IEOR 4500
Final project, due December 16 2018
Please consult TAs for appointments

In this project we will develop a multithreaded C or C++ program for the trade execution problem.

- The program reads an input file that has the following structure. The first line has the parameters J and W. Here,
  1. J is the number of execution problems that we need to solve.
  2. W is the number of workers for the multithreaded setup.
- Following the first line we have J additional lines of the file, each describing one execution problem. Each such line contains four parameters: N, T, α, π. Here
  1. N = number of units to sell
  2. T = number of days in which to execute the sale
  3. $0 \leq \alpha$, $0 \leq \pi$ are parameters. α should be small.
- The trade execution model we use is as follows. On any given day, we can **either** sell a positive integer multiple of 10^3 units, **or** sell a positive integer multiple of 10^2 units but less than 1000 units, **or** sell between 0 and 99 units. Only one of those three choices are allowed.
- As discussed in class, suppose that on a given day we sell k units, and the initial price that day is p. Then price becomes Fp, where the factor F depends on k **and** the prior history as well. This is done as follows.
  1. Suppose k is a positive integer multiple of 10^3.
     IF at least 100 units were sold the day before, $F = 1 - \alpha(\ln k)^{3\pi}$.
     IF fewer than 100 units were sold the day before then $F = 1 - \alpha(\ln k)^{2\pi}$.
  2. Suppose k is a positive integer multiple of 10^2 and $k < 10^3$. Then $F = 1 - \alpha(\ln k)^{\pi}$.
  3. Suppose $0 \leq k \leq 99$. Then $F = 1$, i.e. there is no price reduction.
- You can use any master-worker setup, included those I described in class. Feel free to reuse code that I have provided before. In any case we want at most W workers to be actually computing at any time. Each worker's output will be the optimal revenue.

To handle the history dependent model in the case of big sales (10^3 units) you will have to develop a special version of the dynamic programming procedure.

Please remember: NO RECURSION allowed!!!