Csc429 Homework 1 Answer

Problem 1. (textbook 2.7) What are the major differences between message-passing and shared-address-space computers? Also outline the advantages and disadvantages of the two.

Ans: Please refer to the textbook section 2.3.2

- Message-Passing Architecture
  - In a *distributed memory machine* each processor has its own memory. Each processor can access its own memory faster than it can access the memory of a remote processor (**NUMA for Non-Uniform Memory Access**). This architecture is also known as message-passing architecture and such machines are commonly referred to as **multicomputers**. Interaction between the processes running on different nodes must be accomplished using messages hence the name message passing. This exchange of messages is used to transfer data, work and to synchronize actions among the processes.
  - Advantage:
    - Message passing requires little hardware support, other than a network.
    - The programmer has an explicit control of data locality.
    - Memory is scalable with the number of processors
    - Increase the number of processors, the size of memory increases proportionally
    - Each processor can rapidly access its own memory without interference and without the overhead incurred with trying to maintain cache coherence
    - Cost effectiveness: can use commodity, off-the shelf processors and networking
  - Disadvantage:
    - Difficult to program: Programmer has to handle data communication between processors
    - Nonuniform memory access (NUMA) times
    - It may be difficult to map existing data structures, based on global memory, to distributed memory organization
- Shared-Address-Space Architecture
  - Provides hardware support for read/write to a shared address space that is accessible to all processors. Machines built this way are often called multiprocessors. Processors interact by modifying data objects stored in this shared address space. Memory in shared address space can be local (exclusive to processor) or global (Common to all processor).
    - (1) A ***shared memory*** machine has a single address space shared by all processors (**UMA, for Uniform Memory Access**). The time

taken by a processor to access any memory word in the system is identical.

    - (2) A ***distributed shared memory*** system is a hybrid between the two previous ones. A **global address space is shared** among the processors but is distributed among them.
- Advantage:
    - Global address space provides a user-friendly programming environment to memory access
    - Data sharing between tasks is both fast and uniform due to proximity of memory to CPUs
- Disadvantage:
    - Lack of scalability between memory and CPUs: Adding processors can geometrically increase traffic on the shared memory-CPU path and for cache coherence management
    - Programmer's responsibility for synchronization constructs (correct access to memory)
    - Expensive to design shared memory computers with increasing numbers of processors

Problem 2. (textbook 2.8) Why is it difficult to construct a true shared-memory computer? What is the minimum number of switches for connecting p processors to a shared memory with b words (where each word can be accessed independently)?

Ans: For a true shared-memory computer such as EREW PRAM with p processors and a shared memory with b words, each of the p processors in the ensemble can access any of the memory words, provided that a word is not accessed by more than one processor simultaneously. To ensure such connectivity, the total number of switches must be $\theta(pb)$. For a reasonable memory size, constructing a switching network of this complexity is very expensive. Thus a true shared-memory computer is impossible to realize in practice.

Problem 3. (textbook 2.9) Of the four PRAM models (EREW, CREW, ERCW, and CRCW), which model is the most powerful? Why?

Ans: CRCW PRAM is most powerful since this class allows multiple read and multiple write accesses to a common memory location.

Problem 4. (textbook 2.17) Determine the bisection width, diameter, and total number of switching nodes in $\sqrt{p}\times\sqrt{p}$ mesh.

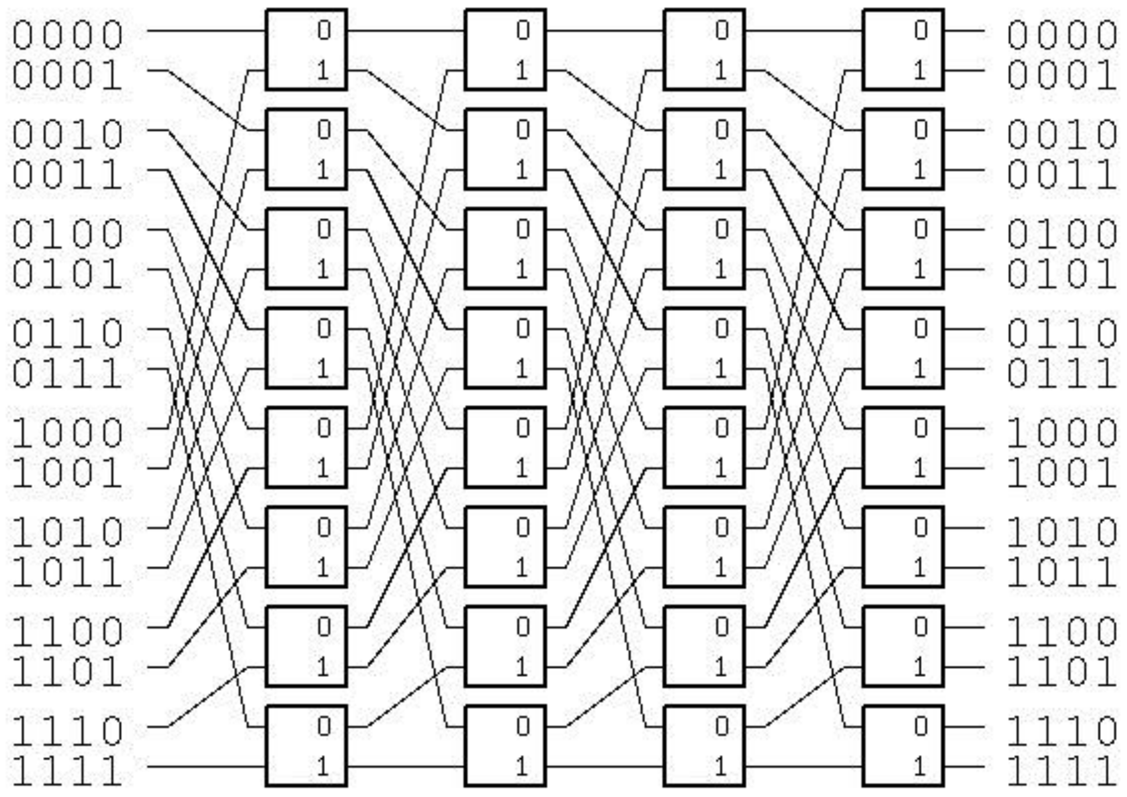
Ans:

For 2D mesh:
The diameter is $2(\sqrt{p}-1)$
Bisection width is $\sqrt{p}$
The total number of switches is p.

For 2D mesh of tree:
An N^2-leaf 2d-MOT (N=√p) consists of N^2 nodes ordered as in a 2d-array $N \times N$ (but without the links). The N rows and N columns of the 2d-MOT form N row CBTs(Complete Binary Tree) and N column CBTs respectively. For such a network, total number of nodes is $|V| = N^2+2N(N-1)$, total number of edges is $|E| = O(N^2)$, the degree of graph is $d = 3$, the diameter of the graph is $D = 4\lg N = 2lgp$, the bisection width of the graph is $bw = N = \sqrt{p}$. The total number of switching nodes is $2N(N-1)=2\sqrt{p}(\sqrt{p}-1)$.

Problem 5. What is the structure of a complete omega network? Please draw a figure of a complete omega network connecting sixteen inputs and sixteen outputs.

Ans:
Refer to your textbook or note for the structure of a complete omega network.



Problem 1.
Consider two algorithms for solving a problem of size N, one that runs in N steps on an N-processor machine and one that runs in √N steps on an N^2 processor machine. Which algorithm is more efficient?

Ans:

$$E_1 = \frac{S_1}{p_1} = \frac{T_s}{T_{p1}p_1} = \frac{T_s}{N*N} = \frac{T_s}{N^2},$$

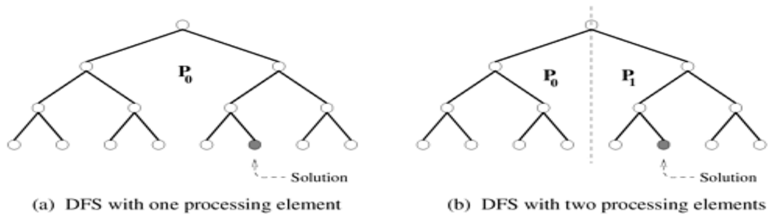
$$E_2 = \frac{S_2}{p_2} = \frac{T_s}{T_{p2}p_2} = \frac{T_s}{\sqrt{N}*N^2} = \frac{T_s}{N^{2.5}}$$

Since N >=1, we have $E_1 \geq E_2$, so algorithm 1 is more efficient.

Problem 5 (textbook 5.2) Consider the search tree shown in Figure 5.10(a), in which the dark node represents the solution.

a. If a sequential search of the tree is performed using the standard depth-first search (DFS) algorithm(section 11.2.1), how much time does it take to find the solution if traversing each arc of the tree takes one unit of time?
b. Assume that the tree is partitioned between two processing elements that are assigned to do the search job, as shown in Figure 5.10(b). If both processing elements perform a DFS on their respective halves of the tree, how much time does it take for the solution to be found? What is the speedup? Is there a speedup anomaly? If so, can you explain the anomaly?

Figure 5.10. Superlinear(?) speedup in parallel depth first search.


P_0
Solution
(a) DFS with one processing element
P_0
P_1
Solution
(b) DFS with two processing elements

Ans:

a. If a sequential search of the tree is performed using the standard depth-first search (DFS) algorithm, 11 edges need to be traversed to access the dark node. If traversing each arc of the tree takes one unit of time, then the running time is 11.
b. If both processing elements perform a DFS on their respective halves of the tree, the running time is 4 to find the dark node.

   By the definition of speedup, we have speedup=Ts/Tp=11/4=2.75 which is greater than the number of processors 2. There is a speedup anomaly. This is called superlinearity effects due to exploratory decomposition. The cause for this superlinearity is that the work performed by parallel and serial algorithms is different.