Real Life

3-15. As Kim knew from her English courses, palindromes are phrases that read the same way right-to-left as they do left-to-right, when disregarding capitalization. The problem title, she recalled, was attributed to Napoleon, who was exiled and died on the island of Elba. Being the mathematically minded type, Kim likened that to her hobby of looking for palindromes on her vehicle odometer. 245542 (her old car), or 002200 (her new car).

Now, in her first job after completing her undergraduate program in Computer Science, Kim is once again working with palindromes. As part of her work, she is exploring alternative security encoding algorithms based on the premise that the encoding strings are NOT palindromes. Her first task is to examine a large set of strings to identify those that are palindromes, so they may be deleted from the list of potential encoding strings.

The strings consist of letters, (a ... z, A ... Z), and numbers, (0 ... 9). Kim has a one-dimensional array of pointers, `mylist[]`, in which each element points to the start of a string. As with all character strings, the sequence of chars terminates with the null character, `'\0'`. Kim's program is to examine each string and print out all string numbers (the subscripts of `mylist[]` identifying the strings) that correspond to palindromes.

3-16. Andy, Tom, Bill, and Fred have spent most of their freshman year playing a simple card game. They deal out a deck of 52 regular playing cards, 13 to each person. The rules are similar to Bridge or Spades: teams of two players, seated so that each player has a member of the other team to his left and right. All 52 cards are dealt one at a time in a clockwise manner. The dealer leads first; players take turns playing in a clockwise manner, and must follow suit if possible; the highest card played of the suit led wins the four-card trick unless a "trump" is played, in which case the highest trump wins. Dealer passes to the left after each hand of 13 tricks has been played. The object is to win the most tricks for your team. Trump is determined by the player who bids the most tricks; i.e., who calls out the highest number of tricks he or she thinks his or her team can win. Bidding starts with dealer and moves clockwise until four consecutive players have announced "no further bid." Each successive bid must be at least one higher than its predecessor. If no one makes a bid, the dealer is stuck with a minimum bid of seven.

Lately, however, one of their group has taken a renewed interest in studying. The result is that fewer than four players are available to play some evenings. Tom decided to write a small game-playing program to fill in for the missing player. Fred wants to make it a parallel computing implementation to allow for the possibility that more than one may be missing. Your job is to assist Fred.

3-17. A small company is having difficulty keeping up with demand for its services: retrieval of data from a huge database. The company used to just hand a clerk the list of items to be retrieved, and he or she would manually look through the files to find them. The company has progressed far beyond that; now it hands a program a list of items, and the program looks through the database and finds them. Lately the list of items has grown so large, and the retrieval process has become so time-consuming, that customers have begun to complain. The company has offered you the job of reimplementing its retrieval process by using multiple machines in parallel and dividing up the list of items to be retrieved among the machines.

Part 1: Identify all the pitfalls or roadblocks facing you in moving the retrieval process to a parallel processing implementation that are not present in the existing serial/single processor one.

Part 2: Identify one or more solutions to each item identified in Part 1.

Part 3: Simulate a composite solution, retrieving (from a large database) all the items on a list, using multiple processors working in parallel.

3-18. Over the past 35 years a series of unmanned radar-mapping missions have produced a very detailed topographic map of the moon's surface. This information has been digitized and is available in a gridlike format known as a Mercator projection. The topographical data for the next unmanned landing area is contained in a 100×100 array of grid points denoting the height above (or below) the average moon surface level for that 10 km $\times$ 10 km region. This particular landing region was chosen for its gradually changing topography; you may assume that linearly interpolating between any two adjacent grid-points will accurately describe the landscape between those grid points.

Upon the rocket landing somewhere within the designated 10 km $\times$ 10 km region, it will discharge a number of autonomous robots. They will conduct a detailed exploration of the region and report their results back to the rocket via a line-of-sight lightwave communications link (flashes of light emitted by the robots and detected by the rocket). Once its exploration is complete, it is essential that a robot be able to find quickly the nearest location from which line-of-sight communication can occur, since it will have only a short battery life.

The rocket designers have assured us that their receiving antenna will be 20 m above the site of the landing; the transmitting antenna on the robot will be 1 m above its site, wherever that may be in the region. Thus, given only the 100×100 array and the grid-point locations of the rocket and a robot, your job is to determine the grid point nearest the robot that will permit line-of-sight communication with the rocket. You may assume that the topographical data array contains only heights in integer values between +100 m and −100 m, and that both the rocket and the robots will be located on grid points when accessing your program.

3-19. You are given a array of 100×10000 floating point values representing data collected in a series of "bake-offs" making up the final exam at the Nella School for the Culinary Arts. As with all grading systems, this data must be massaged (normalized) prior to actually assigning grades. For each of the 100 students (whose data is in a row of 10,000 values), the following operations must be performed:

INS (initial normalized score)

The average of the squares of all data values in a given row that are greater than zero but less than 100.

FNS (final normalized score)

The value computed for this student's INS, compared to all other students' INS scores. The students whose INS scores are in the top 10% overall get an FNS of 4.0; those in the next 20% get an FNS of 3.0; those in the next 30% get an FNS of 2.0; those in the next 20% get an FNS of 1.0; the rest get an FNS of 0.0 and have to enter the "bake-off" again next year as a result.

Your program is to print out the FNS scores two ways:

1. A list of FNS scores, by student (row number, FNS) for all students
2. A list of students, by FNS (FNS, list of all rows [students] with this FNS) for all FNS values

3-20. Recently, there has been somewhat of a public health scare related to the presence of a bacterium, cryptosporidium, in the water supply system of several municipalities. It has come to our attention that a band of literary terrorists is spreading that bacterium by cleverly and secretly embedding it in novels. You have been hired by a major publishing company to search a new novel for the presence of the word *cryptosporidium*. It is known that the terrorists have resorted to insertion of punctuation, capitalization, and spacing to disguise the presence of *cryptosporidium*; finding instances of the word requires more than doing a simple word search of the text

For example, in a highly publicized case, one page ended with the sentence

"Leaving his faithful companion, Ospor, to guard the hallway, Tom crept slowly down the stairs and entered the darkened crypt"

while the next page began with

"Ospor, I dium, HELP!" cried Tom, as the giant bats he had disturbed flew around his head. Disaster was narrowly averted when a clerk luckily caught what he thought was a typo and changed "I dium" to "I'm dying" just as the book went to press. Since this particular publisher handles many books, it is essential that each one be scanned as quickly as possible. To accomplish this you have proposed to use many computers in parallel to divide the task into smaller chunks; each computer would search only a portion of a text. If successful, you stand to make a sizeable commission from the sale of networked computers to the publisher.

Alternative approaches are as follows:

1. Divide the text into equal size sections and assign each section to a single processor. Each processor checks its section and passes information (about whether it found *cryptosporidium* in its section, potential portions of the bacterium as the first or last characters of its section, or no evidence of the bacterium at all) back to a master, which examines what was passed back to it and reports on the book as a whole.
2. Divide the text into many more small sections than there are processors and use a work pool approach, in which faster processors effectively do more of the total work, but in essentially the same manner as described in the preceding approach.

www.cs.uncc.edu/par_prog, or otherwise, showing each body in a color and size to indicate its mass.

4-21. Develop the *N*-body equations for a system of charged particles (e.g., free electrons and positrons) that are governed by Coulumb's law. Write a sequential and a parallel program to model this system, assuming that the particles lie in a two-dimensional space. Produce graphical output showing the movement of the particles. Provide your own initial distribution and movement of particles and solution space.

4-22. (Research problem) Given a set of *n* points in a plane, develop an algorithm and parallel program to find the points that are on the perimeter of the smallest region containing all of the points, and join the points, as illustrated in Figure 4.23. This problem is known as the planar convex hull problem and can be solved by a recursive divide-and-conquer approach very similar to quicksort, by recursively splitting regions into two parts using "pivot" points. There are several sources for information on the planar convex hull problem, including Blelloch (1996), Preparata and Shamos (1985), and Miller and Stout (1996).

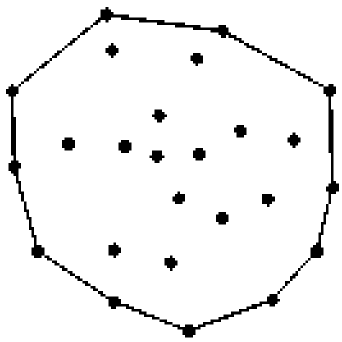


Figure 4.23 Convex hull (Problem 4-22).

Real Life

4-23. Write a sequential and a parallel program to model the planets around the sun (or another astronomical system). Produce graphical output showing the movement of the planets. Provide your own initial distribution and movement of planets. (Use real data if available.)

4-24. A major bank in your state processes an average of 30 million checks a day for its 2 million customer accounts. One time-consuming problem is that of sorting the checks into individual customer-account bundles so they can be returned with the monthly statements. (Yes, the bank handles check sorting for several client banks in addition to its own.) The bank has been using a very fast mainframe-based check sorter and the quicksort method. However, you have told the bank that you know of a way to use *N* smaller computers in parallel, with each sorting 1/*N*th of the 30 million checks, and then merge those partial sorts into a single sorted result. Prior to the bank actually investing in the new technology, you have been hired as a consultant to simulate the process using message-passing parallel programming. Under the following assumptions, simulate this new approach for the bank.

Assumptions:

1. Each check has three identification numbers: a nine-digit bank-identification number, a nine-digit account-identification number, and a three-digit check number (leading zeros are not printed or shown).
2. All checks with the same bank-identification number are to be sorted by customer account for transmittal to that client bank.

Estimate the speedup if N is 10, if N is 1000. Estimate the percentage of time spent in communications versus time spent in processing.

4-25. Sue, 21 years old, comes from a very financially astute family. She has been watching her parents save and invest for several years now, reads the *Wall Street Journal* daily in the university library (for free!), and has concluded that she will not be able to rely on social security when she retires in 49 years. For graduation from college, her parents got her a CD-ROM containing historical daily closing prices covering every exchange-listed security, from January 1, 1900 to the end of last month.

For simplicity you may think of the data on the CD-ROM as being organized into date/symbol/closing price records for each of the 358,000 securities that have been listed since 1900. (Only a fraction are listed at any given date; firms go out of business and new ones start daily.) Similarly, you may assume that the format of a record is given by

date	Last three digits of the year, followed by the "Julian date" (where January 15 is Julian 15, February 1 is Julian 32, etc.)
symbol	Up to 10 characters, such as PCAI, KAUFX, or IBM.AZ, representing a NASDAQ stock (PCA International), a mutual fund (Kaufman Aggressive Growth), and an option to buy IBM stock at a particular price for a particular length of time, respectively.
closing price	Three integers, X (representing a whole number of dollars per unit), Y (representing the numerator of a fractional dollar per unit), and Z (representing the denominator of a fractional dollar per unit).

For example, "996033/PCAI/10/3/4" indicates that on February 2, 1996, PCA International stock closed at $10.75 per share. Sue wants to know how many of the stocks that were listed as of last month's end have had 50 or more consecutive trading days in which they closed either unchanged from the previous day or closed at a higher price, anytime in the CD-ROM's "recorded history."

4-26. The more Samantha recalled her grandfather's stories about the time he won the 1963 World Championship Dominos Match, the more she wanted to improve her skills at the game. She had a basic problem though: she had no playing partners left, having already improved to the point where she consistently won every game against the few friends who still remained!

Samantha knew that computerized versions of Go, chess, bridge, poker, and checkers had been developed, and saw no reason someone skilled in the science of computers could not do the same for dominos. One of her computer science professors at the new campus of the University of Canada, U-Can-II, had told her she could do anything she wanted (within theoretical limitations, of course), and she *really* wanted to win that next World Championship!

Pulling out her slow, old, nearly obsolete 300 MHz/64 Meg (RAM)/6 Gbyte (disk) Pentium, she quickly developed a straightforward, single processor simulator that she could practice against. The basic outline of her approach was to have the program compare every one of its pieces to the pieces already played in order to determine the computer's best move. This appeared to involve enough computation, including rotations and trial placements of pieces, that Samantha found herself waiting for the program to produce the computer's next move, and becoming as bored with its game performance as with that of her old friends. Thus, she is seeking your assistance in developing a parallel processor version.

1. Outline her single processor algorithm.
2. Outline your parallel processor algorithm.
3. Estimate the speedup that could be obtained if you were to network 50 old computers like hers, and make a recommendation to her about either going ahead with the task or spending $2500 to buy the new 800 MHz "Octium," which is reputed to be 50 times faster than her old Pentium for these kinds of simulations.

4-27. Area, Inc., provides a numerical integration service for several small engineering firms in the region. When any of those firms has a continuous function defined over a domain and is unable to integrate it, Area, Inc., gets the call. You have just been hired to help Area, Inc., improve its slow delivery of computed integration results. Area, Inc. has lost money each year of its existence and is so "nonprofit" that payment of next week's payroll is in question. Given your desire to continue eating (and for that to continue, Area, Inc., has to pay you), you have considerable incentive to help Area, Inc.

Given also that you have a considerable background in parallel computing, you recognize the problem immediately: Area, Inc., has been using a single processor to implement a standard numerical integration algorithm.

Step 1: Divide the independent axis into N even intervals.

Step 2: Approximate the area under the function in any interval (its integral over that interval), by the product of the interval width times the function value when it is evaluated at the left edge of the interval.

Step 3: Add up all N approximations to get the total area.

Step 4: Divide the interval width in half.

Step 5: Repeat steps 1 – 4 until the total from the ith repetition differs from the $(i - 1)th$ repetition by less than 0.001% of the magnitude of the ith total.

Since your manager is skeptical about new-fangled parallel computing approaches, she wants you to simulate two different machine configurations: two processors in the first, and eight processors in the second. She has told you that a successful demonstration is key to being able to buy more processors and to your getting paid next week.

$$
\begin{array}{ccc} & A \quad B^T & C \\ AB^T = & \begin{bmatrix} a_0 \\ \cdot \\ \cdot \\ a_{n-1} \end{bmatrix} \begin{bmatrix} b_0 & \cdots & b_{n-1} \end{bmatrix} = & \begin{bmatrix} a_0b_0 & \cdots & a_0b_{n-1} \\ \cdot & \cdots & \cdot \\ \cdot & \cdots & \cdot \\ a_{n-1}b_0 & \cdots & a_{n-1}b_{n-1} \end{bmatrix} \end{array}
$$

Formulate pipeline implementation for this calculation given that the elements of A ($a_0, a_1, \ldots, a_{n-1}$) enter together from the left of the pipeline and one element of B is stored in one pipeline stage (P_0 stores b_0, P_1 stores b_1, etc.). Write a parallel program for this problem.

5-11. Compare implementing the sieve of Eratosthenes by each of the following ways:

(i) By the pipeline approach as described in Section 5.3.3

(ii) By having each process strike multiples of a single number

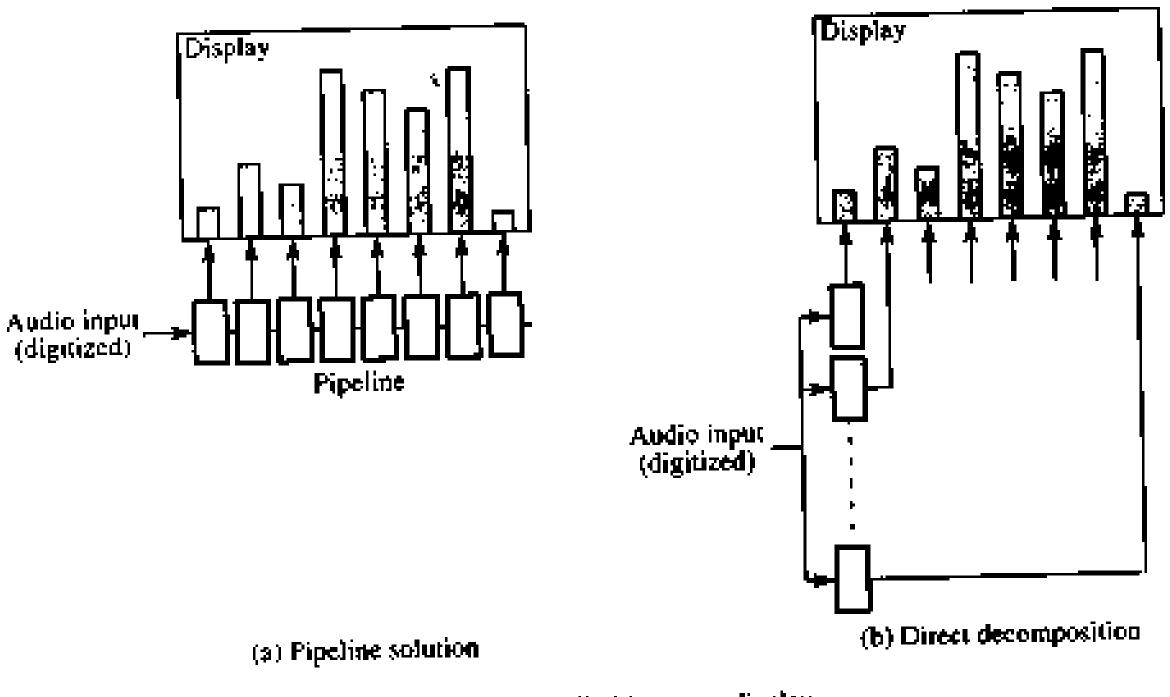
(iii) By dividing the range of numbers into m regions and assigning one region to each process to strike out multiples of prime numbers. Use a master process to broadcast each prime number as found to processes

Perform an analysis of each method.

5-12. (For those with knowledge of computer architecture.) Write a parallel program to model a five-stage RISC processor (reduced instruction set computer), as described in Hennessy and Patterson (1990). The program is to accept a list of machine instructions and shows the flow of instructions through the pipeline, including any pipeline stalls due to dependencies/resource conflicts. Use a single valid bit associated with each register to control access to registers, as described in Wilkinson (1996).

Real Life

5-13. As mentioned in Section 5.1, pipelining could be used to implement an audio frequency-amplitude histogram display in a sound system, as shown in Figure 5.22(a). This application



(a) Pipeline solution

(b) Direct decomposition

Figure 5.22 Audio histogram display.

could also be implemented by an embarrassingly parallel, functional decomposition, where each process accepts the audio input directly, as shown in Figure 5.22(b). For each method, write a parallel program to produce a frequency-amplitude histogram display using an audio file as input. Analyze both methods. (Some research may be necessary to develop how to recognize frequencies in a digitized signal.)

5-14. Due to an unprecedented rise in both automobiles and state-mandated auto inspection requirements, the citizens of the state of New Caroltucky have been complaining that it takes too long to complete the inspection process. Typically, the 35-point inspection checks for brakes (six checks there alone: Each wheel is pulled and brake lining/pad thickness measured, the master cylinder integrity and performance is checked, and general brakeline leaks and cracks are looked for), along with 29 other time-consuming checks. Once a vehicle begins the inspection process, it typically takes a full hour; some claim they have had to wait in a queue just to get to the inspection bay. Legislators have been told of 72-hour queue delays in extreme cases.

The legislature of New Caroltucky is trying to decide whether the state-run inspection stations need a revamping of their operations. Since the legislature has heard that you are taking a course that includes both sequential and parallel programming concepts, it has decided to hire you to do a simulation of both the present inspection system and a proposed system. You are expected to determine the reduction in total time (queue waiting times plus inspection times) if the state revises the inspection process to implement "pipelining" instead of the present "purely sequential" approach.

The present inspection process begins with a driver entering a queue at the inspection station. When an inspector is free, the vehicle at the head of the queue is driven into the inspection bay by the inspector. The inspector then carries out each of the 35 inspections, one at a time. Assuming the vehicle passes, the inspector drives the vehicle out and puts an inspection sticker on it an hour after it was driven into the bay.

The two proposed inspection processes begin the same way, with vehicles entering a single queue. In the proposed new "modified-sequential" system, there will be three inspectors working in three separate bays doing inspections on three vehicles simultaneously. Each draws a vehicle from the head of the queue when ready to begin a new inspection, but sticks with that vehicle until the inspection is complete. Due to space constraints (there are only three bays), it is not possible to add more inspectors to handle more vehicles simultaneously.

In the proposed new "pipelined" system, the state will add some automation to the process so that a vehicle is moved automatically through the inspection bays: entering bay no. 1, moving out of it into bay no. 2 as a new vehicle is moved into bay no. 1, and moving out of it and into bay no. 3 as the second vehicle moves into bay no. 2 and a third vehicle moves into bay no. 1. Under this approach there is plenty of room to add additional inspectors in each bay to speed up the inspection steps handled in that bay. For example, if it would help, inspectors are free to add an inspector for each wheel (each pulls one wheel, measures the pads/shoes, checks for wheel cylinder leaks, replaces that wheel, etc.) plus a fifth who looks for leaks in the lines. Naturally, the state is concerned about cost and efficiency; only the minimum number of inspectors required to achieve the greatest throughput are to be hired. Extra inspectors just standing around will not be tolerated unless eliminating one would cause an increase in the total inspection time once a vehicle enters the first bay.

A table of the tasks assigned to each bay, together with the times each task requires follows. In addition, the loaded labor rate for each inspector (taking into account basic salary, fringe benefits, office and paperwork costs) is given.

Your task is to simulate both new inspection systems to determine several results:

(i) What is the minimum number of inspectors needed under the proposed new pipelined system to achieve the maximum inspection throughput?

(ii) What are the labor costs per inspection performed under each proposed system?

(iii) By how much is the expected inspection delay reduced under each proposed system?

(iv) Without conducting any further simulations (analyzing only what you have obtained from this first part), give an argument for the state investing in additional facilities to expand the number of bays under both systems in order to reduce further the average inspection time. (Naturally, the tasks assigned to each bay under the pipelined approach would have to be changed, but it is assumed that the state inspectors are retrainable.)

Task table:

1.	Pull left front wheel	1 minute
2.	Pull left rear wheel	1 minute
3.	Measure the pads/shoes (per wheel)	1 minute
...		
i.	Replace left front wheel	1 minute
j.	Check wheel alignment	5 minutes
k.	Check exhaust system for leaks	1 minute
l.	Check engine emissions at idle	4 minutes
m.	Check engine emissions under load	3 minutes
...		
z.	Remove old sticker and replace with new one	2 minutes
	Total:	60 minutes

Loaded labor rate table:

1.	Line inspectors (the "worker bees")	\$40,000/yr
2.	Managers (the "drone bees")	\$60,000/yr
3.	Senior managers (the "chairman bees")	\$80,000/yr

Note 1: One manager is needed for every five (or fraction thereof) line inspectors, as well as a senior manager for every four (or fraction thereof) managers beyond the first two. For example, if there are 13 line inspectors, there would be three managers required plus one senior manager.

Note 2: This is an open-ended problem and requires the student to make some assumptions about arrival rates, randomness of arrival times, etc., and is probably more suited to a final project in a course than simply being one of several assigned during a term.

Real Life

6-14. Figure 6.19 shows a room that has four walls and a fireplace. The temperature of the wall is 20°C, and the temperature of the fireplace is 100°C. Write a parallel program using Jacobi iteration to compute the temperature inside the room and plot (preferably in color) temperature contours at 10°C intervals using Xlib calls or similar graphics calls as available on your system. Instrument the code so that the elapsed time is displayed. (This programming assignment is convenient after a Mandelbrot assignment because it can use the same graphics calls.)

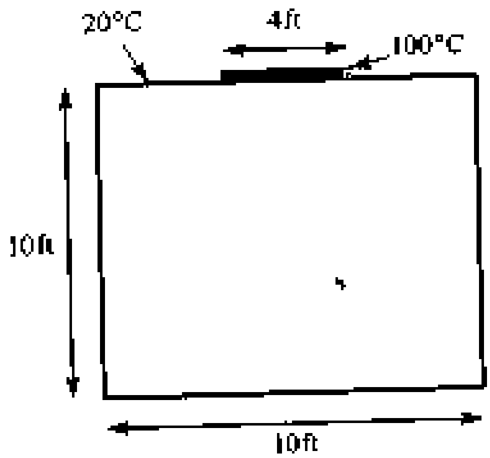


Figure 6.19 Room for Problem 6-14.

6-15. Repeat Problem 6-14 but with a round room of diameter 20 ft and a point heat source in the center at 100°C; the walls are at 20°C.

6-16. Simulate a road junction controlled by traffic lights as shown in Figure 6.20. Vehicles come from all four directions along the roads and either wish to pass straight through the junction to the other side, or turn left, or turn right. On average, 70% of vehicles wish to pass straight through, 10% wish to turn right, and 20% wish to turn left. Each vehicle moves at the same speed up to the junction. Develop a set of driving rules to solve this problem by a cellular automata approach, and implement them in a parallel program using your own test data (vehicle numbers and positions).

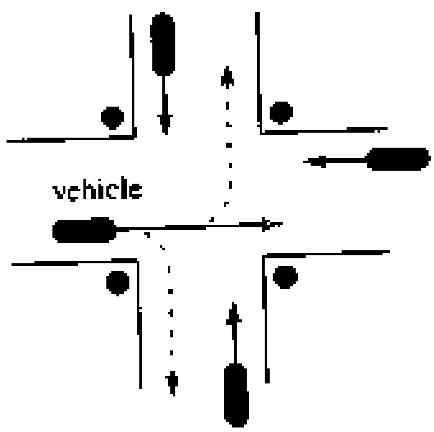


Figure 6.20 Road junction for Problem 6-16

6-17. Write a parallel program to simulate the actions of the sharks and fish as described in Section 6.3.3. The parameters that are input are size of ocean, number of fish and sharks, their initial placement in the ocean, breeding ages, and shark starvation time. Adjacent cells do not include diagonally adjacent cells. Therefore, there are six adjacent cells, except for the edges. For every generation, the fishes' and sharks' ages are incremented by one. Modify the simulation to take into account currents in the water.

6-18. Dr. Michaels was known across campus for being somewhat absentminded. Thus, it was no surprise when he went on a camping trip into the Uwharrie National Forest, but left his map and compass behind. Luckily for him, he had packed his new portable computer that included one of the new cellular modems. Luckier still, he left you sitting back in the computer science building working on a research project!

You had a premonition about this trip; you had downloaded a detailed map of the forest that showed the location of every tree, cliff, and road/path through the forest from the latest NASA satellite pictures. The data is in the form of an array of "cells," with rectangular areas of 0.3 meters on an edge. Each cell contains a 'T', 'C', 'O', or 'R,' which designate what is in that area of the forest:

'T' The area contains an impassable tree.

'C' The area contains an abrupt drop-off (a cliff).

'O' The area is open and passable by the professor.

'R' The area is a roadway or marked path and is passable by the professor.

Thus, you were not particularly surprised when Dr. Michael's e-mail message came through asking for your help. It seems he is suffering from a medical condition that necessitates his exiting the forest as rapidly as possible. He has asked you to write a program to do two things:

1. Identify where in the forest he happens to be.
2. Direct him out of the forest anywhere along the road that borders the southern edge.

Your program can query the professor about what is immediately ahead, behind, left, and right of his present "cell." In response to each query, he will send back four letters. For example, the results of Query() might be 'T', 'O', 'O', 'C' and indicate that he is facing a tree blocking his movement forward, that he can move backward or to his left, and that he finds an impassable cliff to his right. By implication, the cell he is standing in contains 'O' or 'R.'

Your program can tell the professor to move one "cell" in any of the four directions by sending him an 'F' (move forward), a 'B' (move backward), 'L' (move left), or 'R' (move right). The syntax is Move('L') for a move left, etc. Keep in mind that if you tell him to move into a cell containing a tree, your grade will suffer when he finally returns. However, if you direct him to move into a cell containing a cliff, it will not only be on your conscience but also appear as an 'F' on your grade report (which will be filed by Dr. Michael's next of kin).

Your program is to be able to identify the professor's location in as few query/movement combinations as possible, and then direct him by way of the shortest route possible from that location to the road that runs along the forest's southern boundary.

Prototypes of the Query() and Move() functions are as follows:

```
char * Query(void)
/* Query returns a pointer to a string of four characters */

int Move(char direction);
/* if the move is successful, Move returns the value 0. If it is
unsuccessful because you directed him into a tree, Move returns a -1.
If Move is unsuccessful because you directed the professor off a cliff,
Move returns a -100 indicating you just failed your research project
work and need to call the coroner. */
```

Hint: The professor may be facing any of four directions, north, east, south, or west, and does not have a compass. Thus, you will have to match the pattern he returns in response to a Query() to your map data in four possible orientations to narrow down the set of possible locations he is in. Then you will have to Move() him and again Query() him. When you have finally determined where he is located, you have to find the shortest route out of the forest to the southern boundary road.

6-19. Eric was fascinated by the latest episode of "Who Done It?", a mystery thriller he had watched on tape-delay last night. It seems the key to solving the mystery was the ability of Sam Shovel, the detective, to match patterns in various handwriting samples. Eric decided to write a simple program to mimic Sam's pattern-matching behavior. The first thing Eric did was to create a set of 26 "perfect" printed letters on 15×21 grids. These templates would then be compared to actual printing samples, one after the other, to deduce the actual printed characters. His first attempt at writing this program was a total flop! He soon had discovered that none of the actual printed characters was an exact match for his "perfect" characters; he had not recognized any part of the suspect's message.

He then decided to try three radically different approaches. In the first, he used a pipelined solution: scaled the character to a nominal size, centered it in a grid, determined its axes of symmetry, and rotated it to a standard orientation; then compared it to the set of "perfect" characters. In the second, he smoothed the printed character to eliminate noise from the suspect's jittery printing by blurring it slightly using a mathematics-based filter operation, by applying still more mathematics to look at the character in a transform domain, and finally by comparing that to the transforms of the "perfect" characters. In the third approach, Eric decided to simplify things still further; he just counted the number of "matches" between cells on the 15×21 grid and the grid containing the printed character. He moved the printed character around over the "perfect" one until he got the best match, recorded the number of "matches," and then repeated with the next "perfect" character until all 26 had been compared; the best match must be the winner, he thought.

Give a brief analysis of each of his approaches with respect to the one with the best prospects for parallel processing.

6-20. Once upon a time there was an island populated only by rabbits, foxes, and vegetation. The island (conveniently enough!) was the exact shape of a chessboard. Some local geographers have even drawn gridlines that serve to divide the island into 64 squares to facilitate their demographic studies on the populations of each inhabitant.

Within each square the populations of rabbits and foxes are governed by several factors.

- the populations of rabbits and foxes in each square at the start of this "day"
- the reproduction rates of rabbits and foxes (the same over the entire island) during this "day"
- the vegetation growth rate
- the death rates of "old" rabbits and foxes during this "day"
- the eating habits of the foxes (they live entirely on rabbits; rabbits are more difficult to locate in areas with lots of vegetation)
- the eating habits of rabbits (they live on the vegetation; too many rabbits in a square could lead to their starving and/or a lower reproduction rate and/or being easier for foxes to find and eat)
- the migration (from day to day) of rabbits from one square to other squares that are immediately adjacent
- the migration of foxes from one square to any other square within two "leaps"

Since this is an island, there are certain boundary conditions: The 28 squares on the water's edge have no migration possible into or out of the water for either rabbits or foxes. Similarly, there are certain "initial conditions" representing the starting populations of rabbits and foxes in the various squares at the time your program begins its execution.

Your job is to simulate 10 years of life on the island, using time steps of a day in length, and to determine the populations of rabbits and foxes at the end of the period in each square on the island. For each pair of rabbits in a square at the start of a birthing day, which occurs every nine weeks, a litter of babies is born. The size of the litter ranges between two and nine and varies based on both the food supply (vegetation level) and the number of rabbits in that square at the start of the day (population density), as given in Table 6.1. For each pair of foxes in a square at the start of a birthing day, which occurs every six months, a litter of kits is born. The size of the litter ranges between zero and five and varies based on both the food supply (rabbit population) and the number of foxes in that square at the start of the day (population density), as given in Table 6.2

TABLE 6.1 RABBIT BIRTHS FOR PROBLEM 6-20

Vegetation at start of day	Number of rabbits at start of day				
	< 2	2 to 200	201 to 700	701 to 5000	> 5000
< 0.2	0	3	3	2	2
≥ 0.2 but < 0.5	0	4	4	3	3
≥ 0.5 but < 0.8	0	6	5	4	4
≥ 0.8	0	9	8	7	5

TABLE 6.2 RABBIT AND FOX POPULATIONS FOR PROBLEM 6-20

Rabbit population at start of day (per fox)	Number of foxes at start of day				
	< 2	2 to 10	11 to 50	51 to 100	> 100
< 3.0	0	2	2	1	0
≥3.0 but < 10	0	3	3	2	1
≥10 but < 40	0	4	3	3	2
≥40	0	5	4	3	3

A fox can survive on as little as two rabbits per week, but will eat as many as four if they can be found. If the vegetation level is below 0.6, rabbits are more easily found. In that case, on any given day, there is a 4 in 7 chance a fox will eat a rabbit if there are sufficient rabbits available; if there are fewer rabbits than that, or if the vegetation level is at or above 0.6, the foxes will have to make do with a 2 in 7 chance of having a meal — provided there are sufficient rabbits available at that consumption level. (If there are fewer rabbits than the number needed to keep the fox population alive, foxes that didn't get fed have a 10% chance they will die off; that is, in addition to their natural death rate.) The lifespan of a fox is estimated to be four years. Use a random number generator each day to determine whether one or more foxes die a natural death.

A rabbit consumes vegetation; each rabbit consumes 1/10th of 1% of the vegetation in a square per day, under non-food-constrained situations. The normal lifespan of a rabbit is estimated to be 18 months. If the vegetation level is less than 0.35, the death rate due to starvation rises dramatically, as given in Table 6.3.

TABLE 6.3 RABBIT LIFESPAN

Vegetation Level	Rabbit Lifespan
0.1 to 0.15	3 months
0.15 to 0.25	6 months
0.25 to 0.3	12 months
over 0.35	18 months

Use a random number generator each day to determine the number of rabbits that die from a combination of starvation and natural causes. The vegetation level rises quite rapidly when not being eaten by rabbits; growing conditions are ideal on the semitropical island. The vegetation level follows the growth/consumption formula:

$$\text{Vegetation at end of day} =$$
$$(110\% \text{ of vegetation at start of day}) - (0.001 \times \text{number of rabbits at start of day})$$

within the limits that the vegetation level will not drop below 0.1 or grow to be more than 1.0. At the end of each day, 20% of the rabbit population randomly emigrates to adjoining squares. Use a random number generator to determine the number that actually emigrate to each of the possible adjoining squares. Similarly, since foxes are more widely ranging, at the end of each day every fox randomly emigrates zero, one, or two squares distant from their location at the start of that day. Note: All possible migrations are to be considered uniformly likely among the choices available.

Case 1: Uniformly, there are two foxes and 100 rabbits per square initially; the vegetation level is 1.0 everywhere.

Case 2: There are 20 foxes in one corner square and none elsewhere; there are 10 rabbits in every square except in the corner square diagonally opposite the foxes, and it contains 800 rabbits; the vegetation level is 0.3 everywhere.

Case 3: There are no foxes on the island, but there are two rabbits in each square; the initial vegetation level is 0.5 everywhere.

6-21. Develop a cellular automata solution to a real problem and implement it.

6-22. (A research assignment) Develop the rules necessary to model the movement (erosion) of sand dunes at a beach when affected by the waves. (A similar problem is modeling the erosion of the banks of a river due to the water.)

6-23. (A research assignment) Develop the rules necessary to model the airflow across a wing as shown in Figure 6.21 (two dimensions). Select your own dimensions for the solution space and object. Select the number of grid points and write code to solve the problem.

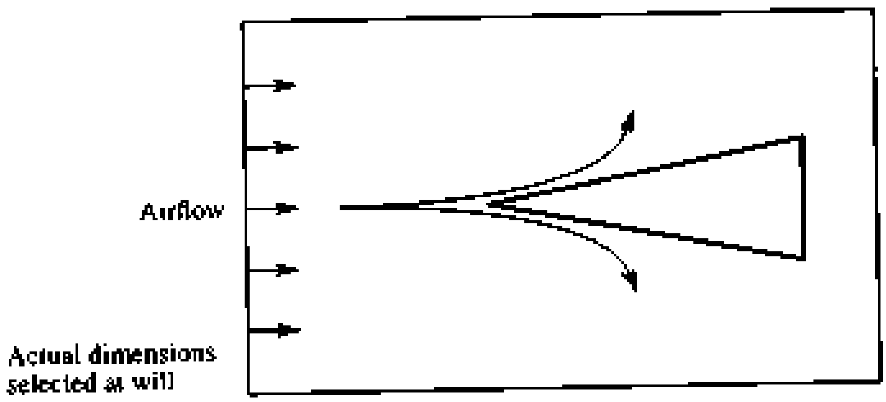


Figure 6.21 Figure for Problem 6-23.

7-5. As noted in the text, the decentralized work pool approach described in Section 7.4 for searching a graph is inefficient in that processes are only active after their vertex is placed on the queue. Develop a more efficient work pool approach that keeps processes more active.

7-6. Write a loading-balancing program using Moore's algorithm and a load-balancing program using Dijkstra's algorithm for searching a graph. Compare the performance of each algorithm and make conclusions.

Real Life

7-7. Single-source shortest-path algorithms can be used to find the shortest route for messages in a multicomputer interconnection network, such as a mesh or hypercube network or any interconnection network one would like to devise. Write a parallel program that will find all the shortest routes through a d-dimensional hypercube, where d is input.

7-8. Modify the program in Problem 7-7 to handle an *incomplete hypercube*. An incomplete hypercube is one with one of more links removed. One form of incomplete hypercube consists of two interconnected complete hypercubes of size 2^n and 2^k ($1 \le k \le n$). More details can be found in Tzeng and Chen (1994).

7-9. You have been commissioned to develop a challenging maze to be constructed at a stately home. The maze is to be laid out on a grid such as shown in Figure 7.19. Develop a parallel program that will find the positions of the hedges that result in the *longest time* in the maze if one uses the maze algorithm: "Keep to the path where there is a hedge or wall on the left" as is illustrated in Figure 7.19, which is guaranteed to find the exit eventually (Berman and Paul, 1997).

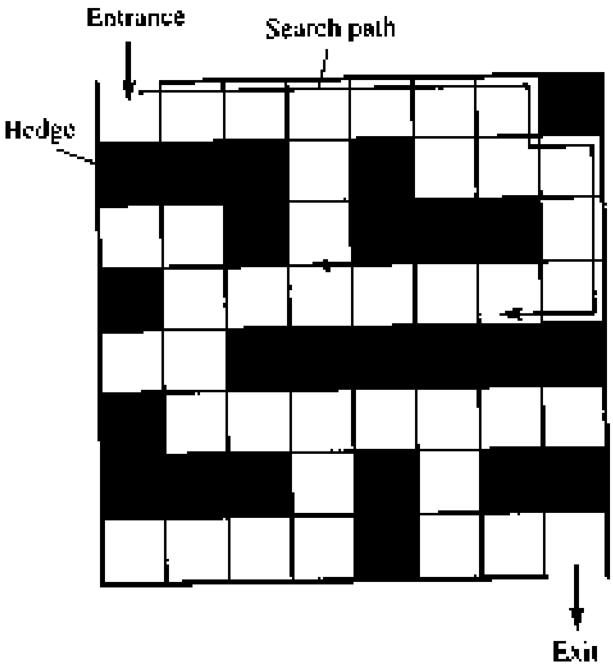


Figure 7.19 Sample maze for Problem 7-9.

7-10. A building has a number of interconnected rooms with a pot of gold in one, as illustrated in Figure 7.20. Draw a graph describing the plan of rooms where each vertex is a room. Doors connecting rooms are shown as vertices between the rooms, as illustrated in Figure 7.21. Write a program that will find the path from the outside door to the chamber holding the gold. Notice that edges are bidirectional, and cycles may exist in the graph.

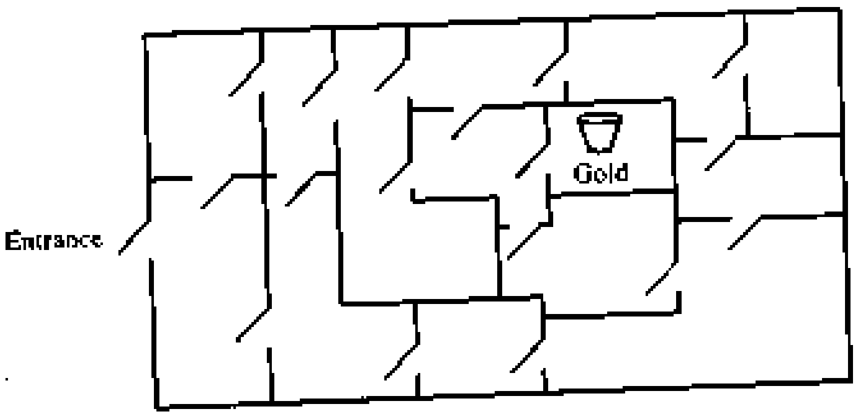


Figure 7.20 Plan of rooms for Problem 7-10.

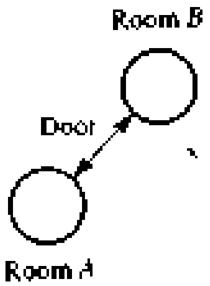


Figure 7.21 Graph representation for Problem 7-10.

7-11. Historically, banks have used one or the other of two competing algorithms to handle the flow of customers at the teller stations within a bank: multiple-queue and single-queue. In the multiple-queue approach, each teller has his or her own queue, similar to the lines at supermarkets. In the purest form of this model, customers enter the bank, choose a queue to enter, and remain in it until served by the teller. One variation that is popular is to permit "queue hopping"; each person in each queue is constantly evaluating whether his or her chances of being served sooner would be enhanced by jumping to another queue. In the single-queue approach, there is only one queue.

The customer at the head of the queue is selected by the first teller completing a transaction. Your task is to simulate the pure multiple-queue and the single-queue approaches using a parallel program and prepare a one-page summary (management report) outlining the perceived advantages and disadvantages of each method given the following set of assumptions. In addition to such items as the average customer waiting times and maximum waiting times, gather whatever other statistics you feel are relevant in documenting your report's conclusions.

Assumptions:

1. There are five tellers.
2. All queues are unlimited in size; customers will snake around the parking lot if necessary. However, the queues are empty at the start of business each day, and although no new customers are allowed into a queue after closing time, those already in the queue are permitted to complete their transactions.
3. Customers arrive randomly at the bank. Due to the bank's location near a major university, customers tend to be concentrated around the end of class times; 10 new customers arrive per minute (on average) between 10 minutes before and 10 minutes after the hour. Two new customers arrive per minute (on average) all other times. The actual arrivals are random and are distributed evenly in the range of one to 19 arrivals per minute near the top of the hour and 0 to 4 arrivals per minute at other times.

4. Each transaction takes a random amount of time to complete. On average, transactions take five minutes but are evenly distributed in the range from one to nine minutes. Each customer is considered a single transaction.

5. Run the simulations between the 9:00 A.M. and 6:00 P.M. (bank opening and closing times) for 100 days to generate the data from which you will draw conclusions for your summary report.

7-12. You are the president of a very large corporation employing nearly a million people. Your firm's personnel department has cleverly organized all employees in a tree-style organization chart in which every employee reports to a supervisor but no supervisor has more than eight or fewer than two employees reporting to him or her. While it may be irrelevant, assume that the average number of employees reporting to a supervisor is five. Thus the average depth of the tree structure is roughly nine. (A little under 1,000,000 lowest-level employees report to about 200,000 first-level supervisors, who report to about 40,000 second-level supervisors, who report to about 8,000 third-level supervisors, etc.)

You have just heard from the U.S. Attorney General that one of your employees was indicted for something that may or may not affect your firm. You did not get the employee's name. Your task is to search the organization for the employee by following the official organization chart personnel hierarchy. You are to do a breadth-first search, starting with the employees you directly supervise, until you identify the individual indicted. Note: You may assume that any nonindicted employee will answer "Not me!", while the employee who was indicted will answer "Yep, the feds got me!".

7-13. A table defines a collection of streets in a section of a major city. Many of those streets are oneway. In addition, there are several tunnels and bridges that allow the driver to skip over or under cross streets. The streets are all numbered. Even numbers are oriented east-west, while odd ones are oriented north-south. Each row of the table has the form

- street number being described
- cross street
- cross street
- mode (oneway or bidirectional)

As an example, one row might look like 13, 4, 6, 1, indicating that it is describing street number 13 in the block where it spans between streets 4 and 6, and is oneway in the direction from street 4 to street 6. (If the line had been 13, 6, 4, 1, then the street would have been oneway between streets 6 and 4.) Another row might look like 13, 6, 22, 2, indicating that street 13 is a two-way street and either a tunnel or a bridge in the section where it links streets 6 and 22 (with no entrances or exits from/to other cross streets between 6 and 22). Complete one (or more) of the following:

1. Find the number of paths that a taxi could use to get from one intersection to another in the city without passing through any intersection more than once.

2. Find the shortest path (fewest blocks traveled) that a taxi could use to get from one intersection to another without passing through any intersection more than once. Note: The only intersections that are associated with bridges or tunnels are those at the two ends.

3. Find the longest path (most blocks traveled) that a taxi could use to get from one intersection to another without passing through any intersection more than once. Note: The only intersections that are associated with bridges or tunnels are those at the two ends.

7-14. A brilliant, yet color-blind, researcher in the biology department has been growing cultured specimens of a dreaded bacterium in Petri dishes. While the culture solution is an opaque white, the bacteria are a pastel pink under visible light. This has hindered her greatly in the daily task of estimating the bacteria growth as she cannot discern yellow/orange/red hues.

She has rigged a digitizing camera that feeds data directly into a computer, and has hired you to write a scanning program that will calculate the percentage of the surface of the Petri dish covered by the bacteria. In addition, your program is to display the surface of the Petri dish in hues of blue/green.

After some initial experimentation, you have determined that an area of the Petri dish, center coordinates (x, y), has an average hue in the range from white to pink that depends upon both the (x, y) coordinates and the length of time, t, that the experiment has been running. For reasons that are not entirely clear, the exact relationship seems to be

$$\frac{t}{100} + \frac{(x+y)}{x_{max} + y_{max}} = Z$$

where the hue throughout a region is white if $Z < 0.95$ and pink otherwise. Your program is to compute and display the bacterium distribution across the Petri dish at a particular experiment time, t. Implement it so that you may zoom in on any particular point. Note: This should be computationally similar to a time-varying fractal, although the picture will not be nearly as jagged.

7-15. Lately, the TV, newspapers, and movies had been filled with stories about aliens, or so it seemed to Tom. Thus, when he was approached by an odd-looking stranger who was posing a multidimensional recursion problem to the people who lived in his apartment complex, Tom simply took it in stride. While it was vaguely discomforting to know that his family might never see him again if he failed to solve the problem, he was confident enough in his math skills to put aside all worries.

The only concern Tom had about the problem was that the aliens seemed much more at ease in dealing with dimensions greater than three than he was. But Tom was confident in his abilities and immediately dug into it.

Given an n-dimensional sphere of radius r, centered at the origin of an n-dimensional coordinate system, compute the number of integer coordinate points inside the sphere. The following are examples provided by the aliens for checking work:

(i) A three-dimensional sphere of radius 1.5 has 19 integer coordinate points within the sphere:

five points in the circle formed when the first coordinate is -1:

$(-1, 0, 0)$, $(-1, 0, 1)$, $(-1, 0, -1)$, $(-1, -1, 0)$, $(-1, 1, 0)$,

five more in the circle when the first coordinate is 1:

$(1, 0, 0)$, $(1, 0, 1)$, $(1, 0, -1)$, $(1, -1, 0)$, $(1, 1, 0)$,

and nine points in the circle when the first coordinate is 0:

$(0, 0, 0)$, $(0, 1, 0)$, $(0, 1, 1)$, $(0, 1, -1)$, $(0, -1, 0)$, $(0, -1, 1)$, $(0, -1, -1)$, $(0, 0, -1)$, and $(0, 0, 1)$.

(ii) A two-dimensional sphere of radius 2.05 has 11 integer coordinate points within the sphere:

$(0, 0)$, $(-1, 0)$, $(-2, 0)$, $(1, 0)$, $(2, 0)$, $(-1, -1)$, $(-1, 1)$, $(1, -1)$, $(1, 1)$, $(0, -2)$, $(0, 2)$

(iii) A one-dimensional sphere of radius 25.5 has 51 integer coordinate points:

$(\pm 25, \pm 24, \pm 23, \ldots \pm 1, 0)$.

8-8. Write a Pthreads program to perform numerical integration as described in Chapter 4, Section 4.2.2. Compare using different decomposition methods (rectangular and trapezoidal).

8-9. Rewrite the Pthread example code in Section 8.4 so that the slaves will take (up to) 10 consecutive numbers to add as a group to reduce the access to the index.

8-10. Condition variables can be used to detect distributed termination. Introduce condition variables into a load-balancing program that has distributed termination such as described in Chapter 7.

8-11. Write a multithreaded program consisting of two threads in which a file is read into a buffer by one thread and written out to another file by another thread.

8-12. Write a multithreaded program to find the roots of the quadratic equation, $ax^2 + bx + c = 0$, using the formula

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

where intermediate values are computed by different threads. Use a condition variable to recognize when each thread has completed its designated computation.

Real Life

8-13. Write a multithreaded program to simulate two automatic teller machines being accessed by different persons on a single shared account. Enhance the program to allow automatic debits to occur

8-14. Write a multithreaded program for an airline ticket reservation system to enable different travel agents access to a single source of available tickets (in shared memory).

8-15. Write a multithreaded program for a medical information system accessed by various doctors who may try to retrieve and update a patient's history (add something, etc.), which is held in shared memory.

8-16. Write a multithreaded program for selling tickets to the next concert of the rock group "Purple Mums" in Ericsson Stadium, Charlotte, North Carolina.

8-17. Write a multithreaded program to simulate a computer network in which workstations are connected by a single Ethernet and send messages to each other and to a main server at random intervals. Model each workstation by one thread making random requests for other workstations and take into account message sizes and collisions.

8-18. Extend Problem 8-17 by providing multiple Ethernet lines (as described in Chapter 1, Section 1.4).

8-19. Write a multithreaded program to simulate a hypercube network and a mesh network both with multiple parallel communication links between nodes. Determine how the performance changes when the number of parallel links between nodes is increased and make a comparative study of the performance of the hypercube and mesh using the results of your simulation. Performance metrics include the number of requests that are accepted in each time period. See Wilkinson (1996) for further details and sample results of this simulation.

8-20. Write a program to simulate a digital system consisting of AND, OR, and NOT gates connected in various user-defined ways. Each AND and OR gate has two inputs and one output. Each NOT gate has one input and one output. Each gate is to be implemented as a thread that receives Boolean values from other gates. The data for this program will be an array defining the interconnections and the gate functions. For example, Table 8.2 defines the logic circuit shown in Figure 8.12. First establish that your program can simulate the specific logic circuit shown in Figure 8.12 and then modify the program to cope with any arrangement of gates given that there are a maximum of eight gates.

TABLE 8.2 LOGIC CIRCUIT DESCRIPTION FOR FIGURE 8.12

Gate	Function	Input 1	Input 2	Output
1	AND	Test1	Test2	Gate1
2	NOT	Gate1		Output1
3	OR	Test3	Gate1	Output2

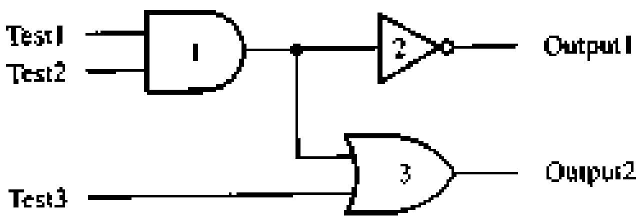


Figure 8.12 Sample logic circuit

8-21. Devise a problem that uses locks for protecting critical sections and condition variables and requires less than three pages of code. Implement the problem

8-22. Many problems in previous chapters can be implemented using threads. Select one and make a comparative study of using Pthreads and using PVM or MPI.

8-23. Write a multithreaded program to implement the following arcade game: A river has logs floating downstream (or to and fro). A frog must cross the river by jumping on logs as the logs pass by, as illustrated in Figure 8.13. The user controls when the frog jumps, which can only be perpendicular to the riverbanks. You win if the frog makes it to the opposite side of the river, and you lose if the frog lands in the river. Graphical output is necessary and sound effects are preferable. Concurrent movements of the logs are to be controlled by separate threads. [This problem was suggested and implemented for a short open-ended assignment (Problem 8-21) by Christopher Wilson, a senior at UNCC in 1997. Other arcade games may be amenable to a thread implementation.]

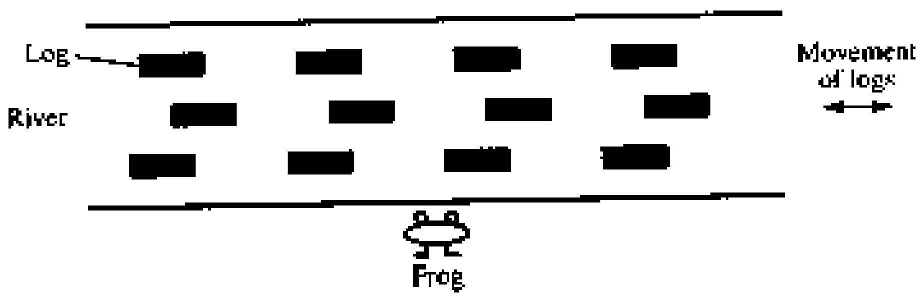


Figure 8.13 River and frog for Problem 8-23

8-24. Write a simple Web server using a collection of threads organized in a master-slave configuration. The master thread receives requests. When a request is received, the master thread checks a pool of slave threads to find a free thread. The request is handed to the first free thread, which then services the request, as illustrated in Figure 8.14. [This problem was suggested and implemented for a short open-ended assignment (Problem 8-21) by Kevin Vaughan, a junior at North Carolina State University in 1997.]

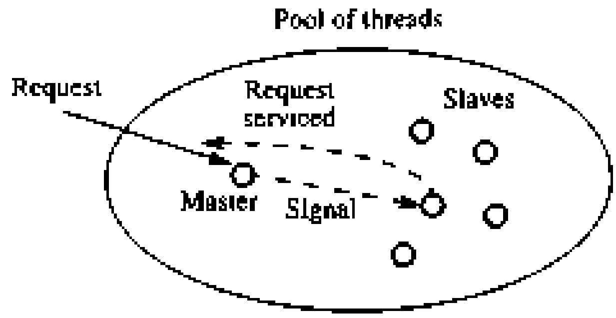


Figure 8.14 Thread pool for Problem 8-24.

9-9. Write a parallel program to implement shearsort.

9-10. Write a parallel program to find the *k*th smallest number of a set of numbers. Use a parallel version of quicksort but apply only to the set of numbers containing the *k*th smallest number. See Cormen, Leiserson, and Rivest (1990) for further discussion on the problem.

9-11. There have been many suggestions that are supposed to improve the performance of sequential quicksort (see, for example, Wainwright, 1985 and references contained therein). The following are two suggestions that have appeared in the literature:

1. Rather than selecting the first number as the pivot, use the median of a group of three numbers picked randomly ("median-of-three" technique)
2. The initial set of numbers is divided into two parts using a first number as the pivot. While the numbers are being compared with the pivot, the sum of the numbers in each of two parts is computed. From these sums, the mean of each part is computed. The mean of each part is used as the pivot for this part in the next partitioning stage. The process of computing two means is done in subsequent stages for the next pivots. This algorithm is called the meansort algorithm.

Determine empirically the prospects for parallelizing these methods.

9-12. Analyze the hypercube quicksort algorithm in Section 9.2.6, in which processors do not maintain sorted numbers at each stage.

9-13. Draw the exchanges of numbers for a four-dimensional hypercube using the parallel hypercube described in Section 9.2.6 that leaves the results in embedded ring order. Hence, determine a general algorithm to handle any sized hypercube.

9-14. Draw the compare-and-exchange circuit configuration for the odd-even mergesort algorithm described in Section 9.2.7 to sort 16 numbers. Sort the following sequence by hand using the odd-even mergesort algorithm:

12 2 11 4 9 1 10 15 5 7 14 3 8 13 6 16

9-15. Repeat Problem 9-14 for bitonic mergesort.

9-16. Compare Batcher's odd-even mergesort algorithm (Section 9.2.7) and his bitonic mergesort algorithm (Section 9.2.8) and assess their relative advantages for parallel implementation on a message-passing multicomputer.

9-17. Prove that the time complexity of odd-even mergesort is $O(\log^2 n)$ with n processors.

Real Life

9-18. Fred has a deck of 52 playing cards that have been thoroughly shuffled He has asked you to determine several things related to reordering them:

1. What modification to bitonic mergesort will be needed to sort the cards, given that there are four suits (spades, hearts, clubs and hearts)?
2. How many friends will Fred have to invite over (and feed) to carry out a modified bitonic mergesort in parallel and how many steps will each friend have to carry out?

9-19. One way to find matching entries in two files is first to sort the contents of the files and then step through the files comparing entries. Analyze this method compared to simply comparing each entry of one file with each entry of the second file without sorting. Write a parallel program to find matching entries of two files by first sorting the files.

9-20. Sorting algorithms can create any permutation of an input sequence by simply numbering the input elements in the required order and sorting on these numbers. Write a parallel program that randomizes the contents of a document file for security purposes by first numbering the words in the file using a random number generator and then sorting on these numbers. (Refer to Chapter 3 for details of random numbers.) Is this a good method of encoding a file for security purposes? Write a parallel program that restores the document file to its original state.

10-16. Write a parallel program to solve the room temperature distribution problem described in Problem 6-15 (Chapter 6) but by the direct means of Gaussian elimination and back substitution rather than by iteration. Only the Gaussian elimination need be computed in parallel; the back substitution may be done on one processor. First, determine the elements of the array **A** of the system of linear equations, **Ax** = 0. This array will always have nonzero elements along the diagonal so that partial pivoting should be unnecessary. Next, decompose the problem so that 10 consecutive rows are handled by one process.

10-17. Write a parallel program for the room temperature distribution problem described in Problem 6-15, but allow for finer grid sizes nearer the fireplace. Provide for user control over the grid sizes. Experiment with adaptive grids.

10-18. Figure 10.26 shows a printed circuit board with various electronic components mounted that generate heat and are at the temperatures indicated. Write a parallel program that computes the temperature distribution. Choose your own components and board dimensions and component placement for this problem. The idea for this problem came from Avila (1994).

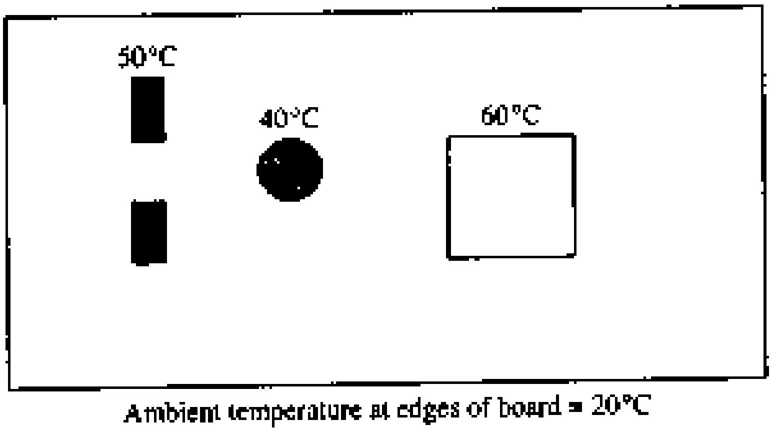


Figure 10.26 Printed circuit board for Problem 10-18.

$$\begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ X_3 \end{bmatrix} = \frac{1}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & w & w^2 & w^3 \\ 1 & w^2 & 1 & w^2 \\ 1 & w^3 & w^2 & w \end{bmatrix}\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Write parallel programs for $N = 4$ using the matrix-vector formulation and using the FFT formulation and measure the execution speeds.

Real Life

11-16. You have been commissioned by a major film studio to develop a really fast "morphing" package that will change one image into another image. You come up with the idea of having two images, the original image and the final image, and changing each pixel on the original image to become closer and closer to the pixels of the final image in a lock-step SIMD fashion. This method is certainly embarrassingly parallel, although it may not create a very smoothly changing shape. Experiment with the method and demonstrate it to the studio using pictures of actors (or your friends).

11-17. The CIA has given you the task of writing a really fast image recognition program (fast enough to be effective on board a missile) for detecting airport runways from images collected by CIA spy planes over a nonfriendly country. It is known that the runways are at least ½ mile long, 25 ft wide, and completely straight. The images contain a 5-mile-by-5-mile area. Develop a parallel program that can detect these runways. Create sample non-perfect images. The CIA will provide the actual images once you have signed your life away. For background information, read the paper on detecting runways by Huertas, Cole, and Nevatia (1990).

11-18. A manufacturer of printed circuit cards is having significant problems with broken printed circuit tracks on its boards and has commissioned you as an independent programmer to find a way to inspect images of the boards automatically as they leave the manufacturing line and recognize broken tracks. You know that all the tracks are straight in one direction or can turn 90 degrees to that direction. Develop a strategy for recognizing the break in the tracks and write a parallel program using realistic test images. For interest, read Mandeville (1985).

8. Also note that two queens at positions (a, b) and (c, d) cannot be on the same diagonal if $a - b = c - d$ or $a + b = c + d$.

12-2. Write a parallel program using a genetic algorithm approach to find the maximum of the function $f(x, y, z) = -x^2 + 1{,}000{,}000x - y^2 - 40{,}000y - z^2$ (as used in Section 12.3.3). Modify the program so that the user may input any three-dimensional function. Instrument the code to determine the number of evaluations your program makes of $f(x, y, z)$.

12-3. Write a parallel program to find the maximum of the preceding function, using the successive-refinement algorithm discussed in Section 12.2.8.

Real Life

12-4. Shortpath Realty has a collection of PCs connected together in its main office. Shortpath needs a parallel program so its real estate agents can plan a schedule of showing houses to minimize the traveling time required. Develop suitable test data and the program.

12-5. Mike was doing just fine traveling around Europe until he discovered a Bäckerei with dozens of delicious-looking pastries and breads. He needs to fill a small bag with edibles for a train ride he is about to take, but he has several constraints:

1. He wants variety; no more than two of any item,
2. He wants the maximum number of calories, which is not necessarily related to the item volume, and
3. He cannot exceed the volume limit of the bag

Develop a parallel program to select quickly the particular items for his trip from the set of 5-tuples describing the Bäckerei items for sale (length, width, thickness, item type, calorie value). In addition to the program, develop a set of 20 test-data items and use them to check your program.

12-6. Given a set of rectangles in which the area of the ith one is given by A_i, write a parallel program to position the rectangles in a large rectangle, subject to the conditions that

1. The individual rectangles must not overlap,
2. All n smaller rectangles must be included in the large one, and
3. The area of the larger one is to be a minimum.

(Note: This is a simplified version of the problem of laying out an integrated circuit. The more complex layout problem simply has more conditions related to relationships between the smaller rectangles.)

12-7. Nat-Ex, a nationwide parcel delivery company, is reassessing the placement of its "hubs" that collect and distribute parcels. Ideally, these hubs should be placed at strategic places across the country to minimize costs and delivery times. You have been commissioned to make a study of possible alternative sites for these hubs and decide to write a parallel program based upon genetic algorithms. You assume that the number of parcels being received is directly proportional to the population, and for a first approximation only the major cities are considered. Write the program, developing suitable input data and constraints. One constraint is the number of hubs.

12-8. A rather similar problem to Problem 12-7 is the siting of airline hubs. Write a parallel program to assist an airline find the best places in the country for its hubs to maximize the airline's profits. Again, input data and constraints must be developed.

12-9. The recently discovered planetoid, Geometrica, has a most unusual surface. By all availabl observations the surface can be modeled by the formula

$$h = 35{,}000 \sin(3\theta)\sin(2\rho) + 9700 \cos(10\theta)\cos(20\rho) - 800 \sin(25\theta + 0.03\pi) + 550 \cos(\rho + 0.2\pi)$$

where h is the height above or below sea level, θ is the angle in the equatorial plane (define: longitude on earth), and ρ is the angle in the polar plane (defines latitude on Earth).

1. Write a sequential program to use hill climbing to find the (θ, ρ) position of the highes point above sea level on Geometrica's surface.
2. Develop an embarrassingly parallel solution to 1.
3. Develop a work pool parallel solution to Part 1 under the assumption that workstations of dramatically varying capabilities are being used to solve the problem.
4. Compare the simulated times required for a solution under the preceding three approaches.

12-10. A commonly occurring problem, known as "bin packing," consists of trying to put k objects of varying characteristics into a smaller number of categories or "bins." While this is commonly encountered in sending a set of presents to a favorite relative on a holiday (you have to find the smallest cost box that will hold all the items), or in industry (a set of machines having different capacities and speeds must be assigned to produce different products optimally), you have been approached to solve a different problem. It has come to the attention of an enterprising (and wealthy) individual that the game of blackjack fits this model also. Each player is trying to obtain anywhere from two to ten or more playing cards whose face values add to 21 or less, while staying at or above the sum of the face values of the dealer's cards. The face values of all previously played cards are known, as are the number of cards of each face value in the deck. You have been hired to outline first a serial algorithm for computing the likelihood that the next card you are dealt will "bust" your hand (result in the sum of face values totalling more than 21) and then to outline a parallel algorithm.

12-11. An enterprising entrepreneur has concluded that the ideal opportunity for him to combine his love of dogs with a vast fortune is to corner the market for a new breed of dog: The Softie. Characteristics of the Softie are as follows:

A.	Length of coat:	8 in. or longer
B.	Coat characteristics:	Extremely soft (or softer)
		Brown areas on a white background
		White paws
		Black tail
C.	Tail characteristics:	Short (4 in. — 6 in.)
		Points straight up
D.	Weight:	Extremely large: 90 kg or heavier
E.	Foot characteristics:	Pawprint area in excess of 9 sq in
		Fully webbed between toes
F.	Disposition:	Extremely mild tempered

Each of these characteristics has a range when viewed across all dogs:

A. Can be represented by 8 bits in which 00000000 corresponds to a hairless dog. 11111111 corresponds to a hairlength of 10.2 in., and 11001000 corresponds to 8 in.

B. Can be represented by six softness bits (in which 000000 corresponds to the ultimate in softness), 000111 is extremely soft, and 111111 is the ultimate in stiffness; three background brightness bits, in which 000 is bright and 111 is dark; three background color bits, in which white is 000 and brown is 001 and all other colors are covered by the remaining six combinations; three foreground brightness bits (000 is bright); three foreground color bits (000 is white, 001 is brown, 010 is red, 011 is yellow, 111 is black, and the remaining bit patterns are other colors); one bit for paw color, in which 0 corresponds to white and 1 is "anything else"; one bit for tail color, in which 0 corresponds to "anything else" and 1 corresponds to black

C. Can be represented by 10 bits, in which eight are related to tail length (00000000 corresponds to a tailless variety, and 11111111 corresponds to a tail-length of 25.5 in. or longer) and two denote tail appearance; 00 corresponds to pointing straight up, 01 corresponds to pointing horizontally, 10 corresponds to pointing straight down, and 11 corresponds to the highly undesirable curly-tailed appearance.

D. Can be represented by 10 bits, in which the first seven correspond to a weight in kg and the second three correspond to a weight in increments of 1/8 kg. Thus, a weight characteristic of 0000101011 would be a weight of 5 3/8 kg.

E. Can be represented by 10 bits, seven bits of which correspond to pawprint area and three to the fraction webbed. In this example, 0000000 corresponds to a pawprint area of 0.5 sq in., 1111111 corresponds to a pawprint area of 13.2 sq in., and 0100011 corresponds to a pawprint area of 4.0 sq in. In the final 3 bits, 000 corresponds to 1/8 webbed, and 111 corresponds to fully webbed.

F. Can be represented by 6 bits, in which 000000 corresponds to the ultimate in mild-tempered disposition, 000100 is extremely mild tempered, and 111111 is "meaner than a junkyard dog" — the ultimate in non-mild-temperedness!

Thus, in total, one needs 64 bits to represent an individual dog in this population. One such dog might be

11001000 001011 000000 111001 01 1000000011 001101100 0100011 011 000100

whose characteristics are

11001000	8 in. coat
001011	Slightly more stiff than desirable
000000	Background color is bright white
111001	Foreground color is dark brown
01	White paws and black tail
1000000011	12.8 in. long, curly-tailed
001101100	13.5 kg weight
0100011	4 sq in. pawprint area
011	50% webbed
000100	Extremely mild mannered

In short, this dog has some desirable characteristics (coat is pretty good, paw size is fine, and its temperament is fine) and some that are undesirable (coat is a little stiffer than the standard, the tail is curly, the dog is extremely small compared to the standard, and it won't be as good a water dog as desired due to its less than fully webbed toes).

The entrepreneur is anxious to hedge his bets on this project and has hired both you and a competitor to develop Softies. Your competitor has a single large kennel complex in which he plans to house 500 dogs; you have opted for five independent kennel locations each housing 100 dogs. Both of you plan to visit animal shelters and take the first 500 nonspayed/neutered dogs you find (i.e., you will start off the initial population with "random" characteristics).

Both your competitor and you plan to keep the population sizes constant in your kennels while breeding successive generations of dogs. You both plan to have each kennel location retain its two parents in the current generation that best match the standard for the Softie breed. In addition, since a pair of parents may have multiple offspring in a given litter (corresponding to making multiple "cuts" and crossovers), both of you plan to give away (to your friends) the offspring that least conform to the standards for Softies, as needed, to keep your local populations constant.

The major difference between the two approaches you and your competitor are using is that he has a single large population while you have five independent kennel "subcontractors" each working with an isolated subpopulation. He will simply produce n generations of offspring; your subcontractors each will produce five generations of offspring in isolation, and only then send their most fit two dogs to their neighbors (A sends one to B and one to E, B sends one to A and one to C, etc.), as in the island model.

1. Write a single processor genetic algorithm solution modeled after your competitor's approach. Run it with several random initial populations and compute the average number of generations it takes before 10% of the population (50 dogs) meets the standards for Softies.
2. Write a multiprocessor genetic algorithm solution modeled after your approach. As in Part 1, run this with several random initial populations and compute the average number of generations it takes before a total of 50 dogs from your kennels meet the standards for Softies.

Hint: You will need to construct an evaluation function that incorporates all of these characteristics, such that a dog meeting the characteristics of the Softie "standard" will exhibit a greater "fitness" than one that does not.

12-12. The 1535 Senators and Representatives of the newly established country of Nella are all under consideration to receive "soft money" campaign financing contributions, expressed in local currency called the Kerf, from a major firm that is encountering some regulatory challenges to the continued sale of its products. It is known that each Senator and Representative has his or her own criteria for assessing such contributions. For example, one might feel that all contributions from this firm should be returned (definitely a "minority viewpoint")! Another might feel that all contributions should be gratefully accepted in light of the superb work he was doing. A few others might have varying thresholds above which they would reject contributions, fearing they might unfairly prejudice their vote on matters concerning this firm.

While the "potential benefit" to the firm is a matter of considerable debate, the firm believes that some Senators and Representatives are more influential than others and therefore considers it important to show them as much "support" through the contribution process as possible. Unfortunately, the firm does not have enough money left after fighting the regulatory agencies to be able to fully "support" all the congressional representatives to the upper limit they might accept. Thus, the firm must be somewhat more selective in allocating its budget for contributions.

The firm's political consultants and attorneys have established an evaluation function that they believe describes each congress member's Support-Interest-as-a-function-of-campaign-Kerfs, SI(Ci, Ki). This function takes into consideration not only the individual's contribution acceptance threshold, but also weights the expected "friendliness to the firm" as a result of the contributions by the influence the congress member has overall in the legislative process. What the firm has lacked up to this point is a mechanism to relate that SI(Ci,Ki) function to a particular allocation of Kerfs among the individuals. As a renowned expert in the application of genetic algorithms, you have been hired to find the optimum allocation of Kerfs to the individual congress members subject to the firm's budget limit.

After a bit of thought you devise a serial genetic algorithm. Each individual in the genetic algorithm sense is a set of 1500 Kerf-amounts, which sum to the total firm budget for this phase. You randomly generate (subject to the budget limits) the Kerf-amounts going to each of the 1535 congress members to produce one genetic algorithm individual, and repeat that to generate 1000 campaign contribution allocations: 1000 genetic algorithm individuals. Fortunately, you quickly recognize that you need a factor of 10 speedup in computation to be able to find an optimum allocation of Kerfs among the congress members in time for your presentation to the firm later today.

Outline a parallel genetic algorithm approach using the networked workstations sitting idle on the desks of ex-employees of the firm who have been downsized as a result of the unfavorable regulatory activities.

12-13. After several recent consolidations in the international banking sector, one of the largest banks, BankWorld, has a problem. It handles some 35 billion credit card transactions per day and has found that while its central computer is still able to record the transactions and process the monthly statements, it has no unused cpu cycles with which to analyze customer spending patterns. It has come to the attention of the Board of Directors of BankWorld that you have skills in the use of networks of workstations to solve problems such as the bank is facing.

Specifically, BankWorld has in excess of 8000 Terabytes (8 million gigabytes) of online storage containing last year's credit card purchase history. This data is the raw point-of-purchase information but is totally unsorted. Each item contains the business ID number, customer account number, date, and amount. BankWorld's internal marketing department proposes to mine this historical data and organize it in ways that will enable BankWorld to earn additional revenue.

As your overall task, BankWorld wants you to design a parallel computing approach to analyzing this data. You are to produce a mailing list of all customers who charged a cumulative total of at least $500 at any combination of home product centers within the past year. BankWorld's marketing department believes it can sell copies of that mailing list (of all "good" home product center customers) back to each of these stores, who will in turn use the lists to send advertising circulars to those customers. Without stopping to analyze the privacy issues involved, you immediately jump into the task of summing purchases, by customer, from a list of businesses identified as being home product centers.

Sketch an algorithm for applying the several thousand idle workstations (made available through the staff reductions occasioned by the latest merger) to this task. Specifically, identify what a typical networked workstation will be doing, how workstations' actions will be coordinated, and how their results will be merged to produce the desired mailing list. Keep in mind that each workstation has a very limited amount of local memory: 512 Mbyte of RAM and 48 Gbyte of disk. Any approach that requires more storage than that on any single workstation is doomed to failure!

12-14. Convert the genetic algorithm example of searching within a volume for the point that maximizes an evaluation function into a hill-climbing problem, implement that solution, and compare it to the time required for an equivalent accuracy genetic algorithm solution:

1. As a sequential approach, and
2. As a parallel approach.

12-15. Convert the hill-climbing solution to the banking application into a genetic algorithm problem, implement that solution, and compare it to the time required for an equivalent accuracy hill-climbing solution:

1. As a sequential approach, and
2. As a parallel approach.