

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC & KỸ THUẬT MÁY TÍNH



LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC

**VIẾT CHƯƠNG TRÌNH TRÒ CHƠI
CỜ TƯỚNG**

HỘI ĐỒNG: Khoa Học Máy Tính

GVHD: ThS. Trần Giang Sơn

GVPB: ThS. Vương Bá Thịnh

—o0o—

SVTH 1: Đỗ Thành Long (1511799)

SVTH 2: Vũ Văn Huynh (1511328)

TP. HỒ CHÍ MINH, THÁNG 12/2019

LỜI CAM ĐOAN

Chúng tôi xin cam đoan Luận văn tốt nghiệp “**Viết chương trình trò chơi cờ tướng**” là công trình của riêng chúng tôi dưới sự hướng dẫn của ThS Trần Giang Sơn và đóng góp của ThS Vương Bá Thịnh. Nội dung nghiên cứu và các kết quả đều là trung thực và chưa từng được công bố trước đây. Các số liệu được sử dụng cho quá trình phân tích, nhận xét được chính chúng tôi thu thập từ nhiều nguồn khác nhau và sẽ được ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, chúng tôi cũng có sử dụng một số nhận xét, đánh giá và số liệu của các tác giả khác, cơ quan tổ chức khác. Tất cả đều có trích dẫn và chú thích nguồn gốc.

Nếu phát hiện có bất kì sự gian lận nào, chúng tôi xin hoàn toàn chịu trách nhiệm về nội dung Luận văn tốt nghiệp của mình. Trường Đại học Bách Khoa thành phố Hồ Chí Minh không liên quan đến những vi phạm tác quyền, bản quyền do chúng tôi gây ra trong quá trình thực hiện.

Nhóm hiện thực:

Đỗ Thành Long Vũ Văn Huynh

LỜI CẢM ƠN

Đầu tiên, chúng tôi xin gửi lời cảm ơn chân thành tới thầy hướng dẫn - ThS Trần Giang Sơn. Đây là người đã cho chúng tôi cơ hội tiếp cận với đề tài và sát cánh cùng chúng tôi trong suốt quá trình làm việc để từng bước hoàn thành đề tài đúng tiến độ. Thầy luôn sẵn lòng trong việc đưa ra góp ý trên cả phương diện kỹ thuật lẫn kỹ năng làm việc, sự hỗ trợ này góp phần rất lớn trong việc hoàn thành đề tài nói chung và hoàn thiện bản thân chúng tôi nói riêng. Ngoài ra, chúng tôi xin gửi lời cảm ơn tới thầy phản biện giai đoạn đề cương và luận văn - ThS Vương Bá Thịnh đã có những góp ý cho chúng tôi trong quá trình thực hiện đề tài này.

Chúng tôi xin cảm ơn các quý thầy cô giảng viên trong trường Đại học Bách Khoa thành phố Hồ Chí Minh đã truyền đạt cho chúng tôi những kiến thức quý báu trong suốt chặng đường học tập tại trường. Xin chúc quý thầy cô những điều tốt đẹp nhất.

Cuối cùng, chúng tôi xin gửi lời cảm ơn đến gia đình, người thân, bạn bè - những người đã quan tâm, động viên, giúp đỡ cả về vật chất lẫn tinh thần để chúng tôi có đủ nghị lực, sức khỏe để hoàn thành tốt đề tài này.

Nhóm hiện thực:

Đỗ Thành Long Vũ Văn Huynh

LỜI MỞ ĐẦU

Cờ tướng là một trong những loại cờ được chơi phổ biến ở Việt Nam trong quá khứ và gắn liền với kĩ ức của nhiều người. Với sự phát triển của máy tính khiến việc dùng máy tính để chơi cờ tướng trở nên phổ biến hơn. Trong đề tài luận văn “Viết chương trình trò chơi cờ tướng”, chúng tôi hướng đến mục tiêu nghiên cứu và áp dụng công nghệ để tạo ra phần mềm chơi cờ tướng giúp cho những người quan tâm về cờ tướng có thể học cách chơi cờ tướng, làm quen với cờ tướng cũng như nâng cao kỹ năng chơi cờ. Với những ý nghĩa thực tiễn, chúng tôi đã bắt tay vào nghiên cứu, hiện thực và đã đạt được những kết quả nhất định. Trong báo cáo này, chúng tôi xin trình bày những gì chúng tôi đã nghiên cứu và hiện thực trong thời gian qua.

Do đây cũng là lần đầu tiên tiếp cận lĩnh vực lập trình này cũng như kiến thức còn hạn hẹp nên khó tránh khỏi những thiếu sót trong quá trình thực hiện, chúng tôi rất mong nhận được góp ý và phản hồi từ thầy cô để hoàn thiện hơn trong tương lai. Ngoài ra, chúng tôi cũng đã cố gắng hết sức để hoàn thiện bài báo cáo một cách tốt nhất. Nhưng không thể nào tránh khỏi những lỗi phát sinh. Rất mong các thầy cô và đọc giả góp ý.

Nhóm hiện thực:

Đỗ Thành Long Vũ Văn Huynh

BỘ CỤC CỦA BÁO CÁO LUẬN VĂN

Dựa trên mục tiêu cụ thể đã trình bày trong phần trước, bài báo cáo được tổ chức thành sáu chương với các nội dung cụ thể như sau:

Chương 1: Giới thiệu. Trong chương này, bài báo cáo sẽ giới thiệu về đề tài, lý do chọn đề tài, mục tiêu của đề tài và khoanh vùng phạm vi nghiên cứu. Chương này cũng giới thiệu về lịch sử, luật chơi cờ tướng cũng như những ứng dụng của chương trình trò chơi cờ tướng.

Chương 2: Kiến thức nền tảng. Trong chương này sẽ đề cập những kiến thức nền tảng liên quan đến giải thuật minimax, Alpha-Beta, Negamax, Hàm lượng giá, Hashing, các giải thuật sort và search mảng, kỹ thuật lập trình trò chơi. Từ đó làm cơ sở cho việc cải tiến và hiện thực đề tài.

Chương 3: Khảo sát một số công trình liên quan. Trong chương này nhóm sẽ đưa ra những công trình nghiên cứu nổi bật về trò chơi cờ tướng mà đã được nghiên cứu trước đây. Từ đó nhóm sẽ khảo sát một số phương pháp hướng tiếp cận và đưa ra những phân tích, đánh giá làm cơ sở nghiên cứu đề tài.

Chương 4: Phân tích và phương pháp đề xuất. Trong chương này, nhóm sẽ phân tích độ phức tạp của cờ tướng và đưa ra các phương pháp cải tiến để đạt được độ hiệu quả trong tìm kiếm.

Chương 5: Trình bày kết quả hiện thực chương trình. Trong chương này, nhóm sẽ trình bày các kết quả đạt được như: Elo của AI, So sánh các thuật toán với nhau, Trình bày các tính năng của trò chơi,...

Chương 6: Tổng kết. Trong chương này, nhóm sẽ đưa ra kết luận, đánh giá ưu nhược điểm và hướng phát triển sắp tới.

Mục lục

1	Giới thiệu	1
1.1	Lý do và mục đích chọn đề tài	1
1.2	Phạm vi và mục tiêu của đề tài	2
1.3	Quá trình hiện thực luận văn	2
1.4	Giới thiệu về cờ tướng	3
1.4.1	Lịch sử của cờ tướng	3
1.4.2	Giới thiệu về bàn cờ và quân cờ	5
1.4.3	Luật chơi của cờ tướng	7
1.4.4	Ứng dụng của chương trình trò chơi cờ tướng	9
2	Kiến thức nền tảng	10
2.1	Cây tìm kiếm	10
2.1.1	Định nghĩa cây trong khoa học máy tính	10
2.1.2	Cây tìm kiếm	11
2.2	Các giải thuật tìm kiếm	12
2.2.1	Các kiểu tìm kiếm	12
2.2.2	Giải thuật tìm kiếm Negamax	12
2.2.3	Giải thuật tìm kiếm Minimax	13
2.2.4	Giải thuật tìm kiếm Alpha-Beta	14
2.2.5	Giải thuật tìm kiếm tìm kiếm nhị phân	15
2.2.6	Giải thuật binary insert	16
2.3	Hàm lượng giá	16
2.3.1	Hình dạng của hàm lượng giá	17
2.3.2	Dấu trong hàm lượng giá	17
2.4	Bảng băm	17
2.4.1	Lý thuyết của bảng băm	17
2.4.2	Giải quyết độ đụng	19
2.5	Lập trình trò chơi	20
2.5.1	Tốc độ khung hình	20
2.5.2	Vòng lặp trò chơi	22
2.5.3	Thời gian trong trò chơi	23
2.5.4	Đối tượng trong trò chơi	25

3 Khảo sát một số công trình liên quan	26
3.1 Phần mềm cờ tướng XQMS	26
3.2 Phần mềm CCBridge	26
3.3 Phần mềm cờ tướng Intella	27
4 Phân tích và phương pháp đề xuất	28
4.1 Phân tích độ phức tạp của cờ tướng	28
4.2 Hàm lượng giá cho cờ tướng	29
4.3 Cắt nhánh khi mất quân tướng	32
4.4 Cải tiến giải thuật Alpha Beta bằng move ordering	34
4.5 Hiện tượng horizon effect và giải thuật quiescence search	34
4.6 Cải tiến AlphaBeta bằng transposition table	35
4.7 Kiểm soát thời gian duyệt bằng giải thuật iterative depth first search	38
4.8 Giải thuật Zobrist Hashing	39
4.9 Cơ sở dữ liệu	40
4.10 Chuỗi FEN	41
5 Trình bày kết quả hiện thực chương trình	43
5.1 Đánh giá sức mạnh của AI	43
5.1.1 Đo chỉ số Elo của AI	43
5.1.2 So sánh độ hiệu quả giữa các thuật toán	44
5.2 Giao diện công cụ nhập liệu	46
5.3 Giao diện trò chơi	46
5.4 Các chức năng của trò chơi	47
5.4.1 Chức năng đi lại quân	47
5.4.2 Chức năng chọn chế độ chơi	47
5.4.3 Chức năng chơi ván mới	47
5.4.4 Chức năng thay đổi quân cờ theo hình ảnh có sẵn	48
5.4.5 Chức năng chọn thể loại quân cờ cho trước	48
5.4.6 Chức năng thay đổi background	49
5.4.7 Chức năng thay đổi nhạc nền trò chơi	49
5.4.8 Chức năng chọn người đi trước	50
5.4.9 Chức năng chọn mức độ chơi	50
5.4.10 Chức năng lưu lại bàn cờ	51
5.4.11 Chức năng thống kê	52
5.4.12 Chức năng sắp cờ thế	52
6 Tổng kết	54
6.1 Kết luận	54
6.2 Ưu điểm	54
6.3 Khuyết điểm	55
6.4 Hướng phát triển	55
Tài liệu tham khảo	56

A	Một vài số liệu thống kê	58
A.1	Thống kê về mã nguồn của hệ thống	58
A.2	Thống kê về bài báo cáo	58
B	Hướng dẫn sử dụng chương trình	59
B.1	Hướng dẫn tải chương trình về	59
B.2	Hướng dẫn chạy chương trình trò chơi cờ tướng	59
B.3	Hướng dẫn chạy công cụ hỗ trợ nhập liệu	60

Danh sách bảng

1.1	Thể loại và số lượng quân của mỗi bên	6
4.1	Bảng so sánh số nút phải xét giữa hai thuật toán Minimax và AlphaBeta .	29
4.2	Giá trị của quân cờ.	30
4.3	Giá trị vị trí của quân xe [1]	31
4.4	Giá trị vị trí của quân mã [1]	31
4.5	Giá trị vị trí của quân pháo [1]	31
4.6	Giá trị vị trí của quân tốt [1]	32
4.7	Giá trị của các quân cờ trong move ordering [1]	34
4.8	Kí hiệu các quân cờ trong FEN	41
5.1	Bảng mô tả cấu hình máy tính xách tay	43
A.1	Bảng thống kê về mã nguồn	58
A.2	Bảng thống kê về báo cáo	58

Tổng số: 12 bảng

Danh sách hình vẽ

1.1	Cờ tướng - nét văn hóa đặc đáo của người Việt.	1
1.2	Bàn cờ tướng.	5
1.3	Quân tướng.	6
1.4	Quân sĩ.	6
1.5	Quân tượng.	6
1.6	Quân xe.	7
1.7	Quân pháo.	7
1.8	Quân mã.	7
1.9	Quân tốt.	7
2.1	Hình ảnh phác họa cây tự nhiên	10
2.2	Cấu trúc dữ liệu cây trong khoa học máy tính	11
2.3	Cây tìm kiếm trong cờ tướng	11
2.4	mô phỏng các bước chạy của giải thuật Minimax	13
2.5	Ví dụ về hashfunction.	18
2.6	Ví dụ về separate chaining.	19
2.7	Hình ảnh tạo chuyển động bằng nhiều hình liên tiếp [2]	20
2.8	So sánh số lượng khung hình giữa 60 FPS và 24 FPS [3]	21
2.9	Hình ảnh vòng lặp trò chơi [4]	22
2.10	Hình ảnh trò chơi FIFA 4 [5]	24
2.11	Hình ảnh trò chơi Super Mario Bros [6]	25
3.1	Phầm mềm cờ tướng CCBridge	27
4.1	Hình ảnh AI xem quân tướng như một quân bình thường	32
4.2	Không mở rộng tiếp những nhánh bị mất tướng	33
4.3	AI chọn nước đi lâu thua nhất	34
4.4	hiện tượng horizon effect	37
4.5	Hình ảnh về một bàn cờ bị lặp trạng thái	38
4.6	Cấu trúc của cơ sở dữ liệu	41
4.7	Chuyển đổi bàn cờ thành chuỗi Fen	42
5.1	Thế trận pháo đầu	44
5.2	Giao diện công cụ nhập liệu	46
5.3	Giao diện trò chơi cờ tướng	46
5.4	Chức năng đi lại quân	47
5.5	Chức năng chọn chế độ chơi	47

5.6	Chức năng chơi ván mới	47
5.7	Chức năng thay đổi quân cờ theo hình ảnh có sẵn	48
5.8	Chức năng chọn thể loại quân cờ cho trước	49
5.9	Chức năng thay đổi background	49
5.10	Chức năng thay đổi nhạc nền trò chơi	50
5.11	Chức năng chọn người đi trước	50
5.12	Chức năng chọn mức độ chơi	51
5.13	Chức năng lưu lại bàn cờ	51
5.14	Chức năng lưu lại bàn cờ	52
5.15	Chức năng thống kê	52
5.16	Chức năng sắp cờ thế	53
5.17	Chức năng sắp cờ thế	53
B.1	59
B.2	Giao diện trò chơi	60
B.3	Giao diện công cụ nhập liệu	61
Tổng số:	48 hình vẽ	

Danh sách từ viết tắt

AI	Artificial Intelligence
CPU	Central Processing Unit
FPS	Frame Per Second
GUI	Graphical User Interface
PC	Personal Computer
RAM	Random Access Memory
GPU	Graphic Processing Unit
JSON	JavaScript Object Notation
SSD	Solid State Drive

Chương 1

Giới thiệu

Trong chương này, bài báo cáo sẽ giới thiệu về đề tài, lý do chọn đề tài, mục tiêu của đề tài và khoanh vùng phạm vi nghiên cứu. Chương này cũng giới thiệu về lịch sử, luật chơi cờ tướng cũng như những ứng dụng của chương trình trò chơi cờ tướng.

1.1 Lý do và mục đích chọn đề tài



Hình 1.1: Cờ tướng - nét văn hóa đặc đáo của người Việt [7]

Chơi cờ là nhu cầu giải trí đã xuất hiện từ rất lâu và vẫn được nhiều người yêu thích cho tới bây giờ. Ở Việt Nam, cờ tướng và cờ vua đang là hai loại cờ phổ biến và được đón nhận mọi người đón nhận. Chắc hẳn trong số chúng ta, không ít người đã từng gắn liền tuổi thơ với những hình ảnh được ngồi xem các trận đấu cờ tướng của các ông cụ hay những người trung niên trong ngôi làng của mình. Điều này trở thành những nét đặc sắc trong văn hóa người Việt Nam và vẫn còn được duy trì ở một số nơi đến tận bây giờ.

Ngày nay, khi đất nước có phát triển vượt bậc về nhiều mặt kéo theo nhu cầu giải trí cũng tăng lên. Điều đó dẫn tới sự phát triển và sinh ra nhiều loại hình giải trí khác nhau. Tuy vậy, những bộ môn thể thao trí tuệ nói chung cũng như cờ tướng nói riêng vẫn có một sức hút nhất định trong xã hội.

Xu hướng phát triển các thiết bị có tính di động hơn như: laptop càng ngày càng nhẹ, điện thoại thông minh, đồng hồ thông minh. Điều này đã giúp mang máy tính và internet đến gần với con người hơn bao giờ hết. Cũng vì thế mà việc chơi cờ bây giờ đã trở thành thuận tiện hơn. Bạn chỉ cần có một chiếc laptop hay điện thoại thông minh có cài phần mềm cờ tướng là có thể chơi cờ khi rảnh rỗi.

Với niềm yêu thích chơi cờ từ nhỏ đã khiến chúng tôi quan tâm đến tài này. Mong muốn của chúng tôi là có thể tạo ra một phần chơi cờ để một phần nào mọi người có thể làm quen với cờ tướng. Đồng thời, chúng tôi cũng muốn áp dụng những gì đã học trong trường để làm ra một dự án thực tế nhằm kiểm chứng và trau dồi những gì đã được học.

1.2 Phạm vi và mục tiêu của đề tài

Trong phạm vi Luận văn tốt nghiệp, nhóm sẽ tập trung nghiên cứu và phát triển phần mềm trò chơi cờ tướng trên một nền tảng PC có những tính năng nhằm hỗ trợ người chơi như: chế độ 2 người chơi, chế độ chơi với AI, tính năng nhắc nước đi, cảnh báo chiếu hết,...

Mục đích của đề tài là tìm hiểu và hiện thực các giải thuật liên quan, đánh giá mức độ hiệu quả của từng giải thuật, hoàn thiện chương trình chơi cờ tướng. Chương trình sau khi hoàn thiện có thể chơi được trong thực tế và giúp những người mới làm quen với bộ môn cờ tướng có thể dễ dàng tiếp cận.

1.3 Quá trình hiện thực luận văn

Luận văn tốt nghiệp được lên kế hoạch và thực hiện trong hai học kỳ chia làm hai giai đoạn Đề cương và sau Đề cương. Chúng tôi tóm tắt quá trình hiện thực luận văn thành bốn giai đoạn sau:

- Giai đoạn 1: Tìm hiểu về các giải thuật trong AI như: Minimax, Negamax, Alpha-Beta và các biện pháp cải tiến để tối ưu việc tìm kiếm trong không gian trạng thái như: move ordering, transposition table, hashing, null pruning, quiescence search,... Đồng thời, nhóm cũng hiện thực một số demo đơn giản để hiểu hơn về cách hoạt động của giải thuật.
- Giai đoạn 2: Tìm hiểu về lập trình trò chơi và framework làm trò chơi (Pygame). Trong giai đoạn này, nhóm vừa tìm hiểu về framework và hiện thực giao diện cho trò chơi cờ tướng.
- Giai đoạn 3: Viết engine cờ tướng và hoàn thiện những tính năng cho trò chơi. Trong giai đoạn này, nhóm tập trung viết engine để có thể chơi cờ tự động với người dùng. Đồng thời, nhóm cũng hoàn thiện trò chơi về giao diện cũng bổ sung tính năng mới.

- Giai đoạn 4: Viết tài liệu liên quan và sửa lỗi chương trình. Trong giai đoạn này, nhóm tập trung viết báo cáo, slide thuyết trình, các đặc tả về thiết kế phần mềm, thiết kế hệ thống. Đồng thời, nhóm cũng tiếp tục duy trì và bảo dưỡng chương trình để giúp chương trình đạt được sự ổn định về sau này.

1.4 Giới thiệu về cờ tướng

1.4.1 Lịch sử của cờ tướng

Hiện nay đã có rất nhiều người yêu thích và đam mê với cờ tướng nhưng khi nói về nguồn gốc xuất xứ của môn cờ tướng này, người ta vẫn không khỏi tranh luận và đang tiếp tục tìm tòi, nghiên cứu. Hiện nay có hai giả thuyết chính về nguồn gốc của cờ Tướng [8]:

1. Cờ tướng do cờ Lục bá phát triển mà thành. Cờ Lục bá du nhập vào Ấn độ, phát triển thành Saturanga. Saturanga sau đó lại du nhập ngược vào Trung Quốc, kết hợp với cờ tướng đang có để trở thành cờ tướng ngày nay.
2. Cờ tướng là do Saturanga, một phát minh của người Ấn độ, du nhập vào Trung Quốc phát triển mà thành, không liên quan gì đến cờ Lục bá.

Giả thuyết 1 được sự ủng hộ chủ yếu là từ các trang web của Trung Quốc. Theo giả thuyết này thì cờ tướng đã có ở Trung Quốc rất lâu trước khi người Ấn có Saturanga. Trong các tác phẩm từ thời Chiến quốc như “Thuyết Uyển” và “Chiêu hồn-Sở từ” đã có nhắc đến cờ tướng (mà người Trung Quốc vẫn gọi là Tượng kỳ). Giả thuyết 2 được các học giả phương Tây ủng hộ. Để làm rõ giả thuyết nào đáng tin hơn, hãy xem cách đi cờ Saturanga:

- Tốt: mỗi lần di chuyển chỉ tấn 1 ô về phía trước, đi thẳng nhưng ăn chéo như cờ vua. Chỉ luôn luôn tấn 1 ô, không bao giờ nhảy 2 ô, vì vậy không có việc ăn tốt qua đường, cũng không có chuyên phong cấp.
- Vua, Xe và Mã có cách di chuyển hoàn toàn giống như cờ vua (mã không bị cản)
- Sĩ: Mỗi lần có 4 vị trí để di chuyển, đi tới ô chéo góc liền kề (giống như cờ tướng)
- Tượng: Đi giống như sĩ nhưng dài gấp đôi (giống như cờ tướng nhưng không bị cản)

Rõ ràng cờ Saturanga có một số điểm giống cờ vua và một số điểm giống cờ tướng. Điều này cũng dễ hiểu vì chính bàn cờ Saturanga khi du nhập sang phương Tây thì trở thành cờ vua, còn du nhập vào Trung Quốc thì trở thành cờ tướng.

Sự cải tiến của Trung Quốc đối với cờ tướng:

Cải tiến đầu tiên và cũng quan trọng nhất là vị trí đặt quân cờ là đặt ở giao điểm các đường chéo không đặt trên ô, quân di chuyển trên đường chéo không nhảy từ ô này sang ô khác. Chỉ với động tác này, bàn cờ tăng thêm số điểm đặt quân từ 64 của Saturanga lên 81, số quân ở hàng cuối từ 8 tăng lên 9. Vua giờ đây đã có thể ở ngay giữa và rất dễ

dàng nhận thấy quân thêm vào bên phải vua chỉ có thể là 1 con sĩ, có vậy mới đảm bảo sự cân đối của bàn cờ. Sau đó là phải vẽ đường cho quân Sĩ, chữ X trước mặt Vua được thêm vào và thế là ta có cửu cung.

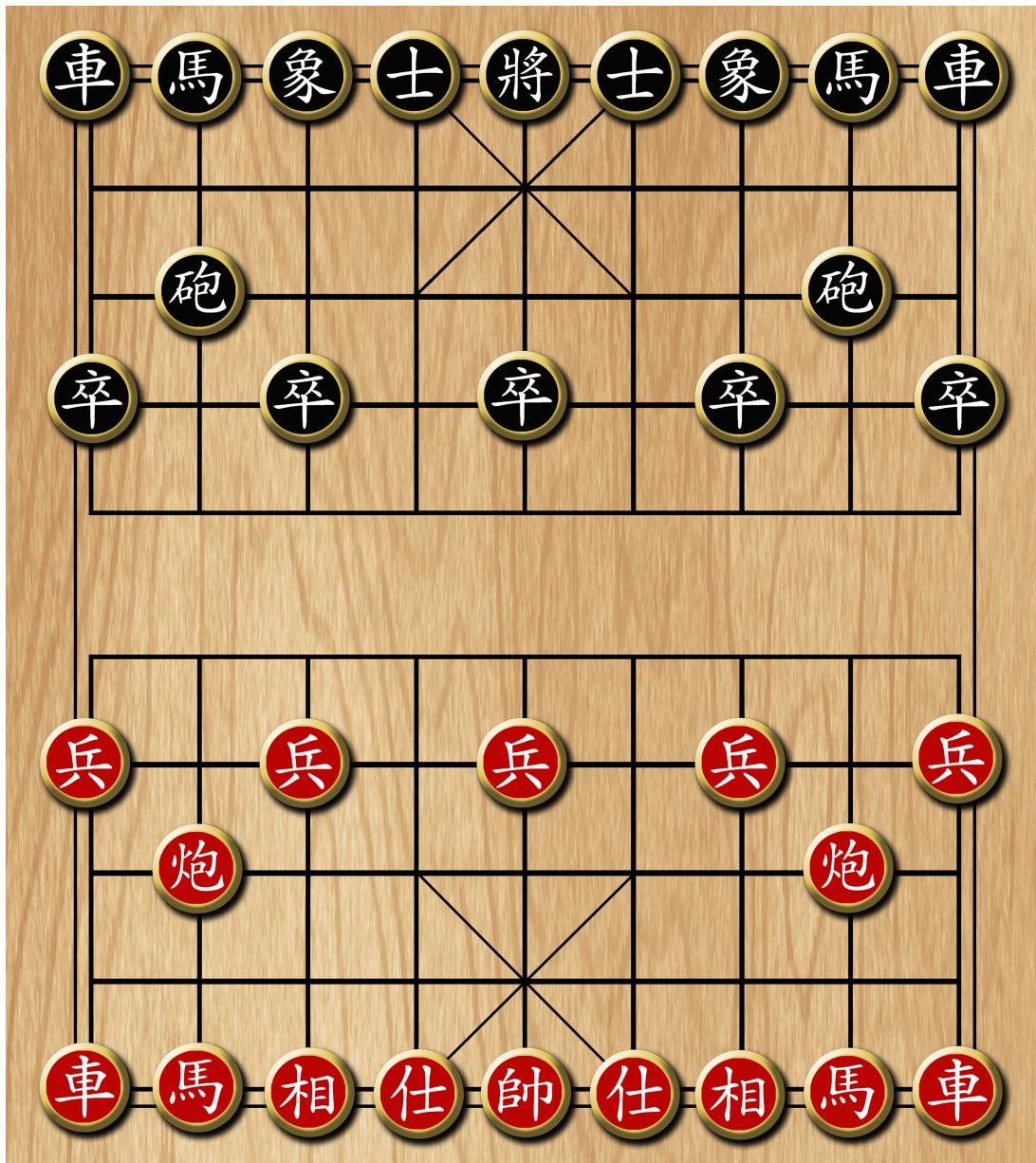
Cờ tướng cổ đại không có quân Pháo. Các nhà nghiên cứu đều thống nhất là quân Pháo được bổ sung từ cuối thời nhà Đường (618-907), là quân cờ ra đời muộn nhất trong bàn cờ tướng. Nhiều người cho rằng quân Pháo xuất hiện muộn là do ngày xưa không có pháo binh. Sự thật hoàn toàn không phải vậy. Pháo binh xuất hiện rất sớm trong chiến tranh thời xưa. Vẫn đề là muốn đưa Pháo vào bàn cờ thì bàn cờ đó phải đủ rộng. Bàn cờ 64 ô nếu muốn thêm quân Pháo thì cũng không biết phải đặt ở đâu khi quân 2 bên đông nghẹt như lô cốt thời nay! Bàn cờ tướng có chỗ đặt quân Pháo là nhờ số điểm đặt quân nhiều hơn (81 so 64). Xin nói thêm là bàn cờ Saturanga khi du nhập vào Thái Lan đã phát triển thành makruk, sang Nhật phát triển thành shogi. Hai loại cờ này đều không có quân Pháo chỉ vì đặt quân trên ô. Chỉ có bàn cờ janggi của Hàn Quốc là có quân Pháo vì loại cờ này xuất thân từ cờ tướng sau khi cải tiến của Trung quốc, cũng đặt quân trên đường.

Người Trung quốc khi ấy phải mất nhiều thời gian hoay tùng vị trí cho quân Pháo này và cuối cùng cũng tìm được vị trí lý tưởng cho quân Pháo như chúng ta thấy trên bàn cờ ngày nay. Tuy nhiên, để có vị trí này thì hàng chốt phải đẩy rất xa lên phía trước. Kết quả là không đầu thủ nào dám tấn chốt vì chỉ cần tiến lên 1 bước thì sẽ bị chốt đổi phương ăn mất ! Thế là Sở hà Hán giới ra đời, tạo thêm không gian ngăn cách 2 bên. Khi "hà" xuất hiện trên bàn cờ, 9 điểm đặt quân nữa được tăng thêm. Như vậy, bàn cờ tướng bây giờ đã có 90 điểm so với 64, đó là một sự mở rộng đáng kể. Diện tích chung của bàn cờ hầu như không tăng mấy (chỉ tăng thêm 8 ô) nhưng số điểm tăng thêm được một phần ba.

Quân cờ trong cờ tướng cũng được cách tân. Theo các tài liệu lịch sử, cờ tướng ở thời Đường được gọi là Tượng hý (du hý - trò chơi) có đặc điểm là quân cờ lập thể, bàn cờ có 64 ô vuông xen kẽ hai màu trắng đen, giống hệt bàn cờ vua hiện nay. Loại bàn cờ này đã được để lại trên các bức tranh dệt "Cầm, Kỳ, Thi, Hoạ" cũng như trên các vật dụng bằng sứ thời Đường. Thế nhưng sang thời Tống (960-1279) thì quân cờ trở nên dẹt và phẳng, trên có ghi chữ như quân cờ tướng ngày nay. Phải chăng đây là sự "cải lùi" ? Yếu tố kinh tế là một giả thuyết tuy dễ thuyết phục nhiều người nhưng chấp nhận giả thuyết này khác nào cho rằng người dân thời Tống nghèo hơn (hay tiết kiệm hơn) người dân thời Đường? Sự thật chăng qua là khi người ta chấp nhận thay đổi vị trí đặt quân thì cũng dễ dàng chấp nhận những thay đổi khác như hình dạng quân cờ. Còn một lý do khác có thể thấy khi hình dung bàn cờ vua ngày nay: Nếu quân cờ đặt trên ô thì quân cờ không thể che hết ô, nhưng nếu đặt trên đường thì quân cờ có thể che hết đường, nghĩa là sẽ khó quan sát hơn.

Không còn nghi ngờ nữa, chính Saturanga là tiền thân của cờ tướng và cả cờ vua ngày nay. Bàn cờ tướng ngày nay là sự phát triển từ Saturanga, không liên quan đến loại cờ nào khác. Khi người dân Trung quốc tiếp xúc với bàn cờ Saturanga, họ đã nhận ra sự ưu việt của loại cờ này so với cờ tướng hiện có lúc đó. Kết quả là hàng ngoại lần lượt hàng nội, chiếm hết "thị phần", thậm chí chiếm luôn cả "thương hiệu". Chuyện này không phải chỉ có ở bộ môn cờ. Bàn cờ tướng đã có ở Trung quốc từ trước khi Saturanga du nhập đã bị thất truyền, không còn ai chơi nữa. Tóm lại, giả thuyết 2 đáng tin cậy hơn.

1.4.2 Giới thiệu về bàn cờ và quân cờ



Hình 1.2: Bàn cờ tướng

Bàn cờ là hình chữ nhật do 9 đường dọc và 10 đường ngang cắt nhau vuông góc tại 90 điểm hợp thành. Một khoảng trống gọi là sông nằm ngang giữa bàn cờ, chia bàn cờ thành hai phần đối xứng bằng nhau. Mỗi bên có một cung Tướng hình vuông do 4 ô hợp thành tại các đường dọc 4, 5, 6 kể từ đường ngang cuối của mỗi bên, trong 4 ô này có vẽ hai đường chéo.

Theo quy ước bàn cờ được đặt đứng bên dưới là bên Đỏ (đi tiên), bên trên là bên Đen (đi hậu). Mỗi ván cờ lúc bắt đầu phải có 32 quân cờ chia đều cho mỗi bên gồm 16 quân Đỏ và 16 quân Đen, gồm bảy loại quân và khi ván đấu bắt đầu mỗi bên phải xếp quân như hình 1.2. Nếu là hai màu Đen và Trắng thì Đỏ coi là Trắng, quân nào chữ đen trên

nền trắng được gọi là quân Trắng, quân nào chữ trắng trên nền đen được gọi là quân Đen. Có thể cách viết khác nhau như Tướng và Soái, Bình và Tốt, hai loại Tượng, hai loại Sĩ để phân biệt nhưng thực chất là một. Tuy tên quân cờ của mỗi bên có thể viết khác nhau (ký hiệu theo chữ Hán) nhưng giá trị và cách đi quân của chúng lại giống nhau hoàn toàn. Bảy loại quân có ký hiệu và số lượng cho mỗi bên như sau:

Quân	Số Lượng
Tướng	1
Sĩ	2
Tượng	2
Xe	2
Pháo	2
Mã	2
Tốt	5

Bảng 1.1: Thể loại và số lượng quân của mỗi bên



Hình 1.3: Quân tướng



Hình 1.4: Quân sĩ



Hình 1.5: Quân tượng



Hình 1.6: Quân xe



Hình 1.7: Quân pháo



Hình 1.8: Quân mã



Hình 1.9: Quân tốt

1.4.3 Luật chơi của cờ tướng

Ván cờ được tiến hành giữa hai người, một người cầm quân Trắng (hay Đỏ), một người cầm quân Đen (hay Xanh). Mục đích của mỗi người là tìm mọi cách di quân trên bàn cờ theo đúng luật để chiếu bí (bắt Tướng hay Soái) của đối phương hoặc làm đối phương hết nước đi hợp lệ. Quân cờ được di chuyển theo luật sau: [9]

- Tướng: Đi ngang hay dọc từng bước một nếu vị trí đi đến là một ô điểm trống hoặc quân của đối phương (bắt quân) nhưng chỉ trong phạm vi cung Tướng. Hai Tướng

không được đối mặt nhau trên một đường thẳng mà không có quân nào đứng giữa (không lộ mặt Tướng).

- Sĩ: Đi chéo từng bước trong phạm vi cung Tướng nếu vị trí đi đến là một điểm trống hoặc quân của đối phương (bắt quân).
- Tượng: Mỗi nước đi chéo tại trận địa bên mình nếu vị trí đi đến là một điểm trống hoặc quân đối phương, không được qua sông. Nếu giữa đường chéo đó có một quân đứng thì quân Tượng bị cản không đi được.
- Xe: Đi dọc hoặc ngang không hạn chế số bước đi miễn là đừng bị quân khác cản đường từ điểm đi tới điểm đến. Nếu quân cản đường là quân đối phương thì nó có thể bắt quân.
- Pháo: Khi không bắt quân nó đi giống Xe, khi bắt quân đối phương thì trên đường đi giữa Pháo và quân bị bắt buộc phải có quân khác đứng làm "ngòi".
- Mã: Đi theo đường chéo hình chữ nhật của hai ô vuông liền nhau nếu vị trí đi đến là điểm trống hoặc quân đối phương (bắt quân). Nếu ở hai giao điểm bước thẳng dọc ngang có một quân khác đứng thì Mã bị cản, không đi được.
- Tốt: Mỗi nước đi một bước, khi qua sông thì tốt chỉ được tiến. Khi qua sông thì tốt được quyền tiến và đi ngang nếu vị trí đến là điểm trống hoặc quân đối phương (bắt quân).

Bắt quân: Đi tới vị trí quân đối phương và nhắc quân đối phương ra khỏi bàn cờ và thay thế bằng quân của bên đi. Việc bắt quân nhằm làm suy yếu đối phương để tiến công tới bắt Tướng và thắng cờ.

Chiếu tướng: Quân của một bên nhắm và Tướng của đối phương để bắt gọi là nước chiếu Tướng. Tướng bị chiếu phải chạy ra vị trí khác hoặc dùng quân để che chắn cho Tướng của mình hoặc bắt quân đang chiếu Tướng mình.

Thắng cờ khi:

- Chiếu bí được Tướng đối phương (chiếu cho Tướng đối phương hết đường chạy)
- làm cho Tướng đối phương bị vây chặt hết nước đi và các quân khác cũng không còn nước đi nữa.

Bị thua cờ khi:

- Tướng bị đối phương chiếu bí
- Tự nguyện xin thua dù ván cờ chưa xong
- Vi phạm luật bị xử thua
- Đi chậm quá thời gian quy định
- Dùng một quân chiếu đi chiếu lại Tướng đối phương liên tục trong 3 lần

Hòa cờ khi:

- Hai bên hết toàn bộ quân tấn công (hết sạch cả Xe, Pháo, Mã, Tốt) thì ván cờ được coi là hòa
- Một bên đề nghị hòa, bên kia đồng ý thì ván cờ được công nhận là hòa, dù bàn cờ còn bao nhiêu quân cũng mặc
- Trong 60 nước đi mà cả 2 bên không có một nước ăn quân nào thì ván cờ được xử hòa

1.4.4 Ứng dụng của chương trình trò chơi cờ tướng

Chương trình trò chơi cờ tướng sẽ có rất nhiều ứng dụng như:

- Đối với người mới tập chơi nó sẽ giúp nhận biết quân cờ và cách đi quân một cách dễ dàng.
- Phục vụ nhu cầu giải trí cho những người yêu thích cờ sau giờ làm việc căng thẳng.
- Phần mềm cờ tướng giúp rèn luyện nâng cao được kĩ nghệ của người chơi cờ.
- Phục vụ cho nhu cầu nghiên cứu và có thể áp dụng các giải thuật và công nghệ cho những trò chơi khác.

Chương 2

Kiến thức nền tảng

Trong chương này sẽ đề cập những kiến thức nền tảng liên quan đến giải thuật minimax, Alpha-Beta, Negamax, Hàm lượng giá, Hashing, các giải thuật sort và search mảng, kỹ thuật lập trình trò chơi. Từ đó làm cơ sở cho việc cải tiến và hiện thực đề tài.

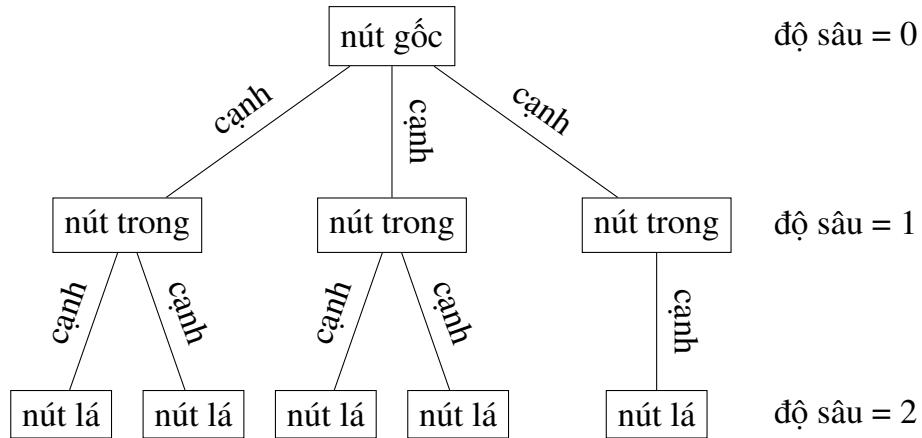
2.1 Cây tìm kiếm

2.1.1 Định nghĩa cây trong khoa học máy tính

Cây là một kiểu dữ liệu trừu tượng để lưu trữ các phần tử một cách có thứ bậc trên dưới [10]. Khác với cây tự nhiên (hình 3.1), gốc (root) của cây tìm kiếm sẽ nằm ở trên và lá (leaf) nằm ở dưới. Những nút không phải là lá hoặc gốc sẽ được gọi là nút trong (inner node). Độ sâu của nốt nút là tổng số cạnh trên đường đi từ nó đến gốc.



Hình 2.1: Hình ảnh phác họa cây tự nhiên

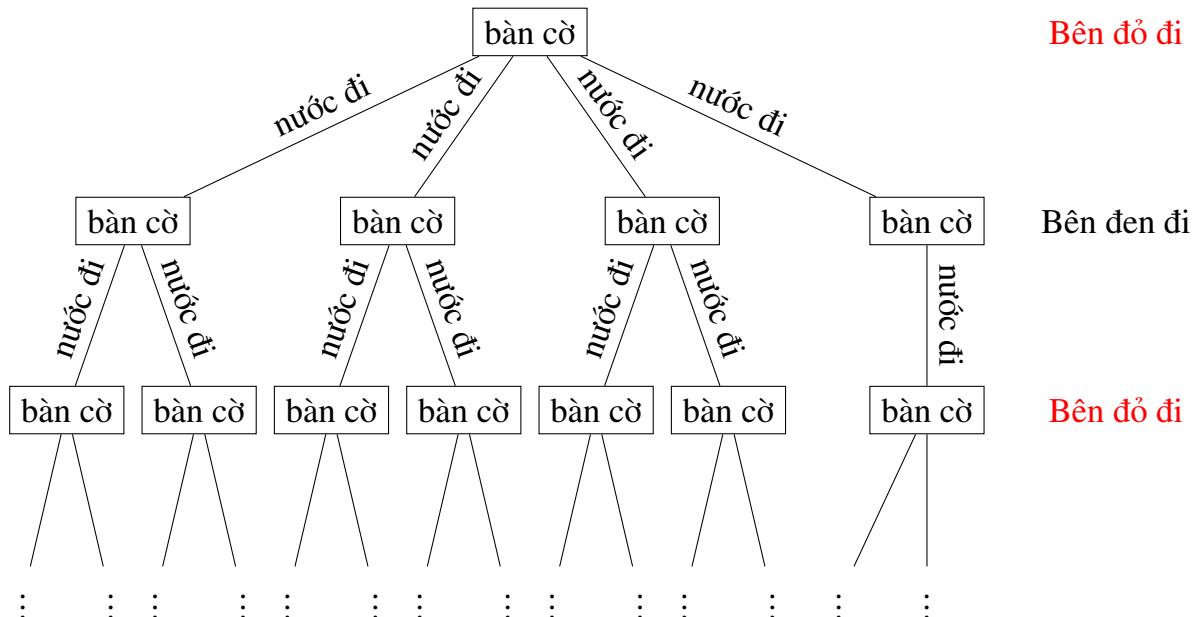


Hình 2.2: Cấu trúc dữ liệu cây trong khoa học máy tính

2.1.2 Cây tìm kiếm

Cây tìm kiếm (search tree) là tập con của không gian tìm kiếm (search space). Cây tìm kiếm là cấu trúc dữ liệu có thứ bậc, là một đồ thị có hướng với các nút còn được gọi là đỉnh (vertice) hay trạng thái (state), các nút được kết nối với nhau bằng cạnh có hướng (directed edge) hay bước chuyển trạng thái (state transition) [11].

Trong cờ Tướng, mỗi trạng thái bàn cờ là một đỉnh trong cây tìm kiếm. Mỗi cạnh tương ứng với nước đi. Chương trình sẽ nhận vào một trạng thái bàn cờ và gán cho nó làm gốc của cây tìm kiếm. Từ gốc cây tìm kiếm, chương trình sẽ sinh ra các nước đi hợp lệ. Từ trạng thái bàn cờ và nước đi hợp lệ sẽ sinh ra tương ứng một trạng thái bàn cờ tiếp theo. Tương tự như vậy ta sẽ xây dựng được một cây tìm kiếm cho một bàn cờ đầu vào.



Hình 2.3: Cây tìm kiếm trong cờ tướng

2.2 Các giải thuật tìm kiếm

Chương trình chơi cờ sẽ sinh ra cây tìm kiếm (phần 2.1.2) và tìm kiếm trong không gian trạng thái đó để dự đoán nước đi tốt nhất tiếp theo. Việc sinh cây tìm kiếm và giải thuật tìm kiếm được thực hiện bằng nhiều giải thuật khác nhau. Mỗi giải thuật có những ưu nhược điểm riêng. Các chương trình hiện nay thường kết hợp các giải thuật lại để bù đắp điểm yếu giữa các giải thuật và phát huy điểm mạnh của những giải thuật đó. Ở phần này, chúng tôi sẽ giới thiệu một số giải thuật tìm kiếm, các kỹ thuật phổ biến để nâng cao độ hiệu quả của các giải thuật tìm kiếm.

2.2.1 Các kiểu tìm kiếm

Ông Claude Shannon đã phân loại cách tìm kiếm ra làm 2 kiểu [12]:

- Kiểu A: là cách tìm kiếm vét cạn (brute-force search) tất cả các trường hợp có thể
- Kiểu B: là cách tìm kiếm chọn lựa (selective search), kiểu này chỉ tìm kiếm những trường hợp "tiềm năng" để giảm bớt số lượng nút tìm kiếm

Đặc điểm của chiến lược tìm kiếm kiểu B là lượng giá để chỉ tìm kiếm các nhánh "quan trọng". Điều này giúp giảm kích thước không gian tìm kiếm cũng như chi phí tìm kiếm. Tuy nhiên, chiến lược này có rủi do. Việc đánh giá nhánh nào xấu hay tốt là một việc khó. Dẫn đến nguy cơ bỏ qua các nhánh tốt do quá trình lượng giá sai hoặc thiếu sót. Chiến lược tìm kiếm kiểu B được sử dụng phổ biến vào thập niên 70. Sau này, do sức mạnh của máy tính càng ngày càng lớn. Các chương trình hiện nay thường có xu thế theo chiến lược tìm kiếm kiểu A hơn.

2.2.2 Giải thuật tìm kiếm Negamax

Algorithm 1 Giải thuật Negamax[13]

```

1: function negaMax(int depth)
2:   if depth == 0 then
3:     return evaluate()
4:   end if
5:   int max = -∞
6:   for all moves do
7:     score = -negaMax( depth - 1 );
8:     if score > max then
9:       max = score;
10:    end if
11:   end for
12:   return max;
13: end function

```

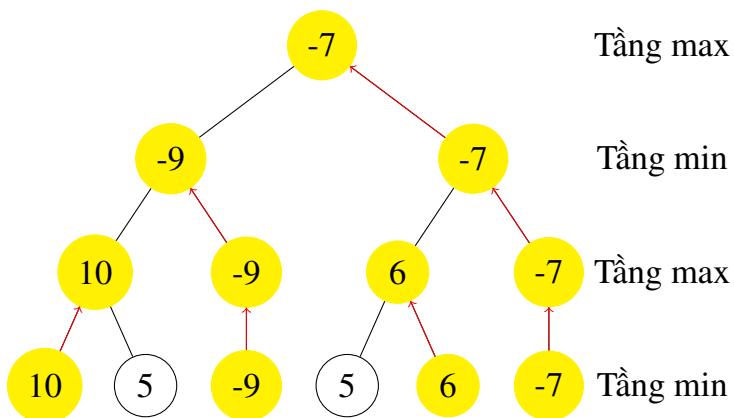
2.2.3 Giải thuật tìm kiếm Minimax

Algorithm 2 Giải thuật Minimax[14]

```

1: function maxi(int depth)
2:   if depth == 0 then
3:     return evaluate()
4:   end if
5:   int max = -∞
6:   for all moves do
7:     score = mini( depth - 1 );
8:     if score > max then
9:       max = score;
10:    end if
11:   end for
12:   return max;
13: end function
14: function mini(int depth)
15:   if depth == 0 then
16:     return evaluate()
17:   end if
18:   int min = +∞
19:   for all moves do
20:     score = maxi( depth - 1 );
21:     if score < min then
22:       min = score;
23:     end if
24:   end for
25:   return min;
26: end function

```



Hình 2.4: mô phỏng các bước chạy của giải thuật Minimax

2.2.4 Giải thuật tìm kiếm Alpha-Beta

Algorithm 3 Giải thuật Minimax sử dụng cắt nhánh Alpha-Beta[15]

```

1: function alphaBetaMax(int alpha, int beta, int depth)
2:   if depth == 0 then
3:     return evaluate()
4:   end if
5:   for all moves do
6:     score = alphaBetaMin(alpha, beta, depth - 1 );
7:     if score >= beta then
8:       return beta;
9:     end if
10:    if score > alpha then
11:      alpha = score
12:    end if
13:   end for
14:   return alpha;
15: end function
16:
17: function alphaBetaMin(int alpha, int beta, int depth)
18:   if depth == 0 then
19:     return evaluate()
20:   end if
21:   for all moves do
22:     score = alphaBetaMax(alpha, beta, depth - 1 );
23:     if score <= alpha then
24:       return alpha;
25:     end if
26:     if score < beta then
27:       beta = score
28:     end if
29:   end for
30:   return min;
31: end function

```

Algorithm 4 Giải thuật Negamax sử dụng cắt nhánh Alpha-Beta [15]

```

1: function alphaBeta(int alpha, int beta, int depth)
2:   if depth == 0 then
3:     return evaluate()
4:   end if
5:   for all moves do
6:     score = -alphaBeta(-beta, -alpha, depth - 1 );
7:     if score >= beta then
8:       return beta;
9:     end if
10:    if score > alpha then
11:      alpha = score;
12:    end if
13:   end for
14:   return max;
15: end function

```

2.2.5 Giải thuật tìm kiếm tìm kiếm nhị phân

Tìm kiếm nhị phân là giải thuật tìm kiếm nhanh với độ phức tạp là $O(\log_n)$ đối với dữ liệu đã được sắp xếp và được sử dụng rất nhiều trong thực tế. Thuật toán này sẽ so sánh phần tử cần tìm với phần tử ở giữa của mảng đã được sắp xếp. Nếu hai phần tử bằng nhau thì giá trị được trả về, nếu phần tử giữa mảng lớn hơn phần tử cần tìm thì quá trình tìm kiếm sẽ trong mảng con bên trái ngược lại sẽ tìm kiếm trong mảng con bên phải.

Algorithm 5 Giải thuật tìm kiếm nhị phân [16]

```

1: function BinarySearch(A[0..N-1], value)
2:   low = 0
3:   high = N - 1
4:   while low <= high do
5:     mid = (low + high) / 2
6:     if A[mid] > value then
7:       high = mid - 1
8:     else if A[mid] < value then
9:       low = mid + 1
10:    else
11:      return mid
12:    end if
13:   end while
14:   return not found
15: end function

```

Giả sử ta có một mảng $A = [2, 3, 4, 5, 6, 7, 8, 9, 20, 25, 45]$ và muốn tìm số 4 trong mảng này.

Vòng lặp 1: Phần tử Midle là 7 và lớn hơn phần tử cần tìm là 4, do đó ta sẽ tìm trên mảng con là $[2, 3, 4, 5, 6]$.

Vòng lặp 2: Phần tử Midle là 4 bằng với phần tử cần tìm nên trả về vị trí của Midle là 2.

2.2.6 Giải thuật binary insert

Khi ra muốn chèn một phần tử vào một mảng đã sắp xếp và vẫn giữ nguyên thứ tự sắp xếp của mảng đã cho ta áp dụng giải thuật Binary Insert. Giải thuật sẽ tương tự như binary search thay vì trả về vị trí của phần tử cần tìm thì nó trả về vị trí cần chèn của phần tử vào một mảng.

Algorithm 6 Giải thuật binary insert [17]

```

1: function BinaryInsert(A[0..N-1], value)
2:   low = 0
3:   high = N
4:   while low < high do
5:     mid = (low + high) / 2
6:     if A[mid] > value then
7:       high = mid
8:     else if A[mid] < value then
9:       low = mid + 1
10:    end if
11:   end while
12:   return low
13: end function
```

Giả sử ta có một mảng $A = [2, 3, 5, 6, 7, 8]$ và muốn chèn số 4 vào trong mảng này.

Vòng lặp 1: Phần tử Midle là 5 và lớn hơn phần tử cần chèn là 4, do đó ta sẽ tìm trên mảng con là $[2, 3, 5]$. Lúc này low = 0 và high = 2.

Vòng lặp 2: Phần tử Midle là 3 nhỏ hơn phần tử cần chèn là 4 nên low = 2 và high = 2 và ngừng vòng lặp trả về vị trí cần chèn là 2.

2.3 Hàm lượng giá

Hàm lượng giá sẽ nhận vào một trạng thái và trả ra một số. Số càng lớn thì nghĩa là bên được lượng giá càng có lợi và ngược lại. Nếu có thể sinh cây tới lúc biết thắng thua thì trạng thái sẽ có ba giá trị là +1(AI thắng), 0(hòa), -1(AI thua). Tuy nhiên, điều đó chỉ áp dụng được khi cây tìm kiếm có kích thước nhỏ như trò chơi caro. Còn với cờ tướng

và các trò chơi có kích thước cây tìm kiếm lớn thì chúng ta sẽ ước lượng khả năng thắng thua bằng những đặc điểm của trạng thái đó.

2.3.1 Hình dạng của hàm lượng giá

Hầu hết các hàm lượng giá hiện nay đều có dạng tuyến tính như sau:

$$eval = \sum_{i=1}^n F_i * W_i$$

Trong đó:

- F là đặc điểm (feature) của trạng thái
- W là trọng số của đặc điểm đó

Để tính ra giá trị lượng giá của một trạng thái, ta sẽ dùng điểm lượng giá của bên AI trừ cho điểm lượng giá của bên người chơi nếu đến lượt AI đi và ngược lại:

$$valueOfState = evalOfAI - evalOfOpponent \quad (2.1)$$

$$valueOfState = evalOfOpponent - evalOfAI \quad (2.2)$$

2.3.2 Dấu trong hàm lượng giá

Khi dùng giải thuật Minimax(hay các giải thuật tương tự), có những lúc ta lượng giá cho bên AI, lúc ta lượng giá cho bên người chơi như 2 công thức ở 2.1 và 2.2. Vì vậy chúng ta cần thêm dấu vào hàm lượng giá.

$$valueOfState = (evalOfAI - evalOfOpponent) * who2move$$

Trong đó $who2move = +1$ nếu đến lượt AI đi, $who2move = -1$ nếu đến lượt người chơi đi.

2.4 Bảng băm

2.4.1 Lý thuyết của bảng băm

Băm (hashing) là một kỹ thuật để định danh một đối tượng cụ thể trong một nhóm các đối tượng tương tự [18]. Ví dụ như:

- Trong một trường đại học mỗi sinh viên có một mã số sinh viên không giống nhau và qua mã số sinh viên đó có thể truy xuất thông tin như họ tên, ngày tháng năm sinh, giới tính,... của sinh viên đó.
- Trong thư viện mỗi cuốn sách có một mã số duy nhất và từ mã số đó có thể biết được thông tin về cuốn sách như tên và vị trí đặt cuốn sách.

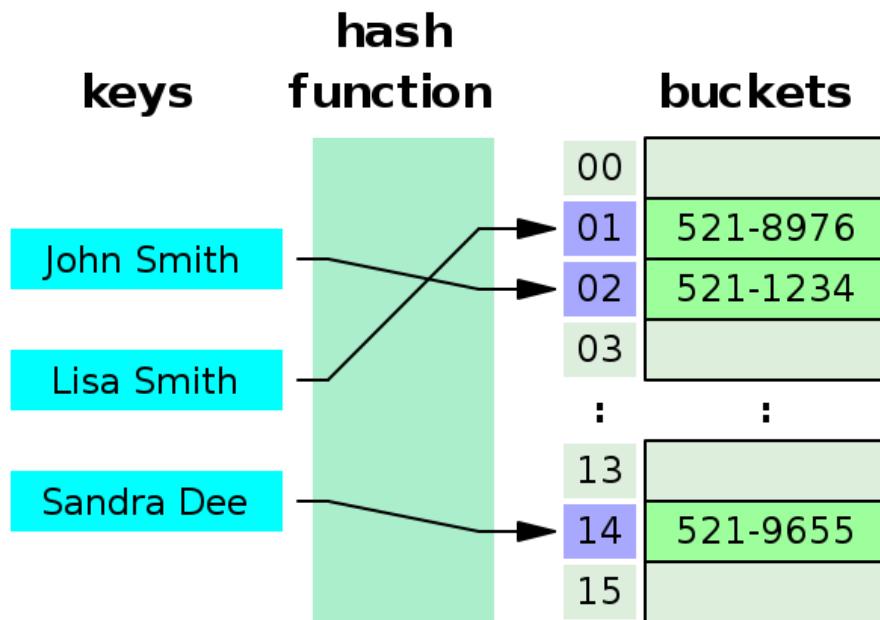
Giả sử rằng ta có một đối tượng và muốn gán cho nó một khóa `<key>` để tìm kiếm dễ dàng hơn. Để lưu giữ cặp `<key, value>`, ta có thể dùng mảng để làm việc này với chỉ số mảng là key và giá trị tại chỉ số đó tương ứng là value. Tuy nhiên với key quá lớn thì ta không thể sử dụng làm chỉ số, khi đó ta sẽ sử dụng băm(hashing).

Ý tưởng của hashing là đưa các cặp `<key, value>` về một mảng thống nhất (hash tables). Trong hashing các key có giá trị lớn sẽ được đưa về các key có giá trị nhỏ hơn gọi là khóa định danh bằng hàm băm (hash functions). Chúng ta có thể truy cập trực tiếp tới phần tử của mảng bằng khóa định danh với độ phức tạp $O(1)$.

Hashing được hiện thực qua hai bước:

- Một phần tử sẽ được chuyển đổi thành một số nguyên bằng cách sử dụng hàm băm. Phần tử này sẽ được sử dụng như một index để lưu trữ phần tử gốc trong bảng băm.
- Có thể truy xuất nhanh phần tử trong bảng băm thông qua khóa băm.

Giả sử ta cần lưu số điện thoại của ba người: John Smith (521-1234), Lisa Smith (521-8976), Sandra Dee (521-9655). Ta sử dụng một buckets gồm 16 ô trống và hàm hashfunction. Giá trị Hash của 3 người này lần lượt là: 1, 2 và 14. Sau khi tính được giá trị Hash của 3 người, ta lưu vào các bucket tương ứng là 1, 2 và 14. Điều này làm cho việc tìm kiếm trở nên rất hiệu quả.



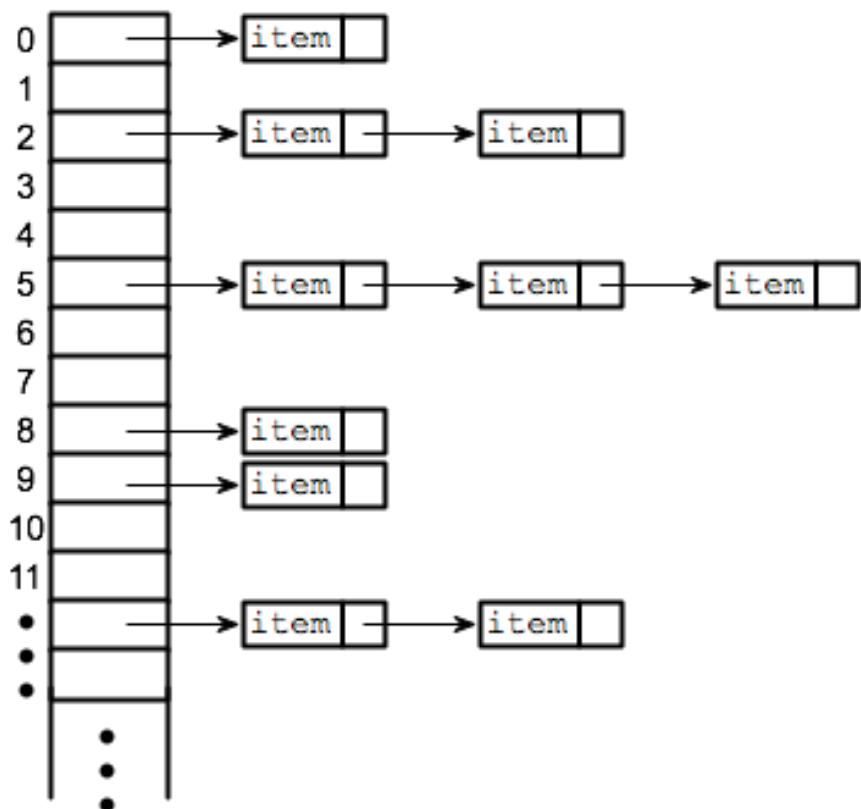
Hình 2.5: Ví dụ về hashfunction. [19]

Khi ta sử dụng hàm hashfunction sẽ có lúc xảy ra hiện tượng hai số điện thoại cùng hash ra cùng một giá trị. Ta gọi hiện tượng đó là đụng độ (collision). Do đụng độ làm quá trình tìm kiếm trở nên phức tạp hơn, ta sẽ cần một chiến lược để đối phó với nó mà ta gọi là chiến lược giải quyết đụng độ. Ta sẽ tìm hiểu mục này ở phần tiếp theo.

2.4.2 Giải quyết đụng độ

Separate chaining

Separate chaining là một kỹ thuật xử lý đụng độ phổ biến nhất. Nó thường được cài đặt với danh sách liên kết. Để lưu giữ một phần tử trong bảng băm, ta phải thêm nó vào một danh sách liên kết ứng với chỉ mục của nó. Nếu có sự đụng độ xảy ra, các phần tử đó sẽ nằm cùng trong một danh sách liên kết.



Hình 2.6: Ví dụ về separate chaining. [18]

Linear probing

Trong kỹ thuật này chúng ta sẽ không dùng linklist để lưu trữ. Khi thêm một phần tử vào mảng băm nếu vị trí đó đã có phần tử rồi thì vị trí lưu sẽ được tính lại theo cơ chế tuần tự.

```

index = index % hashTableSize
index = (index + 1) % hashTableSize
index = (index + 2) % hashTableSize
index = (index + 3) % hashTableSize

```

Và cứ thế theo cách như vậy chừng nào index thu được chưa có phần tử được sử dụng. Tất nhiên phải đảm bảo có đủ không gian cho phần tử mới.

Quadratic Probing

Ý tưởng của thuật toán này cũng khá giống Linear probing chỉ có khác ở công thức khi xảy ra đụng độ.

```
index = index % hashTableSize
index = (index + 12)% hashTableSize
index = (index + 22) % hashTableSize
index = (index + 32) % hashTableSize
```

Quadratic Probing

Ý tưởng của thuật toán này cũng khá giống Linear probing chỉ có khác ở công thức khi xảy ra đụng độ.

```
index = (index + 1 * indexH) % hashTableSize
```

```
index = (index + 2 * indexH) % hashTableSize
```

indexH là giá trị Hash được tính toán bởi hàm hash function khác.

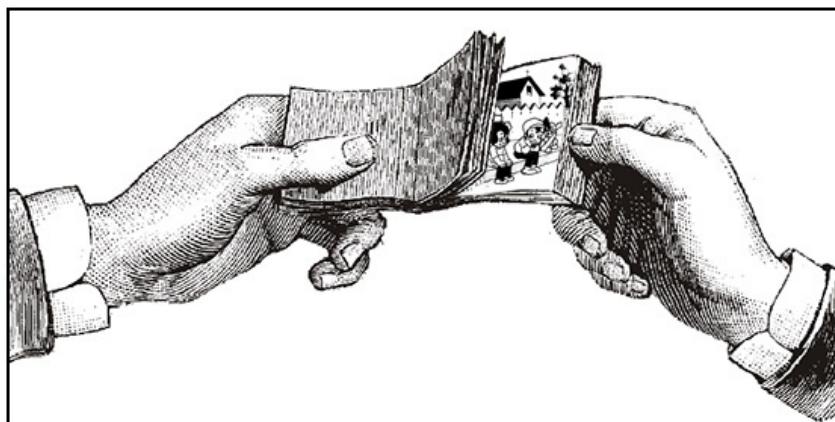
2.5 Lập trình trò chơi

Trong chương này sẽ giới thiệu về một số khái niệm quan trọng trong lập trình trò chơi như tốc độ khung hình, vòng lặp trò chơi, thời gian trong trò chơi và đối tượng trong trò chơi.

2.5.1 Tốc độ khung hình

Tốc độ khung hình (Frame rate) được hiểu là số hình ảnh được chiếu trong một thời gian nhất định. [20].

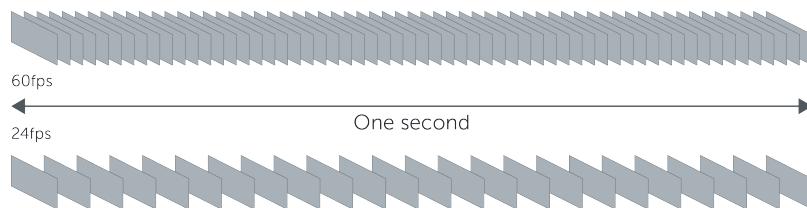
Tưởng tượng bạn vẽ hình chú chó ra tờ note, giờ hãy vẽ thật nhiều tờ note khác miêu tả sự di chuyển rất nhỏ sang bên trái, để khi bạn gộp lại và lật nhanh sẽ nhìn thấy chú chó đang chạy trước mắt mình, những tờ note riêng biệt đó được gọi là **khung hình** (frame) [3].



Hình 2.7: Hình ảnh tạo chuyển động bằng nhiều hình liên tiếp [2]

Đơn vị đếm các frame trong video là FPS (viết tắt của từ Frames – per – second). Ví dụ video của bạn hiển thị 24 khung hình mỗi giây, vậy video của bạn là 24 fps.

Khi FPS quá nhỏ thì mắt người sẽ không cảm nhận được chuyển động mà chỉ là những bức ảnh rời rạc. Khi FPS càng lớn thì chuyển động càng mượt mà (hình 2.8). Tuy nhiên, việc sử dụng FPS bao nhiêu còn tùy mục đích của người sử dụng.



Hình 2.8: So sánh số lượng khung hình giữa 60 FPS và 24 FPS [3]

Dưới đây là một số tốc độ khung hình được sử dụng cho một số trường hợp phổ biến [3]:

- 1-16 FPS:

- Người xem sẽ gần như không thể thấy hiệu ứng chuyển động.
- Hiếm khi sử dụng trong sản xuất phim và video hiện nay.
- Vẫn có thể sử dụng để tái hiện những bộ phim không tiếng ngày xưa.

- 24 FPS:

- Frame rate tiêu chuẩn phổ biến nhất hiện nay.
- Mang lại giao diện điện ảnh, hiệu ứng giống với mắt người nhìn nhất cho video.
- Sử dụng cho các máy chiếu ở rạp chiếu phim trên toàn thế giới.
- Tiêu chuẩn lý tưởng cho phim truyện, ngành công nghiệp điện ảnh và trên TV.

- 30 FPS:

- Giúp tăng chất lượng của các video cần sự chính xác trong điều kiện di chuyển nhanh và trực tiếp.
- Giúp ghi lại các chuyển động chạy hoặc nhảy trông thật và rõ nét hơn.
- Sử dụng phổ biến trên các kênh tin tức, quảng cáo, chương trình truyền hình, sự kiện thể thao hay bất cứ sự kiện nào phát sóng trực tiếp.
- Những tính năng live stream hay quay video trên điện thoại cho ứng dụng Instagram, Facebook...

- 60 FPS:

- Mang lại những cảnh quay chuyển động chân thực và chi tiết.

- Thông thường nhuêng video sẽ được chỉnh sửa sang tốc độ này sau khi quay để mang lại hiệu ứng di chuyển chậm (Slow – motion)
 - Để tạo ra chuyển động chậm mượt mà và chân thực hơn, cameraman sẽ quay video ở tốc độ 60 FPS, sau đó sẽ giảm thành 24 FPS hoặc 30 FPS ở khâu hậu kì.
 - Sử dụng quay video khi chơi game tốc độ cao như đua xe, chiến đấu
 - Những cảnh quay slow- motion.
- 120+ FPS:
 - Mang lại tốc độ khung hình cao cho các hiệu ứng đặc biệt, làm chậm hoặc siêu chậm các di chuyển nhanh, tăng cảm xúc, sự thích thú cho người xem.
 - Lí tưởng để quay lại các di chuyển nhanh trong sự kiện thể thao (vận động viên chạy, chơi thể thao, trượt ván, lướt sóng ...), sự vật (vụ nổ, pháo hoa, bão ...).

2.5.2 Vòng lặp trò chơi

Vòng lặp trò chơi (game loop) là luồng chạy chính của toàn bộ chương trình [21]. Nó có dạng vòng lặp vì trò chơi sẽ thực hiện một chuỗi hành động liên tiếp đến khi người dùng thoát chương trình. Mỗi vòng lặp được gọi là Khung hình (frame). Ví dụ một trò chơi được ghi là 60 FPS (frame per second) nghĩa là một giây trò chơi sẽ xuất ra 60 khung hình hay trò chơi được lặp 60 lần một giây. Thông thường các game thời gian thực sẽ có FPS từ 30 đến 60.

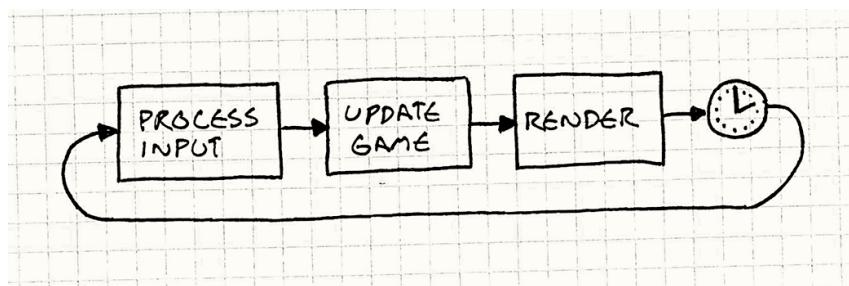
Dưới đây là một vòng lặp trò chơi đơn giản [21]:

Algorithm 7 Vòng lặp trò chơi đơn giản

```

1: while game is running do
2:   process inputs
3:   update game world
4:   generate outputs
5: end while

```



Hình 2.9: Hình ảnh vòng lặp trò chơi [4]

Trong mỗi vòng lặp, một chương trình trò chơi sẽ thực hiện 3 hành động cơ bản. Đầu tiên là nhận những input từ các thiết bị ngoại vi như: bàn phím, chuột, ... Tiếp theo, trò chơi sẽ dùng những thông tin này và các logic khác để cập nhật từng đối tượng trong trò chơi. Bước cuối cùng cũng là bước tốn nhiều chi phí nhất. Trò chơi sẽ xuất hình ảnh, âm thanh, ... ra màn hình, loa và các thiết bị ngoại vi khác.

Dưới đây là một vòng lặp đơn giản của trò chơi Pac-Man:

Algorithm 8 Vòng lặp trò chơi đơn giản của trò chơi Pac-Man[21]

```

1: while player.lives > 0 do
2:   //Process Inputs
3:   JoystickData j = grab raw data from joystick
4:
5:   //Update Game World
6:   update player.position based on j
7:   for each Ghost g in world do
8:     if player collides with g then
9:       kill either player or g
10:    else
11:      update AI for g based on player.position
12:    end if
13:   end for
14:
15:   //Generate Outputs
16:   draw graphics
17:   update audio
18: end while

```

Trong mỗi khung hình, trò chơi Pac-Man sẽ nhận dữ liệu từ tay cầm (dòng 3), sau đó cập nhật vị trí của nhân vật theo thông tin nhận từ tay cầm. Tiếp đó là xử lý logic như: nếu người chơi đụng phải ma thì người chơi sẽ bị tính thua, nếu không thì sẽ cập nhật vị trí của AI dựa vào vị trí của người chơi hiện tại (dòng 6 - 14). Cuối cùng, trò chơi sẽ xuất hình ảnh ra màn hình và âm thanh (dòng 16, 17).

2.5.3 Thời gian trong trò chơi

Thời gian thực và thời gian trò chơi

Thời gian trong trò chơi là một trong những khái niệm quan trọng. Thường thì thời gian trong trò chơi và thời gian thực có tỉ lệ 1:1. Tuy nhiên không phải lúc nào nó cũng vậy! Ví dụ như khi chúng ta tạm dừng chơi, thì thời gian thực vẫn tiếp tục chạy nhưng thời gian trong trò chơi thì ngừng lại mãi đến khi chúng ta tiếp tục hoặc có thể là không bao giờ (máy tính của bạn hư trong thời gian tạm dừng trò chơi chẳng hạn). Một ví dụ khác là trong những trò chơi thể thao (hình 2.10), người chơi không thể bỏ ra 90 phút để chơi một trận cầu được. Vì thế thời gian trong trò chơi sẽ nhanh hơn thời gian

thực tế. Hay trong những trò chơi giả tưởng, người chơi có thể quay ngược thời gian về một vị trí nào đó trong tương lai hoặc quá khứ.



Hình 2.10: Hình ảnh trò chơi FIFA 4 [5]

Delta time

Trong giai đoạn đầu của lập trình trò chơi, người lập trình viên thường lập trình dựa vào tốc độ của CPU. Tuy nhiên có vấn đề xảy ra, ta cùng xem đoạn code dưới đây:

$$\text{enemy.position.x}+ = 10$$

Giả sử nếu đoạn code này được chạy trên máy CPU với tốc độ 8MHz và chạy 30 FPS. Như vậy, mỗi giây nhân vật trong trò chơi sẽ di chuyển 300 pixel. Vậy chuyện gì sẽ xảy ra nếu chạy trên máy có CPU 16 MHZ (giả sử máy này chạy trò này nhanh gấp đôi máy trước)? Mỗi giây ta có nhân vật di chuyển 600 pixel. Đây là vấn đề. Delta time sinh ra để giải quyết vấn đề này. Đoạn code có delta time sẽ khác một chút:

$$\text{enemy.position.x}+ = 300 * \text{deltaTime}$$

Bây giờ, nếu ở 30 FPS thì nhân vật chúng ta sẽ di chuyển 10 pixel/frame. Nếu ở 60 FPS thì nhân vật chúng ta sẽ di chuyển 5 pixel/frame. Như vậy, cho dù ở FPS nào thì khoảng cách nhân vật cũng di chuyển được một đoạn đường giống nhau. Bây giờ chúng ta có vòng lặp trò chơi như sau:

Algorithm 9 Vòng lặp trò chơi đơn giản với delta time[21]

```

1: while game is running do
2:   realDeltaTime = time since last frame
3:   gameDeltaTime = realDeltaTime * gameTimeFactor
4:   Process inputs
5:   update game world with gameDeltaTime
6:   generate outputs
7: end while

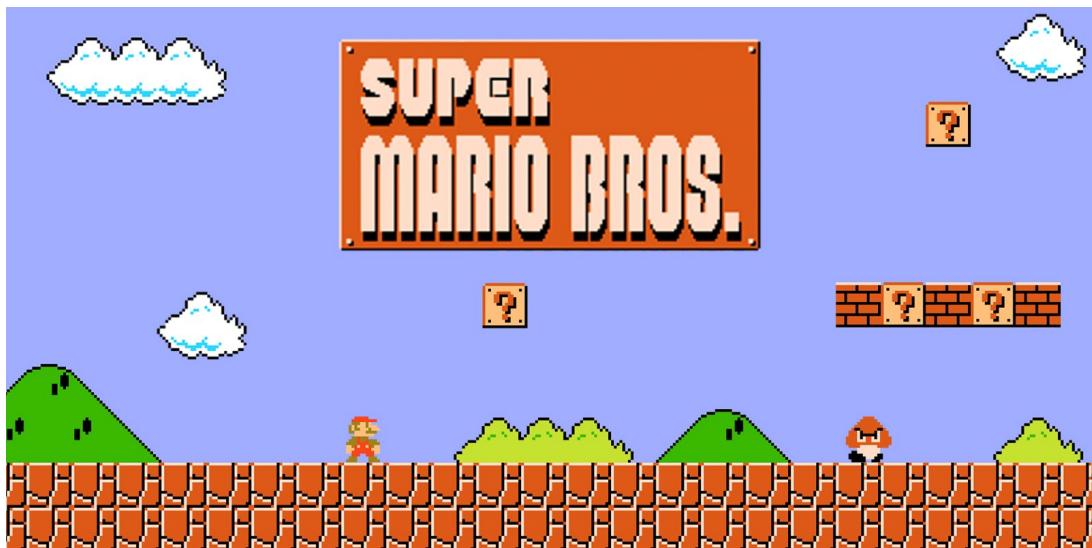
```

Thực ra khi áp dụng delta time vào vòng lặp trò chơi sẽ sinh ra nhiều vấn đề cần giải quyết. Tuy nhiên chúng ta sẽ không đề cập tại đây mà chỉ giới thiệu cơ bản về khái niệm delta time.

2.5.4 Đôi tượng trong trò chơi

Đôi tượng trong trò chơi là bất cứ thứ gì cần được vẽ, cập nhật, hoặc cả vẽ và cập nhật trong trò chơi[21]. Có 3 kiểu đôi tượng trong trò chơi:

- Kiểu đôi tượng vừa được cập nhật và vẽ: Ví dụ như trong trò Mario, mario và các cây nấm là những đôi tượng như vậy.
- Kiểu đôi tượng chỉ cần vẽ: Trong trò Mario, cột nước hay mây là những đôi tượng tĩnh, chỉ cần vẽ chứ không cần cập nhật logic.
- Kiểu đôi tượng chỉ cần cập nhật: Ví dụ là camera, ta không nhìn thấy được camera, vì chúng ta nhìn từ hướng camera. Camera không thể nhìn thấy, nhưng cần cập nhật vị trí để phù hợp với diễn biến của trò chơi.



Hình 2.11: Hình ảnh trò chơi Super Mario Bros [6]

Chương 3

Khảo sát một số công trình liên quan

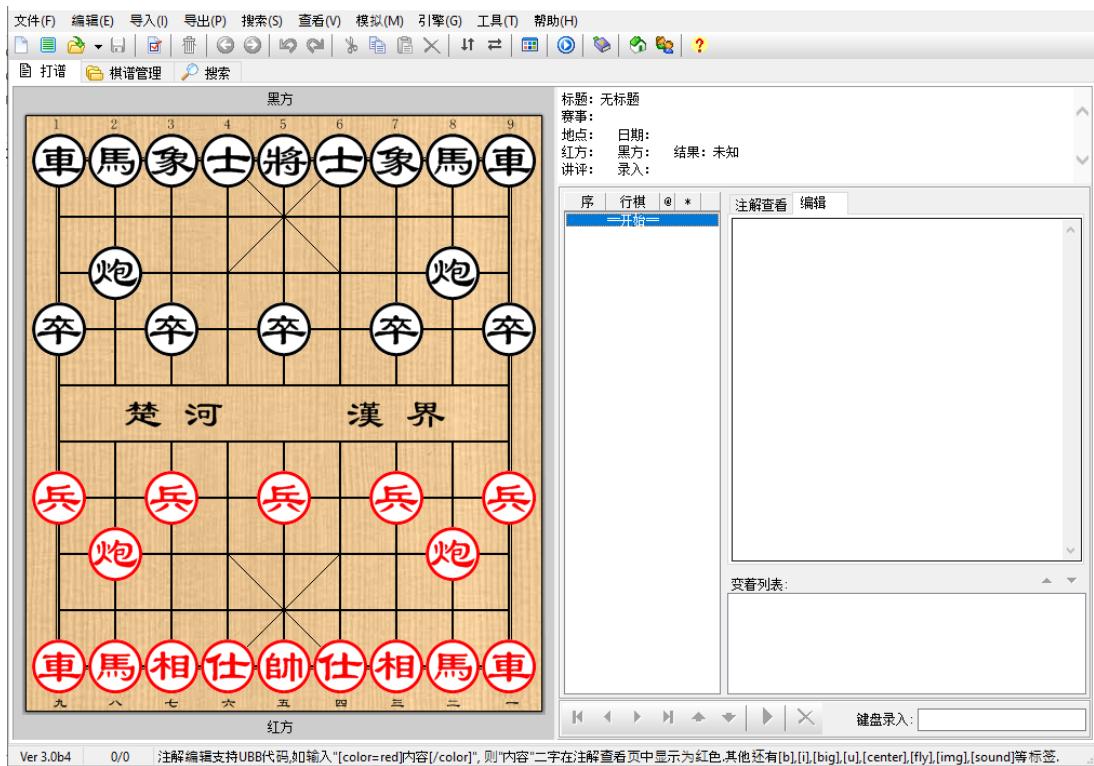
Trong chương chương này nhóm sẽ đưa ra những công trình nghiên cứu nổi bật về trò chơi cờ tướng mà đã được nghiên cứu trước đây. Từ đó nhóm sẽ khảo sát một số phương pháp hướng tiếp cận và đưa ra những phân tích, đánh giá làm cơ sở nghiên cứu đề tài.

3.1 Phần mềm cờ tướng XQMS

Phần mềm cờ tướng thế giới đầu tiên được ra đời năm 1989 có tên NKW. Người soạn thảo là thầy Hoàng Thiếu Long. Sau vài năm phát triển, đặc biệt với sự phát triển của phần cứng máy tính. Phần mềm cờ tướng được cải thiện để nâng cao sức mạnh của mình. Hiện nay, phần mềm cờ tướng xqms 3.23 bản Hoàn Mỹ 2.03 Mb được ra đời năm 2013. Tuy nhiên đến bây giờ vẫn được xem là bản free mạnh nhất. Với phần mềm này, người dùng không bắt buộc cài font tiếng Trung Quốc. Nó hiển thị nước đi 99% là tiếng Việt. Điều này rất thuận lợi khi giải cờ thê và giúp người chơi học tập dễ dàng hơn. Phần mềm cờ tướng sw này bao gồm Gui tiếng Việt (OTZ) 5.53 Mb. Book QQ Extremely 178 Mb, hỗ trợ cho người chơi cách khai cuộc lạ. [22]

3.2 Phần mềm CCBridge

Phần mềm CCBridge là phần mềm quen thuộc của các kỳ thủ chuyên nghiệp. Phần mềm này có khả năng lưu và phân tích các nước đi giúp người chơi có thể xem lại sau khi hoàn thành ván đấu. Ngoài hỗ trợ phiên bản tiếng Trung thì còn có hỗ trợ tiếng Việt. Với nhiều chương trình biên soạn khác nhau nên người dùng có thể sử dụng dễ dàng. Nhiều tính năng nổi bật giúp người chơi có thời gian nghiên cứu hơn. CCBridge được đánh giá là phiên bản mạnh nhất trên android hiện nay. Các bạn có thể tham gia chơi với máy hoặc tìm đối thủ trực tuyến so tài. [22]



Hình 3.1: Phần mềm cờ tướng CCBridge

3.3 Phần mềm cờ tướng Intella

Phần mềm cờ tướng có rất nhiều giao diện khác nhau phù hợp với các dòng máy. Với giao diện đẹp, lực cờ mạnh, giàu tính năng. SA Chess V1.6 32 bit và 64bit giao diện GUI sử dụng phần mềm mềm intella Việt Hóa bản 1.0. Mặc dù chỉ hỗ trợ 2U nhưng nó được nhiều người sử dụng và đánh giá cao. Phần mềm cờ tướng intella hỗ trợ cách chơi kỳ thủ – máy, kỳ thủ – kỳ thủ, chơi qua mạng LAN, Bluetooth, Wifi. Máy có nhiều cấp độ chơi từ nhập môn tới master. Ở Việt Nam, phần mềm intella được nhiều cổng game, nhà phát hành game ứng dụng để cung cấp đến người chơi. Ngoài ra, bạn có thể tải về máy mình để sử dụng một cách nhanh nhất. Ngoài ra phần mềm intella 1.007 được rất nhiều người yêu thích bởi tính đánh ác chiến của nó. Phần mềm này phù hợp với máy có cấu hình yếu. Không có virus và giao diện bằng tiếng Anh, sẽ dễ dàng cho nhiều người sử dụng hơn là tiếng Trung Quốc. [22]

Chương 4

Phân tích và phương pháp để xuất

Trong chương này, nhóm sẽ phân tích độ phức tạp của cờ tướng và đưa ra các phương pháp cải tiến để đạt được độ hiệu quả trong tìm kiếm.

4.1 Phân tích độ phức tạp của cờ tướng

Cờ tướng có bàn cờ 9×10 , lớn hơn bàn cờ vua 8×8 . Tuy nhiên, cờ tướng có nhiều giàng buộc như: tướng và sĩ không được đi khỏi cung, tượng không được đi qua sông, 2 tướng không được đối mặt nhau nên dẫn đến việc viết mã để sinh nước đi cũng khá phức tạp. Cũng vì có nhiều ràng buộc nên dù có bàn cờ lớn hơn nhưng độ phức tạp của cờ vua và cờ tướng là gần giống nhau.

kích thước cây trong cờ tướng

	40
2 tầng	1,600
	64,000
4 tầng	$2.56 \cdot 10^6$
	$1.02 \cdot 10^8$
6 tầng	$4.1 \cdot 10^9$
	$1.64 \cdot 10^{11}$
8 tầng	$6.55 \cdot 10^{12}$

Hệ số phân nhánh trung bình của cờ tướng là 40. Số lượng nút trong cây được tính

theo công thức b^n . Trong đó b là hệ số phân nhánh và n là số tầng. Nếu ta tiến hành dùng kỹ thuật cách nhánh AlphaBeta một cách tối ưu thì ta sẽ được tổng số nút của cây theo công thức $b^{\lceil \frac{n}{2} \rceil} + b^{\lfloor \frac{n}{2} \rfloor} - 1$ [15]. Tuy vậy, trên thực tế thì số nút phải duyệt của chương trình áp dụng AlphaBeta và các cải tiến là nằm trong $[b^{\lceil \frac{n}{2} \rceil} + b^{\lfloor \frac{n}{2} \rfloor} - 1, b^n]$.

Bảng 4.1: Bảng so sánh số nút phải xét giữa hai thuật toán Minimax và AlphaBeta

Số tầng	Minimax		AlphaBeta		tỉ lệ số nút $\frac{\text{Minimax}}{\text{AlphaBeta}}$
	Số nút	Số lần tăng	Số nút	Số lần tăng	
0	1	0	1	0	1
1	40	40	40	40	1
2	1,600	40	79	1.975	20.253
3	64,000	40	1,639	20.747	39.048
4	2,560,000	40	3,199	1.952	800.25
5	102,400,000	40	65,569	20.497	1,561.714
6	4,096,000,000	40	127,999	1.952	32,000
7	163,840,000,000	40	2,623,999	20.5	620,608.411
8	6,553,600,000,000	40	5,119,999	1.951	1,280,000.25

Nhìn vào bảng trên ta có nhận xét là thuật toán AlphaBeta hoàn toàn không chông lại được sự bùng nổ tổ hợp mà chỉ làm giảm tốc độ bùng nổ. Vì số lượng nút của cây rất bự nên dẫn đến khó khăn trong việc lập trình và sửa lỗi. Ngoài ra, việc cờ tướng có luật phức tạp cũng gây khó khăn trong việc sinh nước đi sao cho đúng và nhanh.

4.2 Hàm lượng giá cho cờ tướng

Mỗi bàn cờ được lượng giá theo 2 thành phần là: Giá trị của quân cờ và vị trí của nó trên bàn cờ. Hàm lượng giá được tính bằng công thức sau:

Lượng giá cho quân đen: $\sum_{i=1}^n ValuePiece(i) + Position(i)$

Lượng giá cho quân đỏ: $\sum_{j=1}^n ValuePiece(j) + Position(j)$

Giá trị lượng giá sẽ là:

$$\text{TotalValue} = \sum_{i=1}^n ValuePiece(i) + Position(i) - \sum_{j=1}^m ValuePiece(j) + Position(j) \quad (4.1)$$

Giá trị quân cờ

Mỗi quân cờ được gán cho một giá trị cụ thể tùy thuộc vào tầm quan trọng của nó trong bàn cờ.

Tướng: là quân cờ không có sức mạnh gì nhiều đôi khi nó sẽ trợ giúp các quân trong phe tấn công nhưng có giá trị cao nhất trên bàn cờ bởi vì nếu bị mất tướng thì sẽ bị thua cờ. Tất cả các quân khác đều có thể đổi, hi sinh nhưng tướng thì không thể đổi.

Quân	Tướng	Sĩ	Tượng	Mã	Xe	Pháo	Tốt
Giá trị	6000	120	120	270	600	285	30

Bảng 4.2: Giá trị của quân cờ [1]

Xe: là binh chủng cơ động nhất, nó không chế ngang dọc tối đa 17 điểm. Nên xe có giá trị cao nhất và nó cũng gần tương đương sức mạnh với pháo mã cộng lại.

Pháo: Là binh chủng tác chiến tầm xa rất hiệu quả. Trên đường dọc, Pháo không chế tối đa chỉ 8 điểm nhưng nó còn không chế được cả đường ngang. Mặt khác Pháo là quân có tính cơ động cao, nó có thể đi một bước vượt đến 9 tuyến đường. Nhưng ở cự ly gần, nhiều khi Pháo bất lực, không làm gì được để khống chế đối phương. Do đó so với Mã thì phải thấy mỗi quân có một sở trường riêng, sức mạnh của chúng coi như tương đương nhau. Thế nhưng trong giai đoạn trung cuộc quân số của hai bên tương đối còn nhiều, Mã đi lại dễ bị cản trở, còn Pháo thì nhanh nhẹn hơn, dễ phát huy năng lực hơn nên giá trị cơ bản của Pháo phải hơn Mã một chút.

Mã: Quân Mã trên bàn cờ có khả năng khống chế tối đa 8 điểm. Mỗi bước nhảy của nó vượt được hai tuyến đường ngang hay dọc. So ra thì Mã cũng mạnh đấy nhưng tác chiến ở tầm cự ly ngắn mà thôi. Nếu so với hai Tốt đã qua sông cặp kè nhau thì năng lực của Mã không hơn bao nhiêu.

Tốt: là quân kém năng lực nhất trên bàn cờ. Khi chưa qua sông nó chỉ kiểm soát hay khống chế mỗi một điểm trước mặt nó. Khi Tốt qua sông, nó khống chế đến 2 hay 3 điểm trước mặt và bên cạnh (Tốt biên chỉ khống chế được 2).

Sĩ, Tượng: là loại binh chủng phòng ngự, có nhiệm vụ chính là bảo vệ an toàn cho Tướng, đôi lúc chúng cũng trợ giúp các quân khác tấn công. Trong giai đoạn trung cuộc, Tốt đổi phương qua sông đổi được một Sĩ hoặc một Tượng.

Giá trị vị trí của quân cờ trên bàn cờ

Ngoài tổng giá trị của các quân cờ, vị trí quân cờ cũng rất quan trọng nơi có thể chiếm đóng và đe dọa Tướng đối phương. Vì mỗi quân có một cách di chuyển khác nhau nên ta sẽ trang bị cho chương trình một bảng lưu trữ ước tính các vị trí có thể có cho mỗi quân cờ. Các bảng vị trí của quân Xe, Pháo, Mã, Tốt của quân Đỏ được đưa ra trong các hình dưới đây. Đối với quân Đen thì ta chỉ cần đảo ngược bảng lại.

14	14	12	18	16	18	12	14	14
16	20	18	24	26	24	18	20	16
12	12	12	18	18	18	12	12	12
12	18	16	22	22	22	16	18	12
12	14	12	18	18	18	12	14	12
12	16	14	20	20	20	14	16	12
6	10	8	14	14	14	8	10	6
4	8	6	14	12	14	6	8	4
8	4	8	16	8	16	8	4	8
-2	10	6	14	12	14	6	10	-2

Bảng 4.3: Giá trị vị trí của quân xe [1]

4	8	16	12	14	12	16	8	4
4	10	28	16	8	16	28	10	4
12	14	16	20	18	20	16	14	12
8	24	18	24	20	24	18	24	8
6	16	14	18	16	18	14	16	6
4	12	16	14	12	14	16	12	4
2	6	8	6	10	6	8	6	2
4	2	8	8	4	8	8	2	4
0	2	4	4	-2	4	4	2	0
0	-4	0	0	0	0	0	-4	0

Bảng 4.4: Giá trị vị trí của quân mã [1]

6	4	0	-10	-12	-10	0	4	6
2	2	0	-4	-14	-4	0	2	2
2	2	0	-10	-8	-10	0	2	2
0	0	-2	4	10	4	-2	0	0
0	0	0	2	8	2	0	0	0
-2	0	4	2	6	2	4	0	-2
0	0	0	2	4	2	0	0	0
4	0	8	6	10	6	8	0	4
0	2	4	6	6	6	4	2	0
0	0	2	6	6	6	2	0	0

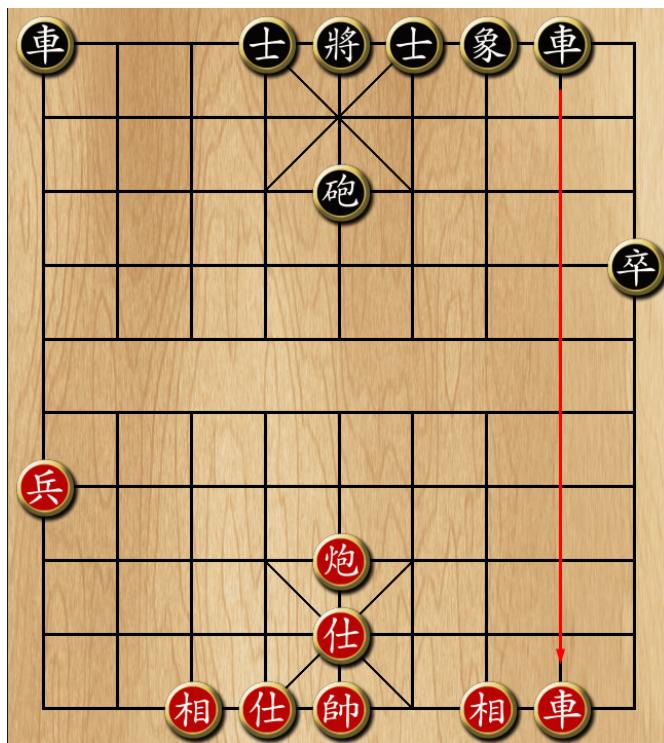
Bảng 4.5: Giá trị vị trí của quân pháo [1]

0	3	6	9	12	9	6	3	0
18	36	56	80	120	80	56	36	18
14	26	42	60	80	60	42	26	14
10	20	30	34	40	34	30	20	10
6	12	18	18	20	18	18	12	6
2	0	8	0	8	0	8	0	2
0	0	-2	0	4	0	-2	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

Bảng 4.6: Giá trị vị trí của quân tốt [1]

4.3 Cắt nhánh khi mất quân tướng

Quân tướng là quân quan trọng trong bàn cờ. Bên nào mất tướng thì coi như thua. Một trong những lỗi cơ bản của mới lập trình cờ đó là để cho AI xem quân tướng như các quân bình thường khác. Chúng ta hãy xem thế cờ dưới đây:



Hình 4.1: Hình ảnh AI xem quân tướng như một quân bình thường

Giả định AI cầm quân đen và đến lượt AI đi, AI sẽ dùng quân xe đen ăn xe đỏ. Tại sao lại như vậy? Vì AI nghĩ rằng: sau khi lấy xe đen ăn xe đỏ, bên đỏ lấy pháo ăn tướng đen thì quân đen lấy pháo đen ăn lại tướng đỏ. Tính ra là bên AI lời được quân xe đỏ.

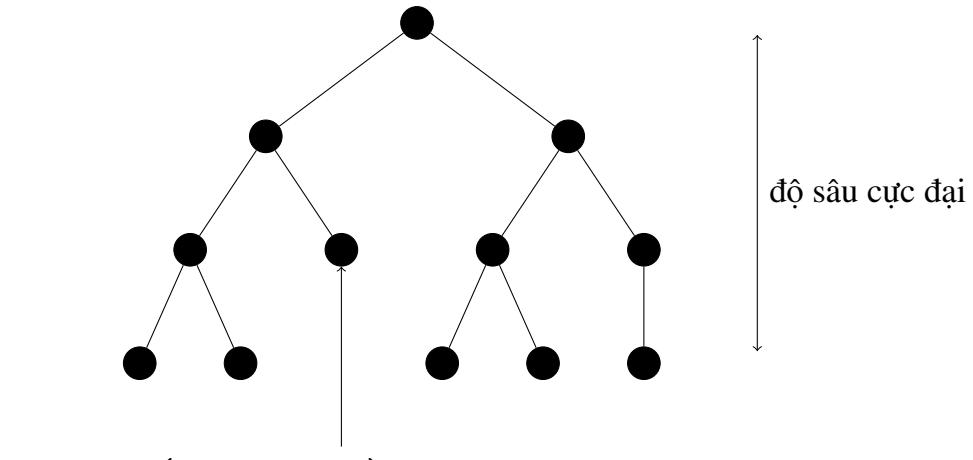
Vì vậy, chúng ta phải có giải pháp để AI xem quân tướng là quân đặc biệt và không

bao giờ được để mất. Để xử lí những trường hợp này, ta cần chuyển Makemove thành một hàm boolean, trả lại giá trị true nếu nước đi thử này lại ăn được tướng đối phương. Nếu ăn được, hiển nhiên điểm sẽ rất cao (được 1000 điểm) và không cần phải tính điểm cho những nhánh con nữa. Phần chương trình đi thử có dạng như sau:

Algorithm 10 Đoạn mã xử lý mất tướng

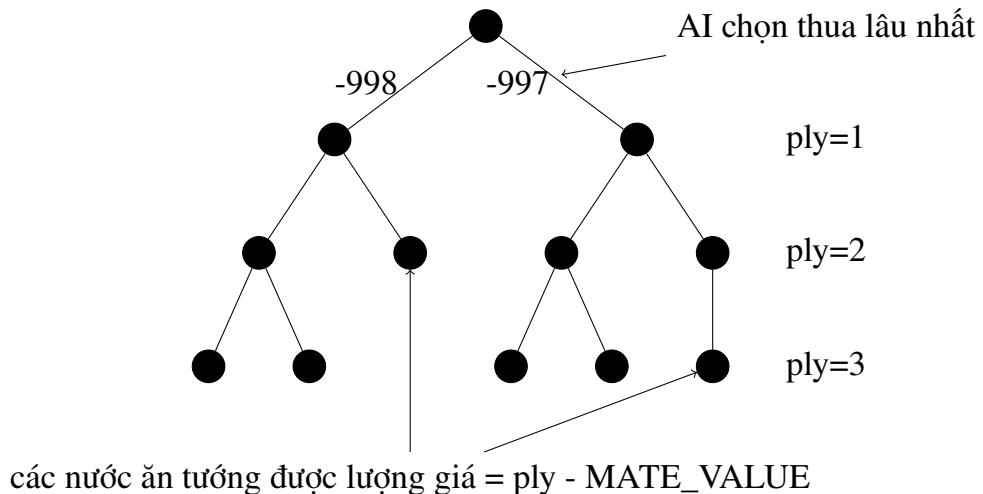
```

1: ...
2: value = ply - MATE_VALUE
3: Array moves = generateAllPossibleMoves();
4: for int i = 0; i < moves.length(); i++ do
5:   if makeMove(moves[i]) == False then
6:     continue
7:   end if
8:   value = -alphaBeta(ply - 1, -beta, -alpha);
9:   unmakeMove(moves[i]);
10: end for
11: ...
  
```



Hình 4.2: Không mở rộng tiếp những nhánh bị mất tướng

Trong đoạn mã trên có đoạn khó hiểu là: value = ply - MATE_VALUE. ply là biến chỉ độ sâu. Câu lệnh đó giúp cho AI có tính chất: nếu có nhiều đường thắng thì sẽ chọn đường thắng nhanh nhất, nếu có nhiều đường thua thì sẽ chọn đường thua lâu nhất. AI khi chọn đường thua lâu nhất để hi vọng đối thủ đi "nhầm" và tránh được bàn thua. Nó cũng giúp AI giống với người chơi hơn. Người chơi khi biết thua vẫn cố gắng để sao cho lâu thua nhất để mong người kia phạm sai lầm và nếu thắng thì cố gắng "dứt điểm" đối thủ nhanh nhất để tránh "đêm dài lắm mộng". Hình minh họa bên dưới.



Hình 4.3: AI chọn nước đi lâu thua nhất

4.4 Cải tiến giải thuật Alpha Beta bằng move ordering

Độ hiệu quả của việc cắt nhánh trong giải thuật Alpha Beta dựa vào cách chúng ta duyệt các nước đi kế tiếp. Giả dụ các nước đi được sắp xếp theo thứ tự từ tốt nhất đến xấu nhất. Nếu chúng ta duyệt nước xấu hơn trước thì giải thuật alpha-beta dường như sẽ trở thành giải thuật minimax bình thường. Từ đó ta thấy được tầm quan trọng của việc sắp xếp các nước (move ordering) trước khi duyệt là quan trọng như thế nào.

Tuy vậy, việc đánh giá nước nào tốt nước nào xấu cũng là vấn đề khó như việc lượng giá một bàn cờ vậy. Ở đây chúng tôi, sẽ đánh giá một nước tốt là nước bắt được quân to hơn. Giá trị các quân sẽ có giá trị như sau:

	Tướng	Sĩ	Tượng	Mã	Xe	Pháo	Tốt
Giá trị	5	2	2	3	4	3	1

Bảng 4.7: Giá trị của các quân cờ trong move ordering [1]

Vòng lặp của giải thuật Alpha Beta sẽ được sửa đổi thành mã giải ở mã dưới đây.

4.5 Hiện tượng horizon effect và giải thuật quiescence search

Horizon effect là hiện tượng gặp phải khi chúng ta tìm kiếm bằng giải thuật Alpha-Beta với số tầng cố định. Chúng ta cùng bàn luận về hình 4.4.

Giả sử bên đi trước là đỏ. Nếu AI duyệt một tầng thì AI sẽ dùng xe đỏ để ăn pháo đen. Tuy nhiên nếu 2 tầng thì AI sẽ thấy xe đen ăn lại pháo đỏ. Nếu duyệt 3 tầng thì AI thấy mã đỏ có thể ăn xe đen lại.

Algorithm 11 Giải thuật Alpha Beta có move ordering [1]

```

1: ...
2: Array moves = generateAllPossibleMoves();
3: for int i = 0; i < moves.length(); i++ do
4:   Find best move among moves[i] to moves[end];
5:   Swap moves[i] with moves[bestMove];
6:   makeMove(moves[i]);
7:   value = -alphaBeta(ply - 1, -beta, -alpha);
8:   unmakeMove(moves[i]);
9: end for
10: ...

```

Quiescence search sẽ giải quyết được vấn đề này (horizon effect) bằng cách gia tăng số tầng Alpha-Beta đến khi nào bàn cờ trở nên "ổn định". Giải thuật quiescence search sẽ được gọi ở nút là nếu có chiếu hoặc bắt quân ở nút lá. Quiescence search sẽ sinh ra tất cả nước bắt quân và chiếu đến khi nào không còn nước bắt quân và chiếu nào nữa.

Quiescence search sẽ tăng thời gian tìm kiếm. Nhưng trong đa số các trường hợp thì thời gian tìm kiếm sẽ gia tăng tầm 20%-30%[1]. Tuy nhiên, đổi lại thì chương trình sẽ thông minh hơn và ít khi bị bắt quân một cách dễ dàng.

4.6 Cải tiến AlphaBeta bằng transposition table

Chương trình cờ tướng tìm nước đi bằng cách xây dựng một cái cây lớn. Tuy nhiên, thực ra đó là một đồ thị có chu trình. Vì một bàn cờ, sau một số hoán vị nước đi sẽ dẫn tới cùng một bàn cờ. Ta nhìn vào hình 4.5. Ví dụ bên đỏ đi xe tiến 2, sau đó đen tiến một, với cách đi {xe đỏ tiến 2, tốt đen tiến 1, xe đỏ bình 1} sinh ra một bàn cờ giống với cách đi {xe đỏ bình 1, tốt đen tiến 1, xe đỏ tiến 2}. Từ đó, ta có nhận xét là trong cây được sinh ra, các nút của cây có thể trùng nhau. Việc hoán vị một chuỗi nước đi đều dẫn tới cùng một trạng thái người ta gọi trong tiếng anh là transposition. Vì thế bảng này có tên là transposition table.

Ý tưởng của bảng transposition table là khi duyệt qua một nút của cây thì sẽ lưu lại thông tin của nút đó, đến khi duyệt lại một nút tương tự thì lấy thông tin ra dùng mà không phải mất thời gian tính lại. Từ đó tiết kiệm được thời gian tính toán.

Transposition table thông thường được hiện thực bằng một mảng một chiều. Mỗi phần tử của mảng chứa thông tin như sau [23]:

- Khoá: Đây là một khoá độc nhất được sinh ra bằng việc mã hoá bàn cờ ra bằng giải thuật Zobrist hoặc BCH.
- Nước đi: mỗi bàn cờ quá trình duyệt AlphaBeta sẽ có một nước đi tốt nhất (tốt nhất ở đây là tại ngay thời điểm duyệt đó và được đánh giá bằng chương trình đó). Nước đi tốt nhất đó sẽ được lưu lại.

Algorithm 12 Giải thuật quiescence search [1]

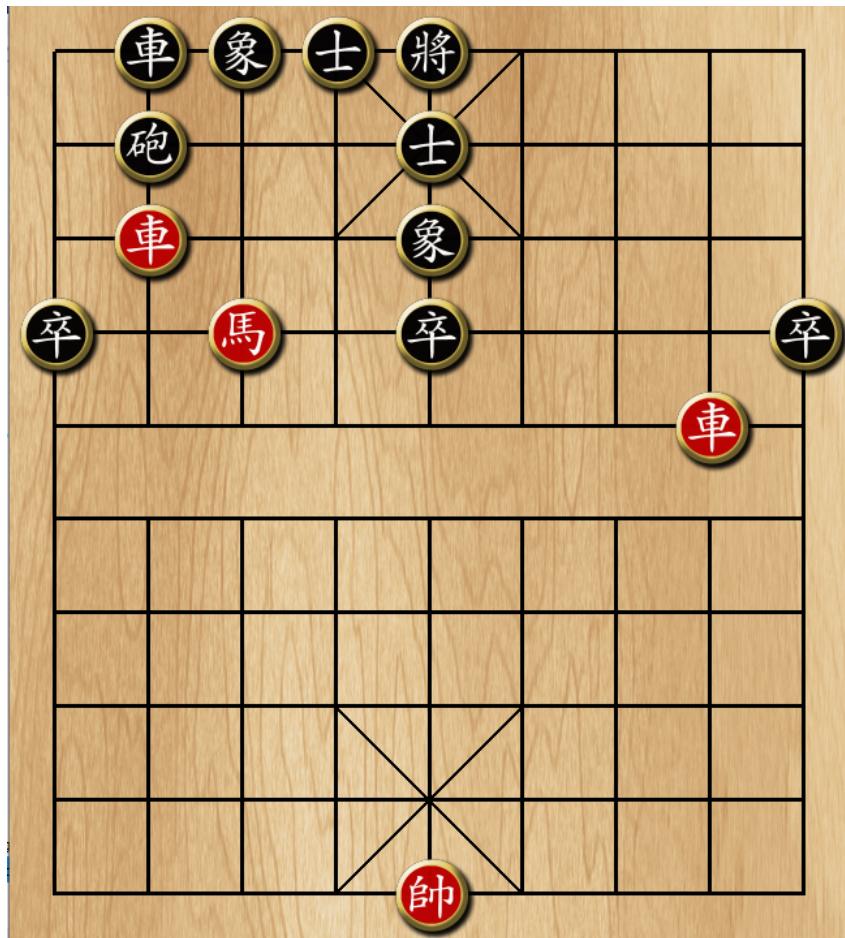
```

1: function alphaBeta(int ply, int alpha, int beta)
2:   ...
3:   for int i = 0; i < moves.length(); i++ do
4:     makeMove(moves[i]);
5:     if ply == 1 && capture is made then
6:       value = -quiescenceSearch(-beta, -alpha);
7:     else
8:       value = -alphaBeta(ply - 1, -beta, -alpha);
9:     unmakeMove(moves[i]);
10:    end if
11:   end for
12:   ...
13: end function
14: function quiescenceSearch(int alpha, int beta)
15:   Array capture = generateAllCaptures();
16:   for int i = 0; i < capture.length(); i++ do
17:     makeMove(capture[i]);
18:     value = -quiescenceSearch(-beta, -alpha);
19:     unmakeMove(capture[i]);
20:     if value >= beta then
21:       return beta;
22:     end if
23:     if value > alpha then
24:       alpha = value;
25:     end if
26:   end for
27:   return alpha;
28: end function

```

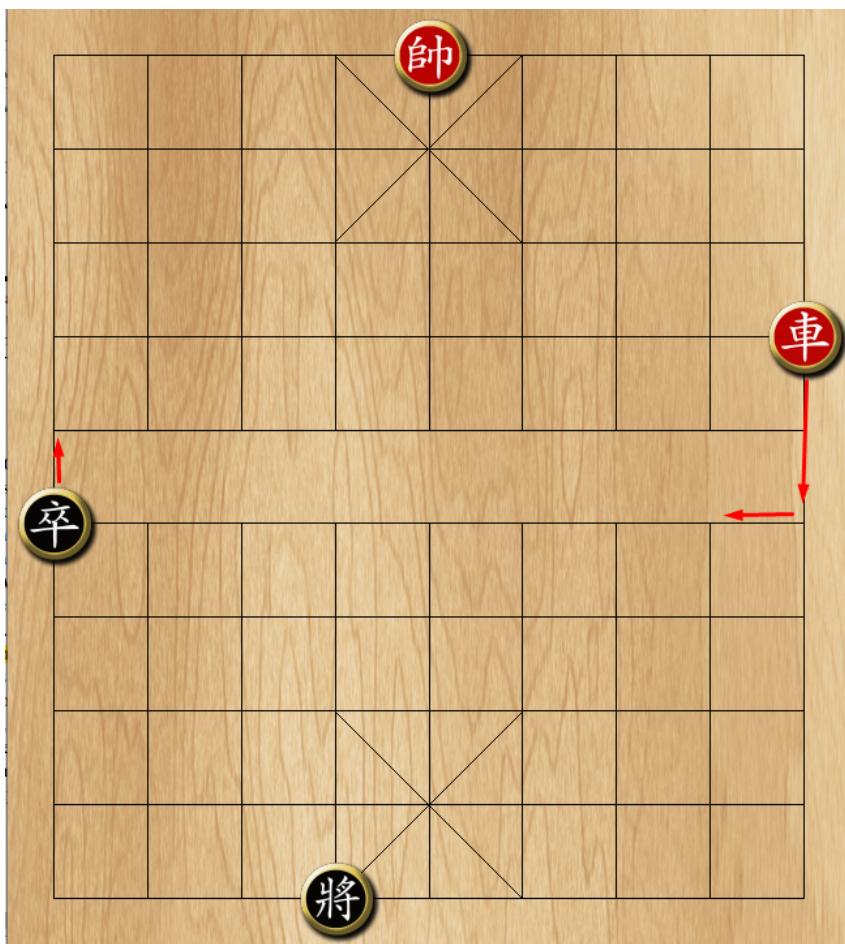
- Điểm: mỗi bàn cờ trong quá trình duyệt AlphaBeta đều có điểm lượng giá (điểm này có thể là điểm thực hoặc điểm được truyền ngược lên từ những nút con của nó).
- Cờ: Ở đây chúng ta sẽ có ba loại cờ là:
 - Chính xác: tức là điểm của trạng thái thỏa mãn điều kiện: $\alpha < \text{điểm} < \beta$.
 - Cận dưới: tức là điểm trạng thái thỏa mãn điều kiện: $\text{điểm} \leq \alpha$
 - Cận trên: tức là điểm trạng thái thỏa mãn điều kiện: $\text{điểm} \geq \beta$
- Độ sâu: nghĩa là khoảng cách từ trạng thái đó tới trạng thái gốc.

Trước khi duyệt một trạng thái, ta sẽ truy xuất xem trong bảng transposition table có trạng thái đó hay chưa? Nếu có thì ta sẽ chia các trường hợp như sau:



Hình 4.4: hiện tượng horizon effect

- Nếu độ sâu trong bảng transposition table nhỏ hơn hoặc bằng độ sâu trạng thái đang duyệt và có cờ là cờ chính xác. Ở trường hợp này thì lập tức trả về điểm trong transposition table.
- Nếu độ sâu trong bảng transposition table nhỏ hơn hoặc bằng độ sâu trạng thái đang duyệt, có cờ là cận dưới và điểm $> \alpha$ của trạng thái đang duyệt thì điều chỉnh $\alpha =$ điểm.
- Nếu độ sâu trong bảng transposition table nhỏ hơn hoặc bằng độ sâu trạng thái đang duyệt, có cờ là cận dưới và điểm $< \beta$ của trạng thái đang duyệt thì điều chỉnh $\beta =$ điểm.
- Nếu độ sâu trong bảng transposition table lớn hơn độ sâu của trạng thái đang duyệt thì chỉ dùng được nước đi trong bảng để áp dụng vào move ordering.



Hình 4.5: Hình ảnh về một bàn cờ bị lặp trạng thái

4.7 Kiểm soát thời gian duyệt bằng giải thuật iterative depth first search

Khi làm một chương trình cho người khác chơi, ngoài những giới hạn về phần cứng của máy tính, thì có một tài nguyên nữa chúng ta nên xem xét đó là thời gian. Một người chơi không thể nào đợi 1 tiếng để chờ máy đánh nước tiếp theo được. Họ không có thời gian như vậy. Với việc tiếp cận cây bằng số tầng biết trước thì sẽ không thể kiểm soát được thời gian. Vì nếu đầu trận, việc cho chương trình sinh 4 tầng có thể tốn 1 phút nhưng đến cuối trận thì chỉ mất khoảng mấy chục giây. Hướng tiếp cận bằng việc từ thời gian mà tính ra được số tầng để tìm kiếm là hướng tiếp cận tốt hơn cho cả người dùng và chương trình. Chương trình có giải thuật iterative depth first search có dạng như sau:

Algorithm 13 Giải thuật iterative depth first search

```

1: function interative_depth_first_search
2:   for int i = 1; i < MAX_DEPTH; i++ do
3:     search with depth i
4:     if over time then
5:       break
6:     end if
7:     if lose or win then
8:       break
9:     end if
10:    end for
11:    return best_move
12: end function

```

4.8 Giải thuật Zobrist Hashing

Khi sử dụng một số giải pháp như transposition table hay cơ sở dữ liệu sẽ dẫn đến nhu cầu lưu trữ bàn cờ lại để truy xuất nhanh chóng và tiết kiệm bộ nhớ. Khi đó giáo sư Albert Zobrist tại trường đại học Wisconsin đã đưa ra giải pháp Zobrist Hashing. Phương pháp này cho phép tạo ra một giá hash duy nhất từ một bàn cờ để tiện trong việc lưu trữ và truy xuất. Để hiện thực giải thuật này ta sẽ trải qua hai bước:

1. Sinh ra một dãy số, mỗi số đại diện cho loại quân cờ và vị trí của nó trong bàn cờ.
2. Với mỗi quân cờ trên bàn cờ ta sẽ XOR với giá trị hash, giá trị hash ban đầu bằng 0.

Algorithm 14 Giải thuật Zobrist Hashing [24]

```

1: Table[ofBoardCells][ofPieces]
2: function findhash(board)
3:   hash = 0
4:   for each cell on the board do
5:     if cell is not empty then
6:       piece = board[cell]
7:       hash ^= table[cell][piece]
8:     end if
9:   end for
10:  return hash
11: end function

```

Chúng ta không cần phải tính lại giá trị Zobrist hash từ đầu sau mỗi lần di chuyển quân cờ mà ta sẽ cập nhật lại Zobrist Hash bằng cách dùng toán tử XOR. Khi muốn xóa

một quân cờ khỏi giá trị Zobrist hash thì ta chỉ cần dùng hash XOR với số đại diện cho quân cờ tại vị trí đó. Bởi vì khi ta XOR một giá trị hai lần với cùng một số thì ta sẽ được giá trị ban đầu. Ví dụ $1110 \text{ XOR } 1001 = 0111$, $0111 \text{ XOR } 1001 = 1110$.

Độ dụng rộng rãi xảy ra khi ta hash hai bàn cờ khác nhau ra cùng một giá trị băm. Mục tiêu ở đây là giảm thiểu khả năng xảy ra độ dụng rộng rãi xuống một giá trị đủ nhỏ để tốc độ đạt được vượt xa ảnh hưởng tiêu cực của độ dụng rộng rãi. Do đó, khi sinh dãy số đại diện cho quân cờ 64 bit thường được sử dụng làm tiêu chuẩn và rất khó xảy ra độ dụng rộng rãi.

4.9 Cơ sở dữ liệu

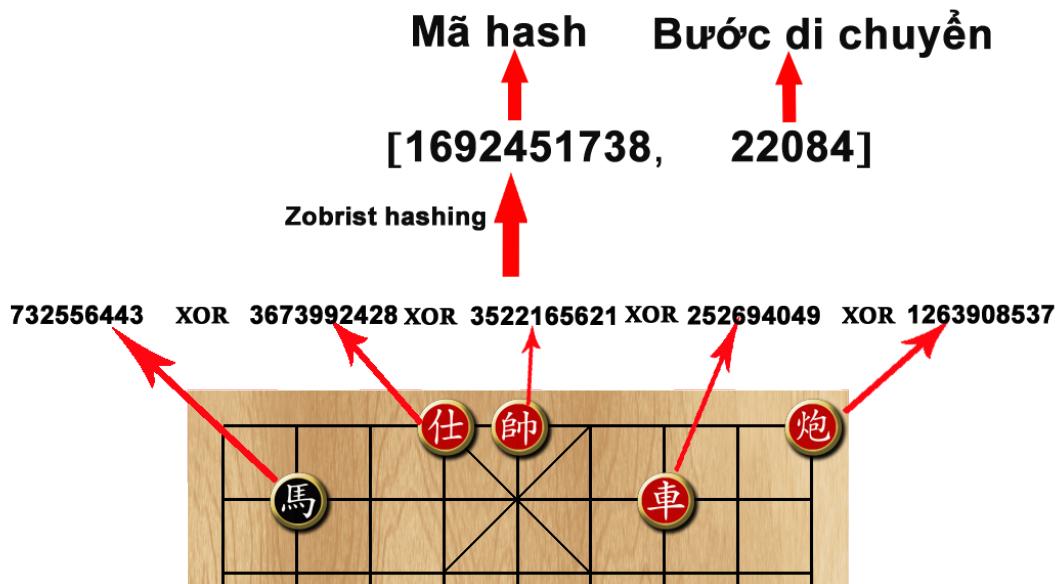
Cơ sở dữ liệu hay trong các phần mềm chơi cờ tướng người ta còn gọi là "Book" là tập hợp dữ liệu của các thế cờ và nước đi tương ứng do người làm Book thu thập từ các cuốn sách về cờ tướng, các giải đấu hay là các phần mềm cờ tướng... Book có 3 loại:

- Book khai cuộc: GUI sẽ tham khảo để chơi ở những nước đầu tiên của ván đấu
- Book tàn cuộc: GUI sẽ tham khảo để kết thúc ván cờ
- Book trung cục: GUI sẽ tham khảo để chơi khi đang ở giữa ván đấu. Tuy nhiên loại này có ít ứng dụng và đã chìm dần vào quên lãng do có rất nhiều biến hóa khác nhau và thường sử dụng AI Engine sẽ hiệu quả hơn.

Trong cờ tướng, khai cục rất quan trọng và chiếm một phần trọng yếu, nếu khai cuộc không tốt thì dẫn đến ván cờ bất ổn và rất khó chơi về sau. Vì vậy sử dụng Book trong khai cục rất quan trọng, Book đó là những kinh nghiệm nhân gian đúc rút được rất nhiều các giải đấu lớn và trong thực tế thì các kì thủ chuyên nghiệp trong phần khai cuộc sẽ đi như một công thức có sẵn. Vậy tại sao không dùng AI Engine khi khai cuộc? Có một số vấn đề mà ta cần nêu ra như sau:

- Thứ nhất về thời gian: Khi khai cuộc thì AI sinh ra một số lượng tầng và nút rất lớn nên thời gian sẽ rất lâu, trong khi đó khi dùng book thì thời gian xảy ra gần như ngay lập tức như thể phần mềm đang có sẵn một bí kíp trong tay.
- Thứ hai là AI Engine sẽ không có hướng khai cục phong phú, không có cái nhìn đại cục và rất dễ rơi vào lối mòn lặp đi lặp lại bị đối thủ bắt bài ngay về sau.

Cơ sở dữ liệu là một danh sách các phần tử và mỗi phần tử gồm hai thành phần: một con số đại diện cho một thế cờ và nước đi tương ứng của thế cờ đó. Để mã hóa bàn cờ ra một con số sử dụng giải thuật Zobrist hashing đã được nêu ở mục 4.8



Hình 4.6: Cấu trúc của cơ sở dữ liệu

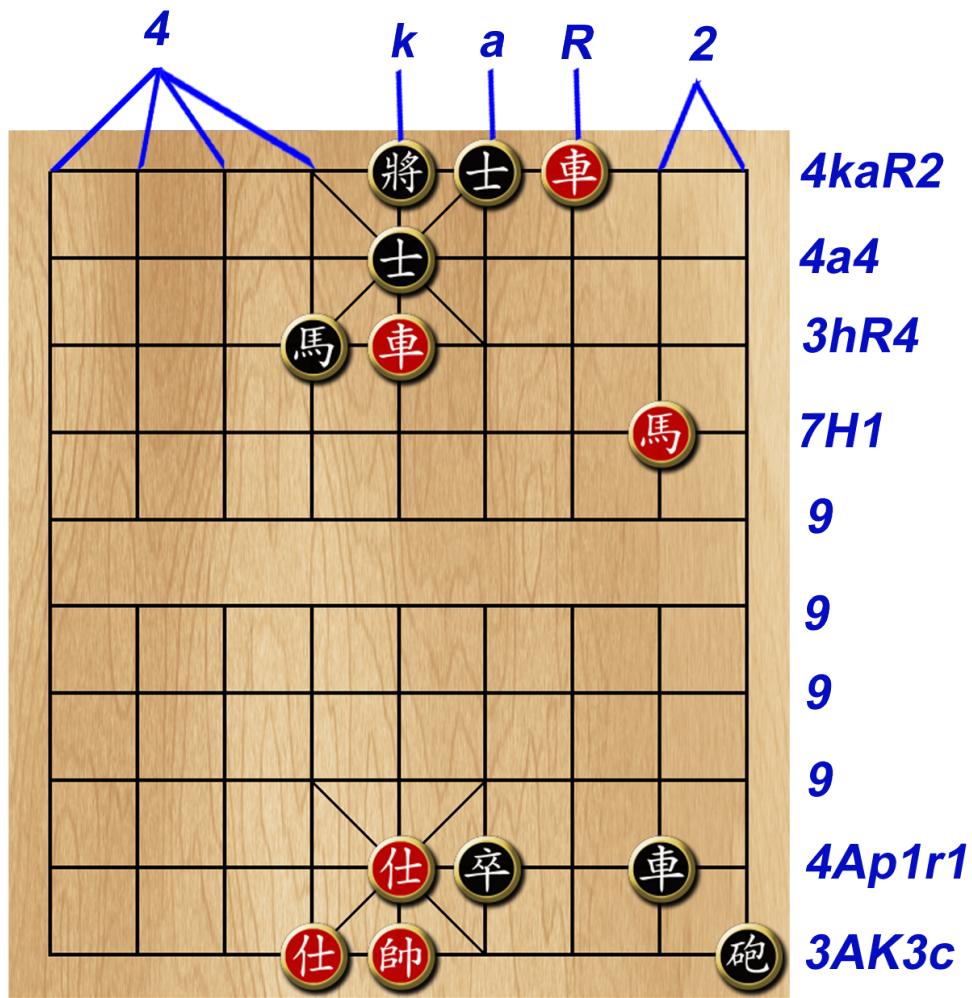
4.10 Chuỗi FEN

FEN là chữ viết tắt từ "Forsyth-Edwards Notation" nó là một tiêu chuẩn để mô tả vị trí cờ Vua (hoặc cờ Tướng) sử dụng bộ kí tự ASCII. Ưu điểm của FEN là tương đối đơn giản dễ ghi, dễ diễn giải, dễ lập trình cho máy và khá ngắn gọn [25].

FEN là một chuỗi kí tự được viết trên một dòng, các quân cờ được thay thế bằng chữ cái . Quân Trắng (Đỏ) được viết bằng chữ cái in hoa ("KAERCHP") và quân Đen (Xanh) được viết bằng chữ cái in thường ("kaerchp"). Những điểm trống liên tiếp nhau trên bàn cờ được viết bằng chữ số từ 1 tới 9. Kí tự gạch chéo "/" dùng để ngăn cách giữa các hàng. Ở cuối chuỗi là chữ thường "w" nếu quân Trắng (Đỏ) đi và chữ "b" nếu là quân Đen (Xanh) đi.

quân	tướng đỏ	sĩ đỏ	tượng đỏ	xe đỏ	pháo đỏ	mã đỏ	tốt đỏ
Kí hiệu	K	A	E	R	C	H	P
quân	tướng đen	sĩ đen	tượng đen	xe đen	pháo	mã đen	tốt đen
Kí hiệu	k	a	e	r	c	h	p

Bảng 4.8: Kí hiệu các quân cờ trong FEN



4kaR2/4a4/3hR4/7H1/9/9/9/4Ap1r1/3AK3c w

Hình 4.7: Chuyển đổi bàn cờ thành chuỗi Fen

Chương 5

Trình bày kết quả hiện thực chương trình

Trong chương này, nhóm sẽ trình bày các kết quả đạt được như: Elo của AI, So sánh các thuật toán với nhau, Trình bày các tính năng của trò chơi,...

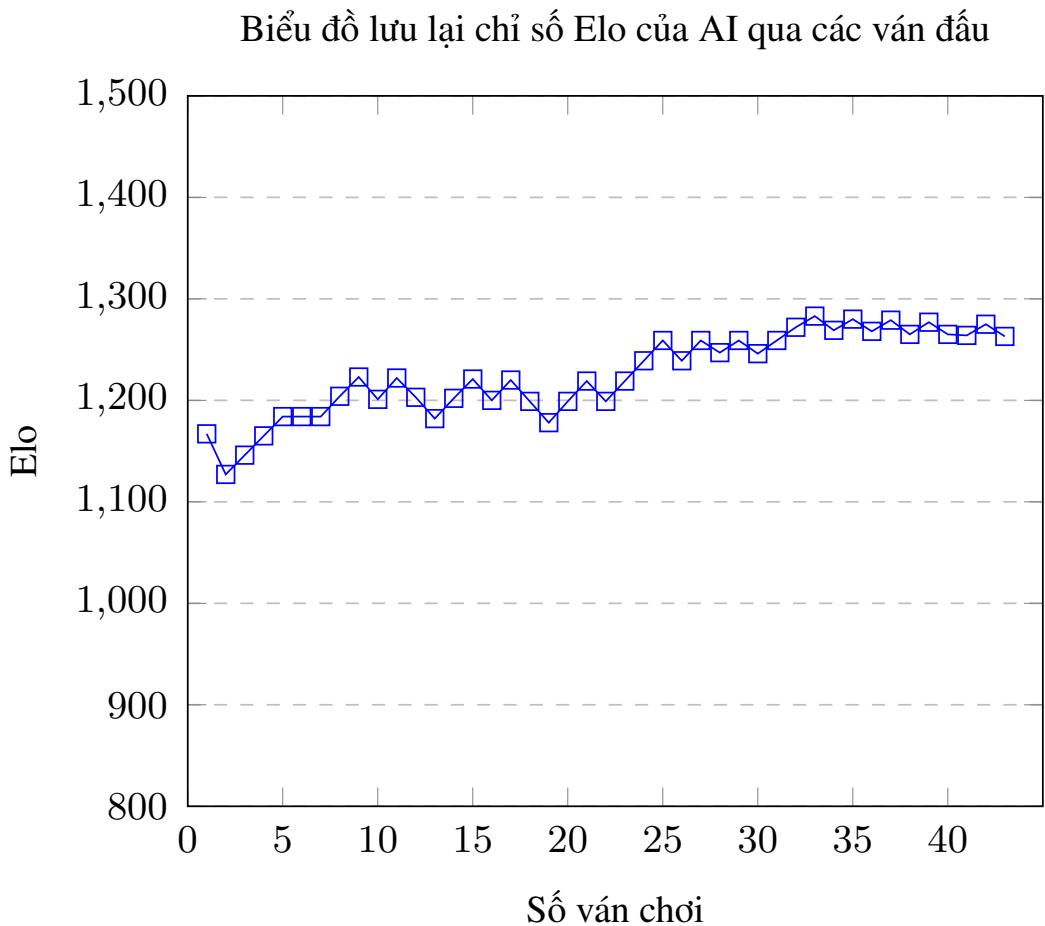
5.1 Đánh giá sức mạnh của AI

5.1.1 Đo chỉ số Elo của AI

Một ứng dụng được làm ra thì việc thử nghiệm trong môi trường thực tế là quan trọng. Nếu không thử nghiệm trong môi trường thực tế thì ứng dụng đó dễ bị "đắp chiếu". Để đánh giá sức mạnh của AI, nhóm đã cho AI chơi trực tuyến trên địa chỉ <https://apps.facebook.com/ziga-game>. Biểu đồ trên thể hiện chỉ số Elo của AI qua từng ván đấu thực tế.

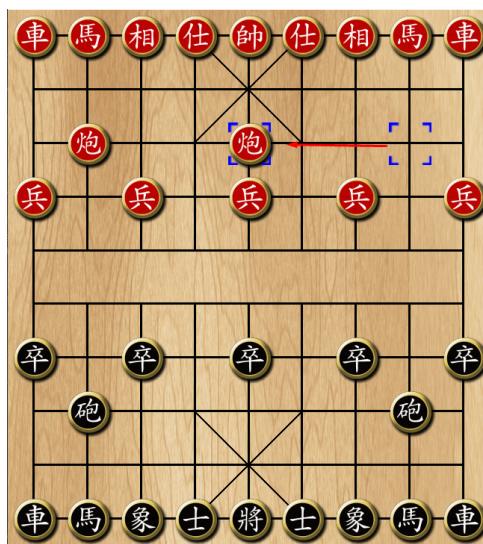
CPU	Itel(R) Core(TM) i5-5200U CPU 2.20GHz
RAM	8GB
Disk type	SSD
GPU	NVIDIA GeForce 820M

Bảng 5.1: Bảng mô tả cấu hình máy tính xách tay



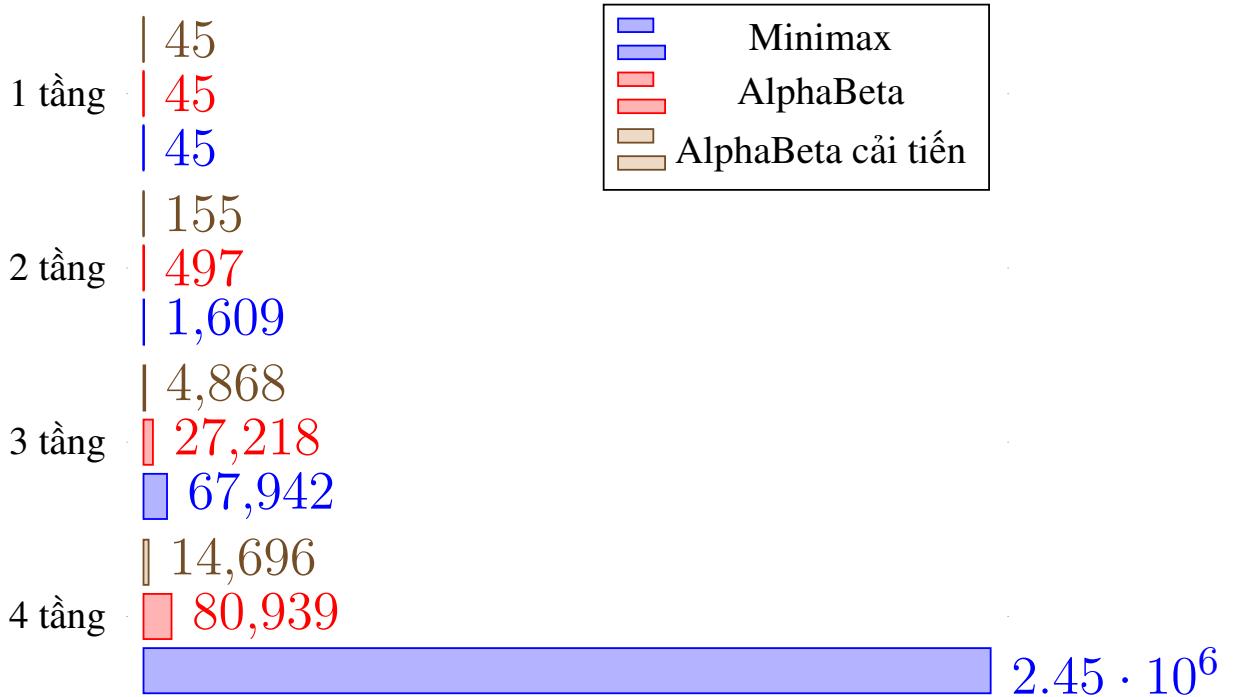
5.1.2 So sánh độ hiệu quả giữa các thuật toán

Nhóm sẽ lấy một trường hợp khai cục như hình 5.1 để xem các thuật toán sẽ phải duyệt bao nhiêu số nút để tìm ra được nước đi. Về cơ bản, nếu cùng số tầng thì thuật toán nào phải duyệt ít nước đi hơn thì sẽ tốt hơn.



Hình 5.1: Thế trận pháo đầu

So sánh tổng số nút giữa các thuật toán



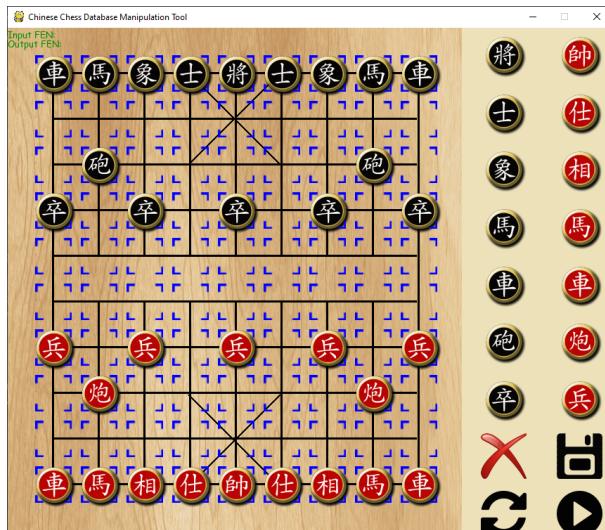
Nhóm cũng so sánh giải thuật bằng thời gian chạy trên cùng chiếc máy tính xách tay có cấu hình như bảng 5.1.

So sánh thời gian chạy (giây) giữa các thuật toán



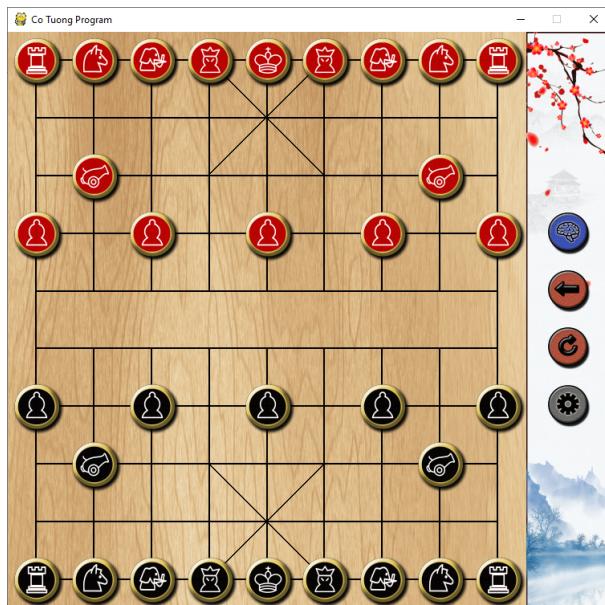
5.2 Giao diện công cụ nhập liệu

Công cụ nhập liệu dùng để nhập các thế cờ rồi lưu xuống dạng JSON. Nếu lần sau gặp lại nước cờ như vậy thì AI sẽ lấy thẳng kết quả ra mà không tính nữa.



Hình 5.2: Giao diện công cụ nhập liệu

5.3 Giao diện trò chơi



Hình 5.3: Giao diện trò chơi cờ tướng

Giao diện trò chơi gồm ba phần:

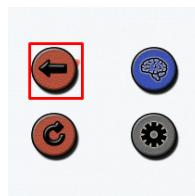
- Phần 1: là bàn cờ. Đây là phần chính dùng để chơi cờ.

- Phần 2: là thanh công cụ bên phải. Ở đây chứa những nút chứa những tính năng phổ biến của chương trình.
- Phần 3: là phần chứa những tính năng chi tiết. Đây là cửa sổ ẩn. Chỉ xuất hiện khi người dùng mở lên.

5.4 Các chức năng của trò chơi

5.4.1 Chức năng đi lại quân

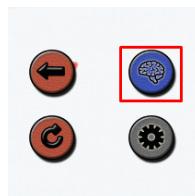
Khi người dùng vì lí do nào đó muốn đi lại thì có thể chọn đi lại bằng cách bấm vào nút có hình mũi tên quay sang trái như hình 5.4.



Hình 5.4: Chức năng đi lại quân

5.4.2 Chức năng chọn chế độ chơi

Chương trình cung cấp cho người chơi 2 chế độ: người chơi với người và người chơi với máy. Người dùng có thể chuyển chế độ bằng cách bấm vào nút như hình 5.5.



Hình 5.5: Chức năng chọn chế độ chơi

5.4.3 Chức năng chơi ván mới

Khi người dùng muốn chơi ván mới có thể chọn nút như hình vẽ

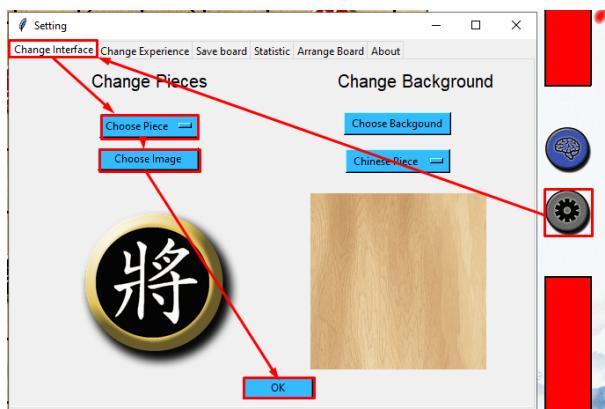


Hình 5.6: Chức năng chơi ván mới

5.4.4 Chức năng thay đổi quân cờ theo hình ảnh có sẵn

Nếu người dùng muốn thay đổi hình ảnh quân cờ theo một hình ảnh khác (ảnh đại diện của người đó chẳng hạn) thì người dùng có thể làm theo các bước sau:

- Bước 1: chọn nút setting. Một cửa sổ mới sẽ xuất hiện.
- Bước 2: chọn tab "Change Interface".
- Bước 3: chọn quân cờ muốn thay đổi.
- Bước 4: nhấn vào nút "Choose image". Sau đó chọn một hình ảnh có trong máy.
- Bước 5: nhấn "OK" để kết thúc.

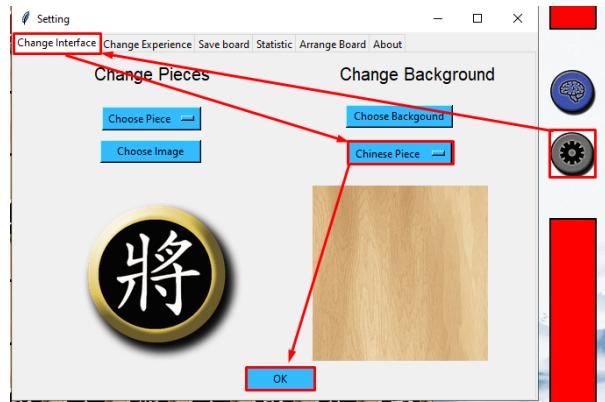


Hình 5.7: Chức năng thay đổi quân cờ theo hình ảnh có sẵn

5.4.5 Chức năng chọn thể loại quân cờ cho trước

Người dùng vẫn có thể thay đổi loại quân cờ cho trước theo những bước sau:

- Bước 1: chọn nút setting. Một cửa sổ mới sẽ xuất hiện.
- Bước 2: chọn tab "Change Interface".
- Bước 3: chọn loại quân cờ yêu thích.
- Bước 5: nhấn "OK" để kết thúc.

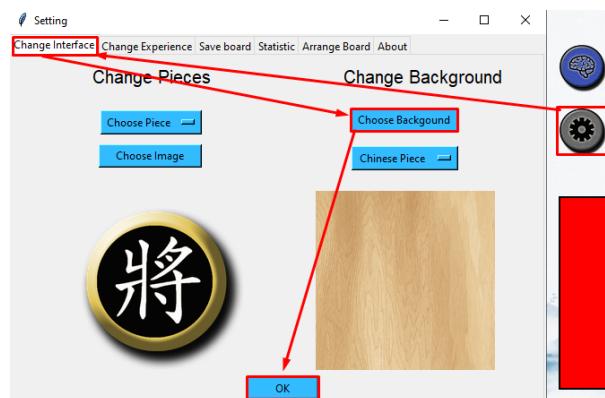


Hình 5.8: Chức năng chọn thể loại quân cờ cho trước

5.4.6 Chức năng thay đổi background

Người dùng có thể thay đổi background của bàn cờ bằng cách:

- Bước 1: chọn nút setting. Một cửa sổ mới sẽ xuất hiện.
- Bước 2: chọn tab "Change Interface".
- Bước 3: bấm vào nút "Choose Background" và chọn hình ảnh yêu thích.
- Bước 5: nhấn "OK" để kết thúc.

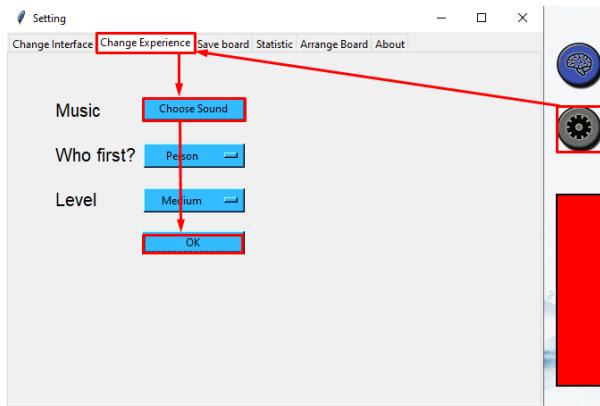


Hình 5.9: Chức năng thay đổi background

5.4.7 Chức năng thay đổi nhạc nền trò chơi

Người dùng có thể thay đổi background của bàn cờ bằng cách:

- Bước 1: chọn nút setting. Một cửa sổ mới sẽ xuất hiện.
- Bước 2: chọn tab "Change Experience".
- Bước 3: bấm vào nút "Choose Sound" và file nhạc bạn yêu thích.
- Bước 5: nhấn "OK" để kết thúc.

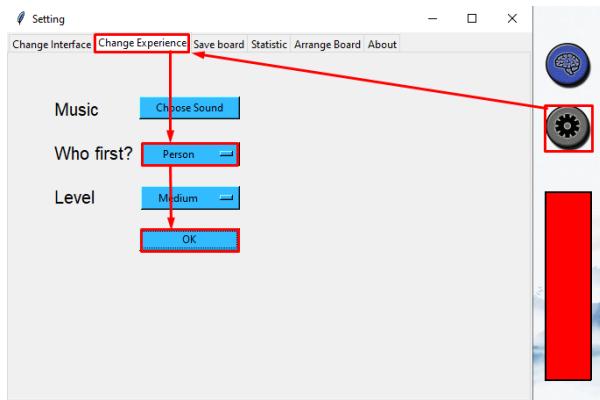


Hình 5.10: Chức năng thay đổi nhạc nền trò chơi

5.4.8 Chức năng chọn người đi trước

Người dùng có thể thay đổi background của bàn cờ bằng cách:

- Bước 1: chọn nút setting. Một cửa sổ mới sẽ xuất hiện.
- Bước 2: chọn tab "Change Experience".
- Bước 3: bấm vào nút "Person" và chọn Person hoặc Computer.
- Bước 5: nhấn "OK" để kết thúc.

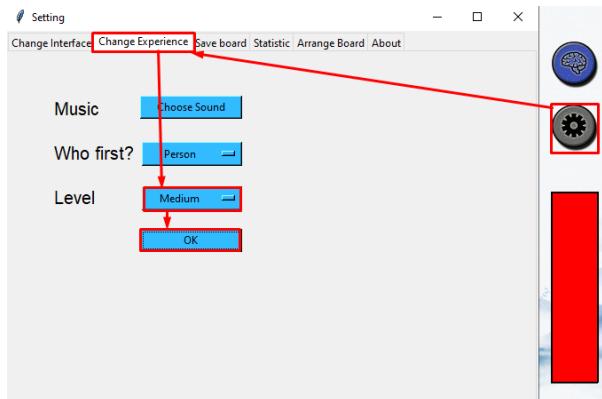


Hình 5.11: Chức năng chọn người đi trước

5.4.9 Chức năng chọn mức độ chơi

Người dùng có thể thay đổi background của bàn cờ bằng cách:

- Bước 1: chọn nút setting. Một cửa sổ mới sẽ xuất hiện.
- Bước 2: chọn tab "Change Experience".
- Bước 3: bấm vào nút "Medium" và chọn Easy hoặc Medium hoặc Difficult.
- Bước 5: nhấn "OK" để kết thúc.

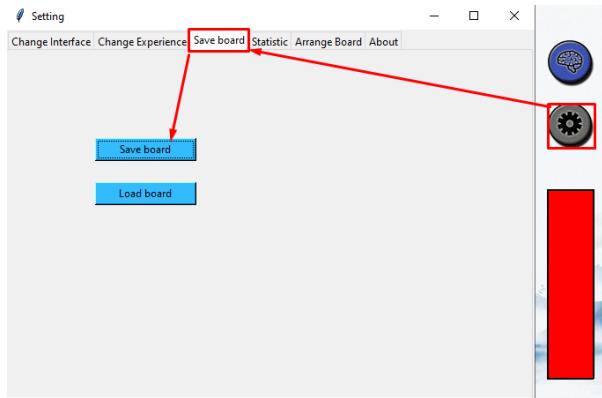


Hình 5.12: Chức năng chọn mức độ chơi

5.4.10 Chức năng lưu lại bàn cờ

Khi đang chơi cờ, người dùng có việc bận và muốn tạm ngưng nhưng vẫn muốn lưu lại bàn cờ. Chương trình có tính năng lưu lại bàn cờ. Người dùng thực hiện theo các bước như sau:

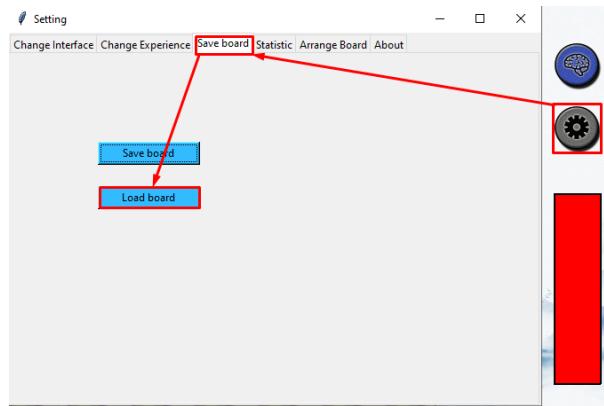
- Bước 1: chọn nút setting. Một cửa sổ mới sẽ xuất hiện.
- Bước 2: chọn tab "Save board".
- Bước 3: bấm vào nút "Save board" để lưu lại bàn cờ.



Hình 5.13: Chức năng lưu lại bàn cờ

Khi người dùng muốn chơi lại bàn cờ đã lưu, người dùng chỉ cần thực hiện theo các bước sau:

- Bước 1: chọn nút setting. Một cửa sổ mới sẽ xuất hiện.
- Bước 2: chọn tab "Save board".
- Bước 3: bấm vào nút "Load board" để mở bàn cờ.

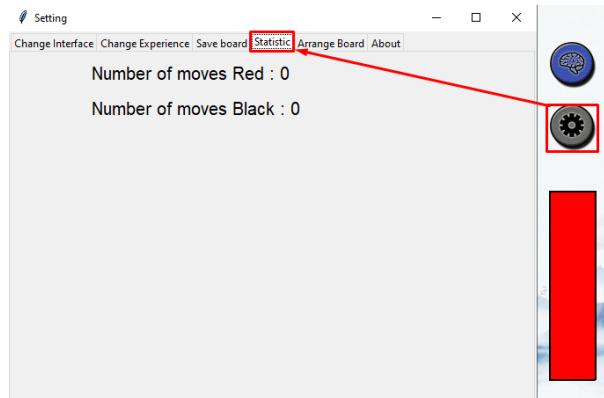


Hình 5.14: Chức năng lưu lại bàn cờ

5.4.11 Chức năng thống kê

Chương trình cũng có tính năng thống kê lại một số thông tin cho người dùng:

- Bước 1: chọn nút setting. Một cửa sổ mới sẽ xuất hiện.
- Bước 2: chọn tab "Statistic".

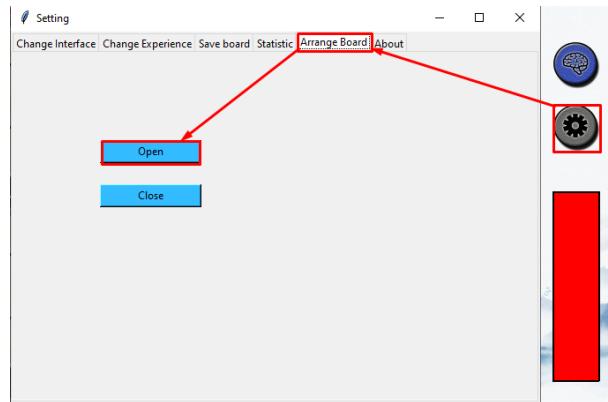


Hình 5.15: Chức năng thống kê

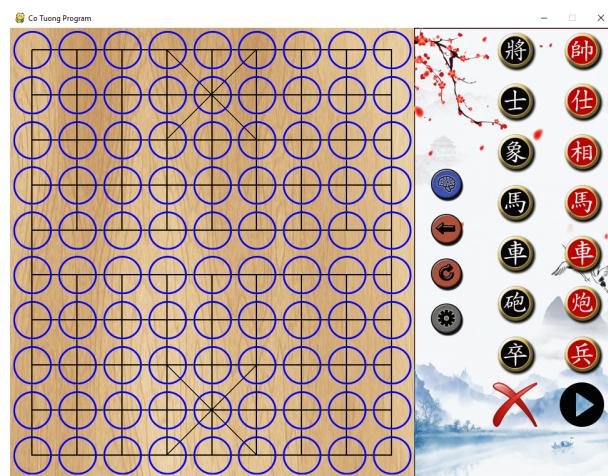
5.4.12 Chức năng sắp cờ thế

Chương trình cũng có tính năng thống kê lại một số thông tin cho người dùng:

- Bước 1: chọn nút setting. Một cửa sổ mới sẽ xuất hiện.
- Bước 2: chọn tab "Arrange Board".
- Bước 3: chọn nút Open để mở cửa sổ sắp cờ.
- Bước 4: sau khi sắp cờ thì người dùng tiến hành chơi bình thường.



Hình 5.16: Chức năng sắp cờ thê



Hình 5.17: Chức năng sắp cờ thê

Chương 6

Tổng kết

Trong chương này, nhóm sẽ đưa ra kết luận, đánh giá ưu nhược điểm và hướng phát triển sắp tới.

6.1 Kết luận

Sau giai đoạn luận văn, nhóm đã xây dựng được một chương trình chơi cờ hoàn chỉnh: giao diện ưa nhìn, chương trình có những tính năng thông dụng, AI có sức mạnh tương đối tốt (mục 5.1). Ngoài ra, nhóm cũng cố gắng viết tài liệu liên quan một cách chi tiết và dễ hiểu để phục vụ cho việc phát triển phần mềm sau này được dễ dàng hơn. Nhóm cũng cấu trúc thư mục, tổ chức mã nguồn, viết mã để làm sao cố gắng mạch lạc nhằm đảm bảo tính dễ đọc, dễ sửa chữa và thay đổi sau này.

6.2 Ưu điểm

Những ưu điểm đạt được sau khi hoàn thành luận văn:

- Chương trình đã được hoàn thiện và có thể sử dụng được.
- Chương trình có giao diện bắt mắt và có hiệu ứng khi chơi.
- Chương trình có các tính năng phổ biến của một chương trình chơi cờ như: lưu bàn cờ, chọn chế độ chơi, chọn bên đi trước, chọn sức mạnh của AI, ...
- AI có sức mạnh tốt và liên tục tăng Elo theo từng lần cải tiến (mục 5.1).
- AI đã được chơi thực tế, đo được Elo nhằm đánh giá khách quan và trung thực hơn.
- Nhóm có xây dựng công cụ hỗ trợ nhập thế cờ vào cơ sở dữ liệu của AI để người dùng có thể nhập thế cờ một cách đơn giản hơn. Từ đó giúp AI có cơ sở dữ liệu phong phú và lớn hơn, kéo theo AI chơi sẽ tốt hơn.
- Cấu trúc thư mục, mã nguồn được sắp xếp theo tinh thần dễ đọc, dễ hiểu, dễ sửa chữa và thêm tính năng mới sau này.

6.3 Khuyết điểm

Những khuyết điểm sau khi hoàn thành luận văn:

- Trò chơi chưa được phát triển trên nhiều nền tảng như: điện thoại thông minh, web để tiếp cận nhiều người dùng hơn.
- Chương trình chưa có tính năng cho phép người chơi trực tuyến với nhau qua mạng máy tính.
- Hiện thực AI engine bằng ngôn ngữ Python nên chưa được hiệu suất cao bằng các ngôn ngữ biên dịch như C, C++, ...
- Hiện thực AI engine chạy đơn luồng và một tiến trình nên chưa tận dụng được hết sức mạnh của những con chíp nhiều nhân hiện nay trên máy tính.
- Vẫn còn nhiều kỹ thuật để nhằm tối ưu việc cắt tỉa cây chưa được nhóm tham khảo và áp dụng vào AI engine.

6.4 Hướng phát triển

Sau khi nhận thấy các ưu nhược điểm, nhóm có những hướng phát triển như sau:

- Viết lại chương trình trên nhiều nền tảng khác để tiếp cận người dùng hơn như: IOS, Android,...
- Tối ưu hình ảnh của chương trình để cho chương trình đẹp hơn nữa.
- Thêm tính năng cho phép người chơi trực tuyến với nhau trên mạng.
- Nghiên cứu khả năng viết giao diện chương trình trò chơi bằng 3D.
- Hiện thực lại AI engine trên ngôn ngữ C++ để đạt được hiệu suất cao hơn.
- Áp dụng kỹ thuật đa luồng, đa tiến trình trên AI engine để tận dụng những con chíp nhiều nhân hiện nay.
- Áp dụng các kỹ thuật cải tiến cắt tỉa nhánh để giúp AI engine có thể duyệt sâu hơn, từ đó nâng cao sức mạnh của AI.
- Viết thêm một AI engine nữa nhưng hướng tiếp cận bằng cây Monte Carlo để so sánh ưu nhược điểm giữa cây Minimax và Monte Carlo.
- Nghiên cứu và phát triển AI engine cho trò chơi cờ úp (chinese dark chess) vì đây là trò chơi này thách thức hơn do thông tin là không hoàn hảo những vẫn rất gần với cờ tướng.

Tài liệu tham khảo

- [1] C. L. Lim, “Exploring strategies through evolutionary algorithms and neural networks,” pp. 12–14, 2009.
- [2] “Design exercise: Flip book animation.” <http://infinitedictionary.com/blog/2015/06/03/design-exercise-flip-book-animation/>. truy cập vào 2019-11-11.
- [3] “Frame rate là gì? sử dụng frame rate thế nào cho hiệu quả?” <https://colorme.vn/blog/frame-rate-la-gi-su-dung-frame-rate-the-nao-cho-hieu-quocolorme>. truy cập vào 2019-11-11.
- [4] “Game loop.” <https://gameprogrammingpatterns.com/game-loop.html>. truy cập vào 2019-11-11.
- [5] <https://www.taigamek.mobi/fifa-online-4-fo4/>. truy cập vào 2019-11-11.
- [6] “Super mario bros.” <https://www.nintendo.co.uk/Games/NES/Super-Mario-Bros-803853.html>. truy cập vào 2019-11-11.
- [7] H. Ngọc, “Sôi nổi ngày hội giỗ tổ nghề mộc truyền thống kim bồng.” <http://cand.com.vn/Chuyen-dong-van-hoa/Soi-noi-ngay-hoi-Gio-to-nghe-moc-truyen-thong-Kim-Bong-382392/>. Truy cập 13-10-2019.
- [8] “Kể chuyện cờ tướng p1 : Nguồn gốc cờ tướng.” <http://vietnamchess.vn/index.php/vi/special-news/55-letter/958-kcct1>. Truy cập 20-10-2019.
- [9] Anonymous, *Chơi cờ tướng như thế nào?*, p. 01. Thành Công.
- [10] R. T. Michael T. Goodrich, Roberto Tamassia, *Data Structures and Algorithms in C++*, p. 269. John Wiley & Sons, Inc, 2011.
- [11] “Search tree.” https://www.chessprogramming.org/Search_Tree. truy cập vào 2019-11-11.
- [12] C. Shannon, “Programming a computer for playing chess,” *Philosophical Magazine*, 1949.
- [13] “Negamax.” <https://www.chessprogramming.org/Negamax>. truy cập vào 2019-11-11.
- [14] “Minimax.” <https://www.chessprogramming.org/Minimax>. truy cập vào 2019-11-11.

- [15] “Alphabeta.” <https://www.chessprogramming.org/Alpha-Beta>. truy cập vào 2019-11-11.
- [16] https://rosettacode.org/wiki/Binary_search. truy cập vào 2019-05-11.
- [17] <https://nachtimwald.com/2018/01/18/binary-search-and-insert/>. truy cập vào 2019-09-08.
- [18] P. Garg, “Basics of hash tables.” <https://www.hackerearth.com/fr/practice/data-structures/hash-tables/basics-of-hash-tables/tutorial/>. Truy cập 30-10-2019.
- [19] <https://vnoi.info/wiki/algo/data-structures/hash-table>. truy cập vào 2019-11-11.
- [20] “Frame rate.” <https://www.techopedia.com/definition/3036/frame-rate>. truy cập vào 2019-11-11.
- [21] S. Madhav, *Game Programming Algorithms and Techniques*. AddisonWesley, 2013.
- [22] <https://www.keonhanh.com/blog-game/phan-mem-co-tuong-manh-nhat-the-gioi/>. truy cập vào 2019-09-08.
- [23] J. U. D.M. Breuker and H. van den Herik, “Information in transposition tables,” *Information in Transposition Tables*, 1970.
- [24] <https://www.geeksforgeeks.org/minimax-algorithm-in-game-theory-set-5-zobrist-hashing/>. truy cập vào 2019-01-11.
- [25] <http://wxf.ca/xq/computer/fen.pdf>. truy cập vào 2019-05-03.

Phụ lục A

Một vài số liệu thống kê

A.1 Thống kê về mã nguồn của hệ thống

EXTENSION NAME : linecount

EXTENSION VERSION : 0.1.7

count time : 2019-12-19 12:11:38

count workspace: c:\Users\nficu\Desktop\New folder\chinesechessai

total files : 199

total code lines : 4557

total comment lines : 617

....

Tổng số tập tin	199
Tổng số dòng lệnh	4557
Tổng số dòng comment	617

Bảng A.1: Bảng thống kê về mã nguồn

A.2 Thống kê về bài báo cáo

Tổng số bảng	12
Tổng số hình	48
Tổng số trang	61

Bảng A.2: Bảng thống kê về báo cáo

Phụ lục B

Hướng dẫn sử dụng chương trình

Chúng tôi giả định là bạn đã có GIT, Python và các framework được sử dụng trong chương trình như: Numpy, Pygame,... Nếu bạn chưa có, bạn có thể dễ dàng tải chúng bằng những hướng dẫn trên mạng. Ở đây, chúng tôi sẽ giả định bạn đã có môi trường, công việc chúng tôi là hướng dẫn các bạn tải và chạy chương trình.

B.1 Hướng dẫn tải chương trình về

Bạn mở GIT và clone thư mục tại địa chỉ: <https://gitlab.com/ThanhLongDo1511799/chinesechessai>.

Khi tải về bạn có cấu trúc thư mục như hình bên dưới:

Name	Date modified	Type	Size
AIEngine	12/19/2019 8:41 PM	File folder	
ChineseChessDatabaseManipulationTool	12/19/2019 11:58 AM	File folder	
ChineseChessGUI	12/19/2019 11:58 AM	File folder	
Notes.md	12/19/2019 8:41 PM	MD File	1 KB

Hình B.1

Lưu ý: Thư mục trên GIT chỉ được mở public khi nhóm đã báo cáo xong

B.2 Hướng dẫn chạy chương trình trò chơi cờ tướng

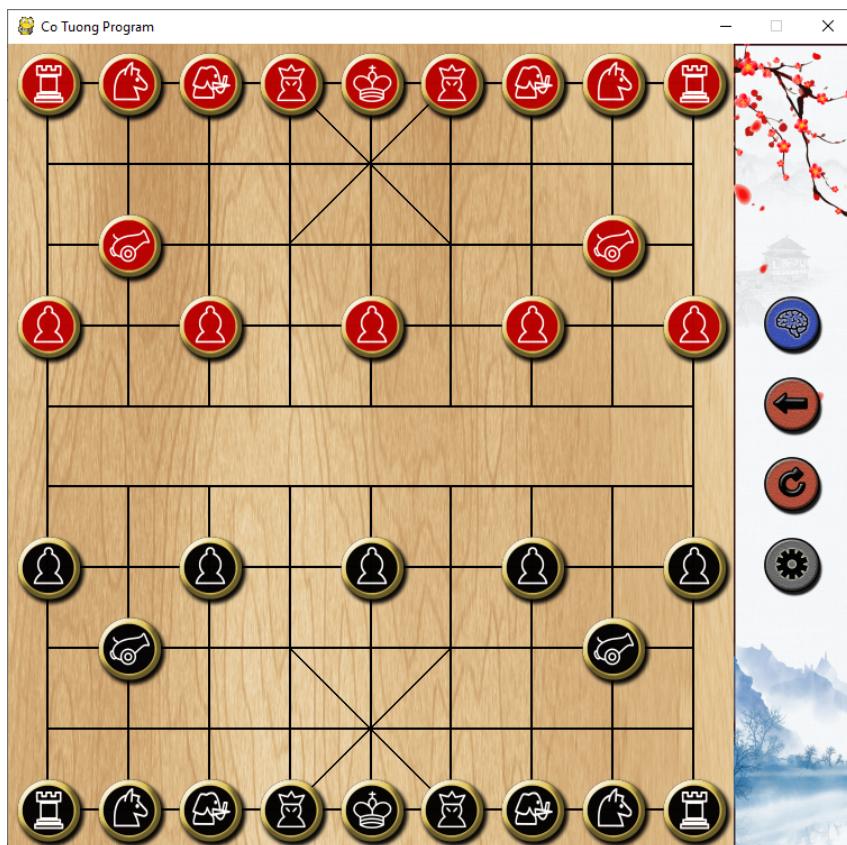
Để chạy được chương trình, bạn phải khởi động AI engine lên trước:

- Nhấn 2 lần chuột vào file: AIEngine\Source\Main.py
- Để im CMD này ở đó

Tiếp tục bạn khởi động chương trình trò chơi cờ tướng:

- Nhấn 2 lần chuột vào file: ChineseChessGUI\Source\Main.py
- Để im CMD này ở đó

Sau đó, chương trình của bạn sẽ được mở lên và có giao diện như thế này:



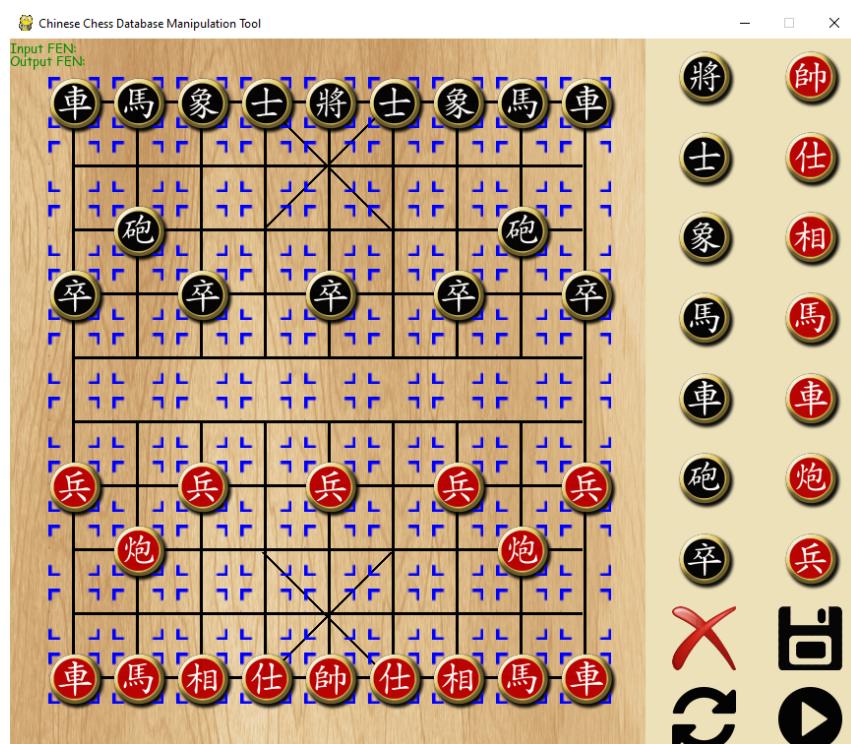
Hình B.2: Giao diện trò chơi

B.3 Hướng dẫn chạy công cụ hỗ trợ nhập liệu

Để chạy chương trình "công cụ hỗ trợ nhập liệu":

- Nhấn 2 lần chuột vào file: ChineseChessDatabaseManipulationTool\Source\Main.py
- Để im CMD này ở đó

Chương trình sẽ được chạy lên và có giao diện như thế này:



Hình B.3: Giao diện công cụ nhập liệu