ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
Εθνικόν και Καποδιστριακόν
Πανεπιστήμιον Αθηνών
ΙΔΡΥΘΕΝ ΤΟ 1837

Τεχνητή Νοημοσύνη Ι

Βασιλική Χριστοφιλοπούλου
1115202000216

Νοέμβριος 2024

Χειμερινό Εξάμηνο 2024-2025

Project 2

Table of Contents

1 Πρόβλημα 1

Απο την θεωρία ξέρουμε ότι η τιμή minimax ενός κόμβου είναι η καλύτερη τιμή χρησιμότητας που μπορεί να επιτευχθεί από τον παίκτη στον κόμβο αυτό, με την προϋπόθεση ότι και οι δύο παίκτες παίζουν βέλτιστα απο εκείνο το σχημείο μέχρι και το τέλος του παιχνιδιού.(σελίδα 13)

Αν ο min παίζει βέλτιστα, η χρησιμότητα για τον max είναι η μέγιστη δυνατή χρησιμότητα που μπορεί να επιτευχθεί όταν ο min παίζει την καλύτερη κίνηση για αυτόν. Αντίθετα, αν ο min παίζει μη βέλτιστα, τότε η χρησιμότητα για τον max θα είναι τουλάχιστον όσο η χρησιμότητα που προκύπτει από την καλύτερη κίνηση του min, αλλά μπορεί να είναι και μεγαλύτερη.

Συνεπώς, ο max θα ωφελείται παραπάνω, καθώς σε κάθε επιλογή θα είναι αναμέσα

- στις τιμές που θα υπήρχαν αν ο min έπαιζε βέλτιστα και
- σε μεγαλύτερες τιμές αφου απο τα δεδομένα ο min δεν παίζει βέλτιστα.

Αυτό έχει ως αποτέλεσμα, ότι ο max να μπορεί να εκμεταλλευτεί την αδυναμία του min να επιλέγει σωστά και να αυξήσει τη χρησιμότητά του. Σημαντικό είναι ότι η στρατηγική του max δεν χρειάζεται να είναι ιδανική για να επωφεληθεί από την κακή επιλογή του min. Ακόμη και μια suboptimal στρατηγική του max μπορεί να αποδώσει καλύτερα αν ο min κάνει κακές επιλογές.

Παρακάτω δίνεται ένα δέντρο παιχνιδιού στο οποίο ο max μπορεί να τα καταφέρει ακόμα καλύτερα χρησιμοποιώντας μια μη βέλτιστη (suboptimal) στρατηγική εναντίον ενός μη βέλτιστου min:

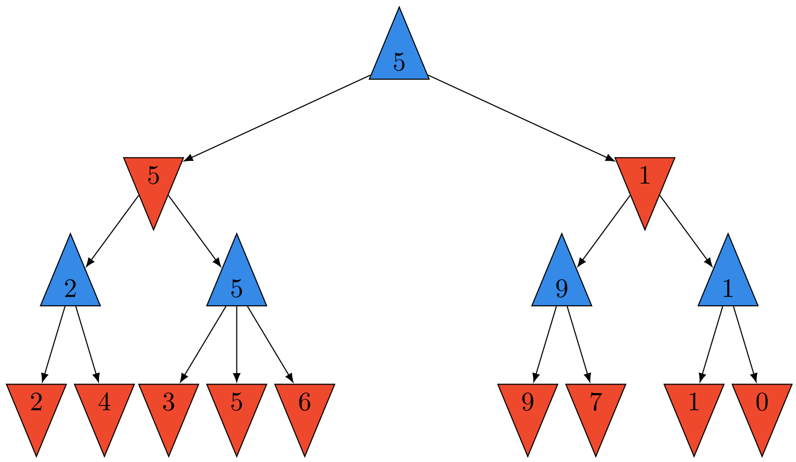

Figure 1: Μη βέλτιστο δέντρο

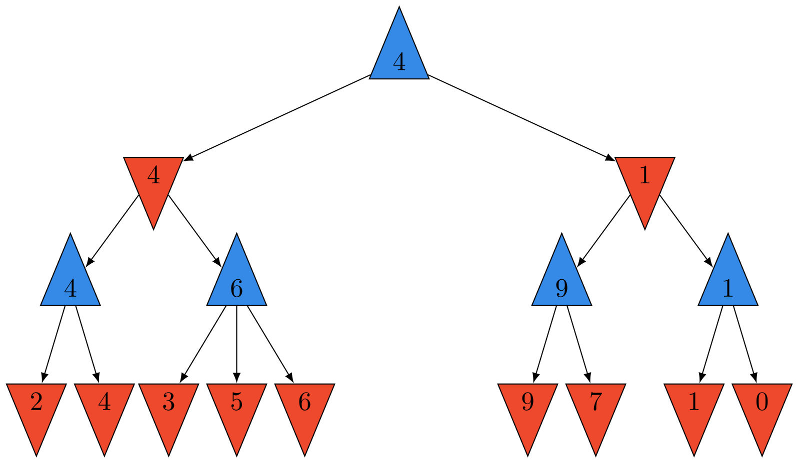

Figure 2: Βέλτιστο δέντρο

Από το παραπάνω δέντρο παιχνιδιού, βλέπουμε ότι:

- Όταν ο min παίζει βέλτιστα, η χρησιμότητα για τον max είναι 4.
- Όταν ο min παίζει μη βέλτιστα αλλά και ο max, η χρησιμότητα για τον max αυξάνεται σε 5.

Συνολικά, οι μη βέλτιστες στρατηγικές μπορούν να οδηγήσουν σε καλύτερα αποτελέσματα για τον MAX εάν ο MIN δεν επιλέγει βέλτιστες κινήσεις.

2 Πρόβλημα 2

Θεωρούμε το παρακάτω δέντρο παιχνιδιού:

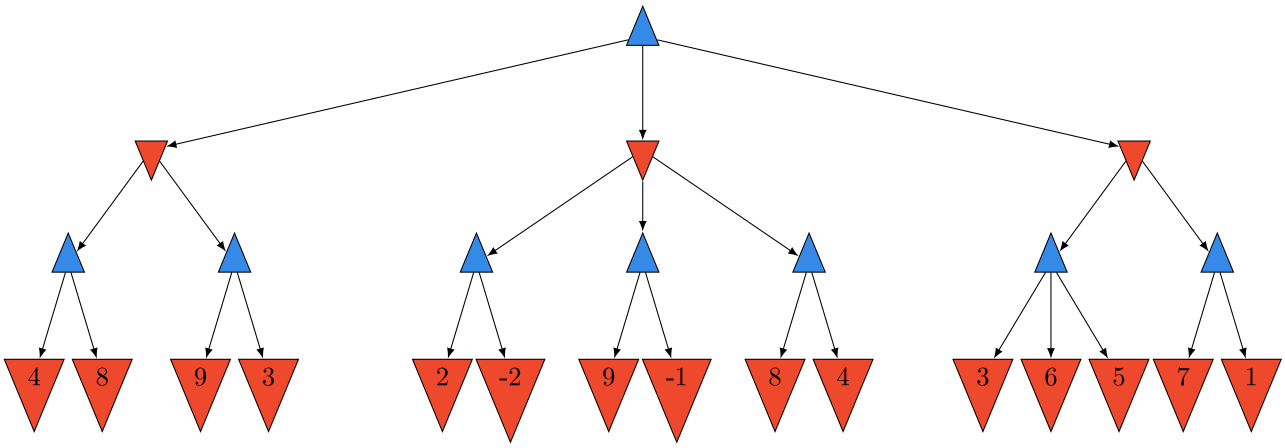

Figure 3: Δεδομένο δέντρο

2.1 Αλγόριθμος Minmax

Εφαρμόζουμε τον αλγόριθμο minmax, για κάθε επίπεδο μαζί για εξοικονόμιση επαναλήψεων και έχουμε:

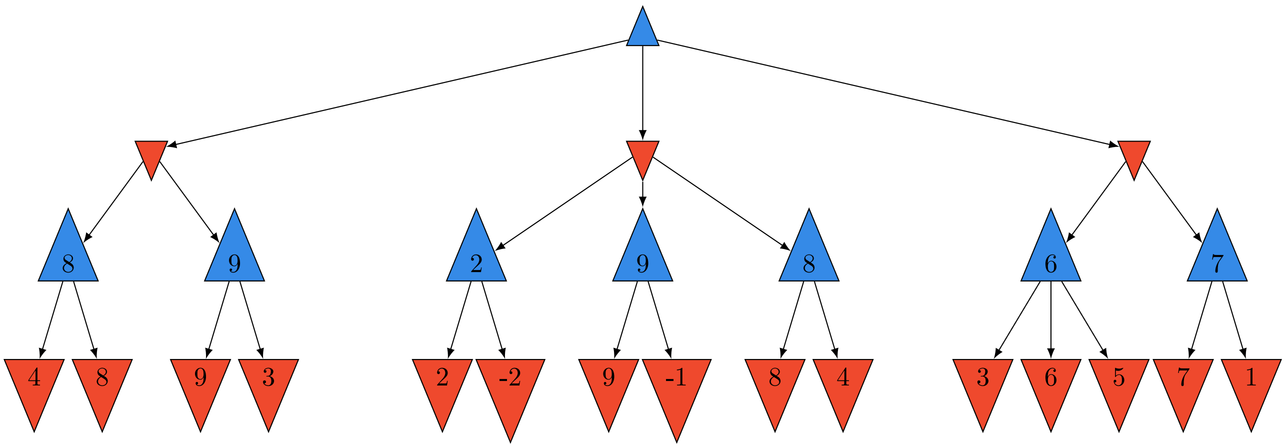

Figure 4: Πρώτη επανάληψη

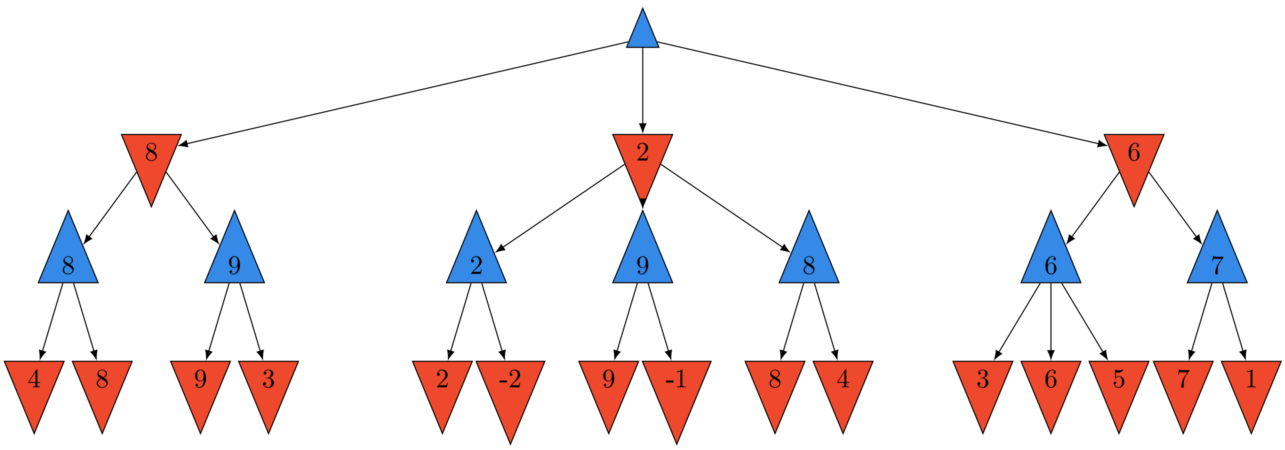

Figure 5: Δεύτερη επανάληψη

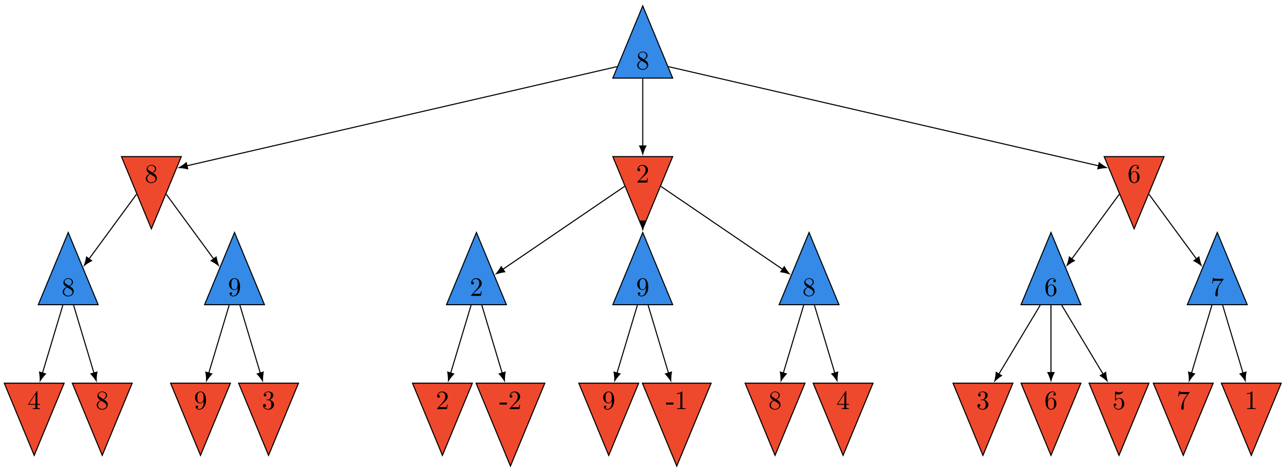

Figure 6: Τελευταία επανάληψη

Συνεπώς η minimax απόφαση στη ρίζα του δέντρου είναι 8.

2.2 Αλγόριθμος Alpha Beta Search

Εφαρμόζοντας τον αλγόρθμος Alpha Beta Search προκύπτουν τα ακόλουθα δέντρα:

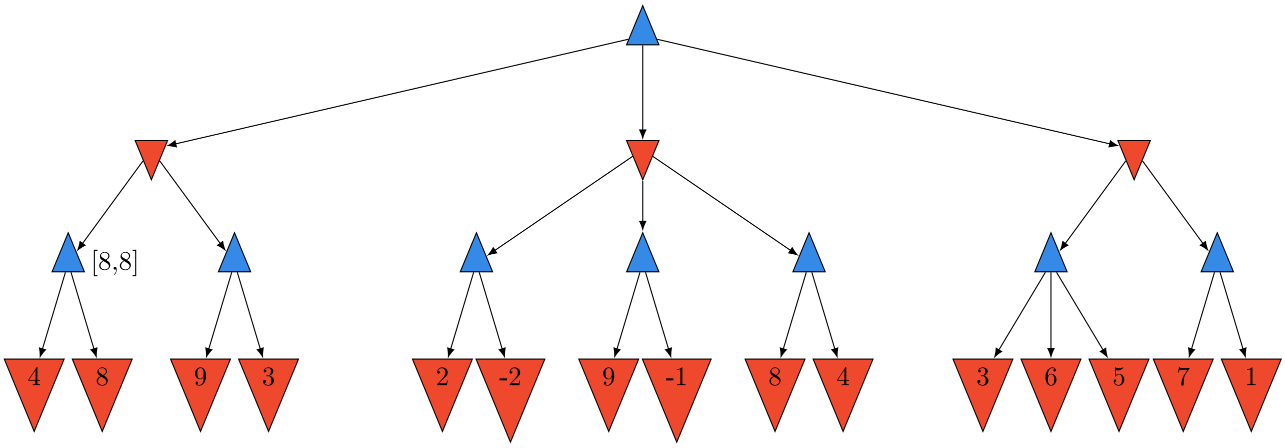

Figure 7: Πρώτη επανάληψη

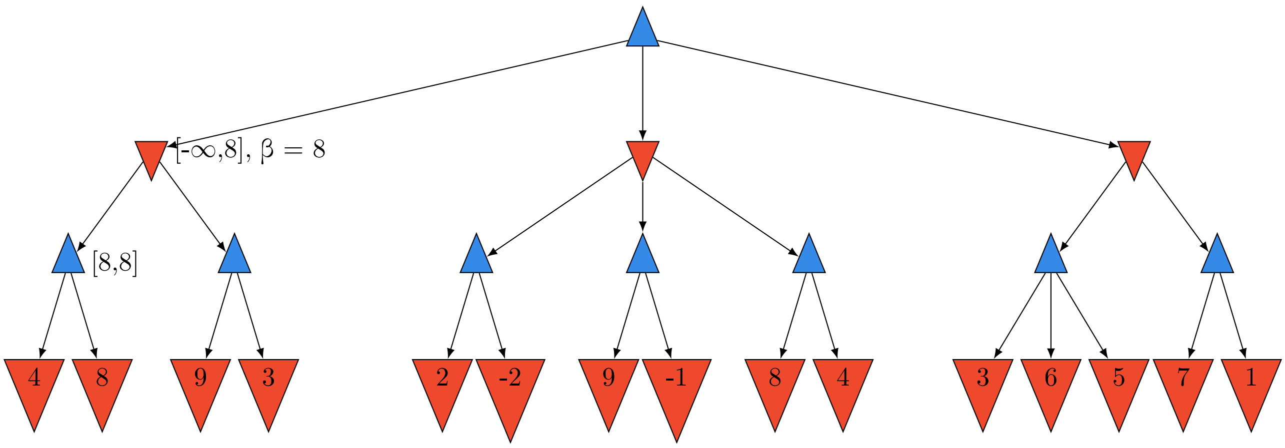

Figure 8: Δεύτερη επανάληψη

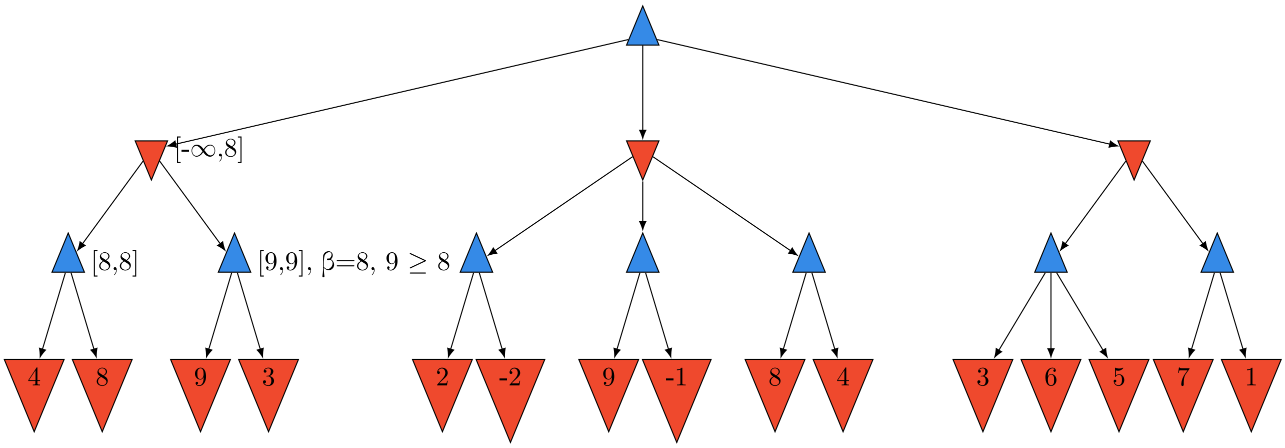

Figure 9: Τρίτη επανάληψη

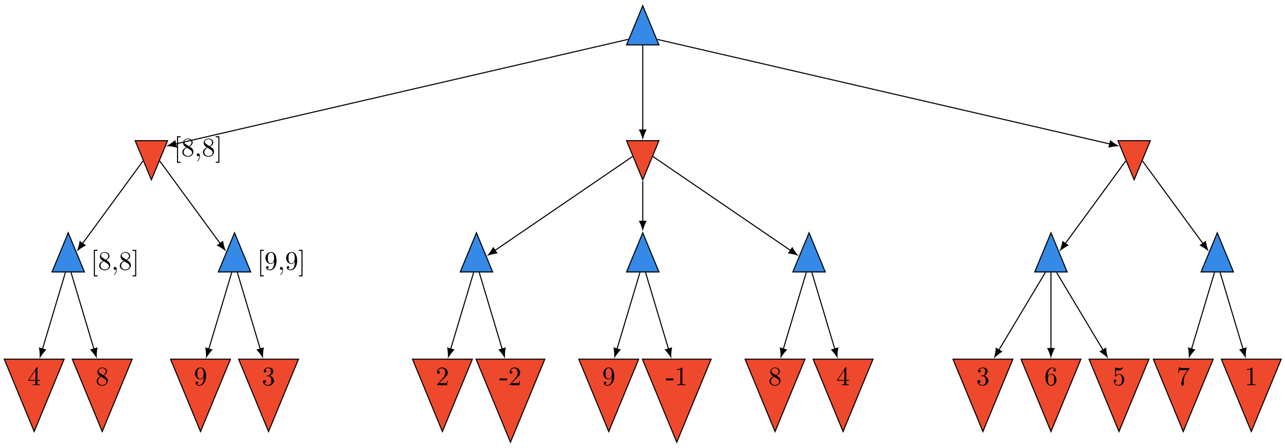

Figure 10: Τέταρτη επανάληψη

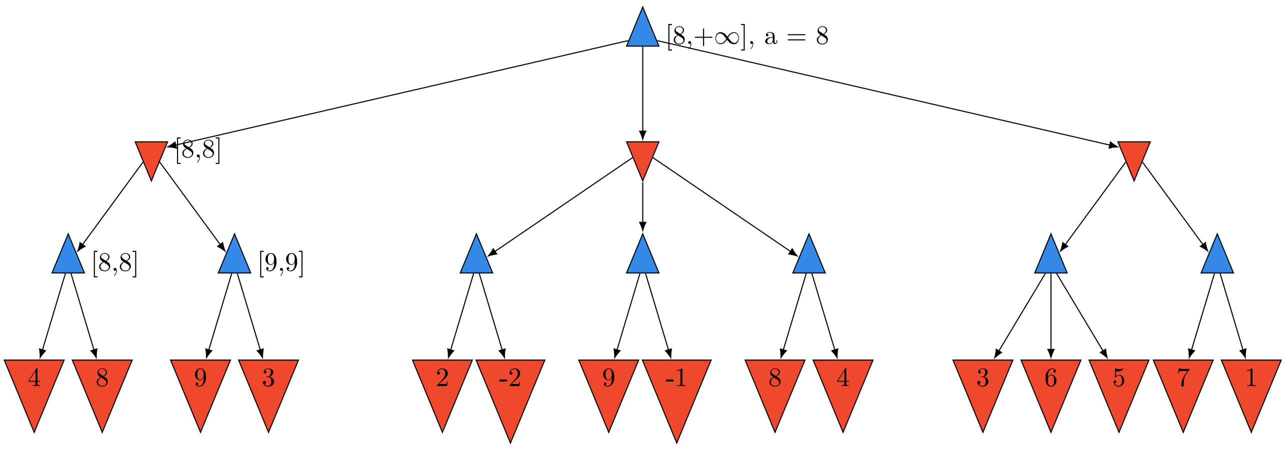

Figure 11: Πέμπτη επανάληψη

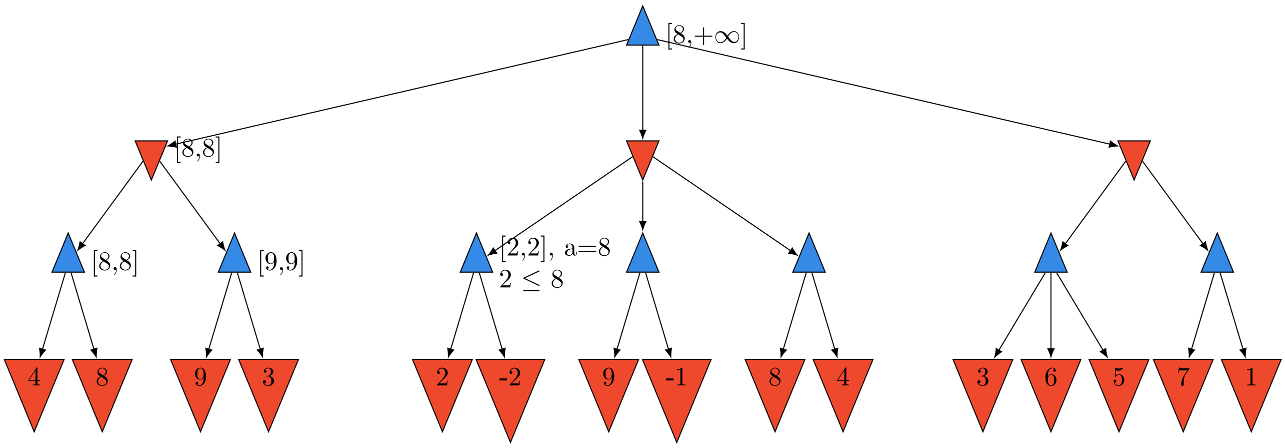

Figure 12: Έκτη επανάληψη

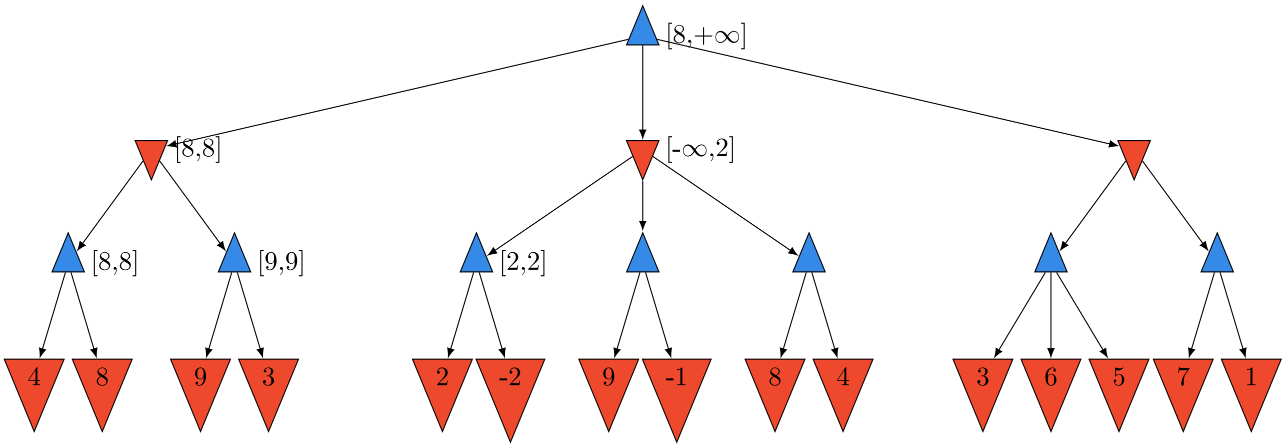

Figure 13: Έβδρομη επανάληψη

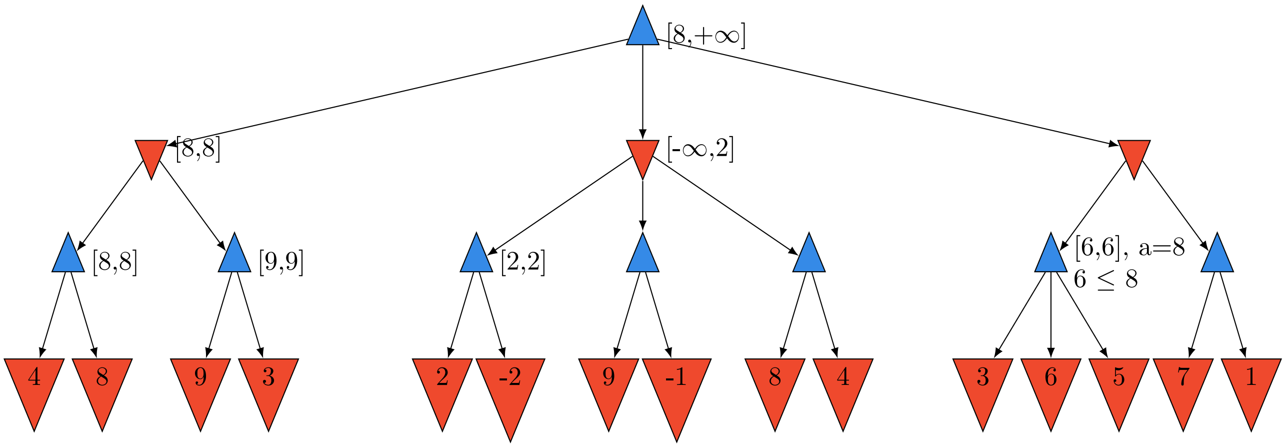

Figure 14: Όγδοη επανάληψη

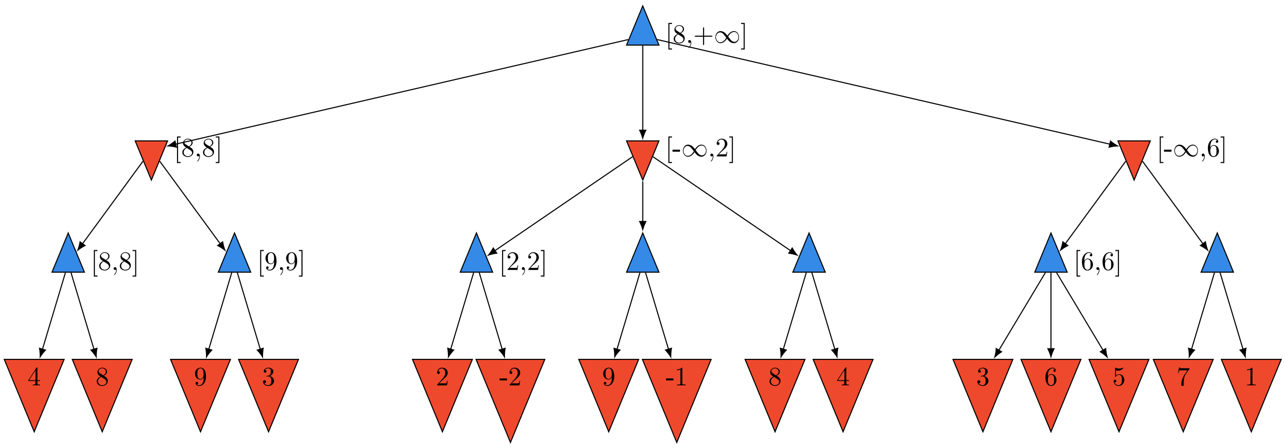

Figure 15: Ένατη επανάληψη

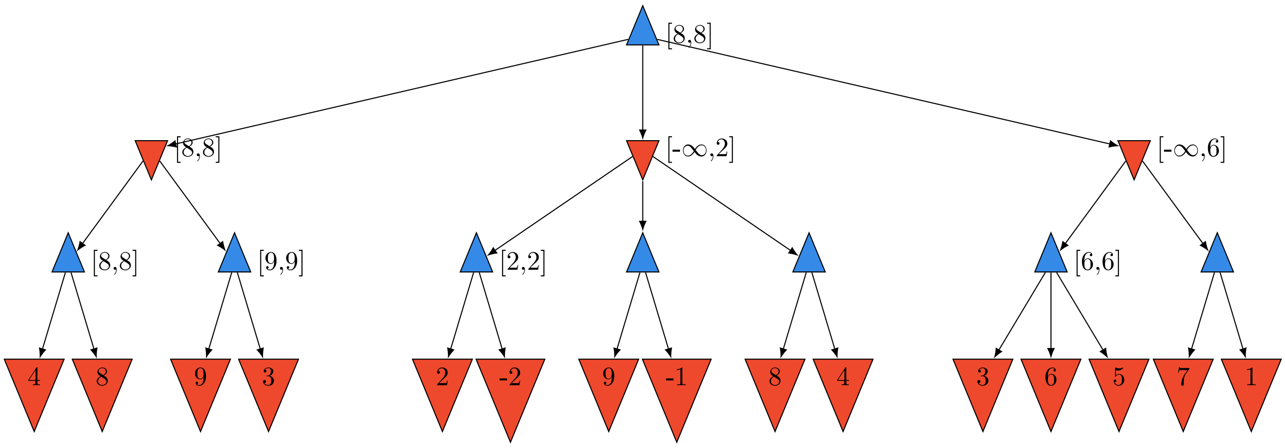

Figure 16: Δέκατη επανάληψη

3 Πρόβλημα 3

Απο τα δεδομένα της εκφώνησης έχουμε το παρακάτω δέντρο:

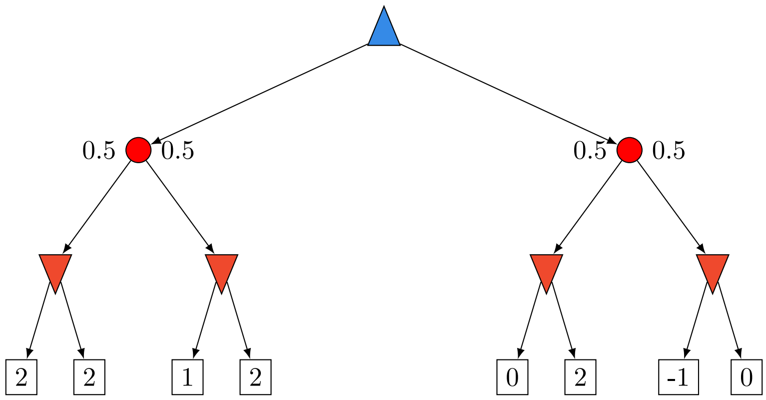

Figure 17: Δεδομένο δέντρο

3.1 Ερώτημα α

Για τον υπολογιμό των εσωτερικών κόμβων, ισχύουν τα ακόλουθα:

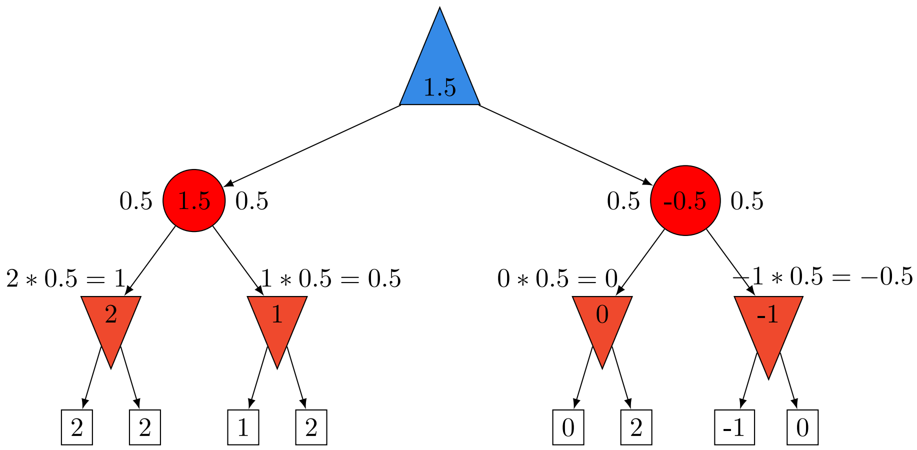

Figure 18: Υπολογισμός κόμβων

Συνεπώς η καλύτερη διαδρομή είναι:

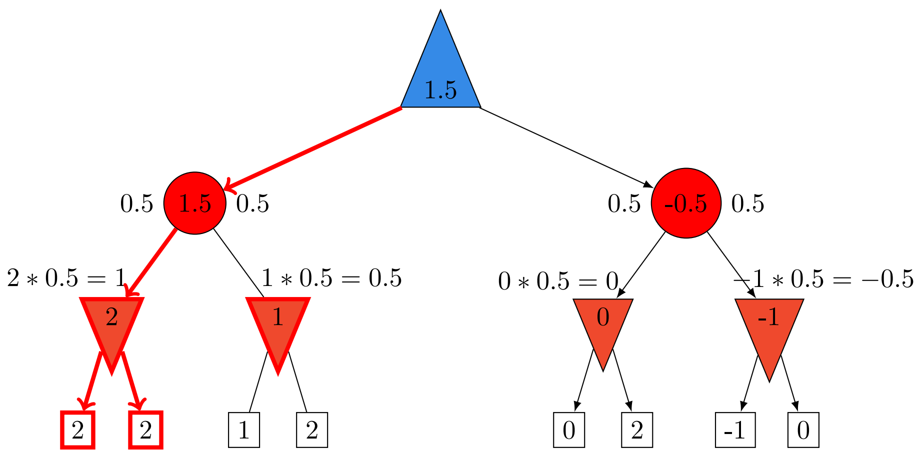

Figure 19: Καλύτερη διαδρομή

3.2 Ερώτημα β

Λύνοντας το ερώτημα α, παρατηρούμε ότι ο ο δεξιότερος κόμβος τύχης εξαρτάται μόνο απο τις τιμές του έβδομου και του όγδοου φύλλου, αφού το πέμπτο και έκτο φύλλο δίνουν ως min μηδένν. Εφόσον οι τιμές για τα φύλλα είναι στο διάστημα $[-\infty, +\infty]$, τότε χωρίς να υπολογίσουμε ακριβως, το έβδομο και όγδοο φύλλο θα έδιναν μια min τιμή, έστω x όπου $x \in [-\infty, +\infty]$. Δηλαδή η τιμή του κόμβου τύχης θα υπολογιζόταν απο την παράσταση $0 * 0.5 + x * 0.5$ Συνεπώς, αν γνωρίζουμε τις τιμές των πρώτων έξι φύλλων, δεν μπορούμε να προσδιορίσουμε την καλύτερη κίνηση για τη ρίζα, και άρα χρειαζόμαστε τουλάχιστον ένα ακόμα φύλλο.

Έστω ότι γνωρίζουμε και το έβδομο φύλλο, τότε έχουμε τις εξής περιπτώσεις:

- **Η τιμή του έβδομου φύλλου $*0.5 \leq$ της τιμής του αριστερότερου κόμβου τύχης**
  Για το δεδομένο παράδειγμα όπου ψάχνουμε το min, και η τιμή του έβδομου κόμβου είναι -1 και του όγδοου ανήκει στο διάστημα $[-\infty, +\infty]$, μπορούμε να σταματήσουμε το ψάξιμο, καθώς ο αριστερότερος κόμβος τύχης έχει τιμή 1.5 και ισχύει ότι $-1 * 0.5 \leq 1.5$. Συνεπώς, ακολουθώντας όμοια λογική με αυτή του Alpha Beta Search σταματάμε το κλάδεμα νωρίτερα καθώς το αριστερό υποδέντρο θα κυριαρχήσει, και μπορούμε να κάνουμε pruning, αγνοώντας τα φύλλα του δεξιού κόμβου.

- **Η τιμή του έβδομου φύλλου $*0.5 \geq$ της τιμής του αριστερότερου κόμβου τύχης**
  Στην περίπτωση που το έβδομο φύλλο έχει τιμή μεγαλύτερη απο το 3 για το συγκεκριμένο παράδειγμα, τότε υποχρεωτικά πρέπει να ελέγξουμε και το όγδοο φύλλο καθώς υπάρχει πιθανότητα το όγδοο να έχει τιμή μικρότερη απο το έβδομο και εμείς αναζητούμε την μικρότερη τιμή των δύο φύλλων.

3.3 Ερώτημα γ

Δεδομένου ότι οι τιμές για τα φύλλα ανήκουν στο διάστημα [-2,2], τότε το δεξί παιδί του αριστερότερου κόμβού τύχης παίρνει τιμές στο διάστημα [-2 * 0.5, 2 * 0.5] = [-1,1]. Συνεπώς, ο κόμβος τύχης λαμβάνει τιμές στο διάστημα [-1 + 1, 1 + 1] = [0,2]

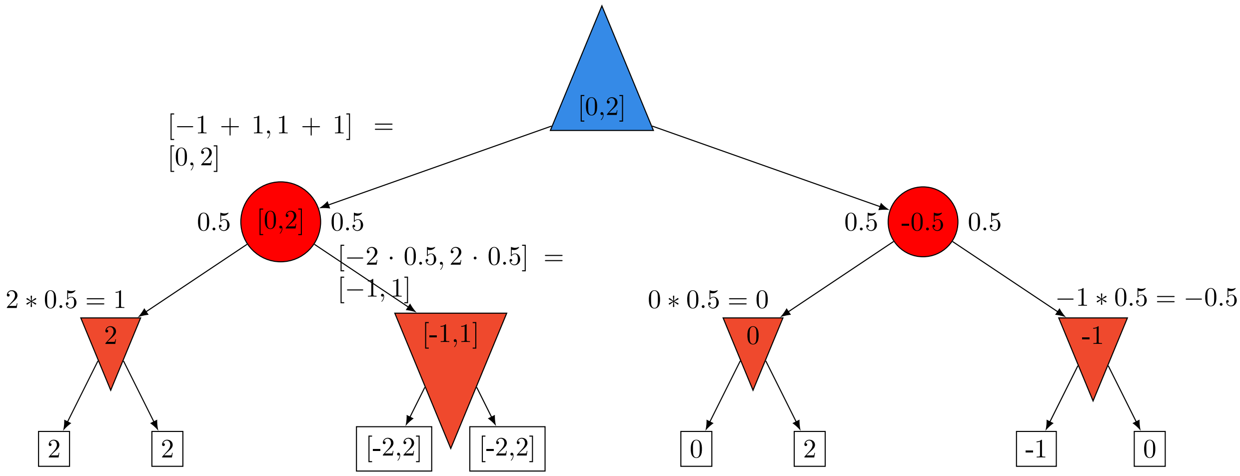

Figure 20: Ερώτημα γ

3.4 Ερώτημα δ

Εφόσον, γνωρίζουμε ότι οι τιμές των φύλλων είναι στο διάστημα [2,2], αν τρέξουμε τον αλγόριθμο άλφα-βήτα για το παραπάνω δένδρο έχουμε ότι θα ελεγχθεί ολό το αριστερο υποδέντρο και απο τα δεξία δεν θα ελεγχθούν:

- Το έκτο φύλλο καθώς κατα τον υπολογισμό του πέμπτου έχουμε ότι η τιμή του κόμβου τύχης με τα μέχρι τότε δεδομένα ειναι 0 * 0.5 + [-2 * 0.5, 2 * 0.5] = [-1,1] όπου η τιμή 1.5 είναι μεγαλύτερη από το διάστημα αυτό και αφού αναζητούμε max τιμή, ο αλγοριθμός δεν θα συνεχίσει τηνα αναζήτηση στο επόμενο φύλλο.

- Το όγδοο φύλλο καθώς κατα τον υπολογισμό του έβδομου έχουμε ότι η τιμή του κόμβου τύχης με τα μέχρι τότε δεδομένα ειναι 0 * 0.5 + -1 * 0.5 = -0,5 $\leq$ 1.5 που είναι η τιμή του άλλου κόμβου τύχης, οπότε και ο αλγόριθμος σταματά καθως βρήκε την βέλτιστη λύση.

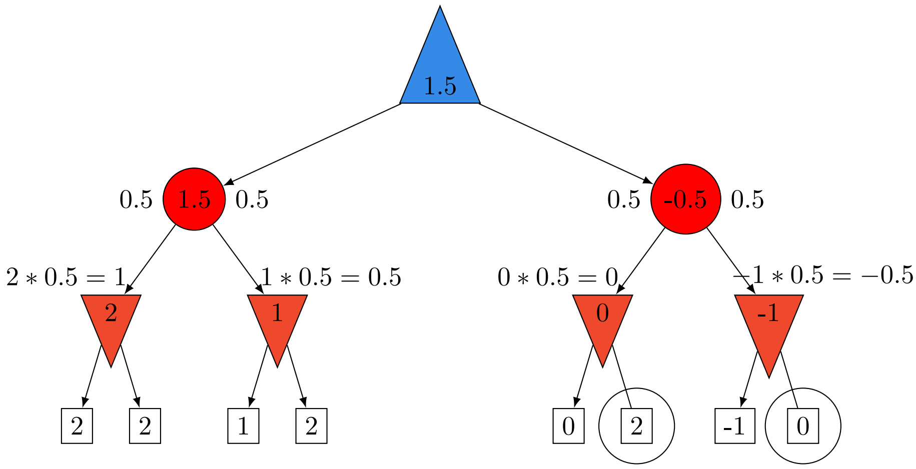

Figure 21: Alpha Beta Tree

4 Πρόβλημα 4

4.1 Ερώτημα 1

Για να κατασκευάσουμε το δέντρο του παιχνιδιού Nim με δύο στοίβες που περιέχουν από δύο αντικείμενα η κάθε μία, ξεκινάμε από την αρχική κατάσταση, όπου κάθε παίκτης μπορεί να αφαιρέσει 1 ή 2 αντικείμενα από μία στοίβα τη φορά. Υποθέτουμε ότι ο παίκτης max παίζει πρώτος και στοχεύει να είναι ο τελευταίος που θα πάρει ένα αντικείμενο, δηλαδή αυτός που θα αφήσει τον αντίπαλο χωρίς επιλογές.

1 . Ξεκινάμε με δύο στοίβες που έχουν από 2 αντικείμενα η καθεμία: (2, 2).

2. Από την κατάσταση (2, 2), ο παίκτης max μπορεί:

- Να αφαιρέσει 1 αντικείμενο από την πρώτη στοίβα, οδηγώντας στην κατάσταση (1, 2).
- Να αφαιρέσει 2 αντικείμενα από την πρώτη στοίβα, οδηγώντας στην κατάσταση (0, 2).
- Να αφαιρέσει 1 αντικείμενο από τη δεύτερη στοίβα, οδηγώντας στην κατάσταση (2, 1).
- Να αφαιρέσει 2 αντικείμενα από τη δεύτερη στοίβα, οδηγώντας στην κατάσταση (2, 0).

Ακολουθώντας όμοια λογική, δημιουργείται το ακόλουθο δέντρο (όπου res εννοτείται reuslt για τον νικητή):

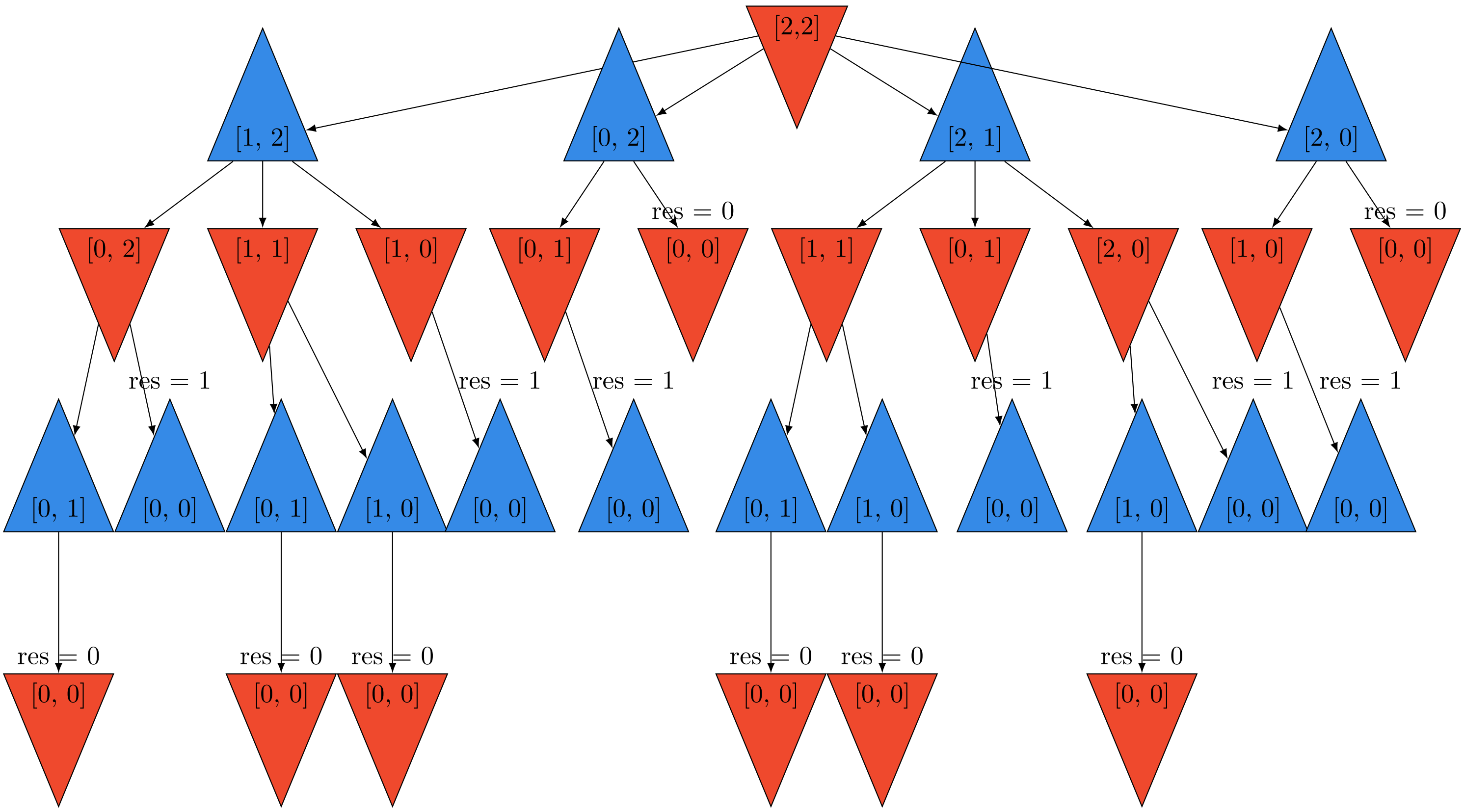

Figure 22: Δέντρο Nim

4.2 Ερώτημα 2

Εφόσον το δέντρο είναι συμμετρικό, δηλαδή η κίνηση [1,2] είναι ίδια με την [2,1] αναπτύχθηκε μόνο μια εκ των δύο, προκειμένου το δέντρο να χωρέσει στο αρχείο. Εφαρμόζοντας τον αλγόρθμο Alpha Beta Search προκύπτει το ακόλουθο δέντρο:

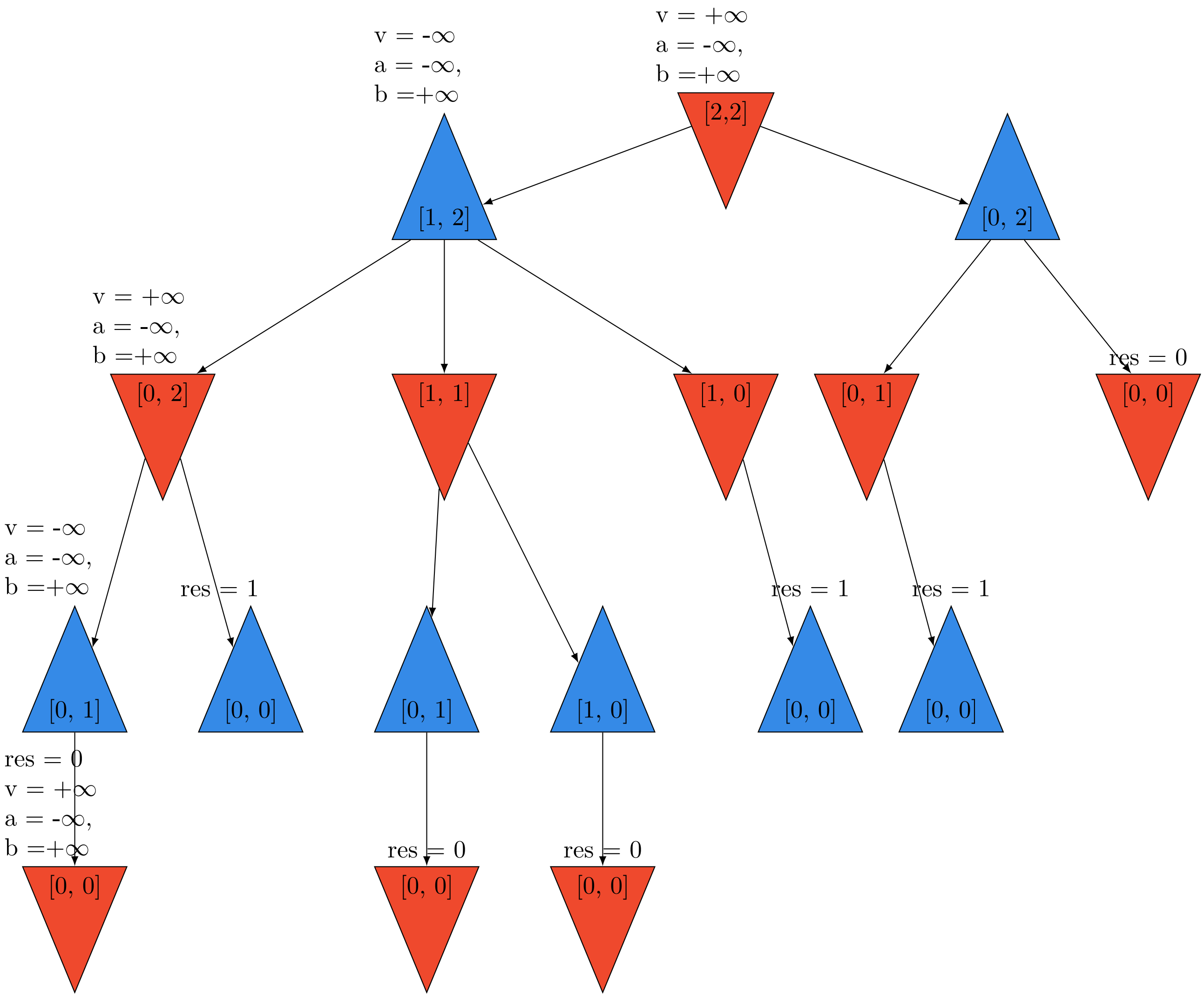

Figure 23: Πρώτη επανάληψη

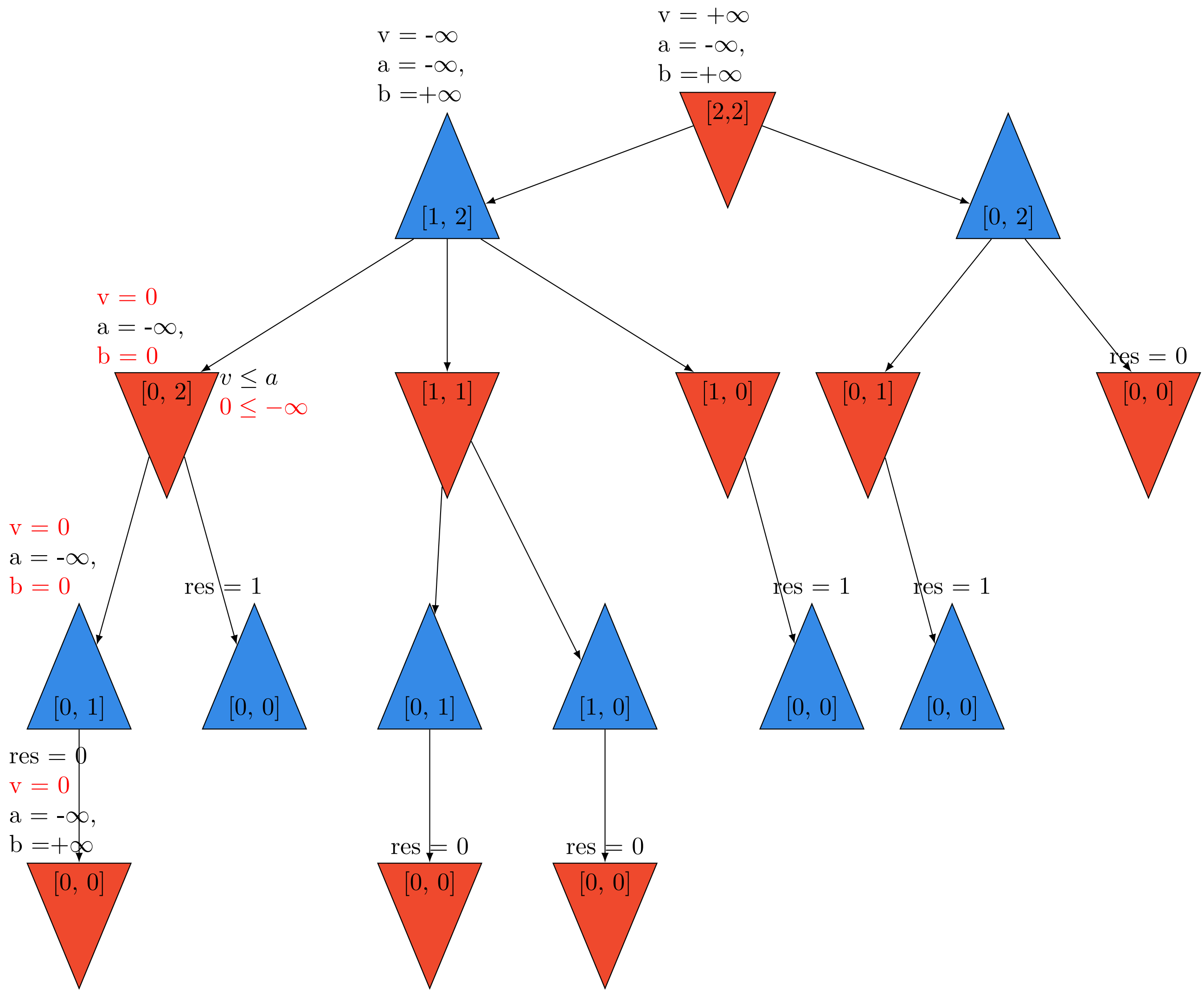

Figure 24: Δεύτερη επανάληψη

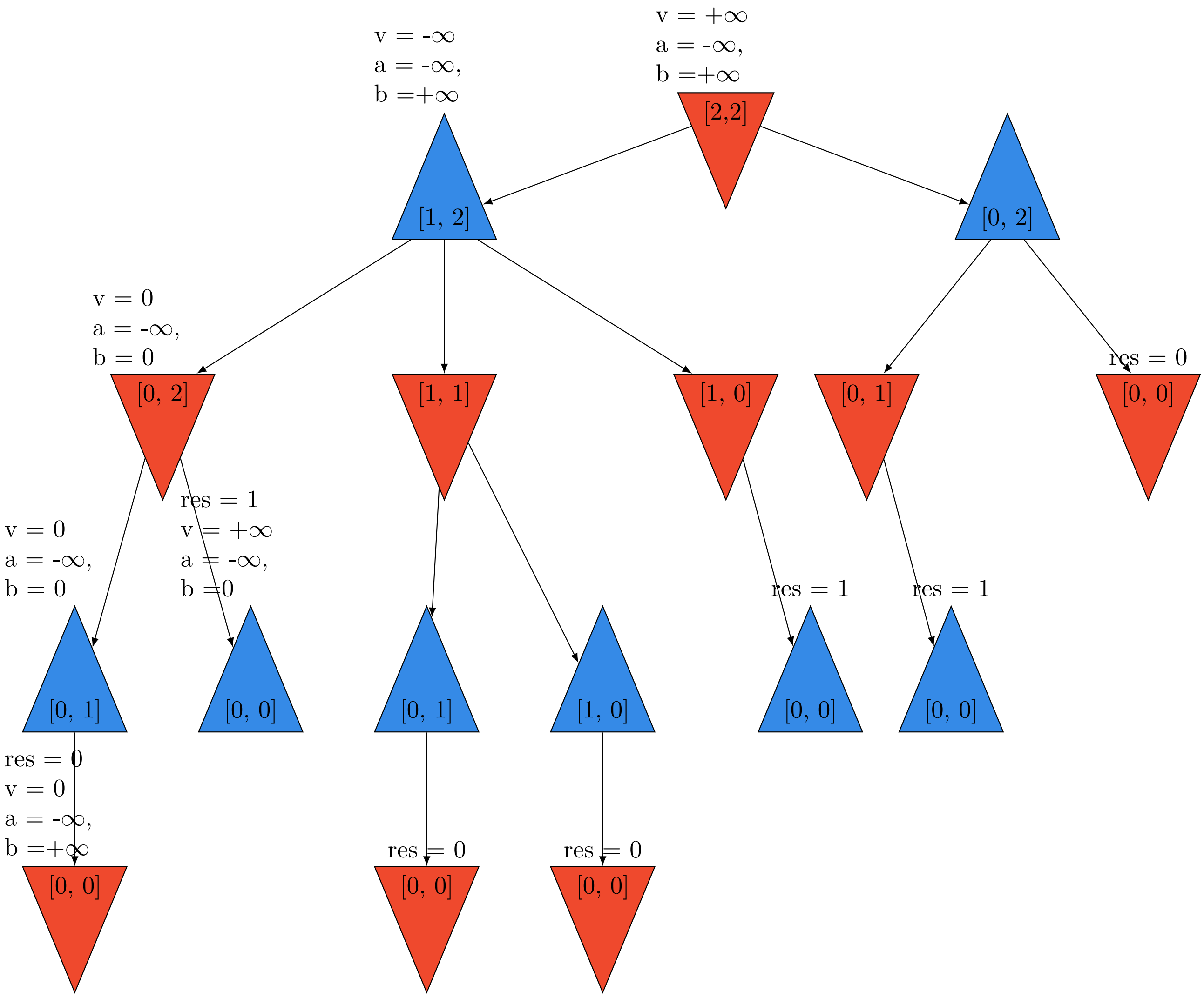

Figure 25: Τρίτη επανάληψη

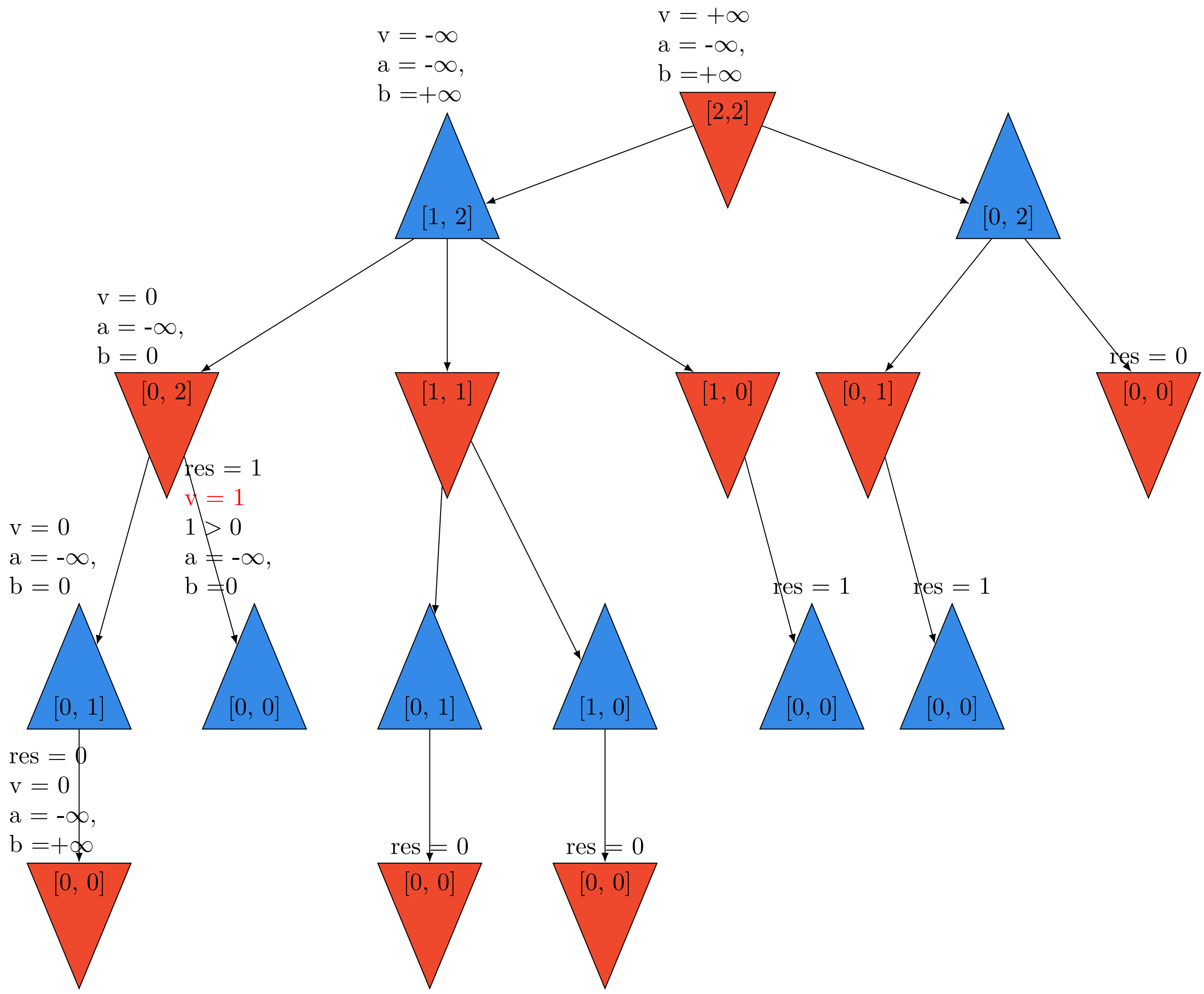

Figure 26: Τέταρτη επανάληψη

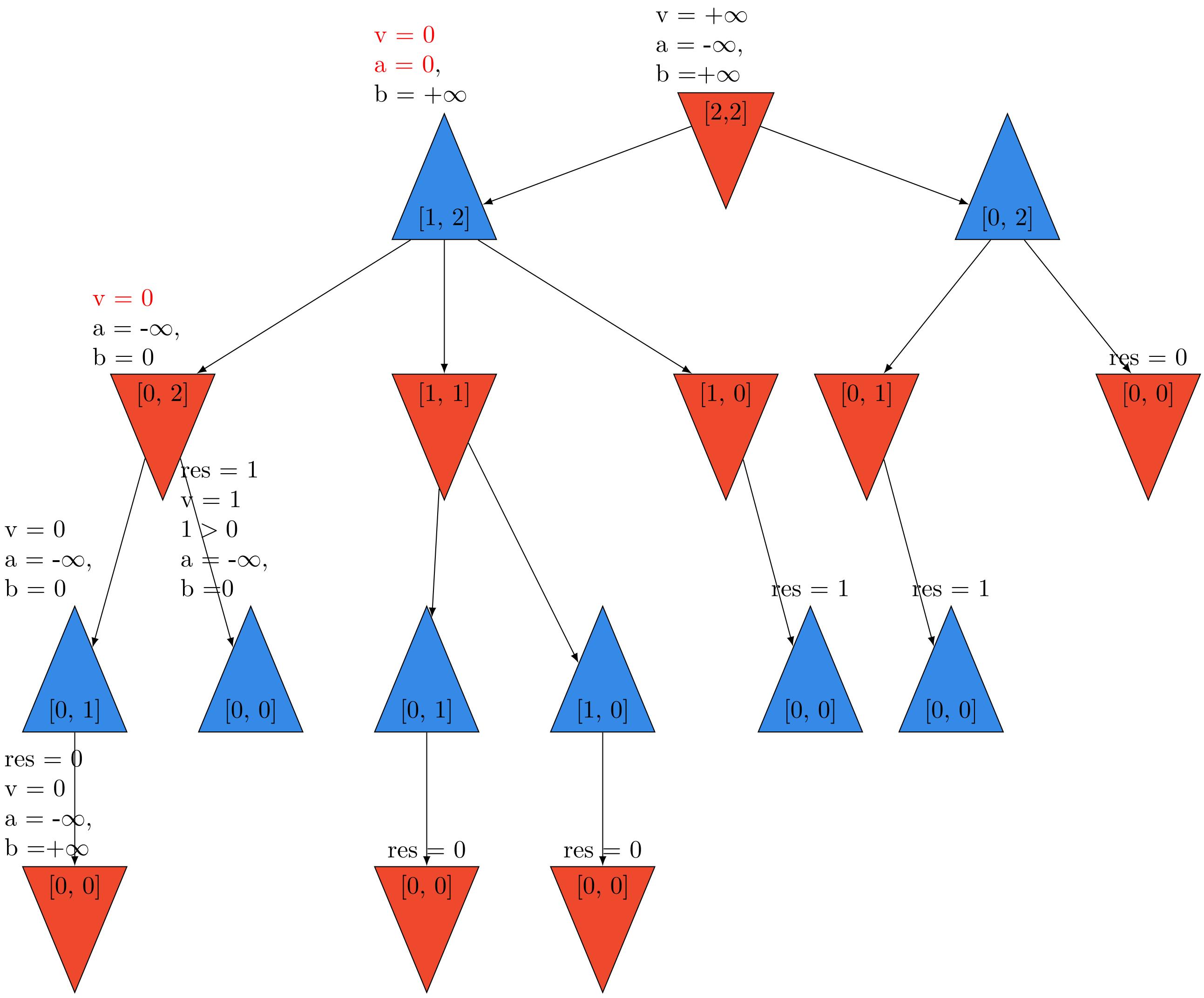

Figure 27: Πέμπτη επανάληψη

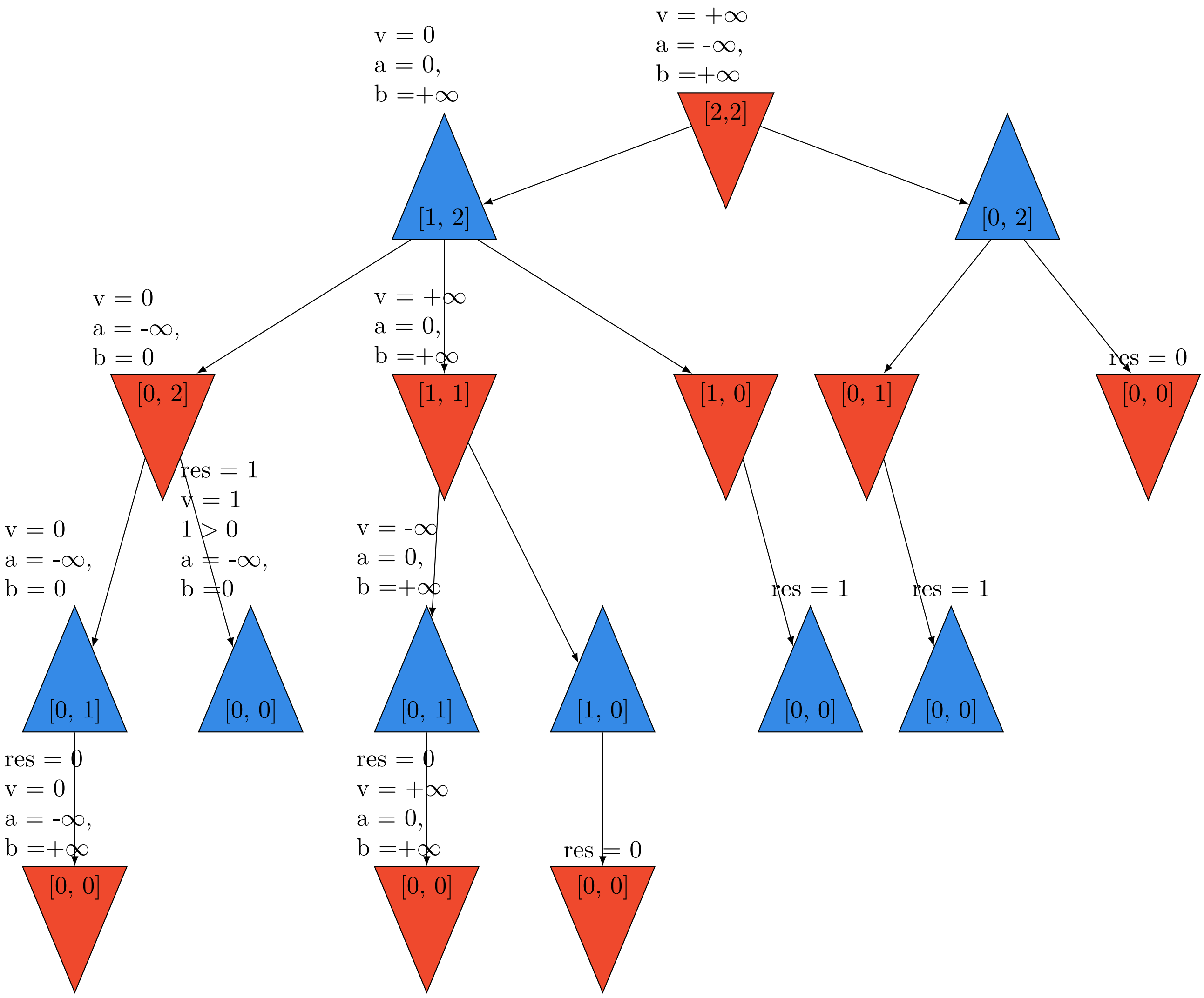

Figure 28: Έκτη επανάληψη

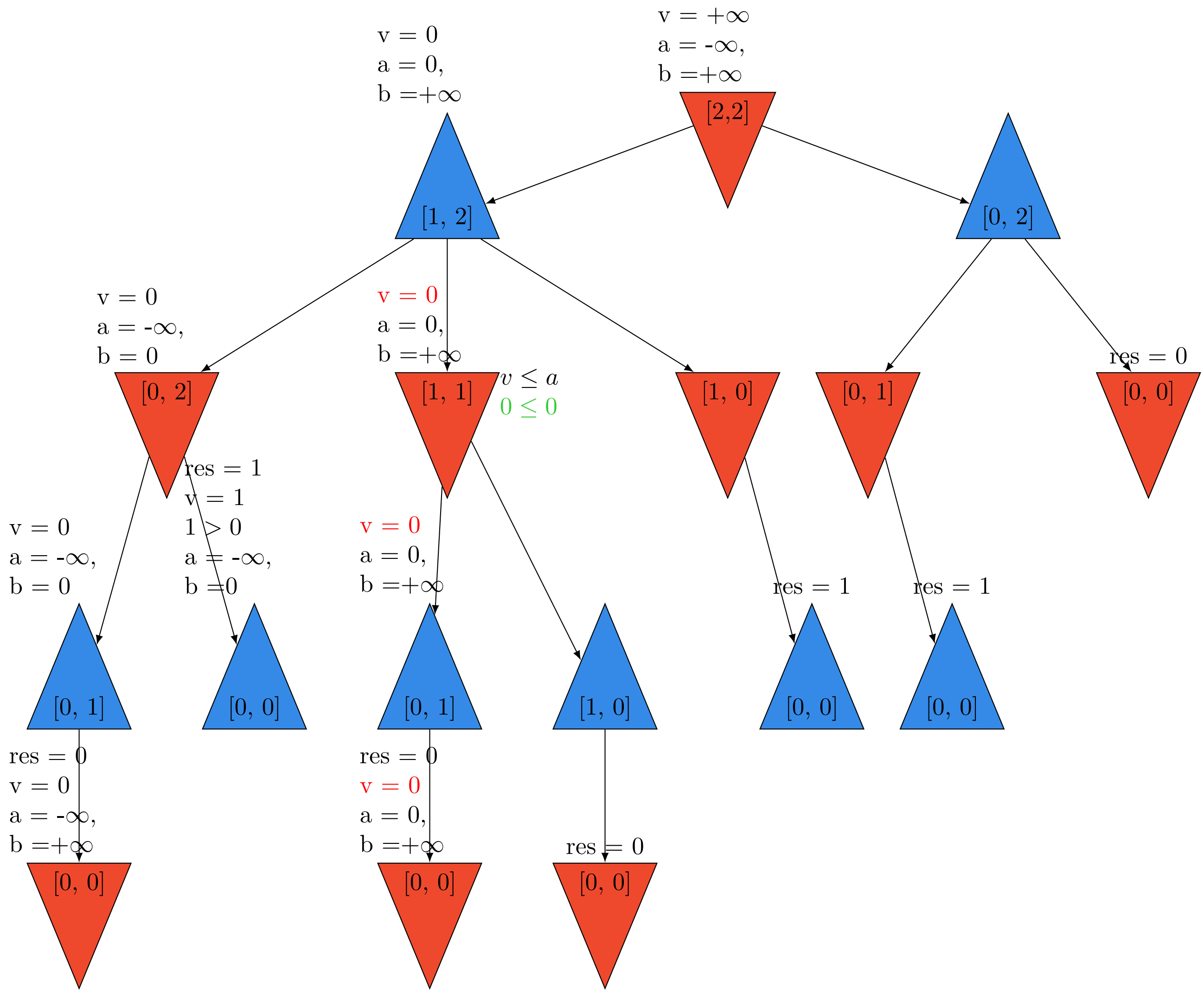

Figure 29: Έβδομη επανάληψη

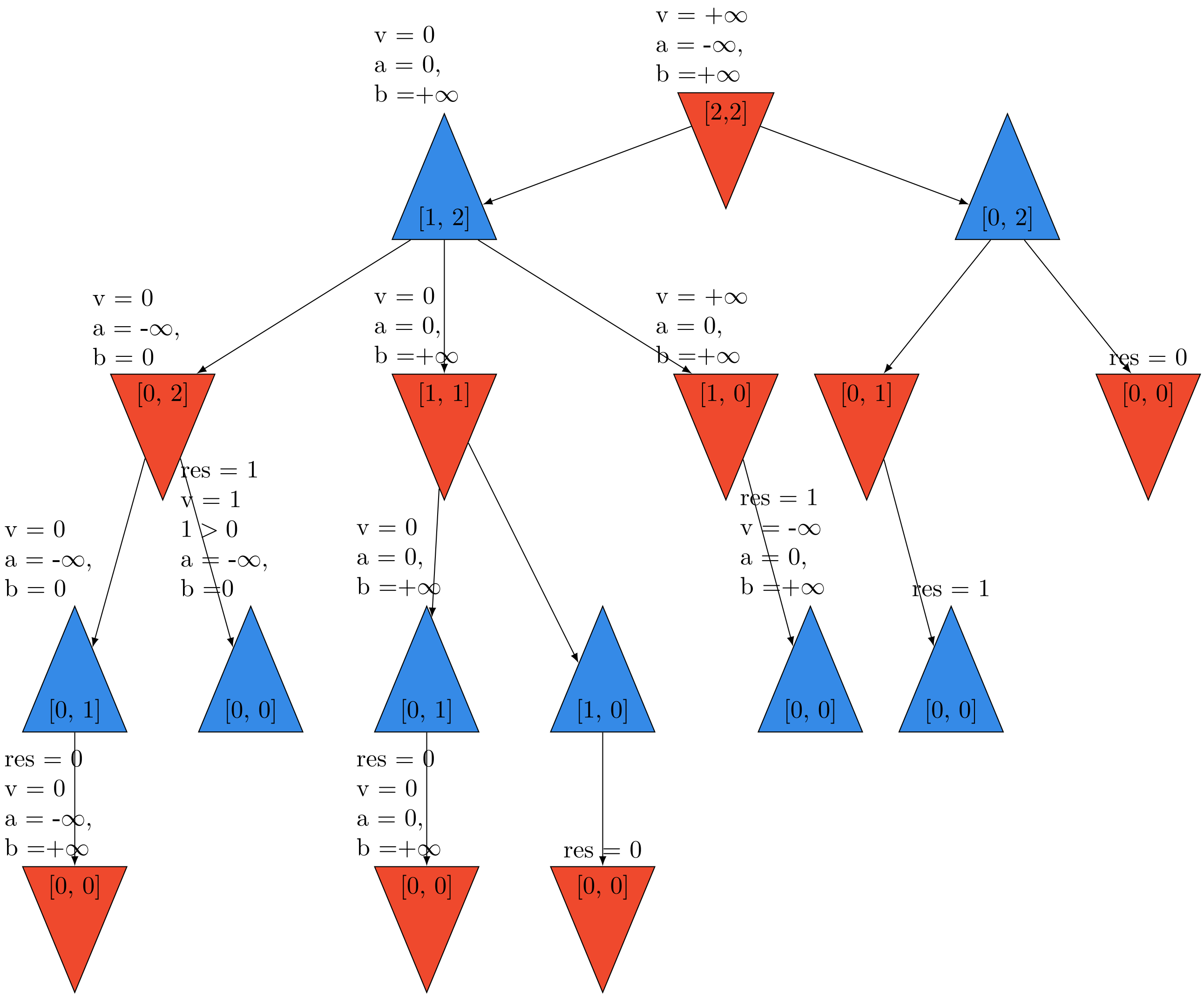

Figure 30: Όγδοη επανάληψη

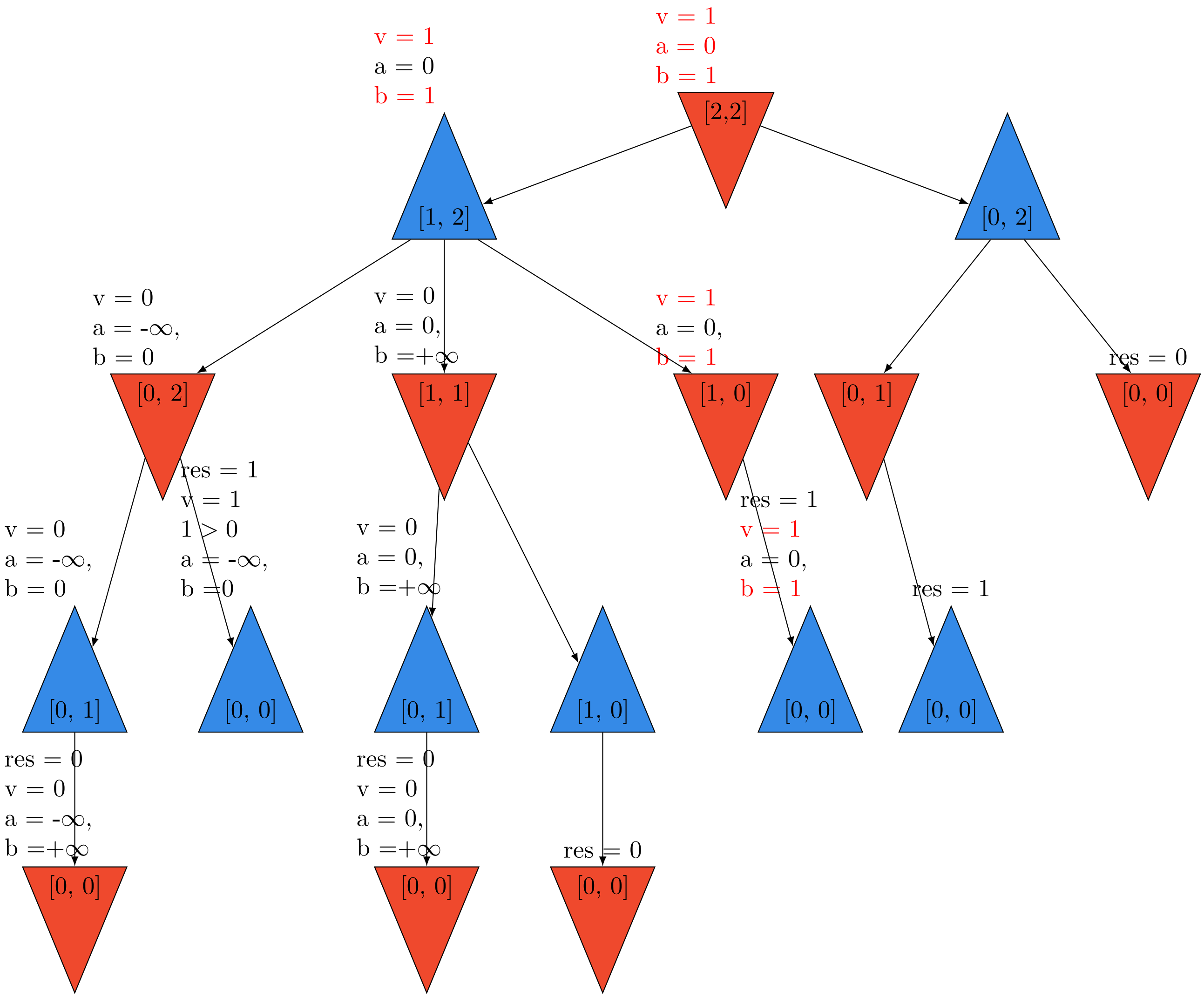

Figure 31: Ένατη επανάληψη

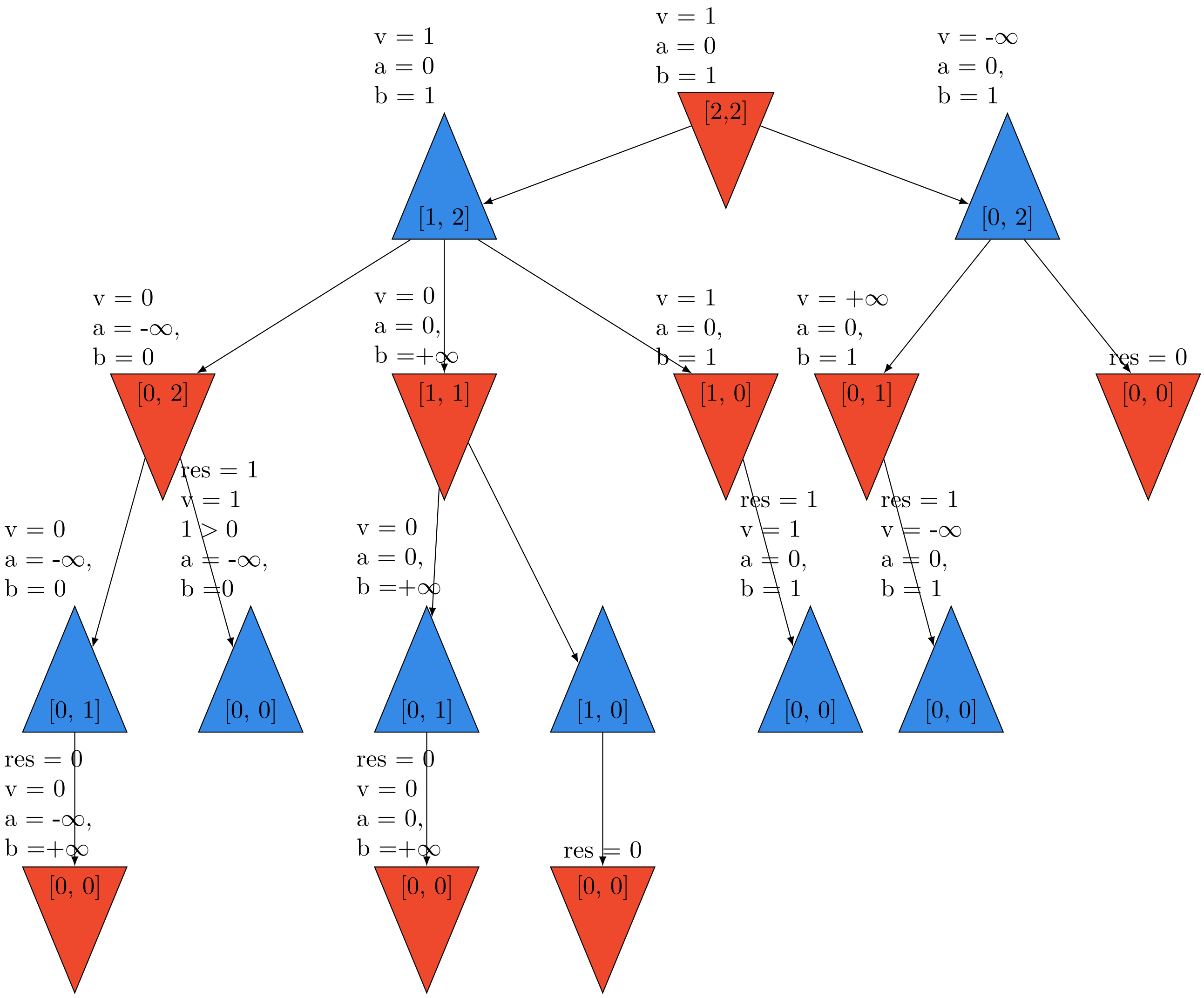

Figure 32: Δέκατη επανάληψη

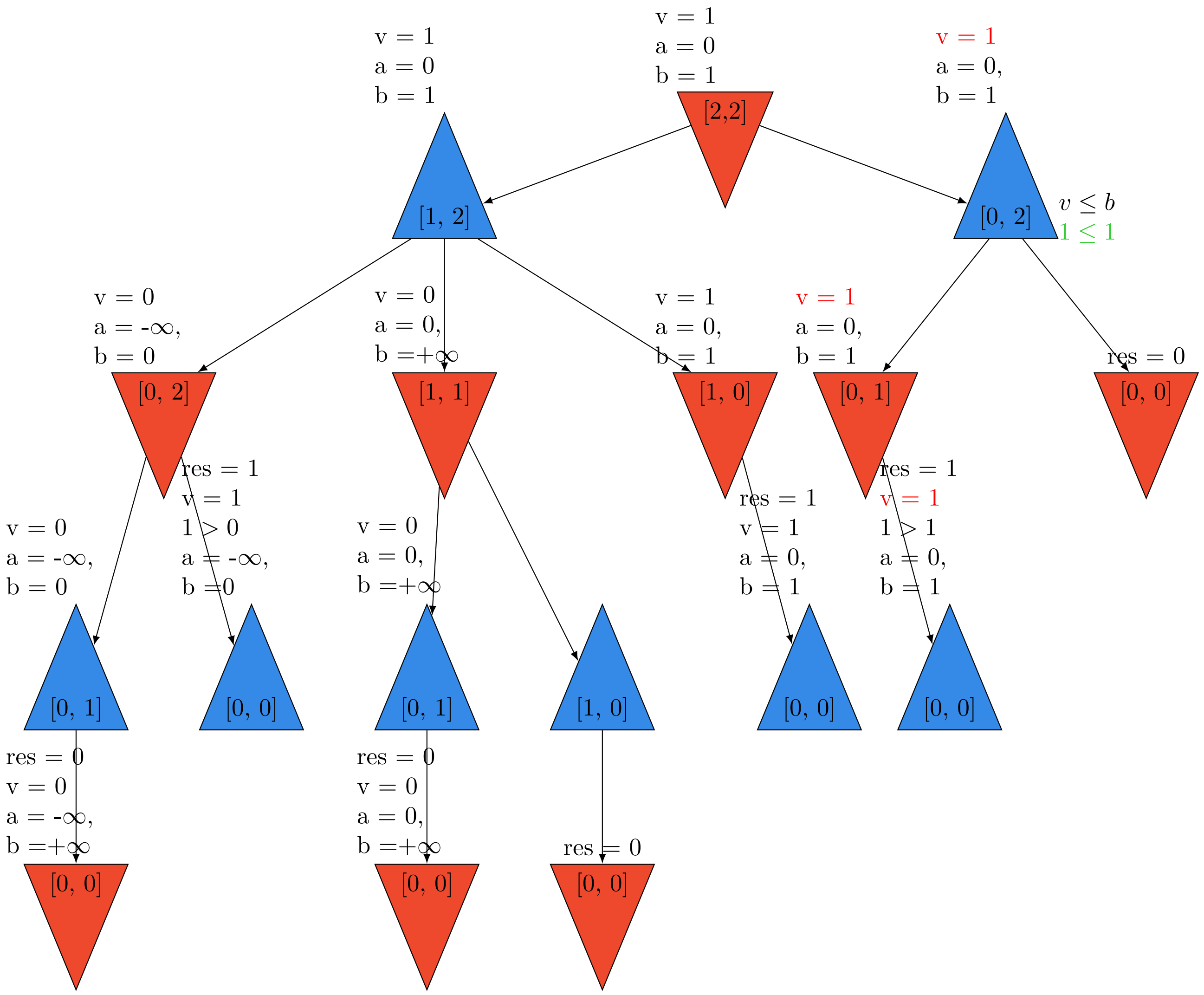

Figure 33: Ενδέκατη επανάληψη

Συνεπώς οι κόμβοι που δεν επισκεφτήκαμε είναι και αυτοί που κλαδεύονται.

4.3 Ερώτημα 3

Στην περίπτωση που και οι δύο παίκτες παίζουν αλάνθαστα ισχύει ότι ο κάθε παίκτης προσπαθεί να πάρει το καλύτερο δυνατό αποτέλεσμα "χειροτερέυοντας" ταυτόχρονα το αποτέλεσμα του αντιπάλου. Εφόσον ο max παίζει πρώτος, και γνωρίζει ότι και ο min παίζει βέλτιστα, θα διαλέξει την μικρότερη δυνατή τιμή απο την πρώτη διακλάδωση. (0 XOR 2 = 2, αφού η χρησιμότητα υπολογίζεται χρησιμοποιώντας την θεωρία του "Nim-sum" - δυαδική πράξη XOR). Στο επόμενο επίπεδο, ο min αφαιρώντας 2 κομμάτια απο την άλλη στοίβα κερδίζει. Συνεπώς, ο νικητής θα είναι όποιος παίξει δεύτερος.