Universitat de Vic
Escola Politècnica
Superior

Treball Final de Carrera

Escacs, un sistema intel·ligent

Manel Segura i Palomar

Enginyeria Tècnica d'Informàtica Especialitat Gestió
Directora: Maria Dolors Antón i Solà

Vic, setembre de 2008

Índex

1. Introducció 3

1.1. Enunciat 3
1.2. Proposta del TFC 4
1.3. Resum del TFC 5
1.4. Resum del TFC en anglés 6
2. Introducció a les escacs 7

2.1. La partida, el tauler i les peces 7
2.1.1. La partida 7
2.1.2. El tauler 7
2.1.3 Les peces 8
2.2. El moviment de les peces 9
2.2.1. La torre, l'alfil i la dama 9
2.2.2. El cavall 12
2.2.3. La captura 12
2.2.4. El peó 13
2.2.4.1. El moviment 13
2.2.4.2. La captura 14
2.2.4.3. La direcció del peó 15
2.2.5. El rei 15
2.3. El xec o escac 18
2.4. El mat 18
2.5. Introducció als sistemes d'anotació i casos generals 21
2.5.1. Introducció als sistemes d'anotació 22
2.5.2. El sistema algebraic 24
2.5.3. El sistema descriptiu 25
2.5.4. La representació lineal de la partida 28
2.5.5. Ús de cadascun dels sistemes d'anotació 29
3. Especificació dels requeriments 30

3.1. La partida 30
3.2. El taulell 30
3.3. Les Peces 30
3.3.1. Peons 30
3.3.2. Torres 31
3.3.3. Cavall 31
3.3.4. Alfils 31
3.3.5. Reina 31
4. Representació del coneixement 32

4.1. Conjunt de regles 39
4.1.1. Moviment de les peces 39
4.1.1.1. Moviments dels peons 39
4.1.1.2. Moviment de les torres 39
4.1.1.3. Moviment dels cavalls 40
4.1.1.4. Moviment dels alfils 40
4.1.1.5. Moviment de la reina 41
4.1.1.6. Moviment del rei 41
4.1.2. Final de partida 41
4.2. Base de Coneixements 41
4.2.1. Base de fets 42
4.2.2. Coneixement del domini 42
4.3. Motor d'inferència 47
5. Disseny de l'aplicació 53

5.1. SDL 53
5.2. GUI, Interfícies gràfiques 55
5.3. Part de codi 55
5.3.1. Game Loop 56

5.3.2. Inicialitzar subsistemes 56
5.3.3. Sortida estàndard 57
5.4. Disseny de la base de coneixements 57
5.5. Disseny del conjunt de regles 57
5.5.1. Moviment de les peces 57
5.5.1.1. Moviment dels peons 58
5.5.1.2. Moviment de les torres 58
5.5.1.3. Moviment d'els cavalls 59
5.5.1.4. Moviment dels alfils 59
5.5.1.5. Moviment de la reina 59
5.5.1.6. Moviments del rei 59
5.5.2. Final de partida 60
5.6. Disseny de la funció d'avaluació 60
6. Implementació 62
6.1. Interfícies gràfiques 62
6.1.1. Peces 62
6.1.1.1. Peces blanques 62
6.1.1.2. Peces negres. 62
6.1.1.3. Peça marcada 62
6.1.1.4. Possible moviment 62
6.1.2. Pantalla del menú 63
6.1.3. Pantalla principal 63
6.1.4. Pantalla blanques guanyen 64
6.1.5. Pantalla negres guanyen 64
6.2. Codi variat 65
6.2.1. Inicialització de subsistemes 65
6.2.2. Banda sonora 66
6.2.3. Game loop 66
6.3. Base de fets 70
6.4. Funció d'avaluació 76
6.5. Conjunt de regles 78
6.5.1. Moviments 78
6.5.2. Final de partida 121
6.6. Motor d'inferència 123
7. Millores i Conclusions 124
7.1. Millores 124
7.1.1. Aprenentatge 124
7.1.2. Connexió en xarxa 124
7.1.3. Altres millores 124
7.2. Conclusions 125
8. Bibliografia 126

1. Introducció

En aquesta introducció, inclouré un enunciat genèric del treball juntament amb tots els documents presentats.

1.1. Enunciat

En aquest treball pretenc descobrir, estudiar i evolucionar la teoria de jocs de la intel·ligència artificial tot fent un joc d'escacs.

La meva principal idea tracta d'implementar un joc d'escacs clàssic, o sigui un taulell amb requadres per on hi passaran certes figures sota certes restriccions. Aquesta implementació en si, seria un joc totalment jugable per dues persones però el que més m'interessa del projecte, és poder crear el que anomenaríem una "Intel·ligència Artificial", aquesta intel·ligència es basaria en codi, un clàssic algorisme Minimax on hauré de cercà la funció d'avaluació més adient per els escacs.

A part de l' Intel·ligència artificial, intentaré crear una interfície gràfica que sigui el mes adient possible i que no sigui gens avorrida, intentant posar fins i tot sons.

1.2. Proposta del TFC

Dades de l'alumne:

Nom:	Manel
Cognoms:	Segura Palomar
Adreça:	C/ Puig-Guardial, 169
Codi postal:	08560
Població:	Manlleu
Província:	Barcelona
E-mail:	msegura@alvic.net

Títol del treball: Escacs, un sistema intel·ligent

Director/a del treball: Mª Dolors Antón i Solà

Descripció dels objectius i les característiques del treball:

En aquest treball pretenc descobrir, estudiar i evolucionar la teoria de jocs de la intel·ligència artificial tot fent un joc d'escacs.

La meva principal idea tracta d'implementar un joc d'escacs clàssic, o sigui un taulell amb requadres per on hi passaran certes figures sota certes restriccions. Aquesta implementació en si, seria un joc totalment jugable per dues persones però el que més m'interessa del projecte, és poder crear el que anomenaríem una "Intel·ligència Artificial", aquesta intel·ligència es basaria en dos conceptes diferenciats, el primer seria purament codi, un clàssic algorisme Minimax on hauré de cercà la funció d'avaluació més adient per els escacs. El segon, seria la creació d'una base de dades on s'haurien d'anar guardant les jugades més freqüents per arribar a la victòria o inclús jugades de l'oponent, el que vulgarment anomenaríem "aprenentatge".

El meu objectiu és centrar-me més en la part de la "Intel·ligència artificial" que no en l'aspecte gràfic, sense deixar-lo descuidat es clar.

Firmes:

L'estudiant El director/a

1.3. Resum del TFC

Resum de Treball Final de Carrera

Enginyeria Tècnica d'Informàtica Especialitat Gestió

Títol: Escacs, un sistema intel·ligent.

Paraules clau: Minimax, intel·ligència artificial, escacs, alfa-beta, SDL.

Autor: Manel Segura Palomar

Direcció: Maria Dolors Antón Solà

Avalador: -

Data: Setembre de 2008

Resum

Des dels inicis dels ordinadors com a màquines programables, l'home ha intentat dotar-los de certa intel·ligència per tal de pensar o raonar el més semblant possible als humans. Un d'aquests intents ha set fer que la màquina sigui capaç de pensar de tal manera que estudiï jugades i guanyi partides d'escacs.

En l'actualitat amb els actuals sistemes multi tasca, orientat a objectes i accés a memòria i gràcies al potent hardware del que disposem, comptem amb una gran varietat de programes que es dediquen a jugar a escacs. Però no hi han només programes petits, hi han fins i tot màquines senceres dedicades a calcular i estudiar jugades per tal de guanyar als millors jugadors del mon.

L'objectiu del meu treball és dur a terme un estudi i implementació d'un d'aquests programes per el que es divideix en dos parts.

La part teòrica o de l'estudi, consta d'un estudi d'els sistemes d'intel·ligència artificial que és dediquen a jugar a escacs, estudi i cerca d'una funció d'avaluació vàlida i estudi dels algorismes de cerca.

La part pràctica del treball és bassa en l' implementació d'un sistema intel·ligent capaç de jugar a escacs amb certa lògica. Aquesta implementació es porta a terme amb l'ajuda de les llibreries SDL, utilitzant l'algorisme minimax amb poda alfa-beta i codi c++.

Com a conclusió del projecte m'agradaria remarcar que l'estudi realitzat m'ha deixat veure que crear un joc d'escacs no era tan fàcil com jo pensava però m'ha aportat la satisfacció d'aplicar tot el que he aprés durant la carrera i de descobrir moltes altres coses noves.

1.4. Resum del TFC en anglés

Summary of Career Final Work

Technical Engineering of Computer Science

Title: Chess, an intelligent system.

Key words: Minimax, artificial intelligence, chess, alpha-beta, SDL.

Author: Manel Segura Palomar

Direction: Maria Dolors Antón Solà

Guarantor: -

Date: September of 2008

Summary

From the beginnings of the computers as programmable machines, the man has attempted to endow them with a certain intelligence in order to think or to reason the most similar possible to the humans. One of these attempts has seven to make that the machine is capable of thinking in such a way that it studies strokes and wins games of chess.

In the present with the current systems fines task, directed at objects and access to memory and thanks to the powerful hardware of what we order, we count on a great variety of programs that dedicate themselves to playing chess. But there are not only small programs, there are even whole machines devoted to calculating and studying the best players strokes in order to win.

The goal of my work is to carry out a study and implementation of one of these programs for the one that divides into two deliveries.

The theoretical part of the study consists in a study of the artificial intelligence systems that are they dedicate to play chess, study and search of a function of valid evaluation and study of the algorithms of search.

The practical part of the work is a pond in the implementation of an intelligent system capable of playing chess with certain logic. This implementation is carried out with the help of the bookshops SDL, using the Minimax algorithm with pruning alpha-beta and code C++.

As a conclusion of the project it would like to remark that the study carried out has let for me see it so easy that to create a game of chess was not as I thought but has brought me the satisfaction of applying all what I have studied during the career and of discovering many other new things.

2. Introducció a les escacs

Per tal de saber les regles bàsiques del joc, he decidit crear tot un apartat dedicat al joc en si, per tal de que puguem estudiar els requeriments de l'aplicació tant en classes com en funcions de la intel·ligència artificial.

2.1. La partida, el tauler i les peces

2.1.1. La partida

En una partida d'escacs, hi participen dos jugadors movent sobre el tauler les peces del seu bàndol, les quals es distingeixen pel seu color: un dels dos jugadors té les peces blanques, i l'altre, les peces negres. La partida representa una batalla entre dos exèrcits -el blanc i el negre- el propòsit dels quals consisteix a capturar el rei de l'adversari. El qui ho aconsegueix guanya; per altra banda, si un dels dos contrincants considera la seva posició com a perdedora sense remei pot abandonar -rendir-se- i concedir així la victòria al seu adversari. En el cas que s'arribés a una situació en què cap dels dos bàndols no pogués assolir el triomf, la partida es declara taules -empat -.

Cada jugador mou una de les seves peces després que ho hagi fet el contrincant; la peça moguda ha de situar-se en una casella o bé lliure o bé ocupada per una peça enemiga, en aquest darrer cas es produeix una captura, la qual, en el joc dels escacs, sempre és voluntària i mai obligatòria. Un jugador només pot capturar -matar, menjar-se- les peces de l'adversari i en cap cas les del seu propi bàndol. La jugada és irreversible, no es pot retirar i aquell jugador que, corresponent-li tirar, toqui una peça que es pugui moure, queda obligat a moure-la; aquesta norma s'expressa amb la dita *peça tocada, peça jugada*; en l'argot de club d'escacs, d'infringir aquesta norma, se'n diu *tocar el piano*. La primera jugada de la partida correspon sempre al jugador que tingui les peces blanques.

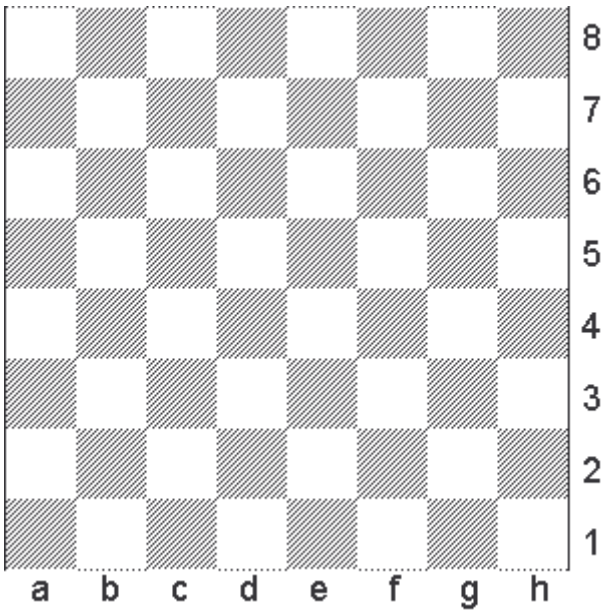
2.1.2. El tauler

El tauler, de forma quadrada, està format per seixanta-quatre caselles -o quadres-, trenta-dues de blanques i trenta-dues de negres; cada casella en té al costat una del color diferent. Dins del tauler, les caselles formen línies; les verticals s'anomenen columnes, les horitzontals, files, i la successió de caselles del mateix color unides pel vèrtex constitueix una diagonal.

El tauler es situa entre els dos jugadors, de tal manera que, sempre, a l'extrem situat a la dreta de cada jugador hi hagi un quadre blanc. Per tal d'orientar-se dins del tauler, les columnes es designen per una lletra -a, b, c, d, e, f, g, h- i les files amb una xifra -1, 2, 3, 4, 5, 6, 7, 8-. Cada casella s'anomena per la lletra de la columna i la xifra de la fila en què es troba situada; així, per exemple, hi ha les caselles c7, g4, f8, b1, etc. Les diagonals es denominen pels noms de les caselles dels seus extrems, com ara la diagonal h3-c8, o la diagonal a5-d8; la diagonal a1-h8 és la gran diagonal negra i la diagonal h1-a8 és la gran diagonal blanca. En una partida, la columna a és la que es troba a l'extrem esquerra de qui juga amb

blanques i a l'extrem dret de qui juga amb negres; les files 1 i 2 són aquelles on es trobaven les blanques a l'inici de la partida mentre que les 7 i 8 són les que ocupaven les negres al principi del joc. En els taulers representats a les pàgines web i en el material imprès -llibres, revistes, notícies de diari, etc-, la columna a sempre es troba a l'extrem esquerre del lector i la h a l'extrem dret; la fila 1 es troba a baix de tot del tauler i la fila 8, a l'extrem superior; tal com pot veure's al diagrama 2.1.2.1.

DIAGRAMA 2.1.2.1

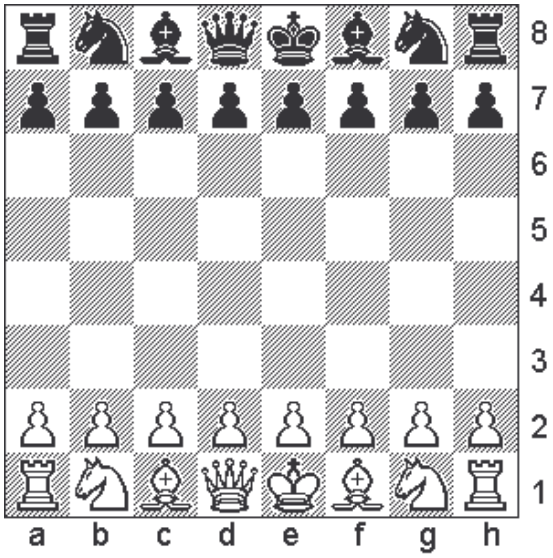


2.1.3 Les peces

En començar la partida, cada jugador té les següents peces: un rei, una dama o reina, dues torres, dos alfils, dos cavalls i vuit peons.

Per iniciar el joc, cadascun dels contrincants situa les seves peces dins de les dues files que li resulten més properes, seguint aquesta pauta: les torres a les caselles dels extrems -a1 i h1 per a les blanques, a8 i h8 per a les negres-, al costat de cada torre, hi va un cavall - a b1 i g1 els cavalls blancs i a b8 i g8 els negres- ; al costat dels cavalls, s'hi situen els alfils - els blancs a c1 i f1, els negres a c8 i f8-; a les caselles centrals de la fila 1 -les blanques- o de la fila 8 -les negres- s'hi col·loquen el rei i la dama - el rei blanc a e1, el rei negre a e8; la dama blanca a d1, la dama negra a d8-; per recordar la situació correcta del rei i la dama cal tenir en compte que, en la posició inicial, el rei negre ocupa una casella blanca i el rei blanc una casella negra, mentre que la dama comença la partida situada en una casella del seu color. A la fila 2 les blanques hi posen els seus vuit peons, mentre que les negres situen els seus a la fila 7; i així les peces queden tal com es veu al diagrama 2..3.1. El quadre que ocupa una peça a l'inici del joc és la seva casella inicial.

DIAGRAMA 2.1.3.1

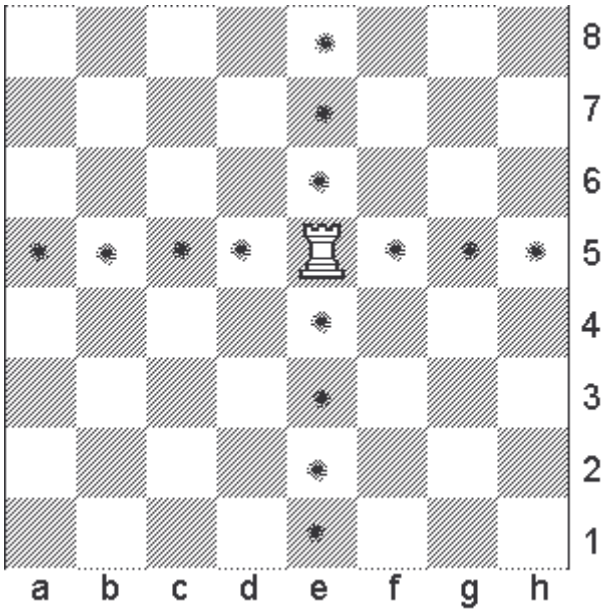


2.2. El moviment de les peces

2.2.1. La torre, l'alfil i la dama

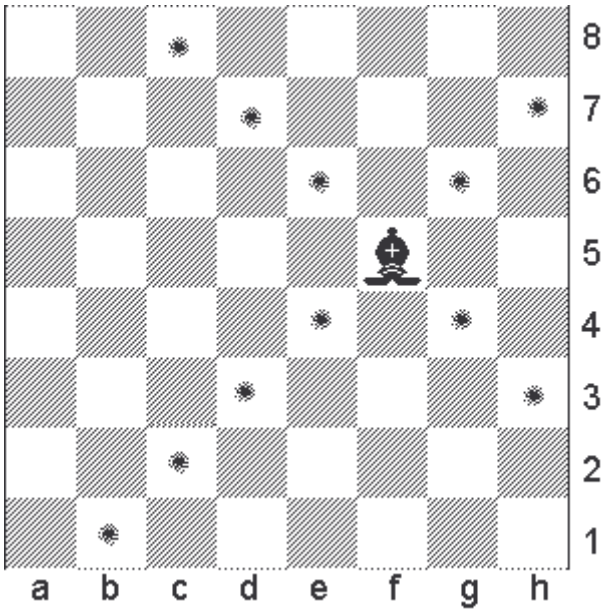
La torre es mou en vertical i en horitzontal, per tant, es desplaça per les files i les columnes, tal com està marcat al diagrama 2.2.1.1, on la torre pot traslladar-se a qualsevol de les caselles assenyalades amb un punt

DIAGRAMA 2.2.1.1



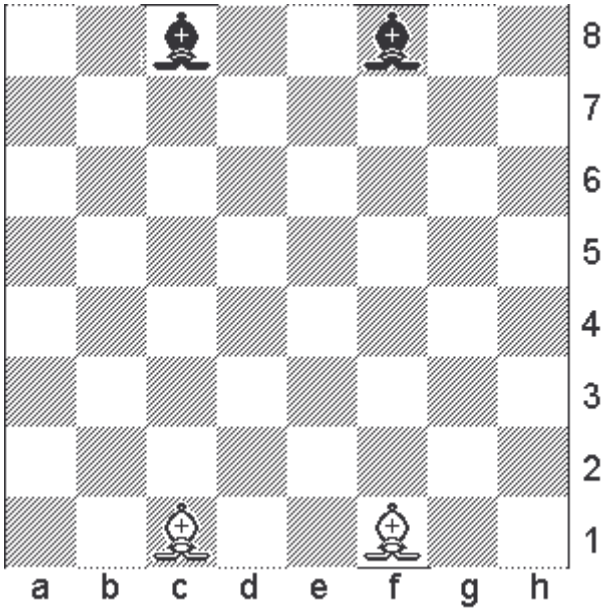
L'alfil es mou en diagonal; al diagrama 2.2.1.2, l'alfil de f5 pot moure's a totes les caselles marcades amb un punt

DIAGRAMA 2.2.1.2



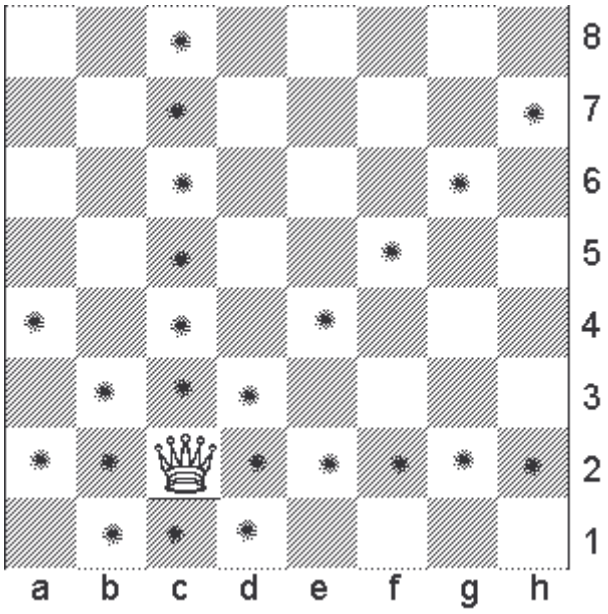
Com pot veure's, per la seva manera de desplaçar-se, l'alfil sempre es mou per caselles d'un mateix color; en començar la partida, cada jugador disposa d'un alfil de caselles blanques i d'un alfil de caselles negres, tal com apareix al diagrama 2.2.1.3, on els alfils es troben en les seves caselles inicials; l'alfil blanc de f1 i el negre de c8 sempre es mouran per caselles blanques, mentre que l'alfil blanc de c1 i el negre de f8 sempre jugaran a caselles negres.

DIAGRAMA 2.2.1.3



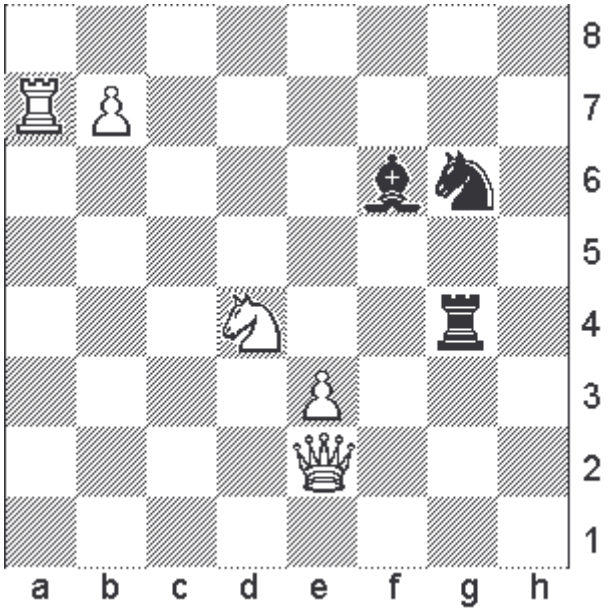
La dama combina els moviments de la torre i de l'alfil, es mou, doncs, en vertical, horitzontal i diagonal. Al diagrama 2.2.1.4, la dama blanca pot anar a qualsevol de les caselles assenyalades amb un punt.

DIAGRAMA 2.2.1.4



La torre, l'alfil i la dama mai no poden saltar per damunt de caselles ocupades per peces pròpies o de l'adversari. Així, al diagrama 2.2.1.5, la torre blanca només pot moure's per la columna a; l'alfil negre pot desplaçar-se per tota la diagonal d8-h4 , però a la diagonal a1-h8 només pot moure's per les caselles e5, g7 i h8; la dama blanca pot anar només a les caselles de la fila 2 i als quadres e1, d1, d3, f1, f3; la torre negra pot traslladar-se únicament a e4, f4, h4, g5, g3, g2, g1.

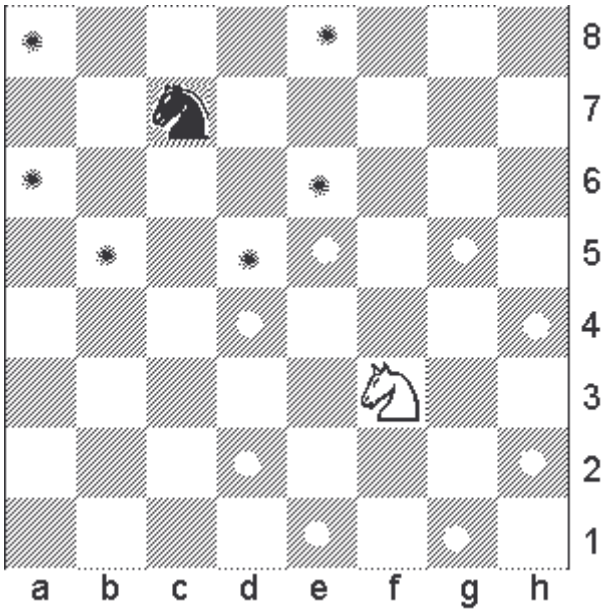
DIAGRAMA 2.2.1.5



2.2.2. El cavall

El cavall es mou fent un pas de torre i un pas d'alfil, o bé un pas d'alfil i un pas de torre, sense anar a parar mai a una casella situada al costat de la que ocupava anteriorment. Al diagrama 2.2.2.1, el cavall blanc pot anar a les caselles marcades amb un punt blanc, i el cavall negre a les assenyalades amb un punt negre. A diferència de la torre, l'alfil i la dama, el cavall sí que pot saltar per damunt de caselles ocupades per peces pròpies o del contrincant; així doncs, la mobilitat dels cavalls del diagrama 2.2.2.1 no es veuria pas afectada si a les caselles del seu voltant hi hagués d'altres peces.

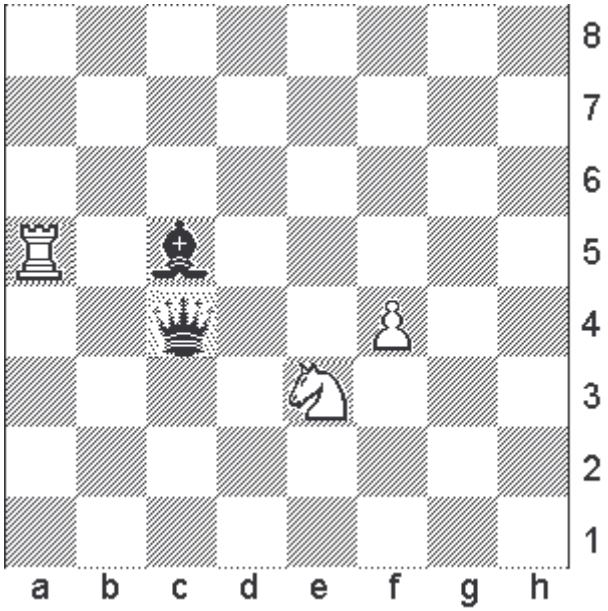
DIAGRAMA 2.2.2.1



2.2.3. La captura

Totes les peces estudiades fins ara capturen de la mateixa manera que es mouen. Efectuar una captura -prendre, matar o menjar-se una peça- consisteix a situar una peça en una casella ocupada per una peça enemiga, la qual es retira del tauler i deixa de jugar. Al diagrama 2.2.3.1, la torre blanca pot capturar l'alfil negre, el qual es podria es menjar el cavall blanc; aquest cavall blanc podria prendre la dama negra, la qual pot matar el peó blanc.

DIAGRAMA 2.2.3.1

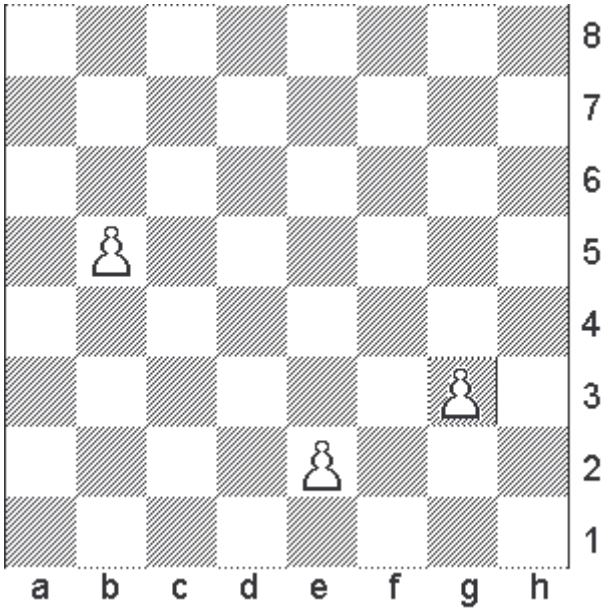


2.2.4. El peó

2.2.4.1. El moviment

El peó es mou avançant sempre un pas cap endavant en vertical, no pot mai ni retrocedir ni moure's en horitzontal. Quan es troba en la seva casella inicial, és a dir, quan encara no ha fet cap moviment, el peó pot optar per avançar dues caselles de cop, dret que perd després d'haver efectuat el seu primer moviment, tant si ha corregut dues caselles com només una. Al diagrama 2.2.4.1.1, el peó de la casella e2 pot decidir si avançar una casella -a e3- o dues -a e4- mentre que els peons de g3 i b5, que ja no ocupen la seva casella inicial i, per tant, s'han mogut, només poden córrer una casella.

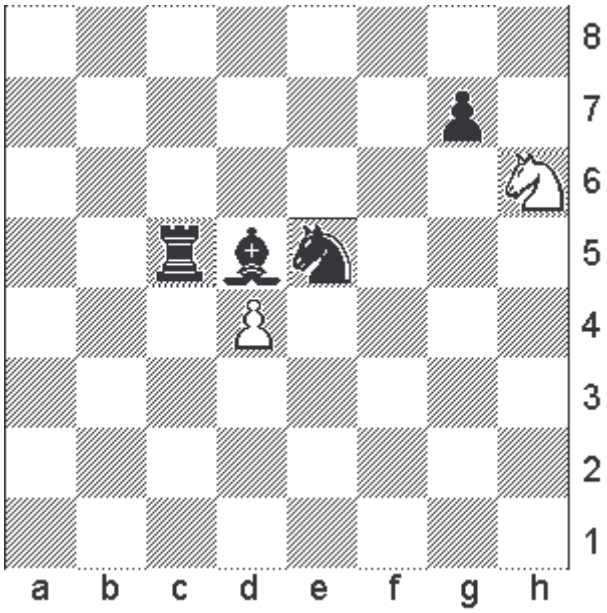
DIAGRAMA 2.2.4.1.1



2.2.4.2. La captura

El peó es mou fent una passa en vertical, però captura fent una passa en diagonal cap endavant. Al diagrama 2.2.4.2.1.1, el peó blanc de d4 pot capturar la torre o el cavall però no pas l'alfil; mentre que el peó negre de g7 pot moure's a g6 o g5 o bé capturar el cavall blanc de h6.

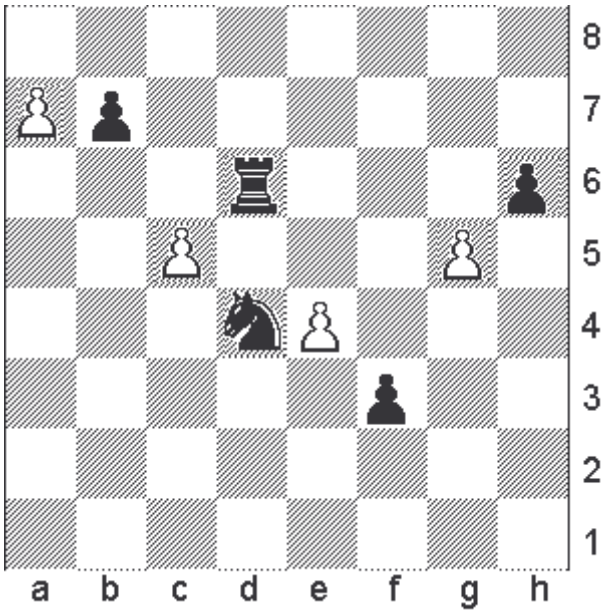
DIAGRAMA 2.2.4.2.1



2.2.4.3. La direcció del peó

Com s'ha dit al principi, en les webs i en el material imprès -llibres, revistes i notícies de diaris-, les files 1 i 2, on es troben les caselles inicials de les blanques, es situen sempre a la part de baix del tauler; mentre que les caselles inicials de les negres -files 7 i 8- es troben sempre a la part superior del tauler. Gràcies a aquest convencionalisme, sempre sabem quina és la direcció dels peons; així, com que els peons blancs sempre avancen cap amunt, i els negres cap avall, al diagrama 2.2.4.3.1 hom veu que el peó blanc de la columna a està a punt de promocionar, mentre que el peó negre de la columna b es troba en la seva casella inicial i li faltarien entre cinc o sis moviments per arribar a la promoció; el peó blanc de c5 pot capturar la torre, però no pas el cavall perquè això significaria anar cap enrere, cosa que un peó no pot fer mai; el peó negre de la columna h i el blanc de la columna g poden capturar-se l'un a l'altre, cosa que no poden fer el peó blanc de la columna e i el negre de la columna f.

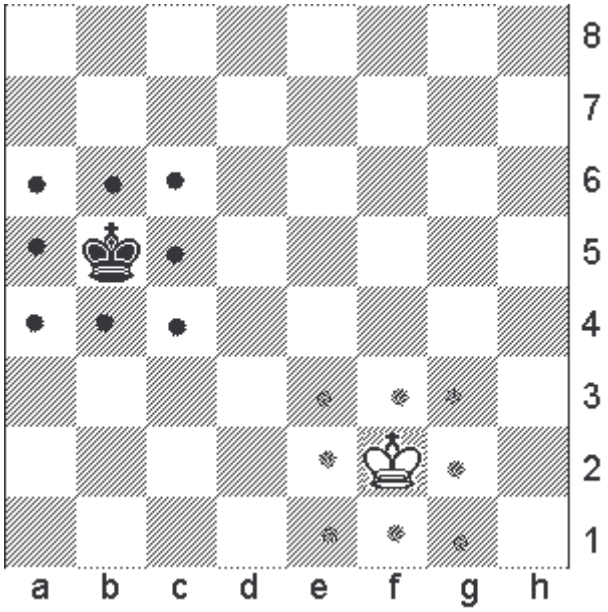
DIAGRAMA 2.2.4.3.1



2.2.5. El rei

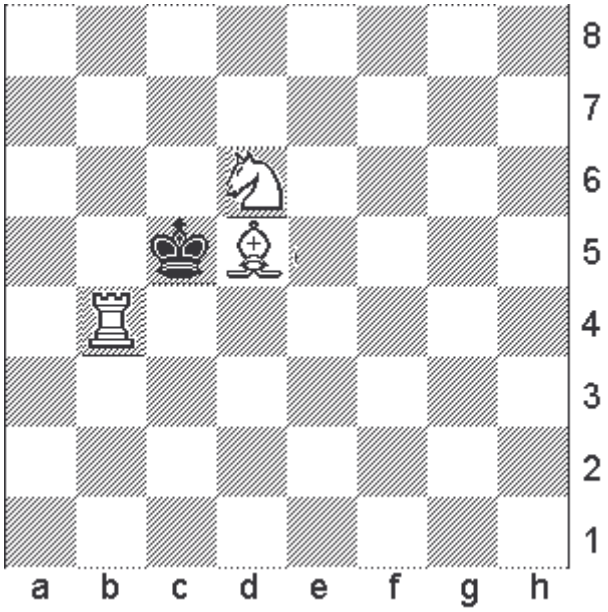
El rei, la peça més important del joc, es mou fent una passa en qualsevol direcció, és a dir, desplaçant-se a una casella que es trobi al costat de la que ocupava abans; al diagrama 2.2.5.1, el rei blanc es pot moure a qualsevol de les caselles marcades amb un punt gris i el rei negre a les caselles marcades amb un punt negre.

DIAGRAMA 2.2.5.1



El rei pot efectuar captures de la mateixa manera com es mou; així, al diagrama 2.2.5.2, el rei negre pot menjar-se la torre, el cavall o l'alfil blancs

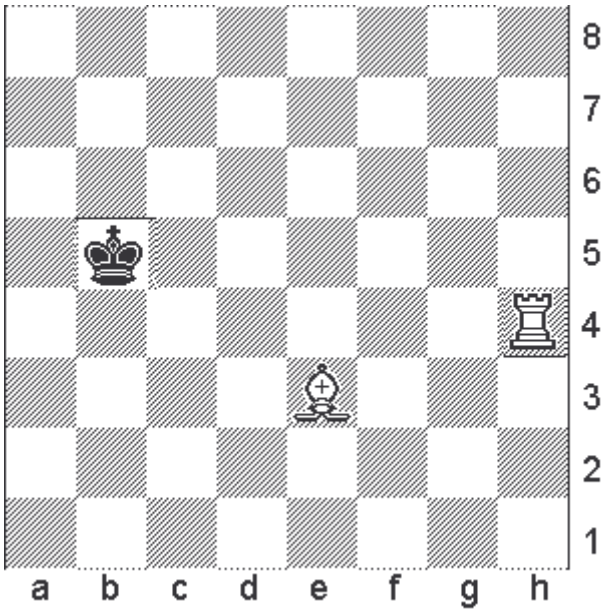
DIAGRAMA 2.2.5.2



A diferència de les altres peces, el rei té una important limitació: no pot situar-se en una casella on s'exposi a ser-hi capturat per una peça enemiga. Al diagrama 2.2.5.3, el rei negre no pot moure's a c5 ni a

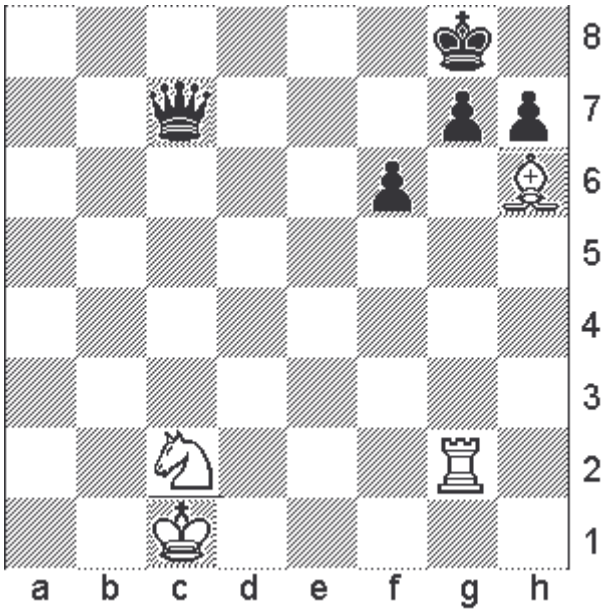
b6 perquè hi seria amenaçat per l'alfil blanc de e3, ni tampoc a les caselles a4, b4 o c4 perquè hi podria ser capturat per la torre blanca de h4

DIAGRAMA 2.2.5.3



A conseqüència d'aquesta limitació, no és lícit fer cap jugada que deixi el rei amenaçat per peces de l'adversari. Al diagrama 2.2.5.4, el cavall no pot moure's perquè, llavors, la dama negra podria capturar el rei blanc, així com el peó negre de g7 pot avançar a g6 o g5, però no pas capturar l'alfil blanc de h6, cosa que deixaria el rei negre amenaçat per la torre blanca

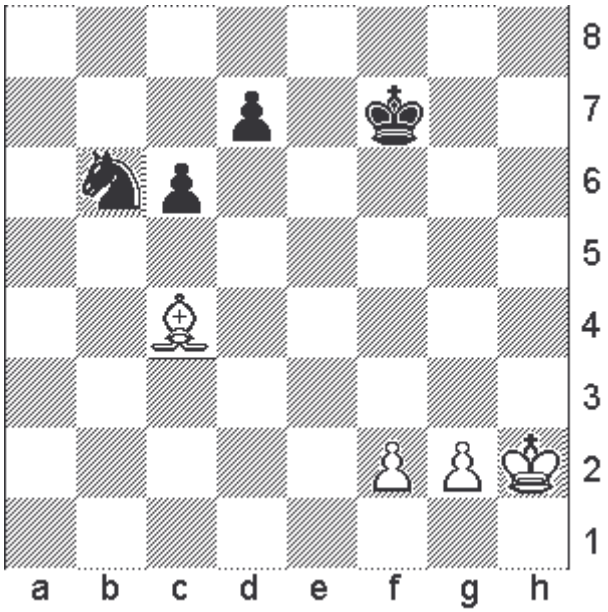
DIAGRAMA 2.2.5.4



2.3. El xec o escac

El xec o escac és aquella jugada amb què un jugador amenaça capturar el rei de l'adversari, el qual queda obligat a defensar el rei del xec perquè no està permès deixar el rei amenaçat.

DIAGRAMA 2.3.1



Al diagrama 2.3.1, les blanques han jugat el seu alfil a c4, on fa xec al rei negre, situat a f7; per defensar-se, les negres poden triar entre tres solucions:

a) retirar el rei de la casella amenaçada, en aquest cas, moure'l a una casella que no sigui ni e6 ni g8, on continuaria estant en xec.

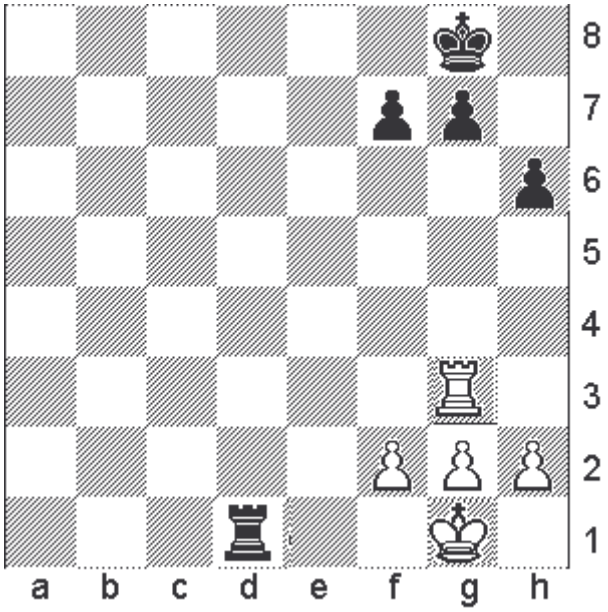
b) capturar la peça que fa xec, és a dir, que el cavall negre de b6 es mengés l'alfil blanc de c4

c) cobrir-se o tapar-se, ço és, situar una peça en una casella que obstrueixi l'acció de la peça amenaçadora, en aquest cas, jugar el peó negre de d7 a d5. Aquesta darrera defensa no és possible quan la peça que fa xec és un cavall o un peó.

2.4. El mat

El xec sempre és obligatori defensar-lo; ara bé, en la posició del diagrama 2.4.1, la torre negra des de d1 fa xec al rei blanc, situat a g1; en aquesta posició, el rei no té casella on refugiar-se perquè a f1 o h1 continuaria estant en xec i, a més, no pot moure's a la fila 2 perquè li ho impedeixen els seus propis peons de f2, g2 i h2. Per altra banda, no hi cap peça blanca que pugui capturar la torre negra de d1, ni tampoc no és possible cobrir-se, és a dir, interposar una peça blanca entre aquesta torre i el rei blanc.

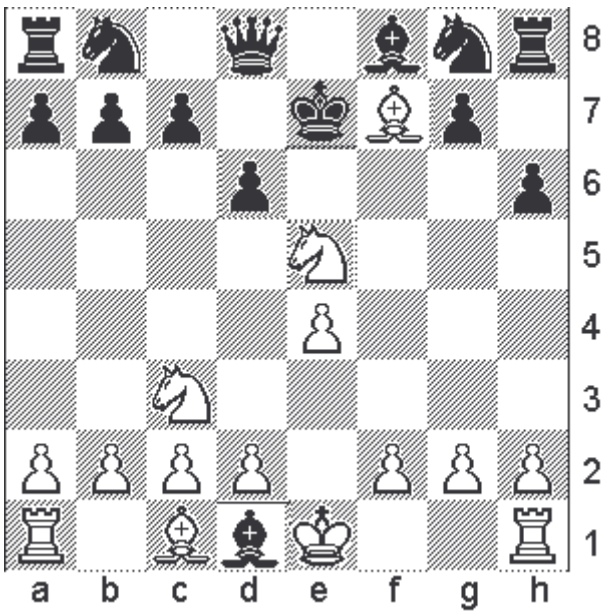
DIAGRAMA 2.4.1



Un xec que no pugui defensar-se és mat; aleshores, la partida s'acaba amb victòria del bàndol que ha fet el mat, perquè és aquesta la manera d'aconseguir l'objectiu de capturar el rei adversari.

El mat és la jugada guanyadora; per això, en una partida convé no deixar passar mai l'oportunitat de fer-la.

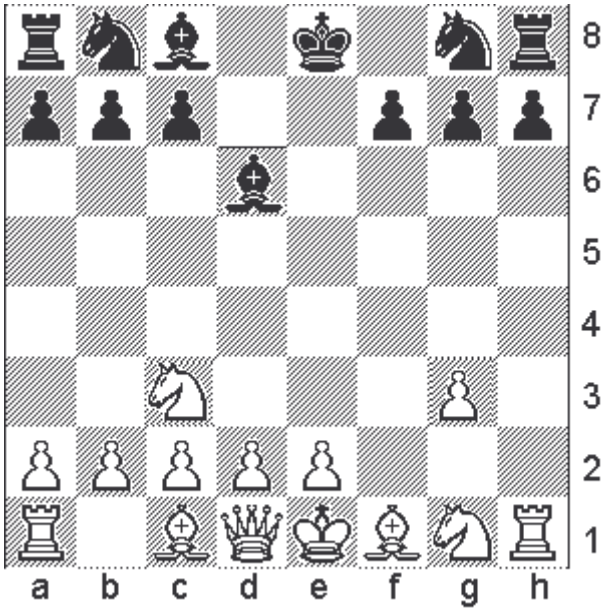
DIAGRAMA 2.4.2
juguen blanques



En la posició del diagrama 2.4.2, toca tirar a les blanques, quina seria la seva millor jugada?. Si el seu cavall de c3 se'n va a d5, el rei negre queda en xec i no pot moure's enlloc -les caselles e8 i e6 les controla

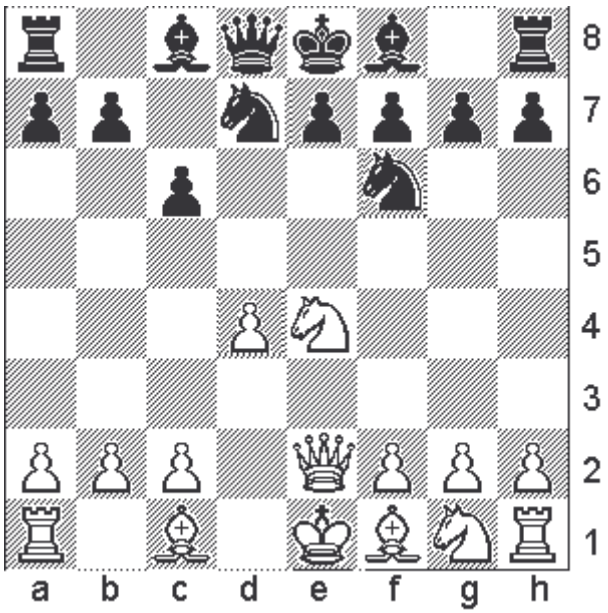
l'alfil blanc de f7, d7 està amenaçada pel cavall de e5-; per altra banda, cap peça negra no pot capturar el cavall de d5 -la peça que està fent xec- i ací tampoc no poden cobrir-se perquè el xec el fa un cavall, la peça que pot saltar per damunt de les altres. Així doncs, ací, les blanques guanyen movent el seu cavall de c3 a d5 perquè fan mat.

DIAGRAMA 2.4.3
juguen negres



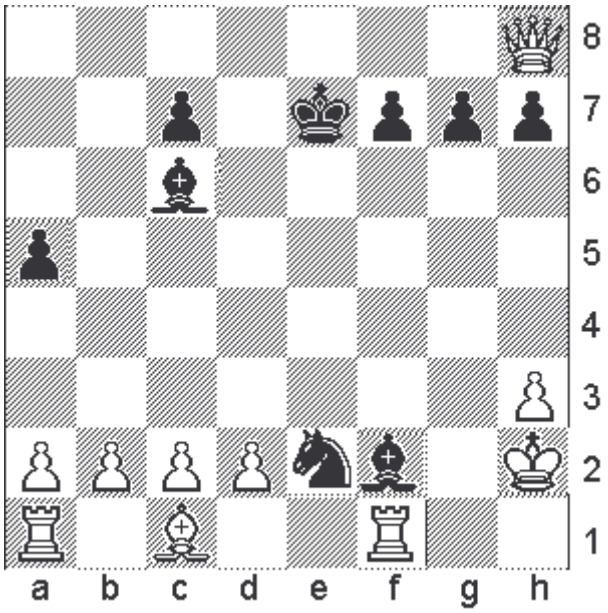
En la posició del diagrama 2.4.3, juguen les negres; si el seu alfil de d6 captura el peó blanc de g3, es produeix una situació de mat, tal com es pot comprovar.

DIAGRAMA 2.4.4
juguen blanques



Al diagrama 2.4.4, juguen les blanques i fan mat movent el seu cavall de e4 a d6, ja que el rei negre no pot moure's enlloc, i el peó de e7 no pot capturar el cavall de d6 ja que, llavors, el rei negre quedaria amenaçat per la dama blanca, situada a e2.

DIAGRAMA 2.4.5
juguen negres



Al diagrama 2.4.5, juguen les negres; si el seu alfil de f2 es situa a g3, el rei blanc no pot defensar-se del xec ja que les caselles h1 i g2 estan dominades per l'alfil de c6, la casella g1 la controla el cavall de e2 i aquest cavall mateix impedeix que el rei pugui capturar l'alfil de g3.

2.5. Introducció als sistemes d'anotació i casos generals

Si s'arriba a una situació en què cap dels dos bàndols no pot alçar-se amb la victòria, la partida es declara taules -empat-. Una partida pot ser taules per qualsevol de les següents raons:

a) **per mutu acord dels dos contrincants**: en qualsevol moment de la partida, un jugador pot proposar a l'adversari deixar la partida en taules, si aquest s'hi avé, la partida es declara taules.

b) **si cap dels dos bàndols no disposa de prou peces per fer el mat a l'adversari**: aquesta situació es dóna si damunt del tauler només queden els dos reis sols, o bé si un jugador té el rei sol i a l'altre únicament li queden el rei i un alfil o el rei i un cavall, casos aquests, en què, com pot comprovar-se, dur a terme el mat és impossible.

c) **en el cas que durant una partida es repeteixi tres vegades la mateixa posició**: aquestes són les dites *taules fotogràfiques*

d) **per xec continu**: si un dels dos bàndols va fent xec a l'adversari a cada jugada, a vegades el xec continu provoca una situació de taules fotogràfiques.

e) **Si han transcorregut més de cinquanta jugades sense haver-se mogut un peó ni haver-se capturat cap peça**: una norma que cal tenir present en els finals de partida

2.5.1. Introducció als sistemes d'anotació

Per tal de poder progressar en el coneixement del joc dels escacs, cal llegir llibres i revistes on hi apareixen comentades i analitzades les partides dels grans mestres, tant dels actuals com dels d'èpoques passades, com també anàlisis i estudis de determinades posicions. Aquest material només es pot llegir si hom coneix els sistemes d'anotació dels escacs.

Hi ha dos sistemes d'anotació: l'algebraic i el descriptiu. Tots dos consisteixen a apuntar les jugades d'una partida, mencionant primer, el número de la jugada dins la partida, després la peça moguda, designada sempre per la seva inicial en majúscula, i, al final, la casella on es desplaça aquesta peça. A la taula 2.5.1.1 mostrem els noms de les peces i les seves inicials.

TAULA 2.5.1.1

PEÇA	INICIAL
REI	R
DAMA	D
TORRE	T
ALFIL	A
CAVALL	C
PEÓ	P

També existeix un conjunt de símbols comuns als dos sistemes d'anotació que apareix descrit a la taula 2.5.1.2.

TAULA 2.5.1.2

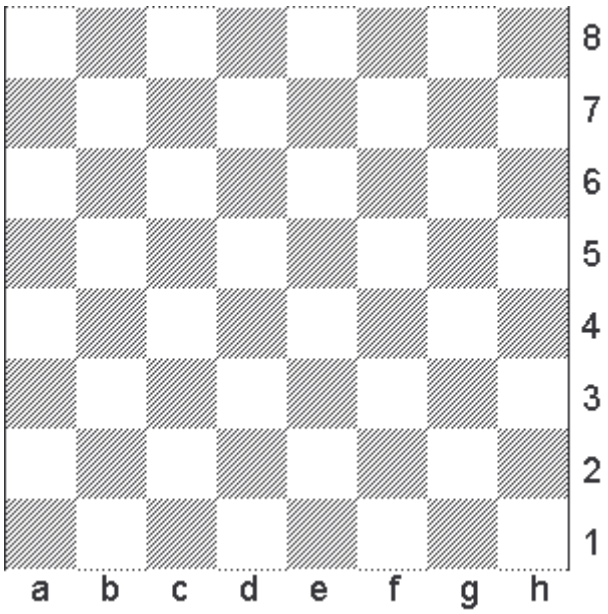
SÍMBOL	SIGNIFICAT
x	CAPTURA
O-O	ENROC CURT
O-O-O	ENROC LLARG
!	BONA JUGADA
!!	JUGADA MOLT BONA
?	JUGADA DOLENTA
??	GREU ERROR
!?	JUGADA DUBTOSA
?!	JUGADA FLUIXA
+	XEC
++	MAT ; alguns llibres marquen el mat amb el símbol #
a.p.	CAPTURA AL PAS
1-0	VICTÒRIA DE LES BLANQUES
0-1	VICTÒRIA DE LES NEGRES
1/2 /1/2	TAULES
e8=D	LA CORONACIÓ DEL PEÓ SEGONS EL SISTEMA ALGEBRAIC
P8R=D	LA CORONACIÓ DEL PEÓ SEGONS EL SISTEMA DESCRIPTIU

La diferència entre tots dos sistemes rau en la manera de designar les caselles del tauler

2.5.2. El sistema algebraic

El sistema algebraic es basa en les combinacions de xifres i lletres que hem vingut usant fins ara i que mostrem al diagrama 2.5.2.1.

DIAGRAMA 2.5.2.1.



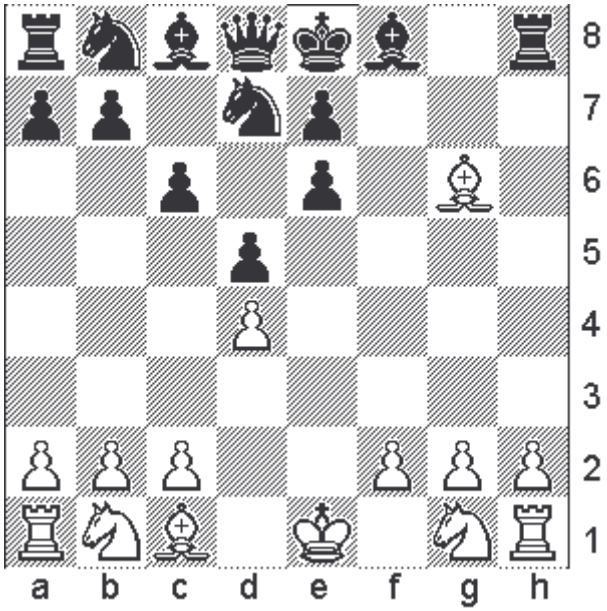
Com hem dit abans, una jugada s'escriu mencionant la inicial de la peça moguda; ara bé, en el sistema algebraic, això no es fa si la peça que es mou és un peó.

Ara presentem un exemple de partida escrita en sistema algebraic. Per altra banda, en aquesta posició final, s'hi produeix una situació de mat. Convé observar-la, fixant-se en quina és la peça que fa xec i per quins motius no pot defensar-se'n el rei enemic.

Partida 2.5.2.1

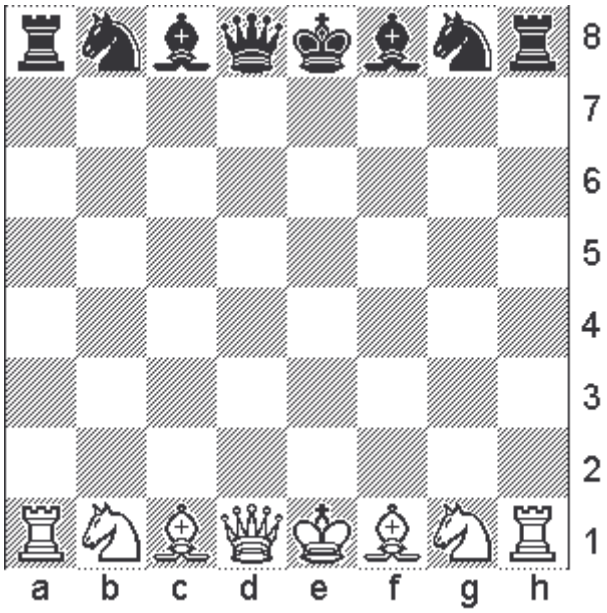
Blanques Negres
1.e4 c6
2.d4 Cf6
3.Ad3 d5
4.e5 Cf6-d7
(Com que a la casella d7 tant s'hi podia traslladar el cavall de f6 com el de b8, ací fem constar de quina casella procedeix el cavall)
5.e6 f7xe6
6.Dh5+ g6
7.Dxg6 h7xg6
8.Axg6++ 1-0

DIAGRAMA 2.5.2.2
posició final de la partida 1



2.5.3. El sistema descriptiu

DIAGRAMA 2.5.3.1



En el sistema descriptiu, les columnes es designen segons el nom de la peça que les ocupa en la posició inicial de la partida; així hi ha la columna de rei (R) i la columna de dama (D). Si observem el diagrama 2.5.3.1, veurem que els jugadors comencen la partida amb dues torres, dos cavalls i dos alfils; aleshores, aquestes peces es distingeixen fixant-se si la seva posició inicial es troba més propera del rei o de la dama. Així, hi ha la torre de dama (TD), el cavall de dama (CD) l'alfil de dama (AD), la torre de rei (TR),

el cavall de rei (CR) i l'alfil de rei (AR). Per veure l'equivalència de les designacions de columnes en el sistema descriptiu respecte de l'algebraic, cal observar aquesta taula

Taula 2.5.3.1

DESIGNACIÓ DE CADA COLUMNA SEGONS EL SISTEMA DESCRIPTIU	DESIGNACIÓ DE CADA COLUMNA SEGONS EL SISTEMA ALGEBRAIC
TD	a
CD	b
AD	c
D	d
R	e
AR	f
CR	g
TR	h

En aquest sistema, hi ha una doble designació de les files: l'una parteix de la fila inicial de les blanques, i l'altra parteix de la fila inicial de les negres. Per tant, la fila que, en el sistema algebraic és la 7, en el sistema descriptiu és la 7 per a les blanques, però la 2 per a les negres; d'igual manera la fila 3 del sistema algebraic, en el descriptiu és la fila 3 de les blanques, però la 6 de les negres. En les webs i el material imprès -llibres, revistes i notícies de diari- la fila de l'extrem inferior del tauler és la 1 de les blanques i la 8 de les negres, de la mateixa manera, la fila de l'extrem superior del tauler és la 8 de les blanques i 1 de les negres. Per comprendre-ho millor, ho exposem en aquesta taula:

Taula 2.5.3.2

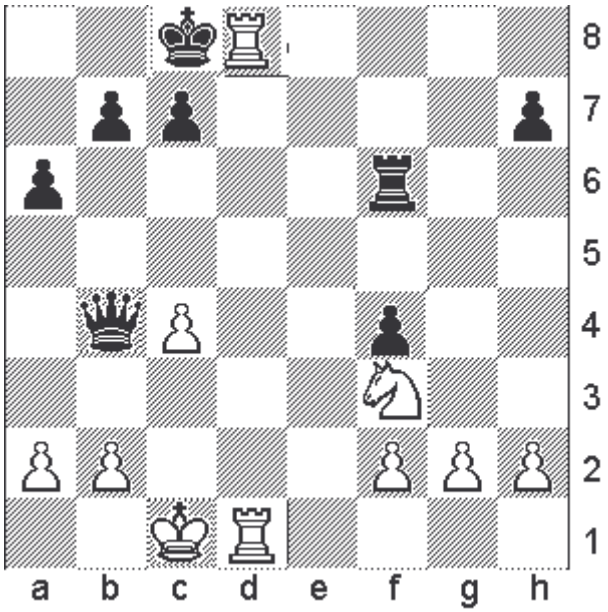
NUMERACIÓ DE CADA FILA SEGONS EL SISTEMA ALGEBRAIC	NUMERACIÓ DE CADA FILA SEGONS EL SISTEMA DESCRIPTIU COMPTANT DES DE LES BLANQUES	NUMERACIÓ DE CADA FILA SEGONS EL SISTEMA DESCRIPTIU COMPTANT DES DE LES NEGRES
1	1	8
2	2	7
3	3	6
4	4	5
5	5	4
6	6	3
7	7	2
8	8	1

A diferència del sistema algebraic, en el sistema descriptiu, apareix sempre la inicial de la peça moguda, fins i tot si es tracta d'un peó; i la casella es designa posant primer la xifra de la fila i després la inicial de la columna en majúscula. En les jugades de les blanques, les files es numeren segons la numeració de les blanques, i en les jugades de les negres, les files es numeren segons la numeració de les negres. Ara, a continuació, mostrarem una partida escrita en sistema descriptiu.

Partida 2.5.3.1
Blanques Negres
1.P4R P4R
(En el sistema algebraic, hauríem escrit 1.e4 e5)
2.P4D PxP
(En el sistema descriptiu després del símbol de captura s'hi posa la inicial de la peça capturada)
3.P3AD PxP
4.A4AD PxP
5.AxP C3AR
6.C3AD A5C
(En teoria, hauria calgut escriure A5CD, però com que la jugada A5CR és impossible en aquesta posició, n'hi ha prou escrivint A5C)
7.C2R CxP
8.O-O CxC
9.CxC AxC
10.AxA D4C
11.T1R+ R1D

12.P4A DxP
13.AXPC T1C
(Amb l'anotació de la jugada 13 de les blanques, indiquem que el peó negre capturat per l'alfil blanc és el situat a la casella dita 7CR comptant des de les blanques, i 2CR, comptant des de les negres. Així també sabem que l'alfil blanc que efectua la captura és el situat a la casella dita 3AD per les blanques i 6AD per les negres)
14.D4C DxD
15 A6A++ 1-0

DIAGRAMA 2.5.3.2
posició final de la partida 2.5.3.2



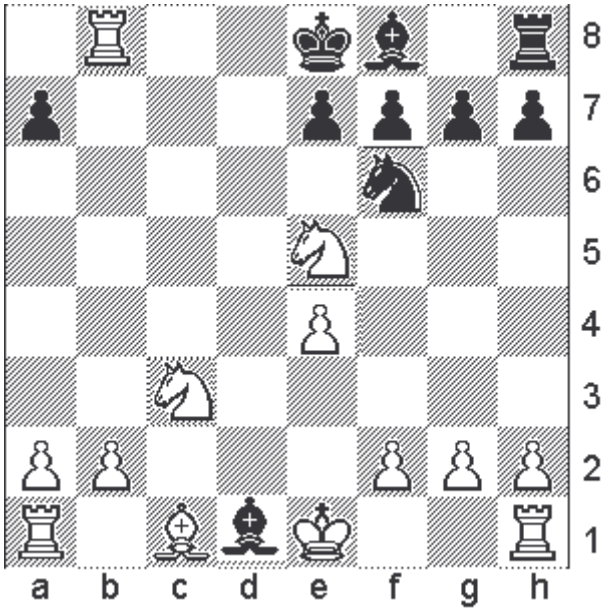
2.5.4. La representació lineal de la partida

Fins ara, ací s'han representat les partides posant-ne les jugades en columna; també és possible fer-ho de forma lineal; aleshores, la jugada mencionada en primer lloc després del número correspon a les blanques i la que va al costat, a les negres. Ara mostrem una partida escrita en forma lineal :

Partida 2.5.4.1

1d4,d5; 2c4,Cf6; 3c4xd5,Cxd5; 4e4,Cf6; 5Cc3,Cc6; 6Cf3,Ag4; 7d5,Ce5; 8Cxe5,Axd1; 9Ab5+,c6; 10d5xc6,Db6; 11c6xb7+,Dxb5; 12b7xa8=T,Db8; 13Txb8++ 1-0

DIAGRAMA 2.5.4.1
posició final de la partida 2.5.4.1



Tant en el sistema algebraic com en el descriptiu, un moviment de les blanques sempre es menciona al costat de la xifra d'ordre de la jugada dins de la partida; si el moviment és de les negres, apareix després de la xifra seguida per tres punts suspensius. En conseqüència, si veiem escrit 14.Ab5, aquesta és una jugada de les blanques, mentre que 14...Tg5 és un moviment de les negres; aquest mateix criteri val per a 1.P4R i 1...P4AD

2.5.5. Ús de cadascun dels sistemes d'anotació

Avui dia, cada vegada s'imposa més el sistema algebraic en detriment del descriptiu, fins al punt que tots els llibres i revistes escrits actualment fan servir el sistema algebraic. Ara bé, fins a mitjan de la dècada de 1980, a l'Estat espanyol, als països hispans i als de parla anglesa hi predominava el descriptiu. En competicions oficials es pot usar qualsevol dels dos, encara que, progressivament, es va fent servir més l'algebraic. Ara bé, cal conèixer el sistema descriptiu perquè encara queden molts llibres que hi estan escrits. Molts diaris tenen una secció on hi apareix un problema o una partida d'escacs; en el cas de la premsa barcelonina, tots els diaris fan servir l'algebraic, mentre que l'*AVUI* continua fent servir el descriptiu; *La Vanguardia* abans feia servir el descriptiu però des de ja fa uns quants anys usa l'algebraic, cosa que també pot dir-se de *El País*.

3. Especificació dels requeriments

Si ens basem en l'explicació del joc podem observar que tenim moltes restriccions i especificacions, així que ho dividiré en subapartats per tal de que quedi més clar.

3.1. La partida

La partida d'escacs està basada en la interacció entre dos jugadors, per tant sabem que únicament hi poden haver dos jugadors i que han d'interactuar un a un o sigui que s'han de definir uns torns de joc per a cada jugador.

La partida s'inicia amb les fitxes ordenades tal qual hem pogut llegir a l'apartat anterior i amb el torn a favor del jugador que controla les fitxes blanques.

Altres aspectes que volem controlar, poden ser el número de moviment actual, el nivell del jugador, que segons sigui amateur o bé un professional, se li podria permetre desfer moviments o no.

Sabem que la partida s'acaba quan un dels dos jugadors perd el Rei o bé quan es fan les taules, tal com podem llegir al apartat anterior.

3.2. El taulell

El taulell de joc a simple vista és una matriu de vuit per vuit quadrats, alternativament pintats d'un color clar i un fosc; aquestes caselles són quadrats exactes i per tant tindran una mida concreta, un atribut que ens indicarà el seu color, també necessitem saber si la casella és buida o per el contrari quina peça hi te a sobre.

3.3. Les Peces

Les peces d'aquest joc ens permeten enregistrar tot un seguit de característiques comuns i un grup de característiques particulars. En comú, hem de guardar el color de la peça, la seva imatge, el seu tipus, si és el primer cop que es mou i el seu valor numèric, que farem servir per tal de calcular la funció d'avaluació per tal que l'aplicació "pensi"

Hem de controlar o limitar els moviments de les peces ja que no podem deixar que una peça es mogui fora del tauler ni faci un moviment il·legal.

3.3.1. Peons

Els peons tenen la característica de que el seu moviment normal és d'una casella, i sempre endavant, o sigui que si ens mirem el tauler com una matriu el que hem de controlar es que cada peó sempre que es vulgui moure, sigui en una posició d'igual columna però de +1 fila en el cas de les blanques i -1 fila en el cas de les negres. L'anterior, seria en el cas d'un moviment normal però com a l' introducció hem vist, els

peons també es poden moure dues caselles el primer cop que es mouen i aquí és on entra en acció el camp que ens controla si la peça mai s'ha mogut.
En el cas de que un peó faci una captura, hem de controlar que la captura sigui correcta, en el cas de les blanques fent +1 fila i +1 o -1 columna, llavors tenim el cas específic de la captura al pas que hem pogut veure en l' introducció i que més endavant controlarem.
Un altre aspecte important, es que passa quan un peó arriba a l'ultima casella del tauler, doncs hem de controlar que si aquesta situació es dona, hem de permetre que l'usuari triï un tipus de peça que no sigui rei, i l'intercanviarem per el peó que ha arribat al final.

3.3.2. Torres

Les Torres tenen la mateixa manera de moure's que de capturar, es mouen tant horitzontalment com verticalment i sense més límit que la casella a la que volem arribar, capturar o el límit del tauler. El que hem de controlar es que quan es mou la torre, ens moguem o bé sobre la mateixa fila o sobre la mateixa col·lumna.

3.3.3. Cavall

Ens trobem davant de l'única peça que pot saltar altres peces i que a més es mou amb un moviment en forma de L, hem de comprovar que el seu moviment sigui de 2 caselles en horitzontal i una en vertical o viceversa.

3.3.4. Alfils

Els alfils ens limiten el moviment a les diagonals, per tant el que hem de comprovar es que el desplaçament es dugi a fila+n i columna+n o bé filar-ne i columna-1.

3.3.5. Reina

La reina es una peça que es comporta com una barreja entre la torre i l'alfil, es pot moure tant en horitzontal com en vertical i en les diagonals, per tant a l'hora de controlar els seus moviments ens basarem en barrejar les normes de la torre amb les de l'alfil.
3.3.6 Rei
El rei és la peça més important per el joc ja que sense rei, el jugador ha perdut, però a l'hora de ficar-li restriccions de moviment, simplement hem de agafar les restriccions que te la reina però limitant-li el moviment a una sola casella.
També és important en aquesta peça controlar que mai es pot moure a una casella en la que estarà amenaçada de captura, i en el pitjor dels casos si no es pot moure enlloc, la partida es declararà perduda.

4. Representació del coneixement

Per tal de representar tot el coneixement que tenim fins ara hem de comprendre que el sistema que estem analitzant dins del camp de l' intel·ligència artificial, és un sistema expert. I aquest sistema expert forma part dels sistemes intel·ligents.

Sistema intel·ligent: Sistema informàtic que compleix els següents requeriments:

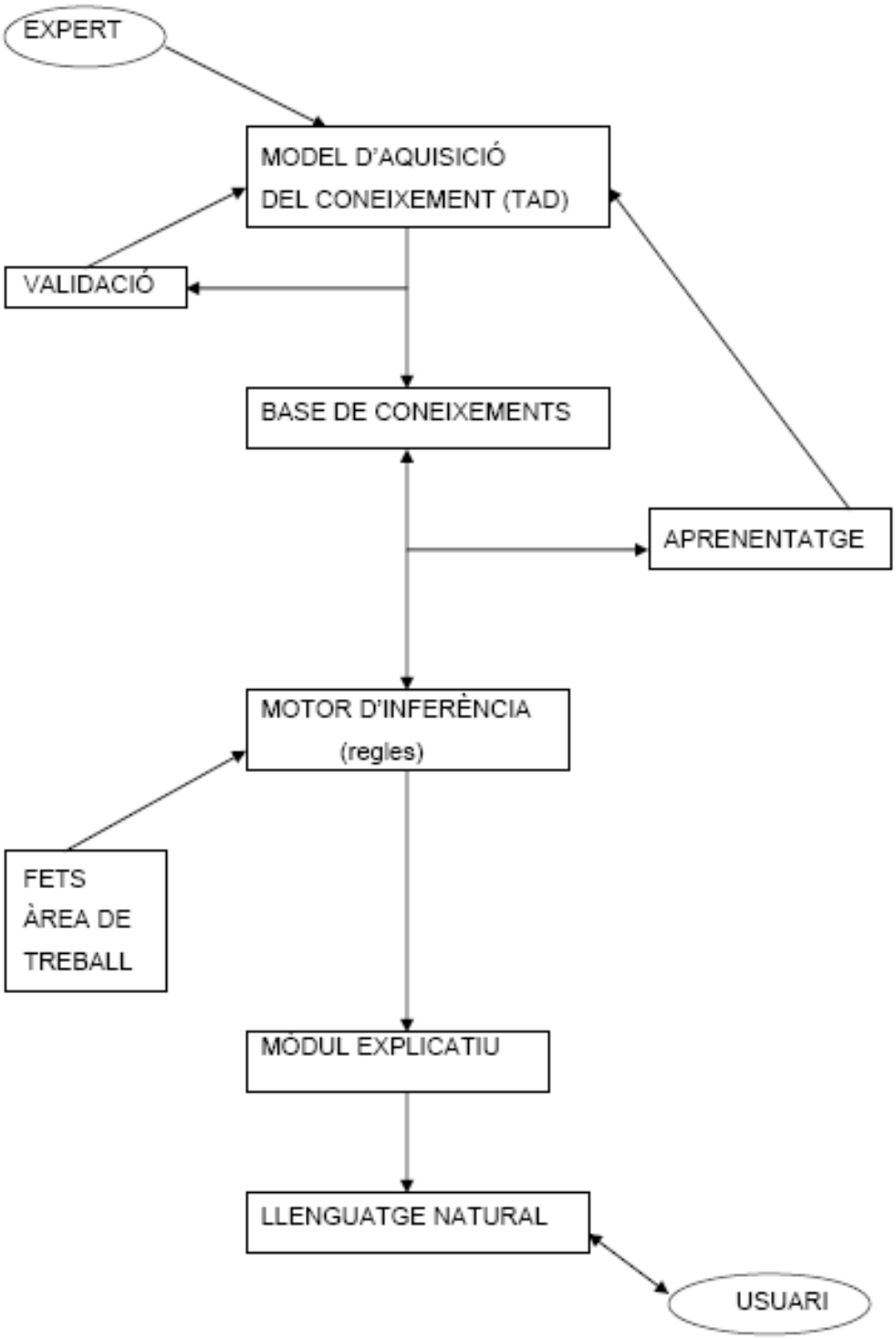
· Aprèn del seu entorn
· Opera amb temps real
· Treballa amb abstraccions simbòliques del coneixement
· Manipula grans volums de dades
· Es comunica amb llenguatge natural.

Un sistema expert és un subconjunt dels sistemes intel·ligents, es podria dir la vessant més comercial de la intel·ligència artificial, que representa la modulació d'un ésser humà a nivell informàtic. Cal dir que són sistemes intel·ligents que treballen en dominis molt concrets, complexos i poc estructurats.
Aquests sistemes tenen unes avantatges i uns inconvenients que es mesuren lligant
amb el que pot fer un expert humà.
Es descriuen a continuació:

Avantatges	Inconvenients
Són capaços de donar diagnòstics.	Treballen només en un domini concret.
Són capaços de fer classificacions.	Dificultat de pas d'informació entre expert i sistema.
Són fàcils de transportar.	No tenen capacitat de planificació.
És fàcil de fer-ne còpies.	

La manera de treballar d'un sistema expert es veu en el següent esquema:

DIAGRAMA 4.1



Els sistemes intel·ligents treballen amb grans volums de dades (coneixement del domini). Llavors interessa poder representar aquest coneixement per tal de poder-lo manipular i fer raonaments.

Els aspectes a tenir en compte a l'hora de representar coneixement són:

a) Quin és el coneixement que es necessita, doncs tenir coneixement innecessari suposa un procés més feixuc de manipulació de dades.

b) Quin mecanisme s'utilitza per la representació (dependrà molt del coneixement del sistema i de les eines de treball que es disposin).

c) Com adquirir el coneixement.

Tenint en compte aquests aspectes s'ha de pensar en mecanismes de representació del coneixement que no sols guardin la informació sinó que també facilitin la manipulació d'aquesta per obtenir la solució del problema a resoldre.

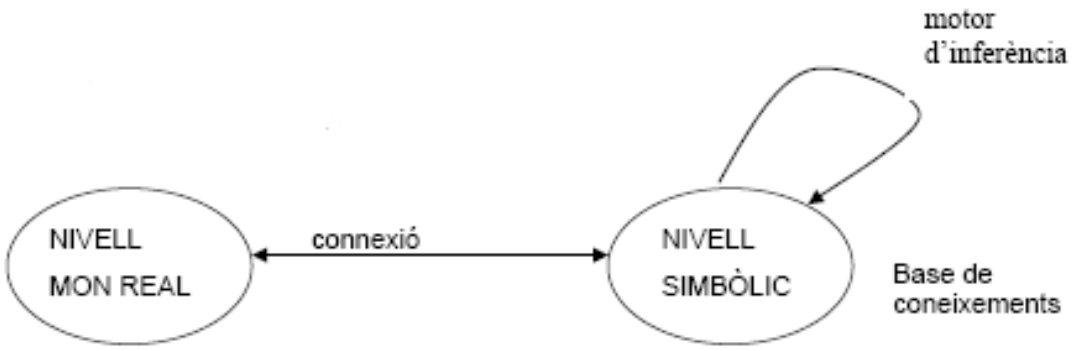
Quan es fa representació del coneixement sempre es treballa a 2 nivells de representació**: Nivell mon real i nivell simbòlic.**

Nivell mon real: equival a tot el coneixement real que es té del domini en que s'està treballant.

Nivell simbòlic: equival a la representació simbòlica de tot el coneixement del nivell anterior per tal de poder ser processat informàticament.

Si esquematitzem aquests dos nivells:

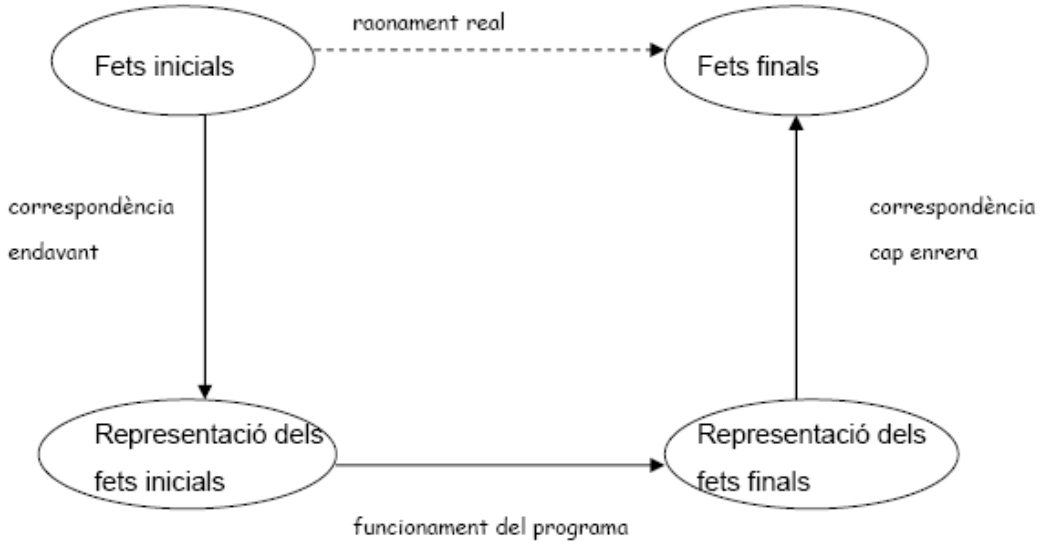
DIAGRAMA 4.2



A nivell mon real tots els fets es guarden en llenguatge natural, llavors es necessita una connexió que faci una abstracció i els representi a nivell simbòlic. Un cop s'han obtingut resultats també es necessita una connexió perquè els tradueixi a llenguatge natural.

Agafant l'esquema anterior i ampliant-lo una mica, s'obté:

DIAGRAMA 4.3



Tot això sembla molt senzill, si es coneixen els mecanismes de representació del coneixement, però a la realitat no ho és tant doncs hi ha tota una sèrie de **problemes** que apareixen a l'hora de fer representació del coneixement. Aquests són:

Ambigüitat referencial: Es fa referència a un mateix objecte però fent servir diferents paraules, el llenguatge natural permet anomenar objectes amb diferents paraules (sinònims). Per defecte el llenguatge natural és ambigu. Per exemple: metge, doctor, facultatiu,..
Ambigüitat lèxica: Fa referència a diferents interpretacions que es poden fer d'un coneixement donat en llenguatge natural. Per exemple: En Joan veu a en Pere amb uns prismàtics dalt de la muntanya. Ens pregunten: Qui porta els prismàtics? Qui és dalt de la muntanya?.

Exemples enfront classes: Saber distingir en la informació donada entre què representa una tipologia d'objectes i què representa un objecte concret és dificultós, però interessa molt poder agrupar sempre que es pugui objectes semblants, és a dir treballant sobre conjunt i no sobre individuals.

Herència: L'herència permet reconèixer característiques dels objectes sense haver d'explicitar-les. Pensant que en IA es tracten grans volums de dades, quan més herència es pugui aplicar millor que millor, doncs menys coneixement a explicitar.

Especificació: No sempre és fàcil obtenir quin és el nivell de detall dels objectes que es volen representar.

Redundància: Tenint en compte tots els problemes anteriors, s'ha d'evitar la redundància d'informació, doncs tenir redundància equival a tenir un cost computacional molt alt.

Per tal de conèixer els diferents tipus de representació del coneixement que hi ha, podem veure aquestes dues definicions:

Representació declarativa: En aquests sistemes la representació del coneixement es basa en un conjunt de fets on s'explica el coneixement que es té del domini, però en cap moment s'indica com s'ha d'usar, per tant aquest tipus de representació sempre va acompanyada d'un programari que especifica el què i el com s'ha de fer amb el coneixement que es té.

Representació procedural: En aquests sistemes la representació no solament hi inclou el coneixement que es té sobre el domini, sinó que també inclou els mecanismes necessaris per saber com usar aquest coneixement.

Per tant, per representar el coneixement del nostre joc d'escacs, farem servir la representació declarativa i concretament el tipus Sistema de Producció, aquests són un tipus de sistemes de representació del coneixement molt usats en programes d'intel·ligència artificial, sobretot en sistemes experts.
Són sistemes de tipus modular que ajuden a guiar, a donar recomanacions, a donar estratègies per trobar la solució,.... .

Tot sistema de producció consta de:

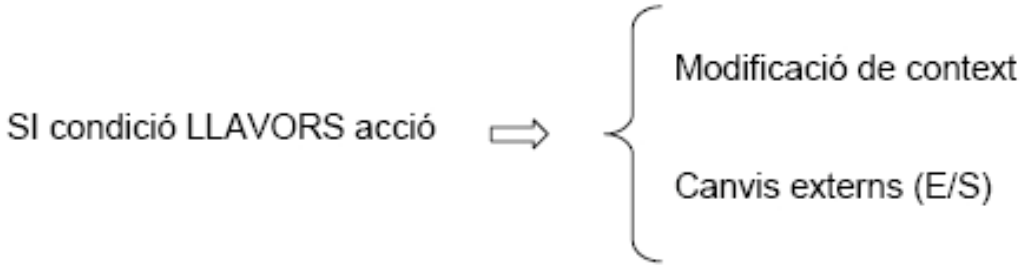
BASE DE CONEIXEMENTS, on s'hi registra tot el coneixement que es té del domini.

Consta de dues components:

BASE DE FETS, on s'hi representa tot el coneixement que és cert de l'estat actual del problema. El seu contingut és canviant en funció de l'estat en que es trobi el problema. La base de fets permet optimitzar el conjunt de regles. Per tal de fer complir les regles, les condicions d'aquestes s'han de trobar dins la base de fets.

CONJUNT DE REGLES, és la component fonamental de la base de coneixements. En aquest si representa tot el coneixement que es té del domini.
Tal i com el seu nom indica està format per un conjunt de regles anomenades **regles de producció**. En aquestes regles tant es pot representar coneixement declaratiu com procedural. Aquestes regles es formalitzen mitjançant càlcul de predicats (un subconjunt d'aquest).
L'estructura d'aquestes regles és:

DIAGRAMA 4.4



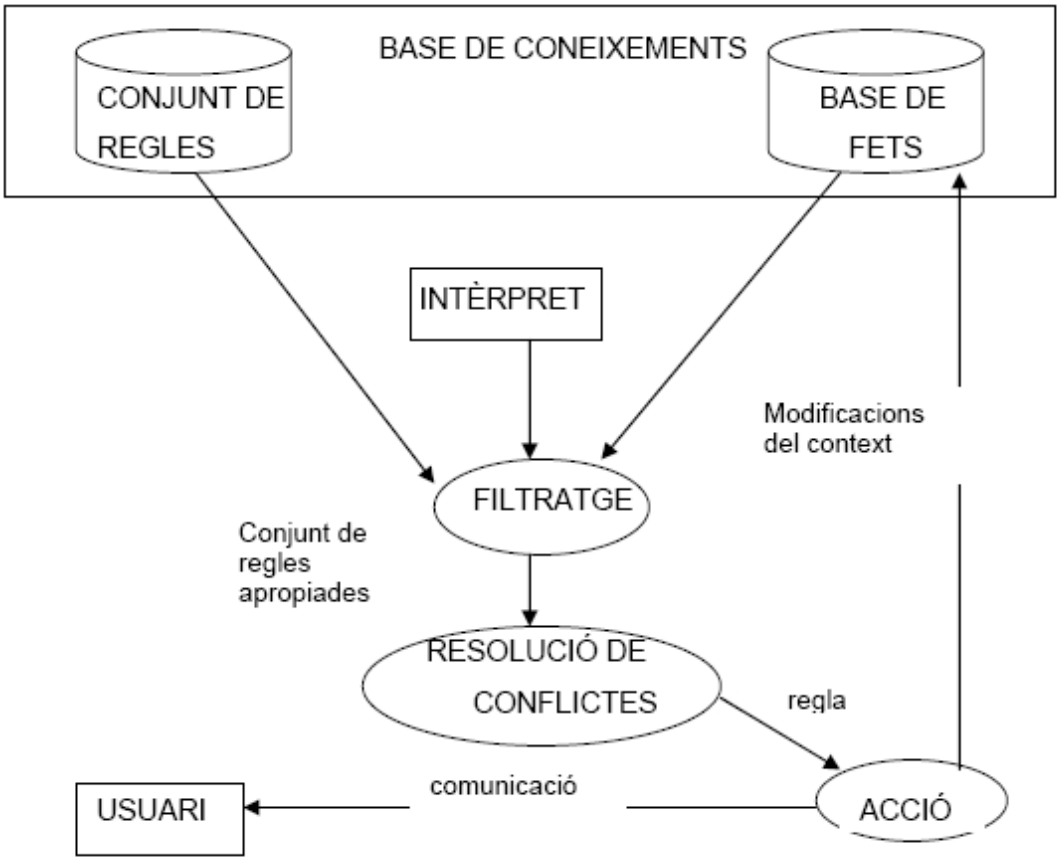
El sistema de producció mira si les condicions de la regla es satisfan o no. En cas de resposta positiva llavors tindrà en consideració la possibilitat d'executar les accions que tingui associada.

Tal i com s'ha anat esmentant el coneixement que es té d'un domini és molt difícil de que sigui complet. Llavors a les regles se'ls associa un grau de certesa que sol ser numèric.

MOTOR D'INFERÈNCIA, és el mecanisme (software) que controla el funcionament del sistema. Aquest decideix que s'ha de fer en cada moment.

El funcionament més simple d'un sistema de producció (el cicle d'execució) es pot veure en el següent esquema:

DIAGRAMA 4.5



Explicant aquest esquema:

- Es parteix d'un estat inicial que es troba dins la base de fets.
- L'intèrpret examina les regles i seguint un ordre mira quines són les possibles aplicar.
- Un cop examinades el mateix intèrpret mitjançant resolució de conflictes decideix quina és la millor a aplicar.
- S'executa la regla. Això pot comportar modificacions de l'estat del problema, que pot implicar la possibilitat d'aplicar noves regles.

Quines són les avantatges d'aquests sistemes i quins en són els inconvenients?

AVANTATGES	**INCONVENIENTS**
Modularitat	Ineficiència
Naturalitat	No universals
Explicatius	No verificació de consistència i Completesa

4.1. Conjunt de regles

En aquest apartat hi incloure la descripció de les regles que seguirà el joc d'escacs, fent servir lògica de predicats. Entre elles trobarem les regles de moviment de peça i les de finalització de partida.

Podem trobar els següents predicats:

P(x)→x és un peó

T(x)→x és una torre

C(x)→x és un cavall

A(x)→x és un alfil

RE(x)→x és una reina

R(x)→x és un rei

B(x)→x és una peça banca

BU(x)→x és una casella buida

POS(x,y,i,j)→La posició y està i caselles per sobre i j caselles cap a la dreta de x

E(x,y)→Hi han caselles ocupades entre x i y

M(x,y)→És possible moure de x a y

I l'objecte PT→Partida.

4.1.1. Moviment de les peces

4.1.1.1. Moviments dels peons

- ∀x,y(P(x)^POS(x,y,1,0)^B(x)^BU(y)→M(x,y))
- ∀x,y(P(x)^POS(x,y,2,0)^¬MA(x)^B(x)^BU(y)→M(x,y))
- ∀x,y(P(x)^POS(x,y,-1,0)^¬B(x)^BU(y)→M(x,y))
- ∀x,y(P(x)^POS(x,y,-2,0)^¬MA(x)^¬B(x)^BU(y)→M(x,y))
- ∀x,y(P(x)^POS(x,y,1,1)^B(x)^¬B(y)→M(x,y))
- ∀x,y(P(x)^POS(x,y,1,-1)^B(x)^¬B(y)→M(x,y))
- ∀x,y(P(x)^POS(x,y,-1,1)^¬B(x)^B(y)→M(x,y))
- ∀x,y(P(x)^POS(x,y,-1,-1)^¬B(x)^B(y)→M(x,y))

4.1.1.2. Moviment de les torres

- ∀x,y(T(x)^ POS(x,y,0,n)^B(x)^¬B(y)^¬E(x,y)→M(x,y))
- ∀x,y(T(x)^ POS(x,y,n,0)^B(x)^¬B(y)^¬E(x,y)→M(x,y))
- ∀x,y(T(x)^ POS(x,y,0,n)^¬B(x)^B(y)^¬E(x,y)→M(x,y))
- ∀x,y(T(x)^ POS(x,y,n,0)^¬B(x)^B(y)^¬E(x,y)→M(x,y))

4.1.1.3. Moviment dels cavalls

- $\forall x,y(C(x) \wedge POS(x,y,2,1) \wedge B(x) \wedge \neg B(y) \rightarrow M(x,y))$
- $\forall x,y(C(x) \wedge POS(x,y,2,-1) \wedge B(x) \wedge \neg B(y) \rightarrow M(x,y))$
- $\forall x,y(C(x) \wedge POS(x,y,-2,1) \wedge B(x) \wedge \neg B(y) \rightarrow M(x,y))$
- $\forall x,y(C(x) \wedge POS(x,y,-2,-1) \wedge B(x) \wedge \neg B(y) \rightarrow M(x,y))$
- $\forall x,y(C(x) \wedge POS(x,y,1,2) \wedge B(x) \wedge \neg B(y) \rightarrow M(x,y))$
- $\forall x,y(C(x) \wedge POS(x,y,1,-2) \wedge B(x) \wedge \neg B(y) \rightarrow M(x,y))$
- $\forall x,y(C(x) \wedge POS(x,y,-1,2) \wedge B(x) \wedge \neg B(y) \rightarrow M(x,y))$
- $\forall x,y(C(x) \wedge POS(x,y,-1,-2) \wedge B(x) \wedge \neg B(y) \rightarrow M(x,y))$
- $\forall x,y(C(x) \wedge POS(x,y,2,1) \wedge \neg B(x) \wedge (B(y) \vee BU(x)) \rightarrow M(x,y))$
- $\forall x,y(C(x) \wedge POS(x,y,2,-1) \wedge \neg B(x) \wedge (B(y) \vee BU(x)) \rightarrow M(x,y))$
- $\forall x,y(C(x) \wedge POS(x,y,-2,1) \wedge \neg B(x) \wedge (B(y) \vee BU(x)) \rightarrow M(x,y))$
- $\forall x,y(C(x) \wedge POS(x,y,-2,-1) \wedge \neg B(x) \wedge (B(y) \vee BU(x)) \rightarrow M(x,y))$
- $\forall x,y(C(x) \wedge POS(x,y,1,2) \wedge \neg B(x) \wedge (B(y) \vee BU(x)) \rightarrow M(x,y))$
- $\forall x,y(C(x) \wedge POS(x,y,1,-2) \wedge \neg B(x) \wedge (B(y) \vee BU(x)) \rightarrow M(x,y))$
- $\forall x,y(C(x) \wedge POS(x,y,-1,2) \wedge \neg B(x) \wedge (B(y) \vee BU(x)) \rightarrow M(x,y))$
- $\forall x,y(C(x) \wedge POS(x,y,-1,-2) \wedge \neg B(x) \wedge (B(y) \vee BU(x)) \rightarrow M(x,y))$

4.1.1.4. Moviment dels alfils

- $\forall x,y(A(x) \wedge POS(x,y,n,n) \wedge B(x) \wedge \neg B(y) \rightarrow M(x,y))$
- $\forall x,y(A(x) \wedge POS(x,y,n,-n) \wedge B(x) \wedge \neg B(y) \rightarrow M(x,y))$
- $\forall x,y(A(x) \wedge POS(x,y,-n,n) \wedge B(x) \wedge \neg B(y) \rightarrow M(x,y))$
- $\forall x,y(A(x) \wedge POS(x,y,-n,-n) \wedge B(x) \wedge \neg B(y) \rightarrow M(x,y))$
- $\forall x,y(A(x) \wedge POS(x,y,n,n) \wedge \neg B(x) \wedge (B(y) \vee BU(x)) \rightarrow M(x,y))$
- $\forall x,y(A(x) \wedge POS(x,y,n,-n) \wedge \neg B(x) \wedge (B(y) \vee BU(x)) \rightarrow M(x,y))$
- $\forall x,y(A(x) \wedge POS(x,y,-n,n) \wedge \neg B(x) \wedge (B(y) \vee BU(x)) \rightarrow M(x,y))$
- $\forall x,y(A(x) \wedge POS(x,y,-n,-n) \wedge \neg B(x) \wedge (B(y) \vee BU(x)) \rightarrow M(x,y))$

4.1.1.5. Moviment de la reina

- $\forall x,y(RE(x) \wedge POS(x,y,0,n) \wedge B(x) \wedge \neg B(y) \wedge \neg E(x,y) \rightarrow M(x,y))$
- $\forall x,y(RE(x) \wedge POS(x,y,n,0) \wedge B(x) \wedge \neg B(y) \wedge \neg E(x,y) \rightarrow M(x,y))$
- $\forall x,y(RE(x) \wedge POS(x,y,n,n) \wedge B(x) \wedge \neg B(y) \rightarrow M(x,y))$
- $\forall x,y(RE(x) \wedge POS(x,y,n,-n) \wedge B(x) \wedge \neg B(y) \rightarrow M(x,y))$
- $\forall x,y(RE(x) \wedge POS(x,y,-n,n) \wedge B(x) \wedge \neg B(y) \rightarrow M(x,y))$
- $\forall x,y(RE(x) \wedge POS(x,y,-n,-n) \wedge B(x) \wedge \neg B(y) \rightarrow M(x,y))$
- $\forall x,y(RE(x) \wedge POS(x,y,n,n) \wedge \neg B(x) \wedge (B(y) \vee BU(x)) \rightarrow M(x,y))$
- $\forall x,y(RE(x) \wedge POS(x,y,n,-n) \wedge \neg B(x) \wedge (B(y) \vee BU(x)) \rightarrow M(x,y))$
- $\forall x,y(RE(x) \wedge POS(x,y,-n,n) \wedge \neg B(x) \wedge (B(y) \vee BU(x)) \rightarrow M(x,y))$
- $\forall x,y(RE(x) \wedge POS(x,y,-n,-n) \wedge \neg B(x) \wedge (B(y) \vee BU(x)) \rightarrow M(x,y))$
- $\forall x,y(RE(x) \wedge POS(x,y,0,n) \wedge \neg B(x) \wedge B(y) \wedge \neg E(x,y) \rightarrow M(x,y))$
- $\forall x,y(RE(x) \wedge POS(x,y,n,0) \wedge \neg B(x) \wedge B(y) \wedge \neg E(x,y) \rightarrow M(x,y))$

4.1.1.6. Moviment del rei

- $\forall x,y(RE(x) \wedge POS(x,y,0,1) \wedge B(x) \wedge \neg B(y) \wedge \neg E(x,y) \rightarrow M(x,y))$
- $\forall x,y(RE(x) \wedge POS(x,y,1,0) \wedge B(x) \wedge \neg B(y) \wedge \neg E(x,y) \rightarrow M(x,y))$
- $\forall x,y(RE(x) \wedge POS(x,y,1,1) \wedge B(x) \wedge \neg B(y) \rightarrow M(x,y))$
- $\forall x,y(R(x) \wedge POS(x,y,1,-1) \wedge B(x) \wedge \neg B(y) \rightarrow M(x,y))$
- $\forall x,y(R(x) \wedge POS(x,y,-1,1) \wedge B(x) \wedge \neg B(y) \rightarrow M(x,y))$
- $\forall x,y(R(x) \wedge POS(x,y,-1,-1) \wedge B(x) \wedge \neg B(y) \rightarrow M(x,y))$
- $\forall x,y(R(x) \wedge POS(x,y,1,1) \wedge \neg B(x) \wedge (B(y) \vee BU(x)) \rightarrow M(x,y))$
- $\forall x,y(R(x) \wedge POS(x,y,1,-1) \wedge \neg B(x) \wedge (B(y) \vee BU(x)) \rightarrow M(x,y))$
- $\forall x,y(R(x) \wedge POS(x,y,-1,1) \wedge \neg B(x) \wedge (B(y) \vee BU(x)) \rightarrow M(x,y))$
- $\forall x,y(R(x) \wedge POS(x,y,-1,-1) \wedge \neg B(x) \wedge (B(y) \vee BU(x)) \rightarrow M(x,y))$
- $\forall x,y(R(x) \wedge POS(x,y,0,1) \wedge \neg B(x) \wedge B(y) \wedge \neg E(x,y) \rightarrow M(x,y))$
- $\forall x,y(R(x) \wedge POS(x,y,1,0) \wedge \neg B(x) \wedge B(y) \wedge \neg E(x,y) \rightarrow M(x,y))$

4.1.2. Final de partida

- $\forall x(\neg(R(x) \wedge B(x)) \vee \neg(R(x) \wedge \neg B(x)) \rightarrow \neg PT)$

4.2. Base de Coneixements

La base de coneixements està composta per la base de fets més el coneixement del domini.

4.2.1. Base de fets

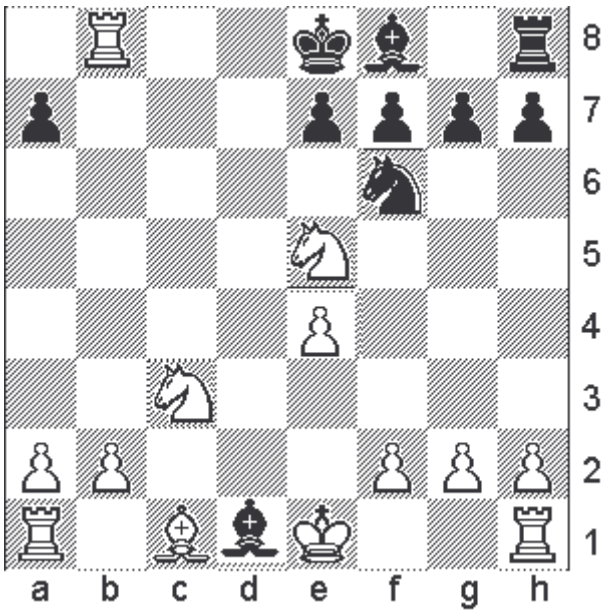
Com que la base de fets conté tot el coneixement real, en la meva aplicació, el que contindrà és la situació actual del tauler en tot moment. Aquesta informació serà guardada en una unitat d'informació anomenada Tauler, però d'això ja en parlarem més endavant.

4.2.2. Coneixement del domini

En aquest cas el coneixement del domini s'assigna a la creació de la funció d'avaluació, aquesta funció el que fa és retornar-nos un valor que pot ser assignat al tauler en un moment donat. Per tant podríem dir que simplement és una funció que "mirant" la base de fets, li assigna un valor.

Una funció d'avaluació clàssica i molt utilitzada és la que compta peces, les compta en positiu si les peces corresponen al jugador que utilitza aquesta funció, i en negatiu si són peces de l'oponent.
Com a exemple podem observar el següent tauler:

DIAGRAMA 4.2.2.1



Si li apliquéssim la funció d'avaluació abans explicada per el jugador blanc, el valor del tauler seria 13 – 10 = 3 i per tant al ser mes gran que 0, seria bona per el jugador blanc. Però en el cas de les escacs, aquesta funció no acaba de ser valida ja que per començar, el rei el compta com una peça normal i podria ser que un jugador perdés el rei i en canvi tingues més peces que el contrincant i per tant la funció d'avaluació li seguiria retornant un positiu mentre que ja hauria perdut.
Aquest problema ens condueix cap a una solució molt evident que és assignar-li un pes més elevat al rei, per exemple fent que en comptes de valer una unitat, en valgués 100. Amb aquest canvi ens asseguraríem de que el valor d'un tauler sense rei seria suficientment baix com per que el sistema intel·ligent no el tries.

Seguint el mateix exemple però amb la nova funció d'avaluació tindríem que el valor de tauler seria de 112 – 109 = 3, el mateix resultat que abans però, si el jugador perdés el rei negre, el resultat seria de 12 – 109 = -97, un resultat tant negatiu, que seria el últim a ser triat.
Amb aquesta funció d'avaluació ja ens podríem assegurar de que el sistema seria prou "intel·ligent" com per guanyar partides, però per una persona esta clar que no és el mateix un peó que una reina i com que el que perseguim és que el sistema o l'aplicació tingui un "pensament" el més semblant a una persona real, el que podem fer és assignar-li certs pesos a les diferents peces fent així que un peó sigui molt menys valuós que un alfil i aquest menys valuós que una reina.
Un possible exemple dels pesos que poden tenir aquestes peces podria ser el següent:

TAULA 4.2.2.1

PEÇA	VALOR
PEÓ	1
TORRE	10
CAVALL	50
ALFÍL	30
REINA	100
REI	1000

Si apliquéssim aquesta funció al exemple abans analitzat, obtenim la següent numeració, 1156 – 1125 = 31, el que ens permet contemplar molt millor les diferencies entre un moviment o un altre i, com abans he comentat, permetrà que l'aplicació pugui "raonar" d'una manera més humana ja que fins hi tot podria sacrificar una peça d'inferior valor per tal de capturar una peça del contrincant d'alt valor.
Però, es podria millorar aquesta funció? La resposta és si, és evident que els humans no donem la mateixa importància a una peça com el cavall si aquesta es situa cap al centre del tauler, que pot realitzar molts moviments diferents, que si la mateixa peça es situa al marge del tauler que podrà fer uns moviments molt més limitats. Per tant la solució que milloraria la funció d'avaluació seria fer que a part de que tinguessin pesos diferents les peces, també en tinguessin diferent pes segons la seva posició.
Amb els següents diagrames, podríem obtenir els pesos de les peces segons la seva posició:

Pes segons posició dels peons:

TAULA 4.2.2.2

5	5	5	5	5	5	5	5
4	4	4	4	4	4	4	4
3	3	3	2	2	3	3	3
3	3	3	2	2	3	3	3
3	3	3	2	2	3	3	3
2	2	2	2	2	2	2	2
1	1	1	1	1	1	1	1
0	0	0	0	0	0	0	0

Pes segons posició de les torres:

TAULA 4.2.2.3

15	15	15	15	15	15	15	15
15	20	20	20	20	20	20	15
15	20	20	20	20	20	120	15
15	20	20	30	30	20	20	15
15	20	20	30	20	20	20	15
15	20	20	20	20	20	20	15
15	20	20	20	20	20	20	15
15	15	15	15	15	15	15	15

Pes segons posició dels cavalls:

TAULA 4.2.2.4

50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	60	60	60	60	50	50
50	50	60	80	80	60	50	50
50	50	60	80	80	60	50	50
50	50	60	60	60	60	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

Pes segons posició dels alfils:

TAULA 4.2.2.5

30	35	35	35	35	35	35	30
35	40	40	40	40	40	40	35
35	40	45	45	45	45	40	35
35	40	45	45	45	45	40	35
35	40	45	45	45	45	40	35
35	40	45	45	45	45	40	35
35	40	40	40	40	40	40	35
30	35	35	35	35	35	35	30

Pes segons posició de la reina:

TAULA 4.2.2.6

100	115	115	115	115	115	115	100
115	130	130	130	130	130	130	115
115	130	140	140	140	140	130	115
115	130	140	150	150	140	130	115
115	130	140	150	150	140	130	115
115	130	140	140	140	140	130	115
115	130	130	130	130	130	130	115
100	120	120	120	120	120	120	100

Pes segons posició del rei:

TAULA 4.2.2.7

1000	1000	1000	1000	1000	1000	1000	1000
1000	1000	1000	1000	1000	1000	1000	1000
1000	1000	1000	1000	1000	1000	1000	1000
1000	1000	1000	1000	1000	1000	1000	1000
1000	1000	1000	1000	1000	1000	1000	1000
1000	1000	1000	1000	1000	1000	1000	1000
1000	1000	1000	1000	1000	1000	1000	1000
1000	1000	1000	1000	1000	1000	1000	1000

Com es pot comprovar, als diagrames només es contempla el pes de les peces blanques, per obtenir el pes de les peces negres simplement s'haurien de girar els diagrames.

Per tal de fer que el programa sigui "intel·ligent" podria triar qualsevol de les dues ultimes funcions d'avaluació, això dependrà en tot cas del nivell d'avaluació que volem que es porti a terme per l'algorisme que triarà la millor jugada ja que si es tracta d'un algorisme que examini molts nivells de jugades, en tenim prou amb l'algorisme que es dedica a comptar per pesos ja que abans o després trobarà la jugada que li portarà a la victòria, en canvi si no volem que examini molts nivells i el que volem es que

faci una jugada decent en un temps ràpid, llavors haurem de fer servir la última funció que ens retornarà un resultat millor.

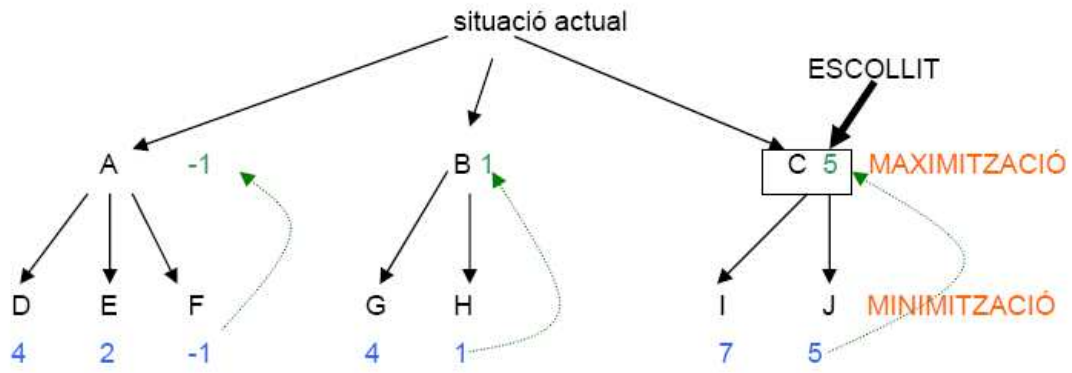
4.3. Motor d'inferència

El motor d'inferència de la meva aplicació correspon a l'algorisme minimax, aquest segueix l'estratègia de pensar que un jugador quan ha de moure ho farà escollint un moviment que li doni el màxim de benefici dins de tots els possibles moviments vàlids. Llavors en el següent torn el contrari intentarà escollir un moviment que minimitzi el benefici del primer jugador.

Això suposa treballar a 2 nivells: un nivell n on el jugador voldrà maximitzar el seu guany i un nivell n+1 on el contrari voldrà minimitzar el guany del jugador. Això implica tenir un nivell de maximització i un de minimització.

Es posa a continuació un exemple abstracte de com pretén treballar aquesta estratègia:

DIAGRAMA 4.3.1



Es veu en l'exemple que es treballa a dos nivells, llavors del per cada possible jugada nivell de minimització escull la que té el valor més petit, i aquest es porta a nivell superior. Un cop a nivell superior s'escull la jugada amb valor més alt. És a dir, dintre de les jugades que pugui escollir el contrari i que menys benefici doni al jugador, s'escull la que li maximitzi el seu benefici.

Tornant a l'exemple també es veu que cada possible jugada porta un valor numèric associat, això vol dir que cada joc amb el que es vulgui treballar caldrà associar-hi una funció d'avaluació que permeti donar un valor gran a una jugada que doni molt benefici al jugador, però alhora també que tingui en compte la jugada que pot fer el contrari i que pot disminuir el guany del jugador, funció que hem estudiat a l'apartat anterior.

És defineix a continuació la funció recursiva associada a l'estratègia MINIMAX:

```
MINIMAX(posicio, nivell)
        /* casos base */
        if (esGuanyador(posicio)) then retorna +1
        else if (esPerdedor(posicio)) then retorna −1
        else if (esEmpate(posicio)) then retorna 0
        else if (nivell = límit) then tornar evaluacio(posicio)
        else
                /* cas recursiu */
                for all succesor i de posicio do
                        valors[i] := MINIMAX(sucesor i, nivell+1)
                        if (esNodeMAX(nivell)) then
                                tornar maxim(valors)
                        end if
                        if (esNodeMIN(nivell)) then
                                tornar minim(valors)
                        end if
                end for
        end if
end minimax
```

Es crida inicialment com:

```
MINIMAX( estat_inicial, 0)
```

Mirant la funció ens podem adonar de que té un paràmetre que és profunditat. Aquest ens permet decidir fins a quin nivell de profunditat es vol arribar per tal de decidir quina és la millor jugada a escollir. Com podem imaginar quanta més visió de futur és pugui tenir amb més seguretat s'aconseguirà una bona jugada. Per tant el més lògic és que per decidir quina jugada es treballi a nivell de profunditat 2n, prenent n el valor més gran possible.

Tot el dit és bo, però com pot arribar a ser l'expansió d'un espai d'estats d'un joc, tenint en compte la voluntat d'un valor n gran?. Pot arribar a ser tant gran que moltes vegades pot ser inaplicable. Llavors cal buscar una solució al problema.

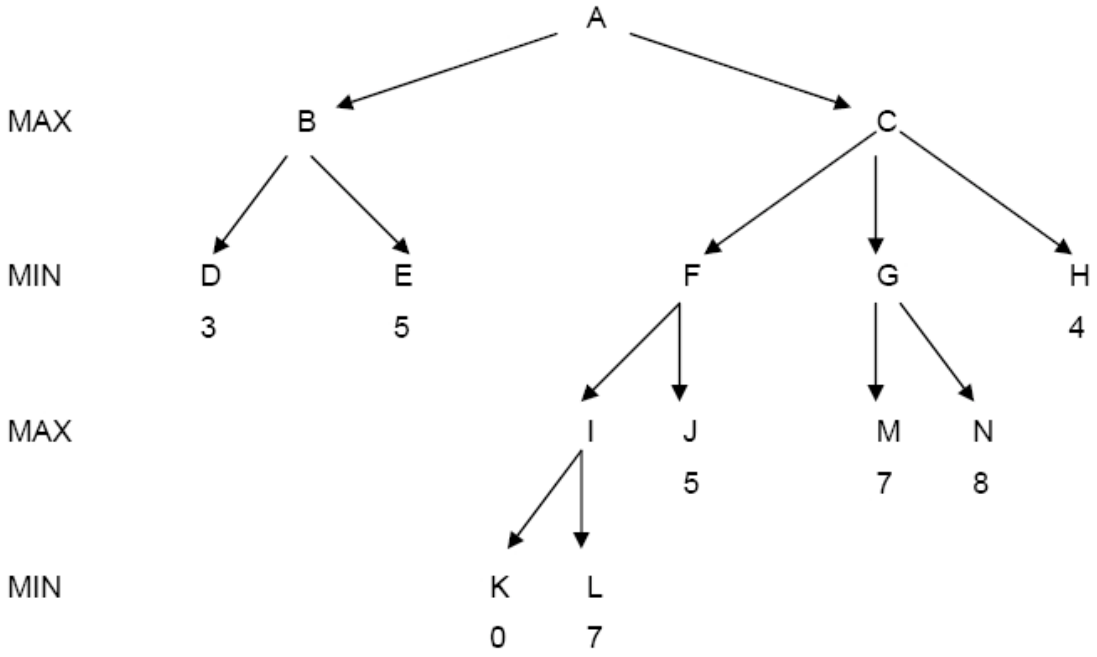
La solució és intentar podar les branques de l'arbre el joc que segur que no seran escollides com a bones jugades. Com? Aplicant l'estratègia **MINIMAX amb PODA o alfa – beta.**

L'estratègia del MINIMAX amb poda consisteix en tenir dues cotes:

- La cota **α:** Aquesta cota es correspon al mínim valor que pot haver-hi en un nivell de maximització.
- La cota **β:** Aquesta cota es correspon al valor màxim que pot haver-hi en un nivell de minimització.

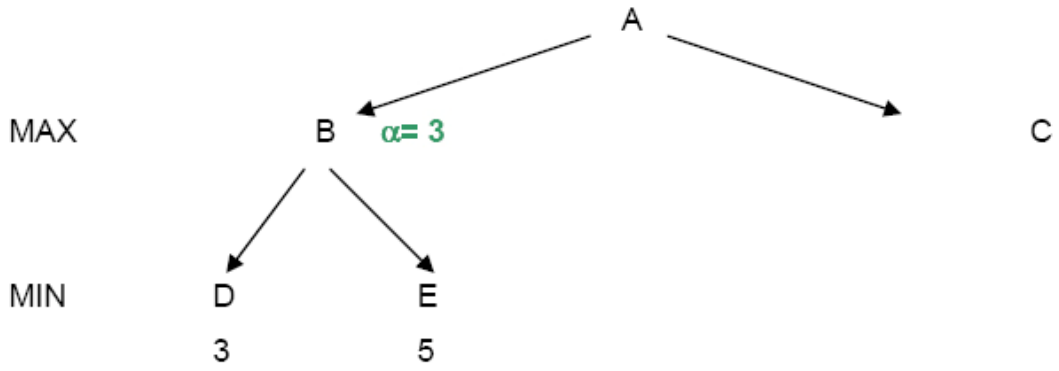
Aleshores quan s'estigui en un nivell de maximització explorant un estat, tant aviat com es pugui assegurar que el seu valor és inferior a α ja s'exclourà, i quan s'estigui en un nivell de minimització, tant aviat com es pugui assegurar que té un valor superior a β ja es pot excloure.
Per veure com funciona aquesta estratègia es proposa el següent exemple:

DIAGRAMA 4.3.2



Pensant que al igual que en l'algorisme MINIMAX, s'explora seguint en profunditat, l'espai d'estats que s'anirà obrint és:

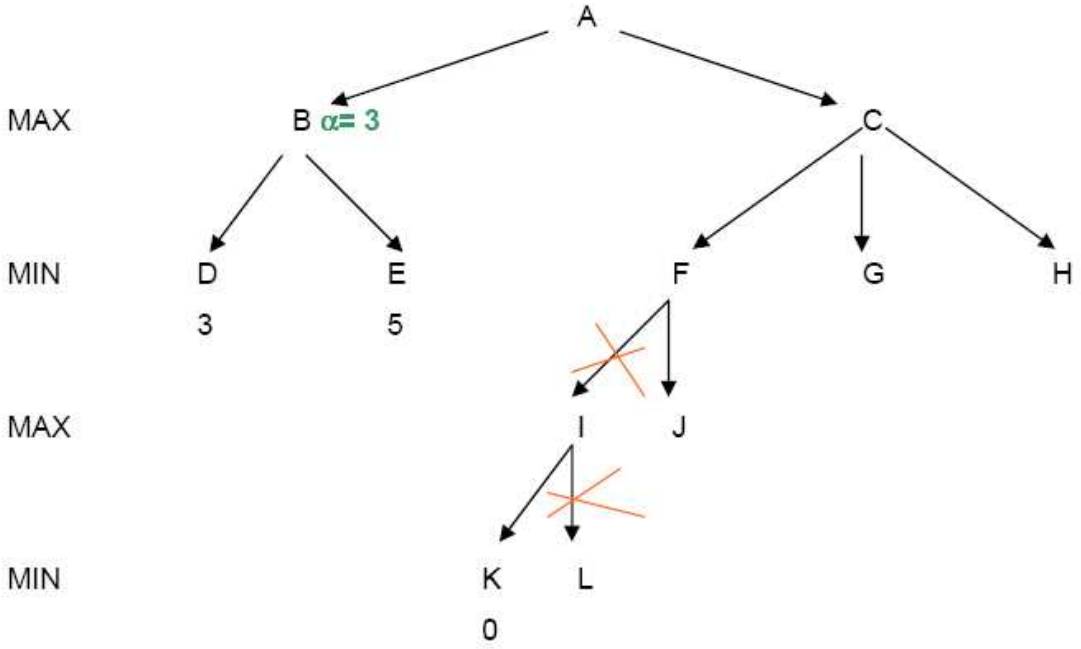
DIAGRAMA 4.3.3



En el nivell de minimització s'escull el 3 i es passa al pare, llavors estan en un nivell de maximització només serà vàlida la possibilitat de que a C hi arribi un valor més gran que 3, per això posem a α el valor de 3. Així doncs en la pròxima exploració si s'està en un nivell de minimització i s'avalua algun estat

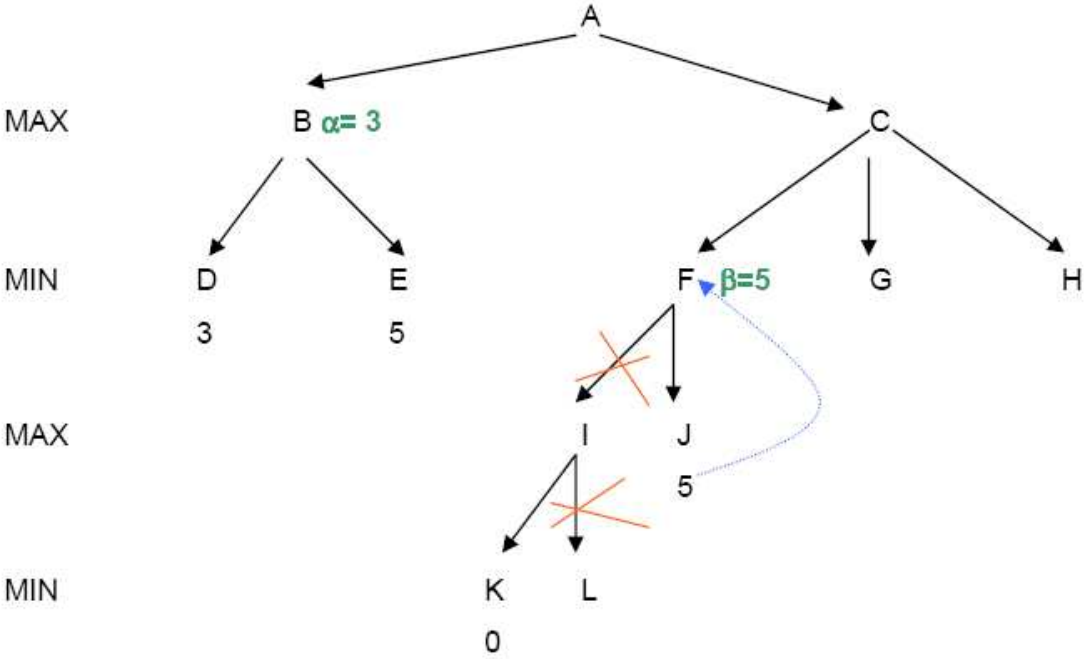
amb un valor inferior a α ja es podrà podar la branca, doncs segur que es passarà un valor inferior a α i no serà contemplat en un nivell de maximització. Continuem explorant:

DIAGRAMA 4.3.4



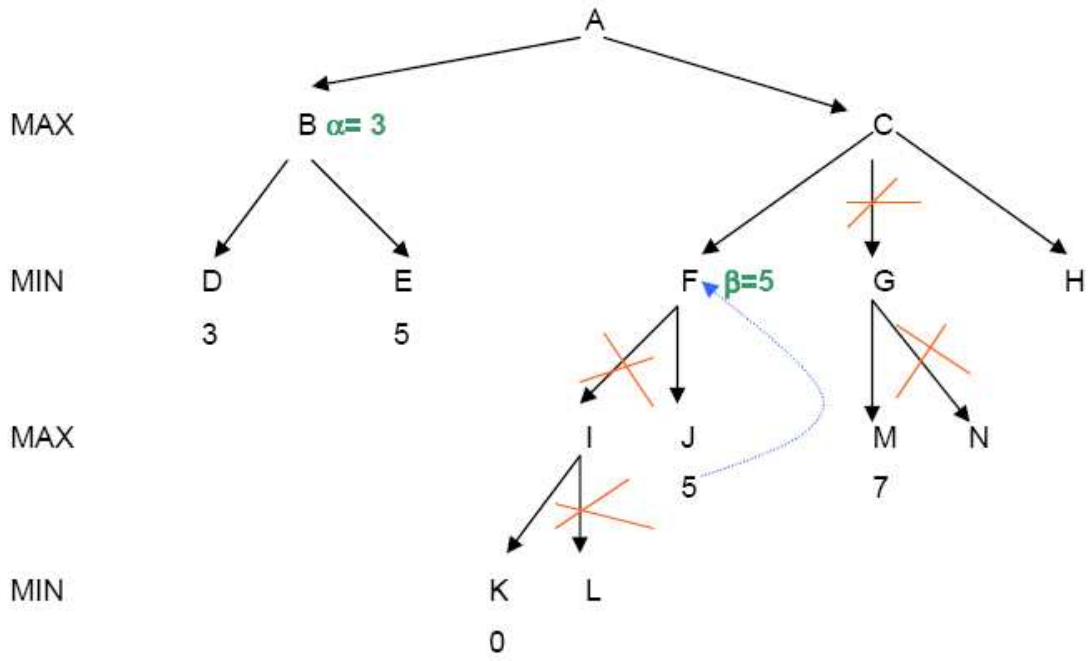
Es poda la branca perquè el valor de K és 0 i és inferior a α. Continuem:

DIAGRAMA 4.3.5



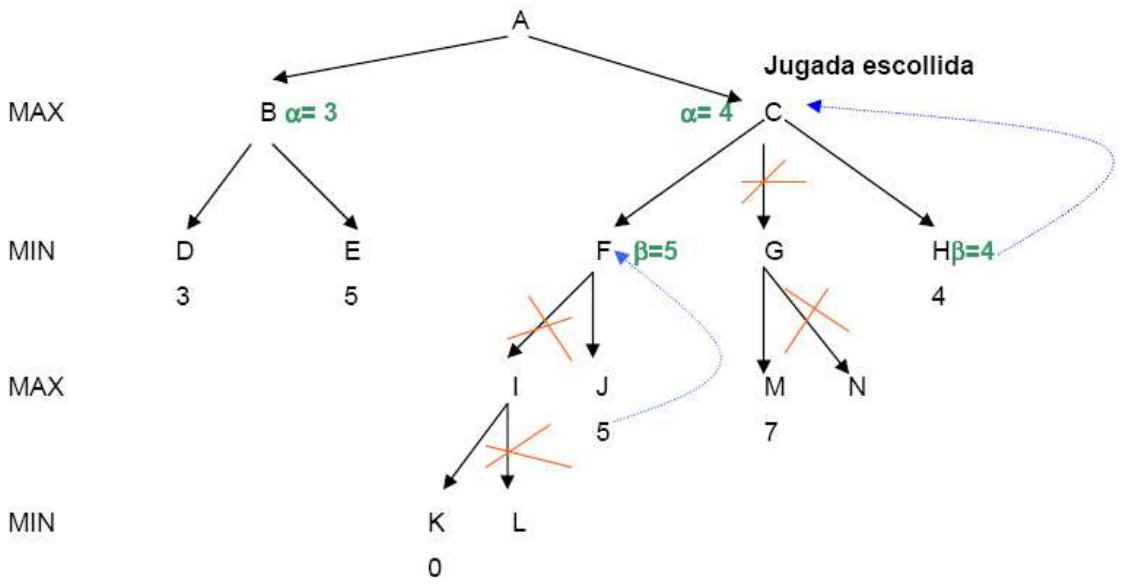
Passem el valor 5 de J al pare, i tot i estant F en un nivell de minimització, al tenir un valor més gran que α, es continua explorant. Si no fos així s'acabaria l'exploració de la branca de C. Al estar explorant sobre un nivell de minimització donem a β el valor de 5. Continuant:

DIAGRAMA 4.3.6



Després d'avaluar M, al estar en un nivell de maximització i tenir un valor més gran que β segur que no serà contemplat en el nivell de minimització, per tant es poda i no s'exploren la resta dels germans. Continuant:

DIAGRAMA 4.3.7



S'explora H i fa que canviï el valor de β, però tot i estant en un nivell de minimització el valor encara és més gran que α i per tant es continua explorant. Al no haver-hi cap més node per explorar, el pare C rep el valor de β i canvia el valor de α. Ara val 4. No hi ha cap més node a explorar, per tant la jugada escollida es C.

Podem veure l'algorisme minimax amb poda alfa-beta a continuació:

```
ALFA_BETA(posicion, α, β, nivell)
if (esGuanyador(posicio)) then tornar +1 then
        else if (esPerdedor(posicio)) then tornar −1 then
        else if (esEmpate(posicio)) then tornar 0 then
        else if (nivell = límit) then tornar evaluacio(posicio) then
        else
        /* cas recursiu */
                if (esNodeMAX(nivell)) then
                        αactual :=  α
                        for all sucesor i de posicion do
                                if (α actual >= β) then
                                        PARAR /* poda BETA */
                                else
                                aux := ALFA_BETA(succesor i, αactual, β, nivell+1)
                                αactual := maximo(αactual, aux)
                                end if
                        end for
                        tornar  αactual
                else if (esNodeMIN(nivell)) then
                βactual := β
                for all succesor i de posicio do
                        if (βactual _ β) then
                                PARAR /* poda ALFA */
                        else
                                aux := ALFA_BETA(sucesor i, α, βactual, nivell+1)
                                βactual := minim(βactual, aux)
                        end if
                end for
                tornar βactual
                end if
        end if
end alfa-beta
```

Crida inicial: ALFA BETA(posició_Actual, −1, +1, 0)

5. Disseny de l'aplicació

En aquesta etapa el que es pretén es dur una mica més enllà el nostre projecte i començar a encarar-lo cap a l' implementació del programa, fent que aquest sigui el més eficient possible, sent també el més senzill que puguem.

Per tal de realitzar aquest apartat, s'han de tenir en compte una sèrie de coses com el llenguatge amb que es programarà, els esdeveniments que es realitzaran o els usuaris que hi treballaran ja que el seu nivell informàtic és el que ens guiarà per realitzar tot l'apartat gràfic.

En resum, podem veure 4 parts bàsiques del disseny del sistema:

- GUI, Interfícies gràfiques
- Part de codi
- Base de coneixements i conjunt de regles
- Motor d'inferència

5.1. SDL

libSDL és l'acrònim de library Simple Directmedia Layer. libSDL va ser creada per Sam Lantinga i el seu grup de programadors de Loki Entertaiment Software, per a portar jocs a Linux principalment. Sam Lantinga és un programador d'àmplia experiència en el món de videojocs la qual li ha permès tenir una visió específica de les necessitats dels creadors de videojocs.

SDL és una llibreria multimèdia multi plataforma, és a dir, totes les aplicacions que desenvolupem amb aquesta llibreria poden ser compilades, sense canviar gens en el codi, en diversos sistemes diferents com Windows, Linux, BeOS... Està dissenyada per a proporcionar accés de baix nivell a l'àudio, teclat, ratolí, maneta de jocs i als dispositius de vídeo 2D. És una llibreria molt utilitzada en diferents tipus d'aplicacions entre elles es troba programari per a reproduir vídeo MPG, emuladors i un gran nombre jocs i adaptacions de jocs entre plataformes.

Podem marcar el començament dels ordinadors personals allà per 1982 quan IBM va comercialitzar la primeres computadores equipats amb PC-DOS, versió del MS-DOS de Microsoft. En els temps d'aquest sistema operatiu la programació de videojocs era molt complexa. Calia conèixer en profunditat el llenguatge assemblador i l'estructura interna de la màquina, així com els elements multimèdia que poguessin posseir, com la targeta de vídeo o de so. La principal complexitat és que la programació es feia gairebé directament sobre el maquinari amb tots els inconvenients d'abstracció, complexitat i portabilitat que suposa això. Resumint, per a fer un videojoc, fos el simple que fos, feia falta un grandíssim esforç.

L'aparició del sistema operatiu Windows i les seves versions no van acabar de millorar el panorama a causa de un rendiment gràfic pobre i la impossibilitat de tenir un accés directe al maquinari en molta de les seves versions. A més l'accés als recursos del sistema varia d'una versió a una altra, d'un sistema operatiu a un altre. Necessitem una interfície comuna per al desenvolupament de videojocs que ens permeti establir-nos en un entorn més "estandarditzat" i amb una interfície comuna als recursos del sistema. En definitiva es va fer fonamental tenir una eina que ens proporcionés una capa d'abstracció que ens permetés dedicar-nos al desenvolupament del videojoc en si alliberant-nos fins a cert punt dels aspectes concrets del sistema on anés a executar-se aquesta aplicació.

L'empresa Loki Games va sortir al rescat, més concretament un experimentat programador anomenat Sam Lantinga. Davant la necessitat de realitzar jocs que fossin portables entre diferents sistemes van crear una llibreria multi anomenada SDL, que ens proporciona accés als recursos del sistema on es faci córrer l'aplicació mitjançant una interfície independent del sistema per al qual anem a realitzar l'aplicació.
SDL suporta Linux, Windows, Windows CE, BeOS, MacOS, MacOS X, FreeBSD, NetBSD, OpenBSD, BSD/Us, Solaris, IRIX i QNX. El codi de la llibreria ens permet treballar en aplicacions per a AmigaOS, Dreamcast, Atari, AIX, OSF/Tru64, RISC Us, SymbianOS i US/2 tot i que aquests sistemes no són suportats oficialment.
SDL està escrita en C però treballa nadivament amb C++. Es pot utilitzar en molts altres llenguatges ja que existeixen adaptacions que permeten fer-ho. Algun dels llenguatges en els quals es pot utilitzar són Ada, C#, Eiffel, ERlang, Euphoria, Guile, Haskell, Java, Lisp, Lua, ML, Objective C, Pascal, Perl, PHP, Pije, Pliant, Python, Ruby i Smalltalk.
És important conèixer que SDL és distribuït sota llicència GNU LGPL en la seva versió 2. Aquesta llicència permet usar SDL lliurement, en programes comercials, sempre que utilitzem un enllaç dinàmic a aquesta llibreria i complim tots els requisits d'aquesta llicència.

SDL proporciona una API que ens permet treballar amb els gràfics d'una manera molt versàtil. Podem treballar amb píxels en cru, directament, construint línies i polígons i fer el que vulguem amb ells o bé carregar imatges que emplenin píxels facilitant així el treball de crear figures o personatges en la nostra aplicació.
També s'encarrega de configurar les maneres de vídeo que demandem, la manera d'accés a memòria de vídeo... ens proporciona finestres o pantalla completa... Podem escriure directament en la memòria de vídeo al seu framebuffer. Crear superfícies amb canals alpha i colors clau. Bolcar les superfícies convertint els formats automàticament al de destinació. Permet realitzar accions accelerades per maquinari sempre que estiguin suportades pel nostre sistema i moltes altres possibilitats.

SDL també ens proporciona entrada per teclat, ratolí i maneta de jocs usant un model d'entrada basat en esdeveniments semblant a l'utilitzat en el desenvolupament d'aplicacions X11, Windows o Mac OS. L'avantatge és que SDL ens proporciona una abstracció d'aquests podent-los utilitzar sense preocupar-nos de per quin sistema operatiu estiguem desenvolupant la nostra aplicació. Podrem oblidar-nos dels esdeveniments específics produïts per cada sistema operatiu que compliquen la tasca d'escriure el codi de l'aplicació i anul·len la possibilitat de portabilitat.
Per a la gestió de l'entrada d'usuari SDL també ens proporciona la possibilitat de conèixer l'estat d'un dispositiu d'entrada en un moment donat sense que aquest hagi d'haver realitzat cap acció. Per exemple podem consultar l'estat d'una determinada tecla o de cert botó del ratolí quan creguem convenient. Aquest mètode de consulta d'estats pot ser-nos útil a l'hora de desenvolupar la nostra aplicació ja que ens permet dictaminar quan volem consultar l'estat dels dispositius d'entrada en contraposició a la gestió de la cua d'esdeveniments.

5.2. GUI, Interfícies gràfiques

Tal com he comentat anteriorment, amb SDL podem treballar amb imatges per tal de crear les interfícies gràfiques de qualsevol aplicació sense haver de fer servir les API de Windows per crear finestres, per tant, podem crear finestres d'aplicació per a qualsevol sistema operatiu treballant amb un llenguatge de programació tal com si estiguéssim realitzant una aplicació sobre consola.

Si només féssim servir les llibreries d' SDL, només podríem treballar amb imatges BMP, cosa que suposa un cost tant a nivell de disseny com a nivell de pes del programa, però per sort, si es busca una mica, SDL disposa d'unes subllibreries que s'han d'instal·lar per separat, que ens permeten treballar amb imatges de format JPEG, TIFF i fins i tot PNG.

A l'hora de dissenyar les interfícies gràfiques, s'ha de tenir en compte diversos factors tals com el tamany que tindran les nostres imatges, com la seva estètica i la dificultat d'us. Basant-me en els dos primers punts crec que el format més convenient d'imatges per la meva aplicació, és el PNG ja que és un format d'última generació orientat cap a disseny de pàgines web, el que comporta que el seu tamany és molt més reduït que el d'una imatge en format BMP. Per altra banda, de cara al disseny, el format PNG, ens permet treballar amb una qualitat molt important que ni BMP ni JPEG ni TIFF posseeixen, aquest pot treballar amb transparències directes sobre el format, fent que no haguem de treballar-les a partir de codi i per tant, fent l'aplicació molt més àgil.

A part de les "característiques tècniques" de les imatges de les interfícies gràfiques, és molt important tenir clar com volem que siguin en si les interfícies ja que aquestes faran en un alt grau que l'aplicació sigui atractiva per els usuaris, a part de ser fàcil de fer servir.

Tenint en compte el punt anterior, crec que les interfícies hauran de ser molt intuïtives per tal de que l'usuari no s'hi senti incòmode així que el primer a tenir en compte al fer l' és que l'usuari podrà fer servir l'aplicació únicament amb el ratolí, minimitzant l'ús de tecles. El segon aspecte a tenir en compte és que les interfícies no poden ser massa monòtones o avorrides ja que el jugador se'n podria cansar molt ràpid i per últim, hauré de fer servir el mateix disseny bàsic de botons per tal de no fer l'aplicació massa difícil.

Al igual que amb les imatges, SDL, posseeix una subllibreria capaç de reproduir sons en format MP3, el que ens permetrà fer encara més amena la GUI, fent que l'aplicació tingui una banda sonora i fins i tot sons al moure les peces.

5.3. Part de codi

En aquest punt ha arribat l'hora de decidir amb quin llenguatge serà programada l'aplicació, i en aquest cas i tenint en compte tot l'exposat anteriorment, crec que el millor llenguatge de programació que puc triar per realitzar l'aplicació és C++. Aquesta decisió es basa en la potencia d'aquest llenguatge ja que és el llenguatge de programació per excel·lència, pot treballar amb objectes, cosa que serà molt útil a la meva aplicació, compta amb herència, és totalment compatible amb SDL, etc. També he de destacar un punt molt important ja que al igual que SDL, és totalment portable a tots els sistemes operatius i per tant, en tot moment podria compilar l'aplicació amb un compilador de Linux o Solaris i tenir l'aplicació totalment operativa en aquests sistemes operatius.

El llenguatge de programació C++ compta amb moltíssimes més avantatges que les esmentades, però no pretenc explicar com es programa amb C++ ja que podria fer dues memòries més però si comentaré les principals característiques que he fet servir per l'aplicació.
També comentar que l'entorn escollit per programar l'aplicació és el famós DEV-C++, entorn totalment gratuït desarrollat per el grup Bloodshed Software sota llicencia GNU. Aquest entorn no és tant potent com altres entorns de pagament però també cal recordar que per programar amb llenguatge C++ ho podríem fer fins i tot amb un simple bloc de notes i el compilador C++.

5.3.1. Game Loop

El primer que hem de tenir clar si programem utilitzant SDL és que les aplicacions i sobretot els jocs que implementem no tindran la estructura clàssica que tenen al ser programats sense ajuda d'. En aquest cas els jocs es basen en el que anomenem Game Loop, aquest no és més que un bucle que en tot moment espera esdeveniments que el jugador li passi, ja siguin moviments de ratolí, pulsació de tecles, pulsació dels botons del ratolí, etc. Per tant nosaltres hem d'implementar totes les funcions que capturin els esdeveniments que necessitem.
L'estructura d'aquest Game Loop, és la següent:

```
Mentre no sortir fer
        Mentre hi hagin events fer
                Si event = ... fer
                Fi si
        Fi mentre
Fi mentre
```

I aplicat al joc d'escacs podem aplicar un Game Loop similar al següent:

```
Mentre no final fer
        Seleccionar peça
        Comprovar possibles moviments
        Moure peça
        Algun jugador guanya?
        Fi torn
Fi mentre
```

5.3.2. Inicialitzar subsistemes

El primer a fer si volem fer servir SDL a la nostra aplicació és inicialitzar SDL i els seus subsistemes, tal com veurem, abans de fer servir qualsevol subsistema d' com el de vídeo, àudio o imatges, l'hem d'inicialitzar; cada subsistema compta amb els seus paràmetres d'inici com el tamany en píxels de la pantalla o la qualitat del so.
De tota manera aquest punt el veurem amb molta més claredat quan ho implementem.

5.3.3. Sortida estàndard

Un altre aspecte molt importat a l'hora de programar fent servir les llibreries SDL hem de saber que la sortida estàndard del programa no serà la pantalla com tenim per costum, SDL ens redirecciona tot el que vulguem fer sortir cap a uns fitxers anomenats stdout.txt o stderr.txt deixant lliure la pantalla per els gràfics.

5.4. Disseny de la base de coneixements

Tal com abans hem vist la base de fets en aquest cas és la situació actual en que es troba el programa en tot moment, per tant el que faré serà guardar en tot moment l'estat del tauler a memòria dinàmica i així controlarem en temps real tota l' informació de la partida.
Com que estaré treballant sobre C++ el millor serà crear un objecte tauler que contingui tota l' referent al tauler com la posició de les peces i certes funcions com dibuixar el tauler.
El més important d'aquesta ED serà, com no, la posició de les peces, així que el que hauré de crear serà una altra ED que representi a una simple casella, aquesta haurà de tenir un tamany, un booleà que marqui si la casella es ocupada i el més important una forma d'identificar la peça que conté, així que el que em proposo és fer que un enter em designi quina peça hi ha, sent 0 que la casella és buida, 1 el peó blanc, etc.
Un cop tingui la ED de la casella, ja podré composar la part més important de la ED tauler, la estructura que emmagatzemarà tot el tauler virtual, així que el més apropiat sabent que farem servir C++ serà crear una matriu bidimensional de caselles de 8 x 8 posicions que composaran el nostre tauler virtual.

5.5. Disseny del conjunt de regles

Tal com tinc pensat implementar l'aplicació, les regles que abans he mostrat en lògica de predicats s'hauran d'implementar en dues funcions, en una que un cop marcada una peça ens retorni un tauler amb tots els possibles moviments marcats i en una altra funció que ens retornarà una llista de possibles "fills" d'un tauler; cada fill representa el moviment d'una peça dins el tauler per tal de poder aplicar la funció minimax i així poder triar la millor jugada.
Com a exemple os mostraré el conjunt de regles en forma de llenguatge de programació.

5.5.1. Moviment de les peces

Cal esmentar que el tipus de peça s'ha assignat al atribut numero de la ED caselles, tal com he explicat al punt anterior. Aquest número ens indica les següents equivalències:

TAULA 5.5.1.1

Número	Peça
0	Casella buida
1	Peó blanc
2	Torre blanca
3	Cavall blanc
4	Alfil blanc
5	Reina blanca
6	Rei blanc
7	Peó negre
8	Torre negra
9	Cavall negre
10	Alfil negre
11	Reina negra
12	Rei negre

5.5.1.1. Moviment dels peons

En el cas dels peons blancs disposaríem de les següents regles:

```
if(taux.caselles[x][y-1].numero==0)
if(taux.caselles[x][y-2].numero==0 && y==6)
if(taux.caselles[x+1][y-1].numero>0 and taux.caselles[x+1][y-1].numero>6)
if(taux.caselles[x-1][y-1].numero>0 and taux.caselles[x-1][y-1].numero>6)
```

I en el cas dels negres:

```
if(taux.caselles[x][y+1].numero==0)
if(taux.caselles[x][y+2].numero==0 && y==1)
if(taux.caselles[x+1][y+1].numero>0 and taux.caselles[x+1][y+1].numero<7)
if(taux.caselles[x-1][y+1].numero>0 and taux.caselles[x-1][y+1].numero<7)
```

5.5.1.2. Moviment de les torres

```
if(taux.caselles[x][y-i].numero==0 or taux.caselles[x][y-i].numero>6)
if(taux.caselles[x][y+i].numero==0 or taux.caselles[x][y+i].numero>6)
if(taux.caselles[x-i][y].numero==0 or taux.caselles[x-i][y].numero>6)
if(taux.caselles[x+i][y].numero==0 or taux.caselles[x+i][y].numero>6)
```

5.5.1.3. Moviment d'els cavalls

```
if(x<6 && y<7 && (taux.caselles[x+2][y+1].numero==0 || taux.caselles[x+2][y+1].numero>6))
if(x<6 && y>0 && (taux.caselles[x+2][y-1].numero==0 || taux.caselles[x+2][y-1].numero>6))
if(x>1 && y<7 && (taux.caselles[x-2][y+1].numero==0 || taux.caselles[x-2][y+1].numero>6))
if(x>1 && y>0 && (taux.caselles[x-2][y-1].numero==0 || taux.caselles[x-2][y-1].numero>6))
if(x<7 && y<6 && (taux.caselles[x+1][y+2].numero==0 || taux.caselles[x+1][y+2].numero>6))
if(x<7 && y>1 && (taux.caselles[x+1][y-2].numero==0 || taux.caselles[x+1][y-2].numero>6))
if(x>0 && y<6 && (taux.caselles[x-1][y+2].numero==0 || taux.caselles[x-1][y+2].numero>6))
if(x>0 && y>1 && (taux.caselles[x-1][y-2].numero==0 || taux.caselles[x-1][y-2].numero>6))
```

5.5.1.4. Moviment dels alfils

```
if(taux.caselles[x+i][y+i].numero==0 or taux.caselles[x+i][y+i].numero > 6)
if(taux.caselles[x-i][y+i].numero==0 or taux.caselles[x-i][y+i].numero > 6)
if(taux.caselles[x-i][y-i].numero==0 or taux.caselles[x-i][y-i].numero > 6)
if(taux.caselles[x+i][y-i].numero==0 or taux.caselles[x+i][y-i].numero > 6)
```

5.5.1.5. Moviment de la reina

```
if(taux.caselles[x][y-i].numero==0 or taux.caselles[x][y-i].numero>6)
if(taux.caselles[x][y+i].numero==0 or taux.caselles[x][y+i].numero>6)
if(taux.caselles[x-i][y].numero==0 or taux.caselles[x-i][y].numero>6)
if(taux.caselles[x+i][y].numero==0 or taux.caselles[x+i][y].numero>6)
if(taux.caselles[x+i][y+i].numero==0 or taux.caselles[x+i][y+i].numero > 6)
if(taux.caselles[x-i][y+i].numero==0 or taux.caselles[x-i][y+i].numero > 6)
if(taux.caselles[x-i][y-i].numero==0 or taux.caselles[x-i][y-i].numero > 6)
if(taux.caselles[x+i][y-i].numero==0 or taux.caselles[x+i][y-i].numero > 6)
```

5.5.1.6. Moviments del rei

```
if(y<7 && (taux.caselles[x][y+1].numero==0 || taux.caselles[x][y+1].numero>6))
if(y>0 && (taux.caselles[x][y-1].numero==0 || taux.caselles[x][y-1].numero>6))
if(x<7 && (taux.caselles[x+1][y].numero==0 || taux.caselles[x+1][y].numero>6))
if(x>0 && (taux.caselles[x-1][y].numero==0 || taux.caselles[x-1][y].numero>6))
if(x<7 && y<7 && (taux.caselles[x+1][y+1].numero==0 || taux.caselles[x+1][y+1].numero>6))
if(x>0 && y<7 && (taux.caselles[x-1][y+1].numero==0 || taux.caselles[x-1][y+1].numero>6))
if(x<7 && y>0 && (taux.caselles[x+1][y-1].numero==0 || taux.caselles[x+1][y-1].numero>6))
if(x>0 && y>0 && (taux.caselles[x-1][y-1].numero==0 || taux.caselles[x-1][y-1].numero>6))
```

5.5.2. Final de partida

```
jblanc=false;
jnegre=false;
for (int i=0; i<8; i++)
{
        for (int j=0; j<8; j++)
        {
                if(taux.caselles[x][y].numero==6) jblanc=true;
                else if (taux.caselles[x][y].numero==12) jnegre=true;
        }
}
if(!jblanc) guanyanegre();
else if(!jnegre) guanyablanc();
```

5.6. Disseny de la funció d'avaluació

Tal com hem vist a la representació del coneixement, el que anomenem funció d'avaluació, ens retornarà un valor numèric amb el qual avaluarem la situació del tauler. A l'hora de dissenyar aquesta funció, podem dir que serà una funció a la que li passarem un *tauler* i ens retornarà el seu valor numèric en un enter.

Per tal de dissenyar aquesta funció, seguirem els passos que tenim a la representació i el resultat podrà ser una funció aproximada a la següent:

```
int tauler::avaluacio()
{
   int r=0;
   for (int i=0; i<8; i++)
   {
      for (int j=0; j<8; j++)
      {
         switch(caselles[i][j].numero)
         {
                        case 1:
                           r=r+pespeob[i][j];
                           break;
                        case 2:
                           r=r+pestorre[i][j];
                           break;
                        case 3:
                           r=r+pescavall[i][j];
                           break;
                        case 4:
                           r=r+pesalfil[i][j];
```

```
                    break;
                case 5:
                    r=r+pesreina[i][j];
                    break;
                case 6:
                    r=r+1000;
                    break;
                case 7:
                    r=r-pespeon[i][j];
                    break;
                case 8:
                    r=r-pestorre[i][j];
                    break;
                case 9:
                    r=r-pescavall[i][j];
                    break;
                case 10:
                    r=r-pesalfil[i][j];
                    break;
                case 11:
                    r=r-pesreina[i][j];
                    break;
                case 12:
                    r=r-1000;
                    break;
            }
        }
    }
    return r;
}
```

6. Implementació

Tenint en copte tots els punts esmentats anteriorment, passaré a mostrar i comentar les parts més rellevants del codi implementat i de les imatges creades per les interfícies, base de fets, funció d'avaluació, conjunt de regles, etc...

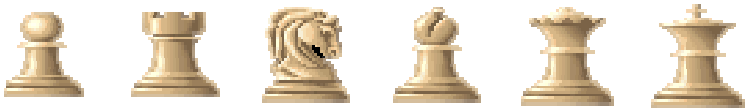
6.1. Interfícies gràfiques

Les pantalles dissenyades per tal de que l'usuari pugui triar les opcions, jugar a les escacs i comunicar-se amb l'aplicació son les següents:

6.1.1. Peces

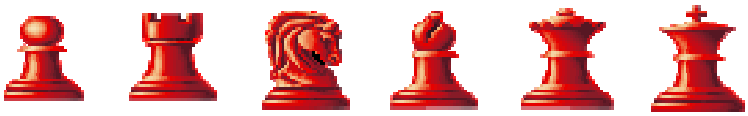
6.1.1.1. Peces blanques

De esquerra a dreta, peó, torre, cavall, alfil, reina i rei.



6.1.1.2. Peces negres.

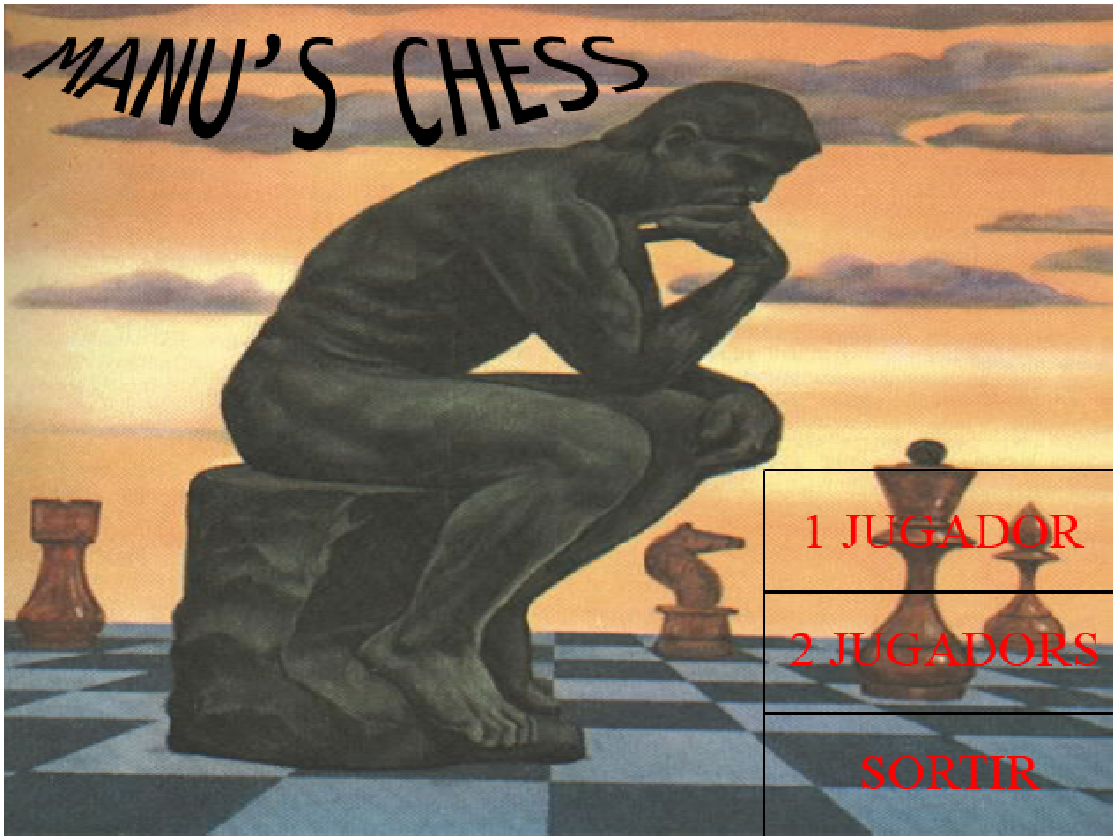
De esquerra a dreta, peó, torre, cavall, alfil, reina i rei.



6.1.1.3. Peça marcada

6.1.1.4. Possible moviment

6.1.2. Pantalla del menú



6.1.3. Pantalla principal

6.1.4. Pantalla blanques guanyen

6.1.5. Pantalla negres guanyen

6.2. Codi variat

En aquest apartat mostraré certes parts de codi, amb els seus comentaris, que em semblen més rellevants del projecte, no posaré tot el codi ja que podria omplir dues memòries més i es faria molt feixuc de llegir, però si aquelles que crec que son més interessants.

6.2.1. Inicialització de subsistemes

Tal com he explicat en punts anteriors, els subsistemes SDL s'han d'inicialitzar:

```
if(SDL_Init(SDL_INIT_VIDEO) < 0)   //Inicialitzem el sistema de vídeo
   {
            cerr << "No es pot iniciar SDL: " <<  SDL_GetError() << endl;        //En cas de fallo sortim
            exit(1);
   }
   atexit(SDL_Quit);                        //Quan l'aplicació finalitzi, tancarà els sistemes SDL
   TTF_Init();                              //Inici del subsistema d'escriptura a pantalla
   atexit(TTF_Quit);                        //Al sortir es tancarà el subsistema
   if(SDL_VideoModeOK(800, 600, 24, SDL_HWSURFACE|SDL_DOUBLEBUF) == 0) //Comprovem
// compatibilitat del mode de vídeo
   {
    cerr << "Mode de vídeo no suportat: " << SDL_GetError();
    exit(1);
   }
   pantalla = SDL_SetVideoMode(800, 600, 24, SDL_HWSURFACE|SDL_DOUBLEBUF);
//Assignem el mode de pantalla
   if(pantalla == NULL)
   {
            cerr << "No es pot establir el mode de vídeo: "<<SDL_GetError()<<endl;
            exit(1);
   }
   fons = IMG_Load("Img/menu.png");                        //Carreguem l'imatge de fons
   if(fons == NULL)
   {
         cerr << "No podem carregar fons: "<<SDL_GetError()<< endl;
                  exit(1);
   }
   SDL_WM_SetCaption("Manu's Chess", NULL);           //Títol i icona de la finestra
   desti.x = 0;
   desti.y = 0;
   SDL_BlitSurface(fons, NULL, pantalla, &desti); //Copiem el fons a la superfície principal
   SDL_Flip(pantalla);   //Actualitzem pantalla
   SDL_FreeSurface(fons);            //Alliberem el rectangle
```

6.2.2. Banda sonora

L'aplicació consta d'una banda sonora que és crea de la següent manera:

```
if(Mix_OpenAudio(44100, MIX_DEFAULT_FORMAT, MIX_DEFAULT_CHANNELS, 4096) < 0)
                                                        //Inicialitzem el subsistema d'àudio
  {
                cer << "Subsistema d'Àudio no disponible"<<SDL_GetError()<< endl;
                        exit(1);
  }                 //Si hi hagués algun error ho notificaria i sortiria
  atexit(Mix_CloseAudio);           //Al sortir tenca el sistema d'àudio
  Mix_Music *bso;                           //Creem una instancia d'àudio
  bso = Mix_LoadMUS("./Audio/BSO/1.mp3");//Carreguem l'arxiu de la BSO
  if(bso == NULL)                                       //Si hi ha error ho capturem i tanquem
  {
      cer << "No es pot carregar la BSO: "<<SDL_GetError()<<endl;
              exit(1);
  }
  int volum = 100;
  Mix_Music *menu;                          //Carreguem la BSO del menú
  menu = Mix_LoadMUS("./Audio/BSO/0.mp3");
  if(menu == NULL)                                      //Si hi ha error ho capturem i tanquem
  {
      cer << "No es pot carregar la BSO: "<<SDL_GetError()<<endl;
              exit(1);
  }
  Mix_PlayMusic(menu, -1);          //I comencem a reproduir la BSO
```

6.2.3. Game loop

El Game loop és l'estructura de joc d' SDL, aquí s'hi capturen els esdeveniments i s'apliquen tots els canvis:

```
while(true)                     //Bucle infinit
  { //Game loop
   while(SDL_PollEvent(&event))     //Captura dels events
   {
    switch(event.type)
    {
     case SDL_MOUSEBUTTONDOWN:                  //Quan s'apreta algun botó
         switch(estat)                          //Actuarà diferent segons l'estat
         {
                 case 0:                        //Estat 0, Menú
                     switch (event.button.button)
                     {
                         case SDL_BUTTON_LEFT:      //Al apretar el botó dret
```

```
                    mx = event.button.x;  //Coordenada X del mouse
                    my = event.button.y;  //Coordenada Y del mouse
                  if(event.button.x > 550 and event.button.y > 425 and event.button.x < 800 and event.button.y
< 512)
                    {                     //Si s'apreta sobre el primer botó
                          estat=1;    //Passem al estat u
                          Mix_PlayMusic(bso, -1);       //Activem la música del primer estat
                          t.dibuixar(49, pantalla);         //I canviem el fons de la pantalla
                    }

                    if(event.button.x > 550 and event.button.y > 425 and event.button.x < 800 and
event.button.y < 512)
                    {                     //Si s'apreta sobre el primer botó
                          estat=2;    //Passem al estat dos
                          Mix_PlayMusic(bso, -1);
                          t.dibuixar(49, pantalla);
                    }
                    if(event.button.x > 550 and event.button.y > 512 and event.button.x < 800 and
event.button.y < 640) return 0;                                   //Si s'apreta el botó sortir es tanca el
programa
                    break;
              }
              break;
          case 1:                         //Primer esta, jugador contra IA
              switch (event.button.button)
              {
                  case SDL_BUTTON_LEFT:
                    mx = event.button.x;  // Coordenada X del mouse
                    my = event.button.y;
                    if(t.torn%2==0)                 //Si el torn és de la IA
                    {
                          t=MINIMAX_ALFABETA(t, 0, 2, -15000, 15000);          //Calcula la millor
jugada
                          t.dibuixar(49, pantalla);                            //Renova el tauler
                          t.torn++;                                      //I passa el torn
                    }
                    Else                                              //Si el torn és del jugador
                    {
                    if(t.marcat)                            //Si el tauler està marcat
                    {
                        t.moure(t.xt, t.yt, mx, my, pantalla);        //Mou la peça
                        m.EscriureSDL("La peca moguda des de", 550, 70, pantalla);
                        m.EscriureSDL("H2", 550, 90, pantalla);
                        m.EscriureSDL("cap a", 550, 110, pantalla);
                        m.EscriureSDL("H5", 550, 130, pantalla);
```

```
                t.marcat=false;                         //Desmarca el tauler
                final();                                //Comprova si la partida s'ha acabat
            t.torn++;                           //I passa el torn
        }
        else                                            //En cas de que no estigui marcat
        {
                t.marcar(mx,my,pantalla);               //Marca la casella premuda
                t=possibilitats(mx,my,t);               //Calcula les possible jugades
                t.dibuixar(49,pantalla);        //Re dibuixa la pantalla
                t.marcat=true;                  //I marca el tauler
        }
        }
        break;
    }
    break;
case 2:                                         //Segon estat, jugador contra jugador
    switch (event.button.button)
    {
        case SDL_BUTTON_LEFT:
            mx = event.button.x;  // Coordenada X del mouse
            my = event.button.y;
            if(t.marcat)
            {
                t.moure(t.xt, t.yt, mx, my, pantalla);
                m.EscriureSDL("La peca moguda des de", 550, 70, pantalla);
                m.EscriureSDL("H2", 550, 90, pantalla);
                m.EscriureSDL("cap a", 550, 110, pantalla);
                m.EscriureSDL("H5", 550, 130, pantalla);
                t.marcat=false;
                final();
            }
            else
            {
                t.marcar(mx,my,pantalla);
                t=possibilitats(mx,my,t);
                t.dibuixar(49,pantalla);
                t.marcat=true;
            }
            break;
    }
    break; //Case2
case 3:                                     //Tercer estat, estats finals
    if(event.button.x > 0 and event.button.y > 500 and event.button.x < 800 and event.button.y < 640)
return 0;
    break;
```

```
            }
            break;
        case SDL_KEYDOWN:                                   //En cas de que es premi una tecla, hi ha un control de
reproducció
            switch(event.key.keysym.sym)
            {
             case SDLK_ESCAPE:
                 Mix_FreeMusic(bso);                    //Amb la tecla esc, para la música
                         return 0;
                         break;
             case SDLK_UP:                              //Amb la fletxa amunt puja el volum
                 volum+= 2;
                  if(volum < 128) Mix_VolumeMusic(volum);
                   else volum = 128;
                 break;
             case SDLK_DOWN:                                //Amb la fletxa avall abaixa el volum
                 volum -= 2;
                 if(volum > -1) Mix_VolumeMusic(volum);
                 else volum = 0;
                 break;
             case SDLK_s:                               //Amb la tecla s pausa i retorna la BSO
                 if(Mix_PlayingMusic() == 1 and Mix_PausedMusic() == 0)
                 {
                     Mix_PauseMusic();
                 }else
                 {
                     Mix_ResumeMusic();
                 }
                 break;

            }
            break;
        case SDL_QUIT:
            return 0;
            break;
        }
       }
    }
```

6.3. Base de fets

Com en punts anteriors he explicat, la base de fets consta de tota l' informació del tauler en un moment donat, per tant passo a mostrar l' implementació de l'objecte tauler:

```
#ifndef TAULER_H
#define TAULER_H
#include "casella.h"
#include <SDL/SDL.h>
#include <SDL/SDL_image.h>
#include "list.h"
// Classe tauler
class tauler
{
    public:
        int xt, yt;    //Variables auxiliars
        bool marcat;            //Variable que ens diu si hi ha alguna peça marcada
        casella caselles[8][8];             //Matriu de caselles (Tauler virtual)
        int torn;     //Variable que guarda en quin torn estem

                    tauler();   //Constructor de tauler
                    void inici();           //Acció que inicialitza el tauler amb les peces allà on toca
                    void trobar_casella(int x, int y, int &xr, int &yr); //Acció que troba una
//casella saben les coordenades
                    void marcar(int x, int y, SDL_Surface *pantalla);             //Acció que marca el Tauler
                    void dibuixar(int mida, SDL_Surface *Pantalla); //Acció que dibuixa el tauler
                    void moure(int xi, int yi,int xf,int yf, SDL_Surface* pantalla); //Acció que mou una
//peça
                    void desmarcar();                //Acció que desmarca el tauler;
                    int avaluacio();                              //Funció d'avaluació
                    List<tauler> obtenirFills(int jugador);  //Funció que ens retorna els successors d'un
//tauler
                    ~tauler(); //Destructor de la classe
};

#endif // TAULER_H
```

I a continuació mostraré l' implementació de les seves funcions:

```
#include "tauler.h"
#include "peca.h"
#include <SDL/SDL_mixer.h>
#include <SDL/SDL_image.h>

tauler::tauler()
{
        torn=1;    //Al constructor, iniciem el torn a 1
}

void tauler::inici()  //Acció que posa totes les peces al tauler
{
    for(int i=0;i<8;i++)
    {
        for(int j=0;j<8;j++)
        {
            caselles[i][j].numero=0;   //Primer ho inicialitza el tauler com buit
        }
    }
    for(int i=0;i<8;i++)
    {
        caselles[i][6].numero=1;                //Posicionem els peons blancs
    }
    caselles[0][7].numero=2;            //Les torres blanques
    caselles[7][7].numero=2;
    caselles[1][7].numero=3;            //Els cavalls blancs
    caselles[6][7].numero=3;
    caselles[2][7].numero=4;            //Els alfils blancs
    caselles[5][7].numero=4;
    caselles[3][7].numero=5;            //La reina blanca
    caselles[4][7].numero=6;            //I el rei blanc
    for(int i=0;i<8;i++)
    {
        caselles[i][1].numero=7;                //Fem el mateix amb els peons negres
    }
    caselles[0][0].numero=8;            //Les torres negres
    caselles[7][0].numero=8;
    caselles[1][0].numero=9;            //Els cavalls negres
    caselles[6][0].numero=9;
    caselles[2][0].numero=10;                   //Els alfils negres
    caselles[5][0].numero=10;
    caselles[3][0].numero=11;                   //La reina negre
    caselles[4][0].numero=12;                   //I el rei negre
```

```
}

tauler::~tauler()
{
}

void tauler::trobar_casella(int x, int y, int &xr, int &yr)                    //Acció que busca una casella segons
coordenades
{
    for(int i=0;i<8;i++)
    {
            for(int j=0;j<8;j++)
            {
                    if(x>(53+i*49) && x<(53+(i+1)*49) && y>(53+j*49) && y<(53+(j+1)*49))
                    {
                                    xr = i;
                                    yr = j;
                    }
            }
    }
}

void tauler::dibuixar(int mida, SDL_Surface *pantalla)                    //Acció que dibuixa el tauler
{
   SDL_Rect desti;                          //Variable necessària
   SDL_Surface *fons, *marcada;             //Superfícies SDL
   fons=IMG_Load("Img/fons.png");           //Carreguem l' imatge de fons
   if(fons == NULL)                                     //Control d'integritat
   {
            cerr << "No podem carregar fons!" << endl;
            exit(1);
   }
   marcada=IMG_Load("Img/marcada.png");     //Fem el mateix per la imatge de marca
   if(fons == NULL)
   {
            cerr << "No podem carregar marcada! " << endl;
            exit(1);
   }

   desti.x = 0;
   desti.y = 0;
   SDL_BlitSurface(fons, NULL, pantalla, &desti); //Copiem el fons a la superfície principal
   char* ruta;
   if(marcat)
   {
```

```
fons=IMG_Load("Img/ok.png");
if(fons == NULL)
{
 cerr << "No podem carregar imatge: Img/ok.png" << endl;
            exit(1);
}

for(int i=0;i<8;i++)          //Pintem a pantalla les possibles jugades
{
      for(int j=0;j<8;j++)
      {
            if(caselles[i][j].pos)
            {
                              desti.x=53+(i*mida);
                              desti.y=53+(j*mida);
                              SDL_BlitSurface(fons, NULL, pantalla, &desti);
            }
            if(caselles[i][j].marcada)
            {
                              desti.x=53+(i*mida);
                              desti.y=53+(j*mida);
                              SDL_BlitSurface(marcada, NULL, pantalla, &desti);
            }
      }
 }
}
 for(int i=0;i<8;i++)                    //Segons la peça carreguem una imatge o una altra
 {
       for(int j=0;j<8;j++)
       {
            if(caselles[i][j].numero!=0)
            {
            switch (caselles[i][j].numero)
            {
                  case 1:
                      ruta=("Img/peob.PNG");
                      break;
                  case 2:
                       ruta=("Img/torreb.PNG");
                       break;
                  case 4:
                       ruta=("Img/alfilb.PNG");
                       break;
                  case 3:
                       ruta=("Img/cavallb.PNG");
```

```
                    break;
                case 5:
                    ruta=("Img/reinab.PNG");
                    break;
                case 6:
                    ruta=("Img/reib.PNG");
                    break;
                case 7:
                    ruta=("Img/peon.PNG");
                    break;
                case 8:
                    ruta=("Img/torren.PNG");
                    break;
                case 10:
                    ruta=("Img/alfiln.PNG");
                    break;
                case 9:
                    ruta=("Img/cavalln.PNG");
                    break;
                case 11:
                    ruta=("Img/reinan.PNG");
                    break;
                case 12:
                    ruta=("Img/rein.PNG");
                    break;
            }
            desti.x=53+(i*mida);
            desti.y=53+(j*mida);
            fons=IMG_Load(ruta);
            if(fons == NULL)
            {
                cerr << "No podem carregar imatge: "<< ruta << endl;
                    exit(1);
            }
            SDL_BlitSurface(fons, NULL, pantalla, &desti);                    //Enganxem la peça sobre el fons
            }
        }
    }
    SDL_Flip(pantalla);                    //Actualitzem la pantalla
    SDL_FreeSurface(fons);                 //Alliberem la superfície
}

void tauler::moure(int xi, int yi,int xf,int yf, SDL_Surface* pantalla)//Acció que mou una peça
{
    trobar_casella(xf,yf,xt,yt);           //Primer localitzem la casella
```

```
    if((xt!=xi || yt!=yi) && caselles[xi][yi].marcada==true) //Si el moviment és legal
    {
     if(caselles[xt][yt].pos)
     {
            caselles[xt][yt].numero=caselles[xi][yi].numero;    //Fem l' intercanvi de peces
            caselles[xi][yi].numero=0;
            caselles[xi][yi].marcada=false;
            torn++;
     }
    }
    desmarcar();                    //Un cop movem, desmarquem el tauler
    Mix_Chunk *so;                          //Creació d'un so
    so = Mix_LoadWAV("./Audio/Chunk/2.wav");            //Carreguem el so
    if(so == NULL)                                      //Comprovem que existeixi
    {
         cerr << "No es pot carregar el so" << endl;
                exit(1);
    }
    int volum = 150;
    Mix_VolumeChunk(so, volum);                         //Li assignem un volum
    Mix_PlayChannel(1, so, 0);                                  //I el reproduïm
    dibuixar(49,pantalla);                              //Per últim re dibuixem la pantalla
}

void tauler::marcar(int x, int y, SDL_Surface *pantalla)
{
   trobar_casella(x,y,xt,yt);
   if(xt>-1 and xt<8 and yt>-1 and yt<8)
   {
            if((torn%2==0 and caselles[xt][yt].numero>6)||(torn%2!=0 and caselles[xt][yt].numero<7))
            {
                    caselles[xt][yt].marcada=true;
            }
   }

}

void tauler::desmarcar()
{
    for(int i=0;i<8;i++)
    {
            for(int j=0;j<8;j++)
            {
                  caselles[i][j].pos=false;
            }
```

```
    }
}
```

Seguit d'aquest codi, vindria l' implementació de la funció d'avaluació i de la funció d'obtenir fills que ja veurem en altres punts.

6.4. Funció d'avaluació

Un cop implementada la funció d'avaluació, tenim que primer declarem unes taules amb els pesos de cada peça segons la seva posició i després n'avaluem el resultat:

```
int tauler::avaluacio()                    //La funció d'avaluació ens retorna un enter
{
        int pespeob[8][8]={{5,5,5,5,5,5,5,5},     //Donem els pesos als peons blancs
                {4,4,4,4,4,4,4,4},                //segons la posició
                {3,3,3,2,2,3,3,3},
                {3,3,3,2,2,3,3,3},
                {3,3,3,2,2,3,3,3},
                {2,2,2,2,2,2,2,2},
                {1,1,1,1,1,1,1,1},
                {0,0,0,0,0,0,0,0}};
        int pespeon[8][8]={{0,0,0,0,0,0,0,0},     //Donem els pesos als peons negres
                {1,1,1,1,1,1,1,1},                //segons la posició
                {2,2,2,2,2,2,2,2},
                {3,3,3,2,2,3,3,3},
                {3,3,3,2,2,3,3,3},
                {3,3,3,2,2,3,3,3},
                {4,4,4,4,4,4,4,4},
                {5,5,5,5,5,5,5,5}};
        int pestorre[8][8]={{15,15,15,15,15,15,15,15},     //Donem els peso a les torres
                {15,20,20,20,20,20,20,15},                 //segons la posició
                {15,20,20,20,20,20,20,15},
                {15,20,20,30,30,20,20,15},
                {15,20,20,30,30,20,20,15},
                {15,20,20,20,20,20,20,15},
                {15,20,20,20,20,20,20,15},
                {15,15,15,15,15,15,15,15}};
        int pescavall[8][8]={{50,50,50,50,50,50,50,50},    //Donem els pesos als cavalls
                {50,50,50,50,50,50,50,50},                 //segons la posició
                {50,50,60,60,60,60,50,50},
                {50,50,60,80,80,60,50,50},
                {50,50,60,80,80,60,50,50},
                {50,50,60,60,60,60,50,50},
                {50,50,50,50,50,50,50,50},
```

```
                {50,50,50,50,50,50,50,50}};
        int pesalfil[8][8]={ {30,35,35,35,35,35,35,30},     //Donem els pesos als alfils
                {35,40,40,40,40,40,40,35},                  //segons la posició
                {35,40,45,45,45,45,40,35},
                {35,40,45,45,45,45,40,35},
                {35,40,45,45,45,45,40,35},
                {35,40,45,45,45,45,40,35},
                {35,40,40,40,40,40,40,35},
                {30,35,35,35,35,35,35,30}};
        int pesreina[8][8]={ {100,120,120,120,120,120,120,100},   //Donem els pesos a
                {120,130,130,130,130,130,130,120},                //les reines segons
                {120,130,140,140,140,140,130,120},                //la posició
                {120,130,140,150,150,140,130,120},
                {120,130,150,130,150,140,130,120},
                {120,130,140,140,140,140,130,120},
                {120,130,130,130,130,130,130,120},
                {100,120,120,120,120,120,120,100}};

int r=0;                                                          //r serà el resultat
for (int i=0; i<8; i++)                                           //Realitzem un recorregut per totes
{                                                                 //les caselles del tauler
   for (int j=0; j<8; j++)
   {
      switch(caselles[i][j].numero)                               //Segons la peça...
      {
                        case 1:                                   //Si és un peó blanc
                            r=r+pespeob[j][i];    //Sumem el seu pes al resultat
                            break;
                        case 2:
                            r=r+pestorre[i][j];   //Si és una torre blanca
                            break;                                //Sumem el seu pes al resultat
                        case 3:
                            r=r+pescavall[i][j];  //Si és un cavall blanc
                            break;                                //Sumem el seu pes al resultat
                        case 4:
                            r=r+pesalfil[i][j];   //Si és un alfil blanc
                            break;                                //Sumem el seu pes al resultat
                        case 5:
                            r=r+pesreina[i][j];   //Si és una reina blanca
                            break;                                //Sumem el seu pes al resultat
                        case 6:                                   //Si és un rei
                            r=r+1000;             //Sumem el seu pes al resultat
                            break;
                        case 7:                                   //Si és un peó negre
                            r=r-pespeon[j][i];    //Restem el seu pes al resultat
```

```
                    break;
                case 8:                     //Si és una torre negre
                    r=r-pestorre[i][j];     //Restem el seu pes al resultat
                    break;
                case 9:                     //Si és un cavall negre
                    r=r-pescavall[i][j];    //Restem el seu pes al resultat
                    break;
                case 10:                    //Si és un alfil negre
                    r=r-pesalfil[i][j];     //Restem el seu pes al resultat
                    break;
                case 11:                    //Si és una reina negre
                    r=r-pesreina[i][j];     //Restem el seu pes al resultat
                    break;
                case 12:                    //Si és un rei negre
                    r=r-1000;               //Restem el seu resultat
                    break;
            }
        }
    }
    return r;                                           //Retornem el valor resultant
}
```

6.5. Conjunt de regles

6.5.1. Moviments

A continuació podrem veure l' implementació del conjunt de regles sobre la funció que obté tots els fills, d'un determinat tauler. Com a codi és bastant feixuga així que per intentar resumir-la diré que el que fa és comprovar quin a quin jugador li toca moure i a partir d'aquí, emprant un tauler auxiliar que cada cop es reseteja, s'elabora una llista de possibles taulers.

Com que aquesta funció és tan llarga i repetitiva només comentaré les parts principals:

```
List<tauler> tauler::obtenirFills(int jugador)          //Aquesta funció ens retornarà una llista de taulers
{
        int a=0;
        bool stop=false;
        List<tauler> llf;
        tauler taux;
        taux.xt=xt;
        taux.yt=yt;
        taux.marcat=marcat;
        taux.torn=torn;
        for(int i=0;i<8;i++)        //Recorrem tot el tauler original per copiar-lo al auxiliar
        {
```

```
        for(int j=0;j<8;j++)
        {
            taux.caselles[i][j]=caselles[i][j];
        }
    }
    for ( int i=0; i<8; i++)                    //Repassem tot el tauler per tal de crear els seus fills
    {
        for ( int j=0; j<8; j++)
        {
            switch (jugador)                    //Filtrem per jugador
            {
                case 1:                         //Si li toca al primer jugador (Blanques)
                    switch (caselles[i][j].numero)      //Filtrem per el tipus de peça
                    {
                        case 1:                         //Si és un peó
                            if(j>0 and caselles[i][j-1].numero==0)  //Apliquem les regles
                            {
                                taux.caselles[i][j].numero=0;       //Realitzem el moviment
                                taux.caselles[i][j-1].numero=1;
                                llf.append(taux);                   //I posem el nou tauler a la llista
                                for(int o=0;o<8;o++)                //Immediatament després reiniciem
                                {                                   //el tauler auxiliar
                                    for(int p=0;p<8;p++)
                                    {
                                        taux.caselles[o][p]=caselles[o][p];
                                    }
                                }
                            }
                            if(j>0 and i>0 and caselles[i-1][j-1].numero>6)   //Apliquem les regles
                            {
                                taux.caselles[i][j].numero=0;
                                taux.caselles[i-1][j-1].numero=1;
                                llf.append(taux);
                                for(int o=0;o<8;o++)
                                {
                                    for(int p=0;p<8;p++)
                                    {
                                        taux.caselles[o][p]=caselles[o][p];
                                    }
                                }
                            }
                            if(j>0 and i<8 and caselles[i+1][j-1].numero>6)
                            {
                                taux.caselles[i][j].numero=0;
                                taux.caselles[i-1][j-1].numero=1;
```

```
                    llf.append(taux);
                    for(int o=0;o<8;o++)
                    {
                        for(int p=0;p<8;p++)
                        {
                            taux.caselles[o][p]=caselles[o][p];
                        }
                    }
                }
                if(j==6 and caselles[i][j+2].numero==0)
                {
                    taux.caselles[i][j].numero=0;
                    taux.caselles[i][j-2].numero=1;
                    llf.append(taux);
                    for(int o=0;o<8;o++)
                    {
                        for(int p=0;p<8;p++)
                        {
                            taux.caselles[o][p]=caselles[o][p];
                        }
                    }
                }
                break;
            case 2:
                a=0;
                stop=false;
                while(!stop)
                {
                    if(j-a>=0)
                    {
                        if(taux.caselles[i][j-a].numero==0)
                        {
                            taux.caselles[i][j].numero=0;
                            taux.caselles[i][j-a].numero=2;
                            llf.append(taux);
                            for(int o=0;o<8;o++)
                            {
                                for(int p=0;p<8;p++)
                                {
taux.caselles[o][p]=caselles[o][p];
                                }
                            }
                        }
                        else
                        {
```

```
                    if(taux.caselles[i][j-a].numero>6)
                    {
                                taux.caselles[i][j].numero=0;
                                taux.caselles[i][j-a].numero=2;
                                llf.append(taux);
                                for(int o=0;o<8;o++)
                                {
                                    for(int p=0;p<8;p++)
                                    {
taux.caselles[o][p]=caselles[o][p];
                                    }
                                }
                                stop=true;
                    }
                    else stop=true;
                }
                a++;
            }
            else stop=true;
        }
        stop=false;
        a=1;
        while(!stop)
        {
            if(j+a<8)
            {
                if(taux.caselles[i][j+a].numero==0)
                {
                        taux.caselles[i][j].numero=0;
                        taux.caselles[i][j+a].numero=2;
                        llf.append(taux);
                        for(int o=0;o<8;o++)
                        {
                            for(int p=0;p<8;p++)
                            {
taux.caselles[o][p]=caselles[o][p];
                            }
                        }
                }
                else
                {
                    if(taux.caselles[i][j+a].numero>6)
                    {
                        taux.caselles[i][j].numero=0;
                        taux.caselles[i][j+a].numero=2;
```

```
                                                llf.append(taux);
                                                for(int o=0;o<8;o++)
                                                {
                                                    for(int p=0;p<8;p++)
                        {
 taux.caselles[o][p]=caselles[o][p];
                                                    }
                                                }
                                                stop=true;
                                    }
                                    else stop=true;
                                }
                                a++;
                            }
                            else stop=true;
                        }
                        stop=false;
                        a=1;
                        while(!stop)
                        {
                            if(i-a>=0)
                            {
                                if(taux.caselles[i-a][j].numero==0)
                                {
                                                taux.caselles[i][j].numero=0;
                                                taux.caselles[i-a][j].numero=2;
                                                llf.append(taux);
                                                for(int o=0;o<8;o++)
                                                {
                                                    for(int p=0;p<8;p++)
                                                    {
taux.caselles[o][p]=caselles[o][p];
                                                    }
                                                }
                                }
                                else
                                {
                                    if(taux.caselles[i-a][j].numero>6)
                                    {
                                                taux.caselles[i][j].numero=0;
                                                taux.caselles[i-a][j].numero=2;
                                                llf.append(taux);
                                                for(int o=0;o<8;o++)
                                                {
                                                    for(int p=0;p<8;p++)
```

```
                                                    {
taux.caselles[o][p]=caselles[o][p];
                                                    }
                                                }
                                                stop=true;
                                            }
                                            else stop=true;
                                        }
                                        a++;
                                    }
                                    else stop=true;
                                }
                                stop=false;
                                a=1;
                                while(!stop)
                                {
                                    if(i+a<8)
                                    {
                                        if(taux.caselles[i+a][j].numero==0)
                                        {
                                                taux.caselles[i][j].numero=0;
                                                taux.caselles[i+a][j].numero=2;
                                                llf.append(taux);
                                                for(int o=0;o<8;o++)
                                                {
                                                    for(int p=0;p<8;p++)
                                                    {
taux.caselles[o][p]=caselles[o][p];
                                                    }
                                                }
                                        }
                                        else
                                        {
                                            if(taux.caselles[i+a][j].numero>6)
                                            {
                                                taux.caselles[i][j].numero=0;
                                                taux.caselles[i+a][j].numero=2;
                                                llf.append(taux);
                                                for(int o=0;o<8;o++)
                                                {
                                                    for(int p=0;p<8;p++)
                                                    {
taux.caselles[o][p]=caselles[o][p];
                                                    }
                                                }
```

```
                                                stop=true;
                                }
                                else stop=true;
                            }
                            a++;
                        }
                        else stop=true;
                    }
                    break;
                case 3:
                    if(i<6 && j<7 && (taux.caselles[i+2][j+1].numero==0 ||
taux.caselles[i+2][j+1].numero>6))
                    {
                        taux.caselles[i][j].numero=0;
                        taux.caselles[i+2][j+1].numero=3;
                        llf.append(taux);
                        for(int o=0;o<8;o++)
                        {
                            for(int p=0;p<8;p++)
                            {
                                taux.caselles[o][p]=caselles[o][p];
                            }
                        }
                    }
                    if(i<6 && j>0 && (taux.caselles[i+2][j-1].numero==0 || taux.caselles[i+2][j-
1].numero>6))
                    {
                        taux.caselles[i][j].numero=0;
                        taux.caselles[i+2][j-1].numero=3;
                        llf.append(taux);
                        for(int o=0;o<8;o++)
                        {
                            for(int p=0;p<8;p++)
                            {
                                taux.caselles[o][p]=caselles[o][p];
                            }
                        }
                    }
                    if(i>1 && j<7 && (taux.caselles[i-2][j+1].numero==0 || taux.caselles[i-
2][j+1].numero>6))
                    {
                        taux.caselles[i][j].numero=0;
                        taux.caselles[i-2][j+1].numero=3;
                        llf.append(taux);
                        for(int o=0;o<8;o++)
```

```
                    {
                        for(int p=0;p<8;p++)
                        {
                            taux.caselles[o][p]=caselles[o][p];
                        }
                    }
                }
                if(i>1 && j>0 && (taux.caselles[i-2][j-1].numero==0 || taux.caselles[i-2][j-
1].numero>6))
                {
                    taux.caselles[i][j].numero=0;
                    taux.caselles[i-2][j-1].numero=3;
                    llf.append(taux);
                    for(int o=0;o<8;o++)
                    {
                        for(int p=0;p<8;p++)
                        {
                            taux.caselles[o][p]=caselles[o][p];
                        }
                    }
                }
                if(i<7 && j<6 && (taux.caselles[i+1][j+2].numero==0 ||
taux.caselles[i+1][j+2].numero>6))
                {
                    taux.caselles[i][j].numero=0;
                    taux.caselles[i+1][j+2].numero=3;
                    llf.append(taux);
                    for(int o=0;o<8;o++)
                    {
                        for(int p=0;p<8;p++)
                        {
                            taux.caselles[o][p]=caselles[o][p];
                        }
                    }
                }
                if(i<7 && j>1 && (taux.caselles[i+1][j-2].numero==0 || taux.caselles[i+1][j-
2].numero>6))
                {
                    taux.caselles[i][j].numero=0;
                    taux.caselles[i+1][j-2].numero=3;
                    llf.append(taux);
                    for(int o=0;o<8;o++)
                    {
                        for(int p=0;p<8;p++)
                        {
```

```
                    taux.caselles[o][p]=caselles[o][p];
                }
            }
        }
        if(i>0 && j<6 && (taux.caselles[i-1][j+2].numero==0 || taux.caselles[i-
1][j+2].numero>6))
        {
            taux.caselles[i][j].numero=0;
            taux.caselles[i-1][j+2].numero=3;
            llf.append(taux);
            for(int o=0;o<8;o++)
            {
                for(int p=0;p<8;p++)
                {
                    taux.caselles[o][p]=caselles[o][p];
                }
            }
        }
        if(i>0 && j>1 && (taux.caselles[i-1][j-2].numero==0 || taux.caselles[i-1][j-
2].numero>6))
        {
            taux.caselles[i][j].numero=0;
            taux.caselles[i-1][j-2].numero=3;
            llf.append(taux);
            for(int o=0;o<8;o++)
            {
                for(int p=0;p<8;p++)
                {
                    taux.caselles[o][p]=caselles[o][p];
                }
            }
        }
        break;
    case 4:
        stop=false;
        a=1;
        while(!stop)
        {
            if(i+a<8 && j+a<8)
            {
                if(taux.caselles[i+a][j+a].numero==0)
                {
                        taux.caselles[i][j].numero=0;
                        taux.caselles[i+a][j+a].numero=4;
                        llf.append(taux);
```

```
                                                    for(int o=0;o<8;o++)
                                                    {
                                                        for(int p=0;p<8;p++)
                                                        {
taux.caselles[o][p]=caselles[o][p];
                                                        }
                                                    }
                                        }
                                        else
                                        {
                                            if (taux.caselles[i+a][j+a].numero > 6)
                                            {
                                                    taux.caselles[i][j].numero=0;
                                                    taux.caselles[i+a][j+a].numero=4;
                                                    llf.append(taux);
                                                    for(int o=0;o<8;o++)
                                                    {
                                                        for(int p=0;p<8;p++)
                                                        {
taux.caselles[o][p]=caselles[o][p];
                                                        }
                                                    }
                                                    stop=true;
                                            }
                                            else stop=true;
                                        }
                                        a++;
                                    }
                                    else stop=true;
                                }
                                stop=false;
                                i=1;
                                while(!stop)
                                {
                                    if(i-a>=0 && j+a<8)
                                    {
                                        if(taux.caselles[i-a][j+a].numero==0)
                                        {
                                                    taux.caselles[i][j].numero=0;
                                                    taux.caselles[i-a][j+a].numero=4;
                                                    llf.append(taux);
                                                    for(int o=0;o<8;o++)
                                                    {
                                                        for(int p=0;p<8;p++)
                                                        {
```

```
taux.caselles[o][p]=caselles[o][p];
                                                    }
                                                }
                                        }
                                        else
                                        {
                                            if (taux.caselles[i-a][j+a].numero > 6)
                                            {
                                                                taux.caselles[i][j].numero=0;
                                                                taux.caselles[i-a][j+a].numero=4;
                                                                llf.append(taux);
                                                                for(int o=0;o<8;o++)
                                                                {
                                                                    for(int p=0;p<8;p++)
                                                                    {
taux.caselles[o][p]=caselles[o][p];
                                                                    }
                                                                }
                                                                stop=true;
                                            }
                                            else stop=true;
                                        }
                                        a++;
                                    }
                                    else stop=true;
                                }
                                stop=false;
                                i=1;
                                while(!stop)
                                {
                                    if(i-a>=0 && j-a>=0)
                                    {
                                        if(taux.caselles[i-a][j-a].numero==0)
                                        {
                                                                taux.caselles[i][j].numero=0;
                                                                taux.caselles[i-a][j-a].numero=4;
                                                                llf.append(taux);
                                                                for(int o=0;o<8;o++)
                                                                {
                                                                    for(int p=0;p<8;p++)
                                                                    {
                                                                        taux.caselles[o][p]=caselles[o][p];
                                                                    }
                                                                }
                                        }
```

```
                else
                {
                    if (taux.caselles[i-a][j-a].numero > 6)
                    {
                                taux.caselles[i][j].numero=0;
                                taux.caselles[i-a][j-a].numero=4;
                                llf.append(taux);
                                for(int o=0;o<8;o++)
                                {
                                    for(int p=0;p<8;p++)
                                    {
                                        taux.caselles[o][p]=caselles[o][p];
                                    }
                                }
                                stop=true;
                    }
                    else stop=true;
                }
                a++;
            }
            else stop=true;
        }
        stop=false;
        a=1;
        while(!stop)
        {
            if(i+a<8 && j-a>=0)
            {
                if(taux.caselles[i+a][j-a].numero==0)
                {
                                taux.caselles[i][j].numero=0;
                                taux.caselles[i+a][j-a].numero=4;
                                llf.append(taux);
                                for(int o=0;o<8;o++)
                                {
                                    for(int p=0;p<8;p++)
                                    {
                                        taux.caselles[o][p]=caselles[o][p];
                                    }
                                }
                }
                else
                {
                    if (taux.caselles[i+a][j-a].numero > 6)
                    {
```

```
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i+a][j-a].numero=4;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                            for(int p=0;p<8;p++)
                                            {
                                                taux.caselles[o][p]=caselles[o][p];
                                            }
                                        }
                                        stop=true;
                                    }
                                    else stop=true;
                                }
                                a++;
                            }else stop=true;
                        }
                        break;
                    case 5:
                        a=0;
                        stop=false;
                        while(!stop)
                        {
                            if(j-a>=0)
                            {
                                if(taux.caselles[i][j-a].numero==0)
                                {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i][j-a].numero=5;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                            for(int p=0;p<8;p++)
                                            {
                                                taux.caselles[o][p]=caselles[o][p];
                                            }
                                        }
                                }
                                else
                                {
                                    if(taux.caselles[i][j-a].numero>6)
                                    {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i][j-a].numero=5;
                                        llf.append(taux);
```

```
                                        for(int o=0;o<8;o++)
                                        {
                                            for(int p=0;p<8;p++)
                                            {
                                                taux.caselles[o][p]=caselles[o][p];
                                            }
                                        }
                                        stop=true;
                                    }
                                    else stop=true;
                                }
                                a++;
                            }
                            else stop=true;
                        }
                        stop=false;
                        a=1;
                        while(!stop)
                        {
                            if(j+a<8)
                            {
                                if(taux.caselles[i][j+a].numero==0)
                                {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i][j+a].numero=5;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                            for(int p=0;p<8;p++)
                                            {
                                                taux.caselles[o][p]=caselles[o][p];
                                            }
                                        }
                                }
                                else
                                {
                                    if(taux.caselles[i][j+a].numero>6)
                                    {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i][j+a].numero=5;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                            for(int p=0;p<8;p++)
                                            {
```

```
                                        taux.caselles[o][p]=caselles[o][p];
                                    }
                                }
                                stop=true;
                            }
                            else stop=true;
                        }
                        a++;
                    }
                    else stop=true;
                }
                stop=false;
                a=1;
                while(!stop)
                {
                    if(i-a>=0)
                    {
                        if(taux.caselles[i-a][j].numero==0)
                        {
                                taux.caselles[i][j].numero=0;
                                taux.caselles[i-a][j].numero=5;
                                llf.append(taux);
                                for(int o=0;o<8;o++)
                                {
                                    for(int p=0;p<8;p++)
                                    {
                                        taux.caselles[o][p]=caselles[o][p];
                                    }
                                }
                        }
                        else
                        {
                            if(taux.caselles[i-a][j].numero>6)
                            {
                                taux.caselles[i][j].numero=0;
                                taux.caselles[i-a][j].numero=5;
                                llf.append(taux);
                                for(int o=0;o<8;o++)
                                {
                                    for(int p=0;p<8;p++)
                                    {
                                        taux.caselles[o][p]=caselles[o][p];
                                    }
                                }
                                stop=true;
```

```
                    }
                    else stop=true;
                }
                a++;
            }
            else stop=true;
        }
        stop=false;
        a=1;
        while(!stop)
        {
            if(i+a<8)
            {
                if(taux.caselles[i+a][j].numero==0)
                {
                                taux.caselles[i][j].numero=0;
                                taux.caselles[i+a][j].numero=5;
                                llf.append(taux);
                                for(int o=0;o<8;o++)
                                {
                                    for(int p=0;p<8;p++)
                                    {
                                        taux.caselles[o][p]=caselles[o][p];
                                    }
                                }
                }
                else
                {
                    if(taux.caselles[i+a][j].numero>6)
                    {
                                taux.caselles[i][j].numero=0;
                                taux.caselles[i+a][j].numero=5;
                                llf.append(taux);
                                for(int o=0;o<8;o++)
                                {
                                    for(int p=0;p<8;p++)
                                    {
                                        taux.caselles[o][p]=caselles[o][p];
                                    }
                                }
                                stop=true;
                    }
                    else stop=true;
                }
                a++;
```

```
        }
        else stop=true;
    }
    stop=false;
    a=1;
    while(!stop)
    {
        if(i+a<8 && j+a<8)
        {
            if(taux.caselles[i+a][j+a].numero==0)
            {
                taux.caselles[i][j].numero=0;
                taux.caselles[i+a][j+a].numero=5;
                llf.append(taux);
                for(int o=0;o<8;o++)
                {
                    for(int p=0;p<8;p++)
                    {
                        taux.caselles[o][p]=caselles[o][p];
                    }
                }
            }
            else
            {
              if (taux.caselles[i+a][j+a].numero > 6)
              {
                taux.caselles[i][j].numero=0;
                taux.caselles[i+a][j+a].numero=5;
                llf.append(taux);
                for(int o=0;o<8;o++)
                {
                    for(int p=0;p<8;p++)
                    {
                        taux.caselles[o][p]=caselles[o][p];
                    }
                }
                stop=true;
              }
              else stop=true;
            }
            a++;
        }
        else stop=true;
    }
    stop=false;
```

```
i=1;
while(!stop)
{
        if(i-a>=0 && j+a<8)
        {
                if(taux.caselles[i-a][j+a].numero==0)
                {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i-a][j+a].numero=5;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                                for(int p=0;p<8;p++)
                                                {
                                                        taux.caselles[o][p]=caselles[o][p];
                                                }
                                        }
                }
                else
                {
                   if (taux.caselles[i-a][j+a].numero > 6)
                   {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i-a][j+a].numero=5;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                                for(int p=0;p<8;p++)
                                                {
                                                        taux.caselles[o][p]=caselles[o][p];
                                                }
                                        }
                                        stop=true;
                   }
                   else stop=true;
                }
                a++;
        }
        else stop=true;
}
stop=false;
i=1;
while(!stop)
{
        if(i-a>=0 && j-a>=0)
```

```
{
        if(taux.caselles[i-a][j-a].numero==0)
        {
                                taux.caselles[i][j].numero=0;
                                taux.caselles[i-a][j-a].numero=5;
                                llf.append(taux);
                                for(int o=0;o<8;o++)
                                {
                                        for(int p=0;p<8;p++)
                                        {
                                                taux.caselles[o][p]=caselles[o][p];
                                        }
                                }
        }
        else
        {
           if (taux.caselles[i-a][j-a].numero > 6)
           {
                                taux.caselles[i][j].numero=0;
                                taux.caselles[i-a][j-a].numero=5;
                                llf.append(taux);
                                for(int o=0;o<8;o++)
                                {
                                        for(int p=0;p<8;p++)
                                        {
                                                taux.caselles[o][p]=caselles[o][p];
                                        }
                                }
                                stop=true;
           }
           else stop=true;
        }
        a++;
}
else stop=true;
}
stop=false;
a=1;
while(!stop)
{
        if(i+a<8 && j-a>=0)
        {
                if(taux.caselles[i+a][j-a].numero==0)
                {
                                taux.caselles[i][j].numero=0;
```

```
                                        taux.caselles[i+a][j-a].numero=5;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                            for(int p=0;p<8;p++)
                                            {
                                                taux.caselles[o][p]=caselles[o][p];
                                            }
                                        }
                        }
                        else
                        {
                            if (taux.caselles[i+a][j-a].numero > 6)
                            {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i+a][j-a].numero=5;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                            for(int p=0;p<8;p++)
                                            {
                                                taux.caselles[o][p]=caselles[o][p];
                                            }
                                        }
                                        stop=true;
                            }
                            else stop=true;
                        }
                        a++;
                }else stop=true;
            }
            break;
        case 6:
            if(j<7 && (taux.caselles[i][j+1].numero==0 || taux.caselles[i][j+1].numero>6))
            {
                taux.caselles[i][j].numero=0;
                taux.caselles[i][j+1].numero=6;
                llf.append(taux);
                for(int o=0;o<8;o++)
                {
                    for(int p=0;p<8;p++)
                    {
                        taux.caselles[o][p]=caselles[o][p];
                    }
                }
```

```
}
if(j>0 && (taux.caselles[i][j-1].numero==0 || taux.caselles[i][j-1].numero>6))
{
    taux.caselles[i][j].numero=0;
    taux.caselles[i][j-1].numero=6;
    llf.append(taux);
    for(int o=0;o<8;o++)
    {
        for(int p=0;p<8;p++)
        {
            taux.caselles[o][p]=caselles[o][p];
        }
    }
}
if(i<7 && (taux.caselles[i+1][j].numero==0 || taux.caselles[i+1][j].numero>6))
{
    taux.caselles[i][j].numero=0;
    taux.caselles[i+1][j].numero=6;
    llf.append(taux);
    for(int o=0;o<8;o++)
    {
        for(int p=0;p<8;p++)
        {
            taux.caselles[o][p]=caselles[o][p];
        }
    }
}
if(i>0 && (taux.caselles[i-1][j].numero==0 || taux.caselles[i-1][j].numero>6))
{
    taux.caselles[i][j].numero=0;
    taux.caselles[i-1][j].numero=6;
    llf.append(taux);
    for(int o=0;o<8;o++)
    {
        for(int p=0;p<8;p++)
        {
            taux.caselles[o][p]=caselles[o][p];
        }
    }
}
if(i<7 && j<7 && (taux.caselles[i+1][j+1].numero==0 ||
taux.caselles[i+1][j+1].numero>6))
{
    taux.caselles[i][j].numero=0;
    taux.caselles[i+1][j+1].numero=6;
```

```
llf.append(taux);
for(int o=0;o<8;o++)
{
    for(int p=0;p<8;p++)
    {
        taux.caselles[o][p]=caselles[o][p];
    }
}
}
if(i>0 && j<7 && (taux.caselles[i-1][j+1].numero==0 || taux.caselles[i-
1][j+1].numero>6))
{
    taux.caselles[i][j].numero=0;
    taux.caselles[i-1][j+1].numero=6;
    llf.append(taux);
    for(int o=0;o<8;o++)
    {
        for(int p=0;p<8;p++)
        {
            taux.caselles[o][p]=caselles[o][p];
        }
    }
}
if(i<7 && j>0 && (taux.caselles[i+1][j-1].numero==0 || taux.caselles[i+1][j-
1].numero>6))
{
    taux.caselles[i][j].numero=0;
    taux.caselles[i+1][j-1].numero=6;
    llf.append(taux);
    for(int o=0;o<8;o++)
    {
        for(int p=0;p<8;p++)
        {
            taux.caselles[o][p]=caselles[o][p];
        }
    }
}
if(i>0 && j>0 && (taux.caselles[i-1][j-1].numero==0 || taux.caselles[i-1][j-
1].numero>6))
{
    taux.caselles[i][j].numero=0;
    taux.caselles[i-1][j-11].numero=6;
    llf.append(taux);
    for(int o=0;o<8;o++)
    {
```

```
                    for(int p=0;p<8;p++)
                    {
                        taux.caselles[o][p]=caselles[o][p];
                    }
                }
            }
            break;
        }
        break;
    case 2:
        switch(caselles[i][j].numero)
        {
            case 7:
                if(j<7 and  caselles[i][j+1].numero==0)
                {
                    taux.caselles[i][j].numero=0;
                    taux.caselles[i][j+1].numero=7;
                    llf.append(taux);
                    for(int o=0;o<8;o++)
                    {
                        for(int p=0;p<8;p++)
                        {
                            taux.caselles[o][p]=caselles[o][p];
                        }
                    }
                }
                if(j<7 and i>0 and caselles[i-1][j+1].numero<7)
                {
                    taux.caselles[i][j].numero=0;
                    taux.caselles[i-1][j+1].numero=7;
                    llf.append(taux);
                    for(int o=0;o<8;o++)
                    {
                        for(int p=0;p<8;p++)
                        {
                            taux.caselles[o][p]=caselles[o][p];
                        }
                    }
                }
                if(j<7 and i<8 and caselles[i+1][j+1].numero<7)
                {
                    taux.caselles[i][j].numero=0;
                    taux.caselles[i+1][j+1].numero=7;
                    llf.append(taux);
                    for(int o=0;o<8;o++)
```

```
            {
                for(int p=0;p<8;p++)
                {
                    taux.caselles[o][p]=caselles[o][p];
                }
            }
        }
        if(j==1 and caselles[i][j+2].numero==0)
        {
            taux.caselles[i][j].numero=0;
            taux.caselles[i][j+2].numero=7;
            llf.append(taux);
            for(int o=0;o<8;o++)
            {
                for(int p=0;p<8;p++)
                {
                    taux.caselles[o][p]=caselles[o][p];
                }
            }
        }
        break;
    case 8:
        a=0;
        stop=false;
        while(!stop)
        {
            if(j-a>=0)
            {
                if(taux.caselles[i][j-a].numero==0)
                {
                        taux.caselles[i][j].numero=0;
                        taux.caselles[i][j-a].numero=8;
                        llf.append(taux);
                        for(int o=0;o<8;o++)
                        {
                            for(int p=0;p<8;p++)
                            {
                                taux.caselles[o][p]=caselles[o][p];
                            }
                        }
                }
                else
                {
                    if(taux.caselles[i][j-a].numero<7)
                    {
```

```
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i][j-a].numero=8;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                            for(int p=0;p<8;p++)
                                            {
                                                taux.caselles[o][p]=caselles[o][p];
                                            }
                                        }
                                        stop=true;
                                    }
                                    else stop=true;
                                }
                                a++;
                            }
                            else stop=true;
                        }
                        stop=false;
                        a=1;
                        while(!stop)
                        {
                            if(j+a<8)
                            {
                                if(taux.caselles[i][j+a].numero==0)
                                {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i][j+a].numero=8;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                            for(int p=0;p<8;p++)
                                            {
                                                taux.caselles[o][p]=caselles[o][p];
                                            }
                                        }
                                }
                                else
                                {
                                    if(taux.caselles[i][j+a].numero<7)
                                    {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i][j+a].numero=8;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
```

```
                                        {
                                            for(int p=0;p<8;p++)
                                            {
                                                taux.caselles[o][p]=caselles[o][p];
                                            }
                                        }
                                        stop=true;
                                    }
                                    else stop=true;
                                }
                                a++;
                            }
                            else stop=true;
                        }
                        stop=false;
                        a=1;
                        while(!stop)
                        {
                            if(i-a>=0)
                            {
                                if(taux.caselles[i-a][j].numero==0)
                                {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i-a][j].numero=8;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                            for(int p=0;p<8;p++)
                                            {
                                                taux.caselles[o][p]=caselles[o][p];
                                            }
                                        }
                                }
                                else
                                {
                                    if(taux.caselles[i-a][j].numero<7)
                                    {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i-a][j].numero=8;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                            for(int p=0;p<8;p++)
                                            {
                                                taux.caselles[o][p]=caselles[o][p];
```

```
                                        }
                                    }
                                    stop=true;
                                }
                                else stop=true;
                            }
                            a++;
                        }
                        else stop=true;
                    }
                    stop=false;
                    a=1;
                    while(!stop)
                    {
                        if(i+a<8)
                        {
                            if(taux.caselles[i+a][j].numero==0)
                            {
                                    taux.caselles[i][j].numero=0;
                                    taux.caselles[i+a][j].numero=8;
                                    llf.append(taux);
                                    for(int o=0;o<8;o++)
                                    {
                                        for(int p=0;p<8;p++)
                                        {
                                            taux.caselles[o][p]=caselles[o][p];
                                        }
                                    }
                            }
                            else
                            {
                                if(taux.caselles[i+a][j].numero<7)
                                {
                                    taux.caselles[i][j].numero=0;
                                    taux.caselles[i+a][j].numero=8;
                                    llf.append(taux);
                                    for(int o=0;o<8;o++)
                                    {
                                        for(int p=0;p<8;p++)
                                        {
                                            taux.caselles[o][p]=caselles[o][p];
                                        }
                                    }
                                    stop=true;
                                }
```

```
                    else stop=true;
                }
                a++;
            }
            else stop=true;
        }
        break;
    case 9:
        if(i<6 && j<7 && (taux.caselles[i+2][j+1].numero==0 ||
taux.caselles[i+2][j+1].numero<7))
        {
            taux.caselles[i][j].numero=0;
            taux.caselles[i+2][j+1].numero=9;
            llf.append(taux);
            for(int o=0;o<8;o++)
            {
                for(int p=0;p<8;p++)
                {
                    taux.caselles[o][p]=caselles[o][p];
                }
            }
        }
        if(i<6 && j>0 && (taux.caselles[i+2][j-1].numero==0 || taux.caselles[i+2][j-
1].numero<7))
        {
            taux.caselles[i][j].numero=0;
            taux.caselles[i+2][j-1].numero=9;
            llf.append(taux);
            for(int o=0;o<8;o++)
            {
                for(int p=0;p<8;p++)
                {
                    taux.caselles[o][p]=caselles[o][p];
                }
            }
        }
        if(i>1 && j<7 && (taux.caselles[i-2][j+1].numero==0 || taux.caselles[i-
2][j+1].numero<7))
        {
            taux.caselles[i][j].numero=0;
            taux.caselles[i-2][j+1].numero=9;
            llf.append(taux);
            for(int o=0;o<8;o++)
            {
                for(int p=0;p<8;p++)
```

```
                    {
                        taux.caselles[o][p]=caselles[o][p];
                    }
                }
            }
            if(i>1 && j>0 && (taux.caselles[i-2][j-1].numero==0 || taux.caselles[i-2][j-
1].numero<7))
            {
                taux.caselles[i][j].numero=0;
                taux.caselles[i-2][j-1].numero=9;
                llf.append(taux);
                for(int o=0;o<8;o++)
                {
                    for(int p=0;p<8;p++)
                    {
                        taux.caselles[o][p]=caselles[o][p];
                    }
                }
            }
            if(i<7 && j<6 && (taux.caselles[i+1][j+2].numero==0 ||
taux.caselles[i+1][j+2].numero<7))
            {
                taux.caselles[i][j].numero=0;
                taux.caselles[i+1][j+2].numero=9;
                llf.append(taux);
                for(int o=0;o<8;o++)
                {
                    for(int p=0;p<8;p++)
                    {
                        taux.caselles[o][p]=caselles[o][p];
                    }
                }
            }
            if(i<7 && j>1 && (taux.caselles[i+1][j-2].numero==0 || taux.caselles[i+1][j-
2].numero<7))
            {
                taux.caselles[i][j].numero=0;
                taux.caselles[i+1][j-2].numero=9;
                llf.append(taux);
                for(int o=0;o<8;o++)
                {
                    for(int p=0;p<8;p++)
                    {
                        taux.caselles[o][p]=caselles[o][p];
                    }
```

```
                    }
                }
                if(i>0 && j<6 && (taux.caselles[i-1][j+2].numero==0 || taux.caselles[i-
1][j+2].numero<7))
                {
                    taux.caselles[i][j].numero=0;
                    taux.caselles[i-1][j+2].numero=9;
                    llf.append(taux);
                    for(int o=0;o<8;o++)
                    {
                        for(int p=0;p<8;p++)
                        {
                            taux.caselles[o][p]=caselles[o][p];
                        }
                    }
                }
                if(i>0 && j>1 && (taux.caselles[i-1][j-2].numero==0 || taux.caselles[i-1][j-
2].numero<7))
                {
                    taux.caselles[i][j].numero=0;
                    taux.caselles[i-1][j-2].numero=9;
                    llf.append(taux);
                    for(int o=0;o<8;o++)
                    {
                        for(int p=0;p<8;p++)
                        {
                            taux.caselles[o][p]=caselles[o][p];
                        }
                    }
                }
                break;
            case 10:
                stop=false;
                a=1;
                while(!stop)
                {
                    if(i+a<8 && j+a<8)
                    {
                        if(taux.caselles[i+a][j+a].numero==0)
                        {
                            taux.caselles[i][j].numero=0;
                            taux.caselles[i+a][j+a].numero=10;
                            llf.append(taux);
                            for(int o=0;o<8;o++)
                            {
```

```
                                        for(int p=0;p<8;p++)
                                        {
                                            taux.caselles[o][p]=caselles[o][p];
                                        }
                                    }
                    }
                    else
                    {
                        if (taux.caselles[i+a][j+a].numero < 7)
                        {
                                    taux.caselles[i][j].numero=0;
                                    taux.caselles[i+a][j+a].numero=10;
                                    llf.append(taux);
                                    for(int o=0;o<8;o++)
                                    {
                                        for(int p=0;p<8;p++)
                                        {
                                            taux.caselles[o][p]=caselles[o][p];
                                        }
                                    }
                                    stop=true;
                        }
                        else stop=true;
                    }
                    a++;
                }
                else stop=true;
            }
            stop=false;
            i=1;
            while(!stop)
            {
                if(i-a>=0 && j+a<8)
                {
                    if(taux.caselles[i-a][j+a].numero==0)
                    {
                                    taux.caselles[i][j].numero=0;
                                    taux.caselles[i-a][j+a].numero=10;
                                    llf.append(taux);
                                    for(int o=0;o<8;o++)
                                    {
                                        for(int p=0;p<8;p++)
                                        {
                                            taux.caselles[o][p]=caselles[o][p];
                                        }
```

```
                                        }
                        }
                        else
                        {
                            if (taux.caselles[i-a][j+a].numero < 7)
                            {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i-a][j+a].numero=10;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                                for(int p=0;p<8;p++)
                                                {
                                                        taux.caselles[o][p]=caselles[o][p];
                                                }
                                        }
                                        stop=true;
                            }
                            else stop=true;
                        }
                        a++;
                }
                else stop=true;
        }
        stop=false;
        i=1;
        while(!stop)
        {
                if(i-a>=0 && j-a>=0)
                {
                        if(taux.caselles[i-a][j-a].numero==0)
                        {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i-a][j-a].numero=10;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                                for(int p=0;p<8;p++)
                                                {
                                                        taux.caselles[o][p]=caselles[o][p];
                                                }
                                        }
                        }
                        else
                        {
```

```
                    if (taux.caselles[i-a][j-a].numero < 7)
                    {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i-a][j-a].numero=10;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                              for(int p=0;p<8;p++)
                                              {
                                                    taux.caselles[o][p]=caselles[o][p];
                                              }
                                        }
                                        stop=true;
                    }
                    else stop=true;
                  }
                  a++;
            }
            else stop=true;
      }
      stop=false;
      a=1;
      while(!stop)
      {
            if(i+a<8 && j-a>=0)
            {
                  if(taux.caselles[i+a][j-a].numero==0)
                  {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i+a][j-a].numero=10;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                              for(int p=0;p<8;p++)
                                              {
                                                    taux.caselles[o][p]=caselles[o][p];
                                              }
                                        }
                  }
                  else
                  {
                    if (taux.caselles[i+a][j-a].numero < 7)
                    {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i+a][j-a].numero=10;
```

```
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                            for(int p=0;p<8;p++)
                                            {
                                                taux.caselles[o][p]=caselles[o][p];
                                            }
                                        }
                                        stop=true;
                                    }
                                    else stop=true;
                                }
                                a++;
                            }else stop=true;
                        }
                        break;
                    case 11:
                        a=0;
                        stop=false;
                        while(!stop)
                        {
                            if(j-a>=0)
                            {
                                if(taux.caselles[i][j-a].numero==0)
                                {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i][j-a].numero=11;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                            for(int p=0;p<8;p++)
                                            {
                                                taux.caselles[o][p]=caselles[o][p];
                                            }
                                        }
                                }
                                else
                                {
                                    if(taux.caselles[i][j-a].numero < 7)
                                    {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i][j-a].numero=11;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
```

```
                                        for(int p=0;p<8;p++)
                                        {
                                            taux.caselles[o][p]=caselles[o][p];
                                        }
                                    }
                                    stop=true;
                    }
                    else stop=true;
                }
                a++;
            }
            else stop=true;
        }
        stop=false;
        a=1;
        while(!stop)
        {
            if(j+a<8)
            {
                if(taux.caselles[i][j+a].numero==0)
                {
                                    taux.caselles[i][j].numero=0;
                                    taux.caselles[i][j+a].numero=11;
                                    llf.append(taux);
                                    for(int o=0;o<8;o++)
                                    {
                                        for(int p=0;p<8;p++)
                                        {
                                            taux.caselles[o][p]=caselles[o][p];
                                        }
                                    }
                }
                else
                {
                    if(taux.caselles[i][j+a].numero < 7)
                    {
                                    taux.caselles[i][j].numero=0;
                                    taux.caselles[i][j+a].numero=11;
                                    llf.append(taux);
                                    for(int o=0;o<8;o++)
                                    {
                                        for(int p=0;p<8;p++)
                                        {
                                            taux.caselles[o][p]=caselles[o][p];
                                        }
```

```
                                        }
                                        stop=true;
                        }
                        else stop=true;
                    }
                    a++;
            }
            else stop=true;
    }
    stop=false;
    a=1;
    while(!stop)
    {
            if(i-a>=0)
            {
                    if(taux.caselles[i-a][j].numero==0)
                    {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i-a][j].numero=11;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                            for(int p=0;p<8;p++)
                                            {
                                                taux.caselles[o][p]=caselles[o][p];
                                            }
                                        }
                    }
                    else
                    {
                        if(taux.caselles[i-a][j].numero < 7)
                        {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i-a][j].numero=11;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                            for(int p=0;p<8;p++)
                                            {
                                                taux.caselles[o][p]=caselles[o][p];
                                            }
                                        }
                                        stop=true;
                        }
                        else stop=true;
```

```
                }
                a++;
            }
            else stop=true;
        }
        stop=false;
        a=1;
        while(!stop)
        {
            if(i+a<8)
            {
                if(taux.caselles[i+a][j].numero==0)
                {
                                taux.caselles[i][j].numero=0;
                                taux.caselles[i+a][j].numero=11;
                                llf.append(taux);
                                for(int o=0;o<8;o++)
                                {
                                    for(int p=0;p<8;p++)
                                    {
                                        taux.caselles[o][p]=caselles[o][p];
                                    }
                                }
                }
                else
                {
                    if(taux.caselles[i+a][j].numero < 7)
                    {
                                taux.caselles[i][j].numero=0;
                                taux.caselles[i+a][j].numero=11;
                                llf.append(taux);
                                for(int o=0;o<8;o++)
                                {
                                    for(int p=0;p<8;p++)
                                    {
                                        taux.caselles[o][p]=caselles[o][p];
                                    }
                                }
                                stop=true;
                    }
                    else stop=true;
                }
                a++;
            }
            else stop=true;
```

```
}
stop=false;
a=1;
while(!stop)
{
        if(i+a<8 && j+a<8)
        {
                if(taux.caselles[i+a][j+a].numero==0)
                {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i+a][j+a].numero=11;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                                for(int p=0;p<8;p++)
                                                {
                                                        taux.caselles[o][p]=caselles[o][p];
                                                }
                                        }
                }
                else
                {
                   if (taux.caselles[i+a][j+a].numero < 7)
                   {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i+a][j+a].numero=11;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                                for(int p=0;p<8;p++)
                                                {
                                                        taux.caselles[o][p]=caselles[o][p];
                                                }
                                        }
                                        stop=true;
                   }
                   else stop=true;
                }
                a++;
        }
        else stop=true;
}
stop=false;
i=1;
while(!stop)
```

```
{
        if(i-a>=0 && j+a<8)
        {
                if(taux.caselles[i-a][j+a].numero==0)
                {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i-a][j+a].numero=11;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                                for(int p=0;p<8;p++)
                                                {
                                                        taux.caselles[o][p]=caselles[o][p];
                                                }
                                        }
                }
                else
                {
                   if (taux.caselles[i-a][j+a].numero < 7)
                   {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i-a][j+a].numero=11;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                                for(int p=0;p<8;p++)
                                                {
                                                        taux.caselles[o][p]=caselles[o][p];
                                                }
                                        }
                                        stop=true;
                   }
                   else stop=true;
                }
                a++;
        }
        else stop=true;
}
stop=false;
i=1;
while(!stop)
{
        if(i-a>=0 && j-a>=0)
        {
                if(taux.caselles[i-a][j-a].numero==0)
```

```
                {
                                taux.caselles[i][j].numero=0;
                                taux.caselles[i-a][j-a].numero=11;
                                llf.append(taux);
                                for(int o=0;o<8;o++)
                                {
                                        for(int p=0;p<8;p++)
                                        {
                                                taux.caselles[o][p]=caselles[o][p];
                                        }
                                }
                }
                else
                {
                    if (taux.caselles[i-a][j-a].numero < 7)
                    {
                                taux.caselles[i][j].numero=0;
                                taux.caselles[i-a][j-a].numero=11;
                                llf.append(taux);
                                for(int o=0;o<8;o++)
                                {
                                        for(int p=0;p<8;p++)
                                        {
                                                taux.caselles[o][p]=caselles[o][p];
                                        }
                                }
                                stop=true;
                    }
                    else stop=true;
                }
                a++;
        }
        else stop=true;
}
stop=false;
a=1;
while(!stop)
{
        if(i+a<8 && j-a>=0)
        {
                if(taux.caselles[i+a][j-a].numero==0)
                {
                                taux.caselles[i][j].numero=0;
                                taux.caselles[i+a][j-a].numero=11;
                                llf.append(taux);
```

```
                                        for(int o=0;o<8;o++)
                                        {
                                            for(int p=0;p<8;p++)
                                            {
                                                taux.caselles[o][p]=caselles[o][p];
                                            }
                                        }
                        }
                        else
                        {
                            if (taux.caselles[i+a][j-a].numero < 7)
                            {
                                        taux.caselles[i][j].numero=0;
                                        taux.caselles[i+a][j-a].numero=11;
                                        llf.append(taux);
                                        for(int o=0;o<8;o++)
                                        {
                                            for(int p=0;p<8;p++)
                                            {
                                                taux.caselles[o][p]=caselles[o][p];
                                            }
                                        }
                                        stop=true;
                            }
                            else stop=true;
                        }
                        a++;
                }else stop=true;
            }
            break;
        case 12:
            if(j<7 && (taux.caselles[i][j+1].numero==0 || taux.caselles[i][j+1].numero<7))
            {
                taux.caselles[i][j].numero=0;
                taux.caselles[i][j+1].numero=12;
                llf.append(taux);
                for(int o=0;o<8;o++)
                {
                    for(int p=0;p<8;p++)
                    {
                        taux.caselles[o][p]=caselles[o][p];
                    }
                }
            }
            if(j>0 && (taux.caselles[i][j-1].numero==0 || taux.caselles[i][j-1].numero<7))
```

```
{
    taux.caselles[i][j].numero=0;
    taux.caselles[i][j-1].numero=12;
    llf.append(taux);
    for(int o=0;o<8;o++)
    {
        for(int p=0;p<8;p++)
        {
            taux.caselles[o][p]=caselles[o][p];
        }
    }
}
if(i<7 && (taux.caselles[i+1][j].numero==0 || taux.caselles[i+1][j].numero<7))
{
    taux.caselles[i][j].numero=0;
    taux.caselles[i+1][j].numero=12;
    llf.append(taux);
    for(int o=0;o<8;o++)
    {
        for(int p=0;p<8;p++)
        {
            taux.caselles[o][p]=caselles[o][p];
        }
    }
}
if(i>0 && (taux.caselles[i-1][j].numero==0 || taux.caselles[i-1][j].numero<7))
{
    taux.caselles[i][j].numero=0;
    taux.caselles[i-1][j].numero=12;
    llf.append(taux);
    for(int o=0;o<8;o++)
    {
        for(int p=0;p<8;p++)
        {
            taux.caselles[o][p]=caselles[o][p];
        }
    }
}
if(i<7 && j<7 && (taux.caselles[i+1][j+1].numero==0 ||
taux.caselles[i+1][j+1].numero<7))
{
    taux.caselles[i][j].numero=0;
    taux.caselles[i+1][j+1].numero=12;
    llf.append(taux);
    for(int o=0;o<8;o++)
```

```
                    {
                        for(int p=0;p<8;p++)
                        {
                            taux.caselles[o][p]=caselles[o][p];
                        }
                    }
                }
                if(i>0 && j<7 && (taux.caselles[i-1][j+1].numero==0 || taux.caselles[i-
1][j+1].numero<7))
                {
                    taux.caselles[i][j].numero=0;
                    taux.caselles[i-1][j+1].numero=12;
                    llf.append(taux);
                    for(int o=0;o<8;o++)
                    {
                        for(int p=0;p<8;p++)
                        {
                            taux.caselles[o][p]=caselles[o][p];
                        }
                    }
                }
                if(i<7 && j>0 && (taux.caselles[i+1][j-1].numero==0 || taux.caselles[i+1][j-
1].numero<7))
                {
                    taux.caselles[i][j].numero=0;
                    taux.caselles[i+1][j-1].numero=12;
                    llf.append(taux);
                    for(int o=0;o<8;o++)
                    {
                        for(int p=0;p<8;p++)
                        {
                            taux.caselles[o][p]=caselles[o][p];
                        }
                    }
                }
                if(i>0 && j>0 && (taux.caselles[i-1][j-1].numero==0 || taux.caselles[i-1][j-
1].numero<7))
                {
                    taux.caselles[i][j].numero=0;
                    taux.caselles[i-1][j-11].numero=12;
                    llf.append(taux);
                    for(int o=0;o<8;o++)
                    {
                        for(int p=0;p<8;p++)
                        {
```

```
                                        taux.caselles[o][p]=caselles[o][p];
                                    }
                                }
                            }
                            break;
                        }
                        break;
                    }
                }
            }
            return llf;
}
```

6.5.2. Final de partida

A continuació, podrem veure l' implementació de les regles de finalització de joc:

```
void final()
{
    jblanc=false;                           //Variables que ens diuen si hi ha els reis
    jnegre=false;
    for (int i=0; i<8; i++)                 //Repassem tota la matriu del tauler
    {
        for (int j=0; j<8; j++)
        {
            if(t.caselles[i][j].numero==6) jblanc=true;          //Si trobem el rei blanc ho indiquem
                    else if (t.caselles[i][j].numero==12) jnegre=true;    //Si trobem el rei negre ho indiquem
        }
    }
    if(!jblanc)                     //Si no hem trobat al rei blanc
    {
            SDL_Surface *final;
            final = IMG_Load("Img/nguan.png");  //Carreguem la imatge final
            if(final == NULL)
            {
                    cerr << "No podem carregar imatge: "<<SDL_GetError()<< endl;
                    exit(1);
            }
            desti.x = 0;
            desti.y = 0;
            SDL_BlitSurface(final, NULL, pantalla, &desti); //Copiem el fons a la superfície principal
            SDL_Flip(pantalla);   //Actualitzem pantalla
            SDL_FreeSurface(final);
            estat=3;                                             //I passem al estat final
    }
```

```
    else if(!jnegre)                            //Si no hem trobat el rei negre
    {
        SDL_Surface *final;
        final = IMG_Load("Img/bguan.png");        //Carreguem l' imatge final
        if(final == NULL)
        {
                cerr << "No podem carregar imatge: "<<SDL_GetError()<< endl;
                exit(1);
        }
        desti.x = 0;
        desti.y = 0;
        SDL_BlitSurface(final, NULL, pantalla, &desti); //Copiem el fons a la superfície principal
        SDL_Flip(pantalla);   //Actualitzem pantalla
        SDL_FreeSurface(final);
        estat=3;                                                    //I passem a l'estat final
    }
}
```

6.6. Motor d'inferència

Tal com l'apartat indica, passo a mostrar d'implementació del motor d'inferència, o el que és el mateix la funció minimax amb poda:

```
Resultat MINIMAX_ALFABETA( tauler t, int profunditat, int jugador, int umbralUS, int umbralPAS)
{
	Resultat j, resultatSuccessor, resultatMillor;		//Variables auxiliars
	List<tauler> successors;
	if (profunditat=3)	//Si arriba a la profunditat especificada
	{
		j.valor = t.avaluacio();  //Assigna el valor actual
		j.putCami(NULL);		//Acaba el camí
		return j;			//I retorna el resultat
	}
	else				//Sinó
	successors = t.obtenirFills(2);		//Obtenim els successors del tauler
	if (successors.is_empty())		//Si no hi ha successors
	{
			j.valor = t.avaluacio();		//Assigna el valor actual
			j.cami = NULL;		//Acaba el camí
			return j;			//I retorna el resultat
	}
	Else				//Si hi ha successors
	{
	  for(int i=0; i < successors.legth(); i++)	//Per cada successor...
	  {
		resultatSuccesor = MINIMAX_ALFABETA (successor.go_to(i), profunditat+1, CONTRARI(jugador),
umbralPAS, umbralUS);	//Obtenim els resultats recursivament
		if (umbralPAS < resultatSucesor.valor)		//Si el valor del resultat és major que
		{						//l'umbral de pas
			umbralPAS = resultatSuccessor.valor;	//Li assignem el nou resultat
			resultatMillor.putCami(resultatSuccessor.cami);		//El nou camí
			resultatMillor.append(successor.go_to(i));	//I li afegim el pas actual al camí
		}
		if (umbralPAS >= umbralUS)		//En cas de que l'umbra usat sigui inferior al de pas
		{
			resultatMillor.valor = umbralPAS;	//Assignem el valor
			return resultatMillor;			//I retornem (Poda)
		}
	  }
	  resultatMillor.valor = umbralPAS;		//Assignem l'umbral de pas al millor valor
	  return resultatMillor;		//I retornem el millor valor i camí.
	}
	}
}
```

7. Millores i Conclusions

Com a finalització de la memòria m'agradaria exposar el que per mi serien les principals millores que es podrien dur a terme i les conclusions que en trec d'haver fet aquest projecte.

7.1. Millores

M'agradaria comentar que les millores que exposo a continuació no les he implementat no per la seva extrema dificultat, ja que totes les he estudiat i he cercat informació per poder-les realitzar, sinó per que la seva implementació és molt laboriosa i m'hagués portat el doble de temps el dur-la a terme.

7.1.1. Aprenentatge

Una part molt important i que m'hagués agradat molt dissenyar i implementar és tot l'apartat d'aprenentatge dins d'intel·ligència artificial ja que és un apartat que m'interessa molt i és la primera millora que hagués fet, aquesta millora, l'hagués fet tot fent servir una base de dades on hi hagués implementat dos mètodes de funcionament.

El primer pas hagués set entrar a la base de dades les partides o jugades més famoses dels millors jugadors del mon amb el que ens estalviaríem tot el pas de calcular el minimax i la funció d'avaluació sempre que el tauler ens coincidís amb alguna jugada que tinguéssim entrada a la base de dades ja que al ser una jugada dels millors jugadors del mon, en principi ens creuríem que és una bona jugada.

El segon pas seria crear el que pròpiament seria l'aprenentatge, en aquest cas l'algorisme que ho controlaria, a part de triar la millor jugada a realitzar, hauria de separar la partida en jugades i anar comptant les vegades que és realitza certa jugada per tal de saber quina és una millor jugada, a l'hora que controla si la jugada dona bons resultats, per tal d'aplicar-la a futures partides.

7.1.2. Connexió en xarxa

La segona millora que estava interessat en implementar és la de tenir possibilitat de fer partides on-line, de fet l'aplicació tal com està programada ara mateix, podria interpretar una trama de 64 enters que es correspondria a les 64 caselles del tauler i les seves peces. Per tant si es volgués fer partides on-line s'hauria d'instal·lar l'aplicació a les dues màquines i llavors establir una connexió via sockets. Llavors, s'hauria de modificar el *game loop* del joc per que al final de cada torn s'enviés la trama del tauler i la informació addicional del joc.

7.1.3. Altres millores

A part de les millores esmentades, estic totalment segur de que es podria millorar aspectes concrets de l'aplicació com la funció d'avaluació, més complerta o un aspecte gràfic millorat.

7.2. Conclusions

Com a conclusió d'aquest treball final de carrera, m'agradaria esmentar que aquest no ha set com un dels múltiples treballs que he realitzat durant la carrera, aquest ha set un treball on he agut d'aplicar moltes de les coses apreses al llarg de la carrera i on he agut de cercar molta altra informació externa amb la que he pogut aprendre moltes coses interessants tant relacionades amb el projecte que he realitzat com amb altres aspectes com entorns de programació, mètodes de programació, gràfics, aplicacions útils, etc.

Ha estat un treball on he pogut experimentar sobretot amb aspectes de l'informàtica com són l' intel·ligència artificial, programació o la programació metòdica. Fent que consolidi encara més els coneixements que vaig adquirir en aquestes assignatures.

8. Bibliografia

ESCACS, Recull de partides, adreces de clubs, curs d'escacs, diccionaris i enllaços. 2002.
http://arcade.ya.com/escacs2002/ [Consulta: Juliol de 2008]

RIBADAS PENA, FRANCISCO JOSÉ. *Búsqueda en Espacios de Estados.* 2004.
http://ccia.ei.uvigo.es/docencia/IA/Tema2.pdf [Consulta: Agost de 2008]

CRIADO, AMADOR. *La comunidad de creadores de videojuegos*. 2008.
http://www.callosenlosdedos.org [Consulta: Agost de 2008]

ANTÓN SOLÀ, DOLORS. *Apunts intel·ligència artificial i metodologia de projectes*. 2008.
http://www.uvic.cat [Consulta: Setembre de 2008]

GARCÍA ALBA, ANTONIO. *Tutorial de libSDL para la programación de videojuegos*. 2007.
http://softwarelibre.uca.es/wikijuegos/Portada [Consulta: Juliol de 2008]

SDL. Simple DirectMedia Layer. 2008.
http://www.libsdl.org [Consulta: Juliol de 2008]

libSDL.es Simple DirectMedia Layer en Español. 2008.
http://www.libsdl.es [Consulta: Juliol de 2008]

LLOPIS, JOSÉ LUIS. *Genericidad en C++*. 2004.
http://www3.uji.es/~llopis/E28/Transparencias/list-template.pdf [Consulta: Setembre de 2008]

ABÁSOLO GUERRERO, MARÍA JOSÉ. *Introducción a la Inteligencia Artificial*. 2008.
http://dmi.uib.es/~abasolo/intart/2-juegos.html [Consulta: Setembre de 2008]