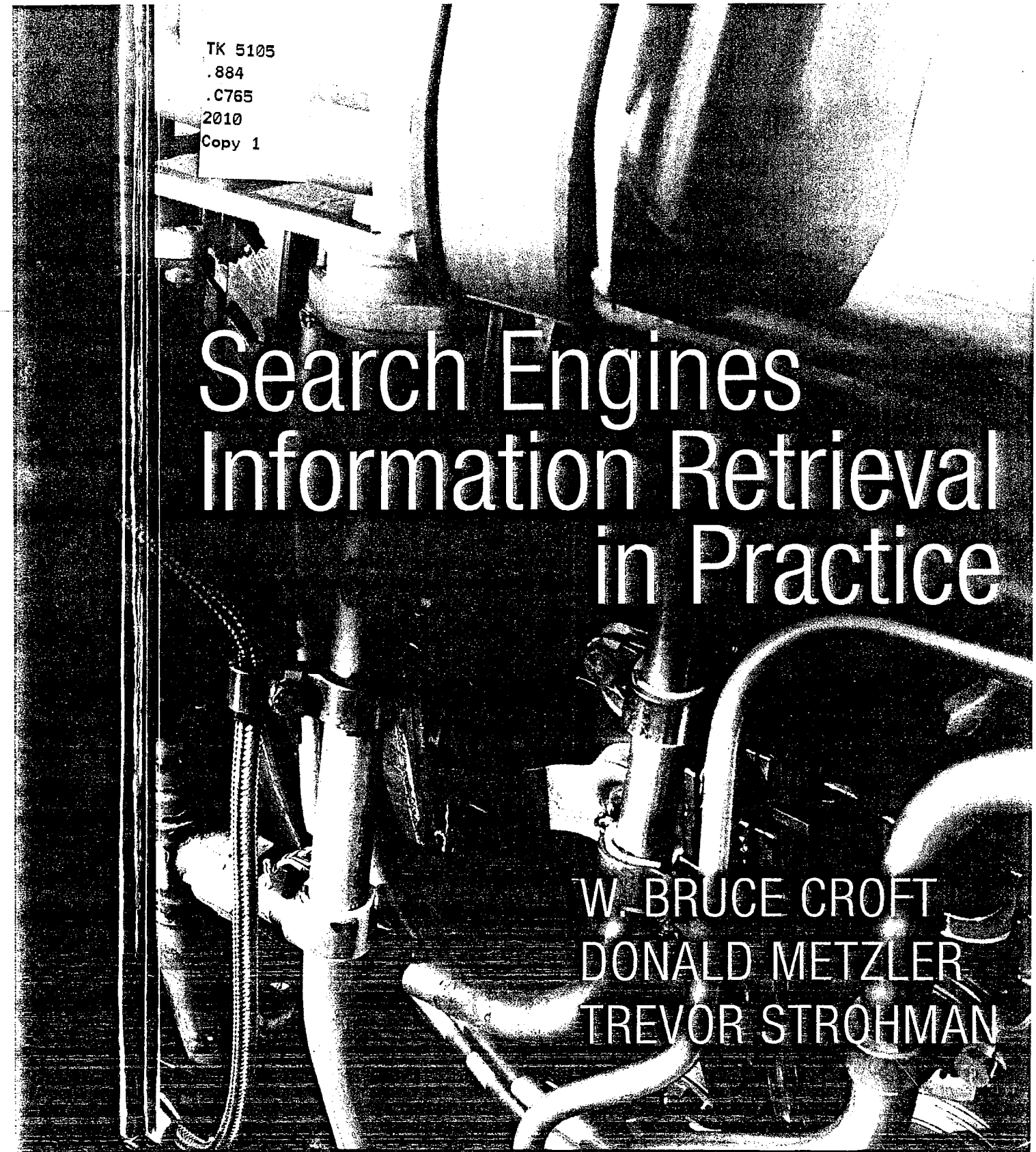

Search Engines Information Retrieval in Practice

W. BRUCE CROFT
DONALD METZLER
TREVOR STROHMAN

Editor-in-Chief	Michael Hirsch
Acquisitions Editor	Matt Goldstein
Editorial Assistant	Sarah Milmore
Managing Editor	Jeff Holcomb
Online Product Manager	Bethany Tidd
Director of Marketing	Margaret Waples
Marketing Manager	Erin Davis
Marketing Coordinator	Kathryn Ferranti
Senior Manufacturing Buyer	Carol Melville
Text Design, Composition, and Illustrations	W. Bruce Croft, Donald Metzler, and Trevor Strohman
Art Direction	Linda Knowles
Cover Design	Elena Sidorova
Cover Image	© Péter Gudella / Shutterstock

Library of Congress Cataloging-in-Publication Data available upon request

ISBN-13: 978-0-13-607224-9
ISBN-10: 0-13-607224-0

1 2 3 4 5 6 7 8 9 10—HP—13 12 11 10 09

We may have more specific goals, too, but usually these fall into the categories of effectiveness or efficiency (or both). For instance, the collection of documents we want to search may be changing; making sure that the search engine immediately reacts to changes in documents is both an effectiveness issue and an efficiency issue.

The architecture of a search engine is determined by these two requirements. Because we want an efficient system, search engines employ specialized data structures that are optimized for fast retrieval. Because we want high-quality results, search engines carefully process text and store text statistics that help improve the relevance of results.

Many of the components we discuss in the following sections have been used for decades, and this general design has been shown to be a useful compromise between the competing goals of effective and efficient retrieval. In later chapters, we will discuss these components in more detail.

2.2 Basic Building Blocks

Search engine components support two major functions, which we call the *indexing process* and the *query process*. The indexing process builds the structures that enable searching, and the query process uses those structures and a person's query to produce a ranked list of documents. Figure 2.1 shows the high-level "building blocks" of the indexing process. These major components are *text acquisition*, *text transformation*, and *index creation*.

The task of the text acquisition component is to identify and make available the documents that will be searched. Although in some cases this will involve simply using an existing collection, text acquisition will more often require building a collection by *crawling* or scanning the Web, a corporate intranet, a desktop, or other sources of information. In addition to passing documents to the next component in the indexing process, the text acquisition component creates a document data store, which contains the text and *metadata* for all the documents. Metadata is information about a document that is not part of the text content, such the document type (e.g., email or web page), document structure, and other features, such as document length.

The text transformation component transforms documents into *index terms* or *features*. Index terms, as the name implies, are the parts of a document that are stored in the index and used in searching. The simplest index term is a word, but not every word may be used for searching. A "feature" is more often used in

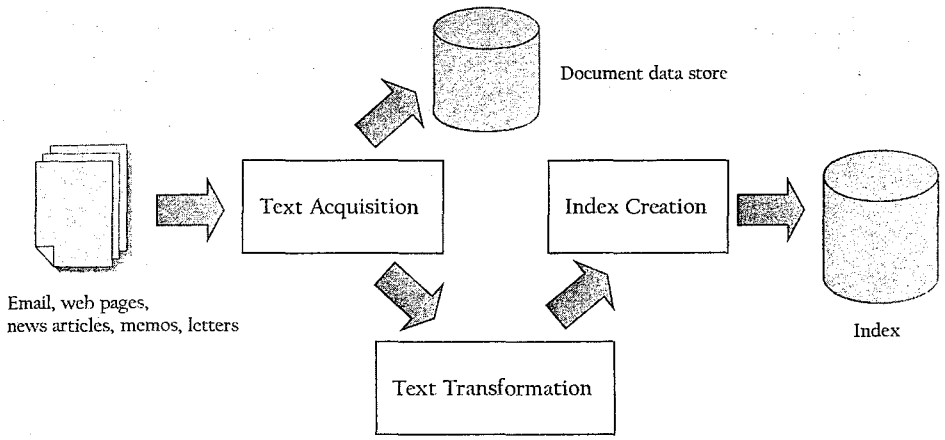

Fig. 2.1. The indexing process

the field of machine learning to refer to a part of a text document that is used to represent its content, which also describes an index term. Examples of other types of index terms or features are phrases, names of people, dates, and links in a web page. Index terms are sometimes simply referred to as "terms." The set of all the terms that are indexed for a document collection is called the *index vocabulary*.

The index creation component takes the output of the text transformation component and creates the indexes or data structures that enable fast searching. Given the large number of documents in many search applications, index creation must be efficient, both in terms of time and space. Indexes must also be able to be efficiently *updated* when new documents are acquired. *Inverted indexes*, or sometimes *inverted files*, are by far the most common form of index used by search engines. An inverted index, very simply, contains a list for every index term of the documents that contain that index term. It is inverted in the sense of being the opposite of a document file that lists, for every document, the index terms they contain. There are many variations of inverted indexes, and the particular form of index used is one of the most important aspects of a search engine.

Figure 2.2 shows the building blocks of the query process. The major components are *user interaction*, *ranking*, and *evaluation*.

The user interaction component provides the interface between the person doing the searching and the search engine. One task for this component is accepting the user's query and transforming it into index terms. Another task is to take the ranked list of documents from the search engine and organize it into the re-

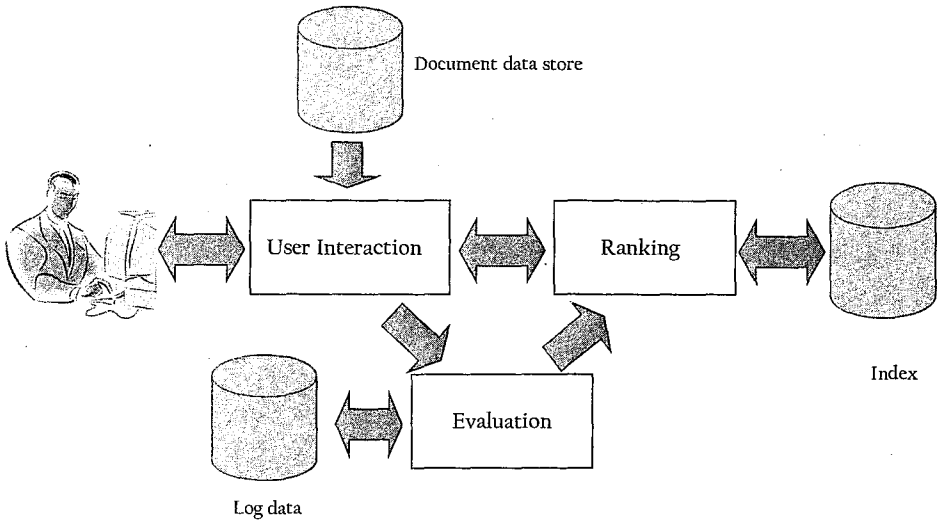

Fig. 2.2. The query process

sults shown to the user. This includes, for example, generating the *snippets* used to summarize documents. The document data store is one of the sources of information used in generating the results. Finally, this component also provides a range of techniques for refining the query so that it better represents the information need.

The ranking component is the core of the search engine. It takes the transformed query from the user interaction component and generates a ranked list of documents using scores based on a retrieval model. Ranking must be both efficient, since many queries may need to be processed in a short time, and effective, since the quality of the ranking determines whether the search engine accomplishes the goal of finding relevant information. The efficiency of ranking depends on the indexes, and the effectiveness depends on the retrieval model.

The task of the evaluation component is to measure and monitor effectiveness and efficiency. An important part of that is to record and analyze user behavior using *log data*. The results of evaluation are used to tune and improve the ranking component. Most of the evaluation component is not part of the online search engine, apart from logging user and system data. Evaluation is primarily an offline activity, but it is a critical part of any search application.

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.