Λύσεις 2ης Σειράς Ασκήσεων

1. (25 **μονάδες**) Ας υποθέσουμε ότι βρισκόμαστε κοντά στο τέλος του εαρινού εξαμήνου, στο οποίο παρακολουθείτε n μαθήματα. Καθένα από αυτά τα μαθήματα απαιτεί την παράδοση μίας τελικής εργασίας την οποία έχετε να ολοκληρώσετε. Κάθε τελική εργασία θα βαθμολογηθεί ως εξής: Θα λάβει ένα ακέραιο αριθμό στην κλίμακα από το 1 μέχρι το g (όπου $g > 1$), και μεγαλύτεροι αριθμοί δείχνουν βέβαια μεγαλύτερους βαθμούς. Ο στόχος είναι να μεγιστοποιήσετε το άθροισμα των βαθμών σας στις n τελικές εργασίες.
'Εχετε συνολικά στη διάθεσή σας $H > n$ ώρες για να εργαστείτε στις n εργασίες που σας έχουν ανατεθεί, και θέλετε να αποφασίσετε την κατανομή των ωρών σας στις n εργασίες. Υποθέτουμε απλοποιητικά ότι ο H είναι ακέραιος, και ότι θα αφιερώσετε ένα ακέραιο αριθμό ωρών στην κάθε τελική εργασία. Για να λάβετε τις σωστές αποφάσεις, θα χρησιμοποιήσετε ένα σύνολο από *συναρτήσεις ωφελιμότητας* $\{f_i \mid 1 \leq i \leq n\}$ με την εξής σημασία: Αν αφιερώσετε $h \leq H$ ώρες στην τελική εργασία για το μάθημα i, αυτή θα βαθμολογηθεί με $f_i(h)$. Μπορείτε να υποθέσετε ότι κάθε συνάρτηση f_i είναι μη φθίνουσα: αν $h < h'$, τότε $f_i(h) \leq f_i(h')$. 'Ετσι, το υπολογιστικό πρόβλημα που έχετε να επιλύσετε είναι το εξής: Με δεδομένες τις συναρτήσεις $\{f_i \mid 1 \leq i \leq n\}$, βρείτε (ακέραιες) ώρες $h_1, \ldots, h_n$ με $\sum_{i=1}^{n} h_i \leq H$ έτσι ώστε το άθροισμα $\sum_{i=1}^{n} f_i(h_i)$ να μεγιστοποιηθεί.
Παρουσιάστε και αναλύστε αλγόριθμο δυναμικού προγραμματισμού για το υπολογιστικό σας πρόβλημα. Η πολυπλοκότητα του αλγορίθμου σας πρέπει να είναι πολυωνυμική ως προς τις παραμέτρους n, g και H.

Λύση:

Τα υποπροβλήματα: Ορίζουμε ως υποπρόβλημα $\Pi(j, h)$, όπου $1 \leq j \leq n$ και $0 \leq h \leq H$, το εξής υπολογιστικό πρόβλημα:

> Με δεδομένες τις συναρτήσεις $\{f_i \mid 1 \leq i \leq j\}$, βρείτε (ακέραιες) ώρες $h_1, \ldots, h_j$ με $\sum_{i=1}^{j} h_i \leq h$ έτσι ώστε το άθροισμα $\sum_{i=1}^{j} f_i(h_i)$ να μεγιστοποιηθεί.

Προφανώς, υπάρχουν $n \cdot (H + 1)$ τέτοια υποπροβλήματα, και το υπολογιστικό πρόβλημα που έχουμε να επιλύσουμε είναι το υποπρόβλημα $\Pi(n, H)$.

Χαρακτηρισμός της δομής της βέλτιστης λύσης: 'Εστω μία βέλτιστη λύση $h_1, \ldots, h_j$ στο υποπρόβλημα $\Pi(j, h)$, όπου $1 \leq j \leq n$ και $0 \leq h \leq H$, στην οποία αφιερώνονται $h_j = \kappa$ ώρες στην εργασία j. Τότε, οι ώρες $h_1, \ldots, h_{j-1}$ αποτελούν μία βέλτιστη λύση για το υποπρόβλημα $\Pi(j - 1, h - \kappa)$.

> Πράγματι, ας υποθέσουμε ότι υπήρχε καλύτερη λύση $h'_1, \ldots, h'_{j-1}$ (με $\sum_{i=1}^{j-1} f_i(h'_i) > \sum_{i=1}^{j-1} f_i(h_i)$) για το υποπρόβλημα $\Pi(j-1, h-\kappa)$. Θεωρούμε τη λύση $h'_1, \ldots, h'_{j-1}, h_j$

για το υποπρόβλημα $\Pi(j,h)$. Έχουμε ότι

$$\begin{aligned}\sum_{i=1}^{j-1} f_i(h'_i) + f_j(h_j) &> \sum_{i=1}^{j-1} f_i(h_i) + f_j(h_j) \\ &= \sum_{i=1}^{j} f_i(h_i)\,.\end{aligned}$$

Έπεται ότι η λύση $h_1,\ldots,h_{j-1},h_j$ δεν είναι βέλτιστη λύση για το υποπρόβλημα $\Pi(j,h)$ (αφού η λύση $h'_1,\ldots,h'_{j-1},h_j$ είναι καλύτερη). Αντίφαση.

Η τιμή της βέλτιστης λύσης: Έστω $\mathsf{V}[j,h]$ η τιμή μίας βέλτιστης λύσης για το υποπρόβλημα $\Pi(j,h)$, όπου $1 \le j \le n$ και $0 \le h \le H$, (Δηλαδή, $\mathsf{V}[j,h]$ είναι η μέγιστη δυνατή τιμή που μπορεί να λάβει το άθροισμα $\sum_{i=1}^{j} f_i(h_i)$.) Ορίζουμε κατά σύμβαση ότι $\mathsf{V}[0,h] = 0$ για κάθε h, όπου $0 \le h \le H$. (Προφανώς, η τιμή $\mathsf{V}[0,h] = 0$ δεν αντιστοιχεί σε κάποιο από τα υποπροβλήματα που έχουμε ορίσει.) Από τον χαρακτηρισμό της δομής της βέλτιστης λύσης, έχουμε ότι

$$\mathsf{V}[j,h] \;=\; \begin{cases} \max_{0\le k\le h}\left(f_j(k) + \mathsf{V}[j-1,h-k]\right)\,, & \text{αν } h > 0 \\ 0\,, & \text{αν } h = 0 \end{cases}\,.$$

Ο αλγόριθμος δυναμικού προγραμματισμού: Παρουσιάζουμε τώρα τον αλγόριθμό μας:

- Συμπλήρωση του πίνακα με τις λύσεις στα υποπροβλήματα: Για κάθε δείκτη j, όπου $1 \le j \le n$, χρησιμοποιούμε την τιμή της βέλτιστης λύσης για να υπολογίσουμε όλες τις τιμές $\mathsf{V}[j,h]$, όπου $0 \le h \le H$. Καταχωρούμε όλες τις τιμές που έχουμε υπολογίσει στη γραμμή j ενός πίνακα $\{1,\ldots,n\} \times \{0,\ldots,H\}$ (από αριστερά προς τα δεξιά). Για κάθε θέση (j,h) του πίνακα με $h \neq 0$, καταχωρούμε, επίσης, την τιμή $\kappa = \kappa(j,h)$ για την οποία λάβαμε ότι

$$\mathsf{V}[j,h] \;=\; f_j(\kappa) + \mathsf{V}[j-1,h-\kappa]\,.$$

- Ανακατασκευή της βέλτιστης λύσης από τον πίνακα με τις λύσεις στα υποπροβλήματα: Η τιμή της βέλτιστης λύσης που ζητούμε είναι η τιμή $\mathsf{V}[n,H]$ από τον πίνακά μας. Επίσης, η βέλτιστη λύση $h_1,\ldots,h_n$ ανακατασκευάζεται αναδρομικά από τον πίνακα $\{1,\ldots,n\} \times \{0,\ldots,H\}$ ως εξής:
  - $h_n := \kappa(n,H)$
  - Χρησιμοποιούμε τον υποπίνακα $\{1,\ldots,n-1\} \times \{0,\ldots,H-\kappa(n,H)\}$ για να ανακατασκευάσουμε αναδρομικά τη βέλτιστη λύση $h_1,\ldots,h_{n-1}$ στο υποπρόβλημα $\Pi(n-1, H-\kappa(n,H))$.

Πολυπλοκότητα: Χρειάζόνται $O(H)$ βήματα για τον υπολογισμό οποιασδήποτε τιμής $\mathsf{V}[j,h]$ στη θέση (j,h) του πίνακα $\{1,\ldots,n\} \times \{0,\ldots,H\}$, όπου $1 \le j \le n$ και $0 \le h \le H$. Ετσι, χρειάζονται συνολικά $O(n^2H)$ βήματα για την συμπλήρωση του πίνακα με τις λύσεις στα υποπροβλήματα. Η (αναδρομική) ανακατασκευή της βέλτιστης λύσης από τον ίδιο πίνακα απαιτεί επιπρόσθετα $O(n)$ βήματα. Έτσι, ο συνολικός αριθμός βημάτων είναι $O(n^2H)$.

2. (25 μονάδες) Ως επενδυτές σε μία χρηματιστηριακή αγορά μετοχών, θεωρούμε τις επόμενες (συνεχόμενες) μέρες $1, \ldots, n$. Η τιμή της μοναδιαίας μετοχής κατά την μέρα i είναι $p(i)$.
Έστω σταθερός (δυνατόν μεγάλος) ακέραιος K με $K \leq n$. Μία *στρατηγική K αγοραπωλήσεων* είναι μία ακολουθία από m ζεύγη ημερών $(b_1, s_1), \ldots, (b_m, s_m)$, όπου $0 \leq m \leq K$, τέτοια ώστε $1 \leq b_1 < s_1 < \ldots b_m < s_m \leq n$. Έτσι, μία τέτοια στρατηγική είναι ένα σύνολο από το πολύ K, μη επικαλυπτόμενα χρονικά διαστήματα, όπου ο επενδυτής αγοράζει $1,000$ μετοχές την μέρα b_i και τις πωλεί την μέρα s_i, όπου $1 \leq i \leq m$. Το *κέρδος* μίας στρατηγικής K αγοραπωλήσεων είναι $1,000 \sum_{i=1}^{m} (p(s_i) - p(b_i))$.
Παρουσιάστε και αναλύστε ένα αλγόριθμο δυναμικού προγραμματισμού ο οποίος θα βρίσκει την στρατηγική K αγοραπωλήσεων με το μέγιστο δυνατό κέρδος. Η πολυπλοκότητα του αλγορίθμου σας πρέπει να είναι πολυωνυμική ως προς τις παραμέτρους n και K.

Λύση:

Προκαταρκτικά και ορισμοί: Συμβολίζουμε ως (i, j) την αγοραπωλησία η οποία συνίσταται στην αγορά $1,000$ μετοχών κατά τη μέρα i και την πώλησή τους κατά τη μέρα j, όπου $1 \leq i < j \leq n$. Το κέρδος από την αγοραπωλησία (i, j) είναι $\mathsf{P}[i, j] = 1,000\,(p(j) - p(i))$. Ορίζουμε ως $\mathsf{Q}[i, j]$ το μέγιστο δυνατό κέρδος από μία αγοραπωλησία η οποία λαμβάνει χώρα μεταξύ των ημερών i και j (συμπεριλαμβανομένων).

Προχωρούμε να ορίσουμε μία παραλλαγή της στρατηγικής K αγοραπωλήσεων, η οποία αποτελεί ειδική περίπτωσή της.

> Μία στρατηγική ακριβώς K αγοραπωλήσεων είναι μία ακολουθία από K ζεύγη ημερών $(b_1, s_1), \ldots, (b_K, s_K)$ τέτοια ώστε $1 \leq b_1 < s_1 < \ldots b_m < s_K \leq n$. Έτσι, μία τέτοια στρατηγική είναι ένα σύνολο από (ακριβώς) K μη επικαλυπτόμενα χρονικά διαστήματα, όπου ο επενδυτής αγοράζει $1,000$ μετοχές την μέρα b_i και τις πωλεί την μέρα s_i, όπου $1 \leq i \leq K$.

Ορίζουμε ως $\mathsf{AM}[m, d]$ το μέγιστο δυνατό κέρδος που μπορούμε να λάβουμε από μία στρατηγική ακριβώς m αγοραπωλήσεων στο διάστημα ημερών $[1, d]$, όπου $0 \leq m \leq K$ και $0 \leq d \leq n$. Ορίζουμε κατά σύμβαση ως $-\infty$ το μέγιστο δυνατό κέρδος που μπορούμε να λάβουμε από μία στρατηγική ακριβώς m αγοραπωλήσεων στο διάστημα ημερών $[1, d]$ όταν δεν είναι δυνατό να εκτελέσουμε m αγοραπωλησίες στο διάστημα ημερών $[1, d]$. (Για παράδειγμα, αυτό συμβαίνει όταν $d < 2m$.) Ορίζουμε κατά σύμβαση ότι $\mathsf{AM}[m, 0] = 0$ για κάθε δείκτη m με $0 \leq m \leq K$ και $\mathsf{AM}[0, d] = 0$ για κάθε δείκτη d με $0 \leq d \leq n$.

Τέλος, ορίζουμε ως $\mathsf{M}[K, d]$ το μέγιστο δυνατό κέρδος που μπορούμε να λάβουμε από μία στρατηγική K αγοραπωλήσεων στο διάστημα ημερών $[1, d]$, όπου $0 \leq d \leq n$. Ετσι, ζητούμε ένα αλγόριθμο δυναμικού προγραμματισμού για τον υπολογισμό της τιμής $\mathsf{M}[K, n]$. Προφανώς, ισχύει ότι

$$\mathsf{M}[K, n] \;=\; \max_{0 \leq m \leq K} \mathsf{AM}[m, n]\,.$$

Έτσι, αρκεί να βρούμε ένα αλγόριθμο δυναμικού προγραμματισμού για τον υπολογισμό όλων των τιμών $\mathsf{AM}[m, n]$, όπου $0 \leq m \leq K$.

Φάση προεπεξεργασίας: Παρουσιάζουμε ένα απλό αλγόριθμο δυναμικού προγραμματισμού για τον υπολογισμό όλων των τιμών $\mathsf{Q}[i,j]$, όπου $1 \leq i < j \leq n$, σε μία φάση προεπεξεργασίας. Οι τιμές αυτές θα χρησιμοποιηθούν στη συνέχεια από τον αλγόριθμο δυναμικού προγραμματισμού που θα σχεδιάσουμε για τον υπολογισμό όλων των τιμών $\mathsf{AM}[m,n]$, όπου $0 \leq m \leq K$.

Θεωρούμε ζεύγος δεικτών i και j, όπου $1 \leq i < j \leq n$. Αν $j-i=1$, τότε $\mathsf{Q}[i,j] = \mathsf{P}[i,j]$. Έστω λοιπόν ότι $j-i>1$. Τότε, προφανώς, η αγοραπωλησία η οποία δημιουργεί το μέγιστο κέρδος $\mathsf{Q}[i,j]$ *(i)* είτε είναι η αγοραπωλησία (i,j), *(ii)* είτε λαμβάνει χώρα σε ένα απο τα διαστήματα ημερών $[i,j-1]$ ή $[i+1,j]$. Έτσι, λαμβάνουμε ότι

$$\mathsf{Q}[i,j] \;=\; \begin{cases} \max\{\mathsf{P}[i,j], \mathsf{Q}[i,j-1], \mathsf{Q}[i+1,j]\}\,, & \text{αν } j-i>1 \\ \mathsf{P}[i,j] & \text{αν } j-i=1 \end{cases} .$$

Μπορούμε να χρησιμοποιήσουμε την αναδρομική αυτή σχέση ως τη βάση για ένα αλγόριθμο δυναμικού προγραμματισμού ο οποίος υπολογίζει όλες τις τιμές $\mathsf{Q}[i,j]$, όπου $1 \leq i < j \leq n$, και τις καταχωρεί σε ένα αντίστοιχο πίνακα. Ο υπολογισμός (και η καταχώρηση) προχωρεί κατά αύξουσα διαφορά $j-i$, όπου $1 \leq i < j \leq n$. (Έτσι, $1 \leq j-i \leq n-1$.) Ο συνολικός αριθμός βημάτων που απαιτούνται είναι ανάλογος του αριθμού των ζευγών (i,j) με $1 \leq i < j \leq n$, δηλαδή είναι $O(n^2)$.

Τα υποπροβλήματα: Ορίζουμε ως υποπρόβλημα $\Pi(m,d)$, όπου $0 \leq m \leq K$ και $0 \leq d \leq n$, το εξής υπολογιστικό πρόβλημα:

> Με δεδομένες τις τιμές $p_1, \ldots, p_d$, υπολόγισε μία στρατηγική ακριβώς K αγοραπωλήσεων η οποία επιτυγχάνει το μέγιστο δυνατό κέρδος $\mathsf{AM}(m,d)$ (και την αντίστοιχη τιμή $\mathsf{M}(m,d)$).

Προφανώς, υπάρχουν $(K+1)\cdot(n+1)$ τέτοια υποπροβλήματα, και το υπολογιστικό πρόβλημα που έχουμε να επιλύσουμε (δηλαδή, ο υπολογισμός όλων των τιμών $\mathsf{AM}[m,n]$, όπου $0 \leq m \leq K$ και των αντιστοίχων βελτίστων στρατηγικών ακριβώς m αγοραπωλήσεων) θα χρησιμοποιήσει τελικά τις λύσεις στα υποπροβλήματα $\Pi(m,n)$, όπου $0 \leq m \leq K$.

Χαρακτηρισμός της δομής της βέλτιστης λύσης: Έστω μία βέλτιστη λύση στο υποπρόβλημα $\Pi(m,d)$ (δηλαδή, μία ακολουθία ακριβώς m αγοραπωλήσεων στο διάστημα ημερών $[1,d]$ η οποία επιτυγχάνει το μέγιστο δυνατό κέρδος $\mathsf{AM}[m,d]$), όπου $0 \leq m \leq K$ και $0 \leq d \leq n$. Ας υποθέσουμε ότι η τελευταία αγοραπωλησία στη βέλτιστη αυτή λύση είναι η αγοραπωλησία (i,j), όπου $1 \leq i < j \leq d$. Τότε, το σύνολο όλων των αγοραπωλησιών πλην της τελευταίας (i,j) αποτελούν μία βέλτιστη λύση για το υποπρόβλημα $\Pi(i-1,m-1)$. (Δηλαδή, αποτελούν μία βέλτιστη στρατηγική ακριβώς $m-1$ αγοραπωλήσεων για το διάστημα ημερών $[1,i-1]$.)

> Πράγματι, αν υπήρχε μία καλύτερη λύση για το υποπρόβλημα $\Pi(m-1,i-1)$, αυτή θα έδινε μαζί με την αγοραπωλησία (i,j) μία καλύτερη λύση για το υποπρόβλημα $\Pi(m,d)$. Αντίφαση, καθώς ξεκινήσαμε αρχικά με μία βέλτιστη λύση για το υποπρόβλημα $\Pi(m,d)$.

Η τιμή της βέλτιστης λύσης: Έστω το υποπρόβλημα $\Pi(m,d)$, όπου $0 \leq m \leq K$ και $0 \leq d \leq n$. Από τον χαρακτηρισμό της δομής της βέλτιστης λύσης, έχουμε ότι

$$\mathsf{AM}[m,d] \;=\; \begin{cases} \max_{1 \leq i < j \leq d} \left(\mathsf{Q}[i,j] + \mathsf{AM}[m-1,i-1]\right) & \text{αν } m>0 \text{ και } d>0 \\ -\infty & \text{αν } m=0 \text{ ή } d=0 \end{cases} .$$

Ο αλγόριθμος δυναμικού προγραμματισμού: Παρουσιάζουμε τώρα τον αλγόριθμό μας:

- Συμπλήρωση του πίνακα με τις λύσεις στα υποπροβλήματα: Για κάθε ζεύγος από δείκτες m και d, όπου $0 \leq m \leq K$ και $0 \leq d \leq n$, χρησιμοποιούμε την τιμή της βέλτιστης λύσης για να υπολογίσουμε την τιμή $\mathsf{AM}[m,d]$. Καταχωρούμε κάθε τιμή που υπολογίζουμε στη αντίστοιχη θέση (m,d) ενός πίνακα $\{0,\ldots,K\} \times \{0,\ldots,n\}$ (από κάτω προς τα πάνω και από αριστερά προς τα δεξιά). Για κάθε θέση (m,d) του πίνακα με $m > 0$ και $d > 0$, καταχωρούμε, επίσης, το ζεύγος $(i,j) = (i(m,d), j(m,d))$ για το οποίο λάβαμε ότι

$$\mathsf{AM}[m,d] \;=\; \mathsf{Q}[i,j] + \mathsf{AM}[m-1, i-1]\,.$$

- Ανακατασκευή των βελτίστων λύσεων από τον πίνακα με τις λύσεις στα υποπροβλήματα: Οι τιμές των βελτίστων λύσεων που ζητούμε είναι όλες οι τιμές $\mathsf{AM}[m,n]$ από τον πίνακά μας, όπου $0 \leq m \leq K$. (Υπενθυμίζουμε ότι από αυτές τις τιμές, η τιμή της βέλτιστης λύσης $\mathsf{M}[K,n]$ θα υπολογιστεί τελικά ως $\mathsf{M}[K,n] = \max_{0 \leq m \leq K} \mathsf{AM}[m,n]$.) Επίσης, κάθε αντίστοιχη βέλτιστη λύση για το υποπρόβλημα $\Pi(m,n)$ ανακατασκευάζεται αναδρομικά από τον πίνακα $\{0,\ldots,m\} \times \{0,\ldots,n\}$ ως εξής:
  - Λαμβάνουμε ως τελευταία αγοραπωλησία την αγοραπωλησία $(i(m,n), j(m,n))$.
  - Χρησιμοποιούμε τον υποπίνακα $\{0,\ldots,m-1\} \times \{0,\ldots,i(m,n)-1\}$ για να ανακατασκευάσουμε αναδρομικά τη βέλτιστη λύση στο υποπρόβλημα $\Pi(m-1, i(m,n)-1)$.
- Υπολογισμός της βέλτιστης λύσης από τις ανακατασκευασμένες βελτίστες λύσεις: Τώρα, θα επιστρέψουμε ως τιμή της βέλτιστης λύσης την ποσότητα

$$\mathsf{M}[K,n] \;=\; \max_{0 \leq m \leq K} \mathsf{AM}[m,n]\,.$$

  Έστω μ ο δείκτης ο οποίος μεγιστοποιεί την τιμή $\mathsf{AM}[m,n]$, όπου $0 \leq m \leq K$, Τότε, επιστρέφουμε ως βέλτιστη λύση τη βέλτιστη λύση για το υποπρόβλημα $\Pi(\mu,n)$ (την οποία έχουμε ανακατασκευάσει).

Πολυπλοκότητα: Υπενθυμίζουμε ότι χρειάζονται $O(n^2)$ βήματα για τον υπολογισμό όλων των τιμών $\mathsf{Q}[i,j]$, όπου $0 \leq i < j \leq n$. Χρειάζονται $O(n)$ βήματα για τον υπολογισμό οποιασδήποτε τιμής $\mathsf{AM}[m,d]$ στη θέση (m,d) του πίνακα $\{0,\ldots,K\} \times \{0,\ldots,n\}$, όπου $0 \leq m \leq K$ και $0 \leq d \leq n$. Έτσι, χρειάζονται συνολικά $O(n^2K)$ βήματα για τη συμπλήρωση του πίνακα με τις λύσιες στα υποπροβλήματα. Η (αναδρομική) ανακατασκευή των βελτίστων λύσεων από τον ίδιο πίνακα απαιτεί επιπρόσθετα $O(n)$ βήματα. Ο τελικός υπολογισμός της βέλτιστης λύσης (από τις ανακατασκευασμένες βέλτιστες λύσεις) απαιτεί $O(K)$ βήματα. Έτσι, ο συνολικός αριθμός βημάτων είναι $O(n^2K)$.

3. (25 μονάδες) Είστε ο ιδιοκτήτης μίας αντιπροσωπείας η οποία εισάγει και πωλεί αυτοκίνητα πολυτελείας. Υπάρχουν προβλέψεις για τον αριθμό των πωλήσεων στους επόμενους μήνες $1, \ldots, n$. Έστω d_i ο αριθμός των προβλεπομένων πωλήσεων για τον μήνα i, όπου $1 \leq i \leq n$. Όλες οι πωλήσεις λαμβάνουν χώραν κατά την πρώτην ημέρα του μήνα και τα αυτοκίνητα που δεν έχουν πωληθεί αποθηκεύονται μέχρι τον επόμενο μήνα. Μπορείτε να αποθηκεύσετε μέχρι S αυτοκίνητα στην αποθήκη σας, και το κόστος για την αποθήκευση ενός αυτοκινήτου είναι C.
Τα αυτοκίνητα αποστέλλονται στην αντιπροσωπεία σας κατόπιν παραγγελίας, και η κάθε παραγγελία στοιχίζει K (ανεξάρτητα από τον αριθμό των αυτοκινήτων που παραγγέλλονται). Αρχικά, δεν υπάρχουν καθόλου αυτοκίνητα στην αντιπροσωπεία σας.
Το υπολογιστικό πρόβλημα που έχετε να επιλύσετε είναι το εξής: Θέλετε να προσδιορίσετε τις παραγγελίες σας (δηλαδή, πόσα αυτοκίνητα θα παραγγείλετε και πότε) έτσι ώστε να ικανοποιήσετε όλες τις προβλεπόμενες πωλήσεις d_i, όπου $1 \leq i \leq n$, με το ελάχιστο δυνατό κόστος. Παρουσιάστε και αναλύστε αλγόριθμο δυναμικού προγραμματισμού για το πρόβλημά σας. Η χρονική πολυπλοκότητα του αλγορίθμου σας πρέπει να είναι πολυωνυμική ως προς τις παραμέτρους n και S.

Λύση:

Τα υποπροβλήματα: Ορίζουμε ως υποπρόβλημα $\Pi(i, s)$, όπου $1 \leq i \leq n$ και $0 \leq s \leq S$, το εξής υπολογιστικό πρόβλημα:

> Με δεδομένες τις προβλεπόμενες πωλήσεις $d_1, \ldots, d_i$, υπολόγισε τις παραγγελίες για τους πρώτους i μήνες έτσι ώστε να ελαχιστοποιηθεί το συνολικό κόστος και να παραμένουν s αυτοκίνητα πολυτελείας στην αποθήκη κατά το τέλος του μήνα i.

Έστω $\mathsf{V}[i, s]$ η τιμή της βέλιστης λύσης για το υποπρόβλημα $\Pi(i, s)$ — δηλαδή, το ελάχιστο δυνατό συνολικό κόστος για τους πρώτους i μήνες εφόσον παραμένουν s αυτοκίνητα πολυτελείας στην αποθήκη κατά το τέλος του μήνα i.

Προφανώς, υπάρχουν $n(S+1) = O(nS)$ τέτοια υποπροβλήματα, και το υπολογιστικό πρόβλημα που έχουμε να επιλύσουμε είναι το υποπρόβλημα $\Pi(n, 0)$. (Προσέξτε ότι δεν υπάρχει λόγος να παραμείνουν αυτοκίνητα πολυτελείας στην αποθήκη κατά το τέλος του μήνα n.)

Χαρακτηρισμός της δομής της βέλτιστης λύσης: Έστω μία βέλτιστη λύση για το υποπρόβλημα $\Pi(i, s)$, όπου $1 \leq i \leq n$ και $0 \leq s \leq S$ — δηλαδή, μία ακολουθία παραγγελιών για τους πρώτους i μήνες η οποία επιτυγχάνει το ελάχιστο δυνατό κόστος $\mathsf{V}[i, s]$. Υποθέτουμε ότι για την βέλτιστη αυτή λύση παραμένουν $z \leq S$ αυτοκίνητα πολυτελείας στην αποθήκη κατά το τέλος του μήνα $i-1$. Τότε, η ακολουθία όλων των παραγγελιών πλην της τελευταίας (για τον μήνα i) αποτελεί μία βέλτιστη λύση για το υποπρόβλημα $\Pi(i-1, z)$.

> Πράγματι, αν υπήρχε μία καλύτερη λύση για το υποπρόβλημα $\Pi(i-1, z)$, αυτή θα έδινε μαζί με την παραγγελία για τον μήνα i, μία καλύτερη λύση για το υποπρόβλημα $\Pi(i, s)$. Αντίφαση, καθώς ξεκινήσαμε αρχικά με μία βέλτιστη λύση για το υποπρόβλημα $\Pi(i, s)$.

Παρατηρούμε ότι οφείλουμε να καταβάλουμε κατά την αρχή του μήνα i το κόστος αποθήκευσης zC (για την αποθήκευση z αυτοκινήτων πολυτελείας κατά τον μήνα $i-1$).

Η τιμή της βέλτιστης λύσης: Θεωρούμε ξανά μία βέλτιστη λύση για το υποπρόβλημα $\Pi(i,s)$, όπου $1 \leq i \leq n$ και $0 \leq s \leq S$, και υποθέτουμε ότι για την βέλτιστη αυτή λύση παραμένουν $z \leq S$ αυτοκίνητα πολυτελείας στην αποθήκη κατά το τέλος του μήνα $i-1$. Διακρίνουμε δύο περιπτώσεις ως προς τη σύγκριση των ποσοτήτων z και $d_i + s$.

- Υποθέτουμε καταρχήν ότι $z \geq d_i + s$. Αφού $z \leq S$, έπεται ότι $d_i + s \leq S$. Τότε, δεν χρειάζεται να κάνουμε παραγγελία κατά τον μήνα i. Επομένως, από τον χαρακτηρισμό της δομής της βέλτιστης λύσης,

$$\begin{aligned} \mathsf{V}[i,s] &= \min_{z \geq d_i + s, z \leq S} \left(\mathsf{V}[i-1,z] + z\,C \right) \\ &= \mathsf{V}[i-1, d_i + s] + (d_i + s)\,C\,. \end{aligned}$$

- Υποθέτουμε τώρα ότι $z < d_i + s$. Τότε, χρειάζεται να κάνουμε παραγγελία κατά τον μήνα i. Επομένως, από τον χαρακτηρισμό της δομής της βέλτιστης λύσης,

$$\begin{aligned} \mathsf{V}[i,s] &= \min_{z < d_i + s, z \leq S} \left(\mathsf{V}[i-1,z] + z\,C + K \right) \\ &= \mathsf{V}[i-1,0] + K\,. \end{aligned}$$

Θα ξαναγράψουμε τώρα την αναδρομική σχέση για την τιμή της βέλτιστης λύσης $\mathsf{V}[i,s]$, όπου $1 \leq i \leq n$ και $0 \leq s \leq S$, με διάκριση δύο περιπτώσεων ως προς τη σύγκριση των ποσοτήτων S και $d_i + s$.

- Υποθέτουμε καταρχήν ότι $d_i + s > S$. Τότε, με αντιθετοαντιστροφή λαμβάνουμε ότι $z < d_i + s$. Ετσι, είμαστε απαραίτητα στη δεύτερη των περιπτώσεων πιο πάνω όπου

$$\mathsf{V}[i,s] \;=\; \mathsf{V}[i-1,0] + K\,.$$

- Υποθέτουμε τώρα ότι $s + d_i \leq S$. Τώρα, μπορούμε να είμαστε στην πρώτη ή τη δεύτερη των περιπτώσεων πιο πάνω, και επομένως λαμβάνουμε ότι

$$\mathsf{V}[i,s] \;=\; \min\{\mathsf{V}[i-1, d_i + s] + (d_i + s)\,C,\, \mathsf{V}[i-1,0] + K\}\,.$$

Συνολικά, λοιπόν, έχουμε ότι

$$\mathsf{V}[i,s] \;=\; \begin{cases} \mathsf{V}[i-1,0] + K\,, & \text{αν } s + d_i > S \\ \min\{\mathsf{V}[i-1, d_i + s] + (d_i + s)\,C,\, \mathsf{V}[i-1,0] + K\}\,, & \text{αν } s + d_i \leq S \end{cases}\,.$$

Ο αλγόριθμος δυναμικού προγραμματισμού: Παρουσιάζουμε τώρα τον αλγόριθμό μας:

- Συμπλήρωση του πίνακα με τις λύσεις στα υποπροβλήματα: Για κάθε δείκτη i, όπου $1 \leq i \leq n$, χρησιμοποιούμε την τιμή της βέλτιστης λύσης για να υπολογίσουμε όλες τις τιμές $\mathsf{V}[i,s]$, όπου $0 \leq s \leq S$. Καταχωρούμε όλες τις τιμές που έχουμε υπολογίσει στη γραμμή i ενός πίνακα $\{1,\ldots,n\} \times \{0,\ldots,S\}$ (από αριστερά προς τα δεξιά).

  Για κάθε δείκτη i, όπου $1 \leq i \leq n$, για τον οποίο ισχύει ότι $s + d_i > S$, καταχωρούμε την απόφασή μας να παραγγείλουμε κατά τον μήνα i. Αφετέρου, αν $s + d_i \leq S$, καταχωρούμε την απόφασή μας να παραγγείλουμε κατά τον μήνα i εφόσον

$$\mathsf{V}[i-1,0] + K \;<\; \mathsf{V}[i-1, d_i + s] + (d_i + s)\,C\,.$$

- Ανακατασκευή της βέλτιστης λύσης από τον πίνακα με τις λύσεις στα υποπροβλήματα: Η τιμή της βέλτιστης λύσης που ζητούμε είναι η τιμή $V[n, 0]$ που έχουμε στη θέση $(n, 0)$ του πίνακά μας. Οι καταχωρημένες αποφάσεις μας να παραγγείλουμε ή όχι κατά τον μήνα i, όπου $1 \leq i \leq n$, ανακατασκευάζουν αυτόματα τη βέλτιστη λύση.

Πολυπλοκότητα: Χρειάζονται $O(1)$ βήματα για τον υπολογισμό οποιασδήποτε τιμής $V[i, s]$ στη θέση i του πίνακα $\{1, \ldots, n\} \times \{0, \ldots, S\}$, όπου $1 \leq i \leq n$ και $0 \leq s \leq S$, Έτσι, χρειάζονται συνολικά $O(nS)$ βήματα για τη συμπλήρωση του πίνακα με τις λύσεις στα υποπροβλήματα. Αφού η ανακατασκευή της βέλτιστης λύσης είναι αυτόματη, αυτός είναι και ο συνολικός αριθμός βημάτων.

4. (25 μονάδες) Ως ιδιοκτήτες ενός πρατηρίου βενζίνης, έχουμε να αντιμετωπίσουμε το εξής πρόβλημα: Υπάρχει μία υπόγεια αποθήκη βενζίνης η οποία μπορεί να αποθηκεύσει μέχρι και L γαλόνια βενζίνης. Η παραγγελία βενζίνης είναι ακριβή. Έτσι, θέλουμε να κρατήσουμε τον αριθμό των παραγγελιών όσο πιο μικρό γίνεται. Για κάθε παραγγελία, έχουμε να πληρώσουμε ένα κόστος K για την διανομή της βενζίνης στο πρατήριό μας (επιπρόσθετα προς το κόστος της βενζίνης). Ωστόσο, υπάρχει ένα κόστος C για την αποθήκευση ενός γαλονιού βενζίνης για μία μέρα. Έτσι, η παραγγελία μεγάλων ποσοτήτων μειώνει μεν τα κόστη διανομής (αφού θα γίνουν λιγότερες παραγγελίες), αυξάνει όμως τα κόστη αποθήκευσης.
Το πρατήριο προγραμματίζει να είναι κλειστό για δύο βδομάδες το καλοκαίρι και θέλει να κρατήσει την αποθήκη του άδεια κατά την περίοδο που θα είναι κλειστό (ώστε να έχει τότε μηδενικό κόστος αποθήκευσης). Υπάρχουν προβλέψεις για την βενζίνη που θα πωληθεί τις επόμενες μέρες $1, \ldots .n$ μέχρι να κλείσει το πρατήριο. Συγκεκριμένα, προβλέπεται ότι θα πωληθούν g_i γαλόνια βενζίνης κατά την μέρα i, όπου $1 \leq i \leq n$. (Υποθέστε ότι η αποθήκη είναι άδεια κατά την μέρα 0.)
Το υπολογιστικό πρόβλημα που έχουμε να επιλύσουμε είναι να προσδιορίσουμε τις παραγγελίες μας (πόσα γαλόνια βενζίνης θα παραγγείλουμε, και πότε) έτσι ώστε να ελαχιστοποιήσουμε το συνολικό μας κόστος. Παρουσιάστε και αναλύστε αλγόριθμο δυναμικού προγραμματισμού για το υπολογιστικό αυτό πρόβλημα. Η πολυπλοκότητα του αλγορίθμου σας πρέπει να είναι πολυωνυμική ως προς τις διάφορες παραμέτρους.

Λύση:

Προκαταρκτικά: Μία λύση προσδιορίζεται από τις ημέρες στις οποίες γίνονται (και διανέμονται) οι παραγγελίες, και τα γαλόνια βενζίνης που παραγγέλλονται. Θα εξετάσουμε μόνο λύσεις στις οποίες αγοράζεται ακριβώς τόση βενζίνη όση προβλέπεται να πωληθεί στις επόμενες μέρες μέχρι είτε την επόμενη παραγγελία είτε την τελευταία μέρα n. (Οποιαδήποτε άλλη λύση δεν μπορεί να είναι βέλτιστη.) Επίσης, σε οποιαδήποτε λύση, πρέπει να γίνει παραγγελία τη μέρα 1 (αφού προβλέπονται πωλήσεις για τη μέρα 1). Για τον προσδιορισμό του συνολικού κόστους μίας (βέλτιστης) λύσης, θα λάβουμε υπόψη τα κόστη διανομής και τα κόστη αποθήκευσης, αλλά όχι τα κόστη αγοράς της βενζίνης της ίδιας, καθώς αυτά τα κόστη είναι τα ίδια σε όλες τις λύσεις (βέλτιστες ή όχι).

Θεωρούμε τώρα μία (βέλτιστη) λύση στην οποία η πρώτη παραγγελία γίνεται τη μέρα 1, και η αμέσως επόμενη παραγγελία γίνεται τη μέρα j_2, όπου $1 < j_2 \leq n$. Τότε, τα γαλόνια στην πρώτη παραγγελία πρέπει να είναι $\sum_{i=1}^{j_2-1} g_i$ ώστε να είναι επαρκή για τις προβλεπόμενες πωλήσεις κατά τις μέρες $1, \ldots, j_2 - 1$. Επίσης, από τον περιορισμό στον χώρο αποθήκευσης, πρέπει η μέρα j_2 να επιλεγεί με τέτοιο τρόπο ώστε να έχουμε ότι $\sum_{i=1}^{j_2-1} g_i \leq L$. Το αντίστοιχο κόστος αποθήκευσης (μέχρι τη μέρα j που θα γίνει η επόμενη παραγγελία) είναι $\sum_{i=1}^{j_2-1} (i-1)\, g_i$ (αφού για κάθε δείκτη i, όπου $1 \leq i \leq j_2 - 1$, χρειάζεται να αποθηκεύσουμε g_i γαλόνια για $i - 1$ μέρες.

Γενικότερα, το κόστος αποθήκευσης για μία ποσότητα από γαλόνια $\sum_{i=1}^{j_2-1} g_i$ που παραγγέλλονται τη μέρα j_1 και αποθηκεύονται μέχρι και τη μέρα $j_2 - 1$ (συμπεριλαμβανομένης) είναι

$$\mathsf{A}[j_1, j_2 - 1] \;\; = \;\; \sum_{i=j_1}^{j_2-1} (i - j_1) g_i \,.$$

(Η επόμενη παραγγελία θα γίνει τη μέρα j_2.)

Τα υποπροβλήματα: Ορίζουμε ως υποπρόβλημα $\Pi(j_1)$, όπου $1 \leq j_1 \leq n$, το εξής υπολογιστικό πρόβλημα:

> Υποθέτουμε ότι η αποθήκη είναι άδεια με τη μέρα j_1. Με δεδομένες τις προβλεπόμενες πωλήσεις $g_{j_1}, \ldots, g_n$, προσδιόρισε τις παραγγελίες για τις μέρες $j_1, \ldots, n$ έτσι ώστε να ελαχιστοποιηθεί το συνολικό κόστος (διανομής και αποθήκευσης).

Προφανώς, υπάρχουν n τέτοια υποπροβλήματα, και το υπολογιστικό πρόβλημα που έχουμε να επιλύσουμε είναι το υποπρόβλημα $\Pi(1, n)$.

Χαρακτηρισμός της δομής της βέλτιστης λύσης: Έστω μία βέλτιστη λύση στο υποπρόβλημα $\Pi(j_1)$, όπου $1 \leq j_1 \leq n$. Ας υποθέσουμε ότι η αμέσως επόμενη παραγγελία (από εκείνη που γίνεται τη μέρα j_1) στη βέλτιστη αυτή λύση θα γίνει τη μέρα j_2. Τότε, η ακολουθία όλων των παραγγελιών πλην της πρώτης αποτελεί μία βέλτιστη λύση για το υποπρόβλημα $\Pi(j_2)$.

> Πράγματι, αν υπήρχε μία καλύτερη λύση για το υποπρόβλημα $\Pi(j_2)$, αυτή θα έδινε μαζί με την παραγγελία για τη μέρα j_1 μία καλύτερη λύση για το υποπρόβλημα $\Pi(j_1)$. Αντίφαση, καθώς ξεκινήσαμε αρχικά με μία βέλτιστη λύση για το υποπρόβλημα $\Pi(j_1)$.

Η τιμή της βέλτιστης λύσης: Έστω το υποπρόβλημα $\Pi(j_1)$, όπου $1 \leq j_1 \leq n$. Έστω $\mathsf{V}(j_1)$ η τιμή της βέλτιστης λύσης για το υποπρόβλημα $\Pi(j_1)$ — δηλαδή, το ελάχιστο δυνατό συνολικό κόστος για τις μέρες $j_1, \ldots, n$. Σε μία βέλτιστη λύση, το συνολικό κόστος αποτελείται από τα εξής κόστη:

- Το κόστος διανομής για την παραγγελία στη μέρα j_1.
- Το κόστος αποθήκευσης $\mathsf{A}[j_1, j_2 - 1]$ για τις μέρες $j_1, \ldots, j_2 - 1$ μέχρι και τη μέρα j_2 που θα γίνει η επόμενη παραγγελία.
- Το συνολικό κόστος για τις υπόλοιπες μέρες $j_2, \ldots, n$.

Αφού η λύση είναι βέλτιστη, η μέρα j_2 πρέπει να επιλεγεί ώστε να ελαχιστοποιεί το συνολικό κόστος. Έτσι, από τον χαρακτηρισμό της δομής της βέλτιστης λύσης, έχουμε ότι

$$\mathsf{V}[j_1] \;\; = \;\; K + \min_{j_2 > j_1 \;\mid\; \sum_{i=j_1}^{j_2-1} g_i \leq L} \mathsf{A}[j_1, j_2 - 1] \,.$$

Ο αλγόριθμος δυναμικού προγραμματισμού: Παρουσιάζουμε τώρα τον αλγόριθμό μας:

- Φάση προεπεξεργασίας: Υπολογίζουμε όλες τις τιμές $\mathsf{A}[j_1, j_2-1]$, όπου $1 \leq j_1 < j_2 \leq n+1$.
- Συμπλήρωση του πίνακα με τις λύσεις στα υποπροβλήματα: Για κάθε δείκτη j_1 κατά φθίνουσα τάξη, όπου $1 \leq j_1 \leq n$, χρησιμοποιούμε την τιμή της βέλτιστης λύσης για να υπολογίσουμε την τιμή $\mathsf{V}[j_1]$. Καταχωρούμε κάθε τιμή $\mathsf{V}[j_1]$ που υπολογίζουμε στην αντίστοιχη θέση j_1 ενός πίνακα $\{1, \ldots, n\}$. Για κάθε θέση j_1 του πίνακά μας, καταχωρούμε επιπρόσθετα τις εξής ποσότητες:
  - Καταχωρούμε την μέρα $j_2 = j_2(j_1)$ για την οποία λάβαμε ότι

    $$\mathsf{V}[j_1] \;=\; K + \mathsf{A}[j_1, j_2 - 1]\,,$$

    ως την μέρα της αμέσως επόμενης παραγγελίας από τη μέρα j_1.
  - Χρησιμοποιούμε τέλος τη μέρα j_2 για να υπολογίσουμε τα γαλόνια βενζίνης $\sum_{i=j_1}^{j_2-1} g_i$ που χρειάζεται να παραγγείλουμε τη μέρα j_1, και καταχωρούμε την ποσότητα αυτή στη θέση j_1 ως τα γαλόνια της παραγγελίας για τη μέρα j_1.
- Ανακατασκευή της βέλτιστης λύσης από τον πίνακα με τις λύσεις στα υποπροβλήματα: Η τιμή της βέλτιστης λύσης που ζητούμε είναι η τιμή $\mathsf{V}[1]$ που έχουμε στη θέση 1 του πίνακά μας. Επίσης, η αντίστοιχη βέλτιστη λύση ανακατασκευάζεται αναδρομικά από τον πίνακα $\{1, \ldots, n\}$ ως εξής:
  - Από τη θέση 1 του πίνακά μας, λαμβάνουμε τα γαλόνια της παραγγελίας μας για τη μέρα 1 και τη μέρα $j_2 = j_2(1)$ της αμέσως επόμενης παραγγελίας από τη μέρα 1.
  - Χρησιμοποιούμε τον υποπίνακα $\{j_2, \ldots, n\}$ για να ανακατασκευάσουμε αναδρομικά τη βέλτιστη λύση στο υποπρόβλημα $\Pi(j_2)$.

Πολυπλοκότητα: Χρειάζονται $O(n^2)$ βήματα για τον υπολογισμό όλων των τιμών $\mathsf{A}[j_1, j_2 - 1]$, όπου $1 \leq j_1 < j_2 \leq n+1$. Ακολούθως, χρειάζονται $O(n)$ βήματα για τον υπολογισμό οποιασδήποτε τιμής $\mathsf{V}[j_1]$, όπου $1 \leq j_1 \leq n$. Έτσι, απαιτούνται συνολικά $O(n^2)$ βήματα για τη συμπλήρωση του πίνακα με τις λύσεις στα υποπροβλήματα. Η (αναδρομική) ανακατασκευή της βέλτιστης λύσης από τον ίδιο πίνακα απαιτεί επιπρόσθετα $O(n)$ βήματα. Έτσι, ο συνολικός αριθμός βημάτων είναι $O(n^2)$.