UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE PATENT AND TRIAL AND APPEAL
BOARD

-----------------------------x

GOOGLE, INC.,
Petitioner, IPR-2015-00343
IPR-2015-00345
vs. IPR-2015-00347
IPR-2015-00348
NETWORK-1 TECHNOLOGIES,
Patent Owner.

-----------------------------x

Patent Nos. 8,640,179
8,205,237
8,010,988
8,056,441

-----------------------------x

VIDEOTAPED DEPOSITION OF GEORGE KARYPIS
New York, New York
Thursday, November 12, 2015
9:05 a.m.

Reported by:
Jennifer Ocampo-Guzman, CRR, CLR

Job No. CS2183243

November 11, 2015
9:05 a.m.

Videotaped Deposition of GEORGE KARYPIS, held at the offices of Amster Rothstein & Ebenstein LLP, 90 Park Avenue, New York, New York, New York, pursuant to notice, before Jennifer Ocampo-Guzman, a Certified Real-Time Shorthand Reporter and Notary Public of the State of New York.

A P P E A R A N C E S:

SKADDEN ARPS SLATE MEAGHER & FLOM LLP
Attorneys for Petitioner
4 Times Square
New York, New York 10038
(212) 735-2419
BY: DOUGLAS R. NEMEC, ESQ.
douglas.nemec@skadden.com
ANDREW GISH, ESQ.
Andrew.gish@skadden.com

DOVEL & LUNER
Attorneys for Patent Owner
201 Santa Monica Boulevard
Santa Monica, California 90401
(310) 656-7066
BY: SEAN LUNER, ESQ.

APPEARANCES (CONTINUED):

AMSTER ROTHSTEIN & EBENSTEIN, LLP
Attorneys for Patent Owner
90 Park Avenue
New York, New York 10016
(212) 336-8074
BY: CHARLES R. MACEDO, ESQ.
cmacedo@arelaw.com
COREY HOROWITZ, ESQ.
(p.m. session)

ALSO PRESENT:
CHRISTOPHER HANLON, Videographer

I N D E X

WITNESS	EXAMINATION BY	PAGE
GEORGE KARYPIS	MR. NEMEC	7

--------------- EXHIBITS ------------------

KARYPIS		FOR I.D.
Exhibit 1	Declaration of George Karypis	23
Exhibit 2	Photocopy of U.S. Patent No. 8,010,988	68
Exhibit 3	Photocopy of U.S. Patent No. 8,205,237	69
Exhibit 4	Photocopy of U.S. Patent No. 8,640,179	69
Exhibit 5	Photocopy of U.S. Patent No. 8,656,441	69
Exhibit 6	Wikipedia entry entitled, "Big O notation"	108
Exhibit 7	Photocopy of U.S. Patent No. 6,188,010	112
Exhibit 8	Single-page chart	123
Exhibit 9	Single-page chart	128
Exhibit 10	Single-page chart	146
Exhibit 11	Photocopy of U.S. Patent No. 5,874,686	160
Exhibit 12	Photocopy of U.S. Patent No. 6,970,886	178

Page 6

THE VIDEOGRAPHER: Good morning. We are now on the record. Please note that the microphones are sensitive and may pick up whispering and private conversations. Please turn off all cellphones or place them away from the microphones, as they can interfere with the deposition audio. Recording will continue until all parties agree to go off the record.

My name is Christopher Hanlon representing Veritext. The date today is November 12, 2015. The time is approximately 9:05 a.m. This deposition is being held at Amster Rothstein & Ebenstein located at 90 Park Avenue, New York, New York and is being taken by counsel for the petitioner.

The caption in this case is Google Incorporated versus Network-1 Technologies, being held before The Patent Trial and Appeal Board, case numbers 343, 345, 347, and 348.

The name of the witness today is Dr. George Karypis. At this time I

Page 7

would ask counsel to please state your appearances for the record.

MR. NEMEC: Douglas Nemec of Skadden Arps for the petitioner, Google. And with me is Andrew Gish, also with Skadden Arps for the petitioner.

MR. LUNER: Sean Luner for Patent Owner Network-1 Technologies from Dovel & Luner.

MR. MACEDO: Charles Macedo from Amster Rothstein & Ebenstein, also for the Patent Owner, Network-1 Technologies.

THE VIDEOGRAPHER: Thank you.

Our court reporter today is Jennifer Ocampo-Guzman, representing Veritext. She will now swear in Dr. Karypis and we can proceed.

G E O R G E K A R Y P I S, called as a witness, having been duly sworn, was examined and testified as follows:

EXAMINATION BY

MR. NEMEC:

Q. Good morning, Dr. Karypis.
A. Good morning.

Page 8

Q. You understand you've just been sworn to testify under oath in the same manner you would if you were testifying in a court of law?
A. Yes.
Q. And do you feel there's any reason that you can't testify fully and accurately today?
A. No.
Q. No medical conditions or health issues that would interfere with your ability to testify?
A. No.
Q. Have you ever given a deposition before?
A. No, I have not.
Q. Have you ever served as an expert witness in a litigation before?
A. No, I have not.
Q. Just a couple of general background comments, then.

Jennifer will be taking down everything we say today on the record. I'm going to try, against my normal tendency, to speak slowly and clearly; but if my questions

Page 9

are not clear to you either because you can't hear them or can't understand them, feel free to ask me to clarify.

Is that fair?
A. Yes.
Q. And likewise so as to avoid talking over each other and making Jennifer's even more difficult than it already is, I would ask you to wait to answer until I've finished my question; and I in turn will wait for your answer before I ask another question. Fair?
A. Fair.
Q. If you would like to take a break during the course of today's proceedings, feel free to speak up. I generally break every 90 minutes or so, but this is not a forced march, so if you need to step out, please speak up.
A. I will.
Q. You understand that you are here to testify today in connection with a declaration that you submitted on behalf of Network-1 Technologies, correct?
A. Correct.
Q. And that declaration was submitted

in connection with four inter partes review proceedings that were instituted at the request of Google?
A. Yes.
Q. And there are four U.S. patents at issue in those IPR proceedings, right?
A. Yes.
Q. And you've referred to those as the IPR patents in your declaration?
A. I believe so.
Q. And the inventor on each of those patents is a man named Dr. Cox, correct?
A. Correct.
Q. So if I occasionally refer to the patents today as the Cox patents, will you understand what I'm talking about?
A. Yes.
Q. Just a couple of other terminology points before we move on. I may refer to the board or the P tab.
Are those terms that you've heard?
A. Yes, I have.
Q. The P tab is the Patent Trial and Appeal Board. You understand that that's the tribunal that will be, in the first instance,

deciding the matters in dispute in this case?
A. Yes.
Q. Okay. In the declaration that you submitted on behalf of Network-1, you expressed certain technical expert opinions, correct?
A. Correct.
Q. And you expressed the opinion that the challenged claims of the Cox patents are not unpatentable, correct?
A. Correct.
Q. You've expressed the opinion that the challenged claims of the Cox patents are not anticipated by the prior art, correct?
A. Correct.
Q. You've also expressed the opinion that the challenged claims of the Cox patents are not obvious, in view of the prior art; is that right?
A. Correct.
Q. Under the umbrella of those opinions would it be fair to say that the, the opinions you've expressed fall roughly into two categories, the first being opinions with regard to how the claim language in the

Cox patents should be interpreted, and the second category being the teachings of the prior art?
A. Correct.
Q. In connection with forming your opinions, what information did you rely upon?
A. The specific information I rely upon I believe is listed in my declaration. I can give you the exact list, if you give me a copy of it.
But on top of my head, it involves what you referred to as the Cox patents, the -- the patents that was submitted by Google as part of the IPR, specifically the Ghias patent, the Iwamura patent, Conwell patent, the Dr. Moulin's declaration and deposition and the IPR filings that Google filed.
Q. The last item, I'm sorry, was the actual filings?
A. Correct.
Q. So the petitions?
And were there also some Wikipedia pages to which you referred?
A. Correct. I believe there were two or three Wikipedia pages. Everything is

fully detailed in my declaration.
Q. Okay. Let me focus for a moment on the Moulin declaration. That's a set of declarations submitted by Dr. Pierre Moulin in support of Google's petitions; is that right?
A. Correct.
Q. In what fashion, generally speaking, did you rely upon Dr. Moulin's declarations in forming your opinions?
A. I read the declarations. I just tried to understand some of the context, you know, behind the IPR filings, and that's about it.
Q. Did the -- aside from providing context for the matters in dispute, did the information presented in the Moulin declaration influence your technical opinions one way or the other?
A. I do not believe so.
Q. And with regard to the deposition testimony of Dr. Moulin, in what fashion did you rely upon that in forming your opinions?
A. I read the deposition and I don't think it affected, you know, my opinions.

Just, you know, fed them.
Q. So once again, with respect to the deposition, would it be fair to say that you relied upon it for contextual purposes?
A. There are a few places my declaration which I specifically, you know, point to certain aspects of documents, declarations, to confirm, you know, some of my beliefs. And I think, you know, to a large extent that's about it, so.
Q. Okay. So for example, in instances where you agreed with what Dr. Moulin had testified, you might point to his deposition for that purpose, right?
A. That would be correct.
Q. If you didn't have Dr. Moulin's deposition testimony, do you think your opinions in this case would be any different?
A. I do not think so.
Q. And you indicated that you had also relied upon the actual filings, the IPR petitions.
In what fashion did you rely upon those materials?
A. I just looked at, you know, the

claim constructions I believe that's what you call it, that the IPR petitions, you know, put forth and how alleged the claims of the Cox patents are anticipated by the prior art.
Q. Have you ever read any deposition testimony from Dr. Cox?
A. No, I do not -- I have not.
Q. Is it correct that your understanding of the law applicable to these inter partes review petitions that you have is derived strictly from your discussions with counsel in the case?
A. I don't think I follow the question.
Q. Is it correct that your understanding of the patent law applicable to the decision in the IPR proceedings is derived strictly from your discussions with counsel?
A. Yes, and also some reading that I've done, you know, kind of cursory notes about some of the material.
Q. In connection with this proceeding or separately?
A. Just in general.

Q. Do you consider yourself to be an expert on patent law?
A. No.
Q. Do you consider yourself to be an expert on patent office procedures?
A. No.
Q. So you wouldn't be qualified to offer expert opinions on legal issues, then; is that fair to say?
A. I think that's a fair statement.
Q. For example, independent of information that may have been conveyed to you by counsel, you have no expertise on what the various burdens of proof are in an inter partes review petition, correct?
A. That is correct.
Q. And you have no independent knowledge of the legal standards for determining anticipation of a patent claim, correct?
A. Not prior to --
(Discussion off the record.)
A. Not prior to --
(Discussion off the record.)
A. Not prior, I said, the answer to

that is yes.
MR. MACEDO: "Not prior to talking to the counsel, yes."
(Discussion off the record.)
Q. And finally, independent of discussions with counsel, you have no expertise in the legal standards governing whether a patent claim is obvious over the prior art or not, correct?
A. I'm familiar with the patent law that has to do with what something is obvious; and if the familiarity is what you refer as expertise, then, yes, if that's just familiarity, then I'm familiar with the law.
Q. My question was limited to the legal standards, and I will ask it a little bit differently.
A. Okay.
Q. Do you purport to be an expert in the legal standards governing whether a patent claim is obvious or not?
A. I will not qualify myself as being an expert in legal standards.
Q. I take it, in the process of your work, you've studied the disclosure of the

Page 18

Cox patents, correct?
A. By "disclosure," you mean specification?
Q. That's a good point, so another terminology issue: If I refer to the specification of prior art or the Cox patents, I'm referring to the text that precedes the claims in the patents.
Is that consistent with your understanding?
A. Yes.
Q. Okay. So have you -- excuse me -- have you studied the specification in the Cox patents in connection with your work on this case?
A. Yes, I have.
Q. How would you characterize the field of the Cox invention?
A. So the general field of the Cox invention falls in the general area of, I would say information retrieval and from a technical point, and, you know, that's about it.
Q. Do you think content recognition would be an accurate characterization of the

Page 19

field of the Cox patents?
A. No -- content recognition, content retrieval, yeah, those would be, you know, the fields.
Q. And based on your review of the Cox patents, what problem or problems do you understand Dr. Cox to have been addressing with his inventions?
A. So the general problem that, you know, the invention addresses is, from the disclosure, has to do on how to identify records in a database that are similar or very similar to a particular query, and how to take actions based on that identification.
Q. Do you understand one of Dr. Cox' goals of his invention to be an efficient search process?
A. I believe I'm recollecting the claims, and again, if the question has to do with the claims, I don't think the specific claim that -- talks about a search process.
Q. So let me ask a somewhat different question, then.
Do you understand Dr. Cox to be asserting in his patents that he's the first

Page 20

to develop any kind of system to identify records in a database that are similar or very similar to a particular query?
A. My recollection from the specification is that the answer to that is no. The specification I believe discloses a bunch of methods to solve the problem.
Q. And generally speaking what is it that distinguishes the method for identifying or system for identifying records that Dr. Cox purports to have invented from those that came before?
MR. LUNER: Can you repeat the question?
MR. NEMEC: Sure. You want it just read back. Why don't you go ahead and read back.
(A portion of the record was read.)
MR. LUNER: Objection to form.
A. So this is a very broad question. So, and I believe in my declaration I kind of, you know, tried to summarize what are the key distinguishing features of the invention that is disclosed.
Now going, I can read you that

Page 21

section, but off the top of my head there are a bunch of different components. One has to do with a nonlinear search. The other one has to do with a non-exhaustive search. Another one has to do with a near neighbor search. So those are the three that I can recall.
Q. And why is it, in the context of these inventions, that Dr. Cox was setting out to identify similar works as opposed to exactly matching works?
MR. LUNER: Objection to form.
A. So I believe the, you know, the specification, you know, put forth certain scenarios in which things like that would be desirable. I don't remember the specific example that they provided, but I can, you know, you know, hypothesize that finding, you know, similar or not necessarily exact, like, would be something that would be tolerant to some, you know, small changes or some transmission error.
Q. So for example, a distortion in an audio file?
A. That can be an example.

Q. Is it your understanding from the disclosure in the Cox patents that Dr. Cox found it undesirable to find an exact match?

A. I don't recall if it was explicitly stated it's undesirable or not, but -- yeah, actually I don't recall if it's saying it's undesirable to find an exact match.

Q. A moment ago you used the term "nonlinear," I believe.

Do you mean that to be synonymous with sublinear?

A. Well, nonlinear is not synonymous with sublinear, but in the context of the Cox patents the nonlinearity that they're talking about is sublinearity.

Q. So you mean in general nonlinearity is not synonymous with sublinearity, separate from the Cox patents?

A. Yes.

Q. What, then separate from the Cox patents, what do you understand nonlinearity to mean?

A. Something that is not linear. I believe I have a precise definition of linearity in my disclosure, but, you know, a

function, you know, of, you know, that increases at a rate that is either higher, greater or smaller than linear is a nonlinear function. For example, a function that is quadratic would be a nonlinear function.

Q. By "rate," do you mean to imply speed?

A. So this is very precisely described in my declaration. I can give you the definition.

Actually, can you give me a copy of the declaration?

MR. NEMEC: Sure. We can go ahead and mark it. We will mark as Karypis 1 the declaration of Dr. George Karypis submitted in the four IPR proceedings.

(Karypis Exhibit 1, Declaration of George Karypis, marked for identification, this date.)

(Discussion off the record.)

THE WITNESS: At some point in time we'll switch to iPad with those?

MR. NEMEC: Yes, that's been tried. I find it difficult in the deposition context, but some people like it.

A. I cannot find the exact place, but repeat your question and I can answer it from my head.

Q. Sure. So the question that I had posed was, what is your understanding of the term "nonlinear," separate and apart from the Cox patents?

A. Sure. So the term "nonlinear," you know, first, you usually do, you know, I have a function that is a parameter of a certain variable, let's say N. Like if I increase that variable by certain fraction, like, so I look at 2N or 4N; if I have an increase in the amount of the value of that function, right, that is not the same proportion, right. It's not 2, a factor of 2 or a factor of 4. If I have the corresponding increase on the integer variable, but then that would be a nonlinear function.

Q. How does the definition that you just gave differ from the definition of sublinear, as you understand it, separate from the Cox patents?

A. I think it is exactly the same definition. The notion of sublinear is a

function in which you find increase by let's say a factor of 2 or a factor of 4, right, an increase in the output of that function would be less than a factor of 2 or a factor of 4.

Q. Now, in your view is the term "sublinear" used differently in the Cox patents?

A. No, I believe this is the use of, this is how the term is used.

Q. You mentioned the term "non-exhaustive" a new moments ago as well, correct?

A. That's correct.

Q. Non-exhaustive is one of the terms that appears in the claims of the Cox patents, right?

A. I believe so.

Q. The actual term "non-exhaustive" is not used in the specification of the Cox patents, though, right?

A. I don't recall.

Q. Dr. Cox, in his disclosure in his patents, doesn't purport to have invented the concept of non-exhaustive searching, right?

A. I believe so.

Q. You believe he does purport to have invented it?
A. He does not.
Q. He does not. So as of 2000, when Dr. Cox' patent applications were filed, non-exhaustive searching was a concept known in the art, correct?
A. Correct.
Q. As of 2000, had you had exposure to the concept of non-exhaustive searching in your work?
A. I believe you mean prior to 2000.
Q. In or prior to, sure.
A. Yes.
Q. In what context?
A. For example, a fairly widely-used algorithm to research a site array is to do a binary search. That would be an example of a non-exhaustive search.
Q. And you've personally worked with such algorithms in or before 2000?
A. Yes.
Q. Do you recall other instances in which you had firsthand experience with non-exhaustive searching in or before 2000?

A. Another approach by 2000 used for non-exhaustive search would be hash tables.
Q. Any other examples that you can recall of --
MR. NEMEC: Excuse me, I'll start that over.
Q. Any other examples you can recall of non-exhaustive search techniques that you worked with in or before 2000?
A. Not that I can recall finding techniques that I have worked with.
Q. You mentioned binary search.
A binary search is a non-exhaustive search; is that correct?
A. That is correct.
Q. Can you explain to me briefly how a binary search works?
A. So assume you have an array of let's assume numbers and solving in increasing order, and the search is trying to answer the question, is a number in the array or not. And, you know, what do you is you check the middle point therein and compare it with your number. If the number is, your number is smaller than what's in the middle

point, then you disregard the first half of the array, and you perform the same search on the second part of the array.
You continue that way until you either find that value in the array, or your end result becomes an array, at which point in time you don't find the value.
Q. Now, if the array you are seeking to search is not sorted, can you still perform a binary search on that array?
A. You can perform a binary search, not and get a correct result. But if your goal is to get the correct answer, you cannot perform binary search.
Q. And what do you mean by "the correct result"?
A. In the example that I gave, if the number exists in the array, then it will return true. If the number does not, it will return false. If the array is not sorted, there are no guarantees that the algorithm will explain, will lead to the correct answer.
Q. So it might return the correct answer, but it also might not?

A. A high probability it will not.
Q. Can you use a binary search to find a near match or only an exact match?
A. The standard binary search on the sorted array can be modified to find a near match.
Q. What sort of modification would be required?
A. There are a couple of ways to implement it, but I would presume a standard way of doing that is after you do your binary search and you get an Mk array, then, you know, conceptually you backtrack to your previous step and, you know, that middle value on your previous step can be returned plus a, you know, near match.
Q. You also mentioned hash tables.
Was the use of hash tables a form of non-exhaustive searching?
A. Yes.
Q. Can you explain how a hash table works? In general terms.
A. So the two prototypical types of hash tables, I'll just describe one of them, right. So it consists of an array, and each

element in the array has a link list associated with it. So the way you store the data there is you have a function that will map the original, let's say keys into some range, you know, that is bonded from one to the length of that array that you use. Then you apply that function on that key. That gives you an index in the array. And then you put the data into that link list associated with that element of the array.

So this is how you populate a hash table, and then when you search, you have a key, you apply exactly the same function, you get to a link list that is associated with an element of that array which your hash value maps to; and then you, you know, sequentially scan that link list to see if that key is there or not.

Q. So by -- you used the term "key" there.

Is key -- what is a key?

A. The key, the equivalent in my previous example are the numbers that we store in that, in that sorted array.

Q. So a key is the entirety of the

reference that's served in the array, or it's some representation thereof?

A. I don't think I -- I fully -- I don't think your question is fully fleshed out. If -- can you rephrase, repeat it?

Q. Sure, sure.

I'm starting at a very fundamental point, which is to understand what exactly the key represents in reference to or in relation to the items that are stored in this array.

A. Okay.

So in a very general setting, right, you know, it's -- for all the stuff we did -- with the patents that we're discussing over here, you know, usually what a key represents is some way of describing, you know, the data they would like to store. So usually you talk about a key value pairs, or like a key is an end file with the data. Sometimes the data can be the key itself. Right? Sometimes the key is just a unique identifier of the data.

So what you store in the array or in a hash table, you store both the key, and

if they're available, the values as was given, the data associated with the keys.

Q. Okay. So now when it comes time to search the hash table populated in the fashion you just described, how does the search process proceed?

MR. LUNER: Objection to form.

A. So in the typical way, if the -- actually the way the search proceeds was already explained, you know, prior to that. But you take the key, you apply the hash function that maps in the range from one to the length of the array, and then you go to the link list and then you do a sequential scan of the link list, and you compare the actual key with the key stroke there. And if they're identical you return back the data, or the key, it depends on what the values are.

Q. So each entry on the link list is associated with a single reference work or multiple reference works?

A. What do you mean by "reference works"?

Q. Let's establish another terminology

thing, because this may come up throughout the day.

When speaking in terms of database searching, I at least use the term "reference work" to refer to what is stored in the database and "query work" to be you are trying to find in the database.

Is that a fair usage?

A. I mean, that's your usage, which is, now that I know what it is. So a reference work is what is stored on the database. A query work is what you use to query, right?

Q. So that's the terminology I used.

Do you feel it's accurate from the standpoint of a person skilled in the art?

A. I think it does make sense, but beyond the type of terms that people, we use in a normal setting. So in the normal setting someone will use a database entry, you know, and query.

Q. I'm not trying to quibble over the terminology. I just want to make sure we're speaking the same language.

So the two terms you just used then

were database entry?
A. Or database record.
Q. Or database record; and query?
A. And the query.
Q. Okay. And I will try to stick to that terminology throughout today's questioning.
A. Okay. So what was your question again?
Q. Okay.
So the question I posed, then, was in the linked list that you were describing, is an individual entry on the linked list associated with a single database entry or a multiple database entries?
A. So in the link list example that I gave you usually each entry in the link list is associated with a single database entry.
Q. So this hash table search process that you're describing, then, the query is compared to each entry on the link list to determine whether there is a match; is that correct?
A. In the way I described it, if you implement a link list in a standard way, yes.

Q. And if a match is found, what happens at that point?
A. If a match is found, so you either return to a match is found; or if there is data associated with that record, you know, you return the data as well.
Q. And this is a search process that you would characterize as a non-exhaustive process?
A. That would be a non-exhaustive search.
Q. What is it about the search that renders it non-exhaustive?
A. So the key element that makes a search to be non-exhaustive is if you think of the hash table in which every row of that table has a link list associated with it, you know, as a result of applying the hash function right away narrows down to one of those link lists, and I would search within that link list. So I never search the rest of the video clips in the hash table, that are associated with the different entries.
(Discussion off the record.)
A. Not associated with the different

entries.
Q. So I'm not sure I followed that last part.
I had understood you to say you compare the query key to each entry on the link list to look for a match.
Do I have that right?
A. The way the hash table querying works, like you apply the hash function, you get a key; based on that key you go to that entry in the table, the table of link lists, right, so you go to that entry in the table that contains that specific link list, and then you traverse that link list.
Q. Okay. Now how is it that I go about finding the right entry in the table to then traverse the link list?
A. So I have the hash function that takes what's called the key or the integer that describes that key and maps it into a range from one to the length of the hash table.
Q. Have you personally designed software systems that use this kind of hash table?

A. Yes, I have.
Q. And did you do so in or before 2000?
A. Yes, I have.
Q. So we've discussed hash tables and binary searches.
Are there any other forms of non-exhaustive searching you were aware of in or before 2000?
A. There are other methods of non-exhaustive searches that have been developed prior to 2000. I believe the Cox patents describe some of those methods. I believe they talk about the k-d trees as an example of a non-exhaustive search. They talk about clustering as an example of a non-exhaustive search. And then they give a bunch of other examples, too.
Q. So I should have asked, in the hash tables --
MR. NEMEC: Strike that.
Q. In the software systems you developed in or before 2000 that used hash tables, what were the hash tables used for?
A. Hash tables are very generic to

Page 38

like a textbook data structure, to store key value pairs. I believe the hash table that I used is just to store the presence or absence of a particular key, so.

Q. So was it for storing any particular type of data?

What I mean by that is text as opposed to audio or video or something like that.

A. So the question has to do with the software system that I designed?

Q. Correct.

A. The answer is they were not really storing any data.

Q. In or before 2000, did you ever use a hash table to locate near matches to a key?

A. No, I never did.

Q. Was it generally known to people skilled in the art in or before 2000 that you could use hash tables to find a near match between a query key and a database reference?

MR. LUNER: Objection to form.

A. So the question is about finding a near match, correct, between the query key and the database key, right.

Page 39

Q. Correct.

A. No, prior to 2000 there were some, you know, I would say early work or some work on using specifically design hash functions that would allow you to do something like that.

Q. But generally speaking, the key matching that would take place in the hash table would be exact matching, as of 2000?

A. That would be correct.

Q. But an exact match between a query key and a key in the database may not correspond to an exact match between the work from which the key was derived, correct?

A. If I understood your question correctly, what you're asking is a fall line, is -- actually, repeat your question, because I don't think I understood your question.

Q. Sure. The key, the key, and I'll back up a little bit. The key is derived from a larger set of data, correct?

A. Again, there are two keys that we are talking about here, right? So one is the key that is associated with the data, right. And then there is that, let's say the hash of

Page 40

that key that you obtain by applying that hash function. So let's call it original key and derived key, right?

Q. Okay.

A. Go ahead.

Q. The derived key being the hash?

A. Yes.

Q. And then this, the hash table lookup that was generally known as of 2000, what you were seeking to do was find an exact match to your hash, correct?

A. No. The way hash tables are implemented in that link list what you actually store is the original key.

Q. Okay. So in the lookup process are you looking to match identical keys?

A. You -- when you compare the query key to each of the record in the link list, so you look to match the identical keys, you know, in the original, of the original keys.

Q. So in practice, a non-exhaustive search would take the form of an algorithm implemented in the software, right?

A. I would expect so, yes.

Q. Dr. Cox doesn't disclose any

Page 41

specific non-exhaustive search algorithms in his patents, does he?

A. You mean he describes a bunch of non-exhaustive search algorithms. I presume that means discloses right?

Q. He describes categories of algorithms, correct?

A. I think he describes some specific algorithms. I believe he talks about k-d trees and vantage point trees. So those are, you know, I mean they are specific algorithms.

Q. Understood. And a k-d tree, for example, could be implemented in a variety of ways, right?

A. Yes.

Q. And Dr. Cox doesn't disclose any specific way of implementing a k-d tree, for example, right?

A. I don't recall if he does.

Q. He doesn't disclose a specific way of implementing any other non-exhaustive algorithm, right?

A. I do not recall he describes specifically of implementing, no.

Q. Do you understand Dr. Cox to have disclosed any new type of non-exhaustive search algorithm?
A. I do not recall Dr. Cox disclosing any new algorithm for a non-exhaustive search.
Q. You understand that the Patent Trial and Appeal Board has proposed a construction of the term "non-exhaustive" for use in these proceedings, correct?
A. Correct.
Q. And the board has stated that "The non-exhaustive search should be construed as a search that locates a match without comparison to all possible matches," right?
I think you might be looking for the chart of the constructions in your report. It's --
A. It's page 39.
Q. Yes. That's the one you're looking for?
A. Yes.
So I believe you said a search that locates a match without a comparison of all possible matches, right? Yep.

Q. What do you understand "possible matches" to mean in that definition?
A. So by possible matches in the definition, my understanding of that would be what is the result of a query. I believe in both the Cox patents as well as some of the patents that Ghias and Iwamura, the result of the query is, let's say a record, is a melody. Right. So that part, right, is a possible match.
Q. So a possible match is something that would, it's something in the data set that would correspond to the query; is that an accurate way of putting?
A. I mean possible match is something in the database that can potentially be returned as an answer to a query.
Q. So if we're querying a database of songs, and the system returned a statement that the query matched the last ten seconds of a song in the database, then what would be the match?
MR. LUNER: Objection to the form.
A. If the system was designed to identify, you know, the specific segments,

right, within a song that the query matches, right, then, you know, that segment would be the result of the query.
Q. In your experience have you ever heard a search that looks at every entry in a reference database characterized as a non-exhaustive search?
MR. LUNER: Can you repeat the question?
MR. NEMEC: Sure.
Q. In your experience, have you ever heard a search that looks at every entry in a reference database characterized as a non-exhaustive search?
A. So if the goal of that query was to return an entry in the database as a result, like, what you described is an exhaustive search.
Q. So you think a person of ordinary skill in the art would be incorrect to characterize a system that looks at every entry in the database as a non-exhaustive search?
A. So again, if the result of that query was to identify an entry in the

database, so yes, that would be an incorrect characterization.
Q. And what if the search looked at every entry in the database but only looked at one byte of data in each entry in the database; would it still be correct to characterize that as a non-exhaustive search?
A. So a query that looks at every entry in the database, right, even if it only looks at a subset of the data in each entry in the database, right, will still be an exhaustive search.
Q. And that's true even in the circumstance where the query you are looking for in the database is one byte, and each entry in the database is ten megabytes?
A. Again, if the goal of the query was to identify one of those records, right, so the fact that it only compares one byte and still has to go through all the records in the database, that still will be a non-exhaustive search -- excuse me, will still be an exhaustive search.
Q. Was that the manner in which you've used and understood the term "non-exhaustive"

throughout your career?
A. Yes.
Q. You're confident that in all the presentations you've given and papers you've authored you've never defined non-exhaustive in a different way?
MR. LUNER: Objection, form.
A. It's hard for me to recall every single paper I wrote and every presentation I gave, but I believe to a large extent that's my, you know, that's my definition of an exhaustive search, the one that would search every record, right.
Q. So it's possible that there may be other ways in which you've used the term "non-exhaustive" throughout your career; is that what you're saying?
MR. LUNER: Objection, form.
A. It can be possible, but I don't recall any other uses.
Q. Do you believe a person skilled in the art would be incorrect to characterize a search where every entry in the database is searched but only a fraction of the data in each entry is searched as non-exhaustive?

A. I believe that would be incorrect.
Q. Take a look at paragraph 80 in your declaration, if you would, Exhibit 1.
A. Okay.
Q. In that example you refer to a situation where there are 100 records in a database.
Do you see that?
A. Yes, I do.
Q. And then you, say, "A non-exhaustive search could use an intelligent algorithm to exclude 75 records from the search such that only 25 would be searched during the comparison process."
Do you see that?
A. Yes. I'm just reading this now.
Yes, I read that. Yes.
Q. So in this example, how would the intelligent algorithm you were referring to go about excluding 75 records from the search?
A. The hash table example I just gave you would be an example of an algorithm.
Q. In general terms, then, how is it in that hash table example that the algorithm

is able to determine that 75 of the 100 records need not be searched?
A. Well, the algorithm will not necessarily determine 75 percent, but it will determine that, you know, a, potentially a subset of the data will not need to be searched.
Q. In your example it was 75 of 100. That's why I used that.
A. Yes.
Q. So how would the algorithm go about determining what subset of information need not be searched?
A. So in the hash table example that I gave you, so if I load the records, using the approach that I outlined before, then given a key that I would like to locate the record for, right, if I applied exactly the same approach to generate the hash key, right, that would get me to a link list that I know that if any other record with the same key had been submitted before, would have been put in the same link list, so let me research that link list.
Q. So the determination of which

records in the database can be excluded from the search is made, at least in part, based on the nature of the query, correct?
A. So the determination of which records not to include, to exclude in the search is a function of the query.
Q. Because we can't know where not to look until we know what we're looking for, correct?
A. I would think so, yeah.
Q. And in that fashion, the content of every entry in the database is taken into account in deciding which entries to search and which not to search, correct?
A. That is not the case, because the hash table example that I gave you, that, you know, I do not take into account the content of every entry in the database at search time to figure out what to search or what not to search.
Q. But you do take into account some information about every entry in the database in order to determine where to search and where not to search, correct?
MR. LUNER: Objection to form.

A. Not in the hash table example I gave you.
Q. What about other examples?
A. I cannot think off the top of my head of a method that would take into account all the entries in the database during the query phase, in order to determine what to search and what not to search.
(Discussion off the record.)
A. In order to determine what to search and what not to search.
Q. So the hash table is the only example that comes to mind?
A. I'm sorry, but I thought your question was for an example of the algorithm, right? So your question was -- what was your question?
Q. Well, I'll ask it a different way.
Aside from a hash table, can you think of any other intelligent algorithm by which you could exclude 75 records from a 100-record database and focus only on the remaining 25?
A. A method like a k-d tree or any other space-partitioning method, you know,

form category.
Q. So how would a space-partitioning method like this allow us to exclude records from the search?
A. So a space-partitioning method is, you know, is to a large extent similar ideas like a hash table; but what you do is you split each of the dimensions of, you know, a high-dimensional feature, right, or a multidimensional feature, right, so that you store the data in the appropriate buckets based on the dimensions of the record, of the key of the record, and then you use exactly the same approach to select the buckets.
Q. So the decision of what database records go in what bucket is based on the nature of the database records, right?
A. The decision of which database record goes to which bucket is a function of the, of the key associated with that database, right.
Q. So all the database records in a given bucket have something in common with one another, correct?
A. They -- all the records have

something in common with respect to the key, right. So there has to be some property of the key that, you know, put those guys in the same bucket.
Q. That property of the key is in some fashion representative of the records in that bucket, correct?
A. Again, if we take the very generic view of a record consisting of a key and some data, right, you know, and the key is a way of retrieving the data which may or may not be similarly be associated with the data, then the answer to your question is no.
Q. Something about that key that represents the bucket is indicative of whether our search should consider the entries in that bucket; isn't that correct?
A. Can you repeat your question again?
Q. Sure. Something about the key that represents the bucket of records is indicative of whether, in a given search, the contents of that bucket should be examined, right?
A. That depends on the data search that you use or the method that you use. For

instance, in a hash table, each, it's the hash value of the original key that returns the bucket. So that may lead to buckets containing records which hash key value is the same, right. So if that's what you mean by indicative, yes, a hash table that falls in the same bucket are keys such that the hash value -- hash key values are the same.
Q. So then in the search process, we would be looking for a correspondence between the query and the key to determine whether to further consider the contents of a given bucket; is that right?
MR. LUNER: Objection to form.
A. I mean a query consists of, I presume, a key, right? So when you say between a query and a key, what do you mean?
Q. Well, let me step back from this a bit, and let's put aside the hash table for a moment.
A. Okay.
Q. And let's talk about this, the space-partitioning example.
A. Okay.
Q. In the space-partitioning example,

and correct me if I'm wrong, but you are essentially creating a decision tree with intelligently organized information down the various branches of the tree, correct?

A. So the decision tree is a correct analogy. The level of intelligence is up from the partition, but the decision tree is a good way of thinking about it.

Q. So it's organized in some fashion such that information with some quality in common is down one branch, and information with a different quality in common is down a different branch, right?

A. So it's organized such that, you know, keys, right, to then put certain dimensions for certain characterization would fall into one branch of the tree versus the other.

Q. Okay. So now when we're looking to search that tree, we start with a query. And our goal is to if there's anything in the tree that matches the query, right?

A. Correct.

Q. So we need to decide which branches are worth looking down in this organized

tree, correct, and which ones we can ignore?

A. So the way the space-partitioning methods work, based on the dimensions of the key, the Pk could be one path down the tree.

Q. So you said the Pk -- I'm sorry?

A. So it's based on the values along the dimensions of the key, they pick one path down the tree.

Q. Oh, I see. So we evaluate whether the dimensions of the query key correspond to the dimensions associated with the keys on one of the branches?

A. Again, there are many ways to implement space-partitioning methods, but let's keep something very simple. Let's assume that we have a key that consists of three dimensions, XYZ. And I split my X dimension into ten things, the Y dimension into ten things, and the Z dimension into ten things. Right.

So given a query, I look at the XYZ values, right. And that would get me to, based on the value of the X value that the key has, that would pick the appropriate bucket to get me down to the X branch,

assuming my decision tree has X, Y and Z, in that order. So go down in X and within that X bucket, as I say, with my range, you know, I will branch, take the path associated with the Y bucket, and then within that, I will take the branch that's in the Z bucket.

Q. If you had a reference database containing a single entry, would it be possible to non-exhaustively search that database?

A. I'm sorry. Your question was if I had a database consisting of one entry, will it be possible to search it in a non-exhaustive way?

Q. Correct.

A. The example that I gave you, from both a hash table and as well as, you know, space-partition method, even if I had a single entry, I can still, you know, my search, I'm not even going to examine that area, that entry under certain conditions.

Q. Is that because you are considering there would be empty branches, for example, in the --

A. In the empty branches, there can be

empty link lists in the hash table.

Q. If you had a database with, if you had a music database that contained one entry with 100 songs appended back to back, would it be possible to non-exhaustively search that database determine whether it contained a particular song?

MR. LUNER: Objection to form.

A. Well, in the example that you gave, a record over there is not one song. It's how many songs you said there were. Right?

So all the examples that we talked so far was looking at things like returning back a record, right? In the example that you said, you know, I have one record, right? And the result of my query would be one record.

Q. So an entry in the database may constitute multiple records; is that what you're saying?

A. No, that's not what I'm saying.

Q. I'm not following that.

In my example, you have a database --

A. Yes.

Q. -- with a single entry containing a file of 100 songs appended back to back.
Do you understand?
A. Sure.
Q. And the question is: Is it possible to non-exhaustively search that database to determine whether a particular song is present?
A. So what are the keys associated with that one --
Q. Pardon?
A. What is the key associated with that one entry?
Q. What is the key associated with it? I'm not specifying any particular structure.
In general, is it possible to construct an algorithm that would search that database in a non-exhaustive fashion?
A. If I understand your question correctly, so what you're asking is the following; if I have a single record containing some data, right, and I would like to perform a query that will tell me whether or not that record contains a certain subset, that is the query that we are asking over

here?
Q. Correct.
A. Can a single record database, right, solve a nonsubset problem using a -- so the subset identification problem, right, using a non-exhaustive search?
Q. Right.
A. I can think of nothing on the top of my head. Either one or the other way, I mean there can be other methods that will allow you to do that thing in a non-exhaustive fashion. Like there can be a method that is non-exhaustive. In this particular case, there may be a method that we may have to search that record.
Q. But it could be done non-exhaustively?
A. If I have a single record in my database and I want to answer the question, does the record contain a particular subset, there can be a way to design the system such that it can answer no, right, without having to visit that one record.
Q. How would you do that?
A. Again, I'm going on top of my head

over here, right. But one possibility for something like that is to create, you know, some sort of a, you know, like some sort of an index or something, you know, that relates to it that will extract, you know, from that single record, you know, some signature of those subsets and then, you know, use those signatures, put them into some sort of a hash table or space-partitioning structure, generate exactly the same thing for my query; and then if I've never seen something like that in the past, then I can answer no without searching another single item.
Q. Turning back to your example in paragraph 80 with the 100-record database whereby you are able to exclude 75 records, the process of excluding those 75 records is not a random selection process, correct?
A. By "random selection," what do you mean?
Q. You don't just randomly exclude 75 records from the search, in order to narrow the search set?
A. There can be methods that, you know, they will, can exclude a random subset

and then can provide some probabilistic recovery damages.
Q. So there are approaches where you could just randomly exclude a certain number of records in order to simplify the search process, at the risk of missing a match?
A. Yes.
Q. Would that be a non-exhaustive search?
A. That would be, that would qualify as a non-exhaustive search.
Q. And likewise, if you have an intelligent process of evaluating which records to review and which records not to review, would that be a non-exhaustive search?
A. What would make it a non-exhaustive search is if I have a mechanism by which to eliminate records without having to review each record, right? So if as part of the elimination I have to review a record, you know, as part of the query processing, prior to eliminating, that would be an exhaustive search, because I have to consider every record.

Page 62

Q. So what you want is a system where you know enough about each record in the database to make a determination as to whether or not to examine that particular record, when presented with a given query, correct?
A. Repeat your question again, because I don't think I follow.
Q. Sure. A system, in designing a non-exhaustive search system, what you want is an arrangement whereby you know something about each record in the database such that when presented with a given query, you can make a determination of whether or not to look at the content of that record?
MR. LUNER: Objection to form.
A. No, that is not how a non-exhaustive method work. I mean in your example, you still tie the determination of not looking something by, you know, making a decision about each record at query time. I mean non-exhaustive search, which at that point in time, you know, the fact that I have to make a determination of not to look something by looking at every object, every

Page 63

record, that would still be an exhaustive search.
A non-exhaustive search is a search that it only looks at a subset, right, of let's say the records, comes up with some matches; and it doesn't do anything with the rest, not even going through to simply determine whether or not it would consider that.
Q. So in the example you just gave, the system is completely blind to the content of the records that you're not searching?
MR. LUNER: Objection to form.
A. So at query time, right, the system only considers the example that it searches. Right? That is the query time.
Q. What do you mean by "query time"?
A. So again, going back to that very simple example that I think illustrates many of those ideas, you know, a hash table, a query time, the algorithms, we never looked at any of algorithms. Right. The data structure is designed in such a way so that by only focusing on the stuff associated with that link list, right, it can guarantee the

Page 64

recovery of the records, whether that record exists, without having to look at anything else.
MR. LUNER: Is it a time for a break?
MR. NEMEC: Sure. I was just noticing it was 10:38.
THE VIDEOGRAPHER: The time is 10:36. We're going off the record. This will be the end of disk number 1.
(A brief recess was taken.)
THE VIDEOGRAPHER: The time is 10:48. We're back on the record. This is the beginning of disk number 2.
Q. Dr. Karypis, are you familiar with the term "linear search"?
A. Yes.
Q. That's a search that -- or, pardon me, that's a term that you used in your declaration in this case, right?
A. Yes.
Q. Is a linear search an exhaustive search?
A. So the term "linear search" is used to characterize the complexity of an

Page 65

algorithm and has nothing to do with whether an algorithm does or it doesn't --
Q. I didn't hear the last part, I apologize.
A. The term linear search has to do with characterizing the complexity of the algorithm and has nothing to do with whether the search is exhaustive or non-exhaustive.
Q. So it's possible to have a linear search that is non-exhaustive? That wouldn't be a contradiction in terms?
A. So if I define the complexity, it's -- so, yes, in the linear search, right, so when I -- the parameter that I have of interest in the number of records, I summons a number of records. So if I have a linear time algorithm that is with respect to the number of records, right, then that will indicate an algorithm I have to search of the records, so that would be an exhaustive search.
Q. Okay. So in the example you just gave, the linear search would be exhaustive, correct?
A. Yes, in the example I just gave,

right, a linear -- an exhaustive search algorithm that has to visit every record will have a complexity that is going to be in the order of the number of records, so it would be a linear complexity.
Q. So is a linear search always exhaustive is my question?
A. If the goal of the search is identify a single record, right, in which the fitness of record is determined entirely by the record itself, right, an exhaustive search would be a linear search.
Q. I think I understand your question, but I'm afraid we may be coming at this from opposite directions.
The question that I'm posing is whether a linear search is always exhaustive.
A. So let me give you an example of a linear search that may not necessarily be exhaustive. Okay.
If the part of my search is to find the best combination of K IDs, of K records in my database. An exhaustive search of that will have to enumerate, will have to visit every, every potential, you know, match,

right? So it would be every potential K subset, so that would be an exhaustive search, which complexity is now going to be linear than on the records, you know.
Q. Is it typical in the art to determine a linear search as a sequential search of all N entries in the database?
A. So can you repeat your question again.
Q. Is it typical in the art to define a linear search as a sequential search of all N entries in a database?
A. So a search that has a linear complexity, okay, and if its goal is to return a single result, right, you know, then a linear search under those conditions, you know, will lead to a sequential scan.
Q. So you've mentioned a couple of times the concept of linear complexity, and my question is simply about the term a linear search.
Are these two separate concepts, or are they one and the same to you?
A. Those two concepts, without any other context associated with them, are

different things.
Q. Are different, okay.
Now what about in the context of the Cox patents, do you understand those terms to be used the same or different?
A. So in the context of the Cox patents the term "linear" or "sublinear" refers to linear time or sublinear time. At which point in time those are measures of complexity, so they are the same things.
Q. Let's take a moment to look at that in the patents. We might as well go ahead and mark all four patents at this point so that you have them available as they come up.
MR. NEMEC: We will mark as Karypis Deposition Exhibit 2, U.S. Patent 8,010,988.
Mark as Karypis Exhibit 3, U.S. Patent 8,205,237.
Karypis Exhibit 4 will be U.S. Patent 8,640,179.
And Karypis Exhibit 5 will be U.S. Patent 8,656,441.
(Karypis Exhibit 2, Photocopy of U.S. Patent No. 8,010,988, marked for

identification, this date.)
(Karypis Exhibit 3, Photocopy of U.S. Patent No. 8,205,237, marked for identification, this date.)
(Karypis Exhibit 4, Photocopy of U.S. Patent No. 8,640,179, marked for identification, this date.)
(Karypis Exhibit 5, Photocopy of U.S. Patent No. 8,656,441, marked for identification, this date.)
Q. Dr. Karypis, you have the four patents in front of you?
A. Uh-huh.
Q. Take a look at, if you would, at Exhibit 2, the '988 patent.
A. Yes.
Q. And in particular column 9.
A. Yes.
Q. And line 25. Do you see a reference to the term "linear search" there?
A. Yes.
Q. What do you understand linear search to mean in that context?
A. Let me read the context.
Q. Sure.

A. So the meaning of linear search in this context is a sequential search.
Q. A sequential search, okay.
That's a different concept from the linear complexity concept that we were discussing a moment ago?
A. So linear, it says over here, describes a way of searching. The linear time complexity describes the complexity.
Q. And when Dr. Cox uses the term in his patent here, he's talking about the way of searching, right?
MR. LUNER: Objection, form.
Q. Is that how you interpret him using the term?
A. So the use of term "linear search" over here has to do with the way he refers to the -- the way the algorithm scans sequentially the entries of the -- the N entries.
Q. And as described here in the Cox patent, that linear search is an exhaustive search, correct?
A. So that would be an example of an exhaustive search, yes.

Q. Now, if you take the example of a search that considers every entry in a multi-entry database but looks at so little of each entry that it can't reliably determine whether a match exists in that entry or not, would you still characterize that as an exhaustive search?
A. So that would be an exhaustive search, because you have to look at every entry in every record in the database.
Q. Even if you're not looking at enough from each entry to really know whether a match exists in that entry or not?
A. The -- I believe that's the case. Yes, that would be the case. It would be an exhaustive search.
Q. So even if we look at a single bit of information in every entry in the database, that's an exhaustive search in your view?
A. That's an exhaustive search in my view, yes.
Q. Let's turn back to your declaration, Exhibit 1, at paragraph 83.
A. Paragraph 83.

Q. Now, in paragraph 83 you're describing a hypothetical search where the query work is ABC, and you're looking in a database for records that would match ABC, correct?
A. Very good.
Yes, so what was your question again?
Q. Just try me out to make sure I'm understanding the example that you've provided here.
A. Yes.
Q. Query work is ABC, and the database contains strings of letters, correct?
A. Correct.
Q. And we're looking for a match to ABC in the database?
A. So the records consists of strings, right. We won't have a query. And what we're trying to find out is we're trying to find the record that, you know, you know, has ABC, whose string is ABC.
I believe the setup over here is very simple. I have records consisting of three-character words, and my query is a

three-character word, and I want to go and find a matching record.
Q. And if I look through the database in your example to seek a match to ABC, I look at the first letter in each database entry and exclude those that don't begin with A, to narrow the data set.
Does that step render the search non-exhaustive?
A. So the fact that I have looked at every record in the database, you know, makes the search to be exhaustive.
Q. So in your example you would not examine any entry that doesn't begin with A, beyond identifying the fact that that entry doesn't begin with A, right?
A. So in my example, we examine every entry, perform a comparison on the first character and then returns folds of those records that did not match.
Q. And it will examine further the entries that do begin with A, correct?
A. That would be correct.
Q. It will move on to the second letter?

A. Well, that would be done on a record-by-record basis. Given a query, given a record, first compare the first character of the query and the first character of the record. If they match, we continue. The second character of the query would be the second character of the record. If there's a match, we'll continue.

But if at any given point in the process there is no match, we terminate, we return false, and we move on to the second one.

Q. Without looking further at the remainder of that reference, correct?

A. You're -- correct.

Q. Now, in your example you're looking for an exact match to ABC, correct?

A. In this particular case, I believe so, yes.

Q. What if you were running this search to look for something that was similar to ABC?

A. How will you define "similar"?

Q. What if we were looking for something that had at least two of the

letters in sequence in common?

A. Okay. So in a scenario like that, in this particular example, I will compare A with B, right. So -- but I can potentially have, you know, B and C matching what comes afterwards. So I will continue, you know, comparing until I can reliably determine that that string cannot qualify as a match, at which point in time I will stop examining any more records.

Q. In that case you may need to get through every letter in the sequence in every entry in the database to determine whether or not there is a match, correct?

A. I would think that depending on the query string and the definition of "similarity" that you have, that would require to actually compare everything, in other words, to not early terminate a comparison.

Q. What about a system in which your database contained nine-letter strings, and your goal was to find the presence of a three-letter string in one of the database entries? Could you perform that search

non-exhaustively?

MR. LUNER: Objection to form.

A. So the question you're asking is, I have a hypothetical database containing strings of length 9, right?

Q. Right.

A. And my query is a string of length 3?

Q. Correct.

A. And I want to solve a subset problem?

Q. Correct.

A. Substring problem, right? It's a substring contained in one of the records.

Q. Right.

A. And your question is, can I think of a method that would allow me to do that thing in a non-exhaustive way?

Q. Right.

A. So the answer to that is yes. This is exactly the same example that, the solution approach that I gave you for your example of a one single record database.

Q. And how would that, how would it work?

A. So if I was to design a system that would allow me to do in that fashion, I will for each query in the -- for each string from the database I will extract every substring of length 3, use those substrings to, to create -- let's say in the context of strings, it would probably make more sense. There's a database structure called an inverted index, which is something similar to a hash table idea; and then given that query string, I will just then search the records that have that and nothing else.

That would be an non-exhaustive search.

Q. So in the process you just described, then, you'd consider all the extracted index values?

A. Not during the query time, if that is what your question is. During the query time I only consider the extracted index values that match my query.

Q. So to form the extracted index values, you take into account the content of each record, correct?

A. I do it -- during the indexing

time, you know, I do it, yes.
Q. And then at the query time, you consider only the, only the index values that have something in common with your query?
A. That is correct.
(Discussion off the record.)
A. That is correct.
Q. I'm going to move on to another term. The term "neighbor" appears in certain of the challenge claims of the Cox patents, right?
A. That is correct.
Q. As well as "near neighbor"?
A. I believe so.
Q. And the term "neighbor search" also appears in some claims?
A. I believe so.
Q. And "identifying a neighbor" appears in some claims?
A. I believe so.
Q. As of 2000, is neighbor searching something that you were familiar with?
A. Yes.
Q. Dr. Cox doesn't purport to have invented neighbor searching in his patents,

does he?
A. I don't believe so.
Q. And in fact it was a concept well known to people skilled in the art in and before 2000?
A. That is true.
Q. In general, what was neighbor searching used for as of 2000?
A. So I can think of two -- two journal applications for that. One is to speed up search, and the second one is to enable -- actually, speeding up search is probably the primary application of that. I mean the driver behind it.
Q. And how is it that use of neighbor searching would speed up the search?
A. Well, the general idea of a neighbor search, right, which is, the goal is to identify a close but not necessarily the closest match, right. It, you know, it makes it, the logic, it makes it easy to make optimizations through the search process that would allow you to get to an algorithm that -- it's faster. Like, for example, it has a sublinear time limit.

Q. Is it your understanding from the disclosure in the Cox patents that Dr. Cox was not concerned or --
MR. NEMEC: Strike that.
Q. -- was not interested in finding an exact match in the data sets to be searched?
A. I don't recall that, you know, in the disclosure that it was an explicit statement, saying it's, you know, we do not want the exact match or the closest match.
Q. Some of the examples Dr. Cox uses for application of his invention in the patents are to identify songs and video works, correct?
A. I believe so.
Q. You think it's within the spirit of that application to ignore exact matches of audio or video works in the search process?
MR. LUNER: Objection to form.
A. So again, I'm not very familiar with those applications from an implementation and from a business standpoint, right; but ignoring the exact matches, right, is -- you know, those things exist, right. There should not be, no.

Q. So if the intent of your system is to identify a song in a database, if that exact work exists in the database, you're going to want to find it, right?
MR. LUNER: Objection to form.
A. I mean to a large extent that has to do with a specific application, right? I can take scenarios in which given a query, if there is something identical in your database, you know, that may not necessarily be what you're after, something that is close to that. And you know, for a typical example, something like that would be like a recommended system.
Q. In your work have you ever designed a system for recognition of audio content?
A. No, I haven't.
Q. Have you ever designed a system for recognition of video content?
A. No, I have not.
Q. Have you ever written any papers about audio or video recognition technology?
A. No, I have not.
Q. As of 2000, if you had a search that was configured to identify close matches

but would also identify an exact match, if one were present in the data set being searched, do you believe a person skilled in the art would have been correct to characterize that as a neighbor search?

MR. LUNER: Objection.

A. So I believe this is an issue that I have not addressed in length in my declaration, and I believe this is addressed in --

Q. Let me stop you for a second.

Before you consult your declaration, are you able to answer that question from your experience in the field?

A. Yeah. I mean -- so your question, if -- just to make sure I understand it is, if I have a search, right, that will return, in addition to a close match, always an exact match, would that be a neighbor search or not --

Q. Correct.

A. -- correct?

So if that search will always return the exact match, right, that would be, that would not be a neighbor search, that

would be a, you know, a nearest neighbor search. Actually in this case it would be an exact search.

Q. I'm asking that based on the knowledge of people skilled in the art as of 2000, so separate from how the term may be used or defined in the context of the patents.

A. I mean if I have a search, right, that is always guaranteed to return the exact, like somebody skilled in the art, you know, in the 2000 frame that you mentioned, that would not characterize the thing as a near neighbor search, right? I mean that is a --

Q. So what would one skilled in the art call a search that is designed to identify close matches but will also always identify an exact match, if one is present in a set?

A. So the term that someone in the art would have used, it would an exact search or a -- a -- some sort of a K nearest neighbor search, right, in which, you know, you identify, you know, all; or something like an

alpha neighbor search, you know. You identify everybody, right, within a certain distance of your query or that came more similar things and so forth.

Q. Okay. Would a nearest neighbor be a form of a neighbor?

A. If I understand your question, you're asking me is an object, right, that is the nearest neighbor, whether or not that is also a near neighbor?

Q. Or a neighbor.

A. Or a neighbor.

What is your definition of a "neighbor"?

Q. I would prefer to work from your definition of a neighbor as it was understood in the year 2000.

So let's start with that.

A. I mean the notion of a neighbor, right, is someone that is close but not necessarily the best or the optimal solution to a search, right, so that would be a neighbor.

So the optimal solution to a search, you know, will definitely be a subset

of the neighbor. So to answer your question, yes, the closest would be close as well.

Q. Okay. Now as of 2000, would the search that always returned the closest match in the database being searched be accurately characterized as a neighbor search?

A. No, it will not be characterized. If it returned the closest, it will not be characterized as a neighbor search.

Q. What would people skilled in the art have called that kind of a search as of 2000?

A. We would call that, use the term the nearest neighbor search.

Q. And is the nearest neighbor search a form of a neighbor search?

MR. LUNER: Objection to form.

A. So the way the neighbor search is, you know, I have -- I have defined in my declaration, I believe it's an agreement, you know, with the board's construction is that a neighbor search, right, is a search, right, that is not guaranteed, you know, to return the -- the nearest neighbor search.

So a nearest neighbor search is

Page 86

guaranteed to return the best result, so the nearest neighbor search is not a type of a neighbor search.

Q. And you say that because of the guarantee that a nearest neighbor search will return the best result; is that right?

A. Correct, so the nearest neighbor search is, you know, we've guaranteed, you know, to return the best result, assuming that, you know, you don't have that threshold. I mean the question of whether or not the nearest neighbor search is to return a result, it will guarantee to return the best result.

Q. And again, is this your view of the meaning of the term, separate and apart from patent office construction in this case?

A. Which term are you referring to, the nearest neighbor search?

Q. Neighbor search.

A. The neighbor search.

Yeah, my view of the term "neighbor search" is independent of the patent office constructions, you know, it's -- actually they're in agreement.

Page 87

Q. So what you're saying is there's no difference between your general understanding of the term "neighbor search" and what you understand the patent office construction to be; do I have that right?

A. So I believe the patent office construction of the neighbor search is a search, right, that is, you know, is guaranteed -- it's a search that will identify a close but not necessarily the best neighbor, right? You know, this is, you know, consistent with my construction, right, and I believe in my declaration I have further clarified that. And I believe, so when I have a search, right, in which it's guaranteed to return a close but not necessarily the closest, all the time, right, so better than neighbor search.

If I have a search in which, you know, always guaranteed to return the closest, okay, regardless of whether or not it returns some things that they're not the closest, right, that it's not a neighbor search. That's a nearest neighbor search.

Q. So in applying the patent office's

Page 88

construction in rendering your opinions in this case, you have excluded from the definition of "neighbor search" anything that guarantees identification of an exact or the closest match, correct?

A. I have excluded anything that always guarantees identification of the exact or closest match, yes.

Q. Likewise, in applying your understanding of the patent office's construction of identifying a neighbor, you have excluded anything that guarantees to always return the exact or the closest match?

A. So identifying a neighbor refers to the search process, right, so it's a search method that identifies, always identifies the closest return, say the closest or the exact match, right. You know, it's not what I refer to as identifying a neighbor.

Q. And if the patent office were to determine that a search is guaranteed to find the closest or an exact match qualifies as a neighbor search in their construction, would you want to revisit the opinions rendered in your declaration?

Page 89

MR. LUNER: Objection to form.

A. Can you restate your question?

Q. Sure. If the patent office were to conclude that searches, a search that guarantees finding an exact or the closest match in a data set qualifies as a neighbor search, would you want to revisit the opinions you've rendered in your declaration?

A. No.

Q. That wouldn't change your view in any way?

A. No. My view is that a neighbor search or identifying a neighbor is a type of search, but, you know, does not guarantee to find the best match. The type of search that always guarantees to find the best match is not the neighbor search or identifying a neighbor.

Q. And what I'm asking you is, if the patent office were to come to the conclusion that a search that does guarantee that it will always find the exact or closest match is a neighbor search, would you want to revisit the opinions in your declaration with regard to whether a neighbor search is

disclosed in the prior art?
MR. LUNER: Objection to form.
A. The written statements are there. So my opinion about what constitutes a neighbor search and define a neighbor was reached independent of the patent office, you know, opinion, right. So I don't think I would be -- I will be needing to revisit, you know, my opinion in those terms.
Q. You understand that the patent office is the one that ultimately decides how these terms are to be construed, correct?
A. I believe so.
Q. So if the patent office were to reach a conclusion different from yours with regard to whether a search that guarantees finding an exact or the closest match qualifies as a neighbor search, would you want to revisit the opinions in your declaration?
A. Opinions regarding what?
Q. Opinions regarding whether the prior art references you've discussed in your declaration disclose a neighbor search.
A. So if the patent office defines a

neighbor search as a type of search that always guarantees to return the best result, right, then my opinion as far as the validity of the prior art, in light of the claims as it relates to the neighbor search in light of the prior art, right, will have to change.
Q. Just for convenience, to have the construction in front of you, if you could turn to paragraph 60 in your declaration.
A. I have 6 here. You sure?
Q. Paragraph 60, the grid showing the constructions that were applied.
A. Yes.
Q. So looking at the construction for neighbor search there, it begins with identifying.
Do you see that?
A. Uh-huh.
Q. And how do you interpret the term "identifying"?
A. Finding.
Q. So if a search is run and returns the title of a matching reference, is it your view that that search has identified a match within the scope of this construction?

A. So if the title of that reference is a unique key, right, a unique identifier of that reference, that would be a result that I didn't find.
Q. And if a search were to return a list of titles and indicate that one of those titles is a match but not specify which one, do you believe that search will have identified a match within the context of the court's claim construction, the board's claim construction?
A. Can you repeat your question again?
Q. Sure. If a search were to return a list of titles and indicate that one of the titles in that list is a close match but not specify which one, do you believe that that search will have identified a neighbor within the scope of the board's construction?
MR. LUNER: Objection to form.
A. So the result of the query is a set of records, right; and I know within that record there is a close match, but I don't know which one it is.
Q. Correct.
A. Right? That will not have

identified the close but not necessarily an exact or close match.
Q. And if a search is run and it returned the title of the closest matching reference, will that search have identified a neighbor within the board's construction?
A. If that search within the construction of the board identified a close but not necessarily an exact or closest match, if it always returned the closest match, right, it's my opinion that that particular board construction with the close but not necessarily exact or closest match, but sort of has two implications, one implication is that if the result -- you know, I mean if the query, that's one, if the query of guarantee to return the closest, right, it's not a neighbor search.
So given your question, if the query always returned the closest match, no, that will not be a neighbor search.
Q. Let's move on to a different term. The term "approximate nearest neighbor search" appears in some of the contested claims, correct?

Page 94

A. Yes.
Q. And you understand that the board has preliminarily construed that term to mean identifying a close match that is not necessarily the closest match?
A. Yes.
Q. Now based on your review of the Cox patents, would you agree with me that Dr. Cox doesn't purport to have invented the concept of the approximate nearest neighbor searching?
A. That is correct.
Q. That's a concept that was known to people skilled in the field as of 2000?
A. The approximate nearest neighbor search as a term, right, has been used, you know, in -- prior to 2000, yes.
Q. Pardon me. Is approximate nearest neighbor searching something that you had experience with as of 2000?
A. Yes.
Q. Now, putting aside the Cox patents and the board's constructions in this case, as of 2000, was it your understanding that an approximate nearest neighbor search must

Page 95

always be sublinear?
A. So outside the context of the patents, the term for approximate nearest neighbor search, you know, it's usually a sublinear algorithm, but it doesn't have to be similar.
Q. Take a look at Exhibit 2, that's in front of you, the '988 patent. And in particular if you look at column 26.
A. Yes.
Q. Element C of claim 15 at the top of the column, do you see the language, "Determining an Action"?
A. So which element do you want me to take a look at?
Q. We're looking at claim 15 in the '988 patent, and this is around line 7.
A. Of the element B, right?
Q. The element C, I'm sorry.
A. Element C.
Q. "Electronically Determining an action," do you see that?
A. Yes.
Q. You've expressed some views in your declaration regarding the term "Determining

Page 96

an action," correct?
A. I believe so.
Q. Do you have an understanding of what constitutes an action within the scope of this claim term?
A. It's been a while since I looked at this particular claim. I need to be able to see my declaration for that.
Okay, I've had a chance to go over it, so what was your question?
Q. Okay, I'll ask it again.
Do you have an understanding of what constitutes an action within the scope of the term "determining an action" in the '988 patent claim 15?
A. I believe so.
Q. What is your understanding?
A. So determining an action over here is, you know, selecting an action to perform.
Q. What sort of actions would fall within the scope of the term "action" in this claim?
A. I presume, you know, fetching a record from a database and transmitting it.
Q. Anything else?

Page 97

A. I cannot think of any --
(Discussion off the record.)
A. No.
Q. Let's go down a little bit further in the '988 patent and look at claim 31.
Do you see that around line 46 in column 26?
A. Yes.
Q. And you see there that the claim specifies "The action comprises providing and/or displacing additional information in association with the electronic work"?
A. Yep.
Q. Do you understand that to be giving an example of an action within the scope of claim 15?
A. That could be one of the actions, the type of action performed.
Q. So displaying additional information in association with the electronic work would be an action that meets the definition of "action" in claim 15, right?
A. Yes.
Q. And might that additional

information include the title of the song that you're searching for?
A. That can be the additional information.
Q. Can you think of anything that you believe would not qualify as an action, as that term is used in claim 15 in the '988 patent?
MR. LUNER: Objection to form.
A. So your question, if I -- actually your question was, can I think of any type of actions that will not fall within the scope of this thing?
Q. Correct.
A. So an action that is not based on the identification would not fall within the scope of that.
Q. Okay. Anything else come to mind?
A. Well, an action that's not determined electronically.
Q. And you say that because the claim specifies that it's electronically determining action, right?
A. Yes.
Q. And you also say that an action not

based on identification of the electronic work wouldn't qualify, because the claim specifies that the action must be based on the identification of the electronic work, right?
A. Yep.
Q. Focusing just on the word "action," can you think of anything that would fall outside the scope of action, as it's used in the context of the claim?
MR. LUNER: Objection to form.
A. That's a very broad question.
I cannot really think of anything. I mean outside, you know, the constraints of the rest of the words in this step.
Q. Let's move to another term. And the word "sublinear" appears in some of the challenge claims issued in the -- correct?
A. I believe so.
Q. And the board has indicated that "sublinear should be construed to mean a search whose execution time scales with a less than linear relationship to the size of the data set to be searched," right?
A. That sounds right.

Q. And if you would like to double-check what I've read, I was reading that from paragraph 60 in your declaration.
A. Yep. I'm in agreement with that interpretation of the sublinear term, to a large extent, yes.
Q. Where the board refers to "size" in that definition, the size of the data set, you've equated that with the number of entries in the reference database; is that right?
A. Number of records, yes.
Q. Number of records.
And is it your view that that's how the term is used in the Cox patents as well?
A. I believe so.
Q. Now, let's take a look at Exhibit 3, if you would, the '237 patent, and in particular column 21. Starting around line 14, if you could take a look at that and let me know if you are familiar with that passage.
A. Okay. I read the passage. Go ahead.
Q. In the example that's described

there, how many entries do you understand there to be in the data set that's being discussed?
A. So any of these example is that 9 million -- 90 million, I'm sorry.
Q. So it's your understanding that there are 90 million entries in the database?
A. So according to the passage, N is defined to be the number of entries, and the database stores each frame as a record. So there are a total of 90 million records in the database.
Q. And this database contains 100,000 commercials; is that right?
A. That's what it says, yes.
Q. So it's your understanding that each commercial doesn't have its own entry in the database, but rather the commercials are deconstructed into their individual frames?
A. So, yes, in this particular example, each frame is considered to be an entry, each frame from the commercial is considered to be an entry in the database.
Q. And that's how we get to N equals 90 million?

Page 102

A. Yes.
Q. Now if we were to construct the database such that each commercial was an individual entry in the database, then N would be 100,000?
A. If the record is a commercial, yes, then it would be 100,000.
Q. Now in your view, in the example here regarding the number of comparisons that would be required to determine whether there is a match among these 100,000 commercials, would it make a difference if the database was constructed with each database entry being one commercial, as opposed to one frame?
A. Again, the question then becomes what is the granularity of the results, right? If what you want to identify is a frame, right, then that's the way to do it. If you want to identify a commercial, right, then I would probably store each, each commercial in its own record.
Q. And what if your goal is to identify a commercial, using a single frame as a query?

Page 103

A. Okay. And what is the question? My goal is what then?
Q. Yes, I'm sorry. If that was your goal then -- then how would you construct the data set?
MR. LUNER: Objection to form.
A. I can see merits for both approaches, so I'm really concerned with the way it was discussed over here, you know, 90,000 -- 90 million records, one frame, which they're base records, or I can construct it as one record for each commercial.
Q. Now, using my example where you are trying to identify a commercial using one frame from the commercial, what would be the merit of organizing the database with each database entry being a single frame?
A. So the merit of doing something like that is, you know, going back to the whole example that you used before, one record database, right? So if I organize it this way, I could potentially develop some, you know, sophisticated basis, allowing me to do it in a non-exhaustive fashion.

Page 104

Q. And if you were to organize the database where each commercial is a database entry, what would be the merit of that approach?
A. So the merit of that approach is, with my goal at the end of the day is to identify a commercial that contains my query frame, then I just need to identify one matching query frame to a video frame in a record. So that I don't have to identify, you know, search every single one video frame, you know, from the record.
Q. Now on line 25, there's a reference to a storage requirement of nine gigabytes.
What does that refer to?
A. The nine gigabytes I believe is 9 million times 1,000 bytes.
Q. So is that the size of the database that would be required to hold all of these frames?
A. It's not here.
Q. The proposal a couple lines above is to take every tenth frame to construct the database from the total of 90 million, correct?

Page 105

A. Correct.
Q. And the assumption is that each vector, meaning each frame, is one kilobyte, right?
A. Okay.
Q. So that gets us to the coverage requirement of 9 gigabytes?
A. Yes.
Q. That's a representation of the size of the database that would be required, right?
A. I believe so, yeah.
Q. In your declaration at paragraph 61, you list two possible interpretations of size of the data set within the construction of sublinear, right? One being the number of entries in the data set and two being the size of an individual entry in the data set; is that right?
A. That's correct.
Q. Why isn't there a third possible interpretation which would be the size on disk required for the database?
A. That can be a valid interpretation.
Q. And you've expressed the opinion

that the number of entries in the data set is the appropriate interpretation.
Why would the size on disk not be an appropriate interpretation?
A. I think that can also be an appropriate interpretation.
MR. NEMEC: Can we go off the record for just a moment?
THE VIDEOGRAPHER: The time is 12:09. We're going off the record. This will be the end of disk number 2.
(A brief recess was taken.)
THE VIDEOGRAPHER: The time is 12:20. We're back on the record. This is the beginning of disk number 3.
Q. Dr. Karypis, I would like to direct you to paragraph 11 in your declaration.
A. Yes.
Q. In paragraph 11 you state your opinion on the qualifications of a person of ordinary skill in the art of the inventions at issue in this case; is that right?
A. That's correct.
Q. Now, is it your view that there is any material difference between your

definition of the level of skill in the art and Dr. Moulin's?
A. In my view, I don't think there's a material difference. I think there is a difference in terms of the degree requirements, but I think once you've had your degree and experience, it all comes out to about the same.
Q. Okay. Now your definition in paragraph 11 refers to two to three years of relevant experience.
What, in your view, would be relevant experience?
A. So relevant experience in this case would be someone working in the area of, you know, content retrieval, content comparison, you know, database systems, information retrieval.
Q. And would it matter what sort of content or information the person had experience with?
A. I don't think it would have been, you know, the type of methods being discussed in those patents, I think are to a large extent, you know, content agnostic.

Q. So you think someone with experience in text information retrieval would be able to apply that knowledge to more highly dimensional problems associated with audio and video retrieval?
A. I do.
Q. At the end of your definition there is reference to a related area, a graduate degree in the same or related area.
What would be the related areas that you're referring to there?
A. So related areas would be probably electrical engineering, possibly statistics.
(Discussion off the record.)
A. Statistics.
Q. If the level of skill in the art was determined to be higher or lower than what you've described here in your declaration, do you believe that would impact any of the opinions you've expressed?
A. I do not think so.
MR. NEMEC: Mark as Exhibit 6 a document, a 13-page document that says "Big O notation" at the top.
(Karypis Exhibit 6, Wikipedia entry

entitled, "Big O notation," marked for identification, this date.)
Q. Do you have Exhibit 6 before you?
A. I do.
Q. Is this a document that you reference and rely upon in your declaration?
A. I believe so.
Q. And this is a printout from Wikipedia?
A. Yes.
Q. Is this something that you downloaded?
A. I don't remember if I downloaded, but it is something that I did find and read, and read.
Q. It is dated at the bottom, it says "September 15, 2015." It actually reads "9/15/2015," do you see that?
A. Uh-huh.
Q. Do you know if that was the date on which this was printed?
A. I'm not sure what date that thing is.
Q. Do you know when this, this document was created?

Page 110

A. No.
Q. Did Wikipedia exist in 2000?
A. I don't really know the answer to that, but there may have been some earlier version of that. I don't know when Wikipedia came online.
Q. Do you know one way or the other whether this discussion of the "Big O notation" that we've marked as Exhibit 6 existed in the year 2000?
A. So this particular content that is, this particular document, you know, I'm not sure it was existing in the year 2000 or not.
Q. And there are a number of other Wikipedia printouts that you reference in your declaration, right?
A. Uh-huh.
Q. Would your answer be the same with respect to those, you don't know one way or the other whether they existed in 2000?
A. That would be correct.
MR. NEMEC: All right. We're on the cusp of jumping into a deeper topic, so this is probably a logical point to break, to grab some lunch.

Page 111

THE WITNESS: Sounds good.
THE VIDEOGRAPHER: The time is 12:28. We're going off the record.
(Lunch recess taken at 12:28 p.m.)

Page 112

A F T E R N O O N S E S S I O N
(Time noted: 1:17 p.m.)
THE VIDEOGRAPHER: The time is now 1:17. We are back on the record.
MR. NEMEC: I would like to mark as the next exhibit, which I believe is 7, Karypis Exhibit 7, a copy of the 6,188,010 to Iwamura.
(Karypis Exhibit 7, Photocopy of U.S. Patent No. 6,188,010, marked for identification, this date.)
G E O R G E K A R Y P I S, resumed.
EXAMINATION (Cont'd.)
BY MR. NEMEC:
Q. Do you recognize Exhibit 7?
A. I do.
Q. Is this one of the prior art references that you expressed opinions about in your declaration?
A. Uh-huh.
Q. Would you agree that the Iwamura patent describes a system for searching for melodies?
A. I do.
Q. And in particular a system whereby

Page 113

use of relative pitch values from a query and compare those to pitch values of known references to look for a match or a near match; is that a fair characterization?
A. Yes. That's one of the ways, yes.
Q. One of the things that's described in the Iwamura reference is a note-by-note comparison process as between a query work and a reference work, correct?
A. Basically, yes.
Q. Would you generally explain your understanding of how that note-by-note comparison process works, as described in Iwamura?
A. My recollection of the particular algorithm or approach, something like that, so I have a query which is for a certain length; and I have a reference melody that's for a certain length. And then I start at the beginning of the reference melody and I compute a distance, you know, between each, what is called data points of the query to the data point in the reference. That would give me some score of distance, I believe.
And then I, you know, shift the

query by one to the right, and I repeat the process and keep on doing that until the end. And at the end I return the score of the match, the best scoring subsegment or the highest scoring subsegment.

Q. The highest or the lowest scorings?

A. I believe they're using distance. That would be the lowest.

Q. Now, if we assume hypothetically that a query is a string, 5, 4, 3, 2, 1, is that a fair representation of a query that we might use in the Iwamura note-by-note search?

A. I believe so, yes. Those would be relative pitches, yes.

Q. So let's assume our query is 5, 4, 3, 2, 1, and let's assume we're going to compare that to a reference work that has the sequence 6, 5, 4, 3, 2, 1.

Is that a reasonable assumption to make, according to Iwamura?

A. Say that again?

Q. Can we assume that 6, 5, 4, 3, 2, 1 would be a valid representation of a reference work, according to Iwamura?

A. You need to check whether or not

the references are centered, which has to do with whether or not it will have present negative values or not.

Q. Whether they will have -- I'm sorry?

A. Whether there would be negative values present in the sequence or not. I believe, you know, both actually, the query and the reference, those are relative pitch differences, so -- and I don't recall if they center those things or not.

Q. What do you mean by "center"?

A. For example, subtract the mean value out of the values.

Q. Okay. So you're not sure whether any given query or reference string would have to have some negative numbers in it?

A. Correct.

Q. Now, you have a couple of examples of strings in your declaration, and we can check those to get the answer.

Paragraph 160 to 161, I believe. Let's see.

And these are sample queries and references that you selected for illustrative

purposes, right?

A. Yes.

Q. So I would be correct in assuming, wouldn't I, that you believe these are accurate representations of what a query work might look like, according to Iwamura?

A. Uh-huh.

Q. Or what a reference work might look like?

A. That is correct.

Q. But you're not certain one way or the other whether the centering you described is required?

A. I don't recall the specific method that they use, but the fact that they have, you know, both positive and negatives that measures a high and a low relative to a baseline, so that's where the negative values comes in.

Q. Now, so let's assume for purposes of this hypothetical that the centering is not required.

So in the hypothetical that I'm describing the query is 5, 4, 3, 2, 1, okay?

A. Okay.

Q. And the reference work that we're comparing it to is 6, 5, 4, 3, 2, 1, okay?

A. Okay.

Q. So when we begin the note-by-note comparison between this query and this reference work, how do we align the pieces or the strings for the first comparison?

MR. LUNER: Objection to form.

A. So I believe Iwamura disclosed a couple of methods. One is one that does initial alignments by computing the scores by relying on the peaks, and I also believe they disclosed an early, less sophisticated method that, you know, they start from the beginning to align with things.

Q. Okay. And is the less sophisticated method that you are describing the note-by-note comparison?

A. I believe so, yes.

Q. So that's the one that I'm asking about at this point now.

If we were to be using the note-by-note comparison, how would we line up the query and the reference in order to make that first comparison?

Page 118

A. So the first position of the query will align against the first position of the reference and so forth.
Q. So for example, the 5 would align with the 6, and 4 would align with 5 and so forth?
A. Correct.
Q. And then in the note-by-note process, after that first set of comparisons is made, the query work would be shifted to the right; is that correct?
A. The query would be shifted to the right by one, correct.
Q. So in the second position, then, the 5 would align with the 5?
A. I don't think that's, so if the query is 6, 5, 4, 3, 2, 1, right?
Q. The query in my hypothetical was 5, 4, 3, 2, 1, and the reference was 6, 5, 4, 3, 2, 1?
A. Yes, so the first 5 with the query will align with the second location of the reference.
Q. And then the 6 in that instance wouldn't be considered, because it does not

Page 119

align with a note in the query?
A. No, correct. It would not make that comparison.
Q. Now you just referenced a few moments ago the peak search approach that's described in Iwamura, correct?
A. Correct.
Q. Now, as compared to the note-by-note search, would a peak search of a given query against a given reference require fewer comparisons?
A. The peak-based approach would require fewer comparisons than the note by note.
Q. And turning back to the note-by-note comparison process, if you had a five-note query in the note-by-note comparison, would the Iwamura process compare that five-note query to every set of five consecutive notes in the reference?
A. In the quote/unquote naive note-by-note approach, the query would compare to every set of five consecutive notes in the reference.
Q. I direct your attention to column 7

Page 120

in the Iwamura patent, in particular the text that begins around 20 and continues down to around 45. If you want to take a moment to look at that and, I will ask my question.
A. Sure.
So you are saying up to line 35, right?
Q. Around to 45, after that second equation.
A. Okay.
Q. The text you just reviewed in column 7 of Iwamura relates to the peak search process, correct?
A. That is correct.
Q. And what we see depicted in the first equation there from about line 26 through 30 is a comparison of a database reference in the top line, correct?
A. Well, the top line actually has part of the database reference.
Q. Part of the database reference being compared to all or part of a query in the second line, correct?
A. That is correct.
Q. And in this particular depiction

Page 121

what we see is the first peak in the query being compared to the first peak in this section of the database reference, right?
A. I believe so, yes.
Q. And then the text here in the second figure describes that in the next comparison, the query is shifted such that the first peak in the query is compared to the second peak in the database reference, correct?
A. Right.
Q. And that's consistent with what you testified to earlier about the peak reference, that the query is shifted with respect to the reference?
A. So the query shifted with the reference, using the peaks as the anchor points to determine the amount of shift.
Q. Now, in contrast, in the examples that you use in your declaration, if you take a look back to paragraph 162, for example, in these examples that you use, you depict a record being shifted with respect to the query, correct?
A. Let me read the paragraph.

I don't think so. I believe the figure at the bottom of page 98 does, you know, imply a shift.

Right, those two things should be named the other way around.

Q. So that was a mistake in these depictions of how Iwamura operates in your declaration?

A. Yeah, it's just the label of the first row should be that of the second.

Q. Do you know if there is any disclosure in Iwamura of shifting the database reference with respect to the query?

A. I don't recall.

Q. In paragraph 162 up above the figure we were just discussing, the description of the shifting process is, in fact, consistent with the mistakenly placed arrow, is it not?

A. Let me read that.

Actually, that statement over there, it's not really clear which one is shifting.

Q. Did you write this portion of your declaration?

A. I did.

Q. Did you write all of your declaration?

A. I did the, wrote the first draft of the declaration; then, you know, I got some language edits from, from counsel.

MR. NEMEC: Mark as the next exhibit, Karypis Deposition Exhibit 8, a single-page chart.

(Karypis Exhibit 8, Single-page chart, marked for identification, this date.)

Q. Dr. Karypis, I've put this together to try to make the next set of questions that I want to walk through a little easier to understand, rather than asking you to memorize long strings of numbers.

Take a look at the top where it says "Query."

Do you see that string of numbers?

A. Yes.

Q. Can you just take a look and confirm that that's the same string of numbers that you used in your example of the Iwamura operation at page 97 of your

declaration, or 101 of 292, whichever you prefer?

A. Yes.

Q. Okay. So in your view that's a proper example of a query that might be run using the Iwamura system?

A. It's a query.

Q. And below that where it says "Reference," do you recognize that string of numbers as the same string of numbers that you used as the record in the database, in your example in paragraph 160 of your report?

A. Yes.

Q. Now take a look at the chart that appears down below.

Do you see that reference sequence in the bottom line of the grid?

A. Yes.

Q. And do you see in comparison number 11 in that row the query appears?

A. Yes, I do.

Q. And do you see that in each of the comparison numbers 1 through 20 depicted in the grid, that the query is shifted by what, one note with respect to the reference?

A. Yes.

Q. Based on your understanding of the note-by-note comparison process described in Iwamura, which of the comparisons in this grid, if any, would take place in the note-by-note comparison process?

A. So my recollection of the note-by-note comparison process.

So my understanding of the note-by-note process that's described in Iwamura, that would be a comparison 11 to 20; but I can easily see that also comparisons 1 through 10 would also be included here.

Q. All right. Now, I ask you the same question but with respect to the peak note search.

If you were comparing this query work to this reference work using the peak note search of Iwamura, which, if any, of the comparisons 1 through 20 would take place?

A. So my understanding of Iwamura, so the comparison that would take place that would be first number 13. And it would be comparison number 18.

Q. Does the Iwamura place any

restriction on the distance between peak notes?

A. I don't recall off the top of my head. If there is any restrictions on the distance between the peak notes. I can take a look.

Q. Sure, I will have you take a look, but before you do that, sir, let me ask another question.

Would distance between the peak notes in either a query or a reference item in Iwamura would be a function of the music, right?

A. The distance between the peak notes in either the query or the reference would be a function of the music and a function of how they're, whether or not they're, how they're doing the centering.

Q. Okay. So let's take that as two pieces, then.

First, can you look at Iwamura and confirm whether or not Iwamura requires centering?

A. Sure.

So I describe both the case in

which, excuse me, they have both absolute peaks as well as relative peaks data, and I don't believe they describe how to get the relative pitch data, so it's both scenarios.

Q. So they describe scenarios that would involve centering and not involve centering; is that what you're saying?

A. It's a relative versus an absolute, yes.

Q. Okay. And then the second half of the question is: Is there anything in the Iwamura disclosure that would place some limit on the number of notes between the peak -- between the peaks?

A. So my understanding of the method they used to identify peaks, there is no limitation in terms of what, the distance between two successful peaks.

Q. And would that be true of both query and the reference?

A. I believe so.

Q. And just to make sure we're clear on this, Iwamura also doesn't set a minimum limit on the number of notes between peaks, correct?

A. So the way a peak is defined has to be a local minimum or local maximum, which by construction would have a minimum distance required.

Q. What would that minimum be?

A. I believe it would be at least one or two notes.

MR. NEMEC: Go ahead and mark another chart as Exhibit 9.

(Karypis Exhibit 9, Single-page chart, marked for identification, this date.)

Q. Do you have Exhibit 9, Dr. Karypis?

A. Yes.

Q. At the top of the page it says, "Query * 5, 4, 3, 2, 1," do you see that?

Can you accept, for purposes of this hypothetical, that that's a query string, according to Iwamura?

A. So the one part that I need to double check is whether or not their peak identification algorithm will allow a peak at the beginning of the string or at the end of the string, as a result of their peak identification approach.

Q. If you look back at paragraph 160 in your declaration, page 101 of 292, the sample query work that you use there begins with peak, correct?

A. Yes, it does.

Q. So at least when you prepared this example you believed that a query string, according to Iwamura, could begin with a peak?

A. It would seem that way, so I'm just trying to verify that.

Q. If we assume that that is an admissible way to begin a query, then can you accept that this query of * 5, 4, 3, 2, 1 is an appropriate theory, according to Iwamura?

MR. LUNER: Objection to form.

A. Is it an appropriate way to start the query, then -- so with a peak note, yes, that would an appropriate query.

Q. And if you look down at the reference, do you see the string of numbers?

A. I do.

Q. And if we again assume that it's appropriate according to the teachings of Iwamura to begin a string with a peak, can

you accept, for purposes of this hypothetical, using that string as a reference?
A. Under all of the above ifs, yes, I do.
Q. And if our assumption about the ability to start a string with a peak is incorrect, then the examples in your declaration are also incorrect, right?
A. The particular query that I used in my declaration would be incorrect.
Q. And looking at the sample query on Exhibit 9, how many peaks does that query have?
A. Probably one.
Q. And which one is that?
A. That would be the one marked with a star or a 5.
Q. And how many peaks does the reference have?
A. That would be two.
Q. Which ones are those?
A. That would be 6 and 6.
Q. Okay. Look down at the grid depicted below. Do you see the reference

string in green at the bottom of the grid?
A. I do.
Q. And do you see the query grid depicted in various, the comparison number rows up above?
A. I do.
Q. We were using the peak search comparison process of Iwamura, which, if any, of the comparisons 1 through 16 would take place?
A. That would be comparison number 5, and that would be comparison number 11.
Q. The second one you said was?
A. 11.
Q. Now, during each comparison, what computation does Iwamura perform?
A. It computes the pairwise distance between the aligned positions, so the pairwise difference, absolute difference.
Q. So using, for example, the comparison between line 5 and the reference, you would first take the difference between 5 and 6; is that right?
A. Correct.
Q. And you would add that to the

difference between 4 and 2?
A. I believe so.
Q. And so forth until you're at the end of the query?
A. Uh-huh.
Q. Do you agree that if you were to perform that comparison and add up the numbers it would total 5?
A. Yes.
Q. And that's the total absolute difference that Iwamura uses to determine whether a reference is a match to a query or not --
A. Yes.
Q. -- is that correct? Okay.
So in the peak-by-peak -- pardon me, in the peak search process, comparisons number 1 through 4 in Exhibit 9 wouldn't take place, correct?
A. That would be correct.
Q. And comparisons 6 through 10 would not take place, correct?
A. That would be correct.
Q. And comparisons 12 through 16 would not take place?

A. That would be correct.
Q. And using this example query and reference, would Iwamura make any comparison that takes into account the one digit, digits in the reference sample?
MR. LUNER: Objection to form.
A. So your question was whether or not the notes marked with 1 would ever be compared?
Q. The question is whether the note in the string that reads 622,221,654321, were the ones --
MR. NEMEC: Strike that.
Q. In the reference string reading 622,221, will that 1 ever be compared to a note in the query?
MR. LUNER: Objection to form.
A. I'm sorry, the question is whether or not that one in the query could be compared to a note in the reference?
Q. Correct.
MR. NEMEC: No, strike that.
Q. The question is whether the 1 in the reference would ever be compared to any note in the query, and this is in the peak

search according to Iwamura.
A. Uh-huh.
So the 1 in the reference would not be compared to any note in the query.
Q. If you take a look at comparison number 12 --
A. Uh-huh.
Q. -- and compare the alignment of the query to the notes in the reference work, at that point, would you agree that the total absolute difference sums to zero?
A. I do.
Q. And is that representative of an exact match between the query and that portion of reference work?
A. That is representative of an exact match, yes.
Q. And in the peak-by-peak search process of the Iwamura, this comparison would not take place, correct?
MR. LUNER: Objection to form.
A. My understanding of how the peak-by-peak approach of Iwamura works, that comparison will not be performed.
(Discussion off the record.)

A. My recollection or understanding of how the peak-by-peak search approach works in Iwamura, that comparison will not be performed.
Q. So in that case Iwamura will not necessarily find the best matching melody segment in the reference; is that correct?
A. So in that case, you know, Iwamura will not find the best matching segment in the melody that has the peak notes aligned.
Q. And using the peak search approach of Iwamura, not all data in the reference is considered, correct?
A. That is correct.
Q. Another teaching in the Iwamura reference is that the user may elect not to consider unimportant portions of the reference melodies, correct?
A. Correct.
Q. Is there any direction or definition provided in Iwamura as to what would constitute an unimportant portion of a music reference?
A. To my recollection there is a very vague description of it. So there's a very

passing description of those from column 9, lines 48 to 50, that indicate that those are geologic send, user defined.
Q. So Iwamura doesn't provide any objective criteria for defining what would be an unimportant portion of the music sample, correct?
A. My recollection of the disclosure, yes, that's correct.
Q. That would be determined at the discretion of the person who is creating the reference database?
A. It's not actually clear. I mean what the specifications disclose, it usually identifies such important portions, that's key word. The other unimportant portions cannot be ignored. What is not clear over here is who is the user.
Q. Does the Iwamura reference require that the query represent an important portion of a music work?
A. Can you repeat your question.
Q. Do the teachings of Iwamura require that the query represent an important portion of the musical work?

A. I don't think the Iwamura, you know, have not had a requirement one way or the other.
Q. So if the reference database in Iwamura were constructed to omit the unimportant portions of the references, and the user constructed a query based on one of those unimportant portions of the music, the search wouldn't result in a match, correct?
A. So if the database does not have the portions of a melody in which, that the user searches, yes, that will not lead to a match.
Q. And let me actually ask that again, because I think I may have misstated how Iwamura operates.
If the reference database in the Iwamura embodiment is structured such that the unimportant portions are to be skipped and a query is based on one of the unimportant portions of a reference work, no match will be found, correct?
MR. LUNER: Objection to form.
A. So the database is structured and the search algorithm is structured to keep

the important portions from the melody from the reference work, and the query contains just the unimportant part?

Q. Correct.

A. Iwamura will still identify a match. It would identify the best match that the user's query matches against the database --

Q. So in that case --

A. -- and as determined by the peaks.

Q. In that case, it may return a match but not the closest or an exact match?

MR. LUNER: Objection to form.

A. In that case we will return a match that is the closest, given the data that the algorithm searches and how it performs a search.

Q. Even though an exact match, in fact, is present in the reference data set?

MR. LUNER: Objection to form.

A. So the way, my understanding of Iwamura, right, so when it goes and preprocesses the melodies to skip over the unimportant portions as part of the search process, you know, this is really what it

does, it does to some extent some type of future extractions. So when it searches, given that you just query, it identifies the best match, you know, with respect to the future that it has there, based on its searches. So in my opinion it will still return the best match, according to a search order.

Q. Now, does Iwamura state that the skipping of unimportant portions takes place in a preprocessing stage?

A. I believe so.

Q. Where is that?

A. I was reading.

So on column 12, lines 5 through 10, roughly that paragraph, it says, "When you build the database, by un-marking peaks, you can select the portion that should not be searched. This avoids searching unnecessary portions and accelerates search speed."

So this is during database building. That's processing step.

Q. And you understand that text in column 12 to refer to a future where unimportant portions can be skipped?

A. I believe that particular part, right, it actually refers to both the, there are two components in the Iwamura search that allows you to skip, right? One has to do with skip of, I believe they call them repeated patterns, so repeating melodies, and the other one is skipping the unimportant parts. So both of those things I believe they are tied together within the same context, and my understanding of this section on column 12, that essentially talks about the whole process of unmarking, because I believe they use a term unmarking, you know, when they were describing those, those approaches.

Q. Look, if you would, at column 8 in Iwamura, and I guess this begins on line 13 or 14, runs down to about line 20.

A. Okay.

Yes.

Q. What do you understand Iwamura to be describing here?

A. So the lines, so there are kind of two components there. So the first component, right, describes instead of

looking at peaks, to look at dips or more dips in here than peaks, as a way to further reduce the number of comparisons they make.

Q. I may have directed you to the wrong portion. I was asking about column 8, lines 13 through 20.

A. Oh, okay.

So my understanding of that, and again this is a very cryptic description is the following: Given a reference I have the different peaks. I have the highest peak, second highest peak and third highest peak. So given a query, then I will first align the highest peak of the query against the highest peak of the reference, you know, compute a score, right? Then align the highest peak of the query against the second highest peak of the reference and compute a score and so forth.

Q. How is it that that approach would further accelerate the search?

A. So that paragraph actually does not discuss how that can further accelerate the search. That's it.

Q. Is Iwamura suggesting here that if

you were to compare the highest peak in the query to the highest peak in the reference and find a match, you could discontinue the comparisons between that query and that reference, return that reference as a match?
A. I don't know how you disclose something like that.
Q. If you were to do that, that would accelerate the search, correct?
A. The question now becomes what we define by match. If the match over here is an exact match, and then, yes, that connects it. But if the match is not an exact match, it doesn't mean that there is no other peak-aligned match that would have that lower score, so that by itself is not a select search.
Q. What if it's a match within a certain threshold?
A. What do you mean by "a match within a certain threshold"?
Q. So in other words, if the highest point in the query is in line with the highest point in the reference, and the calculation yields at least absolute

difference of three or less, it's defined as a match.
MR. LUNER: Object to the form.
A. So what is defined as a match is that of identifying whether or not the reference, you know, contains some melody that is peaked aligned, who is, you know, distance is less than or equal to a user supply constant, right, then, you know, once you find one of those, then we can turn and search and identify that thing as a match.
Q. And by employing that approach you could accelerate the search, correct?
A. So by employing that approach, you can, you know, you will not need to identify, you know, search any other peaks that you've seen in the regular reference work.
Q. And thereby accelerate the search, correct?
A. That will speed up the search, yes.
Q. Let's turn over to column 9, if you would, and take a moment to look over the text from lines 24 through about 32.
A. Yes, I'm ready.
Q. What do you understand Iwamura to

mean there when it says, "Another peak can be used for the search"?
A. It's an example of what they have over there, they provide a clear example of what is the other peak. And in the first case, the peak would be the first one, which in the example it would be that star 5. And the second peak would be the star 5 in location 7.
Q. So what this indicates is that Iwamura doesn't always consider all the peaks in the query, correct?
A. What do you mean by "consider"?
Q. This indicates that if an exact match is found by comparing the first peak in the query to the peaks in the reference, the search won't continue on to another peaks in the query, correct?
A. So that rate of passage does not indicate that the search was stopped.
Q. So where it indicates in line 27, "In this case no exact result is obtained, so another search will be done with the second peak."
Do you see that?

A. Yes.
Q. Does that not indicate that if an exact match were found using comparison to the first peak in the query, that that second search wouldn't take place?
MR. LUNER: Objection to form.
A. So my identification of the paragraph is the fall line, so my understanding of that algorithm is when I use the first peak, right, I align it with a query against a reference, a computer score. If I go through the process by aligning, you know, shifting the first peak to all the peaks in the reference, if I have a -- that would give me a set of scores and a set of matches. So there is no exact match of me doing that. And then in the process of again trying to match a melody against that particular record, I would pick up a second peak and do the same thing.
Q. And if an exact match is found in what you just described, you would not move on that second search, correct?
A. That's what I reference, that is what I appear to have listed.

Q. Now, in the case using the peak search approach in Iwamura, where you have a short query and a longer reference melody, it's possible that the query would match the reference in multiple locations, correct?
A. That is possible, yes.
(Karypis Exhibit 10, Single-page chart, marked for identification, this date.)
Q. I want to take a look at an example sequence here that we will mark as Exhibit 10. Exhibit 10, do you see the top where it says "Query: 5, 4, 3, 2, 1"?
A. Yes.
Q. And can you accept hypothetically that that's a query string for use in a peak search, according to Iwamura?
A. I do.
Q. And below that it has a string of numbers and a reference, do you see that?
A. I do.
Q. Can you accept hypothetically that that is a reference melody string for a comparison, according to the peak search process of Iwamura?

A. I do.
Q. Now, if we were to compare the query 5, 4, 3, 2, 1 to the reference depicted on Exhibit 10, is the first instance of the sequence 5, 4, 3, 2, 1 in the reference a match for the query?
A. It is.
Q. And is the second instance of 5, 4, 3, 2, 1 highlighted in orange also a match?
A. Yes, it is.
Q. And is the yellow sequence 5, 4, 3, 2, 1 also a match for the query?
A. Yes, it is.
Q. And would the blue highlighted sequence 5, 4, 3, 2, 1 again be a match to the query?
A. It is.
Q. So in this example, is it fair to say that there are four exact matches to the query within the reference?
A. So within the reference there are four portions that match exactly to the query, yes.
Q. And in the case of this five-digit query, the match is the corresponding

five-digit sequence in the reference, correct?
A. I believe the match within the context of the Iwamura is a melody, so the match is really the reference.
Q. And you evaluate whether you have a matching melody by virtue of whether the query matches the melody segments within the reference, correct?
A. So you evaluate the quality of the match by evaluating how well the query matches, you know, the best matching segment in the reference.
Q. Let's look back at Iwamura in column 8, please, in particular lines 4 through 12 of column 8. And there Iwamura teaches that in addition to doing a peak search, you could also do a dip search, right?
A. Uh-huh.
Q. Now is it possible any given query or reference work that the best matching melody segment within that reference work would be at a dip?
A. What exactly do you mean by being

at a dip?
Q. Let me make sure that we're finding this the same way.
How do you understand the dip segments to be defined according to Iwamura?
A. My understanding of the definition of a dip is going to be a local minimum and a peak being a local maximum.
Q. So in the dip search process you would align a dip from the query with a dip in the reference, right?
A. That's my understanding of when it says you can use dips and instead of peaks, yes.
Q. And you would compare the notes as aligned in the reference -- pardon me, in the query to the reference aligned at the dip?
A. Correct.
Q. And is it possible that taking that query aligned at the dip would yield the lowest absolute difference of any melody segment in the work?
A. That is a possibility.
Q. Continuing with that hypothetical scenario, if you were then to take the query

Page 150

and compare it to the reference work using the peak search, you would not find that closest matching segment in the reference, would you?

A. Can you repeat that question?

MR. NEMEC: Sure. I'll just read it back off the record here.

Q. Continuing with that hypothetical scenario, if you were then to take the query and compare it to the reference work using the peak search, you would not find that closest matching segment in the reference?

MR. LUNER: Objection to form.

A. So my notion of what is the closest match and what is not is a function of the algorithm that you use to find it. So if I do a peak search approach, what I would return back as the closest would be the closest as a result of the peak search approach.

So with respect to that particular search strategy or algorithm, that's the best, you know, matching segment.

Q. Now, Iwamura teaches that you could also do the comparison using both peaks and

Page 151

dips, right?

A. There is a discussion about that.

Q. And in that scenario, the closest matching segment would be the comparison between either a peak or a dip in the reference that yields the least absolute difference when aligned with a corresponding peak or dip in the reference, correct?

A. So if I do a search in which I use both a peak search approach and also a dip search approach, and I return back the best match of both of them, right. So that would be the best match of the combined approach.

Q. And that best match might occur at a dip as opposed to a peak, right?

MR. LUNER: Objection to form.

A. So that best match can be obtained using a peak search approach or a dip search approach.

Q. And what I'm getting at is that in the approach where you're searching both the dips and the peaks, it's possible that the best match would be found at a dip, right?

MR. LUNER: Objection to form.

A. So when I'm doing a search using a

Page 152

peak search approach and a dip search approach, what you return at the end is the best of the two. The best result can be one that is coming from the dip search portion of the overall search.

Q. All right.

Now in that scenario we take that same query and that same reference, and we have found the best match at a dip. But we now compare that query to that reference using only the peak search, we're not going to find that best matching segment, are we?

A. Again, that goes back to the whole notion of the definition of a search and what is the result. But if I have an algorithm that is designed to find the best peak, let's call it aligned in the segment, right, the peak search approach will be the best. But if an algorithm is designed to find the best dip aligned segment, you know, that would be the best.

I mean for their respective algorithm, in terms of what they're designed to do, you know, what the -- the result that comes out from the dip search approach may

Page 153

not necessarily be an allowable result of the peak search approach. Because this is not what the algorithm is designed to do. But the peak search approach will identify the best match.

Q. So if the algorithm isn't considered, isn't configured to consider a particular match, then that can't be the closest match; is that right?

A. If an algorithm is designed to find a certain type of matches, right, it will not find a match that it is not designed to find.

So in your example, right, the type of match that you consider to be your best is a match in which an algorithm is designed to find. But the peak search approach will find the best match that it's designed to find.

Q. So if it's a match that you don't consider, it can't be the closest match, simply put, correct?

A. But this is a question of a definition of what is a segment that the peak search approach finds. Peak search approach will consider all the matches that satisfy its definition as to what it is after and

Page 154

among those will return the best.
Q. So I'll put it a little bit differently then. If, say, a portion of the reference that you don't consider in the search, then that portion of the reference, by definition, cannot be the closest match?
A. What do you mean when you say you cannot consider a portion of the reference? Is it something that you choose not to consider, or your algorithm, you know, specifically does not consider those things in identifying the allowable matches?
Q. Well, using the example of the peak versus the dip search in Iwamura, if you are running the peak search, you are not considering the dips, correct?
A. You may not be considering, not necessarily. You would still be considering some of the dips, if they include it in their peak.
Q. So that is what I'm trying to convey by the portion of the reference that's not considered, one example of it. Do you follow?
A. No.

Page 155

Q. Okay. Sounded like you did.
Why don't we go back to the two options that you proposed. In one instance you said the algorithm that's excluding a portion of the reference, and in the other you said the user is excluding, right?
A. In what context did I say that?
Q. A couple of minutes ago I think you made reference to it being a difference whether the algorithm excludes searching a part of a reference or whether the user does.
Iwamura describes a limit function that can be employed, correct?
A. Yes, I believe so.
Q. Take a look at column 7, lines 56 and 57.
A. I'm sorry, what was the line numbers again?
Q. Column 7, lines 56 to 57.
A. Yes.
Q. So in practice, if we're running a peak search and we're using this limit function, the search will include, excuse me, any peak segments with error above the threshold that we select, correct?

Page 156

A. That's my understanding of that.
Q. And if every peak segment in that reference that's compared has an error above the limit we've defined, then that particular song will not be considered a match, right?
A. Would you repeat your question again?
Q. If all the peak segments within a reference, have an error above the limit that we've defined, then Iwamura won't consider that reference to be a match, correct?
A. I don't think Iwamura discloses that.
Q. Which part of what I said?
A. That if -- everything that you just said.
Q. How is it that you understand the limit function to operate?
A. So my understanding of the limit function is we've done -- when I called the alignment of the query against that portion of the reference, as I go down each aligned position and compute the actual difference, if that thing, you know, goes above a certain threshold, then I can, you know, stop going

Page 157

down the path, down note by note to compute the difference, the actual difference, and just shift to the next peak.
Q. And then when you shift to that next peak, if you continue doing the calculation, and with each peak comparison throughout that reference you hit the threshold at the end of that process, that reference will not begin to match, correct?
A. Actually Iwamura does not disclose what happens in that scenario.
Q. Do you have any understanding, based on the overall teachings, of what would happen to that particular reference?
A. So one possibility is to not consider that reference as a match.
Q. Is there any teaching in Iwamura on a limit on how strict this error tolerance, pardon me, this limit function can be?
A. I don't recall off the top of my head whether or not they say anything specific to that.
I do not recall that it says something like that, a formula to describe what that thing should be.

Q. I apologize, I didn't hear you. You said you do or don't recall seeing --
A. I do not recall seeing whether or not they provide any guidelines as to what that thing would be.
Q. Do you think it's possible that the limit function could be set at zero?
A. Well, if the limit function is set to zero, then the algorithm that they describe would only identify exact matches if such matches occur, if my understanding is correct.
Q. Would you agree that the search time in the peak note search process of Iwamura is proportional to the number of peaks in the reference work?
A. So the search, the amount of time it takes to search a particular record in the peak search approach, right, will depend, among others, on the number of peaks.
Q. And you believe it would be roughly a proportional relationship between the search time and the number of peaks?
A. The more the peaks, the more time, yes.

Q. And the less peaks, the less time, correspondingly?
A. Correct.
MR. LUNER: I think we're running low on caffeine.
MR. NEMEC: I have a couple more questions on this reference and I will be done, so I'll wrap up the line and we will take a coffee break. Excellent.
Q. One of the file formats that Iwamura discusses is the mini file format, correct?
A. Correct.
Q. Just briefly, how would you characterize what the mini file format is?
A. I'm not an expert on the mini file format, but my understanding is this is a media file format. It predates, you know, MP3 files.
Q. There is also a description or a disclosure of a wav file format in Iwamura, correct?
A. Uh-huh.
Q. How does the wav file format differ from mini, if you know?

A. I do not know.
MR. NEMEC: We will take our coffee break.
THE VIDEOGRAPHER: Okay. The time is 2:40 -- stand by.
The time is 2:48. We're going off the record. This is the end of disk number 3. Thanks.
(A brief recess was taken.)
THE VIDEOGRAPHER: We are now back on the record. The time is 3:01. This is the beginning of disk number 4.
MR. NEMEC: Let's mark as Exhibit 11 U.S. Patent 5,874,686, the Ghias, please.
(Karypis Exhibit 11, Photocopy of U.S. Patent No. 5,874,686, marked for identification, this date.)
Q. Do you have Exhibit 11, Dr. Karypis?
A. Yes, I do.
Q. And this is the Ghias prior art reference that you discuss in your declaration, correct?
A. Correct.

Q. Is it fair to say that Ghias discloses a system where you can hum a tune and the system will try to identify the song that you are humming?
A. Yes, it's identifying melody from a database.
Q. And it does that by comparing the relative pitch values from the hummed query melody to the relative pitch values of known reference melodies; is that right?
A. I believe so.
Q. Ghias discloses comparing a text representation of the notes in the query to a text representation of the notes in the reference, right?
A. That is correct.
Q. In particular, Ghias represents queries and references as strings of the characters U, S and D, right?
A. Yes.
Q. U, S and D referring to whether the note is the same, an up note or a down note, as compared to the preceding note?
A. I believe so.
Q. And what is your understanding of

how Ghias goes about determining whether a query matches a reference?

A. So my understanding of how Ghias goes about doing that is it performs a substring search of the string representing the query, again, the string representing a record in the database; and then based on that it computes a score, and then it returns, you know, some of those melodies as a result of that query.

Q. How does Ghias go about determining which reference most closely matches a query?

A. So I believe the way that they use it is they find a substring that has the least number of character differences when they, it is aligned in a gapless way.

Q. So -- I'm sorry, go ahead.

A. Actually, they can potentially allow insertions and deletions as well, but it's really a substring alignment.

Q. Okay. So by way of example, if a -- if a query string was SSS and a reference string was SSU, how would -- how would Ghias characterize the difference between those two?

A. So Ghias would characterize that those two strings have a one-character mismatch.

Q. And how would Ghias go about determining the difference between a string SSS and SSD?

MR. NEMEC: Strike that.

Q. What would Ghias characterize as the difference between SSS and SSD?

A. I believe will characterize as 1.

Q. If two references have the same number of mismatched characters as compared to a given reference, will Ghias consider them to be equal quality matches?

MR. LUNER: Objection to form.

MR. NEMEC: Let me ask that again, I apologize.

Q. If two queries have the same number of mismatched characters as compared to a given reference, will Ghias consider them to be equal quality matches?

A. As far as I know, you don't match a query to a reference. It's the other way around.

Q. Okay. Let me try that again,

because I'm getting myself tied in knots here.

If two -- if a query has the same number of mismatched references when compared to two different --

MR. NEMEC: Strike that.

Q. If a query has the same number of mismatched characters when compared to reference 1 as it does when compared to reference 2, will Ghias consider reference 1 and reference 2 to be equal quality matches?

A. So if the best, if a substring with the least number of mismatches of the query in reference 1, which is -- if the number of mismatches of the best matching substrate of query to reference 1 is the same as the number of mismatches of the best matching substring of query to reference 2, right, then, you know, I believe Ghias will consider those two things as equal quality melodies or references, good overall matches.

Q. So is Ghias able, in the instance of comparing SSS --

MR. NEMEC: Strike that.

Q. Can Ghias determine whether SSS is

a closer match to SSU than SSD is?

A. So the query is SSS, and the reference is SSU and SSD, so both references are one-character mismatch with a query. And if they, if the metric is in the number of mismatches, those two things will be, those two things would be, you know, the same, equally good matches.

Q. If SSU is the query and the two references are SSS and SSD, will Ghias rank those as equally good matches?

A. I believe so, the one-character difference between the query and the two references.

Q. One of the things that Ghias teaches is that it can output a rank list of matches, correct?

A. That is correct.

Q. If the search in Ghias' run has determined that there are ten references that differ by only one character, how would Ghias go about ranking those ten?

A. So if they rank least that -- so if the reference is that the query matches against had the same number of mismatches, then the, you know, ranking between those things in terms of which one is the best is both irrelevant and arbitrary. You know, all of them are equally good answers to that query, given those constraints.

Q. Now, it may be of that list of ten that one of the referenced melodies itself matches the hummed melody closer than another of the ten, correct?

MR. LUNER: Objection to form.

A. So can you repeat your question?

Q. So notwithstanding the fact that the search in Ghias yields ten references that differ by only one character, and that those ten references, as far as the Ghias search process is concerned, would be equally similar or equally different, the underlying songs --

MR. NEMEC: Strike that.

Q. The fact that a reference, the fact that a query in Ghias differs by the same number of characters as compared to two different references doesn't necessarily mean that the query song is an equal match to both of the reference melodies, correct?

MR. LUNER: Objection to form.

A. So again, within the method that Ghias discloses, right, a query is a string. So based on that query, it identifies, computes a matching scroll to the references. And if there are a number of references that have the same scroll, number of mismatched characters, right, for the purpose of the search, right now, for the purpose of the algorithm disclosed in Ghias, those things are equally good matches.

Q. So putting Ghias aside, if a query piece of music is represented by SSU, which referenced piece of music would be more similar to it, SSS or SSD?

MR. LUNER: Objection to form.

A. So if my query is SSS, and the two references is SSD and SSU.

Q. The query is SSU --

A. The query is SSU.

Q. -- and the references are SSS and SSD.

A. The method that Ghias discloses in which it measures the similarity in terms of just mismatches, those things would be

equivalent; but you can have what is called a weighted mismatch in which certain mismatches weighed less or more, but in which case can create an order is one of those strings.

But if I use a symmetric of similarity or distance between two strings, the number of mismatched characters, those two would be equally good.

Q. My question was separate from Ghias.

A. Okay.

Q. So if we have a piece of music that's represented by SSU, and we have another piece of music that is represented by SSS and another represented by SSD, which is more similar to SSU, SSS or SSD?

MR. LUNER: Objection to form.

A. How do you define similarity?

Q. How different the notes are.

A. Do you consider the -- if I look in the underlying note, that would be a different similarity than what I can compute by just having that another presentation, unless you define a distance between the characters of their presentation.

Q. Column 2 of Ghias discloses an error tolerance, if you look at lines 2 through 50 to 53?

Do you see that?

A. 50 to 53?

Q. I apologize, 52 to 53.

A. 52 to 53. Let me take a minute.

Yes, I see that line.

Q. What do you understand this error tolerance to mean?

A. My understanding of an error tolerance is the maximum number of allowed mismatches between the query and the best matching segment or melody.

Q. So it's not possible that the error tolerance would mean that the output is limited to a set number of matches or maximum number of matches?

A. What do you mean by that?

Q. In other words, is it not possible that the error tolerance could mean that the system only returns the ten best matches?

A. My understanding of the error tolerance, as used in Ghias is, is that it talks about, it's defined in terms of a

maximum number of allowed mismatches. I don't recall seeing it another way.
Q. So we discussed a moment ago, and let me just reconfirm, that one of the possible outputs from Ghias is a ranked list of matching melodies; is that right?
A. Yes, I believe so.
Q. Is it an unlimited list, or is there some cap that Ghias puts on the number of entries included on that ranked list?
A. So my recollection in reading Ghias, the size of the ranked list depends on the error tolerance. So if the error tolerance is said to be high, the ranked list would be larger, potentially; if it's low, it's going to be smaller.
I don't recall seeing a specification of the fixed number of results.
Q. So if, for example, the error tolerance were set at one, the ranked list would return all those references that had a difference of one as compared to the query?
A. What a search will return is all the references with best matching substring that has at most one difference. That means

it will return those that have zero and those that have one, assuming that they are non --
Q. Okay. Understood.
So what we're looking at, then, is the matches with the, the best matches with the substrings as opposed to the overall reference, correct?
A. Can you repeat your question?
Q. Sure.
MR. NEMEC: Let me state it differently.
Q. The error tolerance would refer to the difference between the query work and a substring within the reference?
A. My understanding is the way they talk about error tolerance is that the error tolerance is defined in terms of a best matching, is defined in terms of the best substring match, a query against the reference. If those two things are the same length, it would be a, one would be the same. It will not be a substring match, but if the reference is longer than the query, it will be a substring match.
Q. The top of column 7 in Ghias

explains that a user can perform a new query on a restricted search list consisting of songs just retrieved, correct?
A. Right.
Q. Could the new query that's described there be a different portion of the same song that was searched in the original query?
MR. LUNER: Objection to form.
A. That can be a possibility.
Q. Could the new query be a completely different song?
A. The segments can perform a new query and puts no restrictions on the type of a query.
Q. Now, if we were to return, pardon me. If we were to run a first search --
MR. NEMEC: Strike that.
Q. If we were to run a search according to Ghias, using the chorus from a song, would that necessarily return the same results as if we were to use the guitar solo from the same song as the query run against the reference database?
MR. LUNER: Objection to form.

A. It may or may not.
Q. So the results could be different?
A. It could be different, yes.
Q. Look at figure 1 in Ghias, if you would.
Is the query engine that performs the search process in Ghias --
A. Yes.
Q. -- item 24?
A. Uh-huh.
Q. So it's the query engine that identifies matches?
A. Yes.
Q. And then those matches are output as a ranked list of matching melodies at item 26; is that correct?
A. Based on the result of those matches, the melodies, you know, that satisfy the search criteria are ranked at output, yes.
Q. Do you have an understanding of what form that output would take?
A. What do you mean? Can you clarify your question?
Q. Sure.

Is the user going to get a list of songs by title? Is that your understanding of what Ghias is teaching?
A. I don't recall what is the form of the output, but, you know, a list of titles is a potential output.
Q. We know it's coming in list form, right?
A. Well, it's a ranked list.
Q. If you will look at column 7, lines 36 through 40 of Ghias, please.
A. You said column 7, lines --
Q. 36 to 40.
A. Yes.
Q. So he says there, "As a consequence of using a fast approximate string matching algorithm, search keys can be matched with any portion of the melody rather than just the beginning. As the size of the database grows larger, however, this may not prove to be an advantage."
What is your understanding of why that would not prove to be an advantage?
A. I will presume that by enabling a flexibility to locate a substring within a

bigger string, you know, that would be somewhat slower than looking at just the beginning.
Q. How do you believe a person skilled in the art would circumvent this disadvantage?
MR. LUNER: Objection to form.
A. So if the type of the query that I would like to do is identifying all substrings within a string that can have optic k-mismatches. There are a set of methods that have been developed primarily in searching sequences, DNA sequences that use efficient methods to index strings that would allow you to do fast approximate substring matches.
I can describe them in detail.
Q. That's not something that's disclosed in Ghias, what you just described?
A. I do not think so.
Q. Is that a technique that was known as of 2000?
A. I believe those methods were developed prior to 2000.
Q. Turning back to the description in

Ghias, do you think it's a desirable feature that search keys could be matched with any portion of the melody rather than just the beginning?
MR. LUNER: Objection to form.
A. I can consider cases in which my goal is to find a melody based on something that was hummed on the radio, into a microphone, and they start humming the melody from an arbitrary point within the melody; so doing a substring match, you know, will increase the chances of identifying the melody or not.
Q. Does the fact that the search described here in column 7 of Ghias is an approximate search mean that it doesn't consider all the data within each reference?
MR. LUNER: Objection to form.
A. So I believe the definition of "approximate" that Ghias -- the term "approximate string matching" over here refers to what would be considered a match, and a match is considered a valid match if it has at most K mismatches. So the term "approximate," right, it used to characterize

what is allowed as part of the answer.
Q. So in fact at the top of column 6, lines 9 through 11, Ghias defines when he means by "approximate," right?
A. I believe so.
Q. And he says, "Approximate" -- "By approximate is meant the algorithm should be able to take into account various forms of errors."
Do you see that?
A. That's what it says, yes.
Q. Do you think, from the standpoint of a person skilled in the art in 2000, that's a reasonable definition of "approximate" in this context?
A. The definition of "approximate" being that, something that will allow a certain amount of mismatches, I think that's a reasonable definition.
The definition that Ghias uses is, in terms of the way it formulates the search, is actually more precise. So what it calls approximate string matching is the matching that allows at most k-mismatches.
Q. So you say it's more precise.

Do you think it's an unusual or unacceptable definition?
MR. LUNER: Objection to form.
A. I think this is one way of defining approximate stream matching.
MR. NEMEC: Let's mark as Exhibit 12 US patent 6,970,886.
(Karypis Exhibit 12, Photocopy of U.S. Patent No. 6,970,886, marked for identification, this date.)
Q. Is Exhibit 12 the Conwell prior art reference that is discussed in your declaration?
A. Yes, it is.
Q. Is it fair to say that Conwell describes a system which identifies a query work by hashing the query work and comparing it to hashes and known reference works?
A. That's a fair characterization.
Q. And Conwell can look at a reference work that is a near match to a query work, if the two map the same hash value, correct?
MR. LUNER: Can you repeat the question?
MR. NEMEC: Sure.

Excuse me.
Q. Conwell can look at a reference work that is a near match to a query work, if the two map to the same hash value, correct?
A. By using appropriately designing hash functions, yes.
Q. So in the Conwell system, a determination is made as to whether two works match by comparing whether their hash values are exact matches?
A. Can you repeat the question again?
Q. Conwell determines whether two works match by comparing whether their hash values are exact matches, correct?
A. That is correct.
MR. NEMEC: Can I take a quick five minutes? I apologize.
MR. LUNER: Sure.
THE VIDEOGRAPHER: The time is 3:43. We're going off the record.
(A brief recess was taken.)
THE VIDEOGRAPHER: The time is 3:53. We're now back on the record.
Q. Dr. Karypis, if the database in Conwell doesn't contain a reference to

exactly the same hash as the query, will Conwell locate any match?
A. So if the database has no record that it has the exactly the same hash, I believe Conwell will not locate a match.
Q. And if the database in Conwell contains a reference that is very close match to the query but matched to a slightly different hash, will Conwell determine that that's a match?
A. When you say "a close match to the query," are you talking about in terms of the hash, or in terms of the actual melody?
Q. So both. I'll ask the question again, just so it's clear.
If the database in Conwell contains a reference, it's a very close to match to the query but maps to a slightly different hash, will Conwell determine that that's a match?
A. So if the hash of the query is different than any of the hashes in the database, the Conwell will not determine a match.
Q. Any difference whatsoever in the

hashes will result in a no match determination in Conwell; is that right?
A. If there is no absolutely no entry in the database that has exactly the same hash value as that computed for the query, it will have no match.
Q. Generally speaking, would you consider a search of a dictionary, a paper dictionary, to be a non-exhaustive search?
MR. LUNER: Objection to form.
A. What do you mean by a search of a dictionary, of a paper dictionary?
Q. Looking up a word in the dictionary.
MR. LUNER: Objection to form.
A. That would depend on how you go about doing that.
Q. So what are the different ways that one might look up a word in the dictionary?
A. Well, one way to do that is start from page 1 and go down until you find the page on which that word occurs, or go to the end of the dictionary. That would one way to do it.
Another way to do it would be, you

know, if the dictionary is alphabetized, to go to the set of pages that starts with the same character as your word and then go down that list until you find something.

Q. And would either of those searches be a non-exhaustive search?

A. So the second way in which you directly go to the pages that, let's say start with, your word starts with a K, you do not examine any pages prior to the one that start with the K, you know, that would be a non-exhaustive search.

The first way is not.

Q. Is that why --

MR. NEMEC: Strike that.

Q. In the paper dictionary example that we're discussing, the words are arranged in alphabetical order before your lookup process begins, correct?

A. That would be the way why people do it.

Q. So if someone asked you to go look up a word in the dictionary, which of the two approaches that you just described would you use?

MR. LUNER: Objection to form.

A. Well, I will do another approach that would be more efficient, and that would be the one that will, you know, go the pages containing the words, go to the pages that contain the words that start with the same character as my query.

Q. And that's the approach that you characterized as non-exhaustive, correct?

A. That would be the one I characterized, yes.

Q. Now if we're talking about an electronic dictionary, would a search of an electronic dictionary to look up a word use a binary search?

A. What do you mean by "electronic dictionary"?

Q. So as opposed to a paper dictionary, a collection of words and their associated definitions arranged in a database.

A. Without any more information, I will just go from record one until I find a match or find a word or until I go to the last record.

Q. If you were constructing an electronic database, would you order the entries in some fashion?

A. If I have a database and I would like to be able to perform certain type of queries efficiently, I would probably create indices that would allow me to do those efficiently.

Q. So you would create an ordered database, in other words?

A. Actually, there is no such thing. This is -- you don't necessarily create an order database. No, what you create is you create an index on top of your database that will allow you to, to transverse it in an order that is up to modify the inquiry.

Q. Okay. In general, the purpose of creating such an index would be to make the search more efficient?

A. That is the purpose of most indexes, yes.

Q. Now, in Conwell, the reference table as disclosed has its entries sorted by hash value, correct?

A. Can you tell me which figure you're

referring to?

Q. Sure. I'm referring actually to the text, column 5, lines 59 through 61.

A. Column 5, 59 through 61.

Q. Correct. "For ease of description, the present disclosure assumes the entries are sorted by identifier."

A. Yep. What was the question? I'm sorry.

Q. So do you understand from that that the disclosure here is of a reference table with entries sorted by their hash value?

A. Yes, I believe that's how they have them sorted, yes.

Q. And what is the purpose of sorting the hash entries?

A. So one of the reasons that you like to sort the hash entries is so that it will allow you to potentially implement a search efficiently.

Q. In the fashion that you would, in a paper dictionary, look up a word by its first letter in the word?

MR. LUNER: Objection, form.

A. Well, the keys are sorted in

Page 186

increasing or decreasing order, right? Now I would like to match a key, and if I want to leverage the track of how those things are sorted, I will do a, probably a binary search.
Q. A binary search being an example of a non-exhaustive search, correct?
A. That is correct.
Q. Now, lines 58 to 59 of Conwell in the same column there, column 5, it says, "Maintenance to the table 12 is well understood by those skilled in data structures."
Do you see that?
A. Yes, I do.
Q. And as of about 2000, you think a person skilled in the art of the Cox patents would have been familiar with database structures?
A. Actually, it talks about data structures, not database structures.
Q. Oh, I apologize, okay.
So I will ask that question again.
As of 2000 do think a person skilled in the art would be familiar with

Page 187

data structures?
A. Yes.
Q. And do you think that a person of skill in the art implementing the invention described in the Conwell patent would turn to some form of commercially available data structure software?
MR. LUNER: Objection to form.
A. That can be a possibility, they can use a commercially available software to build it.
Q. As of 2000, are you familiar with any of the commercially available database software that was on the market?
A. Both commercial and open source, yes.
Q. And generally speaking, did such database software employ exhaustive lookup techniques?
A. That would be a direct function on how you choose to configure it. You set up a database, without creating indices. That's the only way to answer the queries that you want would be to add an exhaustive search. Set up a database by creating indexes at

Page 188

which point in time the querying processing engine can make a decision on whether or not that would be more efficient or not, to use the indices or not.
Q. If you made the decision that it would be more efficient to use the indices, that you would be using a non-exhaustive search, correct?
A. So if the indices were enabled, then during querying processing time, the database engine would make a decision whether or not answering that query would be faster relying on the indices or not. And if it's faster, will use the indices; if it's not faster, it will not.
Q. I see.
(Discussion off the record.)
Q. Take a look at Exhibit 4. It's the '179 patent. Take a look at column 21, please, lines 37 to 42.
A. I'm sorry, what were the line numbers again?
Q. Lines 37 to 42.
A. Okay.
Q. The passage that begins, "While

Page 189

databases of this size"?
A. Yes.
Q. Would you agree that here this specification in the Cox patent is explaining that databases that were known as of the date of invention look up entries without comparing a query to all reference items?
MR. LUNER: Objection to form.
A. Okay, so what was the question again?
Q. Would you agree that the specification in the Cox patents here in column 21 is explaining that the databases that were known as of the date of invention, date of invention look up entries without comparing a query to all reference items?
A. So what it here describes, describes that databases have the technology to do that.
Q. So that was an understood approach to database searching as of 2000 when this patent was filed, correct?
A. So as I said before, if I'm using a database system and I would like to perform certain queries, if I can create indices for

those queries, then the database system is intelligent enough to select to use those indices, and that will to lead to a faster query execution time.

Q. Thank you.

MR. NEMEC: Off the record again.

THE VIDEOGRAPHER: Sure. The time is 4:11. We're going off the record.

(A brief recess was taken.)

THE VIDEOGRAPHER: The time is 4:23. We're now back on the record.

MR. NEMEC: Thank you, Dr. Karypis. No further questions from petitioner.

MR. LUNER: Patent owner, Network-1 doesn't have any questions.

THE VIDEOGRAPHER: Okay. Time is 4:23. We're going off the record. This is the end of disk number 4, and that concludes this deposition.

(Time noted: 4:23 p.m.)

STATE OF _______________)

) :ss

COUNTY OF _______________)

I, GEORGE KARYPIS, the witness herein, having read the foregoing testimony of the pages of this deposition, do hereby certify it to be a true and correct transcript, subject to the corrections, if any, shown on the attached page.

____

GEORGE KARYPIS

Sworn and subscribed to before me, this day of , 2015.

____

Notary Public

Job No. CS2183243

C E R T I F I C A T E

STATE OF NEW YORK)

: ss.

COUNTY OF NEW YORK)

I, Jennifer Ocampo-Guzman, a Notary Public within and for the State of New York, do hereby certify:

That GEORGE KARYPIS, the witness whose deposition is hereinbefore set forth, was duly sworn and that such deposition is a true record of the testimony given by the witness.

I further certify that I am not related to any of the parties to this action by blood or marriage, and that I am in no way interested in the outcome of this matter.

IN WITNESS WHEREOF, I have hereunto set my hand this 13th day of November 2015.

____

JENNIFER OCAMPO-GUZMAN, CRR, CLR

ERRATA SHEET
VERITEXT LEGAL SOLUTIONS
800-567-8658
ASSIGNMENT NO. CS2183243
CASE NAME: Network-1 Technologies, Inc. v. Google
DATE OF DEPOSITION: 11/12/2015
WITNESS' NAME: George Karypis

PAGE/LINE(S)/ CHANGE REASON
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________
____/_______/_________________/__________

____
George Karypis
(Notary not required in California)
SUBSCRIBED AND SWORN TO
BEFORE ME THIS______DAY
OF______________, 2015.

____
NOTARY PUBLIC
MY COMMISSION EXPIRES_________________

Veritext Legal Solutions
290 W. Mt. Pleasant Ave. - Suite 3200
Livingston, New Jersey 07039
Toll Free: 800-227-8440 Fax: 973-629-1287

______________, 2015

To: Charles R. Macedo, Esq.

Case Name: Network-1 Technologies, Inc. v. Google

Veritext Reference Number: 2183243

Witness: George Karypis Deposition Date: 11/12/2015

Dear Sir/Madam:

Enclosed please find a deposition transcript. Please have the witness review the transcript and note any changes or corrections on the included errata sheet, indicating the page, line number, change, and the reason for the change. Have the witness' signature at the bottom of the sheet notarized except in California where they are signing under penalty of perjury and forward the errata sheet back to us at the address shown above.

If the jurat is not returned within thirty days of your receipt of this letter, the reading and signing will be deemed waived.

Sincerely,

Production Department

Encl.

cc: Douglas Nemec, Esq.

&

& 2:13 3:4,14 4:3 6:15 7:9,11

0

07039 194:2

1

1 1:6 5:6 6:20 7:8,12 9:23 11:4 23:14,17 47:3 64:10 71:24 114:10,16,18,22 116:24 117:2 118:17,19,20 124:23 125:12,20 128:16 129:14 131:9 132:18 133:8 133:15,23 134:3 146:13 147:3,5,9,12 147:15 163:10 164:9,10,14,16 173:4 181:21 190:15 193:3 194:6
1,000 104:17
10 5:18 125:13 132:21 139:16 146:7,12,12 147:4
100 47:6 48:1,8 50:22 57:4 58:2 60:15
100,000 101:13 102:5,7,11
10016 4:6
10038 3:7
101 124:1 129:2
108 5:14
10:36 64:9
10:38 64:7
10:48 64:13
11 2:8 5:19 106:17 106:19 107:10 124:20 125:11 131:12,14 160:14 160:16,19 177:3
11/12/2015 193:3 194:8
112 5:15
12 1:13 5:21 6:13 132:24 134:6 139:15,24 140:11 148:16 178:7,8,11 186:11
123 5:16
128 5:17
12:09 106:10
12:20 106:14
12:28 111:3,4
13 108:23 125:23 140:17 141:6
13th 192:20
14 100:20 140:18
146 5:18
15 95:11,16 96:15 97:16,22 98:7 109:17
16 131:9 132:24
160 5:20 115:22 124:12 129:1
161 115:22
162 121:21 122:15
178 5:21
179 188:19
18 125:24
1:17 112:2,4

2

2 5:7 24:16,16 25:2 25:4 64:14 68:16,24 69:15 95:7 106:11 114:10,16,18,22 116:24 117:2 118:17,19,20 128:16 129:14 132:1 146:13 147:3 147:5,9,12,15 164:10,11,18 169:1 169:2
20 120:2 124:23 125:11,20 140:18 141:6
2000 26:4,9,12,21,25 27:1,9 37:3,9,12,23 38:15,19 39:2,9 40:9 78:21 79:5,8 81:24 83:6,12 84:17 85:3,12 94:14,17,20 94:24 110:2,10,13 110:20 175:22,24 177:13 186:16,24 187:12 189:21
201 3:16
2015 1:13 2:8 6:13 109:17 191:19 192:21 193:22 194:4
2015-00343 1:4
2015-00345 1:4
2015-00347 1:5
2015-00348 1:5
21 100:19 188:19 189:13
212 3:8 4:7
2183243 194:7
23 5:6
237 100:18
24 143:23 173:9
25 47:13 50:23 69:19 104:13
26 95:9 97:7 120:16 173:16
27 144:21
290 194:1
292 124:1 129:2
2:40 160:5
2:48 160:6
2n 24:13

3

3 5:9 68:18 69:2 76:8 77:5 100:18 106:15 114:10,16 114:18,22 116:24 117:2 118:17,19,19 128:16 129:14 146:13 147:3,5,9,11 147:15 160:8
30 120:17
31 97:5
310 3:18
32 143:23
3200 194:1
336-8074 4:7
343 6:23
345 6:23
347 6:23
348 6:23
35 120:6
36 174:11,13
37 188:20,23
39 42:19
3:01 160:11
3:43 179:20
3:53 179:23

4

4 3:6 5:10 24:17 25:2,4 68:20 69:5 114:10,15,18,22 116:24 117:2 118:5 118:17,19,19 128:16 129:14 132:1,18 146:13 147:3,5,8,11,15 148:15 160:12 188:18 190:19
40 174:11,13
42 188:20,23
45 120:3,8
46 97:6
48 136:2
4:11 190:8
4:23 190:12,18,21
4n 24:13

5

5 5:12 68:22 69:8 114:10,15,18,22 116:24 117:2 118:4 118:5,15,15,17,18 118:19,21 128:16

Google Ex. 1020

129:14 130:18 131:11,21,22 132:8 139:15 144:7,8 146:13 147:3,5,8,11 147:15 185:3,4 186:10
5,874,686 5:20 160:14,17
50 136:2 169:3,5
52 169:6,7
53 169:3,5,6,7
56 155:15,19
57 155:16,19
58 186:9
59 185:3,4 186:9

6

6 5:13 91:10 108:22 108:25 109:3 110:9 114:18,22 117:2 118:5,17,19,24 130:23,23 131:23 132:21 177:2
6,188,010 5:15 112:8,10
6,970,886 5:21 178:7,9
60 91:9,11 100:3
61 105:14 185:3,4
622,221 133:15
622,221,654321 133:11
656-7066 3:18
68 5:8
69 5:9,11,12

7

7 5:3,15 95:17 112:6 112:7,9,15 119:25 120:12 144:9 155:15,19 171:25 174:10,12 176:15
735-2419 3:8
75 47:12,20 48:1,4,8 50:21 60:16,17,21

8

8 5:16 123:8,10 140:16 141:5 148:15,16
8,010,988 1:9 5:8 68:17,25
8,056,441 1:10
8,205,237 1:9 5:9 68:19 69:3
8,640,179 1:8 5:11 68:21 69:6
8,656,441 5:12 68:23 69:9
80 47:2 60:15
800-227-8440 194:2
800-567-8658 193:2
83 71:24,25 72:1

9

9 5:17 69:17 76:5 101:4 104:16 105:7 128:9,10,13 130:13 132:18 136:1 143:21 177:3
9/15/2015 109:18
90 2:13 4:5 6:16 9:16 101:5,7,11,25 103:10 104:24
90,000 103:10
90401 3:17
97 123:25
973-629-1287 194:2
98 122:2
988 69:15 95:8,17 96:15 97:5 98:7
9:05 1:14 2:9 6:14

a

a.m. 1:14 2:9 6:14
abc 72:3,4,13,17,22 72:22 73:4 74:17,22
ability 8:11 130:7
able 48:1 60:16 82:13 96:7 108:3 164:22 177:8 184:5
absence 38:3
absolute 127:1,8 131:19 132:10 134:11 142:25 149:21 151:6
absolutely 181:3
accelerate 141:21,23 142:9 143:13,18
accelerates 139:20
accept 128:17 129:14 130:1 146:15,22
account 49:13,17,21 50:5 77:23 133:4 177:8
accurate 18:25 33:15 43:14 116:5
accurately 8:7 85:5
action 95:13,22 96:1 96:4,13,14,18,19,21 97:10,15,18,21,22 98:6,15,19,23,25 99:3,7,9 192:15
actions 19:14 96:20 97:17 98:12
actual 12:19 14:21 25:18 32:16 156:23 157:2 180:13
add 131:25 132:7 187:24
addition 82:18 148:17
additional 97:11,19 97:25 98:3
address 194:13
addressed 82:8,9
addresses 19:10
addressing 19:7
admissible 129:13
advantage 174:21 174:23
afraid 66:14
agnostic 107:25
ago 22:8 25:11 70:6 119:5 155:8 170:3
agree 6:9 94:8 112:21 132:6 134:10 158:13 189:3,11
agreed 14:12
agreement 85:20 86:25 100:4
ahead 20:16 23:13 40:5 68:12 100:24 128:8 162:17
algorithm 26:17 28:21 40:22 41:23 42:3,5 47:12,19,23 47:25 48:3,11 50:15 50:20 58:17 65:1,2 65:7,17,19 66:2 70:18 79:23 95:5 113:16 128:22 137:25 138:16 145:9 150:16,22 152:15,19,23 153:3 153:6,10,15 154:10 155:4,10 158:9 167:10 174:17 177:7
algorithms 26:21 41:1,4,7,9,12 63:21 63:22
align 117:6,15 118:2 118:4,5,15,22 119:1 141:13,16 145:10 149:10
aligned 131:18 135:10 142:15 143:7 149:16,17,20 151:7 152:17,20 156:22 162:16
aligning 145:12
alignment 134:8 156:21 162:20
alignments 117:11
alleged 15:3
allow 39:5 51:3 59:11 76:17 77:2 79:23 128:22

162:19 175:15 177:17 184:7,15 185:19
allowable 153:1 154:12
allowed 169:12 170:1 177:1
allowing 103:24
allows 140:4 177:24
alpha 84:1
alphabetical 182:18
alphabetized 182:1
amount 24:14 121:18 158:17 177:18
amster 2:12 4:3 6:15 7:11
analogy 54:6
anchor 121:17
andrew 3:11 7:5
andrew.gish 3:12
answer 9:9,11 16:25 20:5 24:2 27:21 28:13,23,25 38:13 43:17 52:13 59:19 59:22 60:12 76:20 82:13 85:1 110:3,18 115:21 177:1 187:23
answering 188:12
answers 166:4
anticipated 11:14 15:4
anticipation 16:19
apart 24:6 86:16
apologize 65:4 158:1 163:17 169:6 179:17 186:22
appeal 1:1 6:22 10:24 42:8
appear 145:25
appearances 4:1 7:2
appears 25:15 78:9 78:16,19 93:24 99:17 124:15,20
appended 57:4 58:2
applicable 15:9,16
application 79:13 80:12,17 81:7
applications 26:5 79:10 80:21
applied 48:18 91:12
apply 30:7,13 32:11 36:9 108:3
applying 35:18 40:1 87:25 88:9
approach 27:1 48:16,19 51:14 76:22 104:4,5 113:16 119:5,12,22 128:25 134:23 135:2,11 141:20 143:12,14 146:2 150:17,20 151:10 151:11,13,18,19,21 152:1,2,18,25 153:2 153:4,16,23,23 158:19 183:2,8 189:20
approaches 61:3 103:8 140:15 182:24
appropriate 51:11 55:24 106:2,4,6 129:15,17,19,24
appropriately 179:5
approximate 93:23 94:10,15,18,25 95:3 174:16 175:15 176:16,20,21,25 177:4,6,7,15,16,23 178:5
approximately 6:14
arbitrary 166:3 176:10
area 18:20 56:21 107:15 108:8,9
areas 108:10,12
arelaw.com 4:9
arps 3:4 7:4,6
arranged 182:17 183:20
arrangement 62:11
array 26:17 27:18 27:21 28:2,3,5,6,8 28:10,18,20 29:5,12 29:25 30:1,6,8,10 30:15,24 31:1,11,24 32:13
arrow 122:19
art 11:14,18 12:3 15:4 17:9 18:6 26:7 33:16 38:19 44:20 46:22 67:5,10 79:4 82:4 83:5,11,17,21 85:11 90:1,23 91:4 91:6 106:21 107:1 108:16 112:17 160:22 175:5 177:13 178:11 186:17,25 187:4
aside 13:15 50:19 53:19 94:22 167:12
asked 37:19 182:22
asking 39:16 58:20 58:25 76:3 83:4 84:8 89:19 117:20 123:16 141:5
aspects 14:7
asserting 19:25
assignment 193:2
associated 30:2,10 30:14 32:2,21 34:14 34:18 35:5,17,23,25 39:24 51:20 52:12 55:11 56:4 58:9,12 58:14 63:24 67:25 108:4 183:20
association 97:12,20
assume 27:18,19 55:16 114:9,15,16 114:22 116:20 129:12,23
assumes 185:6
assuming 56:1 86:9 116:3 171:2
assumption 105:2 114:19 130:6
attached 191:12
attention 119:25
attorneys 3:5,15 4:4
audio 6:8 21:24 38:8 80:18 81:16,22 108:5
authored 46:5
available 32:1 68:14 187:6,10,13
ave 194:1
avenue 2:14 4:5 6:16
avoid 9:6
avoids 139:19
aware 37:8

b

b 75:4,5 95:18
back 20:16,17 32:17 39:20 53:18 57:4,4 57:14 58:2,2 60:14 63:18 64:13 71:23 103:20 106:14 112:4 119:15 121:21 129:1 148:14 150:7,18 151:11 152:13 155:2 160:10 175:25 179:23 190:12 194:13
background 8:20
backtrack 29:13
base 103:11
based 19:5,14 36:10 49:2 51:12,16 55:3 55:6,23 83:4 94:7 98:15 99:1,3 119:12 125:2 137:7,20 139:5 157:13 162:7 167:4 173:17 176:7

Google Ex. 1020

baseline 116:18
basically 113:10
basis 74:2 103:24
beginning 64:14 106:15 113:20 117:14 128:23 160:12 174:19 175:3 176:4
begins 91:15 120:2 129:3 140:17 182:19 188:25
behalf 9:22 11:4
beliefs 14:9
believe 10:10 12:8 12:24 13:20 15:1 19:18 20:6,21 21:13 22:9,24 25:8,17,25 26:1,12 37:12,14 38:2 41:9 42:23 43:5 46:10,21 47:1 71:14 72:23 74:18 78:14,17,20 79:2 80:15 82:3,7,9 85:20 87:6,13,14 90:13 92:8,16 96:2 96:16 98:6 99:19 100:16 104:16 105:12 108:19 109:7 112:6 113:24 114:7,13 115:8,22 116:4 117:9,12,19 121:4 122:1 127:3 127:21 128:6 132:2 139:12 140:1,5,8,13 148:3 155:14 158:21 161:11,24 162:13 163:10 164:19 165:12 170:7 175:4,23 176:19 177:5 180:5 185:13
believed 129:7
best 66:22 84:21 86:1,6,9,14 87:10 89:15,16 91:2 114:4 135:6,9 138:6 139:4 139:7 148:12,22 150:23 151:11,13 151:14,17,23 152:3 152:3,9,12,16,18,19 152:21 153:5,14,17 154:1 164:12,15,17 166:2 169:13,22 170:24 171:5,17,18
better 87:18
beyond 33:18 73:15
big 5:14 108:24 109:1 110:8
bigger 175:1
binary 26:18 27:12 27:13,17 28:10,11 28:14 29:2,4,11 37:6 183:15 186:4,6
bit 17:17 39:20 53:19 71:17 97:4 154:2
blind 63:11
blood 192:16
blue 147:14
board 1:2 6:22 10:20,24 42:8,12 93:8,12 94:2 99:20 100:7
board's 85:21 92:10 92:18 93:6 94:23
bonded 30:5
bottom 109:16 122:2 124:17 131:1 194:12
boulevard 3:16
branch 54:11,13,17 55:25 56:4,6
branches 54:4,24 55:12 56:23,25
break 9:13,15 64:5 110:25 159:9 160:3
brief 64:11 106:12 160:9 179:21 190:10
briefly 27:16 159:14
broad 20:20 99:12
bucket 51:16,19,23 52:4,7,15,17,20,22 53:3,7,13 55:25 56:3,5,6
buckets 51:11,14 53:3
build 139:17 187:11
building 139:22
bunch 20:7 21:2 37:18 41:3
burdens 16:14
business 80:22
byte 45:5,15,19
bytes 104:17

c

c 3:2 75:5 95:11,19 95:20 192:1,1
caffeine 159:5
calculation 142:25 157:6
california 3:17 193:21 194:12
call 15:2 40:2 83:17 85:13 140:5 152:17
called 7:19 36:19 77:8 85:11 113:22 156:20 168:1
calls 177:22
cap 170:9
caption 6:19
career 46:1,16
case 6:19,22 11:1 14:18 15:12 18:15 49:15 59:14 64:20 71:14,15 74:18 75:11 83:2 86:17 88:2 94:23 106:22 107:14 126:25 135:5,8 138:9,11,14 144:6,22 146:1 147:24 168:3 193:3 194:6
cases 176:6
categories 11:24 41:6
category 12:2 51:1
cc 194:25
cellphones 6:6
center 115:11,12
centered 115:1
centering 116:12,21 126:18,23 127:6,7
certain 11:5 14:7 21:14 24:10,12 54:15,16 56:21 58:24 61:4 78:9 84:2 113:17,19 116:11 142:19,21 153:11 156:24 168:2 177:18 184:5 189:25
certified 2:16
certify 191:9 192:8 192:14
challenge 78:10 99:18
challenged 11:9,13 11:17
chance 96:9
chances 176:12
change 89:10 91:6 193:5 194:11,12
changes 21:21 194:11
character 72:25 73:1,19 74:3,4,6,7 162:15 163:2 165:4 165:12,21 166:14 182:3 183:7
characterization 18:25 45:2 54:16 113:4 178:19
characterize 18:17 35:8 44:21 45:7 46:22 64:25 71:6 82:5 83:13 159:15 162:24 163:1,8,10

176:25
characterized 44:6 44:13 85:6,7,9 183:9,11
characterizing 65:6
characters 161:19 163:12,19 164:8 166:22 167:8 168:7 168:25
charles 4:8 7:10 194:5
chart 5:16,17,18 42:17 123:9,11 124:14 128:9,11 146:8
check 27:23 100:2 114:25 115:21 128:21
choose 154:9 187:21
chorus 172:20
christopher 4:15 6:11
circumstance 45:14
circumvent 175:5
claim 11:25 15:1 16:19 17:8,21 19:21 92:10,10 95:11,16 96:5,7,15,22 97:5,9 97:16,22 98:7,21 99:2,10
claims 11:9,13,17 15:3 18:8 19:19,20 25:15 78:10,16,19 91:4 93:25 99:18
clarified 87:14
clarify 9:3 173:23
clear 9:1 122:22 127:22 136:13,17 144:4 180:15
clearly 8:25
clips 35:22
close 79:19 81:11,25 82:18 83:18 84:20 85:2 87:10,16 92:15 92:22 93:1,2,8,12 94:4 180:7,11,17
closely 162:12
closer 165:1 166:8
closest 79:20 80:10 85:2,4,8 87:17,21 87:23 88:5,8,13,17 88:17,22 89:5,22 90:17 93:4,9,10,13 93:17,20 94:5 138:12,15 150:3,12 150:14,18,19 151:3 153:9,19 154:6
clr 1:18 192:23
clustering 37:16
cmacedo 4:9
coffee 159:9 160:2
collection 183:19
column 69:17 95:9 95:12 97:7 100:19 119:25 120:12 136:1 139:15,24 140:11,16 141:5 143:21 148:15,16 155:15,19 169:1 171:25 174:10,12 176:15 177:2 185:3 185:4 186:10,10 188:19 189:13
combination 66:22
combined 151:13
come 33:1 68:14 89:20 98:18
comes 32:3 50:13 63:5 75:5 107:7 116:19 152:25
coming 66:14 152:4 174:7
comments 8:21
commercial 101:17 101:22 102:3,6,14 102:20,22,24 103:13,15,16 104:2 104:7 187:15
commercially 187:6 187:10,13
commercials 101:14 101:18 102:11
commission 193:25
common 51:23 52:1 54:11,12 75:1 78:4
compare 27:23 32:15 36:5 40:17 74:3 75:3,18 113:2 114:17 119:18,23 134:8 142:1 147:2 149:15 150:1,10 152:10
compared 34:21 119:8 120:22 121:2 121:8 133:9,15,20 133:24 134:4 156:3 161:23 163:12,19 164:4,8,9 166:22 170:22
compares 45:19
comparing 75:7 117:2 125:17 144:15 161:7,12 164:23 178:17 179:9,13 189:7,16
comparison 42:15 42:24 47:14 73:18 75:20 107:16 113:8 113:13 117:5,7,18 117:23,25 119:3,16 119:18 120:17 121:7 124:19,23 125:3,6,8,11,22,24 131:4,8,11,12,15,21 132:7 133:3 134:5 134:19,24 135:3 145:3 146:24 150:25 151:4 157:6
comparisons 102:9 118:9 119:11,13 125:4,12,20 131:9 132:17,21,24 141:3 142:4
completely 63:11 172:11
complexity 64:25 65:6,12 66:3,5 67:3 67:14,19 68:10 70:5 70:9,9
component 140:25
components 21:2 140:3,24
comprises 97:10
computation 131:16
compute 113:21 141:15,18 156:23 157:1 168:22
computed 181:5
computer 145:11
computes 131:17 162:8 167:5
computing 117:11
concept 25:24 26:6 26:10 67:19 70:4,5 79:3 94:9,13
concepts 67:22,24
conceptually 29:13
concerned 80:3 103:8 166:16
conclude 89:4
concludes 190:20
conclusion 89:20 90:15
conditions 8:10 56:21 67:16
confident 46:3
configure 187:21
configured 81:25 153:7
confirm 14:8 123:23 126:22
connection 9:21 10:1 12:5 15:23 18:14
connects 142:12
consecutive 119:20 119:23
consequence 174:15
consider 16:1,4 52:16 53:12 61:24

63:8 77:16,20 78:3 135:17 144:11,13 153:7,14,19,24 154:4,8,10,11 156:10 157:16 163:13,20 164:10 164:19 168:20 176:6,17 181:8
considered 101:21 101:23 118:25 135:13 153:7 154:23 156:5 176:22,23
considering 56:22 154:16,17,18
considers 63:15 71:2
consistent 18:9 87:12 121:12 122:18
consisting 52:9 56:12 72:24 172:2
consists 29:25 53:15 55:16 72:18
constant 143:9
constitute 57:19 135:22
constitutes 90:4 96:4,13
constraints 99:14 166:5
construct 58:17 102:2 103:4,12 104:23
constructed 102:13 137:5,7
constructing 184:1
construction 42:9 85:21 86:17 87:4,7 87:12 88:1,11,23 91:8,14,25 92:10,11 92:18 93:6,8,12 105:15 128:3
constructions 15:1 42:17 86:24 91:12 94:23
construed 42:13 90:12 94:3 99:21
consult 82:12
cont'd 112:13
contain 59:20 179:25 183:6
contained 57:3,6 75:22 76:14
containing 53:4 56:8 58:1,22 76:4 183:5
contains 36:13 58:24 72:14 101:13 104:7 138:2 143:6 180:7,16
content 18:24 19:2,2 49:11,17 62:15 63:11 77:23 81:16 81:19 107:16,16,20 107:25 110:11
contents 52:22 53:12
contested 93:24
context 13:12,16 21:8 22:13 23:25 26:15 67:25 68:3,6 69:23,24 70:2 77:6 83:7 92:9 95:2 99:10 140:10 148:4 155:7 177:15
contextual 14:4
continue 6:9 28:4 74:5,8 75:6 144:17 157:5
continued 4:1
continues 120:2
continuing 149:24 150:8
contradiction 65:11
contrast 121:19
convenience 91:7
conversations 6:5
convey 154:22
conveyed 16:12
conwell 12:15 178:11,15,20 179:2 179:7,12,25 180:2,5 180:6,9,16,19,23 181:2 184:22 186:9 187:5
copy 12:10 23:11 112:7
corey 4:10
correct 9:23,24 10:12,13 11:6,7,10 11:11,14,15,20 12:4 12:20,24 13:7 14:15 15:8,15 16:15,16,20 17:9 18:1 25:12,13 26:7,8 27:14,15 28:12,13,16,22,24 34:23 38:12,24 39:1 39:10,14,21 40:11 41:7 42:10,11 45:6 49:3,9,14,24 51:24 52:7,17 54:1,4,5,23 55:1 56:15 59:2 60:18 62:6 65:24 70:23 72:5,14,15 73:22,23 74:14,15 74:17 75:14 76:9,12 77:24 78:5,7,12 80:14 82:4,21,22 86:7 88:5 90:12 92:24 93:25 94:12 96:1 98:14 99:18 104:25 105:1,20 106:23 110:21 113:9 115:18 116:3 116:10 118:7,11,13 119:2,6,7 120:13,14 120:18,23,24 121:10,24 127:25 129:4 131:24 132:15,19,20,22,23 133:1,21 134:20 135:7,13,14,18,19 136:7,9 137:9,22 138:4 142:9 143:13 143:19 144:12,18 145:23 146:5 148:2 148:9 149:18 151:8 153:20 154:16 155:13,25 156:11 157:9 158:12 159:3 159:12,13,22 160:24,25 161:16 165:17,18 166:9,25 171:7 172:3 173:16 178:22 179:4,14,15 182:19 183:9 184:24 185:5 186:7 186:8 188:8 189:22 191:10
corrections 191:11 194:11
correctly 39:16 58:20
correspond 39:13 43:13 55:10
correspondence 53:10
corresponding 24:17 147:25 151:7
correspondingly 159:2
counsel 6:18 7:1 15:12,19 16:13 17:3 17:6 123:6
county 191:3 192:4
couple 8:20 10:18 29:9 67:18 104:22 115:19 117:10 155:8 159:6
course 9:14
court 7:15 8:4
court's 92:10
coverage 105:6
cox 10:12,15 11:9,13 11:17 12:1,12 15:4 15:6 18:1,6,13,18 18:19 19:1,5,7,15 19:24 20:11 21:9

Google Ex. 1020

22:2,2,13,18,20 24:7,23 25:6,15,19 25:22 26:5 37:12 40:25 41:17 42:1,4 43:6 68:4,6 70:10 70:21 78:10,24 80:2 80:2,11 94:7,8,22 100:15 186:17 189:4,12
create 60:2 77:6 168:4 184:6,9,12,13 184:14 189:25
created 109:25
creating 54:2 136:11 184:18 187:22,25
criteria 136:5 173:19
crr 1:18 192:23
cryptic 141:9
cs2183243 1:25 191:25 193:2
cursory 15:21
cusp 110:23

d

d 5:1 37:14 41:9,13 41:18 50:24 161:19 161:21
damages 61:2
data 30:3,9 31:18,20 31:21,23 32:2,17 35:5,6 38:1,6,14 39:21,24 43:12 45:5 45:10 46:24 48:6 51:11 52:10,11,12 52:24 58:22 63:22 73:7 80:6 82:2 89:6 99:24 100:8 101:2 103:5 105:15,17,18 106:1 113:22,23 127:2,4 135:12 138:15,19 176:17 186:12,20 187:1,6
database 19:12 20:2 33:3,6,7,12,20 34:1 34:2,3,14,15,18 38:21,25 39:12 43:16,18,21 44:6,13 44:16,22 45:1,4,6,9 45:11,15,16,21 46:23 47:7 49:1,12 49:18,22 50:6,22 51:15,17,18,21,22 56:7,10,12 57:2,3,6 57:18,23 58:7,18 59:3,19 60:15 62:3 62:12 66:23 67:7,12 71:3,10,19 72:4,13 72:17 73:3,5,11 75:13,22,24 76:4,23 77:4,8 81:2,3,10 85:5 96:24 100:10 101:7,10,12,13,18 101:23 102:3,4,12 102:13 103:17,18 103:22 104:2,2,18 104:24 105:10,23 107:17 120:17,20 120:21 121:3,9 122:13 124:11 136:12 137:4,10,17 137:24 138:8 139:17,21 161:6 162:7 172:24 174:19 179:24 180:3,6,16,23 181:4 183:21 184:2,4,10 184:13,14 186:18 186:21 187:13,18 187:22,25 188:11 189:21,24 190:1
databases 189:1,5 189:13,18
date 6:12 23:19 69:1 69:4,7,10 109:2,20 109:22 112:11 123:12 128:12 146:9 160:18 178:10 189:5,14,15 193:3 194:8
dated 109:16
day 33:2 104:6 191:18 192:20 193:22
days 194:15
dear 194:9
decide 54:24
decides 90:11
deciding 11:1 49:13
decision 15:17 51:15 51:18 54:2,5,7 56:1 62:21 188:2,5,11
declaration 5:6 9:22 9:25 10:9 11:3 12:8 12:16 13:1,3,18 14:6 20:21 23:9,12 23:15,17 47:3 64:20 71:24 82:9,13 85:20 87:13 88:25 89:8,24 90:20,24 91:9 95:25 96:8 100:3 105:13 106:17 108:19 109:6 110:16 112:19 115:20 121:20 122:8,25 123:3,5 124:1 129:2 130:9,11 160:24 178:13
declarations 13:4,10 13:11 14:8
deconstructed 101:19
decreasing 186:1
deemed 194:16
deeper 110:23
define 65:12 67:10 74:23 90:5 142:11 168:18,24
defined 46:5 83:7 85:19 101:9 128:1 136:3 143:1,4 149:5 156:4,10 169:25 171:17,18
defines 90:25 177:3
defining 136:5 178:4
definitely 84:25
definition 22:24 23:10 24:20,21,25 43:2,4 46:11 75:16 84:13,16 88:3 97:22 100:8 107:1,9 108:7 135:21 149:6 152:14 153:22,25 154:6 176:19 177:14,16,19,20 178:2
definitions 183:20
degree 107:5,7 108:9
deletions 162:19
department 194:22
depend 158:19 181:16
depending 75:15
depends 32:18 52:24 170:12
depict 121:22
depicted 120:15 124:23 130:25 131:4 147:3
depiction 120:25
depictions 122:7
deposition 1:12 2:11 6:8,14 8:14 12:16 13:21,24 14:3,13,17 15:5 23:24 68:16 123:8 190:20 191:9 192:10,11 193:3 194:8,10
derived 15:11,18 39:14,20 40:3,6
describe 29:24 37:13 126:25 127:3 127:5 157:24 158:10 175:17
described 23:8 32:5 34:24 44:17 70:21

77:16 100:25 108:18 113:6,13 116:12 119:6 125:3 125:10 145:22 172:6 175:19 176:15 182:24 187:5
describes 36:20 41:3 41:6,8,24 70:8,9 112:22 121:6 140:25 155:12 178:16 189:17,18
describing 31:17 34:12,20 72:2 116:24 117:17 140:14,22
description 122:17 135:25 136:1 141:9 159:20 175:25 185:5
design 39:4 59:21 77:1
designed 36:23 38:11 43:24 63:23 81:15,18 83:17 152:16,19,23 153:3 153:10,12,15,17
designing 62:9 179:5
desirable 21:16 176:1
detail 175:17
detailed 13:1
determination 48:25 49:4 62:3,14 62:19,24 179:8 181:2
determine 34:22 48:1,4,5 49:23 50:7 50:10 53:11 57:6 58:7 63:8 67:6 71:5 75:7,13 88:21 102:10 121:18 132:11 164:25 180:9,19,23
determined 66:10 98:20 108:17 136:10 138:10 165:20
determines 179:12
determining 16:19 48:12 95:13,21,25 96:14,18 98:23 162:1,11 163:5
develop 20:1 103:23
developed 37:12,23 175:12,24
dictionary 181:8,9 181:12,12,14,19,23 182:1,16,23 183:13 183:14,17,19 185:22
differ 24:21 159:24 165:21 166:14
difference 87:2 102:12 106:25 107:4,5 131:19,19 131:22 132:1,11 134:11 143:1 149:21 151:7 155:9 156:23 157:2,2 162:24 163:5,9 165:13 170:22,25 171:13 180:25
differences 115:10 162:15
different 14:18 19:22 21:2 35:23,25 46:6 50:18 54:12,13 68:1,2,5 70:4 90:15 93:22 141:11 164:5 166:17,23 168:19 168:22 172:6,12 173:2,3 180:9,18,22 181:18
differently 17:17 25:6 154:3 171:11
differs 166:21
difficult 9:8 23:24
digit 133:4 147:24 148:1
digits 133:4
dimension 55:18,18 55:19
dimensional 51:9 108:4
dimensions 51:8,12 54:16 55:3,7,10,11 55:17
dip 148:18,24 149:1 149:4,7,9,10,10,17 149:20 151:5,8,10 151:15,18,23 152:1 152:4,9,20,25 154:14
dips 141:1,2 149:13 151:1,22 154:16,19
direct 106:16 119:25 187:20
directed 141:4
direction 135:20
directions 66:15
directly 182:8
disadvantage 175:6
disclose 40:25 41:17 41:21 90:24 136:14 142:6 157:10
disclosed 20:24 42:2 90:1 117:9,13 167:10 175:19 184:23
discloses 20:6 41:5 156:12 161:2,12 167:3,23 169:1
disclosing 42:4
disclosure 17:25 18:2 19:11 22:2,25 25:22 80:2,8 122:12 127:12 136:8 159:21 185:6,11
discontinue 142:3
discretion 136:11
discuss 141:23 160:23
discussed 37:5 90:23 101:3 103:9 107:23 170:3 178:12
discusses 159:11
discussing 31:15 70:6 122:16 182:17
discussion 16:22,24 17:4 23:20 35:24 50:9 78:6 97:2 108:14 110:8 134:25 151:2 188:17
discussions 15:11,18 17:6
disk 64:10,14 105:23 106:3,11,15 160:7,12 190:19
displacing 97:11
displaying 97:19
dispute 11:1 13:16
disregard 28:1
distance 84:3 113:21,24 114:7 126:1,5,10,14 127:17 128:3 131:17 143:8 168:6 168:24
distinguishes 20:9
distinguishing 20:23
distortion 21:23
dna 175:13
document 108:23,23 109:5,25 110:12
documents 14:7
doing 29:11 103:19 114:2 126:18 145:17 148:17 151:25 157:5 162:4 176:11 181:17
double 100:2 128:21
douglas 3:9 7:3 194:25

douglas.nemec 3:10
dovel 3:14 7:8
downloaded 109:12 109:13
dr 6:25 7:18,24 10:12 12:16 13:4,9 13:22 14:12,16 15:6 19:7,15,24 20:10 21:9 22:2 23:15 25:22 26:5 40:25 41:17 42:1,4 64:15 69:11 70:10 78:24 80:2,11 94:8 106:16 107:2 123:13 128:13 160:19 179:24 190:13
draft 123:4
driver 79:14
duly 7:20 192:11

e

e 3:2,2 5:1 7:19,19 112:1,1,12,12 192:1 192:1
earlier 110:4 121:13
early 39:3 75:19 117:13
ease 185:5
easier 123:15
easily 125:12
easy 79:21
ebenstein 2:13 4:3 6:16 7:11
edits 123:6
efficient 19:16 175:14 183:3 184:19 188:3,6
efficiently 184:6,8 185:20
either 9:1 23:2 28:5 35:3 59:9 126:11,15 151:5 182:5
elect 135:16
electrical 108:13
electronic 97:12,21 99:1,4 183:13,14,16 184:2
electronically 95:21 98:20,22
element 30:1,10,15 35:14 95:11,14,18 95:19,20
eliminate 61:19
eliminating 61:23
elimination 61:21
embodiment 137:18
employ 187:18
employed 155:13
employing 143:12 143:14
empty 56:23,25 57:1
enable 79:12
enabled 188:9
enabling 174:24
encl 194:24
enclosed 194:10
engine 173:6,11 188:2,11
engineering 108:13
entirely 66:10
entirety 30:25
entitled 5:13 109:1
entries 34:15 35:23 36:1 49:13 50:6 52:17 67:7,12 70:19 70:20 73:22 75:25 100:10 101:1,7,9 105:17 106:1 170:10 184:3,23 185:6,12,16,18 189:6,15
entry 5:13 32:20 33:20 34:1,13,14,17 34:18,21 36:5,11,12 36:16 44:5,12,16,22 44:25 45:4,5,9,10 45:16 46:23,25 49:12,18,22 56:8,12 56:19,21 57:3,18 58:1,13 71:2,3,4,6 71:10,12,13,18 73:6 73:14,15,18 75:13 101:17,22,23 102:4 102:13 103:18 104:3 105:18 108:25 181:3
enumerate 66:24
equal 143:8 163:14 163:21 164:11,20 166:24
equally 165:8,11 166:4,16,17 167:11 168:8
equals 101:24
equated 100:9
equation 120:9,16
equivalent 30:22 168:1
errata 193:1 194:11 194:13
error 21:22 155:24 156:3,9 157:18 169:2,9,11,15,21,23 170:13,13,19 171:12,16,16
errors 177:9
esq 3:9,11,19 4:8,10 194:5,25
essentially 54:2 140:11
establish 32:25
evaluate 55:9 148:6 148:10
evaluating 61:13 148:11
everybody 84:2
exact 12:9 21:19 22:3,7 24:1 29:3 39:9,11,13 40:10 74:17 80:6,10,17,23 81:3 82:1,18,24 83:3,11,19,22 88:4 88:7,13,17,22 89:5 89:22 90:17 93:2,9 93:13 134:14,16 138:12,18 142:12 142:13 144:14,22 145:3,16,21 147:19 158:10 179:10,14
exactly 21:11 24:24 30:13 31:8 48:18 51:13 60:10 76:21 147:22 148:25 180:1,4 181:4
examination 5:2 7:22 112:13
examine 56:20 62:4 73:14,17,21 182:10
examined 7:20 52:22
examining 75:9
example 14:11 16:11 21:17,23,25 23:4 26:16,18 28:17 30:23 34:16 37:15 37:16 41:14,19 47:5 47:18,22,23,25 48:8 48:14 49:16 50:1,13 50:15 53:23,25 56:16,23 57:9,14,23 60:14 62:19 63:10 63:15,19 65:22,25 66:18 70:24 71:1 72:10 73:4,13,17 74:16 75:3 76:21,23 79:24 81:13 97:15 100:25 101:4,21 102:8 103:14,21 115:13 118:4 121:21 123:24 124:5,12 129:7 131:20 133:2 144:3 144:4,7 146:10 147:18 153:13 154:13,23 162:21 170:19 182:16 186:6
examples 27:3,7 37:18 50:3 57:12

80:11 115:19 121:19,22 130:8
excellent 159:9
exclude 47:12 49:5 50:21 51:3 60:16,21 60:25 61:4 73:6
excluded 49:1 88:2 88:6,12
excludes 155:10
excluding 47:20 60:17 155:4,6
excuse 18:12 27:5 45:22 127:1 155:23 179:1
execution 99:22 190:4
exhaustive 21:4 25:11,14,18,24 26:6 26:10,19,25 27:2,8 27:13 29:19 35:8,10 35:13,15 37:8,11,15 37:17 40:21 41:1,4 41:22 42:2,5,9,13 44:7,14,17,22 45:7 45:12,22,23,25 46:5 46:12,16,25 47:11 56:14 58:18 59:6,12 59:13 61:8,11,15,17 61:23 62:10,18,22 63:1,3 64:22 65:8,8 65:10,20,23 66:1,7 66:11,17,20,23 67:2 70:22,25 71:7,8,16 71:19,21 73:9,12 76:18 77:13 103:25 181:9 182:6,12 183:9 186:7 187:18 187:24 188:7
exhaustively 56:9 57:5 58:6 59:17 76:1
exhibit 5:6,7,9,10,12 5:13,15,16,17,18,19 5:21 23:17 47:3 68:16,18,20,22,24 69:2,5,8,15 71:24 95:7 100:18 108:22 108:25 109:3 110:9 112:6,7,9,15 123:8 123:8,10 128:9,10 128:13 130:13 132:18 146:7,12,12 147:4 160:14,16,19 178:7,8,11 188:18
exhibits 5:4
exist 80:25 110:2
existed 110:10,20
existing 110:13
exists 28:18 64:2 71:5,13 81:3
expect 40:24
experience 26:24 44:4,11 82:14 94:20 107:7,11,13,14,21 108:2
expert 8:17 11:5 16:2,5,8 17:19,23 159:16
expertise 16:13 17:7 17:13
expires 193:25
explain 27:16 28:22 29:21 113:11
explained 32:10
explaining 189:4,13
explains 172:1
explicit 80:8
explicitly 22:4
exposure 26:9
expressed 11:5,8,12 11:16,23 95:24 105:25 108:20 112:18
extent 14:10 46:10 51:6 81:6 100:6 107:25 139:1
extract 60:5 77:4
extracted 77:17,20 77:22
extractions 139:2

f

f 112:1 192:1
fact 45:19 62:23 73:10,15 79:3 116:15 122:18 138:19 166:12,20 166:20 176:14 177:2
factor 24:16,16 25:2 25:2,4,4
fair 9:4,11,12 11:22 14:3 16:9,10 33:8 113:4 114:11 147:18 161:1 178:15,19
fairly 26:16
fall 11:23 39:16 54:17 96:20 98:12 98:16 99:8 145:8
falls 18:20 53:6
false 28:20 74:11
familiar 17:10,14 64:15 78:22 80:20 100:21 186:18,25 187:12
familiarity 17:12,14
far 57:13 91:3 163:22 166:15
fashion 13:8,22 14:23 32:5 49:11 52:6 54:9 58:18 59:12 77:2 103:25 184:3 185:21
fast 174:16 175:15
faster 79:24 188:12 188:14,15 190:3
fax 194:2
feature 51:9,10 176:1
features 20:23
fed 14:1
feel 8:6 9:2,15 33:15
fetching 96:23
fewer 119:11,13
field 18:18,19 19:1 82:14 94:14
fields 19:4
figure 49:19 121:6 122:2,16 173:4 184:25
file 21:24 31:20 58:2 159:10,11,15,16,18 159:21,24
filed 12:17 26:5 189:22
files 159:19
filings 12:17,19 13:13 14:21
finally 17:5
find 22:3,7 23:24 24:1 25:1 28:5,7 29:2,5 33:7 38:20 40:10 66:21 72:20 72:21 73:2 75:23 81:4 88:21 89:15,16 89:22 92:4 109:14 135:6,9 142:3 143:10 150:2,11,16 152:12,16,19 153:10,12,12,16,16 153:17 162:14 176:7 181:21 182:4 183:23,24 194:10
finding 21:18 27:10 36:16 38:23 80:5 89:5 90:17 91:21 149:2
finds 153:23
finished 9:9
first 10:25 11:24 19:25 24:9 28:1 73:5,18 74:3,3,4 117:7,25 118:1,2,9 118:21 120:16 121:1,2,8 122:10 123:4 125:23 126:21 131:22

Google Ex. 1020

140:24 141:13 144:5,6,15 145:4,10 145:13 147:4 172:17 182:13 185:22
firsthand 26:24
fitness 66:10
five 119:17,19,19,23 147:24 148:1 179:17
fixed 170:18
fleshed 31:4
flexibility 174:25
flom 3:4
focus 13:2 50:22
focusing 63:24 99:7
folds 73:19
follow 15:13 62:8 154:24
followed 36:2
following 57:22 58:21 141:10
follows 7:21
forced 9:17
foregoing 191:7
form 20:19 21:12 29:18 32:7 38:22 40:22 43:23 46:7,18 49:25 51:1 53:14 57:8 62:16 63:13 70:13 76:2 77:22 80:19 81:5 84:6 85:16,17 89:1 90:2 92:19 98:9 99:11 103:6 117:8 129:16 133:6,17 134:21 137:23 138:13,20 143:3 145:6 150:13 151:16,24 163:15 166:10 167:1,16 168:17 172:9,25 173:22 174:4,7 175:7 176:5,18 178:3 181:10,15 183:1 185:24 187:6 187:8 189:8
format 159:11,15,17 159:18,21,24
formats 159:10
forming 12:5 13:10 13:23
forms 37:7 177:8
formula 157:24
formulates 177:21
forth 15:3 21:14 84:4 118:3,6 132:3 141:19 192:10
forward 194:13
found 22:3 35:1,3,4 137:22 144:15 145:3,21 151:23 152:9
four 10:1,5 23:16 68:13 69:11 147:19 147:22
fraction 24:12 46:24
frame 83:12 101:10 101:21,22 102:15 102:19,24 103:10 103:16,18 104:8,9,9 104:12,23 105:3
frames 101:19 104:20
free 9:2,15 194:2
front 69:12 91:8 95:8
fully 8:7 13:1 31:3,4
function 23:1,4,4,5 24:10,14,19 25:1,3 30:3,7,13 32:12 35:19 36:9,18 40:2 49:6 51:19 126:12 126:16,16 150:15 155:12,23 156:18 156:20 157:19 158:7,8 187:20
functions 39:4 179:6
fundamental 31:7
further 53:12 73:21 74:13 87:14 97:4 141:2,21,23 190:14 192:14
future 139:2,5,24

g

g 7:19,19 112:12,12
gapless 162:16
general 8:20 15:25 18:19,20 19:9 22:16 29:22 31:13 47:24 58:16 79:7,17 87:2 184:17
generally 9:15 13:8 20:8 38:18 39:7 40:9 113:11 181:7 187:17
generate 48:19 60:10
generic 37:25 52:8
geologic 136:3
george 1:12 2:11 5:3 5:6 6:25 23:15,18 191:6,15 192:9 193:4,20 194:8
getting 151:20 164:1
ghias 12:14 43:7 160:15,22 161:1,12 161:17 162:1,3,11 162:23 163:1,4,8,13 163:20 164:10,19 164:22,25 165:10 165:15,19,21 166:13,15,21 167:3 167:10,12,23 168:10 169:1,24 170:5,9,12 171:25 172:20 173:4,7 174:3,11 175:19 176:1,15,20 177:3 177:20
gigabytes 104:14,16 105:7
gish 3:11 7:5
give 12:9,9 23:9,11 37:17 66:18 113:24 145:15
given 8:14 32:2 46:4 48:16 51:23 52:21 53:12 55:21 62:5,13 74:2,2,9 77:10 81:8 93:19 115:16 119:10,10 138:15 139:3 141:10,13 148:21 163:13,20 166:5 192:12
gives 30:8
giving 97:14
go 6:9 20:16 23:13 32:13 36:10,12,15 40:5 45:20 47:20 48:11 51:16 56:2 68:12 73:1 96:9 97:4 100:23 106:7 128:8 145:12 155:2 156:22 162:11,17 163:4 165:22 181:16,21,22 182:2 182:3,8,22 183:4,5 183:23,24
goal 28:13 44:15 45:17 54:21 66:8 67:14 75:23 79:18 102:23 103:2,4 104:6 176:7
goals 19:16
goes 51:19 138:22 152:13 156:24 162:1,4
going 8:24 20:25 56:20 59:25 63:7,18 64:9 66:3 67:3 78:8 81:4 103:20 106:10 111:3 114:16 149:7 152:11 156:25 160:6 170:16 174:1 179:20 190:8,18
good 6:1 7:24,25 18:4 54:8 72:6 111:1 164:21 165:8 165:11 166:4

167:11 168:8
google 1:3 6:19 7:4 10:3 12:13,17 193:3 194:6
google's 13:5
governing 17:7,20
grab 110:25
graduate 108:8
granularity 102:17
greater 23:3
green 131:1
grid 91:11 124:17 124:24 125:5 130:24 131:1,3
grows 174:20
guarantee 63:25 86:5,13 89:14,21 93:17
guaranteed 83:10 85:23 86:1,8 87:9 87:16,20 88:21
guarantees 28:21 88:4,7,12 89:5,16 90:16 91:2
guess 140:17
guidelines 158:4
guitar 172:22
guys 52:3
guzman 1:18 2:16 7:16 192:6,23

h

half 28:1 127:10
hand 192:20
hanlon 4:15 6:11
happen 157:14
happens 35:2 157:11
hard 46:8
hash 27:2 29:17,18 29:21,24 30:11,15 31:25 32:4,11 34:19 35:16,18,22 36:8,9 36:18,21,24 37:5,19 37:23,24,25 38:2,16 38:20 39:4,8,25 40:2,6,8,11,12 47:22,25 48:14,19 49:16 50:1,12,19 51:7 53:1,2,4,6,8,8 53:19 56:17 57:1 60:8 63:20 77:10 178:22 179:4,6,9,13 180:1,4,9,13,19,21 181:5 184:24 185:12,16,18
hashes 178:18 180:22 181:1
hashing 178:17
head 12:11 21:1 24:3 50:5 59:9,25 126:4 157:21
health 8:10
hear 9:2 65:3 158:1
heard 10:21 44:5,12
held 2:12 6:15,21
hereinbefore 192:10
hereunto 192:20
high 29:1 51:9 116:17 170:14
higher 23:2 108:17
highest 114:5,6 141:11,12,12,14,14 141:16,17 142:1,2 142:22,24
highlighted 147:9 147:14
highly 108:4
hit 157:7
hold 104:19
horowitz 4:10
huh 69:13 91:18 109:19 110:17 112:20 116:7 132:5 134:2,7 148:20 159:23 173:10
hum 161:2
hummed 161:8 166:8 176:8
humming 161:4 176:9
hypothesize 21:18
hypothetical 72:2 76:4 116:21,23 118:18 128:18 130:2 149:24 150:8
hypothetically 114:9 146:15,22

i

i.d. 5:5
idea 77:10 79:17
ideas 51:6 63:20
identical 32:17 40:16,19 81:9
identification 19:14 23:19 59:5 69:1,4,7 69:10 88:4,7 98:16 99:1,4 109:2 112:11 123:11 128:11,22 128:25 145:7 146:8 160:18 178:10
identified 91:24 92:9,17 93:1,5,8
identifier 31:23 92:2 185:7
identifies 88:16,16 136:15 139:3 167:4 173:12 178:16
identify 19:11 20:1 21:10 43:25 44:25 45:18 66:9 79:19 80:13 81:2,25 82:1 83:18,19,25 84:2 87:10 102:18,20,24 103:15 104:7,8,10 127:16 138:5,6 143:11,15 153:4 158:10 161:3
identifying 20:9,10 73:15 78:18 88:11 88:14,19 89:13,17 91:16,20 94:4 143:5 154:12 161:5 175:9
176:12
ids 66:22
ifs 130:4
ignore 55:1 80:17
ignored 136:17
ignoring 80:23
illustrates 63:19
illustrative 115:25
impact 108:19
implement 29:10 34:25 55:14 185:19
implementation 80:22
implemented 40:13 40:23 41:14
implementing 41:18 41:22,25 187:4
implication 93:15
implications 93:14
imply 23:6 122:3
important 136:15 136:20,24 138:1
include 49:5 98:1 154:19 155:23
included 125:13 170:10 194:11
incorporated 6:20
incorrect 44:20 45:1 46:22 47:1 130:8,9 130:11
increase 24:11,13,17 25:1,3 176:12
increases 23:2
increasing 27:20 186:1
independent 16:11 16:17 17:5 86:23 90:6
index 30:8 60:4 77:9 77:17,20,22 78:3 175:14 184:14,18
indexes 184:21 187:25
indexing 77:25

indicate 65:19 92:6 92:14 136:2 144:20 145:2
indicated 14:20 99:20
indicates 144:10,14 144:21
indicating 194:11
indicative 52:15,21 53:6
indices 184:7 187:22 188:4,6,9,13,14 189:25 190:3
individual 34:13 101:19 102:4 105:18
influence 13:18
information 12:6,7 13:17 16:12 18:21 48:12 49:22 54:3,10 54:11 71:18 97:11 97:20 98:1,4 107:17 107:20 108:2 183:22
initial 117:11
inquiry 184:16
insertions 162:19
instance 10:25 53:1 118:24 147:4,8 155:3 164:22
instances 14:11 26:23
instituted 10:2
integer 24:18 36:19
intelligence 54:6
intelligent 47:12,19 50:20 61:13 190:2
intelligently 54:3
intent 81:1
inter 10:1 15:10 16:14
interest 65:15
interested 80:5 192:17
interfere 6:7 8:11
interpret 70:14 91:19
interpretation 100:5 105:22,24 106:2,4,6
interpretations 105:14
interpreted 12:1
invented 20:11 25:23 26:2 78:25 94:9
invention 18:18,20 19:10,16 20:23 80:12 187:4 189:6 189:14,15
inventions 19:8 21:9 106:21
inventor 10:11
inverted 77:9
involve 127:6,6
involves 12:11
ipad 23:22
ipr 1:4,4,5,5 10:6,9 12:14,17 13:13 14:21 15:2,17 23:16
irrelevant 166:3
issue 10:6 18:5 82:7 106:22
issued 99:18
issues 8:11 16:8
item 12:18 60:13 126:11 173:9,15
items 31:10 189:7 189:16
iwamura 12:15 43:7 112:8,21 113:7,14 114:12,20,24 116:6 117:9 119:6,18 120:1,12 122:7,12 123:25 124:6 125:4 125:11,19,21,25 126:12,21,22 127:12,23 128:19 129:8,15,25 131:8 131:16 132:11 133:3 134:1,19,23 135:3,5,8,12,15,21 136:4,19,23 137:1,5 137:16,18 138:5,22 139:9 140:3,17,21 141:25 143:25 144:11 146:2,17,25 148:4,14,16 149:5 150:24 154:14 155:12 156:10,12 157:10,17 158:15 159:11,21

j

jennifer 1:18 2:15 7:16 8:22 192:6,23
jennifer's 9:7
jersey 194:2
job 1:25 191:25
journal 79:10
jumping 110:23
jurat 194:15

k

k 7:19 37:14 41:9,13 41:18 50:24 66:22 66:22 67:1 83:23 112:12 175:11 176:24 177:24 182:9,11
karypis 1:12 2:12 5:3,5,6 6:25 7:18,24 23:14,15,17,18 64:15 68:15,18,20 68:22,24 69:2,5,8 69:11 106:16 108:25 112:7,9 123:8,10,13 128:10 128:13 146:7 160:16,20 178:8 179:24 190:13 191:6,15 192:9 193:4,20 194:8
keep 55:15 114:2 137:25
key 20:23 30:7,13 30:17,19,21,21,22 30:25 31:9,16,19,20 31:21,22,25 32:11 32:16,16,18 35:14 36:5,10,10,19,20 38:1,4,16,21,24,25 39:7,12,12,14,19,19 39:20,24 40:1,2,3,6 40:14,18 48:17,19 48:21 51:13,20 52:1 52:3,5,9,10,14,19 53:2,4,8,11,16,17 55:4,7,10,16,24 58:12,14 92:2 136:16 186:2
keys 30:4 32:2 39:22 40:16,19,20 53:7 54:15 55:11 58:9 174:17 176:2 185:25
kilobyte 105:3
kind 15:21 20:1,21 36:24 85:11 140:23
knots 164:1
know 13:13,25 14:1 14:6,8,9,25 15:2,21 18:22 19:3,10 20:22 21:13,14,18,18,19 21:21 22:25 23:1,1 24:9,9 27:22 29:13 29:14,16 30:5,16 31:14,16,18 32:10 33:10,21 35:5,18 39:3 40:20 41:11 43:25 44:2 46:11 48:5,20 49:7,8,17 50:25 51:6,8 52:3 52:10 54:15 56:3,17 56:19 57:15 60:2,3 60:4,5,6,7,25 61:22 62:2,11,20,23 63:20 66:25 67:4,15,17 71:12 72:21,21 73:11 75:5,6 78:1

79:20 80:7,9,24 81:10,12 83:1,12,24 83:25 84:1,25 85:19 85:21,23 86:8,9,10 86:24 87:8,11,12,20 88:18 89:14 90:7,9 92:21,23 93:16 94:17 95:4 96:19,23 99:14 100:21 103:9 103:20,24 104:11 104:12 107:16,17 107:23,25 109:20 109:24 110:3,5,7,12 110:19 113:21,25 115:8 116:16 117:14 122:3,11 123:5 135:8 137:2 138:25 139:4 140:13 141:15 142:6 143:6,7,9,15 143:16 145:13 148:12 150:23 152:20,24 154:10 156:24,25 159:18 159:25 160:1 162:9 163:22 164:19 165:7 166:1,3 173:18 174:5,7 175:1 176:11 182:1 182:11 183:4
knowledge 16:18 83:5 108:3
known 26:6 38:18 40:9 79:4 94:13 113:2 161:9 175:21 178:18 189:5,14

l

label 122:9
language 11:25 33:24 95:12 123:6
large 14:10 46:10 51:6 81:6 100:6 107:24
larger 39:21 170:15 174:20
law 8:4 15:9,16 16:2 17:10,14
lead 28:22 53:3 67:17 137:12 190:3
legal 16:8,18 17:7 17:16,20,23 193:1 194:1
length 30:6 32:13 36:21 76:5,7 77:5 82:8 113:18,19 171:21
letter 73:5,25 75:12 75:22,24 185:23 194:16
letters 72:14 75:1
level 54:6 107:1 108:16
leverage 186:3
light 91:4,5
likewise 9:6 61:12 88:9
limit 79:25 127:13 127:24 155:12,22 156:4,9,18,19 157:18,19 158:7,8
limitation 127:17
limited 17:15 169:17
line 39:16 69:19 95:17 97:6 100:20 104:13 117:23 120:6,16,18,19,23 124:17 131:21 140:17,18 142:23 144:21 145:8 155:17 159:8 169:8 188:21 193:5 194:11
linear 22:23 23:3 64:16,22,24 65:5,9 65:13,16,23 66:1,5 66:6,12,17,19 67:4 67:6,11,13,16,19,20 68:7,8 69:20,22 70:1,5,7,8,16,22 99:23
linearity 22:25
lines 104:22 136:2 139:15 140:23 141:6 143:23 148:15 155:15,19 169:2 174:10,12 177:3 185:3 186:9 188:20,23
link 30:1,9,14,17 32:14,15,20 34:16 34:17,21,25 35:17 35:20,21 36:6,11,13 36:14,17 40:13,18 48:20,23,24 57:1 63:25
linked 34:12,13
list 12:9 30:1,9,14 30:17 32:14,15,20 34:12,13,16,17,21 34:25 35:17,21 36:6 36:13,14,17 40:13 40:18 48:20,23,24 63:25 92:6,14,15 105:14 165:16 166:6 170:5,8,10,12 170:14,20 172:2 173:15 174:1,5,7,9 182:4
listed 12:8 145:25
lists 35:20 36:11 57:1
litigation 8:18
little 17:16 39:20 71:3 97:4 123:15 154:2
livingston 194:2
llp 2:13 3:4 4:3
load 48:15
local 128:2,2 149:7 149:8
locate 38:16 48:17 174:25 180:2,5
located 6:16
locates 42:14,24
location 118:22 144:9
locations 146:5
logic 79:21
logical 110:24
long 123:17
longer 146:3 171:23
look 24:13 36:6 40:19 47:2 49:8 55:21 62:15,24 64:2 68:11 69:14 71:9,17 73:3,5 74:21 95:7,9 95:15 97:5 100:17 100:20 113:3 116:6 116:8 120:4 121:21 123:18,22 124:14 126:6,7,21 129:1,20 130:24 134:5 140:16 141:1 143:22 146:10 148:14 155:15 168:20 169:2 173:4 174:10 178:20 179:2 181:19 182:22 183:14 185:22 188:18,19 189:6,15
looked 14:25 45:3,4 63:21 73:10 96:6
looking 40:16 42:16 42:20 45:14 49:8 53:10 54:19,25 57:13 62:20,25 71:11 72:3,16 74:13 74:16,24 91:14 95:16 130:12 141:1 171:4 175:2 181:13
looks 44:5,12,21 45:8,10 63:4 71:3
lookup 40:9,15 182:18 187:18
low 116:17 159:5 170:15

lower 108:17 142:15
lowest 114:6,8 149:21
lunch 110:25 111:4
luner 3:14,19 7:7,7 7:9 20:13,19 21:12 32:7 38:22 43:23 44:8 46:7,18 49:25 53:14 57:8 62:16 63:13 64:4 70:13 76:2 80:19 81:5 82:6 85:17 89:1 90:2 92:19 98:9 99:11 103:6 117:8 129:16 133:6,17 134:21 137:23 138:13,20 143:3 145:6 150:13 151:16,24 159:4 163:15 166:10 167:1,16 168:17 172:9,25 175:7 176:5,18 178:3,23 179:18 181:10,15 183:1 185:24 187:8 189:8 190:15

m

macedo 4:8 7:10,10 17:2 194:5
madam 194:9
maintenance 186:11
making 9:7 62:20
man 10:12
manner 8:3 45:24
map 30:4 178:22 179:4
maps 30:16 32:12 36:20 180:18
march 9:17
mark 23:14,14 68:13,15,18 108:22 112:5 123:7 128:8 146:11 160:13 178:6
marked 23:18 68:25 69:3,6,9 109:1 110:9 112:10 123:11 128:11 130:17 133:8 146:8 160:17 178:9
market 187:14
marking 139:17
marriage 192:16
match 22:3,7 29:3,3 29:6,16 34:22 35:1 35:3,4 36:6 38:20 38:24 39:11,13 40:11,16,19 42:14 42:24 43:10,11,15 43:22 61:6 66:25 71:5,13 72:4,16 73:4,20 74:5,8,10 74:17 75:8,14 77:21 79:20 80:6,10,10 82:1,18,19,24 83:19 85:4 88:5,8,13,18 88:22 89:6,15,16,22 90:17 91:24 92:7,9 92:15,22 93:2,10,11 93:13,20 94:4,5 102:11 113:3,4 114:4 132:12 134:14,17 137:9,13 137:22 138:6,6,11 138:12,14,18 139:4 139:7 142:3,5,11,11 142:12,13,13,15,18 142:20 143:2,4,11 144:15 145:3,16,18 145:21 146:4 147:6 147:9,12,15,22,25 148:3,5,11 150:15 151:12,13,14,17,23 152:9 153:5,8,9,12 153:14,15,17,18,19 154:6 156:5,11 157:9,16 163:22 165:1 166:24 171:19,22,24 176:11,22,23,23 178:21 179:3,9,13 180:2,5,7,10,11,17 180:20,24 181:1,6 183:24 186:2
matched 43:20 174:17 176:2 180:8
matches 38:16 42:15,25 43:2,3 44:1 54:22 63:6 80:17,24 81:25 83:18 138:7 145:16 147:19 148:8,12 153:11,24 154:12 158:10,11 162:2,12 163:14,21 164:11 164:21 165:8,11,17 165:24 166:8 167:11 169:17,18 169:22 171:5,5 173:12,14,18 175:16 179:10,14
matching 21:11 39:8,9 73:2 75:5 91:23 93:4 104:9 135:6,9 148:7,12,22 150:3,12,23 151:4 152:12 164:15,17 167:5 169:14 170:6 170:24 171:18 173:15 174:16 176:21 177:23,23 178:5
material 15:22 106:25 107:4
materials 14:24
matter 107:19 192:18
matters 11:1 13:16
maximum 128:2 149:8 169:12,17 170:1
meagher 3:4
mean 18:2 22:10,16 22:22 23:6 26:12 28:15 32:23 33:9 38:7 41:3,11 43:2 43:15 53:5,15,17 59:10 60:20 62:18 62:22 63:17 69:23 79:14 81:6 82:15 83:9,14 84:19 86:11 93:16 94:3 99:14,21 115:12,13 136:13 142:14,20 144:1,13 148:25 152:22 154:7 166:23 169:10,16,19,21 173:23 176:16 181:11 183:16
meaning 70:1 86:16 105:3
means 41:5 170:25 177:4
meant 177:7
measures 68:9 116:17 167:24
mechanism 61:18
media 159:18
medical 8:10
meets 97:21
megabytes 45:16
melodies 112:23 135:18 138:23 140:6 161:10 162:9 164:20 166:7,25 170:6 173:15,18
melody 43:9 113:18 113:20 135:6,10 137:11 138:1 143:6 145:18 146:3,23 148:4,7,8,23 149:21 161:5,9 166:8 169:14 174:18 176:3,7,9,10,13 180:13
memorize 123:17
mentioned 25:10 27:12 29:17 67:18 83:12

Google Ex. 1020

merit 103:17,19 104:3,5
merits 103:7
method 20:9 50:5,24 50:25 51:3,5 52:25 56:18 59:13,14 62:18 76:17 88:16 116:14 117:13,17 127:15 167:2,23
methods 20:7 37:10 37:13 55:3,14 59:10 60:24 107:23 117:10 175:12,14 175:23
metric 165:5
microphone 176:9
microphones 6:3,7
middle 27:23,25 29:14
million 101:5,5,7,11 101:25 103:10 104:17,24
mind 50:13 98:18
mini 159:11,15,16 159:25
minimum 127:23 128:2,3,5 149:7
minute 169:7
minutes 9:16 155:8 179:17
mismatch 163:3 165:4 168:2
mismatched 163:12 163:19 164:4,8 167:7 168:7
mismatches 164:13 164:15,17 165:6,25 167:25 168:2 169:13 170:1 175:11 176:24 177:18,24
missing 61:6
misstated 137:15
mistake 122:6
mistakenly 122:18
mk 29:12
modification 29:7
modified 29:5
modify 184:16
moment 13:2 22:8 53:20 68:11 70:6 106:8 120:3 143:22 170:3
moments 25:11 119:5
monica 3:16,17
morning 6:1 7:24,25
moulin 13:3,4,17,22 14:12
moulin's 12:16 13:9 14:16 107:2
move 10:19 73:24 74:11 78:8 93:22 99:16 145:22
mp3 159:19
mt 194:1
multi 71:3
multidimensional 51:10
multiple 32:22 34:15 57:19 146:5
music 57:3 126:12 126:16 135:23 136:6,21 137:8 167:13,14 168:12 168:14
musical 136:25

n

n 3:2 5:1 24:11 67:7 67:12 70:19 101:8 101:24 102:4 112:1 112:1,1
naive 119:21
name 6:11,24 193:3 193:4 194:6
named 10:12 122:5
narrow 60:22 73:7
narrows 35:19
nature 49:3 51:17
near 21:5 29:3,5,16 38:16,20,24 78:13 83:14 84:10 113:3 178:21 179:3
nearest 83:1,23 84:5 84:9 85:14,15,24,25 86:2,5,7,12,19 87:24 93:23 94:10 94:15,18,25 95:3
necessarily 21:19 48:4 66:19 79:19 81:10 84:21 87:10 87:17 93:1,9,13 94:5 135:6 153:1 154:18 166:23 172:21 184:12
need 9:17 48:2,6,12 54:24 75:11 96:7 104:8 114:25 128:20 143:15
needing 90:8
negative 115:3,6,17 116:18
negatives 116:16
neighbor 21:5 78:9 78:13,15,18,21,25 79:7,15,18 82:5,19 82:25 83:1,14,23 84:1,5,6,9,10,11,12 84:14,16,19,23 85:1 85:6,9,14,15,16,18 85:22,24,25 86:2,3 86:5,7,12,19,20,21 86:22 87:3,7,11,18 87:23,24 88:3,11,14 88:19,23 89:6,12,13 89:17,18,23,25 90:5 90:5,18,24 91:1,5 91:15 92:17 93:6,18 93:21,23 94:10,15 94:19,25 95:4
nemec 3:9 5:3 7:3,3 7:23 20:15 23:13,23 27:5 37:21 44:10 64:6 68:15 80:4 106:7 108:22 110:22 112:5,14 123:7 128:8 133:13 133:22 150:6 159:6 160:2,13 163:7,16 164:6,24 166:19 171:10 172:18 178:6,25 179:16 182:15 190:6,13 194:25
network 1:6 6:20 7:8,12 9:23 11:4 190:15 193:3 194:6
never 35:21 38:17 46:5 60:11 63:21
new 1:13,13 2:14,14 2:14,18 3:7,7 4:6,6 6:16,17 25:11 42:2 42:5 172:1,5,11,13 192:2,4,7 194:2
nine 75:22 104:14 104:16
non 21:4 25:11,14 25:18,24 26:6,10,19 26:25 27:2,8,13 29:19 35:8,10,13,15 37:8,11,15,17 40:21 41:1,4,22 42:2,5,9 42:13 44:7,14,22 45:7,22,25 46:5,16 46:25 47:11 56:9,14 57:5 58:6,18 59:6 59:12,13,17 61:8,11 61:15,17 62:10,18 62:22 63:3 65:8,10 73:9 76:1,18 77:13 103:25 171:2 181:9 182:6,12 183:9 186:7 188:7
nonlinear 21:3 22:9 22:12 23:3,5 24:6,8 24:19

nonlinearity 22:14 22:16,21
nonsubset 59:4
normal 8:24 33:19 33:19
nos 1:8
notarized 194:12
notary 2:17 191:21 192:7 193:21,24
notation 5:14 108:24 109:1 110:9
note 6:2 113:7,7,12 113:12 114:12,12 117:4,4,18,18,23,23 118:8,8 119:1,9,9 119:13,14,16,16,17 119:17,17,19,22,22 124:25 125:3,3,6,6 125:8,8,10,10,15,19 129:18 133:10,16 133:20,25 134:4 157:1,1 158:14 161:22,22,22,23 168:21 194:11
noted 112:2 190:21
notes 15:21 119:20 119:24 126:2,5,11 126:14 127:13,24 128:7 133:8 134:9 135:10 149:15 161:13,14 168:19
notice 2:15
noticing 64:7
notion 24:25 84:19 150:14 152:14
notwithstanding 166:12
november 1:13 2:8 6:13 192:21
number 27:21,24,24 27:25 28:18,19 61:4 64:10,14 65:15,16 65:18 66:4 100:9,12 100:13 101:9 102:9 105:16 106:1,11,15 110:14 124:19 125:23,24 127:13 127:24 131:4,11,12 132:18 134:6 141:3 158:15,20,23 160:8 160:12 162:15 163:12,18 164:4,7 164:13,14,17 165:5 165:25 166:22 167:6,7 168:7 169:12,17,18 170:1 170:9,18 190:19 194:7,11
numbers 6:23 27:19 30:23 115:17 123:17,20,24 124:10,10,23 129:21 132:8 146:20 155:18 188:22

o

o 5:14 7:19 108:24 109:1 110:8 112:1,1 112:1,12
oath 8:2
object 62:25 84:8 143:3
objection 20:19 21:12 32:7 38:22 43:23 46:7,18 49:25 53:14 57:8 62:16 63:13 70:13 76:2 80:19 81:5 82:6 85:17 89:1 90:2 92:19 98:9 99:11 103:6 117:8 129:16 133:6,17 134:21 137:23 138:13,20 145:6 150:13 151:16,24 163:15 166:10 167:1,16 168:17 172:9,25 175:7 176:5,18 178:3 181:10,15 183:1 185:24 187:8 189:8
objective 136:5
obtain 40:1
obtained 144:22 151:17
obvious 11:18 17:8 17:12,21
ocampo 1:18 2:16 7:16 192:6,23
occasionally 10:14
occur 151:14 158:11
occurs 181:22
offer 16:8
office 1:1 16:5 86:17 86:23 87:4,6 88:20 89:3,20 90:6,11,14 90:25
office's 87:25 88:10
offices 2:12
oh 55:9 141:7 186:22
okay 11:3 13:2 14:11 17:18 18:12 31:12 32:3 34:5,8 34:10 36:15 40:4,15 47:4 53:21,24 54:19 65:22 66:20 67:14 68:2 70:3 75:2 84:5 85:3 87:21 96:9,11 98:18 100:23 103:1 105:5 107:9 115:15 116:24,25 117:2,3 117:16 120:10 124:4 126:19 127:10 130:24 132:15 140:19 141:7 155:1 160:4 162:21 163:25 168:11 171:3 184:17 186:22 188:24 189:9 190:17
omit 137:5
once 14:2 107:6 143:9
ones 55:1 130:22 133:12
online 110:6
open 187:15
operate 156:18
operates 122:7 137:16
operation 123:25
opinion 11:8,12,16 90:4,7,9 91:3 93:11 105:25 106:20 139:6
opinions 11:5,22,23 11:24 12:6 13:10,18 13:23,25 14:18 16:8 88:1,24 89:8,24 90:19,21,22 108:20 112:18
opposed 21:10 38:8 102:14 151:15 171:6 183:18
opposite 66:15
optic 175:11
optimal 84:21,24
optimizations 79:22
options 155:3
orange 147:9
order 27:20 49:23 50:7,10 56:2 60:22 61:5 66:4 117:24 139:8 168:4 182:18 184:2,13,16 186:1
ordered 184:9
ordinary 44:19 106:21
organize 103:22 104:1
organized 54:3,9,14 54:25
organizing 103:17
original 30:4 40:2 40:14,20,20 53:2 172:7

outcome 192:17
outlined 48:16
output 25:3 165:16 169:16 173:14,19 173:22 174:5,6
outputs 170:5
outside 95:2 99:9,14
overall 152:5 157:13 164:21 171:6
owner 1:7 3:15 4:4 7:8,12 190:15

p

p 3:2,2 7:19 10:20 10:23 112:12
p.m. 4:11 111:4 112:2 190:21
page 5:2,16,17,18 42:19 108:23 122:2 123:9,10,25 128:10 128:15 129:2 146:7 181:21,22 191:12 193:5 194:11
pages 12:23,25 182:2,8,10 183:4,5 191:8
pairs 31:19 38:2
pairwise 131:17,19
paper 46:9 181:8,12 182:16 183:18 185:22
papers 46:4 81:21
paragraph 47:2 60:15 71:24,25 72:1 91:9,11 100:3 105:13 106:17,19 107:10 115:22 121:21,25 122:15 124:12 129:1 139:16 141:22 145:8
parameter 24:10 65:14
pardon 58:11 64:18 94:18 132:16
149:16 157:19 172:16
park 2:13 4:5 6:16
part 12:14 28:3 36:3 43:9 49:2 61:20,22 65:3 66:21 120:20 120:21,22 128:20 138:3,24 140:1 155:11 156:14 177:1
partes 10:1 15:10 16:15
particular 19:13 20:3 38:4,6 57:7 58:7,15 59:14,20 62:4 69:17 74:18 75:3 93:12 95:9 96:7 100:19 101:20 110:11,12 112:25 113:15 120:1,25 130:10 140:1 145:19 148:15 150:21 153:8 156:4 157:14 158:18 161:17
parties 6:9 192:15
partition 54:7 56:18
partitioning 50:25 51:2,5 53:23,25 55:2,14 60:9
parts 140:8
passage 100:22,23 101:8 144:19 188:25
passing 136:1
patent 1:1,1,7,8 3:15 4:4 5:7,9,10,12,15 5:19,21 6:22 7:7,12 10:23 12:15,15,15 15:16 16:2,5,19 17:8,10,21 26:5 42:7 68:16,19,21,23 68:25 69:3,6,9,15 70:11,22 86:17,23 87:4,6,25 88:10,20 89:3,20 90:6,10,14 90:25 95:8,17 96:15 97:5 98:8 100:18 112:10,22 120:1 160:14,17 178:7,9 187:5 188:19 189:4 189:22 190:15
patents 10:5,9,12,15 10:15 11:9,13,17 12:1,12,13 15:4 18:1,7,8,14 19:1,6 19:25 22:2,14,18,21 24:7,23 25:7,16,20 25:23 31:15 37:13 41:2 43:6,7 68:4,7 68:12,13 69:12 78:10,25 80:2,13 83:8 94:8,22 95:3 100:15 107:24 186:17 189:12
path 55:4,7 56:4 157:1
patterns 140:6
peak 119:5,9,12 120:12 121:1,2,8,9 121:13 125:15,18 126:1,5,10,14 127:14 128:1,21,22 128:24 129:4,9,18 129:25 130:7 131:7 132:16,16,17 133:25 134:18,18 134:23,23 135:2,2 135:10,11 141:11 141:12,12,14,15,16 141:17 142:1,2,15 144:1,5,6,8,15,24 145:4,10,13,20 146:1,16,24 148:17 149:8 150:2,11,17 150:19 151:5,8,10 151:15,18 152:1,11 152:16,18 153:2,4 153:16,22,23 154:13,15,20 155:22,24 156:2,8 157:3,5,6 158:14,19
peaked 143:7
peaks 117:12 121:17 127:2,2,14,16,18,24 130:13,19 138:10 139:17 141:1,2,11 143:16 144:11,16 144:17 145:14 149:13 150:25 151:22 158:16,20 158:23,24 159:1
penalty 194:13
people 23:25 33:18 38:18 79:4 83:5 85:10 94:14 182:20
percent 48:4
perform 28:2,10,11 28:14 58:23 73:18 75:25 96:19 131:16 132:7 172:1,13 184:5 189:24
performed 97:18 134:24 135:4
performs 138:16 162:4 173:6
perjury 194:13
person 33:16 44:19 46:21 82:3 106:20 107:20 136:11 175:4 177:13 186:17,24 187:3
personally 26:20 36:23
petition 16:15
petitioner 1:4 3:5 6:18 7:4,6 190:14
petitions 12:21 13:5 14:22 15:2,10
phase 50:7
photocopy 5:7,9,10 5:12,15,19,21 68:24 69:2,5,8 112:9 160:16 178:8

pick 6:4 55:7,24 145:19
piece 167:13,14 168:12,14
pieces 117:6 126:20
pierre 13:4
pitch 113:1,2 115:9 127:4 161:8,9
pitches 114:14
pk 55:4,5
place 6:6 24:1 39:8 125:5,20,22,25 127:12 131:10 132:19,22,25 134:20 139:10 145:5
placed 122:18
places 14:5
pleasant 194:1
please 6:2,5 7:1 9:18 148:15 160:15 174:11 188:20 194:10,10
plus 29:16
point 14:7,13 18:4 18:22 23:21 27:23 28:1,6 31:8 35:2 41:10 62:23 68:9,13 74:9 75:9 110:24 113:23 117:21 134:10 142:23,24 176:10 188:1
points 10:19 113:22 121:18
populate 30:11
populated 32:4
portion 20:18 122:24 134:15 135:22 136:6,20,24 139:18 141:5 152:4 154:3,5,8,22 155:5 156:21 172:6 174:18 176:3
portions 135:17 136:15,16 137:6,8 137:11,19,21 138:1 138:24 139:10,20 139:25 147:22
posed 24:5 34:11
posing 66:16
position 118:1,2,14 156:23
positions 131:18
positive 116:16
possibility 60:1 149:23 157:15 172:10 187:9
possible 42:15,25 43:1,3,10,11,15 46:14,19 56:9,13 57:5 58:6,16 65:9 105:14,21 146:4,6 148:21 149:19 151:22 158:6 169:15,20 170:5
possibly 108:13
potential 66:25 67:1 174:6
potentially 43:16 48:5 75:4 103:23 162:18 170:15 185:19
practice 40:21 155:21
precedes 18:8
preceding 161:23
precise 22:24 177:22,25
precisely 23:8
predates 159:18
prefer 84:15 124:2
preliminarily 94:3
prepared 129:6
preprocesses 138:23
preprocessing 139:11
presence 38:3 75:23
present 4:14 58:8 82:2 83:19 115:2,7 138:19 185:6
presentation 46:9 168:23,25
presentations 46:4
presented 13:17 62:5,13
presume 29:10 41:4 53:16 96:23 174:24
previous 29:14,15 30:23
primarily 175:12
primary 79:13
printed 109:21
printout 109:8
printouts 110:15
prior 11:14,18 12:3 15:4 16:21,23,25 17:2,9 18:6 26:12 26:13 32:10 37:12 39:2 61:22 90:1,23 91:4,6 94:17 112:17 160:22 175:24 178:11 182:10
private 6:4
probabilistic 61:1
probability 29:1
probably 77:7 79:13 102:21 108:12 110:24 130:15 184:6 186:4
problem 19:6,9 20:7 59:4,5 76:11,13
problems 19:6 108:4
procedures 16:5
proceed 7:18 32:6
proceeding 15:23
proceedings 9:14 10:2,6 15:17 23:16 42:10
proceeds 32:9
process 17:24 19:17 19:21 32:6 34:19 35:7,9 40:15 47:14 53:9 60:17,18 61:6 61:13 74:10 77:15 79:22 80:18 88:15 113:8,13 114:2 118:9 119:16,18 120:13 122:17 125:3,6,8,10 131:8 132:17 134:19 138:25 140:12 145:12,17 146:25 149:9 157:8 158:14 166:16 173:7 182:19
processing 61:22 139:22 188:1,10
production 194:22
proof 16:14
proper 124:5
property 52:2,5
proportion 24:15
proportional 158:15 158:22
proposal 104:22
proposed 42:8 155:3
prototypical 29:23
prove 174:20,23
provide 61:1 136:4 144:4 158:4
provided 21:17 72:11 135:21
providing 13:15 97:10
public 2:17 191:21 192:7 193:24
purport 17:19 25:23 26:1 78:24 94:9
purports 20:11
purpose 14:14 167:8 167:9 184:17,20 185:15
purposes 14:4 116:1 116:20 128:17 130:1
pursuant 2:15
put 15:3 21:14 30:9 48:23 52:3 53:19 54:15 60:8 123:13

153:20 154:2
puts 170:9 172:14
putting 43:14 94:22 167:12

q

quadratic 23:5
qualifications 106:20
qualified 16:7
qualifies 88:22 89:6 90:18
qualify 17:22 61:10 75:8 98:6 99:2
quality 54:10,12 148:10 163:14,21 164:11,20
queries 115:24 161:18 163:18 184:6 187:23 189:25 190:1
query 19:13 20:3 33:6,12,13,21 34:3 34:4,20 36:5 38:21 38:24 39:11 40:17 43:5,8,13,17,20 44:1,3,15,25 45:8 45:14,17 49:3,6 50:7 53:11,15,17 54:20,22 55:10,21 57:16 58:23,25 60:10 61:22 62:5,13 62:21 63:14,16,17 63:21 72:3,13,19,25 74:2,4,6 75:16 76:7 77:3,10,18,19,21 78:2,4 81:8 84:3 92:20 93:16,17,20 102:25 104:7,9 113:1,8,17,22 114:1 114:10,11,15 115:8 115:16 116:5,24 117:5,24 118:1,10 118:12,17,18,21 119:1,10,17,19,22 120:22 121:1,7,8,14 121:16,24 122:13 123:19 124:5,7,20 124:24 125:17 126:11,15 127:20 128:16,18 129:3,7 129:13,14,18,19 130:10,12,13 131:3 132:4,12 133:2,16 133:19,25 134:4,9 134:14 136:20,24 137:7,20 138:2,7 139:3 141:13,14,17 142:2,4,23 144:12 144:16,18 145:4,11 146:3,4,13,16 147:3 147:6,12,16,20,23 147:25 148:8,11,21 149:10,17,20,25 150:9 152:8,10 156:21 161:8,13 162:2,6,10,12,22 163:23 164:3,7,13 164:16,18 165:2,4,9 165:13,24 166:5,21 166:24 167:3,4,12 167:17,19,20 169:13 170:22 171:13,19,23 172:1 172:5,8,11,14,15,23 173:6,11 175:8 178:16,17,21 179:3 180:1,8,12,18,21 181:5 183:7 188:12 189:7,16 190:4
querying 36:8 43:18 188:1,10
question 9:10,11 15:14 17:15 19:19 19:23 20:14,20 24:2 24:4 27:21 31:4 34:8,11 38:10,23 39:15,17,18 44:9 50:15,16,17 52:13 52:18 56:11 58:5,19 59:19 62:7 66:7,13 66:16 67:8,20 72:7 76:3,16 77:19 82:14 82:15 84:7 85:1 86:11 89:2 92:12 93:19 96:10 98:10 98:11 99:12 102:16 103:1 120:4 125:15 126:9 127:11 133:7 133:10,18,23 136:22 142:10 150:5 153:21 156:6 166:11 168:9 171:8 173:24 178:24 179:11 180:14 185:8 186:23 189:9
questioning 34:7
questions 8:25 123:14 159:7 190:14,16
quibble 33:22
quick 179:16
quote 119:21

r

r 3:2,9 4:8 7:19,19 112:1,12,12 192:1 194:5
radio 176:8
random 60:18,19,25
randomly 60:21 61:4
range 30:5 32:12 36:21 56:3
rank 165:10,16,23
ranked 170:5,10,12 170:14,20 173:15 173:19 174:9
ranking 165:22 166:1
rate 23:2,6 144:19
reach 90:15
reached 90:6
read 13:11,24 15:5 20:16,17,18,25 47:17 69:24 100:2 100:23 109:14,15 121:25 122:20 150:6 191:7
reading 15:20 47:16 100:2 133:14 139:14 170:11 194:16
reads 109:17 133:11
ready 143:24
real 2:16
really 38:13 71:12 99:13 103:8 110:3 122:22 138:25 148:5 162:20
reason 8:6 193:5 194:12
reasonable 114:19 177:14,19
reasons 185:17
recall 21:7 22:4,6 25:21 26:23 27:4,7 27:10 41:20,24 42:4 46:8,20 80:7 115:10 116:14 122:14 126:3 157:20,23 158:2,3 170:2,17 174:4
receipt 194:15
recess 64:11 106:12 111:4 160:9 179:21 190:10
recognition 18:24 19:2 81:16,19,22
recognize 112:15 124:9
recollecting 19:18
recollection 20:4 113:15 125:7 135:1 135:24 136:8 170:11
recommended 81:14
reconfirm 170:4

record 6:2,10 7:2 8:23 16:22,24 17:4 20:18 23:20 34:2,3 35:5,24 40:18 43:8 46:13 48:17,21 50:9 50:22 51:12,13,19 52:9 57:10,14,15,17 58:21,24 59:3,15,18 59:20,23 60:6,15 61:20,21,25 62:2,5 62:12,15,21 63:1 64:1,9,13 66:2,9,10 66:11 71:10 72:21 73:2,11 74:2,2,3,5,7 76:23 77:24 78:6 92:22 96:24 97:2 101:10 102:6,22 103:12,22 104:10 104:12 106:8,10,14 108:14 111:3 112:4 121:23 124:11 134:25 145:19 150:7 158:18 160:7 160:11 162:7 179:20,23 180:3 183:23,25 188:17 190:6,9,12,18 192:12
recording 6:8
records 19:12 20:2 20:10 45:18,20 47:6 47:12,20 48:2,15 49:1,5 50:21 51:3 51:16,17,22,25 52:6 52:20 53:4 57:19 60:16,17,22 61:5,14 61:14,19 63:5,12 64:1 65:15,16,18,20 66:4,22 67:4 72:4 72:18,24 73:20 75:10 76:14 77:11 92:21 100:12,13 101:11 103:10,11
recovery 61:2 64:1
reduce 141:3
refer 10:14,19 17:13 18:5 33:5 47:5 88:19 104:15 139:24 171:12
reference 31:1,9 32:21,22,23 33:4,11 38:21 44:6,13 56:7 69:20 74:14 91:23 92:1,3 93:5 100:10 104:13 108:8 109:6 110:15 113:7,9,18 113:20,23 114:17 114:24 115:9,16 116:8 117:1,6,24 118:3,19,23 119:10 119:20,24 120:18 120:20,21 121:3,9 121:14,15,17 122:13 124:9,16,25 125:18 126:11,15 127:20 129:21 130:3,20,25 131:21 132:12 133:3,5,14 133:20,24 134:3,9 134:15 135:7,12,16 135:18,23 136:12 136:19 137:4,17,21 138:2,19 141:10,15 141:18 142:2,5,5,24 143:6,17 144:16 145:11,14,24 146:3 146:5,20,23 147:3,5 147:20,21 148:1,5,9 148:13,22,23 149:11,16,17 150:1 150:3,10,12 151:6,8 152:8,10 154:4,5,8 154:22 155:5,9,11 156:3,9,11,22 157:7 157:9,14,16 158:16 159:7 160:23 161:10,15 162:2,12 162:22 163:13,20 163:23 164:9,10,10 164:11,14,16,18 165:3,24 166:20,25 171:7,14,20,23 172:24 176:17 178:12,18,20 179:2 179:25 180:7,17 184:22 185:11 189:7,16 194:7
referenced 119:4 166:7 167:14
references 90:23 112:18 113:3 115:1 115:25 137:6 161:18 163:11 164:4,21 165:3,10 165:14,20 166:13 166:15,23 167:5,6 167:18,21 170:21 170:24
referred 10:8 12:12 12:23
referring 18:7 47:19 86:18 108:11 161:21 185:1,2
refers 68:8 70:17 88:14 100:7 107:10 140:2 176:22
regard 11:25 13:21 89:25 90:16
regarding 90:21,22 95:25 102:9
regardless 87:21
regular 143:17
related 108:8,9,10 108:12 192:15
relates 60:4 91:5 120:12
relation 31:10
relationship 99:23 158:22
relative 113:1 114:14 115:9 116:17 127:2,4,8 161:8,9
relevant 107:11,13 107:14
reliably 71:4 75:7
relied 14:4,21
rely 12:6,7 13:9,23 14:23 109:6
relying 117:12 188:13
remainder 74:14
remaining 50:23
remember 21:16 109:13
render 73:8
rendered 88:24 89:8
rendering 88:1
renders 35:13
repeat 20:13 24:2 31:5 39:17 44:8 52:18 62:7 67:8 92:12 114:1 136:22 150:5 156:6 166:11 171:8 178:23 179:11
repeated 140:6
repeating 140:6
rephrase 31:5
report 42:18 124:12
reported 1:18
reporter 2:17 7:15
represent 136:20,24
representation 31:2 105:9 114:11,23 161:13,14
representations 116:5
representative 52:6 134:13,16
represented 167:13 168:13,14,15
representing 6:12 7:16 162:5,6
represents 31:9,17 52:15,20 161:17
request 10:3

require 75:18 119:10,13 136:19 136:23
required 29:8 102:10 104:19 105:10,23 116:13 116:22 128:4 193:21
requirement 104:14 105:7 137:2
requirements 107:6
requires 126:22
research 26:17 48:23
respect 14:2 52:1 65:17 110:19 121:15,23 122:13 124:25 125:15 139:4 150:21
respective 152:22
rest 35:21 63:7 99:15
restate 89:2
restricted 172:2
restriction 126:1
restrictions 126:4 172:14
result 28:6,12,16 35:18 43:5,7 44:3 44:16,24 57:16 67:15 86:1,6,9,13 86:14 91:2 92:3,20 93:15 128:24 137:9 144:22 150:19 152:3,15,24 153:1 162:10 173:17 181:1
results 102:17 170:18 172:22 173:2
resumed 112:12
retrieval 18:21 19:3 107:16,18 108:2,5
retrieved 172:3
retrieving 52:11
return 28:19,20,24 32:17 35:4,6 44:16 67:15 74:11 82:17 82:24 83:10 85:23 86:1,6,9,12,13 87:16,20 88:13,17 91:2 92:5,13 93:17 114:3 138:11,14 139:7 142:5 150:18 151:11 152:2 154:1 170:21,23 171:1 172:16,21
returned 29:15 43:17,19 85:4,8 93:4,10,20 194:15
returning 57:13
returns 53:2 73:19 87:22 91:22 162:9 169:22
review 10:1 15:10 16:15 19:5 61:14,15 61:19,21 94:7 194:11
reviewed 120:11
revisit 88:24 89:7,24 90:8,19
right 10:6 11:19 13:6 14:14 24:15,16 25:2,16,20,24 29:25 31:14,22 33:13 35:19 36:7,12,16 38:25 39:23,24 40:3 40:23 41:5,15,19,23 42:15,25 43:9,9 44:1,2 45:9,11,18 46:13 48:18,19 50:16 51:9,10,17,21 52:2,10,23 53:5,13 53:16 54:13,15,22 55:20,22 57:11,14 57:15 58:22 59:4,5 59:7,22 60:1 61:20 63:4,14,16,22,25 64:20 65:13,18 66:1 66:9,11 67:1,15 70:12 72:19 73:16 75:4 76:5,6,13,15 76:19 78:11 79:18 79:20 80:23,24,25 81:4,7 82:17,24 83:9,14,24 84:2,8 84:20,22 85:22,22 86:6 87:5,8,11,12 87:15,17,23 88:15 88:18 90:7 91:3,6 92:2,21,25 93:11,18 94:16 95:18 97:23 98:23 99:5,24,25 100:11 101:14 102:18,19,20 103:22 105:4,11,16 105:19 106:22 110:16,22 114:1 116:1 118:11,13,17 120:7 121:3,11 122:4 125:14 126:13 130:9 131:23 138:22 140:2,4,25 141:16 143:9 145:10 148:19 149:11 151:1,12,15,23 152:6,17 153:9,11 153:13 155:6 156:5 158:19 161:10,15 161:19 164:18 167:3,8,9 170:6 172:4 174:8 176:25 177:4 181:2 186:1
risk 61:6
rothstein 2:13 4:3 6:15 7:11
roughly 11:23 139:16 158:21
row 35:16 122:10 124:20
rows 131:5
run 91:22 93:3 124:5 165:19 172:17,19,23
running 74:20 154:15 155:21 159:4
runs 140:18

s

s 3:2 7:19 112:1,1,1 112:12 161:19,21 193:5
sample 115:24 129:3 130:12 133:5 136:6
santa 3:16,17
satisfy 153:24 173:18
saying 22:6 46:17 57:20,21 80:9 87:1 120:6 127:7
says 70:7 101:15 108:23 109:16 123:19 124:8 128:15 139:16 144:1 146:13 149:13 157:23 174:15 177:6,11 186:10
scales 99:22
scan 30:17 32:15 67:17
scans 70:18
scenario 75:2 149:25 150:9 151:3 152:7 157:11
scenarios 21:15 81:8 127:4,5
scope 91:25 92:18 96:4,13,21 97:15 98:12,17 99:9
score 113:24 114:3 141:16,18 142:16 145:11 162:8
scores 117:11 145:15

scoring 114:4,5
scorings 114:6
scroll 167:5,7
sean 3:19 7:7
search 19:17,21 21:3,4,6 26:18,19 27:2,8,12,13,14,17 27:20 28:2,9,10,11 28:14 29:2,4,12 30:12 32:4,6,9 34:19 35:7,11,12,15 35:20,21 37:15,17 40:22 41:1,4 42:3,6 42:13,14,23 44:5,7 44:12,14,18,23 45:3 45:7,12,22,23 46:12 46:12,23 47:11,13 47:21 49:2,6,13,14 49:18,19,20,23,24 50:8,8,11,11 51:4 52:16,21,24 53:9 54:20 56:9,13,20 57:5 58:6,17 59:6 59:15 60:22,23 61:5 61:9,11,16,18,24 62:10,22 63:2,3,3 64:16,18,22,23,24 65:5,8,10,13,19,21 65:23 66:1,6,8,12 66:12,17,19,21,23 67:3,6,7,11,11,13,16 67:21 69:20,23 70:1 70:2,3,16,22,23,25 71:2,7,9,16,19,21 72:2 73:8,12 74:21 75:25 77:11,14 78:15 79:11,12,16 79:18,22 80:18 81:24 82:5,17,19,23 82:25 83:2,3,9,14 83:17,22,24 84:1,22 84:25 85:4,6,9,11 85:14,15,16,18,22 85:22,24,25 86:2,3 86:5,8,12,19,20,21 86:23 87:3,7,8,9,15 87:18,19,24,24 88:3 88:15,15,21,23 89:4 89:7,13,14,15,17,21 89:23,25 90:5,16,18 90:24 91:1,1,5,15 91:22,24 92:5,8,13 92:17 93:3,5,7,18 93:21,24 94:16,25 95:4 99:22 104:11 114:12 119:5,9,9 120:13 125:16,19 131:7 132:17 134:1 134:18 135:2,11 137:9,25 138:17,24 139:7,20 140:3 141:21,24 142:9,17 143:11,13,16,18,20 144:2,17,20,23 145:5,23 146:2,17 146:24 148:18,18 149:9 150:2,11,17 150:19,22 151:9,10 151:11,18,18,25 152:1,1,4,5,11,14,18 152:25 153:2,4,16 153:23,23 154:5,14 154:15 155:22,23 158:13,14,17,18,19 158:23 162:5 165:19 166:13,16 167:9 170:23 172:2 172:17,19 173:7,19 174:17 176:2,14,16 177:21 181:8,9,11 182:6,12 183:13,15 184:19 185:19 186:5,6,7 187:24 188:8
searched 46:24,25 47:14 48:2,7,13 80:6 82:3 85:5 99:24 139:19 172:7
searches 37:6,11 63:15 89:4 137:12 138:16 139:2,6 182:5
searching 25:24 26:6,10,25 29:19 33:4 37:8 60:13 63:12 70:8,12 78:21 78:25 79:8,16 94:11 94:19 98:2 112:22 139:19 151:21 155:10 175:13 189:21
second 12:2 28:3 73:24 74:6,7,11 79:11 82:11 118:14 118:22 120:8,23 121:6,9 122:10 127:10 131:13 141:12,17 144:8,23 145:4,19,23 147:8 182:7
seconds 43:20
section 21:1 121:3 140:10
see 30:17 47:8,15 55:9 69:19 91:17 95:12,22 96:8 97:6 97:9 103:7 109:18 115:23 120:15 121:1 123:20 124:16,19,22 125:12 128:16 129:21 130:25 131:3 144:25 146:12,20 169:4,8 177:10 186:14 188:16
seeing 158:2,3 170:2 170:17
seek 73:4
seeking 28:8 40:10
seen 60:11 143:17
segment 44:2 135:7 135:9 148:12,23 149:22 150:3,12,23 151:4 152:12,17,20 153:22 156:2 169:14
segments 43:25 148:8 149:5 155:24 156:8 172:13
select 51:14 139:18 142:16 155:25 190:2
selected 115:25
selecting 96:19
selection 60:18,19
send 136:3
sense 33:17 77:7
sensitive 6:3
separate 22:17,20 24:6,22 67:22 83:6 86:16 168:9
separately 15:24
september 109:17
sequence 75:1,12 114:18 115:7 124:16 146:11 147:5,11,15 148:1
sequences 175:13,13
sequential 32:14 67:6,11,17 70:2,3
sequentially 30:16 70:19
served 8:17 31:1
session 4:11
set 13:3 39:21 43:12 60:23 73:7 82:2 83:20 89:6 92:20 99:24 100:8 101:2 103:5 105:15,17,18 106:1 118:9 119:19 119:23 123:14 127:23 138:19 145:15,15 158:7,8 169:17 170:20 175:11 182:2 187:21,25 192:10 192:20
sets 80:6

setting 21:9 31:13 33:19,20
setup 72:23
sheet 193:1 194:11 194:12,13
shift 113:25 121:18 122:3 157:3,4
shifted 118:10,12 121:7,14,16,23 124:24
shifting 122:12,17 122:23 145:13
short 146:3
shorthand 2:17
showing 91:11
shown 191:11 194:13
signature 60:6 194:12
signatures 60:8
signing 194:12,16
similar 19:12,13 20:2,3 21:10,19 51:6 74:21,23 77:9 84:4 95:6 166:17 167:15 168:16
similarity 75:17 167:24 168:6,18,22
similarly 52:12
simple 55:15 63:19 72:24
simplify 61:5
simply 63:7 67:20 153:20
sincerely 194:20
single 5:16,17,18 32:21 34:14,18 46:9 56:8,19 58:1,21 59:3,18 60:6,13 66:9 67:15 71:17 76:23 102:24 103:18 104:11 123:9,10 128:10 146:7
sir 126:8 194:9
site 26:17
situation 47:6
size 99:23 100:7,8 104:18 105:9,15,18 105:22 106:3 170:12 174:19 189:1
skadden 3:4 7:4,6
skadden.com 3:10 3:12
skill 44:20 106:21 107:1 108:16 187:4
skilled 33:16 38:19 46:21 79:4 82:3 83:5,11,16 85:10 94:14 175:4 177:13 186:12,17,25
skip 138:23 140:4,5
skipped 137:19 139:25
skipping 139:10 140:7
slate 3:4
slightly 180:8,18
slower 175:2
slowly 8:25
small 21:21
smaller 23:3 27:25 170:16
software 36:24 37:22 38:11 40:23 187:7,10,14,18
solo 172:22
solution 76:22 84:21 84:24
solutions 193:1 194:1
solve 20:7 59:4 76:10
solving 27:19
somebody 83:11
somewhat 19:22 175:2
song 43:21 44:1 57:7,10 58:8 81:2 98:1 156:5 161:3 166:24 172:7,12,21 172:23
songs 43:19 57:4,11 58:2 80:13 166:18 172:3 174:2
sophisticated 103:24 117:13,17
sorry 12:18 50:14 55:5 56:11 95:19 101:5 103:3 115:5 133:18 155:17 162:17 185:9 188:21
sort 29:7 60:3,3,8 83:23 93:14 96:20 107:19 185:18
sorted 28:9,20 29:5 30:24 184:23 185:7 185:12,14,25 186:4
sorting 185:15
sounded 155:1
sounds 99:25 111:1
source 187:15
space 50:25 51:2,5 53:23,25 55:2,14 56:18 60:9
speak 8:25 9:15,18
speaking 13:9 20:8 33:3,24 39:7 181:7 187:17
specific 12:7 19:20 21:16 36:13 41:1,8 41:11,18,21 43:25 81:7 116:14 157:22
specifically 12:14 14:6 39:4 41:25 154:11
specification 18:3,6 18:13 20:5,6 21:14 25:19 170:18 189:4 189:12
specifications 136:14
specifies 97:10 98:22 99:3
specify 92:7,16
specifying 58:15
speed 23:7 79:11,16 139:20 143:20
speeding 79:12
spirit 80:16
split 51:8 55:17
square 3:6
ss 191:2 192:3
ssd 163:6,9 165:1,3 165:10 167:15,18 167:22 168:15,16
sss 162:22 163:6,9 164:23,25 165:2,10 167:15,17,21 168:15,16
ssu 162:23 165:1,3,9 167:13,18,19,20 168:13,16
stage 139:11
stand 160:5
standard 29:4,10 34:25
standards 16:18 17:7,16,20,23
standpoint 33:16 80:23 177:12
star 130:18 144:7,8
start 27:5 54:20 84:18 113:19 117:14 129:17 130:7 176:9 181:20 182:9,11 183:6
starting 31:7 100:19
starts 182:2,9
state 2:18 7:1 106:19 139:9 171:10 191:1 192:2 192:7
stated 22:5 42:12

statement 16:10 43:19 80:9 122:21
statements 90:3
states 1:1
statistics 108:13,15
step 9:17 29:14,15 53:18 73:8 99:15 139:22
stick 34:5
stop 75:9 82:11 156:25
stopped 144:20
storage 104:14
store 30:2,24 31:18 31:24,25 38:1,3 40:14 51:11 102:21
stored 31:10 33:5,11
stores 101:10
storing 38:5,14
strategy 150:22
stream 178:5
strict 157:18
strictly 15:11,18
strike 37:21 80:4 133:13,22 163:7 164:6,24 166:19 172:18 182:15
string 72:22 75:8,16 75:24 76:7 77:3,11 114:10 115:16 123:20,23 124:9,10 128:19,23,24 129:7 129:21,25 130:2,7 131:1 133:11,14 146:16,19,23 162:5 162:6,22,23 163:5 167:3 174:16 175:1 175:10 176:21 177:23
strings 72:14,18 75:22 76:5 77:7 115:20 117:7 123:17 161:18 163:2 168:4,6 175:14
stroke 32:16
structure 38:1 58:15 60:9 63:23 77:8 187:7
structured 137:18 137:24,25
structures 186:13 186:19,21,21 187:1
studied 17:25 18:13
stuff 31:14 63:24
subject 191:10
sublinear 22:11,13 24:22,25 25:6 68:7 68:8 79:25 95:1,5 99:17,21 100:5 105:16
sublinearity 22:15 22:17
submitted 9:22,25 11:4 12:13 13:4 23:16 48:22
subscribed 191:17 193:21
subsegment 114:4,5
subset 45:10 48:6,12 58:24 59:5,20 60:25 63:4 67:2 76:10 84:25
subsets 60:7
substrate 164:15
substring 76:13,14 77:4 162:5,14,20 164:12,18 170:24 171:14,19,22,24 174:25 175:15 176:11
substrings 77:5 171:6 175:10
subtract 115:13
successful 127:18
suggesting 141:25
suite 194:1
summarize 20:22
summons 65:15
sums 134:11
supply 143:9
support 13:5
sure 20:15 23:13 24:4,8 26:13 31:6,6 33:23 36:2 39:19 44:10 52:19 58:4 62:9 64:6 69:25 72:9 82:16 89:3 91:10 92:13 109:22 110:13 115:15 120:5 126:7,24 127:22 149:2 150:6 171:9 173:25 178:25 179:18 185:2 190:7
swear 7:17
switch 23:22
sworn 7:20 8:2 191:17 192:11 193:21
symmetric 168:5
synonymous 22:10 22:12,17
system 20:1,10 38:11 43:19,24 44:21 59:21 62:1,9 62:10 63:11,14 75:21 77:1 81:1,14 81:16,18 112:22,25 124:6 161:2,3 169:22 178:16 179:7 189:24 190:1
systems 36:24 37:22 107:17

t

t 112:1 192:1,1
tab 10:20,23
table 29:21 30:12 31:25 32:4 34:19 35:16,17,22 36:8,11 36:11,12,16,22,25 38:2,16 39:9 40:8 47:22,25 48:14 49:16 50:1,12,19 51:7 53:1,6,19 56:17 57:1 60:9 63:20 77:10 184:23 185:11 186:11
tables 27:2 29:17,18 29:24 37:5,20,24,24 37:25 38:20 40:12
take 9:13 17:24 19:14 32:11 39:8 40:22 47:2 49:17,21 50:5 52:8 56:4,6 68:11 69:14 71:1 77:23 81:8 95:7,15 100:17,20 104:23 120:3 121:20 123:18,22 124:14 125:5,20,22 126:5,7 126:19 131:9,22 132:18,22,25 134:5 134:20 143:22 145:5 146:10 149:25 150:9 152:7 155:15 159:9 160:2 169:7 173:22 177:8 179:16 188:18,19
taken 6:17 49:12 64:11 106:12 111:4 160:9 179:21 190:10
takes 36:19 133:4 139:10 158:18
talk 31:19 37:14,16 53:22 171:16
talked 57:12
talking 9:6 10:16 17:2 22:14 39:23 70:11 180:12 183:12
talks 19:21 41:9 140:11 169:25 186:20
teaches 148:17 150:24 165:16

teaching 135:15 157:17 174:3
teachings 12:2 129:24 136:23 157:13
technical 11:5 13:18 18:22
technique 175:21
techniques 27:8,11 187:19
technologies 1:6 6:21 7:8,13 9:23 193:3 194:6
technology 81:22 189:18
tell 58:23 184:25
ten 43:20 45:16 55:18,19,19 165:20 165:22 166:6,9,13 166:15 169:22
tendency 8:24
tenth 104:23
term 22:8 24:6,8 25:5,9,10,18 30:19 33:4 42:9 45:25 46:15 64:16,19,24 65:5 67:20 68:7 69:20 70:10,15,16 78:9,9,15 83:6,21 85:13 86:16,18,22 87:3 91:19 93:22,23 94:3,16 95:3,25 96:5,14,21 98:7 99:16 100:5,15 140:13 176:20,24
terminate 74:10 75:19
terminology 10:18 18:5 32:25 33:14,23 34:6
terms 10:21 25:14 29:22 33:3,18,25 47:24 65:11 68:5 90:9,12 107:5 127:17 152:23 166:2 167:24 169:25 171:17,18 177:21 180:12,13
testified 7:21 14:13 121:13
testify 8:2,7,12 9:21
testifying 8:3
testimony 13:22 14:17 15:6 191:8 192:12
text 18:7 38:7 108:2 120:1,11 121:5 139:23 143:23 161:12,14 185:3
textbook 38:1
thank 7:14 190:5,13
thanks 160:8
theory 129:15
thereof 31:2
thing 33:1 59:11 60:10 76:18 83:13 98:13 109:22 143:11 145:20 156:24 157:25 158:5 184:11
things 21:15 55:18 55:19,20 57:13 68:1 68:10 80:24 84:4 87:22 113:6 115:11 117:15 122:4 140:8 154:11 164:20 165:6,7,15 166:2 167:10,25 171:20 186:3
think 13:25 14:9,17 14:19 15:13 16:10 18:24 19:20 24:24 31:3,4 33:17 35:15 39:18 41:8 42:16 44:19 49:10 50:4,20 59:8 62:8 63:19 66:13 75:15 76:16 79:9 80:16 90:7 97:1 98:5,11 99:8 99:13 106:5 107:3,4 107:6,22,24 108:1 108:21 118:16 122:1 137:1,15 155:8 156:12 158:6 159:4 175:20 176:1 177:12,18 178:1,4 186:16,24 187:3
thinking 54:8
third 105:21 141:12
thirty 194:15
thought 50:14
three 12:25 21:6 55:17 72:25 73:1 75:24 107:10 143:1
threshold 86:11 142:19,21 155:25 156:25 157:8
thursday 1:13
tie 62:19
tied 140:9 164:1
time 2:16 6:13,25 23:21 28:7 32:3 49:18 62:21,23 63:14,16,17,21 64:4 64:8,12 65:17 68:8 68:8,9 70:9 75:9 77:18,20 78:1,2 79:25 87:17 99:22 106:9,13 111:2 112:2,3 158:14,17 158:23,24 159:1 160:4,6,11 179:19 179:22 188:1,10 190:4,8,11,17,21
times 3:6 67:19 104:17
title 91:23 92:1 93:4 98:1 174:2
titles 92:6,7,14,15 174:5
today 6:12,24 7:15 8:8,23 9:21 10:15
today's 9:14 34:6
tolerance 157:18 169:2,10,12,16,21 169:24 170:13,14 170:20 171:12,16 171:17
tolerant 21:20
toll 194:2
top 12:11 21:1 50:4 59:8,25 95:11 108:24 120:18,19 123:18 126:3 128:15 146:12 157:20 171:25 177:2 184:14
topic 110:23
total 101:11 104:24 132:8,10 134:10
track 186:3
trademark 1:1
transcript 191:10 194:10,11
transmission 21:22
transmitting 96:24
transverse 184:15
traverse 36:14,17
tree 41:13,18 50:24 54:2,4,5,7,17,20,22 55:1,4,8 56:1
trees 37:14 41:10,10
trial 1:1 6:22 10:23 42:8
tribunal 10:25
tried 13:12 20:22 23:23
true 28:19 45:13 79:6 127:19 191:10 192:12
try 8:24 34:5 72:9 123:14 161:3 163:25
trying 27:20 33:7,22 72:20,20 103:15 129:11 145:18 154:21
tune 161:2
turn 6:5 9:10 71:23 91:9 143:10,21

187:5
turning 60:14 119:15 175:25
two 11:24 12:24 29:23 33:25 39:22 67:22,24 74:25 79:9 79:9 93:14 105:14 105:17 107:10 122:4 126:20 127:18 128:7 130:21 140:3,24 152:3 155:2 162:25 163:2,11,18 164:3,5 164:20 165:6,7,9,13 166:22 167:17 168:6,8 171:20 178:22 179:4,8,12 182:23
type 33:18 38:6 42:2 86:2 89:13,15 91:1 97:18 98:11 107:23 139:1 153:11,13 172:14 175:8 184:5
types 29:23
typical 32:8 67:5,10 81:12

u

u 161:19,21
u.s. 5:7,9,10,12,15 5:19,21 10:5 68:16 68:18,20,22,25 69:3 69:6,9 112:10 160:14,17 178:9
uh 69:13 91:18 109:19 110:17 112:20 116:7 132:5 134:2,7 148:20 159:23 173:10
ultimately 90:11
umbrella 11:21
un 139:17
unacceptable 178:2
underlying 166:17 168:21
understand 8:1 9:2 9:20 10:16,24 13:12 19:7,15,24 22:21 24:22 31:8 42:1,7 43:1 58:3,19 66:13 68:4 69:22 82:16 84:7 87:4 90:10 94:2 97:14 101:1 123:16 139:23 140:21 143:25 149:4 156:17 169:9 185:10
understanding 15:9 15:16 18:10 22:1 24:5 43:4 72:10 80:1 87:2 88:10 94:24 96:3,12,17 101:6,16 113:12 125:2,9,21 127:15 134:22 135:1 138:21 140:10 141:8 145:9 149:6 149:12 156:1,19 157:12 158:11 159:17 161:25 162:3 169:11,23 171:15 173:21 174:2,22
understood 36:4 39:15,18 41:13 45:25 84:16 171:3 186:12 189:20
undesirable 22:3,5 22:7
unimportant 135:17 135:22 136:6,16 137:6,8,19,21 138:3 138:24 139:10,25 140:7
unique 31:22 92:2,2
united 1:1
unlimited 170:8
unmarking 140:12 140:13
unnecessary 139:19
unpatentable 11:10
unquote 119:21
unusual 178:1
usage 33:8,9
use 25:8 29:2,18 30:6 33:4,12,18,20 36:24 38:15,20 42:10 47:11 51:13 52:25,25 60:7 70:16 77:5 79:15 85:13 113:1 114:12 116:15 121:20,22 129:3 140:13 145:9 146:16 149:13 150:16 151:9 162:13 168:5 172:22 175:13 182:25 183:14 187:10 188:3,6,14 190:2
user 135:16 136:3 136:18 137:7,12 143:8 155:6,11 172:1 174:1
user's 138:7
uses 46:20 70:10 80:11 132:11 177:20
usually 24:9 31:16 31:19 34:17 95:4 136:14

v

v 193:3 194:6
vague 135:25
valid 105:24 114:23 176:23
validity 91:3
value 24:14 28:5,7 29:15 30:15 31:19 38:2 53:2,4,8 55:23 55:23 115:14 178:22 179:4 181:5 184:24 185:12
values 32:1,18 53:8 55:6,22 77:17,21,23 78:3 113:1,2 115:3 115:7,14 116:18 161:8,9 179:9,14
vantage 41:10
variable 24:11,12,18
variety 41:14
various 16:14 54:4 131:4 177:8
vector 105:3
verify 129:11
veritext 6:12 7:17 193:1 194:1,7
version 110:5
versus 6:20 54:17 127:8 154:14
video 35:22 38:8 80:13,18 81:19,22 104:9,11 108:5
videographer 4:15 6:1 7:14 64:8,12 106:9,13 111:2 112:3 160:4,10 179:19,22 190:7,11 190:17
videotaped 1:12 2:11
view 11:18 25:5 52:9 71:20,22 86:15 86:22 89:10,12 91:24 100:14 102:8 106:24 107:3,12 124:4
views 95:24
virtue 148:7
visit 59:23 66:2,24
vs 1:5

w

w 194:1
wait 9:9,10
waived 194:16
walk 123:15

want 20:15 33:23 59:19 62:1,10 73:1 76:10 80:10 81:4 88:24 89:7,23 90:19 95:14 102:18,20 120:3 123:15 146:10 186:2 187:24
wav 159:21,24
way 13:19 28:4 29:11 30:2 31:17 32:8,9 34:24,25 36:8 40:12 41:18,21 43:14 46:6 50:18 52:10 54:8 55:2 56:14 59:9,21 63:23 70:8,11,17,18 76:18 85:18 89:11 102:19 103:9,23 110:7,19 116:11 122:5 128:1 129:10,13,17 137:2 138:21 141:2 149:3 162:13,16,21 163:23 170:2 171:15 177:21 178:4 181:20,23,25 182:7,13,20 187:23 192:17
ways 29:9 41:15 46:15 55:13 113:5 181:18
we've 37:5 86:8 110:9 156:4,10,20
weighed 168:3
weighted 168:2
whatsoever 180:25
whereof 192:19
whichever 124:1
whispering 6:4
widely 26:16
wikipedia 5:13 12:22,25 108:25 109:9 110:2,5,15
witness 5:2 6:24 7:20 8:18 23:21 111:1 191:6 192:9 192:13,19 193:4 194:8,10
witness' 194:12
word 73:1 99:7,17 136:16 181:13,19 181:22 182:3,9,23 183:14,24 185:22 185:23
words 72:25 75:19 99:15 142:22 169:20 182:17 183:5,6,19 184:10
work 17:25 18:14 26:11 32:21 33:5,6 33:11,12 39:3,3,13 55:3 62:18 72:3,13 76:25 81:3,15 84:15 97:12,21 99:2,4 113:8,9 114:17,24 116:5,8 117:1,6 118:10 125:18,18 129:3 134:9,15 136:21,25 137:21 138:2 143:17 148:22,23 149:22 150:1,10 158:16 171:13 178:17,17 178:21,21 179:3,3
worked 26:20 27:9 27:11
working 107:15
works 21:10,11 27:17 29:22 32:22 32:24 36:9 80:14,18 113:13 134:23 135:2 178:18 179:8 179:13
worth 54:25
wrap 159:8
write 122:24 123:2
written 81:21 90:3
wrong 54:1 141:5
wrote 46:9 123:4

x

x 1:3,8,10 5:1 55:17 55:23,25 56:1,2,3
xyz 55:17,21

y

y 7:19 55:18 56:1,5 112:12
yeah 19:3 22:5 49:10 82:15 86:22 105:12 122:9
year 84:17 110:10 110:13
years 107:10
yellow 147:11
yep 42:25 97:13 99:6 100:4 185:8
yield 149:20
yields 142:25 151:6 166:13
york 1:13,13 2:14 2:14,14,18 3:7,7 4:6 4:6 6:17,17 192:2,4 192:8

z

z 55:19 56:1,6
zero 134:11 158:7,9 171:1

Federal Rules of Civil Procedure

Rule 30

(e) Review By the Witness; Changes.

(1) Review; Statement of Changes. On request by the deponent or a party before the deposition is completed, the deponent must be allowed 30 days after being notified by the officer that the transcript or recording is available in which:

(A) to review the transcript or recording; and

(B) if there are changes in form or substance, to sign a statement listing the changes and the reasons for making them.

(2) Changes Indicated in the Officer's Certificate. The officer must note in the certificate prescribed by Rule 30(f)(1) whether a review was requested and, if so, must attach any changes the deponent makes during the 30-day period.

DISCLAIMER: THE FOREGOING FEDERAL PROCEDURE RULES ARE PROVIDED FOR INFORMATIONAL PURPOSES ONLY. THE ABOVE RULES ARE CURRENT AS OF SEPTEMBER 1, 2014. PLEASE REFER TO THE APPLICABLE FEDERAL RULES OF CIVIL PROCEDURE FOR UP-TO-DATE INFORMATION.

Filed on Behalf of NETWORK-1 TECHNOLOGIES, INC.

By: Charles R. Macedo (Reg. No. 32,781)
Brian A. Comack (Reg. No. 45,343)
Amster, Rothstein & Ebenstein LLP
90 Park Avenue
New York, NY 10016
Telephone: (212) 336–8074
cmacedo@arelaw.com
N1-Google-IPR@arelaw.com

Gregory Dovel (admitted *pro hac vice*)
Dovel & Luner, LLP
201 Santa Monica Blvd., Suite 600
Santa Monica, CA 90401
Telephone: (310) 656-7066
greg@dovellaw.com

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

GOOGLE INC.
Petitioner

v.

NETWORK-1 TECHNOLOGIES
Patent Owner

Cases IPR2015-00343, IPR2015-00345, IPR2015-00347, and IPR2015-00348
Patents 8,640,179, 8,205,237, 8,010,988, and 8,656,441

DECLARATION OF DR. GEORGE KARYPIS

Table of Contents

I. Background to my opinions in this Declaration. 1
A. Expertise. 1
B. Assignment. 3
C. Approach. 4
D. Understanding of the law. 5
II. Summary of the IPR Patents and asserted art. 8
A. The IPR Patents. 8
1. '237 patent (Ex. 1001 '237 IPR). 8
2. '988 patent (Ex. 1001 '988 IPR). 10
3. '179 patent (Ex. 1001 '179 IPR). 11
4. '441 patent (Ex. 1001 '441 IPR). 13
B. The asserted art. 16
1. Overview of Ghias—Ex. 1010 (addressed in the '237, '988, '179, and '441 IPRs). 16
2. Overview of Iwamura—Ex. 1012 (addressed in the '237 and '988 IPRs). 19
3. Overview of Conwell—Ex. 1009 (addressed in the '179 and '441 IPRs). 22
4. Overview of Philyaw—Ex. 1014 (addressed in the '179 and '441 IPRs as a secondary reference). 24
5. Overview of Chen—Ex. 1008 (addressed in the '237 IPR as a secondary reference). 25
III. General Findings. 27
IV. General concerns with the IPR Petitions and Dr. Moulin's Declarations (Exs. 1004 in each IPR). 30
V. Claim Constructions. 38
A. sub-linear ('237 patent). 40
1. the words used in the construction: "size of the dataset" 41
2. '237 specification. 43
3. Petitioner's Declarant. 47
B. non-exhaustive search ('237, '988, '179, and '441 patents). 54

1. The Board's preliminary construction of "non-exhaustive search" is consistent with the understanding of one of ordinary skill in the art: "a search that locates a match without a comparison of all possible matches." 54

2. The Board properly rejected Petitioner's assertion that a "non-exhaustive search" should be construed as "a search that locates a match without conducting a brute force comparison of all possible matches, and all data within all possible matches." ...56

C. neighbor search / identifying a neighbor / neighbor / near neighbor ('237, '988, '179, and '441 patents).......... 58

D. approximate nearest neighbor search ('237 patent). 60

1. "identifying a close match that is not necessarily the closest match" 61

2. "sublinear" 62

VI. '237 patent. 65

A. '237 Ground 1: The instituted claims of the '237 patent are not anticipated by Iwamura. 65

1. sub-linear time search (claims elements 1(b) and 5(b.2)).........66

2. approximate nearest neighbor search (claim elements 9(b) and 13(b.2)).......... 85

3. nonexhaustive search (claim element 25(b)). 95

4. identify a neighbor / near neighbor (claims elements 1(b), 5(b), and 25(b)).......... 126

5. sublinear approximate nearest neighbor search (claim element 33(b)).......... 132

B. '237 Ground 2: The instituted claims of the '237 patent are not anticipated by Ghias. 134

1. sublinear time search (claim elements 1(b) and 5(b.2)). 134

2. approximate nearest neighbor search (claim elements 9(b) and 13(b.2)).......... 155

C. '237 Ground 3: The instituted claims of the '237 patent are not obvious over Iwamura and Chen.......... 166

VII. '988 patent. 168

A. '998 Ground 1: The instituted claims of the '988 Patent are not anticipated by Ghias. 168

1. non-exhaustive search (claim element 15(b)) 169
2. search identifying a neighbor (claim element 15(b)) 186
3. determining an action based on the identification (claim element 15(c)). 190
B. '988 Ground 2: The instituted claims of the '998 patent are not obvious over Ghias. 193
C. '988 Ground 3: The instituted claims of the '998 patent are not anticipated by Iwamura. 194
1. non-exhaustive search (claim 15(b)) 194
2. identifying a neighbor (claim 15(b)) 203
VIII. '179 patent. 204
A. '179 Ground 1: The instituted claims of the '179 patent are not anticipated by Conwell. 204
1. neighbor search (claims 1, 13, 25). 205
2. non-exhaustive search (claims 1, 13, 25). 231
B. '179 Ground 2: The instituted claims of the '179 Patent are not obvious in view of Ghias and Philyaw. 244
1. non-exhaustive search (claims 1, 13, 25). 245
2. neighbor search (claims 1, 13, 25). 251
IX. '441 Patent. 256
A. '441 Ground 1: The instituted claims of the '441 Patent are not anticipated by Conwell. 256
1. neighbor search (claims 1, 13, 25). 256
2. non-exhaustive search (claims 1, 13, 25). 259
B. '441 Ground 2: The instituted claims of the '441 Patent are not obvious over Ghias and Philyaw. 260
1. non-exhaustive search (claims 1, 13, 25). 261
2. neighbor search (claims 1, 13, 25). 264

I, George Karypis, declare:

I am making this Declaration at the request of Patent Owner Network-1 Technologies, Inc. in the following *Inter Partes* Reviews of U.S. Patent Nos. 8,205,237 ('237 patent), 8,010,988 ('988 patent), 8,640,179 ('179 patent), and 8,656,441 ('441 patent) (collectively the "IPR Patents"):

- IPR2015-00345 ('237 patent),
- IPR2015-00347 ('988 patent),
- IPR2015-00343 ('179 patent), and
- IPR2015-00348 ('441 patent),

(collectively the "IPRs"), all initiated by petitioner Google Inc. ("Petitioner").

I. Background to my opinions in this Declaration.

A. Expertise.

1. I am a Professor in the Department of Computer Science and Engineering at the University of Minnesota. I hold a Ph.D. in Computer Science from the University of Minnesota, granted in 1996. I began my post-graduate school career as a Research Associate in my current department. I became an Assistant Professor in 1999, an Associate Professor in 2004, and a Professor in 2009. I teach courses in Algorithms and Data Structures, Parallel Programming, and Data Mining, among other subjects.

2. I am a member of the Editorial Board of a number of academic journals, and I have chaired a number of academic conferences.[1] I am a co-author of the books *Introduction to Parallel Computing*, and *Introduction to Parallel Computing: Design and Analysis of Algorithms*. I am an author of more than 80 published journal papers, and more than 115 published conference papers.[2]

---

[1] Representative academic conferences include:

- Program Committee co-Chair of the ACM Recommender Systems Conference (RecSys'13), Hong Kong, China (2013);
- Program Committee co-Chair of the 13th International Conference on Data Mining (ICDM), Dallas, Texas (December 2013); and
- Program Committee Co-Chair of the International Conference on Data Science and Advanced Analytics (DSAA 2014), Shanghai, China, (November 2014).

[2] Representative papers include:

- "*L2Knng: Fast Exact K-Nearest Neighbor Graph Construction with L2-Norm Pruning*" David C. Anastasiu and George Karypis, 24th ACM International Conference on Information and Knowledge Management (CIKM), Melbourne, Australia (2015).

3. I have also developed a number of software systems for a variety of functions, including software for analyzing high-dimensional data sets. A copy of my curriculum vitae is attached as Exhibit A to this Declaration. It contains a more complete listing of my professional activities and background.

B. Assignment.

4. I have been retained by Patent Owner Network-1 Technologies, Inc. as a technical consultant. I am being compensated for my time at my standard consulting rate of $350 per hour. I am not receiving any compensation that depends on the outcome of the IPRs.

5. This declaration addresses the validity of:

- "*L2AP: Fast Cosine Similarity Search with Prefix L-2 Norm Bounds*" David Anastasiu and George Karypis, 30th IEEE International Conference on Data Engineering (ICDE), pp. 784—795 (2014).
- "*Comparison of Descriptor Spaces for Chemical Compound Retrieval and Classification*" Nikil Wale and George Karypis, IEEE International Conference on Data Mining (ICDM), pp. 678—689 (2006).
- "*Empirical and Theoretical Comparisons of Selected Criterion Functions for Document Clustering*" Ying Zhao and George Karypis, Machine Learning, 55, pp. 311-331 (2004).

- claims 1, 3–5, 7–9, 11–13, 15, 16, 21–27, 29, 30, 33, 34, 35, 37, and 38 of the '237 patent;
- claims 15–17, 21–28, 31–33, 51, and 52 of the '998 patent;
- claims 1-3, 6, 8–14, 18, 19, 21–27, 29–31, and 34–37 of the '179 patent; and
- claims 1–3, 6, 8–14, 18, 19, 21–27, 29, and 30 of the '441 patent.

C. Approach.

6. To develop my opinions, I have read:

- the four IPR Patents (the '237, '988, '179, and '441 patents);
- the four Petitions for *Inter Partes* Reviews;
- the exhibits accompanying the Petitions, including the four Declarations of Dr. Pierre Moulin (Exs. 1004 in each IPR);
- the four Decisions Instituting the IPRs; and
- the testimony of Dr. Pierre Moulin, dated August 19-20, 2015 (Ex. 2006).[3]

---

[3] In this Declaration, I identify the specific Petition, Declaration, and Decision that I am citing by including the corresponding patent abbreviation in a parenthetical. For example, I refer to the Petition addressing the '237 patent as Pet. ('237) at X; and the Moulin Declaration addressing the '179 patent as Moulin Decl. ('179) ¶X. Because there is only one Dr. Moulin Deposition transcript for all

7. In addition, I relied on my personal knowledge and experience with both research and development in the technology underlying the IPR Patents and the art asserted against the IPR Patents.

D. Understanding of the law.

8. My understanding regarding the law as applicable to this Declaration is based on my discussions with counsel. I have included in the text of my Declaration quotations from or references to certain legal cases or statutes that were provided to me by counsel to provide me with an understanding of the relevant law.

E. Person of ordinary skill in the art.

9. Through my education, experience and training, in academia and industry, and my analysis of the IPR Patents, I am familiar with the knowledge of a person of ordinary skill in the field of the IPR Patents at the time of invention in 2000.

10. For the purposes of this Declaration, I am of the opinion that a person of ordinary skill in the art with respect to the IPR Patents is a person with a Bachelor's degree in computer science, mathematics, or a similar discipline and two to three years of relevant experience, or a graduate degree in the same area.

---

four IPRs (Ex. 2006), I simply refer to Dr. Moulin's deposition testimony as Moulin Depo. Z.

In determining what would be the level of ordinary skill in the field as of the 2000 time frame, I considered the following:

(a) the educational level of the inventor, Ingemar J. Cox (it is my understanding that Dr. Cox has a bachelor's degree in electronics and computer science from University College London (1980) and a Ph.D. from Oxford (1983));

(b) the type of problems encountered in the art—*i.e.*, how to identify a digital work without modifying the work (*see e.g.*, '237, 1:30-36);

(c) the prior art solutions to those problems (*see e.g.*, '237, 1:37-4:4, and the prior art asserted by the Petitioner in the IPRs addressing related problems involving searching, matching, and identifying melodies, audio files, and other digital files within databases—Conwell, Ghias, Iwamura, Chen, and Philyaw);

(d) the rapidity with which innovations are made (based on my observations over the past 20 plus years, major innovations in content identification occur about every 5 to 10 years);

(e) the sophistication of the technology (developing content identification solutions is a moderately sophisticated technology); and

(f) the educational level of workers in the field (workers in the field generally had have at least a bachelor's degree in computer science, mathematics or a similar discipline and at least two to three years of relevant experience).

11. Based on these factors, it is my conclusion that a person of ordinary skill in the art at the time would have been a person with a Bachelor's degree in computer science, mathematics, or a similar discipline and two to three years of relevant experience, or a graduate degree in the same or related area.

12. I note that Dr. Moulin suggests that the person of ordinary skill in the art "would have been highly skilled, and typically would have possessed at least an M.S. in computer science, electrical engineering, or mathematics; knowledge of video and audio processing techniques; and 1-2 years of experience in audio, video, or image processing." *See e.g.*, Moulin Decl. ('237) ¶7; Pet. ('237) at 4. Dr. Moulin's opinion as to the person of ordinary skill in the art is similar to mine with respect to the degrees and years of experience, but I note that: (1) Dr. Moulin does not provide any rational underpinnings for his opinion; and (2) the phrase "highly skilled" used by Dr. Moulin in his description is a relative term and Dr. Moulin does not provide the context for this phrase.

II. Summary of the IPR Patents and asserted art.

13. In this Declaration:

- I use the term "work" to mean the item (*e.g.*, a digital audio or image file) to be identified using the search (*see e.g.*, '237, 6:51-56; '988, 7:17-20; '179, 6:18-21; '441, 6:49-52);
- I use the term "record" to mean one of the units in the reference database that the extracted features of the work may be compared to (*see e.g.*, '237, 6:16-20; '988, 6:46-50; '179, 6:21-24; '441, 6:15-18); and
- I use the term "database," "data set," or "library" to mean the collection of all records to be searched (*see e.g.*, '237, 6:23-30; '988, 6:50-60; '179, 6:30-36; '441, 6:24-30).

A. The IPR Patents.

14. Each IPR Patent (the '237, '179, '988, and '441 patents) involves a search that compares features from a given work to records in a reference database of potential matches to identify an action to be taken.

1. '237 patent (Ex. 1001 '237 IPR).

15. The independent claims of the '237 patent include the following elements:

[1] receiving or obtaining features extracted from a work;

[2] identifying the work using the extracted features to perform a search of the database, where the search is:

- a sub-linear time search to identify a neighbor (claims 1 and 5);
- an approximate nearest neighbor search (claims 9 and 13);
- a non-exhaustive search … to identify a near neighbor (claim 25); or
- a sublinear approximate nearest neighbor search (claim 33); and

[3] either (i) transmitting information about the identified work to the client device, or (ii) determining an action based on the identity of the work.

16. The invention claimed in the '237 patent includes two key features:

17. Feature 1: Although the language varies among the claims, each claim requires that the "identifying" be performed based on a search that has two properties:

(1) a sub-linear or non-exhaustive property (reflected in the underlined language):

- sub-linear time search … to identify a neighbor (claims 1 and 5);
- approximate nearest neighbor search (claims 9 and 13);
- non-exhaustive search … to identify a near neighbor (claim 25); and
- sublinear approximate nearest neighbor search (claim 33).

(2) a neighbor property (reflected in the underlined language):

- identify a neighbor (claims 1 and 5);
- approximate nearest neighbor search (claims 9 and 13);
- non-exhaustive search ... to identify a near neighbor (claim 25); and
- sublinear approximate nearest neighbor search (claim 33).

18. Feature 2: The system must either determine an "action" based on the identification (claims 25 and 33); or transmit information about the identified media work to a "client device" (claims 1, 5, 9, and 13). It is not sufficient to simply identify a match. Rather, an action must also be identified or information about the identified work must be transmitted to the client device.

2. '988 patent (Ex. 1001 '988 IPR).

19. The independent claims of the '988 patent include the following elements:

[1] extracting features from a work;

[2] identifying the work based on the extracted features by performing "a non-exhaustive search identifying a neighbor;"

[3] determining an action based on the identity of the work; and

[4] performing the action.

20. The invention claimed in the '988 patent includes two relevant distinguishing features:

(1) the "identifying" must be performed using a "non-exhaustive search identifying a neighbor;" and

(2) the system must "determin[e] an action" and "perform[] the action" based on the identity of the work. It is not sufficient to identify a match. Rather, "an action" associated with the match must be "determin[ed]" and "perform[ed]." '988, claim 15.

21. I note that the Board did not institute trial for independent claim 1 of the '988 patent and any claims dependent on claim 1. Accordingly, I do not address these claims in this Declaration.

3. '179 patent (Ex. 1001 '179 IPR).

22. The independent claims of the '179 patent (claims 1, 13, and 25) include the following five elements for identifying a work and performing a corresponding action:

[1] a database comprising: (a) electronic representations of works; and (b) electronic data related to an action corresponding to works;

[2] obtaining extracted features of an unknown work;

[3] identifying the unknown work by comparing the extracted features and electronic representations using a "non-exhaustive neighbor search;"

[4] determining an appropriate action based on the electronic data related to an action; and

[5] associating the determined action with the identified work.

'179, claims 1, 13, and 25.

23. The claimed steps are illustrated in Figure 1:

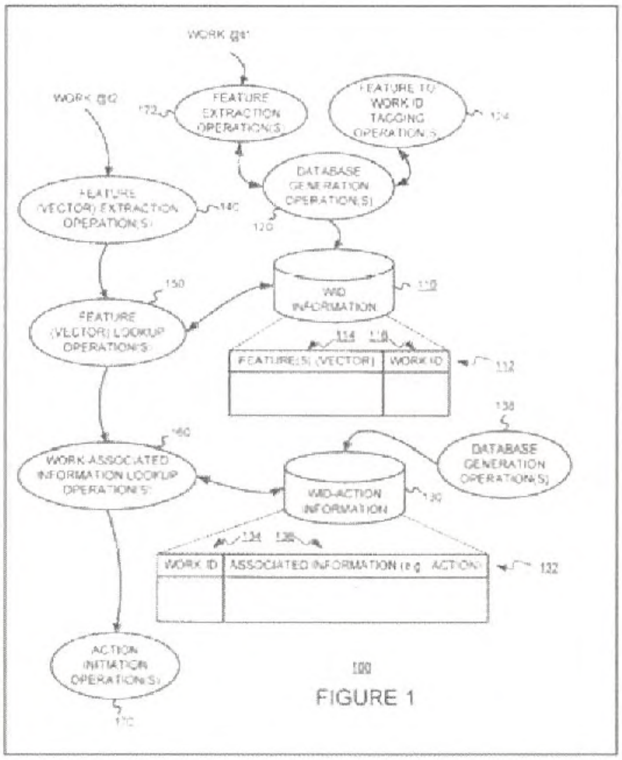


FIGURE 1

Figure 1 illustrates ("for work @t2"):

- "feature (vector) extraction operation(s)" (140) that extract features from the work ('179, 6:45-47);

- "feature (vector) lookup operation(s)" (150) that identify the work by searching for a matching feature vector ('179, 6:50-52);
- "work-associated information lookup operation(s)" (160) that retrieve(s) associated information, such as an action ('179, 6:55-58); and
- "action initiation operation(s)" (170) that perform(s) some action based on the associated information ('179, 6:58-60).

24. The invention claimed in the '179 patent includes two relevant distinguishing features:

(1) the "identifying" must be performed by comparing the extracted features to the electronic representations using a "non-exhaustive neighbor search;" and

(2) the system must determine or associate an "action" based on the identified work. It is not sufficient to simply identify a match. Rather, "an action" associated with the match must be "determined" or "associated."

'179, claims 1, 13, and 25.

4. '441 patent (Ex. 1001 '441 IPR).

25. The independent claims of the '441 patent (claims 1, 13, and 25) include the following five elements for identifying a work and performing a

corresponding action:

[1] a database with (a) first data related to records, and (b) second data related to action information corresponding to the records;

[2] extracting features from a work;

[3] identifying the work by comparing the extracted features and the data related to the records using "a non-exhaustive neighbor search;"

[4] determining an action based on the identity of the electronic work; and

[5] performing the action.

'441, claims 1, 13, and 25.

26. The claimed steps are illustrated in Figure 1:

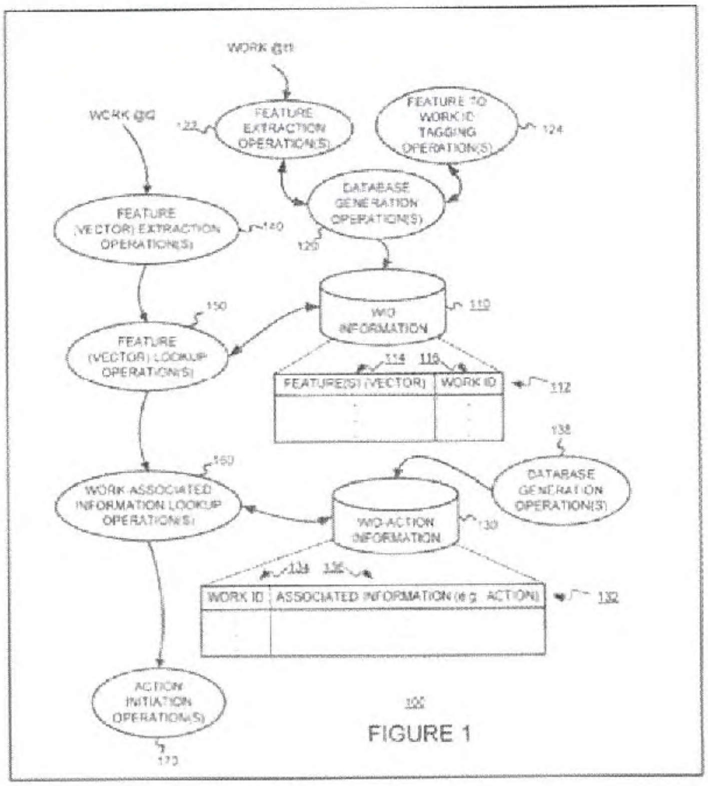


FIGURE 1

Figure 1 illustrates ("for work @t2"):

- "feature (vector) extraction operation(s)" (140) that extract(s) features from the work ('441, 6:39-41);
- "feature (vector) lookup operation(s)" (150) that identify the work by searching for a matching feature vector ('441, 6:44-48);
- "work-associated information lookup operation(s)" (160) that retrieve(s) associated information, such as an action ('441, 6:49-51); and
- "action initiation operation(s)" (170) that perform(s) some action based on the associated information ('441, 6:52-54).

27. The invention claimed in the '441 patent includes two relevant distinguishing features:

(1) the "identifying" must be performed by comparing the extracted features to the electronic representations using a "non-exhaustive neighbor search;" and

(2) the system must determine or associate an "action" based on the identified work. It is not sufficient to simply identify a match. Rather, "an action" associated with the match must be "determined" or "associated."

'441, claims 1, 13, and 25.

B. The asserted art.

28. The four IPRs address three primary references and two secondary references. I address each reference in turn, starting with the primary references and then turning to the secondary references.

1. Overview of Ghias—Ex. 1010 (addressed in the '237, '988, '179, and '441 IPRs).

29. Ghias (Patent No. 5,874,686) discloses "an apparatus [for] searching melodies." Ghias, Abstract. As illustrated in Figure 1 of Ghias, a "tune 12 is hummed by a person 18 into a microphone 20." Ghias, 2:41-42.

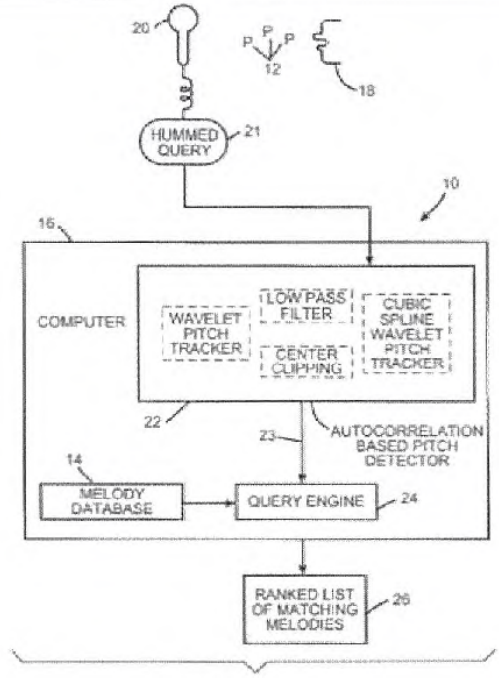


FIG.1

The data from the microphone is fed into "a pitch tracking module 22 in computer 16" which extracts "a contour representation" of the melody (23). Ghias, 2:41-50. The computer uses a "query engine 24" which "searches the melody database 14." Ghias, 2:50-52. The disclosed search can produce a ranked list of matching melodies—"ranked by how well they matched the query" (Ghias, 6:60-63) as illustrated at 26.

30. As I explain below in detail, all searches disclosed in Ghias are linear (not sub-linear) with respect to the size of the data set being searched. In addressing "the problem of approximate string matching," Ghias identifies the running times of several algorithms:

> Several Algorithms have been developed that address the problem of approximate string matching. Running times have ranged from O(mn) for the brute force algorithm to O(kn) or O(nlog(m)), where "O" means "on the order of," m is the number of pitch differences in the query, and n is the size of the string (song).

Ghias, 6:23-28. In each identified instance, the running time of the search is not sub-linear with respect to the data set. As clarified in this passage from Ghias (and as I address in detail below):

- "m is the number of pitch differences in the query" corresponding to the length of the query (highlighted in green in the passage above); and
- "n is the size of the string (song)" corresponding to the size of a record being searched (highlighted in orange in the passage above).

31. The disclosed searches may be sub-linear with respect to the length of the query being searched "m … the number of pitch differences in the query." Specifically, the referenced search with a running time of O(nlog(m)) is sublinear with respect to "m" because it is a function of log(m)). The disclosed searches, however, are never sub-linear with respect to "n…the size of the string (song)" or the size of the data set (N) (*i.e.*, the number of songs to be compared). Rather, the search time will grow linearly with each additional song to be searched and the length of the song.

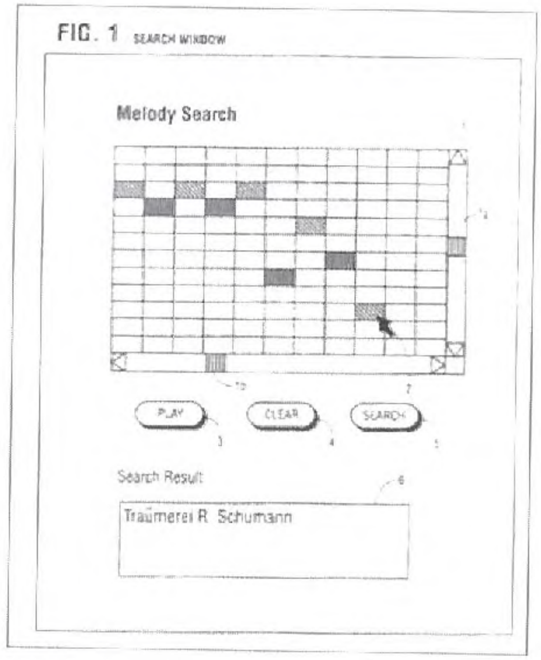
32. Also as I describe in detail below, the searches disclosed in Ghias are exhaustive rather than "nonexhaustive." The "query engine 24" compares the

work (user input 23) to "all the songs" in the melody database 14 (the library). Ghias, 5:66-6:2. After searching all possible matches, the system "output[s] a ranked list of approximately matching melodies." Ghias, 2:50-53.

33. Finally, as I describe in detail below, the searches disclosed in Ghias are not "neighbor" searches because the searches always necessarily identify the exact or closest match—they are guaranteed to identify an exact match or the closest match. Ghias does not identify any search in which an exact or the closest match is not guaranteed to be identified.

2. Overview of Iwamura—Ex. 1012 (addressed in the '237 and '988 IPRs).

34. Iwamura (Patent No. 6,188,010) discloses a "method to enable one to search for a song title when only its melody is known." Iwamura, Abstract. "A remote music database with melody information is searched for the melody entered by the user, using for example, a peak or differential matching algorithm." Iwamura, Abstract. Figure 1 illustrates "an example of a search interface" (Iwamura, 2:45-46):



35. Iwamura discloses a searching algorithm that is designed to be more efficient than alternatives by matching up peak notes from the work to be identified with the peak notes of the records in the database when comparing the notes from the work to be identified with the notes in the records. "Peak notes are also detected and marked when the data base is built." Iwamura, 6:59-60. "A fast search is performed by using a peak or differential matching algorithm." Iwamura, 12:1-2.

36. As I explain in detail below, the search disclosed in Iwamura is exhaustive rather than the claimed "non-exhaustive," "sublinear," or "approximate nearest neighbor" search. While the individual comparisons of a work and a record in the library can be more efficient using the "peak note" approach

disclosed in Iwamura (search speed can be increased), in doing so each record in the library is searched as part of the disclosed algorithm and "[t]he reference melody that gives the least difference is returned as a search result." Iwamura, 7:53-55.

37. Moreover, the Boyer-Moore algorithm referenced in Iwamura searches "word by word from the beginning of the database to the end." Iwamura, 9:51-55. As a result, as I explain in detail below, while the Boyer-Moore algorithm may be sublinear with respect to the length of the query (the work to be identified),[4] it is not sub-linear with respect to the relevant size of the dataset being searched.

4 If the query pre-processing step of the Boyer-Moore algorithm is included as part of the execution time, then the algorithm may be linear in terms of the length of the query. If the query is used repeatedly, however, the pre-processing execution time will only be incurred once. One can think of concatenating all the database strings to give as an aggregate length of n. If "m" is the query length, then the worse-case complexity is Theta(m) O(n), which is linear with respect to both the database n and the query length m.

3. Overview of Conwell—Ex. 1009 (addressed in the '179 and '441 IPRs).

38. Conwell (Patent No. 6,970,866) discloses associating media content, such as MP3 files, with identifiers and URLs. Conwell, Abstract. As illustrated in Figure 3, the identifiers (*e.g.*, "034") are associated with corresponding URLs (*e.g.*, "www.sonymusic.com/catalog/05634.html"):

034	www.sonymusic.com/catalog/05634.html
112	www.sonymusic.com/catalog/00014.html
198	www.supertracks.com/index/artists/taylor.htm
376	www.emusic.com/0555353x.pdf
597	www.cdw.com/music/featured_CDs/index.html
612	www.sonymusic.com/catalog/00231.html
850	www.polygram.com/franklin/adf_234.htm
921	www.loudeye.com/rap/1999/46755646.html

FIG. 3

39. Conwell discloses two approaches to identifying a work: (1) assigning identifiers, or (2) implicitly generating identifiers derived from the data using a hashing algorithm. Conwell, Abstract. The implicit approach (rather than explicitly assigning identifiers) is the approach relevant to the IPR Petitions. *See* Pet. ('179) at 22-23.

40. Conwell relies on hashing algorithms to extract features from the work to be identified:

> "The identifiers can be assigned, or can be implicit (*e.g.*, derived from other data in the content object, as by hashing)."

Conwell, Abstract; 1:65-67 ("some or all of the content data is processed by a hashing algorithm to yield a 128 bit identifier corresponding to that content.") As I explain in detail below, Conwell compares the hashed extracted features of the work to be identified to the features of the records in the database exclusively using an exact match comparison.

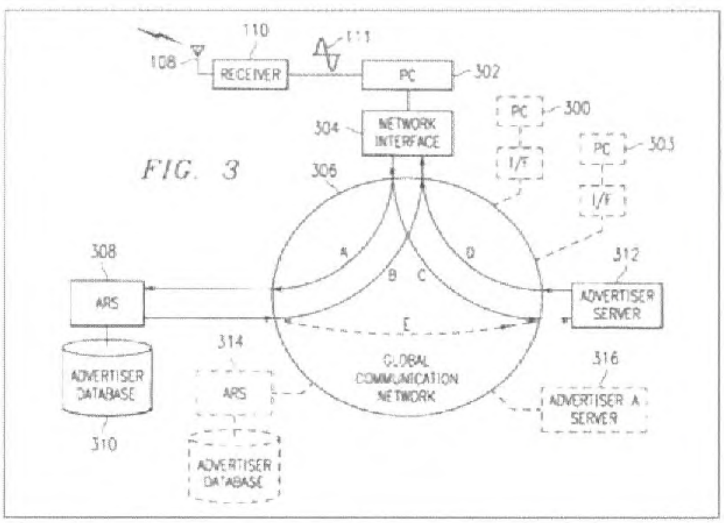
41. When implementing the search in Conwell to identify a match, comparing the hashed extracted features from a work with a record in the library using the disclosed lookup table (Conwell, 3:43-45) produces a binary result: either (1) there is an exact match; or (2) there is no exact match. *See* Conwell, Figure 3. Whether the hash of the extracted features of the work and the hashed extracted features of the record being compared are close (similar) or distant (dissimilar) is not considered, is not relevant, and cannot be determined using a nearness comparison of the hashes extracted in Conwell.

42. In addition, as I explain below in detail, while Conwell discloses using a sorted lookup table to store the hashed extracted features (*see* Conwell Figure 3, 3:43-45), Conwell does not identify any specific algorithm for performing the exact match comparison using the lookup-table and therefore does not identify either an exhaustive search or a non-exhaustive search. Either

approach could theoretically be used; however, neither approach is expressly or inherently disclosed.

4. Overview of Philyaw—Ex. 1014 (addressed in the '179 and '441 IPRs as a secondary reference).

43. Philyaw (Patent No. 6,098,106) discloses a system that uses identifying information embedded into either the sound or video portion of a broadcast signal to view corresponding information. Philyaw, Abstract; 1:66-2:8. "Figure 3 illustrates the system interactions over a global network" (Philyaw, 2:20-21):



44. Rather than using the searches claimed in the IPR patents to identify a work by comparing extracted features to a database of potential matches, the system in Philyaw embeds a "routing signal having routing information contained therein" into a broadcast program to identify what is being broadcast:

> "A program is broadcasted having embedded therein a routing signal having routing information contained therein. The routing signal is then extracted from the broadcast. Thereafter, a personal computer is controlled to allow a user to retrieve the information from a storage region at the defined location, which defined location is located with the extracted routing information, providing it at the personal computer for use by the user."

Philyaw, 2:1-9. Accordingly, Philyaw does not disclose any searching algorithm.[5]

5. Overview of Chen—Ex. 1008 (addressed in the '237 IPR as a secondary reference).

45. Chen (Patent No. 7,444,353) discloses a system for identifying music in a "music and information delivery" system. Chen, Abstract. Figure 1 "illustrates one embodiment of a music and information delivery system according to the teachings" of Chen. Chen, 4:7-9.

---

[5] I note that the '179 and '441 IPR Petitions do not assert that Philyaw discloses the claimed "non-exhaustive neighbor search" (from the '179 and '441 claims) but instead exclusively relies on Ghias for this element in Ground 2 of the '179 and '441 IPRs (the only grounds in which Philyaw is asserted). *See* Pet. ('179) 47-48, 51; Pet. ('441) 47-48; 51.

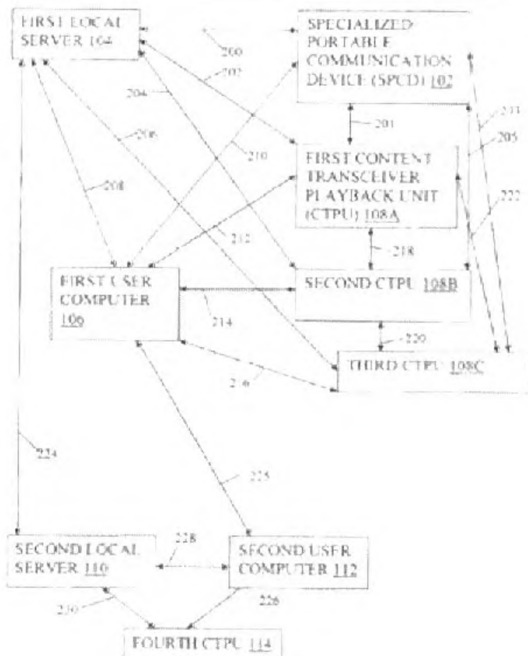


FIG. 1

A user can identify music that they can then access, for example, by downloading to a laptop or home computer. Chen, 1:58-66.

46. The system in Chen can "search a storage medium to identify and access the piece of music from the storage medium." Chen, Abstract. But Chen does not provide any details as to how any search is performed. Accordingly, Chen does not disclose a(n):

- sub-linear time search to identify a neighbor ('237, claims 1 and 5);
- approximate nearest neighbor search ('237, claims 9 and 13);
- non-exhaustive search … to identify a near neighbor ('237, claim 25); or

- sublinear approximate nearest neighbor search (claim 33).

I note that neither the Petition nor the corresponding Declaration relies on Chen for these claimed elements but instead relies on Iwamura in Ground 3 of the '237 IPR, the only ground in which Chen is at issue. *See* Pet. ('237) at 54-57.

III. General Findings.

47. Based on my analysis of:

(a) the IPR Patents (the '237, '988, '179, and '441 patents);

(b) the art asserted against the IPR Patents in the four IPRs;

(c) the IPR Petitions;

(d) Dr. Moulin's Declarations (Exs. 1004 in each IPR) and deposition testimony (Ex. 2006);

(e) the documents cited in the IPR Petitions;

(f) the IPR Decisions; and

(g) the other documents referenced in this Declaration,

I am of the opinion that the instituted claims of the IPR Patents[6] are not unpatentable based on the grounds at issue in the IPRs.

[6] I understand that the instituted claims are claims 1, 3–5, 7–9, 11–13, 15, 16, 21–27, 29, 30, 33, 34, 35, 37, and 38 of the '237 patent; claims 15–17, 21–28, 31–

48. Specifically, for the reasons and based on the analysis that I set forth below, I am of the opinion that:

'237 patent:

- Ground 1: Claims 1, 3–5, 7–9, 11–13, 15, 16, 21–25, 29, 30, 33, 37, and 38 of the '237 patent are not anticipated by Iwamura under 35 U.S.C. § 102(e);
- Ground 2: Claims 1–3, 5–7, 9–11, 13–15, and 21–24 of the '237 patent are not anticipated by Ghias under 35 U.S.C. § 102(b);
- Ground 3: Claims 26, 27, 34, and 35 of the '237 patent are not obvious over Iwamura and Chen under 35 U.S.C. § 103;

'998 patent:

- Ground 1: Claims 15–17, 21–23, 28, 31, and 51 of the '998 patent are not anticipated by Ghias under 35 U.S.C. § 102(b);
- Ground 2: Claims 22, 24–26, and 52 of the '998 patent are not obvious over Ghias under 35 U.S.C. § 103(a);

33, 51, and 52 of the '998 patent; claims 1–3, 6, 8–14, 18, 19, 21–27, 29–31, and 34–37 of the '179 patent; and claims 1–3, 6, 8–14, 18, 19, 21–27, 29, and 30 of the '441 patent.

- Ground 3: Claims 15–17, 21, 23, 27, 28, 31–33, 38, and 51 of the '998 patent are not anticipated by Iwamura under 35 U.S.C. § 102(e);

'179 patent:

- Ground 1: Claims 1–3, 6, 8–14, 19, 21–26, 30, 31, and 34–37 of the '179 patent are not anticipated by Conwell under 35 U.S.C. § 102(e);
- Ground 2: Claims 1–3, 8, 10–14, 18, 19, 21–27, 29, 31, and 34–37 of the '179 patent are not obvious over Ghias and Philyaw under 35 U.S.C. § 103;

'441 patent:

- Ground 1: Claims 1–3, 6, 8–14, 19, 21–26, and 30 of the '441 patent are not anticipated by Conwell under 35 U.S.C. § 102(e); and
- Ground 2: Claims 1–3, 8, 10–14, 18, 19, 21–27, 29, and 30 of the '441 patent are not obvious over Ghias and Philyaw under 35 U.S.C. § 103(a).

49. The information below presents the basis for my opinions that the challenged claims of the IPR Patents are not unpatentable based on the grounds at issue in the IPRs.

IV. General concerns with the IPR Petitions and Dr. Moulin's Declarations (Exs. 1004 in each IPR).

50. It is my understanding that a Petition must set forth how a challenged claim is to be construed. In addition, it is my understanding that a Petition and corresponding declaration must then map the properly construed claim language to the teachings of the asserted art. Based on my review of the Petitions and corresponding Declarations, one skilled in the art would understand that the Petitions and corresponding Declarations fail to comply with these requirements.

51. First, the Petitions (and in particular, the '237 Petition) and corresponding Declarations fail to identify any construction of the phrase "approximate nearest neighbor search." *See* Pet. ('237) at 1-53. Claims 9 and 13 of the '237 patent include an "approximate nearest neighbor search." *See* '237 patent claims 9 and 13. Because the Petition and Declarant do not identify any construction of "approximate nearest neighbor search," they fail to map the properly construed language to the teaching of the prior art (*i.e.*, Iwamura, Ghias, and Chen).

52. Second, although Petitioner and Dr. Moulin identified constructions for certain terms,[7] neither the Petition nor the corresponding Declaration maps the construed claim language to the teachings of the asserted art.

53. Moreover, one skilled in the art would understand that Petitioner's failure to map the properly construed claim language to the teachings of the prior art results in critical mistakes in the IPR Petitions and Declarations, as I illustrate using the following two examples:

54. Example 1: In his Declaration with respect to the '237 patent, Petitioner's Declarant, Dr. Moulin, confirmed (consistent with my understanding and the understanding of one of ordinary skill in the art) that a "sublinear search" is a search that has a sublinear relationship to the database size:

> 53. I understand and agree with Petitioner's position that the term "sublinear search" means "a search whose execution time has a sublinear relationship to database size." For instance, a linear search of a 200-item database would take twice as long as a linear search of a 100-item database. By contrast, a sublinear search of a 200-item database would take less than twice as long as a sublinear search of a 100-item database, perhaps, for instance, 1.5 times as long.

[7] *See, e.g.*, Moulin Decl. ('237) ¶43 (addressing non-exhaustive search); Moulin Decl. ('237) ¶48 (addressing identify a neighbor / identify a near neighbor / nearest neighbor search); Moulin Decl. ('237) ¶53 (addressing sub-linear).

Moulin Decl. ('237) ¶53. In that same Declaration, Dr. Moulin also asserted that Ghias discloses a search that is sublinear based on the disclosure in Ghias of algorithms that have running times of "O(kn) or O(nlog(m)):"

> In particular, Ghias discloses searches whose execution times are proportional to the logarithm of the size of the data set (*id.* at 6:24-28 ("O(kn) or O(nlog(m)")), which, as explained above in Section V(D), are sublinear (Ex. 1001 at 8:54-63).

Moulin Decl. ('237) ¶123.

55. In asserting that Ghias discloses a search that is sublinear, Dr. Moulin did not apply the construction of a sublinear search which requires that the search be sublinear with respect to the "size of the database" and not the number of pitch differences in the query which is the length of the query. Moulin Decl. ('237) ¶53. However, as stated in Ghias (and confirmed by Dr. Moulin at his deposition, *see* the deposition citations that I reference below), (1) "m" refers to the number of pitch differences in the query while "n" refers to the size of the string (song), and (2) each search algorithm identified in Ghias is linear with respect to the size of the data set. Accordingly, one skilled in the art would understand that Ghias does not disclose a sublinear search under Petitioner's (and Dr. Moulin's) own construction of sublinear.

56. Dr. Moulin's deposition transcript addressing the paragraph from his Declaration that I presented above (Moulin Decl. ('237) ¶123) demonstrates to one skilled in the art the problems that result from the Petitioner failing to apply Dr. Moulin's own construction to the art:

Q And by "logarithm of the dataset," you
mean logarithm of the size of the query, not the
dataset to be searched; is that true?
A I believe we discussed that earlier.
Okay. Should have -- here it is the log, indeed, of
the query dataset. So log(m).
Q So it should -- this should be
interpreted -- when you write "dataset," you're --
what you mean is the query dataset, not the dataset
to be searched?
A That's correct. So the logarithm of the
dataset is log(m), which is written below. Yes.
Q Which is the query dataset?
A That's right.

Moulin Depo. 154:14-155:2

Q. [illegible]

MR. BLANCHA: [illegible]

THE WITNESS: [illegible] more precise way. [illegible] it should have been the log of the query dataset. The word "query" [illegible] added there.

BY MR. DOVEL:

Q. Why didn't you write that in [illegible]

MR. BLANCHA: Again, I'm [illegible]

THE WITNESS: I've said it. So the word "query" should be included before "dataset" in the [illegible] line 4.

BY MR. DOVEL:

Q. Why do you object to writing it in there?

A. Why should I write it? I just said it.

Moulin Depo. 155:12-156:6.

A. Because it's very clear. I'm saying the word "query" should have been incorporated before "dataset."

Moulin Depo. 156:22-157:3.

> Well, one possible reason you wanted to give the Board this information is so that they would misread it and be misled.
> That's a possibility; right?
> A It's not at all the -- the reason. So it alone -- there are four documents. Some, they are, like, 90 pages each. Some of the words could have been better chosen. In particular, the word "query" should have been there. I have acknowledged that.
> Q But assume --
> A Again, I have acknowledged that this was not written the best way.

Moulin Depo. 157:6-18.

57. Example 2: As I noted above, in his Declaration with respect to the '237 Patent, Dr. Moulin confirmed that a sublinear search is a search that has a sublinear relationship to the database size:

> 53. I understand and agree with Petitioner's position that the term "sublinear search" means "a search whose execution time has a sublinear relationship to database size." For instance, a linear search of a 200-item database

Moulin Decl. ('237) ¶53. Just a few pages later in that same Declaration, Dr. Moulin asserted:

> 72. It is my opinion that Iwamura further teaches how this search can be sublinear. For example, Iwamura discloses that different "search algorithms may be applied to perform melody searches," (*id.* at 10:2-3), such as the "Boyer-Moore algorithm." (*id.* at 9:63). "On the average the [Boyer-Moore] algorithm has a sublinear behaviour." Ex. 1017 at 1.

Moulin Decl. ('237) ¶72. In asserting that Iwamura discloses a search that is sublinear, Dr. Moulin did not apply his construction of a sublinear to the referenced Boyer-Moore algorithm.

58. As I explain in detail below and confirmed by Dr. Moulin, one skilled in the art would understand that the Boyer-Moore algorithm disclosed in Ghias does not disclose sublinear behaviors with respect to the size of the data set but only with respect to the query (or pattern)[8] to be matched. Again, Dr. Moulin's deposition transcript addressing the paragraph from his Declaration that I presented above (Moulin Decl. ('237) ¶72) demonstrates to one skilled in the art the problems that result from Petitioner failing to apply Dr. Moulin's own construction to the art:

---

[8] Dr. Moulin testified that "query," "pattern," and "probe" are "all synonymous in this context." Moulin Depo. 21:24-22:1. I agree with Dr. Moulin's testimony.

Q When you wrote this, were you trying to convey that searching using the Boyer-Moore algorithm would be sublinear with respect to the size of the dataset being searched?
A No. I did not -- no.

Moulin Depo. 74:20-24.

Q When we read paragraph 72, are you conveying to the reader that the Boyer-Moore algorithm is sublinear with respect to the size of the dataset being searched?
A No. All I do is quote a part of a paper that shows why that algorithm is much faster than brute force. That's all I'm doing. You're inferring things I'm not saying or writing.

Moulin Depo. 69:9-16.

Q When we read paragraph 72, are you conveying to the reader that the Boyer-Moore algorithm is sublinear with respect to the size of the dataset being searched?
A No. All I do is quote a part of a paper that shows why that algorithm is much faster than brute force. That's all I'm doing. You're inferring things I'm not saying or writing.

Moulin Depo. 66:9-18.

> Q. When you wrote this sentence here on page [illegible], did you think that someone at the Board looking at this might think that you meant that -- the Boyer-Moore algorithm was sublinear as used in the patent claim?
>
> A. I didn't think of it that way, no.

Moulin Depo. 75:23-76:3.

> Q. And when you wrote this, were you trying to convey to the Board that the Boyer-Moore algorithm is one that, if you use it, it's sublinear with respect to the size of the database?
>
> A. No. No.

Moulin Depo. 67:17-21.

V. Claim Constructions.

59. In construing the claims of the IPR Patents, I understand that in this proceeding, the claims are interpreted using the broadest reasonable construction in light of the patent in which they appear. I also understand that there is a presumption that a claim term carries its ordinary and customary meaning to one of ordinary skill in the art at the time of the invention. In conducting my analysis of the claim elements below, I apply this understanding.

60. It is also my understanding that, for purposes of evaluating whether the IPR Petitions satisfied an initial threshold, the Board identified certain claim

constructions in its Decisions. It is my understanding that the preliminary constructions identified by the Board are:

"sub-linear"	"a search whose execution time scales with a less than linear relationship to the size of the data set to be searched" Decision ('237) at 7.
"non-exhaustive search" / "non-exhaustive…search"	"a search that locates a match without a comparison of all possible matches" Decision ('237) at 7; Decision ('988) at 7; Decision ('179) at 7; and Decision ('441) at 7.
"neighbor search" / "identifying a neighbor"	"identifying a close, but not necessarily exact or closest, match" Decision ('988) at 7; Decision ('179) at 8; and Decision ('441) at 7.
"neighbor" / "near neighbor"	"a close, but not necessarily exact or closest, match" Decision ('237) at 8.
"approximate nearest neighbor search"	"identifying a close match that is not necessarily the closest match" Decision ('237) at 9.

I address each construction in turn.

A. sub-linear ('237 patent).

61. The Board's preliminary construction of "sub-linear time search" is "a search whose execution time scales with a less than linear relationship to the size of the data set to be searched." Decision ('237) at 7. While the Board's preliminary construction is a correct construction of "sub-linear time search," there are apparently two possible interpretations of the Board's construction:

Interpretation 1: consistent with the meaning of the phrase "size of the data set," the "size of the data set" refers to the number of records in the data set being searched (such that the relevant linear relationship is with respect to the size of the data set, *i.e.*, the number of records in the data set), or

Interpretation 2: the "size of the data set" refers to the length of an individual record in the database (such that the linear relationship is with respect to the length of an individual record to be searched rather than the size of the data set).

62. As I explain below, the first interpretation is correct. Also, as I demonstrate below, the asserted art for this element (Iwamura and Ghias) does not disclose a sub-linear search under either interpretation. The (1) words actually used in the Board's construction, the (2) specification of the '237 patent, and (3) the Petitioner's Declarant, confirm that the proper interpretation of a sub-linear search, in the context of the '237 patent, is a search that is sublinear with the size

of the data set where the data set is the number of records in the database, not the length of an individual record in the data set.

1. the words used in the construction: "size of the dataset"

63. A data set, in the context of the '237, is a database (*i.e.*, set of records), not an individual record in a dataset or database. The Board incorporated the phrase "size of the dataset" in its preliminary construction based on the fact that "database" and "dataset" are "largely consistent" such that "database size" and "size of the data set" are "largely consistent." Decision ('237) at 7. As Petitioner's Declarant confirmed, "dataset" and "database" are the same in the context of the IPR Patents:

> Q When you say "database," is that the same thing as "dataset" in this context?
>
> A Yes.

Moulin Depo. 22:14-16.

> Q When you wrote "sublinear" here, you're referring to sublinear with respect to the size of the dataset being searched?
>
> A Yes. Okay. Just to be clear, the dataset is the whole database, okay, the database of songs.

Moulin Depo. 110:11-15. Consistent with the understanding of one skilled in the art, I agree with Petitioner's Declarant—that "database" and "dataset" are the same in the context of the IPR Patents.

64. A database comprises all records in the database: a single record in a database is not a database. My understanding is consistent with the testimony of Petitioner's Declarant:

> So just to make it clear, N is not the size of the database; N is the size of a single song.

Moulin Depo. 89:4-13.

65. Moreover, I observed that the origin of the phrase "data set" in the Board's construction of sub-liner is the Patent Owner's Preliminary Response for the '237 patent in which Patent Owner specifically clarified that the "'size of the data set' is the number of potential matches in the data set (*i.e.*, the 'number of entries in the search database.'" Preliminary Response ('237) at 9-10 (*quoting* Moulin Decl. ('237) ¶54). I note that neither the Patent Owner (in its Preliminary Response) nor the Petitioner (in its Petition or Declaration) stated or suggested that sublinear should be based on the length of an individual query in the dataset.[9]

---

[9] This is the natural growth mechanism for such a problem. For example, if there are X songs in a database and X additional songs are added, in one dimension the database has doubled (2X). In general, there is no reason to think that the distribution of record sizes changes as the size of the database (number of record in the database) increases. If each record is a fixed length vector, the database size

2. '237 specification.

66. As I noted above, it is my understanding that a claim in an unexpired patent is given its broadest reasonable construction in light of the specification of the patent in which it appears. It is also my understanding that the best source for discerning the proper context of claim terms is the patent specification.

67. "Sub-linear" indicates a relationship between two quantities which is less than linear. Linearity describes "[t]he relationship existing between two quantities when a change in a second quantity is directly proportionate to a change in the first quantity." Ex. 2007 (Modern Dictionary of Electronics) at 425 (1999). "Sub-" is a prefix indicating "under" or "below."

68. The claim language identifies "time" as one of the quantities being related. In the expression "a sub-linear time search," "sub-linear <u>time</u>" is an adjective phrase modifying "search." The '237 specification identifies the number of records in the data set ("N") as the variable that is sub-linear with respect to time.

---

will double when the number of records doubles. Even if each record is a digital representation of an entire song, there is no reason to think that the new songs have a time / length statistical distribution that is significantly different from the songs that already exist in the database.

69. First, the specification identifies a problem with prior art searches—that the searches are "linear" with respect to the number of records in the data set ("N")[10]—not with respect to the length of an individual record in the database being searched:

> "If binary search was possible, then a database containing N vectors would require at most log(N) comparisons. .... In previous work, it was not uncommon to perform a linear search of all N entries, perhaps halting the search when the first match is found. On average, this will require N/2 comparisons.[11] If N is large, this search can be computationally very expensive."

'237, 8:54–63.[12]

---

[10] In the '237 specification, the variable "N" indicates the number of entries in the database being searched—"a database containing N vectors." '237, 8:54–55.

[11] N/2 is the expected average result for an exhaustive or linear search of a dataset with one match.

[12] A search algorithm that requires N/2 comparisons has a running time that scales linearly with respect to N. As N increases by one, the search's running time increases by, on average, one half of the running time of a single comparison. Thus, as N increases, the running time of a search requiring N/2 comparisons increases by a directly proportionate amount, *i.e.*, linearly.

> "Consider a situation in which one out of 100,000 possible commercials is to be identified. Each 30-second commercial consists of 900 video frames. If all 900 frames are stored in the database, then N=90,000,000. Even if only every 10th video frame is stored in the database, its size is still nine million. While databases of this size are now common, they rely of [sic] efficient search to access entries, i.e., they do not perform a linear search. A binary search of a 90,000,000-item database requires less than 20 comparisons. In contrast, a linear search will require an average of 45,000,000!"

'237, 21:14–23.

70. In both of the instances from the '237 specification that I presented above, the '237 specification describes prior art search techniques as "linear" with respect to "N"—the number of records in the database being searched—not with respect to the length of an individual record in the database.

71. Second, the '237 specification identifies search techniques that achieve a sub-linear search time with respect to the number of records in the database being searched (not with respect to the length of an individual record being searched):

> "Other forms of matching include those based on clustering, kd-trees, vantage point trees and excluded middle vantage point forests are possible and will be discussed in more detail later. . . . Thus, for example, a sub-linear search time can be achieved."

'237, 8:64–9:7.

> "A number of possible data structures are applicable including kd-trees and vantage point trees. These data structures and associated search algorithms organize a N-point dataset (N=90,000,000 in out previous example) so that sub-linear time searches can be performed on average."

'237, 21:56–60. Clustering, kd-trees, vantage point trees and excluded middle vantage point all achieve sub-linear behavior by reducing the number of records being search, *e.g.*, by discarding clusters (buckets) of potential matches, not by reducing the length of an individual record being searched. These methods prune parts of the search space (*i.e.*, data records to be searched) and this is why they are efficient.[13]

[13] The Yianilos paper incorporated by reference into the '237 patent ('237, 8:65-9:6) explains: "We introduce the idea of aggressive pruning and give a family of practical algorithms, an idealized analysis, and describe experiments. Our main result is that search complexity measured in terms of d-dimensional inner product operations, is i) strongly sublinear with respect to the data set size n for moderate R, ii) asymptotically, and as a practical matter, independent of dimension." *See* Ex. 2010 (P. N. Yianilos) at 1.

72. Again, in all instances, the '237 specification describes techniques as "linear" with respect to "N"—the number of records in the database being searched—not with respect to the length of an individual record in the database.

3. Petitioner's Declarant.

73. According to Petitioner (consistent with the understanding of one skilled in the art), a "sublinear" search is "a search whose execution time has a sublinear relationship to database size"—where the database size is the number of records in the database, not the length of an individual record in the data set.

74. A "database" consists of all records in the data set; a "database" is not one individual record in the database—an individual record is not a "database." Dr. Moulin confirmed that sublinear in the context of the '237 patent is based on the size of database (a "concept that's common in [his] field" Moulin 8:10-14), not the length of an individual record in the database:

> 53. I understand and agree with Petitioner's position that the term "sublinear search" means "a search whose execution time has a sublinear relationship to database size." For instance, a linear search of a 200-item database would take twice as long as a linear search of a 100-item database. By contrast, a sublinear search of a 200-item database would take less than twice as long as a sublinear search of a 100-item database, perhaps, for instance, 1.5 times as long.

Moulin Decl. ('237) ¶53. As Petitioner's Declarant—Dr. Moulin—confirmed:

Q And what it refers to, to be more precise,
is that if we have a sublinear search, that means
that the search time is going to be -- at less than a
linear relationship compared to the size of the
database as we increase the size of the database?
A That is correct.

Moulin Depo. 13:24-14:4.

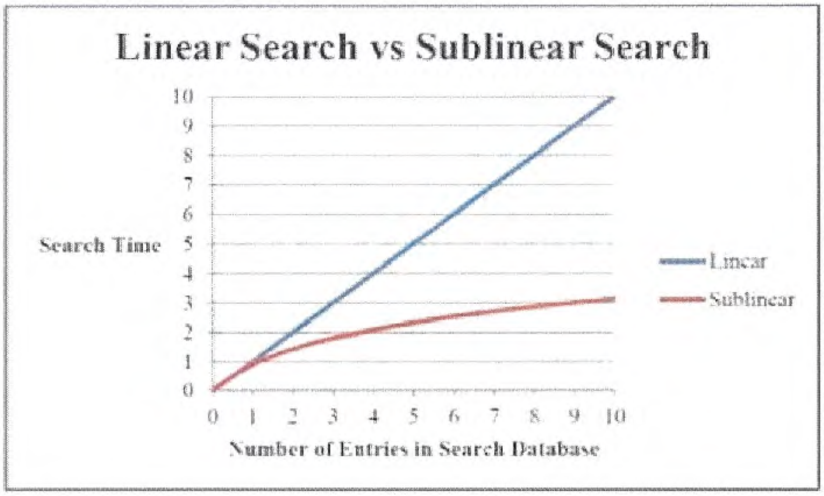
Q Well, as you understood the term
"sublinear" as it's used in the patent, it's about
the execution time as we increase the size of the
database; is that right?
A In the patent, yes.
Q Yes.
A Yes.

Moulin Depo. 103:16-22; 16:4-12; 24:1-12.

75. The Petition and corresponding Declaration interpreted "database size" as the number of records in the database (not the length of an individual record to be searched):

> "For instance, a linear search of a 200-item database would take twice as long as a linear search of a 100-item database. By contrast, a sublinear search of a 200-item database would take less than twice as long as a sublinear search of a 100-item database, perhaps, for instance, 1.5 times as long."

Pet. ('237) at 6; Moulin Decl. ('237) ¶ 53.



Moulin Decl. ('237) ¶54 (showing that the execution time of a sub-linear time search increases with a less than linear relationship to the "number of entries in the search database").

> "[I]t is my opinion that a search whose execution time is proportional to the logarithm of the size of the data set (e.g., a search with execution time proportional to A log (BN), where A and B are constants, and N is the number of entries in the database) is sublinear."

Moulin Decl. ('237) ¶56.

76. One skilled in the art would understand that each explanation of "sub-linear" in the Petition and Declaration demonstrates that the sub-linearity of a search depends on the number of entries in a database, not on other factors, such as the length of an individual record in the database. Nowhere does Petitioner or the

Declarant suggest that the relevant sublinear search is with respect to the length of an individual record to be searched.[14]

77. I note that in its Decision, the Board did not present any analysis or reasoning for interpreting sub-linear relative to the length of an individual record being searched rather than the number of records in the dataset. *See* Pet. ('237) at 7. And the example presented by the Board in its Decision equates "data set" with the number of records in the database ("N"), not the length of an individual record:

> One example of such a sub-linear search would be a search with an execution time proportional to the logarithm of the size of the data set ("N"), where a doubling of N would lead to an execution time proportional to log(2N).

14 In Petitioner's analysis, the length of individual records is not a factor in evaluating whether an algorithm is a sub-linear time search or a linear time search. If the length of individual records were a factor in evaluating the sub-linearity of a search, then such lengths would have to be addressed in determining whether a given algorithm is sub-linear and Petitioner's examples would have to account for those lengths. The Petition does not rely upon any variable other than the number of items in the database to be searched to determine whether a search is sub-linear. The Petition determines whether a search has a sub-linear running time exclusively with respect to the number of entries in the database to be searched.

Decision ('237) at 7.

78. Importantly, one skilled in the art would understand that under any possible interpretation of "sublinear" in the context of the '237 patent (as well as the general context of search algorithms), a search algorithm is sublinear only with respect to the size of the dataset (the size of the reference database), not the size of the query (pattern) of the work to be identified using the search. As Petitioner's expert confirmed, consistent with my understanding, whether a prior art search "scales based upon the size of the query or pattern" would not "be accurately assessing the '237 patent claims." Moulin Depo. 25:22-26:8.

Is it the case that for a search to be
sublinear as it's used in the '237 patent, it's not
enough for it to have execution time that is
sublinear in relationship to the size of the
pattern; it must also be sublinear in relationship
to the size of the database?
A When I read "sublinear" in, say, Claim 5
of the patent, as we just did, I understand
sublinear to mean in relation with the size of the
database. It does not say anything about in
relation with the size of the query.

Moulin Depo. 26:11-21.

Q. Would it be wrong, then, to assess whether a search is sublinear by applying a definition that said, "It's sublinear if this execution time has a sublinear relationship to the size of the query or the pattern"?

A It would not be very relevant. Again, mathematically, it can be done. Everything can be done. But it would not be relevant, from an engineering viewpoint, for an application like this.

Moulin Depo. 25:4-12.

Let's assume that the Patent Trial and Appeal Board was presented with prior art that presented a search that was linear with respect to the size of the database, but it was sublinear with respect to the size of the pattern.

Would that prior art demonstrate a sublinear search as it's used in Claim 25?

A People say it's a linear search, again, because it's in relation with the size of the database. And as you just said, that complexity is still linear; so people would say it is a linear search.

Q It is not a sublinear search?

A It's only sublinear in terms of the size of the query, which is generally not the parameter of -- the relevant parameter.

Moulin Depo. 26:25-27:15.

Q Is it the case that if we had a piece of prior art that was linear with respect to the size of the database but sublinear with respect to the size of the query or the pattern, that that prior art would not teach a sublinear search as it's used in Claim 25?

A Again, if one understands sublinear to be in terms in relation to the size of the database, that would be a -- a linear search.

Moulin Depo. 27:16-24.

Let's assume we've got a piece of prior art that scales at a sublinear relationship with the size of the pattern or query but it scales at a linear relationship with the size of the database that's being searched.

Would that prior art demonstrate or disclose a sublinear search as it's used in the claims of the '237 patent?

A No. Again, because my understanding is the claims of the '237 patent, whenever there's mention of "sublinear," it means in terms of the database size. It does not say it explicitly; it's my inference based on my knowledge and my expertise.

Moulin Depo. 28:4-16. I agree with the Petitioner's Declarant—that "sublinear" in the context of the IPR Patents is with respect to the size of the database, not the size of the query or pattern.

B. non-exhaustive search ('237, '988, '179, and '441 patents).

1. The Board's preliminary construction of "non-exhaustive search" is consistent with the understanding of one of ordinary skill in the art: "a search that locates a match without a comparison of all possible matches."

79. A "non-exhaustive" search is a search that uses an algorithm designed to locate a match without comparing the work to all records in the database. A "non-exhaustive" search uses an intelligent algorithm to narrow the database to only a subset of potential matches. *See e.g.* Moulin Decl. ('988) ¶12 ("algorithms that increased efficiency by intelligently searching only a subset of potential matches (*i.e.*, 'non-exhaustive' algorithms)"); Pet. ('237) at 3 ("search algorithms that increased efficiency by intelligently searching only a subset of potential matches (*i.e.*, 'non-exhaustive' algorithms)").

80. For example, if there are 100 records in a database, a non-exhaustive search could use an intelligent algorithm to exclude 75 records from the search such that only 25 would be searched during the comparison process. As the specifications of the IPR Patents observe, these non-exhaustive "forms of matching include[ing] those based on clustering, kd-trees, vantage point trees and excluded middle vantage point forests" do not systematically compare the work to be identified to each record. *See e.g.*, '179, 9:14-17. Each of these examples uses an intelligent algorithm to narrow the database to only a subset of potential matches.

81. A "non-exhaustive" search can be contrasted with an "exhaustive" search. An exhaustive search systematically checks whether each potential match matches the work to be identified until a match is found, "perhaps halting the search when the first match is found." '237, 8:59-61. An exhaustive search is "a very general problem-solving technique that consists of systematically enumerating all possible candidates for the solution and checking whether each candidate satisfies the problem's statement." Ex. 2001 (the "solution" here refers to a record and not a section within that record).

82. If there are 100 records in the database, an "exhaustive" search does not narrow the potential matches but instead systematically compares the work with each record to determine a match (if there is one). Systematically comparing the work to be identified with each potential match until a match is identified rather than using intelligence to narrow the search candidates is also referred to as using "brute force." Moulin Decl. ('988) ¶44 ("a brute force search conducts a comparison of every item in a search database"); Ex. 2001.

2. The Board properly rejected Petitioner's assertion that a "non-exhaustive search" should be construed as "a search that locates a match without conducting a brute force comparison of all possible matches, and all data within all possible matches."

83. The "all data" clause (that I underlined above) in Petitioner's proposed construction (Pet. ('237) at 5; Decision ('237) at 5-7) would improperly include as a "non-exhaustive" search any search that did not compare "all data" in each record, even if the search were a brute force comparison of each record in the database. As an illustrative example, assume the work to be identified "ABC" is compared with all records in a library, including record "DEF." When comparing "ABC" with "DEF," the algorithm determines that there is no match between "ABC" and "DEF" after just comparing the first letter of the work "A" with the first letter of the record "D." If the algorithm does not unnecessarily compare the second and third letters, then according to Petitioner, the search is not "exhaustive" even though every record is compared.

84. Petitioner's Declarant states that a non-exhaustive search is any search that is not a brute force search, and a "'brute force' search, in turn, is a search wherein a query is compared to every single portion of every single item in a database." Moulin Decl. ('237) ¶43. Petitioner's Declarant, however, provides no

analysis or support for this conclusory assertion which, I understand, is insufficient to satisfy Petitioner's burden in these IPR proceedings.

85. One skilled in the art would understand that the "all data" clause is improper because it is:

- inconsistent with how the non-exhaustive search concept is used in the IPR Patents which describes a linear exhaustive search as one where the search compares the work to all "N entries," not all data within all "N entries" (*see e.g.*, '179, 21:10-42; 8:59-9:54); and
- not part of the ordinary meaning of "non-exhaustive search" (*see* Ex. 2001).

86. Moreover, objective sources confirm my understanding that an "exhaustive" or "brute-force" search systematically compares the work with each record in a database, not all data within each record, for example:

> "In computer science, brute-force search or <u>exhaustive search</u>, also known as generate and test, is a very general problem-solving technique that consists of systematically enumerating all possible candidates for the solution and <u>checking whether each candidate satisfies the problem's statement</u>."

Ex. 2001—each "candidate" is checked, not "all data" within each candidate.

87. Petitioner's own Declarant twice confirmed my understanding—that a "non-exhaustive" search searches a subset of "potential matches," not a subset of "all data within all potential matches":

(1) "Because neighbor searching is computationally intensive, content recognition schemes typically employed search algorithms that increased efficiency by intelligently searching only a subset of potential matches (*i.e.*, 'non-exhaustive' algorithms)." Moulin Decl. ('237) ¶12;

(2) "to maximize search efficiency, persons skilled in the art routinely employed more efficient searches that did not conduct a comparison of every single item in a database, sometimes referred to as non-exhaustive searches." Moulin Decl. ('237) ¶43.

88. For the reasons that I presented above, one skilled in the art would understand that the Board properly rejected Petitioner's "all data" clause. Decision ('237) at 6.

C. neighbor search / identifying a neighbor / neighbor / near neighbor ('237, '988, '179, and '441 patents).

89. One skilled in the art would understand that the Board properly construed a "neighbor search" and "identifying a neighbor" as "identifying a close, but not necessarily exact or closest, match" and "neighbor" and "near neighbor" as

"a close, but not necessarily exact or closest, match." Decision ('237) at 8; Decision ('988) at 7-8; Decision ('179) at 8; Decision ('441) at 7.

90. Petitioner and its Declarant agree with the Board's construction of "neighbor search." *See e.g.*, Petition ('179) at 6 ("The term 'neighbor search' … should be construed to mean 'identifying a close, but not necessarily exact, match.'"); Moulin Decl. ('179) ¶45 ("'neighbor search' means 'identifying a close, but not necessarily exact, match.'"); Moulin Depo. 250:2-5.

91. One skilled in the art would understand that there are two relevant features of a neighbor search under this construction:

92. Feature 1: If a search necessarily identifies an exact or the closest match (*i.e.*, the search is designed to guarantee that an exact or the closest march is identified each time the search is performed), it is not a neighbor or near neighbor search because it is not a search that "identif[ies] a close, but not necessarily exact or closest, match." Rather, such a search necessarily identifies an exact or the closest match.

93. Feature 2: If a search that necessarily identifies an exact or the closest match (*e.g.*, Match 1) but also identifies other matches that, by definition, are not the closest match (Match 2, Match 3, Match 4), the search still necessarily identifies an exact or the closest match (Match 1) and therefore cannot be the claimed neighbor or near neighbor search.

D. approximate nearest neighbor search ('237 patent).

94. As I noted above, the Petitioner did not identify a construction of "approximate nearest neighbor search."

95. The Board preliminary determined that an "approximate nearest neighbor search" is a search "identifying a close match that is not necessarily the closest match." Decision ('237) at 9. One skilled in the art would understand that this construction is correct, but incomplete, as demonstrated by the '237 specification. The '237 specification states that the claimed "approximate nearest neighbor search" is [1] a sub-linear neighbor search that [2] does not always find the closest point to the query—*i.e.*, does not always find the closest match:

> "[1] One example of a sub-linear time search is an approximate nearest neighbor search. [2] A nearest neighbor search always finds the closest point to the query. An approximate nearest neighbor search does not always find the closest point to the query. For example, it might do so with some probability, or it might provide any point within some small distance of the closest point."

'237, 9:12-19.

96. The first feature—that a "approximate nearest neighbor search" is a sub-linear time search—is not reflected in the Board's preliminary construction and, as demonstrated below, should be included in the construction. The second

feature of the claimed "approximate nearest neighbor search" is reflected in the Board's preliminary construction—"identifying a close match that is not necessarily the closest match." I address these two features in reversed order.

1. "identifying a close match that is not necessarily the closest match"

97. This feature of "approximate nearest neighbor search" was properly adopted by the Board. A search that is guaranteed to return the actual closet match is not an "approximate nearest neighbor search." The '237 specification states that an "approximate nearest neighbor search does not always find the closest point to the query." '237, 9:15–16. Accordingly, a search that "always finds" (*i.e.*, is guaranteed to find) the closest match is not an "approximate nearest neighbor search" while a search that is not guaranteed to find the closest match can be an "approximate nearest neighbor search" if it identifies a close match. *See* Pet. ('237) at 19 (stating that a reference discloses an "approximate nearest neighbor search" because the search "identifies a neighbor, but not necessarily the nearest neighbor.").

98. This understanding of "approximate nearest neighbor search" is consistent with the ordinary meaning of the phrase

> "Approximate nearest neighbor In some applications it may be acceptable to retrieve a 'good guess' of the nearest neighbor. In those

> cases, we can use an algorithm which doesn't guarantee to return the actual nearest neighbor in every case, in return for improved speed or memory savings."

Ex. 2008 (http://en.wikipedia.org/wiki/Nearest_neighbor_search#Approximate_nearest_neighbor.) at 5.

99. Similar to the neighbor and near neighbor searches addressed above, one skilled in the art would understand that a search that necessarily identifies both: (1) an exact match or the closest match, and, in addition, (2) "a close match that is not necessarily the closest match" is not an "approximate nearest neighbor search" because it is always guaranteed to identify the closest match.

2. "sublinear"

100. It is my understanding that an inventor may act as his or her own lexicographer in defining terms used in a patent's claims. One skilled in the art would understand that the '237 patent defines "approximate nearest neighbor search" as a type of sub-linear search.

101. Title: In the title of the '237 patent, the patentee identified an "approximate nearest neighbor search" as a type of sub-linear search: "Identifying works, using a sub-linear time search, such as an approximate nearest neighbor search, for initiating a work-based action, such as an action on the internet." '237, Title.

102. Abstract: It is my understanding that the abstract of a patent may be used to determine the scope of the invention. In its Abstract, the '237 patent also describes an "approximate nearest neighbor search" as a "sub-linear time search": "determining an identification of the media work . . . using a sub-linear time search, such as an approximate nearest neighbor search for example." '237, Abstract.

103. Specification: In describing methods for carrying out a sub-linear search of the reference data set, the '237 specification also describes an "approximate nearest neighbor search" as a type of sub-linear search: "One example of a sub-linear time search is an approximate nearest neighbor search." '237, 9:12–14.

104. In its preliminary construction, the Board did not include the sublinear feature of the claimed "approximate nearest neighbor search" based on what appears to be faulty logic. The Board preliminarily found:

> We largely agree with Patent Owner's construction, but note that the Specification refers to "[o]ne example of a sub-linear time search is an approximate nearest neighbor search" (Ex. 1001, 9:12–14), such that we are not persuaded that an "approximate nearest neighbor search," must be a sub-linear search, as that term has been construed above. As such, we are persuaded that the proper construction of "approximate nearest neighbor search" is "identifying a close match that is not necessarily the closest match."

Decision ('237) at 9. The logic underlying the Board's reasoning appears to be as follows: If A is "one example" of B, A is not always B. In my opinion, this logic is faulty.

105. If A is "one example" of B, A is always B even though there may be examples other than A that fall within the scope of B. If A is "one example" of B, the scope of B is not limited to just A (*i.e.*, the scope of B can include C, D, and E) but A is always B. For example, a poodle is "one example" of a dog; a poodle is always a dog (there is no scenario where a poodle is not a dog) but there are other examples that fall within the scope of dog beyond poodles, *i.e.*, terriers, Dalmatians, etc. Just like a "poodle" being "one example" of a dog must be a dog (*e.g.*, a dog bred with a curly coat that is usually clipped …) an "approximate nearest neighbor search" being "one example" of a "sublinear search that ….."

must be a sublinear search (*i.e.*, a "sublinear search identifying a close match that is not necessarily the closest match.")

VI. '237 patent.

106. I understand that the Board instituted the '237 IPR based on three Grounds:

- Ground 1: Claims 1, 3–5, 7–9, 11–13, 15, 16, 21–25, 29, 30, 33, 37, and 38 as unpatentable under 35 U.S.C. § 102(e) as anticipated by Iwamura;
- Ground 2: Claims 1–3, 5–7, 9–11, 13–15, and 21–24 as unpatentable under 35 U.S.C. § 102(b) as anticipated by Ghias; and
- Ground 3: Claims 26, 27, 34, and 35 as unpatentable under 35 U.S.C. § 103 as obvious over Iwamura and Chen.

Decision ('237) at 21-22. I address each Ground in turn.

A. '237 Ground 1: The instituted claims of the '237 patent are not anticipated by Iwamura.

107. The Board instituted Ground 1 based on the following: Claims 1, 3–5, 7–9, 11–13, 15, 16, 21–25, 29, 30, 33, 37, and 38 as unpatentable under 35 U.S.C. § 102(e) as anticipated by Iwamura. Decision ('237) at 21 (I underlined the

independent claims). Ground 1 fails because Iwamura does not disclose the following key elements from each instituted independent claim:

- sub-linear time search (claims 1, 5);
- approximate nearest neighbor (claims 9 and 13);
- nonexhaustive search … to identify a near neighbor (claim 25); and
- sublinear approximate nearest neighbor search (claim 33).

I address each in turn.

1. sub-linear time search (claims elements 1(b) and 5(b.2)).

108. Claims elements 1(b) and 5(b.2) require a "sub-linear time search."

109. As I explained above, a "sub-linear time search" is "a search whose execution time scales with a less than linear relationship to the size of the data set to be searched." Decision ('237) at 7.

110. One skilled in the art would understand that Iwamura does not disclose a "sub-linear time search." Iwamura discloses a searching algorithm that is designed to be more efficient than alternatives by comparing peak notes from the work to be identified with the peak notes in the database. Iwamura, 6:59-60; 12:1-2. While the individual comparisons of a work to a record in the library can be more efficient using this peak note approach, Iwamura does not teach an algorithm that "scales with a less than linear relationship to the size of the data set to be searched" where the data set is either (a) the number of records in the database, or

(b) even the length of an individual record. Instead, each melody in the melody database is processed as part of the disclosed comparison and "[t]he reference melody that gives the least difference is returned as a search result." Iwamura, 7:53-55.

111. Specifically, Iwamura confirms that the referenced Boyer-Moore algorithm (the basis for alleged disclosure of a sub-linear search in the Petition, Declaration, and Decision) searches all items in the database and even searches "word by word from the beginning of the database to the end" and therefore cannot scale with a less than linear relationship to the size of the data set being search—*i.e.*, it is not sublinear:

> "Boyer Moore (discussed below) or other string-matching algorithms do not have this kind of flexibility. They only search word by word from the beginning of the database to the end."

Iwamura, 9:52-55.[15]

112. The search algorithms disclosed in Iwamura do not reduce the number of records to be searched during a search (or even the data to be searched within a record) as the dataset increases. Rather, the disclosed algorithms speed up the comparison of the work to each record by matching peaks. Iwamura, 9:9-11. Accordingly, the disclosed algorithms in Iwamura search all records in the library,

[15] The word-by-word comparison is valid for the worst case.

and the computational time that the disclosed search takes to make such comparisons grows linearly with the number of records in the database (the relevant analysis) and even linearly with the data in each record. Iwamura therefore teaches a linear search rather than the claimed "sublinear" search as the term is used in the IPR Patents, because the computational time that it takes to perform a search grows linearly as new data is added to the database.

113. The Petition fails to satisfy its burden of demonstrating that Iwamura teaches a "sub-linear time search." As support for the "sub-linear" elements, Petitioner (and corresponding Declaration) exclusively relies on the Boyer-Moore algorithm referenced in Iwamura:

114. Petition: The text of the Petition does not address the sub-linear elements or state that Iwamura discloses a "sub-linear time search." Pet. ('237) at 7-10. Neither the word sublinear nor the concept appears in the text of the Petition.

115. Petition Chart: In its chart, Petitioner exclusively relies on the referenced Boyer-Moore algorithm as support for the sub-linear search elements (highlighted in yellow in the passages below):

Claim 1(b):

b) determining, by the computer system, an identification of the media work using the received features extracted from the media work to perform a sub-linear time search of extracted features of identified media works to identify a neighbor; and	Iwamura determines an identification of the media work using the extracted features by "find[ing] the closest melody from the database," which is a neighbor. 9:25-38, 12:1-2. Iwamura discloses searching using the "Boyer-Moore algorithm" (9:63-64, 10:1-3), which is sublinear (Ex. 1017 at 1). Ex. 1004 at ¶ 72.

Pet. ('237) at 10-11.

Claim 5(b.2) (Petitioner references Claim 1):

2) determining, by the computer system, an identification of the media work using the features extracted from the media work to perform a sub-linear time search of extracted features of identified media works to identify a neighbor; and	Petitioner incorporates the above discussion of Iwamura regarding Claim 1b.

Pet. ('237) at 12.

116. Declaration: The Declaration also exclusively relies on the Boyer-Moore algorithm as support for the sublinear search elements:

> 72. It is my opinion that Iwamura further teaches how this search can be sublinear. For example, Iwamura discloses that different "search algorithms may be applied to perform melody searches." (*id.* at 10:2-3), such as the "Boyer-Moore algorithm," (*id.* at 9:63). "On the average the [Boyer-Moore] algorithm has a sublinear behaviour." Ex. 1017 at 1.

Moulin Decl. ('237) ¶72.

117. Declaration Chart: The chart in the Declaration also exclusively relies on the Boyer-Moore algorithm:

Claim 1(b):

b) determining, by the computer system, an identification of the media work using the received media features extracted from the media work to perform a sub-linear time search of extracted features of identified media works to identify a neighbor; and	Iwamura discloses the use of a "search engine" to determine an identification of the media work using the extracted features by "find[ing] the closest melody from the database," which is a neighbor. 9:25-38, 12:1-2. Iwamura discloses searching using the "Boyer-Moore algorithm" (9:63-64, 10:1-3), which is sublinear (Ex. 1017 at 1).

Moulin Decl. ('237) ¶75.

Claim 5(b.2) (the Declarant references Claim 1):

2) determining, by the computer system, an identification of the media work using the features extracted from the media work to perform a sub-linear time search of extracted features of identified media works to identify a neighbor, and	I incorporate my above discussion of Iwamura regarding Claim 1b.

Moulin Decl. ('237) ¶75.

118. Neither the Petition nor Declaration identifies any basis for asserting that Iwamura discloses the sub-linear search elements other than the referenced Boyer-Moore algorithm. Pet. ('237) at 10-12; Moulin Decl. ('237) ¶72. My understanding is confirmed by Petitioner's Declarant:

```
        Q   The only thing you identify in your
Declaration about Iwamura that could disclose a
sublinear time search is the Boyer-Moore algorithm;
correct?
        A   As far as I remember, yes.  In that
Declaration at that time, yes.
```

Moulin Depo. 82:23-83:3.

119. One skilled in the art would understand that the referenced Boyer-Moore algorithm, however, does not disclose or even address a sublinear search—that is "a search whose execution time scales with a less than linear relationship to the size of the data set to be searched." Decision ('237) at 7. Because Iwamura itself does not state that Boyer-Moore algorithm is sublinear, the entire basis in the Petition and corresponding Declaration for the claimed sublinear elements is the single statement in the Petitioner's Declaration:

"On the average the [Boyer-Moore] algorithm has a sub-liner behavior." Moulin Decl. ('237) ¶72 (quoting Ex. 1017 at 1). One skilled in the art would understand that this statement is not accurate with respect to the relevant sub-linear behavior, *i.e.*, with respect to the size of the database. My understanding was confirmed by Petitioner's Declarant who testified that:

(1) he understood that "sub-linear" in the context of the '237 patent is based on the size of the data set searched, not the size of the query or pattern to be matched (from the work to be identified);

(2) the Boyer-Moore algorithm does not disclose a search that is sublinear with respect to the dataset or database or even the length of a record to be search (it does not even address a database or dataset); and

(3) that when he wrote "which is sublinear" in his Declaration, he did not intend the Board to interpret "sublinear" in the context of the '237 patent but instead in a different context unrelated to '237 patent.

120. (1) As I noted above, Petitioner's Declarant understood that "sublinear" in the context of the '237 patent is based on the size of the searched dataset, not the size of the query or pattern of the work to be matched (which is the correct understanding):

> 53. I understand and agree with Petitioner's position that the term "sublinear search" means "a search whose execution time has a sublinear relationship to database size." For instance, a linear search of a 200-item database would take twice as long as a linear search of a 100-item database. By contrast, a sublinear search of a 200-item database would take less than twice as long as a sublinear search of a 100-item database, perhaps, for instance, 1.5 times as long.

Moulin Decl. ('237) ¶53.

Q Is it the case that in doing your analysis to determine whether or not prior art anticipated, you applied the definition that a sublinear search or a sublinear time search was one whose execution time has a sublinear relationship to the database size?

A That is correct, yes.

Q Is that the correct definition that should be applied, or are you applying the wrong definition?

A In my opinion, this is the correct definition to be used.

Moulin Depo. 24:1-12.

Is it the case that for a search to be sublinear as it's used in the '237 patent, it's not enough for it to have execution time that is sublinear in relationship to the size of the pattern; it must also be sublinear in relationship to the size of the database?

A When I read "sublinear" in, say, Claim 5 of the patent, as we just did, I understand sublinear to mean in relation with the size of the database. It does not say anything about in relation with the size of the query.

Moulin Depo. 26:11-21.

Q Would it be wrong, then, to assess whether a search is sublinear by applying a definition that said, "It's sublinear if this execution time has a sublinear relationship to the size of the query or the pattern"?
A It would not be very relevant. Again, mathematically, it can be done. Everything can be done. But it would not be relevant, from an engineering viewpoint, for an application like this.

Moulin Depo. 25:4-12.

Let's assume that the Patent Trial and Appeal Board was presented with prior art that presented a search that was linear with respect to the size of the database, but it was sublinear with respect to the size of the pattern.
Would that prior art demonstrate a sublinear search as it's used in Claim 25?
A People say it's a linear search, again, because it's in relation with the size of the database. And as you just said, that complexity is still linear; so people would say it is a linear search.
Q It is not a sublinear search?
A It's only sublinear in terms of the size of the query, which is generally not the parameter of -- the relevant parameter.

Moulin Depo. 26:25-27:15.

Q Is it the case that if we had a piece of prior art that was linear with respect to the size of the database but sublinear with respect to the size of the query or the pattern, that that prior art would not teach a sublinear search as it's used in Claim 25?

A Again, if one understands sublinear to be in terms in relation to the size of the database, that would be a -- a linear search.

Moulin Depo. 27:16-24.

Let's assume we've got a piece of prior art that scales at a sublinear relationship with the size of the pattern or query but it scales at a linear relationship with the size of the database that's being searched.

Would that prior art demonstrate or disclose a sublinear search as it's used in the claims of the '237 patent?

A No. Again, because my understanding is the claims of the '237 patent, whenever there's mention of "sublinear," it means in terms of the database size. It does not say it explicitly; it's my inference based on my knowledge and my expertise.

Moulin Depo. 28:4-16.

> Q When you wrote your Declaration, you had in mind that sublinear meant having execution time that increased at a less-than-linear relationship compared to the size of the dataset being searched; right?
>
> A That's in the context of database search, yes.
>
> Q And then you wrote this claim chart in order to indicate where in Iwamura it disclosed each part of the claim; right?
>
> A That's correct, yes.

Moulin Depo. 77:14-24.

121. (2) Petitioner's Declarant confirmed my understanding—that the Boyer-Moore algorithm referenced in Iwamura does not disclose a search that is sublinear with respect to the database size (*i.e.*, the size of the data set to be searched)—it does not even address a database (Moulin Depo. 53:19-22 ("There's no database in Boyer-Moore."))—but instead has a relationship to the size of the query pattern from the work to be identified:

> Q Are you familiar with any analysis of the Boyer-Moore algorithm with respect to the size of the dataset being searched?
>
> A It's described here. So, again, this i, if you look at the worst case, i is N minus patlen, then you obtain it. As I said, it will be a linear relationship.

Moulin Depo. 61:18-24; 44:20-46:6; 59:6-9; 61:25-62:9; 68:25-69:4.

122. (3) Petitioner's Declarant confirmed my understanding—that the statement in his Declaration—Petitioner's only support for the sub-linear elements—was wrong. He testified that when he wrote:

> "On the average the [Boyer-Moore] algorithm has a sublinear behaviour." Ex. 1017 at 1.

(Moulin Decl. ('237) ¶72) and wrote just a few pages earlier:

> 53. I understand and agree with Petitioner's position that the term "sublinear search" means "a search whose execution time has a sublinear relationship to database size." For instance, a linear search of a 200-item database

(Moulin Decl. ('237) ¶53), he was not trying to convey that the Boyer-Moore algorithm was sublinear or "has a sublinear behavior" in the context of the '237 patent –*i.e.*, "has a sublinear relationship to the database size":

> Q. When you wrote this, were you trying to convey that searching using the Boyer-Moore algorithm would be sublinear with respect to the size of the dataset being searched?
>
> A. No. I did not -- no.

Moulin Depo. 74:20-24; 74:8-12.

> Q When we read paragraph 72, are you conveying to the reader that the Boyer-Moore algorithm is sublinear with respect to the size of the dataset being searched?
>
> A No. All I do is quote a part of a paper that shows why that algorithm is much faster than brute force. That's all I'm doing. You're inferring things I'm not saying or writing.

Moulin Depo. 69:9-16.

> In paragraph 72, are you representing to the Board that it's your understanding that the Boyer-Moore algorithm has a sublinear behavior with respect to the size of the dataset?
>
> A No. This is just a quote of another paper discussing Boyer-Moore. This Boyer-Moore algorithm has been used in a variety of contexts, including, of course, content recognition. I'm simply quoting from another paper here. I'm not presenting anything about what you asked.

Moulin Depo. 66:9-18.

> When you wrote this sentence here on page 29, did you think that someone at the Board looking at this might think that you meant that a -- the Boyer-Moore algorithm was sublinear as used in the patent claim?
>
> A I didn't think of it that way, no.

Moulin Depo. 75:23-76:3.

Q And when you wrote this, were you trying to convey to the Board that the Boyer-Moore algorithm is one that, if you use it, it's sublinear with respect to the size of the database?
A No. No.

Moulin Depo. 67:17-21.

123. Consistent with my understanding, Petitioner's Declarant clarified that he was not claiming that the Boyer-Moore algorithm referenced in Iwamura discloses a sub-linear search in the context of the '237 patent, *i.e.*, with respect to the size of the dataset:

Q And next to the phrase "a sublinear time search" in the claim, you wrote, "The Boyer-Moore algorithm, which is sublinear"; right?
A It's not me who wrote it. I'm quoting from a reference.
Q Well, you wrote the words "which is sublinear"; right?
A I quoted from a reference, again, showing why there are much faster alternatives to brute force.
Q Let me rephrase it --
A I'm not -- again, to make it very clear. I'm not claiming that using Boyer-Moore simply alone is going to yield a sublinear search for that database problem. I'm not claiming that. Just to be -- to make it clear.

Moulin Depo. 77:25-78:15.

Q Okay. Would you agree, sir, that if -- that one way to read this would be that you were claiming that the claim language, "perform a sublinear time search," was satisfied by searching using the Boyer-Moore algorithm?

A That might be one way of reading it. It's not the way I'm reading this now.

As I said, the way I'm reading this is I'm quoting language from a reference. And, again, to make the record clear, I'm not claiming that Boyer-Moore, as this alone, is going to give us a sublinear time search in a database search problem. I'm not claiming that. And I did not claim it in this document.

Moulin Depo. 78:16-79:6.

Q Would it be reasonable for the Board to have read this as you opining, you asserting, that Iwamura discloses a sublinear time search because it discloses searching using the Boyer-Moore algorithm, which is sublinear?

A Again, I don't know how different people can read it. If there's any ambiguity, I hope I just cleared it up. I'm not claiming that Boyer-Moore alone is going to give us a sublinear time search for the database search problem.

Moulin Depo. 79:9-18

Q. In an email with the Board, wouldn't it have been better to say, "Board, the Boyer-Moore algorithm is linear, not sublinear"?

A. Listen, there are many words that I -- I'm sure I could have chosen better words. I -- I agree with you, there is probably better ways to write that. I don't dispute that.

Q. Do you agree that it would have been better to tell the Board, "The Boyer-Moore algorithm is linear, not sublinear"?

A. I don't -- I'm not saying it's linear either. I -- I don't know. All I'm saying is I'm not representing that the Boyer-Moore algorithm alone is going to give us a sublinear time search for database searching. That's all I'm saying.

Q. Would the Boyer-Moore algorithm alone give you a sublinear time search for searching a string?

A. Again, are you looking at worst case? The answer is no.

Moulin Depo. 79:19-80:12; 80:15-83:3.

124. Accordingly, one skilled in the art would understand that the support in the Petition and Declaration for the sublinear search elements fails to disclose the sublinear search elements.

125. *Board's concerns*: I now address the Board's specific concerns (identified in its Decision) with respect to whether Iwamura discloses the claimed "sub-linear time search." In instituting Ground 1, I note that the Board preliminary found that Iwamura disclosed the "sub-linear time search because (a) a sub-linear

search of the data within the records can be sublinear even if every record in the database is searched, and (b) Patent Owner's argument that Boyer-Moore searches all items in the database therefore does not demonstrate that the Boyer-Moore algorithm is not sub-linear:

> In addition, we note that no claim in the '237 Patent requires the searching, in the determining aspect of the claims, to be both nonexhaustive and sub-linear, such that a sub-linear search of the data within the records, even if every record is searched, can potentially teach the aspect of independent claims 1 and 5 which recite "perform[ing] a sub-linear time search of extracted features." Although the Specification of the '237 Patent discloses that a sub-linear search is performed on the records of the database and not information within the records, the claims do not specify that the sub-linear search must be performed on a subset of all of the records, and not information within individual records.

Decision ('237) at 11.

> Patent Owner also argues that Iwamura's use of the "'Boyer-Moore algorithm' searches all items in the database and therefore is not sublinear." Prelim. Resp. 18–19. As discussed above, we are not persuaded that this is a deficiency with respect to the instant claims.

Decision ('237) at 12. It is my opinion that the Board's preliminary analysis is flawed on multiple levels for the reasons I explain below.

126. First, the Board's preliminary analysis is based on an incorrect interpretation of the construction of sub-linear as it would be understood by one skilled in the relevant art at the time of the inventions. The Board construed a

"sublinear" search as "a search whose execution time scales with a less than linear relationship to the size of the data set to be searched," not the length of any specific record in the database. As I explained above in detail above and reflected in in the Board's analysis of the construction of sub-linear, the data set is the number of records in the database to be searched—"the size of the data set ("N")." Decision ('237) at 7.

127. In addition, as I explained above in detail, those skilled in the art understand that the size of the data set in the context of the '237 patent refers to the number of records in the database to be searched (N) and not the length of any particular record in the database. This understanding is consistent with Dr. Moulin's explanation in his Declaration. *See* Moulin Decl. ('237) ¶53. Accordingly, the Board's preliminary analysis is based on an improper interpretation of the construction of "sublinear."

128. Second, it is my understanding that the Board's preliminary analysis has the relevant burden backwards—it is not the Patent Owner's burden to demonstrate that the referenced Boyer-Moore algorithm does not disclose a sublinear search. Rather it is my understanding that it was the Petitioner's burden to demonstrate that referenced Boyer-Moore algorithm discloses a sublinear search. As I showed above, Petitioner failed to satisfy this burden. As I explained

above, in my opinion, one of ordinary skill in the art would understand that Boyer-Moore algorithm is not a sublinear search in the context of the '237 patent.

129. Third, one skilled in the art would understand that there is no evidence under any interpretation of sublinear in the context of the '237 patent that the referenced Boyer-Moore algorithm discloses a search that is sublinear with respect to either (a) the "size of the dataset" (Decision ('237) at 7); or (b) the length of an individual record being searched. In my opinion, one of ordinary skill in the art would understand that it is not.

130. The two references to the Boyer-Moore algorithm in Iwamura are:

> Boyer-Moore (discussed below) or other string-matching algorithms do not have this kind of flexibility. They only search word by word from the beginning of the database to the end.

Iwamura, 9:52-55.

> There are many studies for fast and efficient string search techniques. For example, the Boyer-Moore algorithm is well-known as one of the best solutions. See

Iwamura, 9:61-64. While the Boyer-Moore algorithm is described as being "efficient," one skilled in the art would understand that neither passage states that the algorithm is sublinear with respect to either the number of references in the database or the length of an individual record to be searched.

131. Fourth, as I explained above, Petitioner's Declarant confirmed my understanding—that the referenced Boyer-Moore algorithm does not disclose a search that is sublinear in the context of the '237 patent.

2. approximate nearest neighbor search (claim elements 9(b) and 13(b.2)).

132. As I presented above, one of ordinary skill in the art would understand that, in the context of the '237 patent, an "approximate nearest neighbor search" is a sub-linear search identifying a close match that is not necessarily the closest match. Also, as I explained above, a search that necessarily identifies the closest match is not an "approximate nearest neighbor search" even if it also identifies other near matches.

133. One skilled in the art would understand that Iwamura does not disclose the claimed "approximate nearest neighbor search" for two independent reasons.

134. Reason 1: One skilled in the art would understand that Iwamura does not disclose an "approximate nearest neighbor search" because Iwamura does not disclose "identifying a close match that is not necessarily the closest match." Iwamura discloses a search that always identifies an exact or the closest match. Consistent with my understanding, Petitioner's Declarant likewise confirmed that

Iwamura will either produce an "exact match" if it finds one, or the "best match it finds using that approximate criterion." Moulin Depo. 271:22-272:12.

135. The system in Iwamura will always find the closest match, even if unimportant peaks are skipped or repeated patterns are avoided. My understanding is consistent with the understanding of Petitioner's Declarant:

- "[W]'re still going to be identifying the closest match" even when "the unimportant peaks are skipped.... Dropping an unimportant part is not going to affect the ability to find the best match." Moulin Depo. 317:14-23.
- "If we implement that feature of Iwamura... skipping a repeated pattern.... It will not affect the ability to find the best match." Moulin Depo. 318:11-18.

136. Petitioner asserts that Iwamura identifies a neighbor because: "the 'search engine will find the closest melody from the database." Pet. ('237) at 8 (*quoting* Iwamura, 9:24-25)); Moulin Decl. ('237) ¶69. A person of ordinary skill in the art would understand that these statements do not disclose an "approximate nearest neighbor search" which is a search identifying a close match that is not necessarily the closest match. Instead, these statements confirm that Iwamura always identifies the closest match—necessarily the closest match—rather than a match that is not necessarily the closest match as required by the claimed

"appropriate nearest neighbor search." *See* '237, 9:15–16 (an "approximate nearest neighbor search does not always find the closest point to the query.").

137. Because the searches disclosed in Iwamura necessarily return the closest match, they are not search algorithms that identify a match that is not necessarily the closest match, as the properly construed claim element requires. Accordingly, in my opinion, Iwamura neither expressly nor inherently (necessarily) discloses an "approximate nearest neighbor search"—a search that does not necessarily find the closest match.

138. Reason 2: One skilled in the art would understand that Iwamura does not disclose an "approximate nearest neighbor search" because Iwamura does not disclose a sublinear search. As I demonstrated above, an "approximate nearest neighbor search" is "one example" of a sublinear search. Also, as I demonstrated above, Iwamura does not disclose a sublinear search. Accordingly, Iwamura does not disclose the claimed "approximate nearest neighbor search."

139. One skilled in the art would understand that the Petition, Declaration, and corresponding charts fail to demonstrate that Iwamura discloses the claimed "approximate nearest neighbor search." As support for the claimed "approximate nearest neighbor search," the Petition and corresponding Declaration rely on (1) the fault tolerance feature, and (2) skipped portions feature, described in Iwamura.

140. Petition: The text of the Petition does not address the claimed "approximate nearest neighbor search"—I note that the words "approximate nearest neighbor search" do not appear in the text of the Petition.

141. Petition Chart: Petitioner provides the following in its claim chart:

Claim 9(b):

b) determining, by the computer system, an identification of the media work using the received features extracted from the media work to perform an approximate nearest neighbor search of extracted features of identified media works, and	Petitioner incorporates the above discussion of Iwamura regarding Claim 1b. Furthermore, Iwamura uses an approximate nearest neighbor "search engine [that] has . . . input fault tolerance capability" (10:17-18), and skips "portions that should not be searched" (12:6-7), such as "repeated patterns" (9:36-44), and "unimportant portion[s]" of the melody (9:44-45).

Pet. ('237) 12.

Claim 13(b.2) (referencing claim element 9(b)):

2) determining, by the computer system, an identification of the media work using the received features extracted from the media work to perform an approximate nearest neighbor search of extracted features of identified media works, and	Petitioner incorporates the above discussion of Iwamura regarding Claim 9b.

Pet. ('237) 13.

142. Declaration: The text of the Declaration also does not address the claimed "approximate nearest neighbor search."

143. Declaration Chart: Petitioner' Declarant provides the following in its claim chart:

Claim 9(b):

b) determining, by the computer system, an identification of the media work using the received features extracted from the media work to perform an approximate nearest neighbor search of extracted features of identified media works, and	I incorporate my above discussion of Iwamura regarding Claim 1b. Furthermore, Iwamura discloses using an approximate nearest neighbor "search engine [that] has input fault tolerance capability" (10:17-18), and skips "portions that should not be searched" (12:6-7), such as "repeated patterns" (9:36-44), and "unimportant portion[s]" of the melody (9:44-45).

Moulin Decl. ('237) ¶75.

Claim 13(b.2) (referencing claim element 9(b)):

2) determining, by the computer system, an identification of the media work using the received features extracted from the media work to perform an approximate nearest neighbor search of extracted features of identified media works, and	I incorporate my above discussion of Iwamura regarding Claim 9b

Moulin Decl. ('237) ¶75.

144. I note that these statements in the Petition (and Declaration) and corresponding passages from Iwamura do not:

(a) provide a construction of "approximate nearest neighbor search,"

(b) explain how Iwamura discloses the claimed "approximate nearest neighbor search,"

(c) explain why the fault tolerance capability and skipped portion are relevant to or disclose an "approximate nearest neighbor search," and

(d) establish that Iwamura discloses an "approximate nearest neighbor search."

145. One skilled in the art would understand that the quoted passages do not disclose an "approximate nearest neighbor search" because the quoted passages do not disclose a search that (a) is not guaranteed to identify the closest match, and (b) is sublinear.

146. First, as I noted above, the passage from element 1(b) cross-referenced in Petitioner's chart ("Petitioner incorporates the above discussion of Iwamura regarding Claim 1b") does not disclose an "approximate nearest neighbor search." As I explained above, one skilled in the art would understand that an "approximate nearest neighbor search" identifies a close match that is not necessarily the closest match. *See* Decision ('237) at 9. The passage cited in the Petition (and corresponding Declaration) confirms that the search disclosed in Iwamura finds "the closest melody from the database." Pet. ('237) at 8 (*quoting* Iwamura, 9:24-35).

147. Second, one skilled in the art would understand that Petitioner's references to searches that have (a) an "input fault tolerance" (Pet. ('237) at 12, *quoting* Iwamura, 10:17-18), or (b) skipped "portions that should not be searched" (Pet. ('237) at 12 *quoting* Iwamura, 12:6-7, 9:36-44, and 9:44-45) do not expressly or inherently (necessarily) disclose a search that does not necessarily identify the closest match and is sublinear. A key issue in addressing whether a search is an "approximate nearest neighbor search" is whether the search is designed to and

will necessarily identify an exact match or the closest match, or whether the search could identify search results that do not include an exact or the closest match. If a sublinear search can return a "close match that is not necessarily the closest match," it is an "approximate nearest neighbor search." But if a search cannot return a "close match that is not necessarily the closest match" (because it is designed to only find the closest match), then it is not an "approximate nearest neighbor search," irrespective of how the search is performed.

148. The input fault tolerance and skipped sections search features describe how a peak note search may be performed. Neither enables a peak note search to return a result other than the closest match. While the Petition identifies these two search features—the way the search is conducted—the Petition does not address the output of the searches much less identify a search that does not necessarily identify the closest match. As demonstrated above, the output from any disclosed Iwamura search always identifies the closest match and therefore is not an "approximate nearest neighbor search"—a search "identifying a close match that is not necessarily the closest match." Iwamura therefore does not disclose an approximate nearest neighbor search. I will specifically address each of the two search features identified by Petitioner is addressed in turn.

<u>input fault tolerance</u>

149. Iwamura discloses that its peak note search can include an "input fault tolerance." Iwamura, 9:20-24. Input fault tolerance allows a user to identify the closest match, even when the melody entered by a user has some errors. Iwamura, 9:33-39 (input fault tolerance enables "a correct search . . . notwithstanding inaccurate input from the user."). Using the fault tolerance feature, the peak note search first performs a search based on a tolerance of no errors, then a tolerance of one error, then a tolerance two errors, *etc*. The search will continue to search based on additional errors only if the search has not identified a match.

150. Accordingly, using the fault tolerance feature, the Iwamura search always produces an exact match or the closest match—it does not produce a result that is not necessarily the closest match. *See e.g.*, Iwamura, 11:43-45 ("The invented input fault tolerance function allows the user to obtain an exact result even when an entered melody has some errors."). Because the record identified using the fault tolerance search is necessarily the closest match, it is not a search that returns a "close match that is not necessarily the closest match," and, as a result, the feature does not disclose the claimed "approximate nearest neighbor search."

skipped sections

151. Iwamura also teaches that the disclosed search has "flexibility on search area" within a record in the reference database. Iwamura, 9:35. For

example, a user can identify the "important" portions of a melody, thereby enabling the search to skip the remaining "unimportant" portions. *See* Iwamura, 9:45-50 ("In a long music selection, there are some important portions that are indispensable to identify the melody. These portions are well recognized and remembered by the user. The user identifies such important portions as a keyword (key-melody). The other unimportant portions [in a long reference melody] can often be ignored."). The skipped sections feature is a pre-processing component, and what remains to be searched can be viewed as the "extracted" features over which an exhaustive search is performed until a match is found.

152. Iwamura does not disclose that flexibility on search area enables the Iwamura search to return a result other than the closest match. *See* Iwamura, 9:35-55. Because the record identified using the skipped portion search feature is still necessarily the closest match, it is not a search that returns a "close match that is not necessarily the closest match" and the feature does not disclose the claimed "approximate nearest neighbor search."

153. Moreover, as I explained above, an "approximate nearest neighbor search" is a sub-linear search, and each of the passages cited by Petitioner does not disclose a sub-linear search.

Board's concerns:

154. I now address the Board's specific concerns (identified in its Decision) with respect to whether Iwamura discloses the claimed "approximate nearest neighbor search." In instituting Ground 1, the Board preliminary found that Iwamura disclosed the "approximate nearest neighbor search" because the "approximate nearest neighbor search" "does not require that all of the records in the library are not used":

> With respect to "approximate nearest neighbor search," Patent Owner argues that the input fault tolerance capability of Iwamura cannot teach the same because it does not state or imply "that all records in the music library are not used in the comparison as required in an 'approximate nearest neighbor search.'" *Id.* at 19–20. Our construction of "approximate nearest neighbor search" to be "identifying a close match that is not necessarily the closest match" does not require that all of the records in the library are not used, so we also do not find this to be a deficiency of the ground.

Decision ('237) at 12. It is my opinion that the Board's preliminary analysis is flawed at multiple levels.

155. First, it is my understanding that the Board's preliminary analysis has the relevant burden backwards—it is not the Patent Owner's burden to demonstrate that the referenced "fault tolerance capability of Iwamura" does not disclose an "approximate nearest neighbor search." Rather it was the Petitioner's burden to demonstrate that Iwamura (and the "fault tolerance capability") discloses an

"approximate nearest neighbor search." As I demonstrated above, Petitioner did not satisfy this burden.

156. Second, as I demonstrated above, one skilled in the art would understand that there is no evidence that the referenced "fault tolerance capacity of Iwamura" teaches a search that identifies a close match that is not necessarily (*i.e.*, not guaranteed to be) the closest match rather than search that is guaranteed to identify the closest match. As I demonstrated above, the evidence confirms the opposite—that Iwamura finds "the closest melody from the database." Pet. ('237) at 8 (*quoting* Iwamura, 9:24-25).

157. Third, as I demonstrated above, an "approximate nearest neighbor search" is a sublinear search, and there is no evidence that the referenced "fault tolerance capability of Iwamura" teaches a sublinear search as the phrase is used in the context of the '237 Patent.

3. nonexhaustive search (claim element 25(b)).

158. As I explained above, a "nonexhaustive search" is "a search that locates a match without a comparison of all possible matches." Decision ('237) at 7.

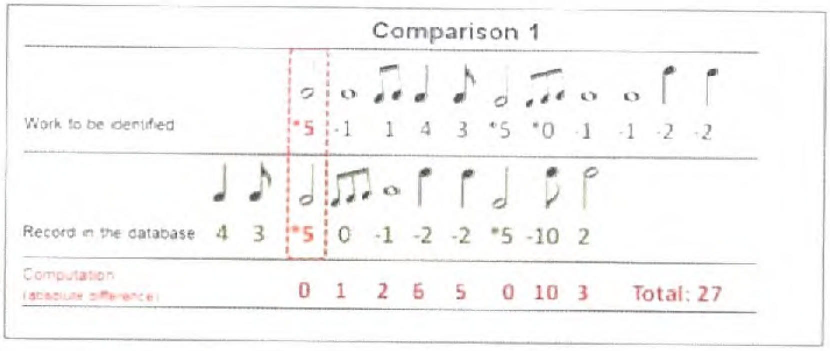
159. One skilled in the art would understand that Iwamura does not disclose a non-exhaustive search as the phrase is used in the context of the '237

Patent. As I described above, Iwamura discloses a searching algorithm that is designed to be more efficient than alternatives by lining up peak notes from the music work to be identified with the peak notes in each record in the music database when comparing the work to each record. Iwamura, 12:1-2. Instead of comparing the work to be identified with a record in the database by (a) preforming a first comparison of the notes in the work and the record, and then (b) shifting the comparison between the work and the record "note by note" to see if there is a match, Iwamura teaches that the shifting can be done peak-note-to-peak-note, thereby reducing the number of comparisons made between the work and a specific record, thus making the comparison more efficient.

> "Peak notes are approximately 20% of the total number of notes in a typical melody. That means search speed using peak notes is 20% of a brute force search which shifts the entered melody, note by note."

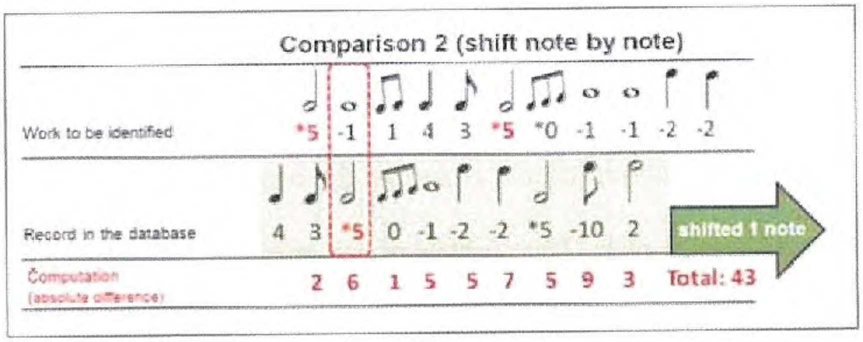
Iwamura, 9:9-11; *see* Iwamura, 5:9-13 ("The peaks in all the melodies stored in the databases are marked in advance. For melody matching, the entered melody is time-shifted . . . so that its peak matches each peak in the reference melody.").

160. This peak note search process can be illustrated using the example notes from Iwamura (Iwamura, 7:11-45). The following illustrates a first comparison between the notes from the work to be identified and the notes in a single record in the database:



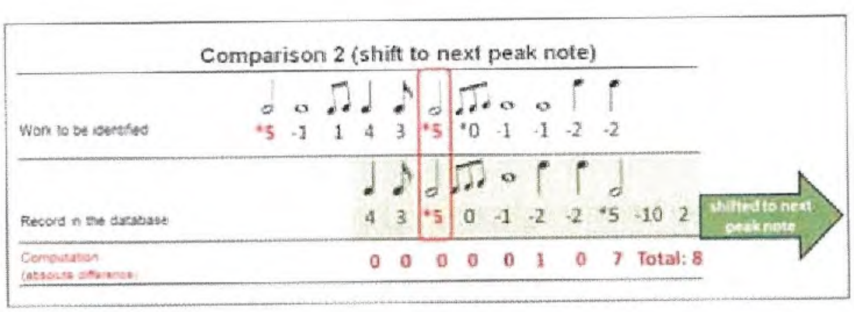
The top row represents the notes in the work to be identified; the middle row (highlighted in green) represents the notes of the record in the database being searched; and the bottom row (text in red) represents the absolute difference between the compared notes. The "peak notes" in the work to be identified and the record being searched are identified by "*". In this first comparison, the first peak note from the work to be identified (*5) and the record (*5) are aligned (as illustrated by the dashed red outline). Note that the computation (the absolute difference between the work to be identified and the record) results in a total value of 27 (0+1+2+6+5+0+10+3).

161. In a second comparison between the work to be identified and this same record in the reference database, the record in the database is shifted to the right by a single note (this is the "note by note" approach referenced in Iwamura (Iwamura, 9:9-11)):



The peak notes are not aligned in this comparison (as illustrated by the dashed red outline). The computation (the absolute difference between the work to be identified and the record) results in a total value of 43 (2+6+1+5+5+7+5+9+3).

162. An alternative to the second comparison presented above is to use the peak note approach taught in Iwamura. Using this peak note approach, the second comparison between the notes of the work to be identified and the notes in the record in the database is not just shifted one note to the right but is shifted to the right to align the next peek note (*i.e.*, five notes to the right), thereby skipping what would have been four intermediate comparisons using the alternative note by note approach:



As a result, the peak note approach taught in Iwamura avoided four unnecessary comparisons between the work to be identified and this reference work, making this peak note search more efficient. Note that the computation now results in a total absolute difference of 8. The number of comparisons that are avoided is 4 * (length of the query) as computing the individual distances between the notes requires a comparison.

163. Each melody in the melody database is compared using this peak note approach and "[t]he reference melody that gives the least difference is returned as a search result." Iwamura, 7:53-55. Because the peak note search algorithm disclosed in Iwamura does not reduce the number of records to be searched or even the notes in each record to be searched but rather speeds up the individual comparison of the work to be identified to each record (by shifting the comparisons by peak notes rather than note by note), the disclosed algorithm searches all records in the library and is therefore an exhaustive search rather than the claimed "non-exhaustive" search. This approach does not reduce the number of records being searched *e.g.*, by discarding clusters of potential matches, like the sub-linear searches addressed in the IPR Patents. *See e.g.*, '237, 8:64-9:7 ("Other forms of matching include those based on clustering, kd-trees, vantage point trees and excluded middle vantage point forests are possible and will be discussed in more detail later. . . . Thus, for example, a sub-linear search time can be achieved.")

While the individual comparisons of a work and a record in the library can be more efficient using the peak note approach disclosed in Iwamura ("search speed can be increased"), in doing so each record in the library is searched as part of the disclosed algorithm and "[t]he reference melody that gives the least difference is returned as a search result." Iwamura, 7:53-55.

164. Accordingly, one skilled in the art would understand that Iwamura teaches an exhaustive search rather than the claimed "non-exhaustive" search, because it searches all records in the database using the peak note approach.

165. I note that Petitioner's Declarant, Dr. Moulin, confirmed that "for all the Iwamura searches…[i]t's understood that you search through every musical work in the database"—*i.e.*, all potential matches (Moulin Depo. 269:19-270:2):

Q You would agree that in Iwamura, the search that's identified there does make a comparison to each of the possible musical works that could be returned as a match?
MR. ELACQUA: Objection.
THE WITNESS: To each of the musical works, yes.

Moulin Depo. 223:2-8.

Q We've looked at each of the possible matches. We go through each record; right? Yes?
A Yes.

Moulin Depo. 247:18-20.

Q Well, if it -- it's going to do a comparison to each reference work; right?

A That's correct.

Moulin Depo. 271:19-21.

Q Now, in the type of search that's identified and described in Iwamura, what it does is it does a comparison of the unknown melody to each of the melody patterns that are in the melody database; right?

A Yes.

Moulin Depo. 207:18-23. As a result, consistent with my understanding and the understanding of one skilled in the art, Petitioner's Declarant confirmed that, based on the proper construction of a non-exhaustive search (adopted by the Board), Iwamura does not disclose a non-exhaustive search:

Q By "nonexhaustive search," we mean a search that does a comparison but doesn't look at each of the melodies in our reference database. It's going to skip over some of them.
By "exhaustive search," we're going to go to each one of them. We might not use all the data for that melody, but we're going to do some comparison of the data to each of the melodies.
Do you understand?
A Yes.
Q With that definition, does Iwamura disclose a -- an exhaustive search or a nonexhaustive search?
A Well, with your, again, incorrect definition, that would be an exhaustive search. But, again, I disagree with your definition.

Moulin Depo. 233:24-234:14.

I want you to assume that by "exhaustive search," we mean a search that does a comparison to -- between an unknown work to each of the musical works in a database, but it doesn't require a comparison to all the data for each of the musical works.

And by a "nonexhaustive search," we mean one that does a comparison but doesn't look at each of the musical works. It just looks at some of them.

Would you agree that Iwamura -- with that definition -- teaches an exhaustive search?

A Under your incorrect premise, it would teach an exhaustive search.

Q By "incorrect premise," you mean the incorrect definition?

A Yes.

Moulin Depo. 225:16-226:7.

Q Before that search is run, each of those works is a possible match; right?
A Yes.
Q Would you agree that the Iwamura search, when it's run, it does a comparison of the unknown work to each of those possible matches?
A To each of those possible music works, yes. An approximate comparison, just to be clear.
Q And by "approximate," you mean that it doesn't necessarily look at every bit of data in every musical work?
A It does not necessarily -- exactly -- use all the data, and then it uses only approximations to the matching criterion.
Q But it does examine each of the possible musical works -- or each of the musical works that could be returned as a possible match?
A Yes.

Moulin Depo. 217:1-18.

166. I note that Petitioner's Declarant also confirmed that (as illustrated in the examples presented above) "all the notes" from each record in the database are compared. As a result, the searches disclosed in Iwamura would not be non-exhaustive even based on Petitioner's construction that includes the improper "and all data within all possible matches" clause (Pet. ('237) at 6):

Q I understand. But if we've got in -- in
database a -- it's more than the peak. We've got a
set of notes -- it doesn't say evaluate just the
peak, does it?
A No.
Q We're evaluating all the notes in that
phrase; right?
A Right. Right.

Moulin Depo. 280:6-13.

Q It says, "In this manner, the entered
melody is shifted to each peak in each reference
melody and compared."
Do you see that?
A Yes.
Q Does this indicate to you that Iwamura is
teaching a peak search method in which it's going to
compare the unknown melody with each peak in the --
each reference melody?
A In this case -- I'm just reading the
context. Okay?
So all the notes are used. Okay. We are
back to this same numerical example. So all the
notes are used in this example, and therefore, he
evaluates the -- you know, the least absolute error
criterion.

Moulin Depo. 277:6-21.

167. The Petition, Declaration, and corresponding charts fail to demonstrate that Iwamura discloses a "nonexhaustive search." Petitioner and its

Declarant identify three features of the Iwamura search as teaching non-exhaustive searching:

(a) peak notes: a search that shifts the comparison of the notes in the work to be identified with the notes in a records by peak notes rather than note-by-note;

(b) limit function: comparing the work to be identified with a specific record in the database can be stopped and shifted to the next peak notes when the computation of the total absolute difference between the notes in the work to be identified and the specific record exceeds a certain limit;

(c) unsearched portions: a search that skips portions that should not be searched, such as "repeated patterns" and "unimportant melodies."

Pet. ('237) at 9-10. Petitioner identifies these three features from Iwamura (labeled ❶, ❷, and ❸ below) as disclosing the non-exhaustive search in its Petition, Declaration, and corresponding charts, addressing either all three features or two of the features:

168. Petition:

> Claim 25 of the '237 patent further requires that the search is "nonexhaustive." Ex. 1001 at Claim 25. Iwamura further teaches how this search can be non-exhaustive. For example, Iwamura teaches a non-exhaustive search that uses "peak notes." Ex. 1012 at 6:31-7:55. "Peak notes are approximately 20% of the total number of notes in a typical melody. That means search speed using peak notes is 20% of a brute force search. . . ." *Id.* at 9:8-11. In another example of non-exhaustive search, Iwamura teaches decreasing search time by stopping the search when computations "exceed[] a certain limit." Ex. 1012 at 7:56-57. In yet another example of non-exhaustive search, Iwamura discloses skipping "portions that should not be searched" (*id.* at 12:6-7), such as "repeated patterns" (*id.* at 9:36-44) and "unimportant portion[s]" of the melody (*id.* at 9:44-45).

Pet. ('237) 9-10.

169. Petition chart:

b) determining, by the computer system, an identification of the media work using the media work extracted features to perform a nonexhaustive search of reference extracted features of reference media works to identify a near neighbor; and	Petitioner incorporates the above discussion of Iwamura regarding Claim 9b. Iwamura further discloses non-exhaustive search algorithms using "peak notes" (6:31-7:55), which "are approximately 20% of the total number of notes in a typical melody," meaning "search speed using peak notes is 20% of a brute force search" (9:9-10). The search is further non-exhaustive because it can be accelerated by stopping the search when computations "exceed[] a certain limit." 7:56-57.

Pet. ('237) 15.

170. Declaration:

> 71. It is my opinion that Iwamura further teaches how this search can be non-exhaustive. For example, Iwamura teaches a non-exhaustive search that uses "peak notes." *See id.* at 6:31-7:55. "Peak notes are approximately 20% of the total number of notes in a typical melody. That means search speed using peak notes is 20% of a brute force search . . . ." *Id.* at 9:8-11.
>
> 73. It is my opinion that Iwamura's disclosure that the search can be accelerated by stopping the search when computations "exceed[] a certain limit" is another example of non-exhaustive searching. Ex. 1012 at 7:56-57.
>
> 74. It is my opinion that Iwamura's disclosure of skipping "portions that should not be searched" (Ex. 1012 at 12:6-7) wherein these skipped portions include "repeated patterns" (*id.* at 9:36-44) and "unimportant portion[s]" of the melody (*id.* at 9:44-5) constitutes another example of non-exhaustive searching.

Moulin ('237) Decl. ¶¶71, 73-74.[16]

171. Declaration Chart:

h) determining, by the computer system, an identification of the media work using the media work extracted features to perform a nonexhaustive search of reference extracted features of reference media works to identify a near neighbor; and	I incorporate my above discussion of Iwamura regarding Claim 9b. Iwamura further discloses using non-exhaustive search algorithms using "peak notes" (6:31-7:55), which "are approximately 20% of the total number of notes in a typical melody," meaning "search speed using peak notes is 20% of a brute force search" (9:9-10). The search is further non-exhaustive because it can be accelerated by stopping the search when computations "exceed[] a certain limit." 7:56-57.

[16] Paragraph 72 of Dr. Moulin's Declaration addresses the "sublinear" rather than the "non-exhaustive" element.

Moulin Decl. ('237) ¶75.

172. One skilled it the art would understand that none of these three Iwamura search features disclose the claimed "non-exhaustive search." Each feature accelerates search speed within a single comparison of a work to be identified with a record in the reference database. No feature, however, enables the disclosed search to locate a match without comparing the work to be identified with each record in the reference database. I address each feature in turn.[17]

173. *peak notes*: A person of ordinary skill in the art would understand that the Iwamura "peak note" approach does not disclose a search that can locate a match without a comparison of all possible matches. As I explained above, a feature of the Iwamura search is that the search speed can be increased if the peaks of a melody input by a user are matched to the peaks of each reference melody, *i.e.*, each record in the reference database and the comparison between the work

---

[17] I observed that Petitioner's Declarant also confirmed that another search feature disclosed in Iwamura—fault tolerance (that was not identified by Petitioner as support for the "non-exhaustive" search element)—also "does a comparison of the unknown work to each of the melodies in our reference database …it compares with every musical work, yes, in the database" and is therefore an exhaustive rather than non-exhaustive search. Moulin Depo. 268:15-20.

and the record is shifted by peak notes rather than note by note. *See* Iwamura 5:9-13 ("The peaks in all the melodies stored in the databases are marked in advance. For melody matching, the entered melody is time-shifted . . . so that its peak matches each peak in the reference melody.").

174. Peak note searching accelerates a search within a single comparison of the work to be identified with an individual record because, when comparing the notes of the work with the notes of the record, it shifts the notes to be compared by peak notes rather than note by note:

> "Peak notes are approximately 20% of the total number of notes in a typical melody. That means search speed using peak notes is 20% of a brute force search which shifts the entered melody, note by note."

Iwamura, 9:8-11.

175. While this search technique may be efficient, the peak note searching disclosed in Iwamura still requires exhaustively searching every reference melody. Iwamura, 9:11-13 (discussing a faster comparison of "each reference melody" with respect to peak note searching); *see also* Iwamura, 7:52-54 (noting that in the search process, "the entered melody is shifted to each reference melody and compared"). As Petitioner's Declarant repeatedly confirmed (consistent with my understanding) "you search through every musical work in the database" for "all

the Iwamura searches" (including the "peak note" approach) (Moulin Depo. 269:19-270:2):

> Q. Would you agree that in Iwamura's peak search note, that it does search across each melody in the melody database?
>
> A. Yes, it does. It's going to compare it with every song in the database, typically.

Moulin Depo. 213:23-214:2; 223:2-8; 247:18-20; 271:19-21.

176. Under the proper construction of "non-exhaustive," the "peak note" approach and the corresponding passages from Iwamura cited in the Petition and Declaration do not disclose a non-exhaustive search because they do not state or suggest that all references in the music library are not compared. Rather, all reference melodies are compared and "[t]he reference melody that gives the least difference is returned as a search result." Iwamura, 7:52-55. Accordingly, a search using "peak notes" is not a non-exhaustive search.

177. Moreover, even applying the "all data" clause in Petitioner's improper construction—a non-exhaustive "search … locates a match without conducting a brute force comparison of … all data within all possible matches"—the peak note search disclosed in Iwamura is still an exhaustive (rather than non-exhaustive) search because it compares "all data within all possible matches." When comparing a work to be identified with each potential match, the peaks of the song

to be identified are lined up with the peaks of the reference work to expedite the comparison: "In this manner, the entered melody is shifted to each peak in each reference melody and compared." Iwamura, 7:52-55. But in doing so, this does not mean that only the peaks from the work to be identified are compared to the peaks of the reference work. Rather, once the peaks are lined up, both the peaks and valleys (all data) are compared in the computation. Dr. Moulin, at his deposition, agreed with this understanding of the peak note search:

> Q It says, "In this manner, the entered melody is shifted to each peak in each reference melody and compared."
> Do you see that?
> A Yes.
> Q Does this indicate to you that Iwamura is teaching a peak search method in which it's going to compare the unknown melody with each peak in the -- each reference melody?
> A In this case -- I'm just reading the context. Okay?
> So all the notes are used. Okay. We are back to this same numerical example. So all the notes are used in this example, and therefore, he evaluates the -- you know, the least absolute error criterion.

Moulin Depo. 277:6-21.

178. While the Petition (Pet. ('237) at 5) quotes a passage from Iwamura that suggests Iwamura avoids a "brute force" search, one skilled in the art would

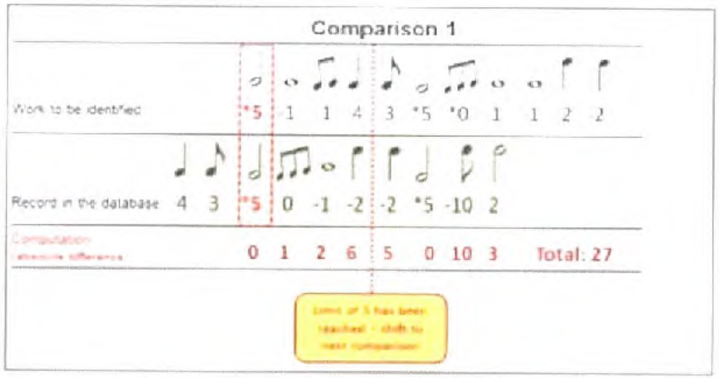
understand that the "brute force" being avoided (and what makes the algorithm efficient) is that peaks are not compared to valleys and valleys are not compared with peaks. Instead, by lining up the peaks when comparing the data, peaks are compared with peaks and valleys are compared with valleys. Therefore, when Iwamura states that its approach is 20% more efficient than a brute force search, one skilled in the art would understand that this does not mean that the peak note approach disclosed in Iwamura does not consider "all possible matches" or even "all data in all possible matches." Rather, it means that by lining up the peaks when doing the comparison, it will save time over comparing the music to be identified with the referenced song without first lining up the peaks; shifting the comparisons by peak notes is more efficient than simply shifting the comparisons "note by note." Iwamura, 9:8-11.

179. *limit function*: One skilled in the art would understand that the limit function approach addressed in Iwamura does not disclose non-exhaustive search under either the proper construction or under the Petitioner's flawed construction. Under the proper construction, a non-exhaustive search locates a match without comparing the work to be identified with all possible records in the reference database. The Iwamura limit function is not a search that locates a match without comparing the work to be identified with all possible matches. The Iwamura limit function accelerates the process of comparing the work to be identified to a single

record in the reference database. The limit function describes the ability of a user to input a "limit" whereby a computation based on comparing the notes of the work to be identified with the notes of an individual record for a particular peak will be stopped and shifted to the next peak for that record when the total absolute difference between the compared notes exceeds a certain value. Iwamura, 7:56-58.[18] Nothing in Iwamura talks about absolute distance calculated for a record (*i.e.*, across all peaks in the record), only for each peak.

180. For example (using the examples provide in Iwamura, 7:11-45), assume a user inputs a limit where the computation comparing the notes of the work to be identified and a single record in the database would be stopped when the total absolute difference in the computation exceeds 5:

18 Computation refers to the process of comparing the absolute difference between the integer values assigned to the notes in the work to be identified (the melody input by the user to be identified) and a single melody record in the reference database for a specific peak comparison.



This comparison would be stopped before all notes have been compared for this specific alignment because comparing the first four notes results in a computation of an absolute difference that exceeds the limit of 5: 0 + 1 + 2 + 6 exceeds the set limit of 5.

181. Once a peak range search is stopped by the limit function (*i.e.*, the total absolute difference exceeds a certain limit so that the computation is stopped), the search shifts to the next peak range comparison within the same record, and continues the search process until each peak in each record is compared against the melody input by the user. A search that uses the limit function disclosed in Iwamura will still compare every record in the reference database: "In this manner, the entered melody is shifted to each peak in each reference melody and compared. The reference melody that gives the least different is returned as a search result." Iwamura, 6:31-7:55. My understanding of how the limit function of Iwamura works was confirmed by Petitioner's Declarant:

Q Now, if we -- and then, as you understand it, what the search algorithm would do, it would then shift this peak over to the next peak and start another calculation; is that right?
A Yes. Yes.

Moulin Depo. 241:24-242:2.

182. Petitioner's Declarant confirmed that "you search through every musical work in the database" for "all the Iwamura searches" (including the limit function approach). Moulin Depo. 269:19-270:2.

Q -- when we throw in this limit on the computation, we are going to still do a comparison to each of the works that are in our database; right?
A We do a comparison of some data within each musical work, yes.
Q You don't read this as saying that if we're doing a computation, and for one of them it exceeds a certain limit, then we stop the search altogether? It doesn't say that, does it?
A No. We move to the next one.
Q By "the next one," you mean the next computation?
A The next possible match. Like after the first line in your example, we move to the second one.

Moulin Depo. 243:17-244:7.

> Q. If we use this parameter in doing the search in Iwamura, is it the case that after we determine that the calculation should be [illegible], then that we stop the search altogether, or do we keep calculating --
>
> A. No. We move to the next one. Or like here, if the parameter value is [illegible], once we reach [illegible], we abandon this and move on to the next one.
>
> Q. Then after we complete the peak search analysis for a given work, then we go to the next work; is that right?
>
> A. Yes.

Moulin Depo. 242:19-243:5.

183. One skilled in the art would understand that the limit function search disclosed in Iwamura is therefore exhaustive.

184. Moreover, even using the "all data" clause from Petitioner's improper construction, one skilled in the art would understand that the limit function algorithm disclosed in Iwamura is still exhaustive rather than non-exhaustive because it compares "all data within all possible matches." While the search comparing a particular peak pattern of a work against a record can be stopped if the difference exceeds a certain limit, this does not mean that the comparison of the work with the record stops. Rather, as I described above, this means that the data in the work will be shifted against the record to match up with the next peak and the comparison of all the data will continue. Nothing in Iwamura expressly states

that all data will not be searched and a search that does not compare all data is also not inherent (*i.e.*, necessarily present).

185. *unsearched portions*: One skilled in the art would understand that this unsearched portion approach disclosed in Iwamura does not disclose a non-exhaustive search. If a search compares the work to be identified to each reference in a database, it is not the claimed non-exhaustive search. Even if certain portions of a reference are skipped, the unsearched portions approach of Iwamura still compares the work to be identified with all potential matches. Consistent with my understanding, Petitioner's Declarant confirmed that all musical records in the reference database are searched under all Iwamura searches (including the unsearched portions approach):

> Q Is it true that for all the Iwamura searches, all the variations that Iwamura teaches, it always teaches doing a comparison to each of the musical works that's a possible match in our database?
>
> A I don't think it says it explicitly every time. It's -- it's often implicit. It's understood that you search through every musical work in the database.

Moulin Depo. 269:19-270:2.

186. Petitioner's Declarant specifically confirmed, consistent with my understanding, that all potential matches in the database are searched using Imamura's unsearched portions approach:

Q All right. So let's assume we've got our
database up. We've identified the unimportant
sections, and we're not going to assess those when
we're doing our Iwamura search.
Does that make sense?
A Yes.
Q In that case, failing to test a melody
against an unimportant part will not result in us
ignoring a match. Would you agree?
A That is correct, assuming it's truly an
unimportant part, yes.

Moulin Depo. 317:2-12.

Q Now, one of the features of Iwamura is an
alternative in which the database can be set up
where you're going to not -- you're going to skip
portions of the reference melodies. You're only
going to use what he identifies as the important
parts; right?
A Yes.
Q In that -- when that version of Iwamura is
used, is it the case that the Iwamura search will do
a comparison to each of the musical works that are
in the database?

Moulin Depo. 267:13-24.

187. Iwamura does not expressly state (nor is it inherent, *i.e.*, necessarily present) that flexibility on search area enables the disclosed search to entirely skip a record in the reference database. Each and every record in the reference database will be searched; therefore, the search is an exhaustive search rather than the claimed non-exhaustive search. Moreover, when a repeated pattern (*e.g.*, "second measure") is skipped, it is a "reasonable engineering assumption" that the search has "already tested" the repeated pattern and, as a result, all data is considered in the search. Moulin Depo. 279:7-14. Moreover, Iwamura states that each repeated portion can be pre-processed and is marked as such in the database. Iwamura, 9:39-42. Accordingly, the unsearched portion process constitutes extracting the features of the melodies to be compared and the resulting search searches all pre-processed data.

188. *Board's concerns*: I now address the Board's specific concerns (identified in its Decision) with respect to whether Iwamura discloses the claimed nonexhaustive search. In instituting Ground 1 of the '237 Petition, the Board determined that one feature of Iwamura identified by Petitioner the "computational limits" feature—discloses a non-exhaustive search because if the computation limit (comparing the notes in the work to be identified with a single record in the database) is reached, the entire search is stopped, independent of how many records in the database have actually been searched:

> Patent Owner also argues that Iwamura's computational limit does not create a nonexhaustive search because "it does not state or suggest that all records in the music library are not use[d] in the comparison." Prelim. Resp 18. We do not agree. If, in Iwamura, the computational limit is reached, the search is stopped, even if not all of the records have been searched. Per our construction of "nonexhaustive search," i.e., "a search that locates a match without a comparison of all possible matches," we are persuaded on this record that the process of Iwamura, with the computational limit, would prevent all of the records of the remote music database from being searched, but would ultimately provide a match because of the input fault tolerance process, discussed above. *See* Ex. 1012, 7:56–57, 9:20–34.

Decision ('237) at 11-12. In making this preliminary finding, it appears that the Board apparently confused:

(a) stopping an individual computation of the absolute difference between the notes in the work to be identified with a specific record in the database for a specific alignment of peak notes and then shifting the peaks to perform another peak comparison with that record, with

(b) stopping the entire search process altogether.

In my opinion, there are at least two reasons why the Board's preliminary interpretation of Iwamura is not correct.

189. <u>Reason 1</u>: Iwamura does not state (or even suggest or imply) that when a given computation (the absolute difference between the compared notes) based on comparing a work to be identified with a specific record in the database

exceeds a certain limit (demonstrating that the particular alignment of work to be identified with the specific record being searched is not a match) the entire search stops. Neither the Petition, Petitioner's Declarant, nor the Board points to such a statement in Iwamura, because one skilled in the art would understand that there is none. Rather Iwamura states that to accelerate comparing the peaks of the work to be identified with a single record in the database, the "computation of the total absolute difference" between the melody and a specific reference work based on that search can be stopped and shifted to the next comparison:

> To accelerate the search, computation of the total absolute difference can be stopped when it exceeds a certain limit.

Iwamura, 7:56-57.

190. The individual computation based on that particular alignment between the peak notes of the work to be identified and the record "can be stopped" when that individual computation exceeds a certain limit. The search process itself is not stopped but rather accelerated: "[t]o accelerate the search." "In this manner, the entered melody is shifted to each peak in each reference melody and compared. The reference melody that gives the least difference is returned as a search result. Iwamura, 6:31-7:55. The specific computation is stopped, not the search: "it would then shift this peak over to the next peak and start another calculation." Moulin Depo. 240:24-242:2.

191. Again, I note that, consistent with my understanding, Petitioner's Declarant confirmed that under the computation limits approach disclosed in Iwamura (as well as all other approaches in Iwamura), all potential matches are searched:

> Q. Is it true that for all the Iwamura searches, all the variations that Iwamura teaches, it always teaches doing a comparison to each of the musical works that's a possible match in our database?
>
> A. I don't think it says it explicitly every time. It's -- it's often implicit. It's understood that you search through every musical work in the database.

Moulin Depo. 269:19-270:2.

Q -- when we throw in this limit on the computation, we are going to still do a comparison to each of the works that are in our database; right?
A We do a comparison of some data within each musical work, yes.
Q You don't read this as saying that if we're doing a computation, and for one of them it exceeds a certain limit, then we stop the search altogether? It doesn't say that, does it?
A No. We move to the next one.
Q By "the next one," you mean the next computation?
A The next possible match. Like after the first line in your example, we move to the second one.

Moulin Depo. 243:17-244:7.

Q If we use this parameter in doing our search in Iwamura, is it the case that after we determine that one calculation should be stopped, then that we stop the search altogether, or do we keep calculating --
A No. We move to the next one. So like here, if the parameter value is 20, once we reach 20, we abandon this and move on to the next one.
Q Then after we complete the peak search analysis for a given work, then we go to the next work; is that right?
A Yes.

Moulin Depo. 242:19-243:5.

192. *Reason 2*: One skilled in the art would understand that the alternative (which is not disclosed in Iwamura)—that the entire search process stops when one peak search comparison between the work to be identified and one record in the database reaches a certain limit—would make the search process inoperable. The purpose of Iwamura is to find a match. Stopping the search when an individual computation exceeds a certain limit would prevent the search from finding a match. For example, assume that:

- there are 10 records in the dataset to be search;
- the computation based on the first peek note alignment between the work to be identified and the first record in the database exceeds the set limit.

Stopping the search at that point—after comparing the work to be identified with just the first alignment of the first record—would identify no match even if records 4, 7, and 8 were close matches and record 9 was an exact match. The system would be inoperable and would fail to identify matches if the search is stopped completely when a computational limit is reached rather than, as disclosed in Iwamura, the search moves on to (a) the next alignment of peak notes between the work to be identified and that same record in the database, or (b) the next potential record in the database to identify a match. Stopping the search when a given computation exceeds a certain limit will speed up comparing the work to be identified with a given record in the database but it does not stop the search process.

193. I note that the Board also noted that if Iwamura disclosed a search that is not a nonexhaustive search, this "does not end the inquiry"—Iwamura could still teach a nonexhaustive search as long as, in addition to disclosing other searches, Iwamura actually disclosed the claimed nonexhaustive search:

> We note that all of the independent claims of the '237 Patent utilize "comprising" language, such that those claimed methods and apparatuses do not exclude additional, unrecited elements or method steps. *See Mars Inc. v. H.J. Heinz Co.*, 377 F.3d 1369, 1376 (Fed. Cir. 2004). Thus, the scope of independent claim 25 can include an exhaustive search, as long as it performs a nonexhaustive search as well. Thus, even if Patent Owner is correct and a particular search in Iwamura is exhaustive, that does not end the inquiry.

Decision ('237) at 11. As I demonstrated above, one of ordinary skill in the art would understand that Iwamura does not disclose any nonexhaustive searches.

4. identify a neighbor / near neighbor (claims elements 1(b), 5(b), and 25(b)).

194. In instituting Ground 1, the Board did not specifically address whether Iwamura disclosed the neighbor or near neighbor properties of the claimed search. Decision ('237) at 11-12. As I demonstrated below, one skilled in the art would understand that Iwamura does not disclose such properties.

195. As I explained above, identifying "a neighbor" or "near neighbor" means identifying "a close, but not necessarily exact or closest, match." Decision ('237) at 8.

196. Iwamura does not disclose a search to identify a neighbor or near neighbor because, as I explained above, the disclosed search always identifies an exact or the closest match. Iwamura confirms that the disclosed search engine will find the "closest" match—the melody that gives the least difference. Iwamura, 9:54-55. Petitioner's Declarant also confirmed that Iwamura will either produce an "exact match" if it finds one, or the "best match it finds using that approximate criterion." Moulin Depo. 271:22-272:12.

197. One skilled in the art would understand that the system in Iwamura will always find the closest match, even if unimportant peaks are skipped or repeated patterns are avoided. At his deposition, Dr. Moulin agreed to my understanding:

- "[W]'re still going to be identifying the closest match" even when "the unimportant peaks are skipped…. Dropping an unimportant part is not going to affect the ability to find the best match." Moulin Depo. 317:14-22.

- "If we implement that feature of Iwamura… skipping a repeated pattern… It will not affect the ability to find the best match." Moulin Depo. 318:11-18.

Because the search algorithms disclosed in Iwamura necessarily return the closest match, they do not identify a match that is not necessarily the closest match, as the neighbor and near neighbor claim elements require. Iwamura does not disclose identifying a neighbor or near neighbor because the disclosed search always identifies an exact or the closest match.

198. The Petition, Declaration, and corresponding charts fail to demonstrate that Iwamura discloses the claimed neighbor or near neighbor searches.

199. The Petition does not address the "neighbor" concepts in the text of the Petition. In its Charts, to establish the claimed search "to identify a neighbor" (elements 1(b) and 5(B.2)) and search "to identify a near neighbor" (element 25(b)), Petitioner asserts:

Claim 1(b):

b) determining, by the computer system, an identification of the media work using the received features extracted from the media work to perform a sub-linear time search of extracted features of identified media works to identify a neighbor; and	Iwamura determines an identification of the media work using the extracted features by "find[ing] the closest melody from the database," which is a neighbor. 9:25-38, 12:1-2. Iwamura discloses searching using the "Boyer-Moore algorithm" (9:63-64, 10:1-3), which is sublinear (Ex. 1017 at 1). Ex. 1004 at ¶ 72.

Pet. ('237) at 10-11.

Claim 5(b.2):

2) determining, by the computer system, an identification of the media work using the features extracted from the media work to perform a sub-linear time search of extracted features of identified media works to identify a neighbor, and	Petitioner incorporates the above discussion of Iwamura regarding Claim 1b

Pet. ('237) at 12.

Claim 25(b): Petitioner incorporates its discussions regarding element 9(b) (the remaining discussion addresses the non-exhaustive component of the claim element):

b) determining, by the computer system, an identification of the media work using the media work extracted features to perform a nonexhaustive search of reference extracted features of reference media works to identify a near neighbor, and	Petitioner incorporates the above discussion of Iwamura regarding Claim 9b [illegible]

Pet. ('237) at 15.[19]

For claim element 9(b), Petitioner asserts:

[19] The referenced claim element 9(b) does not include a search "to identify a near neighbor" but instead includes "an approximate nearest neighbor search."

b) determining, by the computer system, an identification of the media work using the received features extracted from the media work to perform an approximate nearest neighbor search of extracted features of identified media works, and	Petitioner incorporates the above discussion of Iwamura regarding Claim 1b. Furthermore, Iwamura uses an approximate nearest neighbor "search engine [that] has . . input fault tolerance capability" (10:17-18), and skips "portions that should not be searched" (12:6-7), such as "repeated patterns" (9:36-44), and "unimportant portion[s]" of the melody (9:44-45).

Pet. ("237) at 12.

200. The Declaration is essentially the same.

Claim element 1(b):

b) determining, by the computer system, an identification of the media work using the received features extracted from the media work to perform a sub-linear time search of extracted features of identified media works to identify a neighbor, and	Iwamura discloses the use of a "search engine" to determine an identification of the media work using the extracted features by "find[ing] the closest melody from the database," which is a neighbor. 9:25-38, 12:1-2. Iwamura discloses searching using the "Boyer-Moore algorithm" (9:63-64, 10:1-3), which is sublinear (Ex. 1017 at 1).

Moulin Decl. ('277) ¶75.

Claim element 25(b) cross references claim element 9(b):

b) determining, by the computer system, an identification of the media work using the media work extracted features to perform a nonexhaustive search of reference extracted features of reference media works to identify a near neighbor; and	I incorporate my above discussion of Iwamura regarding Claim 9b. Iwamura further discloses using non-exhaustive search algorithms using "peak notes" (6:31-7:55), which "are approximately 20% of the total number of notes in a typical melody," meaning "search speed using peak notes is 20% of a brute force search" (9:9-10). The search is further non-exhaustive because it can be accelerated by stopping the search when computations "exceed[] a certain limit." 7:56-57.

Claim element 9(b):

b) determining, by the computer system, an identification of the media work using the received features extracted from the media work to perform an approximate nearest neighbor search of extracted features of identified media works, and	I incorporate my above discussion of Iwamura regarding Claim 1b. Furthermore, Iwamura discloses using an approximate nearest neighbor "search engine [that] has input fault tolerance capability" (10:17-18), and skips "portions that should not be searched" (12:6-7), such as "repeated patterns" (9:36-44), and "unimportant portion[s]" of the melody (9:44-45)

201. One skilled in the art would understand that these discussions and the cited passages from Iwamura do not demonstrate that Iwamura teaches a search that identifies a neighbor or near neighbor for the reasons that I set forth above.

202. First, the cited passage from element 1(b) does not disclose a search that identifies a neighbor or near neighbor. As I explained above, a search that identifies a neighbor or near neighbor is a search that identifies "a close, but not necessarily exact or closest, match." Decision ('237) at 8. The passage cited in the Petition and corresponding declaration confirms that the Iwamura searches find "the closest melody from the database." Pet. ('237) at 8 *(quoting* Iwamura, 9:24-35).

203. Second, the references to searches that have an "input fault tolerance" or skip "portions that should not be searched" (Pet. ('237) 13 quoting Iwamura 10:13-18, 12:6-7, 9:36-44, and 9:44-45) do not expressly or inherently disclose a search that does not necessarily identify the closest match. As I demonstrated above, the output from any disclosed Iwamura search always identifies the closest match and therefore is not a search that identifies a neighbor or near neighbor—"a

close, but not necessarily exact or closest, match." *See e.g.*, Iwamura, 11:43-45 ("The invented input fault tolerance function allows the user to obtain an exact result even when an entered melody has some errors.").

5. sublinear approximate nearest neighbor search (claim element 33(b)).

204. Claim 33 requires a search that is both (a) a sublinear, and (b) an approximate nearest neighbor search.

205. One skilled in the art would understand that Iwamura does not disclose a "sublinear approximate nearest neighbor search" for at least two independent reasons.

206. Reason 1: As I demonstrated above (with respect to claim elements 1(b) and 5(b.2)), Iwamura does not disclose a "sublinear" search.

207. Reason 2: Also as I demonstrated above (with respect to claim elements 9(b) and 13(b.2)), Iwamura does not disclose an "approximate nearest neighbor search."

208. The Petition, Declaration, and corresponding charts fail to demonstrate that Iwamura discloses the claimed "sublinear approximate nearest neighbor search." For claim 33, the Petition and corresponding Declaration do not address the "sublinear approximate nearest neighbor search" in their respective

texts but instead exclusively cross-reference their respective charts for Claims 1(b) and 9(b).

Petition:

b) determining, by the computer system, an identification of the media work using the media work extracted features to perform a sublinear approximate nearest neighbor search of reference extracted features of reference identified media works, and	Petitioner incorporates the above discussion of Iwamura regarding Claims 1b and 9b

Pet. ('237) at 16.

Declaration:

b) determining, by the computer system, an identification of the media work using the media work extracted features to perform a sublinear approximate nearest neighbor search of reference extracted features of reference identified media works; and	I incorporate my above discussion of Iwamura regarding Claims 1b and 9b

Moulin Decl. ('237) ¶75. As I demonstrated above, the cross-referenced "discussions" and citations to Iwamura fail to demonstrate that Iwamura discloses either a "sublinear" search or an "approximate nearest neighbor search." Accordingly, the Petition fails to satisfy its burden for these two independent reasons.

209. Board's concerns: I addressed the Board's concerns with respect to the "sublinear" component above in Section VI(A)(1). I addressed the Board's

concerns with respect to the "approximate nearest neighbor component" above in Section VI(A)(2).

B. '237 Ground 2: The instituted claims of the '237 patent are not anticipated by Ghias.

210. The Board instituted Ground 2 based on the following: Claims 1–3, 5–7, 9–11, 13–15, and 21–24 as unpatentable under 35 U.S.C. § 102(b) as anticipated by Ghias. Decision ('237) at 21 (I underlined the independent claims). Ground 2 fails because Ghias does not disclose the following key elements from each instituted independent claim:

- sub-linear time search (claim elements 1(b) and 5(b.2)); and
- approximate nearest neighbor search (claim elements 9(b) and 13(b.2)).

I address each in turn below.

1. sublinear time search (claim elements 1(b) and 5(b.2)).

211. Claims elements 1(b) and 5(b.2) require a "sub-linear time search."

212. As I explained above, one of ordinary skill in the art would understand that a "sub-linear time search" is "a search whose execution time scales with a less than linear relationship to the size of the data set to be searched." Decision ('237) at 7.

213. Ghias does not disclose a "sub-linear time search" search but instead teaches a linear search in which the search time grows linearly in relationship to the size of the data set. The searches disclosed in Ghias compare the work (user input 23) with "all the songs" in the library (*i.e.*, what the Petition calls "all possible matches," Pet. 6):

> In order to search the database, songs in the database **14** are preprocessed to convert the melody into a stream of the previously discussed U,D,S characters, and the converted user input (the key **23**) is compared with all the songs.

Ghias, 5:66-6:2.[20] If an increase in a given variable increases the execution time of a given algorithm by an amount that is only a constant multiple of the amount by which that variable was increased, irrespective of the initial value of that variable, then that algorithm scales linearly with regard to that variable.[21] More specifically,

---

20 To compare the "user input" with "all the songs," Ghias must compare the user input with every song in the data set. Ghias does not disclose a search algorithm that does not compare the work to be identified with every record in the data set.

21 As I explained above, linearity describes "[t]he relationship existing between two quantities when a change in a second quantity is directly proportionate to a change in the first quantity." Ex. 2007 (Modern Dictionary of Electronics) at 425 (1999).

if f(n1, …, ni, …, nk) is a function that describes the execution time of an algorithm where variables n1, …, ni, …, nk are the sizes of the different types of data on which the algorithm operates, then if f(n1, …, ni+q, …, nk) = f(n1, …, ni, …, nk) + f(n1, …, q, …, nk), then that algorithm scales linearly with regards to variable ni. Because a constant increase in the size of the data set (*i.e.*, number of records in the reference data set) increases the execution time of the Ghias search algorithm by a constant amount that does not depend on the initial size of the data set, Ghias discloses a linear time search, not a sub-linear time search.

214. In addressing "the problem of approximate string matching," Ghias identifies "the running times of several algorithms:

> Several Algorithms have been developed that address the problem of approximate string matching. Running times have ranged from O(mn) for the brute force algorithm to O(kn) or O(nlog(m)), where "O" means "on the order of," m is the number of pitch differences in the query, and n is the size of the string (song).

Ghias, 6:23-28. In each instance, the running time of the identified search is linear (not sub-linear) with respect to the size of the data set.

215. As clarified in this passage from Ghias:

- "m is the number of pitch differences in the query" corresponding to the length of the query of the work to be identified (highlighted in green); and

- "n is the size of the string (song)" (highlighted in orange); Moulin Depo. 88:13-15.[22]

- "k" refers to the number of mismatched characters permitted in the search results returned by the search: "The problem consists of finding all instances of a pattern string P = p1, p2, p3 . . . pm in a text string T = t1, t2, t3 . . . tn such that there are at most k mismatches (characters that are not the same) for each instance of P in T." Ghias, 6:37–41; Moulin Depo. 96:2-15.

- In the field of computer software, "O" indicates big O notation. Big O notation describes the relationship between an algorithm's execution time and other variables. In computer science, big O notation is used to describe how algorithms respond (*e.g.*, in their processing time or working space requirements) in the worst-case to changes in input size. Ex. 2009 (http://en.wikipedia.org/wiki/Big_O_notation.); Moulin Depo. 16:13-24 (in the field, there is "a common system of notation that's used … when we're taking about how the search time or execution time scales with respect to the size of the database – it's the so-called order of notation … sometimes referred to as the 'big O notation.'")

216. As I explained above, the disclosed searches may be sublinear with respect to "m … the number of pitch differences in the query." O((nlog(m))) is

---

22 Referring to "n" as part of the dataset to be searched (rather than the query of the work to be identified) is standard in the field. Moulin Depo. 18:2-10.

sublinear with respect to "m" (corresponding to the length of the query) because the running time is a function of log(m)). *See* Moulin Depo. 102:9-13.[23]

217. The disclosed searches are never sublinear with respect to "n"—"the size of the string (song)" or the number of records in the data set ("N").[24] As I explained above, if a constant increase in a given variable increases the execution time of a given algorithm by a constant amount, then that algorithm scales linearly with regard to that variable. An incremental increase in the number of records in the data set, or even in the length of a given reference record ("n") in the data set, increases the execution time of every search disclosed by Ghias by a constant amount.

218. O(mn), O(kn), and O(nlog(m)) all describe algorithms whose execution times increase by a constant amount as the length of the record being searched is incrementally increased. The first to run times—O(mn) and O(kn)—are linear with respect to the size of the data set being searched. My

---

[23] "log" stands for taking the logarithm of the following variable; so log(m) means the logarithm of m.

[24] As I noted above, in the IPR Patents, consistent with the literature, the size of the dataset is referred to as "N" where "N" is the number of records in the dataset.

understanding, consistent with the understanding of one skilled in the art, is confirmed by Petitioner's Declarant. *see, e.g.*

> Q. Let's assume we have a search that's execution time is O(mn), where M means the size of the query, N means the size of the database or dataset that we're searching over.
>
> What does that tell us?
>
> A. Well, it means that the search time, [illegible], grows at most linearly.
>
> Q. Is it the case that this would be a sublinear search as it's used in the '137 patent?
>
> A. No. It says, again, it's at most linear in terms of M times N.

Moulin Depo. 28:17-29:2.[25] The third run time—O(nlog(m))—may be sub-linear with respect to the number of pitch differences in the query "m" but is always linear with respect to "n," the size of the string (song) being searched, or the number of records in the dataset being searched. Again, my understanding, consistent with the understanding of one skilled in the art, is confirmed by Petitioner's Declarant *see, e.g.*:

25 Petitioner's Declarant uses "data set" and "database" interchangeable in this context. Moulin Depo. 22:14-16.

If we have a search that is described as set forth in Formula 6, where N refers to the size of the database, what does that tell us about whether it's linear or sublinear?
A Same answer as previously. You have replaced f(m) with log(m). It's just a particular function, f(m). So that would tell me that search time is at most linear in terms of N.
Q If you're presented with that, does that tell you that we have a sublinear search?
A In terms of N, yes.
Q In terms of --
A Sorry. Let me rephrase this.
It's at most linear.
Q Does that tell us that we have a sublinear search?
A It does not tell me that, because of the meaning of the order of notation. It says, at most, linearly.

Moulin Depo. 36:20-37:13.[26]

219. Accordingly, Ghias exclusively disclose searches that are linear—not sublinear—in relationship to the data set to be searched. My understanding is again confirmed by Petitioner's Declarant:

---

26 These "running times" are the times it takes to run a query of length "m" against one record of the length "n" in a dataset including N records. The search time for running the same query against the full dataset would take on average N time longer, since each record in the dataset will need to be searched.

Q Does the information that's presented here suggest to you that these -- that these algorithms are sublinear with respect to the size of the dataset being searched?

A It does not say that. First, there's no database. Okay? Again, this refers to a single song.

Moulin Depo. 88:22-89:3.

Q Reading this, do you interpret this to suggest that these algorithms are sublinear with respect to the size of the dataset being searched?

A No. This, again, means "of the order of." So it means at most linear.

Moulin Depo. 89:14-18.

Q My question, sir, is did you read this to indicate or suggest that these algorithms would -- could be used to perform a sublinear time search with respect to the size of the dataset being searched?

A No.

Moulin Depo. 90:16-21; 93:24-94:5; 98:20-25; 100:8-11; 142:5-10.

220. Petitioner's Declarant confirmed my understanding—that:

(a) any sub linearity referenced in Ghias is with respect to "m"—the number of pitch differences in the query, not "n" the size of the string (song) or the size of the data set (N);

(b) Ghias does not state or suggest that the size of the query is dependent on the size of the data set;

(c) any sub linearity with respect to the query "is not relevant" to the '237 patent, and

(d) as a result, Ghias does not disclose a search that is sublinear with respect to the size of the data set—the relevant sub-linearity inquiry for the '237 patent.

Moulin Depo. 152:20-154:2 (any sub-linearity with respect to the query "is not relevant.")

221. In reviewing Dr. Moulin's deposition, I observed that Petitioner's Declarant, Dr. Moulin, testified that:

(1) he clearly understood that sub-linear in the context of the '237 patent is based on the size of the data set searched, not the size of the query or the pitch differences in the query;

(2) Ghias does not identify a search that is sub-linear with respect to the data set; and

(3) when he wrote in his Declaration that "Ghias discloses searches … which … are sublinear," he did not intend the Board to interpret "sublinear" to be in the context of the '237 patent but rather in a different context irrelevant to the '237 patent.

222. As I noted above, Petitioner's Declarant understood that, consistent with my understanding, "sublinear" in the context of the '237 patent ("a concept that's common in [his] field" (Moulin Depo. 8:10-14)) is based on the size of the data set searched (N), not the size of the query or pattern to be matched ("m"):

> 53. I understand and agree with Petitioner's position that the term "sublinear search" means "a search whose execution time has a sublinear relationship to database size." For instance, a linear search of a 200-item database would take twice as long as a linear search of a 100-item database. By contrast, a sublinear search of a 200-item database would take less than twice as long as a sublinear search of a 100-item database, perhaps, for instance, 1.5 times as long.

Moulin Decl. ('237) ¶53.

Is it the case that for a search to be
sublinear as it's used in the '237 patent, it's not
enough for it to have execution time that is
sublinear in relationship to the size of the
pattern; it must also be sublinear in relationship
to the size of the database?
A When I read "sublinear" in, say, Claim 5
of the patent, as we just did, I understand
sublinear to mean in relation with the size of the
database. It does not say anything about in
relation with the size of the query.

Moulin Depo. 26:11-21.

Q Is it the case that in doing your analysis
to determine whether or not prior art anticipated,
you applied the definition that a sublinear search
or a sublinear time search was one whose execution
time has a sublinear relationship to the database
size?
A That is correct, yes.
Q Is that the correct definition that should
be applied, or are you applying the wrong
definition?
A In my opinion, this is the correct
definition to be used.

Moulin Depo. 24:1-12.

Q Well, my question is if we're trying to
determine whether or not a given search is sublinear
with respect to the size of the database, that means
that we're going to determine execution time
compared to the growth of the database, is that right?
A Yes.

Moulin Depo. 31:13-18.

Q Well, as you understood the term
"sublinear" as it's used in the patent, it's about
the execution time as we increase the size of the
database; is that right?
A In the patent, yes.
Q Yes.
A Yes.

Moulin Depo. 103:16-22.

223. Petitioner's Declarant agreed with my understanding that the algorithms disclosed in Ghias do not disclose a search that is sublinear with respect to the size of the data set but instead has a sub-linear relationship to "m" the pitch differences in the query pattern:

Q When you did that, you knew that what was relevant was sublinearity with respect to the size of the dataset; right?

A That is always the assumption, that -- there are two notions of sublinearity which should not be confused.

And one notion, which is relevant to these proceedings, is sublinearity with respect to the size of the dataset. And then another notion, which occurs in Ghias in the references, is in comparison with brute-force search.

And I reiterate they are not the same

Moulin Depo. 158:14-159:4.

Q Did you ever read this and say to yourself, "What it says here about these algorithms, in this description of these algorithms, tells me that we have a sublinear time search"?

A No. I don't represent that, no.

Moulin Depo. 91:18-22.

Q -- were you trying to convey that these searches here in Ghias were sublinear with respect to the size of the dataset?

A No. No. The discussion there was relative to brute-force search. And it is only sublinear -- it is known to be sublinear in that sense. I do not know whether it would be sublinear in the size of the database for that particular algorithm.

Moulin Depo. 108:9-17

224. Petitioner's Declarant testified that when he wrote the following paragraph in his Declaration (Moulin Decl. ('237) ¶123):

> In particular, Ghias discloses searches whose execution times are proportional to the logarithm of the size of the data set (*Id.* at 6:24-28 ("O(kn) or O(nlog(m)")), which, as explained above in Section V(D), are sublinear (Ex. 1001 at 8:54-63).

he really meant sublinear with respect to the query, not the database or data set being searched:

> Q. Is it the case that Ghias discloses a
> search whose execution time is proportional to the
> log of the size of the dataset being searched?
> A. No. Okay. Again, if you say the dataset
> being searched, that would be N. Okay?
> What I intended there was the query
> dataset, which is M.
> Q. So when you wrote "proportional to the log
> of the size of the dataset," you meant that to mean
> the query dataset?
> A. Yes. Yes.
> Q. Not the -- not the database?
> A. That's -- that's correct. There's no
> database here. It's a problem of matching a query
> to a single song.

Moulin 103:1-15.

Q And by "logarithm of the dataset," you mean logarithm of the size of the query, not the dataset to be searched; is that true?
A I believe we discussed that earlier. Okay. Should have -- here it is the log, indeed, of the query dataset. So log(m).
Q So it should -- this should be interpreted -- when you write "dataset," you're -- what you mean is the query dataset, not the dataset to be searched?
A That's correct. So the logarithm of the dataset is log(m), which is written below. Yes.
Q Which is the query dataset?
A That's right.

Moulin 154:14-155:2

Q Why is it -- why didn't you think it was relevant to tell the Board that information? What would they do with it?

MR. ELACQUA: Calls for speculation. Objection.

THE WITNESS: I don't remember why I wrote it that way. Again, I agree it could be written in a more precise way. So it should have been the log of the query dataset. The word "query" should have appeared there.

BY MR. DOVEL:

Q Why don't you write that in so we don't have that confusion.

MR. ELACQUA: Again, I'm going to object.

THE WITNESS: I've said it. So the word "query" should be included before "dataset" in the box on line 4.

BY MR. DOVEL:

Q Why do you object to writing it in there?

A Why should I write it? I just said it.

Moulin Depo. 155:12-156:6.

A Because it's very clear. I'm saying the word "query" should have been incorporated before "dataset."

Moulin Depo. 156:22-157:3.

> Well, one possible reason you wanted to give the Board this information is so that they would misread it and be misled.
> That's a possibility; right?
> A It's not at all the -- the reason. So it alone -- there are four documents. Some, they are, like, 90 pages each. Some of the words could have been better chosen. In particular, the word "query" should have been there. I have acknowledged that.
> Q But assume --
> A Again, I have acknowledged that this was not written the best way.

Moulin Depo. 157:6-18. I agree with Dr. Moulin—that the referenced passages in Ghias do not disclose a sublinear search with respect to the size of the dataset.

225. Petitioner fails to satisfy its burden of demonstrating that Ghias teaches a "sub-linear time search." As support for the "sub-linear" elements, Petitioner (and corresponding Declaration) exclusively rely on the statement addressed above—that Ghias discloses "searches whose execution times are proportional to the logarithm of the size of the data set" based on the disclosed running times of O(kn) or O(nlog(m)). Pet. ('237) at 41 (*quoting* Iwamura 6:23-35 and 6:24-28):

226. <u>Petition</u>:

Independent claims 1 and 5 of the '237 patent further require that the search be "sublinear." Ex. 1001 at Claims 1, 5. Ghias discloses search algorithms that are substantially faster than "brute force" searches. Ex. 1010 at 6:23-35. In particular Ghias discloses searches whose execution times are proportional to the logarithm of the size of the data set (*id.* at 6:24-28 ("O(kn) or O(nlog(m)")), which, as explained above in Section V(D), are sublinear (Ex. 1001 at 8:54-63). Ex. 1004 at ¶ 123.

Pet. ('237) at 41.

227. <u>Chart in Petition</u>:

<u>Claim 1(b)</u>:

b) determining, by the computer system, an identification of the media work using the received features extracted from the media work to perform a sub-linear time search of extracted features of identified media works to identify a neighbor; and	Ghias determines the identification of a media work by "search[ing] the melody database" (2:50-59) to locate matching "sequence[s] of digitized representations of relative pitch differences," i.e., extracted features (Abstract). This is sublinear because execution time is proportional to the logarithm of the data set. 6:24-28 ("O(nlog(m)"). Ex. 1004 at ¶ 123. This identifies a list of neighbors, i.e., "a ranked list of approximately matching melodies, as illustrated at 26" or "the single most approximate matching melody." 2:50-59, 6:60-63

Pet. ('237) at 42-43.

<u>Claim 5(b.2)</u>:

2) determining, by the computer system, an identification of the media work using the features extracted from the media work to perform a sub-linear time search of extracted features of identified media works to identify a neighbor; and	Petitioner incorporates the above discussion of Ghias regarding Claim 1b.

Pet. ('237) at 44.

228. Declaration:

> 123. It is my opinion that Ghias further discloses the elements of claims 1 and 5 of the '237 patent that require that the search be "sublinear." Ex. 1001 at Claims 1, 5. In particular, Ghias discloses search algorithms that are substantially faster than "brute force" searches. Ex. 1010 at 6:23-35. In particular, Ghias discloses searches whose execution times are proportional to the logarithm of the size of the data set (*id.* at 6:24-28 ("O(kn) or O(nlog(m)")), which, as explained above in Section V(D), are sublinear (Ex. 1001 at 8:54-63).

Moulin Decl. ('237) ¶123 (the paragraph addressed above).

229. Declaration Charts:

Claim 1(b):

b) determining, by the computer system, an identification of the media work using the received features extracted from the media work to perform a sub-linear time search of extracted features of identified media works to identify a neighbor; and	Ghias discloses determining the identification of a media work by "search[ing] the melody database" (2:56-59) to locate matching "sequence[s] of digitized representations of relative pitch differences," i.e., extracted features (Abstract). Ghias further discloses that this search is sublinear because its execution time may be proportional to the logarithm of the data set. 6:24-28 ("O(nlog(m)"). Ghias further discloses that this search identifies a list of neighbors, i.e., "a ranked list of approximately matching melodies, as illustrated at 26" or "the single most approximate matching melody." 2:56-59; 6:60-63.

Moulin Decl. ('237) ¶127

Claim 5(b.2):

2) determining, by the computer system, an identification of the media work using the features extracted from the media work to perform a sub-linear time search of extracted features of identified media works to identify a neighbor, and	I incorporate my above discussion of Ghias regarding Claim 1b

Moulin Decl. ('237) ¶127.[27]

230. As I explained above in detail, this discussion and the passage from Ghias quoted in the Petition and Declaration exclusively address sub-linearity with respect the number of pitch differences in the query ("m"), not the "size of the string (song)" ("n") (Ghias, 6:23-28), much less the size of the data set being searched ("N"), as required by a sub-linear search in the context of the '237 patent. Accordingly, although each individual comparison can be more efficient using the searches disclosed in Ghias, the computational time it takes to search the database always grows linearly with the size of the dataset. As a result, the disclosed searches in Ghias are linear, not sublinear.

231: *Board's concerns*: I now addresses the Board's specific concerns (identified in its Decision in the '237 IPR) with respect to whether Ghias discloses the claimed "sub-linear time search." In instituting Ground 1, the Board

[27] Petitioner's expert confirmed that the other passages that he cites relating to other claim elements do not disclose a search that is sublinear. *See, e.g.*, Moulin Depo. 151:1-5; 151:6-12; 152:3-9 (addressing Ghias 2:50-59).

preliminary found that Ghias disclosed the claimed "sublinear time search" based on the disclosed "sub-linear approximate string matching" disclosed in Ghias:

> Additionally, as we found above, the claims do not specify that the sub-linear search must be performed on a subset of all of the records, and not information within individual records. As such, we are persuaded that the sub-linear approximate string matching, in Ghias, satisfies the claimed recitation of "using the received features extracted from the media work to perform a sub-linear time search of extracted features of identified media works to identify a neighbor."

Decision ('237) at 18-19.

232. As I demonstrated above, however, the "approximate string matching" algorithms disclosed in Ghias are only sub-linear with respect to the "m … the number of pitch differences in the query" not "n… the size of the string (song))" or with respect to N, the size of the dataset:

> Several Algorithms have been developed that address the problem of approximate string matching. Running times have ranged from O(mn) for the brute force algorithm to O(kn) or O(nlog(m), where "O" means "on the order of," m is the number of pitch differences in the query, and n is the size of the string (song).

Ghias, 6:23-28.

233. As I noted above, Petitioner's Declarant confirmed that Ghias, and the approximate string matching algorithms disclosed in Ghias, do not disclose a sub-linear search with respect to the size of the data set—the relevant inquiry in the

context of the '237 patent. Accordingly, the premise underlying the Board's preliminary finding—that the approximate string matching algorithms disclosed in Ghias have sub-linear properties with respect to the dataset—is wrong.[28]

2. approximate nearest neighbor search (claim elements 9(b) and 13(b.2)).

234. As I explained above, an "approximate nearest neighbor search" is a sublinear search identifying a close match that is not necessarily the closest match." Section V(D); Decision ('237) at 9.

235. One skilled in the art would understand that Ghias does not disclose the claimed "approximate nearest neighbor search" for at least two independent reasons.

236. Reason 1: One skilled in the art would understand that Ghias does not disclose an "approximate nearest neighbor search" because Ghias does not disclose "identifying a close match that is not necessarily the closest match."

237. To disclose an approximate nearest neighbor search, Ghias must disclose a search that does not necessarily find the closest match. *See* Section

[28] I note that Petitioner's Declarant also confirmed that searching a subset of information within individual records (*e.g.*, not looking at "every single character in the dataset") does not establish a sub-linear search. Moulin Depo. 37:18-38:5.

V(D). "A nearest neighbor search always finds the closest point to the query. An approximate nearest neighbor search does not always find the closest point to the query. For example, it might do so with some probability, or it might provide any point within some small distance of the closest point." '237, 9:12–19. A search that always (necessarily) identifies an exact or the closest match is not an approximate nearest neighbor search because a neighbor search identifies a "close, but not necessarily exact or closest, match." Section V(D); Decision ('237) at 8.

238. Ghias discloses a search algorithm that necessarily finds the closest match. Ghias does not expressly disclose a search that does not necessarily identify an exact or closest match. And one skilled in the art would understand that such a search is not inherent (necessarily present) in Ghias.

239. Ghias teaches a search that generates three possible outputs:

(1) an exact match (Ghias 2:53-59 ("exact matching melody"));

(2) a "ranked list of approximately matching melodies" (Ghias, 2:50-59; Ghias, 6:60-63 ("a list of songs ranked by how well they matched the query"); Moulin Depo. 118:9-22); or

(3) "the single most approximate matching melody" (Ghias, 2:50-59).

Petitioner's Declarant confirmed, consistent with my understanding, that Ghias teaches these three potential outputs:

> Would you agree that Ghias teaches sort of three options? We've got -- it's going to produce the exact match; it's going to produce the most approximate matching; or it's going to produce this ranked list?
>
> A Yes.

Moulin Depo. 341:16-21.

240. For all three outputs, Ghias always identifies an exact or the closest match:

241. (1) exact match: If the search produces an exact match, it necessary produces an "exact or closest, match" and therefore does not disclose an "approximate nearest neighbor search." Petitioner's Declarant agreed with my understanding:

> If it produces the exact matching melody, would you agree that Ghias teaches providing or identifying the closest match?
>
> A Yes.

Moulin Depo. 341:23-342:1.

242. (2) ranked list: If the search produces a ranked list, it necessarily identifies as part of the ranked list either an exact match (if there is one) or the closest match—*i.e.*, the top ranked match—and therefore does not disclose an "approximate nearest neighbor search" that does not necessarily identify an exact or the closest match. At the top of the ranked list (*i.e.*, the number 1 ranked match

in the list) is an exact or the closest match. For example, assume for illustrative purposes that the work to be identified is 500. Assume that the list outputs in ranked order:

1st closest: 502
2nd closest: 510
3rd closest: 530 and
4th closest: 570.

The ranked list identified the closest match as 502. The closest match will never be excluded from the list of matches returned. As another example, assume that the list outputs in ranked order:

1st closest: 500
2nd closest: 510
3rd closest: 530 and
4th closest: 570.

In this example, the ranked list identified an exact match as 500. The exact match will never be excluded from the list of matches returned.[29] Accordingly, this

---

29 The list of matches within a given error-tolerance includes the full list of matches except those matches outside a given error-tolerance. Because the closest match is among the matches retrieved from the database, and the closest match is

approach necessarily identifies the closest match and therefore is not an "approximate nearest neighbor search."

243. Petitioner's Declarant confirmed, consistent with my understanding, that the ranked list approach identifies the closest match:

Q. Would you agree that for the embodiment in [illegible] that produces a ranked list, that the output is going to identify the match that has been deemed by the algorithm to be the closest match?
A. Yes.
Q. It doesn't merely say, "Here's a set of matches that are close"; it's going to identify the one that it has deemed to be the closest match?
A. So in the embodiment where it outputs a list, you are saying?
Q. Yes.
A. So it will output the list of what it believes it deems are the closest matches.
Q. And in that list, it's also going to identify the one that is the closest match?
A. That is deemed by the algorithm to be -- by the approximate algorithm to be the closest match.
Q. Yes?
A. Yes.

Moulin Depo. 356: 2-21.

---

not subsequently excluded from that list, the closest match will always be among the list of matches returned.

244. (3) <u>single most approximate matching melody</u>: If the search identifies the single most approximate matching melody, it necessarily identifies the closet match and is therefore not the claimed "approximate nearest neighbor search." Petitioner's Declarant agreed with my understanding:

> Q If it returns -- if it identifies anything as a -- as a match, it's going to be identifying the closest possible match; right?
>
> A Yes.

Moulin Depo. 345:16-346:11.

245. Petitioner's expert confirmed that for all three outputs, Ghias teaches a system that will always (necessarily) identify the closest match. Moulin Depo. 352:22-353:2. Accordingly, for all three potential outputs, Ghias necessarily identifies an exact or the closest match. Ghias does not disclose an "approximate nearest neighbor search" which identifies "a close, but not necessarily exact or closest, match."

246. <u>Reason 2</u>: Ghias does not disclose an "approximate nearest neighbor search" because Ghias does not disclose a sublinear search. As I explained above, an "approximate nearest neighbor search" is "one example" of a "sublinear search." Section V(D). Also, as I demonstrated above, Ghias does not disclose a "sublinear search." Section VI(B)(1). Accordingly, Ghias does not disclose the claimed "approximate nearest neighbor search."

247. The Petition, Declaration, and corresponding charts fail to demonstrate that Ghias discloses the claimed "approximate nearest neighbor search." The Petition and corresponding declaration assert that Ghias discloses the "approximate nearest neighbor search" because it produces:

(1) "a ranked list of approximately matching melodies" (labeled ❶); or

(2) "the single most approximate matching melody"(labeled❷):

248. Petition:

> Claims 9 and 13 of the '237 patent further require that the search locate an "approximate nearest neighbor." Ex. 1001 at Claims 9, 13. Ghias discloses that this search locates a neighbor by determining "a ranked list ❶ of approximately matching melodies, as illustrated at 26" or "the single most ❷ approximate matching melody." Ex. 1010 at 2:50-59, 6:60-63; Ex. 1004 at ¶ 124.

Pet. ('237) at 42.

249. Petition Charts:

Claim 9(b):

b) determining, by the computer system, an identification of the media work using the received features extracted from the media work to perform an approximate nearest neighbor search of extracted features of identified media works; and	Petitioner incorporates the above discussion of Ghias regarding Claim 1b. This may be an approximate neighbor search that generates "a ranked list of approximately ❶ matching melodies, as illustrated at ❷ 26" or "the single most approximate matching melody." 2:50-59, 6:60-63.

Pet. ('237) at 45.

Claim 13(b.2) (referencing claim element 9(b)):

2) determining, by the computer system, an identification of the media work using the received features extracted from the media work to perform an approximate nearest neighbor search of extracted features of identified media works, and	Petitioner incorporates the above discussion of Ghias regarding Claim 9b.

Pet. at 46.

250. Declaration:

> 124. It is my opinion that Ghias further discloses the elements of claims 9 and 13 of the '237 patent that require that the search locate an "approximate nearest neighbor." Ex. 1001 at Claims 9, 13. In particular, Ghias discloses that the search locates a near or nearest neighbor by determining "a ranked list ❶ of approximately matching melodies, as illustrated at 26" or "the single most ❷ approximate matching melody." Ex. 1010 at 2:50-59, 6:60-63.

Moulin Decl. ('237) ¶124.

251. Declaration Charts:

Claim 9(b):

b) determining, by the computer system, an identification of the media work using the received features extracted from the media work to perform an approximate nearest neighbor search of extracted features of identified media works, and	I incorporate my above discussion of Ghias regarding Claim 1b. Ghias further discloses that this may be an approximate neighbor search that ❶ generates "a ranked list of approximately matching melodies, as illustrated at 26" or "the single most approximate ❷ matching melody." 2:50-59, 6:60-63.

Moulin Decl. ('237) ¶127.

Claim 13(b.2) referencing claim element 9(b)):

2) determining, by the computer system, an identification of the media work using the received features extracted from the media work to perform an approximate nearest neighbor search of extracted features of identified media works, and	I incorporate my above discussion of Ghias regarding Claim 9b.

Moulin Decl. ('237) ¶127.

252. One skilled in the art would understand that neither of the cited passages discloses the claimed "approximate nearest neighbor search" because, as described above, both the ranked list and single most approximate matching melody always identify the closest match. I address each passage in turn:

253. Passage 1:

> The query engine **24** searches the melody database **14** and outputs a ranked list of approximately matching melodies, as illustrated at **26**. A preselected error tolerance may be applied to the search. The query engine **24** may of course alternatively be programmed to output the single most approximate matching melody or, if desired, to output an exact matching melody. However, by searching for an approximate matching melody, as hereinafter discussed, various forms of anticipated errors may be taken into account.

Ghias, 2:50-59. As noted in the Petition and Declaration, this passage states that the search "outputs a ranked list of approximately matching melodies, as illustrated at 26" or "the single most approximate matching melody." As I explained above, neither approach discloses the claimed "approximate nearest neighbor search." An "approximate nearest neighbor search" must identify "a close, but not necessarily exact or closest, match" Section V(D); Decision ('237) at 8. Both outputs disclosed in this passage necessarily disclose an exact or the closest match and, therefore, are not an "approximate nearest neighbor search."

254. Passage 2:

> The computer 16 may desirably be programmed so that, for a given query, the database 14 returns a list of songs ranked by how well they matched the query, not just one best match.

Ghias, 6:60-63. This passage also does not disclose a neighbor search. As I explained above, a "list of songs ranked by how well they matched the query" necessarily identifies an exact or the closest match, and specifically identifies such a song as the top-ranked song.

255. Moreover, under the proper construction of "approximate nearest neighbor search," the search must be a sub-linear search. '237, 9:12-19 (an approximate nearest neighbor search is an "example of a sub-linear time search"); Section V(D). As demonstrated above, these passages disclose a linear (rather than sublinear) search.

256. *Board's concerns*: I now address the Board's specific concerns (identified in its Decision) with respect to whether Ghias discloses the claimed "approximate nearest neighbor search." In instituting Ground 2 of the '237 IPR, the Board found that Ghias disclosed the "approximate nearest neighbor search"

because the error-tolerance search disclosed in Ghias "allows the user to identify sets of songs that contain similar melodies:"

> Ghias provides that "[t]he number of matches that the database 14 should retrieve depends upon the *error-tolerance* used during the key-search," and "the user can perform a *new query on a restricted search list consisting of songs just retrieved*. This allows the user to identify sets of songs that contain similar melodies." Ex. 1010, 6:63–65, 7:5–8 (emphases added). Thus, Ghias makes clear that the search need not be exhaustive, as Patent Owner has argued, and will act to "identify[] a close, but not necessarily exact or closest, match," per our claim construction.

Decision ('237) at 18-19. The Board did not explain, however, how "Ghias makes clear" that the search in Ghias will "identify[] a close, but not necessarily exact or closest, match" as required by an "approximate nearest neighbor search."

257. The Board noted that using an "error-tolerance," the user can adjust the number of output matches ("The number of matches that the database 14 should retrieve depends upon the error-tolerance used during the key search." Ghias, 6:63-65); and a new query can be performed on the restricted list ("If the list is too large, the user can perform a new query on a restricted search list consisting of songs just retrieved." Ghias, 7:5-8). But nothing in these passages or anywhere else in Ghias states or even suggests that the output of the initial list or the output of the restricted search will "identify a close, but not necessarily exact or closest, match." As I explained above, no such search is expressed in Ghias or is

inherent (*i.e.*, necessarily present). Rather, the search will always ("necessarily") identify an exact or closest match. Accordingly, Ghias does not disclose the claimed "approximate nearest neighbor search."[30]

C. '237 Ground 3: The instituted claims of the '237 patent are not obvious over Iwamura and Chen.

258. It is my understanding that if a combination of two references fails to teach an important claimed element, it is not possible for that combination to render the claim obvious. That is, assuming one of ordinary skill would have thought to combine prior art references, those references would still be missing an important element and therefore, even with the combination, one of ordinary skill would still not possess the invention.

259. Any combination of Iwamura with Chen would still be missing the same elements addressed above in Ground 1.

260. Ground 3 is directed to only dependent claims 26 and 27 which depend either directly or indirectly on independent claim 25; and claims 34 and 35 which depend either directly or indirectly on independent claim 33. Pet. ('237) at

---

30 An approximate nearest neighbor search could miss one or more of the closest matches in the returned search results. The searches disclosed in Ghias never purport to miss one or more of the closest matches in the returned results.

53-56; Decision ('237) at 22. Ground 3 presents two alternative grounds—that the dependent claims "are obvious over Iwamura alone, or alternatively, over Iwamura in view of Chen." Pet. ('237) at 53.

261. As I demonstrated above, Iwamura does not disclose key elements from the independent claims upon which Ground 3 is based (claims 25 and 33) including:

- "non-exhaustive search … to identify a near neighbor" (claim 25(b.2); and
- "approximate nearest neighbor search" (claim 33(b.2)).

I note that Petitioner does not rely on Chen for these elements. Pet. ('237) at 53-56; Moulin Depo. 371:17-20 (addressing sublinear); Moulin Depo. 372:2-4 (addressing non-exhaustive); Moulin Depo. 372:5-7 (addressing approximate nearest neighbor search).

262. Moreover, I note that Petitioner does not assert that these missing elements are obvious in light of Iwamura but rather continues to assert that they are expressly disclosed in Iwamura. *See e.g.*, Pet. ('237) 54 ("For the reasons expressed in Ground 1 [anticipation based on Iwamura], Iwamura discloses all elements of claims 25 and 33."). Accordingly, Ground 3 fails at least because the elements from the independent claims addressed above are missing from Iwamura and the Petition does not identify any basis for correcting these deficiencies based on either Iwamura or Chen.

VII. '988 patent.

263. The Board instituted the '988 IPR based on three Grounds:

- Ground 1: Claims 15–17, 21–23, 28, 31, and 51 under 35 U.S.C. § 102(b) as anticipated by Ghias;
- Ground 2. Claims 22. 24–26, and 52 under 35 U.S.C. § 103(a) as obvious over Ghias; and
- Ground 3: Claims 15–17, 21, 23, 27, 28, 31–33, 38, and 51 under 35 U.S.C. § 102(e) as anticipated by Iwamura;

Decision ('988) at 22. I note that the only instituted independent claim is claim 15. I address each Ground in turn.

A. '998 Ground 1: The instituted claims of the '988 Patent are not anticipated by Ghias.

264. The single independent claim of the '988 patent instituted for trial requires a "non-exhaustive search identifying a neighbor." '988, claim 15. Ghias does not disclose (1) a non-exhaustive search, (2) a search identifying a neighbor, or (3) determining an action based on the identification. I address each deficiency in turn.

1. non-exhaustive search (claim element 15(b)).

265. As I explained above in detail (Section V(B)), a "non-exhaustive search" is "a search that locates a match without a comparison of all possible matches."

266. One skilled in the art would understand that Ghias teaches an exhaustive search that compares the work to be identified (user input 23) with "all the songs" in the database—*i.e.*, "all possible matches." One skilled in the art would understand that all "possible matches" in the system disclosed in Ghias are all of the songs in the database. My understanding is confirmed by Petitioner's Declarant:

> Q Is it the case that the set of all possible matches for Ghias are the set of the musical works in the database?
>
> A Yes.

Moulin Depo. 325:19-22. Ghias discloses a search that compares the work to be identified ("user input") with all possible matches—"all the songs" in the database:

> In order to search the database, songs in the database 14 are preprocessed to convert the melody into a stream of the previously discussed U,D,S characters, and the converted user input (the key 23) is compared with all the songs.

Ghias, 5:66-6:2. As Petitioner's Declarant acknowledged when addressing the paragraph from Ghias quoted above:

Q Well, it says, "In order to search the database, songs are preprocessed to convert the melody into a stream of characters"; right?
A Right.
Q And it says, "The converted user input is compared with all the songs"; right?
A Yeah. That part then makes it clear it's all the songs.

Moulin Depo. 339:23:-340:5.

Q Does Ghias teach doing a search by performing a comparison with all the possible songs that are possible matches in the database?
A Yes, it does say all the songs.

Moulin Depo. 340:6-9.

Q But what you've identified in Ghias, in each case, Ghias is going to be searching each of the records in the database; right?
A Yes.
Q Would it be the case that the Ghias search is going to be performing a comparison to each of the melodies that are possible matches in the database?
A It does a comparison, yes, to each of them.

Moulin Depo. 323:4-13.

267. The user input (23) is not compared with some songs in the melody database (14); rather, it "is compared with all the songs." Ghias does not disclose a

search algorithm that does not compare the query to every record in the reference data set. Petitioner's Declarant confirmed my understanding—that the search disclosed in Ghias compares the song to be identified with each record in the database and is therefore not "non-exhaustive"—"a search that locates a match without a comparison of all possible matches" (Section V(B)); Decision ('998) at 7):

> Let's assume we define "exhaustive search" as a search that does a comparison to each of the possible matches in the database that we're searching over.
>
> In that instance, if we have that definition, would you agree that Ghias does perform an exhaustive search?
>
> MR. ELACQUA: Objection.
>
> THE WITNESS: Under your flawed definition, yes, it would.

Moulin Depo. 327:3-12.

Q. Now, [illegible] the case that -- if we
define -- let's define "nonexhaustive" [illegible]
opposite; that is, "nonexhaustive" [illegible]
does not perform a comparison to each of the
possible matches in our record database.
Would you agree, then, that Ghias does not
disclose a nonexhaustive search?
MR. ELACQUA: Objection.
THE WITNESS: I would have to [illegible] -- [illegible]
the patent [illegible]
not searching the entire database. Clearly, it's
intended to search the entire database.
BY MR. DOVEL:
Q. Ghias is intended to search the entire
database?
A. Yes.

Moulin Depo. 327:14-328:4.

268. The Petition and corresponding Declaration fail to demonstrate that Ghias discloses a non-exhaustive search.

269. Petition: As support for the claimed "non-exhaustive search," the Petition relies on the following assertions (and corresponding references to Ghias) labeled ❶ and ❷:

> successive notes which at least approximately matches . . . the melody."). Ghias further discloses that this search may be non-exhaustive, through the use of "an efficient approximate pattern matching algorithm" rather than an algorithm that is guaranteed to yield a match. *Id.* at 6:7-11. Moreover, Ghias teaches that "Several Algorithms have been developed that address [this] problem" ranging from "brute force" to substantially faster algorithms. *Id.* at 23-35 ("Several Algorithms have been . . . Running times have ranged from O(mn) for the brute force algorithm to O(kn) or O(nlog(m) . . . ."). *Id.* at 6:23-35.
>
> 1
> 2

Pet. ('988) 9-10.

270. Petition Charts: The charts in the Petition rely on the same assertions and passages from Ghias: Petitioner's chart for claim 15, element [c] incorporates the chart for claim 1, element [c]:

b) electronically determining an identification of the electronic work based on the extracted features, wherein the identification is based on a non-exhaustive search identifying a neighbor;	Petitioner incorporates the above discussion of Ghias regarding Claim 1c.

Pet. ('988) at 14. The chart for claim 1, element [c], in turn, provides:

c) receiving at the portable client device from the one or more servers an identification of the electronic work based on the extracted features, wherein the identification is based on a non-exhaustive search identifying a neighbor;	Ghias receives and outputs at the computer, which is a portable client device (Ex. 1004 at ¶ 73), a list of identifications of electronic works. 2:50-52, 6:60-63, 7:4-5, 8:26-28, 8:61-63. Such identifications are determined by "searching the melody database 14" to locate a matching melody. 2:50-59, 6:60-63, 7:4-5, Abstract, 8:26-28, 8:61-63. This search may employ a non-exhaustive (1) "approximate pattern matching algorithm" or another algorithm that operates faster than a (2) brute force search. 6:7-11, 6:23-35. This non-exhaustive search identifies a neighbor, i.e., "a ranked list of approximately matching melodies." 2:50-59, 6:60-63.

Pet. ('988) at 12.

271. Declaration: Petitioner's Declaration relies on the same assertions and passages from Ghias:

> relative pitch differences between successive notes of the melody.") Ghias further discloses that this search may be non-exhaustive. Specifically, Ghias teaches that "it is considered desirable to use an efficient approximate pattern matching algorithm" rather than an algorithm that is guaranteed to yield a match. *Id.* at 6:7-11. Moreover, Ghias teaches that "Several Algorithms have been developed that address [this] problem" ranging from "brute force" to substantially faster algorithms. *Id.* at 23-35 ("Several Algorithms have been developed that address the problem of approximate string matching. Running times have ranged from O(mn) for the brute force algorithm to O(kn) or O(nlog(m)), where 'O' means 'on the order of,' m is the number of pitch differences in the query, and n is the size of the string (song).")[1] *Id.* at 6:23-35. Because these algorithms are faster than brute force searches, they are non-exhaustive under Petitioner's construction.

Moulin Decl. ('988) ¶¶69-70.

272. Declaration Charts: Finally, the charts in the Declaration also rely on the same assertions and passages from Ghias:

c) receiving at the portable client device from the one or more servers an identification of the electronic work based on the extracted features, wherein the identification is based on a non-exhaustive search identifying a neighbor;	Ghias discloses receiving and outputting at the computer, which is a portable client device, a list of identifications of electronic works. 2:50-52, 6:60-63, 7:4-5, 8:26-28, 8:61-63. Ghias further discloses that such identifications are determined by "searching the melody database 14" to locate a matching melody. 2:50-59, 6:60-63, 7:4-5, Abstract, 8:26-28, 8:61-63. Ghias further discloses that this search may employ a non-exhaustive "approximate pattern matching algorithm" or another algorithm that operates faster than a brute force search. 6:7-11, 6:23-35. Ghias further discloses that this non-exhaustive search identifies a neighbor by determining "a ranked list of

Moulin Decl. ('988) ¶75.

273. These are the only passages from Ghias cited by the Petitioner and Declarant to support the sub-linear claim elements. Moulin Depo. 113:15-21. The assertions relating to these passages fails to: (a) apply Petitioner's construction (or any other construction) of non-exhaustive to Ghias; or (b) explain how an "approximate string matching algorithm" is expressly or inherently a non-exhaustive search. One skilled in the art would understand that neither the assertions nor the passages from Ghias disclose the claimed non-exhaustive search. I address each in turn.

274. Passage 1:

> For performing the key-search within the database 14, it is considered desirable to use an efficient approximate pattern matching algorithm. By "approximate" is meant that the algorithm should be able to take into account various forms of errors.

Ghias, 6:7-11. First, this passage does not state that the algorithm is not guaranteed to yield a match (as interpreted by Petitioner). Second, and more importantly, the described algorithm does not state (or even suggest) that all possible matches in the database are not searched. The passage does not state that all matches are not considered, or even that all data in all possible matches is not considered. My understanding is confirmed by Petitioner's Declarant:

> You would agree that the term "approximate matching melody" doesn't expressly say that we're going to be using a search approach where we leave out all -- some of the data; right?
>
> A I agree.

Moulin Depo. 347:13-17.

275. Passage 2:

> Several Algorithms have been developed that address the problem of approximate string matching. Running times have ranged from O(mn) for the brute force algorithm to O(kn) or O(nlog(m), where "O" means "on the order of," m is the number of pitch differences in the query, and n is the size of the string (song). See Ricardo Baeza-Yates and G. H. Gonnet, "Fast String Matching with Mismatches," *Information and Computation*, 1992. A preferred algorithm which is considered to offer better performance in general for this purpose is that described in Ricardo A. Baesa-Yates and Chris H. Perieberg, "Fast and Practical Approximate String Matching, "*Combinatorial Pattern Matching, Third Annual Symposium*," pages 185–192, 1992.

Ghias, 6:23-35. One skilled in the art would understand that the "approximate string matching" algorithms discussed in this passages involve matching a work with a record in the database, where the work to be identified includes an "error" so that "various forms of errors" would not prevent a proper match from being identified. The "approximate string matching" algorithm is applied when the work melody "is compared with all the songs" in the database and all of the data within each record. Ghias, 5:66-6:2; Moulin Depo. 347:13-17. This passage discusses comparing the work with a single record in the database.

276. Accordingly, Ghias does not disclose a search that would even meet Petitioner's improper construction of "non-exhaustive search," because Ghias does not search less than "all possible matches" or even less than "all data within all possible matches."

277. I observed that Petitioner only cited the two passages quoted above as support that Ghias discloses the claimed non-exhaustive search. Petitioner's expert confirmed my observation:

Q. Just now, when you -- we had you go through and examine each of the portions of Ghias that you cite with respect to the nonexhaustive search; right?

A. Yes.

Q. When you did that, did you identify any portions in Ghias that taught doing a search but not examining all the possible matches in the database?

A. I didn't identify them, which does not mean they don't exist. They might exist. If they do exist, I did not cite them.

Moulin Depo. 332:17-333:2. As I demonstrated above, these two passages fail to disclose the claimed non-exhaustive search. Accordingly, Petitioner failed to satisfy its burden of establishing that Ghias discloses the claimed non-exhaustive search.

278. Moreover, Petitioner's expert confirmed that other passages from Ghias cited in his Declaration—in an attempt to establish other claimed elements—also do not establish the claimed non-exhaustive search. Moulin Depo. 330:19-331:24; 239:22-25 (2:50-52 "does not teach excluding a portion of the database from our search"); Moulin Depo. 330:15-18 (2:50-52 ("Q. Does Ghias have any portion in where it teaches affirmatively searching only part of the database. A. Not in that sentence, no."); Moulin Depo. 330:1-14 (2:50-52 (the "natural inference" from the statement that the "query engine 24 searches the melody database 14" is that "it's going to search the entire database"); Moulin Depo.

334:2-21 (Q. "[D]oes Ghias teach looking at only a portion of the database? ... A. It does not do that in this paragraph."); Moulin Depo. 337:7-338:17.

279. *Board's concerns*: I now address the Board's specific concerns (identified in its Decision in the '988 IPR) with respect to whether Ghias discloses the claimed non-exhaustive search. I note that in instituting Ground 2, the Board did not rely on the arguments presented by Petitioner and its Declarant or the passages from Ghias quoted by Petitioner and its Declarant in an attempt to establish the claimed non-exhaustive search. Instead, the Board initially found that Ghias disclosed the "non-exhaustive" search because the search disclosed in Ghias could produce a list of matches based on an error-tolerance and the user can perform a "new query on a restricted search list consisting of songs just retrieved:"

> On the present record, we are not persuaded by Patent Owner's arguments. Ghias provides that "[t]he number of matches that the database 14 should retrieve depends upon the *error-tolerance* used during the key-search." Ex. 1010, 6:63–65 (emphasis added). Ghias further provides that "the user can perform a *new query on a restricted search list consisting of songs just retrieved*. This allows the user to identify sets of songs that contain similar melodies." *Id.* at 7:5–8 (emphasis added). Thus, Ghias makes clear that the search need not be exhaustive, as Patent Owner argues, and will act to "identify[] a close, but not necessarily exact or closest, match." per our claim construction. Additionally, given the

Decision ('988) at 12. There are two reasons why the Board's reliance on the "new query on a restricted search list" does not satisfy Petitioner's burden of demonstrating that the instituted clams are unpatentable based on Ghias.

280. First, had the concept of a new second search based on the restricted list (and these passages from Ghias cited by the Board) disclosed the claimed "non-exhaustive search" (as I demonstrated below, they do not), it is my understanding that it could be improper for the Board to rely on these passages in finding the challenged claims unpatentable because these passages were not identified by the Petitioner as support for the non-exhaustive search.

281. I note that Petitioner never asserted (in the Petition, charts, or Declaration) that Ghias discloses a non-exhaustive search because the "user can perform a new query on a restricted search list consisting of songs." The Petition does not even mention the words or concepts emphasized by the Board in its Decision and that form the basis for the Board's preliminary finding that Ghias discloses a non-exhaustive search: "error-tolerance" and "restricted search list consisting of songs just receive." The only references to Ghias presented by the Petitioner for the claimed non-exhaustive search are Ghias, 6:7-11 and 6:23-35 addressing approximate string matching, not performing a "new query on a restricted search list consisting of songs just retrieved" based on an error tolerance. Petitioner's Declarant did not "cite anything in [his] Declaration that teaches, in

Ghias, performing a search that returns a list of ranked matching songs and then performing a second search on that list." Moulin Depo. 146:21-147:21.

282. I note that the Board, however, relied exclusively on two completely different passages from Ghias not cited by Petitioner—Ghias, 6:63-65 and 7:5-8. Decision ('988) at 12. One skilled in the art would understand that these passages address a different concept than the approximate pattern matching concept identified by Petitioner as support for the nonexhaustive search element.

283. Second, using a "new query" on the "restricted search list consisting of songs just received" does not disclose the claimed "non-exhaustive search." A "non-exhaustive search" is "a search that locates a match without a comparison of all possible matches." *See* Section V(B), ¶¶X. The restricted search can be viewed in one of two ways.[31] Under either view, Ghias does not disclose a non-exhaustive search. I address each view in turn.

284. *First view*: Under the first view, the search to identify the record that matches the song being hummed is viewed as a single search with two stages. Under this view, the second search on the "restricted list" is not an independent search—the two stages of the search are not independent. Rather, the search on the

---

[31] Ghias provides no details or information about the search on the restricted search list.

"restricted list" is the second stage of a two-stage search, dependent on the first stage. *See* Moulin Depo. 336:9-15. The second search depends on the first to generate a candidate set. A single work, *i.e.*, the song being hummed (not two or more works), is being identified in the two-stage search. The two stages refine the identified matches; the second stage does not identify any new matches.

285. To constitute a "non-exhaustive search" under this view, the two-stage search process disclosed in Ghias would have to conduct the search without comparing the work to be identified with all possible matches in the dataset. One skilled in the art would understand that the two-stage search disclosed in Ghias is exhaustive because the first stage compares the query to all possible matches in the dataset —"all the songs." Ghias, 5:66-6:2. My understanding is confirmed by Petitioner's Declarant:

> Q And the user -- the user search has already done a comparison to the other songs in the database; right?
>
> A In the first stage, you know, the one that produced the list, the search was on the entire database presumably, yes.
>
> Q So we have a two-stage search. The first stage, we do a comparison of all possible matches in the database; right?
>
> A Yes.

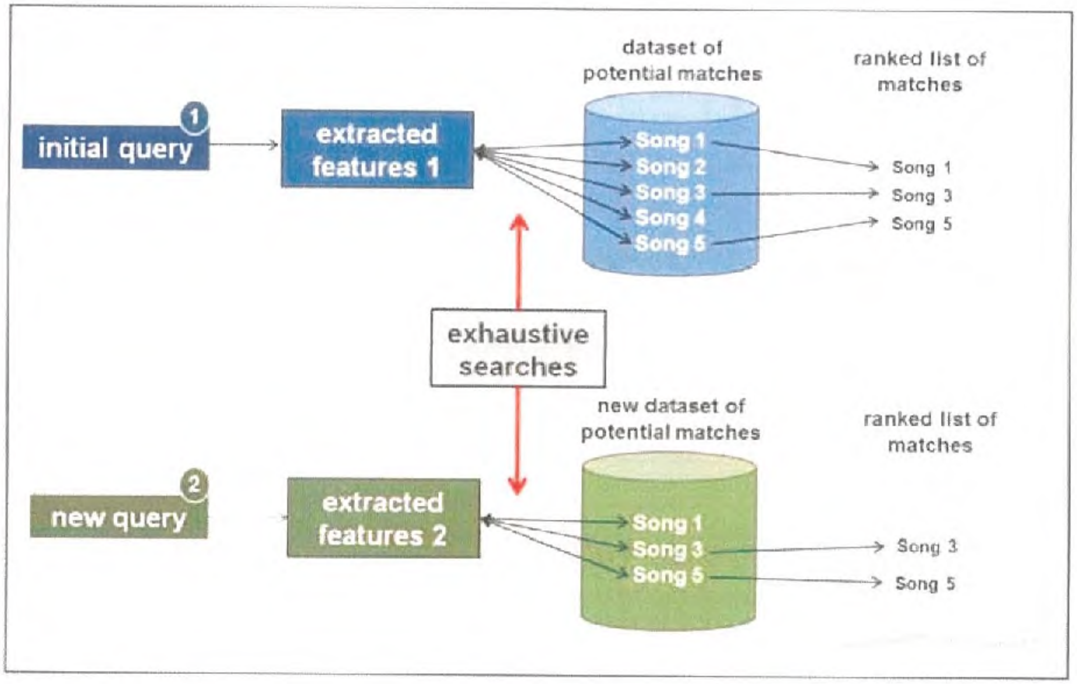
Moulin 336:3-336:12.

Q Does it teach doing a search without doing a comparison of all the possible matches?
A You mean all the possible entries in the original database?
Q Yes.
A No, it does not teach that.
Q The entries in the original database, those are the possible matches?
A Yes.

Moulin Depo. 338:6-14.

286. Under this view, the query on a restricted search list is part of a broader search of every record in the database, which compares the work to be identified with all possible matches—all records in the data set.

287. *Second view*: Alternatively, the second search could be viewed as an independent second search. As disclosed in Ghias, the second search is based on a "new query"—"the user can perform a new query on a restricted search list." Ghias, 7:4-8. The two searches, the first based on an initial query, and the second, based on a second "new query," is reflected in this illustration:



To refine the list of potential matches, the "new query" (2) disclosed in Ghias must be different from the original query (1). This is because Ghias does not teach an alternative search algorithm for searching the restricted list. Rather, Ghias teaches that the same search algorithm is applied to the "new query" (2) that was previously applied to the initial query (1). If the initial query (1) is applied to the restricted list using the same algorithm, the search would produce the same restricted list rather than refine the search as intended by Ghias.[32] Although the

[32] If the query remains constant—the query is not changed—but a different algorithm is applied to the restricted list, this would constitute a single search with