

Κεφάλαιο 4

Αλγόριθμοι Ευριστικής Αναζήτησης

Τεχνητή Νοημοσύνη - Β' Έκδοση

Ι. Βλαχάβας, Π. Κεφαλάς, Ν. Βασιλειάδης, Φ. Κόκκορας, Η. Σακελλαρίου



Αλγόριθμοι Ευριστικής Αναζήτησης

Εισαγωγικά (1/2)

- ❖ Ο χώρος αναζήτησης συνήθως αυξάνεται εκθετικά. Απαιτείται πληροφορία για αξιολόγηση των καταστάσεων (ευριστικός μηχανισμός).
- ❖ Οι αλγόριθμοι που εκμεταλλεύονται τέτοια πληροφορία ονομάζονται **Αλγόριθμοι Ευριστικής Αναζήτησης**.
- ❖ Παράδειγμα ευριστικής αναζήτησης είναι η συναρμολόγηση ενός puzzle.
- ❖ Αν δεν υπήρχαν ευριστικοί μηχανισμοί, τότε τα προβλήματα αυτά θα λύνονταν πολύ δύσκολα, γιατί οι συνδυασμοί που πρέπει να γίνουν είναι πάρα πολλοί.
- ❖ Ο ευριστικός μηχανισμός εξαρτάται από τη γνώση που έχουμε για το πρόβλημα.

Ευριστικός μηχανισμός

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 ευριστική συνάρτηση (heuristic function).
- ❖ Ευριστική τιμή (heuristic value) είναι η τιμή της ευριστικής συνάρτησης και εκφράζει το πόσο κοντά βρίσκεται μία κατάσταση σε μία τελική.
- ❖ 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 (1/3)

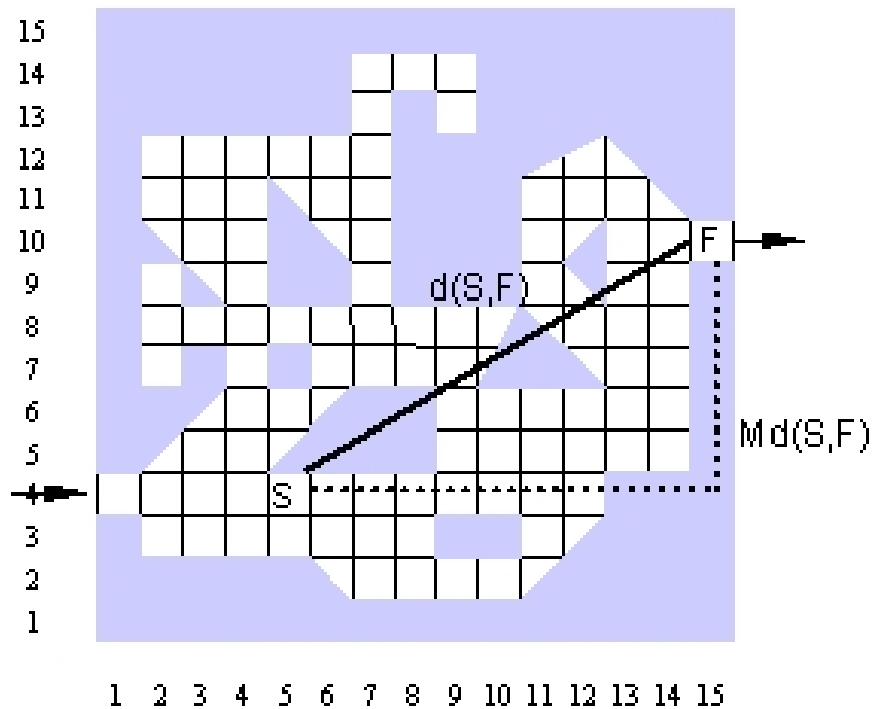
Ευριστικός μηχανισμός και συναρτήσεις σε λαβύρινθο

- ❖ Ευκλείδεια απόσταση (Euclidian distance):

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

- ❖ Απόσταση Manhattan (Manhattan distance):

$$Md(S, F) = |X_S - X_F| + |Y_S - Y_F|$$



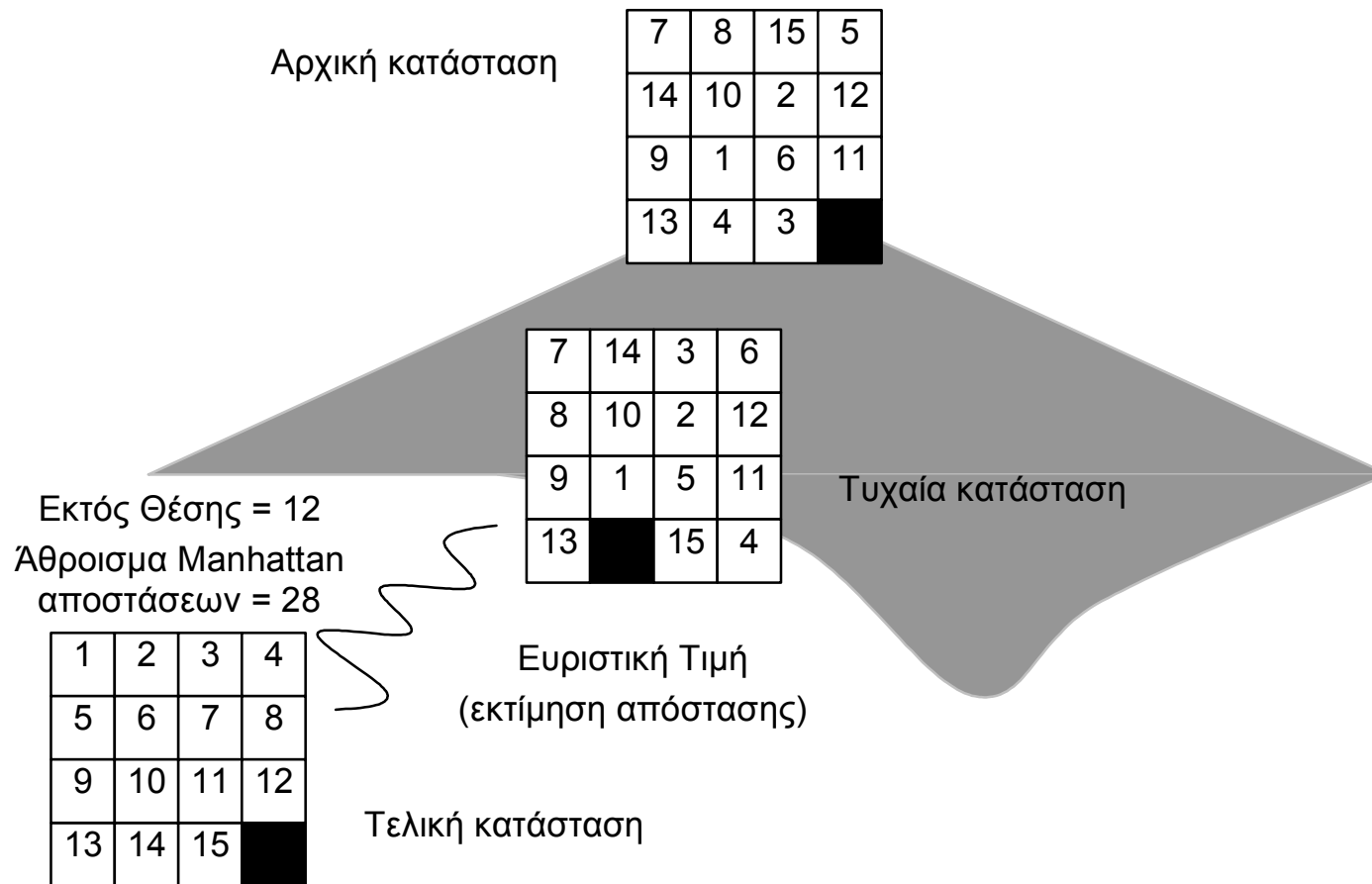
$$d(S, F) = \sqrt{(5 - 15)^2 + (4 - 10)^2} = \sqrt{(100 + 36)} = 11,6$$

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

Ευριστικές Συναρτήσεις σε Μικρά Προβλήματα (2/3)

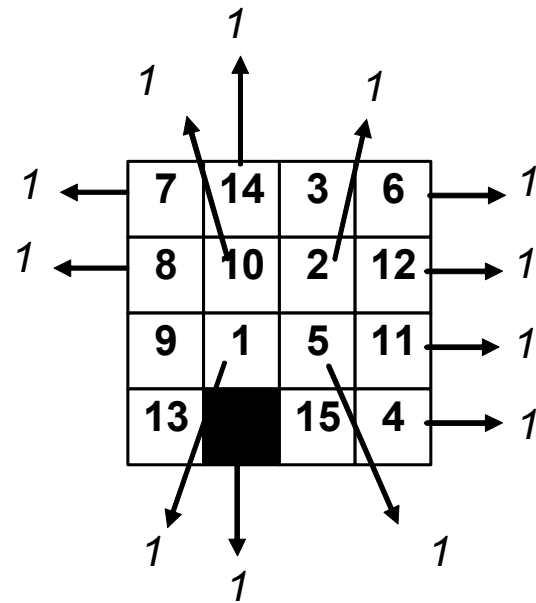
Ευριστικός μηχανισμός και συναρτήσεις στο N-Puzzle

- ❖ Πόσα πλακίδια βρίσκονται εκτός θέσης.
- ❖ Το άθροισμα των αποστάσεων Manhattan κάθε πλακιδίου από την τελική του θέση.

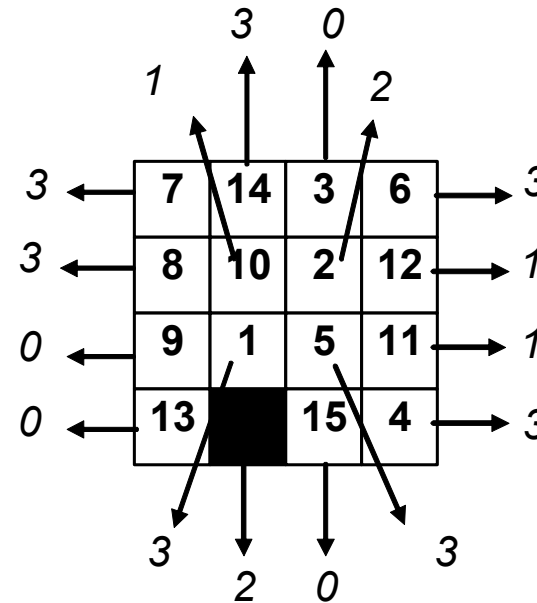


Ευριστικές Συναρτήσεις σε Μικρά Προβλήματα (3/3)

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



Εκτός θέσης = 12



Άθροισμα αποστάσεων Manhattan = 28

Ευριστικός μηχανισμός και συναρτήσεις στο TSP

- ❖ Η κοντινότερη πόλη έχει περισσότερες πιθανότητες να οδηγήσει σε μία συνολικά καλή λύση.

Αναζήτηση με Αναρρίχηση Λόφων


Η αναρρίχηση λόφων (Hill-Climbing Search - HC) είναι ένας αλγόριθμος αναζήτησης που μοιάζει πολύ με τον DFS.

Ο αλγόριθμος HC

1. Η αρχική κατάσταση είναι η τρέχουσα κατάσταση.
2. Αν η κατάσταση είναι μία τελική τότε ανέφερε τη λύση και σταμάτησε.
3. Εφάρμοσε τους τελεστές μετάβασης για να βρεις τις καταστάσεις-παιδιά.
4. Βρες την καλύτερη κατάσταση σύμφωνα με την ευριστική συνάρτηση.
5. Η καλύτερη κατάσταση γίνεται η τρέχουσα κατάσταση.
6. Πήγαινε στο βήμα 2.

Ο αλγόριθμος HC (Ψευδοκώδικας)

```
algorithm hc(InitialState, FinalState)
begin
  CurrentState ← InitialState;
  while CurrentState ≠ FinalState do
    Children ← Expand(CurrentState);
    if Children = ∅ then return failure;
    EvaluatedChildren ← Heuristic(Children);
    bestChild ← best(EvaluatedChildren);
    if hValue(CurrentState) ≥ hValue(bestChild)
      then return failure;
      else CurrentState ← bestChild;
    endif;
  endwhile;
  return success;
end.
```

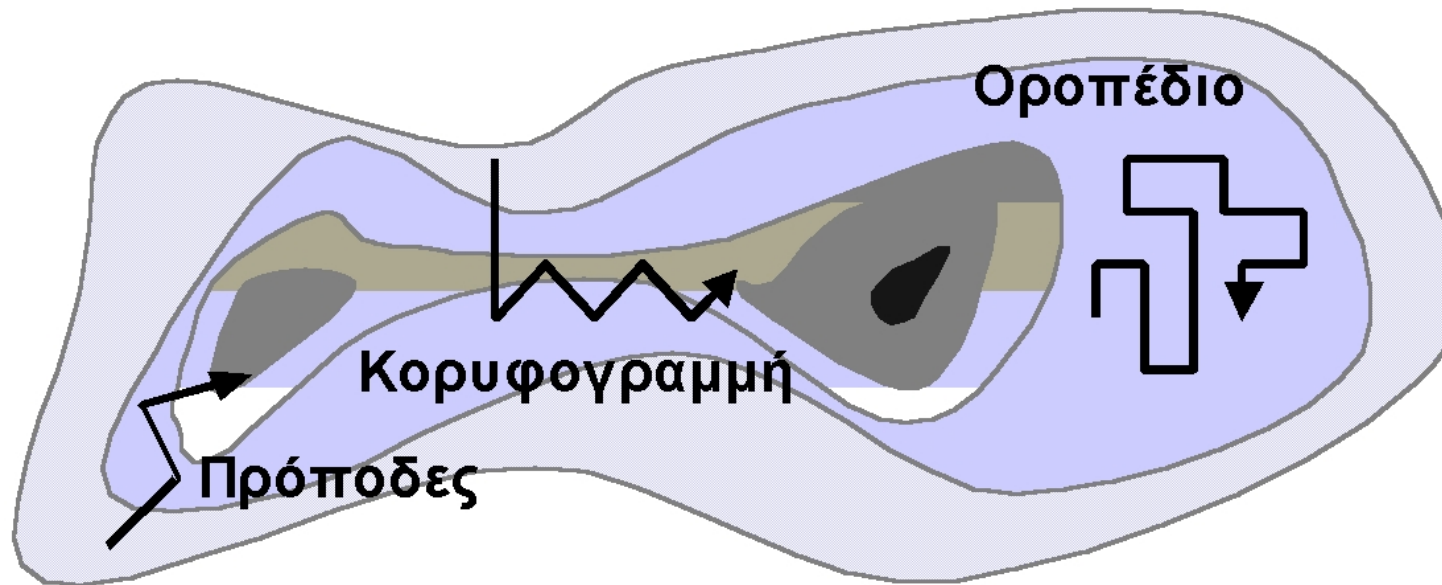
Ο αλγόριθμος HC

Σχόλια (1/2)

- ❖ Ο HC χρησιμοποιείται σε προβλήματα όπου πρέπει να βρεθεί μία λύση πολύ γρήγορα, έστω και αν αυτή δεν είναι η καλύτερη.
- ❖ Πλεονεκτήματα:
 - ❑ Πολύ αποδοτικός και σε χρόνο και σε μνήμη,
- ❖ Μειονεκτήματα:
 - ❑ Είναι μη-πλήρης.
 - ❑ Βασικά προβλήματα του HC:
 - ✓ Πρόποδες (foothill).
 - ✓ Οροπέδιο (plateau).
 - ✓ Κορυφογραμμή (ridges).

Ο αλγόριθμος ΗC

Σχόλια (2/2)



❖ Βελτιώσεις:

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

Ακτινωτή Αναζήτηση

Στον αλγόριθμο ακτινωτής αναζήτησης (*Beam Search - BS*) δεν κλαδεύονται όλες οι υπόλοιπες καταστάσεις όπως στον HC, αλλά ένας σταθερός αριθμός από τις καλύτερες από αυτές κρατείται στο μέτωπο αναζήτησης.

Αναζήτηση Πρώτα στο Καλύτερο

Ο αλγόριθμος αναζήτηση πρώτα στο καλύτερο (Best-First - *BestFS*) κρατά όλες τις καταστάσεις στο μέτωπο αναζήτησης.

Ο αλγόριθμος BestFS

1. Βάλε την αρχική κατάσταση στο μέτωπο αναζήτησης.
2. Αν το μέτωπο αναζήτησης είναι κενό τότε σταμάτησε.
3. Πάρε την πρώτη σε σειρά κατάσταση από το μέτωπο αναζήτησης.
4. Αν η κατάσταση είναι μέλος του κλειστού συνόλου τότε πήγαινε στο 2.
5. Αν η κατάσταση είναι μία τελική τότε ανέφερε τη λύση και σταμάτα.
6. Εφάρμοσε τους τελεστές μεταφοράς για να παράγεις τις καταστάσεις-παιδιά.
7. Εφάρμοσε την ευριστική συνάρτηση σε κάθε παιδί.
8. Βάλε τις καταστάσεις-παιδιά στο μέτωπο αναζήτησης.
9. Αναδιάταξε το μέτωπο αναζήτησης, έτσι ώστε η κατάσταση με την καλύτερη ευριστική τιμή να είναι πρώτη.
10. Βάλε τη κατάσταση-γονέα στο κλειστό σύνολο.
11. Πήγαινε στο βήμα 2.

Ο αλγόριθμος BestFS (Ψευδοκώδικας)

```
algorithm bestfs(InitialState, FinalStates)
begin
  Closed ← ∅;
  EvaluatedInitialState ← Heuristic(<InitialState>)
  Frontier ← <EvaluatedInitialState>;
  CurrentState ← best(Frontier);
  while CurrentState ∉ FinalStates do
    Frontier ← delete(CurrentState, Frontier);
    if CurrentState ∉ ClosedSet then
      begin
        Children ← Expand(CurrentState);
        EvaluatedChildren ← Heuristic(Children);
        Frontier ← Frontier ^ EvaluatedChildren;
        Closed ← Closed ∪ {CurrentState};
      end;
    if Frontier = ∅ then return fail;
    CurrentState ← best(Frontier);
  endwhile;
  return success;
end.
```

Ο αλγόριθμος BestFS

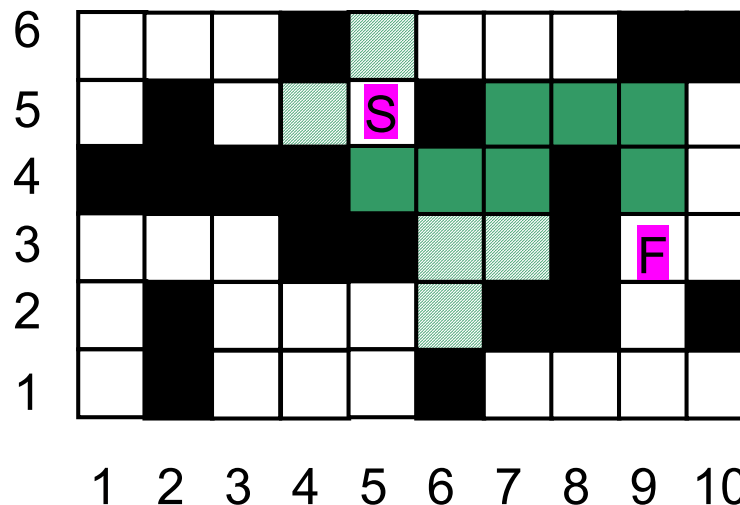
Σχόλια

❖ Πλεονεκτήματα:

- ❑ Προσπαθεί να δώσει μια γρήγορη λύση σε κάποιο πρόβλημα. Εξαρτάται πολύ από τον ευριστικό μηχανισμό.
- ❑ Είναι πλήρης.

❖ Μειονεκτήματα:

- ❑ Το μέτωπο αναζήτησης μεγαλώνει με υψηλό ρυθμό και μαζί του ο χώρος που χρειάζεται για την αποθήκευσή του.
- ❑ Δεν εγγυάται ότι η λύση που θα βρεθεί είναι η βέλτιστη.



Ο αλγόριθμος BestFS: το πρόβλημα του λαβύρινθου

Μέτωπο Αναζήτησης	Κλειστό Σύνολο	Κατάσταση	Παιδιά
<5-5>	◇	5-5	5-4 ⁵ ,5-6 ⁷ ,4-5 ⁷
<5-4 ⁵ ,5-6 ⁷ ,4-5 ⁷ >	<5-5>	5-4	5-5 ⁶ ,6-4 ⁴
<6-4 ⁴ ,5-5 ⁶ ,5-6 ⁷ ,4-5 ⁷ >	<5-5,5-4>	6-4	5-4 ⁷ ,6-3 ³ ,7-4 ³
<6-3 ³ ,7-4 ³ ,5-5 ⁶ ,5-6 ⁷ ,...>	<5-5,5-4,6-4>	6-3	6-4 ⁴ ,6-2 ³ ,7-3 ²
<7-3 ² ,6-2 ³ ,7-4 ³ ,6-4 ⁴ ,5-5 ⁶ ,...>	<5-5,5-4,...>	7-3	6-3 ³ ,6-4 ⁴
<6-3 ³ ,6-2 ³ ,7-4 ³ ,6-4 ⁴ ,5-5 ⁶ ,...>	<...,6-3,...>	6-3	Βρόχος
<6-2 ³ ,7-4 ³ ,6-4 ⁴ ,5-5 ⁶ ,5-6 ⁷ ,...>	<...>	6-2	5-2 ⁵ ,6-3 ³
<7-4 ³ ,6-4 ⁴ ,5-2 ⁵ ,...>	<...>	7-4	7-5 ⁴ ,6-4 ⁴ ,7-3 ²
<7-3 ² ,7-5 ⁴ ,6-4 ⁴ ,5-2 ⁵ ,...>	<...,7-3,...>	7-3	Βρόχος
<4-5 ⁴ ,6-4 ⁴ ,5-2 ⁵ ,...>	<...>	7-5	7-4 ³ ,8-5 ³ ,7-6 ⁵
<8-5 ³ ,7-4 ³ ,6-4 ⁴ ,...>	<...>	8-5	8-6 ⁴ ,7-5 ⁴ ,9-5 ²
<9-5 ² ,7-4 ³ ,6-4 ⁴ ,8-6 ⁴ ,...>	<...>	9-5	8-5 ³ ,9-4 ¹
<9-4 ¹ ,8-5 ³ ,7-4 ³ ,...>	<...>	9-4	9-3 ⁰ ,9-5 ² ,10-4 ²
<9-3 ⁰ ,9-5 ² ,10-4 ² ,...>	<...>	9-3	ΤΕΛΙΚΗ ΚΑΤΑΣΤΑΣΗ
		ΤΕΛΟΣ	

- ❖ Η λύση στο παραπάνω πρόβλημα είναι η διαδρομή που ορίζεται από τη σειρά των θέσεων: $5^5 \rightarrow 5^4 \rightarrow 6^4 \rightarrow 7^4 \rightarrow 7^5 \rightarrow 8^5 \rightarrow 9^5 \rightarrow 9^4 \rightarrow 9^3$.

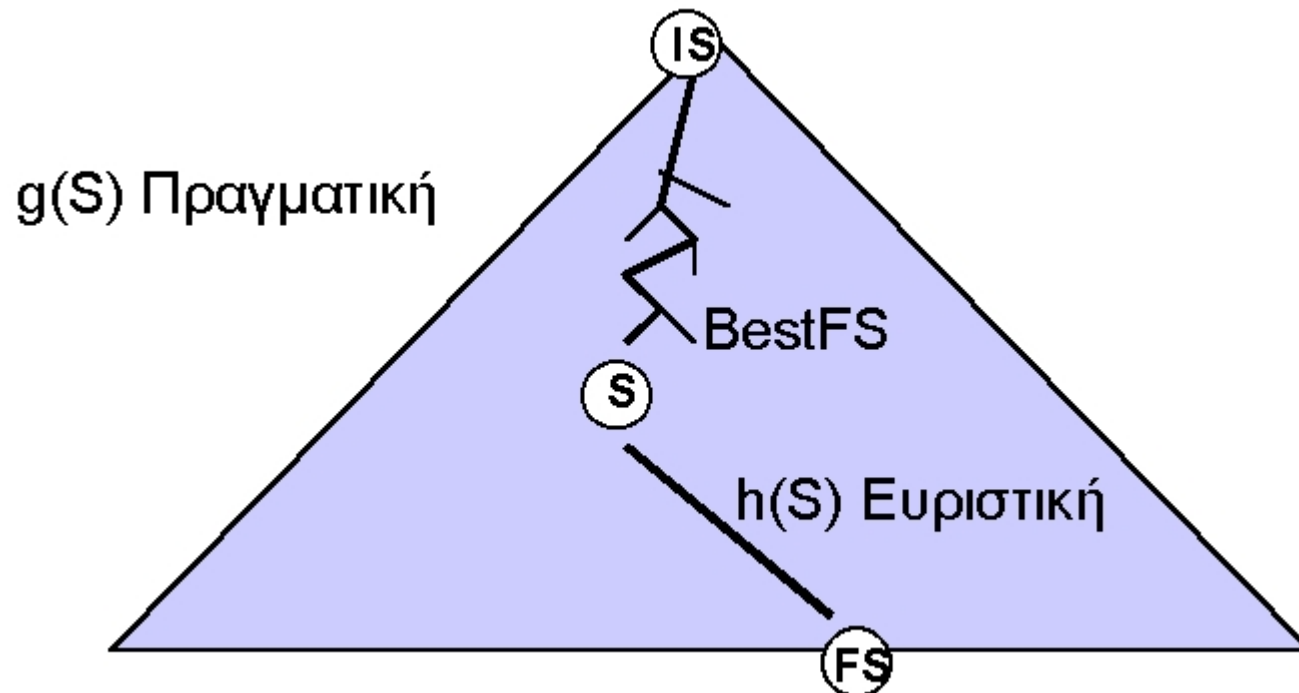
Ο Αλγόριθμος Άλφα-Άστρο (A*)

Ο αλγόριθμος A (Άλφα Άστρο) είναι κατά βάσει BestFS, αλλά με ευριστική συνάρτηση:

$$F(S) = g(S) + h(S)$$

η $g(S)$ δίνει την απόσταση της S από την αρχική κατάσταση, η οποία είναι πραγματική και γνωστή, και

η $h(S)$ δίνει την εκτίμηση της απόστασης της S από την τελική κατάσταση μέσω μιας ευριστικής συνάρτησης, όπως ακριβώς στον BestFS.



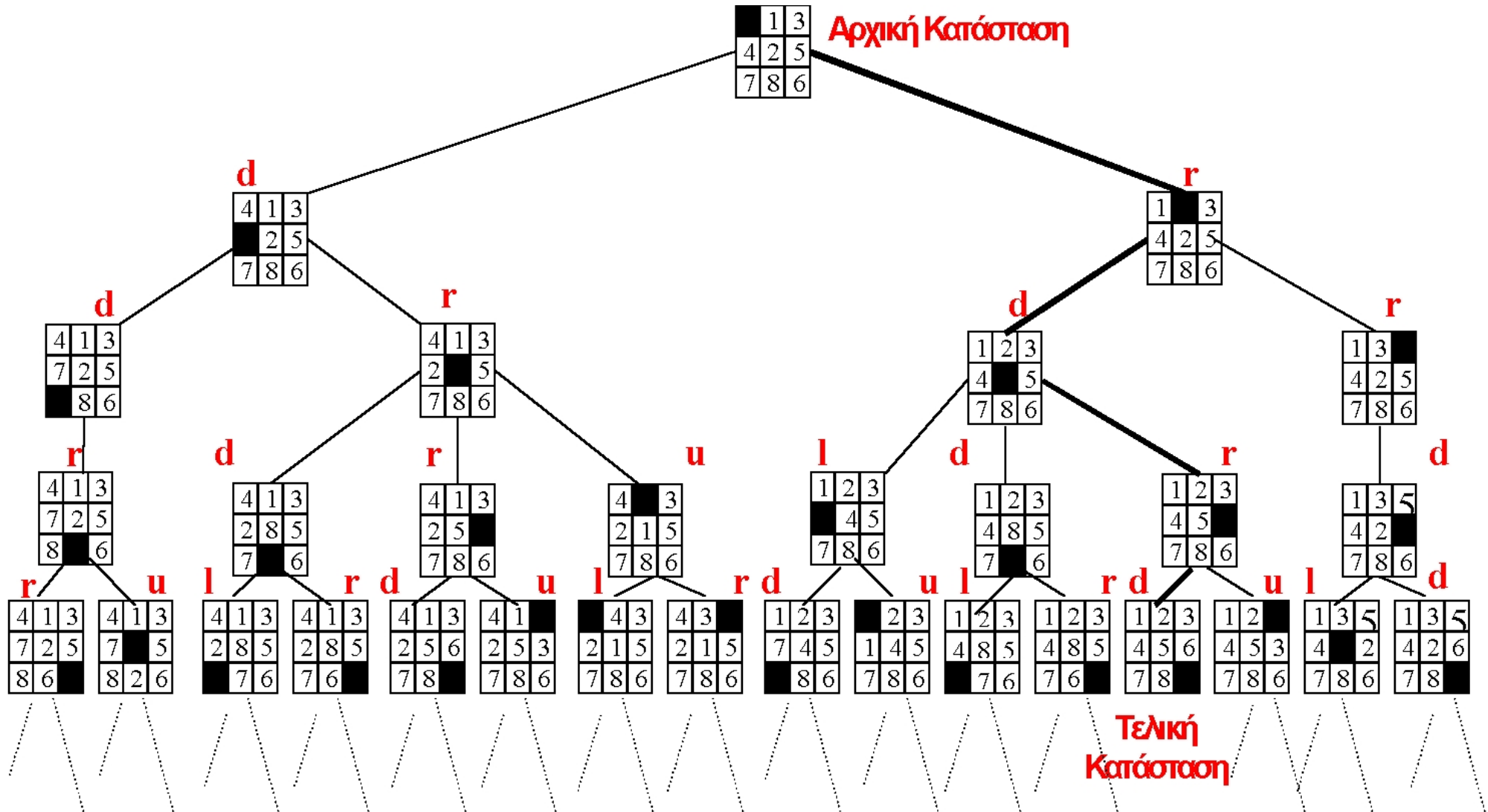
Ο Αλγόριθμος Άλφα-Άστρο (A^*)

Σχόλια

- ❖ Αν για κάθε κατάσταση η τιμή $h(S)$ είναι μικρότερη ή το πολύ ίση με την πραγματική απόσταση της S από την τελική κατάσταση, τότε ο A^* βρίσκει πάντα τη βέλτιστη λύση.
- ❖ Βελτιώσεις:
 - A^* με επαναληπτική εκβάθυνση (Iterative Deepening A^* - IDA)

Εφαρμογή των Αλγορίθμων Ευριστικής Αναζήτησης

Χώρος Αναζήτησης στο 8-puzzle



Εφαρμογή αλγορίθμου BestFS στο 8-puzzle

Αρχική Κατάσταση

