

Τεχνητή Νοημοσύνη

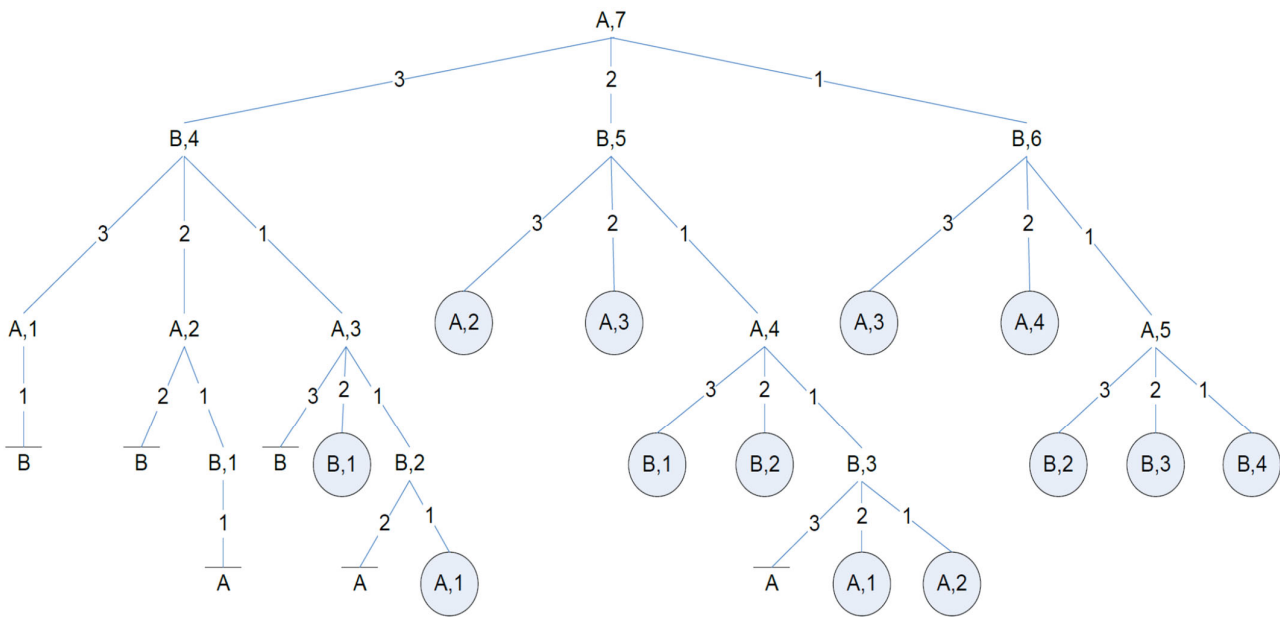
Στρατηγικές Απληροφόρητης Αναζήτησης(Φ)

Δρ. Δημήτριος Κουτσομητρόπουλος

Το παιχνίδι με τα σπέρτα (συνέχεια)

- ▶ Κόστος: Πόσοι κόμβοι θα δημιουργηθούν;
 - ▶ Κόμβοι δημιουργούνται κατά την επέκταση του γονέα
 - ▶ Όμως **BFS**: έλεγχος κατά τη δημιουργία, όχι κατά την επέκταση
 - ▶ Μέχρι να βρεθεί η 1^η λύση;
- ▶ Ποιο είναι το μονοπάτι που βρίσκει ο αλγόριθμος;
- ▶ Αν εφαρμόσετε:
 - ▶ Αναζήτηση κατά βάθος (BFS)
 - ▶ Αναζήτηση κατά πλάτος (DFS)

Σπίρτα - DFS

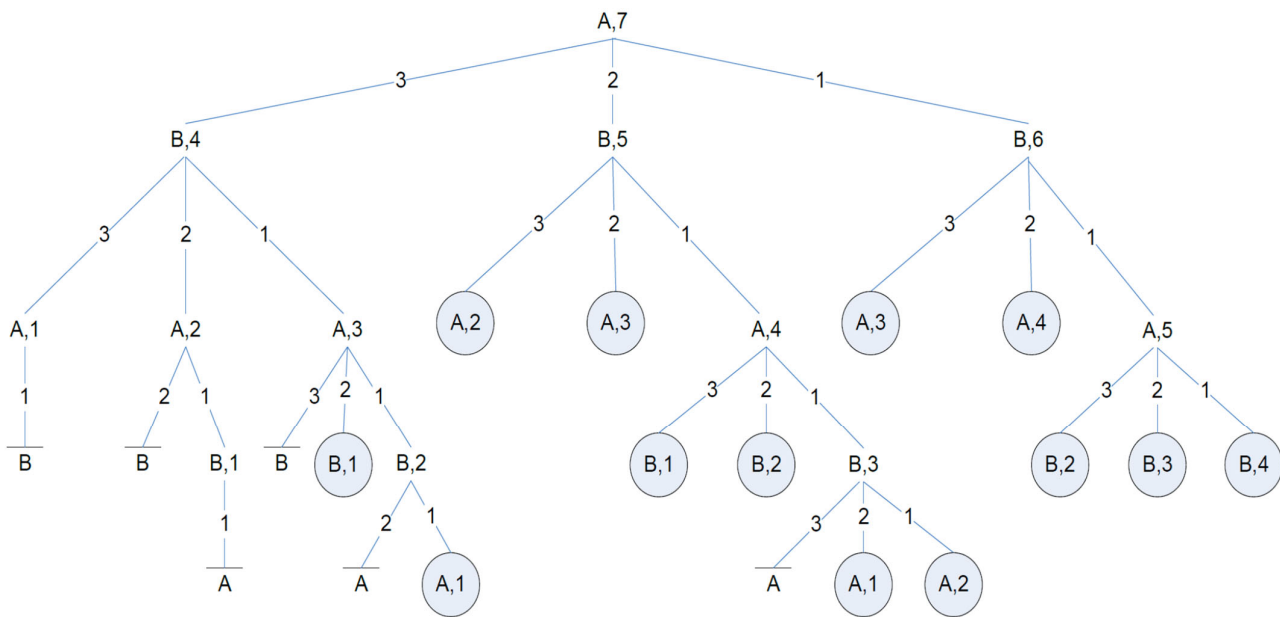


Δημιουργούνται 7+1 κόμβοι (A7, B6, B5, B4, A3, A2, A1, B')

Μονοπάτι: A7, B4, A1, B' μεγέθους 4

▶ 3

Σπίρτα - BFS



Δημιουργούνται 13+1 κόμβοι (A7, B4, B5, B6, A1, A2, A3, A2, A3, A4, A3, A4, A5, B')

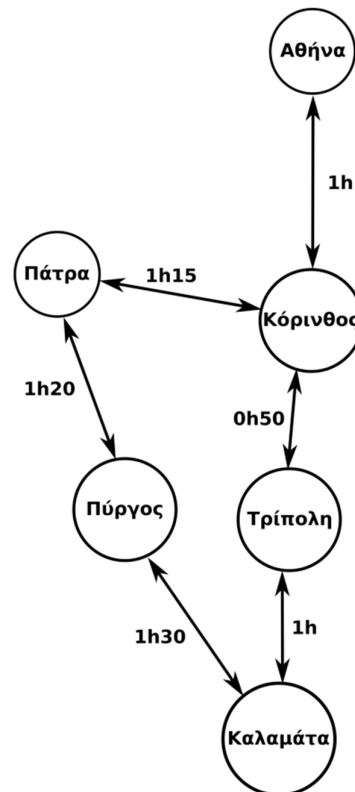
Μονοπάτι: A7, B4, A3, B' μεγέθους 4

▶ 4

Οδικό δίκτυο

Δεξιά βλέπουμε σε γραφική απεικόνιση ένα μέρος του οδικού δικτύου της Ελλάδας. Ένας ταξιδιώτης θέλει να πάει από την Αθήνα στην Καλαμάτα. Ποια διαδρομή θα ακολουθήσει χρησιμοποιώντας: α) αναζήτηση κατά βάθος, β) αναζήτηση κατά πλάτος και γ) αναζήτηση ομοιόμορφου κόστους (uniform cost search);

Θεωρείστε ότι σε περίπτωση ισοβαθμίας επιλέγουμε το παιδί που προηγείται λεξικογραφικά. Θα αλλάξει κάτι στα αποτελέσματα αν επιλέγουμε με την αντίθετη σειρά;



▶ 55

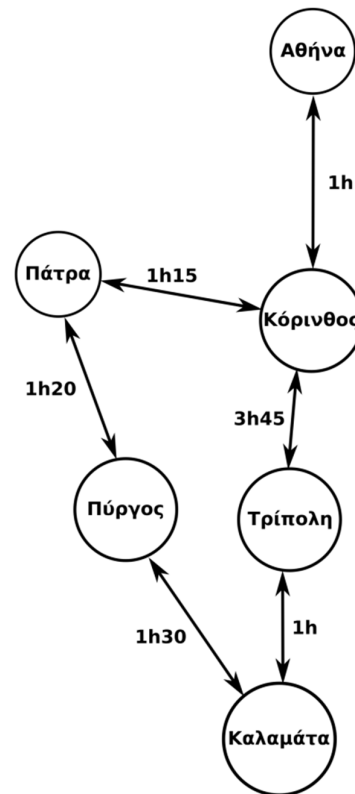
Οδικό δίκτυο – λύση

- ▶ Αν χρησιμοποιήσουμε την **αναζήτηση κατά βάθος**, θα πάρουμε το εξής μονοπάτι:
 - ▶ Αθήνα -> Κόρινθος -> Πάτρα -> Πύργος-> Καλαμάτα
 - ▶ Το συνολικό κόστος της διαδρομής είναι **5h05**.
 - ▶ Αν επιλέγουμε παιδιά με την αντίθετη σειρά, τότε θα πάρουμε το μονοπάτι: Αθήνα -> Κόρινθος -> Τρίπολη -> Καλαμάτα με κόστος **2h50**.
- ▶ Αν χρησιμοποιήσουμε την **αναζήτηση κατά πλάτος**, θα πάρουμε το παρακάτω μονοπάτι:
 - ▶ Αθήνα -> Κόρινθος -> Τρίπολη -> Καλαμάτα.
 - ▶ Το συνολικό κόστος της διαδρομής είναι 2h50.
 - ▶ Αν επιλέγουμε παιδιά με την αντίθετη σειρά, τότε και πάλι θα πάρουμε το ίδιο μονοπάτι.
- ▶ Αν χρησιμοποιήσουμε **αναζήτηση ομοιόμορφου κόστους**, θα πάρουμε:
 - ▶ το ίδιο μονοπάτι με την αναζήτηση κατά πλάτος
 - ▶ το συνολικό κόστος της διαδρομής είναι 2h50
 - ▶ Στην αναζήτηση ομοιόμορφου κόστους στο συγκεκριμένο πρόβλημα, δεν βρισκόμαστε ποτέ σε ισοβαθμία κόστους.

▶ 66

Οδικό δίκτυο - 2

Τι θα αλλάξει στα αποτελέσματα των διαδρομών αν στο δρόμο Κόρινθο->Τρίπολη γίνονται κάποια έργα που απαιτούν παράκαμψη και αλλάζουν οι χρόνοι μετάβασης ως εξής:



▶ 77

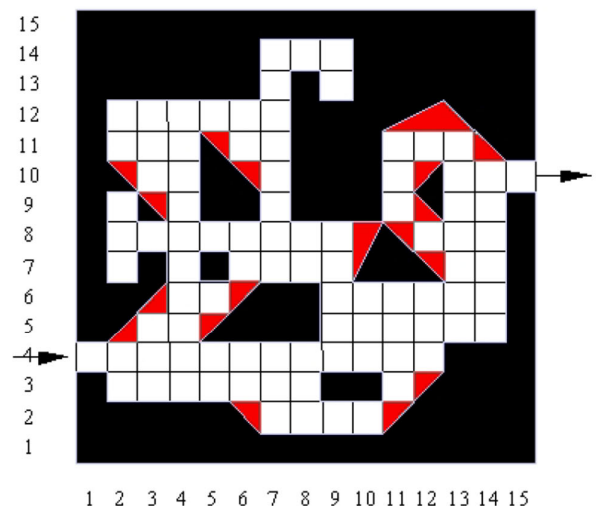
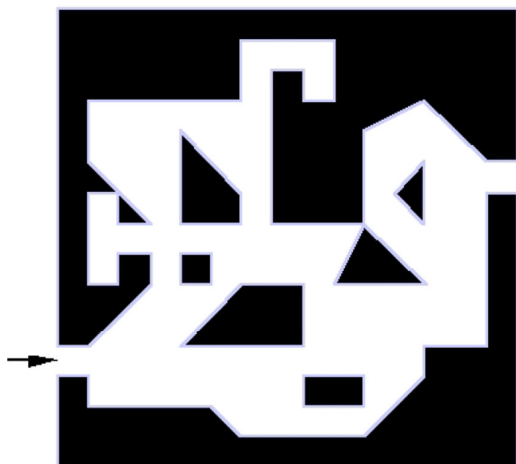
Οδικό δίκτυο -2 - λύση

- ▶ Η αναζήτηση κατά βάθος και η αναζήτηση κατά πλάτος, μιας και δεν αξιοποιούν καμία πληροφορία για το κόστος της διαδρομής:
 - ▶ Θα παράξουν πανομοιότυπα αποτελέσματα με προηγουμένως.
 - ▶ Το κόστος διαδρομής για την **αναζήτηση κατά βάθος** παραμένει **5h05** στην πρώτη περίπτωση.
 - ▶ Όταν επιλέγουμε με αντίθετη σειρά το κόστος γίνεται **5h45**.
 - ▶ Για την **αναζήτηση κατά πλάτος** το κόστος διαδρομής και στις δύο περιπτώσεις γίνεται **5h45**.
- ▶ Η **αναζήτηση ομοιόμορφου κόστους**, μιας και λαμβάνει υπόψη το κόστος της διαδρομής για τις αποφάσεις:
 - ▶ καταφέρνει να βρει κάθε φορά το βέλτιστο μονοπάτι (αφού δεν έχουμε αρνητικά κόστη).
 - ▶ Σε αυτή την περίπτωση (με τα έργα), το βέλτιστο μονοπάτι είναι:
 - ▶ Αθήνα -> Κόρινθος -> Πάτρα -> Πύργος-> Καλαμάτα με κόστος **5h05**.
- ▶ Τι θα συμβεί αν εφαρμόσετε **αμφίδρομη αναζήτηση**;

▶ 8

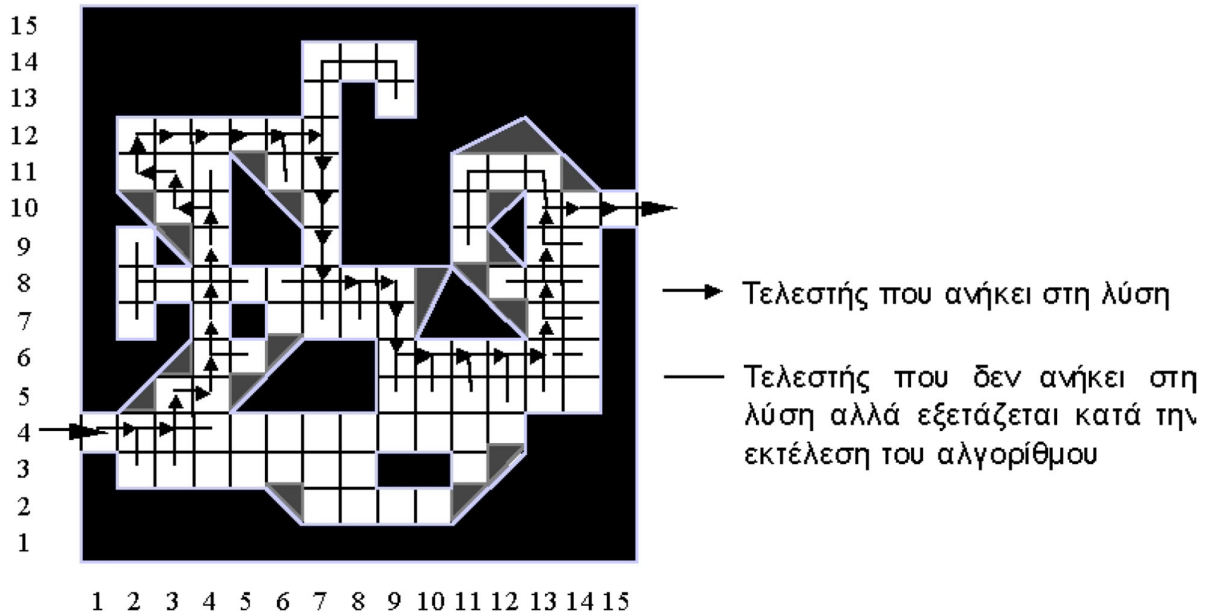
Παράδειγμα Εφαρμογής Αλγορίθμων Απληροφόρητης Αναζήτησης

Το πρόβλημα του λαβυρίνθου



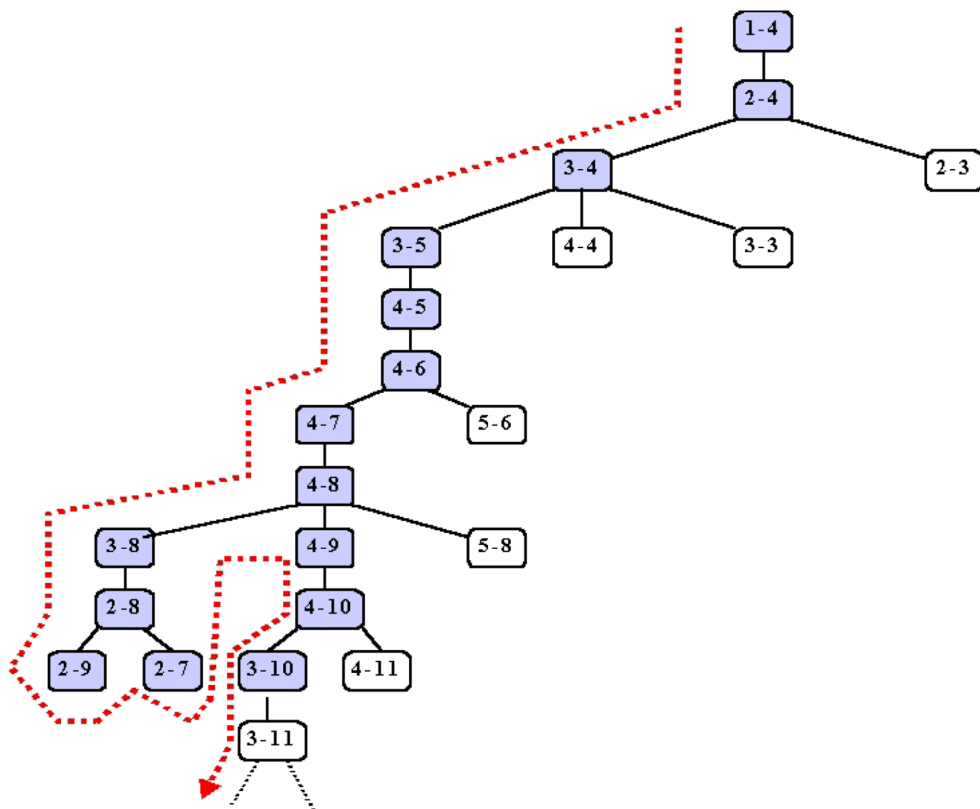
- ▶ Αρχική κατάσταση: (1, 4) (στήλη, γραμμή)
- ▶ Τελική κατάσταση: (15, 10)
- ▶ Τελεστές μετάβασης: **αριστερά, επάνω, δεξιά, κάτω**, εφόσον η θέση είναι ελεύθερη
- ▶ **Με αυτή τη σειρά** σε περίπτωση ισοβαθμίας
- ▶ Χώρος καταστάσεων: Όλες οι ελεύθερες θέσεις, χωρίς εμπόδια, του πλέγματος

Λύση στο πρόβλημα του λαβυρίνθου με χρήση DFS



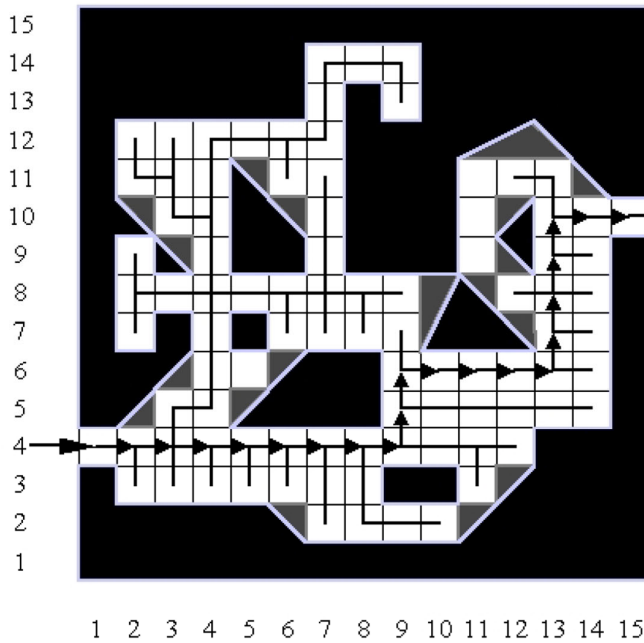
▶ 11

Εφαρμογή του αλγορίθμου DFS



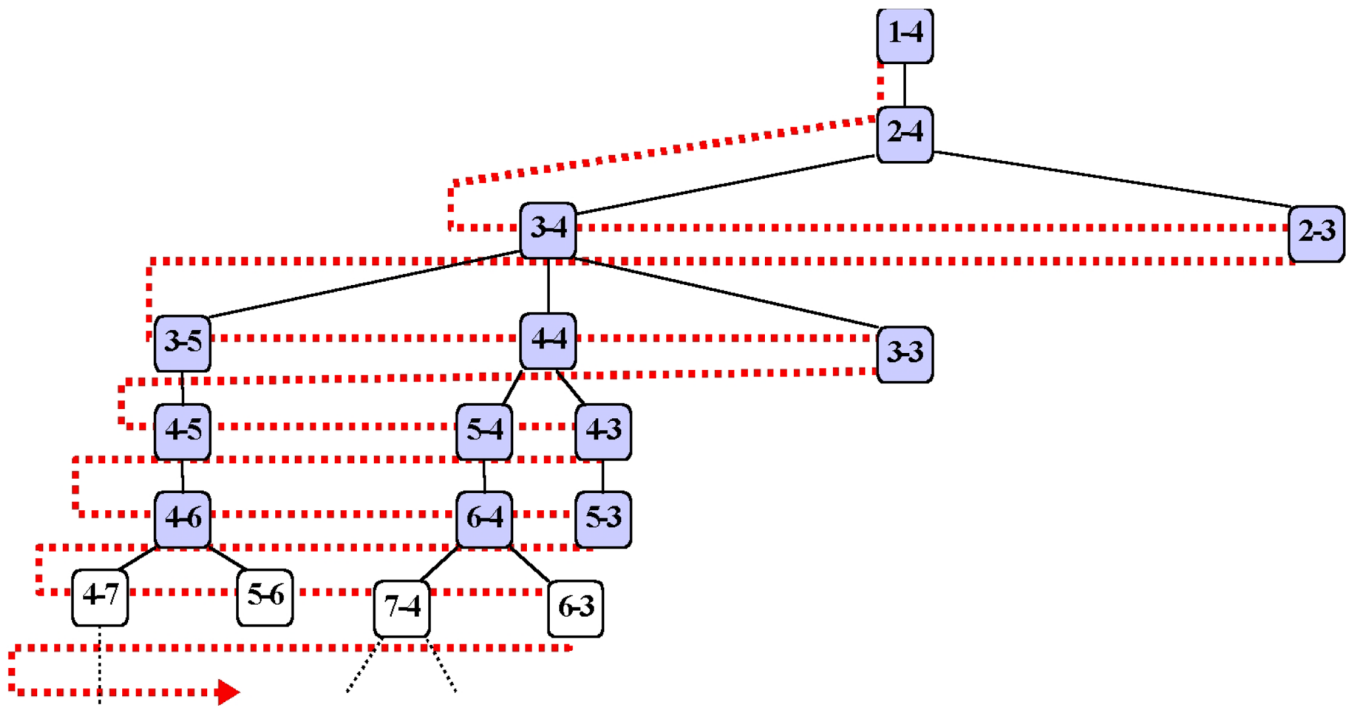
▶ 12

Λύση στο πρόβλημα του λαβυρίνθου με χρήση BFS



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

Εφαρμογή του αλγορίθμου BFS

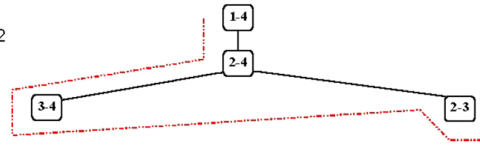


Εφαρμογή του ID στο πρόβλημα του λαβυρίνθου

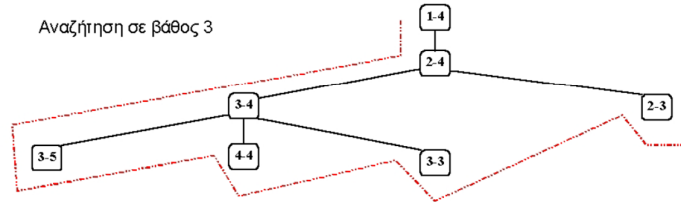
Αναζήτηση σε βάθος 1



Αναζήτηση σε βάθος 2



Αναζήτηση σε βάθος 3



Αναζήτηση σε βάθος 4

