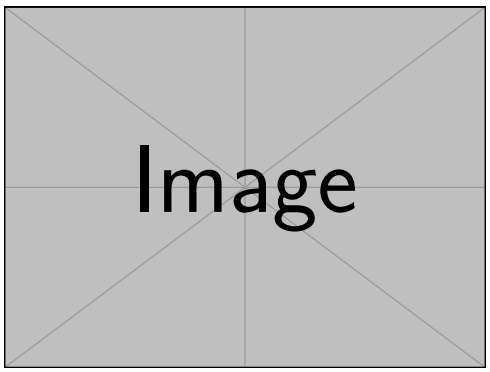
Τεχνητή Νοημοσύνη

Σημειώσεις διαλέξεων

Δεληγιαννάκης Χαράλαμπος

Τμήμα Πληροφορικής ΑΠΘ



Contents

1 Κεφάλαιο 1 - Εισαγωγή

1.1 (Τεχνητή) Νοημουσύνη

Τι είναι η νοημοσύνη
Η ικανότητα για μάθηση, κατανόηση και κρίση. Δηλαδή, να μαθαίνουμε, επικοινωνούμε, λύνουμε προβλήματα.

Τι είναι η τεχνητή νοημοσύνη
Τομέας της επιστήμης των υπολογιστών, που ασχολείται με τη σχεδίαση ευφυών (νοημόνων) υπολογιστικών συστημάτων, δηλαδή συστημάτων που επιδεικνύουν χαρακτηριστικά που σχετίζουμε με τη νοημοσύνη στην ανθρώπινη συμπεριφόρα.

Τι είναι η γνωστική νοημοσύνη
Η επιστήμη που ερευνά τους μηχανισμούς της ανθρώπινης ευφυΐας αναφέρεται ως γνωστική ή γνωσιολογική επιστήμη.

Γενικός Ορισμός Τεχνητής Νοημοσύνης
Τεχνητή Νοημοσύνη (ΤΝ) είναι ο τομέας της Επιστήμης των Υπολογιστών που ασχολείται με τη σχεδίαση και την υλοποίηση προγραμμάτων τα οποία είναι ικανά να μιμηθούν τις ανθρώπινες γνωστικές ικανότητες, εμφανίζοντας έτσι χαρακτηριστικά που αποδίδουμε συνήθως σε ανθρώπινη συμπεριφορά, όπως η επίλυση προβλημάτων, η αντίληψη μέσω της όρασης, η μάθηση, η εξαγωγή συμπερασμάτων, η κατανόηση φυσικής γλώσσας, κλπ.

1.2 Περιοχές της ΤΝ

- Επίλυση προβλημάτων (π.χ σταυρόλεξο)
- Απόδειξη θεωρημάτων
- Επεξεργασία φυσικής γλώσσας
- Τεχνητή όραση
- Μηχανική μάθηση
- Σχεδιασμός ενεργειών και χρονοπρογραμματισμός
- Αυτόνομα robot
- Έμπειρα συστήματα και Συστήματα γνώσης
- Ευφυείς πράκτορες (agents)
- Ευφυείς υπηρεσίες διαδικτύου και σημασιολογικό διαδίκτυο (semantic web)
- Προσαρμοζόμενα και εξελισσόμενα ευφυή συστήματα
- και άλλα...

1.3 Προσεγγίσεις για την ΤΝ

Κλασική / συμβολική ΤΝ

Βασίζεται στην κατανόηση των νοητικών διεργασιών και ασχολείται με την προσομοίωση της ανθρώπινης νοημοσύνης, όπως τα συστήματα κανόνων.

Υπολογιστική νοημοσύνη

Βασίζεται στη μίμηση της βιολογικής λειτουργίας του εγκεφάλου όπως η διαδικασία της εξέλιξης των ειδών ή η λειτουργία του εγκεφάλου, όπως νευωνικά δίκτυα και γενετικοί αλγόριθμοι.

Προσεγγίσεις ανάλογα με τον στόχο

- **Αδύναμη ή περιορισμένη:** ο στόχος είναι η κατασκευή μηχανών που να λειτουργούν **σαν** να είχαν νοημοσύνη
- **Ισχυρή ή γενική:** ο στόχος είναι η κατασκευή μηχανών που σκέφτονται πραγματικά (δεν προσομοιώνουν απλά την σκέψη).

Δοκιμασία Turing

Σειρά ερωτήσεων που υποβάλλει κάποιος σε έναν άνθρωπο και μία μηχανή, χωρίς να ξέρει εκ των προτέρων ποιος είναι τι. Αν στο τέλος δεν καταφέρει να ξεχωρίσει, τότε περνάει το τεστ και η μηχανή θεωρείται ευφυής.

Για να περάσει ένας υπολογιστής το τεστ, απαιτείται συμμετοχή αρκετών επιστημονικών πεδίων:

- Επεξεργασία φυσικής γλώσσας (NLP)
- Αναπαράσταση γνώσης
- Αυτοματοποιημένη συλλογιστική
- Μηχανική μάθηση

Μια επέκταση του τεστ (πλήρες Turing τεστ) περιλαμβάνει και την αναγνώριση εικόνων και αντικειμένων. Άρα, απαταιτείται και η συμμετοχή και άλλων δύο επιστημονικών πεδιών, της μηχανικής όρασης και της ρομποτικής.

1.4 Μηχανική Μάθηση

Τα πολύπλοκα προβλήματα με πολλά δεδομένα και πολύπλοκες σχέσεις μεταξύ των δεδομένων, δεν μπορούν να επιλυθούν μέσω προκαθορισμένης γνώσης, γιατί αυτή είναι πολύ δύσκολο να εξαχθεί εμπειρικά. Αυτές οι δυσκολίες οδήγησαν στην ανάπτυξη συστημάτων ΤΝ τα οποία έχουν την ικανότητα ανακάλυψης γνώσης, εξάγοντας μοντέλα ή πρότυπα από τα πρωτογενή δεδομένα. Αυτή η ικανότητα ονομάζεται **μηχανική μάθηση (machine learning)**.

Η ικανότητα δηλαδή των υπολογιστών να εκτελούν μια συγκεκριμένη λειτουργία για την οποία δεν έχουν ρητά προγραμματιστεί αλλά αυτή προέκυψε ως αποτέλεσμα εκπαίδευσής τους με δεδομένα.

2 Κεφάλαιο 2 - Περιγραφή προβλημάτων και αναζήτηση λύσης

2.1 Επίλυση Προβλημάτων

Απαρτίζεται από τα εξής 3 μέρη:

- **Περιγραφή προβλήματος:** περιγραφή του χώρου καταστάσεων, ορισμός τελεστών μετάβασης, περιγραφή της αρχικής κατάστασης και των τελικών καταστάσεων.
  Σε παράδειγμα εύρεσης διαδρομής, η περιγραφή του προβλήματος περιλαμβάνει τον καθορισμό του ελεύθερου από εμπόδια χώρου, τις κινήσεις που μπορεί να εκτελέσει το ρομπότ κα ιτον καθορισμού του αρχικού και του τελικού σημείου της διαδρομής.
- **Αναπαράσταση γνώσης:** για τον περιβάλλοντα κόσμο με τέτοιο τρόπο ώστε να είναι δυνατή η επεξεργασία αυτής της γνώσης και η εξαγωγή συμπερασμάτων από ένα υπολογιστικό σύστημα.
- **Αναζήτηση:** η λύση σε ένα πρόβλημα δίνεται με την εφαρμογή ενός αλγορίθμου αναζήτησης. Κάθε αλγόριθμος έχει τέτοια χαρακτηριστικά που τον κάνουν ικανό σε κάποιο πρόβλημα να αναζητήσει πιο αποδοτικά τη λύση στο χώρο αναζήτησης από κάποιον άλλο.

2.2 Περιγραφή Προβλημάτων

Σε κάθε πρόβλημα, υπάρχει μια δεδομένη αρχική κατάσταση, μια επιθυμητή τελική κατάσταση και διαθέσιμες ενέργεις που πρέπει να γίνουν για να φτάσουμε από την αρχική στην τελική. Ο ορισμός ή η περιγραφή του προβλήματος πρέπει να είναι ανεξάρτητη από την πολυπλοκότητα της επίλυσής του (η οποία καθορίζεται από την πολυπλοκότητα του επιλεγμένου αλγορίθμου αναζήτησης).

Η περιγραφή προβλημάτων μπορεί να γίνει με τους εξής δύο βασικούς τρόπους:

- Περιγραφή με **Χώρο Καταστάσεων**
- Περιγραφή με **Αναγωγή**

2.3 Περιγραφή Προβλημάτων με Χώρο Καταστάσεων

Για να περιγραφεί ένα πρόβλημα, θα πρέπει να αναπαρασταθεί:

- Πρώτα ο κόσμος του προβλήματος
- Στη συνέχεια τα συγκεκριμένα χαρακτηριστικά του (αρχική-τελική κατάσταση και οι ενέργειες/τελεστές μετάβασης)

Η αναπαράσταση του κόσμου ενός προβλήματος, είναι μία πολύπλοκη διαδικασία για μεγάλα προβλήματα πραγματικού κόσμου (Αναπαράσταση της Γνώσης).

Κόσμος Προβλήματος
Αποτελείται μόνο από τα αντικείμενα που υπάρχουν σε αυτόν, τις ιδιότητές τους και τις σχέσεις που τα συνδέουν. (Είναι υποσύνολο του πραγματικού κόσμου)

Κλειστός Κόσμος
Τίποτα δεν εισάγεται ή εξάγεται από ή προς άλλο κόσμο.

Ανοιχτός Κόσμος
Το αντίθετο του κλειστού.

Κατάσταση προβλήματος
Κατάσταση *(state)* ενός κόσμος είναι ένα στιγμιότυπο ή φωτογραφία *(snapshot)* μιας συγκεκριμένης χρονικής στιγμής της εξέλιξης του κόσμου, δηλαδή μια επαρκής αναπαράσταση του κόσμου σε μία δεδομένη χρονική στιγμή.
Αυτό επιτυγχάνεται με τη λειτουργία της **αφαίρεσης (abstraction)**, σύμφωνα με την οποία για να περιγραφεί μία κατάσταση αφαιρούνται όλες εκείνες οι λεπτομέρειες οι οποίες δεν είναι απαραίτητες για τη μετέπειτα επίλυση ενός προβλήματος.

Τελεστές μετάβασης
Η αντιστοίχιση μιας κατάσταση του κόσμου σε νέες καταστάσεις. Υπάρχουν προϋποθέσεις εφαρμογής που πρέπει να τηρούνται για να εφαρμοστεί ένας τελεστής - μια βασική προϋπόθεση είναι η κατάσταση που προκύπτει να είναι έγκυρη.

Χώρος καταστάσεων
Το σύνολο όλων των έγκυρων καταστάσεων. Απεικονίζεται συνήθως διαισθητικά με έναν γράφο.

Ορισμός προβλήματος
Ένα πρόβλημα ορίζεται ως η τετράδα $P = (I, G, T, S)$ όπου:

- I είναι η αρχική κατάσταση, $I \in S$
- G είναι το σύνολο των τελικών καταστάσεων, $G \subseteq S$
- T είναι το σύνολο των τελεστών μετάβασης, $T : S- > S$
- S είναι ο χώρος καταστάσεων

Λύση προβλήματος
Λύση σε ένα πρόβλημα $P = (I, G, T, S)$ είναι μία ακολουθία από τελεστές μετάβασης $t_1, t_2, ..., t_n \in T$ με την ιδιότητα $g = t_n(...(t_2(t_1(I)))...)$, όπου $g \in G$.
Δηλαδή, λύση σε ένα πρόβλημα είναι η ακολουθία τελεστών που εφαρμόζονται στην αρχική κατάσταση για να προκύψει μία τελική κατάσταση.

Κατηγορίες προβλήματος

- **Προβλήματα σχεδιασμού ενεργειών:** είναι πλήρως γνωστές οι τελικές καταστάσεις και επιδιώκεται η έυρεση μιας σειράς ενεργειών που θα μας οδηγήσουν από την αρχική κατάσταση στην τελική κατάσταση.
- **Προβλήματα χρονοπρογραμματισμού:** σε αυτά είναι γνωστές κάποιες ιδιότητες μόνο της τελικής κατάστασης και επιδιώκεται η εύρεση ενός πλήρου στιγμιότυπου της τελικής κατάστασης (όπως η δημιουργία ωρολόγιου προγράμματος σε έχνα σχολείο, ξεκινώντας από ένα κενό πρόγραμμα αναζητείται το πλήρες πρόγραμμα - το οποίο αποτελεί στιγμιότυπο της τελικής κατάστασης).
- **Προβλήματα ικανοποίησης περιορισμών:** όπως το σταυρόλεξο, όπου στο πλήρες στιγμιότυπο της τελικής κατάστασης όλες οι λέξεις είναι τοποθετημένες στη σωστή τους θέση και αυτό αποτελεί τη λύση του - υπάρχουν περιορισμοί για το ποιες λέξεις θα τοποθετηθούν που.
- **Προβλήματα διαμόρφωσης:** συνδυασμός των παρπάνω κατηγοριών - δηλαδή είναι γνωστές κάποιες ιδιότητες μόνο, της τελικής κατάστασης και επιδιώκεται η εύρεση μίας πλήρως γνωστής τελικής κατάστασης και η σειρά ενεργειών που θα οδηγήσουν σε αυτή (όπως η κατασκευή ενός αυτοκινήτου βάσει κάποιων προδιαγραφών, ζητούμενο είναι όχι μόνο τα εξαρτήματα που θα συνδέσουν το αυτοκίνητο αλλά και η σειρά με την οποία θα τοποθετηθούν).

- **Προβλήματα βελτιστοποίησης:** σε αυτά επίσης η τελική κατάσταση δεν είναι πλήρως γνωστή αλλά είναι γνωστά κάποια χαρκτηριστικά της.
- **Γενετικοί αλγόριθμοι:** αποτελούν ένα ανάλογο της θεωρίας της εξέλιξης, προσομοιάζοντας την εύρεση μιας καλύτερης λύσης σε ένα πρόβλημα με τη συνεχή εξέλιξη άλλων λύσεων, μέσω κατάλληλων διαδικασιών αναπαραγωγής και μετάλλαξης.

2.4 Περιγραφή με Αναγωγή

Μια ακολουθία από τελεστές ανάγουν την περιγραφή ενός προβλήματος σε υποπροβλήματα τα οποία είναι άμεσα επιλύσιμα, **αρχέγονα**. Η βασική δομή δεν είναι η κατάσταση αλλά η ίδια η περιγραφή ενός προβλήματος. Αντί για αρχική κατάσταση έχουμε **αρχική περιγραφή** και αντί για τελεστή μετάβασης έχουμε **τελεστή αναγωγής**, ο οποίος ανάγει ένα πρόβλημα σε υποπροβλήματα.

Ορισμός προβλήματος

Ένα πρόβλημα ορίζεται ως η τετράδα $P = (ID, GD, TR, PP)$ όπου:

- ID είναι η αρχική περιγραφή
- GD είναι ένα σύνολο από τελικές περιγραφές
- TR είναι το σύνολο των τελεστών αναγωγής
- PP είναι ένα σύνολο από αρχέγονα προβλήματα

2.5 Αλγόριθμοι αναζήτησης

Η επιλογή ενός αλγορίθμου αναζήτησης για ένα συγκεκριμένο πρόβλημα είναι σημαντική, διότι οι αλγόριθμοι αυτοί διαφέρουν μεταξύ τους σε αρκετά χαρακτηριστικά. Ο λόγος για την ύπαρξη τόσων αλγορίθμων είναι ότι κάθε ένας έχει διαφορετικά χαρακτηριστικά που τον καθιστούν περισσότερο ή λιγότερο αποδοτικό από άλλους, σε απαιτήσεις μνήμης ή/και χρόνο εκτέλεσης.

Χώρος αναζήτησης

Αλγόριθμοι Αναζήτησης

- Τυφλοί

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

- Ευριστικοί

Hill Climbing	HC	Αναρρίχηση Λόφων
Enforced Hill Climbing	EHC	Εξαναγκασμένη Αναρρίχηση Λόφων
Simulated Annealing	SA	Προσομοιωμένη Ανόπτηση
Tabu Search	TS	Αναζήτηση με Απαγορευμένες Καταστάσεις
Beam Search	BS	Ακτινωτή Αναζήτηση
Best-First Search	BestFS	Αναζήτηση Πρώτα στο Καλύτερο
A* (A-star)	A*	A* (Άλφα Άστρο)

- Παιχνιδιών 2 ατόμων

Minimax	Minimax	Αναζήτηση Μεγίστου-Ελαχίστου
Alpha-Beta	AB	Άλφα-Βήτα

Εικόνα 2.5.1: Βασικότεροι αλγόριθμοι αναζήτησης

Δοθέντος ενός προβλήματος $P = (I, G, T, S)$, χώρος αναζήτησης SP είναι το σύνολο όλων των καταστάσεων που είναι προσβάσιμες από την αρχική κατάσταση. Μια κατάσταση ονομάζεται προσβάσιμη αν υπάρχει ακολουθία τελεστών μετάβασης ώστε να καταλήξουμε σε αυτήν, ξεκινώντας από την αρχική κατάσταση. Ισχύει ότι $SP \subseteq S$.
Ένας αλγόριθμος αναζήτησης δεν μειώνει τον χώρο αναζήτησης (που είναι δεδομένος) αλλά καθορίζει τον αριθμό των καταστάσεων που επισκέπτεται.

Χώρος αναζήτησης ως δένδρο αναζήτησης

Ο χώρος αναζήτησης μπορεί να αναπαρασταθεί με γράφο, ο οποίος μπορεί να μετατραπεί σε **δέντρο αναζήτησης (search tree)**, το οποίο όμως μπορεί να έχει μονοπάτια άπειρου μήκους. Το φαινόμενο της εκθετικής αύξησης του αριθμού των κόμβων του δένδρου ονομάζεται **συνδυ-**

Τμήμα Δένδρου	Αναπαράσταση
Κόμβος (Node)	Κατάσταση
Ρίζα (Root)	Αρχική Κατάσταση
Φύλλο (Tip, Leaf) ή Τερματικός κόμβος	Τελική Κατάσταση ή Αδιέξοδο (Dead Node), δηλαδή κατάσταση στην οποία δεν μπορεί να εφαρμοστεί κανένας τελεστής μετάβασης.
Κλαδί (Branch)	Τελεστής Μετάβασης που μετατρέπει μια κατάσταση-Γονέα (Parent State) σε μία άλλη κατάσταση-Παιδί (Child State).
Λύση (Solution)	Μονοπάτι (Path) που ενώνει την αρχική με μία τελική κατάσταση
Επέκταση (Expansion)	Η διαδικασία παραγωγής όλων των καταστάσεων-παιδιών ενός κόμβου.
Παράγοντας Διακλάδωσης (Branching Factor)	Ο αριθμός των καταστάσεων-παιδιών που προκύπτουν από την επέκταση μιας κατάστασης. Επειδή δεν είναι σταθερός αριθμός, αναφέρεται και ως Μέσος Παράγοντας Διακλάδωσης (Average Branching Factor).

Τα δένδρα είναι Η΄-δένδρα (Or-trees) γιατί σε κάθε κόμβο έχουμε δυνατότητα επιλογής

Εικόνα 2.5.2: Δομή του δέντρου αναζήτησης

αστική έκρηξη.

2.6 Χαρακτηριστικά Αλγορίθμων

Δοθέντος ενός προβλήματος $P = (I, G, T, S)$ και μετά την εφαρμογή κάποιου αλγόριθμου στο χώρο αναζήτησής του, προκύπτει το **επιλυμένο πρόβλημα**, το οποίο ορίζεται ως μια τετράδα $P_s = (V, A, F, G_s)$, όπου:

- V είναι το σύνολο των καταστάσεων που εξέτασε ο αλγόριθμος αναζήτησης
- A είναι ο αλγόριθμος που χρησιμοποιήθηκε
- F είναι το σύνολο των λύσεων που βρέθηκαν
- G_s είναι το σύνολο των τελικών καταστάσεων που εξετάστηκαν

Ο πληθάριθμος του V, δηλαδή ο αριθμός των καταστάσεων που εξέτασε ο αλγόριθμος, και η σχέση του με το χώρο αναζήτησης SP ενός προβλήματος, είναι ένα από τα χαρακτηριστικά της αποδοτικότητας του αλγορίθμου.

Ένας αλγόριθμος ονομάζεται **εξαντλητικός (exhausitive)** όταν το σύνολο των καταστάσεων που εξετάζει ο αλγόριθμος για να βρει τις απαιτούμενες λύσεις είναι ίσο με το χώρο αναζήτησης, δηλαδή $V = SP$

Ένας αλγόριθμος δεν λύνει πάντα κάποιο πρόβλημα, έστω και αν υπάρχει λύση. Τότε, τα σύνολα G_s και F είναι κενά.

Ένας αλγόιρθμος ονομάζεται **πλήρης (complete)** αν εγγυάται ότι θα βρει μία λύση για οποιαδήποτε τελική κατάσταση, αν τέτοια λύση υπάρχει. Σε αντίθεση περίπτωση, ο αλγόριθμος ονομάζεται **μη-πλήρης (incomplete)**.
Η πληρότητα ενός αλγορίθμου μπορεί να αποδειχθεί μόνο μαθηματικά. Ωστόσο, σε ορισμένες περιπτώσεις είναι σίγουρο πως αν ο αλγόριθμος δε βρει λύση, τότε οπωσδήποτε δεν υπάρχει λύση στο πρόβλημα, όπως για παράδειγμα στην περίπτωση που ένας αλγόριθμος είναι εξαντλητικός.

Μία λύση ονομάζεται **βέλτιστη (optimal)** αν οδηγεί στην καλύτερη, σύμφωνα με τη διάταξη, τελική κατάσταση. Όταν δεν υπάρχει διάταξη, μία λύση ονομάζεται βέλτιστη αν είναι η συντομότερη.

Ένας αλγόριθμος αναζήτησης ονομάζεται **αποδεκτός (admissible)** αν εγγυάται ότι θα βρει τη βέλτιστη λύση, αν μια τέτοια λύση υπάρχει.

2.7 Διαδικασία Επιλογής Αλγορίθμου Αναζήτησης

Η επιλογή αλγορίθμου βασίζεται στα εξής κριτήρια:

- Αριθμός καταστάσεων που αυτός επισκέπτεται
- Δυνατότητα εύρεσης λύσεω εφόσον αυτές υπάρχουν
- Αριθμός των λύσεων
- Ποιότητα των λύσεων
- Αποδοτικότητά του σε χρόνο
- Αποδοτικότητά του σε χώρο (μνήμη)
- Ευκολία υλοποίησής του

Στα κριτήρια αυτά εντάσσεται και η έννοια του *κλαδέματος ή αποκοπής καταστάσεων (pruning)* του χώρου αναζήτησης.

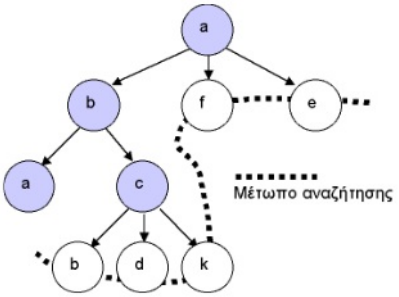
Κλάδεμα/αποκοπή καταστάσεων (pruning)
Είναι η διαδικασία κατά την οποία ο αλγόριθμος απορρίπτει, κάτω από ορισμένες συνθήκες, κάποιες καταστάσεις και μαζί με αυτές όλο το υποδένδρο που εκτυλίσσεται κάτω από τις καταστάσεις αυτές.

2.8 Γενικός Αλγόριθμος Αναζήτησης

Μέτωπο αναζήτησης (search frontier)
Το διατεταγμένο σύνολο (λίστα) των καταστάσεων που ο αλγόριθμος έχει ήδη επισκεφτεί, αλλά δεν έχουν ακόμη επεκταθεί.

Κλειστό σύνολο (closed set)
Το σύνολο όλων των καταστάσεων που έχουν ήδη επεκταθεί από τον αλγόριθμο.
Με έναν απλό έλεγχο, αν η κατάσταση προς επέκταση ανήκει ήδη στο κλειστό σύνολο, αποφεύγονται οι βρόχοι (loops).

Κάθε επέκταση μιας κατάστασης συνοδεύεται από την εισαγωγή της κατάστασης γονέας στο κλειστό σύνολο και την εισαγωγή των καταστάσεων παιδών στο μέτωπο αναζήτησης. Επίσης, οι



Εικόνα 2.8.1: Γράφος αναζήτησς

κόμβοι που είναι σκιαγραμμένοι με μπλε χρώμα, αποτελούν το κλειστό σύνολο (έχει ήδη επισκεπτεί και επεκταθεί). Ακολουθεί η περιγραφή του γενικού αλγόριθμου αναζήτησης σε βήματα και έπειτα σε ψευδογλώσσα.

1. Βάλε την αρχική κατάσταση στο μέτωπο της αναζήτησης.
2. Αν το μέτωπο αναζήτησης είναι άδειο τότε σταμάτησε.
3. Πάρε την πρώτη σε σειρά κατάσταση του μετώπου της αναζήτησης.
4. Αν είναι η κατάσταση αυτή μέρος του κλειστού συνόλου τότε πήγαινε στο βήμα 2.
5. Αν είναι η κατάσταση αυτή τελική κατάσταση τότε τύπωσε τη λύση και πήγαινε στο βήμα 2.
6. Εφάρμοσε τους τελεστές μετάβασης για να παράγεις τις καταστάσεις-παιδιά.
7. Βάλε τις νέες καταστάσεις-παιδιά στο μέτωπο της αναζήτησης.
8. Κλάδεψε τις καταστάσεις που δε χρειάζονται (σύμφωνα με κάποιο κριτήριο), βγάζοντάς τες από το μέτωπο της αναζήτησης.
9. Κάνε αναδιάταξη στο μέτωπο της αναζήτησης (σύμφωνα με κάποιο κριτήριο).
10. Βάλε την κατάσταση-γονέα στο κλειστό σύνολο.
11. Πήγαινε στο βήμα 2.

Εικόνα 2.8.2: Περιγραφή γενικού αλγορίθμου αναζήτησης με βήματα

```
algorithm general(InitialState, FinalState)
begin
  Closed←∅;
  Frontier←<InitialState>;
  CurrentState←First(Frontier);
  while CurrentState ∉ FinalState do
    Frontier←delete(CurrentState,Frontier);
    if CurrentState ∉ ClosedSet then
      begin
        Next←Expand(CurrentState);
        Frontier←insert(Next,Frontier);
        Frontier←prune(Frontier);
        Frontier←reorder(Frontier);
        Closed←Closed∪{CurrentState};
      end;
    if Frontier=∅ then return failure;
    CurrentState←First(Frontier);
  endwhile;
end.
```

Εικόνα 2.8.3: Περιγραφή γενικού αλγορίθμου αναζήτησης με ψευδογλώσσα