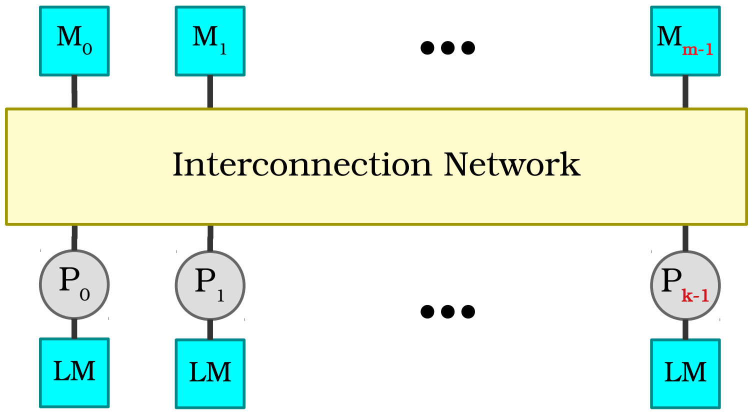
CompSci-131: Homework 1

Due April 24, 2018, in pdf format on EEE .



Consider a system, as the one depicted in the figure above, with k processing elements and m memory modules interconected by a network capable of pairing any processor to any memory.

Time is divided into equal memory cycles. Processing elements request access to memory modules at the beginning of a memory cycle and, following a priority scheme, processing elements are connected to their requested memory module if the module is available. Processing elements that are not granted access to their requested memory module wait for the next memory cycle to renew their request to the same memory module. Access is assumed to occur during the memory cycle and all memory modules are relinquished at the end of the cycle. At the beginning of the next cycle, all processing elements that were granted access generate a new request while those who waited retry access to the previously denied memory module.

Denied accesses to memory modules clearly occur due to the single ported nature of memory modules. If a processing element x is connected to a memory module y, during the same cycle no other processing element z can connect to y. **Processing element z has to wait** and attempt access in the next cycle. The number of cycles each processing element takes to access each distinct memory module request will be dubbed the access time.

Suppose a priority scheme is devised such that processing elements with lower index get access first, in the figure below this is equivalent to scanning the processing elements from left to right and grant access to the first that finds its requested memory module free. In the example of the figure below, P_3 and P_5 have to wait.

ht

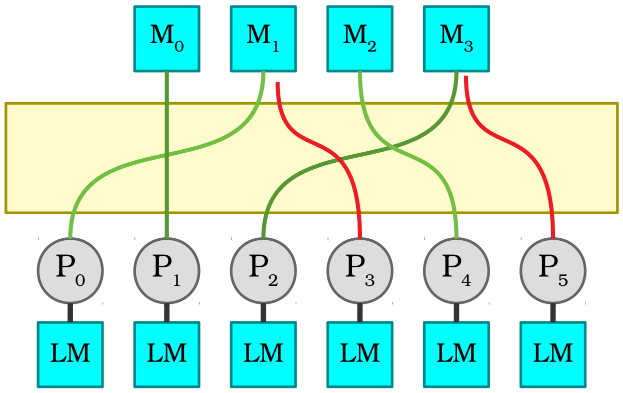


Figure 1: Example

Assignment The main goal of this assignment is to use computer simulation to characterize the access time of processing elements to memory modules under certain workload assumptions.

TWO workloads assumptions will be assumed.

- Each processing element issues a memory module request at the beginning of each memory cycle using a uniform distribution.
- Each processing element issues a memory module request at the beginning of each memory cycle using a Gaussian distribution with a different mean for each processing element.

You are to create a program in C to simulate the above architecture and its operation under the memory module access scenario. The result of the simulation(s) will be plots of the average processing elements memory access times when keeping the number of processing elements fixed and varying the number of memory modules.

- Generate one plot per each fixed number of processors k for $k \in \{2, 4, 8, 16, 32, 64\}$.
- In each plot, the $X - axis$ represents the number of memory modules varying from 1 to 2048.
- In each plot, the $Y - axis$ represents the average, in time and across the processing elements, of the number of memory cycles that requests take. (The average memory access time is the sum of all the waiting times divided by the total number of requests so far.)
- Both axes have to be in a linear scale (e.g. not logarithmic).
- Superpose all plots corresponding to each fixed number of processing elements into one chart.

You are free to design your simulation program in any which way you like. The problem was solved in advance using the following considerations/specifications:

- Use 3 equal size arrays to represent the processing elements, the wait counters for each of them, and the priority of connection.
- Use another array to represent the memory modules (remember that the number of memory modules doesn't need to be equal to the number of processors). Elements in this array are will hold a 1 if the represented memory is already connected to a processing element and a 0 if free.
- To avoid processing element starvation, prioritize the ones that have been waiting longer.
- The termination condition to characterize average wait time is when the current value differs from the previous one by less than 0.02%.
- In each cycle, only the processors that successfully connected to their requested memory modules will generate a new request, the other processors (that couldn't connect) will maintain the same request.

Report Please submit a report that clearly explains the logic of your program and includes a listing of your commented program and the plots that result for each run of the program. Please label plots clearly !!! Do include also a discussion and interpretation of results.