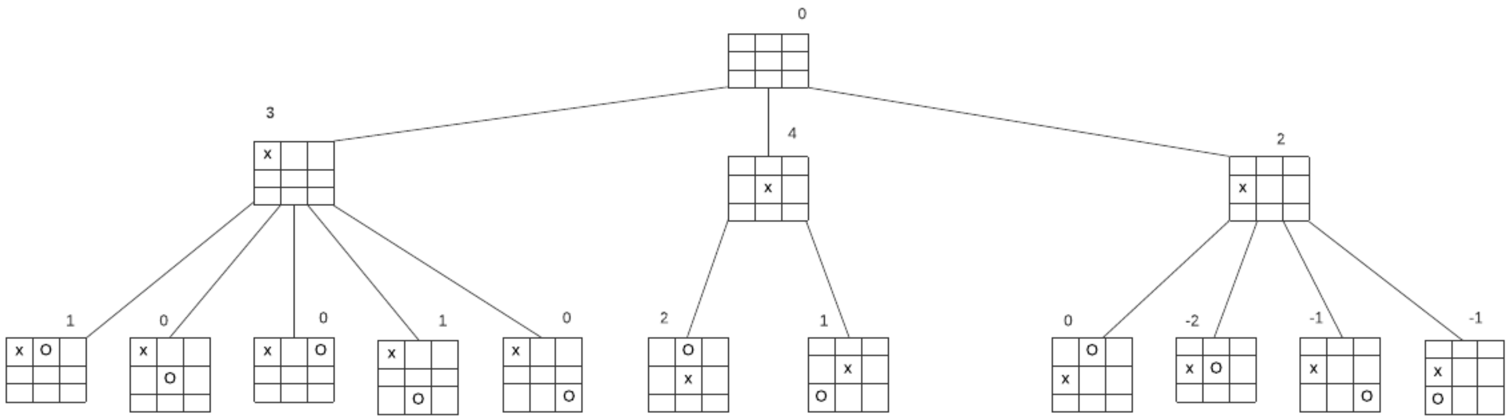
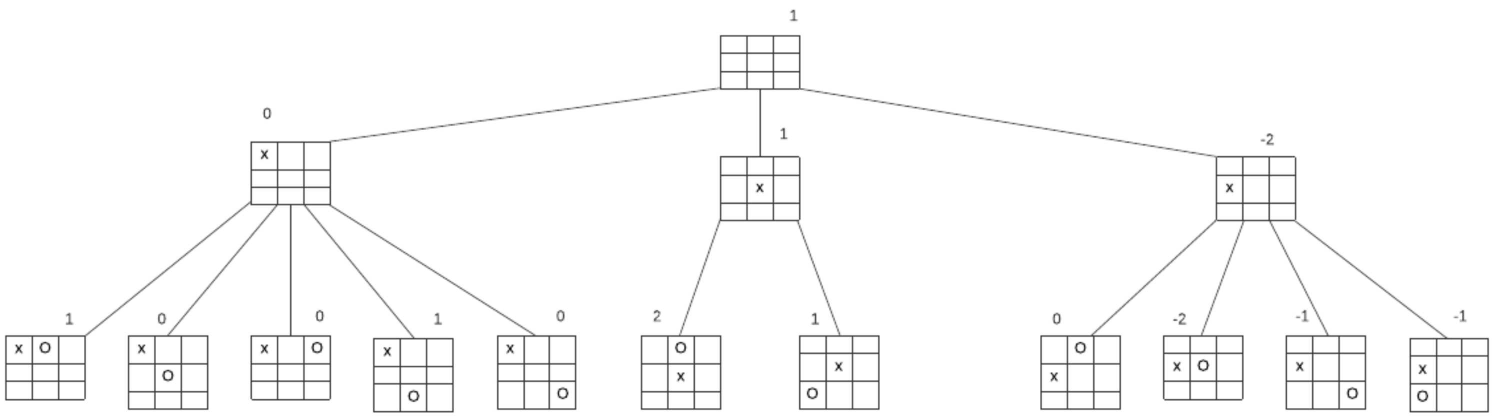
Εργασία 2
Χαράλαμπος Τσιτσιρίγγος
sdi1900198

Πρόβλημα 1

Άσκηση 2,3



Άσκηση 4



Άσκηση 5

Ο αλγόριθμος alpha-beta κόβει τους 3 τελευταία φύλλα στο πιο δεξιά node. Όταν ελέγχει την utility function του αριστερού παιδιού-φύλλου , βρίσκει πως είναι 0 (0 < 1) , δεν ελέγχει τις υπόλοιπες και αναθέτει την τιμή minimax 1 στη ρίζα.

Αν οι κόμβοι παράγονται με την αντίθετη σειρά θα κόψει τα τελευταία 4 φύλλα (από δεξιά στα αριστερά) από το πιο αριστερό παιδί. Βλέπει πως η utility function είναι 0(0<1) , οπότε δεν ελέγχει τις υπόλοιπες και αναθέτει την minimax 1 στη ρίζα.
Στην προκειμένη περίπτωση , η δεύτερη μέθοδος είναι καλύτερη καθώς παράγει 11 nodes και η πρώτη παράγει 12 nodes.

Εργασία 2
Χαράλαμπος Τσιτσιρίγγος
sdi1900198

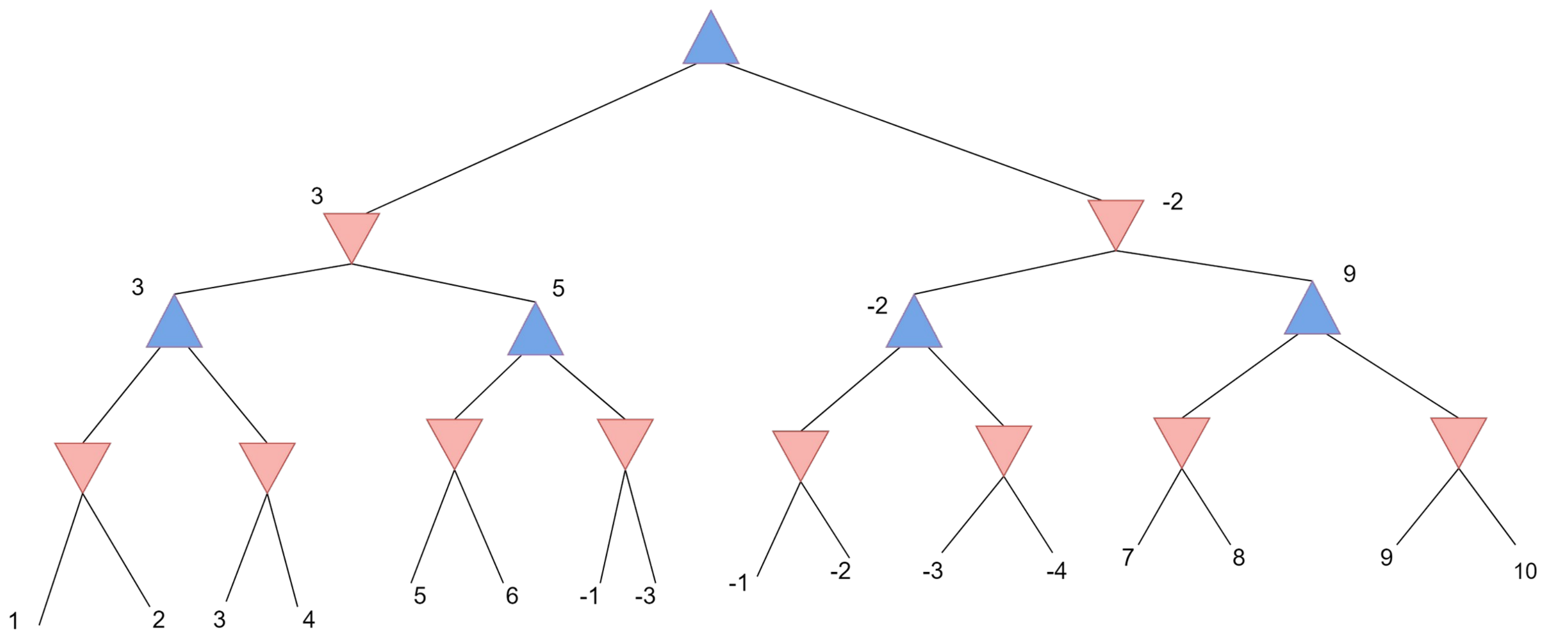
Πρόβλημα 2

Για να είναι βέλτιστος ο alpha-beta αλγόριθμος , πρέπει να παράγονται πρώτα οι κόμβοι που αποτελούν την καλύτερη λύση για τον κάθε παίκτη. Δηλαδή όταν παίζει ο παίκτης Χ , να παράγονται με φθίνουσα σειρά οι κόμβοι και όταν παίζει ο Ο να παράγονται με αύξουσα σειρά.

Για την αντίθετη περίπτωση παρατηρείται η μέγιστη παραγωγή κόμβων από τον αλγόριθμο

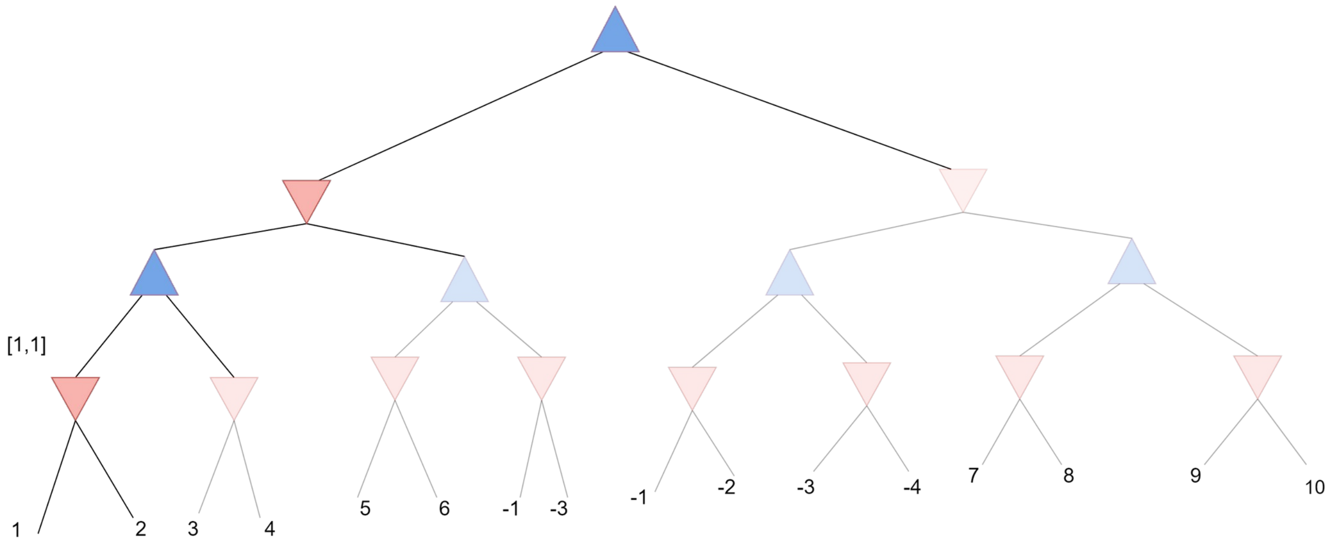
Πρόβλημα 3

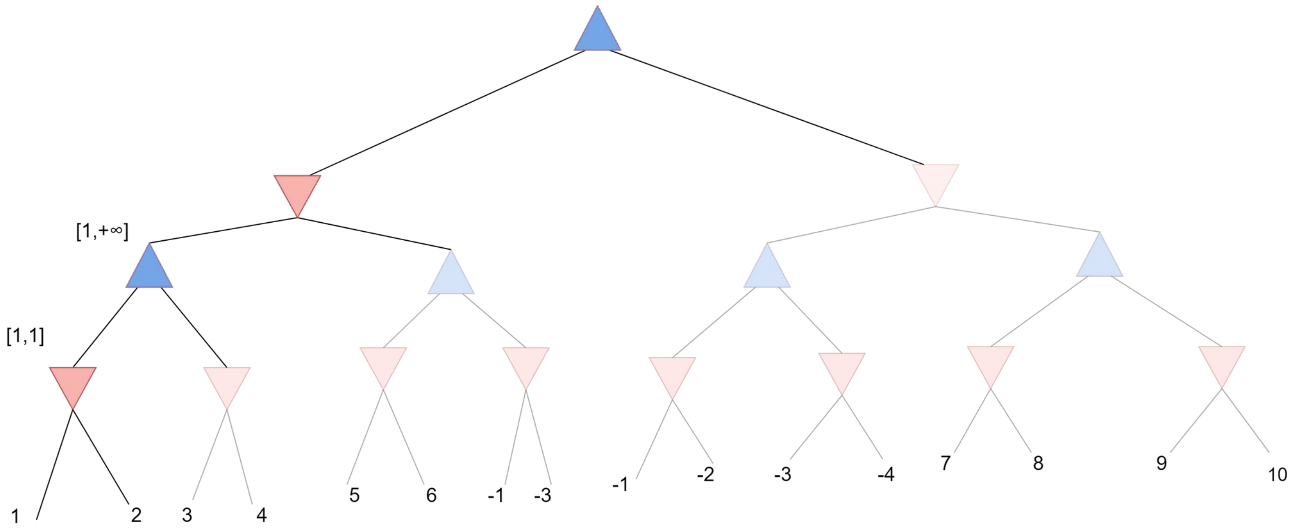
Στη ρίζα θα διαλέξει τον αριστερό κόμβο ,καθώς έχει το μεγαλύτερο minimax value.

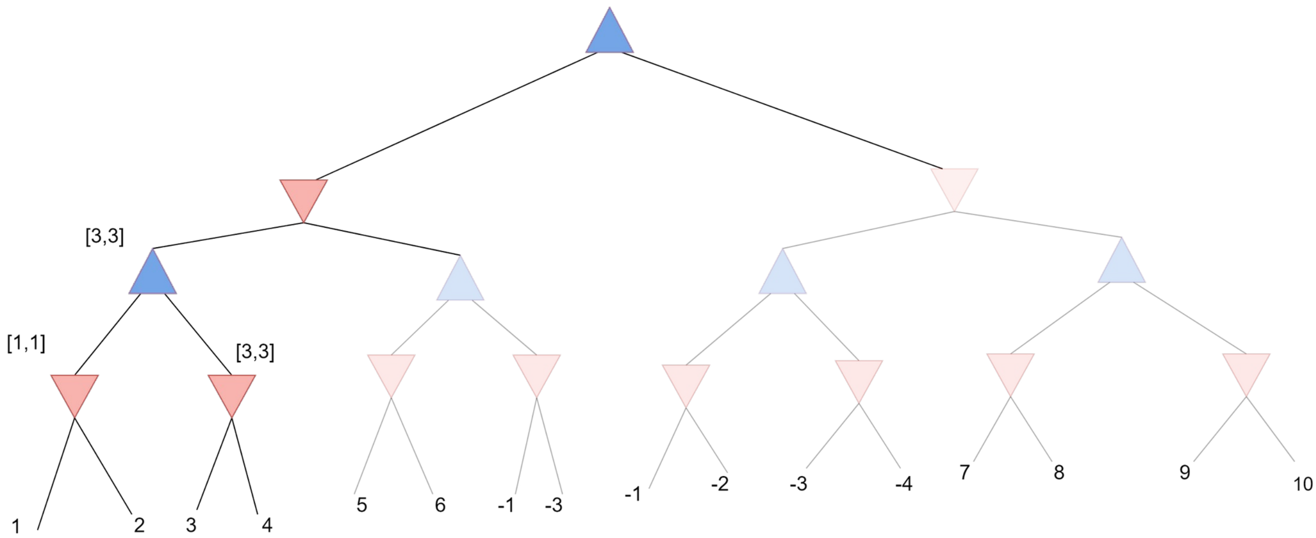


Alpha-Beta:

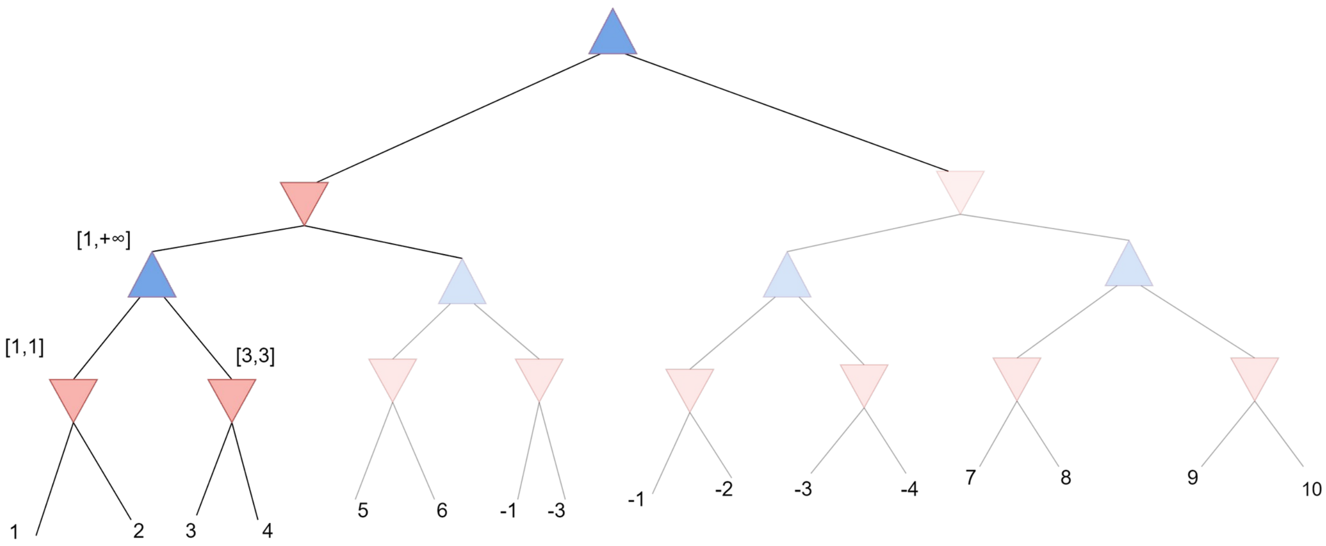
Εργασία 2
Χαράλαμπος Τσιτσιρίγγος
sdi1900198

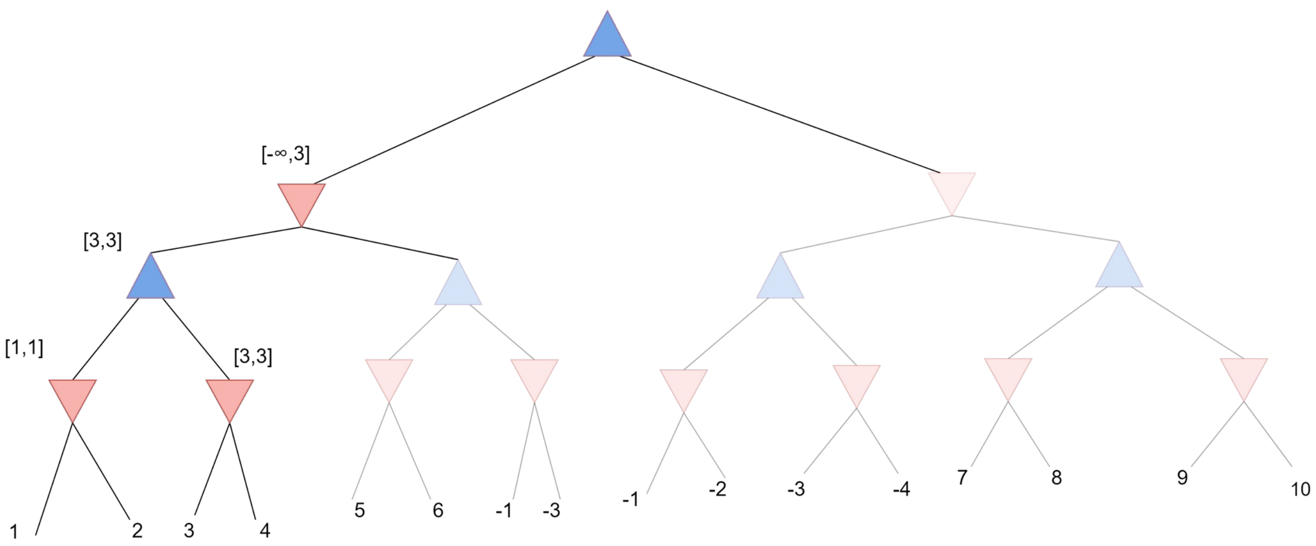


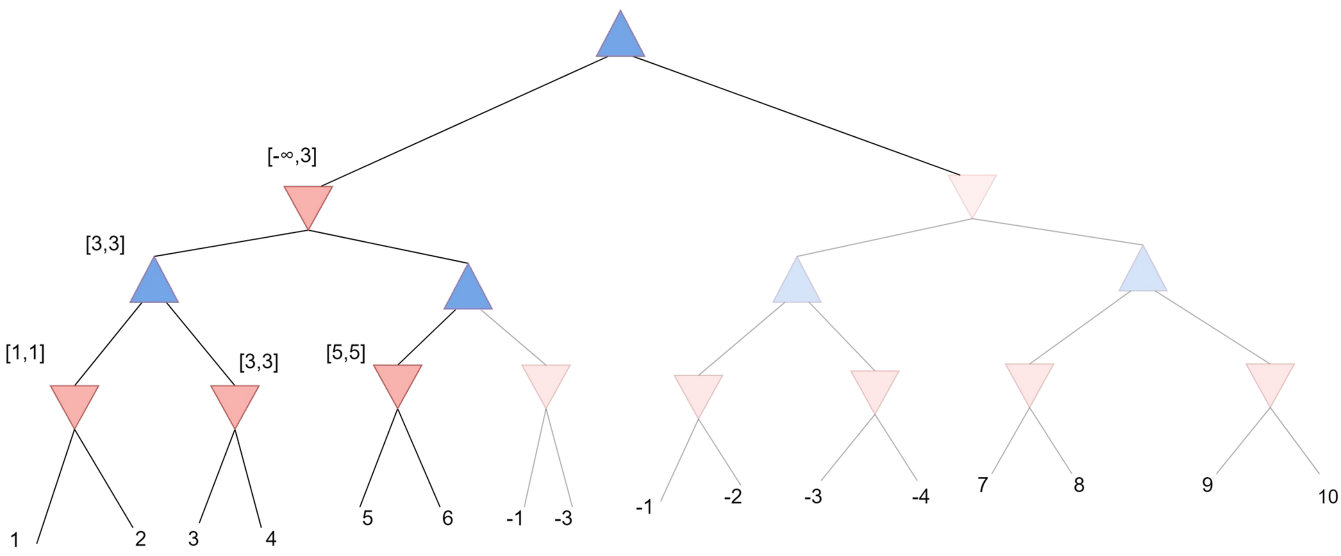




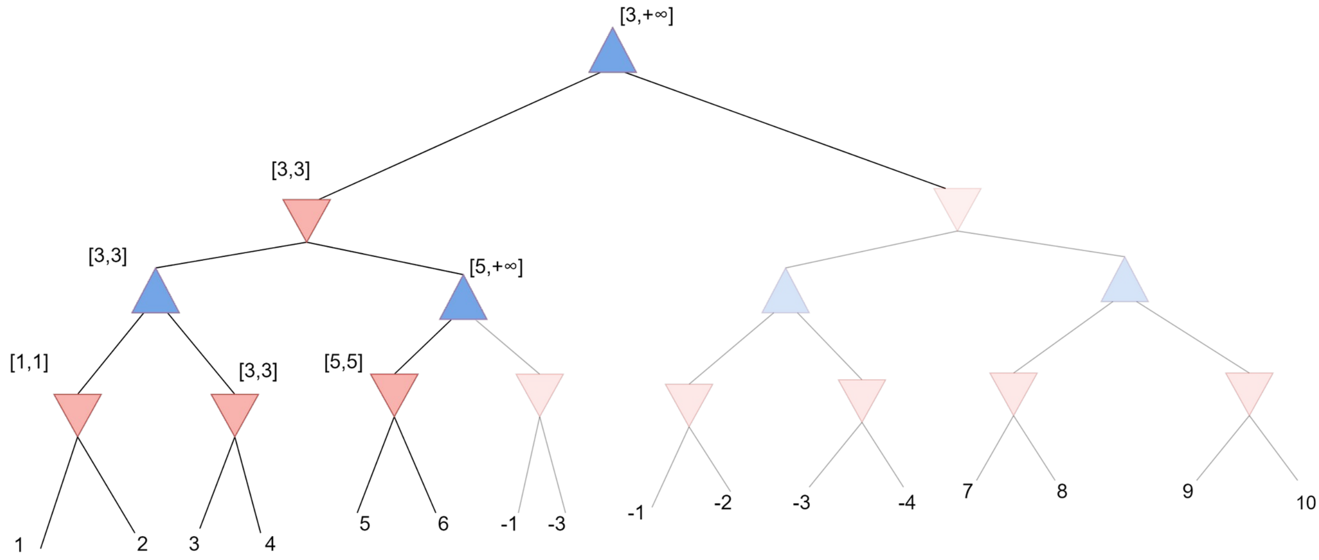
Εργασία 2
Χαράλαμπος Τσιτσιρίγγος
sdi1900198

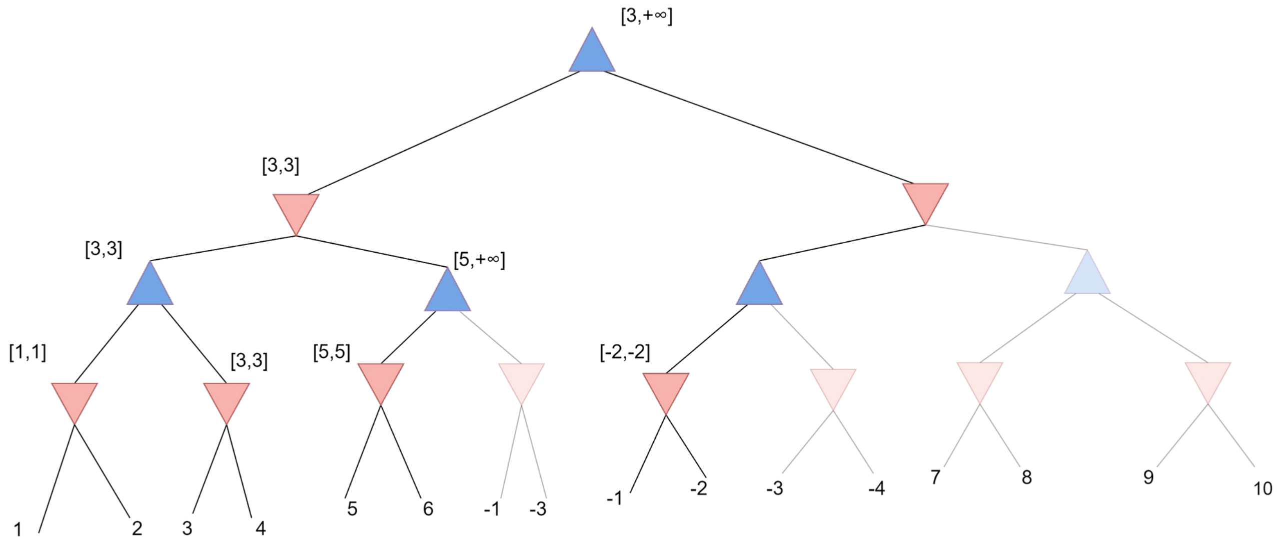


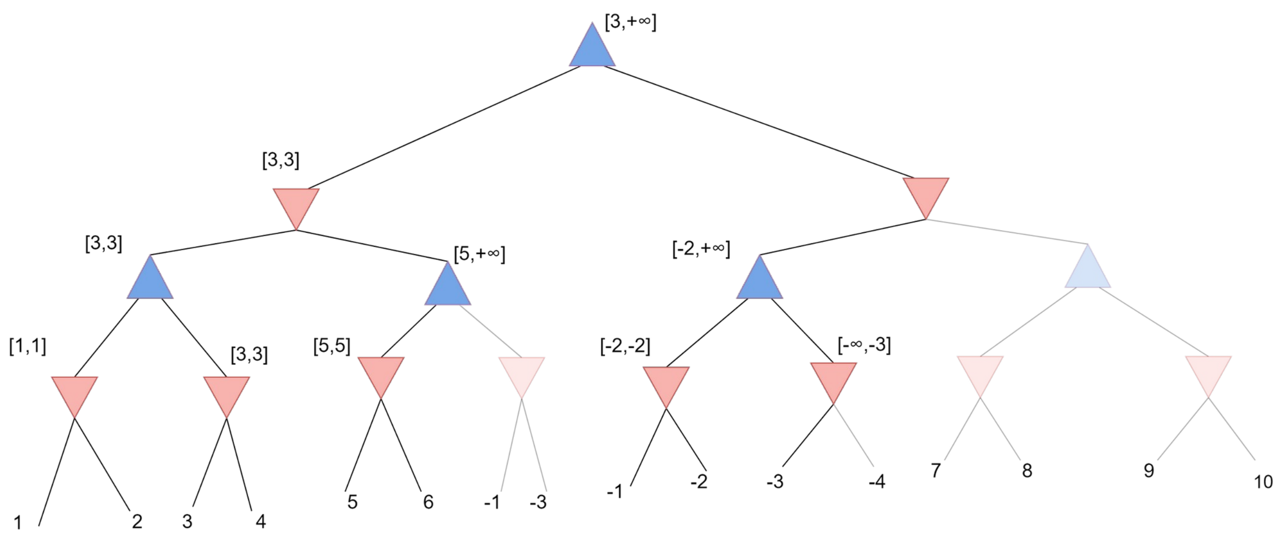




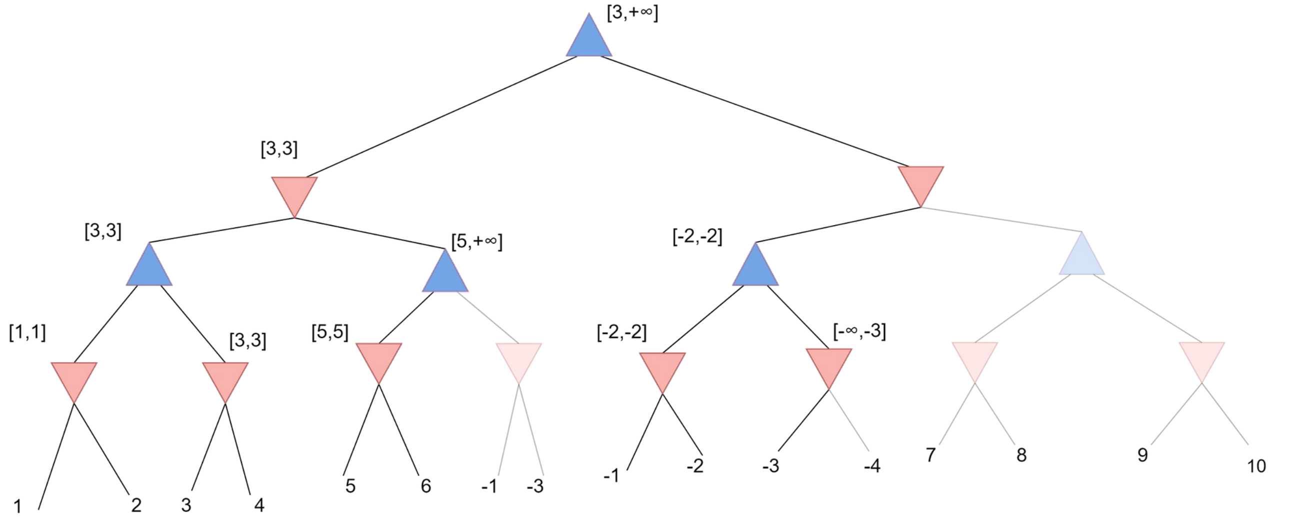
Εργασία 2
Χαράλαμπος Τσιτσιρίγγος
sdi1900198

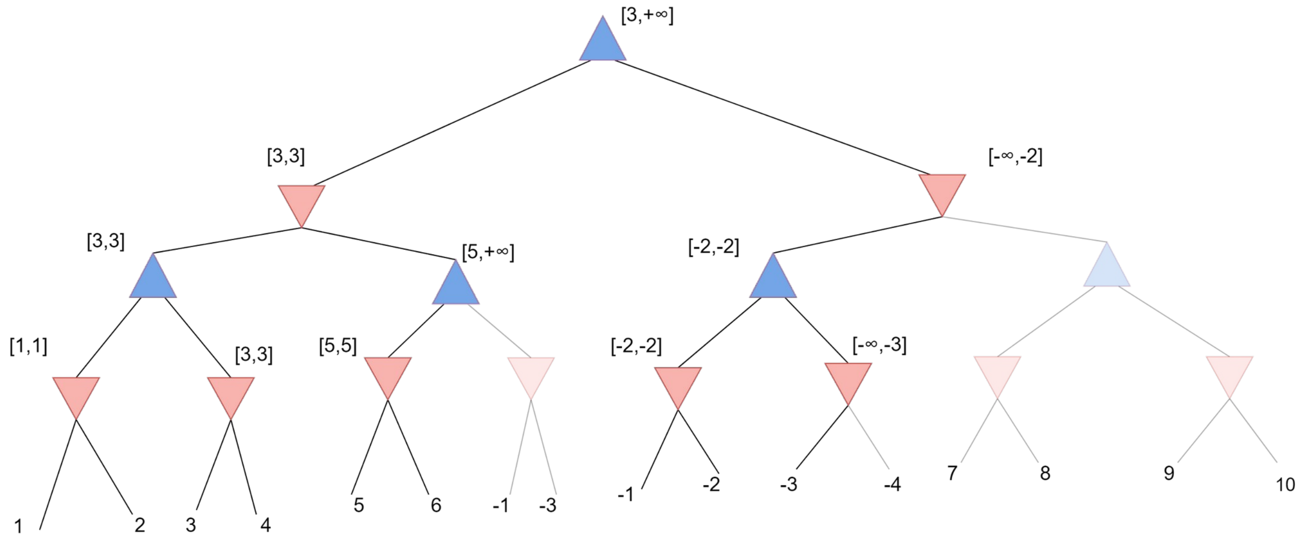


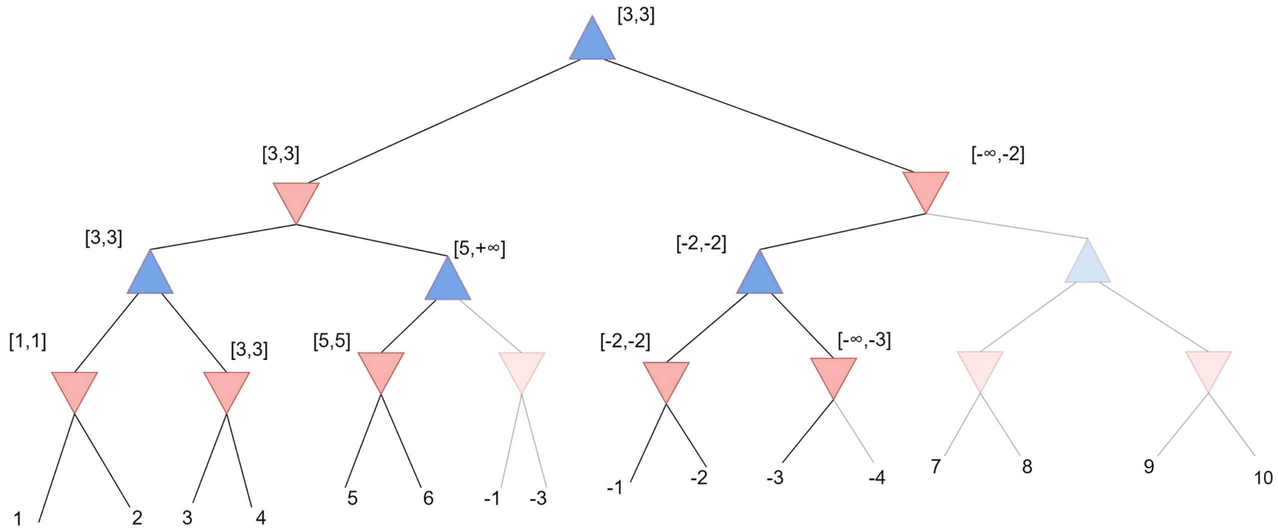




Εργασία 2
Χαράλαμπος Τσιτσιρίγγος
sdi1900198







Εργασία 2
Χαράλαμπος Τσιτσιρίγγος
sdi1900198

Πρόβλημα 4

α) Δεν υπάρχει κάποιος αλγόριθμος για να κλαδέψουμε ένα unordered max-tree καθώς δεν υπάρχει ένα upper bound στα values των παιδιών ενός node . Αν όμως παράγουμε τα nodes με σειρά από το βέλτιστο στο λιγότερο χρήσιμο , τότα εμφανίζεται ένα upper bound για τα παιδιά του κάθε node. Συγκεκριμένα θα είναι , με $c_0, c_1, \ldots, c_n$ παιδιά:

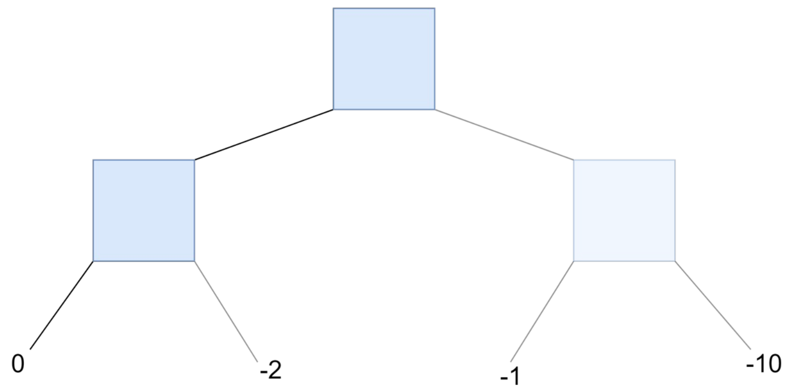
$$eval(c_0) >= eval(c_1), \ldots, >= eval(c_n)$$

Έτσι μπορούμε να κρατάμε ένα γενικό max με το οποίο θα ελέγχουμε το $eval(c_0)$. Αν είναι η eval είναι μικρότερη , τότε , δεν θα παράγουμε τα υπόλοιπα nodes

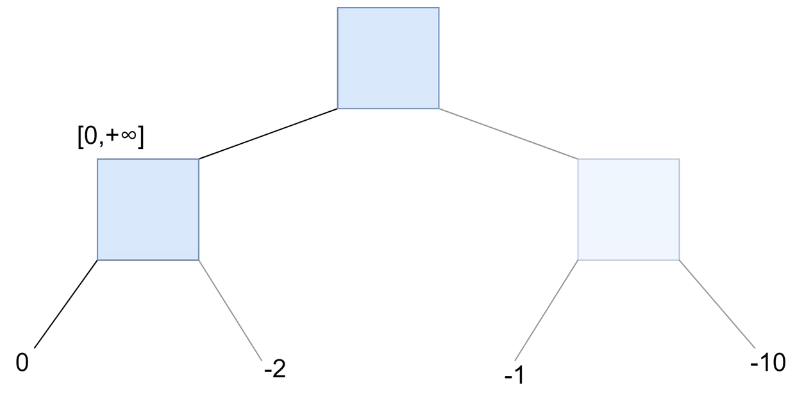
β) Δεν υπάρχει κάποιος αλγόριθμος να κλαδέψουμε ένα expectiamax tree καθώς από τον ορισμό , πρέπει να παραχθούν όλα τα nodes για να έχουμε τιμές.

γ) Ναι καθώς τώρα υπάρχει upper bound το 0.(αν υποθέσουμε ότι δεν έχουμε ισοπαλίες)
Πχ:

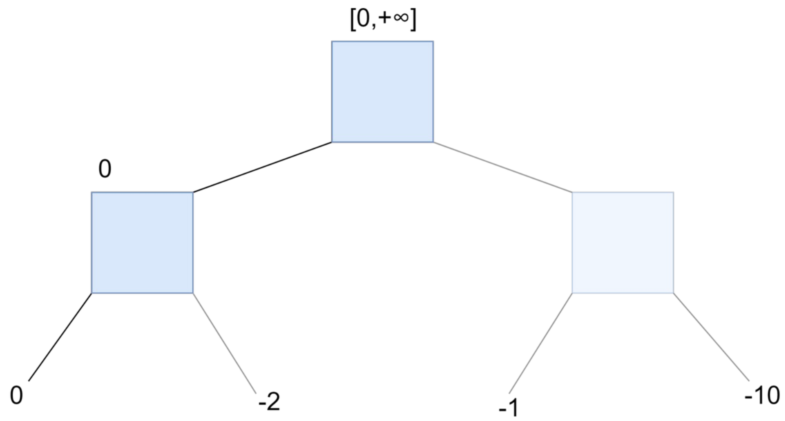
1) Πηγαίνοντας στα αριστερά βρίσκουμε το 0



2) Άρα ο κόμβος έχει τιμή σίγουρα [0,+∞] , άρα 0 , αφού έχουμε upper bound το 0. Έτσι δεν χρειάζεται να παράξουμε το node με τιμή -2.



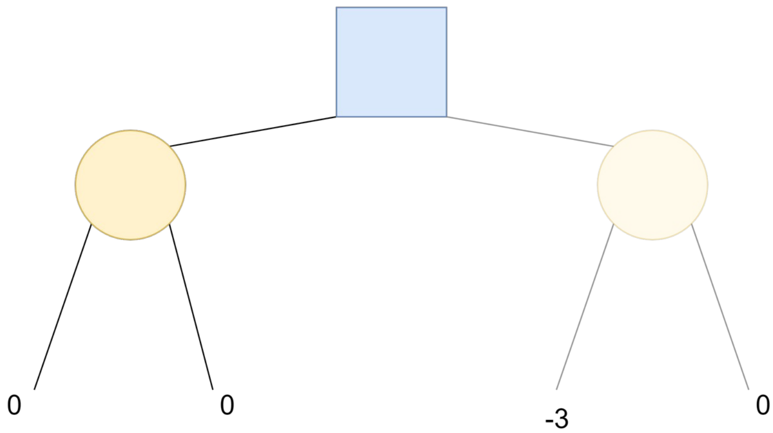
Εργασία 2
Χαράλαμπος Τσιτσιρίγγος
sdi1900198



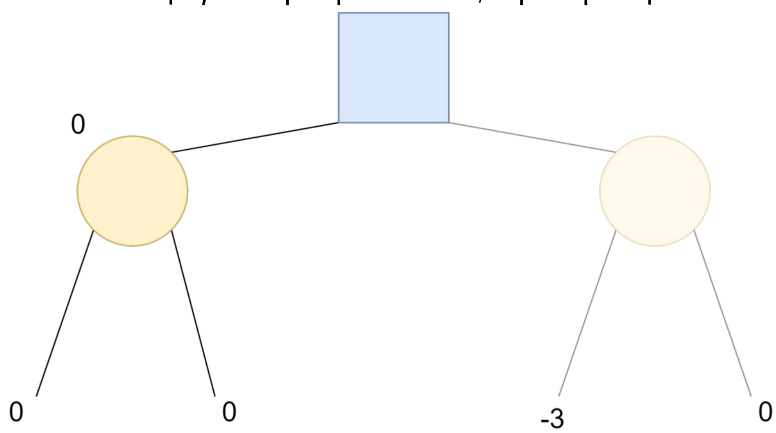
3)Ομοίως για το parent node. Έχει τιμή μεγαλύτερη ή ίση του 0 , άρα ίση του μηδενός

δ) Ναι.(αν υποθέσουμε ότι δεν έχουμε ισοπαλίες) Πχ:

1) Υπολογίζουμε τον αριστερά κόμβο.



2) Είναι ίσος με 0. Το 0 είναι upper bound
3) Το max ειναι μεγαλύτερο ή ίσο του 0 , δηλαδή ίσο με το 0 και άρα ίσο με την max τιμή . Οπότε δεν χρειάζεται να παράξουμε το δεξί παιδί.



ε) Όχι καθώς δεν υπάρχει upper bound , άρα πρέπει να παραχθούν όλοι οι κόμβοι

στ) Όχι καθώς δεν υπάρχει upper bound , άρα πρέπει να υπολογίσουμε όλες τις τιμές , άρα να παραχθούν όλοι οι κόμβοι

Εργασία 2
Χαράλαμπος Τσιτσιρίγγος
sdi1900198

ζ) Ναι. Κάθε φορά που σε κάποιο childNode συναντάμε το 1 (δηλαδή το μέγιστο) ,δεν χρειάζεται να παράξουμε και τα υπόλοιπα παιδιά , αφού η τιμή θα είναι σίγουρα 1.

η) Μπορούμε μόνο αν συναντήσουμε κάποιο chanceNode με τιμή 1, δηλαδή όλα τα παιδιά του είναι ίσα με 1. Άρα η τιμή του κόμβου max θα είναι μεγαλύτερη ή ίση του 1 , άρα 1. Αν δεν έχουμε ισοπαλίες , τότε ο αλγόριθμος θα σταματούσε εκεί . Αν μπορεί να έχουμε , τότε θα έπρεπε να παράγουμε τα childNodes του επόμενου chanceNode μέχρι να βρούμε 0 . Όταν βρούμε 0 , είναι σίγουρο πως η τιμή του chanceNode θα είναι λιγότερο από 1 , άρα ο αλγόριθμος προχωράει.