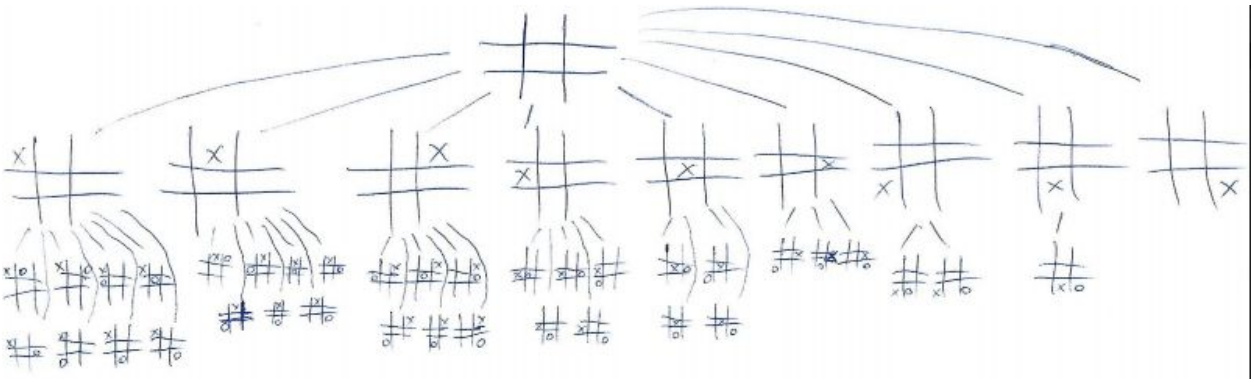
Ονοματεπώνυμο: Παντελεήμων Μαλέκας
Α.Μ: 1115201600268

- Πρόβλημα 1

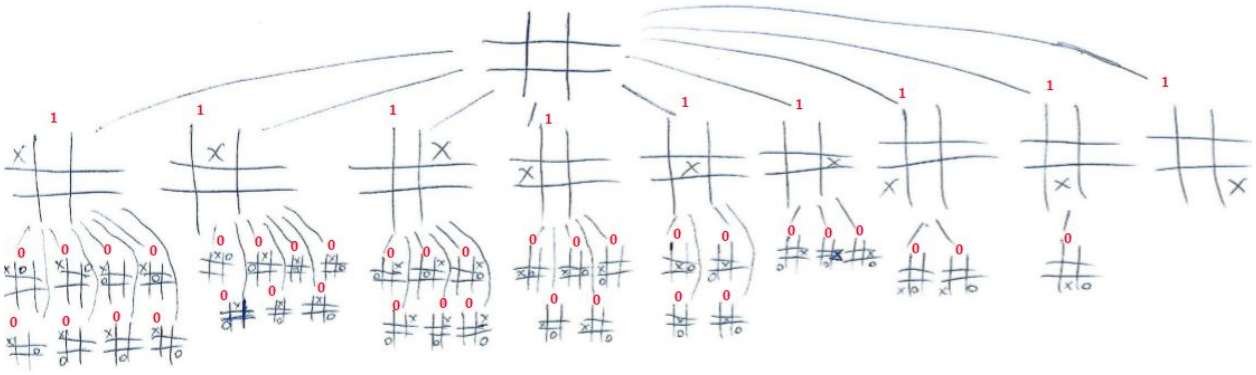
1. Έχουμε ένα σύνολο από 9! = 362880 πιθανές κινήσεις στο κλασικό παιχνίδι τρίλιζας. Αυτός ο αριθμός υπολογίζεται βλέποντας ότι εφόσον έχουμε 9 κουτάκια υπάρχουν 9 πιθανοί τρόποι να τοποθετηθεί το πρώτο σημάδι, μετά μένουν 8 τρόποι για το επόμενο κ.ο.κ. Από αυτόν τον αριθμό έχουμε ότι τα νικητήρια παιχνίδια (χωρίς να λάβουμε υπόψιν συμμετρικές καταστάσεις) μπορούν να είναι 255168 τα οποία υπολογίζονται παρακάτω:
   Τα παιχνίδια που τελειώνουν μετά από 5 κινήσεις είναι 8 * 3! * 6 * 5 = 1440. Έχουμε 8 νικητήριες γραμμές για τον παίκτη Χ (οι 3 κάθετες, οι 3 οριζόντιες και οι 2 διαγώνιοι) και μετά τα άλλα δύο Ο θα είναι τοποθετημένα στα υπόλοιπα 6 κουτάκια σε οποιαδήποτε σειρά.
   Τα παιχνίδια που τελειώνουν μετά από 6 κινήσεις είναι 5328. Αρχικά βλέπουμε με παρόμοια λογική με πριν ότι έχουμε 8 * 3! * 6 * 5 * 4 = 5760 τελικά παιχνίδια. Όμως επειδή δεν μπορούν να υπάρχουν ταυτόχρονα τρεις νικητήριες γραμμές και για τον Χ και για τον Ο αφαιρούμε τις διπλότυπες περιπτώσεις: δεν μπορούμε να έχουμε διαγώνιους και αν έχει καταληφθεί μία γραμμή τότε μένουν δύο άλλες γραμμές. Άρα αφαίρουμε 6 * 3! * 2 * 3! = 432 περιπτώσεις. Οπότε έχουμε συνολικά 5760 - 432 = 5328 παιχνίδια.
   Με παρόμοια λογική βρίσκουμε ότι τα παιχνίδια που τελειώνουν μετά από 7 κινήσεις είναι 47952, μετά από 8 κινήσεις είναι 72576 και μετά από 9 κινήσεις είναι 127872. Προσθέτοντας αυτούς τους αριθμούς βρίσκουμε ότι όλα τα διαφορετικά παιχνίδια τρίλιζας που μπορούν να παιχτούν είναι 255168. Επίσης, δεν υπάρχουν παιχνίδια που να τελειώνουν μετά από 1,2,3 ή 4 κινήσεις γιατί δεν αρκούν αυτές οι κινήσεις για να σχηματιστεί τριάδα.

2. Ακολουθεί σχηματική απεικόνιση του ζητούμενου δέντρου:



Να σημειωθεί ότι στο δέντρο έχουν αφαιρεθεί οι συμμετρικές καταστάσεις. Το τελευταίο υποδέντρο π.χ. λοιπόν δεν θα έχει κανένα υποδέντρο από κινήσεις του Ο γιατί έχουν καλυφθεί στα προηγούμενα υποδέντρα.

3. Από την evaluation function που μας δίνει η εκφώνηση:
   (Eval(s) = $3X_2(s) + X_1(s) - (3O_2(s) + O_1(s))$) παρατηρούμε ότι μπορεί να γραφτεί ως:
   $X_1(s) - O_1(s)$ μιας και στο ζητούμενο δέντρο δεν έχουμε καμία δυάδα. Οπότε στις πρώτες 9 κινήσεις του Χ θα δοθεί η τιμή 1 από την συνάρτηση χρησιμότητας γιατί έχουμε μόνο ένα Χ. Στα υπόλοιπα υποδέντρα έχουμε ένα Χ και ένα Ο άρα η συνάρτηση δίνει σε όλα 0. Ακολουθεί σχηματική απεικόνιση με τις τιμές της συνάρτησης σημειωμένες με κόκκινο χρώμα.



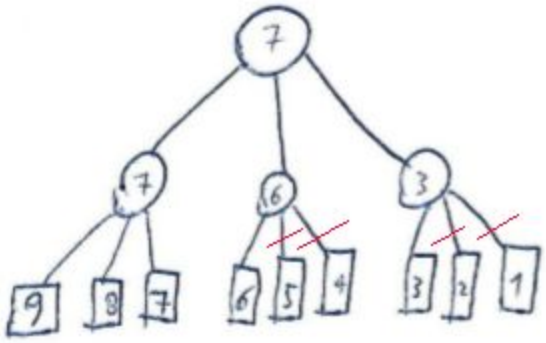
4. Η minimax απόφαση στην ρίζα του δέντρου θα είναι 1. Παρατηρούμε ότι αν εκτελέσουμε τον αλγόριθμο minimax με το που φτάσουμε σε μία κατάσταση φύλλο εφόσον δεν έχουμε άλλη επόμενη κατάσταση καλούμε την συνάρτηση χρησιμότητας η οποία δίνει την τιμή 0 για τους λόγους που εξηγήθηκαν στο προηγούμενο ερώτημα. Ωστόσο ο τελευταίος Min κόμβος (αυτός δηλαδή που έχει το Χ στην κάτω δεξιά γωνία) είναι και αυτός κατάσταση φύλλο αλλά η συνάρτηση του δίνει τιμή χρησιμότητας 1. Άρα έχουμε ότι ο αλγόριθμος στο επίπεδο του Min (δηλάδη στις καταστάσεις όπου έχει τοποθετηθεί μόνο ένα Χ) θα επιλέξει πάντα το 0 με εξαίρεση τον τελευταίο που θα έχει 1. Ο ρίζα είναι Max κόμβος άρα ο αλγόριθμος θα επιλέξει την μεγαλύτερη τιμή η οποία θα είναι 1.

5. Το πόσοι κόμβοι θα κλαδευτουν εξαρτάται αν οι εντολές κλαδέματος χρησιμοποιούν γνήσιες ανισότητες ή όχι. Στα προηγούμενα ερωτήματα είδαμε ότι οι καταστάσεις φύλλα στο ζητούμενο δέντρο έχουν όλες τιμή 0. Άρα αν ο αλγόριθμος χρησιμοποιεί γνήσιες ανισότητες τότε δεν θα κλαδευτεί κανένας κόμβος γιατί στην σύγκριση θα έχουμε πάντα 0<0 το οποίο δεν ισχύει ποτέ άρα δεν θα έχουμε κανένα κλάδεμα. Αν δεν χρησιμοποιούμε γνήσιες ανισότητες τότε στον πρώτο Min κόμβο θα εξεταστούν όλα τα υποδέντρα και το β θα γίνει 0. Προχωρώντας στην ρίζα το α θα γίνει με την σειρά του 0. Οπότε λοιπόν σε κάθε επόμενο Min κόμβο θα εξεταστεί ο πρώτος γείτονας του και θα

κλαδευτουν όλοι οι υπόλοιποι εφόσον θα ισχύει 0≤0. Άρα σε αυτήν την περίπτωση θα κλαδευτουν 21 κόμβοι. Αν οι κόμβοι παράγονται με την αντίστροφη σειρά δεν θα υπάρξει καμία διαφορά. Ο λόγος είναι επειδή αν παραχθεί πρώτα ο τελευταίος κόμβος (εκείνος δηλαδή που έχει το Χ στην κάτω δεξιά γωνία) θα χρειαστεί να παράξει τα οκτώ επόμενα υποδέντρα γιατί τώρα δεν έχουμε συμμετρικές καταστάσεις σε κάποιο προηγούμενο υποδέντρο. Άρα και με την αντίστροφη σειρά το ζητούμενο δέντρο θα είναι αντίστοιχο με το πρώτο, οπότε δεν θα αλλάξει κάτι. Για αυτόν τον λόγο δεν υπάρχει βέλτιστη σειρά παραγωγής των κόμβων.

- Πρόβλημα 2

Για να είναι μέγιστος ο ζητούμενος αριθμός χρειάζεται οι τιμές στα φύλλα του δέντρου να παράγονται με φθίνουσα ταξινόμηση. Π.χ. στο δέντρο που ακολουθεί:

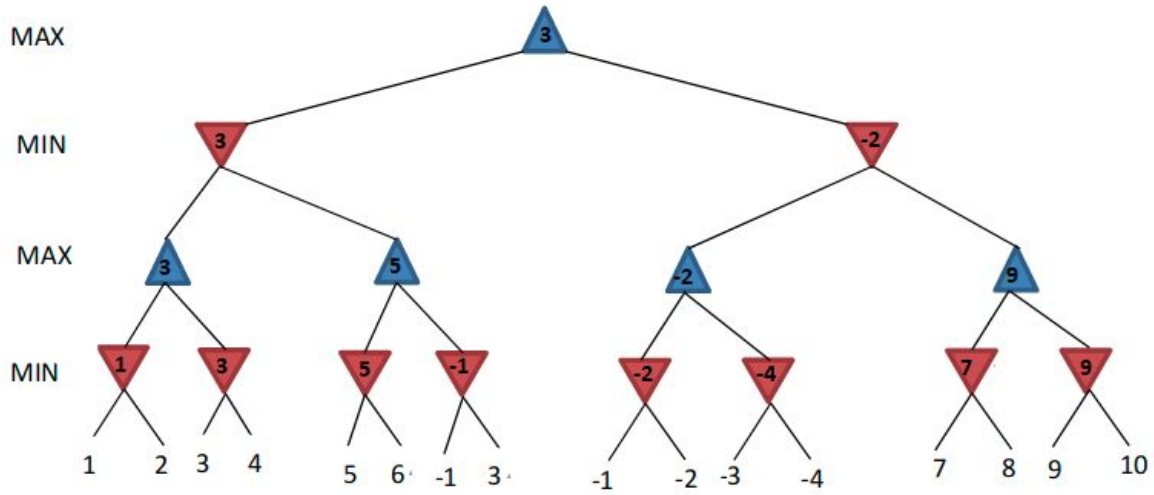


Αφού παραχθούν οι πρώτες τρεις τερματικές καταστάσεις ο α θα έχει πάρει την τιμή 7. Οπότε στον επόμενο min κόμβο θα εξεταστεί αν 6≤7 που θα ισχύει άρα θα κλαδέψουμε τους υπόλοιπους κόμβους. Αντίστοιχα θα έχουμε το ίδιο αποτέλεσμα και στον 3ο min κόμβο. Οπότε θα έχουμε κλαδέψει τον μεγαλύτερο αριθμό κόμβων που μπορούμε σε αυτό το δέντρο.

Για να είναι ελάχιστος αυτός ο αριθμός τότε οι τιμές αρκεί να παράγονται με αύξουσα σειρά. Στο ίδιο δέντρο με πριν, αν οι τιμές ήταν με την σειρά 123456789 τότε ο α θα είχε την τιμή 1 μετά από τον έλεγχο των τριων πρώτων τερματικών καταστάσεων. Οπότε στον επόμενο min κόμβο θα εξεταστεί αν 4≤1 που δεν θα ισχύει. Θα γίνουν ανάλογοι έλεγχοι μέχρι την τελευταία τερματική κατάσταση και εν τέλει δεν θα έχει κλαδευτεί κανένας κόμβος.

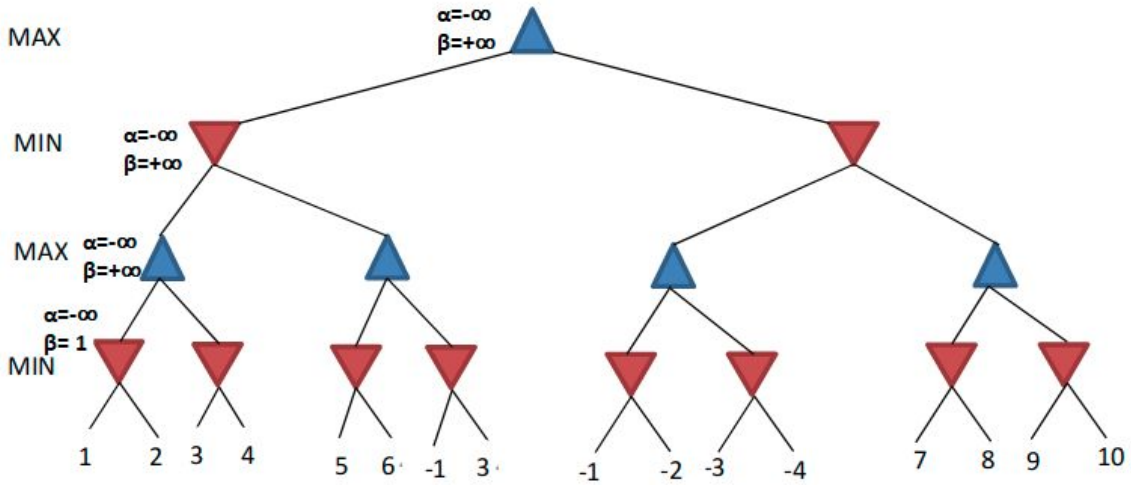
- Πρόβλημα 3

1. Ακολουθεί απεικόνιση με τις ζητούμενες τιμές:

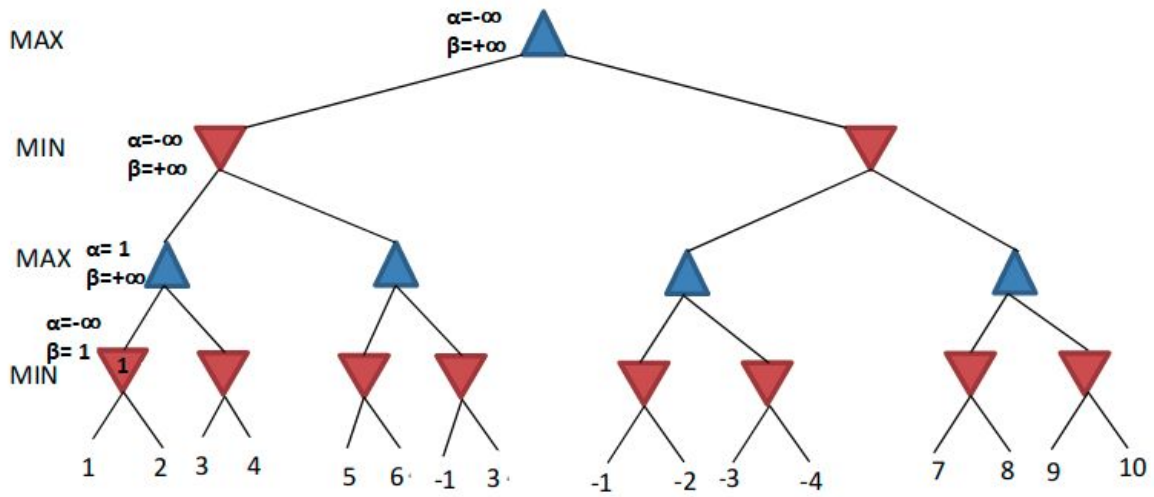


2. Η minimax απόφαση στην ρίζα του δέντρου θα είναι 3 όπως φαίνεται στο προηγούμενο σχήμα.

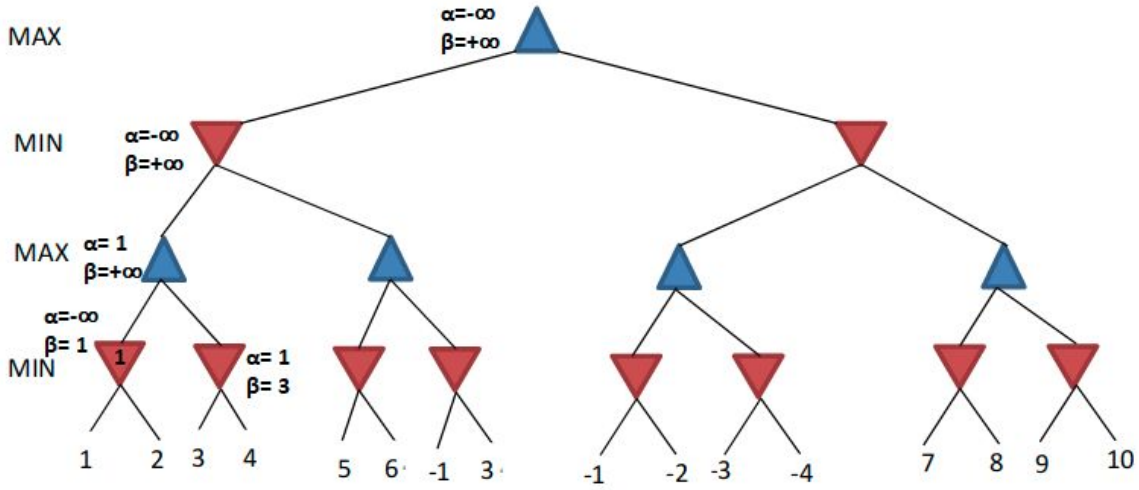
3. Ακολουθεί αναλυτική εξήγηση από την εκτέλεση του αλγόριθμου



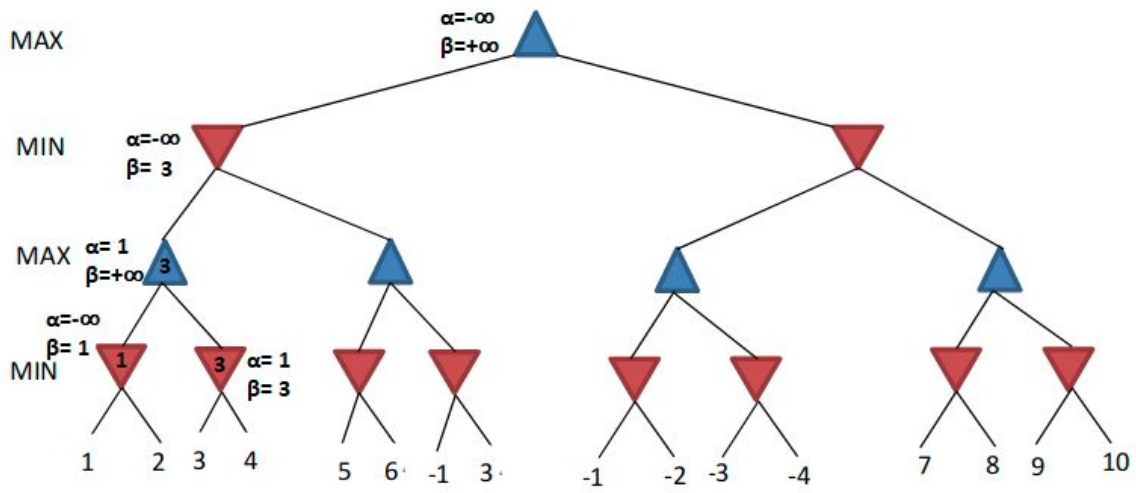
Ξεκινάμε δίνοντας τις τιμές -∞ και +∞ στο α και β, αντίστοιχα. Προχωράμε σε κάθε γειτονικό κόμβο περνώντας τις τιμές των α και β. Στον πιο αριστερό κόμβο βλέπουμε ότι έχει τις τερματικές καταστάσεις 1 και 2. Το β ανανεώνεται ως 1 μιας και είναι το μικρότερο από αυτά.



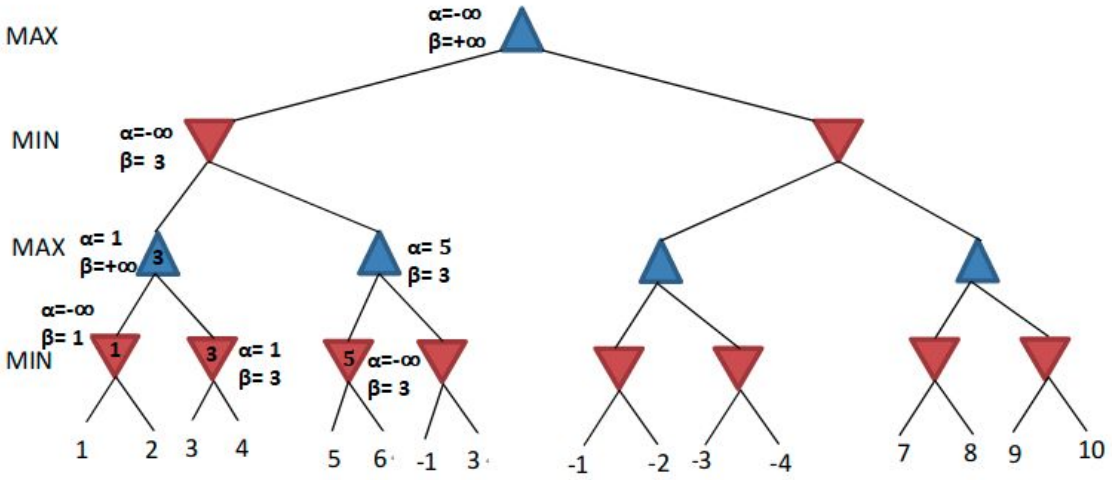
Ο αλγόριθμος επιλέγει να επιστρέψει την τιμή 1 στον κόμβο Min μιας και είναι η μικρότερη τιμή. Προχωρώντας προς τα πάνω, η τιμή του α ανανεώνεται σε 1 μιας και είναι μεγαλύτερη από το $-\infty$.



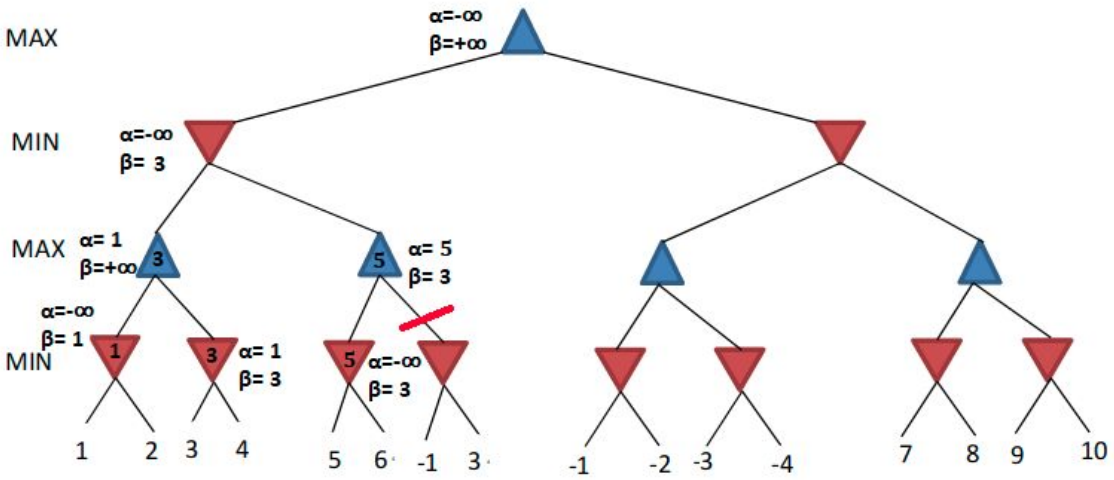
Περνάμε τις τιμές των α και β από τον Max κόμβο στον επόμενο Min κόμβο. Αυτός έχει τις τερματικές καταστάσεις 3 και 4 και η τιμή του β από $+\infty$ ανανεώνεται σε 3.



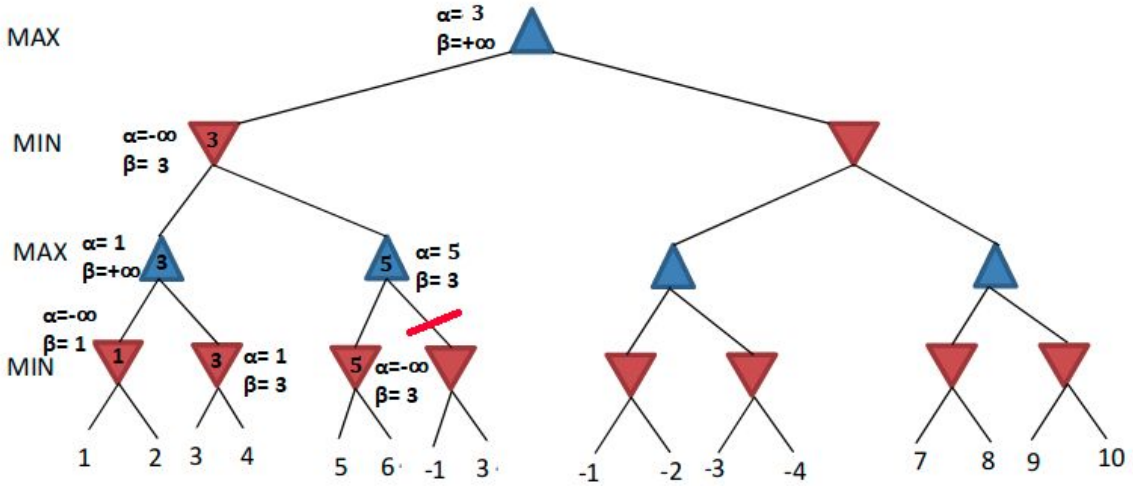
Ο αλγόριθμος Minimax επιστρέφει 3 στον Min κόμβο μιας και είναι η μικρότερη τιμή. Προχωρώντας στον επάνω Max κόμβο ο αλγόριθμος επιστρέφει την τιμή 3 μιας και έχουμε βρει τις τιμές από τους γείτονες του. Επίσης η τιμή β από τον επάνω Min κόμβο ανανεώνεται σε 3.



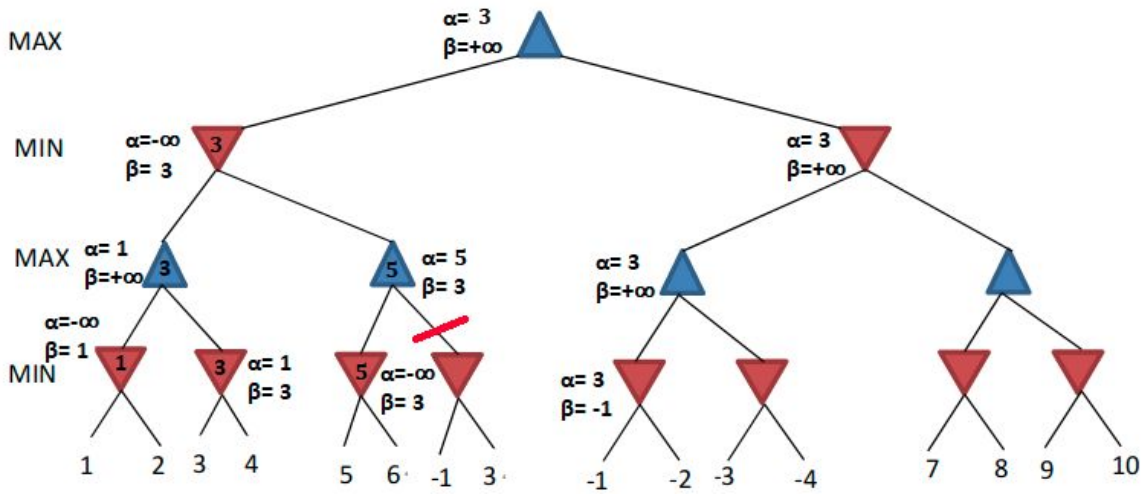
Περνάμε τις τιμές του Min κόμβου μέχρι που φτάνουμε στον Min κόμβο που έχει τις τερματικές καταστάσεις 5 και 6. Ο αλγόριθμος επιστρέφει την τιμή 5 σε αυτόν τον κόμβο και η τιμή του α στον επάνω Max κόμβο γίνεται 5.



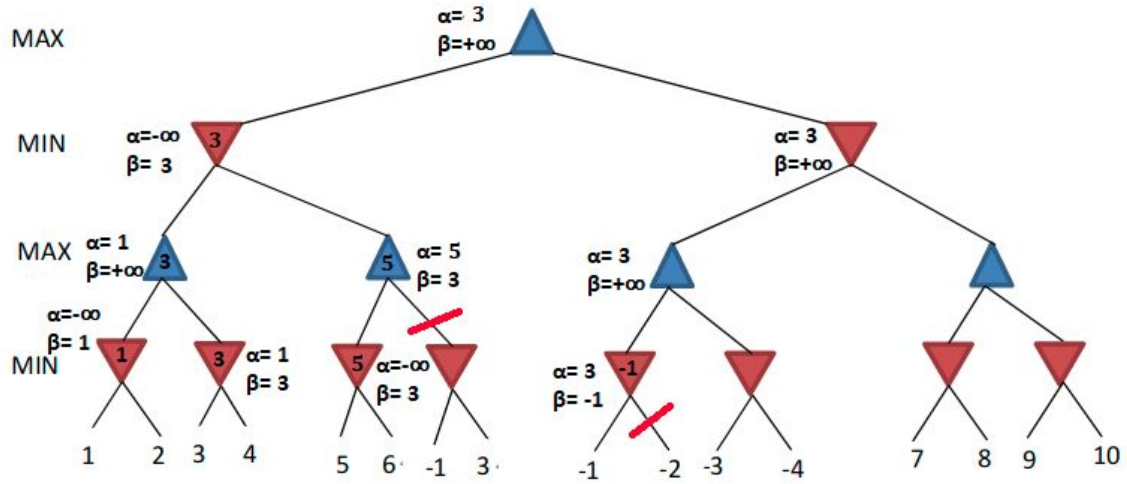
Σε αυτό το σημείο ο αλγόριθμος βρίσκεται στον Max κόμβο και συγκρίνει αν η τιμή του γείτονα είναι μεγαλύτερη του β, δηλαδή αν ισχύει 5≥3. Το οποίο είναι αληθές άρα επιστρέφεται η τιμή 5 στον κόμβο και κλαδευεται ο άλλος γειτονικός Min κόμβος.



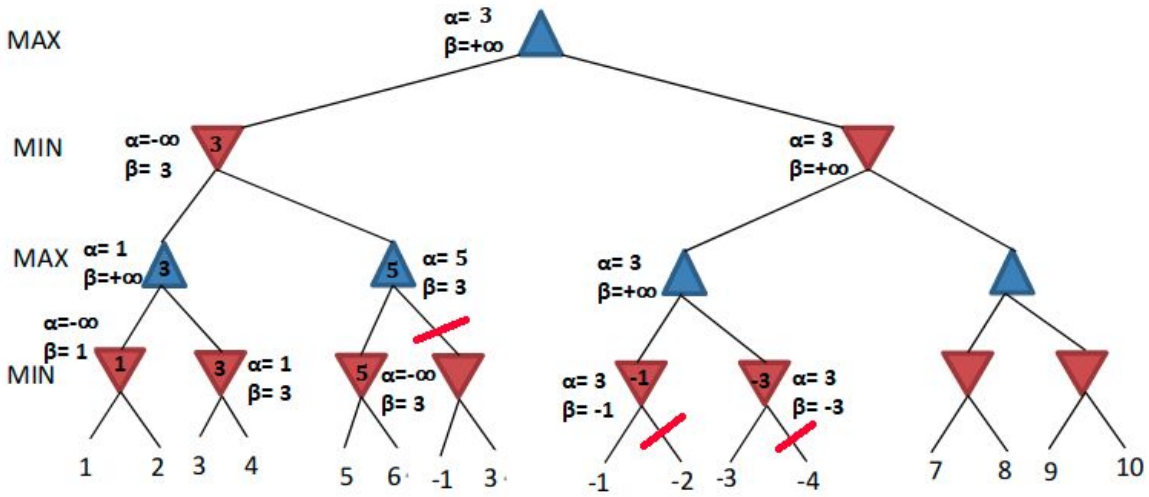
Στον αμέσως επάνω Min κόμβο επιλέγεται η τιμή 3 εφόσον γνωρίζουμε τις τιμές των γειτόνων. Επισης η τιμή του α στην ρίζα γίνεται 3.



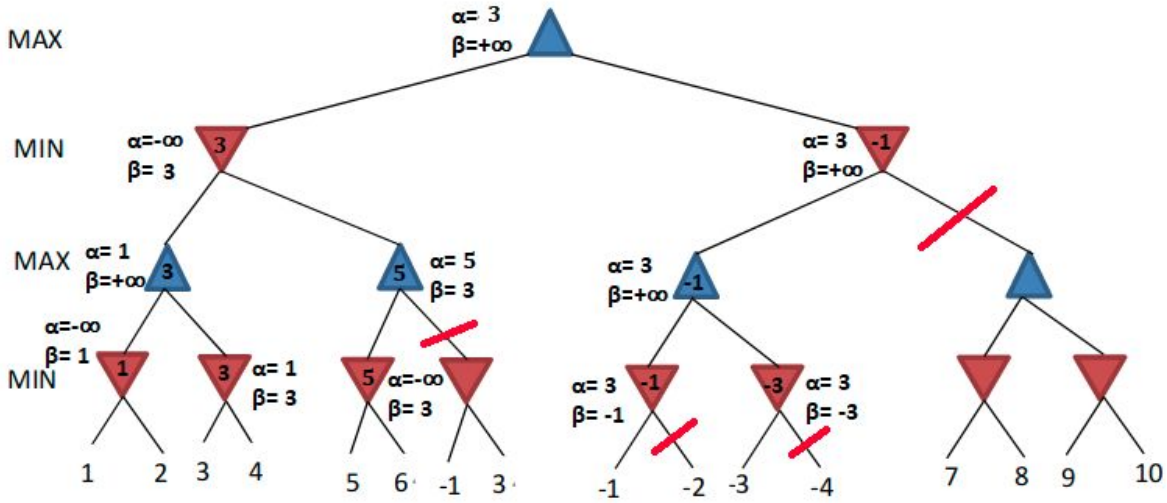
Περνάμε τις τιμές των α και β απο την ρίζα μέχρι τον Min κόμβο που έχει τις τερματικές καταστάσεις -1 και -2. Εκεί πέρα η τιμή του β γίνεται -1.



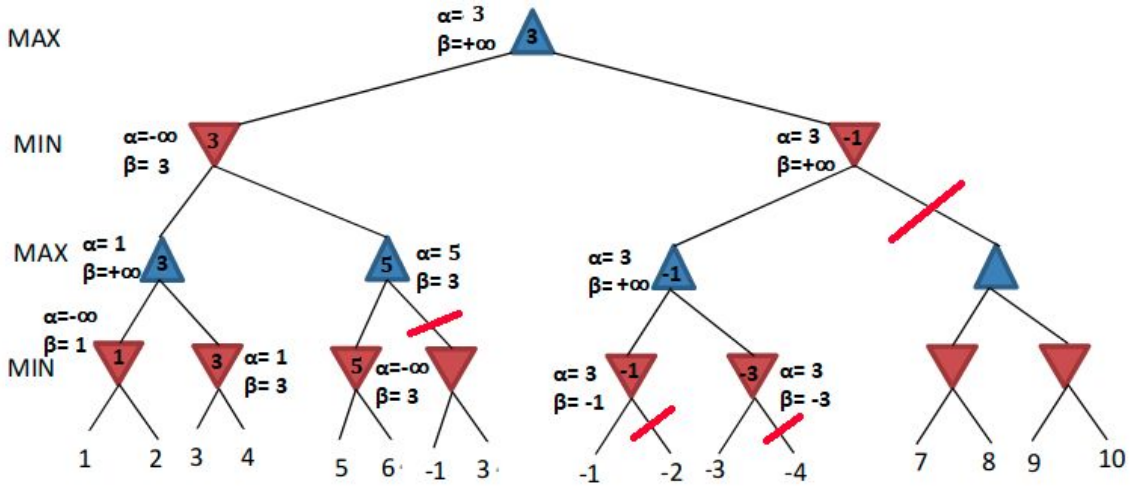
Σε αυτό το σημείο ο αλγόριθμος συγκρίνει την τιμή του γείτονα με την τιμή του α, δηλαδή αν -1≤3, που ισχύει. Άρα επιστρέφεται η τιμή -1 στον Min κόμβο και κλαδεύεται ο άλλος γείτονας του.



Γυρνάμε λοιπόν πίσω στον Max κόμβο και ελέγχουμε τον επόμενο Min κόμβο δηλαδή αυτό με τις τερματικές καταστάσεις -3 και -4. Εκεί η τιμή του β γίνεται -3. Ο αλγόριθμος θα συγκρίνει την τιμή του γείτονα με τον α. Δηλαδή ελέγχουμε αν -3≤3, που ισχύει. Άρα επιστρέφεται η τιμή -3 στον Min κόμβο και κλαδεύουμε τον αλλο γείτονα του.



Στον επάνω Max κόμβο λοιπόν επιστρέφεται η τιμή -1. Στον αμέσως επάνω Min κόμβο ο αλγόριθμος θα συγκρίνει την τιμή του γείτονα με την τιμη του α, δηλαδή αν -1≤3 που ισχύει. Άρα κλαδεύεται ο άλλος γειτονικός Max κόμβος.



Εν τέλει επιστρέφεται η τιμή 3 στην ρίζα μιας και έχουμε βρεί τις τιμές από όλους τους γείτονες. Συνολικά θα έχουμε κλαδέψει 12 κόμβους, όπως φαίνεται στο τελικό σχήμα.

- Πρόβλημα 4

α) Ο αλγόριθμος alpha-beta pruning δεν προτείνεται. Από την στιγμή που έχουμε μόνο Max κόμβους θα γίνεται πάντα η σύγκριση για το αν ενας κόμβος είναι μεγαλύτερος του β. Όμως εφόσον δεν υπάρχει Min κόμβος δεν θα ανανεωθεί ποτέ η τιμή του β, άρα θα είναι πάντα +∞. Επειδή κανένας πραγματικός αριθμός δεν πρόκειται να ξεπεράσει το +∞ δεν θα γίνει ποτέ κανένα κλάδεμα, άρα η χρήση του αλγορίθμου alpha-beta pruning δεν έχει ιδιαίτερη αξία.

β) Στην περίπτωση του expectimax δέντρου επίσης δεν προτείνεται κάποιος αλγόριθμος κλαδέματος. Εδώ πέρα ο λόγος είναι επειδή δεν υπάρχει η έννοια της βέλτιστης κίνησης για τον αντίπαλο του Max. Εφόσον ο Chance κόμβος δέχεται μια τιμή βασισμένη στην τύχη, δεν μπορούμε να ορίσουμε μια τιμή πχ για το β γιατί το περιεχόμενο των μη εξερευνημένων παιδιών μπορεί να φέρει τεράστιες αλλαγές στις τιμές των Chance κόμβων.

γ) Όπως εξηγήσαμε στο α), από την στιγμή που δεν υπάρχουν Min κόμβοι δεν θα αλλάξει ποτέ η τιμή του β άρα δεν θα γίνει κλάδεμα σε κανένα κόμβο του δέντρου. Οπότε οι αρνητικές ή μηδενικες τιμές δεν επηρεάζουν το max δέντρο.

δ) Για τον ίδιο λόγο που εξηγήθηκε στο β), δεν θα γίνει κλάδεμα σε κανέναν κόμβο. Δεν υπάρχει η έννοια της βέλτιστης κίνησης για τον αντίπαλο του Max, οπότε οι αρνητικές ή μηδενικες τιμές δεν θα επηρεάσουν το αποτέλεσμα.

ε) Δεν θα επηρεάσουν με κάποιον τρόπο το max δέντρο οι θετικές ή μηδενικές τιμες, οπότε ισχύει ό,τι είπαμε και στο γ).

στ) Δεν θα επηρεάσουν με κάποιον τρόπο το expectimax δέντρο οι θετικές ή μηδενικές τιμες, οπότε ισχύει ό,τι είπαμε και στο δ).

ζ) Το διάστημα [0,1] δεν θα φέρει κάποια αλλαγή στο max δέντρο, οπότε δεν θα γίνει κανένα κλάδεμα για τους ίδιους λόγους που αναφέραμε και στα προηγούμενα ερωτήματα για το max δέντρο.

η) Το διάστημα [0,1] δεν θα φέρει κάποια αλλαγή στο expectimax δέντρο, οπότε δεν θα γίνει κανένα κλάδεμα για τους ίδιους λόγους που αναφέραμε και στα προηγούμενα ερωτήματα για το expectimax δέντρο.