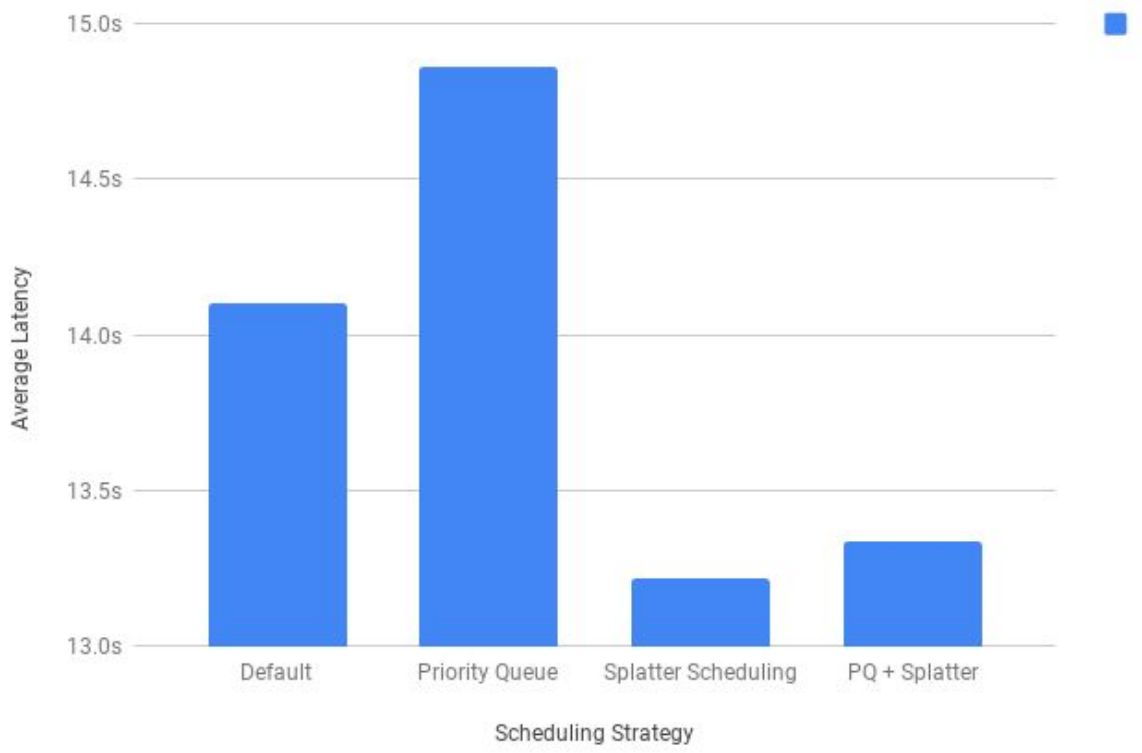
freeBSD Scheduling

A brief write up about numbers and such



Above is a graph showing average latency when running many instances of the same program using different scheduling strategies. The program computes the first n square numbers, storing them in a small circular array. A script launched 3 different versions, each running a different order of magnitude of n. Each version had 30 instances running.

Analysis:
The results we got were a bit counter intuitive. One would expect the priority queue implementation to have the lowest latency, but the opposite appears to be true. Furthermore, one expects splatter scheduling to lead to worse performance, but again our expectations are not met. This could be contributed to the overhead of running queue operations. Our PQ implementation does an insertion sort to make a tail queue behave like a priority queue. This has O(length of queue) performance, which is undesirable given how often the scheduler runs. In the case of splatter, spreading the processes out will likely cause many queues to only have one element, and thus have fewer queue ops to add/remove.

Next time..:
Results could be made more robust by running a wider variety of programs with more predictable priorities. This would allow for a more complete picture of the effect of the different strategies.