that would pass back and forth between the processor and disk system that enables the disk system to implement the managing strategy that you outline. In your explanation describe each command by listing its name and operands, followed by a description of what the command does.

Example: READ RECORD <track number> <sector number>. This command tells the disk system to obtain the specified record and transmit the record to the processor.

b) Assume that the processor that uses the disk system as a paging system for virtual memory must read pages during page faults and must write back pages that have been altered. What should the disk system do to manage this type of access? To implement this management policy, what should be the commands and replies between processor and disk system?

c) Assume that the processor is using the disk system to store the database for a banking center that serves customers and tellers through on-line queries. How might this database be managed by the disk cache? What should the command and reply interface be for this type of access to implement your suggested management algorithm?

2.16 The object of this exercise is to confirm the footprint model.

a) Measure the footprints for Trace 1 and Trace 2. The footprint is the number of distinct lines touched.

b) Now measure the cache-reload transient under two different conditions. Run Trace 2, then Trace 1, then Trace 2. The reload transient for the second running of Trace 2 is equal to the size of its footprint minus the number of lines of Trace 2 that are resident in the cache when Trace 2 begins to run the second time. Measure the reload transient for one-way and two-way set associative caches of size 32, 64, 128, 256, 512, and 1024. (Twelve different caches.)

c) Calculate the reload transient by using the size of the footprints and the cache structures in the mathematical model for the reload transient as described in the textbook.

d) Compare your answers.

e) Repeat the three parts of this question assuming that you run Trace 1, then Trace 2, then Trace 1. Measure and calculate the reload transient for the second running of Trace 1.

2.17 This problem concerns footprints in caches and in virtual memories.

a) The footprint model developed in the text appears for caches. What are the appropriate values for the cache-footprint model that describe a virtual-memory system in which two programs, *A* and *B*, execute alternately? Define the parameters you need to make the virtual-memory problem correspond to the cache-footprint model.

b) Now consider a collection of programs to execute concurrently in a virtual-memory system. How can the cache-footprint model help you determine which subset of programs to run together?

2.18 The purpose of this question is to analyze the relative advantages and disadvantages of two cache designs. In a computer system that uses both virtual memory and a high-speed cache, a cache tag can be either a virtual address or a physical address. If the tag is a virtual address, it is compared to the virtual address produced by a program before the virtual address is mapped to a real address by a segment and page transformation. If the tag is a real address, it is compared to the virtual address of a reference after that virtual address has been mapped to a real address.

These are the basic cache schemes mentioned in the following parts of this exercise. In response to the questions, you are asked to add more capability or other functions to one or both caches to gain higher performance.

a) Consider the relative performance of the two basic approaches. Which of the two, if any, is the faster? Explain your answer.

b) Consider the problem of handling references to main memory by an input/output processor while maintaining the cache to reflect changes made by the input/output operations. Also, consider how changes made by the central processor to data in the cache can be made available to the input/output processor so that the input/output processor always accesses fresh data when it reads from memory. Which of the two basic schemes, if any, leads to higher performance? Explain your answer. If you have found one scheme to be slower than the other, find a way to improve the performance of the slower scheme to bring it as close to the performance of the faster scheme by augmenting the cache structure, the control of the cache, or its implementation.

c) A cache flush is a purging process that is performed in some virtual-memory systems. When a process in a virtual-memory system relinquishes the processor, and a new process takes its place, the second process may generate the same virtual addresses as the former process, but the second process refers to totally different items. We assume that when a process relinquishes the processor, all of its data held in the cache that might be accessed by another process are purged from the cache. With regard to the cache flush to purge these data, which of the two schemes—real or virtual address tags—leads to higher performance, or are they about equal in performance? Explain your answer. How can the slower of the two schemes be augmented to improve its performance in handling the cache flush? In answering these questions, assume that the caches are four-way set associative.

d) It may be possible to reduce the overhead of cache flushes. Consider how you might augment both of the cache designs (real-address tags and virtual-address tags) so that data resident in the cache for a process need not be purged each time the process relinquishes the processor. Discuss how both cache schemes can be modified to support this behavior.

2.19 Having studied the design of cache-memory systems, consider the parts from which caches are made. Each memory chip in a cache is a standard random-access memory (RAM) chip that contains M bits of information, where M is a power of 2. The memory chip can be designed to have any one of several different organizations. It can, for example, have 1 bit at each of M different addresses, or with a slightly different design, have 2 bits at each of $M/2$ different addresses, or 4 bits at each of

$M/4$ addresses, and so forth. A single cache line made up of, for example, 16 bytes (128 bits) can be built from 128 $M \times 1$ chips or from 64 $M/2 \times 2$ chips, and so forth. If $M \times 1$ chips are used, the chips create not just one cache line, but a total of M sets of cache lines.

a) Show a scheme for organizing $M \times 1$ chips to form a memory with 1024 sets, four-way set associativity, and 16 bytes per cache line. For what value or values of M do you achieve the minimum number of chips in the memory? For this scheme how many address and data bits have to be supplied to a memory chip for each access?

b) Show a scheme for organizing $M/4 \times 4$ chips to form the memory described in the previous part of this exercise. How many address and data bits have to be supplied to each chip?

c) Suppose that $M = 1024\text{K} (= 2^{20})$. What size cache would you design, and how would you organize the memory chips to achieve this size?

2.20 We want to explore the behavior of a multilevel memory hierarchy. Let Level 1 be small and very fast, Level 2 be much larger than Level 1 and somewhat slower, and Level 3 be a very large and very slow memory. The objective is to retain information in Level 2 that will be needed in the near future. All transfers occur on demand, and no prefetching is used.

Assume that both Level 1 and Level 2 are maintained as LRU caches. When a miss occurs, an item is moved immediately to Level 1 from main memory or from Level 2, wherever it is found. If an item is moved from Level 2 to Level 1, no copy of the item remains in Level 2. When an item ages out of Level 2, it is discarded, and future accesses for the item are made to main memory.

a) Assume that Level 1 is organized as N sets and is K-way set associative. Assume that Level 2 is organized as N sets and is J-way set associative. Consider a situation in which a Process A runs, then Process B runs, and then A is to be resumed. Find expressions that show the expected number of lines of A in Level 1, in Level 2, and in main memory at the time that A is resumed.

b) Repeat the previous part of this exercise under the assumption that Level 2 is organized as a $2N$-set cache, J-way set associative.

c) Repeat the first part of this exercise assuming that Level 2 is an RN-set cache, J-way set associative where R is some power of 2.

2.21 This problem examines models for preallocating cache to different functions.

a) Assume that a particular computer architecture produces one instruction reference followed by one data reference for each instruction executed. The stream of instruction references has a miss rate $M_I(x)$ in a cache of size x. Similarly, the stream of data references has a miss-rate function $M_D(x)$ for a cache of size x. (Ignore the exact structure of the cache, and just assume that the miss rate depends on the total number of bytes available.) Assume that both of these functions are known to you. Given C bytes to use for a cache, how should it be partitioned into a data cache and an instruction cache in a way that minimizes the overall miss rate? (Assume that the partitioning can be of any size, not necessarily into pieces that are good for cache structures.)

b) Now assume that r data references occur on the average for each instruction reference, for some real number r. What partitioning of cache produces the least miss ratio in this case?

c) Now assume that the program in the first part is running without having instructions and data prepartitioned. Assume that the cache allocation between data and instructions varies randomly as misses occur and cache replacements change the allocation between data and instructions. Show that when the cache reaches a state in which the number of bytes holding data is equal to the optimum number of bytes to assign to data in a partition, then the cache is in equilibrium. That is, the rate at which bytes change from holding instructions to holding data is equal to the rate at which bytes change from holding data to holding instructions.

d) Assume that the unpartitioned cache in the previous part is in equilibrium. Assume that the program changes so that its M_I and M_D functions change as well. Construct a mathematical model whose solution describes the number of data lines and instruction lines in cache as a function of time. (This model could be very complex, so simply construct the model, but make no attempt to solve it.)

2.22 For this problem assume that two different programs are sharing a computer. Each program takes over the processor, runs for a fixed quantum time Q, and then relinquishes control while the other program takes over. The programs have identical cache footprints. Their streams of address references produce miss rates at the rate of $M_1(x)$ and $M_2(x)$, respectively, in caches of size x. As above, assume that you know both of these functions. Compare two different ways of using C bytes of cache storage for this situation.

The first method is to use cache in a conventional way so that each program runs, uses all cache available as best it can, and then yields the processor to the other program. The second method preallocates a fixed amount of cache storage to each program. The cache storage allocated to a program is private to that program, and is inaccessible to the other program. All storage is allocated to one program or the other.

To answer the following questions, develop the mathematical models required for the comparisons assuming that the miss-rate functions and footprint functions have been given to you.

a) For the scheme that uses a fixed cache allocation, what fixed allocation of cache produces the lowest composite miss rate? (Hint: Express the composite miss-rate as a function of the miss-rate functions and the cache allocations. Then find the minimum of that function.)

b) The characteristics of working sets allow some simplification to the answer of *a* in some cases. Consider three cases: Neither footprint fits fully in cache, both footprints fit fully in cache, and one footprint fits in cache together with a fraction α of the second footprint. In which cases can you give simple allocations and what are these allocations?

c) Now compare the fixed-allocation strategy to conventional allocation as described in the opening discussion. Which of the two schemes produces the lower miss rate in each of the three cases of *b*? Indicate for which of the three cases, one scheme is clearly better than another and in which cases you cannot tell.

2.23 The purpose of this question is to examine slightly less expensive implementations of replacement policies than conventional LRU replacement for set-associative caches. For 4-way set-associative caches, at least 5 bits per set are required to keep track of the order of the cache entries in each set from most recently used entry to least recently used entry.

a) Construct some means for approximating LRU replacement that uses only 4 bits per set for LRU purposes. Describe your replacement algorithm and describe how you use the 4 bits to keep track of which item to replace.

b) Simulate the 4-way set associative caches of Exercise 2.5 on Trace 1 and Trace 2, and compare the effectiveness of your replacement algorithm with true LRU replacement. Your algorithm should do about as well as LRU, possibly marginally better or marginally worse, provided that your algorithm never replaces the most recently used item when it does not replace the least recently used item.

2.24 The purpose of this problem is to explore cache-memory allocations based on miss-rate derivatives analogous to the main memory allocation strategy based on fault-rate derivatives.

Assume that you must design separate caches to hold instructions and data. The total cache size you have available is 16K bytes. Assume that data fetches and instruction fetches occur with equal frequency on your computer system. Assume that the misses as a function for cache size and structure for data fetches is the function that you measure for Trace 1, and that the misses for instruction fetches is the function that you measure for Trace 2.

a) Use your trace data to fit a smooth continuous curve through your data points for caches up to size 16K. You may find that straight lines give reasonably good fits to the log/log plots of these functions. For both instruction and data caches, assume caches are 4-way set-associative. Find where to divide 16K bytes between the two caches so that the number of misses is minimized. The division point can be any integer, and need not be a power of 2 for your answer. The line sizes are 8 bytes for your data.

b) Working analytically, take the derivative of the number of misses as a function of cache size for each of the two analytic functions you created. Calculate the value of the derivatives at the division point you have chosen. Are the derivatives nearly equal?

c) Prove that the division point that gives the fewest misses is the division point at which the miss-rate derivatives are equal. That is, if you take away a little cache from the instructions and give it to the data accesses, then extra misses created for instructions will be approximately equal to the extra misses saved for data fetches. (Hint: find an expression for miss rate as a function of cache allocations, take the derivative of this expression, and find which allocations produce a value of zero for the derivative.)

2.25 The purpose of this problem is to practice calculations of cycles per instruction measures from performance data.

Assume that you are investigating two different cache memory designs, one of which uses separate caches for data and instructions, and one of which uses a

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.