ΠΑΝΕΠΙΣΤΗΜΙΟ
ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΜΕΛΕΤΩΝΤΑΣ ΚΙΝΗΣΕΙΣ ΚΑΙ ΣΤΡΑΤΗΓΙΚΕΣ ΣΤΟ ΣΚΑΚΙ ΜΕ ΤΗ ΧΡΗΣΗ ΤΟΥ MATLAB

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΜΑΤΑΝΑΣ ΧΡΗΣΤΟΣ

Α.Μ 2397

ΝΤΑΣΗΣ ΕΥΑΓΓΕΛΟΣ

Α.Μ. 2741

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ: *ΔΡ. ΜΑΥΡΑΤΖΑΣ ΣΤΥΛΙΑΝΟΣ*

Καστοριά, 2021

ΠΕΡΙΛΗΨΗ

Στην παρούσα εργασία θα μελετήσουμε τις κινήσεις και τις στρατηγικές στο σκάκι. Η πολύπλοκη φύση του παιχνιδιού αυτού μας προσφέρει την δυνατότητα να εφαρμόσουμε αλγόριθμους τεχνητής νοημοσύνης, στατιστικές αναλύσεις και ευρετικές μεθόδους. Βλέπουμε δηλαδή ότι συνδυάζει ένα ευρύ φάσμα από επιστήμες πράγμα που το ξεχωρίζει από τα υπόλοιπα επιτραπέζια παιχνίδια. Η υλοποίηση διαλέξαμε να είναι σε MATLAB γιατί είναι μια γλώσσα με πολλές ενσωματωμένες συναρτήσεις που μας διευκολύνουν κατά πολύ στην εφαρμογή πολύπλοκων μαθηματικών αλγορίθμων. Επιπλέον, θα δούμε πώς η υλοποίηση μπορεί να ενισχυθεί από το UCI (συγκεκριμένα από το Stockfish 14) που είναι ένα ανοιχτό πρωτόκολλο επικοινωνίας και επιτρέπει στις μηχανές σκακιού να επικοινωνούν με διεπαφές χρήστη. Στις επόμενες σελίδες θα δούμε αναλυτικά τι χρειάζεται για να παίξει σκάκι ένας υπολογιστής και πιο συγκεκριμένα:

- Τον τρόπο αναπαράστασης της σκακιέρας στη μνήμη, έτσι ώστε να γνωρίζει ποια είναι η κατάσταση του παιχνιδιού. *(θα το δούμε στο Κεφ. 3 , Σελίδες 37 – 38)*
- Κανόνες για τον καθορισμό του τρόπου δημιουργίας νόμιμων κινήσεων, έτσι ώστε να μπορεί να παίζει σωστά *(θα το δούμε αναλυτικά στο Κεφ. 1 , Σελίδες 13 – 18)*
- Κανόνες για τον καθορισμό του τρόπου υπολογισμού της αξίας που έχει το κάθε πιόνι, την αναπαράσταση(σκακιστική γραφή) και τον τερματισμό του παιχνιδιού *(θα τα δούμε αναλυτικά στο Κεφ. 1, Σελίδες 19 – 27)*

- Μια τεχνική για να επιλέξετε την κίνηση που θα πραγματοποιήσει ανάμεσα σε όλες τις νόμιμες δυνατότητες, έτσι ώστε να μπορεί να επιλέξει μια κίνηση αντί να αναγκαστεί να επιλέξει μια τυχαία. Επίσης, έναν τρόπο σύγκρισης κινήσεων και θέσεων, ώστε να κάνει έξυπνες επιλογές. *(θα τα δούμε στο Κεφ.3 , Σελίδες 39 – 46)*
- Τον κώδικα *(θα τον δούμε στο Κεφ. 4, Σελίδες 49 – 77)*
- Κάποιο είδος διεπαφής χρήστη - Graphical User Interface και αν θέλουμε κάποιον εξωτερικό κινητήρα σκακιού. *(θα τα δούμε στο Κεφ. 2, Σελίδες 27-32 και στο Κεφ.5 , Σελίδες 78 – 88)*

ABSTRACT

In this project we will study movements and strategies of chess. The complex nature of this game gives us a chance to implement artificial intelligence algorithms, statistical analysis and heuristic methods. Therefore, we see that it combines a wide variety of sciences which makes it different from other board games. We chose to use MATLAB for the implementation as it is a programming language with many embedded functions that make it easier for us to apply complex algorithms. In addition, we will see how our implementation can be strengthened by the UCI tool (Stockfish 14) which is an open communication protocol that allows chess machines to communicate with the user interface. In the next pages we will study in detail the whole procedure of a computer that makes it able to play chess.

To be more specific:

- The way chessboard uses memory in order to be updated with the game status (Section 3, page 37-38)
- The rules to determine a way to create legal movements in order to play right (Section 1, page 13-18)
- The rules to determine a way to calculate the value of each piece, the presentation (chess writing) and the termination of the game (Section 1, page 19-27)
- A technique in order to choose a move to apply among all legal choices and not just make a random move. Also, we will look through a way of comparing the moves so the computer can make smart choices (Section 3, page 39-46)

- Look through the code (Section 4, page 49-77)
- Some kind of Graphical User Interface and a chess engine (Section 5, 78-88)

ΕΙΣΑΓΩΓΗ

Ένα από τα πιο ενδιαφέροντα επιτραπέζια παιχνίδια είναι αδιαμφισβήτητα και το Σκάκι. Ένα παιχνίδι που σχεδόν όλοι το έχουμε παίξει κάποια στιγμή στην ζωή μας ή συνεχίζουμε να ασχολούμαστε με αυτό. Το Σκάκι έχει την ικανότητα να προσελκύει το ενδιαφέρον μικρών και μεγάλων, δεν έχει ηλικία, μπορεί να παιχτεί από ανθρώπους αλλά και από μηχανές και συνδυάζει αγωνία, στρατηγική και λογική. Σήμερα το σκάκι χαρακτηρίζεται «βασιλιάς των παιγνιδιών» ως το δυσκολότερο, επιστημονικότερο και ωραιότερο επιτραπέζιο παιχνίδι. Παίζεται από εκατομμύρια ανθρώπους σε όλο τον κόσμο, ενώ νέοι τύποι του παιχνιδιού, που διαφέρουν από τον κλασσικό ως προς τους κανόνες, έχουν επίσης καταστεί σημαντικά δημοφιλής.

ΠΕΡΙΕΧΟΜΕΝΑ

Περιεχόμενα Εικόνων

Κεφάλαιο 1^{ο} – Το σκάκι

1.1 Ιστορική Αναδρομή

Η ιστορία του σκακιού ξεκινάει αιώνες πριν και συγκεκριμένα εμφανίζεται στην Ινδία και στην Περσία με την ονομασία Chatrang. Την εποχή εκείνη οι κανόνες ήταν διαφορετικοί σε σχέση με σήμερα. Τα πιόνια είχαν διαφορετικές ιδιότητες και νικητής ήταν αυτός που κατατρόπωνε τον ‘βασιλιά’. Στη συνέχεια, η Περσία κατακτήθηκε από τους Άραβες, οι οποίοι ενσωμάτωσαν το σκάκι στην καθημερινότητά τους και διατήρησαν τα ονόματα των κομματιών, εξίσου και τους κανόνες.

Τον 9^{ο} αιώνα το σκάκι μεταφέρθηκε στην νότια Ευρώπη από τους Μαυριτανούς, οι οποίοι ήταν υπεύθυνοι για την διεύρυνση του σε όλη την Ευρώπη. Ύστερα, ακολούθησαν τα βόρεια παράλια της Αφρικής, το Βυζάντιο και η μακρινή Ανατολή. Η δημοτικότητα του στη Βόρεια Ευρώπη αναπτύχθηκε όταν δημιουργήθηκαν τα πιόνια-φιγούρες. Τον 15^{ο} αιώνα οι κανόνες άρχισαν να αλλάζουν και το σκάκι πέρασε από πολλές διαφορετικές περιόδους μέχρι να φτάσει τη σημερινή μορφή.

Στα μέσα του 19^{ου} αιώνα το σκάκι απέκτησε τα δικά του τουρνουά, ένα από αυτά, πραγματοποιήθηκε στο Λονδίνο το 1851 και μετά από 35 χρόνια ακολούθησε το πρώτο επίσημο παγκόσμιο πρωτάθλημα. Έτσι γράφτηκε μία νέα ιστορία με τον πρώτο παγκόσμιο πρωταθλητή τον Βίλχελμ Στάϊνιτς.

Τέλος, τον 21^{ο} αιώνα το σκάκι ενώνεται με τον τομέα της τεχνολογίας και το παιχνίδι ενσωματώνεται με επιτυχία στους ηλεκτρονικούς υπολογιστές δίνοντας τη δυνατότητα εκμάθησης, συμμετοχής αλλά και παρακολούθησης αγώνων ζωντανά. {2} {3}

1.2 Γενικά χαρακτηριστικά

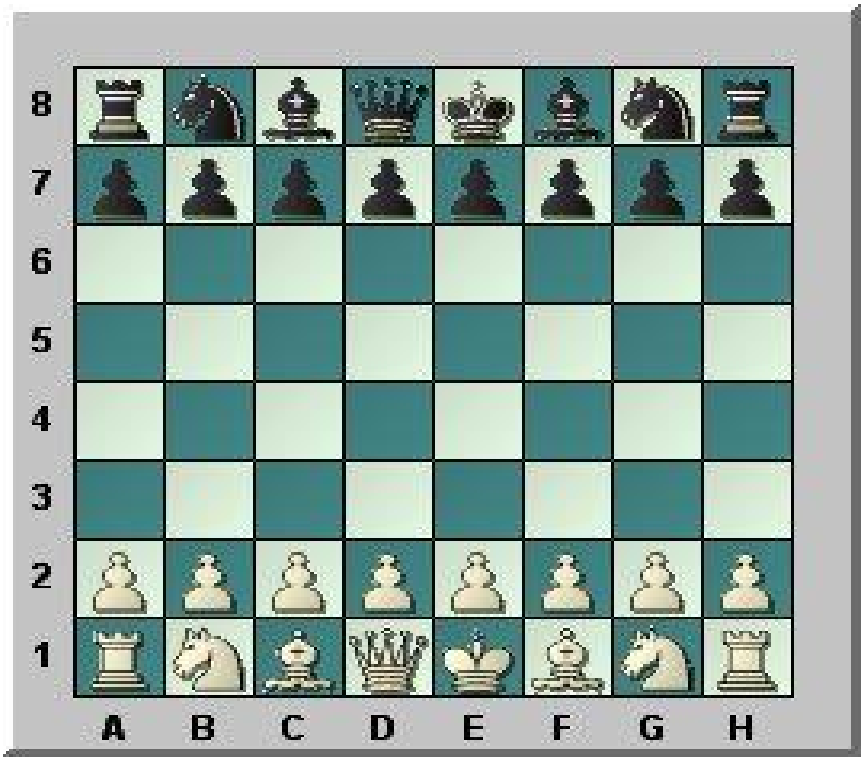
Το σκάκι είναι ένα επιτραπέζιο παιχνίδι στρατηγικής, που παίζεται ανάμεσα σε δύο παίχτες. Παίζεται στην σκακιέρα και αποτελείται από ένα διάγραμμα 64 τετραγώνων βαμμένων με μαύρο και λευκό χρώμα οριζοντίως και καθέτως σε διαστάσεις 8×8. Οι στήλες αναγνωρίζονται από τα γράμματα a έως h ενώ οι σειρές από τους αριθμούς 1 έως 8. Έτσι σε καθένα από τα 64 τετράγωνα αντιστοιχεί μία συγκεκριμένη συντεταγμένη. Κάθε παίχτης έχει στη διάθεση του 16 πεσσούς, οι οποίοι είναι ένας βασιλιά, μία βασίλισσα, δύο πύργους, δύο ίππους, δύο αξιωματικούς και οχτώ πιόνια. Κάθε ένας από τους έξι τύπους πεσσών εκτελεί διαφορετικές κινήσεις στην σκακιέρα.

Στην Εικόνα(1) απεικονίζονται όλα τα σύμβολα των διαφόρων πεσσών,

Κομμάτι		Σύμβολο στην ελληνική	Σύμβολο στην αγγλική
	Βασιλιάς	Ρ	K (King)
	Βασίλισσά	Β	Q (Queen)
	Πύργος	Π	R (Rook)
	Αξιωματικός	Α	B (Bishop)
	Ίππος	Ι	N (Knight)
	Τα πιόνια δεν σημειώνονται με κάποιο σύμβολο		

Εικόνα 1: *Οι πεσσοί*

ενώ στην Εικόνα(2) εμφανίζεται η αρχική τοποθέτηση όλων των πεσσών σε μία σκακιέρα μαζί με την αντίστοιχη αναγραφή των σειρών και των στηλών της.



Εικόνα 2:*Η αρχική τοποθέτηση τους στην σκακιέρα*

Στο σκάκι ανταγωνίζονται δύο παίκτες, όπου ένας κινεί τα λευκούς πεσσούς και ο άλλος τους μαύρους. Η πρώτη κίνηση εκτελείται από τον «Λευκό» και μετά ακολουθεί η κίνηση του «Μαύρου». Ο σκοπός κάθε παίκτη είναι να αιχμαλωτίσει τον Βασιλιά του αντιπάλου με τέτοιο τρόπο ώστε ο αντίπαλος να μην έχει την δυνατότητα να έχει κανονική κίνηση. Ο παίχτης που θα πετύχει τον σκοπό αυτό, λέμε ότι κάνει “ματ” τον βασιλιά του αντιπάλου και κερδίζει το παιχνίδι. Επίσης, πέρα από την επίτευξη ματ, γίνεται αν ο αντίπαλος παραιτηθεί αυτοβούλως, καθώς κρίνει την ήττα του αναπόφευκτη. Συχνό φαινόμενο σε έναν αγώνα είναι και η ισοπαλία, η οποία προκύπτει με διάφορους τρόπους. Τέλος, σε περίπτωση που υπάρχει ρολόι, χάνει εκείνος που θα τελειώσει ο χρόνος εκτός και αν νωρίτερα έχει γίνει ματ σε κάποιον βασιλιά.{3}

1.3 Κανόνες και κινήσεις

Οι κανόνες καθορίζονται έως σήμερα από την FIDE, το διεθνές κυβερνητικό όργανο για το σκάκι. Μερικές τροποποιήσεις υλοποιούνται από ορισμένους εθνικούς οργανισμός για τους δικούς τους στόχους. {4}{5}

Καταρχάς, το παιχνίδι ξεκινάει αυτός που κατέχει τους λευκούς πεσσούς αφού είχε προηγηθεί πριν ένας πολύ κοινός τρόπος. Ουσιαστικά, ένας από τους δύο παίκτες κρύβει έναν λευκό και έναν μαύρο πεσσό στο χέρια του και να βάλει τον αντίπαλο του να διαλέξει έναν χέρι. Οι παίκτες κάνουν κινήσεις εναλλάξ μεταξύ τους. Η πρώτη κίνηση των λευκών αν και δίνει κατά γενική πεποίθηση ένα μικρό προβάδισμα σύμφωνα με την τεράστια συσσωρευμένη εμπειρία δεν είναι καθοριστική ούτε αρκετή για να εξαναγκάσει ο λευκός την νίκη αν ο μαύρος παίζει σωστά. Οι πεσσοί είτε στο κενό είτε στις κατειλημμένες θέσεις των πεσσών του αντιπάλου. Επίσης, η αιχμαλωσία των κομματιών είναι πολύ σημαντική ενέργεια στο παιχνίδι. Η αφαίρεση των εχθρικών κομματιών από τα κομμάτια μπορεί να γίνει στα τετράγωνα που καταλαμβάνουν οι αντίπαλοι πεσσοί, με μία ελαφριά εξαίρεση τα πιόνια. Η αιχμαλωσία του βασιλιά δεν γίνεται εφικτή με αυτό τον τρόπο. Επιτυγχάνεται μόνο με το ματ, με το οποίο λήγει άμεσα το παιχνίδι. Ο παίκτης δεν πρέπει σε καμία περίπτωση να αφήσει τον βασιλιά του εκτεθειμένο, δηλαδή σε απειλή από τον αντίπαλο.{6}

Σαχ πραγματοποιείται όταν ο βασιλιάς απειλείται από έναν ή περισσότερους πεσσούς, ωστόσο υπάρχουν τρόποι αντιμετώπισης ενός σαχ:

- Η μετακίνηση του βασιλιά σε ασφαλέστερο τετράγωνο όπου δεν θα απειλείται.
- Η αιχμαλώτιση του κομματιού που απειλεί τον βασιλιά.
- Η παρεμβολή ενός κομματιού ανάμεσα στον βασιλιά και στο κομμάτι που κάνει το σαχ.

Αν καμία από τις παραπάνω περιπτώσεις δεν υφίσταται, τότε έχουμε την περίπτωση του ματ και ο παίκτης χάνει το παιχνίδι. Επιπλέον, υπάρχει το λεγόμενο διπλό σαχ, στο οποίο ένα κομμάτι κινείται κάνοντας σαχ και ταυτόχρονα ένα άλλο κομμάτι κάνει επίσης σαχ. Σε αυτή την περίπτωση μόνο ο πρώτος τρόπος αντιμετώπισης είναι νόμιμος. {7}

Η νίκη ενός παιχνιδιού μπορεί να πραγματοποιηθεί ως εξής:

- Με κίνηση ματ στον αντίπαλο βασιλιά,
- Με την εγκατάλειψη του αντιπάλου και
- Με το τέλος χρόνου το ρολογιού.

Σε περίπτωση επίτευξης ισοπαλίας ισχύουν τα εξής:

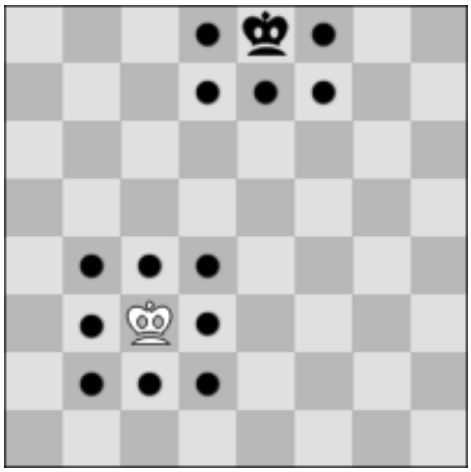
- Αν ένας παίκτης δεν έχει νόμιμες κινήσεις αλλά ο βασιλιάς του δεν είναι σαχ,
- Αν η επίτευξη ματ είναι αδύνατη,
- Με κοινή συμφωνία των δύο παικτών,
- Γίνουν 50 κινήσεις από κάθε παίκτη χωρίς να αιχμαλωτιστεί κανένας πεσσός και να κινηθεί κανένα πιόνι και
- Προκύψει τρεις φορές η ίδια θέση, με τον ίδιο παίκτη να έχει σειρά και τα ίδια δικαιώματα στις κινήσεις.

Σε περίπτωση που κριθεί αδύνατη η επίτευξη ματ, όταν οι πεσσοί που απέμειναν στο παιχνίδι είναι οι εξής:

- Βασιλιάς εναντίον βασιλιά.
- Βασιλιάς και ίππος εναντίον βασιλιά.
- Βασιλιάς και αξιωματικός εναντίον βασιλιά.
- Βασιλιάς και αξιωματικοί εναντίον βασιλιά και αξιωματικών, με τους αξιωματικούς και των δύο πλευρών να καταλαμβάνουν τετράγωνα ίδιου χρώματος.

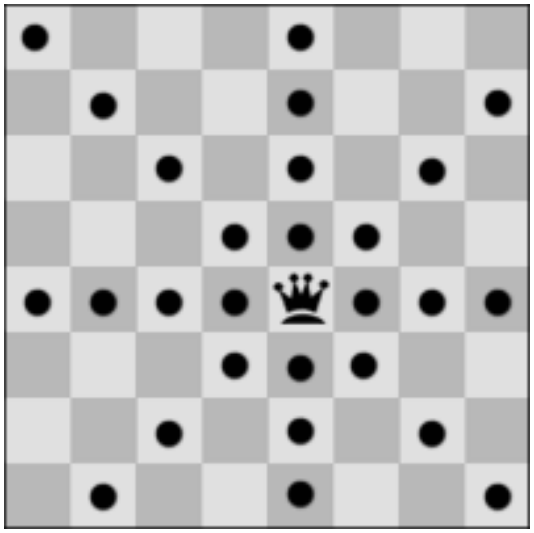
Η θέση στην οποία κανένας παίκτης δεν μπορεί να δώσει ματ με μία σειρά νομικών κινήσεων ονομάζεται **νεκρή θέση**(dead position).{6}{8}

Στις προηγούμενες ενότητες είχε αναφερθεί ότι σε ένα παιχνίδι σκάκι υπάρχουν έξι διαφορετικοί τύπο πεσσών, οι οποίοι ξεχωρίζουν από τους κανόνες των κινήσεων τους. **Ο βασιλιάς** είναι από τα πιο σημαντικά κομμάτια στην σκακιέρα διότι μπορεί να κρίνει άμεσα την παρτίδα ενός παιχνιδιού. Ωστόσο η κίνηση του δεν είναι τόσο ισχυρή, επειδή κινείται μόνο ένα τετράγωνο οριζόντια, κάθετα ή διαγώνια. Είναι σημαντικό να αναφερθεί ότι οι δύο βασιλιάδες να βρίσκονται σε γειτονικά τετράγωνα, άρα πρέπει να έχουν απόσταση το λιγότερο ένα τετράγωνο. {9}



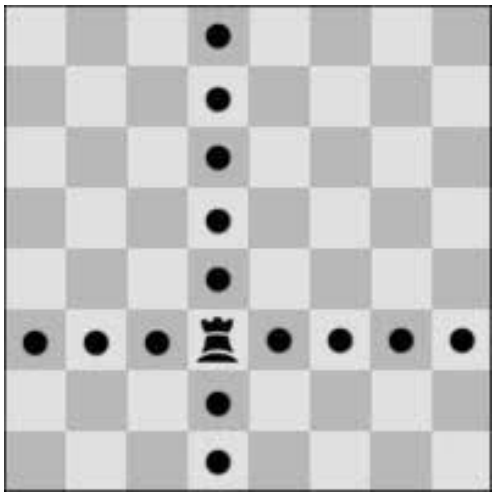
***Εικόνα 3**:Κινήσεις του βασιλιά*

Η **βασίλισσα** αποτελεί το ισχυρότερο κομμάτι του παιχνιδιού, διότι έχει την ικανότητα να μετακινείται σε οποιοδήποτε αριθμό τετραγώνων είτε είναι αυτό κάθετα, οριζόντια ή διαγώνια, συνδυάζοντας την κίνηση του πύργου και του αξιωματικού. Επίσης, δεν μπορεί να υπερπηδήσει πάνω από κάποιο άλλο κομμάτι της σκακιέρας. {10}



Εικόνα 4: *Κίνηση της βασίλισσας*

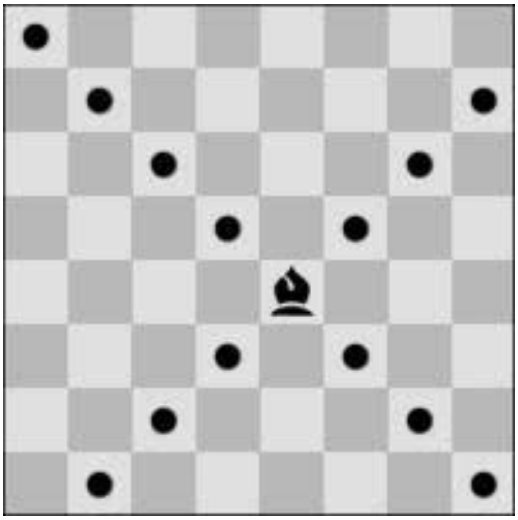
Ο **πύργος** μπορεί να μετακινηθεί οριζόντια ή κάθετα σε οποιοδήποτε αριθμό τετραγώνων. Το τετράγωνο πρέπει να είναι κενό ή κατειλημμένο από κάποιο αντίπαλο πεσσό. Επίσης, δεν μπορεί να υπερπηδήσει πάνω από κάποιο άλλο κομμάτι της σκακιέρας. {11}



Εικόνα 5: *Κινήσεις του πύργου*

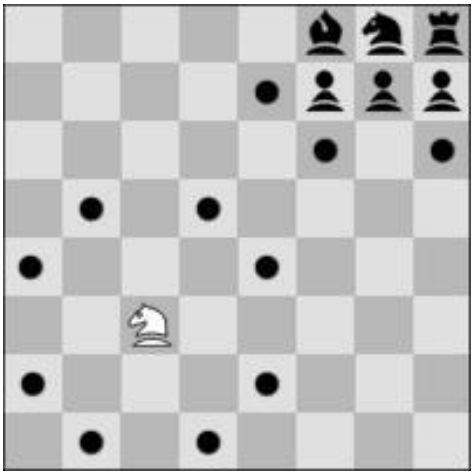
Ο **αξιωματικός** μπορεί να μετακινηθεί διαγώνια σε οποιοδήποτε αριθμό τετραγώνων. Το τετράγωνο πρέπει να είναι κενό ή κατειλημμένο από κάποιο αντίπαλο πεσσό. Δεν μπορεί να υπερπηδήσει πάνω από κάποιο άλλο

κομμάτι της σκακιέρας. Επίσης, μία επιπλέον ιδιότητα του αξιωματικού είναι ότι λόγο της διαγώνιας κίνησης του παραμένει πάντα στο χρώμα από το οποίο ξεκίνησε.{12}



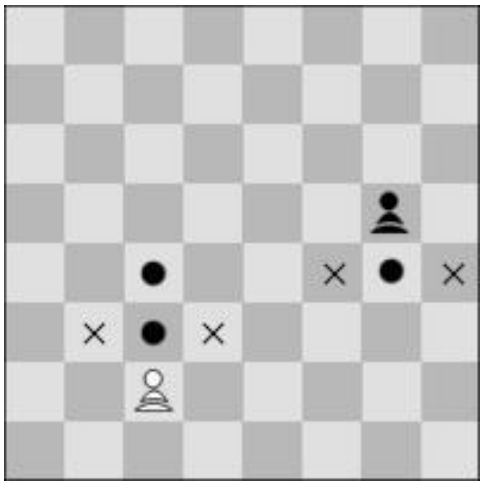
***Εικόνα 6**:Κινήσεις του αξιωματικού*

Ο **ίππος** έχει δική του ξεχωριστή κίνηση σε σχέση με τους άλλους πεσσούς μιας και μπορεί να κινείται δύο τετράγωνα κάθετα και ένα οριζόντια ή δυο τετράγωνα οριζόντια και ένα κάθετα, σχηματίζοντας το γράμμα Γ, με αυτό τον τρόπο μπορεί να έχει έως και 8 κινήσεις. Επίσης, ο ίππος είναι το μόνο μπορεί να υπερπηδήσει πάνω από κάποιο άλλο κομμάτι της σκακιέρας και να μετακινηθεί. Τέλος, μπορεί να μετακινηθεί στην αρχική του θέση.{13}



***Εικόνα 7**:Κινήσεις του ίππου*

Ο πιο αδύναμος πεσσός και ταυτόχρονα πολυάριθμος είναι το **πιόνι**. Είναι ο μοναδικός πεσσός που δεν έχει την δυνατότητα να μετακινηθεί προς τα πίσω. Το πιόνι μετακινείται μόνο μπροστά κατά ένα κενό τετράγωνο, όμως εάν κινείται για πρώτη φορά στο παιχνίδι έχει την επιλογή να κινηθεί κατά ένα ή δυο κενά τετράγωνα. Δεν μπορεί να υπερπηδήσει πάνω από κάποιο άλλο κομμάτι της σκακιέρας. Επίσης, το πιόνι δεν κινείται στην ίδια κατεύθυνση με την οποία αιχμαλωτίζει και κατά αιχμαλώτιση ενός πεσσού μετακινείται κατά ένα τετράγωνο διαγωνίως προς τα μπροστά. Αυτό που παρεκκλίνει από τον κανόνα είναι η ειδική κίνηση «εν διελεύσει» ή αλλιώς “en passant”.{14}



***Εικόνα 8**:Κινήσεις του πιονιού*

1.4 Ειδικές κινήσεις

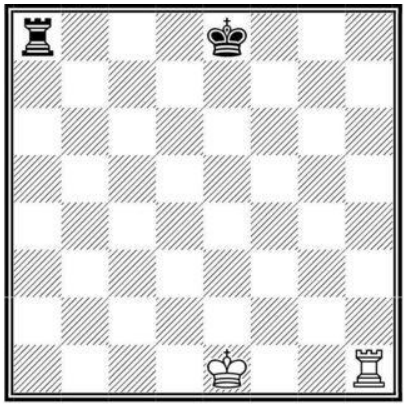
Αναφορικά υπάρχουν τρεις ειδικές κινήσεις οι οποίες δεν εντάσσονται στους κανόνες των κινήσεων που έχουν αναφερθεί προηγουμένως και είναι οι εξής κινήσεις:

- Ροκέ (castling)
- Προαγωγή (promotion)
- Εν διελεύσει (en passant)

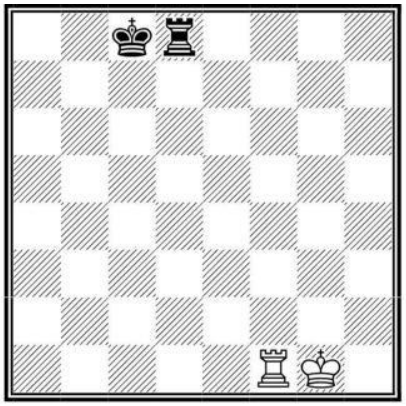
Στην ειδική κίνηση ροκέ συμμετέχουν ο βασιλιάς και ένας από τους δύο πύργους. Ο βασιλιάς μετακινείται δύο τετράγωνα προς τον πύργο και ο πύργος μετακινείται στο τετράγωνο ανάμεσα στην αρχική και τελική θέση του βασιλιά. Είναι η μοναδική κίνηση που μπορεί ένας παίκτης να μετακινήσει στη σειρά του δύο πιόνια την ίδια στιγμή.

Το **ροκέ** χωρίζεται σε δύο κατηγορίες:

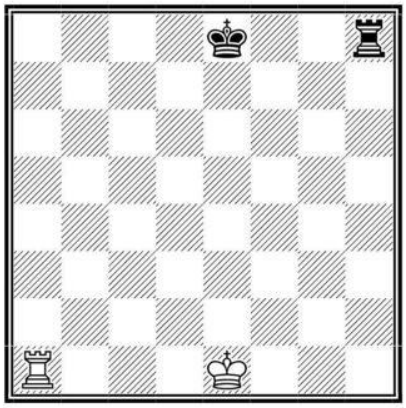
- Το μικρό ροκέ (kingside castling), που γίνεται με τον πύργο του βασιλιά
- Το μεγάλο ροκέ (queen castling), που γίνεται με τον πύργο της βασίλισσας.



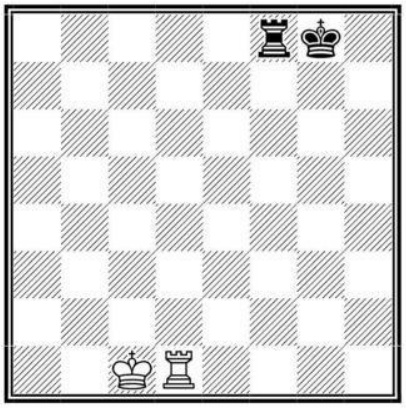
Πριν από ροκέ του λευκού στην πτέρυγα του βασιλιά
Πριν από ροκέ του μαύρου στην πτέρυγα της βασίλισσας



Μετά από ροκέ του λευκού στην πτέρυγα του βασιλιά
Μετά από ροκέ του μαύρου στην πτέρυγα της βασίλισσας



Πριν από ροκέ του λευκού στην πτέρυγα της βασίλισσας
Πριν από ροκέ του μαύρου στην πτέρυγα του βασιλιά



Μετά από ροκέ του λευκού στην πτέρυγα της βασίλισσας
Μετά από ροκέ του μαύρου στην πτέρυγα του βασιλιά

***Εικόνα 9**:Κινήσεις μικρού και μεγάλου ροκέ*

Στην συνέχεια, υπάρχουν μερικές απαιτήσεις για να πραγματοποιηθεί το **ροκέ** και είναι οι εξής:

- Ο βασιλιάς δεν πρέπει να έχει μετακινηθεί ποτέ από την αρχική θέση του.
- Ο πύργος που θα συμμετάσχει επίσης δεν πρέπει να έχει μετακινηθεί ποτέ από την αρχική του θέση.

- Ο βασιλιάς δεν πρέπει να είναι υπό την απειλή σαχ την δεδομένη στιγμή.
- Το ενδιάμεσο τετράγωνο από το οποίο θα περάσει ο βασιλιάς δεν πρέπει να απειλείται,
- Το τελικό τετράγωνο που θα καταλήξει ο βασιλιάς βεβαίως δεν πρέπει να απειλείται.
- Δεν πρέπει να παρεμβάλλονται κομμάτια μεταξύ του βασιλιά και του πύργου.

Ο σκοπός του **ροκέ** είναι η καλύτερη δυνατή προστασία του βασιλιά στα μέσα της παρτίδας και οι πύργοι να αξιοποιηθούν καλύτερα. Γενικά, είναι σημαντική κίνηση για τον παίκτη. Το πιο συνηθισμένο είναι το μικρό ροκέ, αφού είναι ταχύτερο και ασφαλέστερο για τον βασιλιά χωρίς να χρειάζεται να απομακρυνθούν πολλά κομμάτια από τις θέσεις τους. {15}

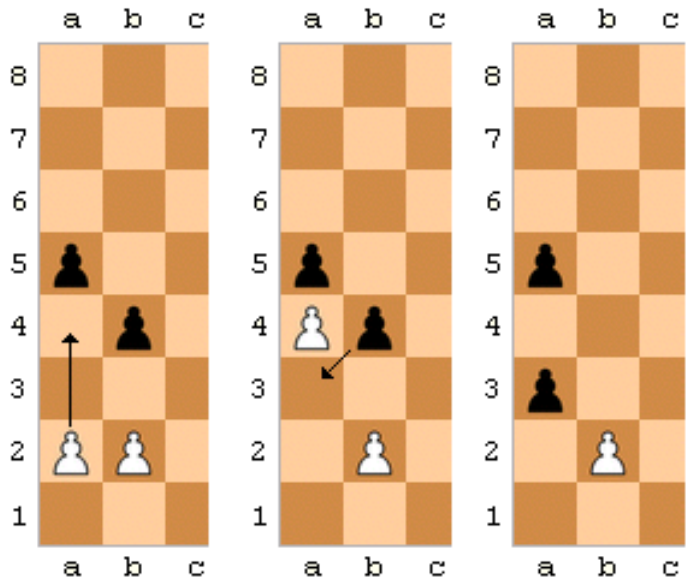
Η **προαγωγή** είναι ένας όρος στο σκάκι στον οποίο ένα πιόνι φτάνει στην τελευταία γραμμή και αλλάζει με έναν βασιλιά ή πύργο η αξιωματικό η ίππο ίδιου χρώματος, αυτομάτως το πιόνι αντικαθιστάτε στην ίδια κίνηση. Δεν υπάρχει κανένας περιορισμός σε κομμάτια που είχαν ήδη αιχμαλωτιστεί, με αποτέλεσμα ένας παίχτης μπορεί να έχει στην κατοχή του την δυνατότητα δύο ή περισσότερες βασίλισσες, τρείς ή περισσότερους αξιωματικούς. Επειδή η βασίλισσα είναι το ισχυρότερο κομμάτι του παιχνιδιού υπάρχει μια συντριπτική πλειοψηφία που την αφορούν.{16}

Η κίνηση **‘’εν διελεύσει’’ ‘’en passant’’** είναι κανόνας που αφορά την αιχμαλώτιση των πιονιών στο σκάκι, όταν ένα πιόνι κάνει

κίνηση δύο τετραγώνων από το αρχικό του τετράγωνο και θα μπορούσε να είχε αιχμαλωτιστεί από ένα εχθρικό αν είχε προχωρήσει μόνο ένα τετράγωνο. Φυσιολογικά ένα πεσσός αιχμαλωτίζει αντίπαλους πεσσούς καταλαμβάνοντας την θέση τους. Η εν διελεύσει αποτελεί μια εξαίρεση, διότι η κίνηση αυτή επιτρέπεται να συμβεί μόνο μεταξύ πιονιών και είναι μία κίνηση όπου το επιτιθέμενο κομμάτι δεν μετακινείται στο τετράγωνο του κομματιού που έχει αιχμαλωτίσει.{17}

Οι προϋποθέσεις για την επίτευξη **en passant** είναι οι ακόλουθες:

- Το επιτιθέμενο πιόνι πρέπει να βρίσκεται στην πέμπτη θέση με το αντίπαλο πιόνι.
- Το αντίπαλο πιόνι πρέπει να έχει μόλις μετακινηθεί δύο τετράγωνα σε μία κίνηση.
- Αφού το αντίπαλο πιόνι έχει κινηθεί ήδη μπροστά το επιτιθέμενο πιόνι πρέπει να είναι στην ίδια σειρά με το αντίπαλο πιόνι.
- Η σύλληψη μπορεί να γίνει με την επόμενη κίνηση , αλλιώς χάνεται το δικαίωμα να το συλλάβει.{17}

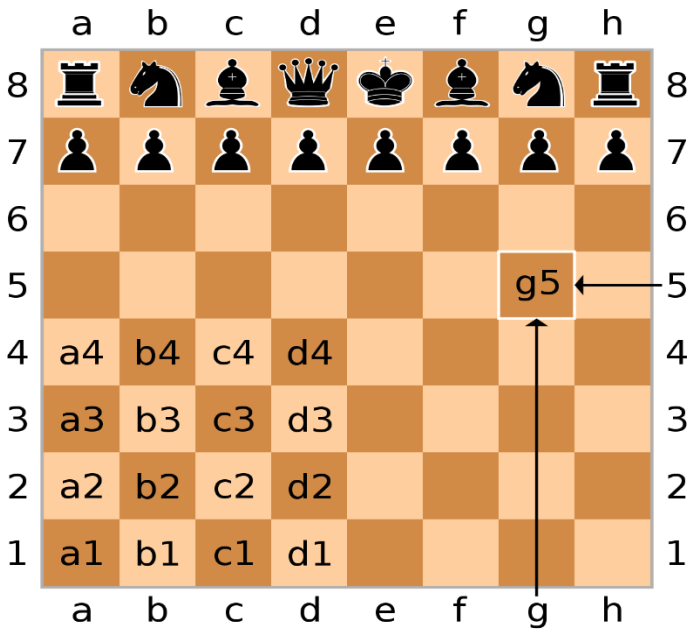


***Εικόνα 10**:Η κίνηση Εν διελεύσει*

1.5 Σκακιστική γραφή

Η σκακιστική γραφή είναι σύστημα το οποίο χρησιμοποιείται για την καταγραφή και την περιγραφή των σκακιστικών παρτίδων, επίσης βασίζεται σε ένα σύστημα συντεταγμένων για τον μοναδικό προσδιορισμό κάθε τετραγώνου στη σκακιέρα. Η αλγεβρική σκακιστική σημειογραφία εξακολουθεί να χρησιμοποιείται από παίχτες, αλλά δεν αναγνωρίζεται πλέον από το FIDE το διεθνές κυβερνητικό όργανο σκακιού.

Κάθε τετράγωνο της σκακιέρας αναγνωρίζεται από ένα μοναδικό ζεύγος συντεταγμένων, ένα γράμμα και έναν αριθμό. Οι κάθετες στήλες των τετραγώνων, που ονομάζονται αρχεία, φέρουν την ένδειξη a έως h ενώ οι οριζόντιες σειρές τετραγώνων, που ονομάζονται τάξεις, αριθμούνται από 1 έως 8. Έτσι κάθε τετράγωνο έχει μια μοναδική αναγνώριση του γράμματος της στήλης ακολουθούμενο από τον αριθμό της σειράς, για παράδειγμα το αρχικό τετράγωνο της βασίλισσας του λευκού είναι στην “d1”



***Εικόνα 11**:Συντεταγμένες των τετραγώνων*

Κάθε τύπος πεσσού (εκτός από πιόνια) αναγνωρίζεται με ένα κεφαλαίο γράμμα. Τα γράμματα που χρησιμοποιούνται είναι K(King) για τον βασιλιά, Q(Queen) για την βασίλισσα, R(Rook) για τον πύργο, B(Bishop) για τον αξιωματικό και N (Knight) για τον ίππο. Κάθε κίνηση ενός πεσσού σε κενό τετράγωνο υποδεικνύεται από το κεφαλαίο γράμμα του πεσσού, συν τη συντεταγμένη του τετραγώνου προορισμού. Για παράδειγμα Be5(μετακίνηση αξιωματικού στο e5), Nf3(μετακίνηση ίππου στο f3). Για κινήσεις πιονιού δεν χρησιμοποιείται γράμμα, δίνεται μόνο το τετράγωνο προορισμού, για παράδειγμα στο a5 περιγράφεται ως a5.

Όταν ένα κομμάτι κάνει μια σύλληψη, ένα γράμμα “x” εισάγεται αμέσως πρίν από το τετράγωνο προορισμού. Για παράδειγμα, ο Nxa5(ο ίππος αιχμαλωτίζει το αντίπαλο κομμάτι στο a5). Όταν ένα πιόνι κάνει μία σύλληψη τότε το αρχείο από το οποίο αναχώρησε το πιόνι χρησιμοποιείται για την αναγνώριση της κίνησης. Για παράδειγμα, η exd5(ένα πιόνι στην πέμπτη στήλη έχει αιχμαλωτίσει έναν αντίπαλο πεσσό στο d5).

Οι συλλήψεις en passant υποδεικνύονται καθορίζοντας το αρχείο αναγνώρισης του πιονιού που αιχμαλωτίζει, το γράμμα “x” και το τετράγωνο προορισμού(όχι το τετράγωνο του πιονιού που αιχμαλωτίστηκε).Επιπλέον μπορεί να έχει ως κατάληξη το “e.p.” που υποδεικνύει ότι η κίνηση ήταν en passant.

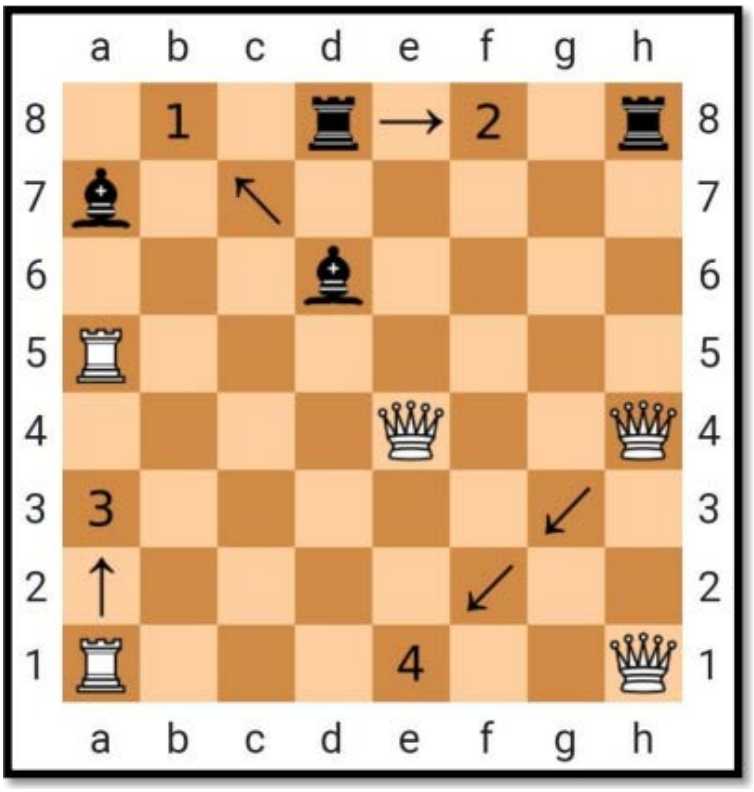
Όταν δύο οι περισσότερα πανομοιότυπα κομμάτια μπορούν να κινηθούν στο ίδιο τετράγωνο, το κινούμενο κομμάτι αναγνωρίζεται μοναδικά καθορίζοντας έτσι το γράμμα του κομματιού ακολουθούμενο σε φθίνουσα σειρά προτίμησης:

- Το γράμμα της στήλης αφετηρίας (εάν διαφέρουν).
- Τον αριθμό της σειράς αφετηρίας (εάν οι στήλες είναι οι ίδιες αλλά οι σειρές διαφέρουν).
- Το γράμμα της στήλης αφετηρίας και τον αριθμό της σειράς αφετηρίας (εάν κανένα από αυτά δεν είναι αρκετό για να προσδιορίσει ο πεσσός).

Την τελευταία περίπτωση δεν την συναντάμε συχνά, η οποία συμβαίνει όταν ένα ή περισσότερα πιόνια έχουν προαχθεί, με αποτέλεσμα ένας παίκτης να έχει την δυνατότητα να φτάσουν στο ίδιο τετράγωνο τρεις ή περισσότεροι ίδιοι πεσσοί. Για παράδειγμα, στην Εικόνα 12 το Bb8 θα ήταν ασαφές, καθώς ένας από τους αξιωματικούς στα a7 και d6 θα μπορούσε να μετακινηθεί στο b8. Η κίνηση του αξιωματικού που βρίσκεται στο d6 καθορίζεται επομένως ως Bdb8, υποδεικνύοντας ότι ήταν ο αξιωματικός από την τέταρτη στήλη ο οποίος μετακινήθηκε. Αν και θα μπορούσαν επίσης να διαφοροποιηθούν από τους αριθμούς των σειρών, το γράμμα της στήλης υπερισχύει.

Οι μαύροι πύργοι που βρίσκονται στην όγδοη θέση, θα μπορούσαν ενδεχομένως να μετακινηθούν στο f8, οπότε Rdf8 περιγράφεται η μετακίνηση του πύργου από το d8 στο f8 περιγράφεται. Επίσης οι λευκοί πύργοι που βρίσκονται στην πρώτη στήλη, θα μπορούσαν και οι δύο να

μετακινηθούν στο a3, επομένως είναι απαραίτητο να δοθεί η σειρά του κινούμενου πεσσού. Άρα η κίνηση αυτή χαρακτηρίζεται ως R1a3. Σε ότι αφορά την λευκή βασίλισσα στο h4 που μετακινείται στο e1, ούτε η στήλη ούτε η σειρά αποτελούν επαρκή δεδομένα για να αποσαφηνιστεί η κίνηση της από τις άλλες λευκές βασίλισσες. Επομένως αυτή η κίνηση γράφεται ως Qh4e1. Οι κινήσεις Bdxb8, Rdxf8, R1xa3 και Qh4xe1 θεωρούνται περιπτώσεις αιχμαλώτισης.



***Εικόνα 12**: Παραδείγματα αποσαφήνισης κινήσεων*

Το κομμάτι που προάγεται εμφανίζεται με το σύμβολο ίσον "=" στο τέλος της σημειωμένης κίνησης όταν σε μία προαγωγή ένα πιόνι μετακινείται στην τελευταία σειρά. Για παράδειγμα, η κίνηση e8=Q εφαρμόζεται όταν ένα πιόνι μετακινηθεί στο e8 και αντικατασταθεί από μία βασίλισσα. Η κίνηση αυτή συμβαίνει όταν το πιόνι βρισκόταν στην τέταρτη στήλη και αιχμαλώτιζε

έναν πεσσό στο e8 ενώ ταυτόχρονα έκανε προαγωγή. Όσον αφορά την περίπτωση του ροκέ, το μικρό ροκέ αναγράφεται ως Ο-Ο ενώ το μεγάλο ροκέ ως Ο-Ο-Ο. Επίσης, με το προστιθέμενο σύμβολο "+" περιγράφεται μία κίνηση που υποβάλει τον αντίπαλο βασιλιά σε σαχ. Ενώ όταν η κίνηση οδηγεί το παιχνίδι σε ματ, καταγράφεται με το προστιθέμενο σύμβολο "#" στην κατάληξη της. Στο τέλος του παιχνιδιού αν νικητής είναι ο παίχτης με τα λευκά το σκορ αναγράφεται ως 1-0, αντίστοιχα 0-1 αν έχει νικήσει ο παίκτης με τα μαύρα και το ½ - ½ αν έχει έρθει ισοπαλία.{18}

Κεφάλαιο 2^{ο} – Σκακιστικές Μηχανές

2.1 Deep Blue

2.1.1 Ιστορική Αναδρομή

Στις 11 Μαΐου 1997, ο παγκόσμιος πρωταθλητής σκακιού Γκάρι Κασπάροφ νικήθηκε από έναν υπολογιστή της IBM (International Business Machines) που ονομάζεται IBM Deep Blue μετά από έναν αγώνα έξι παιχνιδιών: για την Deep Blue δύο νίκες, για τον πρωταθλητή μία νίκη και τρεις ισοπαλίες. Η διάρκεια του αγώνα κράτησε αρκετές ημέρες και είχε ως αποτέλεσμα την γρήγορη διάδοση από τα Μέσα Μαζικής Ενημέρωσης σε όλο το κόσμο. Αποτελούσε την πρώτη γραμμή επικοινωνίας μεταξύ του ανθρώπου και της μηχανής. Πέρα από τον διαγωνισμό, η επιστήμη των

υπολογιστών ανέπτυξε την ικανότητα διαχείρισης πολύπλοκων υπολογισμών που ήταν απαραίτητη για την ανακάλυψη νέων ιατρικών φαρμάκων. Χάρης σε αυτό, μπορεί να χειριστεί στην αναζήτηση μεγάλου όγκου βάσεων δεδομένων και να εκτελέσει τεράστιους υπολογισμούς που απαιτούνται σε πολλούς τομείς της επιστήμης.{19} {20}

Στα τέλη της της δεκαετίας του 1940, από την εμφάνιση της τεχνητής νοημοσύνης και των πρώτων υπολογιστών, οι επιστήμονες των υπολογιστών σύγκριναν την απόδοση των «γιγαντιαίων εγκεφάλων» με τον ανθρώπινο εγκέφαλο και εφάρμοσαν το σκάκι ως τρόπο δοκιμής των υπολογιστικών ικανοτήτων των υπολογιστών. Στις αρχές της δεκαετίας του 1950, οι επιστήμονες υπολογιστών της IBM είχαν ενδιαφέρον για τον υπολογισμό του σκακιού. Το 1985, ένας μεταπτυχιακός φοιτητής στο Πανεπιστήμιο Carnegie Mellon ο Feng-hsiung Hsu, άρχισε να εργάζεται στο έργο της διατριβής του: μία μηχανή παιχνιδιού σκακιού που ονόμασε ChipTest. Με την βοήθεια περισσότερων επιστημόνων που εργάζονταν στην IBM Research δημιούργησαν την μηχανή αυτή γνωστή ως Deep Blue. Ο πρωταθλητής ανθρώπου σκακιού κέρδισε το 1996 ενάντια σε μια προηγούμενη έκδοση του Deep Blue. Ο αγώνας του 1997 τιμολογήθηκε ως «επαναληπτικός αγώνας».

Ο πρωταθλητής και ο υπολογιστής συναντήθηκαν στο Equitable Center της Νέας Υόρκης στην οποία παραβρέθηκαν και παρακολούθησαν εκατομμύρια το αποτέλεσμα. Οι εργάτες της IBM γνώριζαν ότι το μηχάνημα τους θα μπορούσε να υπολογίσει έως και 200 εκατομμύριες πιθανές θέσεις σκακιού ανά δευτερόλεπτό, ωστόσο γνώριζαν ότι οι πιθανότητες νίκης του Deep Blue δεν ήταν με το μέρος τους, αλλά η επιστήμη παρέμενε σταθερή. Ο πρωταθλητής του σκακιού κέρδισε το πρώτο παιχνίδι, ο Deep Blue πήρε το επόμενο, και οι δύο παίκτες τράβηξαν τα τρία ακόλουθα παιχνίδια. Το παιχνίδι 6 τερμάτισε τον αγώνα με μια συντριπτική ήττα του πρωταθλητή από το Deep Blue. Το αποτέλεσμα του αγώνα έγινε πρωτοσέλιδο σε όλο τον

κόσμο και βοήθησε ένα ευρύ κοινό να κατανοήσει καλύτερα τον υπολογιστή υψηλής ισχύος. Χάρης την παγκόσμια αναγνώριση από τα μέσα ενημέρωσης που δόθηκε στο Deep Blue είχε ως αποτέλεσμα περισσότερες από τρεις δισεκατομμύρια εντυπώσεις σε όλο τον κόσμο.{20}

2.1.2 Υπολογιστής και Deep Blue

Ο Deep Blue είναι ένα παράλληλο (με 32 κόμβους) RISC σύστημα το οποίο τρέχει σε έναν υπολογιστή RS/6000 SP που κατασκεύασε η IBM. Σχεδιάστηκε για να παίζει σκάκι στο υψηλότερο δυνατό επίπεδο. Ο Deep Blue ζυγίζει 1400 κιλά. Χρησιμοποιεί επεξεργαστές P2SC (Power Two Super Chip). Ο κάθε ένας από τους 32 κόμβους είναι εξοπλισμένος με 8 κάρτες ειδικών μικροεπεξεργαστών (συνολικά 256). Χρειάστηκαν 5 χρόνια και εκατομμύρια δολάρια από την IBM για την κατασκευή του. Ο Deep Blue δεν χρησιμοποιεί τεχνητή νοημοσύνη. Στηρίζεται αποκλειστικά στην υπολογιστική του ισχύ και την εκτιμητική του λειτουργία.{21}

Το Deep Blue είχε αντίκτυπο στον υπολογιστή σε πολλές διαφορετικές βιομηχανίες. Προγραμματίστηκε για να λύσει το περίπλοκο, στρατηγικό παιχνίδι του σκακιού, έτσι επέτρεψε στους ερευνητές να εξερευνήσουν και να κατανοήσουν τα όρια της μαζικής παράλληλης επεξεργασίας. Αυτή η έρευνα βοήθησε τους προγραμματιστές δίνοντας πληροφορίες σχετικά με τους τρόπους με τους οποίους θα μπορούσαν να σχεδιάσουν έναν υπολογιστή για την αντιμετώπιση πολύπλοκων προβλημάτων σε άλλους τομείς, χρησιμοποιώντας βαθιά γνώση για την ανάλυση μεγαλύτερου αριθμού πιθανών λύσεων. Η αρχιτεκτονική εφαρμογή που χρησιμοποιήθηκε στο Deep Blue υλοποιήθηκε στις τάσεις της αγοράς, στην μοριακή δυναμική, ένα πολύτιμο εργαλείο για την ανακάλυψη και ανάπτυξη νέων φαρμάκων. {20}

2.2 Stockfish

2.2.1 Τί είναι το Stockfish;

Το **Stockfish** είναι ο ισχυρότερος κινητήρας σκακιού που διατίθεται στο κοινό και υπήρξε για μεγάλο χρονικό διάστημα. Είναι μια δωρεάν μηχανή ανοιχτού κώδικα που αναπτύσσεται επί του παρόντος από μια ολόκληρη κοινότητα. Το Stockfish δεν είναι μόνο η πιο ισχυρή διαθέσιμη μηχανή σκακιού, αλλά είναι επίσης εξαιρετικά προσβάσιμη. Είναι άμεσα διαθέσιμο σε πολλές πλατφόρμες, συμπεριλαμβανομένων των Windows, Mac OS X, Linux, iOS και Android.{22}

2.2.2 Ιστορική Αναδρομή

Το πρόγραμμα προήλθε από το Glaurung , μία ανοιχτή πηγή κώδικα που δημιουργήθηκε από τον Νορβηγό Romstad και κυκλοφόρησε για πρώτη φορά το 2004. Στην συνέχεια μετά από τέσσερα χρόνια, συνεργάστηκε με τον Ιταλό Costalba, ο οποίος εμπνευσμένος από το ισχυρό κίνητρο ανοιχτού κώδικα αποφάσισε να παρακάμψει το έργο δίνοντας την ονομασία Stockfish, διότι παράγεται στην Νορβηγία. Η πρώτη έκδοση κυκλοφόρησε τον Νοέμβριο του 2008. Ακολούθησαν νέες ιδέες και αλλαγές κώδικα μεταξύ τον δύο προγραμμάτων, μέχρι την στιγμή που ο Romstad αποφάσισε τον Δεκέμβριο του 2008 να εγκαταλείψει το Glaurung και να αφοσιωθεί στη Stockfish, η οποία είχε καλύτερες προδιαγραφές. Γύρω στο 2011, ο Romstad αποφάσισε να αποσυρθεί από τη Stockfish για να ασχοληθεί με την νέα του εφαρμογή σκακιού στην iOS.{23}

Μερικά χρόνια αργότερα, η αρχιτεκτονική νευρωνικών δικτύων NNUE (Efficiently Updatable Neural Networks) συνεργάστηκε με την κοινότητα Shogi, οι οποίοι οδήγησαν ένα πολλά υποσχόμενο τομέα του Stockfish NNUE. Η Shogi είναι μια ιαπωνική παραλλαγή του σκακιού. Στις 2 Σεπτεμβρίου του 2020, κυκλοφόρησε η 12η έκδοση στην οποία σημειώθηκε μια σημαντική βελτίωση ισχυρότερη από κάθε προηγούμενη, διότι συνήθως κέρδιζε δέκα φορές περισσότερα παιχνίδια από ό,τι έχανε όταν αγωνίστηκε με την προηγούμενη έκδοση του. Η κυκλοφορία του Stockfish 13 έγινε στις 19 Φεβρουαρίου του 2021, η οποία προκλήθηκε από την έναρξη των πωλήσεων του Fat Fritz 2, βασισμένη στην πρόσφατη ανάπτυξη του με μικρές τροποποιήσεις. Το Fat Fritz 2 είναι μια τεχνολογία της NNUE και της Stockfish 12, λειτουργεί σε CPU, χωρίς να απαιτούνται ακριβές κάρτες γραφικών GPU για παιχνίδι όπως ο προκάτοχός του. Στον ακόλουθο πίνακα μπορούμε να δούμε όλες τις εκδόσεις του Stockfish.{24}

Έκδοση (από – έως)	Ημερομηνία (από – έως)
Stockfish 1.0 – 1.2	Νοέμβριος 2, 2008 - Δεκέμβριος 29, 2008
Stockfish 1.3 – 1.6.2	Μάιος 2, 2009 - Δεκέμβριος 31, 2009
Stockfish 1.6.3 – 1.9.1	Φεβρουάριος 2, 2010 – Οκτώβριος 5,2010
Stockfish 2.0 – 2.2	Ιανουάριος 1, 2011 – Δεκέμβριος 29,2011
Stockfish 2.2.1– 2.3.1	Ιανουάριος 6, 2012 – Σεπτέμβριος 22,2012
Stockfish 3 – 5	Απρίλιος 30, 2013 – Μάιος 31,2014
Stockfish 6 – 10	Ιανουάριος 27, 2015 – Νοέμβριος 29,2018
Stockfish 11 – 13	Ιανουάριος 18, 2020 – Φεβρουάριος 19,2021

2.2.3 Τα επιτεύγματα του Stockfish

Η Stockfish κυριάρχησε στο TCEC (Top Chess Engine Championship) από τότε που άρχισε να συμμετέχει. Έχει κερδίσει οκτώ πρωταθλήματα TCEC και έχει επίσης έξι τερματισμούς στη δεύτερη θέση - κατέλαβε την πρώτη ή δεύτερη θέση σε κάθε σεζόν που συμμετείχε από το 2013 με μία μόνο εξαίρεση. Από το 2018-2020 κέρδισε επτά από τις εννέα σεζόν TCEC μπροστά από τους Komodo, Leela Chess Zero, Shredder, Houdini και άλλους κινητήρες ανώτερου επιπέδου.{22}

2.4 Komodo Chess

2.4.1 Τί είναι το Komodo Chess;

Η Komodo είναι μία μηχανή σκακιού που δημιουργήθηκε και προγραμματίστηκε από τους Don Dailey, Mark Lefler και Larry Kaufman. Επίσης, είναι ένας αυτόνομος κινητήρας σκακιού που υποστηρίζει το πρωτόκολλο UCI και είναι διαθέσιμος για πολλαπλές πλατφόρμες και λειτουργικά συστήματα, επομένως είναι συμβατός με δωρεάν και εμπορικά συμβατά με UCI γραφικά περιβάλλοντα χρήστη και διεπαφές βάσης δεδομένων και κατατάσσεται στην κορυφή της λίστας μαζί με το Stockfish.{25}

2.4.2 Ιστορική Αναδρομή

Η δημιουργία του Komodo προήλθε τον Ιανουάριο του 2010, ωστόσο η πρώτη έκδοση πολλαπλών επεξεργαστών κυκλοφόρησε τον Ιούνιο του 2013 ως Komodo 5.1 MP. Η παρούσα έκδοση ήταν μια σημαντική επαναδιατύπωση και μία «ανοιχτή θύρα» του Komodo στη C++ 11. Μια έκδοση ενός επεξεργαστή του Komodo κυκλοφόρησε ως αυτόνομο προϊόν

λίγο πριν την κυκλοφορία του 5.1 MP. Η έκδοση αυτή ονομάστηκε Komodo CCT, η οποία βασίζεται στον παλαιότερο κώδικα C και ήταν περίπου 30 Elo (Σύστημα αξιολόγησης) ισχυρότερη από την έκδοση 5.1 MP.

Τον Οκτώβριο του 2013, ο Don Dailey ανακοίνωσε την κυκλοφορία του Komodo 6 και άσχημα νέα σχετικά με τη μελλοντική κατάσταση του Komodo λόγω θανατηφόρας ασθένειάς του. Παρουσίασε τον Mark Lefler ως νέος μέλος της ομάδας Komodo. Ο Don Dailey απεβίωσε στις 22 Νοεμβρίου 2013 στην ηλικία των 57 ετών.

Ένα μήνα μετά συμμετείχε στον τελικό του TCEC 5 και ανακηρύχθηκε νικητής το Komodo TCEC, το οποίο κυκλοφόρησε στις 04 Δεκεμβρίου 2013 και το βελτιωμένο Komodo 7 στις 21 Μαΐου 2014. Το Komodo 8 κυκλοφόρησε στις 5 Σεπτεμβρίου 2014 και ήταν μια σημαντική βελτίωση σε σχέση με το Komodo 7a, το οποίο είχε ήδη βαθμολογηθεί μεταξύ των τριών κορυφαίων σκακιστικών μηχανών σε σχεδόν όλες τις λίστες βαθμολογίας.{25}

Το Komodo 9, που κυκλοφόρησε στις 28 Απριλίου 2015, συνέχισε την πρόοδο με περίπου 50 Elo περισσότερο έναντι του Komodo 8 λόγω βελτιωμένης αναζήτησης, αξιολόγησης και διαχείρισης χρόνου, καθώς και αποτελεσματικότερης χρήσης της πολλαπλής επεξεργασίας. Πιο αισθητή είναι η αύξηση του βάθους αναζήτησης που επιτυγχάνεται σε ένα δεδομένο χρονικό διάστημα, η οποία οφείλεται σε καλύτερους κανόνες επέκτασης.

Στις 23 Μαΐου του 2016 ανακοινώθηκε το Komodo 10. Είναι σαφώς πιο βελτιωμένο από το Komodo 9, λόγω την αναθεώρηση της ασφάλειας του βασιλιά και συνέχισε το πρόοδο με περίπου 60 Elo ισχυρότερο. Στις 23 Μαΐου 2017, το Komodo 11.01 κυκλοφόρησε, το οποίο διόρθωσε ορισμένα δευτερεύοντα ζητήματα του Komodo 11 και εκτιμάται περίπου 55 Elo ισχυρότερο από το Komodo 10.

Το Komodo 12 κυκλοφόρησε τον Μάιο του 2018 και διαθέτει μια προαιρετική αναζήτηση Monte-Carlo Tree (MCTS), δηλαδή έναν αλγόριθμο αναζήτησης δέντρων που βασίζεται σε τυχαίες αναπαραγωγές. Τον Μάιο του 2019, το Komodo 13 κυκλοφόρησε με σημαντικές βελτιώσεις στη λειτουργία MCTS με την παράθεση του Larry Kaufman. Το Komodo 14 κυκλοφόρησε τον Μάιο του 2020 με τις ακόλουθες βελτιώσεις που ανακοινώθηκαν από τον Larry Kaufman. Η ανάπτυξη του Komodo συνεχίζεται ακόμα σήμερα με τελευταία έκδοση το Μάϊο του 2021 με ονομασία Komodo Dragon 2.{26}

2.5 Fritz

2.5.1 Τί είναι το Fritz;

Το Fritz είναι ένα εμπορικό πρόγραμμα σκακιού που αναπτύχθηκε από τον Frans Morsch. Διατίθεται σε Windows και σε άλλες πλατφόρμες αλλά και σε κονσόλες. Οποιοσδήποτε διαθέτει συσκευή με δυνατότητα σύνδεσης στο διαδίκτυο μπορεί να κατεβάσει έναν ισχυρό κινητήρα σκακιού δωρεάν, αλλά δεν συνέβαινε πάντα. Ο Fritz εξακολουθεί να είναι πολύ ισχυρός εμπορικός κινητήρας σκακιού από το τις αρχές της δεκαετίας του 1990 έως και σήμερα.{27}

2.5.2 Ιστορική Αναδρομή

Το 1991 η γερμανική εταιρεία ChessBase πλησίασε τον ολλανδό προγραμματιστή σκακιού Frans Morsch για να γράψει μια μηχανή σκακιού για να προσθέσει στο πρόγραμμα βάσεις δεδομένων που πούλησαν. Στη συνέχεια, το 1995, ο Fritz 3 κέρδισε το Παγκόσμιο Πρωτάθλημα Σκακιού Υπολογιστών, νικώντας μια πρώιμη έκδοση του Deep Blue. Το Fritz 5 κυκλοφόρησε το 1998, συμπεριλαμβανόμενης της Friend Mode, η οποία θα

έκανε τη μηχανή να προσαρμόσει το παιχνίδι κατά τη διάρκεια ενός σκακιστικού παιχνιδιού με βάση το επίπεδο που παίζει ο αντίπαλος.

Με την πάροδο του χρόνου, δεν σταμάτησε η βελτίωση του Fritz, σε κάθε νέα έκδοση πραγματοποιήθηκε ενσωμάτωση νέων λειτουργιών, καθώς και βελτιωμένο γραφικό περιβάλλον και διορθώσεις ή τροποποιήσεις του προγράμματος.

Το 2003 εφαρμόστηκε για πρώτη φορά η X3D Fritz της έκδοσης Deep Fritz η τρισδιάστατη απεικόνιση, επιπλέον το 2004 το Fritz 8 πρόσθεσαν την λειτουργεία Fun mode, η οποία έδινε την δυνατότητα στον χρήστη να επιλέξει βαθμολογία ή επίπεδο δυσκολίας της μηχανής. Αυτό είχε ως αποτέλεσμα το 2010 η έκδοση του Deep Fritz 12 να καταλάβει την έκτη θέση στη λίστα της Σουηδικής Ένωσης Υπολογιστών Σκακιού και της CCRL(Computer Chess Rating Lists). Το 2013, το Deep Fritz 14 κυκλοφόρησε με νέο δημιουργό τον Gyula Horváth και αρκετές αλλαγές όσον αφορά την μηχανή.

Το Fritz 15 κυκλοφόρησε στις 25 Νοεμβρίου του 2015 με νέα χαρακτηριστικά συμπεριλαμβανομένης της μετάβασης στη περίφημη μηχανή Rybka του Vasik Rajlich. Το Fritz 16 κυκλοφόρησε στις 12 Νοεμβρίου 2017 με μια νέα λειτουργία παιχνιδιού Easy game που υπολογίζει τις σωστές κινήσεις με πράσινο και λανθασμένες κινήσεις με κόκκινο, αυτή η έκδοση χρησιμοποίησε ξανά την μηχανή Rybka. Τέλος, το Fritz 17 κυκλοφόρησε στις 12 Νοεμβρίου 2019 και χρησιμοποιεί τον κινητήρα Ginkgo.{28}

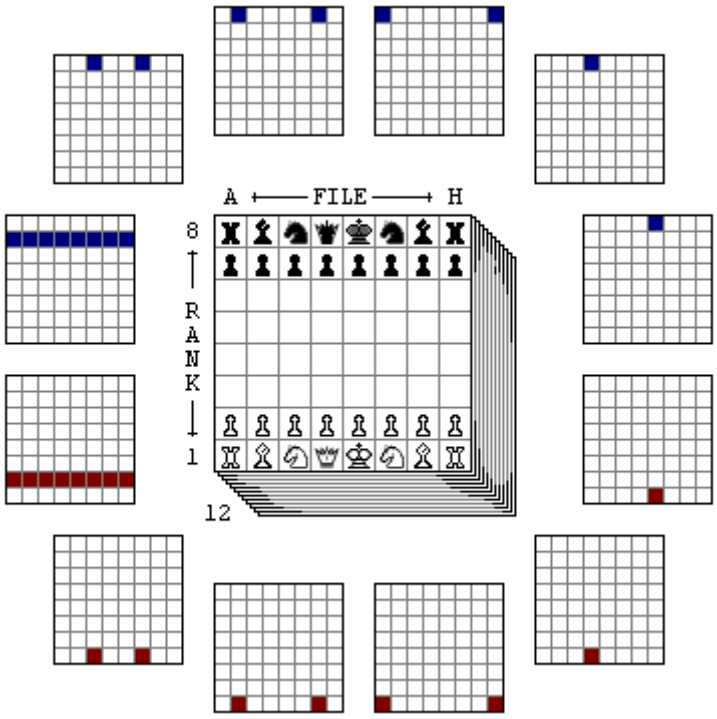
***Εικόνα 13**:Γραφικό περιβάλλον εργασίας χρήστη του Fritz (1991)*

Κεφάλαιο 3^{ο} – Λειτουργία του Stockfish και ο Αλγόριθμος minimax

3.1 Εισαγωγή

Στις πρώτες μέρες του προγραμματισμού του σκακιού, η μνήμη ήταν εξαιρετικά περιορισμένη μια τυπική σκακιέρα εφαρμοζόταν ως πίνακας 8x8, με κάθε τετράγωνο να αντιπροσωπεύεται από ένα μόνο byte. Όταν οι προγραμματιστές σκακιού άρχισαν να εργάζονται σε σταθμούς εργασίας 64-bit και mainframes, εμφανίστηκαν πιο περίτεχνες αναπαραστάσεις πίνακα βασισμένες σε "bitboards". Εφευρεμένο στη Σοβιετική Ένωση στα τέλη της δεκαετίας του 1960, ο πίνακας bit (bitboard) είναι μια λέξη 64-bit που περιέχει πληροφορίες σχετικά με μια εκδοχή της κατάστασης του παιχνιδιού. Για παράδειγμα, ένα bitboard μπορεί να περιέχει "το σύνολο των τετραγώνων που καταλαμβάνουν τα μαύρα πιόνια", ένα άλλο "το σύνολο των

τετραγώνων στα οποία μπορεί να μετακινηθεί μια βασίλισσα που βρίσκεται στο e3" και ένα άλλο "το σύνολο των λευκών πιονιών που είναι σε επίθεση επί του παρόντος από αντίπαλα πιόνια ". Οι πίνακες Bitboard είναι ευέλικτοι και επιτρέπουν γρήγορη επεξεργασία, επειδή πολλές λειτουργίες που επαναλαμβάνονται πολύ συχνά κατά τη διάρκεια ενός σκακιστικού παιχνιδιού μπορούν να εφαρμοστούν ως λογικές πράξεις 1 κύκλου σε bitboard. Στην ακόλουθη εικόνα 14 βλέπουμε ένα παράδειγμα bitboard.

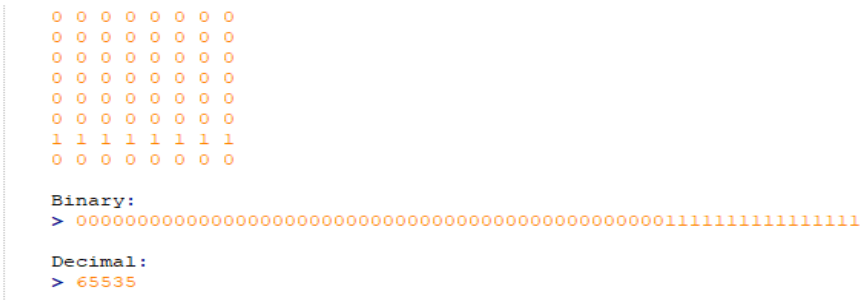


***Εικόνα 14**: Παράδειγμα bitboard*

3.1.1 Bitboards

Όπως αναφέρθηκε προηγουμένως, μια σκακιέρα αποτελείται από 64 τετράγωνα, πράγμα που σημαίνει ότι μπορεί να αποθηκεύσει τις θέσεις ενός δεδομένου κομματιού για έναν παίκτη σε μια μεταβλητή 64-bit. Κάθε bit αντιστοιχεί σε ένα τετράγωνο και αν έχει οριστεί σε 1, τότε υπάρχει ένα

κομμάτι σε αυτό το τετράγωνο. Για παράδειγμα, το παρακάτω bitboard αντιπροσωπεύει τα πιόνια του λευκού παίκτη στην αρχή ενός παιχνιδιού.



***Εικόνα 15**:Απεικόνιση πιονιών του λευκού παίκτη σε bitboard*

Στο Stockfish, οι πίνακες bitboards ξεκινούν από το τετράγωνο a1, προχωρούν κατά σειρά και οι θέσεις αποθηκεύονται σε μια μικρή ενδειακή κωδικοποίηση, όπως απεικονίζεται παραπάνω (Binary). Με αυτόν τον τρόπο, οι χειρισμοί στον πίνακα μπορούν να μεταφραστούν ως λειτουργίες δυαδικών ψηφίων:

- Μετακίνηση ενός κομματιού ένα τετράγωνο αριστερά: αριστερή μετατόπιση 1 bit,
- Μετακίνηση ενός κομματιού ένα τετράγωνο μπροστά: αριστερή μετατόπιση 8 bit,
- Ανάκτηση όλων των κομματιών ενός παίκτη: λογικό Ή όλων των bitboard του,
- Επαλήθευση εάν ένα τετράγωνο είναι κατειλημμένο: λογικό ΚΑΙ του πίνακα και μια μάσκα θέσης του τετραγώνου.

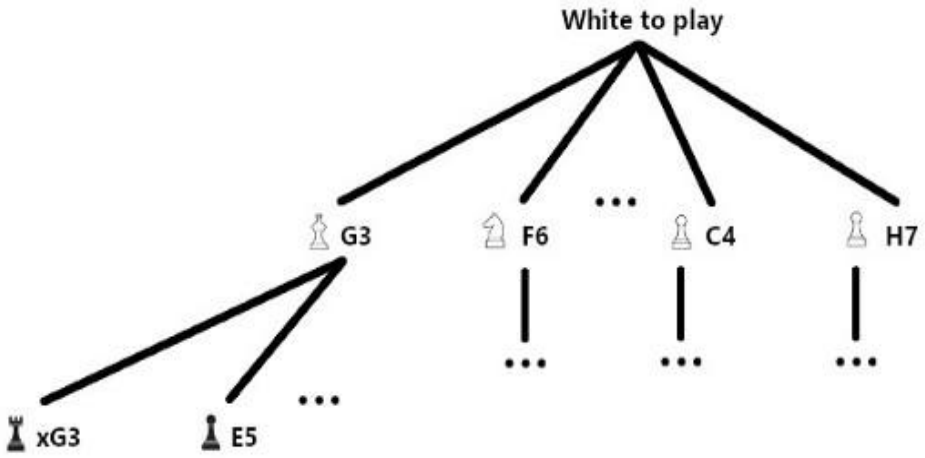
Για παράδειγμα, εδώ είναι ο κώδικας Stockfish για να επιτεθούν τα τετράγωνα από ένα bitboard πιόνων (bitboard.h):

```
template<Color C>
constexpr Bitboard pawn_attacks_bb(Bitboard b) {
  return C == WHITE ? shift<NORTH_WEST>(b) | shift<NORTH_EAST>(b)
                    : shift<SOUTH_WEST>(b) | shift<SOUTH_EAST>(b);
}
```

Με Bitboard shift <D> (Bitboard b) μια πρότυπη συνάρτηση που μετακινεί τον bitboard b σε μια δεδομένη κατεύθυνση D χρησιμοποιώντας λειτουργίες μετατόπισης bit.{30}

3.1.2 Εύρεση υποψήφιων κινήσεων

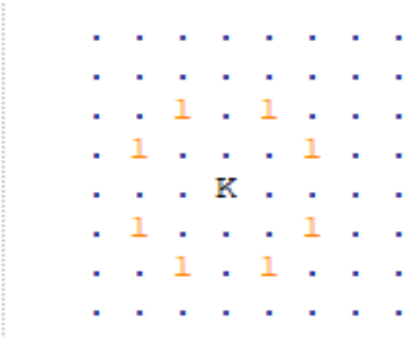
Η σκακιστική μηχανή πρέπει να βρει όλες τις υποψήφιες κινήσεις για έναν παίκτη προκειμένου να δημιουργήσει το δέντρο όλων των πιθανών κινήσεων (Εικόνα 16).{30}



***Εικόνα 16**: Υποψήφιες κινήσεις λευκού παίκτη*

3.1.3 Κομμάτια μοτίβου

Οι υποψήφιες κινήσεις των κομματιών που κινούνται σε μοτίβα (πιόνια, ιππότες και βασιλιάς) είναι ένας σταθερός αριθμός στοιχείων ενός συνόλου. Για παράδειγμα, ένας λευκός ιππότης στο D4 θα έχει πάντα τους ακόλουθους υποψηφίους:



***Εικόνα 17**:Υποψήφιες κινήσεις του λευκού ιππότη(K)*

Οι λειτουργίες μετατόπισης bit στο bitboard που περιέχουν όλους ιππότες χρησιμοποιούνται για τον υπολογισμό αυτών των υποψήφιων κινήσεων:

```
candidates  = bitboard << 15; // North-north-west
candidates |= bitboard << 17; // North-north-east
candidates |= bitboard << 6;  // North-west-west
...
candidates |= bitboard >> 15; // South-south-east
```

Οι πιθανές κινήσεις είναι ο συνδυασμός OR όλων αυτών των μετατοπίσεων. Ωστόσο, εάν ο ιππότης είναι σε πλάγια θέση ή αρχείο (Ax, Hx, x1, x8), δεν επιτρέπεται να μετακινηθεί έξω από τον πίνακα. Επομένως, η απόκρυψη του bitboard με την πλαϊνή βαθμίδα ή αρχείο θα διασφαλίσει ότι εάν ο ιππότης βρίσκεται σε τέτοια θέση, δεν θα δημιουργηθούν υποψήφιοι.

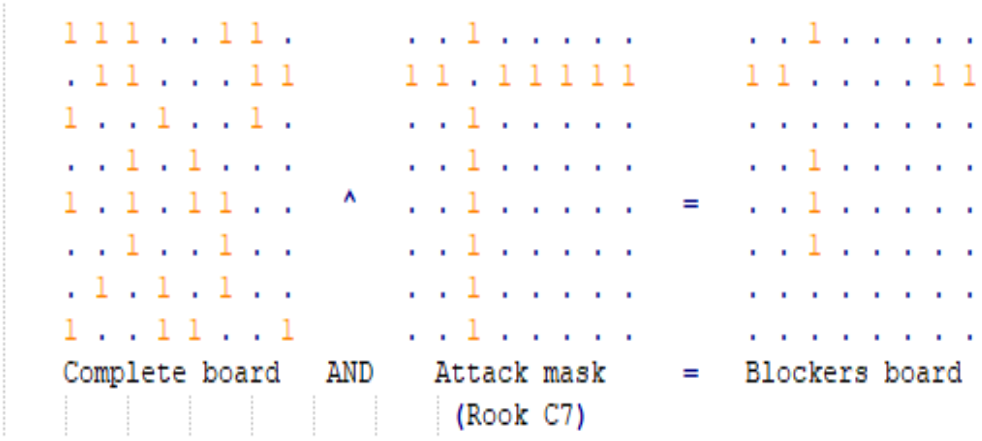
```
candidates = (bitboard & !(FileA | Rank8)) << 15; // N-N-E
...
```

Το Stockfish εκτελεί στην πραγματικότητα τους ελέγχους μετά τη δημιουργία των υποψηφίων και χρησιμοποιεί μια μέθοδο Bitboard safe_destination (Square s, int step) (bitboard.cpp) για να διασφαλίσει ότι οι κινήσεις δεν είναι εκτός πίνακα. Τέλος, υπάρχουν μερικές επιπλέον επιθεωρήσεις, όπως το μπλοκάρισμα κομματιών ή οι ανακαλυφθέντες έλεγχοι. Για το σκοπό αυτό, η Stockfish έχει μια μέθοδο bool Position :: legal (Move m) (position.cpp), η οποία ελέγχει εάν μια ψευδο-νόμιμη κίνηση είναι πράγματι νόμιμη.{30}

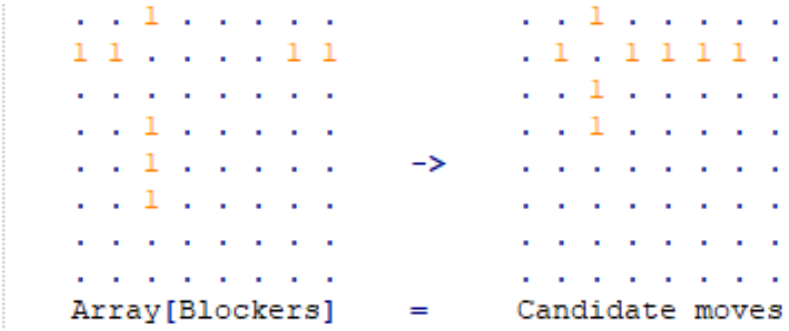
3.1.4 Συρόμενα κομμάτια

Οι πύργοι, οι αξιωματικοί και η βασίλισσα μπορούν να μετακινήσουν έναν αόριστο αριθμό τετραγώνων κατά μήκος μιας κατεύθυνσης μέχρι να φτάσουν στην άκρη του πίνακα ή έως ότου ένα άλλο κομμάτι εμποδίσει τη μετατόπιση τους. Ο υπολογισμός των υποψήφιων κινήσεών τους εν κινήσει είναι εφικτός χρησιμοποιώντας προκαθορισμένες ακτίνες. Μια επιθετική ακτίνα του κομματιού είναι AND με το bitboard όλων των κατειλημμένων τετραγώνων και μια εντολή επιπέδου επεξεργαστή που ονομάζεται bitscan χρησιμοποιείται για να βρει το πρώτο μη τετριμμένο bit του υπολογισμένου αποτελέσματος. Ωστόσο, η λειτουργία πρέπει να επαναλαμβάνεται για κάθε επιθετική ακτίνα κάθε κομματιού, η οποία οδηγεί σε μια υπολογιστικά εντατική μέθοδο για χρήση σε πραγματικό χρόνο.

Ως εκ τούτου, το Stockfish χρησιμοποιεί μια μέθοδο η οποία είναι υπολογιστικά πιο αποτελεσματική κάνοντας μια αναζήτηση σε έναν πίνακα που περιέχει τις υποψήφιες κινήσεις για τα συρόμενα κομμάτια. Για να δημιουργήσετε ένα ευρετήριο για αναζήτηση σε αυτόν τον πίνακα, απαιτείται ακόμα η εύρεση όλων των αποκλειστών για ένα συρόμενο κομμάτι:



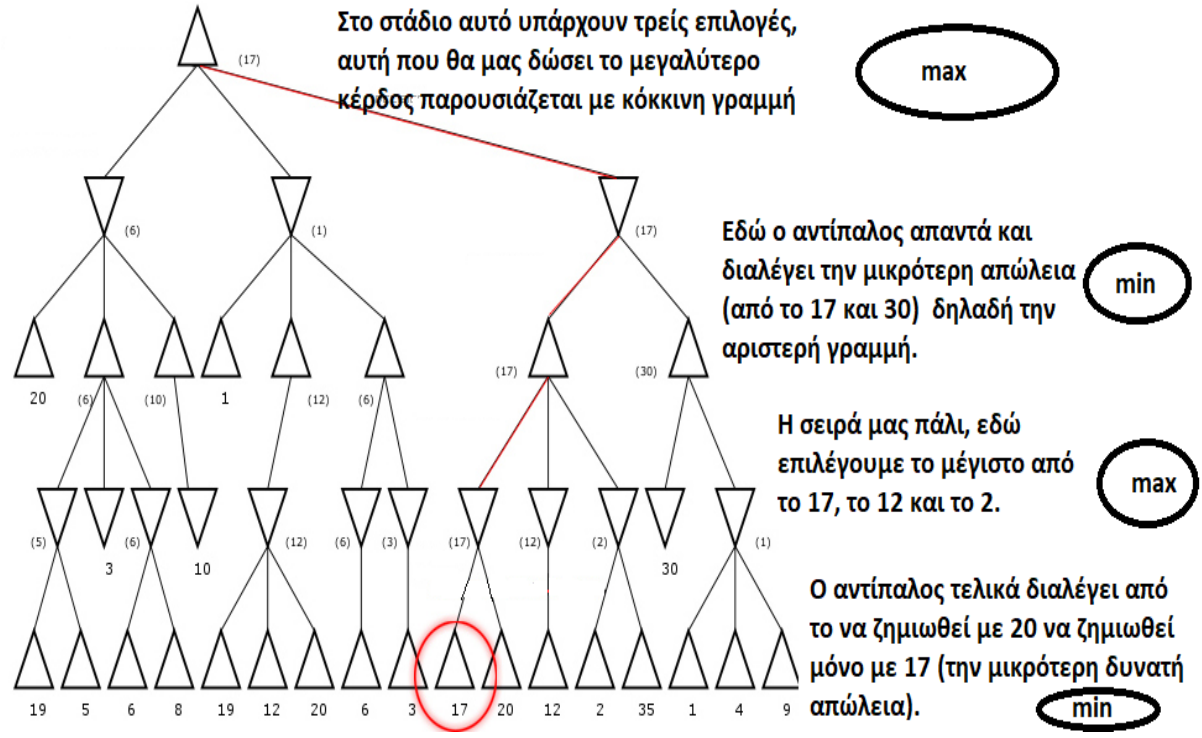
Τότε, το Stockfish θα χρησιμοποιήσει τον πίνακα αποκλειστών ως δείκτη στον πίνακα που περιέχει τους προκαθορισμένους πίνακες bitboard των υποψήφιων κινήσεων.{30}



3.2 Αλγόριθμος minimax

Για έναν υπολογιστή, δεν είναι καθόλου προφανές ποια από τις πολλές κινήσεις στο σκάκι είναι "καλή" και ποια "κακή". Ο καλύτερος τρόπος να το διακρίνει αυτό είναι να εξετάσει τις συνέπειές τους δηλαδή, να αναζητήσει σειρά κινήσεων, για παράδειγμα βάθους 4 κινήσεων για κάθε πλευρά και να δει τα αποτελέσματα. Επίσης, για να βεβαιωθεί ότι κάνει όσο το δυνατόν λιγότερα λάθη υποθέτει ότι ο αντίπαλος είναι εξίσου καλός. Αυτή είναι η γενική ιδέα όλων των σκακιστικών προγραμμάτων και βασίζεται στον αλγόριθμο αναζήτησης **minimax**, ο οποίος είναι η ρίζα όλων αυτών των προγραμμάτων που υλοποιούν το σκάκι. Το δέντρο minimax είναι απλά ένας αλγόριθμος διαλογής δέντρων που μεγιστοποιεί τις κινήσεις μας και υποθέτει ότι ο αντίπαλος θα ελαχιστοποιήσει τη δική του βαθμολογία. Με απλά λόγια το δέντρο minimax, είναι απλώς **ένα μονοπάτι**, για να λάβουμε την καλύτερη απόφαση αφού εξετάσουμε όλα τα μονοπάτια μέχρι ένα ορισμένο βάθος. Στο ακόλουθο σχήμα μπορούμε να το δούμε αυτό αναλυτικά. Αρχικά υπάρχουν τρείς επιλογές, η πιο επικερδής (max) από τις τρείς είναι η τρίτη (με κέρδος 17), φαίνεται με κόκκινη γραμμή στο σχήμα

(Εικόνα 18). Ο αντίπαλος αντίθετα με εμάς έχει πάντα κατά νου να ελαχιστοποιεί τις απώλειές του (min). Συνεπώς στο συγκεκριμένο παράδειγμα όταν θα είναι η σειρά του αντίπαλου να παίξει θα προτιμήσει κάποια κίνηση που θα του επιφέρει την μικρότερη ζημιά (min), εδώ από το 17 και το 30 προτιμάει το 17. Έπειτα οι επιλογές μας είναι ανάμεσα στο 17, το 12 και το 2. Διαλέγουμε το μέγιστο(max) που είναι το 17. Ο αντίπαλος έχοντας να επιλέξει ανάμεσα στο 17 και το 20 προτιμάει να θυσιάσει το 17 (min απώλεια).



***Εικόνα 18**:Περιγραφή του αλγορίθμου minimax*

Η πολυπλοκότητα του αλγορίθμου αυτού είναι: **O(b[sup]n[/sup])**

όπου **n** είναι το βάθος των κινήσεων, **b** είναι ο παράγοντας Διακλάδωσης (Branching Factor), δηλαδή ο αριθμός των καταστάσεων-παιδιών που προκύπτουν εκείνη την χρονική στιγμή από την επέκταση του δέντρου. Το supremum (**sup**) ενός υποσυνόλου κάποιου συνόλου είναι το μικρότερο στοιχείο που περιέχεται στο υποσύνολο.

Όπως φαίνεται από την πολυπλοκότητα του αλγορίθμου minimax όσο μεγαλύτερο είναι το βάθος (το n) τόσο αυξάνονται και οι υπολογισμοί που πρέπει να γίνουν (που δυστυχώς γίνονται στη συνέχεια όλο πιο χρονοβόροι και αργοί). Ο παράγοντας πολλαπλασιασμού των πράξεων είναι περίπου 10^3 σε κάθε επίπεδο, οπότε καταλαβαίνουμε ότι σε βάθος πολλών κινήσεων οι υπολογισμοί είναι σχεδόν αδύνατο να πραγματοποιηθούν. Αυτό το πρόβλημα το βελτίωσαν οι αλγόριθμοι Alphabeta, NegaScout και MTD(f) οι οποίοι είναι μεν πιο γρήγοροι αλλά σαφώς πιο πολύπλοκοι και βασίζονται σε τεχνικές τεχνητής νοημοσύνης (Artificial Intelligence) και σε ευρεστικές μεθόδους (Heuristics), δηλαδή καταγράφουν τις κινήσεις του αντιπάλου και δημιουργούν ένα ιστορικό κινήσεων και ένα προφίλ για το στυλ παιξίματος. Αυτές οι πληροφορίες είναι πολύτιμες και σε συνδυασμό με τον αλγόριθμο περιορίζουν σημαντικά τα πιθανά μονοπάτια. Ο Deep Blue II της IBM χρησιμοποίησε αυτές τις τεχνικές και κατάφερε να νικήσει τον Κασπάροφ στους περίφημους αγώνες που είχαν δώσει το 1996 και το 1997.

Σε αυτό το σημείο είναι απαραίτητο να αναφερθούμε και στο **"horizon effect"** που στα ελληνικά θα μπορούσαμε να το μεταφράσουμε ως **φαινόμενο ορίζοντα** και περιγράφηκε για πρώτη φορά από τον Hans Berliner. Ας δούμε ένα παράδειγμα ώστε να κατανοήσουμε την γενική ιδέα του φαινομένου αυτού. Υποθέτουμε ότι έχουμε δημιουργήσει ένα πρόγραμμα που εξετάζει τις κινήσεις σε βάθος 8 (n=8). Καθώς τρέχει ο αλγόριθμος ανακαλύπτει δυστυχώς ότι θα έχουμε μελλοντικά μια σοβαρή απώλεια και συγκεκριμένα στην έκτη κίνηση. Ο αλγόριθμος προσαρμόζει τις μελλοντικές κινήσεις με τέτοιο τρόπο ώστε σε βάθος 8 (μέχρι εκεί μπορεί να υπολογίσει) να μην υπάρξει κάποια απώλεια. Αυτό όμως δεν αποκλείει την απώλεια, απλά την μεταθέτει π.χ. στην δέκατη κίνηση. Για το πρόγραμμα όλα λειτουργούν ρολόι, στην ουσία όμως απλά μετέθεσε το πρόβλημα για αργότερα διότι δεν μπορεί να «δει» παραπέρα.

Ο Deep Blue II της IBM κατάφερε και αυτό το πρόβλημα να το αντιμετωπίσει με επιτυχία μέσο της αναζήτησης Quiescence. Ο αλγόριθμος αυτός χρησιμοποιείται συνήθως για την επέκταση της αναζήτησης σε ασταθείς κόμβους. Είναι μια επέκταση της λειτουργίας αξιολόγησης για την αναβολή της αξιολόγησης έως ότου η θέση είναι αρκετά σταθερή για να αξιολογηθεί στατικά, δηλαδή χωρίς να λαμβάνεται υπόψη το ιστορικό της θέσης ή οι μελλοντικές μετακινήσεις από τη θέση. Με απλά λόγια χρησιμοποιώντας αυτή την τεχνική μπορούμε να βλέπουμε σε μεγαλύτερο βάθος με μικρότερο υπολογιστικό και χρονικό κόστος. Αυτός ήταν και ο λόγος που κατάφερε η IBM να λυγίσει τον Κασπάροφ καθότι μπορούσε να δει αρκετές κινήσεις μπροστά σε σύγκριση με τον ανθρώπινο εγκέφαλο του διάσημου Ρώσου σκακιστή.

Παρόλο που ο Deep Blue II της IBM κατάφερε να τα συνδυάσει όλα αυτά, για έναν απλό υπολογιστή και μια απλή εφαρμογή είναι δύσκολο όλα αυτά να συνδυαστούν διότι η υπολογιστική ισχύς είναι περιορισμένη και η δυνατότητα ανάλυσης και επεξεργασίας του στυλ του παιχνιδιού του αντιπάλου είναι μια χρονοβόρα και απαιτητική διαδικασία. Επίσης, ο ανθρώπινος εγκέφαλος μπορεί να στερείται των ικανοτήτων του υπολογιστή αλλά διαθέτει κάτι που ακόμη οι υπολογιστές δεν έχουν καταφέρει να διαθέτουν και αυτό είναι η Πονηριά! Αυτό το χαρακτηριστικό που έχουν οι άνθρωποι οδήγησε τον Κασπάροφ στους περίφημους αγώνες του να «κάψει» εσκεμμένα ένα πιόνι με σκοπό να μπερδέψει τον Deep Blue II της IBM που ανέλυε τις κινήσεις του και τελικά κατάφερε να δώσει τις λάθος πληροφορίες που ήθελε για να σχηματιστεί ένα εσφαλμένο παικτικό προφίλ.{33} {34}

3.2.1 Εναλλακτικές προσεγγίσεις υλοποίησης

Η διάσχιση του δένδρου αναζήτησης στο σκάκι είναι απαραίτητη προκειμένου να εντοπιστούν πιθανές ακολουθίες κινήσεων στο παιχνίδι. Η λογική είναι να εντοπισθεί το καλύτερο μονοπάτι από την ρίζα μέχρι τέρμα κάτω στα φύλλα. Αυτός ο τρόπος διάσχισης όμως γεννά κάποια προβλήματα όπως είδαμε επειδή απαιτεί την γένεση εκθετικού αριθμού κινήσεων και θέσεων, κατά τον υπολογισμό κάθε δυνατής περίπτωσης.

Μια εναλλακτική προσέγγιση είναι με τα **τυχαία μονοπάτια**. Ξεκινώντας από τον αρχικό κόμβο, ένα τυχαίο μονοπάτι σχηματίζεται επισκέπτοντας ακολουθιακά γειτονικούς κόμβους, μέχρι να εντοπιστεί ένας τελικός κόμβος. Διασχίζοντας τον γράφο επαναληπτικά με τυχαίο τρόπο, υπολογίζουμε τις μέσες αποστάσεις μεταξύ των κόμβων έως μια τερματική να βρεθεί (ματ ή ισοπαλία). Η δημιουργία πολλών τυχαίων μονοπατιών μπορεί να μας κάνει ικανούς να μπορούμε να εκτιμήσουμε την μακροπρόθεσμη εξέλιξη του παιχνιδιού με έναν **στατιστικό μη απωλεστικό τρόπο.** Μια πρακτική χρήση θα μπορούσε να ήταν η ακόλουθη: δεδομένης μιας θέσης, εκτίμησε την απόσταση από την νίκη του μαύρου παίκτη, του άσπρου και της ισοπαλίας.

Υπάρχει πολύ μεγάλη δυναμική στα τυχαία μονοπάτια διότι η επίδραση ορίζοντα εξαφανίζεται όσο διασχίζουμε το δένδρο αναζήτησης μέχρι το τέλος του παιχνιδιού (ματ ή ισοπαλία) και όχι έως ένα συγκεκριμένο ρηχό βάθος. Επιπλέον, η χρονική πολυπλοκότητα της γένεσης τυχαίων μονοπατιών είναι **γραμμική** αναφορικά με το βάθος, σε αντίθεση με την **εκθετική πολυπλοκότητα** των άλλων μεθόδων, συνεπώς είναι πολύ καλύτερη ενώ διατηρεί την υψηλή ποιότητα εκτίμησης χωρίς περιορισμό στο βάθος.

Η γενική φιλοσοφία εδώ είναι ότι η καλύτερη κίνηση είναι αυτή που **θα είναι βελτιωτική μακροπρόθεσμα**. Το αποτέλεσμα είναι μια πολύ βαθιά στρατηγική ανάλυση της θέσης. Ένα συμπληρωματικό πλεονέκτημα είναι ότι η φύση των αλγορίθμων αυτών επιτρέπει την χρήση του **παραλληλισμού,** δηλαδή μπορούν να τρέξουν ταυτόχρονα σε πολλούς επεξεργαστές πράγμα που τους δίνει πολλαπλασιαστική ισχύ.{29} {31} {32}

Κεφάλαιο 4^{ο}- MATLAB και ανάλυση των κλάσεων της ψηφιακής σκακιέρας

Στο κεφάλαιο αυτό θα αναφερθεί η υλοποίηση της εργασίας, ξεκινώντας με μία αναφορά στο MATLAB ως προς τον τρόπο χρήσης και λειτουργίας. Στη συνέχεια, θα αναφερθούν οι κλάσεις που χρησιμοποιήθηκαν για την εκτέλεση του προγράμματος, καθώς και η περιγραφή των κλάσεων με την βοήθεια σχεδιαγραμμάτων, σημειώνοντας επίσης και τις κλάσεις που κληρονομούνται από άλλες κλάσεις.

4.1 Τι είναι το MATLAB;

Το **MATLAB** είναι ένα σύγχρονο ολοκληρωμένο μαθηματικό λογισμικό πακέτο που χρησιμοποιείται σε πανεπιστημιακά μαθήματα αλλά και ερευνητικές και άλλες εφαρμογές με επιστημονικούς υπολογισμούς. Το όνομά του προέρχεται από τα αρχικά γράμματα των λέξεων MATtrix LABoratory (εργαστήριο πινάκων). {35} {36}

4.1.1 Τρόπος χρήσης και οι δυνατότητες του MATLAB

Χρησιμοποιείται κατά κύριο λόγο για την επίλυση μαθηματικών προβλημάτων, ωστόσο είναι πολύ ισχυρό και μπορεί να χρησιμοποιηθεί και για προγραμματισμό καθώς περιέχει εντολές από την C++ όπως την while, την switch και την if. Στον τομέα των γραφικών όσον αφορά τον μαθηματικό κλάδο μπορεί να υλοποιήσει συναρτήσεις πραγματικές, μιγαδικές, πεπλεγμένες συναρτήσεις δύο μεταβλητών και άλλες.

Μάλιστα, δεν είναι υπερβολικό να θεωρηθεί ότι η συγκεκριμένη γλώσσα προγραμματισμού του MATLAB ίσως είναι καταλληλότερη και υψηλότερου επιπέδου για εφαρμογές Μηχανικού, οι οποίες απαιτούν ισχυρά υπολογιστικά εργαλεία, κάτι το οποίο είναι δύσκολο να επιτευχθεί από τις κλασσικές γλώσσες προγραμματισμού.

Παρέχει στο χρήστη ένα διαδραστικό περιβάλλον με χιλιάδες ενσωματωμένες συναρτήσεις, κατάλληλες για την υλοποίηση απαιτητικών υπολογιστικών αναλύσεων, γραφημάτων καθώς επίσης και για την παραγωγή διαφόρων animations. Επίσης, το φάσμα των εφαρμογών του συγκεκριμένου πακέτου λογισμικού διευρύνεται συνεχώς και περισσότερο, αναδεικνύοντας με αυτό τον τρόπο τις πολλαπλές δυνατότητες του, όπως:

- Υψηλή απόδοση και ταχύτητα υπολογιστικών αναλύσεων
- Δυνατότητα προσομοίωσης φυσικών συστημάτων
- Δυνατότητα υλοποίησης αλγορίθμων
- Δυνατότητα αμφίδρομης επικοινωνίας με πληθώρα άλλων προγραμμάτων και εφαρμογών
- Δυνατότητα σύνδεσης με διάφορες συσκευές καταγραφής
- Φιλικότητα προς το χρήστη και διαδραστικός χαρακτήρας

4.1.2 Περιγραφή τρόπου λειτουργίας προγράμματος

Αρχικά, ζητείται από το χρήστη να επιλέξει πιόνι που θα θελήσει να μετακινήσει. Αφού δώσει θέση προέλευσης, γίνεται έλεγχος αν στη συγκεκριμένη θέση υπάρχει πιόνι του παίκτη και, σε περίπτωση που υπάρχει, ελέγχεται αν μπορεί να μετακινηθεί το συγκεκριμένο πιόνι, π.χ. πριν ξεκινήσει το παιχνίδι, δεν μπορεί να επιλεγεί ο στρατηγός ή η βασίλισσα, γιατί έχουν άλλα πιόνια γύρω τους. Αφού επιλεγεί, λοιπόν, η θέση προέλευσης, ζητείται από το χρήστη να δώσει θέση που επιθυμεί να μετακινήσει το πιόνι που διάλεξε. Αφού δώσει θέση προορισμού, γίνεται έλεγχος αν μπορεί να πάει εκεί το συγκεκριμένο πιόνι (αν βρίσκεται μέσα στις επιτρεπτές κινήσεις του) στη θέση που επιλέχθηκε. Αν δεν είναι επιτρεπτή η κίνηση, ζητείται νέα θέση, αλλιώς πραγματοποιείται η μετακίνηση. Μετά είναι η σειρά του υπολογιστή και, αφού παίξει, έρχεται ξανά η σειρά του παίκτη. Η διαδικασία αυτή ακολουθείται μέχρι να χάσει ο παίκτης ή ο υπολογιστής το βασιλιά του.{37}

4.2- Κλάσεις του προγράμματος

Bishop.m: Η κλάση για τον χειρισμό του αξιωματικού.

BoardEditor.m: Κλάση που σχεδιάζει την σκακιέρα.

BoardGeometry.m: Κλάση που αφορά την γεωμετρία και την σχεδίαση της σκακιέρας.

BoardState.m: Κλάση που αποθηκεύει την τρέχουσα κατάσταση της σκακιέρας.

ChessClock.m: Κλάση για την δημιουργία και τον χειρισμό του ρολογιού.

ChessEngine.m: Κλάση χειρισμού και ελέγχου του Graphical User Interface (GUI).

ChessHighlight.m: Κλάση που διαχειρίζεται τον φωτισμό των τετραγώνων της σκακιέρας.

ChessMaster.m: Η βασική κλάση του προγράμματος.

ChessOptions.m: Κλάση για τις βασικές ρυθμίσεις του παιχνιδιού.

ChessPiece.m: Η βασική κλάση την οποία κληρονομούν όλα τα πιόνια.

ChessPieceData.m: Κλάση για τον χειρισμό των γραφικών δεδομένων των πιονιών.

EngineInterface.m: Κλάση που διαχειρίζεται την ασύγχρονη επικοινωνία με το πρωτόκολλο UCI.

EngineLog.m: Κλάση που επικοινωνεί με τα αντικείμενα της κλάσης ChessEngine.

EngineOptions.m: Κλάση για τον χειρισμό επιλογών του Engine.

FigureManager.m: Κλάση για τον χειρισμό των φιγούρων του GUI.

GameAnalyzer.m: Κλάση για την ανάλυση του παιχνιδιού.

HelpWindow.m: Κλάση για την βοήθεια.

King.m: Κλάση για τις κινήσεις του βασιλιά.

Knight.m: Κλάση για τις κινήσεις που κάνει το αλογάκι.

Move.m: Κλάση για την δημιουργία αντικειμένων που περιγράφουν την κίνηση.

MoveList.m: Κλάση για τον χειρισμό των διαθέσιμων κινήσεων.

MutableList.m: Κλάση χειρισμού στοιχείων της λίστας κινήσεων.

OptionsWindow.m: Υπερκλάση κληρονομούμενη απ' όλες τις <Class>Options κλάσεις.

Pawn.m: Κλάση για το στρατιωτάκι και τον χειρισμό του.

Queen.m: Κλάση χειρισμού των κινήσεων της βασίλισσας.

Rook.m: Κλάση χειρισμού του πύργου.

ThemeEditor.m: Κλάση για την δημιουργία color themes.

4.2.1 Περιγραφή κλάσεων

<Αξιωματικός>

Bishop.m

Περιλαμβάνει την συνάρτηση ValidMoves η οποία ελέγχει την ορθότητα των κινήσεων του αξιωματικού. Θυμίζουμε ότι ο αξιωματικός κινείται διαγωνίως όσα τετράγωνα θέλει μπροστά ή πίσω. Ένας επιπλέον περιορισμός είναι ότι δεν μπορεί να κινηθεί σε τετράγωνο που βρίσκεται κομμάτι διαφορετικού χρώματος ή αν υπάρχει πιόνι ίδιου χρώματος με αυτό. Όλα αυτά ελέγχονται μέσα από την συνάρτηση.

<Σχεδίαση σκακιέρας>

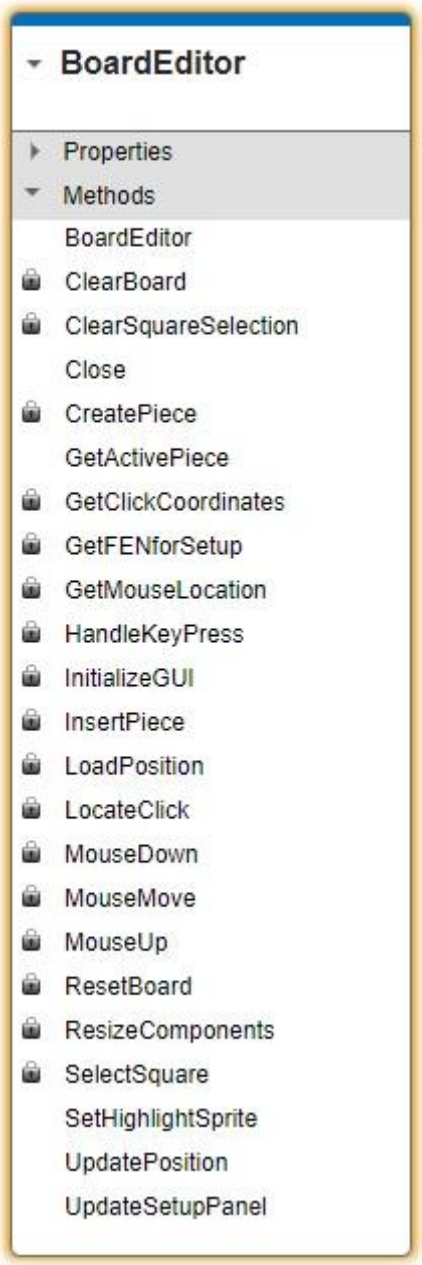
BoardEditor.m

Η συνάρτηση MouseMove παρακολουθεί την κίνηση του ποντικιού από την στιγμή που μαρκάρουμε ένα πιόνι της σκακιέρας μέχρι το σημείο που θέλουμε να το τοποθετήσουμε. Η συνάρτηση SelectSquare ενεργοποιείται μόλις γίνει κλικ επάνω σε κάποιο τετραγωνάκι της σκακιέρας και το χρωματίζει λίγο διαφορετικά για να ξεχωρίζει. Μετά την τοποθέτηση του πιονιού στην Highlighted θέση καλείται η συνάρτηση ClearSquareSelection και ο αρχικός χρωματισμός επανέρχεται.

Η συνάρτηση LoadPosition φορτώνει κάθε φορά την νέα θέση.

Η συνάρτηση CreatePiece δημιουργεί ένα-ένα όλα τα στοιχεία της σκακιέρας (Βασιλιά, αλογάκι, πύργο κτλ). Η συνάρτηση InsertPiece εισάγει όλα αυτά τα στοιχεία στην σκακιέρα.

Η συνάρτηση InitializeGUI είναι για τον καθορισμό της γραμματοσειράς, του φόντου της σκακιέρας και τις διαστάσεις της.

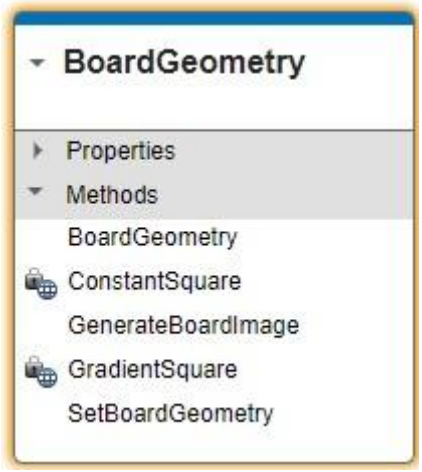


<Γεωμετρική σχεδίαση σκακιέρας>

BoardGeometry.m

Η κλάση αυτή είναι πολύ σημαντική και περιλαμβάνει τις διαστάσεις των βασικών δομικών στοιχείων όπως: μέγεθος τετραγώνου, διαστάσεις σκακιέρας, όρια σκακιέρας, μέγιστη και ελάχιστη διάσταση, μέγεθος γραμμής.

Περιλαμβάνει έναν Κατασκευαστή (Constructor) για την αρχικοποίηση όλων αυτών των χαρακτηριστικών όταν δημιουργείται ένα αντικείμενο.



<Αποθήκευση τρέχουσας κίνησης>

BoardState.m

Είναι η κλάση που αποθηκεύει την τρέχουσα κατάσταση της σκακιέρας. Διαμοιράζεται με τα αντικείμενα των άλλων κλάσεων την τρέχουσα κατάσταση της σκακιέρας. Περιέχει τις συναρτήσεις AddPiece και RemovePiece, η πρώτη προσθέτει ένα στοιχείο στην σκακιέρα και η δεύτερη το αφαιρεί. Επίσης, τσεκάρει αν η τρέχουσα κατάσταση είναι Σαχ (Check) με την GetCheckStatus και την IsInCheck. Αντίστοιχα η GetMateStatus και η IsInMate ελέγχει αν είμαστε σε κατάσταση Ματ (Mate) οπότε και το παιχνίδι θα λάβει τέλος.

Η συνάρτηση AnyLegalMoves ελέγχει τις έγκυρες θέσεις που μπορεί να κινηθεί ένα πιόνι.

Η συνάρτηση IsValidCastle τσεκάρει αν η επόμενη κίνηση είναι μια έγκυρη κίνηση του Πύργου.

Το εν διελεύσει είναι ένας κανόνας σχετικός με την κίνηση και την αιχμαλώτιση των πιονιών στο σκάκι. Ο κανόνας εφαρμόζεται όταν ένα πιόνι χρησιμοποιήσει το δικαίωμα να κινηθεί δυο τετράγωνα μπροστά αντί για ένα. Η συνάρτηση IsValidEnPassant κάνει αυτόν τον έλεγχο.

Η συνάρτηση RecordMove κάνει καταγραφή τις κινήσεις. Οι συναρτήσεις UndoMove και RedoMove μας δίνουν την δυνατότητα να πάμε μπρος και πίσω μια κίνηση.

Η GetRandomMove μας δίνει μια τυχαία και έγκυρη κίνηση.

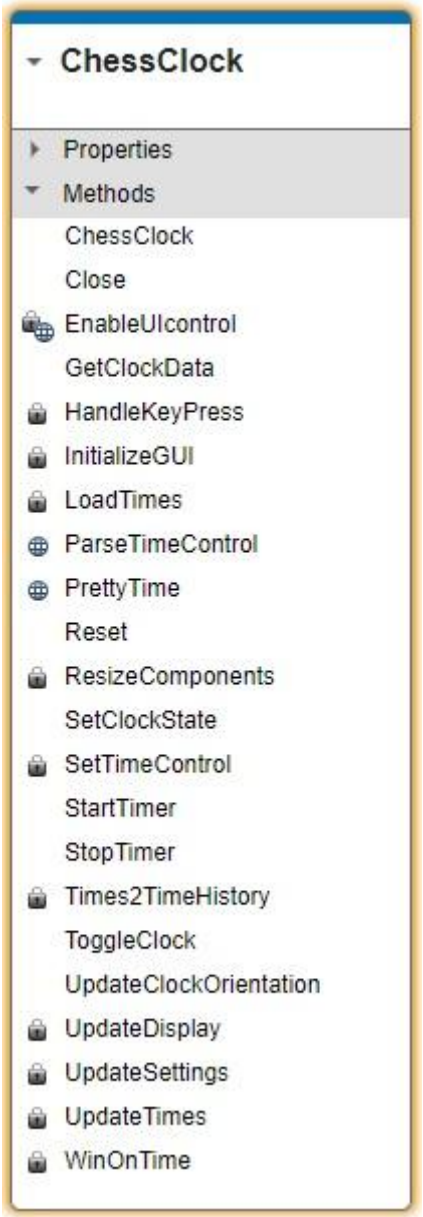
Ο κανόνας της τριπλής επανάληψης θέσης αναφέρει ότι ένας παίκτης μπορεί να διεκδικήσει μια ισοπαλία, αν η ίδια θέση εμφανιστεί τρεις φορές στην παρτίδα, ή πρόκειται να προκύψει μετά την επόμενη κίνησή τους, με τον ίδιο παίκτη να έχει την κίνηση. Η συνάρτηση Is3FoldRep κάνει αυτόν τον έλεγχο. Για να τσεκάρει την περίπτωση αυτή γίνεται κλήση της συνάρτησης MoveList που περιέχει όλο το ιστορικό κινήσεων. Ένας μετρητής υπολογίζει το άθροισμα και όταν αυτό φθάσει στο 3 τότε η Is3FoldRep που είναι μια συνάρτηση με τύπο επιστροφής Boole επιστρέφει την τιμή true.


BoardState
Properties
Methods
AddPiece
AddRightsToState
AnyLegalMoves
Attackers
BaseEncoding
BoardState
ClearEditList
ColorAt
FilterCastlingRights
FilterEnPassantTarget
GenerateFEN
GetCheckStatus
GetCurrentEncoding
GetCurrentFENstr
GetMateStatus
GetRandomMove
GetReversibleMoves
Is3FoldRep
IsBlack
IsEmpty
IsInCheck
IsInMate
IsUnderAttack
IsValidCastle
IsValidEnPassant
IsValidPromotion
IsWhite
KingOfColor
MovePiece
ParseFEN
PieceAt
RecordEdit
RecordMove
RedoMove
RefreshPieces
RemovePiece
Reset
SufficientMatingMaterial
UndoMove
UpdateChecks

<Δημιουργία και χειρισμός ρολογιού>

ChessClock.m

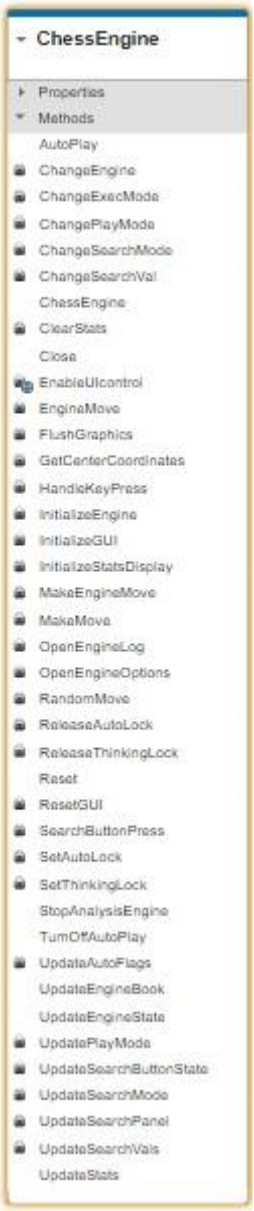
Είναι η κλάση για την δημιουργία και τον χειρισμό του ρολογιού. Η κλάση αυτή εκκινεί τα χρονόμετρα (ένα για τα λευκά και ένα ακόμη για τα μαύρα), μας δίνει την δυνατότητα να τα κάνουμε παύση και επανεκκίνηση. Τέλος, γίνεται και καταγραφή του ιστορικού των χρόνων.



<Χειρισμός και έλεγχος του GUI>

ChessEngine.m

Κλάση χειρισμού και ελέγχου του Graphical User Interface (GUI) ώστε να συντονίζεται με μια σκακιστική μηχανή (Chess Engine). Η σκακιστική μηχανή είναι κώδικας που αναλύει τις θέσεις παραλλαγής στο σκάκι και δημιουργεί μια κίνηση ή μια λίστα κινήσεων που θεωρεί ως ισχυρότερη μέσω αλγορίθμων τεχνητής νοημοσύνης. Η κλάση αυτή αποτελεί την γέφυρα επικοινωνίας του προγράμματος με την σκακιστική μηχανή. Καθορίζει τον τρόπο επικοινωνίας και χειρισμού του GUI από την μηχανή.

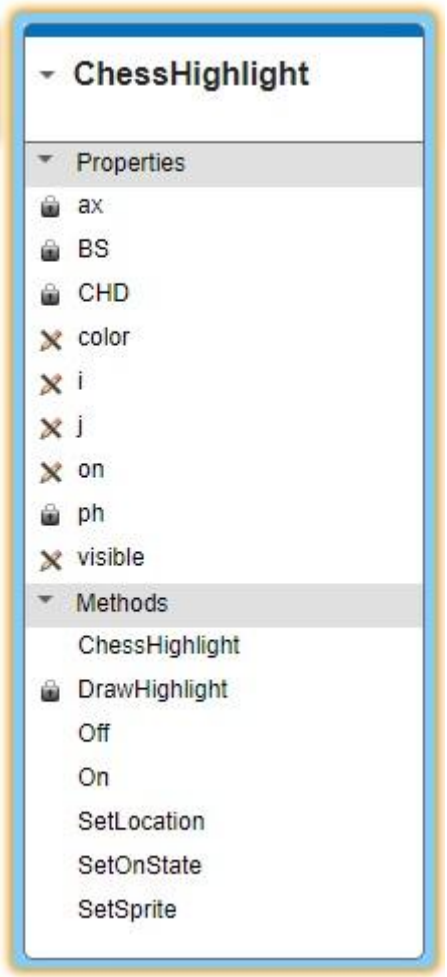


<Διαχωρισμός φωτισμού τετραγώνων>

ChessHighlight.m

Είναι η κλάση που διαχειρίζεται τον φωτισμό των τετραγώνων της σκακιέρας. Αυτό που ίσως κάνει το σκάκι στον υπολογιστή πιο ελκυστικό είναι και η δυνατότητα χρωματισμού των επιτρεπτών κινήσεων. Τουλάχιστον για αρχάριους χρήστες αυτό είναι πολύ χρήσιμο.

Οι συναρτήσεις (SetOnState, SetSprite, On Off) που περιλαμβάνει η κλάση αυτή καθορίζουν την έναρξη και τον τερματισμό του χρωματισμού.



<Βασική κλάση>

ChessMaster.m

Η βασική κλάση του προγράμματος. Η έξοδος του (output) είναι ένα αντικείμενο CM με δημόσιες μεθόδους τις ακόλουθες:

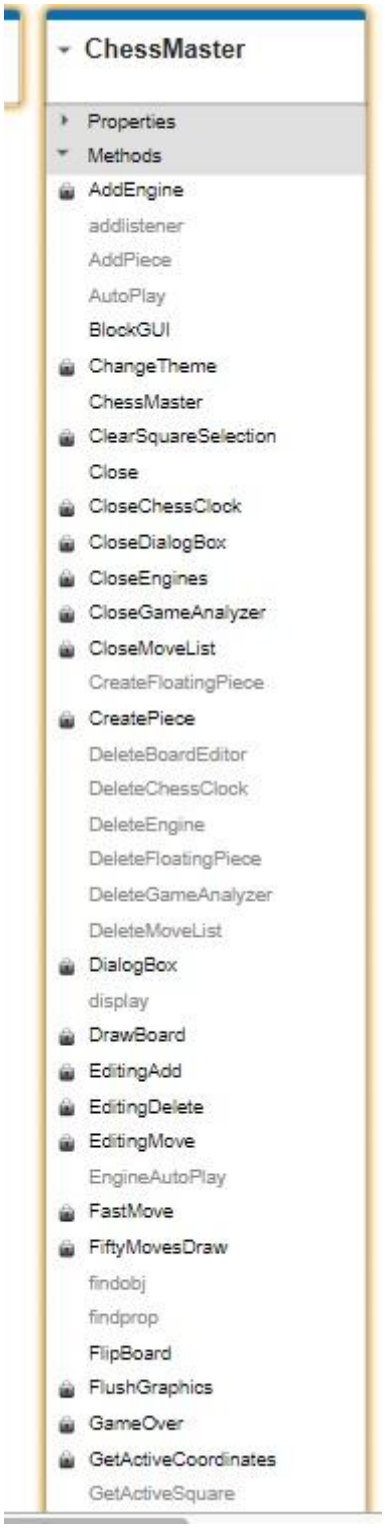
Outputs:
SANstr = CM.MakeMove(LANstr);
SANstr = CM.RandomMove();
CM.Undo();
CM.UndoAll();

CM.Redo();
CM.RedoAll();
CM.GoToHalfmove(n);
success = CM.LoadPosition(FENstr);
FENstr = CM.GetFENstr();
LANstrs = CM.GetLANstrs();
SANstrs = CM.GetSANstrs();
CM.FlipBoard();
CM.RefreshBoard();
CM.BlockGUI(bool);
CM.LoadGame(path,start);
CM.SaveGame(path);
CM.ResetBoard();
CM.Close();

Η κεντρική διαχείριση του προγράμματος γίνεται από εδώ. Η έναρξη μιας παρτίδας, ο τερματισμός, η διαμόρφωση του παιχνιδιού, τα χαρακτηριστικά του περιβάλλοντος, ο χρόνος, το ιστορικό και οι κινήσεις. Ανάλογα με το τι από όλα αυτά θέλουμε να κάνουμε γίνεται και η επικοινωνία με τις αντίστοιχες κλάσεις.

Εδώ έχουμε την συνάρτηση LoadGame για την φόρτωση ενός παιχνιδιού που είναι αποθηκευμένο, την SaveGame για την αποθήκευση ενός παιχνιδιού, την RefreshBoard για την αρχικοποίηση της σκακιέρας, τις SelectSquare και ClearSquareSelection για την επιλογή και αποεπιλογή ενός τετραγωνιδίου της σκακιέρας, την endOfGame για τον τερματισμό του παιχνιδιού, την SaveMove για την αποθήκευση της τελευταίας κίνησης και την OfferDraw για την ισοπαλία. Για την τελευταία συνάρτηση να σημειώσουμε ότι στο σκάκι για να υπάρξει ισοπαλία θα πρέπει να γίνει αποδοχή και από τους δύο παίκτες οπότε έπρεπε να ληφθεί και αυτό στα υπόψιν για την υλοποίηση που έγινε με question dialog (κώδικας selection = questdlg([color ' offers a draw. Accept?'], ... this.version.name,'Yes','No','Yes');

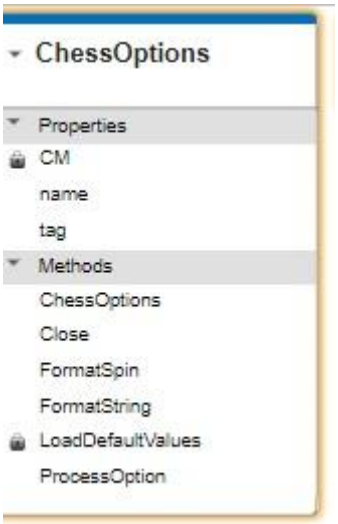
Τέλος η συνάρτηση Resign δίνει την δυνατότητα σ’ έναν παίκτη να εγκαταλείψει το παιχνίδι και να το τερματίσει, τότε καλείτε και η συνάρτηση GameOver.



<Ρυθμίσεις παιχνιδιού>

ChessOptions.m

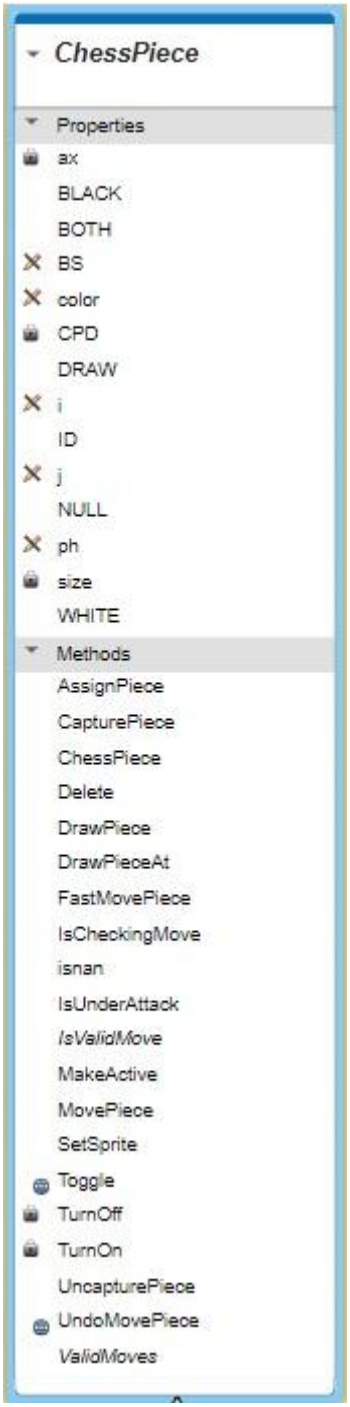
Είναι η κλάση για τις βασικές ρυθμίσεις του παιχνιδιού. Περιλαμβάνει την συνάρτηση ProcessOption όπου καθορίζονται διάφορες ρυθμίσεις του παιχνιδιού όπως αν θα υπάρξει χρονόμετρο ή όχι, αν θα επιτρέπονται αναλυτές για το παιχνίδι κτλ.



<Κληρονόμηση των πιονιών>

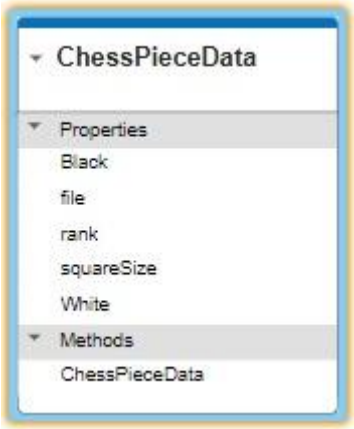
ChessPiece.m

Η βασική κλάση την οποία κληρονομούν όλα τα πιόνια της σκακιέρας. Περιλαμβάνει έναν κατασκευαστή ο οποίος αρχικοποιεί τα πιόνια (χρώμα, θέση και κατάσταση) και τις συναρτήσεις DrawPiece, DrawPieceAt και MakeActive που ενεργοποιούν ένα στοιχείο στην σκακιέρα. Τέλος, γίνεται έλεγχος μέσω της συνάρτησης IsUnderAttack αν κάποιο στοιχείο απ' αυτά είναι σε κίνδυνο.


ChessPiece
Properties
ax
BLACK
BOTH
BS
color
CPD
DRAW
i
ID
j
NULL
ph
size
WHITE
Methods
AssignPiece
CapturePiece
ChessPiece
Delete
DrawPiece
DrawPieceAt
FastMovePiece
IsCheckingMove
isnan
IsUnderAttack
IsValidMove
MakeActive
MovePiece
SetSprite
Toggle
TurnOff
TurnOn
UncapturePiece
UndoMovePiece
ValidMoves

< Χειρισμός των γραφικών δεδομένων των πιονιών>

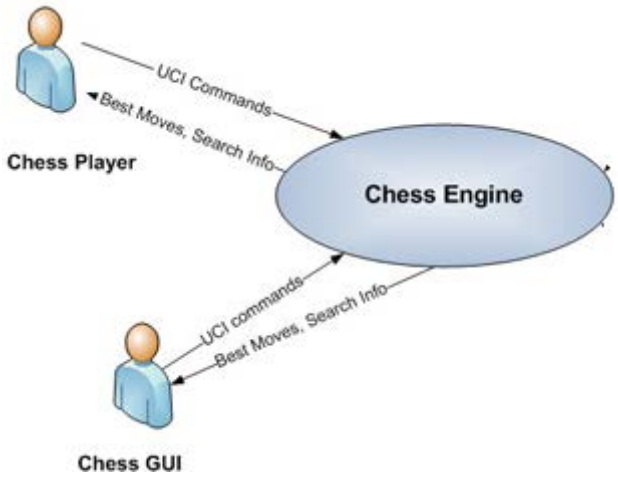
ChessPieceData.m



<Διαχείριση ασύγχρονης επικοινωνίας με το UCI>

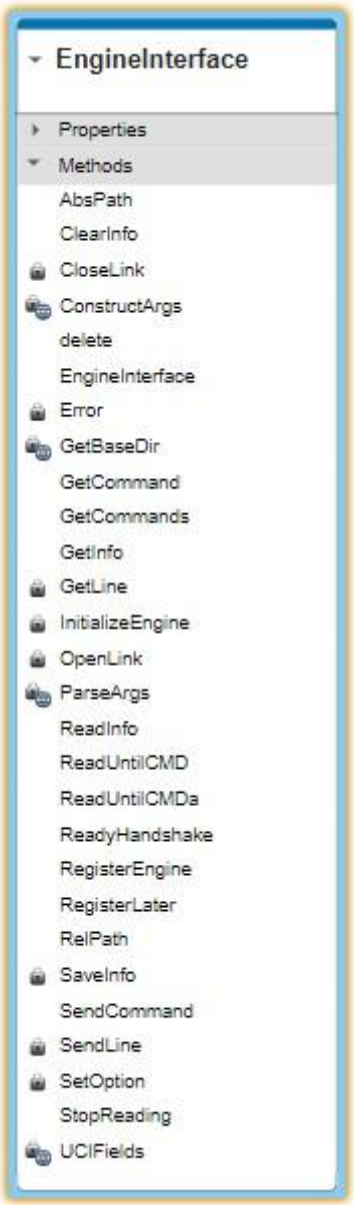
EngineInterface.m

Κλάση που διαχειρίζεται την ασύγχρονη επικοινωνία με το πρωτόκολλο UCI. Να θυμίσουμε σε αυτό το σημείο ότι το UCI (Universal Chess Interface) είναι ένα ανοιχτό πρωτόκολλο επικοινωνίας που επιτρέπει στις μηχανές σκακιού να επικοινωνούν με διεπαφές χρήστη (όπως φαίνεται στην ακόλουθη Εικόνα 19).



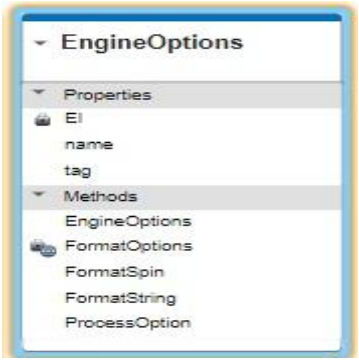
Εικόνα (19): το UCI (Universal Chess Interface)

Για να γίνει αυτό πρέπει να ανοίξει ένας δίαυλος επικοινωνίας. Οι συναρτήσεις SendCommand, ReadUntilCMD και ReadyHandshake διαχειρίζονται αυτή την επικοινωνία σύμφωνα με το UCI πρωτόκολλο.



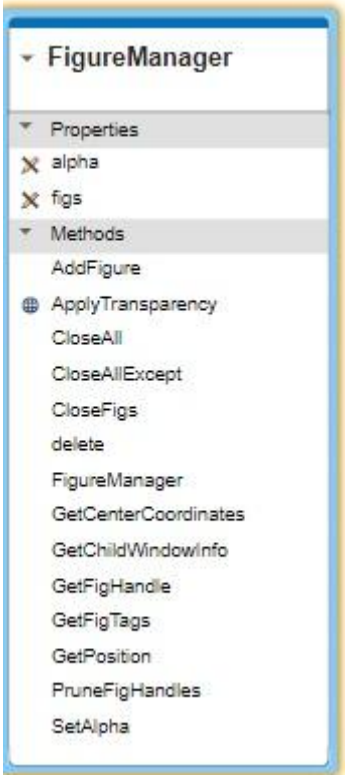
<Χειρισμός επιλογών του Engine>

EngineOptions.m



<Χειρισμός στοιχείων του GML>

FigureManager.m



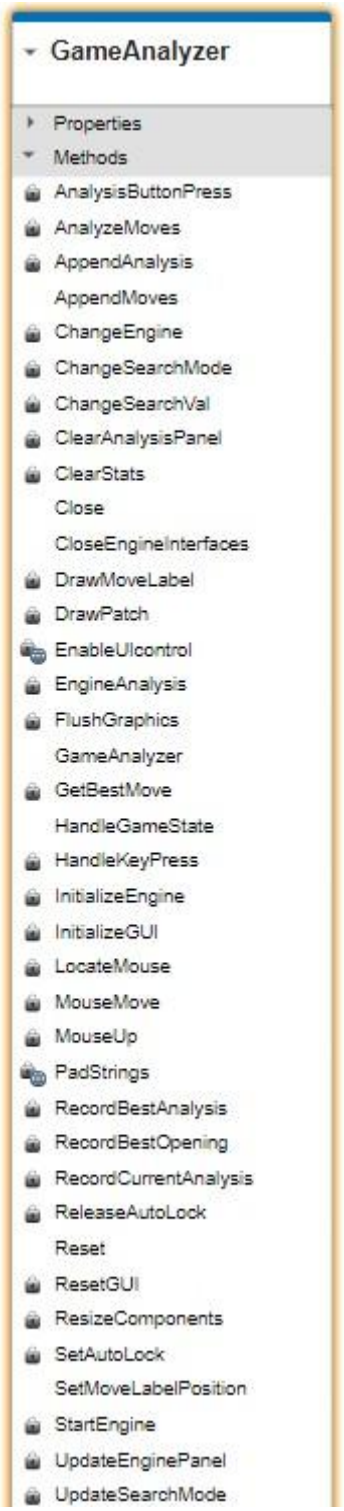
<Ανάλυση του παιχνιδιού>

GameAnalyzer.m

Κλάση για την ανάλυση του παιχνιδιού. Η ανάλυση ενός παιχνιδιού μπορεί να γίνει σε βάθος κάποιων κινήσεων. Προφανώς όσο μεγαλύτερο είναι το βάθος αυτό τόσο και αυξάνεται η επεξεργαστική ισχύς που απαιτείται. Στο πρόγραμμα ορίζεται ότι η χρήση της CPU είναι στο MAX.

Η ανάλυση εδώ γίνεται με την βοήθεια των engines. Οι συναρτήσεις StartEngine και EngineAnalysis ανοίγουν έναν δίαυλο επικοινωνίας με την μηχανή ώστε να έχουμε την καλύτερη δυνατή επιλογή. Κάθε κίνηση καταγράφετε και αναλύεται, στο User Interface υπάρχει και διαγραμματική απεικόνιση της τρέχουσας κατάστασης με στατιστικά στοιχεία.

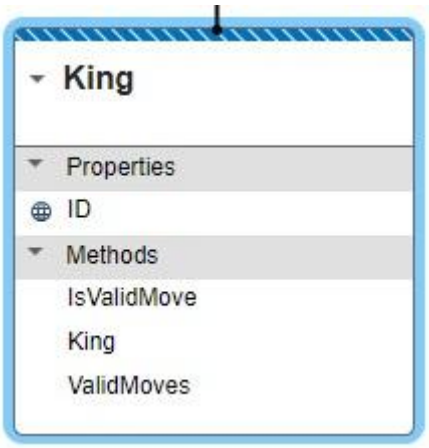
Εδώ θα πρέπει να θυμίσουμε ότι στο σκάκι υπάρχει ένα σύστημα βαθμολόγησης για κάθε κίνηση. Μια λάθος κίνηση μπορεί να μας στοιχίσει μελλοντικά οπότε παίρνουμε μια αρνητική βαθμολογία. Στο πρόγραμμα υπάρχουν τρία επίπεδα «λάθους», 1.το απλό λάθος, 2. Το σοβαρό λάθος και 3. Το μοιραίο λάθος. Η συνάρτηση AppendAnalysis κάνει αυτή την κατηγοριοποίηση, ενώ η UpdateStats ενημερώνει ανάλογα τα στατιστικά.


GameAnalyzer
Properties
Methods
AnalysisButtonPress
AnalyzeMoves
AppendAnalysis
AppendMoves
ChangeEngine
ChangeSearchMode
ChangeSearchVal
ClearAnalysisPanel
ClearStats
Close
CloseEngineInterfaces
DrawMoveLabel
DrawPatch
EnableUIcontrol
EngineAnalysis
FlushGraphics
GameAnalyzer
GetBestMove
HandleGameState
HandleKeyPress
InitializeEngine
InitializeGUI
LocateMouse
MouseMove
MouseUp
PadStrings
RecordBestAnalysis
RecordBestOpening
RecordCurrentAnalysis
ReleaseAutoLock
Reset
ResetGUI
ResizeComponents
SetAutoLock
SetMoveLabelPosition
StartEngine
UpdateEnginePanel
UpdateSearchMode

<Κινήσεις βασιλιά>

King.m

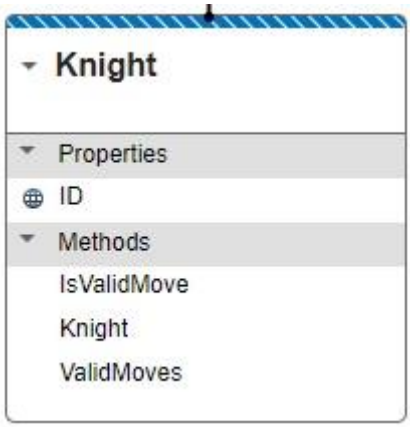
Κλάση για τις κινήσεις του βασιλιά. Η συνάρτηση ValidMoves εξετάζει σε ποια θέση είναι δυνατή η μετακίνηση του βασιλιά.



<Κινήσεις αλόγου>

Knight.m

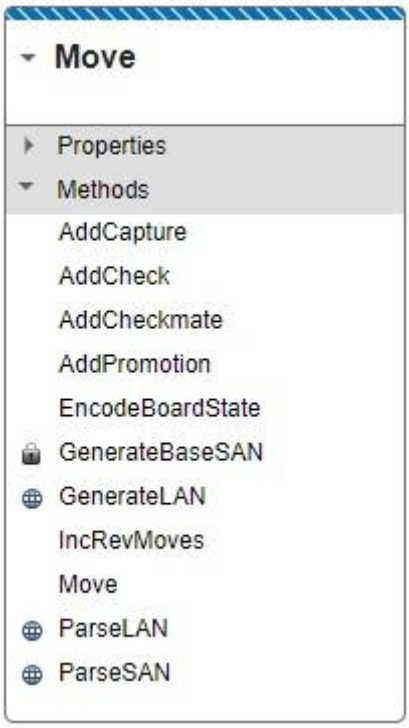
Το αλογάκι έχει μια ευρεία γκάμα από κινήσεις. Η κλάση αυτή καθορίζει όλες αυτές τις κινήσεις μέσω της συνάρτησης IsValidMove.



<Αντικείμενα περιγραφής κίνησης>

Move.m

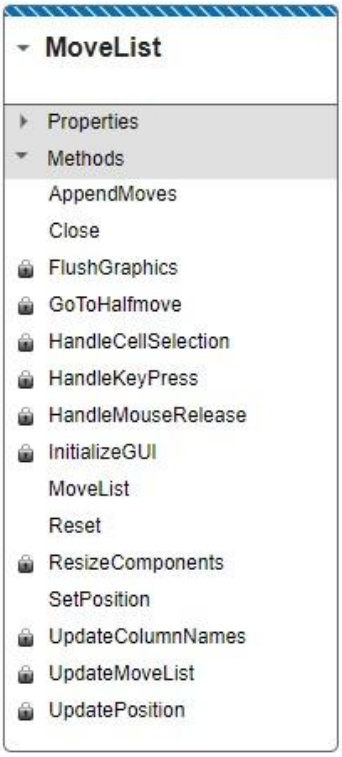
Κλάση για την δημιουργία αντικειμένων που περιγράφουν την κίνηση. Η κάθε κίνηση εξετάζεται για τσεκ ή ματ.



<Χειρισμός διαθέσιμων κινήσεων>

MoveList.m

Κλάση για τον χειρισμό των διαθέσιμων κινήσεων.



OptionsWindow.m

Υπερκλάση κληρονομούμενη απ’όλες τις <Class> Options κλάσεις.

<Κινήσεις του πιονιού>

Pawn.m

Κλάση για το στρατιωτάκι και τον χειρισμό του. Περιέχει την συνάρτηση ValidMoves για τον εντοπισμό έγκυρων θέσεων στην σκακιέρα. Εδώ γίνεται και ο έλεγχος αν μπορεί το στρατιωτάκι να κάνει δύο βήματα (βασικός κανόνας στο σκάκι που ισχύει μόνο μια φορά στην αρχή).

<Κινήσεις της Βασίλισσας>

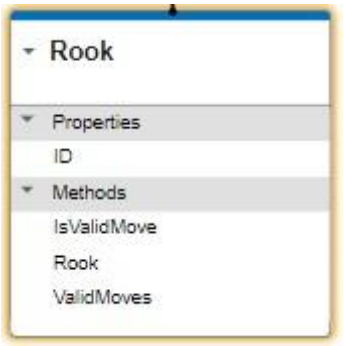
Queen.m

Κλάση χειρισμού των κινήσεων της Βασίλισσας. Η συνάρτηση IsValidMove ελέγχει την εγκυρότητα των κινήσεων. Η βασίλισσα έχει μια ευρεία γκάμα κινήσεων (οριζόντια, κάθετα, πίσω, μπροστά) και μπορεί να κινείται στα λευκά αλλά και στα μαύρα τετραγωνάκια. Αυτή η πληθώρα κινήσεων αυξάνει και τους ελέγχους που πρέπει να γίνουν στην σκακιέρα. Η συνάρτηση ελέγχει μία-μία την κάθε κίνηση προς όλες τις κατευθύνσεις.

<Κινήσεις πύργου>

Rook.m

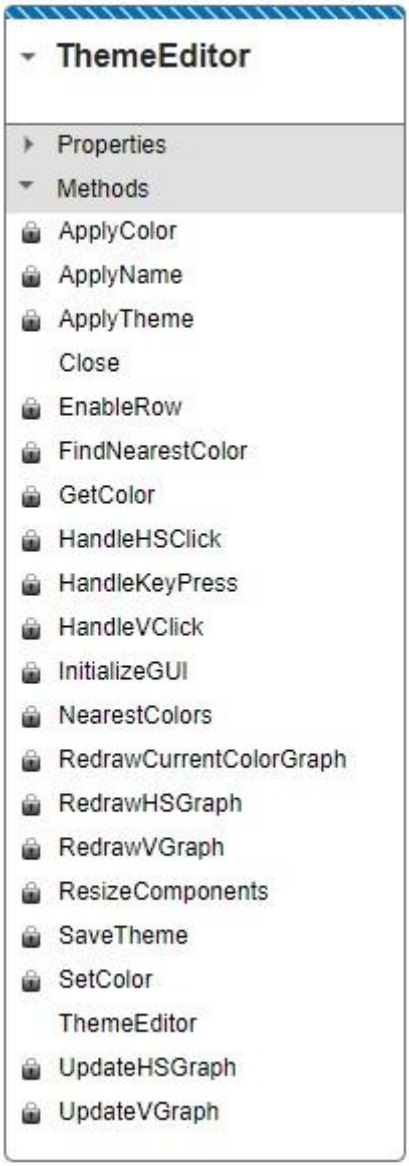
Είναι η κλάση χειρισμού του πύργου. Η συνάρτηση IsValidMove ελέγχει την εγκυρότητα των κινήσεων οι οποίες θα πρέπει να είναι κάθετες και οριζόντιες με κατεύθυνση μπροστά ή πίσω.



<Χρώμα θεμάτων>

ThemeEditor.m

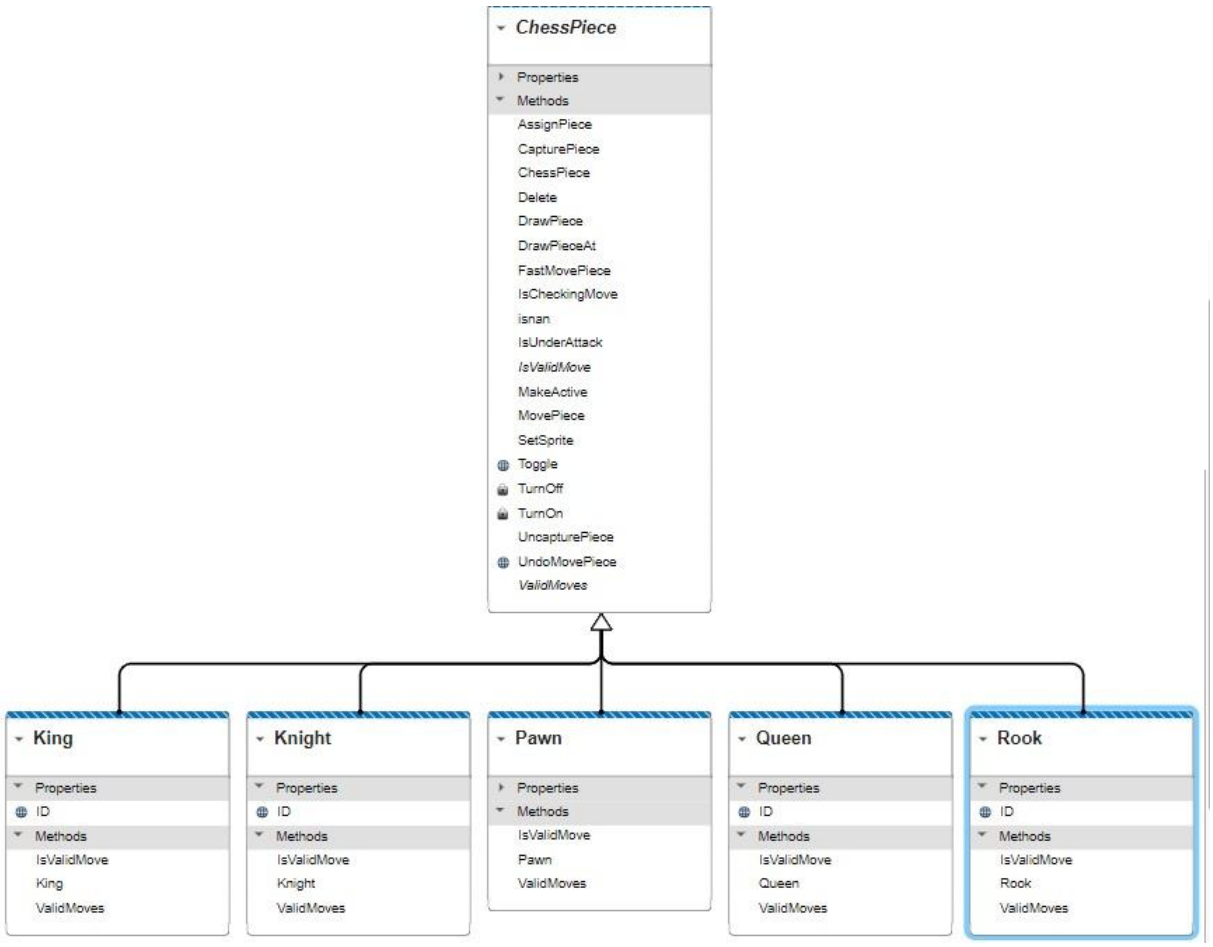
Κλάση για την δημιουργία color themes. Οι συναρτήσεις ApplyColor και RedrawVGraph σχεδιάζουν και χρωματίζουν το διάγραμμα.


ThemeEditor
Properties
Methods
ApplyColor
ApplyName
ApplyTheme
Close
EnableRow
FindNearestColor
GetColor
HandleHSClick
HandleKeyPress
HandleVClick
InitializeGUI
NearestColors
RedrawCurrentColorGraph
RedrawHSGraph
RedrawVGraph
ResizeComponents
SaveTheme
SetColor
ThemeEditor
UpdateHSGraph
UpdateVGraph

4.2.2 Κληρονόμηση κλάσεων

Οι κλάσεις ChessOptions και EngineOptions κληρονομούνται από την κλάση OptionsWindow.

Τα πιόνια (βασιλιάς, βασίλισσα, πύργος κτλ.) κληρονομούνται από την βασική κλάση ChessPiece.



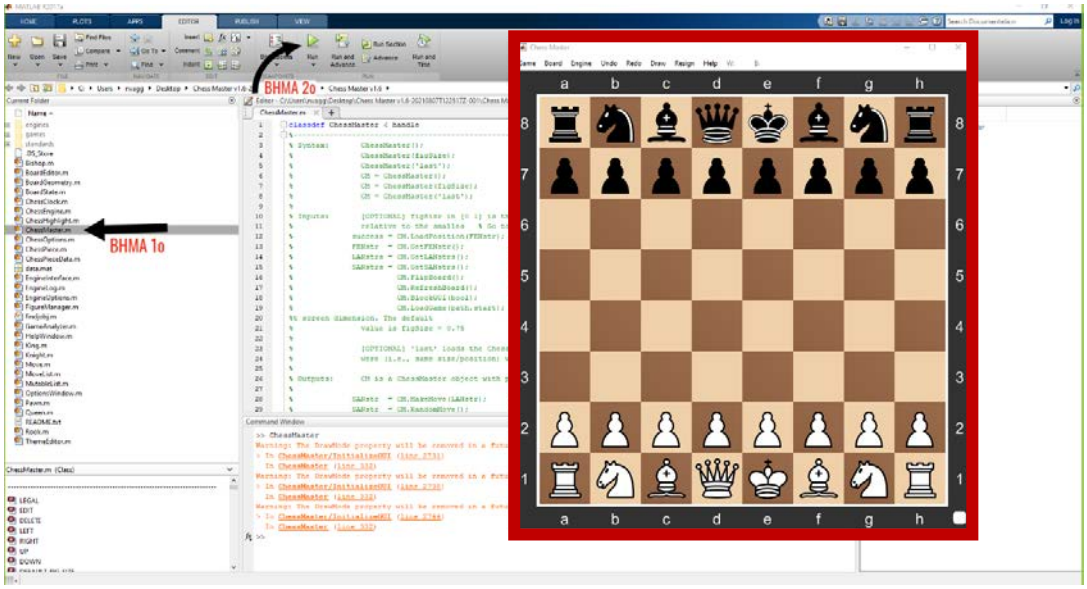
Κεφάλαιο 5^{o} - Εκτέλεση προγράμματος

5.1 Απαιτήσεις συστήματος

Η υλοποίηση του προγράμματος πραγματοποιήθηκε σε λειτουργικό περιβάλλον Windows 10. Επίσης, το πρόγραμμα δημιουργήθηκε με την χρήση του Matlab, με την προϋπόθεση ότι με αυτό το λειτουργικό σύστημα μπορεί να εκτελεστεί σε υπολογιστή, καλύπτοντας τις απαιτήσεις του προγράμματος. Η έκδοση που χρησιμοποιήθηκε είναι η MATLAB 9.2 R2017a.

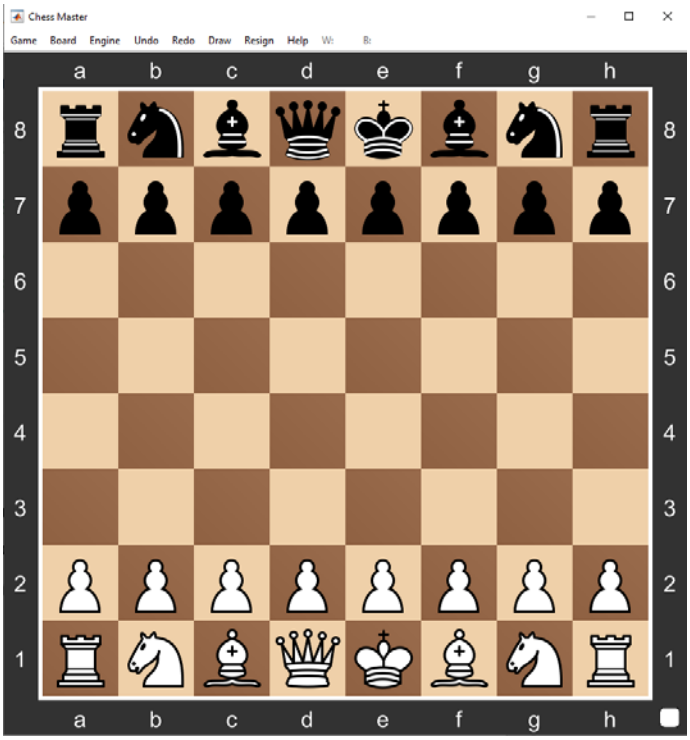
5.2 Εκτέλεση και περιγραφή της σκακιέρας

Καταρχάς, για να τρέξουμε την ψηφιακή σκακιέρα θα πρέπει να εκτελέσουμε την κλάση ChessMaster.m στο MATLAB για να εμφανισθεί στην οθόνη του υπολογιστή. Η παρακάτω εικόνα (Εικόνα 19) παρουσιάζει τα δύο βήματα μέχρι την εμφάνιση της σκακιέρας.



***Εικόνα 19**: Βήματα εμφάνισης σκακιέρας στο Matlab*

Η σκακιέρα που εμφανίσθηκε στην οθόνη απεικονίζει τα λευκά και μαύρα πιόνια των δύο παικτών στις αντίστοιχες αρχικές θέσεις, μαζί με την αντίστοιχη αναγραφή των σειρών και των στηλών της (**Εικόνα 20**).



***Εικόνα 20**:Η σκακιέρα*

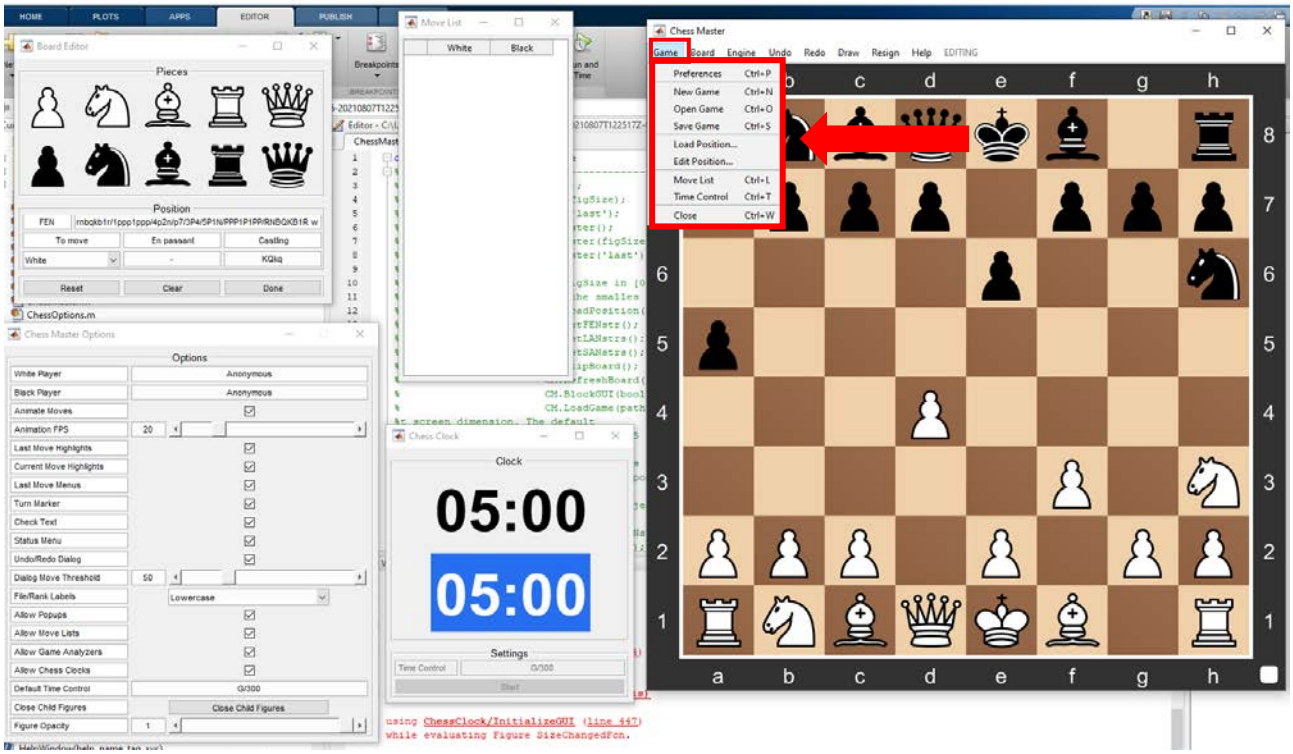
Πάνω από την ψηφιακή σκακιέρα υπάρχει μια γραμμή εργαλείων, η οποία περιέχει ρυθμίσεις σχετικές με το παιχνίδι (**Εικόνα 21**).

***Εικόνα 21**:Η γραμμή εργαλείων*

5.3 Ανάλυση γραμμής εργαλείων

Η πρώτη επιλογή που συναντούμε στην γραμμή εργαλείων (Εικόνα 22) είναι το **Game**, η οποία περιέχει τα εξής:

- Preferences: Αφορά τις ιδιότητες της σκακιέρας, όπως ονομασία λευκού και μαύρου παίκτη, την ενεργοποίηση της λίστας των κινήσεων και της ανάλυσης του παιχνιδιού κ.λπ.
- New Game: Δημιουργία νέου παιχνιδιού.
- Open Game: Άνοιγμα ενός αποθηκευμένου παιχνιδιού.
- Save Game: Αποθήκευση του υπάρχοντος παιχνιδιού.
- Load Position: Εκτέλεση της τελευταίας θέσης κίνησης του πιονιού.
- Edit Position: Επεξεργασία θέσης ενός πιονιού.
- Move List: Λίστα κινήσεων.
- Time Control: Έλεγχος χρόνου
- Close: Κλείσιμο σκακιέρας.



***Εικόνα 22**:Επιλογές του Game*

Η δεύτερη επιλογή στην γραμμή εργαλείων (Εικόνα 23) είναι το **Board**, το οποίο έχει τα εξής:

- Refresh Board: Ανανέωση σκακιέρας.
- Flip Board: Περιστροφή σκακιέρας.
- Change Theme: Αλλαγή θέματος σκακιέρας.
- Edit Theme: Επεξεργασία χρώματος των τετραγώνων της σκακιέρας.
- Screenshot: Στιγμιότυπο σκακιέρας

***Εικόνα 23**:Επιλογές του Board*

Τρίτη επιλογή είναι το **Engine** (Εικόνα 24), το οποίο έχει τα εξής:

- New Engine: Ενσωμάτωση και ρύθμιση της μηχανής σκακιού.
- Analyze Game: Ανάλυση του παιχνιδιού.
- Manage Engine: Διαχείριση μηχανών σκακιού.
- Add Engine: Προσθήκη μηχανής σκακιού, όπως Stockfish, Komondo.

***Εικόνα 24**:Η επιλογή Engine*

Τέταρτη επιλογή είναι το **Undo** (Εικόνα 25), το οποίο έχει τα εξής:

- Undo Move: Αναίρεση προηγούμενης κίνησης.
- Undo All: Αναίρεση όλων των κινήσεων (δηλαδή στην αρχική θέση).

***Εικόνα 25**:Επιλογή Undo*

Πέμπτη επιλογή είναι το **Redo** (Εικόνα 26), το οποίο έχει τα εξής:

- Redo Move: Επιστροφή κίνησης μετά από αναίρεση της (Undo Move).
- Redo All: Επιστροφή όλων των κινήσεων μετά από αναίρεση (Undo All).

***Εικόνα 26**: Επιλογή Redo*

Έκτη επιλογή είναι το **Draw** (Εικόνα 27), το οποίο περιέχει το Offer Draw. Αυτό πραγματοποιείται, όταν ένας παίκτης προσφέρεται να λήξει το παιχνίδι ισοπαλία και το δεχτεί ο αντίπαλος παίκτης.

***Εικόνα 27**:Επιλογή Draw*

Έβδομη επιλογή είναι το **Resign** (Εικόνα 28), το οποίο σημαίνει την παραίτηση του παίκτη, δηλαδή παραδίδει την νίκη στον αντίπαλο.

***Εικόνα 28**:Επιλογή Resign*

Όγδοη και τελευταία επιλογή είναι το **Help** (Εικόνα 29), το περιέχει τα εξής:

- Help: Οδηγός εκμάθησης της ψηφιακής σκακιέρας (τρόπος κίνησης πιονιών, επεξεργασία πιονιών, ιστορικό κινήσεων, ρύθμιση ρολογιού, ρύθμιση μηχανών, επιλογή ανάλυσης παιχνιδιού και επεξεργασία σκακιέρας).
- About: Σχετικά με την ψηφιακή σκακιέρα.

***Εικόνα 29**:Επιλογή Help*

Κεφάλαιο 6ο – Συμπεράσματα και Επίλογος

Το **σκάκι** δεν αποτελεί μόνο ένα επιτραπέζιο παιχνίδι αλλά και αποκωδικοποιημένες σκέψεις, ενέργειες, μεθόδους. Διότι το σκάκι είναι ένα παιχνίδι στο οποίο μπορούμε να εφαρμόσουμε αλγόριθμους τεχνητής νοημοσύνης, στατιστικές αναλύσεις και πιθανότητες. Παράγοντες όπως οι δυνατότητες του επεξεργαστή και η μνήμη παίζουν επίσης σημαντικό ρόλο στην διαχείριση του παιχνιδιού διότι ο έλεγχος για την βέλτιστη κίνηση

φτάνει πολλές φορές σε βάθος πολλών κινήσεων πράγμα που απαιτεί εκ των πραγμάτων να γίνονται πολλές αριθμητικές πράξεις σε ελάχιστο χρόνο.

Στην παρούσα εργασία προσπαθήσαμε να δούμε την υλοποίηση για το πολύ γνωστό και διαδεδομένο αυτό παιχνίδι. Είδαμε ότι η υλοποίηση μπορεί να ενισχυθεί από το **UCI** (συγκεκριμένα από το Stockfish 14) που είναι ένα ανοιχτό πρωτόκολλο επικοινωνίας και επιτρέπει στις μηχανές σκακιού να επικοινωνούν με διεπαφές χρήστη. Έτσι μπορούμε να έχουμε έναν ισχυρό αντίπαλο που χρησιμοποιεί υψηλές τεχνικές παιχνιδιού και ταυτόχρονα να εξετάζουμε την απόδοση μας μέσα από το γράφημα και το σκορ στο ταμπλό.

Είδαμε επίσης ότι η χρήση της γλώσσας **MATLAB** για την υλοποίηση ενός μαθηματικού προβλήματος έχει πλεονεκτήματα όπως:

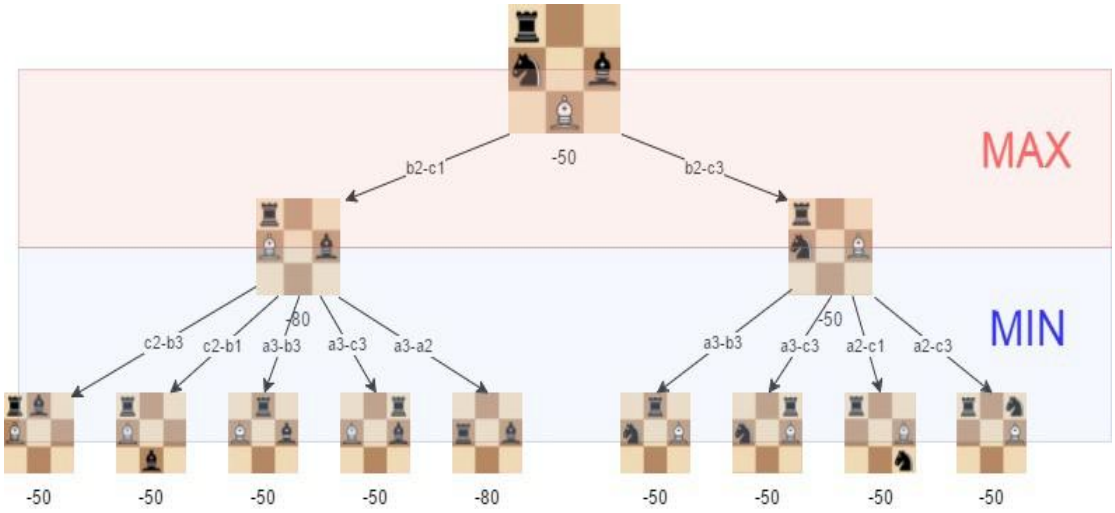
- Η υλοποίηση και ο έλεγχος των αλγορίθμων είναι πιο εύκολος
- Υπάρχει μια πληθώρα ενσωματωμένων συναρτήσεων που διευκολύνουν κατά πολύ την υλοποίηση πολύπλοκων αλγορίθμων. Επίσης μπορούμε πολύ εύκολα να χρησιμοποιούμε και εξωτερικές βιβλιοθήκες.
- Υπάρχει η δυνατότητα της δημιουργίας GUI (Graphics User Interface)
- Υπάρχει η δυνατότητα της εκτενής και σε βάθος ανάλυσης των αποτελεσμάτων και η παρουσίασή τους με γραφήματα.
- Ειδικά η υλοποίηση μαθηματικών αλγορίθμων γίνεται ευκολότερα και ακριβέστερα.

Γενικά αυτά που χρειάζεται ο υπολογιστής για να προγραμματιστεί και να παίξει σκάκι είναι τα ακόλουθα:

- Έναν τρόπο αναπαράστασης της σκακιέρας στη μνήμη, έτσι ώστε να γνωρίζει ποια είναι η κατάσταση του παιχνιδιού.
- Κανόνες για τον καθορισμό του τρόπου δημιουργίας νόμιμων κινήσεων, έτσι ώστε να μπορεί να παίζει σωστά.

- Κανόνες για τον καθορισμό του τρόπου υπολογισμού της αξίας που έχει το κάθε πιόνι, την αναπαράσταση(σκακιστική γραφή) και τον τερματισμό του παιχνιδιού.
- Μια τεχνική για να επιλέξετε την κίνηση που θα πραγματοποιήσει ανάμεσα σε όλες τις νόμιμες δυνατότητες, έτσι ώστε να μπορεί να επιλέξει μια κίνηση αντί να αναγκαστεί να επιλέξει μια τυχαία. Επίσης, έναν τρόπο σύγκρισης κινήσεων και θέσεων, ώστε να κάνει έξυπνες επιλογές.
- Τον κώδικα.
- Κάποιο είδος διεπαφής χρήστη - Graphical User Interface και αν θέλουμε κάποιον engine.

Η γενική ιδέα όλων των σκακιστικών προγραμμάτων και βασίζεται στον αλγόριθμο αναζήτησης **minimax**, ο οποίος είναι η ρίζα όλων αυτών των προγραμμάτων που υλοποιούν το σκάκι. Το δέντρο minimax είναι απλά ένας αλγόριθμος διαλογής δέντρων που μεγιστοποιεί τις κινήσεις μας και υποθέτει ότι ο αντίπαλος θα ελαχιστοποιήσει τη δική του βαθμολογία. Με απλά λόγια το δέντρο minimax, είναι απλώς **ένα μονοπάτι**, για να λάβουμε την καλύτερη απόφαση αφού εξετάσουμε όλα τα μονοπάτια μέχρι ένα ορισμένο βάθος. Στο ακόλουθο σχήμα(Εικόνα 30) μπορούμε να το δούμε ένα παράδειγμα.



***Εικόνα 30**:Αλγόριθμος Minimax*

Η διάσχιση του δένδρου αναζήτησης στο σκάκι είναι απαραίτητη προκειμένου να εντοπιστούν και πιθανές ακολουθίες κινήσεων στο παιχνίδι. Αυτός ο τρόπος διάσχισης όμως γεννά κάποια προβλήματα επειδή απαιτεί την γένεση εκθετικού αριθμού κινήσεων και θέσεων.

Όπως είδαμε από την πολυπλοκότητα του αλγορίθμου minimax όσο μεγαλύτερο είναι το βάθος τόσο αυξάνονται και οι υπολογισμοί που πρέπει να γίνουν (που δυστυχώς γίνονται στη συνέχεια όλο πιο χρονοβόροι και αργοί). Ο παράγοντας πολλαπλασιασμού των πράξεων είναι περίπου 10^3 σε κάθε επίπεδο, οπότε καταλαβαίνουμε ότι σε βάθος πολλών κινήσεων οι υπολογισμοί είναι σχεδόν αδύνατο να πραγματοποιηθούν. Αυτό το πρόβλημα το βελτίωσαν οι αλγόριθμοι Alphabeta, NegaScout και MTD(f) οι οποίοι είναι μεν πιο γρήγοροι αλλά σαφώς πιο πολύπλοκοι και βασίζονται σε τεχνικές τεχνητής νοημοσύνης (Artificial Intelligence) και σε ευρεστικές μεθόδους (Heuristics), δηλαδή καταγράφουν τις κινήσεις του αντιπάλου και δημιουργούν ένα ιστορικό κινήσεων και ένα προφίλ για το στυλ παιξίματος. Αυτές οι πληροφορίες είναι πολύτιμες και σε συνδυασμό με τον αλγόριθμο περιορίζουν σημαντικά τα πιθανά μονοπάτια.

Μια εναλλακτική προσέγγιση είναι τα **τυχαία μονοπάτια**. Ξεκινώντας από τον αρχικό κόμβο, ένα τυχαίο μονοπάτι σχηματίζεται επισκέπτοντας ακολουθιακά γειτονικούς κόμβους, μέχρι να εντοπιστεί ένας τελικός κόμβος. Διασχίζοντας τον γράφο επαναληπτικά με τυχαίο τρόπο, υπολογίζουμε τις μέσες αποστάσεις μεταξύ των κόμβων έως μια τερματική να βρεθεί (ματ ή ισοπαλία). Η δημιουργία πολλών τυχαίων μονοπατιών μπορεί να μας κάνει ικανούς να μπορούμε να εκτιμήσουμε την μακροπρόθεσμη εξέλιξη του παιχνιδιού με έναν **στατιστικό μη απωλεστικό τρόπο.** Μια πρακτική χρήση θα μπορούσε να ήταν η ακόλουθη: δεδομένης μιας θέσης, εκτίμησε την απόσταση από την νίκη του μαύρου παίκτη, του άσπρου και της ισοπαλίας.

Υπάρχει πολύ μεγάλη δυναμική στα τυχαία μονοπάτια διότι η επίδραση ορίζοντα εξαφανίζεται όσο διασχίζουμε το δένδρο αναζήτησης μέχρι το τέλος του παιχνιδιού (ματ ή ισοπαλία) και όχι έως ένα συγκεκριμένο ρηχό βάθος. Επιπλέον, η χρονική πολυπλοκότητα της γένεσης τυχαίων μονοπατιών είναι **γραμμική** αναφορικά με το βάθος, σε αντίθεση με την **εκθετική πολυπλοκότητα** των άλλων μεθόδων, συνεπώς είναι πολύ καλύτερη ενώ διατηρεί την υψηλή ποιότητα εκτίμησης χωρίς περιορισμό στο βάθος.

Το πρόγραμμα που παρουσιάστηκε στην παρούσα πτυχιακή εργασία αφενός υλοποίησε μόνο τη βασική ραχοκοκαλιά του σκακιού. Προφανώς θα μπορούσε να ενισχυθεί με περισσότερους αλγόριθμους και τεχνητή νοημοσύνη. Επιπλέον, το γραφικό περιβάλλον αν και επαρκεί για τις απλές λειτουργίες που απαιτούνται για τη διεξαγωγή μίας παρτίδας, θα μπορούσε να έχει επιπλέον δυνατότητες με περισσότερα γραφήματα και στατιστικές αναλύσεις.

Τέλος, χρήσιμη θα ήταν μια δοκιμή στρατηγικών για το παιχνίδι και πως αυτές μπορούν να εκφραστούν μέσα από μετρήσιμα χαρακτηριστικά του

ταμπλό, για να αξιοποιηθούν από μια πιο σύνθετη και ακριβή ευρετική συνάρτηση.

Βιβλιογραφία

{1}https://www.akadimiaskaki.gr/blog/39-istoria-tou-skakiou Ημερομηνία χρήσης: 25/05/2021

{2}http://www.kolivas.de/archives/371961?fbclid=IwAR1sDp6O3bX8mZ6asNNtUvN19nhoneM04R_jHBtdpx6UJIjJ9fUfyfCYbgI Ημερομηνία χρήσης: 25/05/2021

{3}https://el.wikipedia.org/wiki/Σκάκι Ημερομηνία χρήσης: 25/05/2021

{4}https://www.ebooks4greeks.gr/κανονες-σκακιου Ημερομηνία χρήσης: 25/05/2021

{5}https://en.wikipedia.org/wiki/Rules_of_chess Ημερομηνία χρήσης: 25/05/2021

{6}https://el.wikipedia.org/wiki/%CE%9A%CE%B1%CE%BD%CF%8C%CE%BD%CE%B5%CF%82_%CF%84%CE%BF%CF%85_%CF%83%CE%BA%CE%B1%CE%BA%CE%B9%CE%BF%CF%8D Ημερομηνία χρήσης: 30/05/2021

{7}https://el.wikipedia.org/wiki/%CE%A3%CE%B1%CF%87 Ημερομηνία χρήσης: 30/05/2021

{8}https://en.wikipedia.org/wiki/Draw_(chess) Ημερομηνία χρήσης: 01/06/2021

{9}https://en.wikipedia.org/wiki/King_(chess) Ημερομηνία χρήσης: 01/06/2021

{10}https://en.wikipedia.org/wiki/Queen_(chess) Ημερομηνία χρήσης: 01/06/2021

{11}https://en.wikipedia.org/wiki/Rook_(chess) Ημερομηνία χρήσης: 01/06/2021

{12}https://en.wikipedia.org/wiki/Bishop_(chess) Ημερομηνία χρήσης: 01/06/2021

{13}https://en.wikipedia.org/wiki/Knight_(chess) Ημερομηνία χρήσης: 01/06/2021

{14}https://en.wikipedia.org/wiki/Pawn_(chess) Ημερομηνία χρήσης: 01/06/2021

{15}https://en.wikipedia.org/wiki/Castling Ημερομηνία χρήσης: 01/06/2021

{16}https://el.wikipedia.org/wiki/Προαγωγή_(σκάκι) Ημερομηνία χρήσης: 01/06/2021

{17}https://en.wikipedia.org/wiki/En_passant Ημερομηνία χρήσης: 01/06/2021

{18}https://en.wikipedia.org/wiki/Algebraic_notation_(chess) Ημερομηνία χρήσης: 15/06/2021

{19}https://el.wikipedia.org/wiki/Deep_blue?fbclid=IwAR1jgJLgnCh0RPYouyjR8geaHD8w9TmI89LYzvyqAxqThUZFFXUaCvQdIls Ημερομηνία χρήσης: 17/06/2021

{20}https://www.ibm.com/ibm/history/ibm100/us/en/icons/deepblue/ Ημερομηνία χρήσης: 17/06/2021

{21}http://users.sch.gr/jenyk/index.php/artificialintelligence/ai-historicalreview/10-deepblue Ημερομηνία χρήσης: 17/06/2021

{22}https://www.chess.com/terms/stockfish-chess-engine#accomplishments Ημερομηνία χρήσης: 25/06/2021

{23}https://en.wikipedia.org/wiki/Stockfish_(chess)?fbclid=IwAR1ztq_vSSRgrH41fOgOD8OteWGUuaW8Cj7gSwMMaDqOYCMUgWUCYO5BZWg#cite_note-12 Ημερομηνία χρήσης: 25/06/2021

{24}https://www.chessprogramming.org/Stockfish?fbclid=IwAR1tnhrulzWAdZSVdy9IAxE1bge9SI162J-ivOl2YsZx7QQPNkMuHVRUAu8 Ημερομηνία χρήσης: 25/06/2021

{25}https://en.wikipedia.org/wiki/Komodo_(chess) Ημερομηνία χρήσης: 01/07/2021

{26}https://www.chessprogramming.org/Komodo#Komodo_5 Ημερομηνία χρήσης: 01/07/2021

{27}https://www.chess.com/terms/fritz-chess-engine#what Ημερομηνία χρήσης: 02/07/2021

{28}https://en.wikipedia.org/wiki/Fritz_(chess) Ημερομηνία χρήσης: 02/07/2021

{29} T. A. Marsland and M. Campbell, “Parallel search of strongly ordered game trees,” ACM Comput. Surv., vol. 14, no. 4, pp. 533–551, 1982.

{30}https://towardsdatascience.com/dissecting-stockfish-part-1-in-depth-look-at-a-chess-engine-7fddd1d83579 Ημερομηνία χρήσης: 25/07/2021

{31} M. S. Campbell and T. A. Marsland, “A comparison of minimax tree search algorithms”, Artificial Intelligence, vol. 20, no. 4, pp. 347 – 367, 1983.

{32} J. Pearl, “Asymptotic properties of minimax trees and gamesearching procedures”, Artif. Intell. , vol. 14, no. 2, pp. 113–138, 1980.

{33}https://www.gamedev.net/tutorials/programming/artificial-intelligence/chess-programming-part-i-getting-started-r1014/ Ημερομηνία χρήσης: 27/07/2021

{34}http://www.infernodevelopment.com/how-computer-chess-engines-think-minimax-tree Ημερομηνία χρήσης: 27/07/2021

{35}https://el.wikipedia.org/wiki/MATLAB Ημερομηνία χρήσης: 30/07/2021

{36}https://www.mathworks.com/discovery/what-is-matlab.html Ημερομηνία χρήσης: 30/07/2021

{37} Λουκογεωργάκη Ε και Αγγελίδης Δ. (2006), "Εισαγωγικές Διδακτικές σημειώσεις για το λογισμικό MATLAB", Πανεπιστημιακές σημειώσεις, Α.Π.Θ., Θεσσαλονίκη