Άσκηση 1: Χημικά Απόβλητα

Καλούμαστε να λύσουμε το πρόβλημα του διαχωρισμού των N ουσιών σε K φιάλες, διατηρώντας την διάταξη των ουσιών, ώστε να ελαχιστοποιήσουμε τη συνολική ενέργεια που προκύπτει από την συσσώρευση διαφορετικών χημικών στις φιάλες. Θα χρησιμοποιήσουμε την προσέγγιση του Δυναμικού Προγραμματισμού. Συγκεκριμένα, έστω $dp(i, j)$ η ελάχιστη ενέργεια για τοποθέτηση των χημικών 1 έως j σε i-φιάλες. Η ζητούμενη έξοδος του προβλήματος προκύπτει ως $dp(K, N)$. Ξεκινώντας από το τέλος και επιλέγοντας μέχρι ποια ουσία θα τοποθετηθεί στην τελευταία φιάλη, προκύπτει η ακόλουθη αναδρομική σχέση για το πρόβλημα:

$$dp(i, j) = min_{l<j}(dp(i-1, l) + \sum_{a=l+1}^{j} \sum_{b=a+1}^{j} A_{ab})$$

Στη συνέχεια θα εξετάσουμε διαφορετικούς τρόπους για τον υπολογισμό της αναδρομικής σχέσης, που θα οδηγήσουν σε διαφορετικές πολυπλοκότητες για το πρόβλημα.

Λύση 1: Άμεσος Υπολογισμός - $O(KN^4)$
Μία απλή επίλυση της παραπάνω αναδρομικής θα οδηγήσει στον υπολογισμό KN-υποπροβλημάτων. Κάθε υποπρόβλημα πρέπει να υπολογίσει το ελάχιστο πάνω σε $O(N)$ ποσότητες και για κάθε μία ποσότητα, πρέπει να υπολογίσουμε το διπλό άθροισμα της αναδρομής σε $O(N^2)$, οδηγώντας σε μία συνολική πολυπλοκότητα $O(KN^4)$.

Λύση 2: Προϋπολογισμός Αθροισμάτων - $O(KN^2)$
Μία πρώτη βελτίωση είναι να εξαλείψουμε την τετραγωνική πολυπλοκότητα που απαιτείται για τον υπολογισμό του διπλού αθροίσματος μέσα στο ελάχιστο. Αφού διαβάσουμε τον πίνακα A, υπολογίζουμε τις ποσότητες $C(x, y)$ που ορίζονται ως το άθροισμα των στοιχείων του πίνακα A με δείκτη γραμμής $i \leq x$ και δείκτη στήλης $j \leq y$. Για τον υπολογισμό αυτών των ποσοτήτων, μπορούμε να χρησιμοποιήσουμε την αναδρομική σχέση

$$C(x, y) = C(x-1, y) + C(x, y-1) - C(x-1, y-1) + A[x, y]$$

η οποία θα μας επιτρέψει να υπολογίσουμε και να αποθηκεύσουμε τις τιμές $C(x, y)$ για κάθε $x, y \in [1, N]$ σε χρόνο $O(N^2)$. Έπειτα, κάθε φορά που χρειαζόμαστε να υπολογίσουμε κάποιο διπλό άθροισμα στην αναδρομική σχέση, κάνουμε χρήση της σχέσης

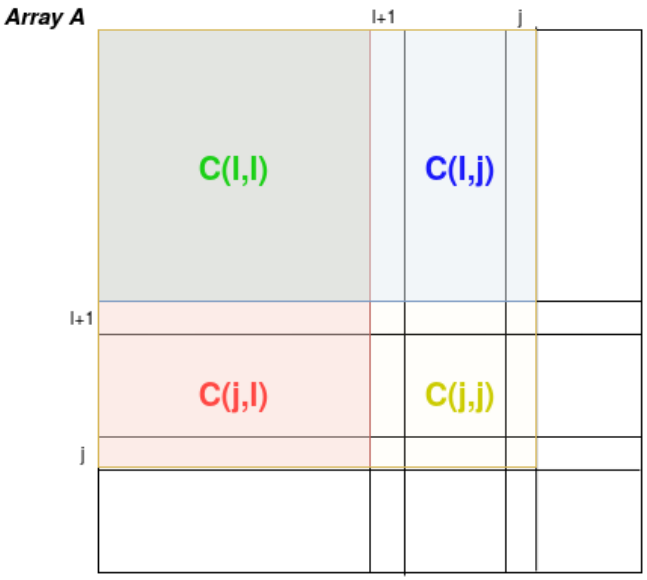
$$\sum_{a=l+1}^{j} \sum_{b=a+1}^{j} A_{ab} = \frac{1}{2}(C(j, j) - C(j, l) - C(l, j) + C(l, l))$$

(βλέπε Σχήμα 1) η οποία μας δίνει την ζητούμενη τιμή σε χρόνο $O(1)$ και ρίχνει την συνολική πολυπλοκότητα της λύσης σε $O(KN^2)$.

Λύση 3: Divide & Conquer Optimization - $O(N^2 + KNlogN)$
Η βασική ιδέα είναι να χρησιμοποιήσουμε μία ιδιότητα μονοτονίας πάνω στους minimizers της αναδρομικής σχέσης ώστε να μην χρειάζεται να κοιτάμε όλες τους ουσίες $l < j$ στα ελάχιστα, αλλά μόνο κάποιες από αυτές χρησιμοποιώντας προηγούμενα αποτελέσματα. Ορίζουμε τους minimizers της αναδρομικής ως

$$opt(i, j) = argmin_{l<j}(dp(i-1, l) + \sum_{a=l+1}^{j} \sum_{b=a+1}^{j} A_{ab}),$$

δηλαδή $opt(i, j) \in [1, j-1]$ είναι το τελευταίο χημικό που δεν μπαίνει στην τελευταία φιάλη της βέλτιστης λύσης με i-φιάλες και χημικά 1 έως j. Μπορούμε να αποδείξουμε οτι για τους παραπάνω minimizers ισχύει πως $opt(i, j) \leq opt(i, j+1)$, δηλαδή αν έχω περισσότερα χημικά και ίδιες φιάλες, με συμφέρει πάντα να "κόψω" την τελευταία παρτίδα αργότερα. Αυτή η ιδιότητα μονοτονίας μας επιτρέπει να χρησιμοποιήσουμε τον ακόλουθο αλγόριθμο:



Σχήμα 1: Υπολογισμός Διπλών Αθροισμάτων

1. Υπολογισμός αρχικής συνθήκης $opt(1, j) = 0$ και $dp(1, j) = \frac{1}{2}C(1, j)$ για όλα τα $j = 1, \ldots, N$.
2. Υπολογισμός δεύτερης γραμμής πινάκων $opt, cost$.

   ⋮

3. Υπολογισμός K-οστής γραμμής πινάκων $opt, cost$ και τύπωμα $opt(K, N)$.

Η κάθε γραμμή υπολογίζεται με Divide & Conquer, αξιοποιώντας την μονοτονία των $opt(i, j)$. Δηλαδή, δεδομένης της γραμμής $i - 1$ υπολογίζουμε την γραμμή i ως:

$$opt(i, \frac{N}{2}) = argmin_{l<\frac{n}{2}}(cost(i-1, l) + \sum_{a=l+1}^{j} \sum_{b=a+1}^{\frac{N}{2}} A_{ab}) = B$$

$$opt(i, \frac{N}{4}) = argmin_{l<min(\frac{n}{4},B)}(cost(i-1, l) + \sum_{a=l+1}^{j} \sum_{b=a+1}^{\frac{N}{4}} A_{ab})$$

$$opt(i, \frac{3N}{4}) = argmin_{B\leq l<min(\frac{n}{4},B)}(cost(i-1, l) + \sum_{a=l+1}^{j} \sum_{b=a+1}^{\frac{3N}{4}} A_{ab})$$

κλπ. Ουσιαστικά απορρίπτουμε πιθανές τιμές από τους δείκτες των ελαχίστων και τελικά αντί για N^2 πετυχαίνουμε τον υπολογισμό γραμμής σε $O(NlogN)$. Για K γραμμές του πίνακα και $O(N^2)$ χρόνο για τον προϋπολογισμό των τελικών αθροισμάτων, τελικά λύνουμε το πρόβλημα σε $O(N^2 + KNlogN)$.

Λύση 4: Knuth Optimization - $O(N^2)$
Το Knuth Optimization είναι μία τεχνική βελτιστοποίησης προβλημάτων Δυναμικού Προγραμματισμού που χρησιμοποιήθηκε από τον Knuth για να λύσει σε $O(n^2)$ το πρόβλημα του Optimal Binary Search Tree. Ουσιαστικά, επεκτείνει το Divide and Conquer Optimization, βασιζόμενο στην πιο ισχυρή ιδιότητα μονοτονίας

$$opt(i-1, j) \leq opt(i, j) \leq opt(i, j+1),$$

δηλαδή τώρα χρειαζόμαστε και κάποιο monotonicity ως προς τις φιάλες, ενώ πριν χρειαζόμασταν μόνο monotonicity ως προς τις ουσίες. Η ιδιότητα αυτή μας επιτρέπει να υπολογίσουμε όλους τους minimizers $opt(i, j)$ με σταθερή διαφορά $i - j$ σε συνολικό χρόνο $O(j)$, ρίχνοντας την συνολική πολυπλοκότητα του προβλήματος σε $O(N^2)$ και βελτιώνοντας το Divide and Conquer optimization για μεγάλες τιμές της

παραμέτρου K. Για περισσότερες αναφορές σχετικά με το Knuth Optimization, και γενικά την βελτιστοποίηση αναδρομικών σχέσεων Δυναμικού Προγραμματισμού, τις περιπτώσεις στις οποίες εφαρμόζεται, τις αντίστοιχες τεχνικές και παραδείγματα προβλημάτων, σας παραπέμπουμε εδώ.

Άσκηση 2: Αγορές στο Φαράγγι

Προκειμένου να λύσουμε το εν λόγω πρόβλημα θα ξεκινήσουμε με μια πιο απλή εκδοχή. Συγκεκριμένα, έστω ότι είχαμε έναν έμπορο και ένα προιόν και θέλαμε να υπολογίσουμε το ελάχιστο ποσό χρημάτων που πρέπει να διαθέσουμε ώστε να αποκτήσουμε τουλάχιστον N προιόντα.

Ορίζουμε $dp[i, n]$ ως το ελάχιστο ποσό που πρέπει να διαθέσουμε ώστε να αγοράσουμε ακριβώς n προιόντα χρησιμοποιώντας τις πρώτες i προσφορές του εμπόρου. Προκύπτει επομένως η ακόλουθη αναδρομική σχέση:

$$dp[i, n] = min(dp[i-1, n], dp[i-1, n-a_i] + p_i)$$

Όπου a_i, p_i το πλήθος προιόντων καθώς και η τιμή της i-οστής προσφοράς. Μπορούμε πολύ εύκολα να χρησιμοποιήσουμε τον παραπάνω πίνακα προκειμένου να υπολογίσουμε το ελάχιστο ποσό που πρέπει να διαθέσουμε ώστε να αποκτήσουμε τουλάχιστον n προιόντα ως εξής:

$$total[n] = min(total[n+1], dp[M, n])$$

Όπου total[n] το ελάχιστο ποσό που χρειαζόμαστε για τουλάχιστον n προιόντα και M το συνολικό πλήθος προσφορών του εμπόρου.

Λύνοντας το γενικό πρόβλημα Έστω πως υπολογίζουμε την παραπάνω αναδρομική σχέση για κάθε συνδυασμό εμπόρων και είδος προιόντων. Συγκεκριμένα, έστω $total[x][y][n]$ το ελάχιστο ποσό που πρέπει να διαθέσουμε ώστε να αποκτήσουμε τουλάχιστον n προιόντα τύπου y απ' τον έμπορο x (π.χ total[1][A][10] είναι τα χρήματα που πρέπει να διαθέσουμε για να αγοράσουμε τουλάχιστον 10 προιόντα τύπου A απ' τον έμπορο 1).

ορίζουμε $set\_cost(x, k)$ ως ελάχιστο ποσό που πρέπει να δώσουμε ώστε να αποκτήσουμε k σετ προιόντων απ' τον έμπορο x. Προκειμένου να γίνει αυτό χρειαζόμαστε τουλάχιστον k αντικείμενα τύπου A, B και C απ'τον έμπορο x, καταλήγουμε επομένως στον ακόλουθο τύπο:

$$set\_cost(x, k) = \sum_{y \in \{A,B,C\}} total[x][y][k]$$

Για να καταλήξουμε στην λύση του προβλήματος συνδυάζουμε τα παραπάνω. Θέλουμε να κατασκευάσουμε N τω πλήθος sets απ' τους 3 εμπόρους αλλά μέχρι στιγμής γνωρίζουμε τον βέλτιστο τρόπο να κατασκευάσουμε sets για τον καθένα ξεχωριστά (μας δίνεται απ' την τιμή της συνάρτησης set_cost). Συνεπώς η βέλτιστη λύση του προβλήματος δίνεται απ' τον παρακάτω τύπο:

$$\min_{0 \le a \le N, 0 \le b \le N} (set\_cost(1, a) + set\_cost(2, b) + set\_cost(3, N-a-b))$$

Ανάλυση Πολυπλοκότητας. Χρειαζόμαστε $\Theta(NM)$ χρόνο για τον υπολογισμό των πινάκων dp και total αντίστοιχα και επιπλέον $\Theta(N^2)$ για την εύρεση της βέλτιστης λύσης μέσω των συνδυασμών. Συνεπώς η συνολική πολυπλοκότητα του αλγορίθμου είναι $\Theta(N^2 + NM)$.

Γενίκευση του προβλήματος. Στην γενική περίπτωση όπου έχουμε D εμπόρους και D προιόντα χρειαζόμαστε χρόνο $O(D^2NM + N^{D-1})$ υπάρχει τρόπος να λύσουμε το πρόβλημα σε πολυωνυμικό χρόνο ως προς M και D(Έχοντας δηλαδή μόνο μια ψευδοπολυωνυμική εξάρτηση απ' το N);