## Choosing data structure for scheduling

Having a scalable and efficient simulation environment is very dependent on the data structure we use to maintain the events and the agents for scheduling. MUSE has a two tier scheduling system. The very top tier is the scheduler and it maintains the agents and knows which agent to process at any given time. The second tier is in the agent. All incoming events to a given agent must be stored and correctly sorted in increasing fashion according to the time of the delivery. Both tier seemed a great fit for the heap data structure. The heap data structures we thought of are the Fibonacci heap (Fredman and Tarjan) and the Binary Heap. Binary heaps are heaps that are implemented with binary trees (Wikipedia). Fibonacci heaps have very impressive runtime results, however these results are amortized. The following table shows the runtimes of both binary and Fibonacci heaps.

|  |  |  |
| --- | --- | --- |
| Standard Operations | Fibonacci Heap | Binary Heap |
| Insert | O(1) | O(log\*n) |
| Get Min | O(1) | O(1) |
| Delete Min | O(log\*n) <amortized> | O(log\*n) |
| Decrease Key | O(1) <amortized> | O(log\*n) |
| Delete | O(log\*n) <amortized> | O(n) |
| Merge | O(1) | O(m log(n+m)) |