ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

ΕΝΟΤΗΤΑ 1.4

Εισαγωγή στις Βασικές Έννοιες Αλγορίθμων Εύρεσης Μη Βέλτιστης Λύσης σε Χώρο Καταστάσεων

Διδάσκων Ενότητας:
Κώστας Κοντογιάννης

ΜΕΡΟΣ 1: Επίλυση Προβλημάτων στο Χώρο των καταστάσεων

Περίληψη Ενότητας 1.4

Θέματα

- Ταξινόμηση Αλγορίθμων Αναζήτησης Λύσης
- Αλγόριθμοι Εύρεσης μη Βέλτιστης Λύσης σε χώρο καταστάσεων

Υλικό

- Διαφάνειες παρουσίασης
- Κεφάλαια 3, 4 Βιβλίου

Βιβλιογραφία - Αναφορές

3

- Υλικό για αυτή τη διάλεξη συγκεντρώθηκε από
  - το βιβλίο “Artificial Intelligence” by P.H. Winston, Addison Wesley, 1984 και
  - τις συνοδευτικές παρουσιάσεις του βιβλίου «Τεχνητή Νοημοσύνη», Ι. Βλαχάβας κ.α, Εκδόσεις Β. Γκιούρδας

Γενικά

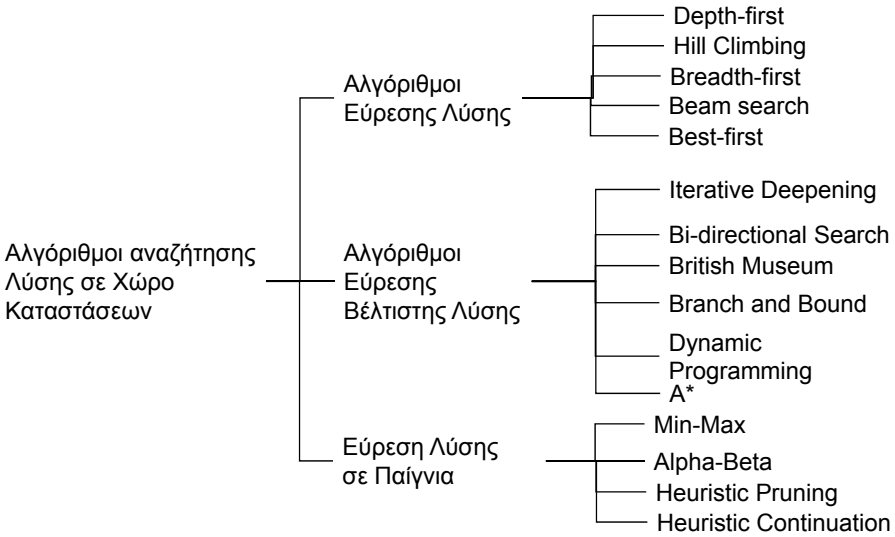
4

- Ο σκοπός των επόμενων τριών διαλέξεων θα είναι αυτής της ενότητας είναι να παρουσιάσει :

1. Αλγόριθμους που βρίσκουν κάποια λύση στο χώρο καταστάσεων
2. Αλγόριθμους που βρίσκουν την βέλτιστη λύση στο χώρο καταστάσεων
3. Αλγόριθμους που βρίσκουν λύσεις όταν ο χώρος καταστάσεων κατασκευάζεται κατά τη διάρκεια ενός παιγνίου με δύο αντιπάλους που και οι δύο εκτελούν κινήσεις στη προσπάθειά τους να κερδίσουν το παιχνίδι

Ταξινόμηση Αλγόριθμων Αναζήτησης

5



Αναζήτηση Λύσης

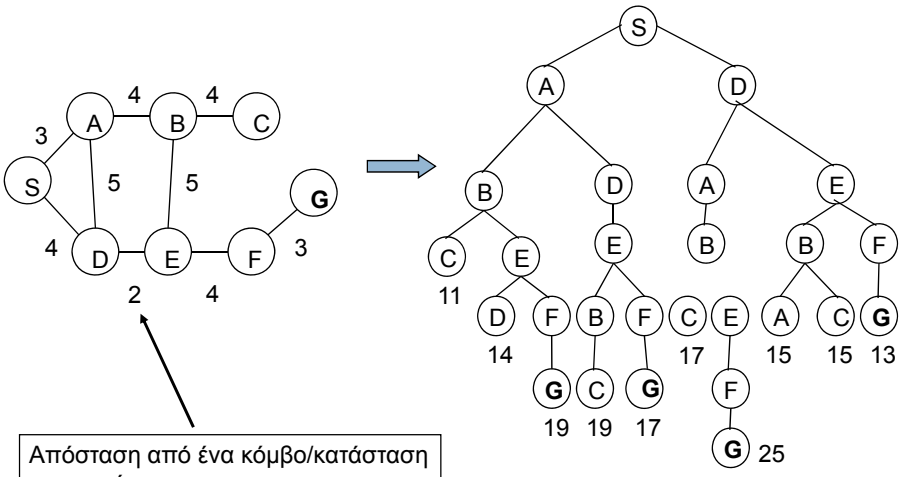
6

- Γενικά όταν λέμε ότι αναζητούμε μια λύση σε ένα χώρο καταστάσεων ουσιαστικά μιλάμε για την αναζήτηση ενός μονοπατιού στο χώρο των καταστάσεων που μας οδηγεί από την αρχική κατάσταση σε μια «επιτυχή» τελική κατάσταση

- Οι διαφορετικές καταστάσεις που μπορούν να προσπελασθούν όταν ήδη είμαστε σε μια κατάσταση ορίζονται από τα χαρακτηριστικά του προβλήματος και δημιουργούν αυτό που ονομάζουμε *δένδρο αναζήτησης*

Παράδειγμα Δένδρου Αναζήτησης

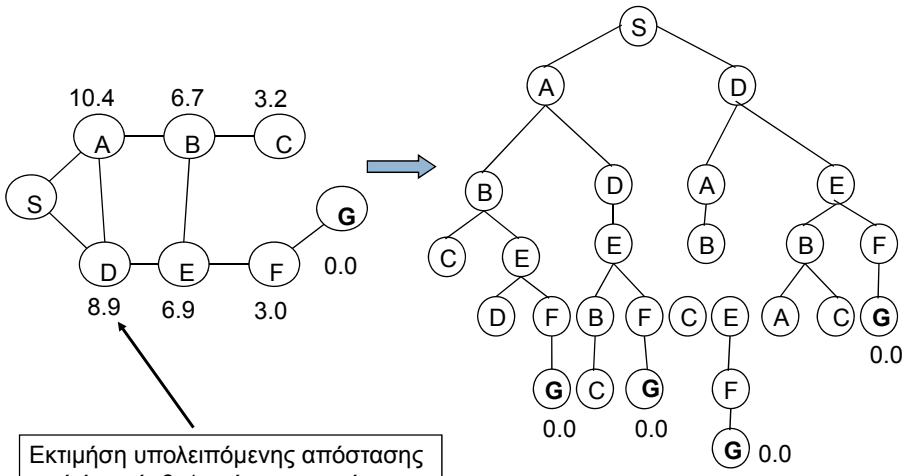
7



Απόσταση από ένα κόμβο/κατάσταση στην επόμενη.

Παράδειγμα Δένδρου Αναζήτησης

8



Εκτιμήση υπολειπόμενης απόστασης από ένα κόμβο/κατάσταση σε ένα κόμβο «στόχο».

Χαρακτηριστικά του Δένδρου Αναζήτησης

9

- Αρχικός Κόμβος (Root node)
- Τελικοί Κόμβοι (Terminal Nodes)
- Κόμβος Πρόγονος (Ancestor node)
- Κόμβος Απόγονος (Descendant node)
- Παράγοντας Διακλάδωσης (Branching Factor)
- Κλειστοί Κόμβοι (Closed nodes)

Ευριστικοί και Μη-Ευριστικοί Αλγόριθμοι Αναζήτησης

10

- Όταν η απόσταση / κόστος για τη μοντελοποίηση κάθε κόμβου / κατάστασης στο χώρο αναζήτησης από τον κόμβο / κατάσταση που θεωρούμε στόχο δίνεται με βάση κάποια ευριστική μέθοδο τότε λέμε ότι έχουμε ένα αλγόριθμο ***ευριστικής αναζήτησης***
- Όταν δεν έχουμε κάποια μέθοδο να εστιάσουμε το επόμενο μας βήμα, δηλαδή ψάχνουμε στα τυφλά, τότε έχουμε ένα αλγόριθμο μη-ευριστικής αναζήτησης ή αλλιώς ένα αλγόριθμο ***τυφλής αναζήτησης***

Αλγόριθμοι Τυφλής Αναζήτησης

11

- Οι *αλγόριθμοι τυφλής αναζήτησης* (*blind search algorithms*) εφαρμόζονται σε προ-βλήματα στα οποία δεν υπάρχει πληροφορία που να επιτρέπει την αξιολόγηση των καταστάσεων του χώρου αναζήτησης (Δηλαδή ψάχνουμε στα τυφλά)
- Έτσι οι αλγόριθμοι αυτοί αντιμετωπίζουν με τον ίδιο ακριβώς τρόπο οποιοδήποτε πρόβλημα καλούνται να λύσουν
- Για τους αλγορίθμους τυφλής αναζήτησης, το τι απεικονίζει κάθε κατάσταση του προβλήματος είναι παντελώς αδιάφορο. Σημασία έχει η χρονική σειρά με την οποία παράγονται οι καταστάσεις από το μηχανισμό επέκτασης.

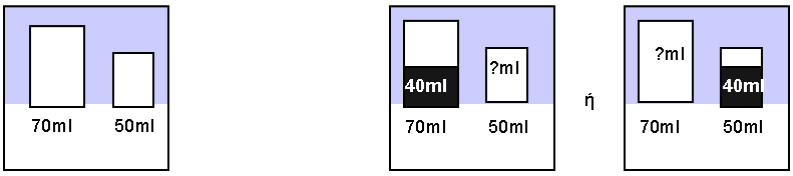
Λίστα Αλγορίθμων Τυφλής Αναζήτησης

12

Όνομα Αλγορίθμου	Συντομογραφία	Ελληνική Ορολογία
Depth-First Search	DFS	Αναζήτηση Πρώτα σε Βάθος
Breadth-First Search	BFS	Αναζήτηση Πρώτα σε Πλάτος
Iterative Deepening	ID	Επαναληπτική Εκβάθυνση
Bi-directional Search	BiS	Αναζήτηση Διπλής Κατεύθυνσης
Branch and Bound	B&B	Επέκταση και Οριοθέτηση
Beam Search	BS	Ακτινωτή Αναζήτηση

Παράδειγμα (Ποτήρια) – (i)

13



Τελεστής (1)
Γέμισε το ποτήρι των X ml μέχρι το χείλος από τη βρύση
Προϋποθέσεις
Το ποτήρι των X ml έχει 0 ml
Αποτελέσματα
Το ποτήρι των X ml έχει X ml

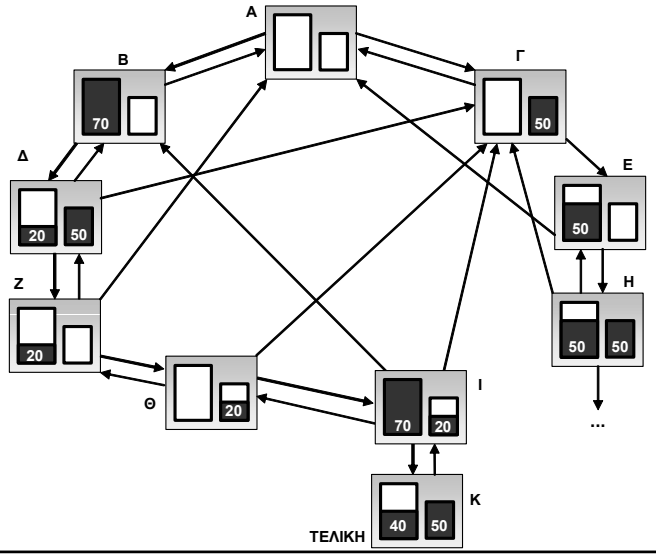
Παράδειγμα (Ποτήρια) – (ii)

14

Τελεστής (2)
Γέμισε το ποτήρι των X ml από το ποτήρι των Y ml
Προϋποθέσεις Το ποτήρι των X ml έχει Z ml Το ποτήρι των Y ml έχει W ml (W≠0)
Αποτελέσματα Το ποτήρι των X ml έχει X ml και Το ποτήρι των Y ml έχει W-(X-Z), αν W≥X-Z ή Το ποτήρι των X ml έχει Z+W ml και Το ποτήρι των Y ml έχει 0, αν W<X-Z
Τελεστής (3)
Άδειασε το ποτήρι των X ml στο νεροχύτη
Προϋποθέσεις Το ποτήρι έχει περιεχόμενο
Αποτελέσματα Το ποτήρι των X ml έχει 0 ml

Μέρος του Χώρου Αναζήτησης

15



Αλγόριθμοι Ευριστικής Αναζήτησης

16

- Οι αλγόριθμοι τυφλής αναζήτησης προχωρούν σε βάθος ή πλάτος χωρίς ένδειξη για το αν πλησιάζουν σε τερματική κατάσταση.
- Σε πολλά προβλήματα ο χώρος αναζήτησης αυξάνεται ραγδαία (συνδυαστική έκρηξη).
  - **Σκοπός**: Μείωση του αριθμού των καταστάσεων που επισκέπτεται ο αλγόριθμος.
  - **Απαιτείται**: Πληροφορία για αξιολόγηση των καταστάσεων που θα οδηγήσει γρηγορότερα στη λύση ή θα βοηθήσει στο κλάδεμα καταστάσεων που δεν οδηγούν πουθενά.
- Οι αλγόριθμοι που εκμεταλλεύονται τέτοιες πληροφορίες (κάποιον ευριστικό μηχανισμό) ονομάζονται Αλγόριθμοι Ευριστικής Αναζήτησης.

Αλγόριθμοι Ευριστικής Αναζήτησης

17

- Παράδειγμα ευριστικής αναζήτησης είναι η συναρμολόγηση ενός puzzle.
  - Με τυφλή αναζήτηση θα έπρεπε να προσπαθήσουμε να το φτάξουμε, χωρίς να δώσουμε καμία σημασία στο χρώμα ή το σχήμα των κομματιών.
  - Στην πραγματικότητα όμως, ενεργούμε εντελώς διαφορετικά. Χωρίζουμε τα κομμάτια σε κατηγορίες, π.χ. αυτά που ανήκουν στην περιφέρεια (μία άκρη τους είναι ευθεία τομή), σε αυτά που πιθανά ανήκουν στο χρώμα του ουρανού ή στη θάλασσα ή στα δένδρα κ.ο.κ.
  - Οι ενέργειες αυτές αντιστοιχούν στη χρήση ενός ευριστικού μηχανισμού που έχουμε αναπτύξει μέσω της εμπειρίας μας.
  - Ωστόσο, ποτέ δεν είμαστε σίγουροι ότι ο διαχωρισμός των κομματιών είναι και ο σωστός. Για παράδειγμα, υπάρχουν κομμάτια του puzzle που μπορεί να ανήκουν στη θάλασσα ή τον ουρανό.
- Αν δεν υπήρχαν ευριστικοί μηχανισμοί, τότε τα προβλήματα αυτά θα λύνονταν πολύ δύσκολα, γιατί οι συνδυασμοί που πρέπει να γίνουν είναι πάρα πολλοί.
  - Θα ήταν σα να προσπαθεί κάποιος να συναρμολογήσει ένα puzzle από την ανάποδη πλευρά ή σα να ψάχνει την κεντρική πλατεία της πόλης με τα μάτια κλειστά.
- Ο ευριστικός μηχανισμός εξαρτάται από τη γνώση που έχουμε για το πρόβλημα.

Ευριστικός Μηχανισμός

18

- **Ευριστικός μηχανισμός (heuristic) είναι μία στρατηγική, βασισμένη στη γνώση για το συγκεκριμένο πρόβλημα, η οποία χρησιμοποιείται σα βοήθημα στη γρήγορη επίλυσή του.**
- Ο ευριστικός μηχανισμός υλοποιείται με ευρι-στική συνάρτηση (heuristic function), που έχει πεδίο ορισμού το σύνολο των κατα-στάσεων ενός προβλήματος και πεδίο τιμών το σύνολο τιμών που αντιστοιχεί σε αυτές.
- Σε όμοια προβλήματα χρησιμοποιούμε παρόμοιους ευριστικούς μημανισμούς.
- Ευριστική τιμή (heuristic value) είναι η τιμή της ευριστικής συνάρτησης και εκφράζει το πόσο κοντά βρίσκεται μία κατάσταση σε μία τελική.
- Δεν είναι πραγματική αλλά μια εκτίμηση (πολλές φορές λανθασμένη)
- Η ευριστική τιμή δεν είναι η πραγματική τιμή της απόστασης από μία τερματική κατάσταση, αλλά μία εκτίμηση (estimate*)* που πολλές φορές μπορεί να είναι και λανθασμένη.

Ευριστικές Συναρτήσεις σε Μικρά Προβλήματα (i)

19

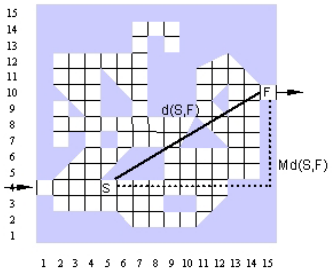
Ευριστικός μηχανισμός και συναρτήσεις σε λαβύρινθο

❖Ευκλείδεια απόσταση (Euclidian distance):

$$d(S,F) = \sqrt{(X_S - X_F)^2 + (Y_S - Y_F)^2}$$

❖Απόσταση Manhattan (Manhattan distance):

Md(S,F) = | XS - XF | + | YS -YF |



$$d(S,F) = \sqrt{(5-15)^2 + (4-10)^2} = \sqrt{(100+36)} = 11{,}6$$

Md(S,F)=|5-15| + |4-10|=10+6=16

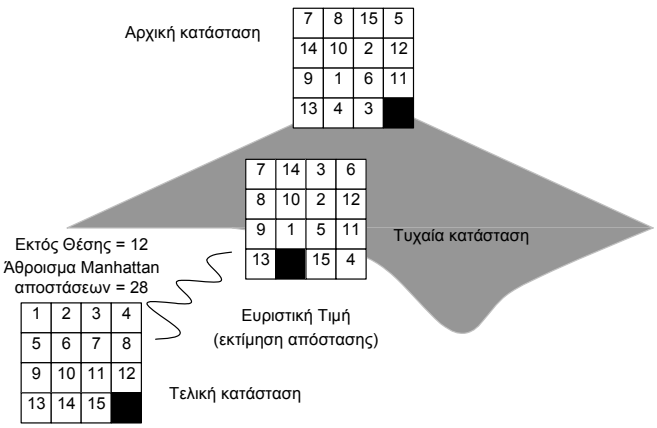
Ευριστικές Συναρτήσεις σε Μικρά Προβλήματα (ii)

20

Ευριστικός μηχανισμός και συναρτήσεις στο N-Puzzle

•Πόσα πλακίδια βρίσκονται εκτός θέσης.

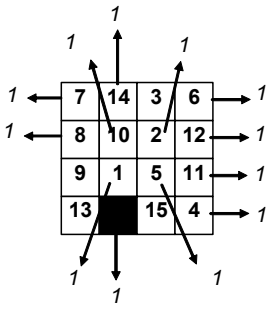
•Το άθροισμα των αποστάσεων Manhattan κάθε πλακιδίου από την τελική του θέση.



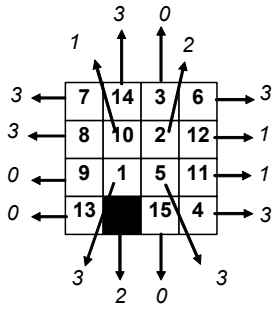
Ευριστικές Συναρτήσεις σε Μικρά Προβλήματα (iii)

21

Αναλυτικός υπολογισμός ευριστικής τιμής για μία τυχαία κατάσταση του 15-puzzle.



Εκτός θέσης = 12



Άθροισμα αποστάσεων Manhattan = 28

Λίστα Αλγορίθμων Ευριστικής Αναζήτησης

22

Όνομα Αλγορίθμου	Συντομογραφία	Ελληνική Ορολογία
Hill Climbing	**HC**	Αναρρίχηση Λόφων
Beam Search	**Beam SearchFS**	Ακτινωτή Αναζήτηση
Best First	**BestFS**	Πρώτα στο Καλύτερο
Iterative Deepening	**ID**	Επαναληπτικής Εκβάθυνσης
Bidirectional Search	**BiS**	Διπλής Κατεύθυνσης
A*	**A Star**	Αναζήτηση Άλφα Άστρο

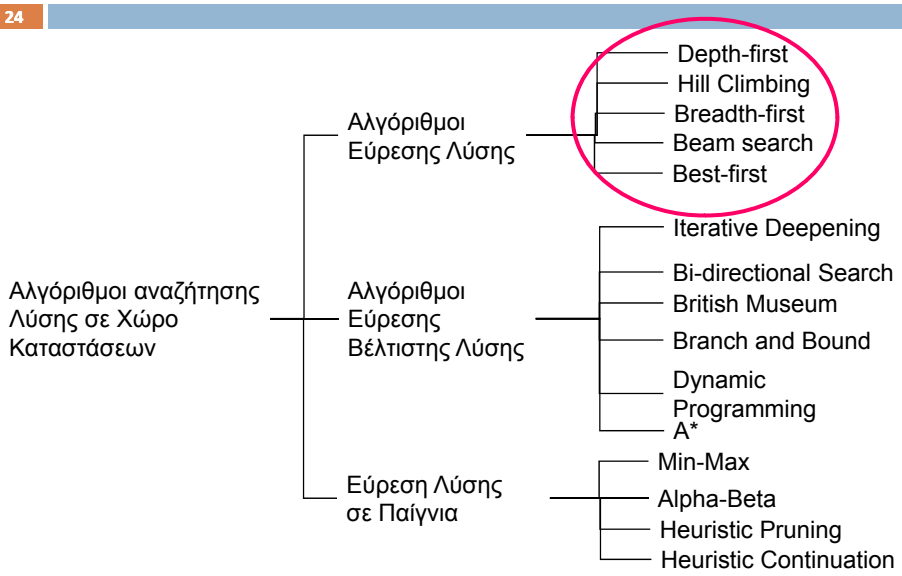
Αλγόριθμοι Μη-Βέλτιστης Λύσης

23

- Στις επόμενες διαφάνειες θα μιλήσουμε για μη-ευριστικούς και ευριστικούς αλγόριθμους που μπορούν να βρούν κάποια λύση στο χώρο καταστάσεων χωρίς απαραίτητα αυτή η λύση να είναι η καλύτερη.

Ταξινόμηση Αλγόριθμων Αναζήτησης

24



Αναζήτηση Κατά-Βάθος (Depth First)

25

- Ο Αλγόριθμος Αναζήτησης κατά Βάθος (Depth First Search) ουσιαστικά είναι ο αλγόριθμος που κατασκευάζει / διασχίζει κατά βάθος το δένδρο αναζήτησης
- Δηλαδή ο αριστερότερος κόμβος είναι αυτός που επεκτείνεται πρώτος, και σε συνέχεια η σειρά επέκτασης είναι από αριστερά προς τα δεξιά του δένδρου
- Το πρόβλημα που υπάρχει με την Κατά Βάθος αναζήτηση είναι ότι εάν φτάσουμε σε κάποιο κόμβο που μπορεί να επεκτείνεται χωρίς τέλος (πχ. A→B→A) τότε ο αλγόριθμος μπαίνει σε ένα ατελείωτο βρόχο (infinite loop)
- Εάν περιορίσουμε το μέγιστο επιτρεπτό βάθος που ο αλγόριθμος μπορεί να εξετάσει (έστω το βάθος *l*) τότε η τεχνική ονομάζεται Περιορισμένη Κατά-Βάθος Αναζήτηση (Depth Limited Search)

Αλγόριθμος Αναζήτησης Depth First

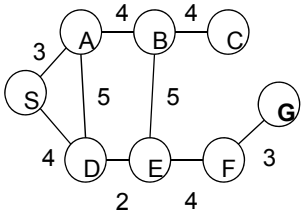
26

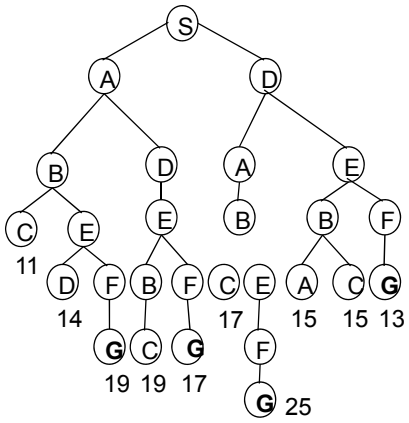
- **Βήμα 1**: Κατασκεύασε μια λίστα ουρά που περιέχει τη ρίζα του δένδρου (αρχική κατάσταση)
- **Βήμα 2**: Μέχρι που η λίστα να αδειάσει ή να βρεθεί ένας τελικός κόμβος στόχος
  - **Βήμα 2.α**: Εάν ο πρώτος κόμβος στη λίστα είναι κόμβος στόχος μη κάνεις τίποτε
  - **Βήμα 2.β**: Εάν ο πρώτος κόμβος στη λίστα δεν είναι κόμβος στόχος, τότε βγάλε το πρώτο κόμβο από τη λίστα, και βάλε στη **αρχή** της λίστας τα παιδιά του κόμβου (δηλ. όλες τις καταστάσεις που μπορούν να προσπελασθούν με ένα μόνο βήμα από το κόμβο/κατάσταση που μόλις αφαιρέσαμε από τη λίστα)
  - **Βήμα 2.γ**: Εάν ο πρώτος κόμβος δεν έχει παιδία απλά αφαίρεσέ τον από τη λίστα και πήγαινε στο βήμα 2
- **Βήμα 3**: Εάν βρήκαμε ένα κόμβο στόχο τότε ανακοινώνουμε επιτυχία αλλιώς ανακοινώνουμε αποτυχία

Παράδειγμα Κατά Βάθος Αναζήτησης

27

Στο παρακάτω πρόβλημα, παρουσιάζεται το μοντέλο, το ολικό δένδρο αναζήτησης, και η λίστα που δημιουργείται στη κατά-βάθος αναζήτηση του δένδρου η *Κατά Βάθος* αναζήτηση δημιουργεί τη παρακάτω λίστα.





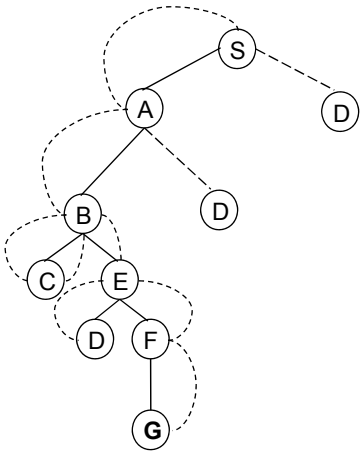
[S]
[A,D]
[B, D, D]
[C, E, D, D]
[E, D, D]
[D, F, D, D]
[F, D, D]
[G, D, D]
Sucess

Παράδειγμα Κατά Βάθος Αναζήτησης στο Δένδρο Αναζήτησης

28

Λίστα

[S]
[A,D]
[B, D, D]
[C, E, D, D]
[E, D, D]
[D, F, D, D]
[F, D, D]
[G, D, D]
Sucess



Ψευδοκώδικας του Αλγόριθμου Αναζήτησης Hill Climbing

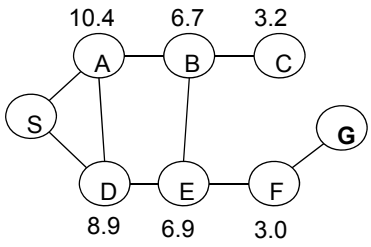
29

- **Βήμα 1**: Κατασκεύασε μια λίστα ουρά που περιέχει τη ρίζα του δένδρου (αρχική κατάσταση)
- **Βήμα 2**: Μέχρι που η λίστα να αδειάσει ή να βρεθεί ένας τελικός κόμβος στόχος, εξέτασε εάν ο πρώτος κόμβος στη λίστα είναι κόμβος στόχος
  - **Βήμα 2.α**: Εάν ο πρώτος κόμβος στη λίστα είναι κόμβος στόχος τότε μη κάνεις τίποτε και η επανάληψη σταματά, και πάμε στο Βήμα 3
  - **Βήμα 2.β**: Εάν ο πρώτος κόμβος στη λίστα δεν είναι κόμβος στόχος, τότε βγάλε το πρώτο κόμβο από τη λίστα, βρες στο παιδιά αυτού του κόμβου (δηλ. όλες τις καταστάσεις που μπορούν να προσπελασθούν με ένα μόνο βήμα από το κόμβο/κατάσταση που μόλις αφαιρέσαμε από τη λίστα), ταξινόμησε αυτά τα παιδιά σε αύξουσα σειρά με βάση τη υπολογιζόμενη υπόλοιπη απόστασή τους από το στόχο και βάλε στη αρχή της λίστας τα ταξινομημένα παιδιά αυτού του κόμβου
  - **Βήμα 2.γ**: Εάν ο πρώτος κόμβος δεν έχει παιδία απλά αφαίρεσέ τον από τη λίστα και πήγαινε στο βήμα 2
- **Βήμα 3**: Εάν βρήκαμε ένα κόμβο στόχο τότε ανακοινώνουμε επιτυχία αλλιώς ανακοινώνουμε αποτυχία

Παράδειγμα Αλγόριθμου Hill Climbing

30

Στο παρακάτω πρόβλημα* η *Hill Climbing* αναζήτηση δημιουργεί τη παρακάτω λίστα.



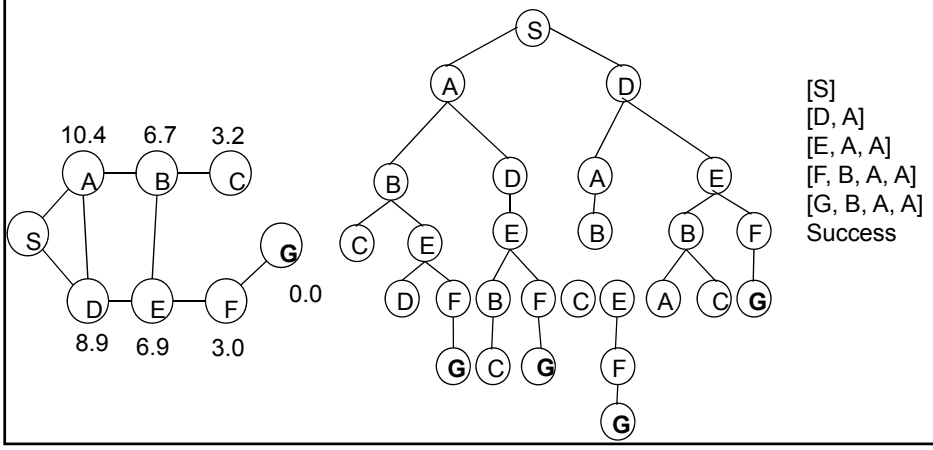
[S]
[D, A]
[E, A, A]
[F, B, A, A]
[G, B, A, A]
Success

Υποθέτουμε εδώ ότι η υπολογιζόμενη υπολειπόμενη απόσταση
από τον κόμβο A στον Στόχο G είναι: 10.4
από τον κόμβο Δ στον Στόχο G είναι: 8.9
από τον κόμβο E στον Στόχο G είναι: 6.9
από τον κόμβο B στον Στόχο G είναι: 6.7
από τον κόμβο F στον Στόχο G είναι: 3.0

Παράδειγμα Αναζήτησης Hill Climbing

31

Στο παρακάτω πρόβλημα, παρουσιάζεται το μοντέλο, το ολικό δένδρο αναζήτησης, και η λίστα που δημιουργείται στη hill climbing αναζήτηση του δένδρου



[S]
[D, A]
[E, A, A]
[F, B, A, A]
[G, B, A, A]
Success

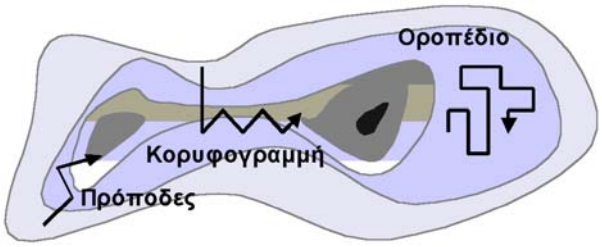
Προβλήματα με τη Τεχνική Hill Climbing

32

- Η τεχνική Hill Climbing είναι σαν να προσπαθούμε να βρούμε κάποιο σημείο όπου όλα τα σημεία που μπορούμε να πάμε από αυτό το σημείο με ένα βήμα έχουν χειρότερα χαρακτηριστικά από αυτό το σημείο
- Παραδείγματα είναι όταν μπαίνουμε σε ένα δωμάτιο και αρχίζουμε να ρυθμίζουμε το θερμοστάτη θέρμανσης/ψύξης μέχρι να βρούμε κάποιο σημείο που αισθανόμαστε άνετα. Εδώ δεν υπάρχει κάποιο συγκεκριμένο σημείο «στόχος», αλλά σταματάμε σε κάποιο σημείο (θέση του θερμοστάτη) όπου όλες οι άλλες γειτονικές ρυθμίσεις δεν μας δίνουν καλύτερα αποτελέσματα απ' ότι αυτό το σημείο (θέση του θερμοστάτη).
- Δηλαδή, η τεχνική Hill Climbing, προσπαθεί να βρει μια λύση (σημείο στο χώρο καταστάσεων) όπου έχει βελτιστοποιηθεί η τιμή μιας παραμέτρου του προβλήματος (π.χ. Πόσο άνετα αισθανόμαστε) ή ένας συνδυασμός τιμών παραμέτρων του προβλήματος
- Όπως καταλαβαίνουμε η τεχνική είναι απλή, δηλαδή εύκολο να υλοποιηθεί και να εφαρμοστεί, αλλά έχει και κάποια σημαντικά προβλήματα όπως τα προβλήματα
  - Πρόποδες (foothill).
  - Οροπέδιο (plateau).
  - Κορυφογραμμή (ridges).

Προβλήματα με τη Τεχνική Hill Climbing

33

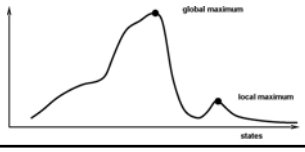


Βελτιώσεις:
- *Προσομοιωμένη εξέλιξη* (*Simulated Annealing - SA*)
- *Εξαναγκασμένη αναρρίχηση λόφου* (*Enforced Hill-Climbing - EHC*)
- *Αναζήτηση με απαγορευμένες καταστάσεις* (*Tabu Search - TS*).

Αλγόριθμος Προσομοιωμένη εξέλιξη (Simulated Annealing - SA)

34

- Στον αλγόριθμο Hill Climbing παίρνουμε πάντα την επόμενη καλύτερη κίνηση που μπορούμε να εκτελέσουμε την κάθε στιγμή
- Είδαμε όμως ότι ο αλγόριθμος αυτός μπορεί να μας «κολλήσει» σε τοπικά μέγιστα "local maxima"
- Μια λύση σε αυτό το πρόβλημα είναι να επιτρέψουμε στον αλγόριθμο να κάνει και «κακές» κινήσεις. Στην αρχή δίνουμε μικρή ποινή για τη πρώτη «κακή» κίνηση αλλά πιο μεγάλη ποινή για την επόμενη «κακή» κίνηση κλπ.
- Το κόστος των ποινών αυξάνεται εκθετικά. Η ιδέα είναι να επιτρέψουμε κάποιες κακές κινήσεις με την ελπίδα ότι σύντομα θα βρεθούμε εκτός του «τοπικού μέγιστου» και θα βρούμε ένα καλύτερο μονοπάτι που θα μας οδηγήσει στο (ιδανικά) στο "γενικό μέγιστο" (global maximum)



Αλγόριθμος Simulated Annealing - Ψευδοκώδικας

35

Function SIMULATED-ANNEALING(problem, schedule) **returns** state

Input: *problem*: a problem

schedule: a mapping from time to "temperature"

Local Variables: *current*: a node, *next*: a node,

T: a "temperature" controlling the probability of a bad move

current ← MAKE-NODE(INITIAL-STATE[problem])

for t ← 1 to ∞ **do**

T ← *schedule[t]*

if *T* == 0 then return *current*

next ← a randomly selected successor of *current*

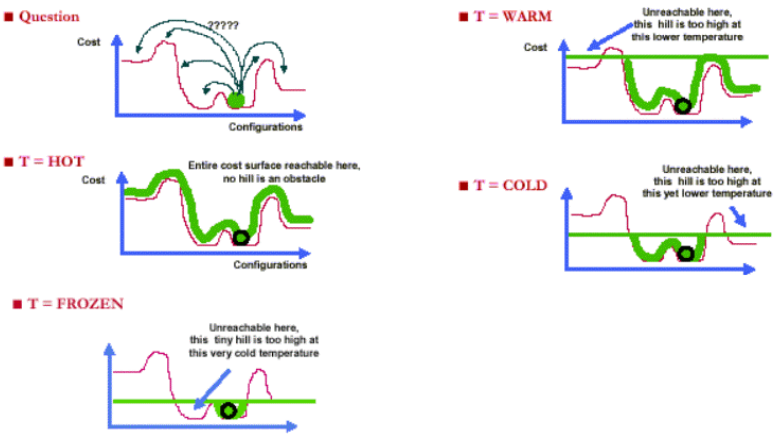
ΔE ← VALUE[*next*] – VALUE[*current*]

if *ΔE* > 0 then *current* ← *next*

else *current* ← *next* only with probability $e^{\Delta E/T}$

Ο ρόλος της μεταβλητής "Temperature" στον Αλγόριθμο SA

36



Αναζήτηση κατά Πλάτος (Breadth first)

37

- Όταν ο αλγόριθμοι της Κατά-Πλάτος αναζήτησης και του Hill Climbing δεν είναι εφικτοί ή κατάλληλοι, τότε μπορούμε επίσης να εξετάσουμε τη χρήση του αλγόριθμου αναζήτησης Κατά-Βάθος
- Ο Αλγόριθμος Αναζήτησης κατά Πλάτος (Breadth First Search) ουσιαστικά είναι ο αλγόριθμος που κατασκευάζει / διασχίζει κατά πλάτος το δένδρο αναζήτησης
- Σύμφωνα με τον Κατά-Πλάτος αλγόριθμο αναζήτησης, ψάχνουμε για τον κόμβο «στόχο» σε όλους τους κόμβους του δένδρου που βρίσκονται κάθε φορά στο ίδιο επίπεδο. Σε κάθε κίνηση του αλγόριθμου κοιτάμε τους κόμβους ένα επίπεδο πιο βαθιά στο δένδρο

Αλγόριθμος Αναζήτησης Breadth First

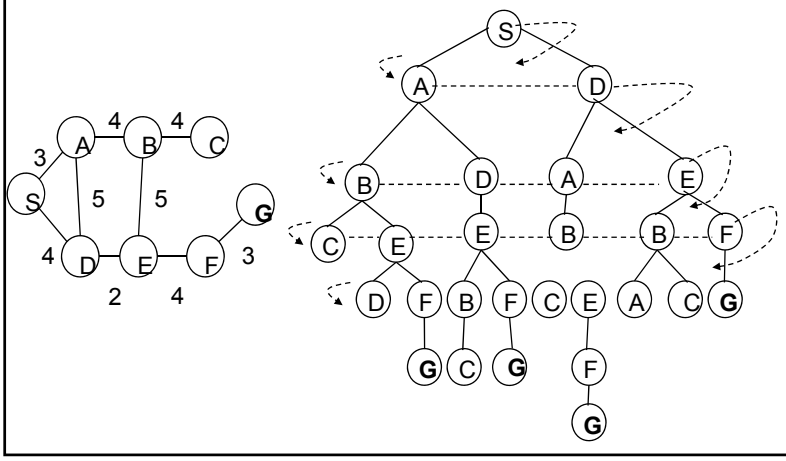
38

- **Βήμα 1**: Κατασκεύασε μια λίστα ουρά που περιέχει τη ρίζα του δένδρου (αρχική κατάσταση)
- **Βήμα 2**: Μέχρι που η λίστα να αδειάσει ή να βρεθεί ένας τελικός κόμβος στόχος
  - **Βήμα 2.α**: Εάν ο πρώτος κόμβος στη λίστα είναι κόμβος στόχος μη κάνεις τίποτε
  - **Βήμα 2.β**: Εάν ο πρώτος κόμβος στη λίστα δεν είναι κόμβος στόχος, τότε βγάλε το πρώτο κόμβο από τη λίστα, και βάλε στο **τέλος** της λίστας τα παιδιά του κόμβου (δηλ. όλες τις καταστάσεις που μπορούν να προσπελασθούν με ένα μόνο βήμα από το κόμβο/κατάσταση που μόλις αφαιρέσαμε από τη λίστα)
  - **Βήμα 2.γ**: Εάν ο πρώτος κόμβος δεν έχει παιδία απλά αφαίρεσέ τον από τη λίστα και πήγαινε στο βήμα 2
- **Βήμα 3**: Εάν βρήκαμε ένα κόμβο στόχο τότε ανακοινώνουμε επιτυχία αλλιώς ανακοινώνουμε αποτυχία

Παράδειγμα Breath First

39

Στο παρακάτω πρόβλημα, παρουσιάζεται το μοντέλο, το ολικό δένδρο αναζήτησης, και η λίστα που δημιουργείται στη breadth-first αναζήτηση του δένδρου



[S]
[A, D]
[D, B, D]
[B, D, A, E]
[D, A, E, C, E]
[A, E, C, E, E]
[E, C, E, E, B]
[C, E, E, B, F]
[E, E, B, F]
[E, B, F, D, F]
……..

Αλγόριθμος Αναζήτησης Beam Search

40

- Ο *αλγόριθμος ακτινωτής αναζήτησης* (*Beam Search - BS*) είναι παρόμοιος με τον αλγόριθμο αναζήτησης Κατά-Πλάτος διότι προχωράμε στην αναζήτηση ανά επίπεδο του δένδρου αναζήτησης. Όμως η διαφορά είναι ότι δεν κοιτάμε όλους τους κόμβους του επόμενου επιπέδου, αλλά μόνο του *w* καλύτερους κόμβους όσον αφορά στη βαθμολόγησή τους σε σχέση με την υπολειπόμενη απόσταση από κάποιο κόμβο «στόχο». Δηλαδή σε κάθε βήμα κρατάμε τους *w* καλύτερους κόμβους στο μέτωπο αναζήτησης

- Αυτή η μέθοδος όπως και η Hill Climbing είναι ευριστική στη βάση ότι χρησιμοποιεί μια ευριστική εκτίμηση της υπολειπόμενης απόστασης ενός κόμβου (σημείου του χώρου κατάστασης) από ένα κόμβο «στόχο»

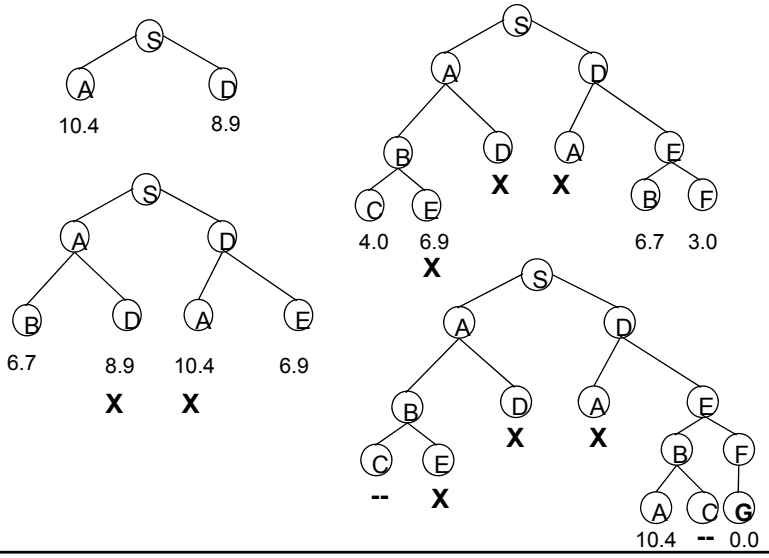
Αλγόριθμος Αναζήτησης Beam Search

41

- **Βήμα 1**: Κατασκεύασε μια λίστα ουρά που περιέχει τη ρίζα του δένδρου (αρχική κατάσταση)
- **Βήμα 2**: Μέχρι που η λίστα να αδειάσει ή να βρεθεί ένας τελικός κόμβος στόχος
  - **Βήμα 2.α**: Εάν ο πρώτος κόμβος στη λίστα είναι κόμβος στόχος μη κάνεις τίποτε
  - **Βήμα 2.β**: Εάν ο πρώτος κόμβος στη λίστα δεν είναι κόμβος στόχος, τότε βγάλε το πρώτο κόμβο από τη λίστα, και βάλε στο **τέλος** της λίστας τα w το πλήθος καλύτερα παιδιά του κόμβου (δηλ. όλες τις καταστάσεις που μπορούν να προσπελασθούν με ένα μόνο βήμα από το κόμβο/κατάσταση που μόλις αφαιρέσαμε από τη λίστα). όσον αφορά την ευριστική υπολειπόμενη απόσταση του κόμβου από ένα κόμβο «στόχο»
  - **Βήμα 2.γ**: Εάν ο πρώτος κόμβος δεν έχει παιδία απλά αφαίρεσέ τον από τη λίστα και πήγαινε στο βήμα 2
- **Βήμα 3**: Εάν βρήκαμε ένα κόμβο στόχο τότε ανακοινώνουμε επιτυχία αλλιώς ανακοινώνουμε αποτυχία

Παράδειγμα Διεύρυνσης του Δένδρου Αναζήτησης κατά την Beam Search

42



Αλγόριθμος Αναζήτησης Best First Search

43

- Ο *αλγόριθμος αναζήτησης Best First*) είναι όπως ο αλγόριθμος αναζήτησης Hill Climbing, μόνο που εδώ επεκτείνουμε όχι τον καλύτερο κόμβο από τα παιδιά του κόμβου που είμαστε, αλλά επεκτείνουμε κάθε φορά τον καλύτερο κόμβο από όλους τους κόμβους που βρίσκονται στο μέτωπο αναζήτησης του δένδρου
- Ο αλγόριθμος αναζήτησης Best First Search έχει την πιθανότητα να παράγει τα μικρότερα μονοπάτια από την αρχική κατάσταση (ρίζα του δένδρου αναζήτησης) σε κάποιο κόμβο «στόχο»
- Η βασική διαφορά αυτού του αλγόριθμου από τον Hill Climbing είναι ότι αντί να ταξινομούμε τα παιδιά του κόμβου που επεκτείνουμε και να εισάγουμε αυτή τη ταξινομημένη λίστα στην ολική λίστα, εισάγουμε τα παιδιά του κόμβου που επεκτείνουμε στη λίστα, και μετά ταξινομούμε όλη τη λίστα. Οπότε ο καλύτερος κόμβος συνολικά πηγαίνει στη κορυφή της λίστας για επέκταση στο επόμενο βήμα.

Ψευδοκώδικας του Αλγόριθμου Αναζήτησης Best First Search

44

- **Βήμα 1**: Κατασκεύασε μια λίστα ουρά που περιέχει τη ρίζα του δένδρου (αρχική κατάσταση)
- **Βήμα 2**: Μέχρι που η λίστα να αδειάσει ή να βρεθεί ένας τελικός κόμβος στόχος, εξέτασε εάν ο πρώτος κόμβος στη λίστα είναι κόμβος στόχος
  - **Βήμα 2.α**: Εάν ο πρώτος κόμβος στη λίστα είναι κόμβος στόχος τότε μη κάνεις τίποτε και η επανάληψη σταματά, και πάμε στο Βήμα 3
  - **Βήμα 2.β**: Εάν ο πρώτος κόμβος στη λίστα δεν είναι κόμβος στόχος, τότε βγάλε το πρώτο κόμβο από τη λίστα, βρες στο παιδιά αυτού του κόμβου (δηλ. όλες τις καταστάσεις που μπορούν να προσπελασθούν με ένα μόνο βήμα από το κόμβο/κατάσταση που μόλις αφαιρέσαμε από τη λίστα), βάλε στη λίστας τα παιδιά αυτού του κόμβου, και μετά **ταξινόμησε σε αύξουσα σειρά ολόκληρη τη λίστα** σε σχέση με την υπολειπόμενη απόσταση του κάθε κόμβου στη λίστα από ένα κόμβο «στόχο»
  - **Βήμα 2.γ**: Εάν ο πρώτος κόμβος δεν έχει παιδία απλά αφαίρεσέ τον από τη λίστα και πήγαινε στο βήμα 2
- **Βήμα 3**: Εάν βρήκαμε ένα κόμβο στόχο τότε ανακοινώνουμε επιτυχία αλλιώς ανακοινώνουμε αποτυχία

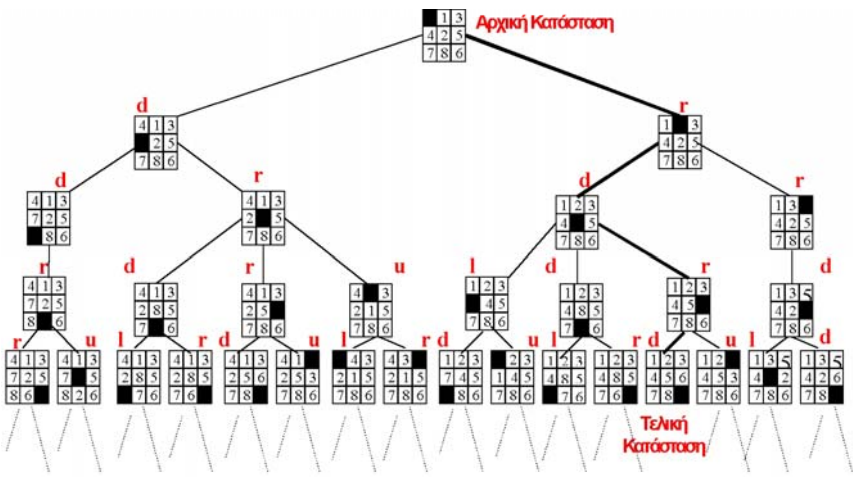
Σχόλια για τον Αλγόριθμο Best First Search

45

- Πλεονεκτήματα:
  - Προσπαθεί να δώσει μια γρήγορη λύση σε κάποιο πρόβλημα. Το αν τα καταφέρει ή όχι εξαρτάται πολύ από τον ευριστικό μηχανισμό.
  - Είναι πλήρης – (εάν υπάρχει λύση ο αλγόριθμος θα τη βρεί)
- Μειονεκτήματα:
  - Το μέτωπο αναζήτησης μεγαλώνει με υψηλό ρυθμό και μαζί του ο χώρος που χρειάζεται για την αποθήκευσή του, καθώς και ο χρόνος για την επεξεργασία των στοιχείων του.
  - Δεν εγγυάται ότι η λύση που θα βρεθεί είναι η βέλτιστη.

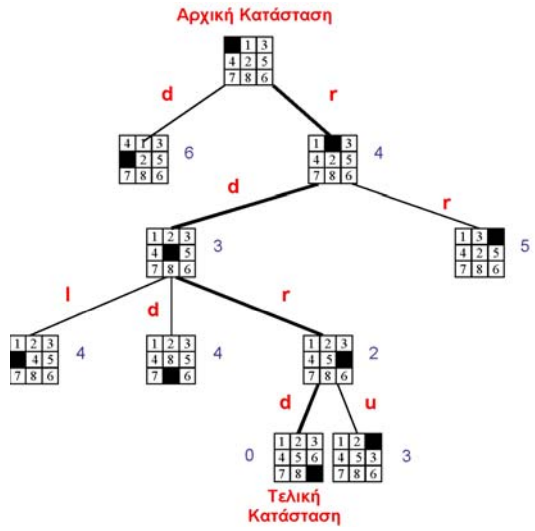
Παράδειγμα Δένδρου Αναζήτησης

46



Εφαρμογή του Αλγόριθμου Best First

47



Γενικά Σχόλια για την Επιλογή του Κατάλληλου Αλγόριθμου

48

- Η Κατά-Βάθος αναζήτηση είναι καλή όταν το δένδρο αναζήτησης δεν βαθαίνει πολύ
- Η Κατά-Πλάτος αναζήτηση είναι καλή όταν ο Παράγοντας Διακλάδωσης δεν είναι μεγάλος
- Η τεχνική Hill-Climbing είναι καλή όταν έχουμε μια καλή ευριστική μετρική για την υπολειπόμενη απόσταση και όταν μια καλή επιλογή είναι συνήθως σε αυτές που οδηγούν σε λύση (όχι πολλά τοπικά μέγιστα ή τοπικά ελάχιστα)
- Η Τεχνική Beam Search είναι καλή όταν έχουμε μια καλή ευριστική μετρική για την υπολειπόμενη απόσταση και όταν ένα καλό μονοπάτι είναι συνήθως σε αυτά που μπορούν να επιλεγούν σε κάθε επίπεδο του δένδρου
- Η τεχνική Best First είναι καλή όταν έχουμε μια καλή ευριστική μετρική για την υπολειπόμενη απόσταση και όταν ένα καλό μονοπάτι μπορεί να μη φαίνεται καλό στην αρχή