Synchronization

Learning Outcomes

- Data Race, Mutual Exclusion, Deadlocks
- Atomics, Locks, Barriers
- Reduction
- Prefix Sum

Classwork

- Write CUDA code for the following functionality.
  - Assume following data type, filled with some values.

    ```
    struct Point { int x, y; } arr[N];
    ```
  - Each thread should operate on 4 elements.
  - Find the average AVG of x values.
  - If a thread sees y value above the average, it replaces all 4 y values with AVG.
  - Otherwise, it adds y values to a global sum.
  - Host prints the number of elements set to AVG.

Data Race

- A datarace occurs if ***all*** of the following hold:
  1. Multiple threads
  2. Common memory location
  3. At least one write
  4. Concurrent execution
- Ways to remove datarace:
  1. Execute sequentially
  2. Privatization / Data replication
  3. Separating reads and writes by a barrier
  4. Mutual exclusion

Classwork

If initially flag == 0, then S2 executes before S1.

If initially flag == 1, then S2 executes and after that S1 may execute or T1 may hang.

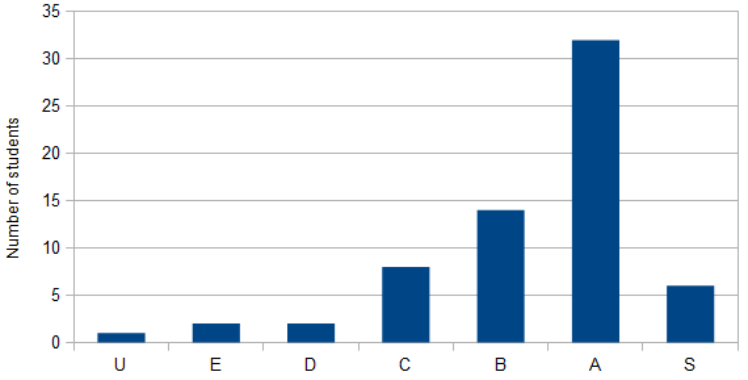
- Is there a datarace in this code?
- What does the code ensure?
- Can mutual exclusion be generalized for N threads?

T1	T2

flag = 1;
while (flag)
;
S1;
	while (!flag)
	;
	S2;
	flag = 0;

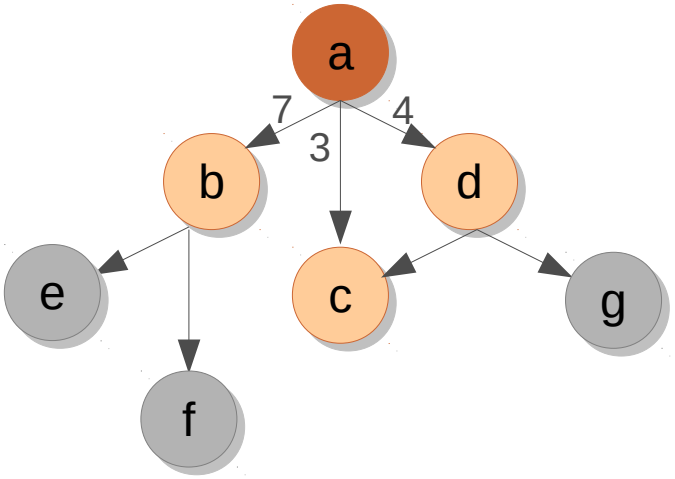
Classwork: Grading

- Given roll numbers and marks of 80 students in GPU Programming, assign grades.
  - S = 90, A = 80, B = 70, …, E = 40, and U.
  - Use input arrays and output arrays.
- Compute the histogram.
  - Count the number of students with a grade.



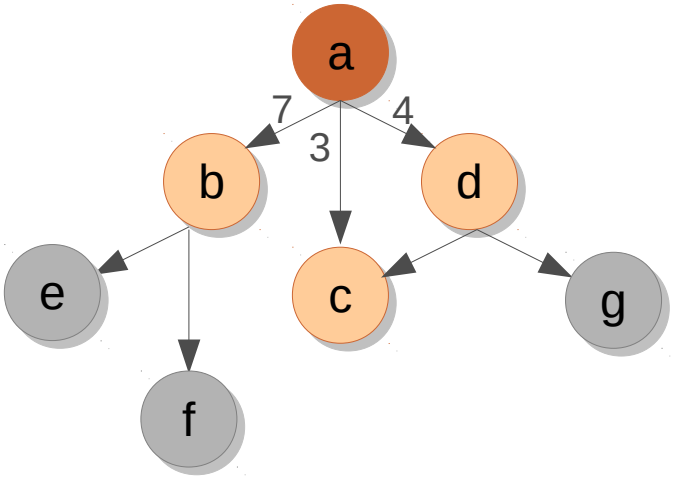
Let's Compute the Shortest Paths

- You are given an input graph of India, and you want to compute the shortest path from Nagpur to every other city.
- Assume that you are given a GPU graph library and the associated routines.
- Each thread operates on a node and settles distances of the neighbors (Bellman-Ford style).



Let's Compute the Shortest Paths

- You are given an input graph of India, and you want to compute the shortest path from Nagpur to every other city.
- Assume that you are given a GPU graph library and the associated routines.




```
__global__ void dsssp(Graph g, unsigned *dist) {
    unsigned id = ...
    for each n in g.allneighbors(id) {      // pseudo-code.
        unsigned altdist = dist[id] + weight(id, n);
        if (altdist < dist[n]) {
            dist[n] = altdist;
}   }   }
```

What is the error in this code?