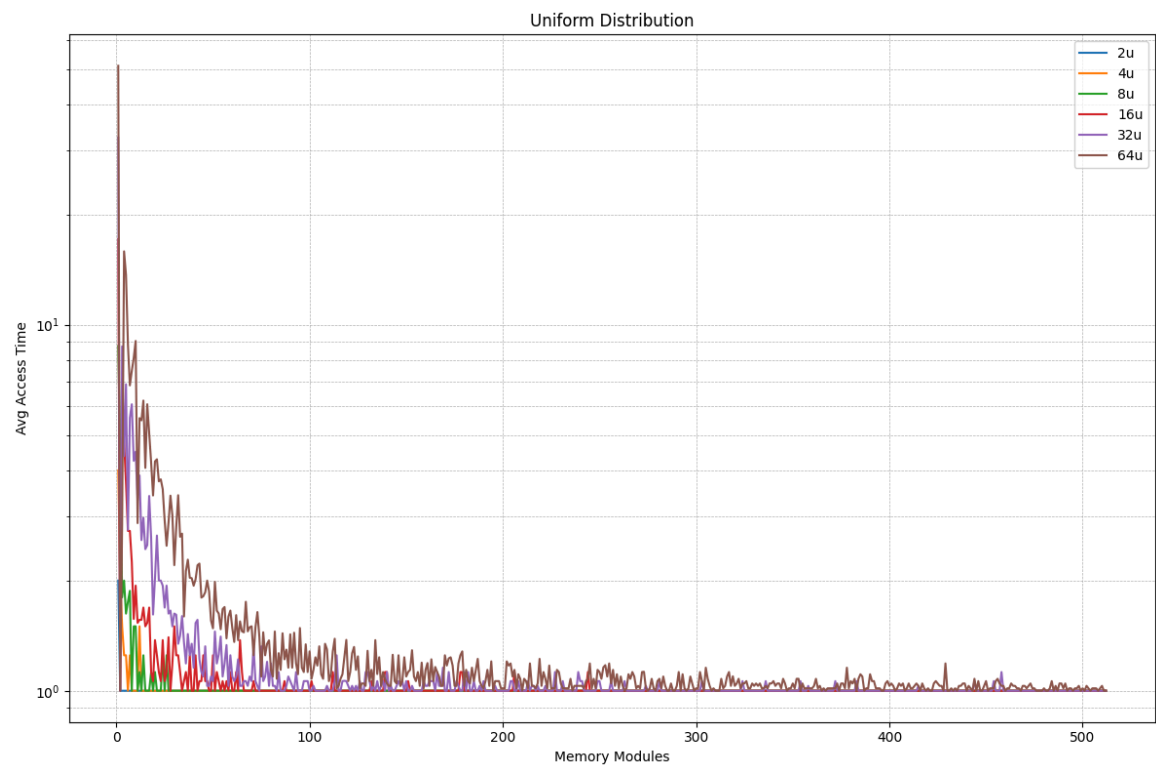
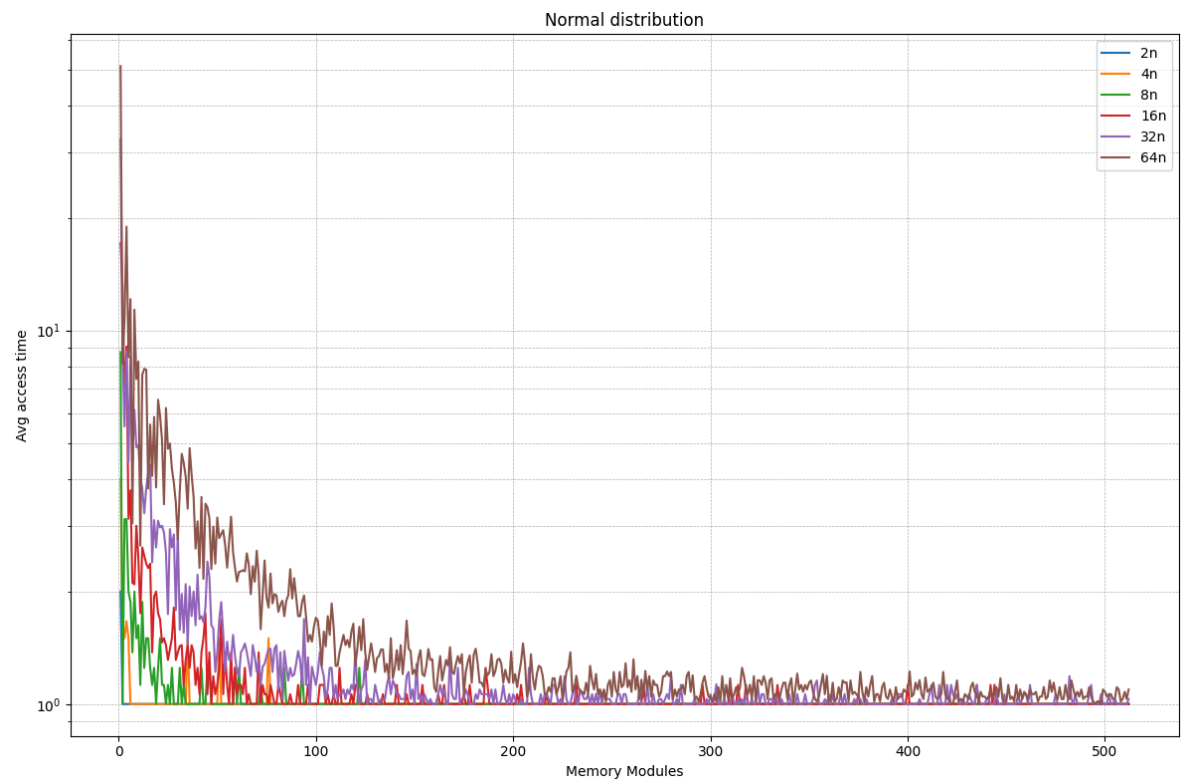


Homework 1 Report

| Name | UCI NETID |
|----------------------------|-----------|
| Mitali Manojkumar Deshmukh | deshmum1 |
| Nisha Rajesh Gharpure | ngharpur |
| Bhoomi Ketan Patani | bpatani |

Charts



Discussion

We use a circular assignment scheme to distribute memory requests evenly across modules, ensuring fairness and load balancing. By rotating through processors in a round-robin manner, the scheme prevents any single memory module from being overloaded, which is especially important for skewed access patterns like those in a normal distribution. This approach also avoids starvation, as no processor is skipped or delayed indefinitely—every request eventually gets a chance to be fulfilled. As a result, the system maintains consistent and low average access times across varying processor counts, as reflected in both the uniform and normal distribution plots.

In both uniform and normal distributions, the average access time decreases sharply as the number of memory modules increases, then flattens out. For higher processor counts (32, 64), this plateau is reached later **around 120 modules for uniform distribution**, where requests are evenly spread, and **around 200 modules for normal distribution**, where clustered access causes more contention.

1) How does the memory request distribution affect the behavior of the system? Which one is more realistic?

The distribution affects contention and locality. A normal distribution, which has a peak around a central point, tends to overload certain memory modules, causing more contention and longer access times until enough modules are available to spread the load. In contrast, a uniform distribution spreads memory requests more evenly, reducing contention earlier.

A normal distribution is generally more realistic in many computing systems because access patterns tend to cluster (due to cache locality, frequently used data, etc.).

2) If this simulation was in the context of making a decision to buy expensive memory modules for a given number of powerful processors, what would you recommend? Why?

For a small to medium number of processors (e.g., 2 to 16): No need to invest in a large number of memory modules. Performance stabilizes quickly with fewer modules. **For a large number of processors (32, 64):** It's beneficial to invest in more memory modules, but **only up to the point where average access time flattens** beyond that, the return on investment diminishes (Amdahl's Law).

This is our recommendation because, after a certain point, adding more memory modules yields **diminishing returns**. For **normal distribution**, contention is higher due to clustered access patterns, so performance stabilizes around **200 modules**. For **uniform distribution**, load is spread evenly, and optimal performance is reached with about **120 modules**. Beyond these thresholds, extra memory provides little benefit and would not be a cost-effective investment.