ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA KHOA HỌC MÁY TÍNH

======*===*===*===*

ĐỒ ÁN MÔN HỌC TRÍ TUỆ NHÂN TẠO

ଌ★ଌ

Game cờ vua

Sinh viên thực hiện:

NGUYỄN KHẮC AN DƯƠNG – 17520384

NGUYỄN NGỌC TÀI – 17520997

PHẠM TRUNG THÀNH – 17521057

TRẦN TẤN ĐẠT – 17520342

TRƯƠNG BÁ LUÂN – 17520723

Lớp: CS106.J21

Năm học: 2018 – 2019

MỤC LỤC

LỜI NÓI ĐẦU

Nói đến cách mạng công nghiệp là nói đến sự thay đổi lớn lao mà nó mang lại trong các lĩnh vực kinh tế, văn hóa, và xã hội. Nhìn lại lịch sử, con người đã trải qua nhiều cuộc cách mạng khoa học kỹ thuật lớn. Mỗi cuộc cách mạng đều đặc trưng bằng sự thay đổi về bản chất của sản xuất và sự thay đổi này được tạo ra bởi các đột phá của khoa học và công nghệ. Cuộc cách mạng lần nhất là cuộc cách mạng cơ khí hóa với sự phát minh của máy thủy lực và máy hơi nước. Cuộc cách mạng lần hai với sự phát hiện ra điện năng, tạo một bước tiến đột phá lớn lao cho nền công nghiệp của Thế giới. Cuộc cách mạng lần ba là kỷ nguyên của máy tính, sự dần thay thế của các thiết bị máy móc đã giúp tăng cao năng suất lao động. Cuộc cách mạng lần bốn đang diễn ra rất sôi nổi với việc kết nối vạn vật từ thế giới thực đến thế giới ảo.

Nhìn lại hai cuộc cách mạng gần nhất với chúng ta hiện tại là cuộc cách mạng lần thứ ba và lần thứ tư. Chúng ta có thể thừa nhận rằng sự đóng góp rất lớn của máy tính trong mọi lĩnh vực đời sống, từ các dây chuyền nhà máy đến các thiết bị gia dụng. Để làm được điều đó bộ môn "Trí tuệ nhân tạo" đã ra đời để làm nên sự "thông minh" cho các thiết bị vô tri vô giác và từ những thiết bị đó đã đang sẽ phục vụ cho cuộc sống loài người, đồng hành cùng xã hội phát triển.

Trí tuệ nhân tạo phát triển song hành với sự phát triển của máy tính. Đóng góp ngày càng lớn cho mọi mặt của đời sống, trong đó có lĩnh vực giải trí. Từ đó nhóm sinh viên chúng em quyết định lựa chọn đề tài "Game Cờ Vua" trong đồ án này. Sự lựa chọn này là gắn với thực tiễn phát triển mạnh mẽ của công nghệ, tạo nên sự thuận lợi cho mọi người khi không cần phải tốn chi phí mua bàn cờ hay tốn thời gian gặp nhau thi đấu. Dù là một game đã có từ rất lâu đời và đã được áp dụng AI từ hàng chục năm về trước nhưng lựa chọn này là nền tảng và bước đi đầu tiên cho những dự án lớn sau này.

Với đồ án này chúng em làm rõ các vấn đề sau. Tổng quan cờ vua là gì?! Thuật toán được sử dụng cho game?! Cách cài đặt chương trình?! Đánh giá và hướng phát triển cho game?!

LỜI CẢM ƠN

Nhóm sinh viên chúng em xin gửi lời cảm ơn chân thành và sâu sắc tới Thầy ThS. Phạm Nguyễn Trường An. Suốt một học kì qua nhóm sinh viên chúng em đã nhận được sự chỉ bảo rất tận tình, tâm huyết từ thầy. Trải qua một kì học tập với nhiều kiến thức thu nhặt được từ môn "Trí tuệ nhân tạo", chúng em có được kiến thức bổ ích về lý thuyết cũng như thực hành.

Trong quá trình học môn này, chúng em được nhận đồ án với chủ đề "Game Cờ Vua". Đây là một thách thức với nhóm sinh viên chúng em, nhưng khi nhìn lại đây là một cơ hội rất tốt cho chúng em được chứng minh năng lực của mình, vận dụng những kiến thức đã được học vào thực tiễn để tạo ra một sản phẩm ích lợi. Hơn thế nữa, đây là dịp cho chúng em học hỏi kinh nghiệm, được lắng nghe những lời nhận xét chân thành từ Thầy. Đó là nền tảng cho chúng em có được nhiều kiến thức và trải nghiệm để học tập và làm việc sau này.

Nhóm sinh viên chúng tôi cũng gửi lời cảm ơn đến các bạn lớp CS106.J21, trong quá trình học tập và làm đồ án chúng tôi đã nhận được sự tư vấn chia sẻ rất quý báu về kiến thức và sự động viên rất chân thành từ các bạn. Có thể khẳng định rằng đó là nguồn động lực và là yếu tố rất cần thiết để chúng tôi hoàn thành đồ án này.

Để hoàn thành đồ án này là sự nỗ lực rất lớn của nhóm sinh viên chúng em. Chúng em đã cố gắng hết sức mình cho công việc, từ việc tìm kiếm nguồn tài liệu, tìm tòi thông tin trên các sách, báo, trang mạng cho đến những buổi họp nhóm thường xuyên để làm tốt đồ án. Đồ án này là sự tâm huyết, lòng nhiệt thành, sự bao dung của tất cả các thành viên trong nhóm. Nhưng dù đã nỗ lực hết sức vẫn không thể tránh được sai xót và khuyết điểm trong đồ án này. Nhóm sinh viên chúng em mong muốn nhận được sự nhận xét và đóng góp của Thầy cũng như các bạn để chúng em hoàn thiện và làm tốt hơn nữa.

Xin chân thành cảm ơn!

Tp. Hồ Chí Minh, ngày 20 tháng 06 năm 2019

CHƯƠNG I: TỔNG QUAN CỜ VUA

1. Giới thiệu

Giải vô địch cờ vua thế giới (World Chess Champion - WCC) được xem là một trong những cuộc thi về cờ vua lớn nhất Thế giới ở thời điểm hiện tại. Cuộc thi được tổ chức lần đầu tiên vào năm 1886, sau đó cuộc thi được tổ chức qua nhiều năm nhưng không có sự thống nhất về chu kì tổ chức. Mãi đến năm 2014, cuộc thi mới được thống nhất tổ chức với chu kì 2 năm 1 lần và tổ chức vào năm chẵn.

2. Lịch sử [1]

Giải đấu lần đầu tiên được tổ chức vào năm 1886, dù có nhiều tranh cãi về mốc thời gian này do trước năm 1886 thì đã có rất nhiều giải đấu không chính thức với nhiều quy mô lớn nhỏ khác nhau được tổ chức. Sau đó, khi hai cầu thủ hàng đầu ở châu Âu và Hoa Kỳ, lần lượt là Johann Zukertort và Wilhelm Steinitz thi đấu với nhau, do sự quảng cáo ở quy mô lớn về cuộc thi và sự nổi tiếng của hai nhân vật nên giải đấu năm 1886 được chú ý hơn cả và xem như giải vô địch đầu tiên trên Thế giới.

Từ năm 1888 đến năm 1948 giải đấu bị trì hoãn nhiều năm vì nhiều lí do khách quan cũng như chủ quan, cụ thể: do chiến tranh thế giới thứ nhất và thứ hai đã tạo khó khăn về ngoại giao cho một giải đấu tầm cỡ liên quốc gia. Ngoài ra, việc chưa xác định được quy luật chơi thống nhất, thể lệ thi đấu cũng chưa thực sự cụ thể, tài chính cho giải đấu là một vấn đề quan trọng khi chưa kiếm được nguồn tài trợ chính thức.

Có các cuộc thảo luận và thương phán nhưng không tìm được sự đồng tình và vì vậy giải đấu bị trì hoãn liên tục.

Từ năm 1948, qua nhiều nỗ lực đàm phán mà kết quả cuối cùng là giải đấu được quản lí bởi Liên đoàn cờ vua thế giới (Fédération Internationale des Échecs - FIDE). Sau đó, giải đấu được tổ chức với chu kì dày đặc hơn qua các năm 1951, 1954, 1957, 1958, 1960, 1961, 1963. Qua những năm tổ chức nhiều vấn đề cấp bách nổi lên. Giải đấu dường như bị chính trị hóa khiến nó không còn đơn thuần là một giải thi đấu tranh tài bổ ích nữa. Liên Xô lúc bấy giờ chiếm đa số ứng viên tham gia nên tỷ lệ vô địch cao hơn các quốc gia khác, điều đó bị coi là không công bằng nên nhiều nỗ lực thay đổi quy định giới hạn số lượng người tham gia của một quốc gia được triển khai. Điều này làm Liên Xô không hài lòng và Liên Xô cho rằng quy định đó sẽ làm giảm vị thế của Liên Xô

trong giải đấu. Nhiều vấn đề xảy ra xung quanh giải đấu liên quan đến Liên Xô và đỉnh điểm năm 1962, Bobby Fischer công khai cáo buộc rằng Liên Xô đã thông đồng để ngăn chặn bất kỳ người không thuộc Liên Xô nào chiến thắng, và Liên Xô sử dụng chiêu trò bẩn thỉu trong giải đấu. Từ đó dẫn đến việc thay đổi thể lệ chơi từ giải đấu vòng tròn được thay thế bằng các trận đấu loại trừ. Các giải đấu với thể lệ mới được tổ chức qua những năm 1966, 1969, 1972, 1978, 1981, 1984, 1985, 1986, 1987, 1990.

Năm 1993, nhà vô địch Garry Kasparov đã tách ra khỏi FIDE, điều này dẫn đến việc tạo ra chức vô địch PCA là đối thủ của giải WCC. Các danh hiệu đã được thống nhất tại Giải vô địch cờ vua thế giới 2006. Sau khi thống nhất có nhiều điều khoản được quy định lại, trong đó quy định rằng các trận đấu vô địch thế giới trở lại định dạng của trận đấu giữa nhà vô địch và người thách đấu. Các giải đấu với quy định này được tổ chức vào những năm 2008, 2010, 2012.

Kể từ năm 2013, các thí sinh tham gia giải phải thi đấu theo luật vòng tròn 8 người, với người chiến thắng chơi một trận đấu với nhà vô địch cho danh hiệu. Những điều này đã diễn ra theo chu kỳ 2 năm: vòng loại cho các thí sinh trong năm lẻ, giải đấu của thí sinh vào đầu năm chẵn và trận tranh đai vô địch thế giới vào cuối năm chẵn.

3. Thông tin giải đấu

Người vô địch và địa điểm tổ chức của các giải đấu gần nhất (từ 2006-nay)

Tên	Năm tổ chức	Quốc gia đăng cai
Anatoly Karpov	**1993–1999**	**Russia**
Alexander Khalifman	**1999–2000**	**Russia**
Viswanathan Anand	**2000–2002**	**India**
Ruslan Ponomariov	**2002–2004**	**Ukraine**
Rustam Kasimdzhanov	**2004–2005**	**Uzbekistan**
Veselin Topalov	**2005–2006**	**Bulgaria**
Vladimir Kramnik	**2006–2007**	**Russia**
Viswanathan Anand	**2007–2013**	**India**
Magnus Carlsen	**2013–present**	**Norway**

4. Thể lệ thi đấu của năm gần nhất (Giải vô địch năm 2018)

4.1. Đối tượng

Các ứng viên sẽ thi đấu để chọn ra người giỏi nhất, người đó được gọi là người thách đấu. Sau đó người thách đấu sẽ thi đấu với đương kim vô địch. Nếu người thách đấu thắng thì họ sẽ trở thành tân vô địch của giải. Nếu người thách đấu thua thì đương kim vô địch bảo vệ thành công ngôi vô địch.

4.2. Thời gian

Quy định thời gian cho mỗi bàn đấu là: 100 phút cho 40 lần di chuyển đầu tiên, 50 phút cho 20 lần di chuyển tiếp theo và sau đó 15 phút cho phần còn lại của trò chơi cộng thêm 30 giây cho mỗi lần di chuyển bắt đầu từ di chuyển 1.

Các trò chơi sẽ được chơi bằng đồng hồ và bảng điện tử được FIDE phê duyệt.

4.3. Điều kiện thắng

Giải đấu sẽ được chơi trong tối đa mười hai ván và người chiến thắng trong trận đấu sẽ là người chơi đầu tiên đạt được 6,5 điểm trở lên. Nếu người chiến thắng ghi được 6,5 điểm sau ít hơn 12 trận thì trận đấu sẽ kết thúc sớm. Nếu sau 12 ván mà có điểm hòa thì các ứng viên hòa nhau sẽ tiếp tục thêm 4 ván gọi là tie-break. Với thời gian 25 phút cho mỗi ứng viên và sau mỗi lần di chuyển quân cờ thời gian sẽ tăng thêm 10 giây cho ứng viên đó.

Nếu một người chơi từ chối tham gia Giải vô địch thế giới, họ sẽ được thay thế như sau: Người vào chung kết giải vô địch 2016 GM Serge Karjakin sẽ thay thế nhà vô địch thế giới Magnus Carlsen và á quân giải 2018 sẽ thay thế người thách đấu. Trong trường hợp bất kỳ hoặc cả hai người chơi từ chối tham gia khi được mời hoặc cho bất kỳ sự thay thế nào cần thiết, danh sách xếp hạng sẽ được họp bàn công bố sau.

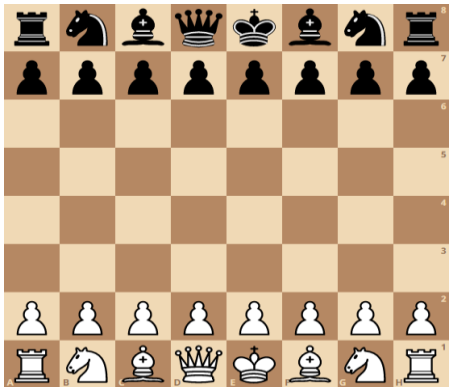
(Match, 2018)

5. Luật chơi [3]

Trò chơi này diễn ra trên một bảng hình vuông, gọi là bàn cờ, gồm 8 hàng (đánh số từ 1 đến 8) và 8 cột (đánh các chữ cái từ a đến h), tạo ra 64 ô hình vuông với các màu đậm và nhạt xen kẽ nhau, ô phải dưới và ô trái trên của bàn cờ vua sẽ luôn là màu nhạt. Mỗi người sẽ bắt đầu ván cờ với 16 quân cờ và sẽ lần lượt đi các quân của mình sau khi đối phương đã đi xong một nước (hoàn thành nước đi).

Các quân cờ của mỗi bên bao gồm: 8 Tốt, 2 Mã, 2 Tượng, 2 Xe, 1 Hậu và 1 Vua. Người cầm quân trắng luôn là người đi đầu tiên; người kia cầm quân đen.

Vị trí khai cuộc:



Cờ vua là một môn thể thao trí tuệ, sự thắng thua của trò chơi dựa vào chiến thuật cũng như chiến lược của trò chơi. Tuy nhiên một người chơi không thể bao quát hết tất cả các phương án xảy ra của bàn cờ bởi vì số lượng nước đi của nó là một số không thể đong đếm. Có rất nhiều biến thể được sinh ra từ cờ vua như cờ tướng (Trung Quốc, Việt Nam...), Shogi (Nhật Bản), Janggi (Triều Tiên).

Mục tiêu cuối cùng của người chơi là tìm cách tiêu diệt quân Vua (King) của đối phương càng sớm càng tốt. Để thực hiện được điều này người chơi phải nắm rõ được các nước đi của các quân cũng như luật chơi. Điều này được mô tả như sau:

Quân Trắng luôn luôn đi trước.

Quân Tốt – Pawn kí hiệu P

Quân Tốt trong cờ vua là quân thường xuyên bị ngó lơ nhất trong tất cả các quân cờ vua. Một phần vì quân tốt quá đơn giản và là quân cờ nhỏ nhất trong bộ cờ vua, sức mạnh ban đầu kém nhất và đứng ở hàng đầu. Tuy nhiên càng về sau quân tốt càng có giá trị, những người chơi giỏi rất biết tận dụng sức mạnh của quân này.

Quân Tốt là quân cờ đơn giản nhất và cũng phức tạp trong cách nó di chuyển.

Quân Tốt trong cờ vua có rất ít sự lựa chọn trong cách di chuyển. Chỉ có Tiến lên về phía trước cho đến khi nó đến cuối bàn cờ và được thăng hạng thành những quân cờ khác. Quân Tốt có cách di chuyển có thể mô tả ngắn gọn như sau:

Quân tốt chỉ có thể di chuyển về phía trước 1 ô và trong trường hợp là nước đi đầu tiên của nó thì có thể lựa chọn giữa việc đi 1 ô hoặc 2 ô.

Quân Tốt có thể tiến chéo để ăn quân đối thủ.

Khi quân Tốt tiến đến cuối bàn cờ bên kia, nó được phong cấp thành bất kỳ con nào khác nhưng không phải là vua.

Quân Xe – Rook kí hiệu R

Quân Xe có cách di chuyển đơn giản nhất trên bàn cờ vua.

Quân Xe có thể di chuyển thẳng theo phương tiến, lùi, sang ngang bất kỳ lúc nào.

Một lần di chuyển, quân Xe có thể thực hiện từ 1 đến 7 ô miễn là đường đi không bị cản bởi quân nào.

Sức mạnh của quân Xe vì vậy rất lớn trong trung cuộc và tàn cuộc, vì lúc đó số lượng quân của hai bên đã vơi bớt, ít cản đường xe hơn.

Trong cờ vua, quân Xe có thể di chuyển cùng lúc với quân Vua trong nước đi nhập thành (sẽ được đề cập trong phần sau).

Quân Mã – Knight kí hiệu N

Khi nói đến cờ vua, quân Mã thường được phân tích như là một quân có nhiều cơ hội sáng tạo nhất. Đây cũng là quân khó đoán nhất bàn cờ. Nhiều chiến thuật cờ vua cực kỳ độc đáo được tạo ra với quân Mã hạ gục đối phương thần sầu.

Cách di chuyển quân Mã:

Quân Mã có thể di chuyển về phía trước, phía sau, trái hoặc phải hai ô vuông và sau đó phải di chuyển một hình vuông theo hướng vuông góc. Nói cách khác, quân Mã di chuyển theo hình chữ L.

Quân Mã có thể đi đến bất kỳ vị trí không có quân cùng màu với nó.

Quân Mã có thể nhảy qua các quân khác.

Quân Mã thường được sử dụng trong vùng trung tâm bàn cờ vua vì nó là quân đầu tiên có thể tiến lên khu vực này.

Quân Mã có cách di chuyển khác các quân khác, vì vậy quân Mã lấp lại yếu điểm của quân khác.

Quân Tượng – Bishop

Quân Tượng là một trong những quân mạnh trong bộ cờ Vua. Ở khai cuộc cờ vua, sức mạnh của Tượng có thể bị kiềm hãm vì quân Tốt chặn đường đi khá nhiều, nhưng một khi đường đi được giải phóng, quân Tượng sẽ tỏa sáng tốt. Quân Tượng luôn xuất sắc đỡ cho quân Vua nhưng nhiều cờ thủ vẫn thà thí Tượng để giữ Xe hơn.

Cách di chuyển quân Tượng:

Quân Tượng di chuyển theo đường chéo theo bất kỳ hướng nào, miễn là không có quân cản trở.

Quân Tượng sẽ bị kẹt nếu nếu có quân cản trên đường đi của nó, khác với quân Mã có thể nhảy qua quân khác.

Quân Tượng có thể ăn bất kỳ quân nào trên bước đi.

Quân Hậu – Queen kí hiệu Q

Quân Hậu được xem là quân cờ vua nguy hiểm và linh hoạt nhất trên bàn cờ vua, vì vậy quân hậu rất quan trọng để giữ trong cờ vua bên cạnh quân Vua. Mất quân Vua coi như thua cờ, nhưng mất quân Hậu thì cũng … gần với thua cờ rồi trừ khi bạn là một tay cờ có nghề. Hầu hết mọi kỳ thủ đều sẵn lòng thí quân khác để cứu Hậu. Mỗi bên bắt đầu với một quân Hậu nhưng nếu di chuyển được quân Tốt đến cuối bàn cờ, thì nó sẽ biến thành quân hậu thứ hai và khi đó hầu như bạn sẽ nắm chắc lấy chiến thắng.

Cách di chuyển quân Hậu

Quân Hậu là tổng hợp cách di chuyển của 2 quân Xe và Tượng.

Quân Hậu có thể di chuyển theo đường chéo và đường thẳng bất kỳ đâu.

Quân Hậu không thể nhảy qua quân khác.

Quân Hậu có thể ăn bất kỳ quân nào của đối thủ.

Quân Hậu là một quân phòng thủ ưa thích của nhiều kỳ thủ cờ vua vì khả năng di chuyển và ăn quân siêu phàm của nó. Quân Hậu là một nhân tố không thể thiếu trong tàn cuộc cờ vua. Tuy nhiên, tốt hơn là vẫn đẩy Hậu vào trung tâm bàn cờ để phòng thủ được tốt hơn phần bàn cờ của mình. Hậu là quân bảo vệ của quân Vua. Tuy nhiên cần cẩn thận quân Mã của đối phương vì quân Hậu không có khả năng tấn công lại quân Mã khi bị nó tấn công.

Quân Vua – King kí hiệu K

Quân cuối cùng là quân Vua, đây là quân cờ chiến thắng, nghĩa là đoạt được quân này thì đối phương sẽ thắng bất kể tỷ số. Nhiều kỳ thủ kỳ cựu còn biết tận dụng quân vua để giành lợi thế so với đối thủ. Tuy vậy, quân Vua phải luôn được bảo vệ xuyên suốt ván cờ.

Cách di chuyển quân Vua:

Quân Vua có thể di chuyển bất kỳ hướng nào 1 ô.

Quân Vua không thể di chuyển đến quân cờ bên mình đang đứng.

Quân Vua không thể đi nước mà nó bị chiếu tướng

Quân Vua có thể dùng trong thế nhập thành là quân Vua có thể di chuyển 3 ô và đổi chỗ cho quân Xe.

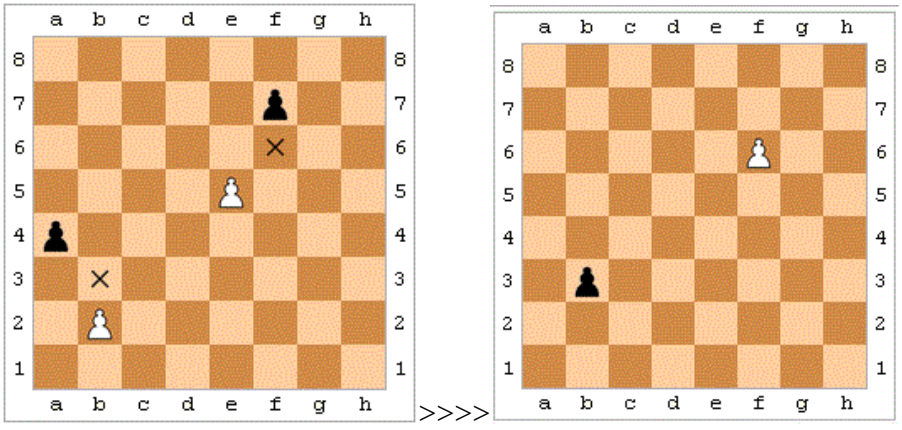
Kỳ thủ hay có thể dùng quân Vua để nhử mồi đặt bẫy rồi ăn quân đối thủ, nhưng vẫn phải dùng vua sao cho an toàn.

Thông thường, để giữ an toàn cho quân Vua, kỳ thủ dồn Vua vào các góc để ít bị tấn công bằng nhập thành.

Các nước đi đặc biệt trong cờ vua

1. Những nước đi đặc biệt của chốt (tốt):

- Nước ăn chốt qua đường: khi Tốt đối phương từ vị trí ban đầu tiến hai ô vượt qua ô cờ đang bị Tốt bên có lượt đi kiểm soát, thì Tốt bên có lượt đi có thể bắt Tốt đối phương vừa đi hai ô như khi Tốt đó thực hiện nước đi một ô. Nước bắt này chỉ có thể thực hiện ngay sau nước tiến Tốt hai ô và được gọi là "bắt Tốt qua đường". (Hình sau)



- Phong cấp: khi một Tốt tiến tới hàng ngang cuối cùng nó phải được đổi thành Hậu, hoặc Xe, hoặc Tượng, hoặc Mã cùng màu, ngay trong nước đi này. Sự lựa chọn để đổi quân của đấu thủ không phụ thuộc vào các quân đã bị bắt trước đó. Sự đổi Tốt thành một quân khác được gọi là "phong cấp" cho Tốt và quân mới có hiệu lực ngay lập tức.

2. Nước nhập thành:

Vua di chuyển ngang hai ô từ vị trí ban đầu sang phía Xe tham gia nhập thành, tiếp theo Xe nói trên di chuyển nhảy qua Vua tới ô cờ quân Vua vừa đi qua.

[1]. Không được phép nhập thành, nếu:

(a) Nếu Vua đã di chuyển rồi, hoặc

(b) Nếu Xe phía nhập thành đã di chuyển rồi.

[2]. Tạm thời chưa được nhập thành, nếu:

(a). Nếu ô Vua đang đứng, ô Vua phải đi qua hoặc ô Vua định tới đang bị một hay nhiều quân của đối phương tấn công, trấn giữ.

(b). Nếu giữa Vua và quân Xe định nhập thành có quân khác đang đứng.

(c). Quân Vua được coi là "bị chiếu" nếu như nó bị một hay nhiều quân của đối phương tấn công, thậm chí cả khi những quân này không thể tự di chuyển. Việc thông báo nước chiếu Vua là không bắt buộc.

CHƯƠNG II: NỘI DUNG ĐỒ ÁN

1. Mục tiêu

Qua quá trình học tập nhóm sinh viên muốn sử dụng kiến thức đã học để tạo nên một sản phẩm vừa để hoàn thành mục tiêu môn học vừa tạo nên nền tảng cho công việc và việc học sau này.

Đồ án giải quyết các vấn đề sau:

- Bài toán đặt ra
- Cách giải quyết
- Kết quả đạt được
- Tổng kết và đánh giá.

2. Chức năng

2.1. Chọn chế độ:

2.1.1 Một người chơi:

• Cờ thế.

• Đánh thường.

• Cờ chớp

2.1.2 Hai người chơi:

Tùy chỉnh (thời gian, thế cờ, số lần undo/redo…) ở các chế độ đánh thường hay cờ thế.

2.2. Cờ thế

• Người chơi được chỉ được chọn phe thắng trong thế cờ.

• Các thế cờ đã được lưu sẵn theo mã Fen

• Trong số bước đi đã được quy định (không quan tâm là đi như thế nào) nếu người chơi không kết thúc được ván đấu thì xử thua người chơi.

Lưu ý: số bước đi sẽ được báo trước ở mỗi ván đấu.

2.3. Undo, Redo

- Standard: được undo tối đa 3 lần. Mỗi lần 1 nước, sau khi undo đi nước khác thi không được redo.

- Custom: tùy chỉnh theo người chơi.

2.4. Tính giờ

- Cờ chớp: Cho mỗi bên số thời gian nhất định theo chuẩn quốc tế

2.5. Ghi biên bản trận đấu

- Trong trận: khung hiện thị các nước đã đi của cả 2 bên.

- Khi người chơi bấm save game: lưu các nước đã đi thành một file định dạng *.PGN.

2.6. Reset bàn cờ:

- Load game lại từ đầu.

3. Ngôn ngữ và công nghệ sẽ sử dụng

3.1. Ngôn ngữ: sử dụng C# [5]

3.2. Giao diện:

• Bàn cờ theo chuẩn: cột A -> H, hàng 1 -> 8

• Sử dụng winform / WPF

3.3. Các thuật toán sẽ dùng để làm AI:

• Alpha-Beta search with Pricipal-variation (PVS)

• Minimax

• Iterative Deepening

3.4. Các thuật toán có thể dùng thêm:

• Hashtable (transition table) using Zobrist Keys. (Separate Pawn & Check hashtables)

• Null-move forward pruning (verified and non-verified)

• 0x88 board representation

• Move ordering using Hash table and History Heuristic

- Quiescience Search (MVV/LVA and SEE)
- Extensions (Check, Re-Capture, Pawn-promotion)
- Futility pruning (Standard & Extended)
- "n moves in x minutes" Move time-allocation algorithm

4. Tổ chức

4.1. Cơ sở dữ liệu:

- Thông tin trạng thái bàn cờ.
- Thông tin các quân cờ.
- Tạo các nước đi hợp lệ.
- Kiểm tra thắng thua.

4.2. Giao diện:

- Thiết kế bàn cờ.
- Tìm kiếm các hình ảnh đại diện cho quân cờ.
- Các khung chức năng khác: thời gian của người chơi, khung lưu lịch sử các nước đi, các nút chức năng new game undo redo, …

5. Giải quyết vấn đề:

5.1. Lập trình trên WPF bằng ngôn ngữ C#:

- Thực hiện song song phần cấu trúc dữ liệu và thiết kế giao diện.
- Làm chức năng 2 người chơi trước.

5.2. Cấu trúc dữ liệu

5.2.1 Quân cờ:

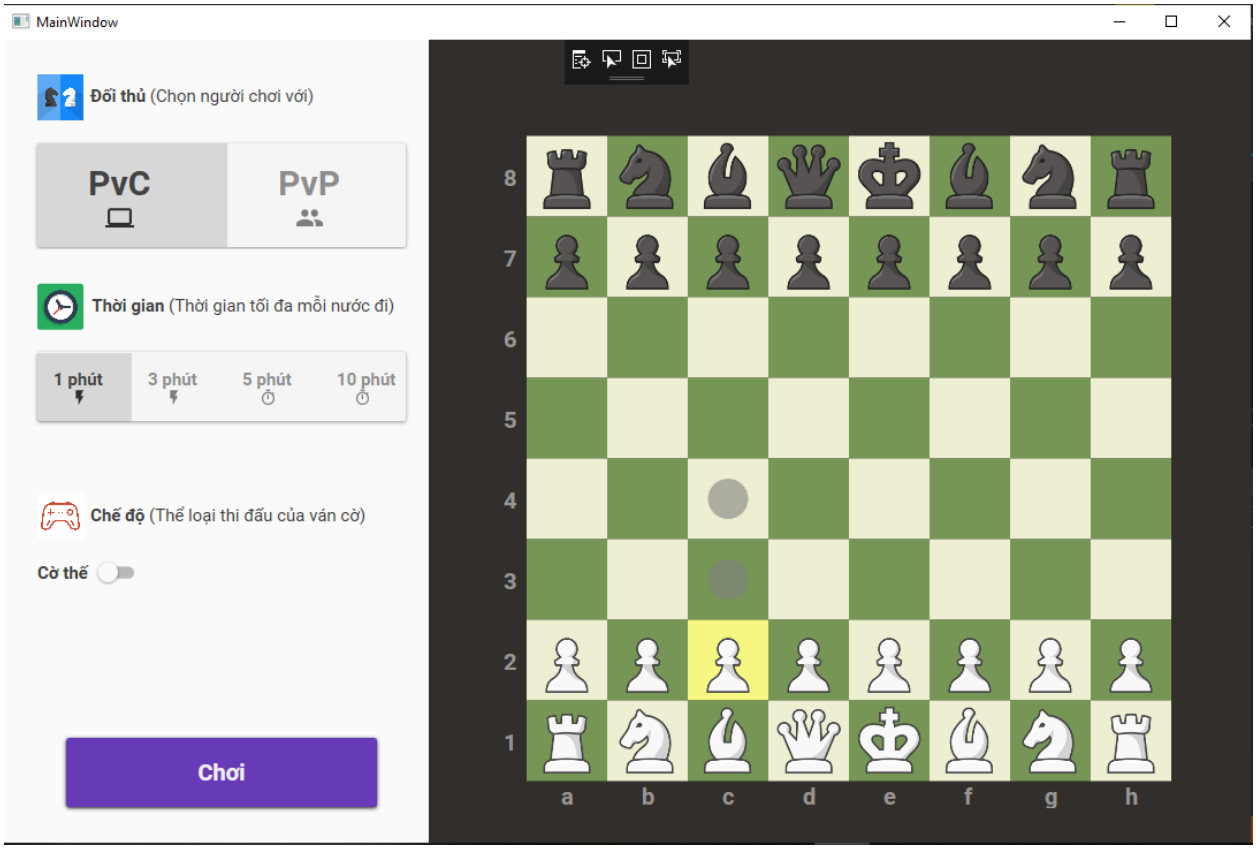
- Tạo một class Piece đại diện cho các quân cờ, gồm các thuộc tính:
- Color: màu quân cờ, 0 là đen, 1 là trắng, -1 là ô trống.
  - Type: loại quân cờ, 0 là ô trống, 1-> 6 lần lượt là pawn, knight, bishop, rook, queen, king.
  - Int x, int y: tọa độ của quân cờ trên bàn cờ.

– Res: là image hình ảnh hiển thị của quân cờ trên bàn cờ

– Mỗi quân cờ sẽ là một class riêng kế thừa class Pieces, và có một số thuộc tính riêng của quân cờ đó, và phương thức sinh nước đi từ trạng thái hiện tại.

5.2.2 Lưu trạng thái bàn cờ và sinh nước đi cho quân cờ:

- Dùng một mảng 2 chiều 64 phần tử Piecies (List<List<Piece>> chess_mtr;) để biểu diện trạng thái bàn cờ hiện tại.
- Khi người chơi click vào Image của quân cờ trên bàn cờ, chương trình sẽ kiểm tra đối tượng được click có phải turn của người chơi hiện tại không, sau đó sẽ chạy phương thức sinh nước đi và vẽ các ô tròn tượng trưng cho các nước có thể đi.

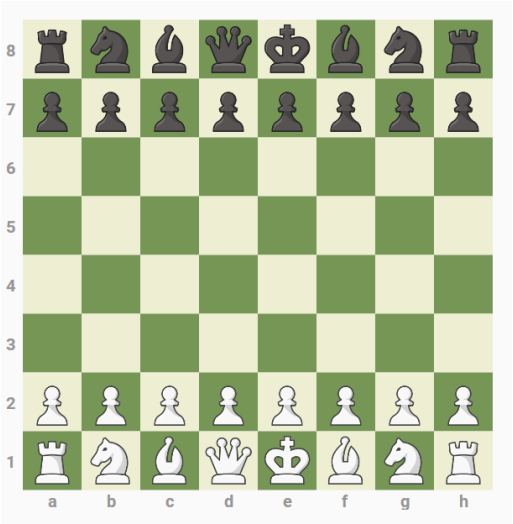


5.2.3 Giao diện:

- Mỗi ô trên bàn cờ là một rectangle.
- Mỗi quân cờ sẽ được hiển thị bằng một image tương ứng.
- Các thao tác trên quân cờ sẽ được thực hiện trực tiếp trên image.

5.2.4 Ưu điểm và khuyết điểm:

- Ưu điểm:
  - Giao diện đồ họa:
    + WPF cung cấp nhiều tính năng lập trình giao diện trong cùng một công nghệ đơn nhất, điều này giúp cho quá trình tạo giao diện người dùng trở nên dễ dàng hơn đáng kể.
    + WPF cung cấp nhiều tính năng đồ họa giao diện rất đẹp mặt, và có nhiều thư viện đồ họa sẵn rất dễ tái sử dụng.



Hình trên là giao diện bàn cờ chúng em dùng WPF và thư viện Material Design In XAML ver 2.5.1, nhìn chung rất đẹp và bắt mắt.

  - Bộ nhớ

+ Việc sử dụng cấu trúc dữ liệu trên mảng 2 chiều giúp chương trình tiết kiệm rất nhiều bộ nhớ, chỉ tốn 1 mảng 8x8 và 1 số biến phụ xuyên suốt quá trình chạy. Điều này giúp chương trình khi chạy sẽ ít ngốn RAM hơn.

- Nhược điểm
  - WPF tuy là một GUI framework hỗ trợ rất mạnh về đồ họa nhưng lại có một điểm yếu là không thể đa nền tảng.
  - WPF là một nền tảng cho phép developer có thể tạo ra các ứng dụng trên nền .NET framework cho Windows nói chung vì thế các sản phẩm làm ra chỉ có thể chạy trên Windows, dẫn đến các hệ điều hành như Linux, Mac, ...

- Tốc độ
  - Việc cấu trúc dữ liệu trên mảng 2 chiều rất chương trình tiết kiệm nhiều bộ nhớ, việc truy vấn dữ liệu sẽ nhanh hơn nhưng nó lại tạo ra nhiều nhược điểm khác.
  - Với cấu trúc dữ liệu này, ta chỉ có thể thực hiện sinh nước đi cho các quân Xe, Tượng, Hậu bằng các vòng for thông thường, điều này làm tăng độ phức tạp của chương trình.
  - Dẫn đến hệ lụy nghiêm trọng cho việc AI, như ví dụ trên trang jsfidlle chess, với độ sâu bằng 5 (mặc dù có hàm heuristic) chương trình mất 43.23s để xử lý cho những nước đi đầu tiên. Đây là vấn đề nghiêm trọng cần được thay đổi, khắc phục.

5.2.5 Cách giải quyết

- Đa nền tảng
  - Qt là một khung ứng dụng đa nền tảng và bộ công cụ tiện ích để tạo giao diện người dùng đồ họa cổ điển và nhúng, và các ứng dụng chạy trên nhiều nền tảng phần mềm và phần cứng khác nhau hoặc ít thay đổi trong codebase cơ bản.
  - Thế nên chúng em tận dụng Qt (một trong các bộ công cụ thầy giới thiệu) để giúp cho chương trình có thể đa nền tảng, ngoài ra Qt sử dụng ngôn ngữ C++ và đa số các thành viên trong nhóm đều được học về ngôn ngữ này nên việc tiếp cận sẽ dễ dàng và nhanh chóng.
- Tốc độ
  - Phương pháp cấu trúc dữ liệu bằng mảng 2 chiều và sinh nước đi bình thường dường như không phù hợp để chạy AI nên chúng em quyết định đổi mới.
  - Sau khi nghiên cứu và tìm hiểu thì thấy phương pháp cấu trúc dữ liệu bitboard và phương pháp sinh nước đi Magic bitboard rất phù hợp để cải thiện tốc độ. Hai phương pháp này tuy làm tăng bộ nhớ cần lưu trữ (tăng không đáng kể) nhưng bù lại cải thiện tốc độ rất nhanh so với phương pháp cũ.

6. Phân chia công việc

Số thứ tự	**Chủ đề**	**Công việc cụ thể**	**Thành viên thực hiện**
1	Qt	Tìm hiểu công cụ	Cả nhóm
2	Cơ sở lý thuyết	Tìm hiểu các giải thuật và phương pháp liên quan đến AI chess	Cả nhóm
3	GUI	Thiết kế giao diện	Dương
		Dùng material design	
4	Engine	Xây dựng cấu trúc bàn cờ, các hàm tạo nước đi	Sử dụng code Shallow Blue của Rhys Rustad-Elliott
5	AI	Xây dựng thuật toán alpha-beta pruning	Dương, Tài
6	Tính năng	Chế độ chơi	Thành, Luân
		Undo, redo	Đạt, Thành
		Ghi biên bản	Luân, Tài
		Time	Đạt, Dương
		Cờ thế	Tài, Luân
7	Logic game	Promotion	Đạt, Thành, Dương
		Chiếu tướng	
		Endgame	

Bảng phân chia công việc

Thành viên	Tỷ lệ công việc %
Tài	30
Dương	25
Thành	15
Đạt	15
Luân	15

Bảng tỷ lệ công việc

Làm việc nhóm chia rất hiệu quả, bảng trên không phản ánh được gì đâu thầy, cứ cho cao điểm hết đi thầy!

CHƯƠNG III: CƠ SỞ LÝ THUYẾT

1. Tìm hiểu Qt [6]

Qt là một Application Framework. Mục tiêu của các nhà phát triển nên Qt chính là tạo ra một framework có khả năng thiết kế những phần mềm có thể chạy trên nhiều nền tảng phần mềm lẫn phần cứng khác nhau mà không phải thay đổi nhiều về code. Qt không chỉ là thứ giúp bạn viết giao diện cho phần mềm của mình, nó có đầy đủ các khía cạnh để tạo nên một phần mềm hoàn chỉnh ở nhiều góc độ, cho dù phần mềm đó có giao diện hay không. Bạn có thể dùng Qt viết ra những phần mềm chạy bằng dòng lệnh, hoặc là các ứng dụng console chạy trên server, thậm chí là các web framework,….

Lịch sử các công ty phát triển Qt:

- Trolltech (1991 – 2008)
- Nokia (2008 – 2011)
- Digia (2012 – 2014)
- Qt Project (2011 – nay) cùng phát triển bản Qt mã nguồn mở, được Nokia thành lập, sau này nhân sự và công nghệ về mảng này đều được Digia mua lại và quản lý
- Hiện tại Qt Company đã được thành lập (Digia + Qt Project) để thống nhất phát triển cho Qt..

Một số nền tảng mà Qt hỗ trợ

- Windows
- Linux
- OS X
- Android
- iOS
- WinRT (Windows 8/8.1 và Windows Phone 8/8.1)
- Blackberry 10

Sắp tới sẽ hỗ trợ thêm Tizen, hệ điều hành dựa trên Linux cho nhiều loại thiết bị, OS phát triển bởi Linux Foundation, Samsung, Intel, LG, Vondafone,… (các thành viên của Tizen Association)

Các hệ điều hành nhúng như: Android/Linux/Windows Embedded

2. Board representation

Sử dụng phương pháp cấu trúc biboard để thực hiện

Bitboard là một cấu trúc dữ liệu được dùng rất phổ biến trong game cờ vua. Bitboard là một con số gồm có 64 bits với mỗi bit đại diện cho một vị trí/ ô trên bàn cờ. Biểu diễn bàn cờ bằng bitboard giúp tăng tốc độ tính toán khi sinh nước đi cho quân cờ hơn.

Ví dụ biểu diễn quân xe trên bàn cờ:

	A	B	C	D	E	F	G	H
8	R							R
7								
6								
5								
4								
3								
2								
1	R							R

	A	B	C	D	E	F	G	H
8	1	0	0	0	0	0	0	1
7	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	1

	A	B	C	D	E	F	G	H
8	56	57	58	59	60	61	62	63
7	48	49	50	51	52	53	54	55
6	40	41	42	43	44	45	46	47
5	32	33	34	35	36	37	38	39
4	24	25	26	27	28	29	30	31
3	16	17	18	19	20	21	22	23
2	8	9	10	11	12	13	14	15
1	0	1	2	3	4	5	6	7

Với thao tác bật những bit ở những vị trí có quân xe đứng ta được dãy 64 bits, bit có trọng số nhỏ nhất sẽ ứng với ô A1 và bits có trọng số lớn nhất sẽ ứng với ô H8:

10000001 00000000 00000000 00000000 00000000 00000000 00000000 10000001

3. Move generation [7]

Sử dụng phương pháp magic bitboard để thực hiện

3.1. Định nghĩa [8]

Bản chất magic bitboard là một thuật toán băm cộng với sử dụng phép dịch bit để tạo và lưu nước đi cho các quân cờ. Cùng với việc dùng phép nhân bit, đây là một giải thuật giúp tăng rất đáng kể tốc độ duyệt nước đi (chuyện rất quan trọng trong quá trình ứng dụng AI).

Được đánh giá là giải thuật cốt lõi trong engine tiêu chuẩn của bitboard thời điểm hiện tại.

Cách thức hoạt động

Gồm 4 bước chính:

Đánh dấu các bit “chiếm chỗ” có liên quan với vị trị và quân cờ đang xét

Vd: Quân xe ở ô A1 – Các ô từ A2-A7 và B1-G1 có quân cờ khác sẽ được đánh dấu lại

```
blockers              RAYS[NORTH_WEST][c3]    maskedBlockers
. 1 . 1 . . . 1       . . . . . . . 1         . . . . . . . 1
. . . . . . . .       . . . . . . 1 .         . . . . . . . .
1 . . . 1 1 1 .       . . . . . 1 . .         . . . . . 1 . .
. . . . . . . .       . . . . 1 . . .         . . . . . . . .
. . . . . . . .   &   . . . 1 . . . .     =   . . . . . . . .
. . B . . . . .       . . B . . . . .         . . . . . . . .
1 . . 1 . . . .       . . . . . . . .         . . . . . . . .
. . . . . . . .       . . . . . . . .         . . . . . . . .
```

2. Nhân giá trị bit đánh dấu này với một con số gọi là “magic number” (Về phần con số này sẽ được nói sâu hơn ở phần sau) để tạo ánh xạ chỉ mục

3. Thực hiện dời bit phải 64-n bit. Con số “magic number” tìm được càng tốt thì giá trị dời càng lớn, hay nói cách khác là thu nhỏ được không gian lưu trữ cho trạng thái.

4. Sử dụng giá trị lưu để truy xét giá trị trước đó (duyệt AI,…)

```
relevant occupancy                    combination of
bishop b1, 5 bits                     the masked bits
. . . . . .     . . . . . .           . . .[C D E F G]
. . . . . .     . 1 . . . .           . . . . . .
. . . . . G .   . 1 . . . .           . . . . . .
. . . . . F . . . 1 . . . .           . . . . . .
. . . . E . . * . 1 . . . . =         . . garbage . .    >> (64- 5)
. . . D . . . .   . 1 . . . .         . . . . . .
. . C . . . . .   . . . . . .         . . . . . .
. . . . . . . .   . . . . . .         . . . . . .

                                      any consecutive
relevant occupancy                    combination of
bishop d4, 9 bits                     the masked bits
. . . . . .     . . . . . .           2 4 5 B C E F G]
. . . . . G .   . . .some . . .       . . . . . .[1
. 5 . . . F . . . . . . . . .         . . . . . .
. . 4 . E . . . . . .magic. . .       . . . . . .
. . . . . . . * . . . . . . . =       . . garbage . .    >> (64- 9)
. . C . 2 . . . . .bits . . .         . . . . . .
. B . . . 1 . . . . . . . . .         . . . . . .
. . . . . . . .   . . . . . .         . . . . . .

                                      any consecutive
relevant occupancy                    combination of
rook d4, 10 bits                      the masked bits
. . . . . . . .   . . . . . . . .     4 5 6 B C E F G]
. . . 6 . . . .   . . .some . . .     . . . . . .[1 2
. . . 5 . . . .   . . . . . . . .     . . . . . .
. . . 4 . . . .   . . .magic. . .     . . . . . .
. B C . E F G . * . . . . . . . =     . . garbage . .    >> (64-10)
. . . 2 . . . .   . . .bits . . .     . . . . . .
. . . 1 . . . .   . . . . . . . .     . . . . . .
. . . . . . . .   . . . . . . . .     . . . . . .

                                      any consecutive
relevant occupancy                    combination of
rook a1, 12 bits                      the masked bits
. . . . . . . .   . . . . . . . .     5 6 B C D E F G]
6 . . . . . . .   . . .some . . .     . . . .[1 2 3 4
5 . . . . . . .   . . . . . . . .     . . . . . .
4 . . . . . . .   . . .magic. . .     . . . . . .
3 . . . . . . . * . . . . . . . =     . . garbage . .    >> (64-12)
2 . . . . . . .   . . .bits . . .     . . . . . .
1 . . . . . . .   . . . . . . . .     . . . . . .
. B C D E F G .   . . . . . . . .     . . . . . .
```

3.2. Ưu điểm

Nhanh chóng tìm ra các nước đi có thể từ một vị trí quân cờ bất kì mà không phải sử dụng số lượng lớn bộ nhớ để lưu trữ một mảng các bit tấn công có kích thước khoảng 147.57 exabyte. Không hợp lí để kết hợp thêm AI khi có thêm độ xâu làm cho việc sinh ra các nước đi rất chậm không đáp ứng được yêu cầu. Bằng cách sử dụng hằng số “magic number”. Sử dụng phép và giữa 2 bảng bit nước đi và bảng bit các nước đi bị chặn sẽ ra được maskedblocker. Ở đây maskedblocker được xem như key để ánh xạ đến tập bit các nước đi(đã được tạo đầy đủ từ trước) bằng cách nhân key với hằng số trên sẽ ra được các nước đi có thể từ vị trí được chọn.

3.3. Cách cài đặt [9]

• Phương pháp magic bitboard sử dụng tính toán bằng xử lý bit nên việc cài đặt rất khó khăn. Chúng em quyết định sẽ thảm khảo code của những người đi trước.

• Chúng em quyết định tham khảo code Shallow Blue của ông Rhys Rustad – Elliott khi thực hiện code này ông là sinh viên năm 3 của trường đại học University of Toronto.

• Trong đó chúng em tham khảo các class attack, bitboard, move, move_gen, defs, bitultis mục đích chung tạo cấu trúc dữ liệu bitboard và phương pháp sinh nước đi magic biboard.

4. Giải thuật Minimax [10]

4.1. Định nghĩa

Minimax là giải thuật là một thuật toán đệ quy lựa chọn bước đi kế tiếp trong một trò chơi có hai người bằng cách định giá trị cho các Node trên cây trò chơi sau đó tìm Node có giá trị phù hợp để đi bước tiếp theo.

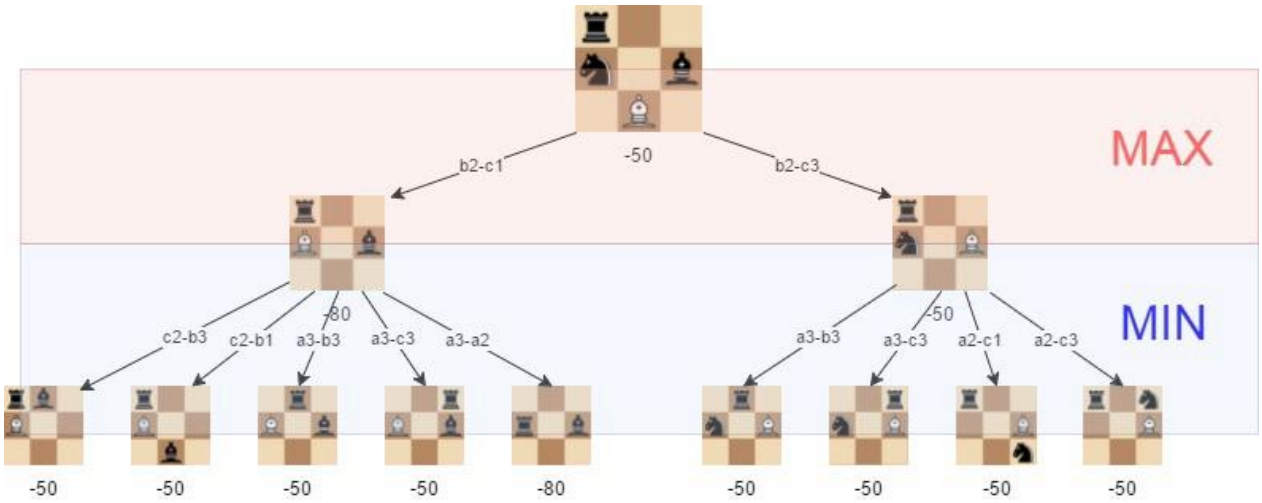
Có rất nhiều thuật toán tìm kiếm để làm AI trong game như A, Heuristic… Mỗi thuật toán thì sẽ phù hợp với từng loại game cho nó. Những game đối kháng trong đối người chơi luân phiên đánh như cờ vua, cờ tường, caro… Khi chơi bạn có thể khai triển hết không gian trạng thái nhưng khó khăn chủ yếu là bạn phải tính toán được phản ứng và nước đi của đối thủ mình như thế nào? Cách xử lý đơn giản là bạn giả sử đối thủ của bạn cũng sử dụng kiến thức về không gian trạng thái giống bạn. Giải thuật Minimax áp dụng giả thuyết này để tìm kiếm không gian trạng thái của trò chơi. Trường hợp này thuật toán minimax sẽ đáp ứng những gì mình cần.

4.2. Mã giả [11]

```
function minimax(board, depth,isMaximizingPlayer):
   if(CheckStateGame(curMove) == WIN_GAME)
      return MAX - depth
   if(CheckStateGame(curMove) == LOSE_GAME)
       return MIN + depth
   if( CheckStateGame(curMove) == DRAW_GAME)
      return DRAW_VALUE
    if isMaximizingPlayer :
        bestVal = -INFINITY
        for each move in board :
           value = minimax(board, depth +1, false)
           bestVal = max( bestVal, value)
       return bestVal

    else :
       bestVal = +INFINITY
       for each move in board :
            value = minimax(board, depth + 1,true)
            bestVal = min( bestVal, value)
        return bestVal
```

4.3. Minh họa [12]



5. Giải thuật Alpha-beta pruning

Giải thuật này được sử dụng chính trong AI

5.1. Định nghĩa [13]

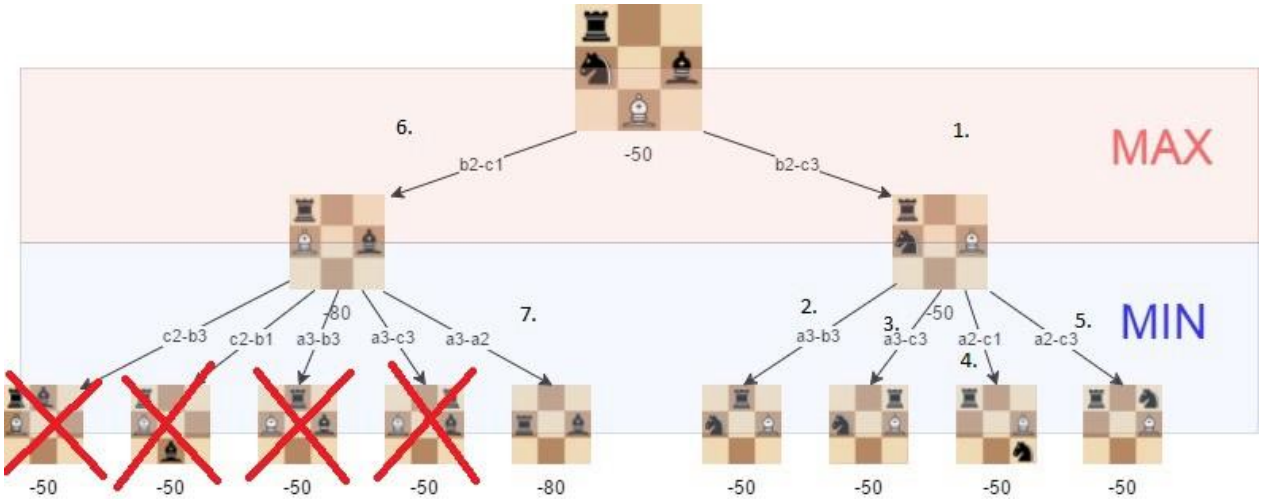
Cắt tỉa Alpha Alpha beta là một thuật toán tìm kiếm nhằm giảm số lượng nút được đánh giá bởi thuật toán minimax trong cây tìm kiếm của nó. Nó là một thuật toán tìm kiếm nghịch cảnh

thường được sử dụng để chơi máy hai trò chơi (Tic-tac-toe , Chess , Go , v.v.). Nó dừng đánh giá một động thái khi có ít nhất một khả năng đã được tìm thấy chứng tỏ động thái đó tồi tệ hơn một động thái được kiểm tra trước đó. Những động thái như vậy không cần phải được đánh giá thêm. Khi được áp dụng cho cây minimax tiêu chuẩn, nó sẽ trả về động thái tương tự như minimax, nhưng cắt bớt các nhánh không thể ảnh hưởng đến quyết định cuối cùng

5.2. Mã giả [14]

```
function alphabeta(node, depth, α, β, maximizingPlayer) is
    if depth = 0 or node is a terminal node then
        return the heuristic value of node
    if maximizingPlayer then
        value := −∞
        for each child of node do
            value := max(value, alphabeta(child, depth − 1, α, β, FALSE))
            α := max(α, value)
            if α ≥ β then
                break (* β cut-off *)
        return value
    else
        value := +∞
        for each child of node do
            value := min(value, alphabeta(child, depth − 1, α, β, TRUE))
            β := min(β, value)
            if α ≥ β then
                break (* α cut-off *)
        return value
```

5.3. Minh họa [15]



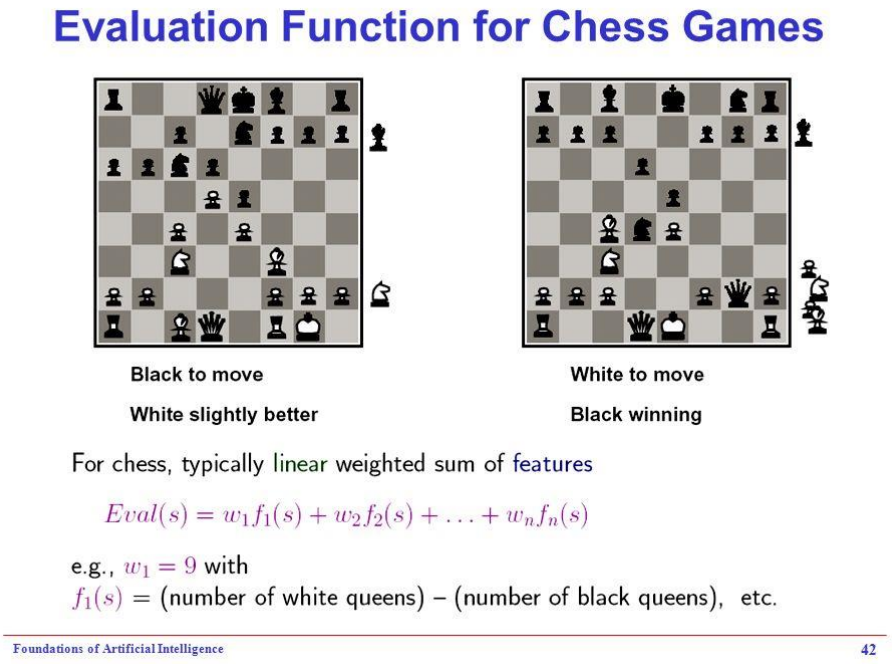
6. Board evaluation

6.1. Định nghĩa [16]

Là một chức năng heuristic để xác định giá trị tương đối của một trạng thái bàn cờ , tức là cơ hội chiến thắng của phe nào đó. Nếu chúng ta có thể thấy đến cuối trò chơi ở mọi dòng, đánh giá sẽ chỉ có các giá trị -1 (thua), 0 (hòa) và 1 (thắng). Tuy nhiên, trên thực tế, chúng tôi không biết giá trị chính xác của một trạng thái, vì vậy chúng tôi phải thực hiện xấp xỉ. Người chơi cờ bắt đầu học cách làm điều này bắt đầu với giá trị của các quân cờ . Các chức năng đánh giá máy tính cũng sử dụng giá trị của cân bằng vật chất làm khía cạnh quan trọng nhất và sau đó thêm các cân nhắc khác.

Hoạt động [17]

Board evaluation sẽ tính toán một trạng thái bàn cờ bằng cách cộng các điểm giá trị quân cờ với các điểm tọa độ quân cờ để cho ra điểm của mỗi bên, sau đó kết quả sẽ là tích của điểm 2 phe.



Ví dụ minh họa board evaluation

6.2. Ứng dụng

Ta sẽ dùng board evaluation như một hàm heuristic cho giải thuật minimax/ alpha beta pruning để khi giải thuật search đến depth = 0 nếu không có kết quả thì sẽ trả về giá trị này để thay thế.

Kết quả này giúp AI có thể thông minh hơn trong việc di chuyển, hạn chế các trường hợp lặp lại như 1 quân cờ đi qua đi lại ví trí cũ...

6.3. Cài đặt

Chúng em sử dụng lại class psquaretable của Shallow Blue để thao tác tính toán các giá trị trên cấu trúc dữ liệu bitboard, nhưng các giá trị quân cờ và tọa độ sẽ thay đổi.

Về giá trị quân cờ [18], chúng em dùng giá trị trên trang chessprogaming vì đã được khảo sát đánh giá là rất hiệu quả

```
P = 100
N = 320
B = 330
R = 500
Q = 900
K = 20000
```

Về giá trị tọa độ [19], chúng em dùng giá trị của phần mềm cờ vua trên web mà thầy đã demo. Nhưng để phù hợp với giá trị quân cờ nên chúng em x10 vừa đảm bảo phù hợp vừa không phải dùng số thực.

```
[ -3.0, -4.0, -4.0, -5.0, -5.0, -4.0, -4.0, -3.0],
[ -3.0, -4.0, -4.0, -5.0, -5.0, -4.0, -4.0, -3.0],
[ -3.0, -4.0, -4.0, -5.0, -5.0, -4.0, -4.0, -3.0],
[ -3.0, -4.0, -4.0, -5.0, -5.0, -4.0, -4.0, -3.0],
[ -2.0, -3.0, -3.0, -4.0, -4.0, -3.0, -3.0, -2.0],
[ -1.0, -2.0, -2.0, -2.0, -2.0, -2.0, -2.0, -1.0],
[  2.0,  2.0,  0.0,  0.0,  0.0,  0.0,  2.0,  2.0 ],
[  2.0,  3.0,  1.0,  0.0,  0.0,  1.0,  3.0,  2.0 ]
```

```
[ -2.0, -1.0, -1.0, -0.5, -0.5, -1.0, -1.0, -2.0],
[ -1.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0, -1.0],
[ -1.0,  0.0,  0.5,  0.5,  0.5,  0.5,  0.0, -1.0],
[ -0.5,  0.0,  0.5,  0.5,  0.5,  0.5,  0.0, -0.5],
[  0.0,  0.0,  0.5,  0.5,  0.5,  0.5,  0.0, -0.5],
[ -1.0,  0.5,  0.5,  0.5,  0.5,  0.5,  0.0, -1.0],
[ -1.0,  0.0,  0.5,  0.0,  0.0,  0.0,  0.0, -1.0],
[ -2.0, -1.0, -1.0, -0.5, -0.5, -1.0, -1.0, -2.0]
```

```
[  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0],
[  0.5,  1.0,  1.0,  1.0,  1.0,  1.0,  1.0,  0.5],
[ -0.5,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0, -0.5],
[ -0.5,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0, -0.5],
[ -0.5,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0, -0.5],
[ -0.5,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0, -0.5],
[ -0.5,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0, -0.5],
[  0.0,   0.0, 0.0,  0.5,  0.5,  0.0,  0.0,  0.0]
```

```
[ -2.0, -1.0, -1.0, -1.0, -1.0, -1.0, -1.0, -2.0],
[ -1.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0, -1.0],
[ -1.0,  0.0,  0.5,  1.0,  1.0,  0.5,  0.0, -1.0],
[ -1.0,  0.5,  0.5,  1.0,  1.0,  0.5,  0.5, -1.0],
[ -1.0,  0.0,  1.0,  1.0,  1.0,  1.0,  0.0, -1.0],
[ -1.0,  1.0,  1.0,  1.0,  1.0,  1.0,  1.0, -1.0],
[ -1.0,  0.5,  0.0,  0.0,  0.0,  0.0,  0.5, -1.0],
[ -2.0, -1.0, -1.0, -1.0, -1.0, -1.0, -1.0, -2.0]
```

```
[-5.0, -4.0, -3.0, -3.0, -3.0, -3.0, -4.0, -5.0],
[-4.0, -2.0,  0.0,  0.0,  0.0,  0.0, -2.0, -4.0],
[-3.0,  0.0,  1.0,  1.5,  1.5,  1.0,  0.0, -3.0],
[-3.0,  0.5,  1.5,  2.0,  2.0,  1.5,  0.5, -3.0],
[-3.0,  0.0,  1.5,  2.0,  2.0,  1.5,  0.0, -3.0],
[-3.0,  0.5,  1.0,  1.5,  1.5,  1.0,  0.5, -3.0],
[-4.0, -2.0,  0.0,  0.5,  0.5,  0.0, -2.0, -4.0],
[-5.0, -4.0, -3.0, -3.0, -3.0, -3.0, -4.0, -5.0]
```

```
[0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0],
[5.0,  5.0,  5.0,  5.0,  5.0,  5.0,  5.0,  5.0],
[1.0,  1.0,  2.0,  3.0,  3.0,  2.0,  1.0,  1.0],
[0.5,  0.5,  1.0,  2.5,  2.5,  1.0,  0.5,  0.5],
[0.0,  0.0,  0.0,  2.0,  2.0,  0.0,  0.0,  0.0],
[0.5, -0.5, -1.0,  0.0,  0.0, -1.0, -0.5,  0.5],
[0.5,  1.0, 1.0,  -2.0, -2.0,  1.0,  1.0,  0.5],
[0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0]
```

CHƯƠNG IV: PHÂN TÍCH VÀ THỰC HIỆN

1. Phân tích và tổ chức

1.1. Ý tưởng

Tạo ra một bàn cờ vua 64 ô, mỗi ô được lưu dưới dạng bitboard, do bitboard là một con số với 64 bits nên sẽ đại diện được cho 64 ô của bàn cờ. Dùng giải toán magic bitboard để tính toán các bước đi cho quân cờ, với lợi thế duyệt nước đi nhanh nên là lựa chọn tốt nhất cho đồ án này.

Với nền tảng ngôn ngữ C++ đã có, nhóm kết hợp sử dụng với hỗ trợ giao diện Qt để tạo nên hình ảnh hiển thị cho game. Sử dụng Qt sẽ không thay đổi nhiều về cú pháp code và được hỗ trợ chạy trên nhiều hệ điều hành.

Có ba lựa chọn cho chế độ chơi. Chế độ PvP là chế độ chơi giữa người với người dùng cho việc giải trí giữa bạn bè hoặc người thân. Chế độ PvC là chế độ chơi người với máy, với phần máy sử dụng thuật toán minimax cắt tỉa alpha-beta để tạo nên "thông minh" cho nó. Chế độ Chall là chế độ cờ thế, nó giống với chế độ PvC nhưng khác biệt là tạo ra thế cờ tùy chọn, thường là thế cờ sắp kết thúc.

1.2. Thiết kế

1.2.1 Coding:

Sử dụng phương pháp lập trình hướng đối tượng. Code được chia làm các phần thể hiện riêng biệt :

1.2.2 Engine :

Def : định nghĩa các enum cho dự án ,ví dụ : màu sắc, loại quân cờ, giá trị của các dòng, cột trong bàn cờ, chế độ chơi.

Board: Biểu diễn một trạng thái của bàn cờ, kèm theo các phương thức để thao tác, ví dụ một số thao tác quan trọng, doMove(Move) để thực hiện nước đi và thay đổi trạng thái hiện tại thành trạng thái mới, kiểm tra chiếu tướng, setFen(string) để set trạng thái của bàn cờ bằng mã fen và nhiều phương thức tiện ích khác.

Attacks, rays: Hỗ trợ class board trong việc sinh nước đi bằng magic bitboard.

Bitutils: chứa các hàm giúp cho việc thao tác trên bit được thực hiện một cách dễ dàng, và nhanh chóng hơn.

Move: biểu diễn cho một nước đi, với đầy đủ các thông tin như cờ hiệu của nước đi đó, đi từ ô nào đến ô nào, có ăn quân cờ nào không, loại quân cờ được phong cấp, nước đi này là của loại quân cờ nào, cùng với các thao tác cần thiết khác.

Movegen: Chủ yếu dùng để sinh danh sách các nước đi hợp lệ của một trạng thái bàn cờ cụ thể một cách nhanh chóng, moveGen sử dụng các class actacks, rays, move, bitUtils để hỗ trợ trong việc sinh các nước đi hợp lệ một cách nhanh chóng

Psquaretable: Lưu các giá trị của quân cờ, các giá trị này phụ thuộc vào vị trí của quân cờ, loại quân cờ và gamephase, dùng trong việc đánh giá để cài đặt thuật toán alpha beta trong AI .

AI: Sử dụng giải thuật alpha-beta pruning kết hợp với đánh giá bằng psquaretable.

1.2.3 GUI

Tiles: Định nghĩa các công việc liên quan đến quân cờ (click lần 1, click lần 2, click được chấp nhận – hợp lệ, hiển thị quân cờ, kiểm tra chiếu, gọi AI, gọi engine,…)

Validation: Kiểm tra sinh nước đi hợp lệ

Material: Các công cụ nâng cấp giao diện (button, dialog, lib,…)

Game (tinh chỉnh giao diện người dùng): Bao gồm các nút chức năng và các hiển thị trên bàn cờ người chơi (khởi tạo bàn cờ, nút play, các nút đổi chế độ chơi,…)

Sử dụng Qt cho việc code giao diện để tận dụng tính đa nền tảng của nó.

2. Đặc tả kỹ thuật

2.1. Kho dữ liệu

Tất cả dữ liệu của sản phẩm đồ án đều do chúng em thu thập từ nhiều nguồn khác nhau, có cái tự design, chứ không copy hoàn toàn của 1 sản phẩm cờ vua nào khác.

2.1.1 Hình ảnh

Logo [20]

Logo do chúng em tự design trên trang web hỗ trợ online

Thắng Thua [21]

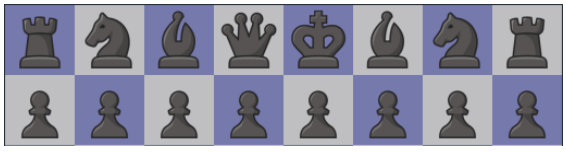
Hình ảnh thông báo thắng thua cũng tự design trên trang web hỗ trợ online

Chiếu tướng [22]

Hình ảnh quân vua bị chiếu được design bằng Paint 3D và quân vua thì lấy ở trang chess.com

Quân cờ [23]

Chúng em hình ảnh quân cờ của trang chess.com vì cảm thấy đồ họa rất đẹp dễ nhìn.



Thay wr.png bằng quân cờ tương ứng (ví dụ wr = white rook)

2.1.2 Âm thanh [24]

Âm thanh bắt đầu trận đấu [24a]

Âm thanh chiến thắng [24b]

Âm thanh thua cuộc [24c]

Âm thanh di chuyển [24d]

Âm thanh tấn công [24e]

2.2. Class defs

Mục đích: định nghĩa các enum cho dự án. Trong đó có định nghĩa màu sắc, loại quân cờ, giá trị các dòng, các cột trong bàn cờ, chế độ chơi.

Hỗ trợ cho class: ai, attacks, bitutils, board, move, movegen, psquaretable.

2.3. Class bitutils

Mục đích: giúp cho việc thao tác trên bit dễ dàng, các class board, move, moveGen, attack đều cần dùng

Nội dung: chứa các chức năng chính như là đảo bit, trả về vị trí gặp bit 1 đầu tiên, set các giá trị cho bit, di chuyển bit

Tác dụng: hỗ trợ các class board, move, movegen, attack trong việc sử lí bit

2.4. Class rays

Mục đích: giữ các chức năng giúp tạo nhanh các đường đi theo 8 hướng.

Tác dụng: hỗ trợ cho class attack tạo nhanh các mask (trong phần mô tả magic bitboard sẽ nói rõ hơn) tương ứng với từng ô trên bàn cờ. (sử dụng để sinh nước đi).

2.5. Class attacks

Mục đích: giúp cho việc tạo nước đi một cách nhanh chóng bằng cách tạo và lưu trữ sẵn các mask của các quân cờ của 64 ô trên bàn cờ, các giá trị magicbitboard tương ứng với từng quân cờ.

Hoạt động: các nước đi được sinh ra bằng kỹ thuật magic bitboard.

Tác dụng: hỗ trợ class board trong việc giảm thời gian sinh các nước đi.

Chức năng chính: trả về các danh sách các nước có thể đi của tất cả quân cờ trong tất cả trường hợp.

2.6. Class board

Mục đích: tạo ra bàn cờ và cung cấp các method liên quan tới bàn cờ như: doMove để thực hiện nước đi và thay đổi trạng thái bàn cờ (nhập thành, phong cấp, bắt cờ, bắt chốt qua sông,…)

Tác dụng: Vì class board cung cấp các phương thức liên quan tới bàn cờ nên nó áp dụng cho hầu hết các class khác để lưu, truyền hay thay đổi trạng thái bàn cờ.

2.7. Class move

Mục đích: Thực hiện các thao tác liên quan tới di chuyển quân cờ (lấy kí hiệu nước tới, nước vừa mới rời khỏi, lấy loại cờ ở ô hiện hành – dùng cho thao tác bắt cờ, định nghĩa loại cờ được phong cấp, thực hiện phong cấp cờ - chuyển từ chốt sang hâu, xe, tượng hoặc mã, lấy loại nước đi – nước đi phong cấp, bắt cờ,…

Tác dụng: Chủ yếu tương tác với nước đi. Hỗ trợ class movegen trong việc tìm nước đi hợp lệ, hỗ trợ class tile phong cấp, hỗ trợ class board cài lại bàn cờ,…

2.8. Class movegen

Mục đích: tạo ra danh sách các nước đi hợp lệ dựa đối với từng trạng thái cụ thể của bàn cờ.

Hoạt động: nhận vào một trạng thái bàn cờ cụ thể, tương ứng với từng quân cờ, và lượt đi, tạo và lưu các nước đi vào một danh sách để khi dùng chỉ cần gọi lại.

Tác dụng: hỗ trợ class tile trong việc lấy nhanh nước đi với ô cụ thể mà người chơi click, và class AI cho việc tìm kiếm nước đi tốt nhất nhanh chóng hơn.

2.9. Class psquaretable

Mục dích: class chứa hàm đánh giá board evaluation

Hoạt động: tính toán các giá trị trạng thái bàn cờ dựa vào tham số board truyền vào, kết quả về là lợi thế của 1 phe

Tác dụng: hỗ trợ hàm heuristic cho class ai để đánh khi depth = 0

2.10. Class ai

Mục đích: là phần chứa thuật toán tạo nên trí tuệ cho trò chơi. Cụ thể ở đây là thuật toán minimax có sử dụng cắt tỉa alpha-beta. Class này có tác dụng chính là tính toán để cho ra bước đi tốt nhất trong chế độ PvC hoặc Chall.

Hỗ trợ cho các class trong Engine của trò chơi: attacks, board, move, movegen, ray, psquaretable.

2.11. Class tile

Mục đích: tạo ra quân cờ và các phương thức liên quan tới quân cờ như: Hiển thị quân cờ, hiển thị ô phong cấp, thay đổi thông số các biến thông qua tương tác với quân cờ, kiểm tra chiếu, hiển thị chiếu,…

Tác dụng: Hỗ trợ class game trong việc hiển thị giao diện người dùng, đóng vai trò điều khiển, hiển và thiết lập thông qua các quân cờ. Đóng vai trò trong việc hiển thị giao diện nhận tín hiệu từ người dùng để engine xử lí và xuất lại kết quả thu được thông qua đối tượng chính của class là các quân cờ.

2.12. Class validation

Mục đích: class tạo ra để chứa các thao tác sinh nước đi có thể, tô màu các nước đi có thể

Tác dụng: hỗ trợ cho các tile để tạo nước đi và khai báo các biến extern để dùng cho toàn chương trình

2.13. Class game

Mục đích: class game là class chính chứa tất cả cả các class để tạo ra 1 game cờ hoàn chỉnh

Tác dụng: class game được gọi bởi hàm main để khởi động chương trình

Nội dung:

Chứa toàn bộ các biến toàn cục chạy xuyên suốt chương trình

Khởi tạo bàn cờ và quân cờ

Khởi tạo thời gian

Khởi tạo các nút tính năng

Chạy các sự kiện

3. Đặc tả chức năng

3.1. Giao diện

3.1.1 Tạo bàn cờ, quân cờ, thao tác trên quân cờ

Để tạo giao diện bàn cờ, chúng em sử dụng một Qlabel để hiện thị một ô trong bàn cờ, cho nên một bàn cờ cần 64 Qlabel. Đối với những ô có mà tại đó có quân cờ, thì Qlabel đó sẽ được set pixmap với hình quân cờ, và màu sắc tương ứng.

Để thao tác được trên các Qlabel, các thao tác ấy tượng trưng cho việc chọn và di chuyển các quân cờ hợp lệ, chúng em tạo một hàm gọi sẽ được gọi khi người chơi click chọn vào bất kì ô nào trên bàn cờ, hàm đó có tên là MousePressEvent();

Sau khi người chơi click chọn (lần click đầu tiên), hàm MousePressEvent() sẽ kiểm tra xem đối tượng được click là hợp lệ hay không dựa vào lượt chơi hiện tại, màu sắc quân cờ, tại vị trí đó có quân cờ hay là ô trống.

Sau khi kiểm tra các điều kiện nếu là hợp lệ thì hàm MousePressEvent() sẽ gọi tiếp hàm sinh nước đi hợp lệ từ vị trí đã được chọn và sau đó highlight các ô có thể đi để người chơi chọn.

Hàm sinh nước đi hợp lệ sẽ gọi class MoveGen để lấy danh sách các nước đi, và dựa vào danh sách đó để đổi nền background cho Qlabel (highlight) các ô có thể đi đến tương ứng.

Hình minh họa sau lần click đầu tiên

Ở lần click tiếp theo nếu người chơi click vào một trong các ô đã được highlight (các nước đi hợp lệ) thì hàm MousePressEvent() sẽ gọi hàm vẽ lại bàn cờ từ trạng thái hiện tại, và highlight các ô vừa di chuyển.

Hàm để vẽ lại bàn cờ từ trạng thái hiện tại là boardDisplay(Board), hàm này sẽ nhận vào trạng thái hiện tại và ứng với mỗi ô nếu có quân cờ sẽ set hình ảnh quân cờ với màu tương ứng cho ô đó.

Hình minh họa sau lần click thứ 2

3.1.2 Phong cấp

Ý tưởng:

Khi quân tốt lên đến hàng cuối cùng (hàng đầu tiên bên đối phương) của bàn cờ sẽ thực hiện phong cấp, người chơi sẽ được chọn 4 con cho việc phong cấp Hậu, Xe, Tượng và Mã.

Cài đặt:

Khởi tạo một mảng _promotion gồm 4 phần tử, mỗi phần tử là một button chứa hình ảnh của 4 quân cờ có thể phong cấp. Khi quân tốt đến hang cuối cùng sẽ kích hoạt hàm setPieceTypeforPromotion(). Trong hàm này sẽ khởi tạo một sender() nhằm mục đích lấy thao tác của chuột. Vòng lặp sẽ duyệt mảng _promotion để người chơi chọn quân cờ mình muốn phong cấp.

Nếu người chơi chưa chọn quân phong cấp thì chưa tính hết lượt của người chơi đó.

Sau khi người chơi chọn quân cờ phong cấp thì sẽ cập nhật lại bàn cờ với current-1. Giải thích cho vấn đề current-1 mà không phải là current ?! Giá trị current là giá trị sau khi đã thực hiện thao tác chọn phong cấp, chúng ta cần trừ current một đơn vị để trở về trạng thái trước bước chọn phong cấp. Giả sử con tốt phong hậu cần trải qua các giai đoạn sau: đến ô cuối -> chọn quân phong cấp -> biến thành quân đó, bây giờ chúng ta cần bỏ đi bước trung gian là chọn quân phong cấp nên chúng ta sẽ trừ giá trị current 1 đơn vị.

Cập nhật lại board và vẽ lại bàn cờ. Trả biến isFinishPromotion về true (biến thông báo việc phong cấp đã thành công). Nếu là chế độ PvC hoặc cờ thế thì tiếp tục chạy AI.

3.2. Chế độ

3.2.1 AI [25]

Ý tưởng :

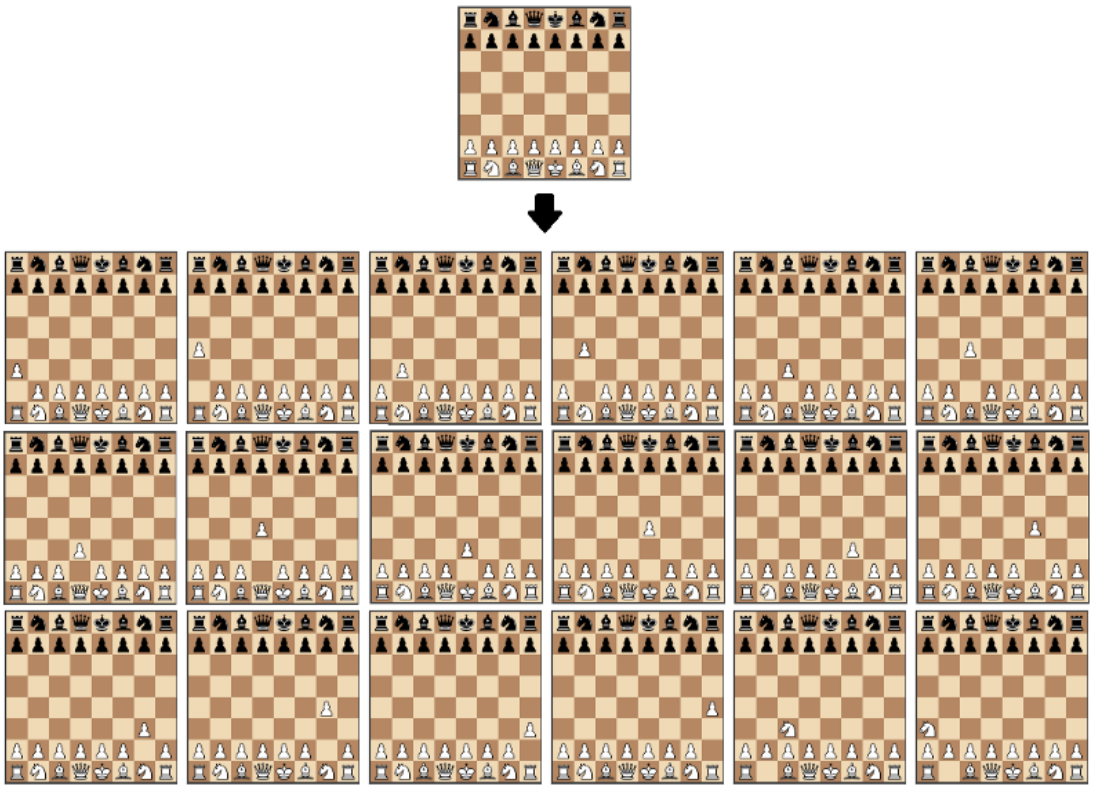
Khi người chơi chọn chế độ pvc hoặc challenge: hàm AI() sẽ được gọi sau khi người chơi thao tác di chuyển một quân cờ.

Hàm AI() sẽ tìm ra nước đi tốt nhất trong các nước đi có thể có của trạng thái bàn cờ hiện tại cho bên Đen, thuật toán tìm kiếm được sử dụng ở đây là thuật toán alpha – beta pruning.

Cài đặt:

Để xây dựng hàm AI() , chúng ta cần có sử dụng các class sau :

moveGen : dùng để sinh các nước đi tiếp theo.

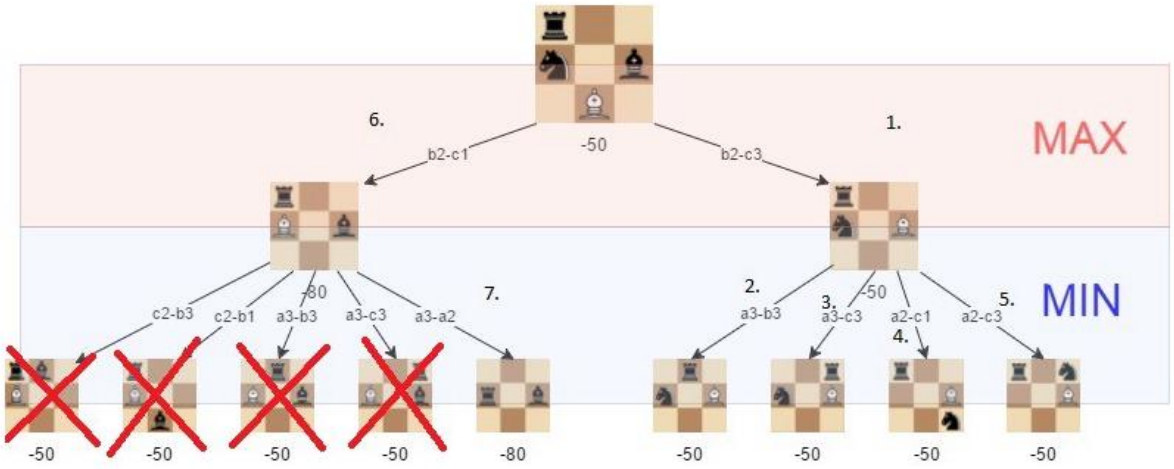


Hình minh họa chức năng của class moveGen cho quân trắng [26];

Psquaretable : dùng để ước lượng giá trị của trạng thái bàn cờ ứng với từng bên (đen và trắng)

Hàm minimax (sử dụng cắt tỉa alpha – beta) dùng để ước lượng giá trị của các nước có thể đi từ trạng thái hiện tại, hàm rootmax () : trả về nước đi có lợi nhất cho bên Đen.

Mô tả thuật toán minimax dùng alpha-beta để cắt tỉa: thuật toán này sẽ chạy đệ quy với tất cả các nước có thể có với độ sâu cho sẵn (người chơi có thể tùy chọn độ sâu của cây tìm kiếm), cây sẽ đệ quy đến khi nào độ sâu bằng 0 thì sẽ ước lượng giá trị của trạng thái tại nút lá đó, sau đó trả về giá trị ước lượng max hoặc min của nút con cho nút cha của nó, giá trị trả về là max hay min dựa vào đó là lượt đi của quân đen (max) hay quân trắng (min). các giá trị alpha beta sẽ giúp cho thuật toán này chạy nhanh hơn bằng cách dừng việc xét tiếp các nút con không cần thiết.



Hình minh họa cho thuật toán minimax-alphabeta [27]

Trạng thái bàn cờ sẽ được đánh giá như sau: mỗi quân cờ sẽ có trọng số khác nhau cụ thể: Pawn = 100, Knight = 320, Bishop = 330, Rook = 500, Queen = 900, King = 20000, tương ứng với từng vị trí cụ thể của từng quân cờ và gamephase (đầu game hay cuối game), mà quân cờ đó sẽ có giá trị thực bằng giá trị quân cờ mặc định + giá trị tại vị trí nó đang đứng (ví dụ quân hậu khi ở những ô ở giữa bàn cờ sẽ được cộng nhiều điểm hơn ở biên bàn cờ vì có nhiều nước để đi hơn).

Công việc ước lượng giá trị của trạng thái bàn cờ cụ thể sẽ được class psquaretable của tác giả shallow thực hiện, giá trị của tại nút lá trong cây tìm kiếm minimax -alphabeta sẽ giá trị của toàn bộ quân đen trừ đi giá trị của toàn bộ quân trắng tại trạng thái bàn cờ ở nút lá đó.

3.2.2 PVP

PvP là chức năng được thiết kế cho phép ta đấu cờ với một người khác. Trong project này, PvP là một nút chức năng được thiết kế khá đơn giản, sau khi bấm chọn vào nút (button) PvP trên giao diện, game sẽ được chuyển qua chế độ PvP việc tiếp theo đơn giản là bấm Play và đấu với bạn mình thôi

Về nội dụng nút (button) PvP, nút này sẽ khởi tạo lại các biến xác định mode chơi (tắt chức năng đấu máy, cờ thế và khởi động chức năng chơi với người) và lượt chơi sẽ được trả về cho bên trắng. Nút bấm trong Qt có một hàm gọi là connect, hàm này truyền vào đối tượng nút bấm (PvP), nếu nhận được tín hiệu bấm (click) chuột, tín hiệu đó sẽ được truyền gọi hàm pvp(), và trong hàm này các thông số về mode chơi sẽ được thay đổi, cụ thể là: _isAI= 0 (tắt chế độ đánh máy), _isChall

= 0 (tắt chế độ cờ thế) và _isPlayer = 1 (bật chế độ đấu với người), turn = 0 (lượt chơi quay về bên trắng).

Về thiết kế nút bấm, nút bấm sử dụng QtMaterialFlatButton, được canh chỉnh nhờ hàm đặt vị trí setGeometry và đặt màu,kiểu chữ bằng setForegroundColor và setFont.

3.2.3 PVC

Tương tự PvP, PvC chỉ khác ở điểm là dùng AI (trí tuệ nhân tạo) để tạo cho người chơi một đối thủ hay còn gọi là đấu với máy. Ở mode chơi này, người dùng có thể chọn độ khó cho game cờ bằng cách chọn depth (sẽ nói tiếp ở phần sau).

Cũng gần như tương tự nút PvP, nội dung nút PvC được thiết kế bằng các hàm tương tự cũng như sử dụng hàm connect để liên kết thao tác bấm chuột và hàm đổi chế độ chơi. Cụ thể là: _isAI= 1 (tắt chế độ đánh máy), _isChall = 0 (tắt chế độ cờ thế) và _isPlayer = 0 (bật chế độ đấu với người), turn = 0 (lượt chơi quay về bên trắng).

3.2.4 CHALLENGE

Cờ thế là một chế độ chơi thú vị mà người chơi sẽ được thử sức với phe tấn công và nhiệm vụ của người chơi là kết thức game đấu trong một số ít nước đi theo quy định. Ở chế độ này, vị trí các quân cờ và số lượng của chúng thường là đã gần hết ván cờ, người chơi cần suy nghĩ cân nhắc để kết thúc máy do nhóm tạo ra để dành chiến thắng. Các thế cờ này là các nước kết thúc ở các ván đấu của các giải vô địch trên thế giới được nhóm tổng hợp lại có chọn lọc để đưa ra thế cờ thử thách cho người chơi.

Chế độ này sẽ bao gồm 20 màn chơi. Ở mỗi màn chơi người chơi sẽ có 1’30s tất cả để suy nghĩ và đưa ra quyết định kết thúc các thế cờ trong 4 nước hoặc thua.

Lưu ý: Đánh số thứ tự từ 1 đến 20 không liên quan đến độ khó của màn chơi.

Về nội dung nút Challenge, cũng lại tương tự như PvP và PvC, nút sẽ khởi tạo lại các biến xác định chế độ chơi (mode). . Cụ thể là: _isAI= 0 (tắt chế độ đánh máy), _isChall = 1 (tắt chế độ cờ thế) và _isPlayer = 0 (bật chế độ đấu với người), turn = 0 (lượt chơi quay về bên trắng).

3.3. Chức năng

3.3.1 Undo

Nút Undo giúp người chơi cờ trở lại bước đi trước đó. Đối với chế độ PvP, chỉ có thể Undo nếu cả người chơi đã đi ít nhất một bước đi. Đối với chế độ PvC, chỉ có thể Undo nếu người chơi đã đi ít nhất một bước đi. Chúng ta có thể Undo đến khi nào trở về trạng thái bàn cờ ban đầu.

Cài đặt:

Tạo một nút undo trên giao diện cửa sổ. Thực hiện undo bằng cách bấm chuột (click) vào nút undo. Khi đó sẽ kích hoạt để gọi hàm undo().

Trong hàm undo() sẽ kiểm tra điều kiện current có lớn hơn bằng 2 hay không, nếu lớn hơn thì mới cho thực hiện undo. Khi bắt đầu chơi cờ (nhấn nút play) thì ứng với mỗi lượt đi của mỗi bên, giá trị current sẽ tăng lên 1 đơn vị, điều kiện current ≥ 2 nghĩa là cả hai người chơi đã đều đã đi ít nhất một bước đi, thỏa điều kiện có thể undo được.

Sau khi kiểm tra điều kiện có thể undo sẽ tiến hành undo. Trừ biến current đi 2 đơn vị, nghĩa là trở về trạng thái trước đó 1 bước mà mỗi bên đã đi. Tăng biến _undo lên một đơn vị, biến _undo này nhằm phục vụ cho lệnh redo() mà ta sẽ nói ở phần dưới. Cập nhật lại bàn cờ (board) ở trạng thái vừa undo (current). Vẽ lại bàn cờ bằng lệnh boardDisplay(). Sau đó ta phải kiểm tra lệnh chiếu vua bằng lệnh check(), điều này giúp đảm bảo sẽ có thông báo cho đối phương biết nếu có chiếu vua. Cập nhật mới danh sách các bước đã đi writeHistory() sau khi đã thực hiện undo.

3.3.2 Redo

Nút Redo giúp người chơi cờ trở về trạng thái trước đó sau khi đã nhấn Undo. Chỉ có thể thực hiện Redo nếu trước đó có ít nhất 1 lần nhấn Undo. Redo cho phép người chơi trở về trạng thái y hệt như trước khi nhấn Undo.

Cài đặt:

Tạo một nút redo trên giao diện cửa sổ. Thực hiện undo bằng cách bấm chuột (click) vào nút redo. Khi đó sẽ kích hoạt để gọi hàm redo().

Trong hàm redo() sẽ kiểm tra điều kiện _undo > 0 hay không, như đã nói ở trên mỗi khi thực hiện lệnh undo() thì biến _undo sẽ tăng một đơn vị, điều kiện _undo > 0 giúp xác nhận trước khi thực hiện redo() thì người chơi đã thực hiện undo() trước đó, thỏa điều kiện có thể redo được. Nếu _undo = 0 nghĩa là người chơi không có nhấn undo trước đó hoặc đã hết lần có thể redo được.

Sau khi kiểm tra điều kiện, biến current tăng lên 2 đơn vị, nghĩa là trở về trạng thái ban đầu. Cập nhật lại bàn cờ (board) ở trạng thái vừa redo (current). Vẽ lại bàn cờ bằng lệnh boardDisplay(). Sau đó ta phải kiểm tra lệnh chiếu vua bằng lệnh check(), điều này giúp đảm bảo sẽ có thông báo cho đối phương biết nếu có chiếu vua. Cập nhật mới danh sách các bước đã đi writeHistory() sau khi đã thực hiện redo.

3.3.3 Depth

Với việc dùng alpha – beta pruning, depth (số node lấy theo chiều sâu) sẽ ảnh hưởng trực tiếp tới độ khó của game, nói cách khác AI (máy) sẽ có thể tạo ra những nước đi bất lợi hơn cho người chơi cũng như có lợi hơn cho máy.

Cài đặt:

Sử dụng một biến depthChoosed để lưu độ sâu nói trên. Giá trị của biến này sẽ được thay đổi thông qua việc chọn depth trên giao diện trò chơi thông qua một nút Depth và một combo box

Về nội dung nút "Depth": Được khởi tạo là một QtMaterialFlatButton có tác dụng truyền vào tín hiệu bấm chuột (click) để kích hoạt hàm depthChooser tạo combo box có giá trị từ 1 tới 6, với mức độ khó tăng dần theo số depth này.

Đối với combo box (nút mũi tên chỉ xuống ngay bên phải Depth) nói trên: Sau khi bấm vào nút mũi tên combo box sẽ xổ ra một list gồm 6 cấp độ đã đề cập ở trên. Ngày khi người chơi bấm chuột vào giá trị nào đó tín hiệu sẽ truyền và kích hoạt hàm setAiDepth. Từ đây giá trị depth sẽ được cập nhật tương ứng với giá trị được chọn.

3.3.4 Time

Chế độ PvC sẽ không có đồng hồ đếm thời gian.

Khi chọn chế mỗi bên quân cờ sẽ có một đồng hồ đếm ngược thời gian, hai đồng hồ này sẽ có khởi điểm là bằng nhau và giá trị do người chơi tùy chọn. Khi bấm nút play sẽ bắt đầu tính giờ. Khi tới lượt bên trắng đi thì đồng hồ bên trắng sẽ bắt đầu đếm ngược, khi quân trắng đã hoàn tất nước đi thì đồng hồ bên trắng sẽ dừng lại và đồng hồ bên đen sẽ bắt đầu đếm ngược, cứ liên tục và thay phiên nhau đến khi một trong hai đồng hồ hết giờ thì trận đấu sẽ kết thúc và đưa ra thông báo.

Cài đặt:

Để có thể tính được thời gian cho trận đấu và hiện thị được thời gian trên khung game cần phải khai báo 2 thư viện QTime và QTimer, trong đó QTimer dùng để tính toán chạy thời gian và QTime dùng để hiển thị thời gian.

Chọn thời gian cho trận đấu

Tạo một button Time để chọn thời gian cho trận đấu QtMaterialFlatButton() . Khi nhấn vào nút Time sẽ kích hoạt hàm timelist(). Người chơi sẽ nhấn vào nút mũi tên hướng xuống cạnh đó để hiển thị các lựa chọn thời gian đã lập trình sẵn theo dạng danh sách dọc. Khi nhấn vào các số trong listtime lựa chọn thời gian thì sẽ kích hoạt hàm setTime(). Hàm này có tác dụng thiết lập thời gian tổng cho trận đấu, nghĩa là thời gian hiển thị ở mỗi đồng hồ của 2 bên.

Hiển thị đồng hồ đếm ngược mỗi bên

Khởi tạo các biến time1, time2, _totalTime dùng để tính thời gian. Khởi tạo biến timer để hiển thị thời gian.

Tạo mỗi bên một đồng hồ đếm ngược, trước khi chọn thời gian và bấm play thì chỗ đồng hồ sẽ hiển thị WHITE ở vị trí đồng hồ bên trắng và BLACK ở vị trí đồng hồ bên đen. Nếu người chơi không chọn thời gian cho trận đấu thì sẽ được mặc định là 10 phút.

Khi nhấn nút play thì đồng hồ sẽ bắt đầu chạy, tức là hàm time() bắt đầu chạy. Trong hàm time() sẽ kiểm tra đang đến lượt bên nào thông qua biến _player, giả sử đang là lượt đi của bên trắng (_player=0) thì đồng hồ bên trắng sẽ giảm từ từ từng giây qua lệnh addSecs(-1). Đó sẽ xuất ra thành chuỗi với hình thức “mm:ss”, nghĩa là phút và giây. Sau khi bên trắng hoàn thành bước đi thì giá trị biến _player được gán bằng 1. Đồng hồ bên trắng sẽ dừng lại và hàm time() kiểm tra điều kiện nếu biến _player khác 0 thì đồng hồ bên sẽ giảm, tương tự như cách hoạt động lúc nãy bên trắng. Cứ tuần tự đến thi hết thời gian thì thông báo hết giờ.

3.3.5 Play

Cũng được tạo bằng QtMaterialFlatButton và được đặt vị trí kiểu chữ ,… tương tự như các nút khác.

Tác dụng của nút Play: Ngay sau khi bấm nút Play trên giao diện, tương tự các nút khác, hàm connect của nút Play sẽ được kích hoạt nhân vào tín hiệu bấm chuột và gọi hàm play tương ứng.

Ở hàm play ta sẽ có một số hành động được thực hiện:

Xóa lịch sử nước đi (mục ghi lại nước cờ đã đi) thông qua phương thức clear.

Gán (lại) biến posi (nhãn Position) với giá trị là một xâu rỗng.

Gán (lại) biến tim (nhãn Clock) cũng với giá trị là một xâu rỗng.

Gán _start (giá trị kiểm tra để khởi động game) bằng 1 (bắt đầu chơi).

Gán lại biến _player (người chơi hiện đang có lượt đi) bằng 0.

Gán lại biến turn (lượt chơi) bằng 0 (chuyển lượt về cho quân trắng)

Gán lại biến time1 (đồng hồ đếm giờ của người chơi 1) bằng với thời gian được chọn đã được gán trong biến _totalTime . Tương tự vơi time 2, time2 (đồng hồ đếm giờ của người chơi 2) = _totalTime và dừng đếm giờ

timer->stop

Gán lại biến đếm lượt đi current bằng 0 và phase AI (cách tính giá trị nước đi) bằng OPENING (sẽ nói rõ hơn trong phần AI)

Nhận về giá trị các biến _isAI, _isPlayer, _isChall từ các nút đã đề cập ở trên, dùng câu lệnh if khởi tạo bàn cờ theo chế độ chơi tương ứng (chế độ PvP sẽ có thêm đồng hồ đếm giờ) và chế độ chơi cờ thế sẽ có cách chơi riêng (đã đề cập ở trên) và vẽ bàn cờ dựa theo thế cờ đã được chuẩn bị trước.

Gán lại giá trị lưu nước đi hiện tại boardHistory[current] bằng trạng thái bàn cờ mới được đặt lại và kết thức bằng việc vẽ lại bàn cờ đã đặt – BoardDisplay.

3.3.6 Ghi biên bản

Nội dung :

Ghi biên theo luật cờ vua quốc tế [28]

Quy định theo ký hiệu viết tắt của quân cờ (viết chữ in) + ô nó di chuyển đến (viết chữ thường).

– Nước bắt quân được thể hiện bằng ký tự “x” chen giữa. Ví dụ: Mxc4

– Không ghi tên viết tắt của chốt. Nước đi chốt được biểu đạt bằng tên ô mà nó đi tới. Ví dụ: d4 nghĩa là chốt đi đến ô d4. Nước chốt bắt chốt được thể hiện bằng tên cột nó đang đứng + “x” + ô nó bắt quân. Ví dụ: dxc4 nghĩa là chốt cột d bắt quân ở c4....

Tiến hành ghi biên bản như trên nhưng là tiếng Anh

Thực hiện :

Sử dụng QlistWidget để làm khung biên bản, mỗi nước đi được thể hiện trên biên bản bằng Qstring.

3.4. Kiểm tra trạng thái đặc biệt

3.4.1 Chiếu tướng

Tác dụng :

Khi quân vua của một phe bị tấn công sẽ thực hiện chiếu tướng để thông báo cho người chơi biết mà phòng thủ.

Trong chương trình hàm chiếu tướng là check sẽ được gọi sau khi thực hiện bước đi moveEngine

Thực hiện :

check sẽ kiểm tra bằng hàm có sẵn board.ColorIsInCheck nếu true thì đổi ảnh của tile vua bằng ảnh vua bị chiếu

Đây là hình ảnh vua bị chiếu được dùng

3.4.2 Kết thúc game

Tác dung :

Endgame dùng để kết thúc ván đấu, hiển thị thông báo chiến thắng hay thua cuộc tùy vào kết quả.

Endgame dùng cho các trường hợp :

• Một phe bất kỳ bị chiếu và không thể đi nước nào khác

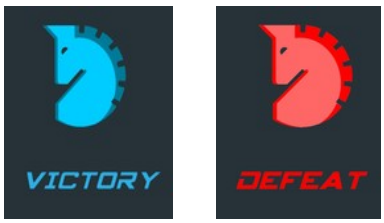
• Một phe bất kỳ hết thời gian chơi

• Khi chơi Challenge ở thế công đánh quá 4 nước

Trong chương trình hàm Endgame là endgameCheck sẽ được gọi sau khi thực hiện bước đi moveEngine

Thực hiện :

Kiểm tra các điều kiện tương ứng như trên và trả về kết quả thắng hoặc thua bằng 1 QmessageBox

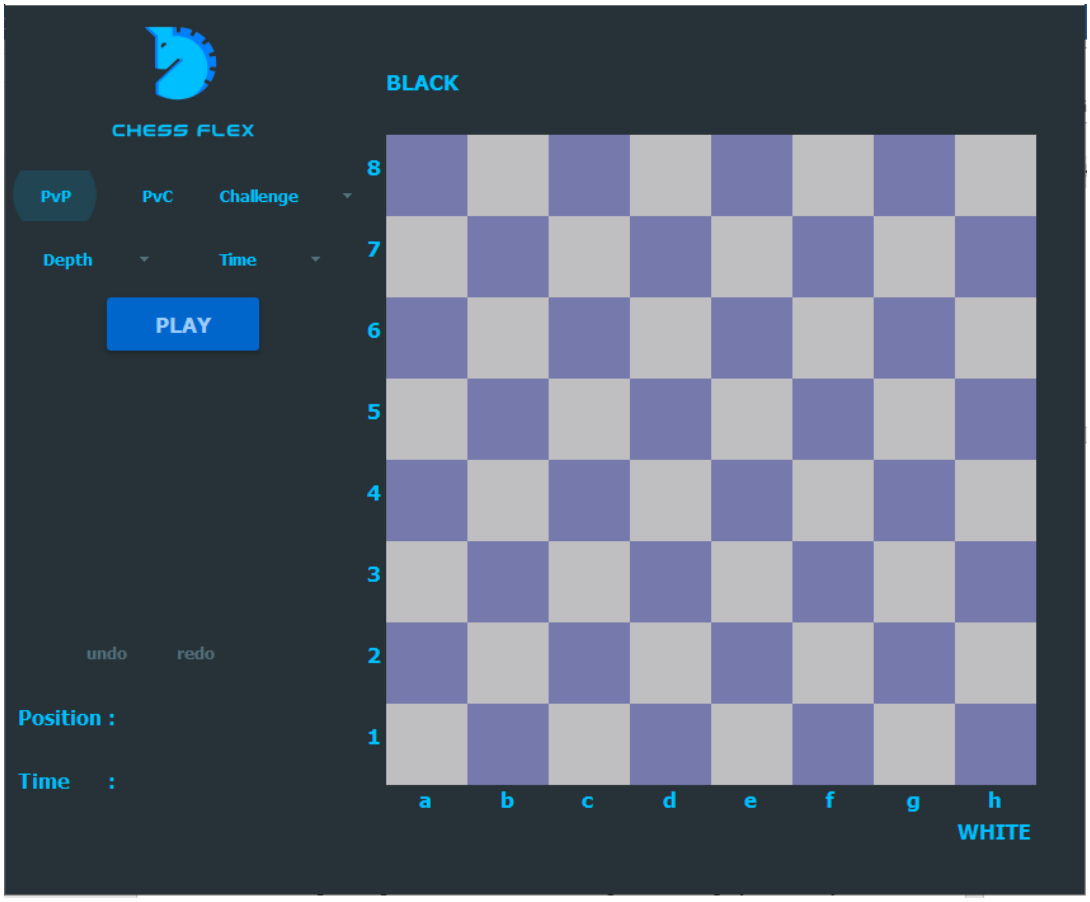


Đây là hình ảnh tương ứng thông báo kết quả

CHƯƠNG V: TỔNG KẾT

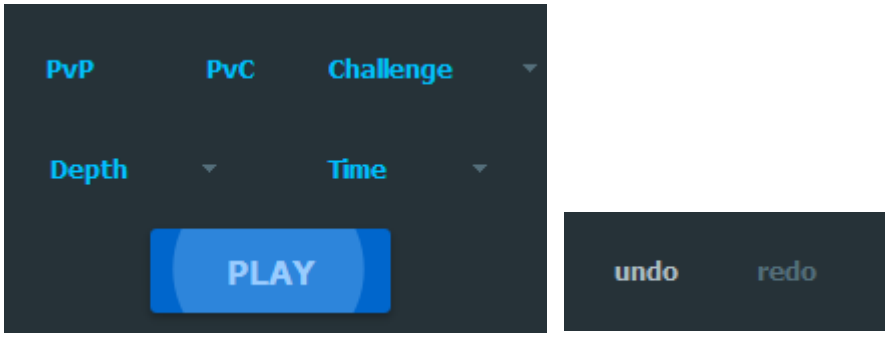
1. Kết quả sản phẩm

1.1. Giao diện



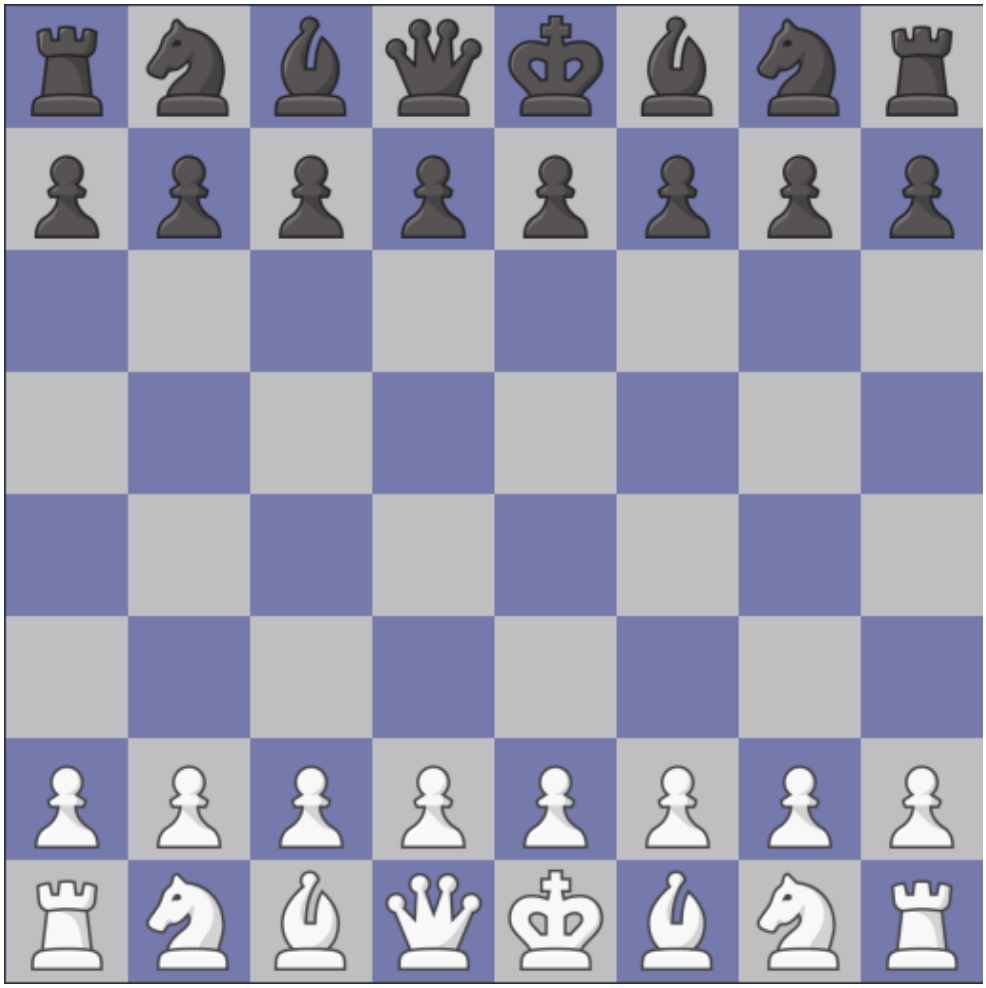
Giao diện khi bắt đầu chơi

1.2. Tính năng



Các nút tính năng

1.3. Bàn cờ

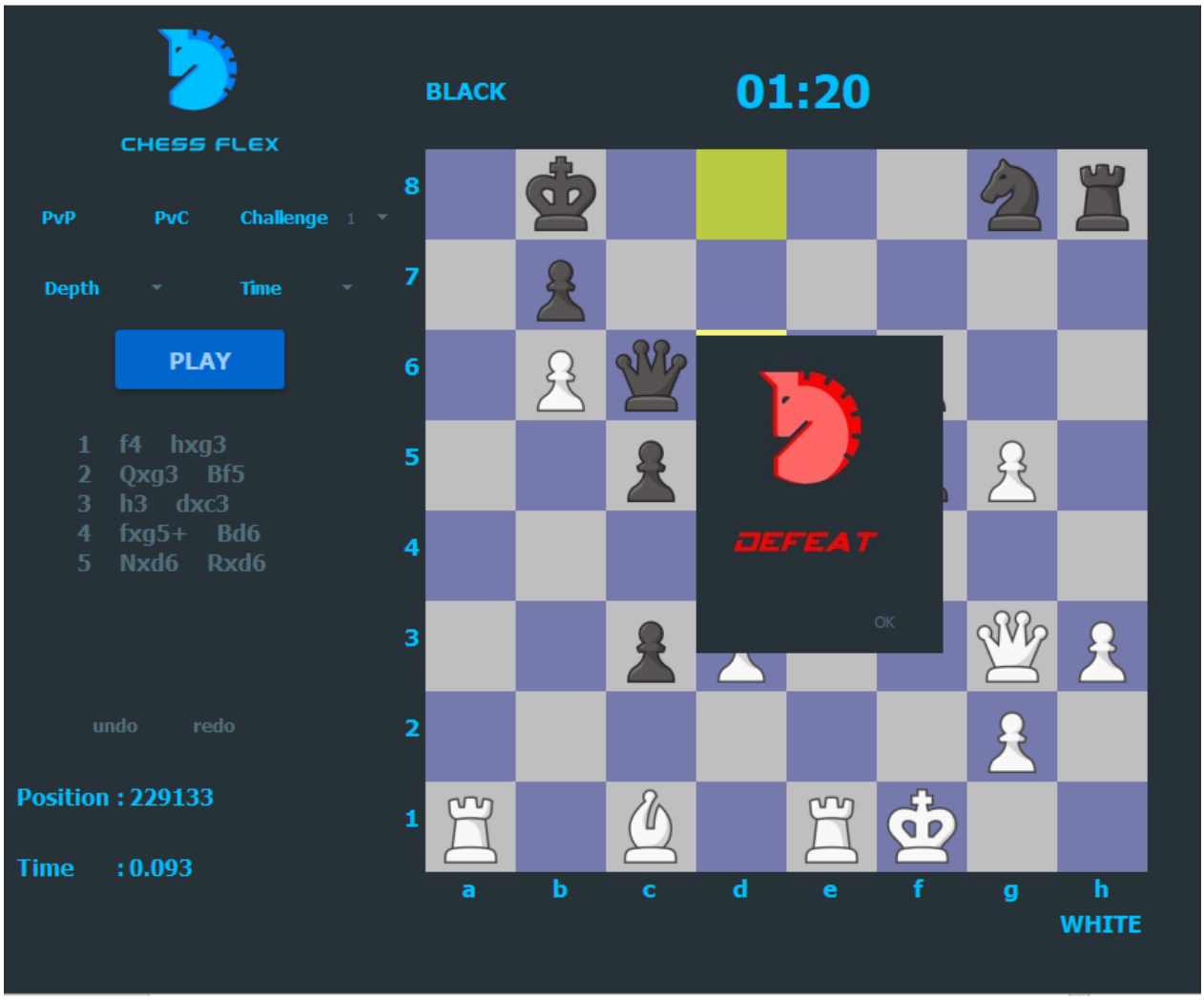


Bàn cờ mặc định

1.4. Thắng

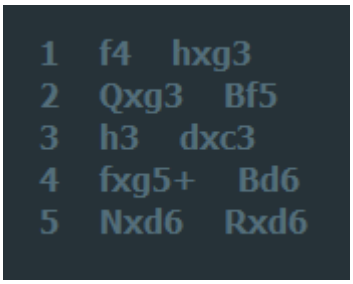
Thông báo chiến thắng

1.5. Thua



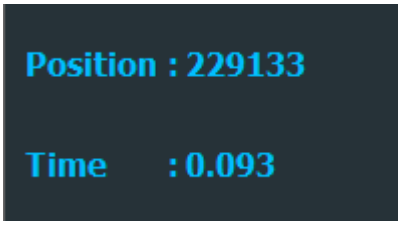
Thông báo thua cuộc

1.6. Biên bản



Lịch sử ghi biên bản

1.7. Thông số AI



Thống kê thống số position và time

2. Thử nghiệm

Các thành viên trong nhóm đều bại trận trước AI ở độ sâu 5, do trình độ cá nhân thấp nên chưa kiểm định được AI

3. Khó khăn

Tìm hiểu để sử dụng công cụ và ngôn ngữ lập trình như: WPF – C# và Qt – C++

Tìm hiểu các giải thuật và phương pháp liên quan đến AI, vì những kiến thức này khó thực hiện và hiểu

Tập trung đầy đủ các thành viên vì lịch học dày đặc

Các lỗi phát sinh rất khó sửa, vì các tính năng được chia nhỏ cho các thành viên

Chưa có kinh nghiệm làm việc nhóm và các đồ án lớn

Gặp nhiều bỡ ngỡ trong công tác quản lý và phân chia công việc

4. Giải quyết

Đọc các tài liệu liên quan đến xây dựng cờ vua AI

Tham khảo những người đi trước và tái sử dụng code của họ

Tham khảo cách tổ chức oop, cấu trúc dữ liệu

Các thành viên nổ lực đóng góp thời gian công sức để bù đắp khó khăn

Giảm bớt thời gian cá nhân lại và tập trung họp nhóm nhiều hơn

5. Đánh giá và bài học

Sản phẩm đã hoàn thiện tương đối so với mục tiêu đồ án đề ra lúc đầu, hạn chế những sai lầm ngu ngốc như thí cờ vô nghĩa, đi qua đi lại... Nhưng còn hạn chế là chưa có chiến lược

Học được các phương pháp tổ chức 1 đồ án

Học được các giải thuật cải tiến tốc độ xử lý

Học được cách làm việc nhóm hiệu quả

Tiếp tận được các công cụ và ngôn ngữ lập trình mới

Tiền đề phát triển môi trường việc làm sau này

Tích lũy kinh nghiệm tìm kiếm, tham khảo, ứng dụng

6. Cải tiến

AI:

Một số cải tiến mà chúng em có thể thực hiện đối với sản phẩm này:

Move ordering: giúp alpha beta pruning được tối đa

Board evaluation: cải tiến đánh giá thêm về cuối trận

Opening Book: thêm chiến lược cho AI

Chức năng :

SAVE load trận đấu

Cờ thể cho người chơi tự custom

CHƯƠNG VI: TÀI LIỆU KHAM KHẢO

[1] Giải vô địch cờ vua Thế giới.
<https://en.wikipedia.org/wiki/World_Chess_Championship>
Xem 04/05/2019

[2] Các nhà vô địch.
<https://en.wikipedia.org/wiki/World_Chess_Championship#World_champions>
Xem ngày 04/05/2019

[3] Luật chơi cờ vua.
<https://en.wikipedia.org/wiki/Rules_of_chess>
Xem 04/05/2019

[4] Cờ nhanh.
<https://vi.wikipedia.org/wiki/C%E1%BB%9D_nhanh?fbclid=IwAR2yvQ-wvxBKFMM74x_HoZocW0g8k_J6IXVQlOSFMHhns-XAcHqV3H0M>
Xem 10/05/2019

[5] Chess in C#.
<https://www.c-sharpcorner.com/article/sharpchess-in-C-Sharp/>
Xem 10/05/2019

[6] Qt
<http://devnt.org/qt-tut-series-1/>
Xem 20/05/2019.

[7] Fast chess move generation with magic bitboard.
<https://rhysre.net/2019/01/15/magic-bitboards.html>
Xem 20/05/2019

[8] Magic Bitboards.
<https://www.chessprogramming.org/Magic_Bitboards>
Xem 20/05/2019

[9] Chess engine in C++.
<https://github.com/GunshipPenguin/shallow-blue>
Xem 27/05/2019

[10] Thuật toán MiniMax
<https://viblo.asia/p/thuat-toan-minimax-ai-trong-game-APqzeaVVzVe>
Xem 27/05/2019

[11] Mã giả thuật giải MiniMax.
<https://viblo.asia/p/thuat-toan-minimax-ai-trong-game-APqzeaVVzVe>
Xem 27/05/2019

[12] Minh họa cây trong thuật giải MiniMax.
<https://www.freecodecamp.org/news/simple-chess-ai-step-by-step-1d55a9266977/>
Xem 27/05/2019

[13] Định nghĩa giải thuật Alpha-Beta.
< https://en.wikipedia.org/wiki/Alpha%E2%80%93beta_pruning>
Xem 27/05/2019

[14] Mã giả giải thuật Alpha-Beta.
<https://en.wikipedia.org/wiki/Alpha%E2%80%93beta_pruning>
Xem 27/05/2019

[15] Cây cắt tỉa trong giải thuật Alpha-Beta.
<https://www.freecodecamp.org/news/simple-chess-ai-step-by-step-1d55a9266977/>
Xem 27/05/2019

[16] Evaluation.
<https://www.chessprogramming.org/Evaluation>
Xem 27/05/2019

[17] Evaluation Function for Chess.
<https://slideplayer.com/slide/8174134/>
Xem 27/05/2019

[18] Simplified Evaluation Function
<https://www.chessprogramming.org/Simplified_Evaluation_Function>
Xem 03/06/2019

[19] Chess step by step.
<https://www.freecodecamp.org/news/simple-chess-ai-step-by-step-1d55a9266977/>
Xem 03/06/2019

[20] Hình ảnh logo.
<https://www.logodesign.net/logos/chess>
Xem 03/06/2019

[21] Hình ảnh báo hiệu thắng thua.
<https://www.logodesign.net/logos/chess>
Xem 03/06/2019

[22] Hình ảnh chiếu vua.
<http://images.chesscomfiles.com/chess-themes/pieces/neo/150/wk.png>
Xem 03/06/2019

[23] Hình ảnh quân cờ cho 2 bên.
<http://images.chesscomfiles.com/chess-themes/pieces/neo/150/wr.png>
Xem 03/06/2019>

[24] Âm thanh được dùng trong game.
<https://freesound.org/people/>
Xem 10/06/2019

[24a] Âm thanh bắt đầu trận đấu
<https://freesound.org/people/qubodup/sounds/159552/>

[24b] Âm thanh chiến thắng
<https://freesound.org/people/joshuaempyre/sounds/404025/>

[24c] Âm thanh thua cuộc
<https://freesound.org/people/LittleRobotSoundFactory/sounds/270403/>

[24d] Âm thanh di chuyển
<https://freesound.org/people/ChaosEntertainment/sounds/396705/>

[24e] Âm thanh tấn công

<https://freesound.org/people/joshuaempyre/sounds/404025/>

[25] AI

<https://www.freecodecamp.org/news/simple-chess-ai-step-by-step-1d55a9266977/> Xem ngày 15/06/2019.

[26] Minh họa moveGen.

< https://www.freecodecamp.org/news/simple-chess-ai-step-by-step-1d55a9266977/> Xem ngày 15/06/2019.

[27] Minh họa cho thuật giải MiniMax có cắt tỉa Alpha-Beta.

<https://www.freecodecamp.org/news/simple-chess-ai-step-by-step-1d55a9266977/> Xem 15/06/2019

[28] Cách ghi biên bản cờ vua Quốc tế.

<https://chess.edu.vn/kien-thuc-chung/bien-ban-van-co-va-cach-ghi-chep-nuoc-di/> Xem 15/06/2019