Simulation:

Generally speaking, it is a simple producer-consumer problem. The produced/consumed data can be abstracted as “floor request”. (That means, for certain floor, only the event that ‘up’ or ‘down’ is from unpressed to pressed could influence the scheduling and running of carts.)

1. Cart as consumer: Each cart should be a thread

Considering the elevators in reality, most elevators would take “greedy picking” strategy.

That means, each upgoing cart would check if there is upgoing request of the incoming floor along the way.(That is, when cart A is at floor 2 and is going to floor 7, a floor request at 5 come up at this time and the cart would pick the people there.)

This feature determines that each cart should be a thread to do the parallel linear scan.

1. Each floor’s floorRequestMgr as producer: Data feed

In reality, the waiting people would show up arbitrarily. So the best simulation is set each floorRequestMgr as a thread and generate waiting people at their own rhythm. Each floorRequestMgr has two queue for up and down people for FIFO fairness.

As we also want the input to be both from input file, or from a user interactive input(with pause). For file input, we can input an time-relevant array (idx as time) for each floor and give them to each floor to feed (maybe once per second). For interactive input, we can parse the user input and feed to certain floor(with a single feedThread or in main thread).

For pausing the program, we just set a pause cv and bool to pause each running carts when they start checking the next floor.

1. Exchanged data format

Exchanged data(floor request) can be stored and operated in a single vector(called RequestEachFloor<pair<bool,bool>>) with lock. However, the lock is definitely coarse grained. We would use a lock per floor.

1. Launching: how and when carts start running

For initial state, three carts are idle. After the first request comes, we have to pick one cart to respond to that. We use a cv and put all idle carts there waiting for new request.

Evaluation:

We can evaluate the design by average waiting time of each people. However, it may be changed by the ratio of intensity of coming people per unit time.