Performance Modeling of Computer Systems and Networks

Prof. Vittoria de Nitto Personè

Case study 2
An Inventory system

Università degli studi di Roma Tor Vergata
Department of Civil Engineering and Computer Science Engineering

1

A simple inventory system

- Distributes items from current inventory to customers
- Customer demand is discrete
- Simple: one type of item

2

1. Goals
2. Conceptual model
3. Specification model
4. Computational model
5. Verify
6. Validate

Discrete-event simulation
Trace-driven simulation

Inventory policy

- **Transaction reporting**
  – Inventory review after each transaction
  – Significant labor may be required
  – Less likely to experience “shortage”
- **Periodic inventory review**
  – Inventory review is periodic
  – Items are ordered, if necessary, only at review times
  – (s, S) are the min,max inventory levels, 0≤s<S

Search for (s, S) that minimize cost

Prof. Vittoria de Nitto Personè 3

3

1. Goals
2. Conceptual model
3. Specification model
4. Computational model
5. Verify
6. Validate

Discrete-event simulation
Trace-driven simulation

Inventory System Costs
- *Holding* cost: for items in inventory
- *Shortage* cost: for unmet demand
- Setup cost: fixed cost when order is placed
- Item cost: per-item order cost
- Ordering cost: sum of setup and item costs

Additional Assumptions
- Back ordering is possible
- No delivery lag
- Initial inventory level is S
- Terminal inventory level is S

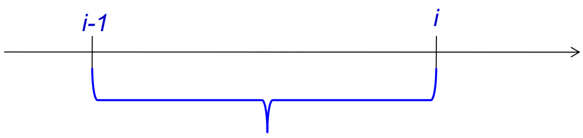
l : inventory level
o : amount ordered
d : demand quantity

Prof. Vittoria de Nitto Personè 4

4

1. Goals
2. Conceptual model
3. Specification model
4. Computational model
5. Verify
6. Validate

- Time begins at t=0
- Review times are t=0, 1, 2, …
- l_{i-1}: inventory level at beginning of i^{th} interval
- o_{i-1}: amount ordered at time t=i-1, ($o_{i-1} \geq 0$)
- d_i: demand quantity during i^{th} interval ($d_i \geq 0$)
- Inventory at end of interval can be negative (back ordering)



5

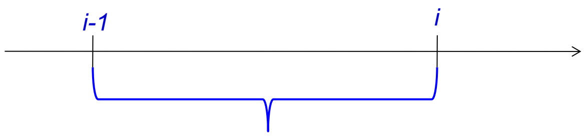
Inventory level considerations

- Inventory level is reviewed at t=i-1
- If at least s, no order is placed
  if less than s, inventory is replenished to S:

$$o_{i-1} = \begin{cases} 0 & l_{i-1} \geq s \\ S - l_{i-1} & l_{i-1} < s \end{cases}$$

- Items are delivered immediately
- At end of i^{th} interval, inventory diminished by d_i

$$l_i = l_{i-1} + o_{i-1} - d_i$$



6

Discrete-event simulation
Trace-driven simulation

Time Evolution of Inventory Level

Algorithm 3: trace-driven simulation

$l_0 = S;$ /* the initial inventory level is S */
$i = 0;$
while (more demand to process) {
    i++;
    if ($l_{i-1} < s$)
        $o_{i-1} = S - l_{i-1}$;
    else
        $o_{i-1} = 0$;
    d_i = GetDemand(); ← Read data from file
    $l_i = l_{i-1} + o_{i-1} - d_i$;
    }
    $n = i$;
    $o_n = S - l_n$;
    $l_n = S$; /* the terminal inventory level is S */

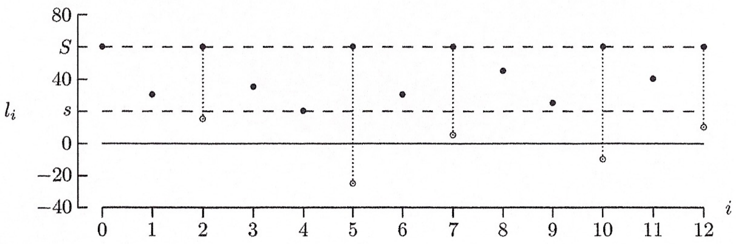
return $l_1, l_2, \ldots, l_n$ and $o_1, o_2, \ldots, o_n$;

Discrete-event simulation
Trace-driven simulation

example

Let (s, S)=(20, 60) and consider n=12 time intervals, and the following input demands:

i :	1	2	3	4	5	6	7	8	9	10	11	12
d_i :	30	15	25	15	45	30	25	15	20	35	20	30



- What statistics to compute?
- *average demand* and *average order*

$$\bar{d} = \frac{1}{n}\sum_{i=1}^{n} d_i \qquad \bar{o} = \frac{1}{n}\sum_{i=1}^{n} o_i$$

- For our example data:

$$\bar{d} = \bar{o} = 305/12 \approx 25.42$$ items per time interval

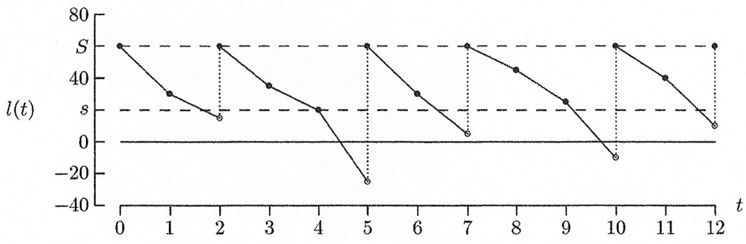
9

- Average demand and order *must* be equal
- Ending inventory level is S
- Over the simulated period, all demand is satisfied
- Average “flow” of items in equals average “flow” of items out

Flow balance

10

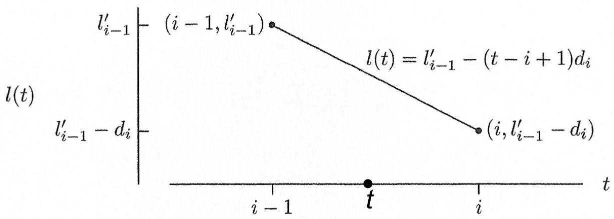
- Holding and shortage costs are proportional to time-averaged inventory levels
- Must know inventory level for all *t* (not only at review times)
- Assume the demand rate is constant between review times



The inventory level at any time t in i^{th} interval is

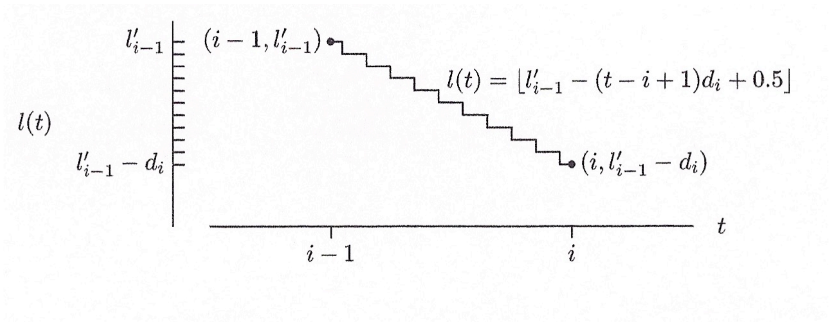
$$l(t) = l'_{i-1} - (t - i + 1)d_i$$

If demand rate is constant between review times



$l'_{i-1} = l_{i-1} + o_{i-1}$ represents inventory level *after* review

- Inventory level at any time *t* is an integer
- *l*(*t*) should be rounded to an integer value
- *l*(*t*) is a stair-step function



Time-Averaged Inventory Level

Case 1

l(*t*) remains non-negative over i^{th} interval

$$\bar{l}_i^+ = \int_{i-1}^{i} l(t)dt$$

Case 2

l(*t*) becomes negative at some time τ

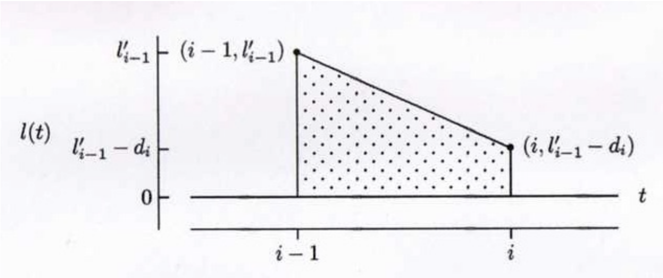
$$\bar{l}_i^+ = \int_{i-1}^{\tau} l(t)dt \qquad \bar{l}_i^- = -\int_{\tau}^{i} l(t)dt$$

$\bar{l}_i^+$ is the time-averaged *holding level*

$\bar{l}_i^-$ is the time-averaged *shortage level*

Case 1 no back-ordering

No shortage during i^{th} time interval iff $d_i \leq l'_{i-1}$

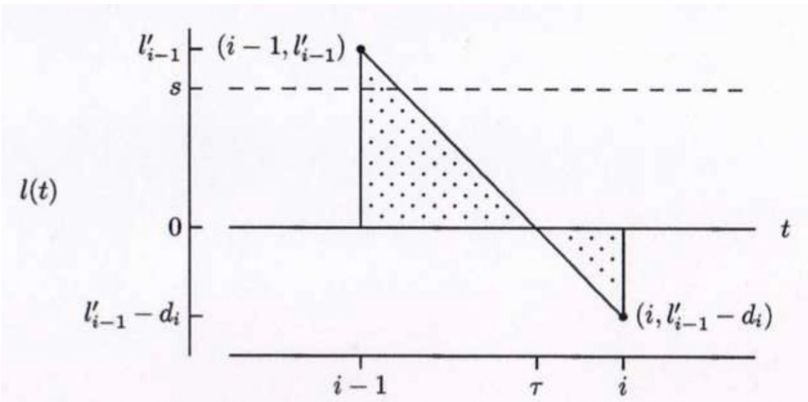


Time-averaged holding level is computed as area of a trapezoid:

$$\bar{l}_i^+ = \int_{i-1}^{i} l(t)dt = \frac{l'_{i-1} + \left(l'_{i-1} - d_i\right)}{2} = l'_{i-1} - \frac{1}{2}d_i$$

Case 2 back-ordering

Inventory becomes negative iff $d_i > l'_{i-1}$



Case 2 back-ordering (cont.)

- $l(t)$ becomes negative at time $t = \tau = i\text{-}1+(l'_{i-1}/\ d_i)$

- time-averaged holding and shortage levels for i^{th} interval computed as the areas of triangles

$$\bar{l}_i^+ = \int_{i-1}^{\tau} l(t)dt = \ldots = \frac{\left(l'_{i-1}\right)^2}{2d_i}$$

$$\bar{l}_i^- = -\int_{\tau}^{i} l(t)dt = \ldots = \frac{\left(d_i - l'_{i-1}\right)^2}{2d_i}$$

Time-Averaged Inventory Level

- *Time-averaged holding level* and *time-averaged shortage level*

$$\bar{l}^+ = \frac{1}{n}\sum_{i=1}^{n}\bar{l}_i^+ \qquad \bar{l}^- = \frac{1}{n}\sum_{i=1}^{n}\bar{l}_i^-$$

- The *time-averaged inventory level* is

$$\bar{l} = \frac{1}{n}\int_0^n l(t)dt = \bar{l}^+ - \bar{l}^-$$

Goals
Conceptual model
Specification model
Computational model
Verify
Validate

Discrete-event simulation

- Computes the statistics
  $\bar{d}, \bar{o}, \bar{l}^+, \bar{l}^-$

  and the order frequency

  $$\bar{u} = \frac{n.\,of\ orders}{n}$$

- Consistency check:
  compute $\bar{o}, \bar{d}$ separately, then compare

Prof. Vittoria de Nitto Personè 19

19

sis1.c

```
#include <stdio.h>

#define FILENAME  "sis1.dat"
#define MINIMUM   20
#define MAXIMUM   80
#define sqr(x)    ((x) * (x))

long GetDemand(FILE *fp)
{
       long d;
       fscanf(fp, "%ld\n", &d);
       return (d);}
```

20

```c
int main(void)
{
   FILE *fp;
   long index     = 0;
   long inventory = MAXIMUM;
   long demand;
   long order;
   struct {
        double setup;
        double holding;  /* inventory held (+) */
        double shortage; /* inventory short (-) */
        double order;
       double demand;
   } sum = { 0.0, 0.0, 0.0, 0.0, 0.0 };

   fp = fopen(FILENAME, "r");
   if (fp == NULL) {    fprintf(stderr, "Cannot open input file
   %s\n", FILENAME);
   return (1);
}
```



```c
   while (!feof(fp)) {
     index++;
     if (inventory < MINIMUM) {
          order = MAXIMUM - inventory;
          sum.setup++;
          sum.order     += order;
     }
     else order          = 0;
     inventory += order;      /* there is no delivery lag */
     demand=GetDemand(fp);
     sum.demand      += demand;
     if (inventory > demand)
          sum.holding  += (inventory - 0.5 * demand);
     else {
          sum.holding  += sqr(inventory) / (2.0 * demand);
          sum.shortage += sqr(demand - inventory) / (2.0 * demand);
     }
     inventory      -= demand;
  }
```

```c
 if (inventory < MAXIMUM) {
        order = MAXIMUM - inventory;
        sum.setup++;
        sum.order+= order;
        inventory+= order;
   }

   printf("\nfor %ld time intervals ", index);
   printf("with an average demand of %6.2f\n", sum.demand / index);
   printf("and policy parameters (s, S) = (%d, %d)\n\n", MINIMUM, MAXIMUM);
   printf("   average order ........... = %6.2f\n", sum.order / index);
   printf("   setup frequency ......... = %6.2f\n", sum.setup / index);
   printf("   average holding level .... = %6.2f\n", sum.holding / index);
   printf("   average shortage level ... = %6.2f\n", sum.shortage / index);

   fclose(fp);
   return (0);
}
```


- trace file sis1.dat contains data for n=100 time intervals
- with (s, S)=(20, 80)

$$\bar{o} = \bar{d} = 29.29 \qquad \bar{u} = 0.39 \qquad \bar{l}^{+} = 42.40 \qquad \bar{l}^{-} = 0.25$$

Operating Costs

A facility's cost of operation is determined by:

- c_{item} : unit cost of new item
- c_{setup} : fixed cost for placing an order (setup cost)
- c_{hold} : cost to hold one item for one time interval
- c_{short} : cost of being short one item for one time interval

Case study

- Automobile dealership that uses weekly periodic inventory review
- The facility is the showroom and surrounding areas
- The items are new cars
- The supplier is the car manufacturer
- Simple inventory system: one type of car

Case study

- Limited to a maximum of S=80 cars
- Inventory reviewed every Monday
- If inventory falls below s=20, order cars sufficient to restore to S=80
- For now, ignore delivery lag
- Costs:
  - Item cost is c_{item} = \$8000 per item
  - Setup cost is c_{setup} = \$1000
  - Holding cost is c_{hold} = \$25 per week
  - Shortage cost is c_{short} = \$700 per week

Per-Interval Average Operating Costs

- The average operating costs per *time interval* are:
  - item cost: $c_{item} \cdot \bar{o}$
  - setup cost: $c_{setup} \cdot \bar{u}$
  - holding cost: $c_{hold} \cdot \bar{l}^{+}$
  - shortage cost: $c_{short} \cdot \bar{l}^{-}$
- The average *total* operating cost *per time interval* is their sum
- For the stats and costs of the hypothetical dealership:
  - item cost \$8000·29.29 = \$234.320 per week
  - setup cost \$1000·0.39 = \$390 per week
  - holding cost \$25·42.40 = \$1.060 per week
  - shortage cost \$700·0.25 = \$175 per week

Cost Minimization

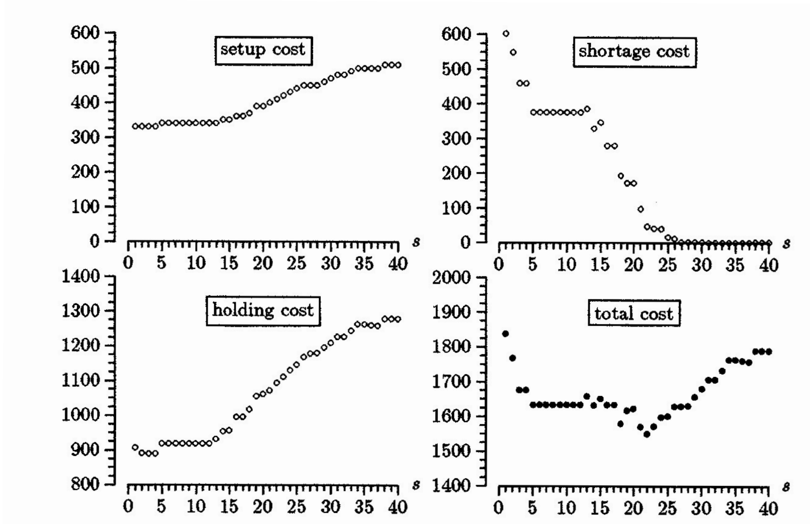
- By varying s (and possibly S), an optimal policy can be determined
- Optimal ⇔ minimum average cost
- Note that $\bar{o} = \bar{d}$ and $\bar{d}$ depends only on the demands. Hence, item cost is independent of (s, S)
- Average dependent cost is:

  avg setup cost + avg holding cost + avg shortage cost

- Let S and the demand sequence be fixed
- If <u>s is systematically increased</u>, we expect:
  - average setup cost and holding cost will increase
  - average shortage cost will decrease
- average dependent cost will have 'U' shape, yielding an optimum

  avg setup cost +
  avg holding cost +
  avg shortage cost
- From results (next slide), minimum cost is $1550 at s=22

Discrete-event simulation
invs – case study



31

Discrete-event simulation
Trace-driven simulation

Exercises

- Study program sis1.c
- Run with sis1.dat; analyze the results
- Generate a data file with the values on p.8
- Run with the new data, verify that the results confirm the expectations
- Ex 1.3.1 and Ex 1.3.8 on p.36 of the textbook

32