Andrew Branum

HW4

The graph provided was implemented within a python script as a directed graph where nodes are process checkpoints and edges represent the time between those checkpoints. The time taken between is either deterministic or a uniform distribution between 2 numbers. The purpose of the simulation is to evaluate the performance of each path, find the criticality of each path, and offering up suggestions to improve the graph design. There are 5 given paths:

|  |  |
| --- | --- |
| 1 | 1->2->3->4->7 |
| 2 | 1->2->4->7 |
| 3 | 1->5->3->4->7 |
| 4 | 1->5->4->7 |
| 5 | 1->5->6->7 |

Each edge also has a weight that is either deterministic or uniformly distributed:

|  |  |
| --- | --- |
| 1,2 | U(3,5) |
| 1,5 | 6 |
| 2,3 | 6 |
| 2,4 | U(7,9) |
| 3,4 | U(5,8) |
| 4,7 | 4 |
| 5,3 | 7 |
| 5,4 | 9 |
| 5,6 | U(7,10) |
| 6,7 | U(8,12) |

A diagram of a network

AI-generated content may be incorrect.

Both the paths and the edge weights are implemented within the script, and a Monte Carlo approach is used with random sampling. All paths are enumerated and for each iteration for every edge that has a uniform distribution has a random value generated within its constraints and the deterministic edges are kept constant. For each path, the total time is calculated by summing up the times of its edges. Each iteration also keeps track of which path takes the longest to complete marking it as the critical path. A counter tracks every time a path is marked critical. After running the simulation 10,000 times, my simulation found the following results:

Average Duration Per Path:

|  |  |
| --- | --- |
| 1 | 20.49 |
| 2 | 15.99 |
| 3 | 23.50 |
| 4 | 19 |
| 5 | 24.5 |

Critical Path % Chance

|  |  |
| --- | --- |
| 3 | 28.46 |
| 5 | 71.54 |
| All Others | <1 |

A white background with black text

AI-generated content may be incorrect.

After analyzing the results, the paths that cross node 5 tend to be a critical path 99% of the time. This makes sense considering the weights moving from node 5 to its following nodes (7, 9, U(7,10)). Optimizing node 5 to reduce its costs would help lower the critical rates of paths that traverse through it. Breaking up node 5 into different nodes may help lower the cost some or if the uniform distributions were able to be converted to deterministic may also lower costs. Another option is to offload the work occurring at node 5 and to continue the process while the functionality at node 5 continues as an asynchronous operation. The system itself could be altered to dynamically chose the path based on the current load of the system to balance out the execution times as well. In addition to this, the system could use the critical paths only when they are needed to lower the risk of a bottleneck and slower processing times of the system.