Simulation report 1 mm/1
075741B 桃原岳史

mm.java

この mm1 シミュレータは、次のような処理を行う。

1.客の人数、それぞれが来る時間、それぞれの処理にかかる時間のデータを乱数をもとに生成する。

2.そのデータをもとに、客がキューに入る時間と、客がキューから出る時間を計算する。

3.そのデータをもとに、客の平均待ち時間を計算する。

mm_p.java は、gnuplot にて実行結果を出力することができるようにしたものである

---1 の処理の主要なソース---

```
    for (int i=0;i<pnum;i++){
      Passengers[i] = new Passenger();
      Passengers[i].setTasktime( (rnd.nextDouble()+1.0));
      time+=rnd.nextDouble();
      Passengers[i].setArrivetime(time);
    }
```

これでは、客の処理時間は 1 から 2 までの間、客が来る間隔は 0 から 1 までの間になる。
　客の処理時間や出現頻度は、本来は実際の統計的な情報より引用されるべきかもしれない。ここでは単純に乱数で生成している。

---2 の処理の主要なソース---
長いので、まず処理の概要のみを解説する。

　シミュレータ上では２つのイベントが存在する。それは(客が入る)と(客がぬける)である。
　シミュレータが回っているあいだ、次の(客が入る)時間と、次の(客がぬける)時間が随時更新される。
シミュレータのループ処理は次のようになる。

1.次のイベントが、(客が入る)と(客がぬける)のどちらかを確認する。

2. シミュレータ上の現在時刻から、次回イベントの予定時刻までの間、列に並んだ客の分だけ待ち時間を計算する。

3.(客が入る)と(客がぬける)のどちらかが起こり、列に並んだ客の数が更新される。次回のイベントの時間も更新される。

これらを繰り返すと、いつかデータ上の最後の客が来店する。
こうなると(客が抜ける)以外のイベントは発生しない。そうなると。

4.場合分けを考慮しない計算方法に移行する。
この計算はすぐに終わる。

シミュレーションの結果を GNUPLOT で出力した図。
x 軸が時間、y が待ち人数。
y=-1 は客の処理をしていない(暇な)状態を表している。
平均待ち時間は　(待ち行列の長さ)×時間　の総和から計算されるが、
y=-1 の部分は計算からスキップされるので、-1 の値は 0 と同じふるまいをする