Αναζήτηση Επαναληπτικής Εκβάθυνσης (Iterative Deepening, ID)

- **Ο αλγόριθμος *επαναληπτικής εκβάθυνσης* (*ID*) αποτελεί ένα υβριδικό αλγόριθμο που συνδυάζει τον DFS με τον BFS**

Ο αλγόριθμος ID:

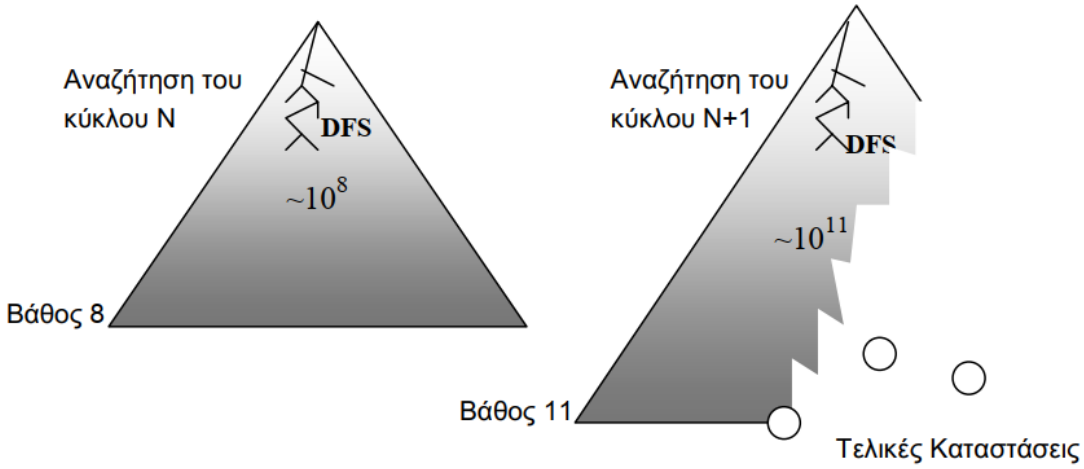
1. Όρισε το αρχικό βάθος αναζήτησης T (συνήθως 1)
2. Εφάρμοσε τον αλγόριθμο DFS μέχρι αυτό το βάθος αναζήτησης
3. Αν έχεις βρει λύση σταμάτησε
4. Αύξησε το βάθος αναζήτησης κατά k (συνήθως κατά 1)
5. Πήγαινε στο βήμα 2

Algorithm ID (Initial, Final, T, k)

```
Begin
    depth ← T;
    do
       result ← boundedDFS(Initial,Final,depth);
       depth ← depth + k;
    while (result = failure);
    return result;
End.
```

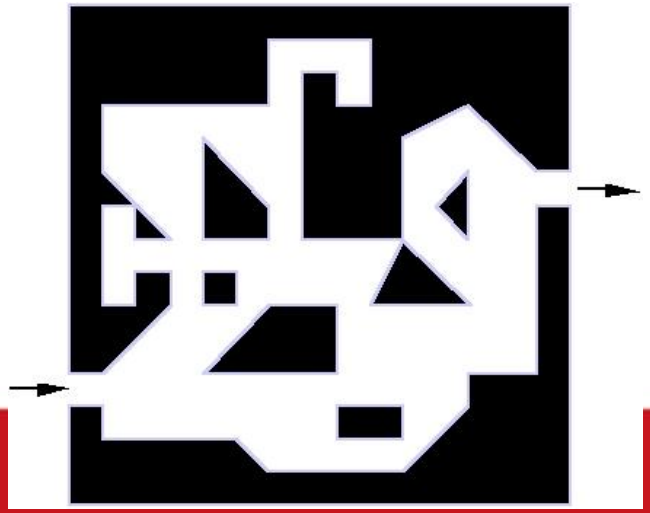
ID - Σχόλια

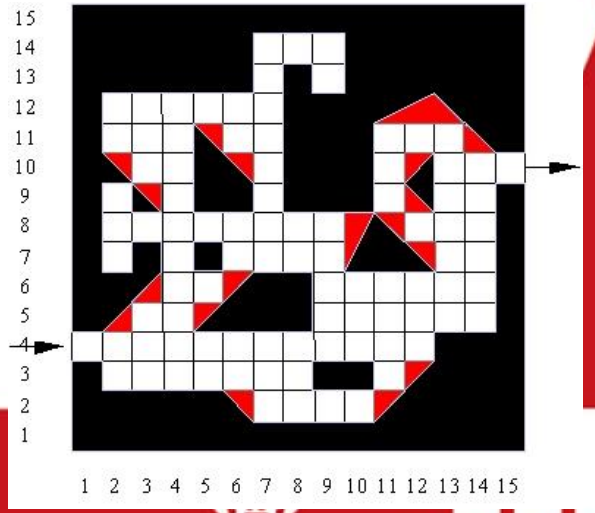
- Μειονεκτήματα:
  - *Δε θυμάται τίποτα από τον προηγούμενο κύκλο αναζήτησης.*
- Πλεονεκτήματα:
  - *Είναι πλήρης.*
  - *Αν το βάθος αυξάνεται κατά 1 σε κάθε κύκλο και ο ID βρει λύση, τότε αυτή η λύση θα είναι η καλύτερη.*
  - *Έχει αποδειχθεί ότι έχει την ίδια πολυπλοκότητα σε χώρο και χρόνο με τους DFS και BFS.*
  - *Δεν κινδυνεύει να χαθεί σε κάποιο κλαδί απείρου μήκους.*



Παράδειγμα 2 – Το πρόβλημα του Λαβύρινθου

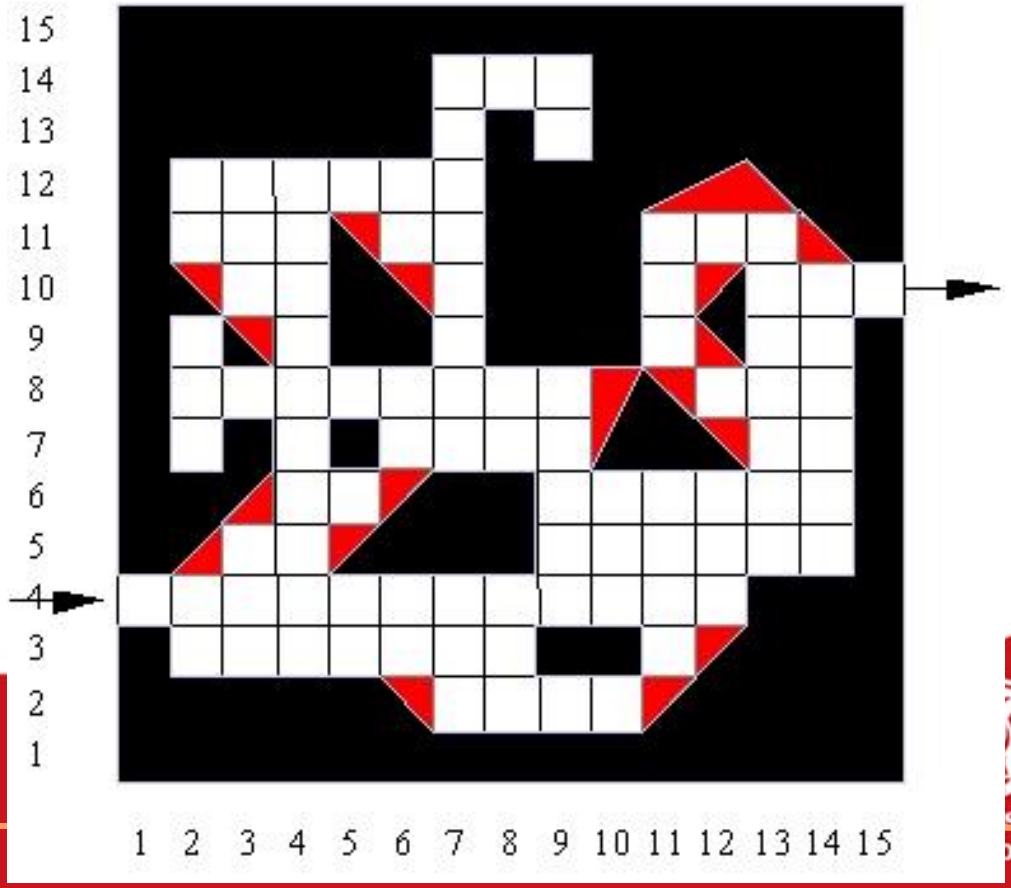
- Για να εφαρμοστούν οι αλγόριθμοι αναζήτησης, πρέπει να οριστεί το πρόβλημα **(I,G,T,S)**
- Το πρώτο βήμα είναι η μετατροπή του χάρτη σε πλέγμα
  - Υπάρχουν διακριτές θέσεις στο χάρτη
  - Μία θέση ορίζεται ως **pos(X,Y)**
  - Μία θέση μπορεί να είναι κενή ή όχι (**clear(X,Y)** ή **¬clear(X,Y)**)
  - Μπορούν να δηλωθούν οι επιτρεπτές κινήσεις



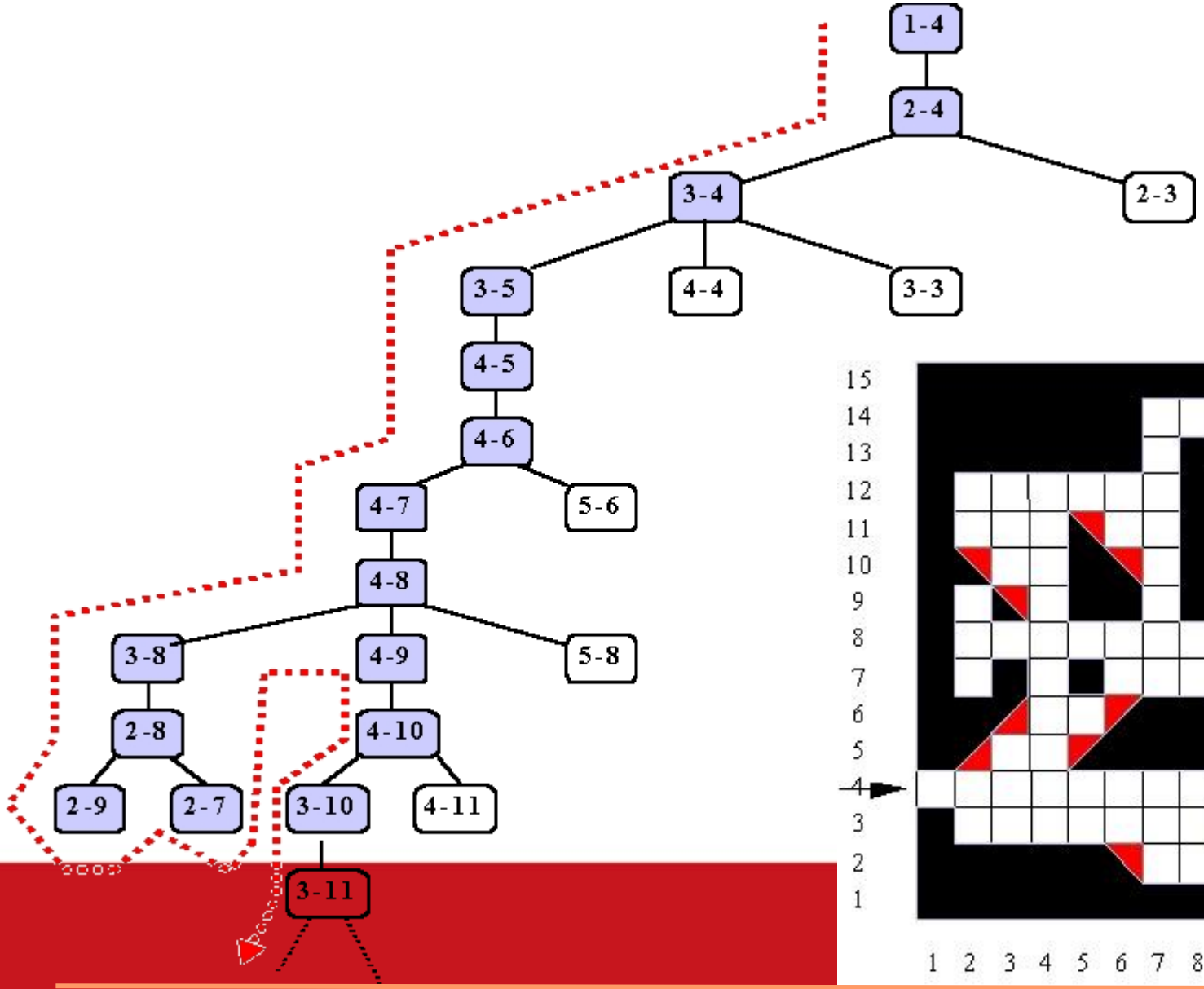


Παράδειγμα 2 – Ορισμός προβλήματος

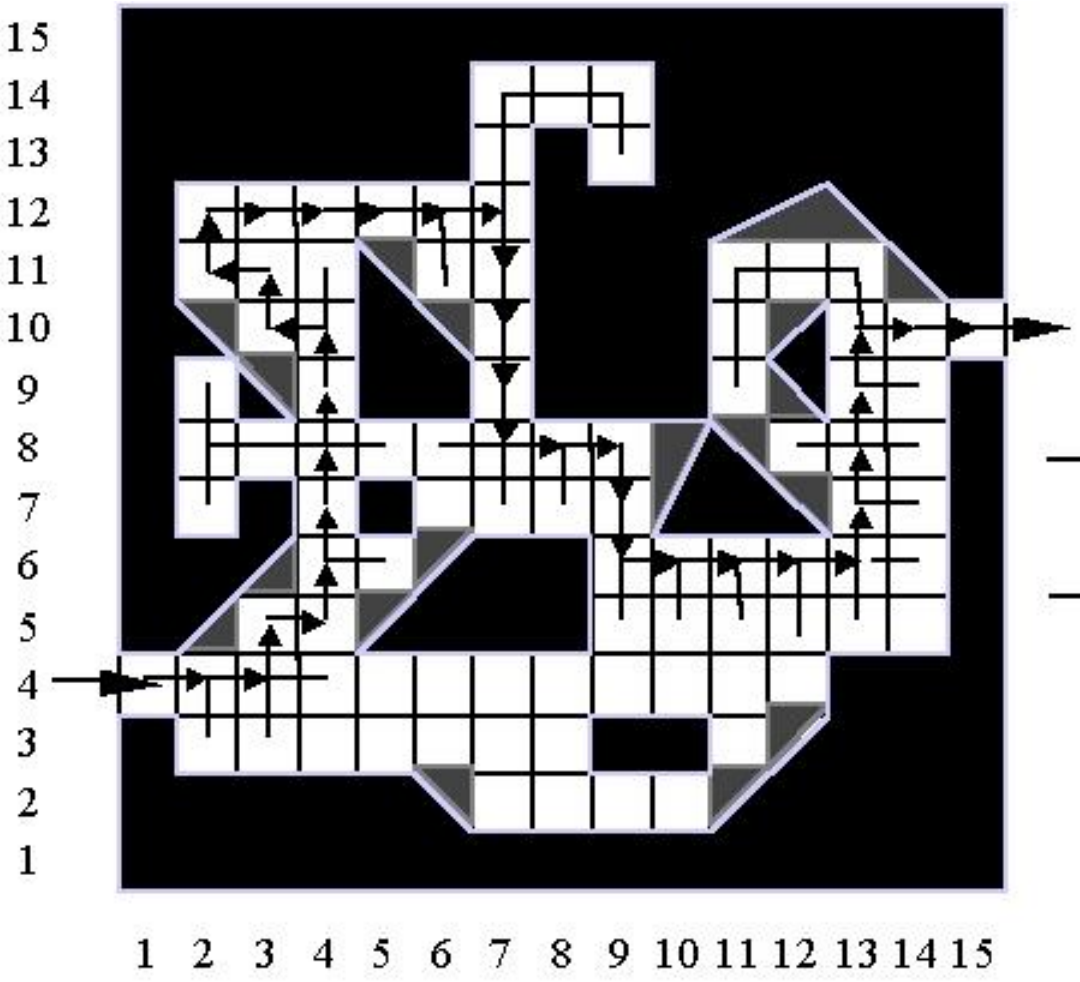
- Πρόβλημα
  - I = pos(1,4)
  - G ={pos(15,10)}
  - T = {MoveLeft, MoveUp, MoveRight, MoveDown}
  - S = {pos(X,Y) : clear(X,Y)} // Το σύνολο των ελεύθερων θέσεων



Παράδειγμα 2 – Εφαρμογή DFS



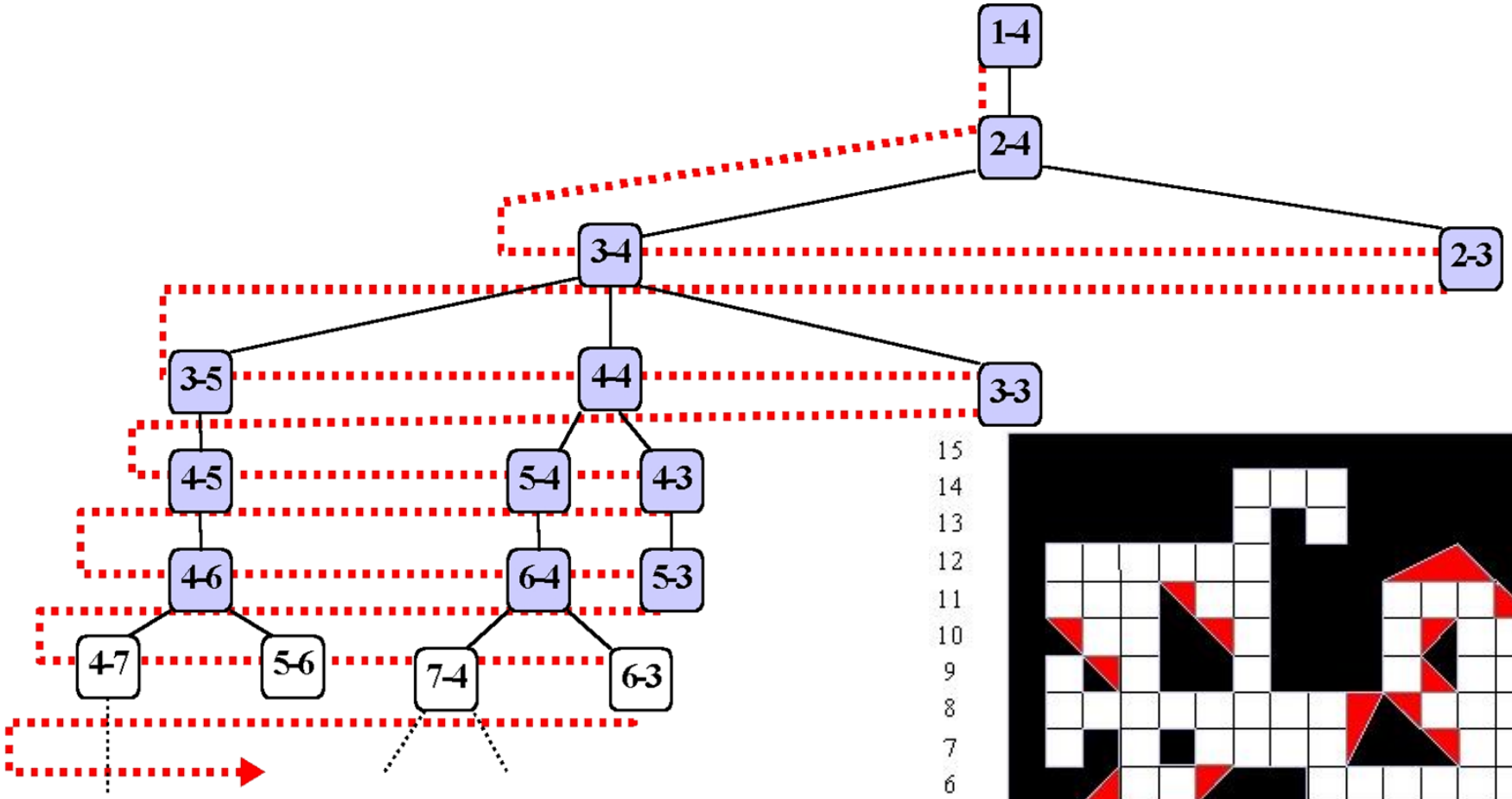
Παράδειγμα 2 – Λύση με DFS

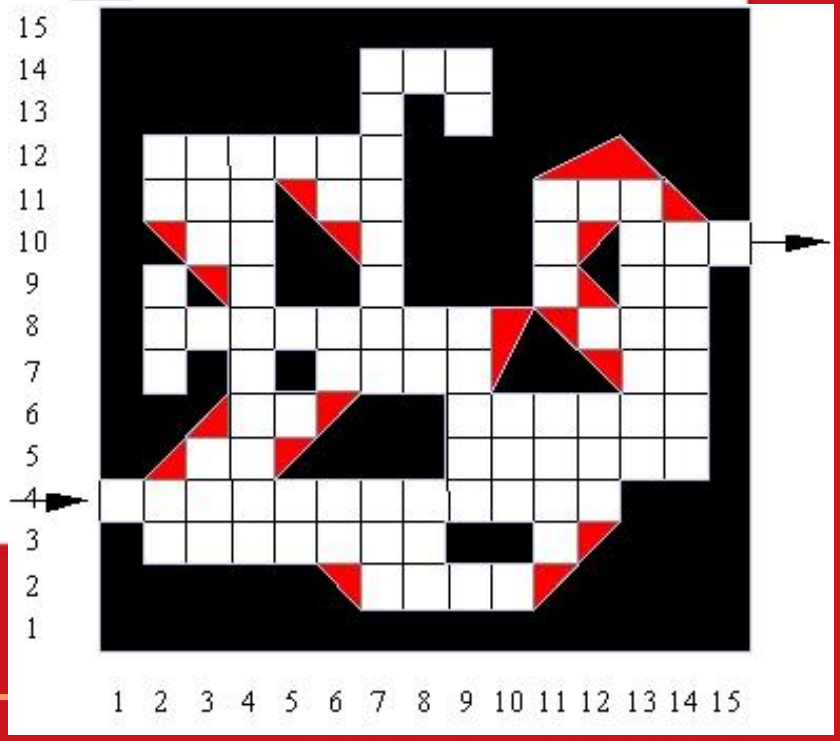


→ Τελεστής που ανήκει στη λύση

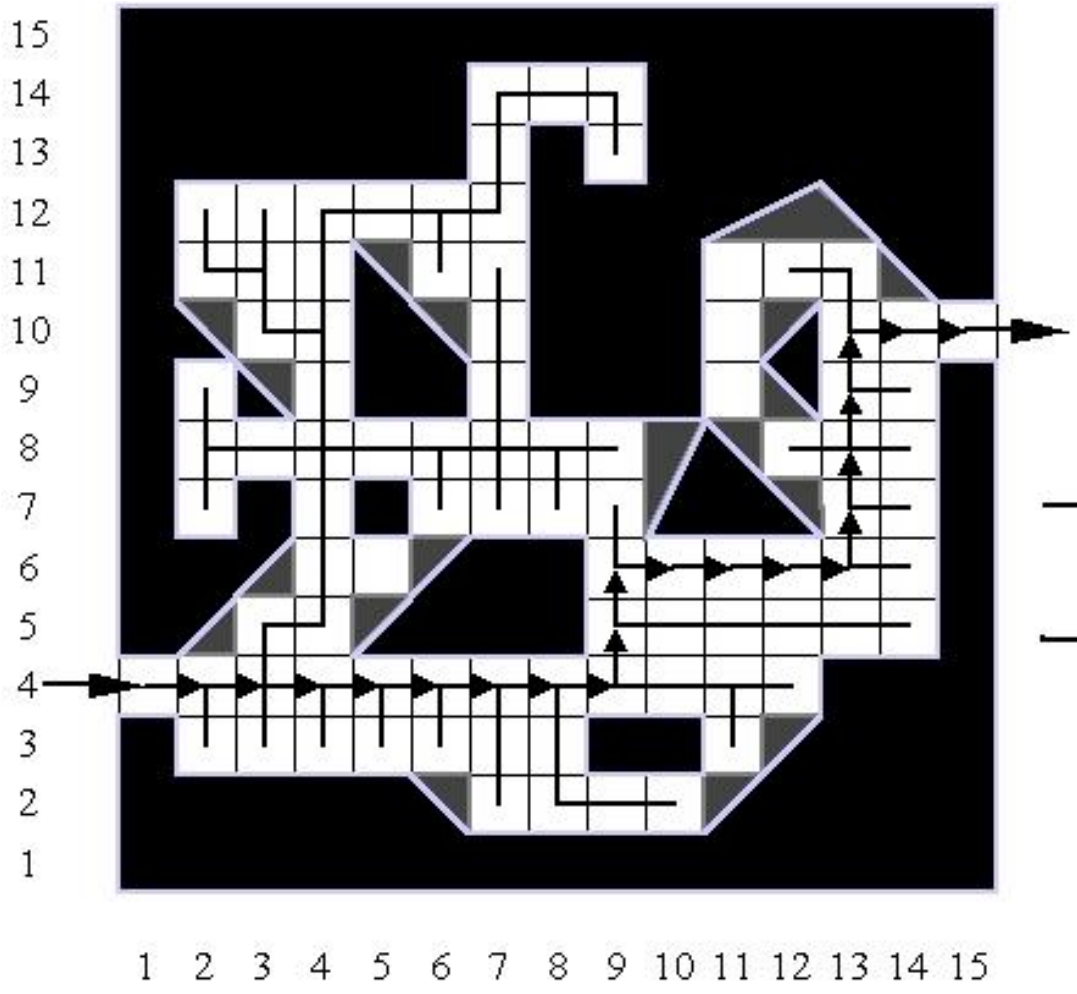
— Τελεστής που δεν ανήκει στη λύση αλλά εξετάζεται κατά την εκτέλεση του αλγορίθμου

Παράδειγμα 2 – Εφαρμογή BFS





Παράδειγμα 2 – Λύση με BFS

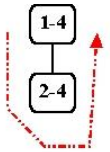


→ Τελεστής που ανήκει στη λύση

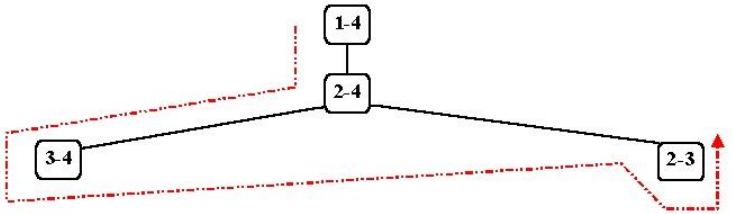
— Τελεστής που δεν ανήκει στη λύση αλλά εξετάζεται κατά την εκτέλεση του αλγορίθμου

Παράδειγμα 2 – Εφαρμογή ID

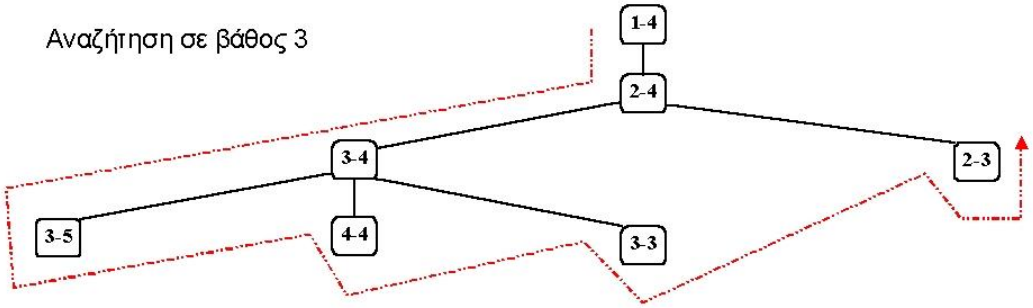
Αναζήτηση σε βάθος 1



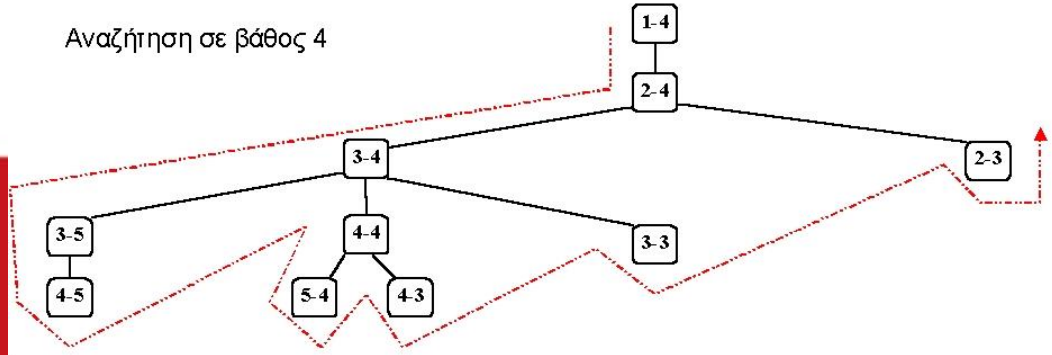
Αναζήτηση σε βάθος 2

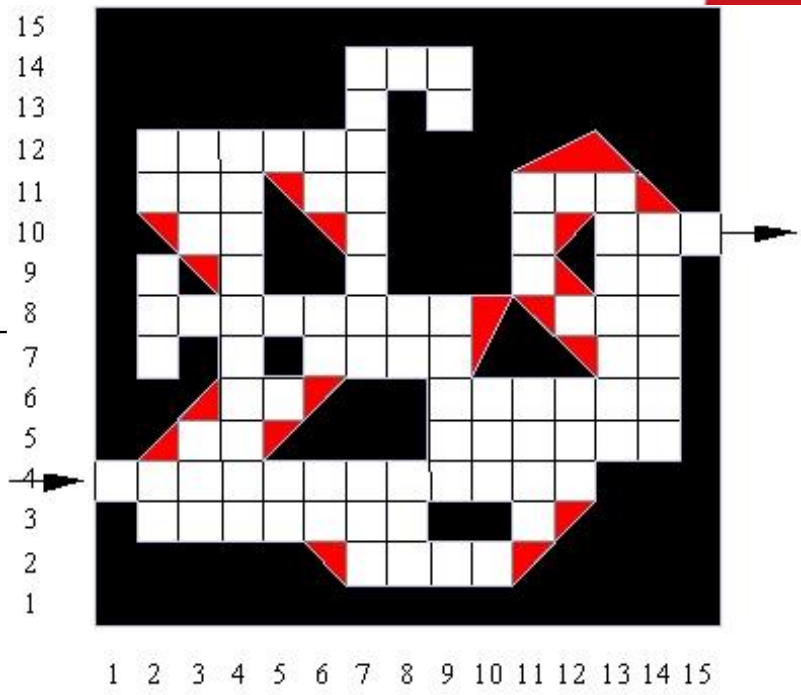


Αναζήτηση σε βάθος 3



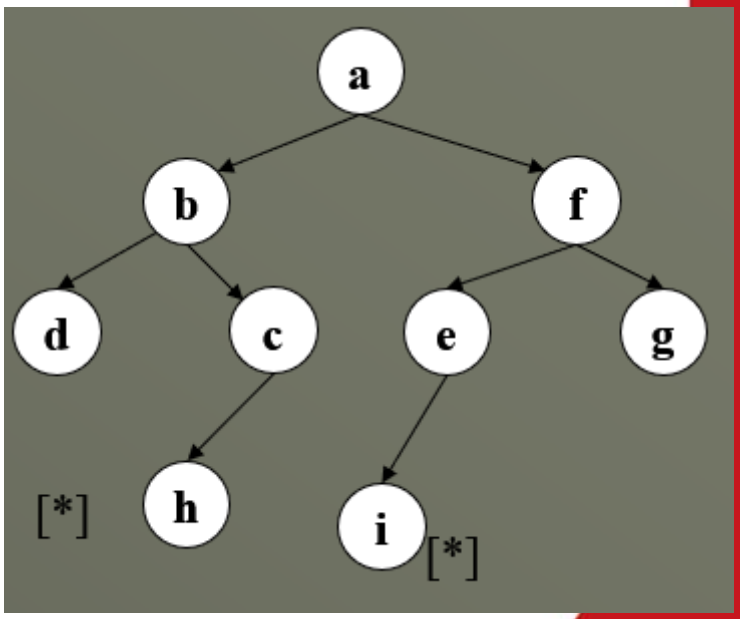
Αναζήτηση σε βάθος 4



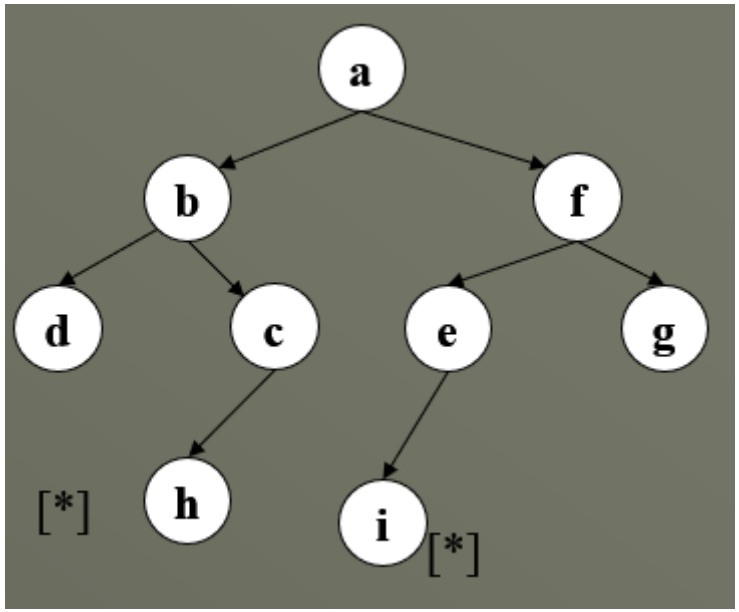


Άσκηση 1

- Στο διπλανό σχήμα παρουσιάζεται ένα δένδρο αναζήτησης. Οι κόμβοι που είναι σημειωμένοι με ένα * , είναι οι τερματικοί κόμβοι της αναζήτησης.
- Γράψτε την σειρά με την οποία θα εξεταστούν οι κόμβοι του δέντρου (π.χ. a,b,c,...) μέχρι να βρεθεί τερματική κατάσταση από τους ακόλουθους αλγορίθμους:
  - DFS
  - BFS
  - ID (με **αρχικό βάθος 1** και **βήμα 1**)



Άσκηση 1 - Λύση



DFS: a → b → d → c → **h** → f → e → **i** → g

BFS: a → b → f → d → c → e → g → **h** → **i**

ID: a → b → f → a → b → d → c → f → e → g → a → b → d → c → **h** → f → e → **i** → g

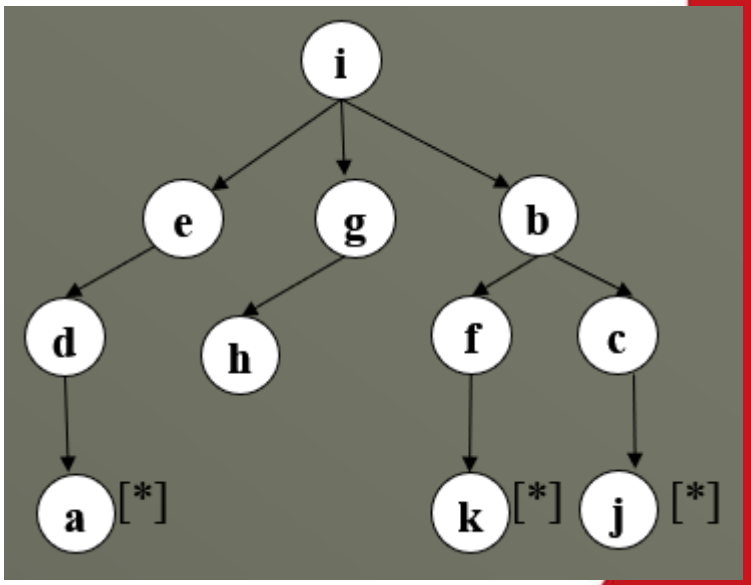
1st iteration *2nd iteration* *3rd iteration*

University of East London

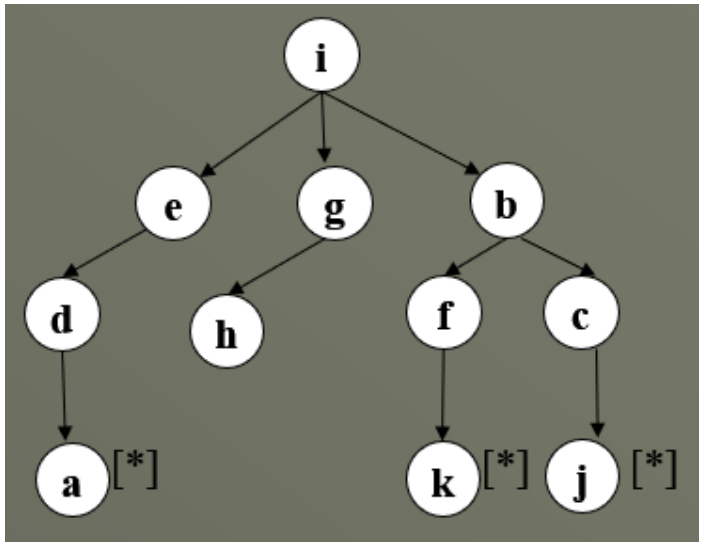
METROPOLITAN COLLEGE

Άσκηση 2

- Στο διπλανό σχήμα παρουσιάζεται ένα δένδρο αναζήτησης, Οι κόμβοι που είναι σημειωμένοι με ένα * , είναι οι τερματικοί κόμβοι της αναζήτησης.
- Γράψτε την σειρά με την οποία θα εξεταστούν οι κόμβοι του δέντρου (π.χ. a,b,c,…) μέχρι να βρεθεί τερματική κατάσταση από τους ακόλουθους αλγορίθμους:
  - DFS
  - BFS
  - ID (με αρχικό βάθος 1 και βήμα 2)



Άσκηση 2 – Λύση



DFS: i → e → d → **a** → g → h → b → f → **k** → c → **j**

BFS: i → e → g → b → d → h → f → c → **a** → **k** → **j**

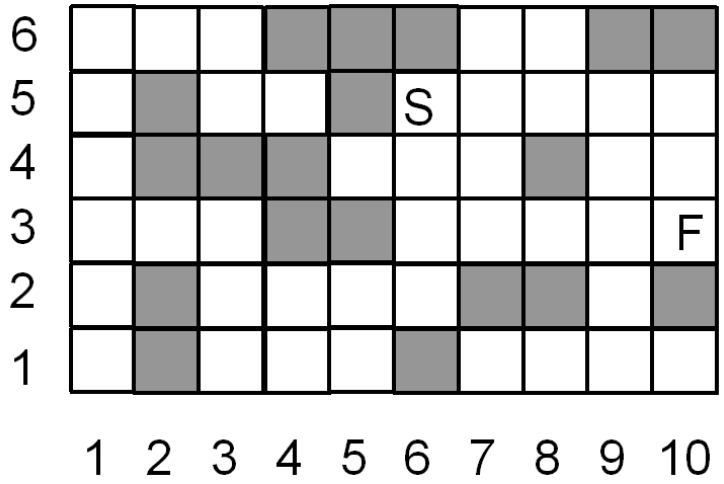
ID *(με αρχικό βάθος 1 και βήμα 2)*:

i → e → g → b → i → e → d → a → g → h → b → f → k → c → j

1st iteration *2nd iteration*

Άσκηση 3

- Εφαρμόστε (μόνο 5 βήματα) τον αλγόριθμο αναζήτησης κατά βάθος (DFS) στο διπλανό πρόβλημα του λαβυρίνθου ξεκινώντας από την αρχική θέση S (τελική θέση είναι η F). **Σημείωση**: Δεν επιτρέπονται διαγώνιες κινήσεις.



Μέτωπο Αναζήτησης	Κλειστό Σύνολο	Κατάσταση	Παιδιά

Περισσότεροι αλγόριθμοι τυφλής αναζήτησης

Αναζήτηση διπλής κατεύθυνσης

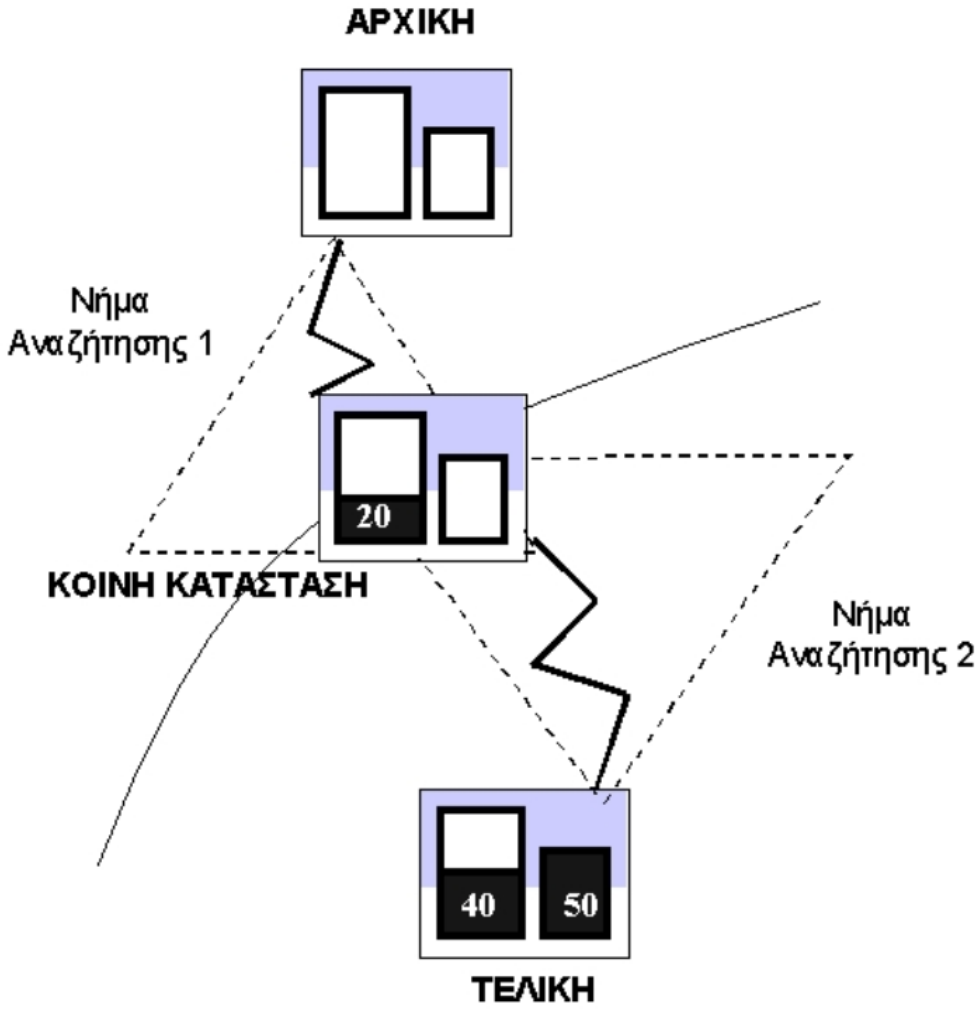
Η ιδέα της *αναζήτησης διπλής κατεύθυνσης* (Bidirectional Search - *BiS*) πηγάζει από τη δυνατότητα του παραλληλισμού (parallelism) στα υπολογιστικά συστήματα.

- Προϋποθέσεις κάτω από τις οποίες μπορεί να εφαρμοστεί:
  - *Οι τελεστές μετάβασης είναι αντιστρέψιμοι (reversible), και*
  - *Είναι πλήρως γνωστή η τελική κατάσταση.*

- Εφαρμογή
  - *Αρχίζει την αναζήτηση από την αρχική και τελική κατάσταση ταυτόχρονα.*
  - *Αν κάποια κατάσταση που επεκτείνεται είναι κοινή, τότε βρέθηκε λύση.*
  - *Λύση είναι η ένωση των μονοπατιών από την κοινή κατάσταση έως την αρχική και έως την τελική κατάσταση.*

- Μειονεκτήματα:
  - *Υπάρχει επιπλέον κόστος επικοινωνίας μεταξύ των δύο αναζητήσεων.*

Αναζήτηση διπλής κατεύθυνσης



Επέκταση και οριοθέτηση (B&B)

Ο *αλγόριθμος επέκτασης και οριοθέτησης* (Branch and Bound - *B&B*) εφαρμόζεται σε προβλήματα όπου αναζητείται η βέλτιστη λύση (ελάχιστο κόστος).

- Ο B&B κλαδεύει καταστάσεις (pruning) και μειώνει το χώρο αναζήτησης.
- Παράδειγμα: βρέθηκε μια λύση με κόστος 159. Κατά την αναζήτηση για άλλες λύσεις μια κατάσταση έχει ήδη κόστος 167. Άρα κλαδεύεται αυτή η κατάσταση.

Ο αλγόριθμος B&B (Branch & Bound)

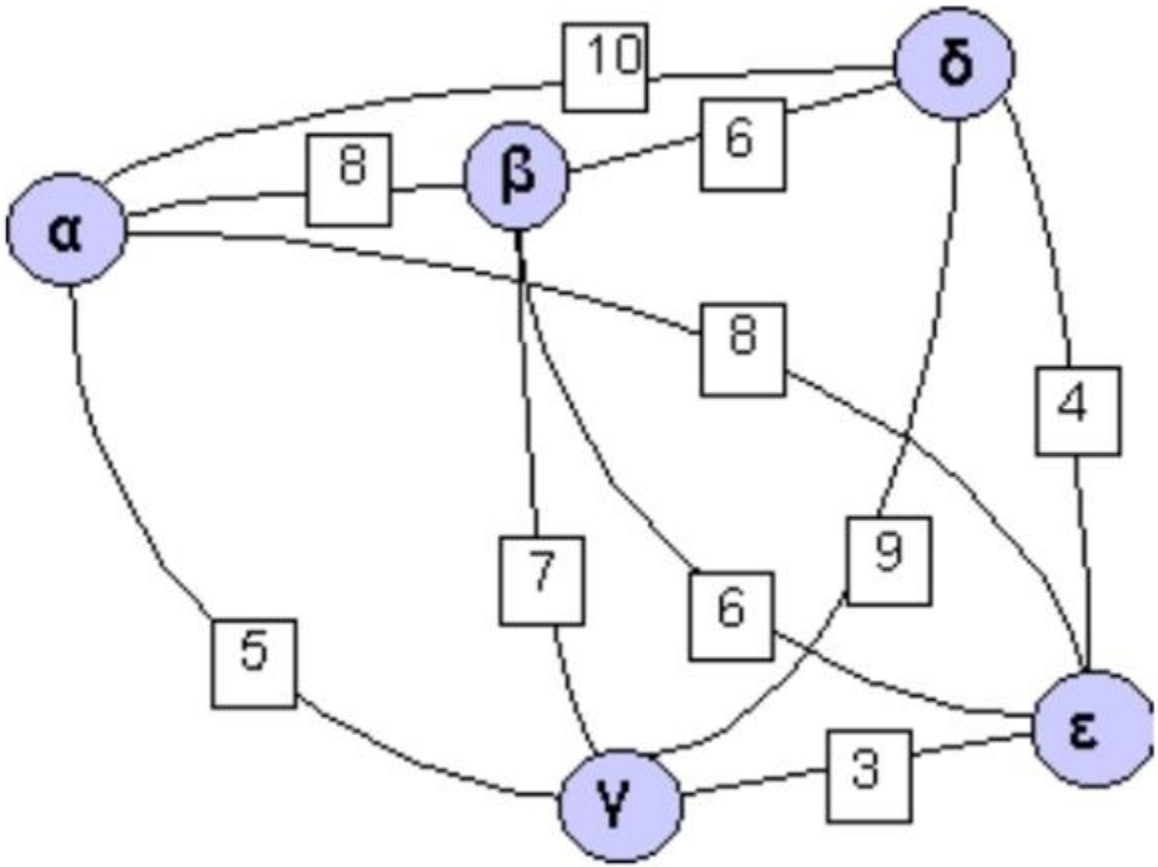
1. Βάλε την αρχική κατάσταση στο μέτωπο της αναζήτησης.
2. Αρχική τιμή της καλύτερης λύσης είναι το +∞ (όριο).
3. Αν το μέτωπο της αναζήτησης είναι κενό, τότε σταμάτησε.
   Η καλύτερη μέχρι τώρα λύση είναι και η βέλτιστη.
4. Βγάλε την πρώτη σε σειρά κατάσταση από το μέτωπο της αναζήτησης.
5. Αν η κατάσταση ανήκει στο κλειστό σύνολο, τότε πήγαινε στο 3.
6. Αν η κατάσταση είναι τελική, τότε ανανέωσε τη λύση ως την καλύτερη μέχρι τώρα και ανανέωσε την τιμή του ορίου με την τιμή που αντιστοιχεί στην τελική κατάσταση. Πήγαινε στο 3.
7. Εφάρμοσε τους τελεστές μεταφοράς για να παράγεις τις καταστάσεις-παιδιά και την τιμή που αντιστοιχεί σε αυτές.
8. Βάλε τις καταστάσεις-παιδιά, των οποίων η τιμή δεν υπερβαίνει το όριο, μπροστά στο μέτωπο της αναζήτησης. (*)
9. Βάλε την κατάσταση-γονέα στο κλειστό σύνολο.
10. Πήγαινε στο 3.

(*) Αυτός είναι DFS-B&B γιατί οι νέες καταστάσεις μπαίνουν μπροστά στο μέτωπο αναζήτησης. Υπάρχει και BestFS-B&B.

Ο αλγόριθμος B&B - Ψευδοκώδικας

```
algorithm b&b(InitialState, FinalStates)
begin
  Closed←∅;
  Frontier←<InitialState>;
  BestCost←∝;
  BestState←null;
  while Frontier≠ ∅ do
    CurrentState←First(Frontier);
    CurrentCost←Cost(Current_State);
    Frontier←delete(CurrentState,Frontier);
    if CurrentCost < BestCost then
  if CurrentState ∈ FinalStates then
        BestState←CurrentState;
        BestCost←CurrentCost;
      else
        Next←Expand(CurrentState);
        ChildrenStates←
        {s| s∈Next ∧ s∉Frontier ∧ s∉Closed};
        Frontier←ChildrenStates ^ Frontier;
        Closed←Closed∪{CurrentState};
      endif;
    endif;
  endwhile;
   if BestState = null
      then return fail
    else return BestState and BestCost;
end.
```

Το πρόβλημα του πλανόδιου πωλητή (Traveling Salesman Problem, TSP)

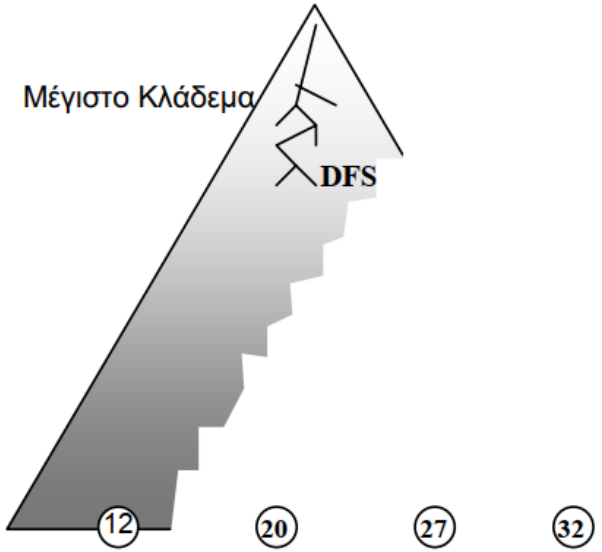


Ο B&B στο πρόβλημα TSP

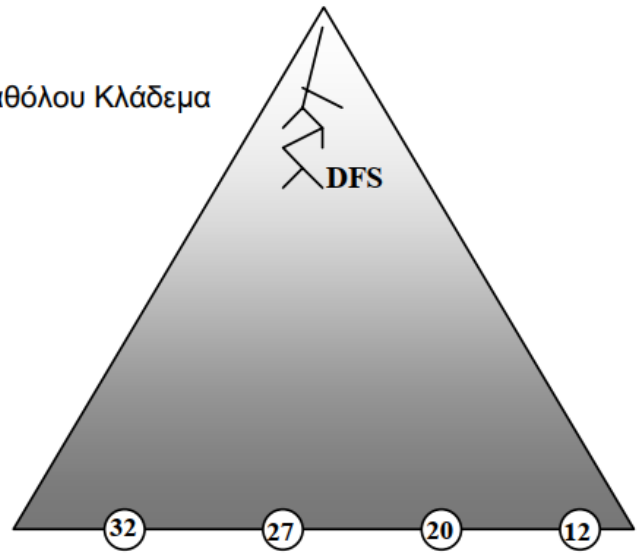
Μέτωπο της αναζήτησης	Κόστος Λύσης	Κατάσταση	Παιδιά
<α>	+∞	α	$αβ^{8}$,$αγ^{5}$,$αδ^{10}$,$αε^{8}$
<$αβ^{8}$,$αγ^{5}$,$αδ^{10}$,$αε^{8}$>	+∞	αβ	$αβγ^{15}$,$αβδ^{14}$, $αβε^{14}$
<$αβγ^{15}$,$αβδ^{14}$,$αβε^{14}$,$αγ^{5}$,...> αδ, αε	+∞	αβγ	$αβγδ^{24}$, $αβγε^{18}$
<$αβγδ^{24}$,$αβγε^{18}$, $αβδ^{14}$,$αβε^{14}$ ...>	+∞	αβγδ	$αβγδε^{28}$
<$αβγδε^{28}$,$αβγε^{18}$, $αβδ^{14}$,...>	+∞	αβγδε	$αβγδεα^{36}$
< $αβγδεα^{36}$, $αβγε^{18}$, $αβδ^{14}$,..>	36	αβγδεα	Τελική Κατάσταση
<$αβγε^{18}$, $αβδ^{14}$,... >	36	αβγε	$αβγεδ^{22}$
<$αβγεδ^{22}$,$αβδ^{14}$,...>	36	αβγεδ	$αβγεδα^{32}$
< $αβγεδα^{32}$,$αβδ^{14}$,$αβε^{14}$ ...>	32	$αβγεδα^{32}$	Τελική Κατάσταση
...	...	...	...
<$αβδεγα^{26}$, ...>	26	αβδεγα	Τελική Κατάσταση
...	...	...	...
<$αβεγδ^{26}$, ...>	26	αβεγδ	Κλάδεμα
....	...	...	...
<$αεβγδ^{30}$, ...>	26	αεβγδ	Κλάδεμα
. . .	. . .	. . .	. . .
<>	ΕλάχιστηΤιμή	ΤΕΛΟΣ	

Ο αλγόριθμος B&B

- Ο B&B εφαρμόζεται όταν υπάρχει μια πραγματική εκτίμηση του κόστους.
- Το κέρδος από το κλάδεμα εξαρτάται από το πόσο γρήγορα θα βρεθεί μια καλή λύση.
- Υπάρχει περίπτωση να μη γίνει καθόλου κλάδεμα αν οι λύσεις είναι διατεταγμένες από τη χειρότερη προς την καλύτερη.
- Στη χειρότερη περίπτωση συμπεριφέρεται σαν τον DFS.



Τελικές Καταστάσεις Διατεταγμένες Ορθά



Τελικές Καταστάσεις Διατεταγμένες Αντίστροφα

Πηγές

- **“Τεχνητή Νοημοσύνη – Β έκδοση»: Ι. Βλαχάβας, Π. Κεφαλάς, Ν. Βασιλειάδης, Φ. Κόκκορας, Η. Σακελλαρίου**