ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ

(2020-2021)

1η Σειρά Γραπτών Ασκήσεων

Ονοματεπώνυμο:

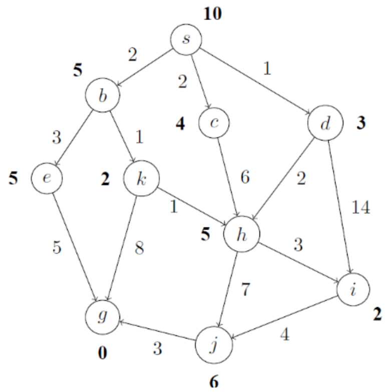
- Χρήστος Τσούφης

Αριθμός Μητρώου:

- 03117176

1η Άσκηση

1. Ακολουθεί η εκτέλεση των αλγορίθμων για τον παρακάτω γράφο:



Hill Climbing

Ο αλγόριθμος Hill Climbing είναι ένας greedy αλγόριθμος αναζήτησης ο οποίος όταν εξετάζει μια κατάσταση, ταξινομεί τις καταστάσεις – παιδιά της με βάση την τιμή ενός heuristic function και τοποθετεί στο ανοικτό μέτωπο την κατάσταση που η heuristic function θεωρεί καλύτερη μόνο αν η τιμή της συνάρτησης για την κατάσταση – παιδί είναι καλύτερη από την τιμή της συνάρτησης για την τρέχουσα κατάσταση αλλιώς, ο αλγόριθμος σταματάει. Η εκτέλεση του αλγορίθμου για το γράφο είναι η ακόλουθη:

Open Set	Closed Set	Current State	Children States	Heuristic
[(s,10)^{s}]	n/a	s	[b, c, d]	[(c, 4), (b, 5), (d, 3)]
[(d,3)^{s,d}]	s	d	[h, i]	[(i, 2), (h, 5)]
[(i,2)^{s,d,i}]	s, d	i	[j]	[(j, 6)]

Best First

Ο αλγόριθμος Best First τοποθετεί όποια κατάσταση δεν έχει ξαναδεί στο ανοικτό σύνολο αλλά, κάθε φορά το ανοικτό σύνολο το έχει διατεταγμένο ως προς τις τιμές που δίνει το heuristic για κάθε κατάσταση που έχει συναντήσει. Έτσι, κάνει expand την κατάσταση που φαίνεται περισσότερο ελπιδοφόρα από όσες έχει ήδη συναντήσει. Η εκτέλεση του αλγορίθμου για το γράφο είναι η ακόλουθη:

Open Set	Closed Set	Current State	Children States	Heuristic
[(s,10)^{s}]	n/a	s	[b, c, d]	[(b, 5), (c, 4), (d, 3)]
[(c,4)^{s,c}, (d,3)^{s,d}, (b,5)^{s,b}]	s	d	[h, i]	[(h, 5), (i, 2)]
[(i,2)^{s,d,i}, (c,4)^{s,c}, (b,5)^{s,b}, (h,5)^{s,d,h}]	s, d	i	[j]	[(j, 6)]
[(c,4)^{s,c}, (b,5)^{s,b}, (h,5)^{s,d,h}, (j,6)^{s,d,i,j}]	s, d, i	c	[h]	[(h, 5)]
[(h,5)^{s,d,h}, (b,5)^{s,b}, (j,6)^{s,d,i,j}]	s, d, i, c	b	[e, k]	[(e, 5), (k, 2)]
[(k,2)^{s,b,k}, (e,5)^{s,b,e}, (h,5)^{s,d,h}, (j,6)^{s,d,i,j}]	s, d, i, c, b	k	[g, h]	[(h, 5) , (g, 0)]
[(g,0)^{s,b,k,g}, (e,5)^{s,b,e}, (h,5)^{s,b,k,h}, (j,6)^{s,d,i,j}]	s, d, i, c, b, k	g	[]	[]

A* Star

Ο αλγόριθμος <u>A* Star</u> λειτουργεί με τρόπο που μοιάζει με τον αλγόριθμο Best First, μόνο που το ανοικτό μέτωπο είναι ένα διατεταγμένο σύνολο ως προς άλλη μετρική. Η μετρική εδώ είναι ένας γραμμικός συνδυασμός των τιμών που έχει δώσει το heuristic για την κάθε κατάσταση (h(n)) και του κόστους που πραγματικά έχει δαπανηθεί ώστε να φτάσει σε αυτή την κατάσταση g(n). Έτσι, προκύπτει η μετρική: f (n) = g(n) + h(n) , με βάση την οποία διατάσσεται το ανοικτό σύνολο κάθε φορά που εξετάζεται κάποια κατάσταση. Η εκτέλεση του αλγορίθμου για το γράφο είναι η ακόλουθη:

Open Set	**Closed Set**	**Current State**	**Children States**	**Heuristic**
[(s,0,10)^{s}]	n/a	s	[b, c, d]	[(c,2,6), (b,2,7), (d,1,4)]
[(d,1,4)^{s,d}, (c,2,6)^{s,c}, (b,2,7)^{s,b}]	s	d	[h, i]	[(h,3,8), (i,15,17)]
[(c,2,6)^{s,c}, (i,15,17)^{s,d,i}, (b,2,7)^{s,b}, (h,3,8)^{s,d,h}]	s, d	c	[h]	[(h,8,13)]
[(b,2,7)^{s,b}, (h,3,8)^{s,d,h}, (i,15,17)^{s,d,i}]	s, d, c	b	[e, k]	[(k,3,5) , (e,5,10)]
[(h,3,8)^{s,d,h}, (k,3,5)^{s,b,k}, (e,5,10)^{s,b,e}, (i,15,17)^{s,d,i}]	s, d, c, b	k	[g, h]	[(g,11,11), (h,4,9)]
[(h,3,8)^{s,d,h}, (g,11,11)^{s,b,k,g}, (e,5,10)^{s,b,e}, (i,15,17)^{s,d,i}]	s, d, c, b, k	h	[i, j]	[(i,6,8), (j,10,16)]
[(i,6,8)^{s,d,h,i}, (e,5,10)^{s,b,e}, (g,11,11)^{s,b,k,g}, (j,10,16)$^{s,d,h,j/s,d,h,i,j}$]	s, d, c, b, k, h	i	[j]	[(j,10,16)]
[(e,5,10)^{s,b,e}, (g,11,11)^{s,b,k,g}, (j,10,16)$^{s,d,h,j/s,d,h,i,j}$]	s, d, c, b, k, h, i	e	[g]	[(g,10,10)]
[(g,10,10)^{s,b,e,g}, (j,10,16)$^{s,d,h,j/s,d,h,i,j}$]	s, d, c, b, k, h, i, e	g	[]	[]

2. Όσον αφορά τις λύσεις του προβλήματος:

Το γράφημα δεν έχει κύκλους άρα, ο αριθμός των λύσεων θα είναι πεπερασμένος. Συγκεκριμένα, υπάρχουν 9 λύσεις οι οποίες είναι οι ακόλουθες ακολουθίες κορυφών:

- s → b → e → g
- s → b → k → g
- s → c → h → j → g
- s → d → h → j → g
- s → d → h → i → j → g
- s → b → k → h → j → g
- s → b → k → h → i → j → g
- s → d → i → j → g
- s → c → h → i → j → g

Η καλύτερη ακολουθία είναι αυτή η s → b → e → g με κόστος 10.

Hill Climbing Algorithm	Δεν βρίσκει αποτέλεσμα.
Best First Algorithm	$s \rightarrow b \rightarrow k \rightarrow g$
A* Star Algorithm	$s \rightarrow b \rightarrow e \rightarrow g$

Αναλυτικότερα,

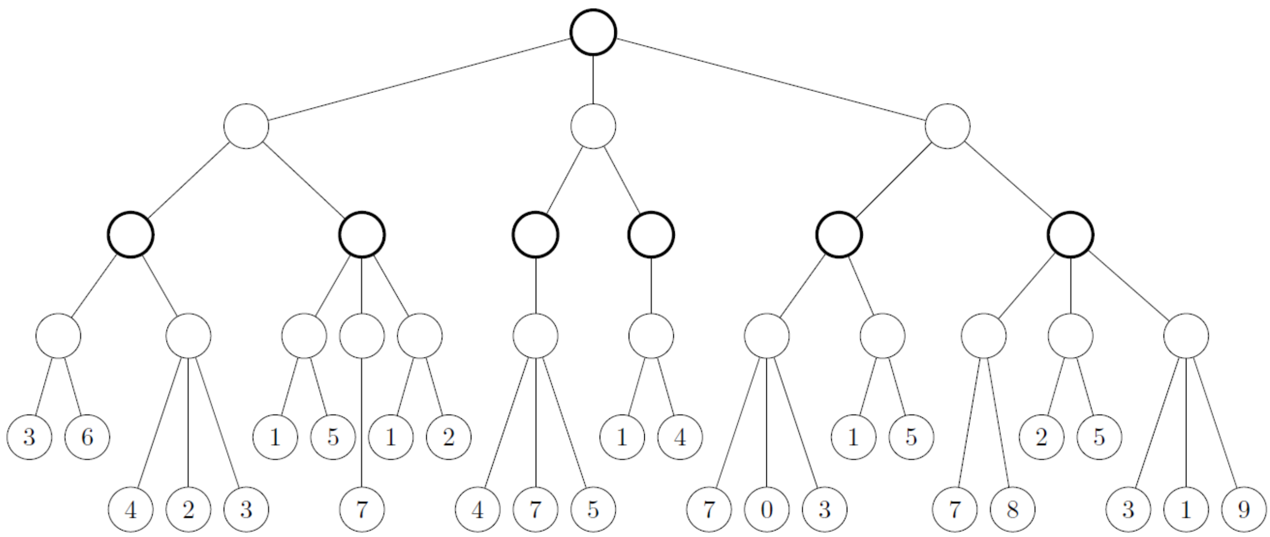
- Ο <u>Hill Climbing</u> αλγόριθμος, δεν βρίσκει καμία λύση αφού στο συγκεκριμένο γράφο, το παιδί του i με την μικρότερη τιμή της heuristic function έχει μεγαλύτερη τιμή από το i. Εν γένει, ο Hill Climbing αλγόριθμος δίνει λύσεις σε convex προβλήματα, υποσύνολο των οποίων είναι τα προβλήματα με ένα ακρότατο. Εδώ όμως υπάρχουν περισσότερα από ένα ακρότατα οπότε ο Hill Climbing δεν μπορεί να λύσει το πρόβλημα.

- Ο <u>Best First</u> αλγόριθμος, βρίσκει λύση με το μονοπάτι που φαίνεται στον ανωτέρω πίνακα. Δεν υπάρχει κάποια θεωρητική θεμελίωση για το λόγο που βρίσκει τη σωστή λύση στο συγκεκριμένο πρόβλημα, αλλά το γεγονός ότι βρίσκει λύση εξαρτάται καθαρά από το instance του προβλήματος.

- Ο <u>A* Star</u> αλγόριθμος, βρίσκει λύση και μάλιστα τη βέλτιστη, η οποία φαίνεται στον ανωτέρω πίνακα. Εν γένει, είναι γνωστό ότι βρίσκει εγγυημένα τη βέλτιστη λύση αν η heuristic function δεν υπερεκτιμά ποτέ την απόσταση ενός κόμβου από τον κόμβο προορισμού. Εδώ αντιθέτως συμβαίνει το εξής: στην τελευταία επανάληψη γίνεται μια υπερεκτίμηση, τα δύο ισοβάθμια μονοπάτια προς τον κόμβο j απορρίπτονται όμως το g έχει ήδη μπει στο τελικό μέτωπο οπότε, έχει ήδη βρεθεί η βέλτιστη λύση στο συγκεκριμένο πρόβλημα.

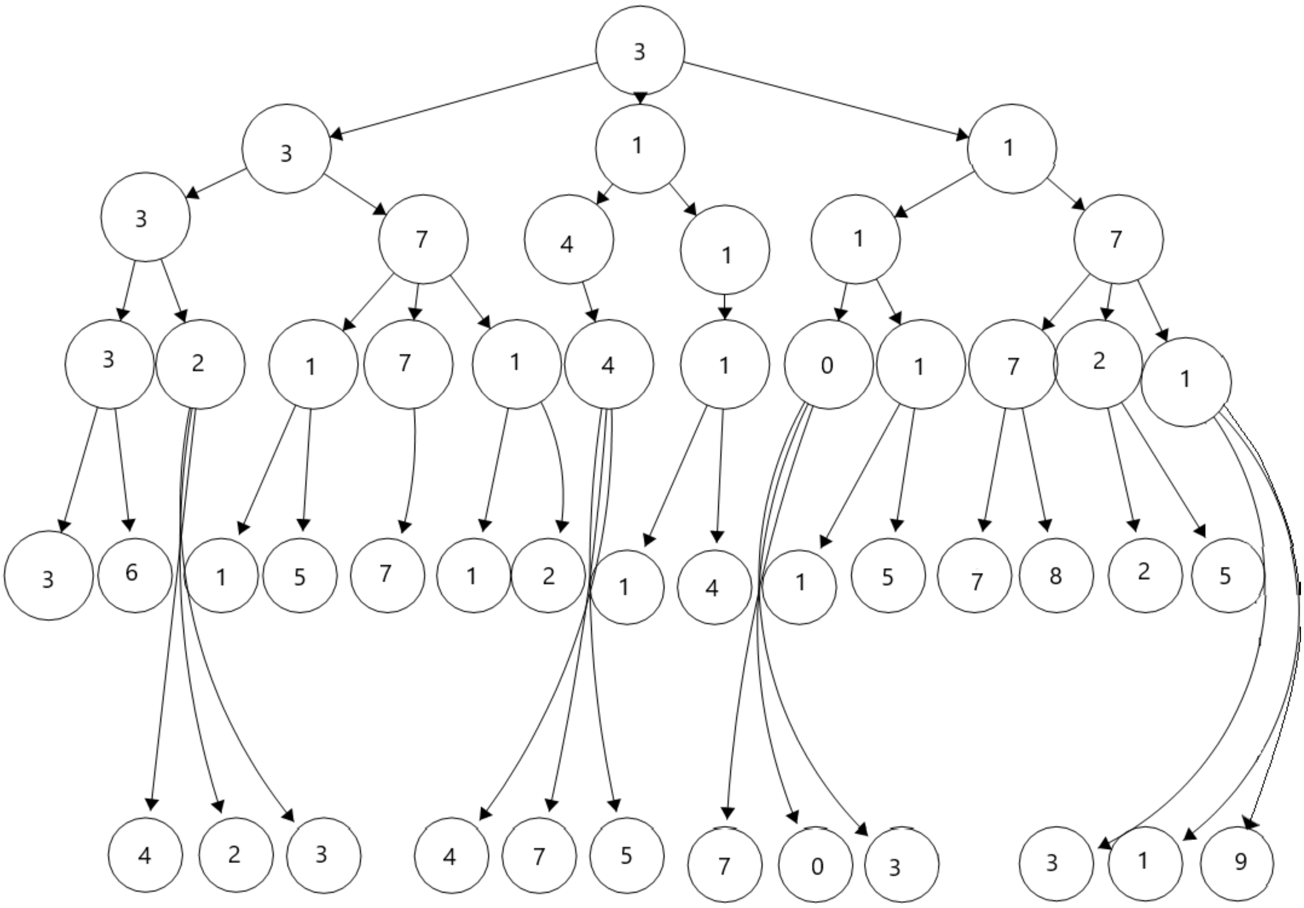
<u>2η Άσκηση</u>

Δίνεται το παρακάτω δέντρο:



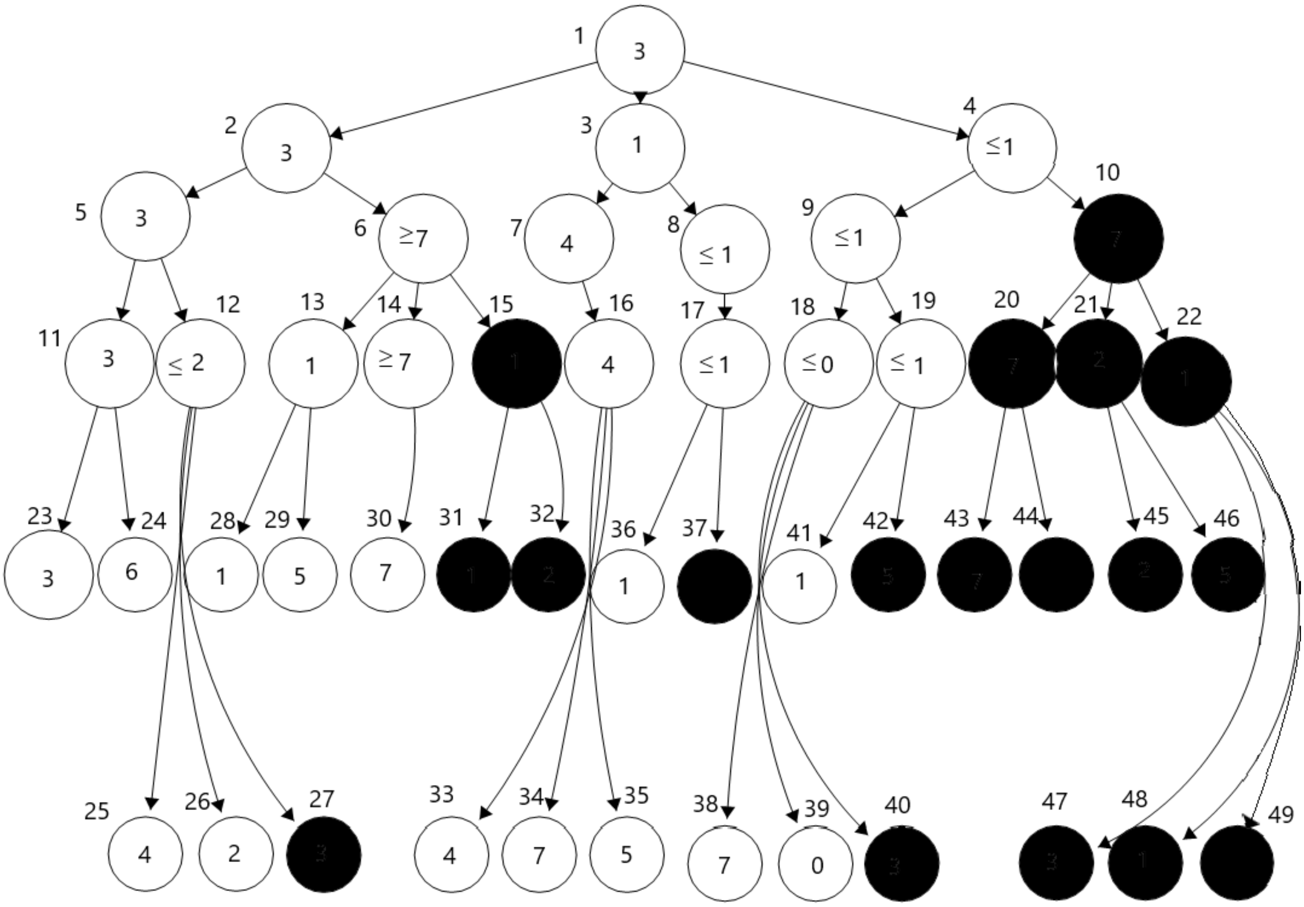
1. *Minmax*

Ο αλγόριθμος Minmax είναι ένας αλγόριθμος που επιχειρεί να βρει τη στρατηγική νίκης σε ένα παιχνίδι δύο αντιπάλων που παίζουν ο ένας μετά τον άλλον εναλλάξ. Κάθε κατάσταση αξιολογείται με έναν αριθμό που δείχνει ένα heuristic function. Αν αυτός ο αριθμός είναι πάνω από ένα threshold τότε κερδίζει ο παίκτης Α ενώ, αν είναι κάτω από ένα threshold κερδίζει ο παίκτης Β. Έτσι, ο παίκτης Α προσπαθεί να μεγιστοποιήσει τον αριθμό της κατάστασης στην οποία βρίσκεται διαλέγοντας από τις πιθανές του κινήσεις εκείνη που του δίνει τον μεγαλύτερο αριθμό. Αντίθετα, ο παίκτης Β προσπαθεί να ελαχιστοποιήσει τον αριθμό της κατάστασης στην οποία βρίσκεται διαλέγοντας από τις πιθανές κινήσεις του εκείνη που θα δώσει τον ελάχιστο αριθμό. Η εκτέλεση του αλγορίθμου για το γράφο της εκφώνησης φαίνεται παρακάτω:



2. *AB Pruning*

Ο <u>AB Pruning</u> αλγόριθμος αποτελεί μια βελτίωση του αλγορίθμου <u>Minmax</u>. Σε μια πραγματική εφαρμογή του Minmax ο υπολογισμός των τιμών του heuristic πάνω στα φύλλα είναι πραγματικά χρονοβόρα ενέργεια αφού οι τιμές των φύλλων καθορίζουν τις τιμές των υπολοίπων κόμβων. Ο AB Pruning αλγόριθμος πρόκυψε από την παρατήρηση ότι ορισμένες τιμές φύλλων (και άλλων κόμβων) δεν χρειάζεται να υπολογιστούν καθώς οι στρατηγικές των παικτών (Max για τον A και Min για τον B) εισάγουν περιορισμούς που καθιστούν αδύνατο να επηρεάζουν οι συγκεκριμένοι κόμβοι το αποτέλεσμα και συνεπώς ο υπολογισμός τους είναι περιττός. Παρακάτω, φαίνεται η εκτέλεση του AB Pruning αλγορίθμου για τον γράφο της εκφώνησης:



Η *σειρά* με την οποία επισκέπτεται ο αλγόριθμος AB Pruning τους κόμβους είναι η ακόλουθη:

1 → 2 → 5 → 11 → 23 → 24 → 12 → 25 → 26 → 6 → 13 → 28 → 29 → 14 → 30 → 3 → 7 →

→ 16 → 33 → 34 → 35 → 8 → 17 → 36 → 4 → 9 → 18 → 38 → 39 → 19 → 41