ΚΩΝΣΤΑΝΤΙΝΟΣ ΝΙΚΟΛΕΤΟΣ

1115201700104

Τεχνητή Νοημοσύνη-Εργασία 2

Πρόβλημα 1

Δεδομένα :

- $Eval(s)$ = 3$X2$ (s) + $X1$ (s) − (3$O2$ (s) + $O1$)
- $X3$ = 1 τότε utility(s,X) = +1
- $O3$ = 1 τότε utility(s,O) = -1
- Σε κάθε άλλη περίπτωση utility(s,X||O) = 0

1.

Παρατηρούμε ότι υπάρχουν 3^9 διαφορετικοί συνδυασμοί παιχνιδιών τρίλιζας που μπορούν να παιχτούν.(σε κάθε κουτάκι 3 πιθανές καταστάσεις : κενό , Ο , Χ και σύνολο 9 κουτάκια)

Κάθε φορά που πάει να παίξει ένας παίκτης έχει n αριθμό επιλογών να τοποθετήσει το σύμβολο του. Στην αρχή , ο πρώτος παίκτης “βλέπει” 9 κενές θέσεις και συμπληρώνει σε μια από αυτές το σύμβολο του. Στον επόμενο γύρο , ο άλλος παίκτης θα δει 8 διαθέσιμες θέσεις και θα συνεχίσει έτσι το παιχνίδι μέχρι κάποιος να κερδίσει ή οι θέσεις να εξαντληθούν.

Οπότε: 9*8*7*…*1 = 9! = 362.880

Ωστόσο μπορεί ένα παιχνίδι να τελειώσει πριν συμπληρωθούν όλες οι θέσεις και πρέπει να συνδέσουμε στο αποτέλεσμα τους περιορισμούς ,όπως για παράδειγμα όταν ένα σύμβολο μπεί σε μια στήλη αυτή απορρίπτεται άρα τα κουτάκια της πρέπει να αφαιρεθούν από τα πιθανά

Αν ένα παιχνίδι τελειώσει στις :

- **5 κινήσεις** (8*3!*6*5 , 8 πιθανές γραμμές: οριζόντια ,κάθετα ,διαγώνια και συμπληρώνονται με οποιονδήποτε σειρά τα δύο Χ ή Ο σε αυτούς τους συνδυασμούς): **1.440** πιθανά διαφορετικά παιχνίδια
- **6 κινήσεις** (ομοίως με το πρώτο 8*3!*6*5*4 με την διαφορά πως εάν έχει τοποθετηθεί κάποιο αντίπαλο σύμβολο μερικές στήλες ,

διαγώνιες θα έχουν καταληφθεί, οπότε πρέπει να τις αφαιρέσουμε 8*3!*6*5*4 - 6*3!*2*3!) : **5.328** πιθανά διαφορετικά παιχνίδια

- **7 κινήσεις** : **47.952** πιθανά διαφορετικά παιχνίδια
- **8 κινήσεις** : **72.576** πιθανά διαφορετικά παιχνίδια
- **9 κινήσεις** : **81.792** πιθανά διαφορετικά παιχνίδια
- **Ισοπαλία** : **46.080**

Επομένως υπάρχουν συνολικά :

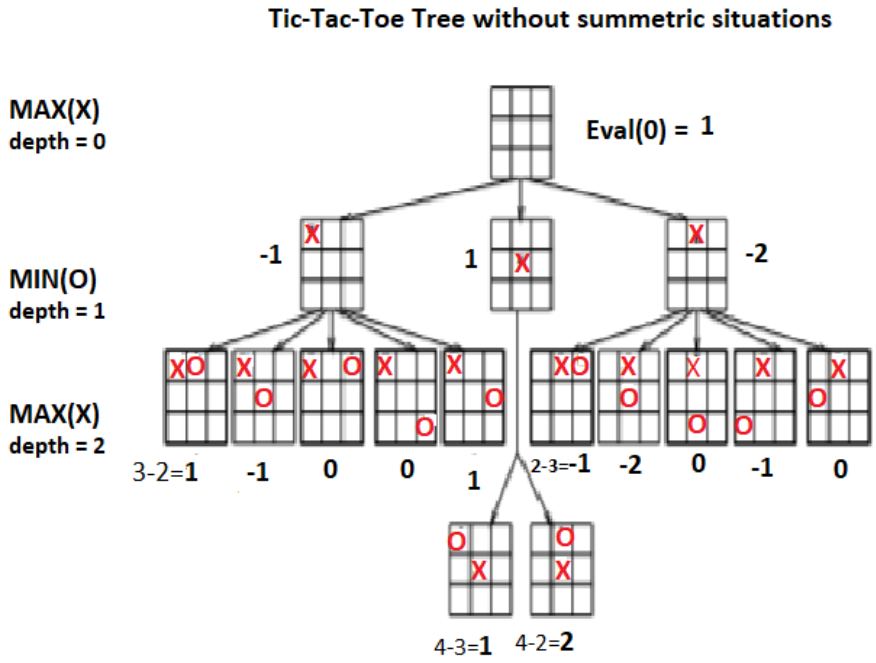
1.440 + 5.328 + 47.952 + 72.576 + + 81.792 + 46.080 = 255.168

*Οι μετρήσεις προήλθαν από έναν ιστότοπο που μελετά αυτές τις περιπτώσεις

*Παραλείπονται οι περιπτώσεις μετά τις 6 κινήσεις καθώς γίνονται με παρόμοια λογική

2. **και**
3.

Το παιχνίδι της τρίλιζας μπορεί να περιγράφει από τον αλγόριθμο minimax , όποτε για βάθος 2 , το δέντρο παιχνιδιού θα είναι (χωρίς τις συμμετρικές καταστάσεις) :



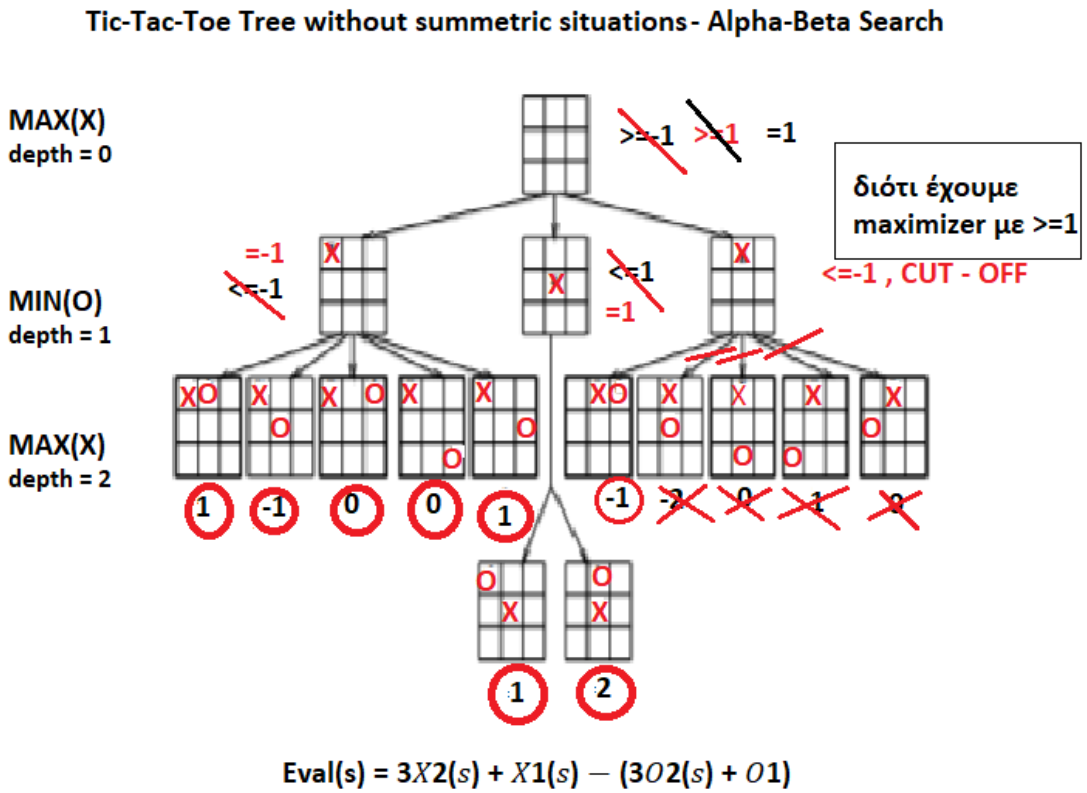
Eval(s) = 3X2(s) + X1(s) − (3O2(s) + O1)

*Οι υπολογισμοί έγιναν με την χρήση του παραπάνω τύπου

4.

Σε εκτέλεση του αλγορίθμου **MINIMAX-DECISION** , η απόφαση στη ρίζα του δέντρου θα είναι: (δέχομαι ότι η ρίζα είναι MAX) το μέγιστο από τα παιδιά δηλαδή το 1,1,1 ,όπως φαίνεται και στο σχήμα.

5. ALPHA-BETA-SEARCH



Eval(s) = 3X2(s) + X1(s) − (3O2(s) + O1)

- **Αν** οι κόμβοι παράγονταν με αντίστροφη σειρά (τα φύλλα από όσο καταλαβαίνω) , το κλάδεμα θα γινόταν στον αριστερό κλάδο και όχι στον δεξί κλάδο. Αυτό θα συνέβαινε καθώς η λύση βρίσκεται στην μέση και στις 2 περιπτώσεις.
- Η βέλτιστη σειρά παραγωγής των κόμβων είναι να είναι στην πρώτη θέση που ανακαλύπτει η αναδρομή (έστω στο παράδειγμα η αριστερή) , έτσι ώστε να ανακαλύπτεται πρώτη. Ακόμη εάν η επιθυμητή θέση είναι στα αριστερά και είναι ο maximizer στην ρίζα ,τότε εάν ο maximizer(>=1) έχει το α μεγαλύτερο από το μικρότερο που θα επιστρέψει ο minimizer (<=-1) ,τότε θα γίνει κλάδεμα. Συνοψίζοντας , η σειρά που μας εξυπηρετεί είναι ο maximizer να βρίσκει στις αρχικές αναδρομές τον μεγαλύτερο και ο minimizer ομοίως

Πρόβλημα 2

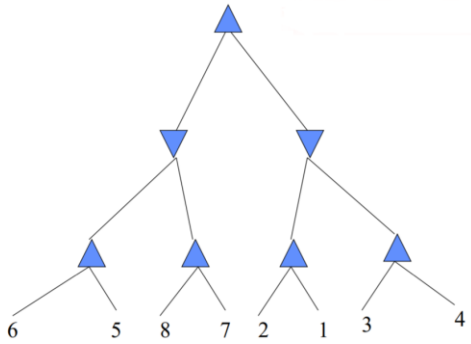
Η τεχνική άλφα-βήτα , γενικά , εφαρμόζεται σε δέντρα minimax και επιστρέφει την ίδια κίνηση που θα επέστρεφε η minimax ,αλλά εξαλείφει τους κλάδους που δεν είναι δυνατόν να επηρεάσουν την τελική απόφαση . Η αποτελεσματικότητα του κλαδέματος άλφα-βήτα εξαρτάται από σε μεγάλο βαθμό από την σειρά με την οποία εξετάζονται οι διάδοχοι. Οπότε :

1. Κλάδεμα μέγιστου αριθμού κόμβων (Best case)

Οι τιμές χρησιμότητας στα φύλλα θα πρέπει να είναι σε φθίνουσα σειρά για κάθε τελικό κόμβο, έτσι ώστε να εξερευνώνται πρώτοι οι επιθυμητοί κόμβοι. Έτσι στη θέση άλφα θα είναι πάντα η καλύτερη επιλογή του MAX (ο μεγαλύτερος αριθμός -> φθίνουσα σειρά) και στην βήτα η καλύτερη αντίστοιχα του βήτα.

Πολυπλοκότητα: $O(sqrt(b^d))$

Για παράδειγμα (με μεγαλύτερο ύψος από 2) :

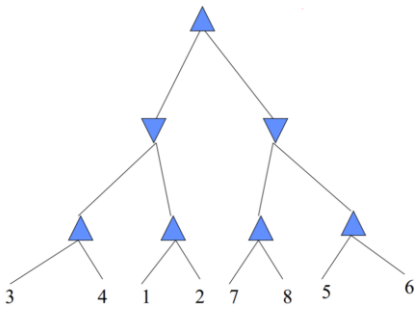


2. Κλάδεμα ελάχιστου αριθμού κόμβων (Worst case)

Οι τιμές χρησιμότητας στα φύλλα θα πρέπει να είναι σε αύξουσα σειρά για κάθε τελικό κόμβο, έτσι ώστε να εξερευνώνται τελευταίοι οι επιθυμητοί κόμβοι.

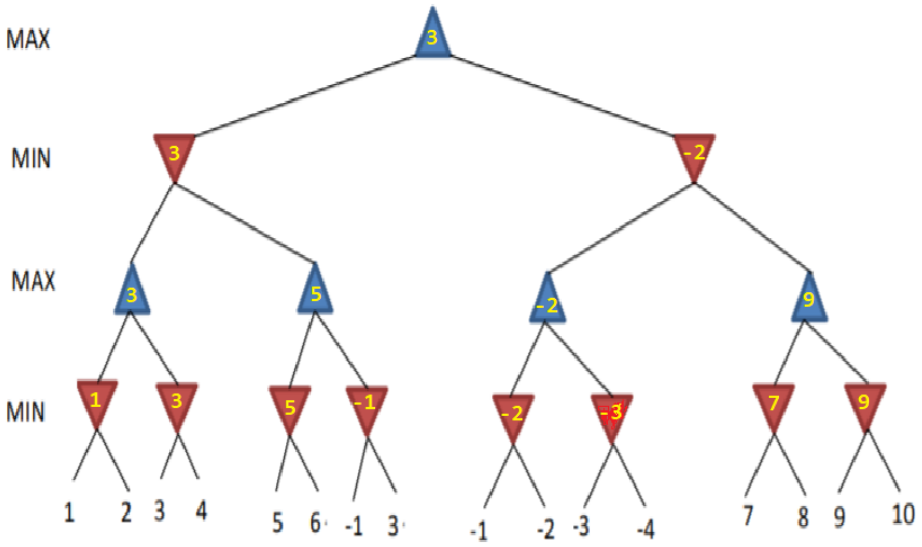
Πολυπλοκότητα: $O(b^d)$

Για παράδειγμα: (με μεγαλύτερο ύψος από 2)



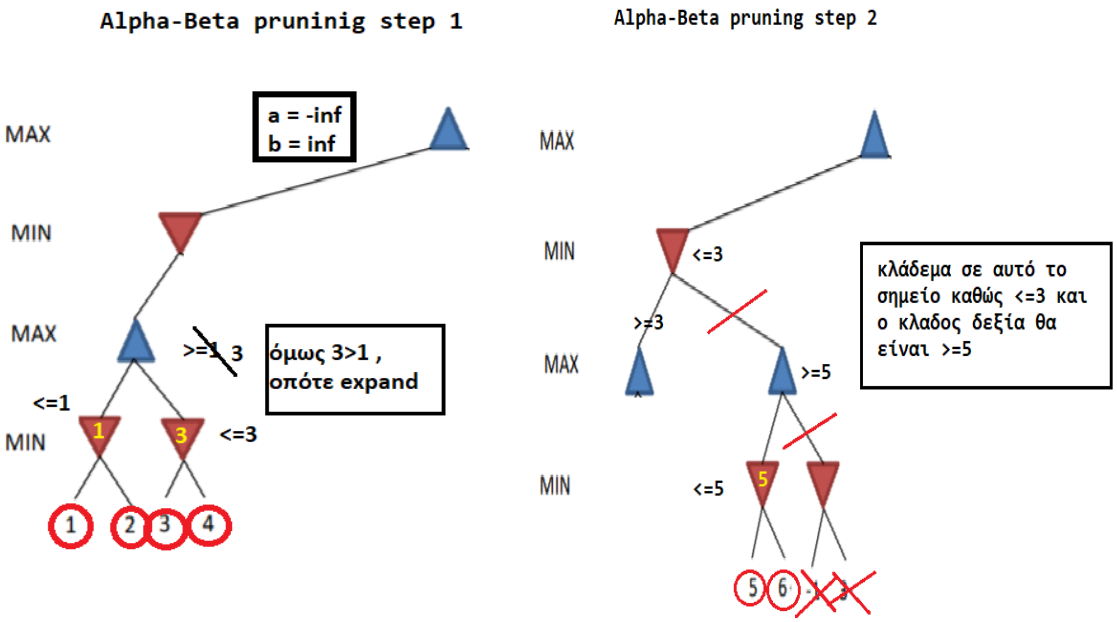
Πρόβλημα 3

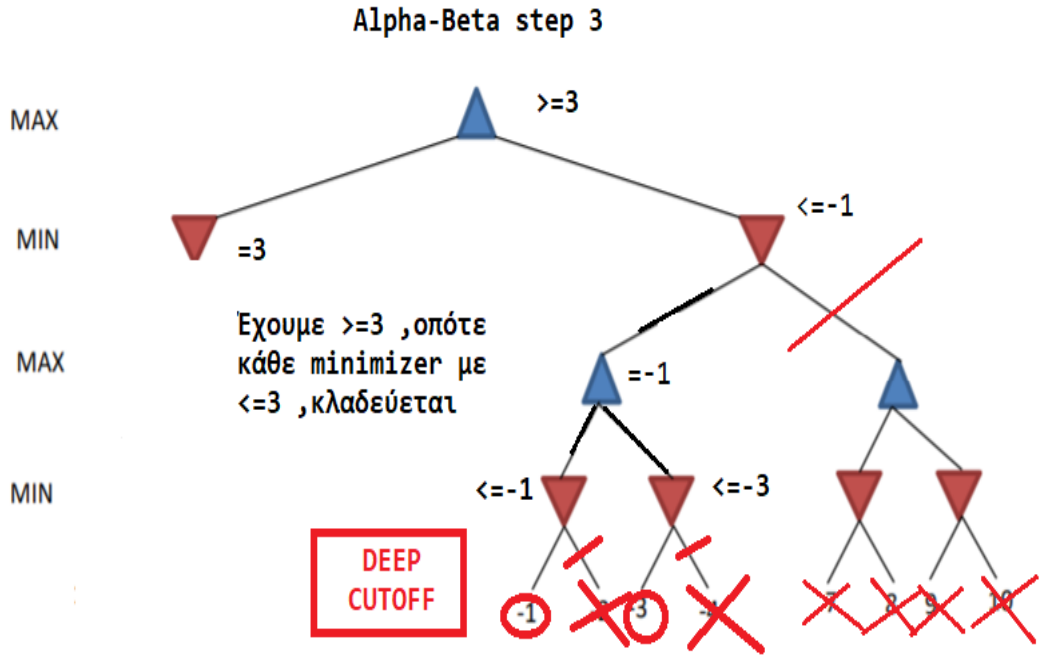
(α) minimax τιμή για κάθε κόμβο:



(β) Η minimax απόφαση στη ρίζα του δέντρου είναι το 3 , δηλαδή θα ακολουθήσει το μονοπάτι 3,3,3,3

(γ) ALPHA-BETA-SEARCH :



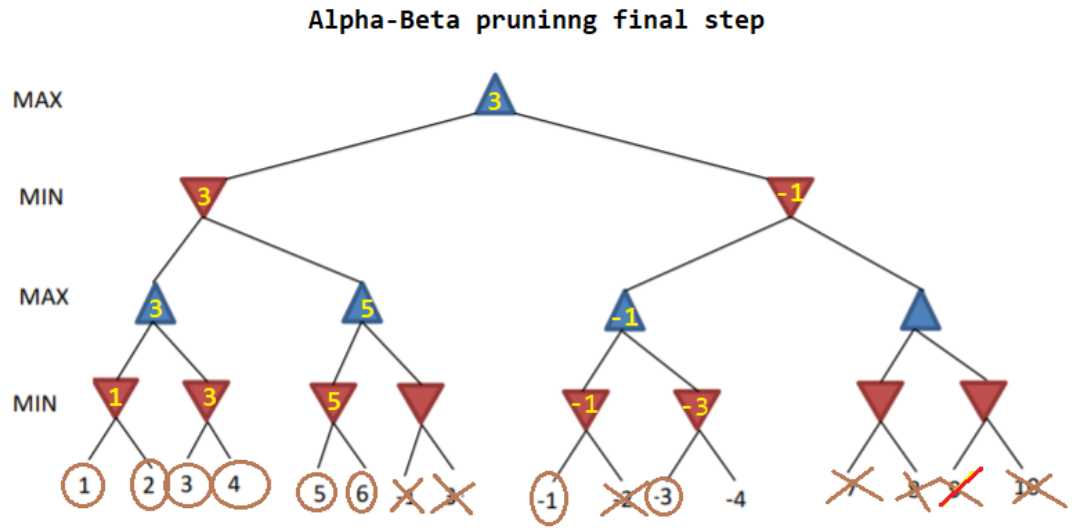


- Οι κόμβοι που τελικώς κλαδεύονται φαίνονται στο σχήμα , καθώς και οι επιλογές που γίνονται
- Η αναδρομή ξεκίνα από αριστερά προς τα δεξιά και για αυτό το λόγο στον δεξί κλάδο της ρίζας παρατηρείται το φαινόμενο DEEP-CUTOFF ,με το οποίο κλαδεύεται ολόκληρος κλάδος , όπως φαίνεται και στο σχήμα.

*Με κόκκινο κύκλο οι κόμβοι που επισκέφθηκαν

*Με Χ αυτοί που κλαδεύτηκαν

Τελικά:



Πρόβλημα 4

(α)

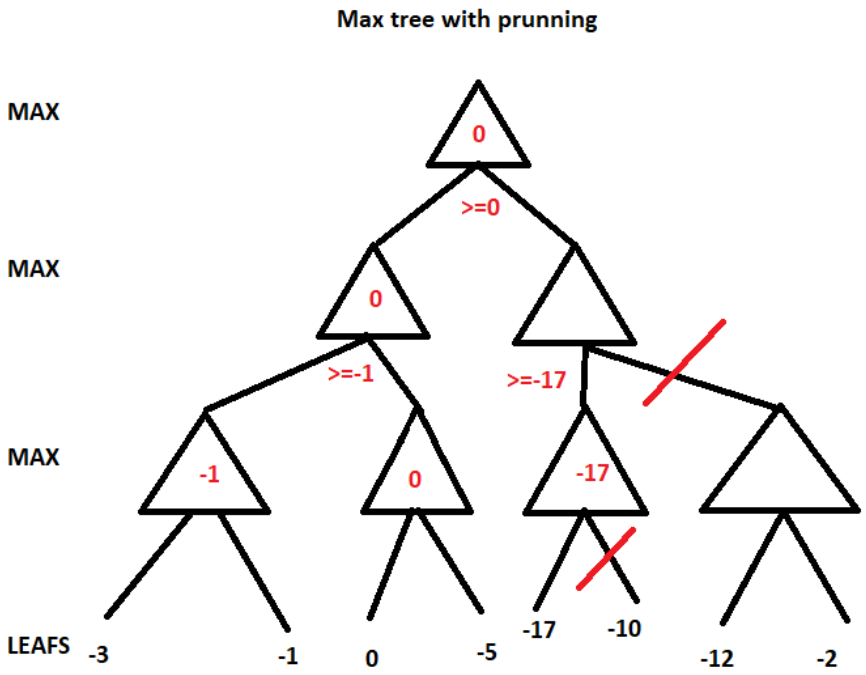
Ο αλγόριθμος αυτός που ζητείται δεν είναι εφικτός . Αυτό συμβαίνει καθώς σε κάθε επίπεδο έχουμε maximizer παίκτη και συνεπώς τα διαστήματα που αναζητούμε για κλάδεμα δεν θα υπάρξουν . Γενικά ένα κλάδεμα προέρχεται από σύγκριση διαστημάτων και ύπαρξη άνω η κάτω φράγματος. Επομένως αυτός ο αλγόριθμος δεν είναι εφικτός.

(β)

Στην περίπτωση ενός δέντρου expectimax , αυτό που ρωτάτε είναι αδύνατο ,καθώς για να υπολογιστεί η πιθανότητα στους κόμβους, χρειάζεται να τους επισκεφθούμε όλους ,να προσπελάσουμε τις τιμές στα φύλλα και τελικά να υπολογίσουμε την πιθανότητα .Επομένως είναι αδύνατο το κλάδεμα σε αυτήν την περίπτωση.

(γ)

Στην περίπτωση ενός δέντρου max με αρνητικές και μηδενικές τιμές , είναι εφικτό να κλαδέψουμε κλαδιά ,καθώς το κλάδεμα δεν εξαρτάται από το είδος των αριθμών αλλά από την σύγκριση μεταξύ τους. Αυτό που συμβαίνει σε αυτήν την περίπτωση είναι ότι έχουμε το άνω φράγμα που ζητάμε και είναι το 0 . Δηλαδή σε περίπτωση που σε έναν κλάδο βρεθεί το 0 , μπορούμε να κλαδέψουμε τους άλλους.



(δ)

Αντίθετα με την περίπτωση (γ) ,αν και έχουμε άνω φράγμα , πρέπει να υπολογίσουμε την πιθανότητα για τους κόμβους CHANCE ,που σημαίνει πως πρέπει να τους επισκεφθούμε. Επομένως αυτή η πρόταση δεν είναι εφικτή .

(ε)

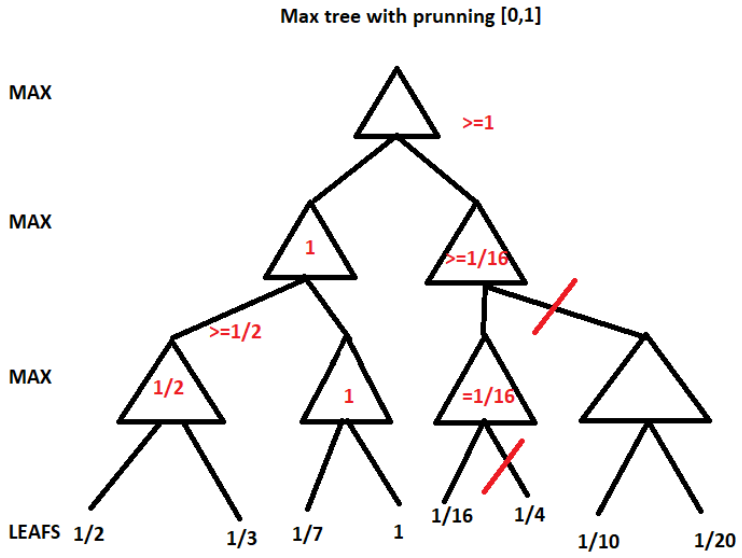
Επειδή το δέντρο είναι max χρειαζόμαστε άνω φράγμα για να κλαδέψουμε . Στο διάστημα [0,+inf] το άνω φράγμα είναι το +inf που σημαίνει πως δεν έχουμε κάποιο συγκεκριμένο άνω φράγμα και συνεπώς θα πρέπει να εξετάσουμε όλους τους κόμβους για να βρούμε τον μέγιστο κάθε φορά. Η πρόταση δεν είναι εφικτή.

(στ)

Για λόγους που ανέφερε δεν μπορούμε να έχουμε κλάδεμα σε αλγόριθμο expectiminimax . Οπότε η πρόταση αυτή είναι αδύνατη.

(ζ)

Στο διάστημα [0,1] το κλάδεμα σε δέντρο max είναι εφικτό . Δεχόμαστε ως άνω φράγμα το 1. Για παράδειγμα:



(η)

Για λόγους που ανέφερε δεν μπορούμε να έχουμε κλάδεμα σε αλγόριθμο expectiminimax . Οπότε η πρόταση αυτή είναι αδύνατη.

Πρόβλημα 5

Έγινε υλοποίηση του **Pacman-project-2**