Ονοματεπώνυμο: Ορφανίδης Ελευθέριος
Α.Μ.: 1115201400133

Προβλημα 2:

Αν ο κόμβος - στόχος με μικρότερο βάθος βρίσκεται σε βάθος g <= d και το δεντρο έχει παραγοντα διακλαδωσης b τότε χρησιμοποιώντας τον IDDFS ξέρουμε οτι ο αλγοριθμος έχει "τρέξει" g-1 φορές χωρίς να βρει αποτέλεσμα. Άρα μέχρι αυτό το σημείο έχουν ελεγχθεί :

(1): $g + (g-1) \cdot b + (g-2) \cdot b^2 + ... + 1 \cdot b^{g-1}$

Αυτό εν συντομία σημαίνει ότι η ρίζα έχει ελεγχθεί g φορές , οι κόμβοι με βάθος 1 έχουν ελεγχθεί g-1 φορές κοκ. με τους κόμβους σε βάθος g-1 να έχουν ελεγχθεί 1 φορά. Ο συνολικός αριθμός κόμβων που θα ελεγχθούν μέχρι να βρεθεί ο κόμβος - στόχος εξαρτάται από την θέση του κόμβου στο επίπεδο g. Συγκεκριμένα αν ο κόμβος- στόχος είναι ο πιο αριστερά κόμβος στο βάθος g τότε θα χρειαστεί ο έλεγχος g ακόμα κόμβων. Αυτή είναι και η περίπτωση που θα ελεγχθούν οι ελάχιστοι κόμβοι από τον αλγόριθμο καθώς στην g-οστή επανάληψη θα καταλήξει κατευθείαν στον κόμβο-στόχο χωρίς να κάνει expand κανένα παιδί εκτός από τα αριστερά παιδιά κάθε κόμβου σε βάθος μικρότερο του g. Η χειρότερη περίπτωση είναι ο κόμβος - στόχος να βρίσκεται στο δεξιότερο σημείο του δέντρου καθώς έτσι θα πρέπει να γίνουν expand όλοι οι κόμβοι των προηγούμενων επιπέδων (με βάθος < g). Σε αυτή την περίπτωση θα ελεγχθούν:

(2): $(g+1) + g \cdot b + (g-1) \cdot b^2 + ... + 2 \cdot b^{g-1} + b^g$

Οπότε καταλήγουμε στο συμπέρασμα ότι :
min = (1) + g
max = (2)

Προβλημα 3:

Breadth-First Search: Στην BFS χρησιμοποιούμε ως fringe μια FIFO queue οπότε θα εισαχθούν σε αυτή με την σειρά οι κόμβοι: S, A, B, D . Συγκεκριμένα θα εισαχθεί αρχικά ο S, θα γίνει pop() και θα τον κάνουμε expand. Άρα θα εισαχθούν στην συνέχεια με την σειρά οι A, B, D . Τώρα θα γίνει pop() ο A από το queue και θα γίνει expand, οπότε θα βρούμε τον G1 που αποτελεί και κόμβο -στόχο οπότε σταματάει η αναζήτηση. Άρα με την BFS φτάνουμε στον κόμβο G1 και η κόμβοι που εξάγονται από το fringe είναι με την σειρά: S, A . (Ο έλεγχος αν κάποιος κόμβος είναι κόμβος -στόχος γίνεται πριν αυτός εισαχθεί στο fringe.)

Depth-First Search: Η DFS χρησιμοποιεί ως fringe μια Stack. Στην υλοποίηση της BFS αυτής όταν κάποιος κόμβος γίνει expand εισάγω τους successor του στο fringe με αντίστροφη από την αλφαβητική σειρά, ώστε να γίνονται pop() με αλφαβητική σειρά Αρχικά εισάγεται στο fringe ο κόμβος

S , γίνεται pop() και στην συνέχεια γίνεται expand. Κάνοντας expand τον S εισάγονται στο fringe με την σειρά οι: D, B, A. Κάνουμε pop() από το fringe τον A (η Stack λειτουργει ως LIFO) και τον κάνουμε expand. Φτάνουμε έτσι στον κόμβο G1 που αποτελεί και κόμβο στόχο και σταματάει η αναζήτηση Άρα α) ο κόμβος - στόχος που βρίσκουμε με την DFS είναι ο G1 και οι κόμβοι που εξάγονται από το fringe είναι με την σειρά οι: S,A .

Iterative Deepening DFS: Για depth = 0 , ο κόμβος που ελεγχεται είναι ο S. Για depth = 1 , οι κόμβοι που ελεγχονται: S(γίνεται απλά expand), A, B, D. Για depth = 2 οι κόμβοι που ελέγχονται είναι οι: S(expand),A(expand),G1. Άρα και με τον iterative deepening DFS καταλήγουμε στον G1.

Best-First Search: Τόσο η Best-First Search όσο και η A* Search χρησιμοποιούν για fringe ένα priority queue. Με την άπληστη αναζήτηση πρώτα στον καλύτερο καταλήγουμε στον G2, με τους κόμβους που έχουν εξαχθεί από το fringe να ειναι οι: S,B,C. Συγκεκριμένα εισάγεται στο fringe ο κόμβος S και γίνεται expand . Μετά το expand εισάγονται στο fringe οι εξής κόμβοι: (A,7), (B,3), (D,6). Από αυτούς κάνουμε pop() αυτόν με το μικρότερο κόστος, άρα τον B , και τον κάνουμε expand. Μετά από αυτό το expand το fringe είναι της μορφής: (A,7), (D,6), (C,4) .(Το A είναι ήδη στο fringe οπότε δεν το ξαναβάζουμε).Με τον ίδιο τρόπο κάνουμε pop() από το fringe το C καθώς έχει την μικρότερη τιμή και βρίσκουμε τον G2 .

A* Search: Αρχικά εισάγουμε στο fringe το (S,5) και το κάνουμε pop() και μετά expand. Οπότε εισάγονται στο fringe τα : (SA,12), (SB,12), (SD,12). Από αυτά κάνουμε pop() το SA (αλφαβητική προτεραιότητα) και το κάνουμε pop() και expand, με το fringe να έχει την μορφη: (SB,12), (SD,12), (SAB,11), (SAG1,14). Από αυτά κάνουμε pop() και expand το SAB με το fringe να είναι πια: (SB,12), (SD,12), (SAG1,14), (SABC,13). Στο επόμενο βήμα κάνουμε pop() και expand το SB και το fringe είναι : (SD,12), (SAG1,14), (SABC,13), (SBC,14), (SBA,18) . Στην συνέχεια κάνουμε pop() και expand τον SD με το fringe να διαμορφώνεται ως εξής: (SAG1,14), (SABC,13), (SBC,14), (SBA,18), (SDC,12), (SDE,13). Παρατηρούμε ότι το μικρότερο κόστος το έχει ο SDC οπότε τον κάνουμε pop() και expand με το fringe να γίνεται: (SAG1,14), (SABC,13), (SBC,14), (SBA,18), (SDE,13), (SDCF,21), (SDCG2, 13). Λόγω αλφαβητικής προτεραιότητας γίνεται pop και expand αρχικά ο SDE ο οποίος θα προσθέσει στο fringe το (SDEG3,15). Τέλος γίνεται pop() ο SDCG2 που αποτελεί κόμβο -στόχο. Άρα ο κόμβος στόχος στον οποίο καταλήγουμε με τον A* είναι ο G2 και οι κόμβοι που εξήχθησαν από το fringe είναι οι εξής: S, SA, SAB, SB, SD, SDC, SABC, SDE, SDCG2.