ΠΛΗ31

ΕΝΟΤΗΤΑ 1: ΑΝΑΖΗΤΗΣΗ

Μάθημα 1.3:
Ευρετική Αναζήτηση

Δημήτρης Ψούνης

www.psounis.gr

ΠΕΡΙΕΧΟΜΕΝΑ

Α. Σκοπός του Μαθήματος

Επίπεδο Α
- Συναρτήσεις Κόμβων
- Ο άπληστος αλγόριθμος ευρετικής αναζήτησης
- Ο αλγόριθμος UCS
- Ο αλγόριθμος A*

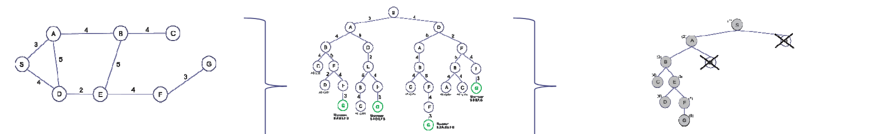
Επίπεδο Β
- Ευρετικές Συναρτήσεις για συγκεκριμένα προβλήματα

Επίπεδο Γ
- (-)

Β. Θεωρία

1. Συναρτήσεις Κόμβων

- Στα προηγούμενα:
  - Ορίσαμε τον **χώρο καταστάσεων** που περιέχει όλους τους συνδυασμούς καταστάσεων-μεταβάσεων
  - Ορίσαμε τον **χώρο αναζήτησης** να περιέχει όλα τα μονοπάτια από την αφετηρία προς τον κόμβο-στόχο
  - Ορίσαμε το **δένδρο αναζήτησης** ως το μέρος του χώρου αναζήτησης που παράγεται από εναν αλγόριθμο αναζήτησης



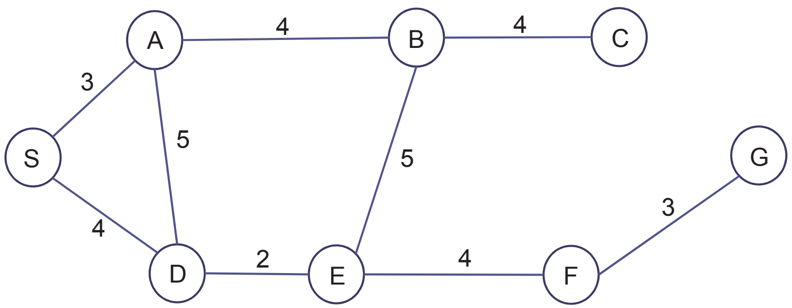
ΧΩΡΟΣ ΚΑΤΑΣΤΑΣΕΩΝ — ΧΩΡΟΣ ΑΝΑΖΗΤΗΣΗΣ — ΔΕΝΔΡΟ ΑΝΑΖΗΤΗΣΗΣ ΤΗΣ ΚΑΤΑ ΒΑΘΟΣ

  - Μελετήσαμε τους αλγόριθμους **τυφλής αναζήτησης** κατά βάθος και κατά πλάτος που παράγουν ένα μονοπάτι χωρίς να λαμβάνουν υπόψιν τα κόστη των ακμών
  - Οι αλγόριθμοι **ευρετικής αναζήτησης** θα παράγουν ένα μονοπάτι λαμβάνοντας υπόψιν τα κόστη των ακμών.

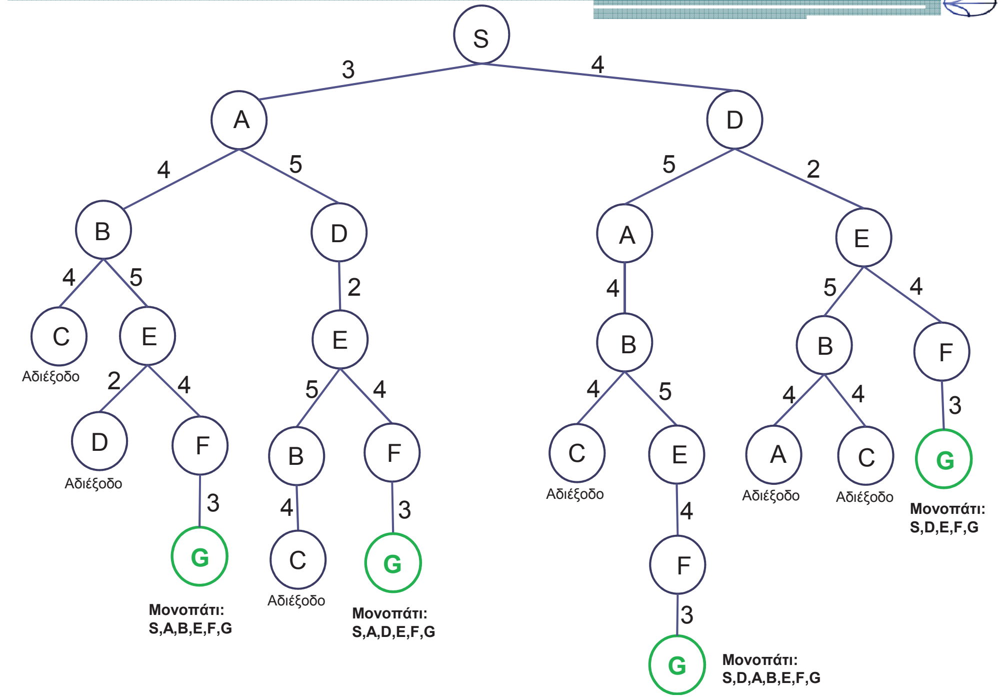
Β. Θεωρία

1. Συναρτήσεις Κόμβων

Το παράδειγμα ενός χώρου καταστάσεων στο οποίο θα εργαστούμε:



- Αρχική κατάσταση είναι η S και τελική κατάσταση είναι η G. Στην επόμενη διαφάνεια φαίνεται ο χώρος αναζήτησης



Β. Θεωρία

1. Συναρτήσεις Κόμβων Δένδρου

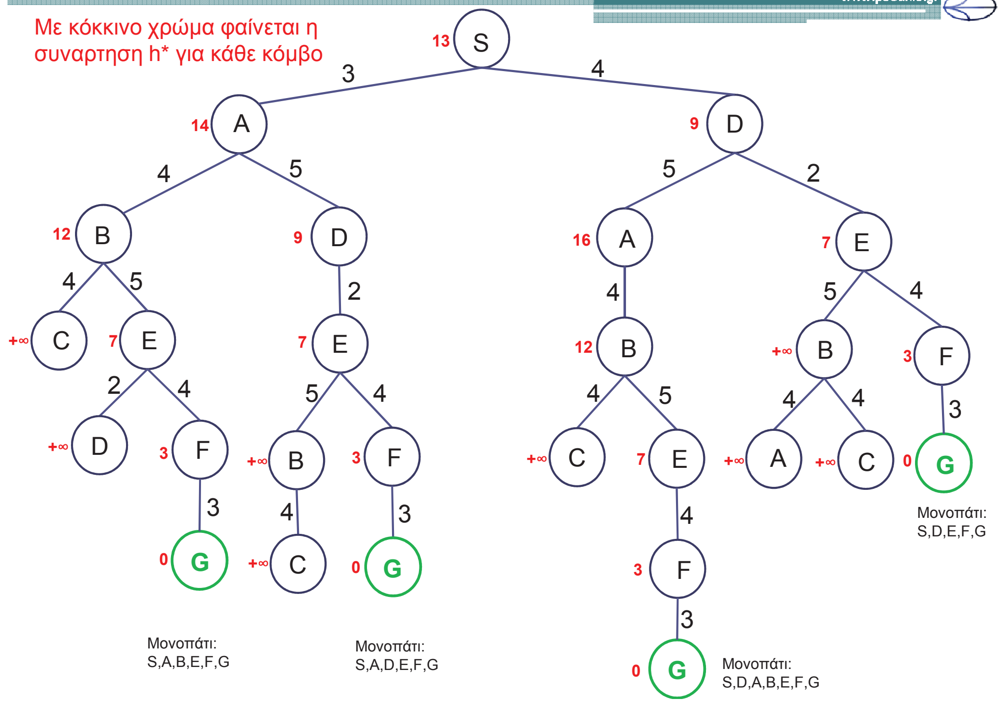
1. Συνάρτηση πραγματικού κόστους h*(v)

Ορίζουμε την <u>συνάρτηση πραγματικού κόστους h*(v)</u> του κόμβου v του χώρου αναζήτησης ως το κόστος (σε άθροισμα βαρών ακμών) του <u>βέλτιστου</u> μονοπατιού από την v προς τον κόμβο-στόχο.

- Αν υπάρχουν περισσότερα από ένα μονοπάτια που οδηγούν από την v στον κόμβο-στόχο, τότε η τιμή της συνάρτησης h*(v) είναι το κόστος του καλύτερου από τα διαθέσιμα μονοπάτια.
- Έπεται άμεσα:
  - Για την τελική κατάσταση T: h*(T)=0
  - Για μία κατάσταση v που δεν οδηγεί στην τελική κατάσταση: h*(v)=+∞

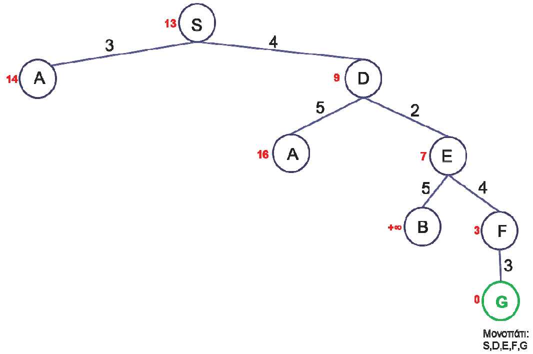
Στην επόμενη διαφάνεια έχει σημειωθεί σε κάθε κόμβο η συνάρτηση πραγματικού κόστους από κάθε κατάσταση προς τον κόμβο στόχο.

Με κόκκινο χρώμα φαίνεται η συναρτηση h* για κάθε κόμβο



Β. Θεωρία
1. Συναρτήσεις Κόμβων Δένδρου
1. Συνάρτηση πραγματικού κόστους h*(v)

- Αν γνωρίζαμε την συνάρτηση πραγματικού κόστους, τότε θα μπορούσαμε να αναπτύξουμε τους κόμβους με βάση την κορυφή που έχει την μικρότερο κόστος.
- Συγκεκριμένα ο αλγόριθμος θα έτρεχε ως εξής:



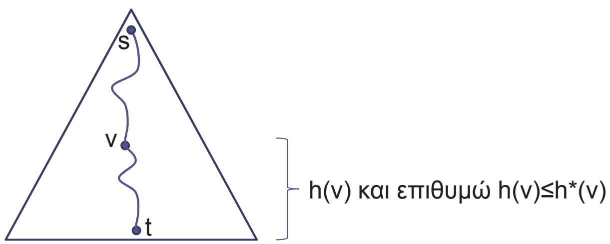
- Δηλαδή θα μπορούσαμε «εύκολα» να υπολογίσουμε την βέλτιστη λύση.

Β. Θεωρία
1. Συναρτήσεις Κόμβων Δένδρου
2. Ευρετική Συνάρτηση h(v)

- Στην πράξη δεν μπορούμε να γνωρίζουμε την συνάρτηση πραγματικού κόστους.
- Για παράδειγμα στο σκάκι αν είχαμε τέτοια συνάρτηση θα μπορούσαμε να ξέρουμε ποια είναι η «καλύτερη» κίνηση σε κάθε κατάσταση της παρτίδας.
- Ορίζουμε την Ευρετική Συνάρτηση h(v) του κόμβου v του χώρου αναζήτησης ως μία εκτίμηση του πόσο απέχει ο κόμβος από έναν κόμβο-στόχο.
  - Η συνάρτηση h(v) «μαντεύει» το κόστος μετάβασης, εκτιμώντας πόσο υποσχόμενη είναι η κατάσταση v για να μας οδηγήσει στον κόμβο-στόχο.
  - Η κατασκευή της γίνεται ενσωματώνοντας ευρετικούς κανόνες που εξαρτώνται από το πρόβλημα.
- Μια ευρετική συνάρτηση λέγεται παραδεκτή αν δεν υπερεκτιμά το πραγματικό κόστος του κόμβου, δηλαδή: $h(v) \leq h^*(v)$ για κάθε κόμβο v του χώρου αναζήτησης.

Β. Θεωρία
1. Συναρτήσεις Κόμβων Δένδρου
2. Ευρετική Συνάρτηση h(v)

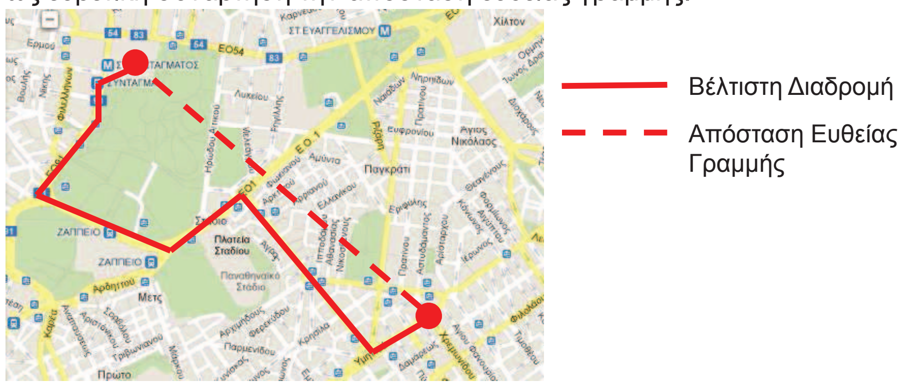
- Η ευρετική συνάρτηση *εκτιμάει* ποια είναι η απόσταση του τρέχοντος κόμβου από τον κόμβο στόχο.



- Επιθυμούμε να έχουμε εκτιμήσεις που δεν υπερβαίνουν το βέλτιστο κόστος, διότι τότε θα δουλεύουν αποδοτικά οι αλγόριθμοί μας.
- Ας δούμε δύο παραδείγματα παραδεκτών ευρετικών συναρτήσεων.

Β. Θεωρία
1. Συναρτήσεις Κόμβων Δένδρου
2. Ευρετική Συνάρτηση h(v): Προβλήματα Χαρτών

- Σε προβλήματα εύρεσης μονοπατιού σε ένα χάρτη,μπορούμε να ορίσουμε ως ευρετική συνάρτηση την απόσταση ευθειας γραμμής.



- Η απόσταση ευθείας γραμμης δεν υπερεκτιμά ποτέ τη βέλτιστη απόσταση, άρα πρόκειται για μια παραδεκτή ευρετική συνάρτηση.

Β. Θεωρία

1. Συναρτήσεις Κόμβων Δένδρου

2. Ευρετική Συνάρτηση h(v): Προβλήματα Λαβυρίνθων

- Σε προβλήματα λαβυρίνθων ορίζεται η απόσταση Manhattan του κόμβου από τον κόμβο-στόχο σύμφωνα με τον τύπο:

$$manhattan\,((x_1,y_1),(x_2,y_2)) = |x_1-x_2|+|y_1-y_2|$$

Y

X	1	2	3	4
1				
2				
3			T	
4				

Κόμβος Στόχος Είναι το τετράγωνο (3,3)

Y

X	1	2	3	4
1	4	3	2	3
2	3			2
3	2		T	1
4	3	2	1	2

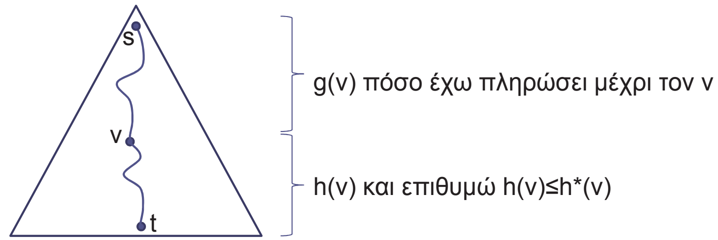
- Η απόσταση Manhattan υπολογίζει το πλήθος μετακινήσεων σαν μην υπάρχουν εμπόδια, άρα δεν υπερεκτιμά ποτέ το πραγματικό κόστος.

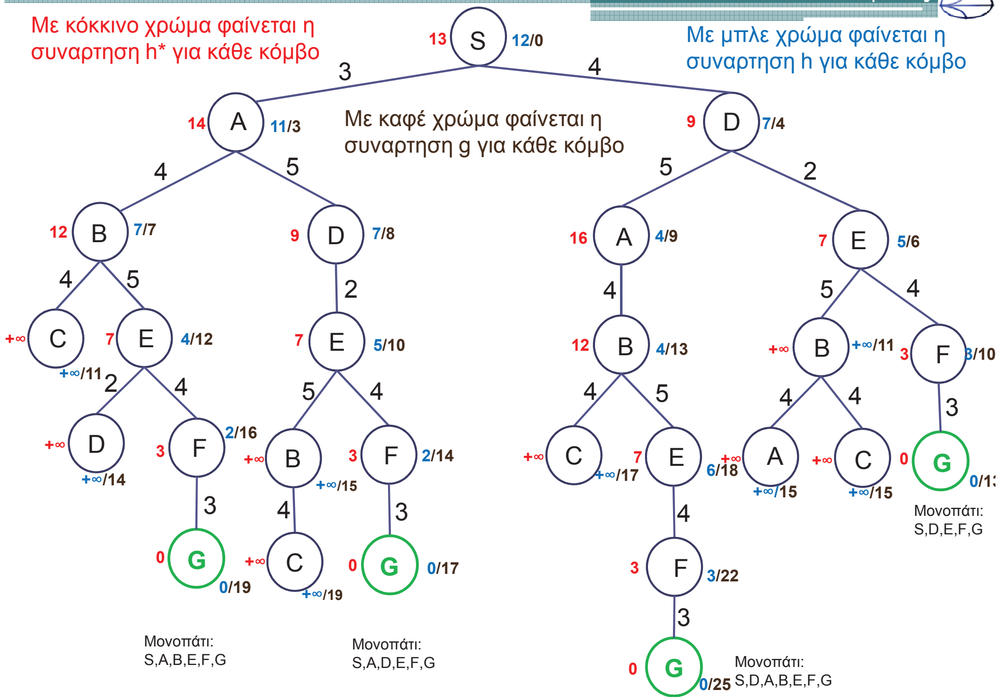
Β. Θεωρία

1. Συναρτήσεις Κόμβων Δένδρου

3. Συνάρτηση κόστους διαδρομής g(v)

- Η <u>Συνάρτηση Κόστους διαδρομής g(v)</u> του κόμβου v του χώρου αναζήτησης είναι το άθροισμα των βαρών των ακμών από την ρίζα μέχρι και τον κόμβο v.
  - Η συνάρτηση αυτή υπολογίζει το κόστος που ήδη έχουμε πληρώσει για να φτάσουμε μέχρι τον κόμβο v.
  - Χρησιμοποιείται από τους αλγόριθμους A* και UCS για τον υπολογισμό του επόμενου κόμβου που θα αναπτυχθεί.





Β. Θεωρία

3. Αλγόριθμοι Ευρετικής Αναζήτησης

1. Ο απληστος αλγόριθμος

- Οι αλγόριθμοι Ευρετικής Αναζήτησης λαμβάνουν υπόψιν τα βάρη των ακμών με στόχο να βρουν ένα μονοπάτι από την αρχική κατάσταση στην τελική.
- Ο 1ος αλγόριθμος ευρετικής αναζήτησης που θα μελετήσουμε είναι ο <u>άπληστος αλγόριθμος</u> ο οποίος σε κάθε βήμα επιλέγει να αναπτύξει εκείνο τον κόμβο που έχει το μικρότερο ευρετικό κόστος, δηλαδή αυτήν που *φαίνεται* ότι θα οδηγήσει πιο γρήγορα στην βέλτιστη λύση.

Β. Θεωρία

3. Αλγόριθμοι Ευρετικής Αναζήτησης

1. Ο άπληστος αλγόριθμος (1.Ψευδογλώσσα)

```
GREEDY(S,T)
ΑΝΟΙΚΤΕΣ=[(S,f(S))]
ΚΛΕΙΣΤΕΣ=[]
Επανέλαβε:
   1.Αφαίρεσε την κατάσταση v της λίστας ΑΝΟΙΚΤΕΣ με την ιδιότητα
     f(v)≤f(u) για κάθε άλλη κατάσταση u της λίστας ΑΝΟΙΚΤΕΣ
   2.Βάλε το v στην λίστα ΚΛΕΙΣΤΕΣ
   3.Δημιουργησε τους διάδοχους της v. Κάθε διάδοχος κρατάει ότι η v
     είναι προκάτοχός της.
   4.Εάν το v είναι ο κόμβος στόχος τερμάτισε με επιστροφή
       μονοπατιού
     Αλλιώς επανέλαβε τα ακόλουθα για κάθε διάδοχο u της κατάστασης v:
     4.1 Υπολόγισε την τιμή f(u)
     4.2 Εάν (η u δεν ανήκει ούτε στις ΑΝΟΙΚΤΕΣ, ούτε στις ΚΛΕΙΣΤΕΣ) τότε
             Πρόσθεσε την τιμή (u,f(u)) στις ανοικτές
         Αλλιώς εάν (η u ήδη ανήκει στις ΑΝΟΙΚΤΕΣ ή ΚΛΕΙΣΤΕΣ) τότε
             Συγκρίνουμε την νέα τιμή της f(u) με την παλαιά τιμή της)
             Εάν (νεα>=παλαιά) τότε αφαιρούμε την u από τους διάδοχους της v
             Αλλιώς Αφαιρούμε το (u,παλαία) από την λίστα που ανήκει
                    Προσθέτουμε το (u,νέα) στην λίστα ΑΝΟΙΚΤΕΣ
Εως ότου η λίστα ΑΝΟΙΚΤΕΣ είναι κενή.
```

Στον ψευδοκωδικα νοείται ως f(x) η συνάρτηση h(x)

Β. Θεωρία

2. Αλγόριθμοι Ευρετικής Αναζήτησης

1. Ο άπληστος αλγόριθμος(1.Ψευδογλώσσα)

Εμπειρικά:

ΑΠΛΗΣΤΟΣ ΑΛΓΟΡΙΘΜΟΣ

Αρχικά:
- Βάζουμε την αφετηρία στο δένδρο με τιμή f(x)=h(x).

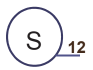
Επαναληπτικά:
- **Πατάμε** *στον ανοιχτό κόμβο με την μικρότερη τιμή.*
- **Ανοίγουμε** τους γείτονες του (που δεν είναι πρόγονοί του) στον γράφο και τους θέτουμε ως παιδιά του με τιμή **f(v)=h(v)**.
- Απο κάθε παιδί κρατάμε την καλύτερη εμφάνιση του στο δένδρο:
  - Αν δεν προϋπήρχε στο δένδρο τότε το αφήνουμε ανοιχτό.
  - Αν προυπήρχε στο δένδρο με μικρότερη ή ίση τιμή, τότε το **διαγράφουμε**.
  - Αν προϋπήρχε στο δένδρο με μεγαλύτερη τιμή, το κρατάμε και διαγράφουμε την προϋπάρχουσα τιμή.

Εως ότου:
- Πατήσουμε στον κόμβο-στόχο

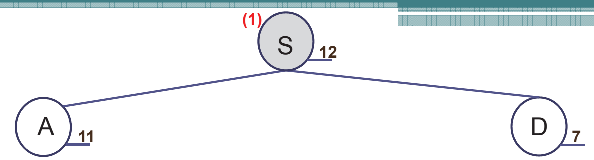
Β. Θεωρία

2. Αλγόριθμοι Ευρετικής Αναζήτησης

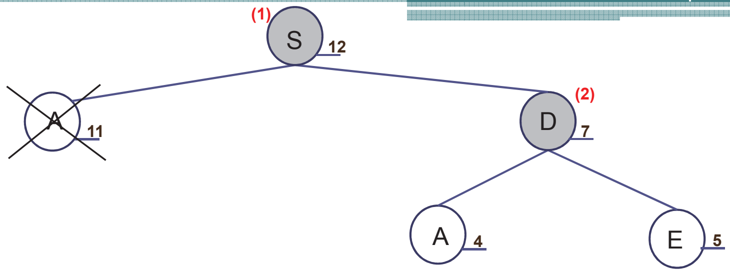
1. Ο άπληστος αλγόριθμος(2.Παράδειγμα Εκτέλεσης)



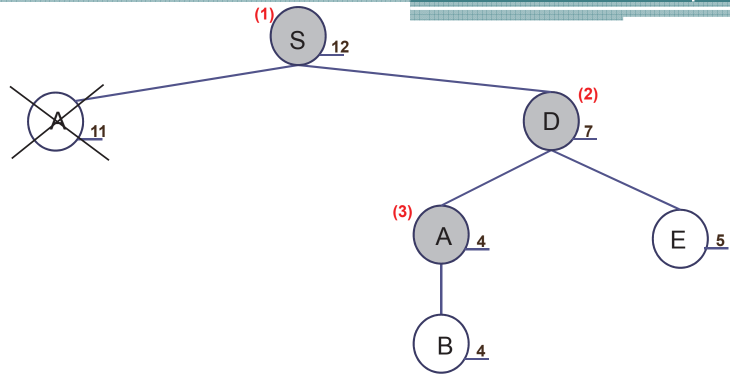
Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,12)}	{}



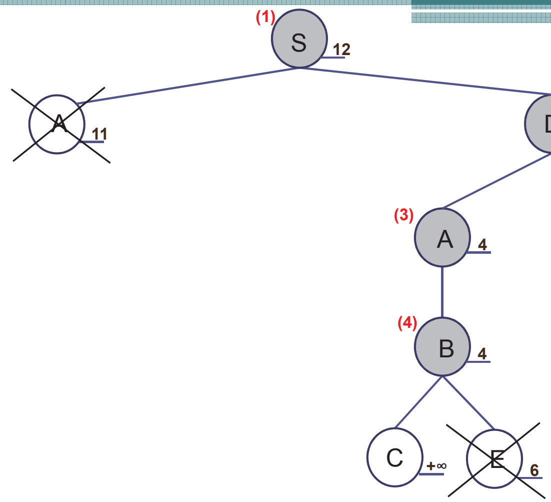
Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,12)}	{}
1	{(A,11),(D,7)}	{(S,12)}



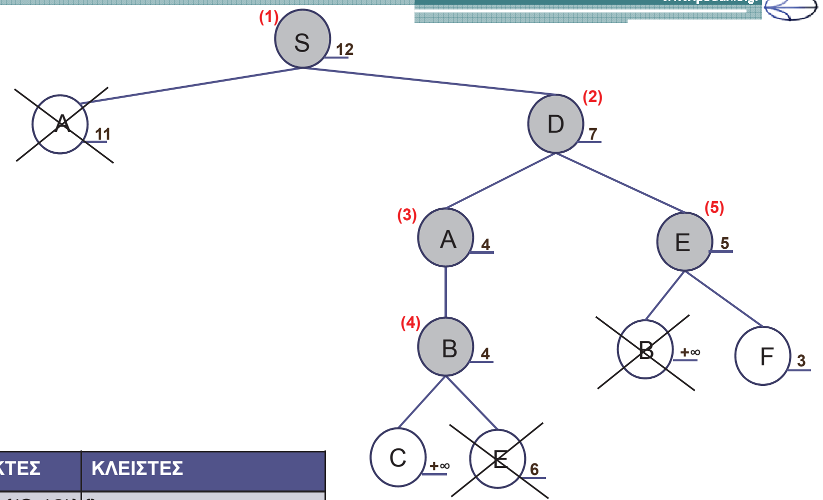
Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,12)}	{}
1	{(A,11),(D,7)}	{(S,12)}
2	{(A,4),(E,5)}	{(S,12),(D,7)}



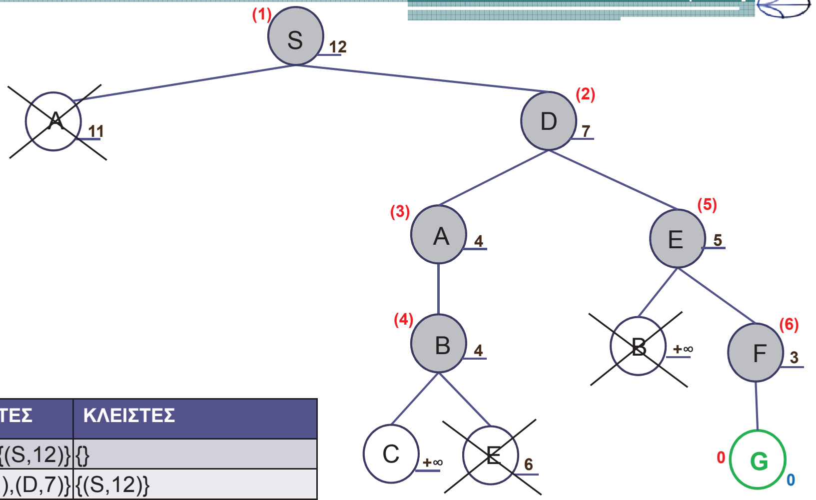
Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,12)}	{}
1	{(A,11),(D,7)}	{(S,12)}
2	{(A,4),(E,5)}	{(S,12),(D,7)}
3	{(B,4),(E,5)}	{(S,12),(D,7),(A,4)}



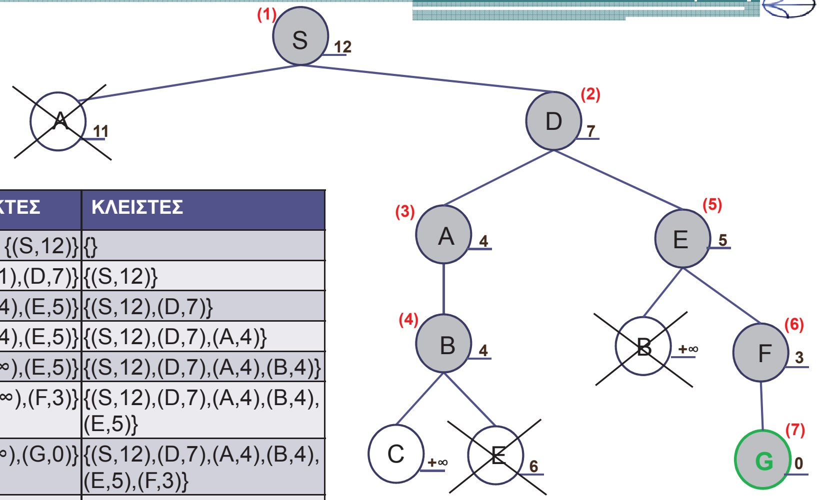
Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,12)}	{}
1	{(A,11),(D,7)}	{(S,12)}
2	{(A,4),(E,5)}	{(S,12),(D,7)}
3	{(B,4),(E,5)}	{(S,12),(D,7),(A,4)}
4	{(C,+∞),(E,5)}	{(S,12),(D,7),(A,4),(B,4)}



Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,12)}	{}
1	{(A,11),(D,7)}	{(S,12)}
2	{(A,4),(E,5)}	{(S,12),(D,7)}
3	{(B,4),(E,5)}	{(S,12),(D,7),(A,4)}
4	{(C,+∞),(E,5)}	{(S,12),(D,7),(A,4),(B,4)}
5	{(C,+∞),(F,3)}	{(S,12),(D,7),(A,4),(B,4),(E,5)}



Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,12)}	{}
1	{(A,11),(D,7)}	{(S,12)}
2	{(A,4),(E,5)}	{(S,12),(D,7)}
3	{(B,4),(E,5)}	{(S,12),(D,7),(A,4)}
4	{(C,+∞),(E,5)}	{(S,12),(D,7),(A,4),(B,4)}
5	{(C,+∞),(F,3)}	{(S,12),(D,7),(A,4),(B,4),(E,5)}
6	{(C,+∞),(G,0)}	{(S,12),(D,7),(A,4),(B,4),(E,5),(F,3)}



Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,12)}	{}
1	{(A,11),(D,7)}	{(S,12)}
2	{(A,4),(E,5)}	{(S,12),(D,7)}
3	{(B,4),(E,5)}	{(S,12),(D,7),(A,4)}
4	{(C,+∞),(E,5)}	{(S,12),(D,7),(A,4),(B,4)}
5	{(C,+∞),(F,3)}	{(S,12),(D,7),(A,4),(B,4),(E,5)}
6	{(C,+∞),(G,0)}	{(S,12),(D,7),(A,4),(B,4),(E,5),(F,3)}
7	{(C,+∞)}	{(S,12),(D,7),(A,4),(B,4),(E,5),(F,3),(G,0)}

<u>Μονοπάτι</u>: S-D-E-F-G
<u>Κόστος Μονοπατιού</u>: 13

<u>Σειρά Επίσκεψης</u>: S-D-A-E-B-F-G
<u>Βήματα</u>: 7

Β. Θεωρία
2. Αλγόριθμοι Ευρετικής Αναζήτησης
1. Άπληστος Αλγόριθμος (3.Παρατηρήσεις)

ΠΑΡΑΤΗΡΗΣΗ 1: Σειρά Τοποθέτησης των Παιδιών
- Υιοθετούμε την εξής σύμβαση: Όταν εισάγουμε τα παιδιά ενός κόμβου η σειρά με την οποία τα απτυπώνουμε στο σχήμα είναι <u>λεξικογραφική</u> (αλφαβητική)
- Δεν έχει μεγάλη σημασία πάντως, γιατί ο κόμβος που επιλέγεται είναι αυτός με την μικρότερη αριθμητική τιμή.

ΠΑΡΑΤΗΡΗΣΗ 2: Κριτήριο Τερματισμού
- Στα βιβλία ΕΑΠ προτείνονται και τα δύο κριτήρια τερματισμού:
  - Να πατήσουμε στον κόμβο-στόχο
  - Να εμφανιστεί ο κόμβος στόχος στους απογόνους του τρέχοντος κόμβου

ΠΑΡΑΤΗΡΗΣΗ 3: Ισοπαλία στην επιλογή κόμβου για ανάπτυξη
- Προτείνονται συνήθως δύο κριτήρια:
  - Επιλέγεται ό κόμβος που είναι σε υψηλότερο επίπεδο και αν είναι στο ίδιο επίπεδο, τότε επιλέγεται ο κόμβος που είναι πιο αριστερά.
  - Λεξικογραφική επιλογή

Β. Θεωρία
2. Αλγόριθμοι Ευρετικής Αναζήτησης
1. Άπληστος Αλγόριθμος (4.Χαρακτηριστικά)

Τα χαρακτηριστικά του άπληστου αλγορίθμου είναι τα ακόλουθα:

ΠΛΗΡΟΤΗΤΑ
- Ναι, εγγυάται την εύρεση μονοπατιού εφόσον αυτό υπάρχει.

ΒΕΛΤΙΣΤΟΤΗΤΑ
- Οχι. Δεν βρίσκει την βέλτιστη λύση

ΧΡΟΝΙΚΗ ΠΟΛΥΠΛΟΚΟΤΗΤΑ
- Εκθετική: $O(b^d)$

ΧΩΡΙΚΗ ΠΟΛΥΠΛΟΚΟΤΗΤΑ
- Εκθετική: $O(b^d)$

Β. Θεωρία
2. Αλγόριθμοι Ευρετικής Αναζήτησης
2. Ο αλγόριθμος UCS

- Ο 2ος αλγόριθμος ευρετικής αναζήτησης που θα μελετήσουμε είναι ο UCS (Uniform Cost Search) ο οποίος σε κάθε βήμα επιλέγει να αναπτύξει εκείνο τον κόμβο που έχει το μικρότερο κόστος διαδρομής, δηλαδή αυτήν που είναι πιο κοντά στην ρίζα.
- Η UCS βρίσκει πάντα την βέλτιστη λύση γιατι στην ουσία κατασκευάζει διαδοχικά όλα τα μονοπάτια από την αφετηρία με αύξουσα σειρά κόστους (λειτουργεί όπως ο αλγόριθμος του Dijkstra)

Β. Θεωρία
2. Αλγόριθμοι Ευρετικής Αναζήτησης
2. Ο αλγόριθμος UCS (1.Ψευδογλώσσα)

```
UCS(S,T)
ΑΝΟΙΚΤΕΣ=[(S,f(S))]
ΚΛΕΙΣΤΕΣ=[]
Επανέλαβε:
   1.Αφαίρεσε την κατάσταση v της λίστας ΑΝΟΙΚΤΕΣ με την ιδιότητα
     f(v)≤f(u) για κάθε άλλη κατάσταση u της λίστας ΑΝΟΙΚΤΕΣ
   2.Βάλε το v στην λίστα ΚΛΕΙΣΤΕΣ
   3.Δημιουργησε τους διαδόχους της v. Κάθε διάδοχος κρατάει ότι η v
     είναι προκάτοχός της.
   4.Εάν το v είναι ο κόμβος στόχος τερμάτισε με επιστροφή
       μονοπατιού
     Αλλιώς επανέλαβε τα ακόλουθα για κάθε διάδοχο u της κατάστασης v:
     4.1 Υπολόγισε την τιμή f(u)
     4.2 Εάν (η u δεν ανήκει ούτε στις ΑΝΟΙΚΤΕΣ, ούτε στις ΚΛΕΙΣΤΕΣ) τότε
             Πρόσθεσε την τιμή (u,f(u)) στις ανοικτές
         Αλλιώς εάν (η u ήδη ανήκει στις ΑΝΟΙΚΤΕΣ ή ΚΛΕΙΣΤΕΣ) τότε
             Συγκρίνουμε την νέα τιμή της f(u) με την παλαιά τιμή της)
             Εάν (νεα>=παλαιά) τότε αφαιρούμε την u από τους διαδόχους της v
             Αλλιώς Αφαιρούμε το (u,παλαία) από την λίστα που ανήκει
                    Προσθέτουμε το (u,νέα) στην λίστα ΑΝΟΙΚΤΕΣ
Εως ότου η λίστα ΑΝΟΙΚΤΕΣ είναι κενή.
```

Στον ψευδοκωδικα νοείται ως f(x) η συνάρτηση g(x)

Β. Θεωρία
2. Αλγόριθμοι Ευρετικής Αναζήτησης
2. Ο αλγόριθμος UCS (1.Ψευδογλώσσα)

Εμπειρικά:

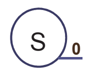
ΑΛΓΟΡΙΘΜΟΣ UCS

Αρχικά:
- Βάζουμε την αφετηρία στο δένδρο με τιμή f(x)=g(x).

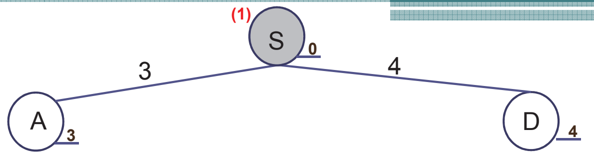
Επαναληπτικά:
- **Πατάμε** *στον ανοιχτό κόμβο με την μικρότερη τιμή.*
- **Ανοίγουμε** τους γείτονες του (που δεν είναι πρόγονοί του) στον γράφο και τους θέτουμε ως παιδιά του με τιμή **f(v)=g(v)**.
- Απο κάθε παιδί κρατάμε την καλύτερη εμφάνιση του στο δένδρο:
  - Αν δεν προϋπήρχε στο δένδρο τότε το αφήνουμε ανοιχτό.
  - Αν προυπήρχε στο δένδρο με μικρότερη ή ίση τιμή, τότε το **διαγράφουμε**.
  - Αν προϋπήρχε στο δένδρο με μεγαλύτερη τιμή, το κρατάμε και διαγράφουμε την προϋπάρχουσα τιμή.

Εως ότου:
- Πατήσουμε στον κόμβο-στόχο

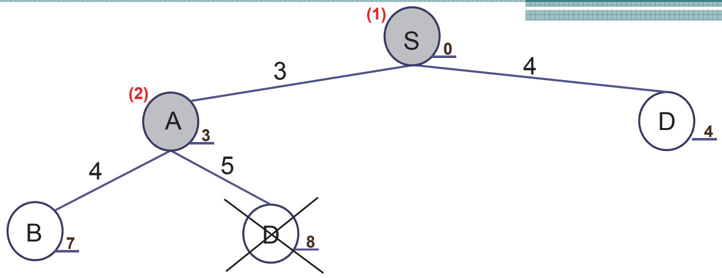
Β. Θεωρία
2. Αλγόριθμοι Ευρετικής Αναζήτησης
2. Ο αλγόριθμος UCS (2.Παράδειγμα Εκτέλεσης)



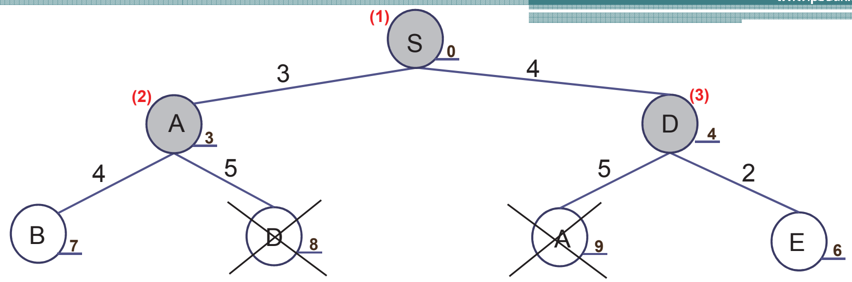
Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,0)}	{}



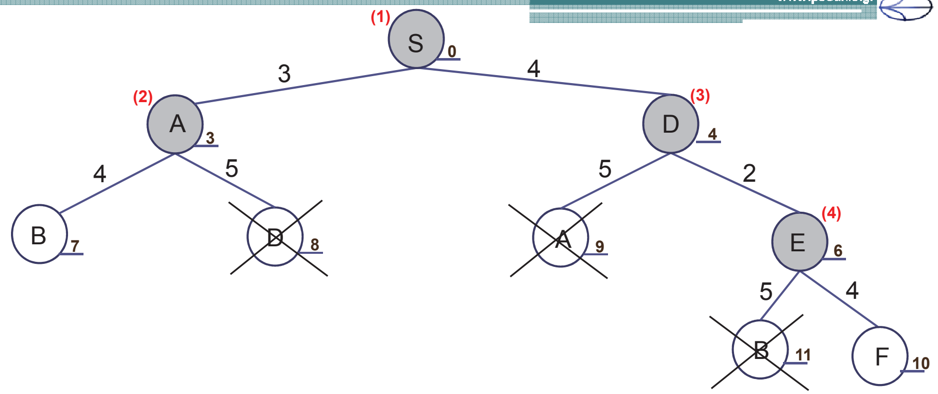
Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,0)}	{}
1	{(A,3),(D,4)}	{(S,0)}



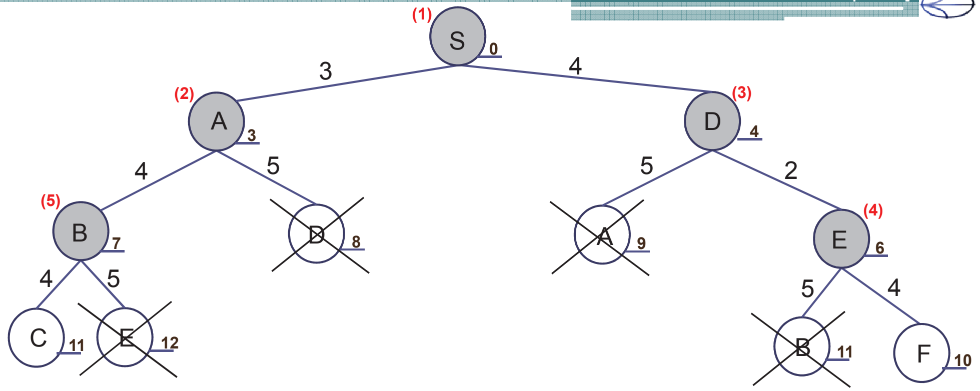
Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,0)}	{}
1	{(A,3),(D,4)}	{(S,0)}
2	{(D,4),(B,7)}	{(S,0),(A,3)}



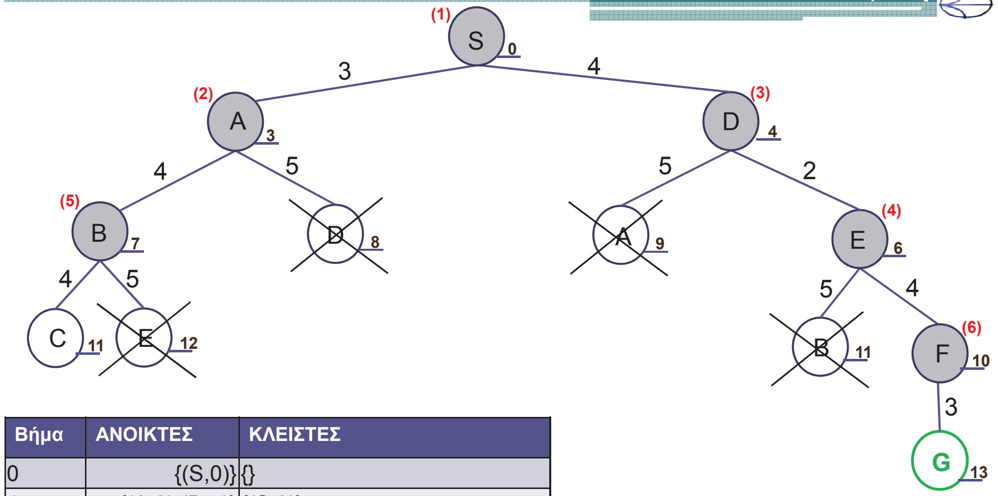
Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,0)}	{}
1	{(A,3),(D,4)}	{(S,0)}
2	{(D,4),(B,7)}	{(S,0),(A,3)}
3	{(B,7),(E,6)}	{(S,0),(A,3),(D,4)}



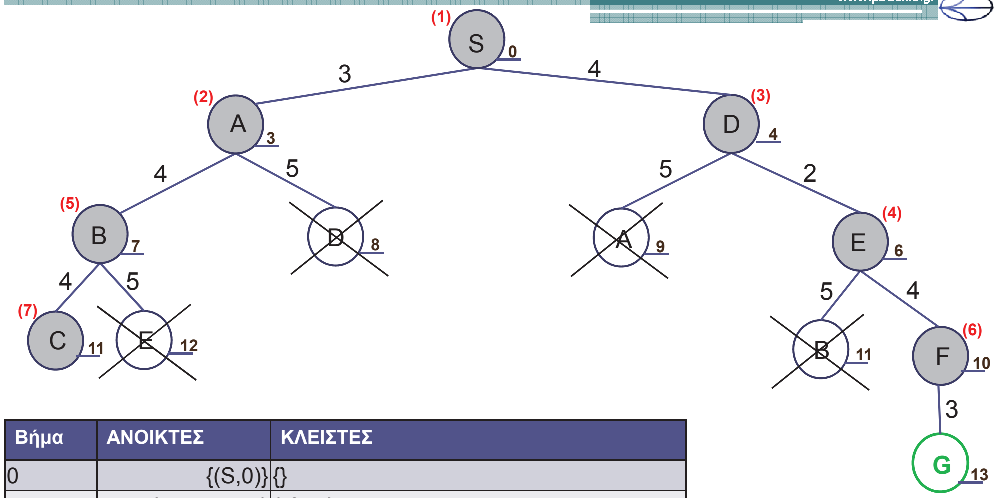
Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,0)}	{}
1	{(A,3),(D,4)}	{(S,0)}
2	{(D,4),(B,7)}	{(S,0),(A,3)}
3	{(B,7),(E,6)}	{(S,0),(A,3),(D,4)}
4	{(B,7),(F,10)}	{(S,0),(A,3),(D,4),(E,4)}



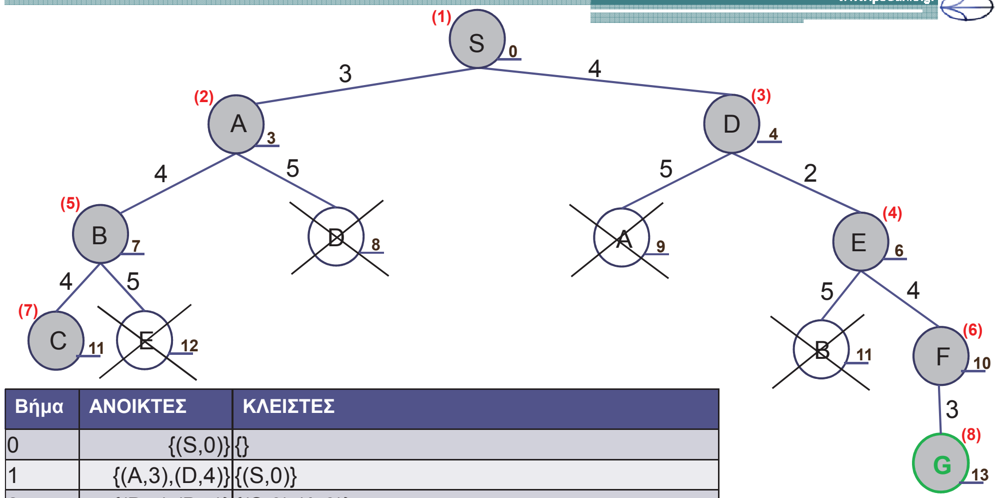
Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,0)}	{}
1	{(A,3),(D,4)}	{(S,0)}
2	{(D,4),(B,7)}	{(S,0),(A,3)}
3	{(B,7),(E,6)}	{(S,0),(A,3),(D,4)}
4	{(B,7),(F,10)}	{(S,0),(A,3),(D,4),(E,4)}
5	{(F,10),(C,11)}	{(S,0),(A,3),(D,4),(E,4),(B,7)}



Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,0)}	{}
1	{(A,3),(D,4)}	{(S,0)}
2	{(D,4),(B,7)}	{(S,0),(A,3)}
3	{(B,7),(E,6)}	{(S,0),(A,3),(D,4)}
4	{(B,7),(F,10)}	{(S,0),(A,3),(D,4),(E,4)}
5	{(F,10),(C,11)}	{(S,0),(A,3),(D,4),(E,4),(B,7)}
6	{(G,13),(C,11)}	{(S,0),(A,3),(D,4),(E,4),(B,7), (F,10)}



Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,0)}	{}
1	{(A,3),(D,4)}	{(S,0)}
2	{(D,4),(B,7)}	{(S,0),(A,3)}
3	{(B,7),(E,6)}	{(S,0),(A,3),(D,4)}
4	{(B,7),(F,10)}	{(S,0),(A,3),(D,4),(E,4)}
5	{(F,10),(C,11)}	{(S,0),(A,3),(D,4),(E,4),(B,7)}
6	{(G,13),(C,11)}	{(S,0),(A,3),(D,4),(E,4),(B,7),(F,10)}
7	{(G,13)}	{(S,0),(A,3),(D,4),(E,4),(B,7),(F,10),(C,11)}



Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,0)}	{}
1	{(A,3),(D,4)}	{(S,0)}
2	{(D,4),(B,7)}	{(S,0),(A,3)}
3	{(B,7),(E,6)}	{(S,0),(A,3),(D,4)}
4	{(B,7),(F,10)}	{(S,0),(A,3),(D,4),(E,4)}
5	{(F,10),(C,11)}	{(S,0),(A,3),(D,4),(E,4),(B,7)}
6	{(G,13),(C,11)}	{(S,0),(A,3),(D,4),(E,4),(B,7),(F,10)}
7	{(G,13)}	{(S,0),(A,3),(D,4),(E,4),(B,7),(F,10),(C,11)}
8	{}	{(S,0),(A,3),(D,4),(E,4),(B,7),(F,10),(C,11),(G,13)}

<u>Μονοπάτι</u>: S-D-E-F-G
<u>Κόστος Μονοπατιού</u>: 13

<u>Σειρά Επίσκεψης</u>:
S-A-D-E-B-F-C-G
<u>Βήματα</u>: 8

Β. Θεωρία
2. Αλγόριθμοι Ευρετικής Αναζήτησης
2. Αλγόριθμος UCS (3.Παρατηρήσεις)

ΠΑΡΑΤΗΡΗΣΗ 1: Σειρά Τοποθέτησης των Παιδιών
- Υιοθετούμε την εξής σύμβαση: Όταν εισάγουμε τα παιδιά ενός κόμβου η σειρά με την οποία τα απτυπώνουμε στο σχήμα είναι λεξικογραφική (αλφαβητική)
- Δεν έχει μεγάλη σημασία πάντως, γιατί ο κόμβος που επιλέγεται είναι αυτός με την μικρότερη αριθμητική τιμή.

ΠΑΡΑΤΗΡΗΣΗ 2: Κριτήριο Τερματισμού
- Στα βιβλία ΕΑΠ προτείνονται και τα δύο κριτήρια τερματισμού
  - Να πατήσουμε στον κόμβο-στόχο
  - Να εμφανιστεί ο κόμβος στόχος στους απογόνους του τρέχοντος κόμβου

ΠΑΡΑΤΗΡΗΣΗ 3: Ισοπαλία στην επιλογή κόμβου για ανάπτυξη
- Προτείνονται συνήθως δύο κριτήρια:
  - Επιλέγεται ό κόμβος που είναι σε υψηλότερο επίπεδο και αν είναι στο ίδιο επίπεδο, τότε επιλέγεται ο κόμβος που είναι πιο αριστερά.
  - Λεξικογραφική επιλογή

Β. Θεωρία
2. Αλγόριθμοι Ευρετικής Αναζήτησης
2. Αλγόριθμος UCS (4.Χαρακτηριστικά)

Τα χαρακτηριστικά του άπληστου αλγορίθμου είναι τα ακόλουθα:

ΠΛΗΡΟΤΗΤΑ
- Ναι, εγγυάται την εύρεση μονοπατιού εφόσον αυτό υπάρχει.

ΒΕΛΤΙΣΤΟΤΗΤΑ
- Ναι. Εγγυάται την εύρεση της βέλτιστης λύσης

ΧΡΟΝΙΚΗ ΠΟΛΥΠΛΟΚΟΤΗΤΑ
- Εκθετική: $O(b^d)$

ΧΩΡΙΚΗ ΠΟΛΥΠΛΟΚΟΤΗΤΑ
- Εκθετική: $O(b^d)$

Β. Θεωρία
2. Αλγόριθμοι Ευρετικής Αναζήτησης
3. Ο αλγόριθμος Α*

- Ο 3ος αλγόριθμος ευρετικής αναζήτησης που θα μελετήσουμε είναι ο Α*, ο οποίος σε κάθε βήμα επιλέγει να αναπτύξει εκείνο τον κόμβο που έχει τη μικρότερη τιμή στο άθροισμα των τιμών της ευρετικής συνάρτησης h και της συνάρτησης κόστους g, δηλαδή αυτήν που είναι πιο κοντά στην λύση.
- Ο Α* βρίσκει πάντα την βέλτιστη λύση, εφόσον η ευρετική συνάρτηση που χρησιμοποιεί είναι παραδεκτή.

Β. Θεωρία
2. Αλγόριθμοι Ευρετικής Αναζήτησης
3. Ο αλγόριθμος Α* (1.Ψευδογλώσσα)

```
A*(S,T)
ΑΝΟΙΚΤΕΣ=[(S,f(S))]
ΚΛΕΙΣΤΕΣ=[]
Επανέλαβε:
   1.Αφαίρεσε την κατάσταση v της λίστας ΑΝΟΙΚΤΕΣ με την ιδιότητα
     f(v)≤f(u) για κάθε άλλη κατάσταση u της λίστας ΑΝΟΙΚΤΕΣ
   2.Βάλε το v στην λίστα ΚΛΕΙΣΤΕΣ
   3.Δημιουργησε τους διαδόχους της v. Κάθε διάδοχος κρατάει ότι η v
     είναι προκάτοχός της.
   4.Εάν το v είναι ο κόμβος στόχος τερμάτισε με επιστροφή
       μονοπατιού
     Αλλιώς επανέλαβε τα ακόλουθα για κάθε διάδοχο u της κατάστασης v:
     4.1 Υπολόγισε την τιμή f(u)
     4.2 Εάν (η u δεν ανήκει ούτε στις ΑΝΟΙΚΤΕΣ, ούτε στις ΚΛΕΙΣΤΕΣ) τότε
             Πρόσθεσε την τιμή (u,f(u)) στις ανοικτές
         Αλλιώς εάν (η u ήδη ανήκει στις ΑΝΟΙΚΤΕΣ ή ΚΛΕΙΣΤΕΣ) τότε
             Συγκρίνουμε την νέα τιμή της f(u) με την παλαιά τιμή της)
             Εάν (νεα>=παλαιά) τότε αφαιρούμε την u από τους διαδόχους της v
             Αλλιώς Αφαιρούμε το (u,παλαία) από την λίστα που ανήκει
                    Προσθέτουμε το (u,νέα) στην λίστα ΑΝΟΙΚΤΕΣ
Εως ότου η λίστα ΑΝΟΙΚΤΕΣ είναι κενή.
```

Στον ψευδοκωδικα νοείται ως f(x) η συνάρτηση g(x)+h(x)

Β. Θεωρία

2. Αλγόριθμοι Ευρετικής Αναζήτησης

2. Ο αλγόριθμος Α* (1.Ψευδογλώσσα)

Εμπειρικά:

<u>ΑΛΓΟΡΙΘΜΟΣ Α*</u>

<u>Αρχικά:</u>

- Βάζουμε την αφετηρία στο δένδρο με τιμή f(x)=g(x)+h(x).

<u>Επαναληπτικά:</u>

- **<u>Πατάμε</u>** *στον ανοιχτό κόμβο με την μικρότερη τιμή.*
- **<u>Ανοίγουμε</u>** τους γείτονες του (που δεν είναι πρόγονοί του) στον γράφο και τους θέτουμε ως παιδιά του με τιμή **f(v)=g(v)+h(v)**.
- Απο κάθε παιδί κρατάμε την καλύτερη εμφάνιση του στο δένδρο:
  - Αν δεν προϋπήρχε στο δένδρο τότε το αφήνουμε ανοιχτό.
  - Αν προυπήρχε στο δένδρο με μικρότερη ή ίση τιμή, τότε το **<u>διαγράφουμε</u>**.
  - Αν προϋπήρχε στο δένδρο με μεγαλύτερη τιμή, το κρατάμε και διαγράφουμε την προϋπάρχουσα τιμή.

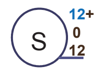
<u>Εως ότου:</u>

- Πατήσουμε στον κόμβο-στόχο

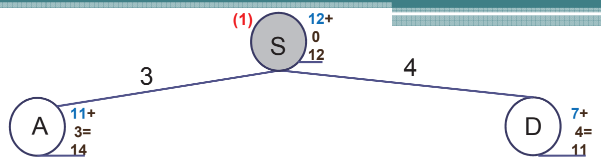
Β. Θεωρία

2. Αλγόριθμοι Ευρετικής Αναζήτησης

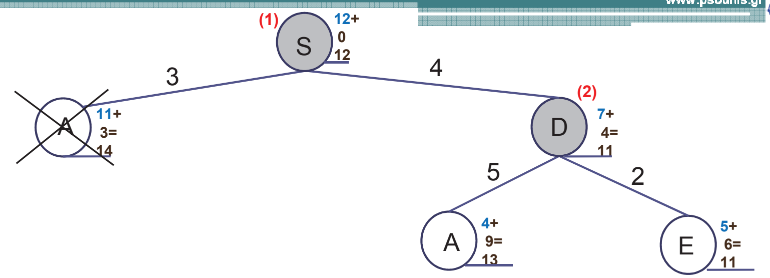
3. Ο αλγόριθμος Α* (2.Παράδειγμα Εκτέλεσης)



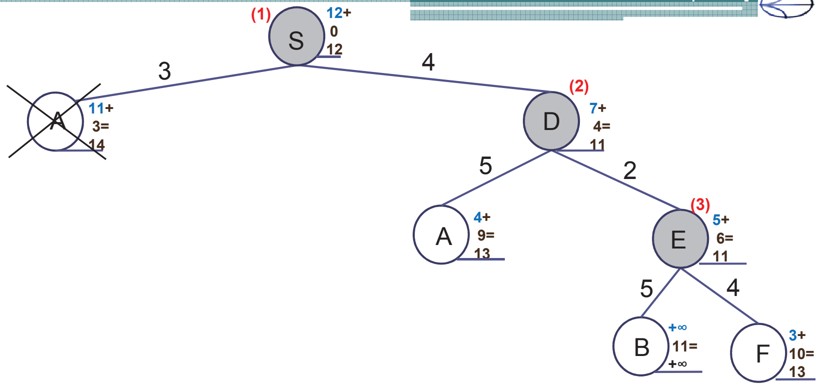
Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,12)}	{}



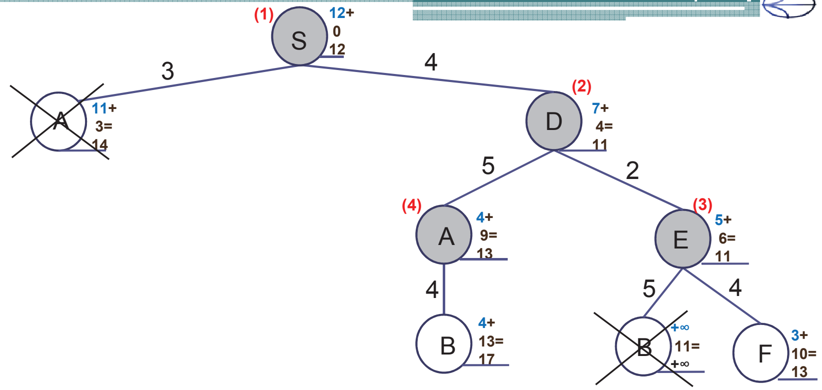
Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,12)}	{}
1	{(A,14),(D,11)}	{(S,12)}



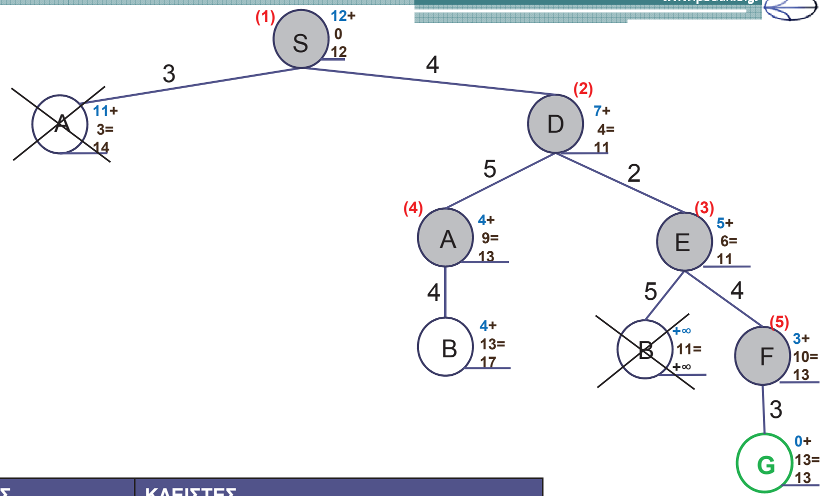
Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,12)}	{}
1	{(A,14),(D,11)}	{(S,12)}
2	{(A,13),(E,11)}	{(S,12),(D,11)}



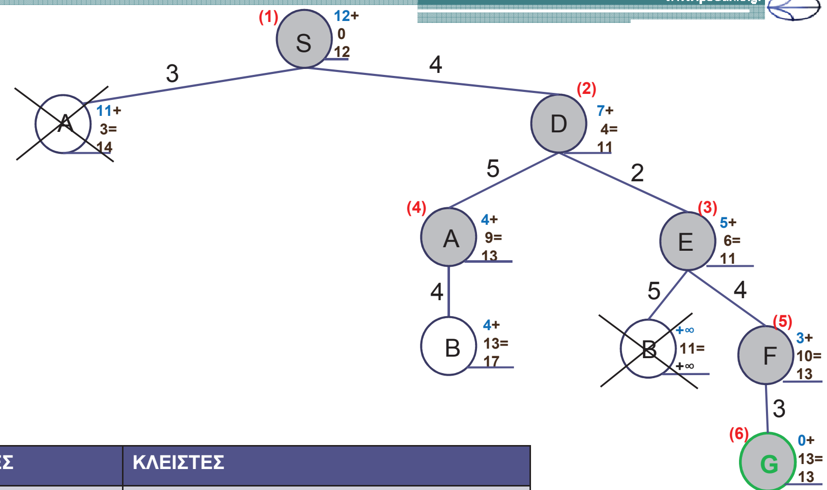
Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,12)}	{}
1	{(A,14),(D,11)}	{(S,12)}
2	{(A,13),(E,11)}	{(S,12),(D,11)}
3	{(A,13),(B,+∞)(F,13)}	{(S,12),(D,11),(E,11)}



Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,12)}	{}
1	{(A,14),(D,11)}	{(S,12)}
2	{(A,13),(E,11)}	{(S,12),(D,11)}
3	{(A,13),(B,+∞)(F,13)}	{(S,12),(D,11),(E,11)}
4	{(B,17),(F,13)}	{(S,12),(D,11),(E,11),(A,13)}



Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,12)}	{}
1	{(A,14),(D,11)}	{(S,12)}
2	{(A,13),(E,11)}	{(S,12),(D,11)}
3	{(A,13),(B,+∞)(F,13)}	{(S,12),(D,11),(E,11)}
4	{(B,17),(F,13)}	{(S,12),(D,11),(E,11),(A,13)}
5	{(B,17),(G,13)}	{(S,12),(D,11),(E,11),(A,13),(F,13)}



Βήμα	ΑΝΟΙΚΤΕΣ	ΚΛΕΙΣΤΕΣ
0	{(S,12)}	{}
1	{(A,14),(D,11)}	{(S,12)}
2	{(A,13),(E,11)}	{(S,12),(D,11)}
3	{(A,13),(B,+∞)(F,13)}	{(S,12),(D,11),(E,11)}
4	{(B,17),(F,13)}	{(S,12),(D,11),(E,11),(A,13)}
5	{(B,17),(G,13)}	{(S,12),(D,11),(E,11),(A,13),(F,13)}
6	{(B,17)}	{(S,12),(D,11),(E,11),(A,13),(F,13),(G,13)}

Μονοπάτι: S-D-E-F-G
Κόστος Μονοπατιού: 13

Σειρά Επίσκεψης:
S-D-A-E-F-G
Βήματα: 6

Β. Θεωρία
2. Αλγόριθμοι Ευρετικής Αναζήτησης
3. Αλγόριθμος Α*(3.Παρατηρήσεις)

ΠΑΡΑΤΗΡΗΣΗ 1: Σειρά Τοποθέτησης των Παιδιών
- Υιοθετούμε την εξής σύμβαση: Όταν εισάγουμε τα παιδιά ενός κόμβου η σειρά με την οποία τα απτυπώνουμε στο σχήμα είναι λεξικογραφική (αλφαβητική)
- Δεν έχει μεγάλη σημασία πάντως, γιατί ο κόμβος που επιλέγεται είναι αυτός με την μικρότερη αριθμητική τιμή.

ΠΑΡΑΤΗΡΗΣΗ 2: Κριτήριο Τερματισμού
- Έχει διαλευκανθεί ότι το κριτήριο του τερματισμού είναι πάντα:
  - Να πατήσουμε στον κόμβο-στόχο!

ΠΑΡΑΤΗΡΗΣΗ 3: Ισοπαλία στην επιλογή κόμβου για ανάπτυξη
- Προτείνονται συνήθως δύο κριτήρια:
  - Επιλέγεται ό κόμβος που είναι σε υψηλότερο επίπεδο και αν είναι στο ίδιο επίπεδο, τότε επιλέγεται ο κόμβος που είναι πιο αριστερά.
  - Λεξικογραφική επιλογή

Β. Θεωρία
2. Αλγόριθμοι Ευρετικής Αναζήτησης
3. Αλγόριθμος Α* (4.Χαρακτηριστικά)

Τα χαρακτηριστικά του άπληστου αλγορίθμου είναι τα ακόλουθα:

ΠΛΗΡΟΤΗΤΑ
- Ναι, εγγυάται την εύρεση μονοπατιού εφόσον αυτό υπάρχει.

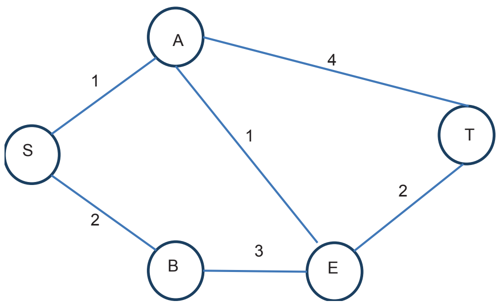
ΒΕΛΤΙΣΤΟΤΗΤΑ
- Ναι αν η ευρετική συνάρτηση είναι παραδεκτή.

ΧΡΟΝΙΚΗ ΠΟΛΥΠΛΟΚΟΤΗΤΑ
- Εκθετική: $O(b^d)$

ΧΩΡΙΚΗ ΠΟΛΥΠΛΟΚΟΤΗΤΑ
- Εκθετική: $O(b^d)$

Γ. Ασκήσεις
Άσκηση Κατανόησης 1

Δίδεται ο ακόλουθος γράφος καταστασεων με κόμβο αφετηρία τον κόμβο S και κόμβο-στόχο T:



και η ακόλουθη ευρετική συνάρτηση: h(S)=3, h(A)=2, h(B)=1, h(E)=2

Α. Εξετάστε αν η ευρετική συνάρτηση είναι παραδεκτή.

Β. Κατασκευάστε τον χώρο αναζήτησης

Γ. Δώστε την εκτέλεση της κατά βάθος με αφετηρία το S και προορισμό το T

Δ. Δώστε την εκτέλεση της κατά πλάτος με αφετηρία το S και προορισμό το T

Ε. Δώστε την εκτέλεση του Άπληστου Αλγορίθμου με αφετηρία το S και προορισμό το T

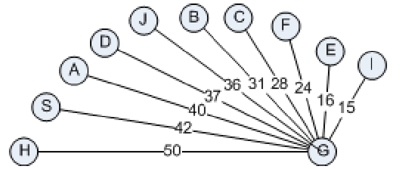
Ζ. Δώστε την εκτέλεση της UCS με αφετηρία το S και προορισμό το T

Η. Δώστε την εκτέλεση του A* με αφετηρία το S και προορισμό το T

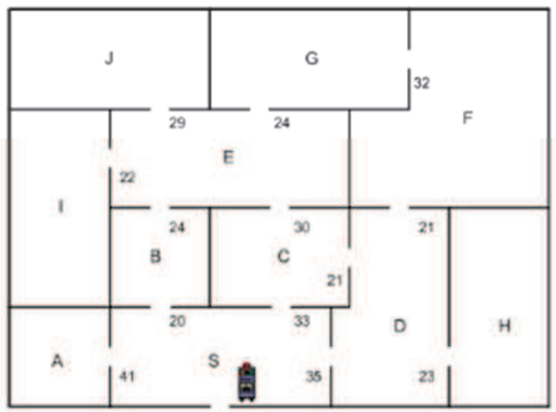
Γ. Ασκήσεις
Εφαρμογή 1

Το ρομπότ Robbie για να αποφασίσει ποια διαδρομή θα ακολουθήσει κατά την πλοήγηση του στο χώρο που παρουσιάζεται στο σχήμα λαμβάνει υπόψη την κατανάλωση ενέργειας της μπαταρίας του. Στο σχήμα 1 φαίνεται ο χώρος πλοήγησης του Robbie. Κάθε δωμάτιο χαρακτηρίζεται από το όνομά του και περιέχει θέσεις πρόσβασης σε γειτονικό του τετράγωνο. Κάθε θέση πρόσβασης χαρακτηρίζεται από έναν αριθμό που περιγράφει το ποσό ενέργειας πραγματικής κατανάλωσης της μπαταρίας του Robbie.

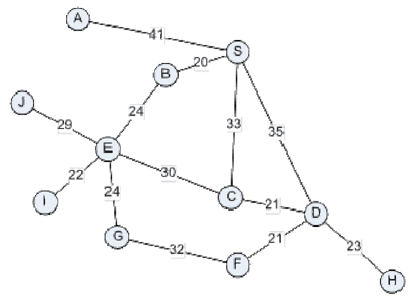
Ο γράφος καταστάσεων που σχεδιάσαμε στο Μάθημα 1.1 είναι στο σχήμα 3, ενώ στο σχήμα 2 φαίνεται η ευρετική συνάρτηση για την εκτιμώμενη κατανάλωση ενέργειας του Robbie για να κάνει την μετάβαση από κάθε κόμβο ως τον κόμβο-στόχο G.



Σχήμα 2 Εκτιμώμενη κατανάλωση ενέργειας του Robbie.



Σχήμα 1 Χώρος πλοήγησης του Robbie.



Σχήμα 3: Ο γράφος καταστάσεων του λαβυρίθου του Robbie:

(A) Εκτελέστε τον άπληστο αλγόριθμο με αφετηρία το S και προορισμό το G

(B) Εκτελέστε τον αλγόριθμο UCS με αφετηρία το S και προορισμό το G

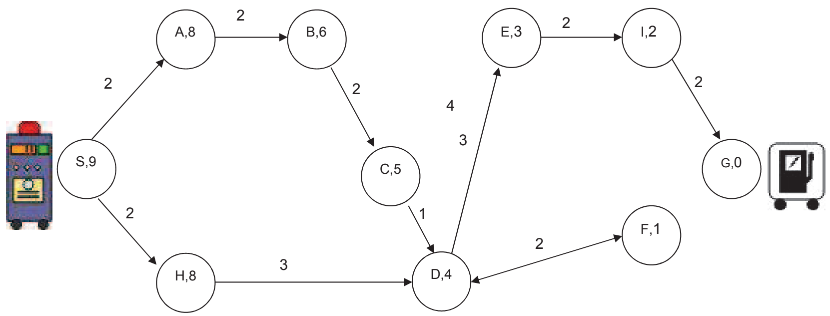
(Γ) Εκτελέστε τον αλγόριθμο A* με αφετηρία το S και προορισμό το G

Γ. Ασκήσεις
Εφαρμογή 2

Ο Robbie, το ρομπότ του παρακάτω σχήματος-χάρτη, κατά τη διάρκεια των εργασιών που κάνει διαπιστώνει ότι πρέπει να γυρίσει όσο το δυνατόν πιο γρήγορα, από την τρέχουσα θέση S, στην κινητή βάση ανεφοδιασμού, στη θέση G, προκειμένου να φορτίσει τις μπαταρίες του. Ο Robbie μπορεί να πλοηγηθεί σύμφωνα με τα βέλη του χάρτη. Πάνω σε κάθε βέλος υπάρχει ένας αριθμός που αντιπροσωπεύει την απόσταση μεταξύ των συνδεόμενων κόμβων. Κάθε κόμβος χαρακτηρίζεται από ένα γράμμα. Δίπλα στο γράμμα κάθε κόμβου βρίσκεται ένας αριθμός που αντιπροσωπεύει την ευθεία απόσταση του κόμβου από τον κόμβο G. Αγνοείστε προς το παρόν το διακεκομμένο βέλος.

Ο Robbie διαθέτει μνήμη στην οποία έχουν καταχωρηθεί τα στοιχεία του χάρτη. Ο Robbie αποφασίζει να βρει την καλύτερη διαδρομή χρησιμοποιώντας διάφορους αλγορίθμους, όπως παρακάτω. Κατά την εκτέλεση των αλγορίθμων χρησιμοποιούμε τις εξής συμβάσεις σε περιπτώσεις ισότιμων κόμβων (εκτός αν αναφέρεται διαφορετικά):

- Όταν ισότιμοι κόμβοι βρίσκονται στο ίδιο βάθος (επίπεδο) επιλέγεται ο ευρισκόμενος αριστερότερα.
- Όταν ισότιμοι κόμβοι βρίσκονται σε διαφορετικό βάθος (επίπεδο) επιλέγεται ο ευρισκόμενος σε μικρότερο βάθος (υψηλότερα στο δέντρο).



α. Κατά βάθος αναζήτηση.

Α. Να σχεδιάσετε το (τμήμα από το) δέντρο αναζήτησης που θα παραχθεί κατά την εκτέλεση του αλγορίθμου της κατά βάθος αναζήτησης (depth-first search), αν ο Robbie αποφασίσει να κινείται όλο αριστερά. Καταγράψτε το μονοπάτι της λύσης και υπολογίστε το κόστος του. Καταγράψτε την σειρά επέκτασης των κόμβων και υπολογίστε πόσα βήματα έκανε ο αλγόριθμος.

Β. Να κάνετε το ίδιο, αν ο Robbie αποφασίσει να κινείται όλο δεξιά.

β. Κατά πλάτος αναζήτηση.

Να σχεδιάσετε το (τμήμα από το) δέντρο αναζήτησης που θα παραχθεί κατά την εκτέλεση του αλγορίθμου της κατά πλάτος αναζήτησης (breadth-first search). Καταγράψτε το μονοπάτι της λύσης και υπολογίστε το κόστος του. Καταγράψτε την σειρά επέκτασης των κόμβων και υπολογίστε πόσα βήματα έκανε ο αλγόριθμος.

γ. Άπληστη Αναζήτηση.

Να σχεδιάσετε το (τμήμα από το) δέντρο αναζήτησης που θα παραχθεί κατά την εκτέλεση του αλγορίθμου της άπληστης αναζήτησης (greedy search), η οποία αναπτύσσει κάθε φορά τον κόμβο που απέχει λιγότερο από τη θέση ανεφοδιασμού. Καταγράψτε το μονοπάτι της λύσης και υπολογίστε το κόστος του. Καταγράψτε την σειρά επέκτασης των κόμβων και υπολογίστε πόσα βήματα έκανε ο αλγόριθμος.

δ. Αναζήτηση με βάση το κόστος διαδρομής.

Να σχεδιάσετε το (τμήμα από το) δέντρο αναζήτησης που θα παραχθεί κατά την εκτέλεση του αλγορίθμου της αναζήτησης με βάση το κόστος της διαδρομής (branch and bound search), η οποία αναπτύσσει κάθε φορά τον κόμβο που το κόστος της διαδρομής από την αρχική θέση μέχρι αυτόν είναι το μικρότερο. Καταγράψτε το μονοπάτι της λύσης και υπολογίστε το κόστος του. Καταγράψτε την σειρά επέκτασης των κόμβων και υπολογίστε πόσα βήματα έκανε ο αλγόριθμος.

ε. Αναζήτηση με τον A*.

Να σχεδιάσετε το (τμήμα από το) δέντρο αναζήτησης που θα παραχθεί κατά την εκτέλεση του αλγορίθμου A*. Καταγράψτε το μονοπάτι της λύσης και υπολογίστε το κόστος του. Καταγράψτε την σειρά επέκτασης των κόμβων και υπολογίστε πόσα βήματα έκανε ο αλγόριθμος.