ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

Τεχνητή νοημοσύνη και μηχανική μάθηση

Εισηγητής
Αναστάσιος Κεσίδης

Τυφλή αναζήτηση

Αναζήτηση λύσεων

Αναπαραστάσεις για αλγόριθμους αναζήτησης

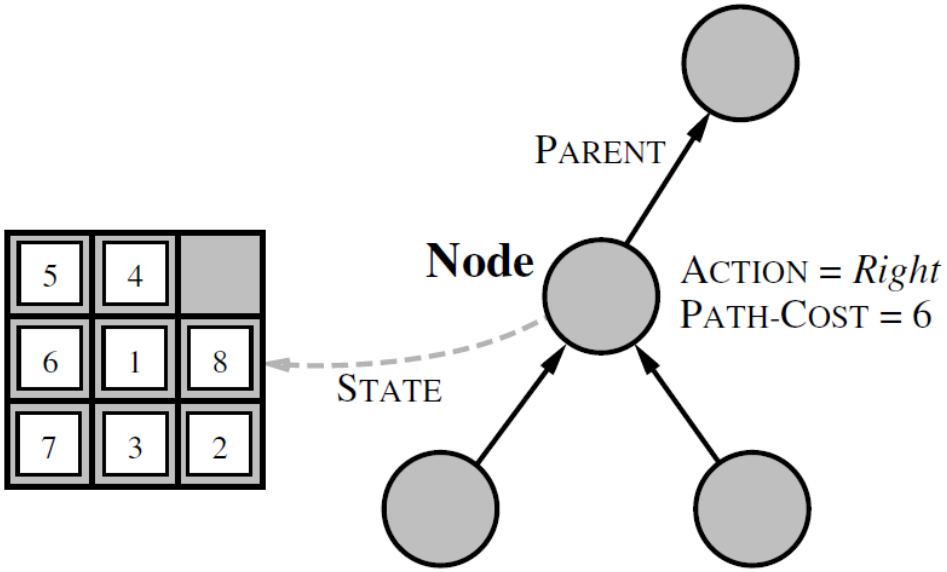
Δομή δεδομένων

- STATE: Η κατάσταση του χώρου καταστάσεων στην οποία αντιστοιχεί ο κόμβος
- PARENT: Ο κόμβος του δένδρου αναζήτησης (γονικός κόμβος) ο οποίος παρήγαγε τον τρέχοντα κόμβο
- ACTION: Η ενέργεια που εφαρμόστηκε στον γονικό κόμβο για να παραχθεί ο τρέχον κόμβος
- PATH-COST: το κόστος (συνήθως συμβολίζεται $g(n)$) της διαδρομής από την αρχική κατάσταση μέχρι τον τρέχοντα κόμβο όπως προκύπτει από την ακολουθία των γονικών δεικτών
- DEPTH: Ο αριθμός των βημάτων πάνω στην διαδρομή από την αρχική κατάσταση

Αναζήτηση λύσεων

- Αναπαραστάσεις για αλγόριθμους αναζήτησης

Παράδειγμα



Κόμβος
- Δομή δεδομένων για αναπαράσταση δένδρου αναζήτησης

≠

Κατάσταση
- Αντιστοιχεί σε μια διάταξη του κόσμου

Δύο διαφορετικοί κόμβοι μπορεί να περιέχουν την ίδια κατάσταση του κόσμου η οποία όμως παράγεται από δύο διαφορετικές διαδρομές αναζήτησης.

Αναζήτηση λύσεων

- Αναπαραστάσεις για αλγόριθμους αναζήτησης

Δημιουργία νέων κόμβων

```
function EXPAND(node, problem) returns a set of nodes

successors ← an empty set

for each action, result in SUCCESSOR-FN(problem,STATE[node]) do

    s ← a new NODE

    PARENT-NODE[s] ← node

    ACTION[s] ← action

    STATE[s] ← result

    PATH-COST[s] ← PATH-COST[node]+ STEP-COST(STATE[node], action, result)

    DEPTH[s] ← DEPTH[node] + 1

    add s to successors

return successors
```

- Η συνάρτηση EXPAND δημιουργεί καινούργιους κόμβους, συμπληρώνει τα πεδία τους και χρησιμοποιεί την SUCCESSOR-FN του προβλήματος ώστε να δημιουργηθούν καινούργιες καταστάσεις

Αναζήτηση λύσεων

- Αποτελεσματικότητα αλγορίθμων αναζήτησης

Αξιολογείται με τους ακόλουθους τρόπους:

Πληρότητα (completeness): Είναι εξασφαλισμένο ότι ο αλγόριθμος θα βρει την λύση εάν υπάρχει;

Βέλτιστη συμπεριφορά (optimality): Βρίσκει πάντα μια λύση με το χαμηλότερο κόστος;

Χρονική πολυπλοκότητα (time complexity): Πόσο χρόνο χρειάζεται για να βρει την λύση;

Χωρική πολυπλοκότητα (space complexity): Πόση μνήμη χρειάζεται για να βρει την λύση;

Αναζήτηση λύσεων

- Αποτελεσματικότητα αλγορίθμων αναζήτησης

Πολυπλοκότητα

Εκφράζεται με τρεις όρους

b μέγιστος **παράγοντας διακλάδωσης** του δέντρου αναζήτησης
d **βάθος** της λύσης με το χαμηλότερο κόστος
m **μέγιστο βάθος** του χώρου κατάστασης (μπορεί να είναι άπειρο)

Στην πράξη

Η **χρονική** πολυπλοκότητα συχνά εκφράζεται με το **πλήθος των κόμβων** που παράγονται

Η **χωρική** πολυπλοκότητα συχνά εκφράζεται με το **μέγιστο αριθμό κόμβων** που αποθηκεύονται στην **μνήμη**

Αναζήτηση λύσεων

Κατηγορίες στρατηγικής αναζήτησης

Τεχνικές τυφλής αναζήτησης (blind search)

Δεν υπάρχουν διαθέσιμες άλλες πληροφορίες πέρα από την διατύπωση του προβλήματος.

Μεθοδολογία επίλυσης:

- Δημιουργία κόμβων
- Έλεγχος εάν κάποιος κόμβος είναι κόμβος-λύση

Οι μεθοδολογίες διαχωρίζονται μεταξύ τους με βάση την **προτεραιότητα επέκτασης** των κόμβων.

Τεχνικές πληροφορημένης αναζήτησης (informed search)

Βασίζονται σε κάποια **ευριστική συνάρτηση** που εκτιμά το κόστος μιας λύσης

Τυφλή αναζήτηση

- Αναζήτηση πρώτα σε πλάτος (breadth-first search)

Μεθοδολογία

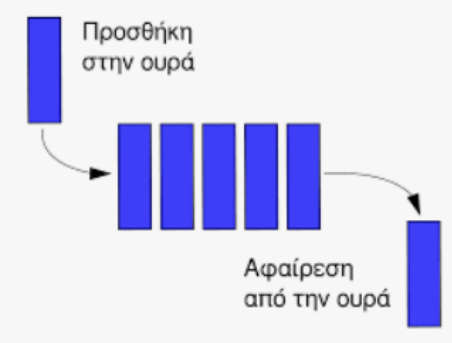
Αρχικά επεκτείνεται ο ριζικός κόμβος.

Έπειτα οι διάδοχοι του ριζικού κόμβου.

Έπειτα οι διάδοχοι των διαδόχων κόμβων κοκ.

Όλοι οι κόμβοι επεκτείνονται σε ένα δεδομένο βάθος πριν από οποιονδήποτε κόμβο στο επόμενο επίπεδο.

Χρήση **ουράς First In First Out - FIFO**



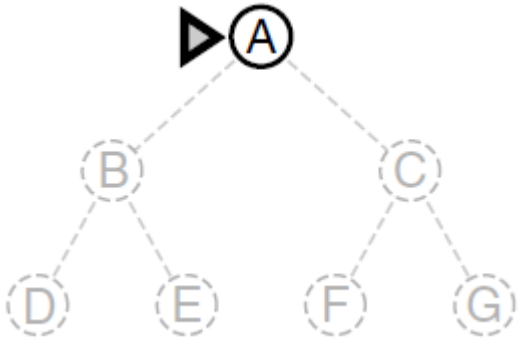
Τυφλή αναζήτηση

- Αναζήτηση πρώτα σε πλάτος (breadth-first search)

Παράδειγμα

1ο Βήμα

Επέκταση του ριζικού κόμβου



Ουρά
[A]

Επιλογή A από ουρά
και επέκταση (B, C)

Ουρά
[B, C]

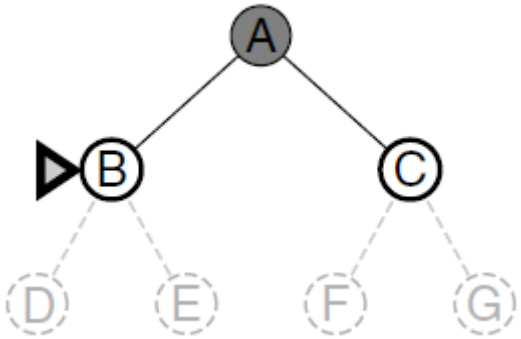
Τυφλή αναζήτηση

- Αναζήτηση πρώτα σε πλάτος (breadth-first search)

Παράδειγμα

2ο Βήμα

Επέκταση του ρηχότερου κόμβου



Ουρά
[B, C]

Επιλογή Β από ουρά
και επέκταση (D, E)

Ουρά
[C, D, E]

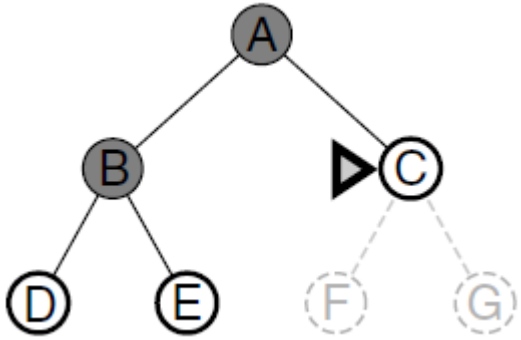
Τυφλή αναζήτηση

- Αναζήτηση πρώτα σε πλάτος (breadth-first search)

Παράδειγμα

3ο Βήμα

Επέκταση του επόμενου ρηχότερου κόμβου



Ουρά
[C, D, E]

Επιλογή C από ουρά
και επέκταση (F, G)

Ουρά
[D, E, F, G]

Τυφλή αναζήτηση

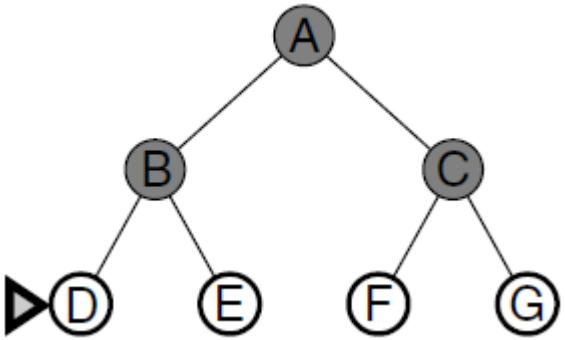
- Αναζήτηση πρώτα σε πλάτος (breadth-first search)

Παράδειγμα

4ο Βήμα

Επέκταση του επόμενου ρηχότερου κόμβου

(εάν υπάρχει)



Ουρά
[D, E, F, G]

Ουρά
[E, F, G]

Ουρά
[F, G]

Ουρά
[G]

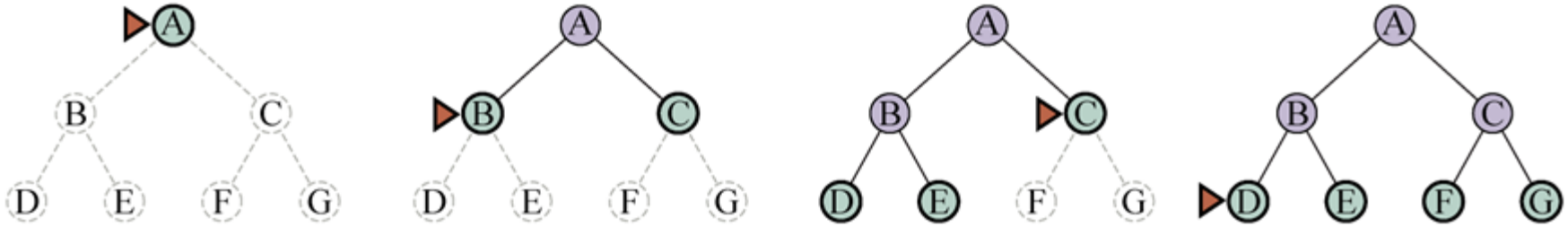
Ουρά
[]

Τυφλή αναζήτηση

- Αναζήτηση πρώτα σε πλάτος (breadth-first search)

Παράδειγμα

Συνοπτικά



Πράσινο: τρέχον σύνορο

Τρίγωνο: κόμβος που θα επεκταθεί στην συνέχεια

Μωβ: κόμβοι που έχουν επεκταθεί

Διακεκομμένη γραμμή: πιθανοί μελλοντικοί κόμβοι

Τυφλή αναζήτηση

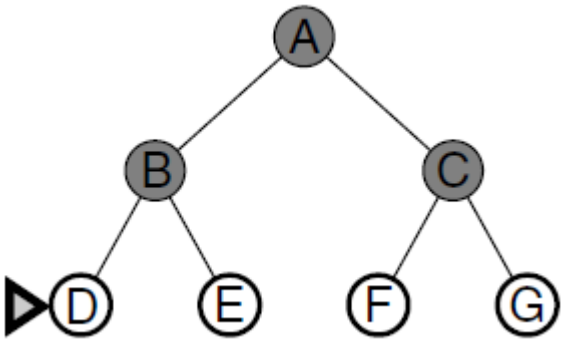
- Αναζήτηση πρώτα σε πλάτος (breadth-first search)

Είναι πλήρης

Εάν ο ρηχότερος κόμβος-στόχος είναι σε κάποιο πεπερασμένο βάθος d, η αναζήτηση θα τον βρει μετά τη δημιουργία όλων των ρηχών κόμβων (θεωρώντας ότι ο παράγοντας διακλάδωσης b είναι **πεπερασμένος**).

Μόλις βρεθεί ένας κόμβος-στόχος, είναι και ο **ρηχότερος** καθώς όλοι οι ρηχότεροι κόμβοι έχουν ήδη εξετασθεί και απέτυχε η δοκιμή στόχου.

Ο ρηχότερος κόμβος στόχου **δεν** είναι απαραίτητα ο **βέλτιστος**



Βέλτιστη αναζήτηση, **εάν** το κόστος διαδρομής είναι **μη-φθίνουσα** συνάρτηση του βάθους του κόμβου

Τυφλή αναζήτηση

➢ Αναζήτηση πρώτα σε πλάτος (breadth-first search)

Χρονική πολυπλοκότητα

Εάν κάθε κόμβος έχει b απόγονους και ο κόμβος-στόχος είναι σε βάθος d τότε το πλήθος των κόμβων που δημιουργούνται είναι

$$b + b^2 + b^3 + \cdots + b^d = \boldsymbol{O(b^d)}$$

Εκθετικό ως προς d

$b = 2$

2 κόμβοι

$2 + 4 = 6$ κόμβοι

Τυφλή αναζήτηση

- Αναζήτηση πρώτα σε πλάτος (breadth-first search)

Χωρική πολυπλοκότητα

Το σύνολο των κόμβων που έχουν ελεγχθεί είναι

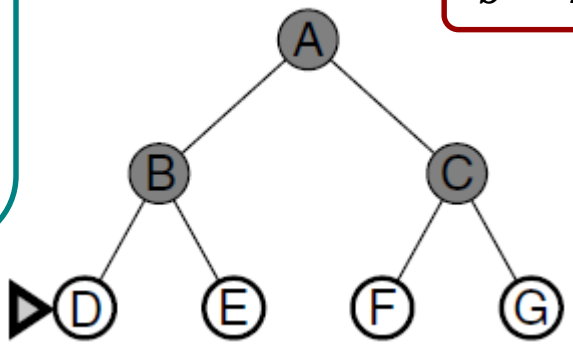
$$b + b^2 + b^3 + \cdots + b^d = \boldsymbol{O(b^d)}$$

Ομοίως, εκθετικό ως προς d

Κάθε κόμβος που δημιουργείται κρατείται στην μνήμη

(για να δημιουργηθεί στο τέλος η διαδρομή-λύση)

$b = 2$



Τυφλή αναζήτηση

- Αναζήτηση πρώτα σε πλάτος (breadth-first search)

$b = 10$

Βάθος	Κόμβοι	Διάρκεια	Μνήμη
2	110	0.00011 δευτερόλεπτα	107 kilobytes
4	11110	0.011 δευτερόλεπτα	10.6 megabyte
6	10^6	1.1 δευτερόλεπτα	1 gigabyte
8	10^8	2 λεπτά	10^3 gigabyte
10	10^{10}	3 ώρες	10 terabyte
12	10^{12}	13 ημέρες	1 petabyte
14	10^4	3.5 χρόνια	99 petabytes
16	10^{16}	350 χρόνια	10 exabytes

Μεγαλύτερο πρόβλημα η **χωρική** πολυπλοκότητα

Τυφλή αναζήτηση

➢ Αναζήτηση ομοιόμορφου κόστους (uniform-cost search)

Μεθοδολογία

Αντί να επεκτείνεται ο ρηχότερος κόμβος, επεκτείνεται ο κόμβος που έχει το **λιγότερο κόστος διαδρομής** $g(n)$.

Ο έλεγχος κόμβου-στόχου γίνεται **αφού έχει επιλεγεί ο κόμβος** (όχι όταν δημιουργείται).

- Εξήγηση: Ο πρώτος από τους κόμβους που δημιουργούνται (κατά την επέκταση του κόμβου-γονέα) μπορεί να **μην** είναι ο βέλτιστος με βάση το κριτήριο ελάχιστου κόστους διαδρομής.

Χαρακτηριστικά

- Όποτε η αναζήτηση ομοιόμορφου κόστους επιλέγει έναν κόμβο για επέκταση, η βέλτιστη διαδρομή προς αυτόν τον κόμβο έχει ήδη βρεθεί.
- Με δεδομένο ότι οι συναρτήσεις κόστους είναι μη-αρνητικές, το κόστος του μονοπατιού δεν ελαττώνεται ποτέ όταν προστίθενται κόμβοι.
- Δεν εξαρτάται από το πλήθος των βημάτων του μονοπατιού.

Τυφλή αναζήτηση

➢ Αναζήτηση ομοιόμορφου κόστους (uniform-cost search)

Πληρότητα

Εξασφαλίζεται όταν το κόστος για κάθε βήμα είναι **μεγαλύτερο** από μια μικρή **θετική σταθερά** $\varepsilon > 0$

Χρονική πολυπλοκότητα

Δίνεται από το πλήθος των κόμβων που έχουν κόστος μονοπατιού

$g(n) \leq C^*$

Έλεγχος **αφού** επιλεγεί ο κόμβος

$$O\left(b^{1+\left\lfloor \frac{C^*}{\varepsilon} \right\rfloor}\right)$$

όπου C^* το κόστος της βέλτιστης λύσης

- Όταν όλα τα κόστη είναι ίδια τότε η πολυπλοκότητα είναι $O(b^{1+d})$ δηλαδή παρόμοια με την αναζήτηση πρώτα σε πλάτος

Τυφλή αναζήτηση

➢ Αναζήτηση πρώτα σε βάθος (depth-first search)

Μεθοδολογία

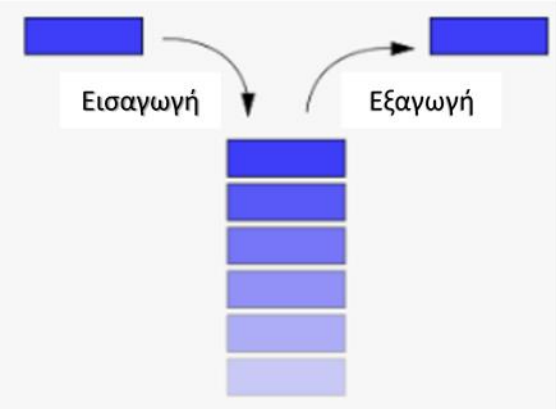
Επεκτείνεται πάντα ο **βαθύτερος** κόμβος.

Οι κόμβοι που έχουν ελεγχθεί αφαιρούνται από το μέτωπο αναζήτησης

Όταν ελεγχθεί όλο το βάθος του δέντρου, ο αλγόριθμος «οπισθοχωρεί» σε ρηχότερο επίπεδο.

Επιλέγεται για επέκταση ο κόμβος που επεκτάθηκε πιο πρόσφατα

Χρήση **στοίβας Last In First Out - LIFO**



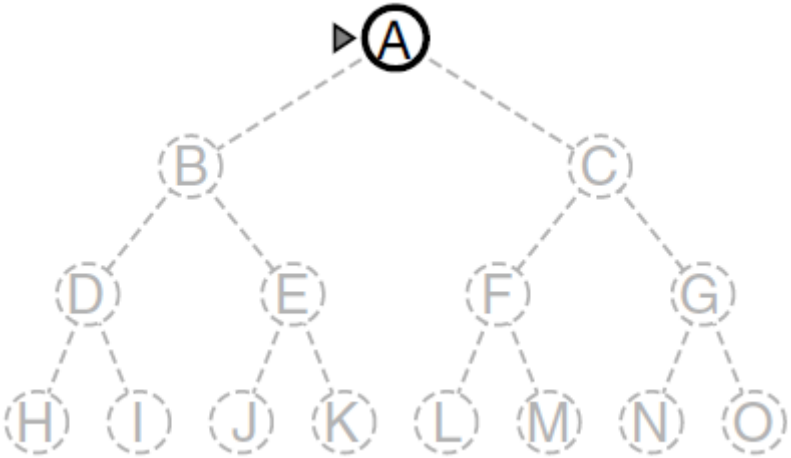
Τυφλή αναζήτηση

- Αναζήτηση πρώτα σε βάθος (depth-first search)

Παράδειγμα

1ο Βήμα

Επέκταση του ριζικού κόμβου



Στοίβα
[A]

⇩

Επιλογή A από στοίβα
και επέκταση (B, C)

⇩

Στοίβα
[B, C]

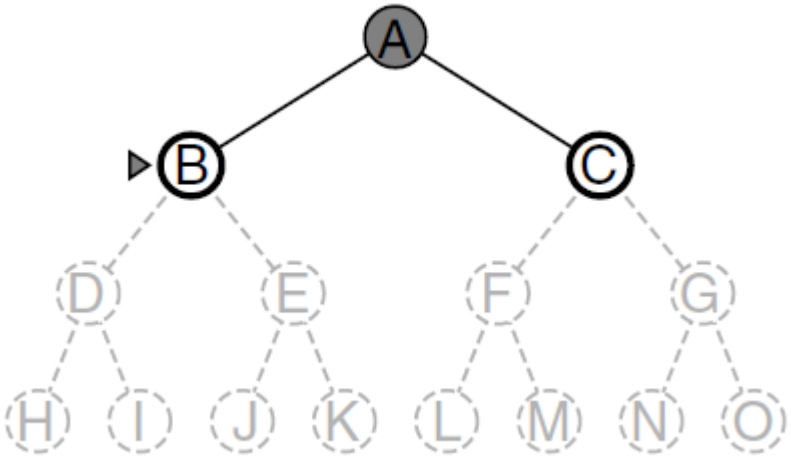
Τυφλή αναζήτηση

- Αναζήτηση πρώτα σε βάθος (depth-first search)

Παράδειγμα

2ο Βήμα

Έλεγχος του βαθύτερου κόμβου που δεν έχει ακόμη ελεγχθεί



Στοίβα
[B, C]

Επιλογή B από στοίβα
και επέκταση (D, E)

Στοίβα
[D, E, C]

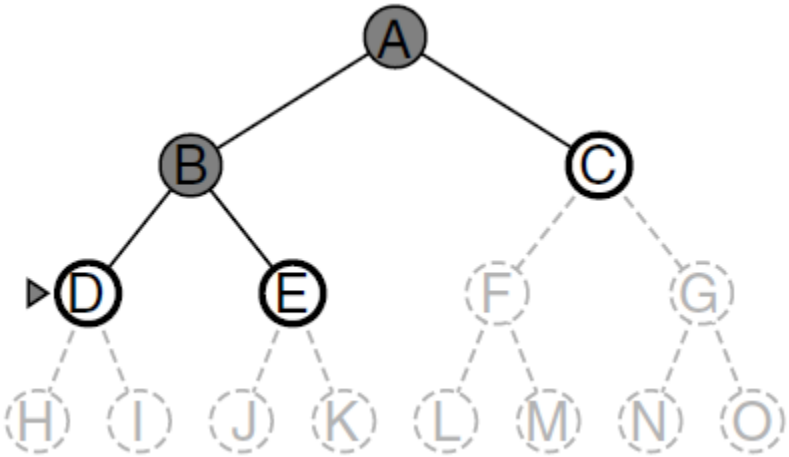
Τυφλή αναζήτηση

- Αναζήτηση πρώτα σε βάθος (depth-first search)

Παράδειγμα

3ο Βήμα

Έλεγχος του βαθύτερου κόμβου που δεν έχει ακόμη ελεγχθεί



Στοίβα
[D, E, C]

Επιλογή D από στοίβα
και επέκταση (H, I)

Στοίβα
[H, I, E, C]

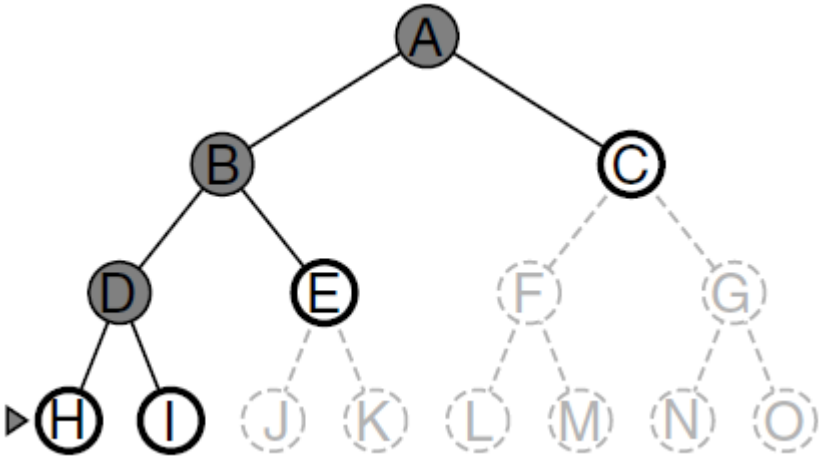
Τυφλή αναζήτηση

- Αναζήτηση πρώτα σε βάθος (depth-first search)

Παράδειγμα

4ο Βήμα

Έλεγχος του βαθύτερου κόμβου που δεν έχει ακόμη ελεγχθεί



Στοίβα
[H, I, E, C]

Στοίβα
[I, E, C]

Στοίβα
[E, C]

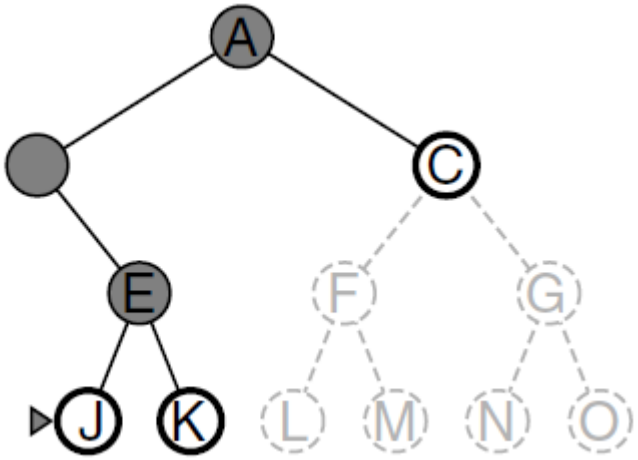
Τυφλή αναζήτηση

- Αναζήτηση πρώτα σε βάθος (depth-first search)

Παράδειγμα

5ο Βήμα

Έλεγχος του βαθύτερου κόμβου που δεν έχει ακόμη ελεγχθεί



Στοίβα
[E, C]

Επιλογή E από στοίβα
και επέκταση (J, K)

Στοίβα
[J, K, C]

Τυφλή αναζήτηση

- Αναζήτηση πρώτα σε βάθος (depth-first search)

Παράδειγμα

Συνοπτικά

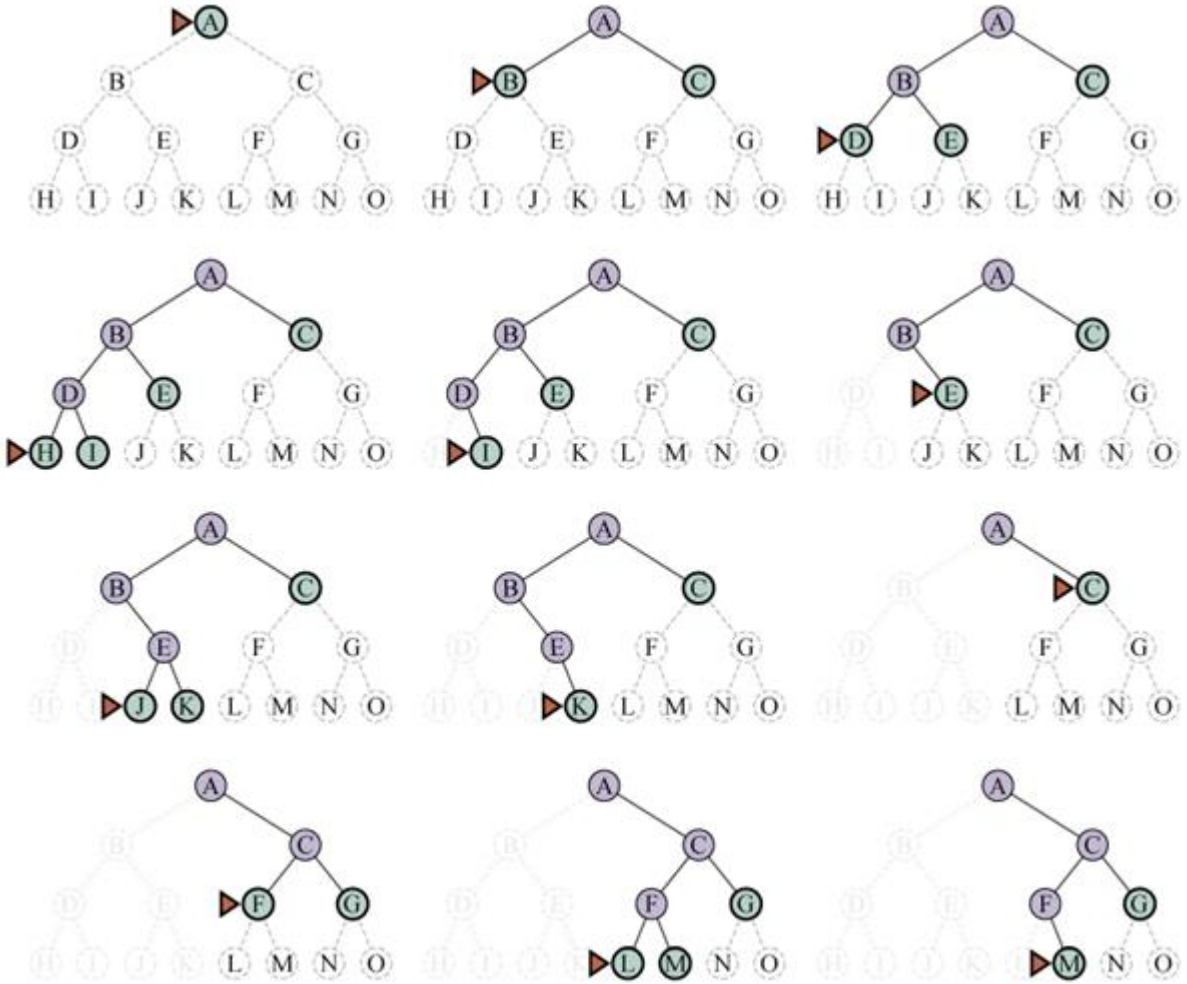
Πράσινο: τρέχον σύνορο

Τρίγωνο: κόμβος που θα επεκταθεί στην συνέχεια

Μωβ: κόμβοι που έχουν επεκταθεί

Διακεκομμένη γραμμή: πιθανοί μελλοντικοί κόμβοι

Αχνή γραμμή: Κόμβοι που έχουν επεκταθεί και δεν έχουν απογόνους στο σύνορο (μπορούν να απορριφθούν)



Τυφλή αναζήτηση

➢ Αναζήτηση πρώτα σε βάθος (depth-first search)

Πληρότητα

- Σε δομή TREE-SEARCH: **Όχι**

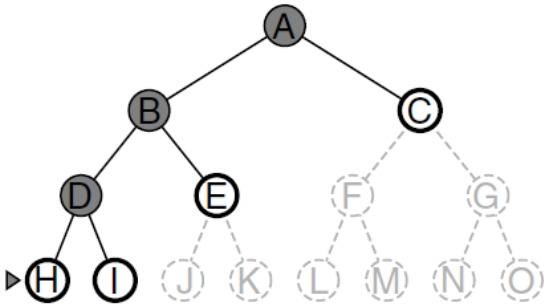
  Μπορεί να εγκλωβιστεί σε ατέρμονους βρόχους

  Π.χ. $Arad \rightarrow Sibiu \rightarrow Arad$

- Σε δομή GRAPH-SEARCH: **Ναι**

  Αγνοούνται οι ατέρμονοι βρόχοι

  Εξετάζονται όλοι οι κόμβοι



Δεν είναι βέλτιστη αναζήτηση

Π.χ. εξετάζεται όλο το αριστερό υπο-δέντρο ενώ μπορεί ο κόμβος C να είναι κόμβος-στόχος

Τυφλή αναζήτηση

➢ Αναζήτηση πρώτα σε βάθος (depth-first search)

Χρονική πολυπλοκότητα

Καθορίζεται από το μέγιστο βάθος του δέντρου m

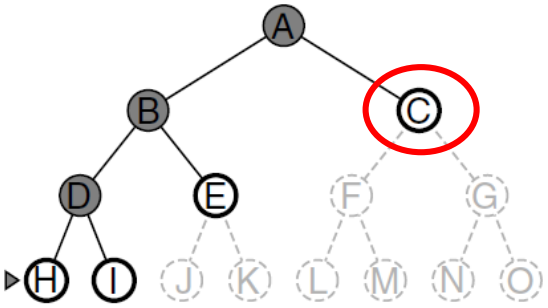
$$O(b^m)$$

- Το μέγιστο βάθος m μπορεί να είναι **πολύ μεγαλύτερο** του d (βάθος ρηχότερου κόμβου – στόχου)

Χειρότερη από αναζήτηση πρώτα σε πλάτος

Για κόμβο-στόχο C είναι

$$d = 1$$
$$m = 3$$



Τυφλή αναζήτηση

➢ Αναζήτηση πρώτα σε βάθος (depth-first search)

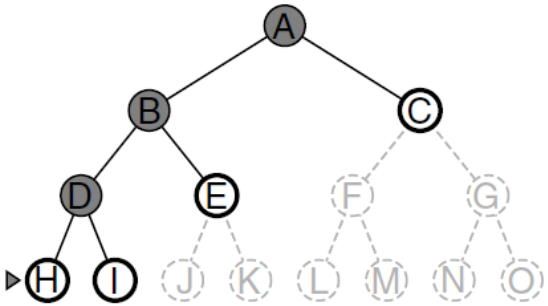
Χωρική πολυπλοκότητα

Για μέγιστο παράγοντα διακλάδωσης του δέντρου αναζήτησης b και μέγιστο βάθος του χώρου κατάστασης m, είναι **γραμμική**

$$\boldsymbol{O(bm)}$$

Σημαντικό πλεονέκτημα έναντι της αναζήτησης πρώτα σε πλάτος

- Η αναζήτηση του δέντρου πρέπει να αποθηκεύει μόνο **μία διαδρομή από τη ρίζα προς τον κόμβο** μαζί με τους υπόλοιπους γειτονικούς κόμβους που δεν έχουν ακόμη επεκταθεί για κάθε κόμβο στη διαδρομή.
- Μόλις επεκταθεί ένας κόμβος, μπορεί να **αφαιρεθεί** από τη μνήμη.



Τυφλή αναζήτηση

➢ Αναζήτηση περιορισμένου βάθος (depth-limited search)

Μεθοδολογία

Παραλλαγή της αναζήτησης πρώτα σε βάθος όπου ορίζεται ένα **μέγιστο βάθος** αναζήτηση l.

- Κόμβοι σε βάθος l θεωρείται ότι δεν έχουν διαδόχους.

Επιλύει το πρόβλημα των ατέρμονων βρόχων

Προβλήματα

- **Μη πλήρες**, εάν επιλεγεί $l < d$ (ο κόμβος−στόχος είναι πιο πέρα από το μέγιστο βάθος αναζήτησης)
- **Δεν** είναι βέλτιστο όταν $l > d$

Τυφλή αναζήτηση

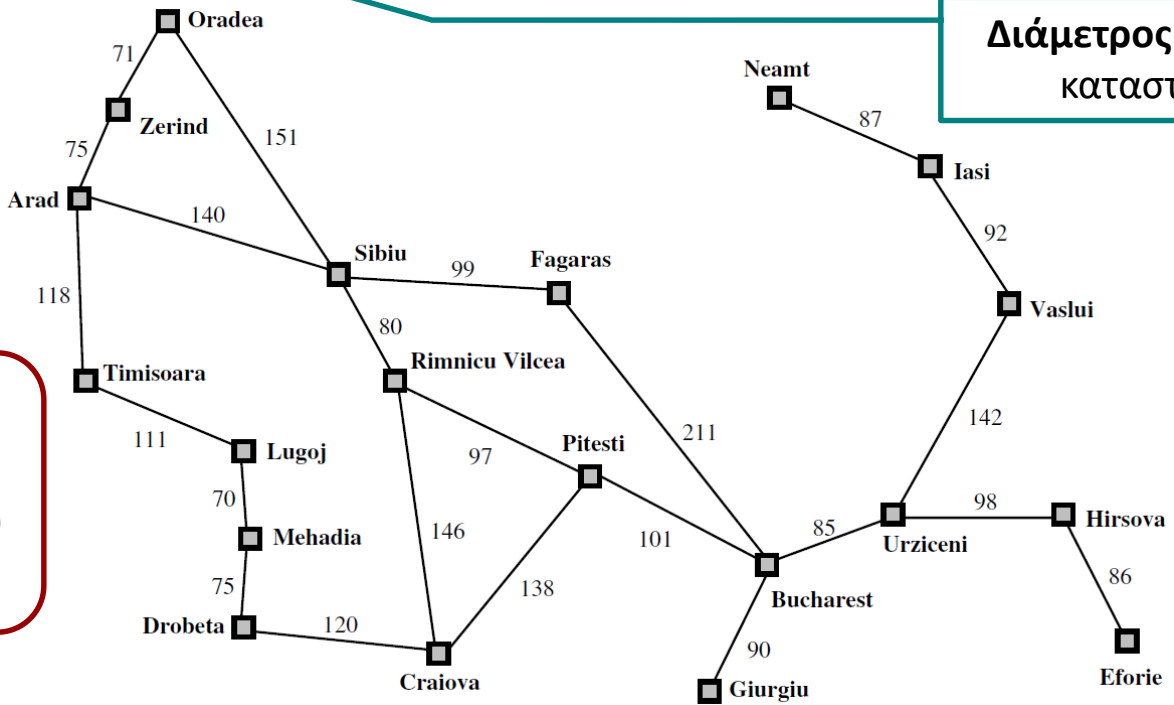
- Αναζήτηση περιορισμένου βάθος (depth-limited search)

Παράδειγμα

Το πλήθος των πόλεων του χάρτη είναι 20.

- Θα μπορούσε να οριστεί $l = 19$
- Όμως κάθε πόλη μπορεί να προσεγγιστεί από οποιαδήποτε άλλη πόλη σε $l = 9$ βήματα **το πολύ**

Διάμετρος του χώρου καταστάσεων



Απαιτείται γνώση του **συγκεκριμένου** προβλήματος

Τυφλή αναζήτηση

- Επαναληπτική αναζήτηση εμβάθυνσης (iterative deepening search)

Μεθοδολογία

Το όριο l αυξάνεται σταδιακά $l = 0, l = 1, l = 2, \ldots$ μέχρι να βρεθεί κόμβος - στόχος για $l = d$.

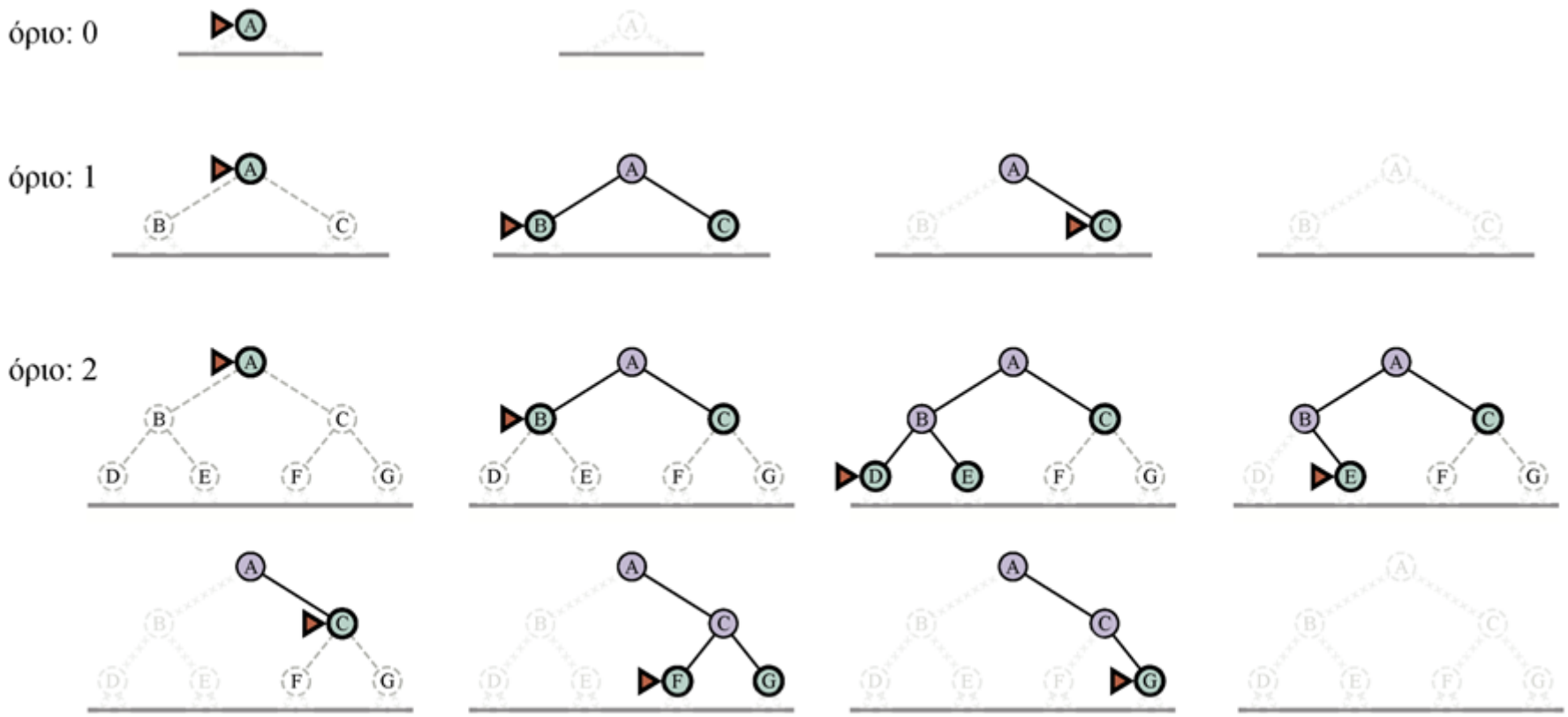
Χαρακτηριστικά

- Συνδυάζει τα πλεονεκτήματα της αναζήτησης πρώτα σε πλάτος με την αναζήτηση πρώτα σε βάθος
- Η πολυπλοκότητα είναι $O(bd)$
- Είναι **πλήρης** όταν ο μέγιστος παράγοντα διακλάδωσης του δέντρου αναζήτησης b είναι πεπερασμένος
- Είναι **βέλτιστη** όταν το κόστος διαδρομής είναι **μη-φθίνουσα** συνάρτηση του βάθους του κόμβου

Τυφλή αναζήτηση

- Επαναληπτική αναζήτηση εμβάθυνσης (iterative deepening search)

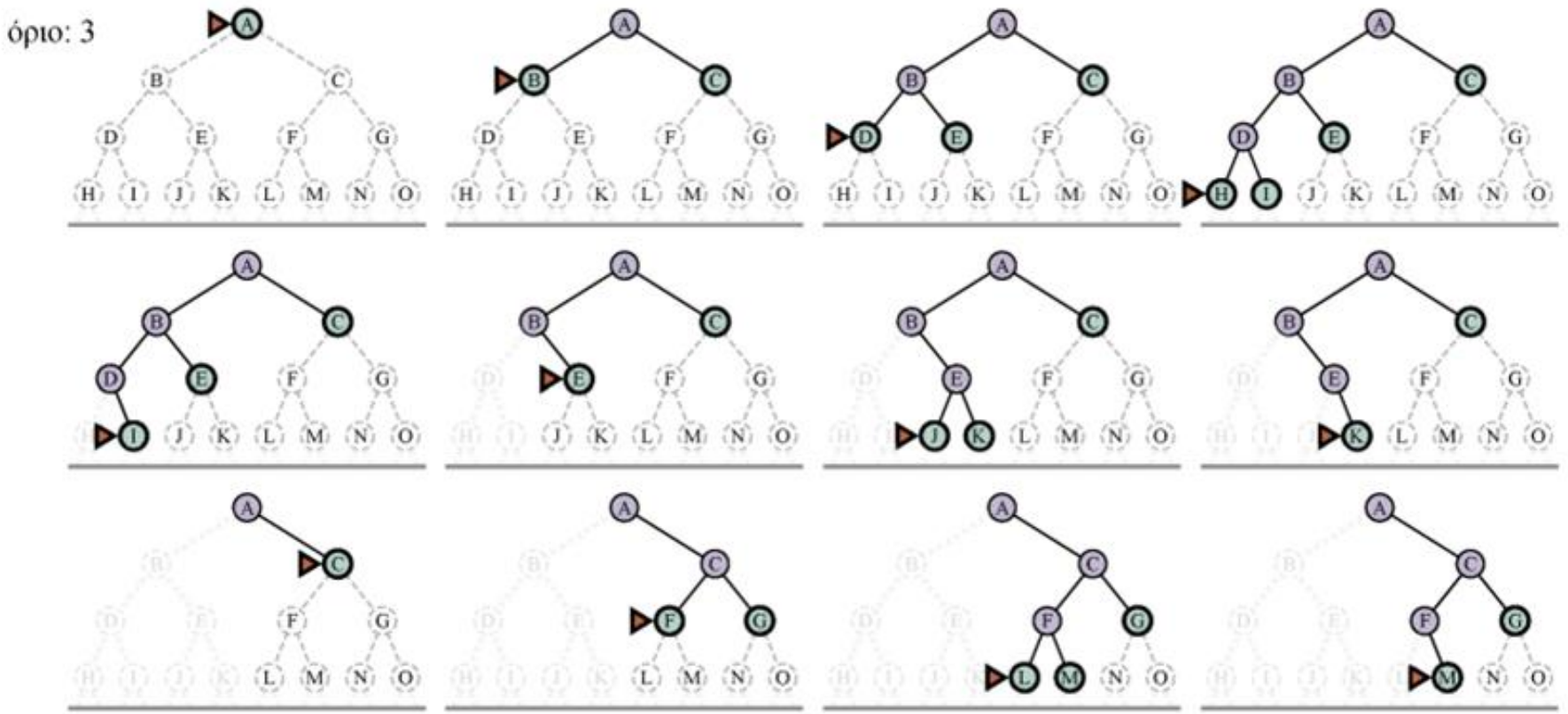
Παράδειγμα



Τυφλή αναζήτηση

- Επαναληπτική αναζήτηση εμβάθυνσης (iterative deepening search)

Παράδειγμα



Τυφλή αναζήτηση

- Επαναληπτική αναζήτηση εμβάθυνσης (iterative deepening search)

Χρονική Πολυπλοκότητα

Οι κόμβοι στο επίπεδο d υπολογίζονται μια φορά.

Οι κόμβοι στο επίπεδο $d-1$ υπολογίζονται δύο φορές.

Οι κόμβοι στο 1ο επίπεδο υπολογίζονται d φορές.

Παρόμοια με αναζήτηση πρώτα κατά πλάτος

Σύνολο παραγόμενων κόμβων

$$N(IDS) = (d)b + (d-1)b^2 + \cdot \cdot \cdot + (1)b^d$$

- Επιπλέον **επαναλαμβανόμενοι υπολογισμοί** για τα άνω επίπεδα (που δεν επιβαρύνουν σημαντικά τον συνολικό υπολογιστικό φόρτο)

Τυφλή αναζήτηση

- Επαναληπτική αναζήτηση εμβάθυνσης (iterative deepening search)

Παράδειγμα

Για

μέγιστο παράγοντα διακλάδωσης του δέντρου αναζήτησης $b = 10$

και

βάθος κόμβου-στόχου $d = 5$

είναι

$$N(IDS) = 50 + 400 + 3000 + 20000 + 100000 = 123450$$
$$N(BFS) = 10 + 100 + 1000 + 10000 + 100000 = 111110$$

Παρόμοια με αναζήτηση πρώτα κατά πλάτος

Η επαναληπτική εμβάθυνση είναι η **προτιμώμενη τυφλή μέθοδος αναζήτησης** όταν ο χώρος αναζήτησης είναι μεγάλος και το βάθος της λύσης δεν είναι γνωστό.

Τυφλή αναζήτηση

Αμφίδρομη αναζήτηση (bidirectional search)

Μεθοδολογία

Δύο παράλληλες αναζητήσεις: (i) Από την αρχική κατάσταση προς τον κόμβο-στόχο και (ii) Από τον κόμβο-στόχο προς την αρχική κατάσταση

Κατασκευάζονται δύο δένδρα, το καθένα με το δικό του σύνορο αναζήτησης.

Μπορεί να χρησιμοποιηθεί οποιοσδήποτε αλγόριθμος αναζήτησης για την κατασκευή κάθε δένδρου, υπό την προϋπόθεση ότι διατηρεί όλα τα κλαδιά στη μνήμη (π.χ. όχι αναζήτηση πρώτα σε βάθος)

- Π.χ., στην αμφίδρομη αναζήτηση πρώτα στο καλύτερο, ο κόμβος που επεκτείνεται είναι πάντα αυτός με την ελάχιστη τιμή της συνάρτησης αξιολόγησης, σε οποιοδήποτε από τα δύο σύνορα.

Στόχος

Οι δύο διαδικασίες να «συναντηθούν» στην μέση του χώρου καταστάσεων.

Η πολυπλοκότητα είναι:

$$O\left(b^{d/2} + b^{d/2}\right) \ll O\left(b^{d}\right)$$

Τυφλή αναζήτηση

- Αμφίδρομη αναζήτηση (bidirectional search)

Παράδειγμα

Πρόβλημα με βάθος λύσης $d = 6$.

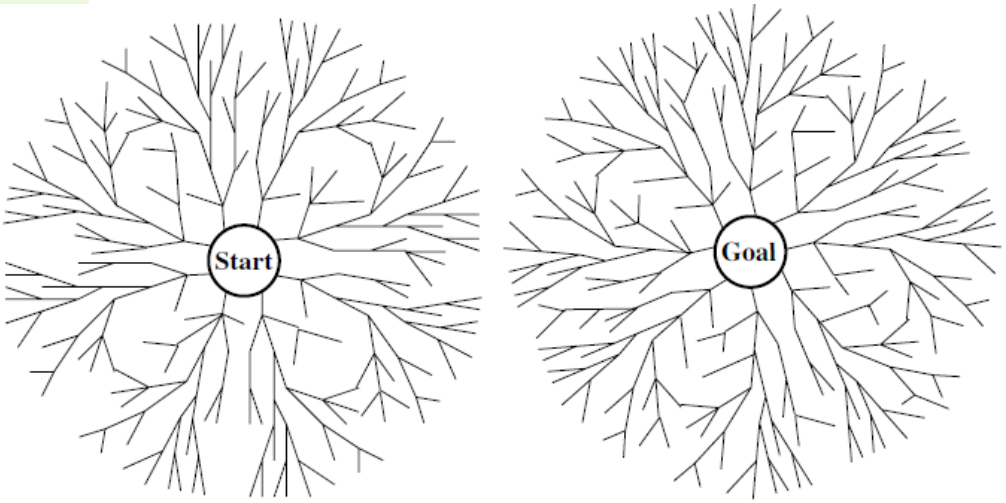
Κάθε κατεύθυνση τρέχει αναζήτηση πρώτα κατά πλάτος.

Συνάντηση σε $d = 3$.

Για $b = 10$ είναι

$$N(breadth - first\ search) = 1111110 \text{ κόμβοι}$$

$$N(bidirectional\ search) = 2220 \text{ κόμβοι}$$



Τυφλή αναζήτηση

Αμφίδρομη αναζήτηση (bidirectional search)

Χαρακτηριστικά

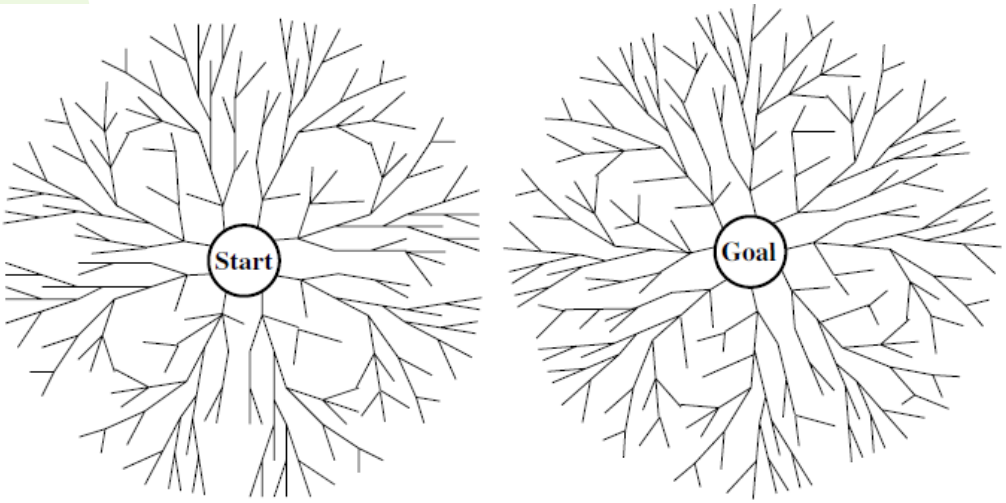
Στην αντίστροφη αναζήτηση απαιτείται ο υπολογισμός των **προκατόχων** ενός κόμβου

- Εύκολο όταν οι ενέργειες στους κόμβους είναι αντιστρέψιμες.

  Π.χ. ένας κόμβος-στόχος (8-puzzle, υπολογισμός διαδρομής)

- Όχι πάντα εφικτό.

  Π.χ. σκάκι



Τυφλή αναζήτηση

➢ Σύγκριση μεθόδων

Κριτήριο	Πρώτα σε πλάτος	Ομοιόμορφου κόστους	Πρώτα σε βάθος	Περιορισμένου βάθους	Επαναληπτικής εμβάθυνσης	Αμφίδρομη (εάν εφαρμόζεται)
Πλήρης;	Ναι[α]	Ναι[α],[β]	Όχι	Όχι	Ναι[α]	Ναι[α],[δ]
Βέλτιστη;	Ναι[γ]	Ναι	Όχι	Όχι	Ναι[γ]	Ναι[γ],[δ]
Χρόνος	$O(b^d)$	$O(b^{1+\lfloor C^*/\varepsilon \rfloor})$	$O(b^m)$	$O(b^l)$	$O(b^d)$	$O(b^{d/2})$
Χώρος	$O(b^d)$	$O(b^{1+\lfloor C^*/\varepsilon \rfloor})$	$O(bm)$	$O(bl)$	$O(bd)$	$O(b^{d/2})$

όπου

b = ο παράγοντας διακλάδωσης
d = το βάθος της ρηχότερης λύσης
m = το μέγιστο βάθος του δέντρου
l = το όριο βάθους
ε = το μικρότερο κόστος βήματος
C^* = το κόστος της βέλτιστης λύσης

[α] **πλήρης** εάν το b είναι πεπερασμένο
[β] **πλήρης** για κόστος βήματος $\geq \varepsilon$ με ε μικρό θετικό αριθμό
[γ] **βέλτιστο** για ενιαίο κόστος βήματος
[δ] εάν και οι δύο κατευθύνσεις είναι αναζητήσεις πρώτα πλάτους

Τυφλή αναζήτηση

Σύνοψη

Διάφορες πρωτότυπες ιδέες αναζήτησης στην Τεχνητή Νοημοσύνη που εμπνέονται από μελέτες επίλυσης ανθρώπινων προβλημάτων, π.χ., παζλ, μαθηματικά και παιχνίδια, αλλά και πολλές άλλες εργασίες ΤΝ απαιτούν την εφαρμογή κάποιας μορφής **αναζήτησης**.

Η **διατύπωση του προβλήματος** συνήθως απαιτεί την απόκρυψη λεπτομερειών του πραγματικού κόσμου (**abstraction**) για τον καθορισμό ενός χώρου καταστάσεων που να μπορεί να διερευνηθεί.

Υπάρχει **ποικιλία στρατηγικών** τυφλής αναζήτησης, η καθεμία με τα ιδιαίτερα χαρακτηριστικά της.

Η επαναληπτική αναζήτηση εμβάθυνσης έχει **γραμμικές** χωρικές και χρονικές απαιτήσεις συγκρινόμενη με άλλους αλγόριθμους.

Πρέπει να αποφεύγονται **ατέρμονοι βρόχοι** και επανάληψη καταστάσεων.