ΟΝ/ΜΟ: ΚΩΝΣΤΑΝΤΙΝΟΣ ΝΙΚΟΛΕΤΟΣ

Α.Μ: 1115201700104

Προβλήματα – Εργασία 1

Πρόβλημα 1

PacMan/Project1

Πρόβλημα 2

Ο συνολικός αριθμός κόμβων που παράγονται από την αναζήτηση επαναληπτικής εκβάθυνσης (IDS) είναι :

N(IDS) = (d)b + (d-1)b^2 +...+ (1)b^d

όπου b:παράγοντας διακλάδωσης και d:βάθος .

- Ο μικρότερος αριθμός κόμβων που μπορούν να παραχθούν από αυτόν τον αλγόριθμο είναι **MIN(IDS) = g*b** , όπου g<=d , κάτι που συμβαίνει στην περίπτωση όπου η πρώτη επανάληψη επιτύχει , δηλαδή για ύψος 1 η πρώτη κλήση DFS βρει τον επιθυμητό στόχο.

-Ο μεγαλύτερος αριθμός κόμβων , είναι : **MAX(IDS) = (g)b + (g-1)b^2 +...+ (1)b^g**

Στην περίπτωση που ο κόμβος βρίσκεται στην τελευταία DFS και έχει γίνει expand σε όλους τους κόμβους αρκετές φορές.

Πρόβλημα 3

- Αλγόριθμος πρώτα σε πλάτος (BFS)

Η λίστα σύνορο είναι μια Ουρά

(α) **G1**

(β) Εξέλιξη λίστας:

S → A,B,D → B,D,G1 → D,G1,C → G1,C,E

→ C,E,G2 → E,G2,F→G2,F,G3 → F,G3 →G3 → EMPTY

Σειρά αφαίρεσης των κόμβων απο το frontier: **S,A,B,D,G1,C,E,G2,F,G3**

- Αλγόριθμος πρώτα σε βάθος (DFS)

Η λίστα σύνορο είναι μια Στοίβα

(α) **G3**

(β) Εξέλιξη λίστας:

S → A,B,D → A,B,C,E → A,B,C,G3 → A,B,C → A,B,F,G2 → A,B,F

→ A,B → A → G1→ EMPTY

Σειρά αφαίρεσης των κόμβων από το frontier: **S,D,E,G3,C,G2,F,B,A,G1**

- Αλγόριθμος πρώτα σε βάθος με επαναληπτική εκβάθυνση (IDS)

Η λίστα σύνορο είναι μια Στοίβα

(α) **G1**

(β) Εξέλιξη λίστας:

S (Depth = 0)→ S,A,B,D (Depth = 1)→ S,A,G1,B,C,E,G3 (Depth = 2)

→ S,A, G1,G2,B,C,F,G2,D,E,G3 (depth = 4)→ EMPTY

Σειρά αφαίρεσης των κόμβων απο το frontier:
S,S,A,B,D,S,A,G1,B,C,D,E,S,A,G1,G2,B,C,F,D,E,G3

- Αλγόριθμος άπληστη αναζήτηση πρώτα στον καλύτερο

Η λίστα σύνορο είναι μια PriorityQueue με προτεραιότητα το αποτέλεσμα της ευρετικής.

(α) **G2**

(β) Εξέλιξη λίστας:

(S,5) → (B,3),(D,6),(A,7) → (C,4),(D,6),(A,7) → (G2,0),(D,6),(F,6),(A,7) →

(D,6),(F,6),(A,7) → (E,5),(F,6),(A,7) → (G3,0),(F,6),(A,7)→ (F,6),(A,7) → (A,7)
→ (G1,0) → EMPTY

Σειρά αφαίρεσης των κόμβων απο το frontier: **S,B,C,G2,D,E,G3,F,A,G1**

- Αλγόριθμος A*

Η λίστα σύνορο είναι μια PriorityQueue με προτεραιότητα το αποτέλεσμα της ευρετικής συν το κόστος.

(α) **G1**

(β) Εξέλιξη λίστας:

(S,5) →(A,12),(B,12),(D,12) → (B,12),(D,12),(G1,14) → (D,12),(C,14),(G1,14) → (E,13),(C,14),(G1,14)→(C,14),(G1,14),(G3,15)→ (G1,14),(G2,15),(G3,15), (F,23) → (G2,15),(G3,15),(F,23) → (G3,15),(F,23) → (F,23) → EMPTY

Σειρά αφαίρεσης των κόμβων απο το frontier: **S,A,B,D,E,C,G1,G2,G3,F**

Πρόβλημα 4

Το πρόβλημα των Σφακιανών πιτών μοιάζει με το γνωστό πρόβλημα Pancakes sorting.

Η συνάρτηση f(n) εκφράζει τον μικρότερο αριθμό απο αναποδογυρίσματα .Επομένως :

(α.) Ο ελάχιστος αριθμός από αναποδογυρίσματα για την πιο αταξινόμητη στοίβα από πίτες είναι ο f(n).

-Αν n=1 , τότε δεν χρειάζεται να αναποδογυρίσουμε καμία σφακιανή πίτα .

-Αν n=2 , τότε f(n) = 1 , καθώς με μια φορά οι πίτες θα έρθουν στην σωστή σειρά

-Αν n=3 , τότε f(n) = 3

-Αν n=4 , τότε f(n) = 4

(β.)

Έστω πως έχουμε μια στοίβα από πίτες όπου συμβολίζουμε με x,y δύο από αυτές.

-Έστω πως οι πίτες δεν είναι διαδοχικές.

Σε αυτή την περίπτωση θα χρειαστεί να βάλουμε την σπάτουλα και να αλλάξουμε το πλήθος απο πίτες [x,..,y] φτιάχνοντας την σειρά σε μια ή περισσότερες πίτες.

Οπότε f(n)>=n

-Έστω πως οι πίτες είναι διαδοχικές

Σίγουρα κάποια στιγμή θα πρέπει η σπάτουλα να μπει ανάμεσα απο αυτές τις 2 πίτες ,έτσι ώστε να καταλήξουμε στο επιθυμητό αποτέλεσμα. Επομένως θα πρέπει για Β διαδοχικά λάθος ζευγάρια να κάνουμε Β αλλαγές. Δηλαδή , για περισσότερες απο 4 πίτες(είτε άρτιο-είτε περιττό αριθμό πιτών) , στο χειρότερο σενάριο ,όπου όλες οι διαδοχικές πίτες αποτελούν λάθος ζευγάρι , θα χρειαστούν n αλλαγές.

Επομένως f(n)>=n .

(γ.)

-Για n=1 ,τότε δεν χρειάζεται να αναποδογυρίσουμε καμία σφακιανή πίτα.

-Για n=2 , μπορούμε να φέρουμε την μεγαλύτερη πίτα στο τέλος της στοίβας με 1 κινήση .

-Για n>2 ,μπορούμε να φέρουμε την μεγαλύτερη πίτα στο τέλος της στοίβας με 1 κινήση και να φτίαξουμε τις υπόλοιπες αναδρομικά. Δηλαδή:

f(1) = 0 and f(2)<=1 ,επομένως f(n) <= 1+f(n-1) <= 2*n (κατ'επέκταση του 2n)

δηλαδή f(n) <= 2n.

(δ.) Διατύπωση σαν πρόβλημα ανάζητησης :

- Αρχική κατάσταση : Αταξινόμητη στοίβα απο πίτες
- Διαθέσιμες ενέργειες : Αναποδογύρισμα ενός αριθμού απο πίτες
- Συνάρτηση κόστους : Η συνάρτηση f(n) = αριθμός απο κινήσεις
- Κόστος βήματος: 1 αναποδογύρισμα
- Έλεγχος στόχου : Αν οι πίτες είναι ταξινομημένες
- Καταστάσεις στόχου : Σε οποιαδήποτε σημεία της στοίβας να έχουμε ταξινομημένα κομμάτια
- Στόχος : Να ταξινομηθεί η στοίβα απο πίτες
- Συνάρτηση διαδοχής : Επιστρέφει την κατάσταση της στοίβας και το αναποδογυρίσματα που μπορεί να γίνει και να διατηρηθούν οι ιδιότητες της στοίβας και την κατάσταση της στοίβας
- Λύση : Η ταξινομημένη στοίβα απο πίτες

Το μέγεθος του χώρου καταστάσεων θα είναι n! .

Πρόβλημα 5

- **Διατύπωση σαν πρόβλημα ανάζητησης :**

– Αρχική κατάσταση : Κ ανθρωπάκια σε τυχαίες θέσεις, εκτός από τα μαύρα κουτάκια και 1 σε κάθε κουτάκι
– Διαθέσιμες ενέργειες : Δεξία,αριστερά,πάνω,κάτω κατά 1 τετραγωνάκι
– Συνάρτηση κόστους : Κόστος όλων των βημάτων και για τα Κ ανθρωπάκια
– Κόστος βήματος: Κάθε βήμα έχει κόστος 1 ή 2 ανάλογα με το τετραγωνάκι
– Συνάρτηση διαδοχής: Επιστρέφει ζεύγη με την επιτρεπτή κίνηση(καφέ ή γεμάτο κουτάκι) για τα Κ ανθρωπάκια και την κατάσταση τους

(Κίνηση Κ-0 , Κατάσταση Κ-0, …, Κίνηση-(Κ-1) , Κατάσταση (Κ-1))

– Έλεγχος στόχου : Αν τα Κ ανθρωπάκια έχουν φθάσει στον στόχο τους
– Καταστάσεις στόχου :Να φθάσουν τα ανθρωπάκια στα Ε
– Στόχος : Να βρεθούν ταυτόχρονα τα ανθρωπάκια στα Ε
– Λύση : Κ μονοπάτια από την αρχική κατάσταση μέχρι τους στόχους

* Επιτρεπτές κινήσεις είναι όσες δεν πάνε σε μαύρο κουτάκι , κατειλημμένο ή κουτάκι στόχο , αν δεν είναι και όλα τα Κ-1 ανθρωπάκια σε απόσταση 1 από τον δικό τους στόχο.

- Το μέγεθος του χώρου καταστάσεων θα είναι **(n*n)^K.**

- Ο μέγιστος παράγοντας διακλάδωσης είναι: **K^4** , όπου 4 είναι οι πιθανές κινήσεις για κάθε ανθρωπάκι και Κ ο αριθμός των ανθρωπακίων

- Στην περίπτωση που περιγράφει το πρόβλημα δεν γίνεται(δεν βρήκα κάποιο πιθανό μονοπάτι), καθώς τα ανθρωπάκια απέχουν περιττό αριθμό ενώ ,τα ευρώ άρτιο αριθμό. Αυτό έχει σαν αποτέλεσμα σε κάθε προσπάθεια το ένα να κάνει άρτιο αριθμό και το άλλο περιττό αριθμό βημάτων.

  Έτσι δεν φτάνουν ταυτόχρονα και το πρόβλημα δεν παρουσιάζει λύση.

- Μια **ευρετική συνάρτηση** που θα χρησιμοποιούσα στον A* θα είχε ως κριτήριο την τρέχουσα απόσταση των Κ σημείων σε σχέση με την απόσταση στόχου , σε συνδυασμό με το πλήθος τον μαύρων τετραγώνων που βρίσκονται μεταξύ τους. Συγκεκριμένα η ευρετική θα επέστρεφε :

  Sum(Απόσταση από το ανθρωπάκι Κ μέχρι το Ε) *για κάθε Κ ανθρωπάκι*

  + Sum(Αριθμός από μαύρα κουτάκια στη μεταξύ τους απόσταση) *ομοίως*

εναλλακτικά πιθανές ευρετικές θα επέστρεφαν:

1. **Sum(Απόσταση από το ανθρωπάκι Κ μέχρι το Ε στόχου**)
2.

Sum(Απόσταση από το ανθρωπάκι Κ μέχρι το Ε)

+Sum(Αριθμός από μαύρα κουτάκια στη μεταξύ τους απόσταση)

+Sum(Αριθμός από γεμάτα κουτάκια στη μεταξύ τους απόσταση)

+Sum(Αριθμός από γεμάτα κουτάκια στη μεταξύ τους απόσταση)

- **Παρατηρήσεις:**

  -Οι ευρετικές και γενικότερα οι συναρτήσεις που προαναφέρονται , χρησιμοποιούν τα δεδομένα όλων των Κ δεδομένων καθώς δεν μπορούμε να εξετάσουμε μεμονωμένα και ξεχωριστά τα ανθρωπάκια . Σε μια τέτοια περίπτωση θα έφταναν τα ανθρωπάκια ξεχωριστά και όχι ταυτόχρονα.

  -Τα μονοπάτια που θα επιλέγονται από αυτόν τον αλγόριθμο μπορεί να μην είναι μεμονωμένα βέλτιστα καθώς μπορεί κάποιο ανθρωπάκι να επιλέξει μια μακρύτερη διαδρομή "περιμένοντας" τα υπόλοιπα.

  *Η συνάρτηση Sum(πεδίο) επιστρέφει το άθροισμα των πεδίων για όλα τα Κ ανθρωπάκια