UNIVERSIDADE FEDERAL DE UBERLÂNDIA – UFU

Graduação em Ciência da Computação

Atividade Prática 01

GBC065 – Modelagem e Simulação

Uberlândia
2018

Faculdade de Computação

UFU

Atividade Prática 01

Trabalho apresentado à disciplina de Modelagem e Simulação (GBC065), ministrada pelo professor Anderson Rodrigues dos Santos, para o curso de Bacharelado em Ciência da Computação, no período 2018-2, na Universidade Federal de Uberlândia.

Grupo 02 – Integrantes:

Antonio Carlos Neto
11611BCC054

Ronistone Gonçalves dos Reis Júnior
11521BCC018

Uberlândia
2018

Exercise 1.2.2 :

(a) Modify program ssq1 to output the additional statistics (¯l), (¯q), and (¯x).

R: Conforme a imagem a seguir, temos que:

- (¯x) é 0.72;
- (¯q) é 1.88;
- (¯l) é 2.60.

```
C:\Windows\System32\cmd.exe

C:\Users\netii\Documents\MS\code>gcc ssq1.c -o teste1

C:\Users\netii\Documents\MS\code>teste1 < ssq1.dat

for 1000 jobs
   average interarrival time................ =   9.87
   average service time .................... =   7.12
   average delay ........................... =  18.59
   average wait ............................ =  25.72
   time-averaged number in service.......... =   0.72
   time-averaged number in the queue........ =   1.88
   time-averaged number in the node......... =   2.60

C:\Users\netii\Documents\MS\code>
```

(b) Similar to the case study, use this program to compute a table of (¯l), (¯q), and (¯x) for traffic intensities of 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, and 1.2.

R: Considerando que:

$$\frac{1/\bar{r}}{1/\bar{s}} = \frac{\bar{s}}{\bar{r}} = \frac{\bar{s}}{a_n/n} = \left(\frac{c_n}{a_n}\right)\bar{x}.$$

temos que:

service = GetService(fp) * (timeService[i] / 0.72);

sendo que o timeService[i] é o tráfico de intensidade escolhido, 0.6 a 1.2, e 0.72 é o tráfego padrão do exemplo ssq1.dat. Portanto a cada leitura estamos modificando o tempo que o job ficará no processador, aumentando ou diminuindo todos proporcionalmente, e assim alterando o tráfego.

```
C:\Windows\System32\cmd.exe

C:\Users\netii\Downloads\trabalho1>gcc 1-2-2.c -o 1-2-2

C:\Users\netii\Downloads\trabalho1>1-2-2 < ssq1.dat

for 1000 jobs with traffic intensity 0.60
   average interarrival time............... =   9.87
   average service time ................... =   5.94
   average delay .......................... =   9.57
   average wait ........................... =  15.51
   time-averaged number in service......... =   0.60
   time-averaged number in the queue....... =   0.97
   time-averaged number in the node........ =   1.57
   traffic intensity is ................... =   0.60
   traffic intensity is ................... =   0.60

for 1000 jobs with traffic intensity 0.70
   average interarrival time............... =   9.87
   average service time ................... =   6.93
   average delay .......................... =  16.67
   average wait ........................... =  23.60
   time-averaged number in service......... =   0.70
   time-averaged number in the queue....... =   1.68
   time-averaged number in the node........ =   2.38
   traffic intensity is ................... =   0.70
   traffic intensity is ................... =   0.70

for 1000 jobs with traffic intensity 0.80
   average interarrival time............... =   9.87
   average service time ................... =   7.92
   average delay .......................... =  29.89
   average wait ........................... =  37.81
   time-averaged number in service......... =   0.80
   time-averaged number in the queue....... =   3.02
   time-averaged number in the node........ =   3.82
   traffic intensity is ................... =   0.80
   traffic intensity is ................... =   0.80

for 1000 jobs with traffic intensity 0.90
   average interarrival time............... =   9.87
   average service time ................... =   8.91
   average delay .......................... =  75.27
   average wait ........................... =  84.18
   time-averaged number in service......... =   0.90
   time-averaged number in the queue....... =   7.58
   time-averaged number in the node........ =   8.48
   traffic intensity is ................... =   0.90
   traffic intensity is ................... =   0.90
```

```
for 1000 jobs with traffic intensity 0.90
   average interarrival time............... =   9.87
   average service time ................... =   8.91
   average delay .......................... =  75.27
   average wait ........................... =  84.18
   time-averaged number in service......... =   0.90
   time-averaged number in the queue....... =   7.58
   time-averaged number in the node........ =   8.48
   traffic intensity is ................... =   0.90
   traffic intensity is ................... =   0.90

for 1000 jobs with traffic intensity 1.00
   average interarrival time............... =   9.87
   average service time ................... =   9.90
   average delay .......................... = 264.05
   average wait ........................... = 273.95
   time-averaged number in service......... =   0.99
   time-averaged number in the queue....... =  26.50
   time-averaged number in the node........ =  27.50
   traffic intensity is ................... =   1.00
   traffic intensity is ................... =   1.00

for 1000 jobs with traffic intensity 1.10
   average interarrival time............... =   9.87
   average service time ................... =  10.89
   average delay .......................... = 766.37
   average wait ........................... = 777.25
   time-averaged number in service......... =   0.99
   time-averaged number in the queue....... =  70.04
   time-averaged number in the node........ =  71.04
   traffic intensity is ................... =   1.10
   traffic intensity is ................... =   1.10

for 1000 jobs with traffic intensity 1.20
   average interarrival time............... =   9.87
   average service time ................... =  11.87
   average delay .......................... = 1268.75
   average wait ........................... = 1280.63
   time-averaged number in service......... =   1.00
   time-averaged number in the queue....... = 106.42
   time-averaged number in the node........ = 107.41
   traffic intensity is ................... =   1.20
   traffic intensity is ................... =   1.20
  X       Q       L
 0.60    0.97    1.57
 0.70    1.68    2.38
 0.80    3.02    3.82
 0.90    7.58    8.48
 0.99   26.50   27.50
 0.99   70.04   71.04
 1.00  106.42  107.41
```

(c) Comment on how (¯l), (¯q), and (¯x) depend on the traffic intensity.

R: Quanto maior o tráfego de intensidade, temos que maior o tempo que o server fica ocupado. Tomando que **(¯x)** seja o número médio de jobs no server em um determinado tempo, então para que o server fique mais ocupado, temos que o **(¯x)** deve aumentar. Considerando um aumento no uso do server, podemos concluir que a probabilidade dos jobs chegarem e encontrarem o server ocupado é maior, assim o **$(\bar{q})$** aumentará. Como **$(\bar{l}) = (\bar{q}) + (\bar{x})$**, temos que quanto maior o tráfego maior o **$(\bar{l})$**.

(d) Relative to the case study, if it is decided that $(\bar{q})$ greater than 5.0 is not acceptable, what systematic increase in service times would be acceptable?

R: Consultando a tabela abaixo, temos que a partir de 0.85 não é aceitável.

X	Q	L
0.81	3.30	4.11
0.82	3.64	4.45
0.83	3.99	4.82
0.84	4.35	5.19
0.85	4.75	5.59
0.86	5.22	6.07
0.87	5.75	6.61

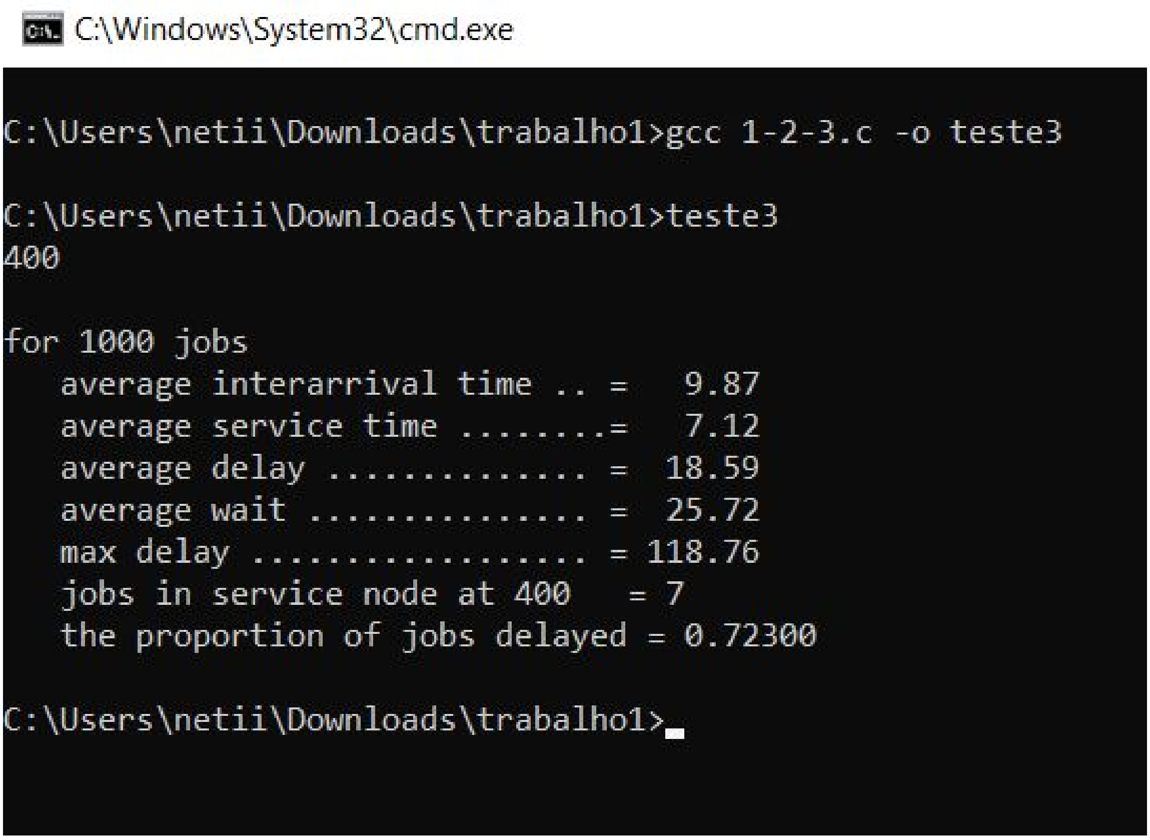
Exercise 1.2.3 :

(a) Modify program ssq1 by adding the capability to compute the maximum delay, the number of jobs in the service node at a specified time (known at compile time) and the proportion of jobs delayed.

R: O cálculo do máximo delay é fácil, já que a cada iteração comparamos o delay do job atual com o máximo. O cálculo do número de jobs dentro do nó de serviço em um determinado tempo consiste basicamente em analisar se o tempo de entrada é maior ou igual ao tempo lido e se o tempo de saída é menor que o tempo lido, analisando a cada iteração temos que um incremento na variável inServiceNode. A proporção de jobs que entraram na fila é calculado na comparação do tempo de chegada e o tempo de saída do último nó, departure, assim dividimos essa variável pela quantidade de nós(index).

(b) What was the maximum delay experienced?

R: O máximo delay é 118.76.



```
C:\Windows\System32\cmd.exe

C:\Users\netii\Downloads\trabalho1>gcc 1-2-3.c -o teste3

C:\Users\netii\Downloads\trabalho1>teste3
400

for 1000 jobs
   average interarrival time .. =   9.87
   average service time ........=   7.12
   average delay .............. =  18.59
   average wait ............... =  25.72
   max delay .................. = 118.76
   jobs in service node at 400    = 7
   the proportion of jobs delayed = 0.72300

C:\Users\netii\Downloads\trabalho1>_
```

(c) How many jobs were in the service node at t = 400 and how does the computation of this number relate to the proof of Theorem 1.2.1?

R: No tempo 400 tem 7 jobs, podendo ser visto na imagem do item b. Essa relação é exatamente l(t), ou seja, foi implementado a função l(t) e calculado l(t) para t = 400.

(d) What proportion of jobs were delayed and how does this proportion relate to the utilization?

R: Quanto maior a utilização, maior essa proporção. Já que os jobs ficam mais tempo no server, implica que aumentará a chance dos próximos jobs terem que esperar na fila.

Exercise 1.2.6 :

The text file ac.dat consists of the arrival times $a_1, a_2, \ldots, a_n$ and the departure times $c_1, c_2, \ldots, c_n$ for $n = 500$ jobs in the format

$$\begin{array}{cc} a_1 & c_1 \\ a_2 & c_2 \\ \ldots & \ldots \\ a_n & c_n \end{array}$$

(a) If these times are for an initially idle single-server FIFO service node with infinite capacity, calculate the average service time, the server's utilization and the traffic intensity.

R: Tomando em consideração que o tempo de serviço(S_i) de um job(i) pode ser calculado da seguinte forma, considerando que temos o tempo de chegada(A_i) e saída(S_i) do job anterior.

$$S_i = C_i - C_{i-1} \text{ se } A_i < C_{i-1}$$
$$S_i = C_i - A_i \text{ se } A_i >= C_{i-1}$$

Calculando a soma do tempo de interarrival($A_i - A_{i-1}$), a soma do tempo de serviço(S_i) e o tempo de saída(C_n) do último job(n), temos capacidade de calcular tudo solicitado.

C:\Windows\System32\cmd.exe

```
C:\Users\netii\Downloads\trabalho1>gcc 1-2-6.c -o 1-2-6

C:\Users\netii\Downloads\trabalho1>1-2-6 < ac.dat
        average of service time is        3.03
        average of interarrival is        4.08
        server utilization is             0.74
        the traffic intensity is          0.74

C:\Users\netii\Downloads\trabalho1>
```

(b) Be explicit: for i = 1, 2, . . . , n how does si relate to ai−1, ai , ci−1, and ci?

R: Conforme explicado no item anterior, a partir desses valores conseguimos calcular todos os outros, mostrado pela figura 1.2.4 e 1.2.5 do livro texto base.

- **Case I.** If $a_i < c_{i-1}$, i.e., if job i arrives before job $i-1$ departs then, as illustrated, job i will experience a delay of $d_i = c_{i-1} - a_i$. Job $i-1$'s history is displayed above the time axis and job i's history is displayed below the time axis in Figures 1.2.4 and 1.2.5.

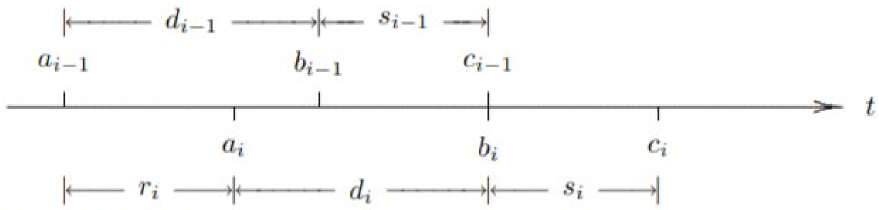


Figure 1.2.4. Job i arrives before job $i-1$ departs.

- **Case II.** If instead $a_i \geq c_{i-1}$, i.e., if job i arrives after (or just as) job $i-1$ departs then, as illustrated, job i will experience no delay so that $d_i = 0$.

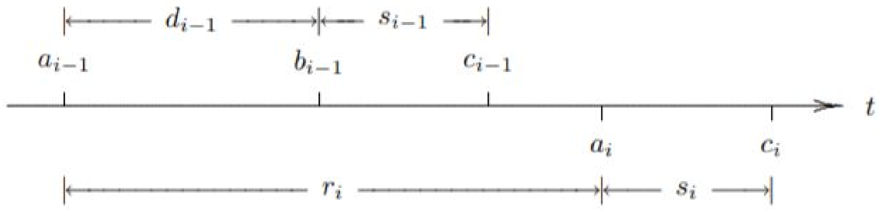


Figure 1.2.5. Job i arrives after job $i-1$ departs.

Exercise 1.2.8 :

(a) Similar to Exercise 1.2.2, modify program ssq1 to output the additional statistics (¯l), (¯q), and (¯x).

R: Copiando o código ssq1 e alterando o retorno da função GetService, colocando uma constante, conseguimos atender o solicitado. Exemplo: Constante = 10.0;

C:\Windows\System32\cmd.exe

```
C:\Users\netii\Downloads\trabalho1>gcc 1-2-8-a.c -o teste4

C:\Users\netii\Downloads\trabalho1>teste4

for 1000 jobs
   average interarrival time................ =   9.87
   average service time .................... =  10.00
   average delay ........................... = 138.46
   average wait ............................ = 148.46
   time-averaged number in service.......... =   0.99
   time-averaged number in the queue........ =  13.71
   time-averaged number in the node......... =  14.70

C:\Users\netii\Downloads\trabalho1>
```

(b) By using the arrival times in the file ssq1.dat and an appropriate constant service time in place of the service times in the file ssq1.dat, use the modified program to compute a table of ($\bar{l}$), ($\bar{q}$), and ($\bar{x}$) for traffic intensities of 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, and 1.2.

R: Mantendo as alterações do item anterior e incluindo o código usado na 1.2.2-b, temos que a grande mudança foi:

```
service = GetService(fp) *(timeService[i]/(SERVICETIME/9.87));
```

Basicamente fizemos a mesma mudança do exercício 1.2.2-b com a diferença de calcularmos o traffic intensities padrão com base na constante lida(SERVICETIME), lembrando que o **($\bar{r}$)** é constante, já que o tempo de chegada não é alterado, sendo 9.87.

(c) Comment on how ($\bar{l}$), ($\bar{q}$), and ($\bar{x}$) depend on the traffic intensity.

R: Quanto maior o tráfego de intensidade, temos que maior o tempo que o server fica ocupado. Tomando que **($\bar{x}$)** seja o número médio de jobs no server em um determinado tempo, então para que o server fique mais ocupado, temos que o **($\bar{x}$)** deve aumentar. Considerando um aumento no uso do server, podemos concluir que a probabilidade dos jobs chegarem e encontrarem o server ocupado é maior, assim o **($\bar{q}$)** aumentará. Como **($\bar{l}$) = ($\bar{q}$) + ($\bar{x}$)**, temos que quanto maior o tráfego maior o **($\bar{l}$)**.

Exercise 1.3.1 :

Verify that the results in Example 1.3.1 and the averages in Examples 1.3.2 and 1.3.3 are correct.

R:

- Example 1.3.1:
  - s = 20, S = 60, n = 12;
  - On = 50, portanto, o exemplo 1.3.1 está correto;

i	1	2	3	4	5	6	7	8	9	10	11	12	Total
d	30	15	25	15	45	30	25	15	20	35	20	30	305
li	30	15	35	20	-25	30	5	45	25	-10	40	10	-
oi	0	45	0	0	85	0	55	0	0	70	0	50	305
l'i+	45	22.5	47.5	27.5	4.45	45	17.5	52.5	35	8.93	40	25	370.88
l'i-	0	0	0	0	6.94	0	0	0	0	1.42	0	0	8.36

- Example 1.3.2:
  - **(¯d) = (¯o) = 305/12 ~= 25.42**;
  - $\Sigma di = \Sigma oi$, portanto, o exemplo 1.3.2 está correto;

- Example 1.3.3:
  - **(¯l+) = 31.74, (¯l-) = 0.70**;
  - Average number of items held = 31.74;
  - Average number of items short = 31.04 = (370.88/12);
  - Average inventory level was = 0.70 = (8.36/12);
  - O exemplo está correto.

Exercise 1.3.2 :

(a) Using the cost constants in Example 1.3.5, modify program sis1 to compute all four components of the total average cost per week.

R: Utilizando o Example 1.3.5, Definition 1.3.6 e Example 1.3.6, obtivemos os seguintes resultados. Lembrando que houve uma diferença de arredondamento entre o Example 1.3.6 e o programa sis1 modificado.

C:\Windows\System32\cmd.exe

```
C:\Users\netii\Desktop\final>gcc sis1.c -o teste7

C:\Users\netii\Desktop\final>teste7

for 100 time intervals with an average demand of  29.29
and policy parameters (s, S) = (20, 80)

   average order .................. =  29.29
   setup frequency ................ =   0.39
   average holding level .......... =  42.40
   average shortage level ......... =   0.25
   average cost per week Citem .... = 234320.00
   average cost per week Csetup ... = 390.00
   average cost per week Chold .... = 1060.03
   average cost per week Cshort ... = 172.47

C:\Users\netii\Desktop\final>_
```

Tabela de preços bases do Example 1.3.5:

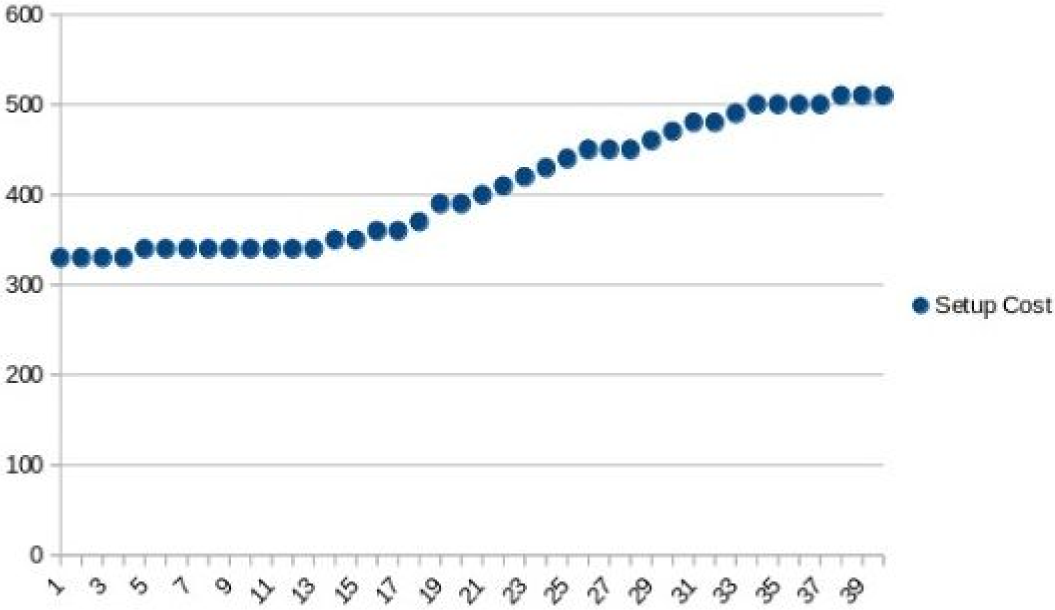
Citem	\$8,000.00
Csetup	\$1,000.00
Chold(one week)	\$25.00
Cshort(one week)	\$700.00

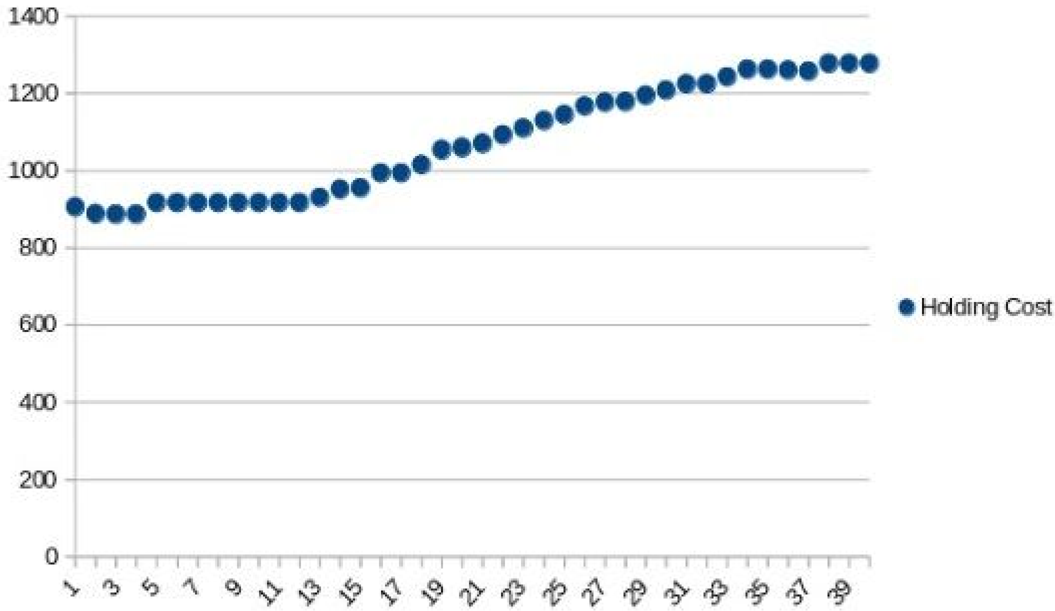
(b) These four costs may differ somewhat from the numbers in Example 1.3.6. Why?

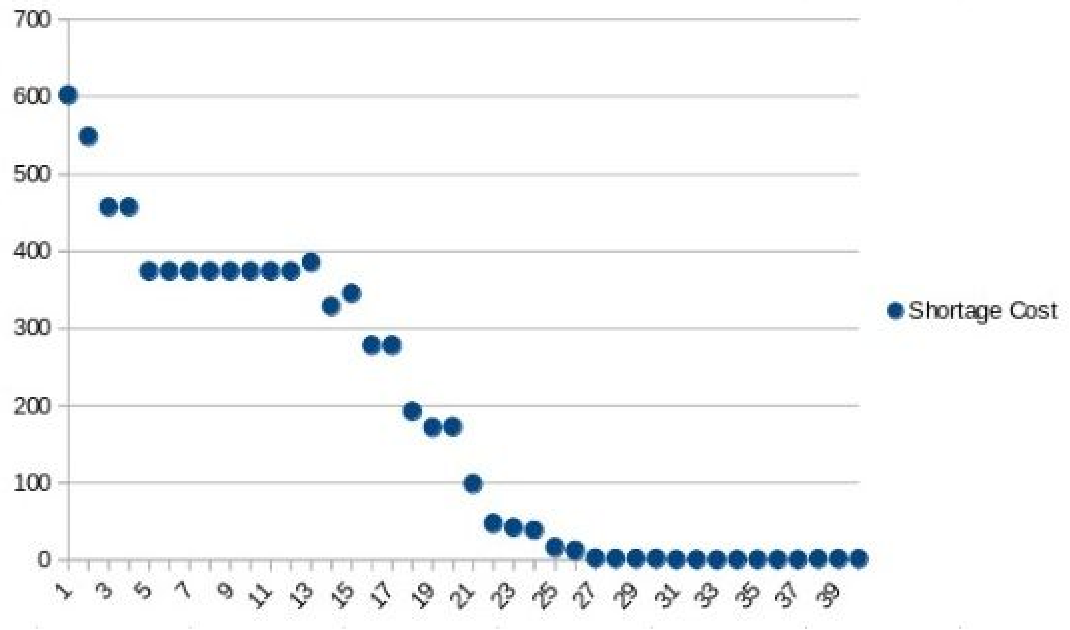
R: Sim, já que o Example 1.3.6 utiliza os valores arredondados para efetuar esse cálculo, enquanto o programa sis1.c modificado efetua a conta para depois arredondar.

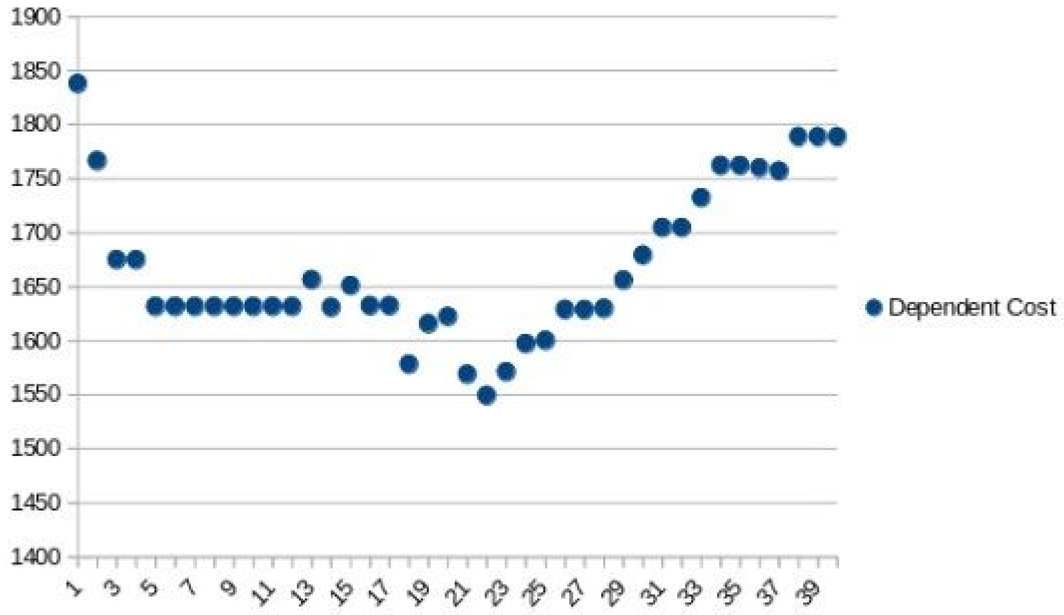
(c) By constructing a graph like that in Example 1.3.7, explain the trade-offs involved in concluding that s = 22 is the optimum value (when S = 80).

R: Modificando o código sis1.c, incluímos um laço de repetição de s mínimo até s máximo, simulando Setup Cost, Holding Cost, Shortage Cost, Dependent Cost. Assim concluímos que com o s mínimo = 1 e s máximo = 40, temos que o menor Dependent Cot(1549.29) é quando o s = 22.









(d) Comment on how well-defined this optimum is.

R: Considerando s sendo um inteiro, temos que ele(s = 22) é ótimo se mantermos o S fixo, ou seja, para todos valores no intervalo 0<s<41, com S = 80, ele será ótimo, se modificarmos o S, esse valor pode deixar de ser ótimo.

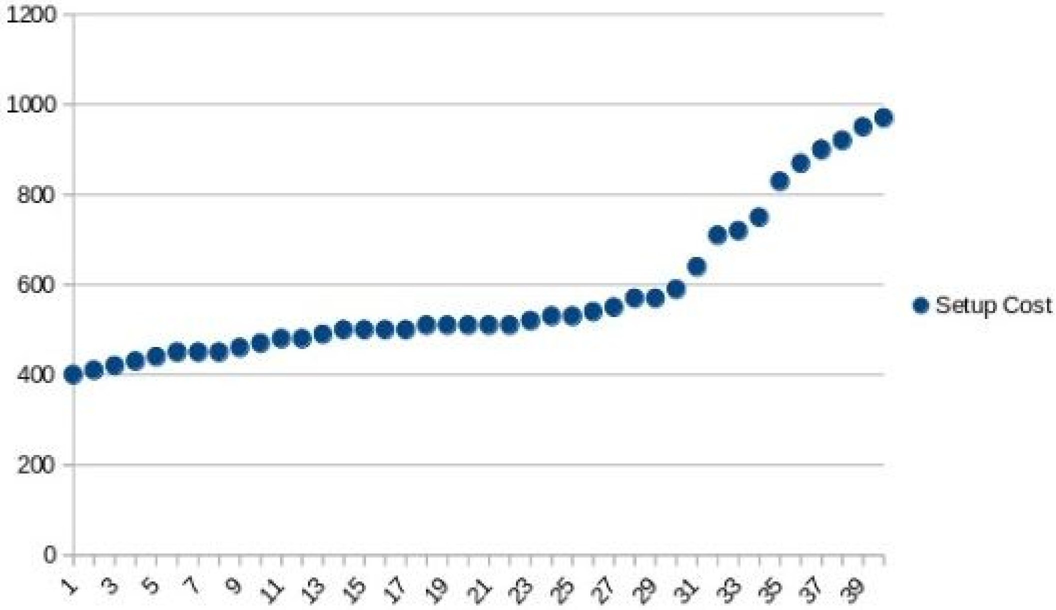
Exercise 1.3.4 :

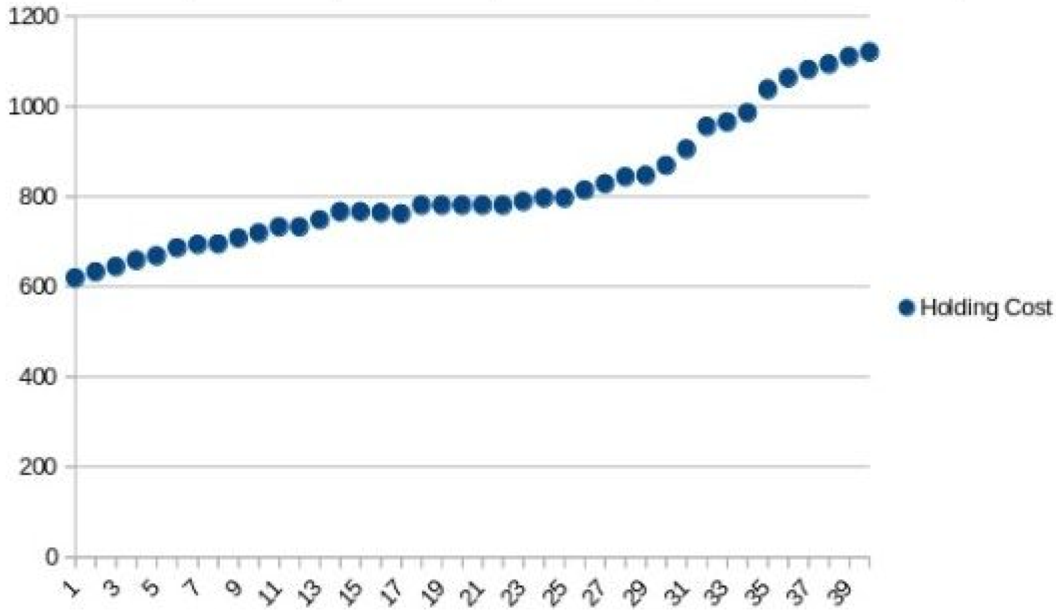
(a) Construct a table or figure similar to Example 1.3.7 but for S = 100 and S = 60.

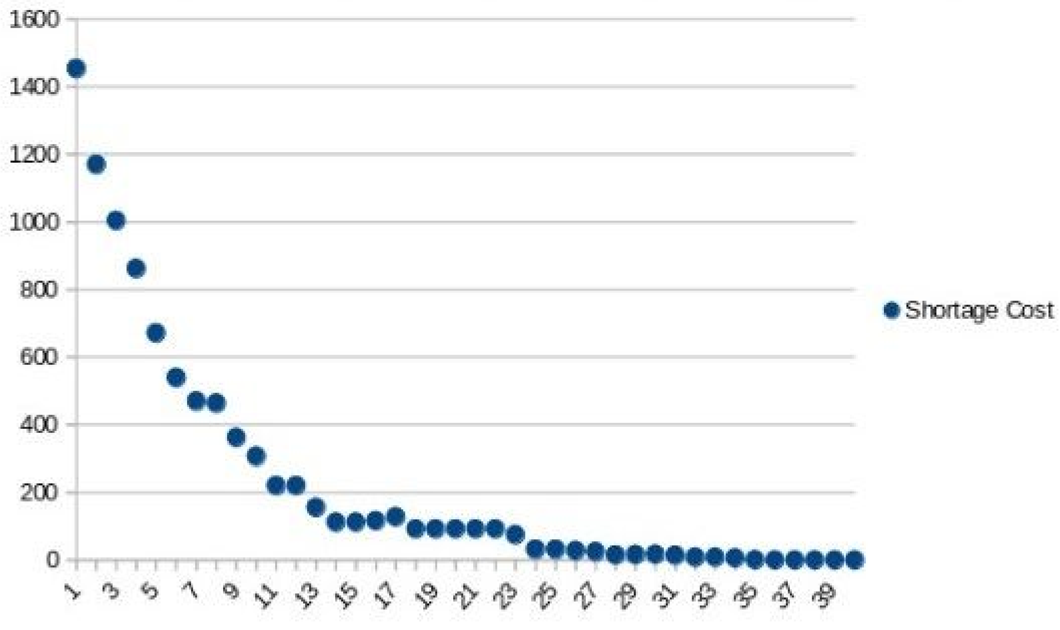
R:

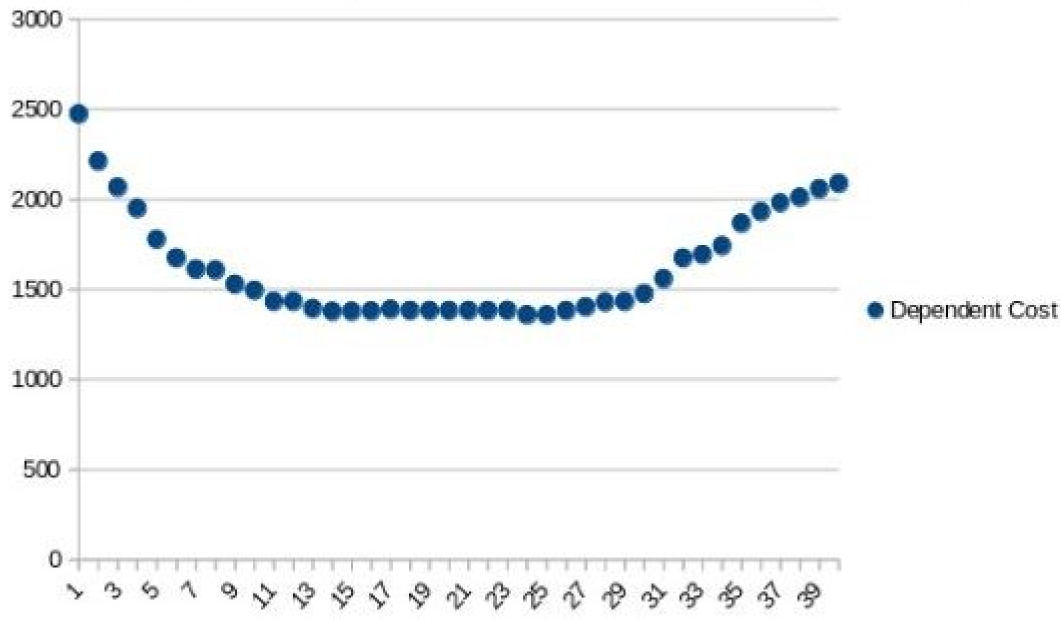
- Com S = 100 e sabendo que s é inteiro e 0<s<41, temos que o s ótimo é 20 e seu Dependent Cost = 1674.70.
- Com S = 60 e sabendo que s é inteiro e 0<s<41, temos que o s ótimo é 24 e seu Dependent Cost = 1358.76.

- Os gráficos seguintes são do S = 60, com 0 < s < 41:

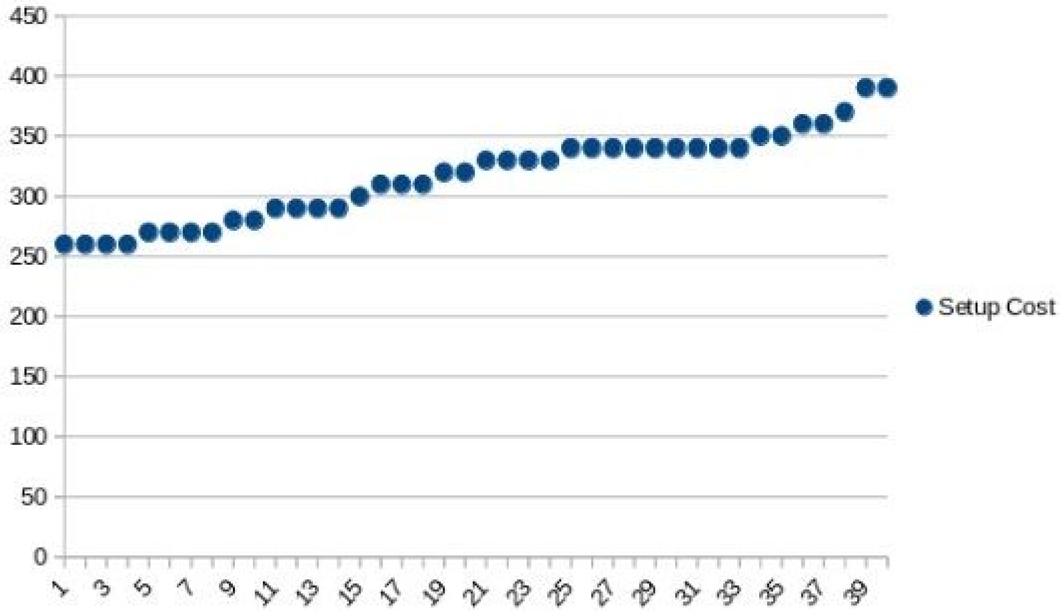




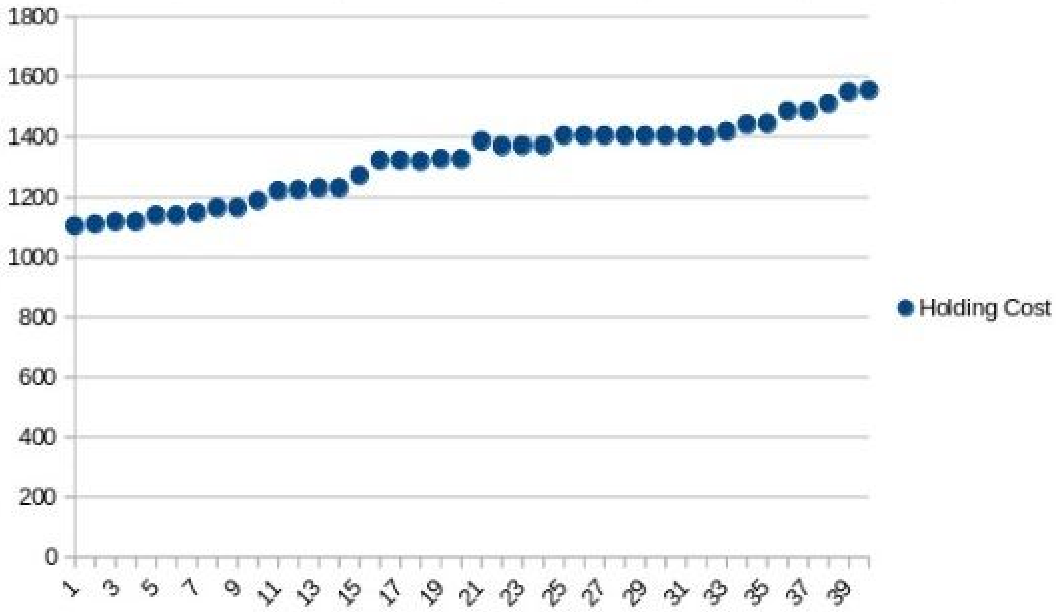




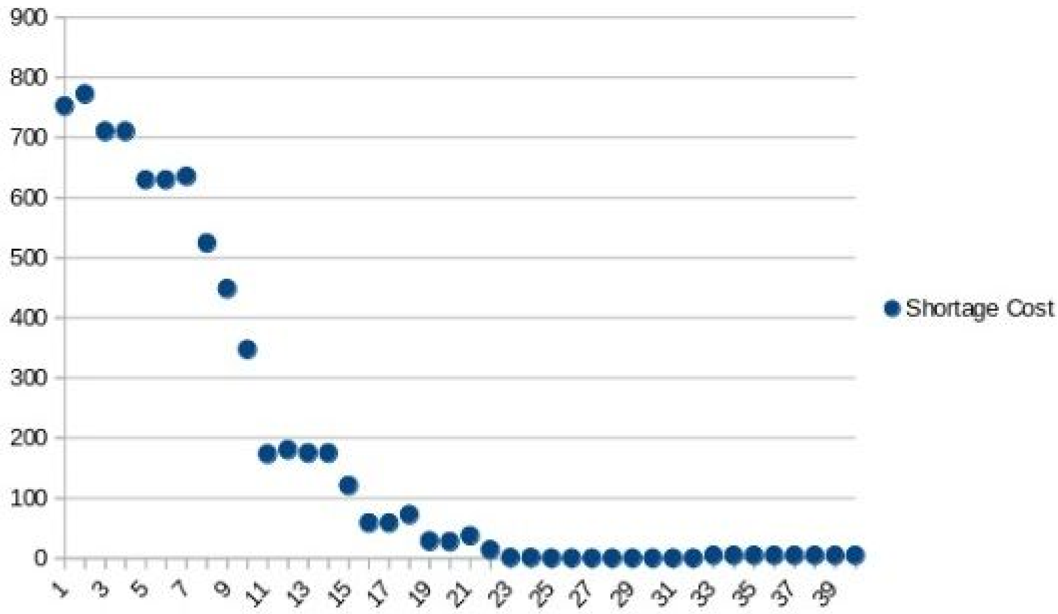
- Os gráficos seguintes são do S = 60, com 0 < s < 41:

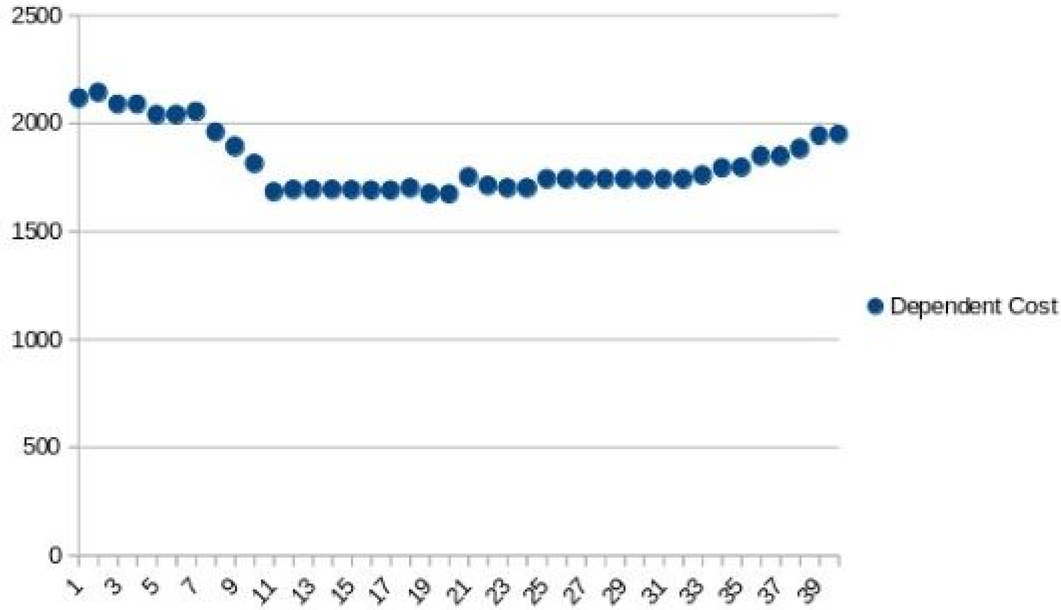


450
400
350
300
250
200
150
100
50
0
1
3
5
7
9
11
13
15
17
19
21
23
25
27
29
31
33
35
37
39
Setup Cost



1800
1600
1400
1200
1000
800
600
400
200
0
1
3
5
7
9
11
13
15
17
19
21
23
25
27
29
31
33
35
37
39
Holding Cost





(b) How does the minimum cost value of s seem to depend on S? (See Exercise 1.3.2.)

R: Analisando esses três pontos:

- S = 100, s = 20, Dependent Cost = 1674.70;
- S = 80, s = 22, Dependent Cost = 1549.29;
- S = 60, s = 24, Dependent Cost = 1358.76;

Temos que esses pontos são pontos ótimos para seus respectivos valores de S, portanto, quanto maior o S, maior o minimum cost, e o inverso também é válido.