CPP1702 Coding Assignment 2

庫存系統-訂購決策問題

電子檔 2018/03/27 09:00 前上傳至 moodle
書面檔 2018/03/27 09:15 前於課堂繳交

運用在工業與資訊管理的範圍

基本邏輯推導、庫存系統。

相關 C++課程內容

基本 C/C++語法、陣列、迴圈、讀檔。

問題背景與概述

「庫存系統管理」為「作業管理」領域中極為重要的一個主題，每日 i 的庫存即存貨量 $I[i]$（件/日）指的是商家在每天打烊後，尚存於其倉庫內的商品總量。適量的存貨有益於應付突發的需求量：若前一日之存貨量 $I[i\text{-}1]$加上當日之新進貨量 $O[i]$仍不足以應付當日之需求量 $D[i]$，將導致當日產生缺貨量 $L[i]$。而每單位的缺貨每持續一日，將會引發一筆缺貨成本 B（$/件日）；反之，若滿足需求量之後還有多餘的存貨量 $I[i]$，每單位的存貨每滯在倉庫一日，將會引發一筆持有成本 H（$/件日）。一般商家只能被動地因應顧客的需求變化，在打烊後依庫存是否足夠，來決定是否訂購（$Y[i]$=1 或 0）新的進貨量 $O[i+1]$於隔日營業前入庫，以維持適量的庫存。每次訂購皆會產生一固定額的單次訂購成本 A（$/次），且每單位的新訂購貨物需要付一筆進貨成本 C（$/件日）。如何決定商品的進貨時機與進貨量為庫存管理最重要的決策。本次作業將要求同學實作常見的兩種庫存**訂購策略**，讓同學對需求、進貨、存貨、缺貨四者之間的消長關係有初步的理解。

假設某公司已有其每日 i=1,…,T 之商品需求量預測值 $D[i]$，想測試不同訂購策略對其 T 日之總成本(Total Cost，TC)的影響，以選出最有利的訂購策略。總成本由以下四組成本加總而成：

- 存貨成本（Inventory Cost，IC）$H\sum_{i=1}^{T} I[i]$，請注意 $I[0]$因木已成舟，所以不算成本
- 缺貨成本（Backlog Cost，BC）$B\sum_{i=1}^{T} L[i]$，請注意 $L[0]$因木已成舟，所以不算成本
- 訂購成本（Ordering Cost，OC）$A\sum_{i=0}^{T-1} Y[i]$，請注意其可能的訂購期間為 i=0,…,T-1
- 進貨成本（Purchasing Cost，PC）$C\sum_{i=1}^{T} O[i]$

以上每日 i 之存貨量 $I[i]$、缺貨量 $L[i]$、訂購與否 $Y[i]$、進貨量 $O[i]$皆為商家的「決策變數」，除 $Y[i]$僅可為 0 或 1 外，其餘三者皆為正整數或零。又可依其主被動的變動關係得知 $Y[i]$與 $O[i]$為主動之決策，而 $I[i]$與 $L[i]$則被動地受其影響。舉例來說，若第 i 日之需求量 $D[i]$可被前日存貨 $I[i\text{-}1]$與當日進貨 $O[i]$所滿足，則當日之存貨量 $I[i]=I[i\text{-}1]+O[i]\text{-}D[i]$；反之，當日之缺貨量 $L[i]= D[i]\text{-}I[i\text{-}1]\text{-}O[i]$。當然，$I[i]$若為正，$L[i]$必為 0（同理，$L[i]$若為正，$I[i]$必為 0）。

以下先定義兩種訂購策略計算公式中會用到的統計變數：

T 期內的總需求量 $D_T=\sum_{i=1}^{T} D[i]$，平均需求量 $\bar{D}_T=\left\lfloor D_T/T \right\rfloor$，標準差 $\sigma_T=\left\lfloor \sqrt{\sum_{i=1}^{T}(D[i]-\bar{D}_T)^2/(T-1)} \right\rfloor$

本次作業將實作以下兩種訂購策略，它們將以不同的準則來決定 $Y[i]$與 $O[i]$，再隨之決定 $I[i]$與 $L[i]$，最後計算 TC = IC + BC + OC + PC：

- **(s_T, Q_T)訂購策略：**

  當$I[i] \le s_T$（i=0,...,T-1）時，隨即訂購 Q_T 單位的貨，於隔日一大早進貨 $O[i+1]=Q_T$ 單位商品

  s_T:再訂購點，當存貨量少或等於此值時，隨即引發一次訂購，我們先設定 $s_T^0=\bar{D}_T+\sigma_T;\ s_T=s_T^0$

  Q_T:經濟訂購批量，為每次訂購量（隔天早上抵達），我們先設定 $Q_T^0=\left\lfloor\sqrt{2AD_T/H}\right\rfloor;\ Q_T=Q_T^0$

- **(R, S)訂購策略：**

  初始進貨日 V 之後每隔 R 日（第 $V, V+R, ..., V+\lfloor (T-V)/R \rfloor R$ 日）將進貨至其庫存量為 S 單位

  R：訂購週期，亦即每隔 R 日即進貨一次，本作業中 R 可能的值將介於 1 與$\lfloor T/2 \rfloor$ 之間。

  S：進貨將達成之預期期初庫存量，假設某預期進貨日 i 之前一天 $I[i\text{-}1]<S$，則第 i 日將進貨 $O[i]=S\text{-}I[i\text{-}1]$，以使期初庫存量在進貨之後變成 S 單位，在此我們先設定 V=2, $S=7\bar{D}_T$。

作業目的及假設

本次作業擬要求同學寫兩個程式：其中程式一在 PART1 先讀入一個特定格式的輸入檔，依該輸入檔之資料再實作出 PART2: **(s_T, Q_T)** 與 PART3: **(R, S)** 等兩種訂購策略，最後再於程式二實作 PART4，讓同學們嘗試尋找針對 data2.txt 之 PART2 與 PART3 中最佳的 **s_T, Q_T** 與 **V, R, S** 之可能設定值，以使其總成本 TC 最小。

由於此為第二個作業，為簡化問題，我們假設所有的一維陣列長度(32)已給定。同學們可依需要自行再宣告其它新的變數或陣列，我們皆假設使用者會輸入正確的資料與格式（亦即先不用防呆）。

程式要求及作法

本作業大概可切割成以下四個 PART：(PART1、2、3 在程式一；PART4 在程式二)

PART1 宣告與讀檔：(程式一)

請使用相同的命名方法宣告下列變數與陣列，額外需求的自訂變數可自行命名。

```
int  T,           //規劃期間為 T 天
     H,           //單位持有成本
     B,           //單位缺貨成本
     A,           //單次訂購成本
     C,           //單位進貨成本
     s,     //再訂購點,程式一假設 s_T = s_T^0,程式二需測試 σ_T +1 組 s_T 值
     Q,     //經濟訂購批量,程式一假設 Q_T = Q_T^0,程式二需測試 11 組 Q_T 值
     V,     //初始進貨日,程式一假設 V = 2,程式二需測試 V = 2,4,6 等 3 組數值
     R,     //訂購週期,程式一假設 R = 7,程式二需測試 R = 1,3,5,...,13,15 等 8 組值
     S;     //預期期初庫存量,程式一假設 S = 7D̄_T,程式二需配合 8 組 R 值測試 S = RD̄_T 值
int  D[32]={0}, //儲存第 i(=1,…,31)天的預測需求量，先初始化為 0
     I[32]={0}, //儲存第 i(=0,…,31)天的期末存貨量，先初始化為 0
     L[32]={0}, //儲存第 i(=0,…,31)天的期末缺貨量，先初始化為 0
     Y[32]={0}, //儲存第 i(=0,…,30)天的訂購決定，先初始化為 0
     O[32]={0}; //儲存第 i(=1,…,31)天的期初進貨量，先初始化為 0
```

請使用者輸入檔名，譬如讀取 data1.txt 檔案後，輸出以下結果。(紅色部分為使用者輸入)

```
Enter filename: data1.txt
T=10, D= 34 31 26 40 33 23 22 25 41 17
H=2, B=20, A=300, C=10, V=2, R=7, I[0]=45, L[0]=0
D=292, D_bar=29, stdev=7; (s,Q)=(36,295); (R,S)=(7,203)
```

輸入檔 data1.txt 格式:

10	(期數 T)
34 31 26 40 33 23 22 25 41 17	(預測需求量 D[i], i=1, ... ,10)
2 20 300 10 2 7 45 0	(H、B、A、C、V、R、I[0]、L[0])

PART2 實作(s_T, Q_T)訂購策略：(程式一)

實作完成後，請印出以下格式之結果：(請分別用 data1.txt、data2.txt 測試並印出)

```
(s=36,Q=295) model:
    t     D     O     I     L
-----------------------------
    0     0     0    45     0
    1    34     0    11     0
    2    31   295   275     0
    3    26     0   249     0
    4    40     0   209     0
    5    33     0   176     0
    6    23     0   153     0
    7    22     0   131     0
    8    25     0   106     0
    9    41     0    65     0
   10    17     0    48     0
===============================
Total cost: 6096 = 2846(IC) + 0(BC) + 300(OC) + 2950(PC)
```

左表為 data1.txt 之結果(供核對用)
以下為 data2.txt 之總成本(供核對用)

```
D=960, D_bar=30, stdev=12;
(s,Q)=(42,536) model:
Total cost: 27622 = 16302(IC) + 0(BC) +
600(OC) + 10720(PC)
```

PART3 實作(R, S)訂購策略：(程式一)

實作完成後，請印出以下格式之結果：(請分別用 data1.txt、data2.txt 測試並印出)

```
(R=7,S=203) model:
    t     D     O     I     L
-----------------------------
    0     0     0    45     0
    1    34     0    11     0
    2    31   192   172     0
    3    26     0   146     0
    4    40     0   106     0
    5    33     0    73     0
    6    23     0    50     0
    7    22     0    28     0
    8    25     0     3     0
    9    41   200   162     0
   10    17     0   145     0
===============================
Total cost: 6312 = 1792(IC) + 0(BC) + 600(OC) + 3920(PC)
```

左表為 data1.txt 之結果(供核對用)
以下為 data2.txt 之總成本(供核對用)

```
(R,S)=(7,210) model:
Total cost: 18798 = 5828(IC) + 1920(BC)
+ 1500(OC) + 9550(PC)
```

PART4 讀取 [data2.txt]()，搜尋最佳之 s_T, Q_T 與 V, R, S 設定值，以使其總成本 TC 最小：（程式二）

- 針對(s_T, Q_T)訂購策略，原先使用之公式 $s_T = s_T^0$ 、 $Q_T = Q_T^0$ 並無法保證可算出小總成本，本題要求同學們測試 $\sigma_T + 1$ 組 s_T 可能的值： $s_T \in \{s_T^0 - 2\sigma_T, s_T^0 - 2\sigma_T + 2, s_T^0 - 2\sigma_T + 4, ..., s_T^0 - 2, s_T^0\}$ ，以及 11 組 Q_T 可能的值： $Q_T \in \{Q_T^0 - 50, Q_T^0 - 40, ..., Q_T^0 + 40, Q_T^0 + 50\}$ ，亦即找出這 $11(\sigma_T + 1)$ 個(s_T, Q_T)組合當中，哪一組（**s_T, Q_T**）設定能測得最小之總成本。
- 針對(R, S)訂購策略，原先使用 V=2, R=7, $S = 7\bar{D}_T$ 並無法保證可算出最小總成本，本題要求同學們計算多組 **V, R, S** 可能的值，以搭配得到測試組合中總成本最小者。在此請同學測試 V=2, 4, 6 共 3 組； $(R,S) \in \{(1,\bar{D}_T),(3,3\bar{D}_T),...,(13,13\bar{D}_T),(15,15\bar{D}_T)\}$ 共 8 組，找出這 $3 \times 8 = 24$ 個組合當中，哪一組 **V, R, S** 設定能測得最小之總成本。
- 再比較上述兩種訂購策略之最佳設定，印出其所有組合中最佳之訂購策略(即最佳之 **s_T, Q_T 與 V, R, S 設定值**)、總成本與各項成本：$TC = IC + BC + OC + PC$

解題建議

- 老師估計初學者必須花至少 3hr 才能將本題作好，因為是第二個作業，老師給大家兩週多來做。如果連這題你都必須花超過 5hr 才能作好，代表之後你必須要更早開始寫作業。
- 本題分成 4 個 PART 處理，前 3 個 PART 應該可以很快解決（1hr 內），不過仍建議你先確認 PART1 對了之後再做 PART2、3。前三個 PART 建議命名為 **hw2_123.cpp**
- PART4 的程式可先 copy/paste 前三 PART 的程式當基礎來改，最簡單的作法就是先把先前單一變數的 s,Q,V,R,S 重新宣告成陣列(譬如 s[13],Q[11],V[3],R[8],S[8]）再使用多重 for loop 分別計算不同組合之總成本，當然你要記得記錄當下為止曾經計算過的最小總成本以及其相關參數設定。PART4 只需列印最佳組合的相關參數設定值以及成本，不用像 PART2、3 那樣列印個別之 D[],O[],I,L[]。PART4 建議命名為 **hw2_4.cpp**
- 記得兩個 cpp 檔案的 header comment 應該會稍有不同
- 其實 PART1、2、3 可以用 Excel 來檢查

***作業繳交應注意事項**

1. 作業需繳交電子檔以及書面(列印程式檔及程式結果)
2. 電子檔請於作業繳交截止時間以前上傳至 http://moodle.ncku.edu.tw
   2.1 請同學先建立一個**資料夾**，資料夾名稱為 "**學號_hw2**"，例如學號為 hxxxx，則資料夾名稱則為 **hxxxx_hw2**
   2.2 將程式檔案名稱存為 "**hw2_123.cpp**、**hw2_4.cpp**"，並將此程式檔案存於上述設立之 **學號_hw2** 資料夾中
   2.3 最後將整個**學號_hw2** 資料夾壓縮成 zip 檔(**學號_hw2.zip**)，再上傳至 moodle 系統
   (！注意！：**請勿**將 cpp 檔 copy/paste 至 word 檔而上傳之)
3. 書面作業請於 **2018/03/27 上課 5 分鐘內(09:15 前)**繳交至講台，其中需要註明程式是否能被編譯與執行、撰寫人、程式之目的、如何編譯及執行等資料(詳見 https://www.dropbox.com/s/pvac59tfefggokd/programming.html?dl=0)。