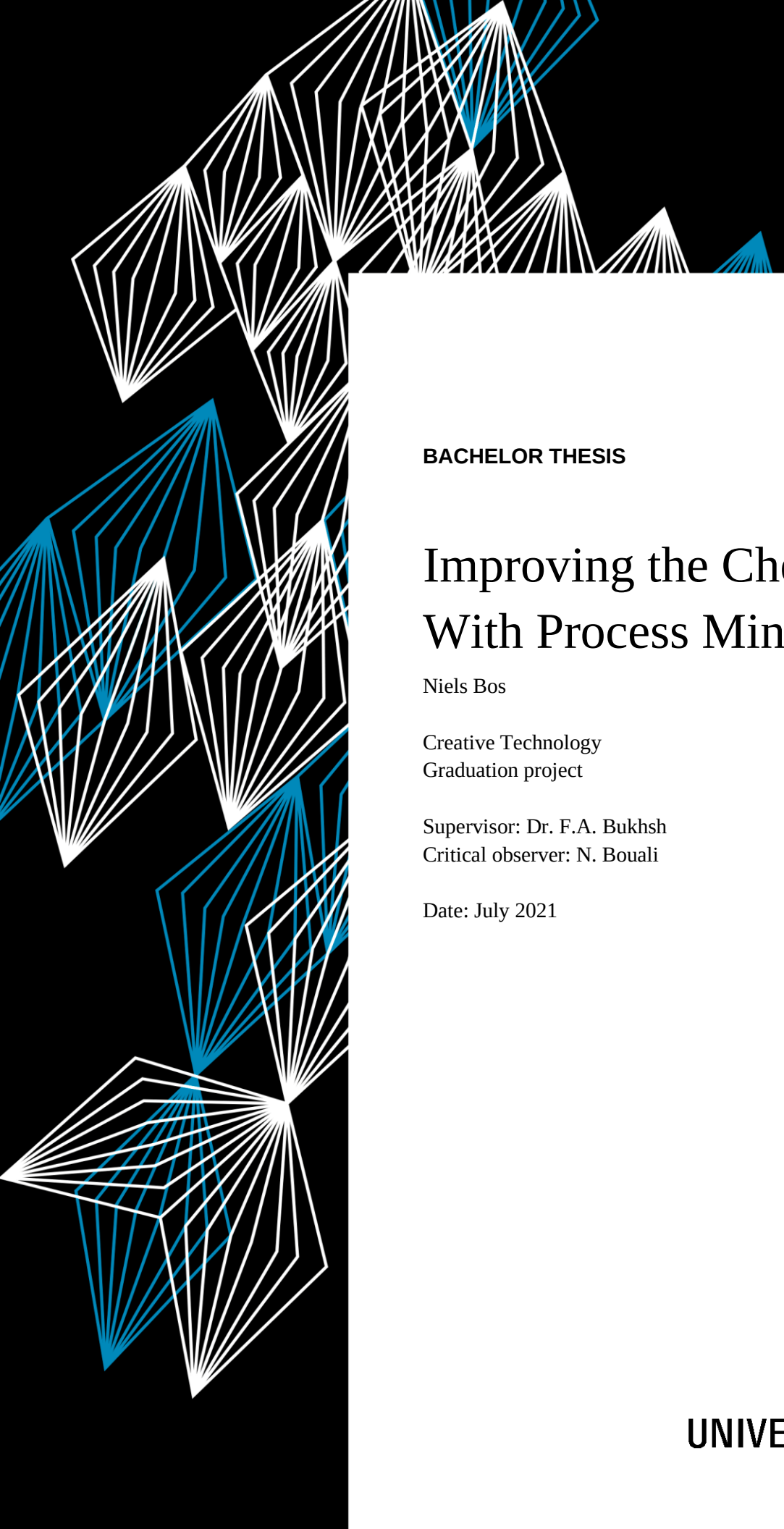


BACHELOR THESIS

Improving the Chess Elo System With Process Mining

Niels Bos

Creative Technology
Graduation project

Supervisor: Dr. F.A. Bukhsh
Critical observer: N. Bouali

Date: July 2021

UNIVERSITY OF TWENTE.

Abstract

Over the last decade, the amount of data generated by software applications e.g. information systems, websites, mobile applications etc. has increased tremendously. Process mining, a subdiscipline of data science, uses this data to analyse and improve processes. In this research, the possibilities of process mining on chess event logs are explored, to ultimately improve the chess Elo system. The chess Elo system is a widely used and well accepted rating system. The Elo system is, however, flawed in multiple ways. Two major flaws of the Elo system, are its incapability to review a player's strength and the excessive time needed to gain the appropriate Elo rating. This research explores the potential of process mining to identify chess expertise. To be more specific, multiple process mining techniques are applied on chess event logs, and the generated process models are analysed to identify chess expertise. This research presents a method to analyse the differences between high and low rated players. This is achieved by comparing process models generated from high and low rated chess games. The results show that by comparing the process models differences between high and low rated players can be observed. Process mining is therefore a promising approach to improve the Elo system and might be applicable to other software too. However, only the first twelve moves of a game were used. To gain more insight into the differences between high and low rated players, the mid and end games should be included in the event logs as well. Future research should be conducted with more chess games added to the event logs to increase the validity.

Acknowledgement

First of all, I would like to thank my supervisors Faiza Bukhsh and Nacir Bouali for their optimism, support and critical view during my bachelor project. They have helped me stay on track and provided tools and examples. Secondly, I would like to thank Tijs Zandt for all the motivational support during the hours we worked together. Lastly, I would like to thank my family for their support and encouragement.

Table of Contents

List of Figures

List of Tables

Chapter 1 –Introduction

Over the last decade, the amount of data generated by software applications e.g. information systems, websites, mobile applications etc. has increased tremendously. Whenever an account is registered, a file uploaded, button clicked or message sent, data is generated and stored. With the advancements in gathering and storing data, the interest in the field of data science and analytics has increased significantly. Aalst defines **data science** as a combination of classical disciplines like statistics, data mining, databases and distributed systems and aims at turning data into usable information [1] . Process mining is one of the subdisciplines of data science. It is also a subdiscipline of process science. **Process science** refers to the broader field that combines the knowledge of information technology and knowledge from management sciences [1]. Process science aims at improving and running operational processes. In Figure 1.1 the position of process mining between the two disciplines is illustrated.

Figure 1.1: A visualisation of how process mining covers both process science and data science [1]

Process mining was introduced in 2004 by Aalst [22]. **Process mining** is a group of techniques in the field of data analytics, to extract process related information [1]. Information is extracted from event logs, often by visualising the behaviour in the event logs in process models. Event logs contain traces, sequences of events. An event is some activity that occurred in a certain time and has certain properties. An example of a simple event log can be found in Table 1.1. With a simple event log, process mining techniques can be used to discover a process model and this model can be validated. Thereafter, the process can be analysed to extract imperative information about the process, and possibly improve the process and process model. Information about a process can include bottlenecks, workflows, frequency of events and much more.

In process mining, there are three main techniques: process discovery, conformance checking and process enhancement [1]. In **process discovery**, process models that capture the behaviour in the event logs are constructed. The process model is generated by an algorithm and may therefore not represent reality adequately. **Conformance checking** is a process mining technique that checks the fitness of a process

model. Conformance checking compares the event log to the process model, to analyse to what degree they correspond. Lastly, **process enhancement** aims at extending or improving a process model by utilizing properties from the event log. Often, event logs contain additional information next to the activity. For example, events must contain timestamps to gain knowledge of the time perspective in a process.

Case id	Event id	Properties				
	35654423	Timestamp	Activity	Resource	Cost	...
1	35654424	30-12-2010:11.02	Register request	Pete	50	...
	35654425	31-12-2010:10.06	Check ticket	Sue	400	...
	35654426	05-01-2011:15.12	Decide	Mike	100	…
	35654427	06-01-2011:11.18	Reject Request	Sara	200	…
2	35654483	30-12-2010:11.32	Register request	Mike	200	...
	35654485	07-01-2011:14.24	Decide	Mike	100	...
...	...	...	….	...	...	…

Table 1.1: A sample event log [1]

Process mining is used in many business areas such as healthcare, public, transportation, education, and can be used on software to gain insights in the workflow and use of the interface [24]. In addition, it is used in the gaming industry to analyse player behaviour and discover strategies [25]. Process mining can even be applied to one of the oldest games, chess [26]. **Chess** is a turn based, 2 player, strategy game which recently rose in popularity by a large amount because of the Netflix series, The Queen's Gambit [27].

Daily, a large number of chess games are played on online chess platforms, which all use similar **Elo systems** to determine the player's strength. The Elo system is a matchmaking system, which provides every player with an Elo rating that indicates the player's strength, and players with similar ratings are matched up against each other. A player's rating decreases after a loss and increases after a win. The magnitude of which a player's rating is increased or decreased is based on the difference in Elo rating of the players. Larger differences in Elo rating lead to larger increases in rating when the lower rated player wins, and smaller increases in rating when the higher rated player wins.

Regardless of its wide use, the Elo system has multiple flaws. First, the Elo system does not consider how well a player played when determining the Elo score, it simply looks at the win/loss/draw ratio. Second, a player's rating does not change as fast as one's actual expertise. For example, when a player stopped playing for a certain amount of time, their Elo rating remained the same while their chess expertise decreased. Third, to determine a player's chess rating when the player starts playing for the first time, a lot of games need to be played to calculate an accurate score. During these games, the player might play against opponents far beyond or below their game level.

Therefore, a new method to rate new players must be found. This method should not solely consider the difference in rating between the players, but also consider to what extent the players played correctly.

Eventually, this method could be applied to other games that incorporate an Elo rating or even to find the expertise level of users for other types of software. For software programs, knowing the user's expertise and experience with the software is valuable since it can assist in personalising the software. Users with a lot of experience with the software could be provided with advanced features and users with little experience could get explanation modals or a tour. Ultimately, it would provide a better user experience.

The first step in improving the Elo system is to determine chess expertise based on the moves of a player, instead of their win/loss/draw ratio. Therefore, the aim of this research is to determine if **the chess expertise of a chess player can be estimated with only a few games as input, and to distinguish between inexperienced and intermediate players**. This research will also explore the possibilities of process mining in the chess domain. Chess can be seen as a complex process. Consequently, methods to apply process mining techniques on complex chess event logs must be explored. Next, a method to define experienced and inexperienced chess players must be found using the event logs of a game. To achieve these goals, the following research questions were constructed:

- *RQ1: What are the differences in gameplay and mindset between novice and expert chess players and how can they be identified?*
  First, literature regarding the differences between novice and expert chess players will be explored. This is necessary to eventually recognize certain behaviours in the process models. Section 2.4 explains the state of the art of identifying chess expertise.
- *RQ2: How can process mining techniques be applied on complex event logs?*
  Second, various process mining techniques will be explored to find methods to deal with complex processes. In chapter 2, the state of the art of process mining on complex processes is described. Chapter 3 elaborates the methods and techniques used to perform process mining techniques on chess event logs.
- *RQ3: How can process mining techniques be utilized to identify chess expertise?*
  Lastly, process mining techniques and the produced process models will be analysed to identify chess expertise. In chapter 5, the process of retrieving the event logs, generating process models and analysing them is elaborated.

This research starts with a literature review (chapter 2) of process mining on software logs. The most commonly used process mining techniques and where they are applied will be described. Also, the state of the art of identifying chess expertise is described. Next, the methodology (chapter 3) of this research will be explained, including where the data is extracted from, how it is processed and converted to event logs and which attributes are used. The structure of the succeeding chapters of this research is based on the Creative Technology Design Process (see Figure 1.2), which is a framework to design products, developed by Mader and Eggink [23]. The framework consists of four phases: ideation, specification, realisation, and evaluation. In chapter 4, the ideation phase is elaborated. The ideation phase consists of a stakeholder analysis, acquisition of relevant information, the research goals and the final concepts developed in the phase. The specification and realisation phase are processed together in chapter 5. In this chapter, data formatting choices are elaborated, and the results of the process mining techniques are illustrated and explained. Subsequently, the results are discussed (chapter 6) and a conclusion (chapter 8) is formed. Lastly, the evaluation phase of the framework is stated in chapter 9, in which experts review the results.

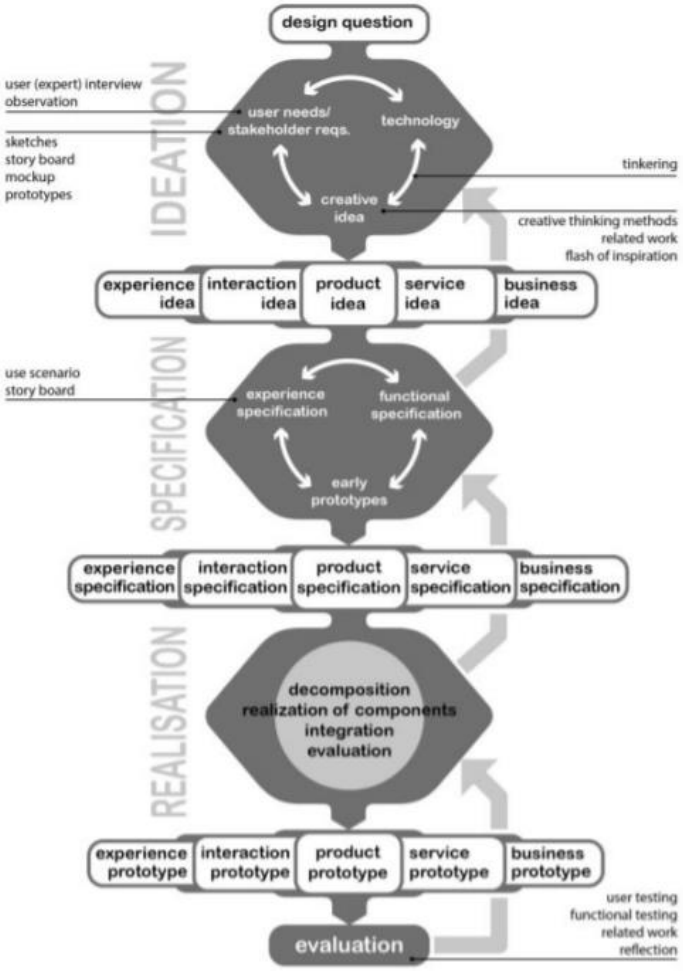


Figure 1.2: Creative Technology Design Model

Chapter 2 –Background Research

In this chapter, published research that relates to applying process mining on chess event logs is examined. When applying process mining on chess event logs, a set of challenges emerges. To begin, chess is a complex process, which complicates the process discovery. Next to that, chess expertise must be defined and methods to estimate chess expertise must be found. This chapter will start with explaining the basic process mining techniques and overall process mining challenges. Subsequently, various methods to deal with complex processes will be explained. Lastly, published research explaining methods to identify chess expertise will be explained, followed by non-literary existing software that estimate ELO ratings.

2.1 Process mining techniques

Process mining can be used for various purposes with multiple techniques. Van der Aalst states that there are three main types of process mining: discovery, conformance, and enhancement [1]. To begin with, discovery techniques are used to discover a process from event logs. They produce a model representing a process without prior knowledge, based on the event logs. Secondly, conformance techniques are used to check if reality corresponds to the found event logs. The behaviour of an existing process model is compared with the event logs to measure the accuracy of the model and deficiencies can be removed. For instance, if the event logs show a certain sequence of events which cannot be generated in the model, the model is incomplete. The last type of process mining is enhancement, which is used to improve or extend an existing process model. For example, a process model could be extended by adding attributes like timestamps to visualise bottlenecks and it could be improved by modifying the model to better reflect reality [1].

In these three process mining categories, multiple different algorithms and techniques exist, each with different advantages and disadvantages. First, four main discovery algorithms are described, subsequently techniques of conformance checking are explained.

The **α-algorithm** is a process discovery technique that produces a Petri net given an event log. A Petri net is a simple process modelling language that clearly and intuitively visualises a process. The α-algorithm is relatively simple compared to other discovery techniques, and it can deal adequately with concurrency but has some major limitations. Van der Aalst states that the models produced by the algorithm are not always sound, meaning that either the model is incomplete, there are redundant events or the end point is not reached. In addition, the algorithm cannot handle noise in the event logs and might produce unnecessary complex routes [1]. The same limitations were stated by Weerapong et al. in a case study using the alpha algorithm [3]. Amelia Effendi and Sarno also mention these limitations and add that the algorithm is unable to mine duplicate and hidden tasks [4]. Therefore, this algorithm sees little use in practical applications but is the foundation of many other techniques [1].

A more advanced technique is the **heuristic mining algorithm**. This algorithm utilizes the frequency of events in the event logs. This allows for better noise control since paths that occur infrequently can be left out of the model. However, the heuristic mining algorithm can still produce unsound models and needs larger event logs to generate an accurate model [1].

Inductive mining is another technique that produces a model based on a process tree. A process tree shows the sequence of events through branches. This technique always produces sound models since process trees are sound by construction. Moreover, research from Nuritha and Mahendrawathi shows that the inductive mining technique is better at dealing with noise compared to the heuristic mining technique

[5]. Inductive mining is currently one of the leading techniques because of its scalability, flexibility, and soundness of the models [1].

The **Fuzzy miner** is an algorithm developed by Gunther and Aalst in 2007, that addresses the problems of large numbers of events and transitions [28]. The fuzzy miner outputs a Fuzzy model. In this model, the less important events are suppressed in clusters, which makes the overall model clearer. However, these clusters do add a layer of abstraction and can make it harder to follow the actual flow of events. The Fuzzy miner is mainly used with complex and unstructured event logs.

The second category of process mining techniques is **conformance checking**. One of the most used conformance checking techniques is token-replay. From the given event log, the traces (a certain sequence of events) are replayed in the model. If the trace is able to complete within the model, the model represents the behaviour correctly. This is repeated for all the traces in the event log. To what extent the behaviour matches the model is defined as fitness. A high fitness means that the model represents the data to a high extent. The fitness is calculated by the amount of completed traces against the total amount of traces. Even though token-replay is one of the most used techniques, it has some drawbacks. The most important drawbacks are that the technique can only be applied on Petri nets and that the fitness of problematic models tends to be too high. When applying token replay on problematic models the fitness could be too high, creating the illusion that the model behaves correctly [1].

To overcome these drawbacks, the alignment technique was introduced. This technique still calculates the fitness of a model but does this by aligning the found traces to the model. Whenever an event does not align, a skip marker is added to indicate the misalignment. Given the total amount of events and misalignments, the fitness can be calculated [1].

2.2 Main process mining challenges

When applying process mining techniques in general, one might face multiple challenges in the different stages of process mining. In this section, various common challenges are stated and described.

Van der Aalst states two main challenges related to the event logs [1]. Among others, these two main challenges are also described by R'bigui and Cho [6]. To begin, proper event logs are essential to discover processes, and noise in the data can therefore be a problem. In this case, noise is defined as events that occur very infrequently, i.e., outliers. To ensure valid models, the noise needs to be filtered out. A threshold needs to be set correctly to filter out the noise while preserving the correct data [1][2]. The second main challenge stated by Van der Aalst is incompleteness of data [1]. Whereas noise is caused by too much data, incompleteness is the result of too little data. With incomplete data, important possible traces may not be incorporated into the model and cannot be analysed.

When event logs have been established, the right parameters must be chosen for the discovery techniques. Keith and Vega explain that selecting the optimal parameters for process discovery is an important but complex task because of the large number of possibilities [2]. Finding the right algorithms and optimal parameters is very time consuming and is therefore one of the challenges in process mining. Next to that, they mention that process mining must be integrated with other methodologies and techniques to gain the desired information. For example, this could be tools for process improvement or to analyse processes to redesign and advance the process [2][6].

Lastly, a challenge described by R'bigui and Cho is that a representative benchmark to compare different process mining techniques is missing [6]. This results in trial and error to find the best process mining technique or tool to use at the start of anyone's process mining.

2.3 Process mining and complex processes

Most common process mining techniques are unable to handle complex processes. Complex processes have a high concurrency or many different tasks and transitions. One way to deal with complex processes is to simplify the model or simplify the event logs. Chapela-Campa et al. [11] introduced UBeA, which is a technique to abstract non-core behaviour from a process model. UBeA takes as input an event log, the Petri-net and the index of traces that should not be abstracted. The technique produces the abstracted Petri-net and event log with the non-core behaviour narrowed down in new activities. This simplifies the model as it removes large amounts of unnecessary events and keeps the model clearer.

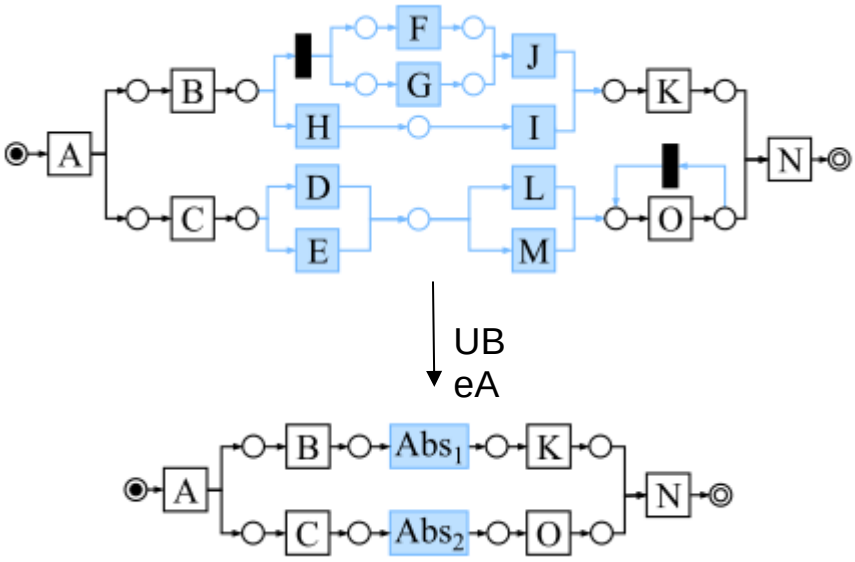


Figure 2.1: Simplifying process models by abstracting infrequent behaviour.

UBeA is an independent technique that can be used for various purposes. In their research, they also introduce IBeA, an algorithm to simplify process models by abstracting infrequent behaviour. IBeA is a technique that combines UBeA and WoMine [12]. WoMine is an algorithm that extracts frequent behavioural patterns from process models. It extracts these patterns by searching for frequently occurring sequences, selections, loops or parallels. IBeA uses WoMine to detect and abstract infrequent behaviour, to produce simpler process models. In addition, it produces a simplified event log which could be analysed with other process mining techniques. These techniques, however, do make the overall process less precise and reduce the overall fitness to gain clarity.

Common structures that make process models more complex are concurrency and loops. Research by Sun et al. focusses on dealing with the combination of these two structures, multiple-concurrency short-loop structures (MCLS) [13]. In their research, they propose an Alpha Mining technique that can mine incomplete logs with MCLS structures effectively. They claim that the algorithm improves the fitness,

precision, recall, simplification, and generalisation of the mined process model. However, the algorithm is still incomplete since it does not consider the impact of duplicate or visible events.

According to Ferilli and Angelastro, chess can be cast as a complex process [7]. Chess is complex due to four main factors. First, chess has a high concurrency, meaning that there are many pieces on the board at the same time. Secondly, chess has a high number of tasks, meaning that there are many different possible tasks due to the combination of pieces and squares. Thirdly, chess has a huge number transition, meaning that there are factors accompanying a move like checks and captures. Lastly, chess has a huge number of possible cases, since the number of possibilities in chess is almost endless.

To overcome these complexities, Ferelli and Angelastro used the Workflow Management framework (WoMan framework). The WoMan framework is a framework based on First-Order Logic. By focussing on tasks and transitions, the framework is able to discover logical workflows from complex processes [8] [9]. The WoMan framework takes trace elements in the form, entry (T, E, W, P, A, O, R), as input. Ferelli and Angelastro [7] filled the trace elements with the following parameters:

T a progressive number indicating the event timestamp;

E one of the allowed event types:

- begin of process — the start of a match;
- begin of activity — a certain piece is placed in a certain square;
- end of activity — a certain piece is removed from a certain square;
- end of process — the end of a match.

W the name of the process model the entry is referred to;

P a unique match identifier, obtained by concatenation of the following data: name of white player, name of black player, place and date of the match;

A the name of the activity;

O the progressive number of occurrences of A in P;

R the player (white or black) responsible for the beginning or end of activity.

With this approach, they were able to discover and analyse a process from 200 chess games. The WoMan framework was proven to be able to handle complex processes. In addition, a study by Ferilli et al. on process mining on traffic, also states that the WoMan framework is able to deal with complex processes [10]. However, they also state that the framework might not be able to handle huge amounts of data, which is a possible shortcoming.

To conclude, when applying process mining techniques on chess event logs, chess should be treated as a complex process in order to get useful information. This could be by abstracting infrequent behaviour or using the WoMine framework. Ultimately, this would lead to a process model that can be analysed and information can be extracted.

2.4 Identifying chess expertise

Before applying process mining techniques to identify chess expertise, chess expertise must be defined and methods to measure chess expertise must be found. In this section, chess expertise will be defined and factors that influence it will be identified. Next, methods to measure or estimate chess expertise will be described and analysed.

2.4.1 Differences between novice and expert chess players

Before the differences in chess expertise can be identified, chess expertise must be defined. Chess expertise is the skill and knowledge to: find the best move in a certain chess position, analyse the strengths and weaknesses of positions and having knowledge of common patterns and openings. Unlike most other disciplines of expertise, chess expertise has an objective and valid indicator, the ELO ranking system [14]. When a player participates in an official chess tournament and wins a game, his or her ELO rating is slightly increased based on the opponents rating. As mentioned in Chapter 1, a player gains more after a win against an opponent with a higher rating compared to a win against a lower rated opponent. The same logic applies in case of defeat, the player's rating is decreased with an amount based on the opponents rating. Low rated players are players with an ELO score of sub 900, intermediate players have ratings of 1000-1500, high rated players have ratings of 1500-2200 and masters have a rating of 2200 to over 3000.

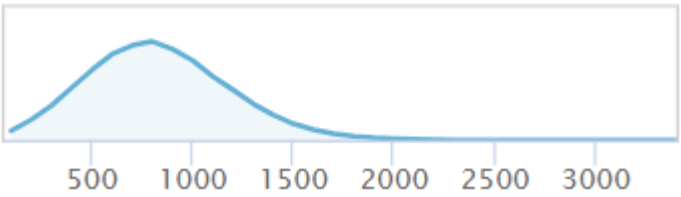


Figure 2.2: ELO distribution of chess.com[1]

A study by Grabner et al. investigated the individual differences between higher and lower rated players [15]. Their study showed that intelligence is related to chess expertise. More specifically, the skill to recognize patterns, think multiple moves ahead and memory are predictors for a strong chess player. However, high intelligence alone is not enough to become a strong chess player. According to Grabner et al., the main predictor of a player's chess expertise is their chess experience [15]. They showed that it accounted for approximately 25% of the skill variance of the players. They also found that a small percentage of the skill variance was caused by personality factors, such as motivation and emotion control. Similar results were found by Campitelli and Gobet when studying the importance of practice in chess [16]. They concluded that a higher daily amount of practice resulted in a higher rating. They also found that although masters and experts have similar daily practice, masters had a higher rating than expert players. This is most often the result of starting at a younger age, therefore building up more experience over the years.

Next to individual differences, there are major differences in chess play between differently rated players. In the book The Improving Chess Thinker, Heisman explains how to improve one's chess thinking process and describes the differences between the thought process of higher and lower rated chess players [17]. To begin, low rated players play "Hope Chess", a way of playing in which moves are made without considering the opponents counterplay. In addition, the players would often not consider tactics (a sequence of forced moves leading to a win of material) of the opponent or themselves. Winning material is chess terminology for "capturing opponents' pieces". Next to that, low rated players' analysis is inconsistent, non-systematic, and incomplete, often missing lines, captures, checks or attacks. Missing certain moves of the opponent results in false assumptions of a position, leading to mistakes. Moreover, low rated players have difficulty analysing moves on positional ground. For example, moves that do not win material are often not

[1] https://www.chess.com/leaderboard/live/rapid

considered, while they would make the player's position better. They lack the understanding of which trades, that do not change material count, are beneficial for the position.

High rated players tend to analyse games to a better degree, but masters arrive at more accurate conclusions among high rated players. High rated players also have a better understanding of possible threats of the opponent, strengths and weaknesses in positions and better time management. However, most high rated players could still improve by following the advice, “if you see a good move, look for a better one” [17].

All in all, there are multiple predictors for chess expertise. Chess expertise can be recognized by patterns, mistakes, and analysis in a chess game. The difference between weak and strong players is often clear, but between strong and master players, the differences become very subtle and harder to identify. The main differences are their capability to analyse board positions, play consistently and that low rated players play “Hope Chess”, while higher rated players do not.

2.4.2 Estimate chess expertise

Using the previously stated predictors and playing traits of different rating levels, chess expertise can be estimated. Research by Ferreira, showed that the Elo rating can be estimated using a chess engine [18]. A chess engine is a program that determines the best move, after looking ahead for a certain number of moves. The engine uses the values of the pieces (P: 1, N: 3, B: 3, T: 5, Q: 9) to give a move a certain score. If the move improves the position, the score increases and if the position degrades, the score decreases. Ferreira’s approach to estimate ELO ratings uses the gain or loss in the score, to determine how well the moves of the player are. With this information, the Elo rating is estimated.

Another approach was taken by Scheible and Schütze to predict chess player strength [21]. They looked at game annotations that players made for their own games and used machine learning to process this data. The game annotations consisted of the player’s own commentary on the games. From the game annotations, they found that low rated players had a short-term nature, focusing mostly on captures and attacks while missing long-term positional tactics. Next, low rated players often lack confidence, instead of using terms of confidence, “Know” or “Will”, terms with uncertainty were used, “Think”, “Believe”, “Maybe” or “Hoping”. This corresponds with the findings of Heisman , he defined this as “Hope Chess” [17]. Ultimately, certain terms used in a player’s game annotation indicated a rating. With these indications, the models succeeded in predicting the player’s rating.

A well-known and common method to estimate chess expertise is with the use of questionnaires. Often, the questionnaire consists of chess puzzles, finding the best move or analysing certain complex positions. Van der Maas and Wagenmakers use this approach in their Amsterdam Chess Test (ACT) [19]. The ACT estimates chess expertise through five tasks: a choose-a-move task, a motivation questionnaire, a predict-a-move task, a verbal knowledge questionnaire and a recall task. The ACT was proven to be a very reliable and valid test to predict chess expertise. Research by Junior and Thomaz used a chess questionnaire in another way [20]. They provided a chess questionnaire and analysed participants' eye movements. They found that expertise is consistently associated with the ability to process visual information. They propose that these findings could be used in predicting chess expertise. However, one drawback of using questionnaires to estimate chess expertise is the time constraint.

Lasty, multiple programs can be found online, which claim to be able to determine chess expertise or estimate Elo ratings. First, de Booy[2] developed a program that estimates a player's Elo rating based on one game. Like the method of Ferreira , a chess engine is used and the computer's moves are compared to the player's moves [18]. Similarly, Chess.com[3] developed a Computer Aggregated Precision Score (CAPS) to determine a player's rating. The CAPS system looks at four factors:

1. How many top moves (moves that matched the engine's top choice or were equal in score to that choice).
2. How many inaccuracies (a move that changes the position's evaluation slightly in the negative direction).
3. How many blunders (a move that changes the position's evaluation drastically in the negative direction).
4. Patterns of strength (their own algorithm that determines the sequencing of these scores per game timeline).

Using these factors, the system is able to calculate the CAPS of a player. Unlike the Elo rating, the CAPS is not dependent on other players or your opponent in particular.

The final two online Elo estimators are similar. The Elometer[4] and Chessmaniac Elo estimator[5] both use chess questionnaires to estimate the Elo rating of a player. Like one of the components in the ACT, a player is given a certain chess position, for which the player has to determine the best move [19]. Based on the moves made by the player, the Elo rating is estimated.

Many different approaches have been explored to determine a player's chess expertise. Using a chess engine or questionnaire seems to be the most effective. However, an approach that has not been tried yet is to use a large database of matches played by certain rated players. A player's moves could be compared to the most commonly played moves by players of a certain rating. This comparison could lead to an accurate estimation of a player's Elo rating.

2.5 Conclusion

To conclude, there are multiple process mining techniques, each with different advantages and disadvantages. When process mining is used to gain insights in a process, process discovery techniques are used. There are many different discovery algorithms, the most common being the alpha-algorithm, the fuzzy algorithm, heuristic mining, and inductive mining. When applying process mining techniques, the following challenges need to be considered: filtering out noise in data, dealing with incomplete data, choosing the right parameters and choosing the right tools and techniques. However, most common discovery algorithms have difficulty discovering complex processes, since complex processes have a high concurrency and many different tasks and transitions. Possible solutions to this problem are simplifying the process model and event logs or using other process discovery techniques. Simplifying the model or event logs could be done with UBeA, WoSimp or an improved Alpha mining technique. A promising discovery framework (which can be used for complex processes) is WoMan, which is proven to be effective in different complex cases including chess.

[2] https://www.debooy.eu/Java/caissatools_en.html
[3] https://www.chess.com/article/view/better-than-ratings-chess-com-s-new-caps-system
[4] http://www.elometer.net/
[5] http://www.chessmaniac.com/ELORating/ELO_Chess_Rating.shtml

In order to use process mining techniques to identify a player's expertise, differences between higher and lower rated players have been analysed, together with methods to associate this with the ratings. To begin with, research showed that experience is the biggest predictor of great chess performance, while the influence of intelligence is a smaller predictor, contradicting common belief. Gameplay of high rated and low rated players also show clear differences. Lower rated players often play, "Hope Chess", are worse at analysing the board state, have less insight in positional plays and often make short term moves. Higher rated players are better at analysing positions, regard possible opponent's counterplay before making moves and make more long-term moves.

These insights are used in various ways to determine the chess expertise of a player. A common way to determine chess expertise is by using a chess questionnaire, e.g., the ACT. Other methods are: using a chess engine and comparing the best moves to the moves played by the player or using the commentary of a player on their own game.

All in all, literature would suggest that it is possible to use process mining on chess event logs to identify a player's expertise. Literature shows that many techniques exist to use process mining on complex processes like chess. Moreover, there are analysis techniques to identify a player's expertise or examine the differences between high and low rated players. By combining the previously mentioned techniques, process mining can be used to identify the differences between novice and expert players.

Chapter 3 -Methods and Techniques

This chapter will explain the main framework and techniques used throughout this research. First, the main process mining framework will be briefly described and how certain sections correspond to process mining steps in the framework. Next, multiple process mining algorithms and plug-ins are shown and explained.

3.1 Process mining framework

For the process mining steps in this research, the process mining framework developed by van der Aalst is used, see Figure 3.1 [1]. The top of the diagram shows the external "world" from which data is obtained. In the case of this research, the external world is the online chess environment, explained more comprehensively in section 3.1.1. Information systems extract information from the chess environment and convert this information into event logs. Lichess and Chess.com are information systems, they store game logs of the games played on the platforms. Section 3.1.2 will go more in depth on these platforms.

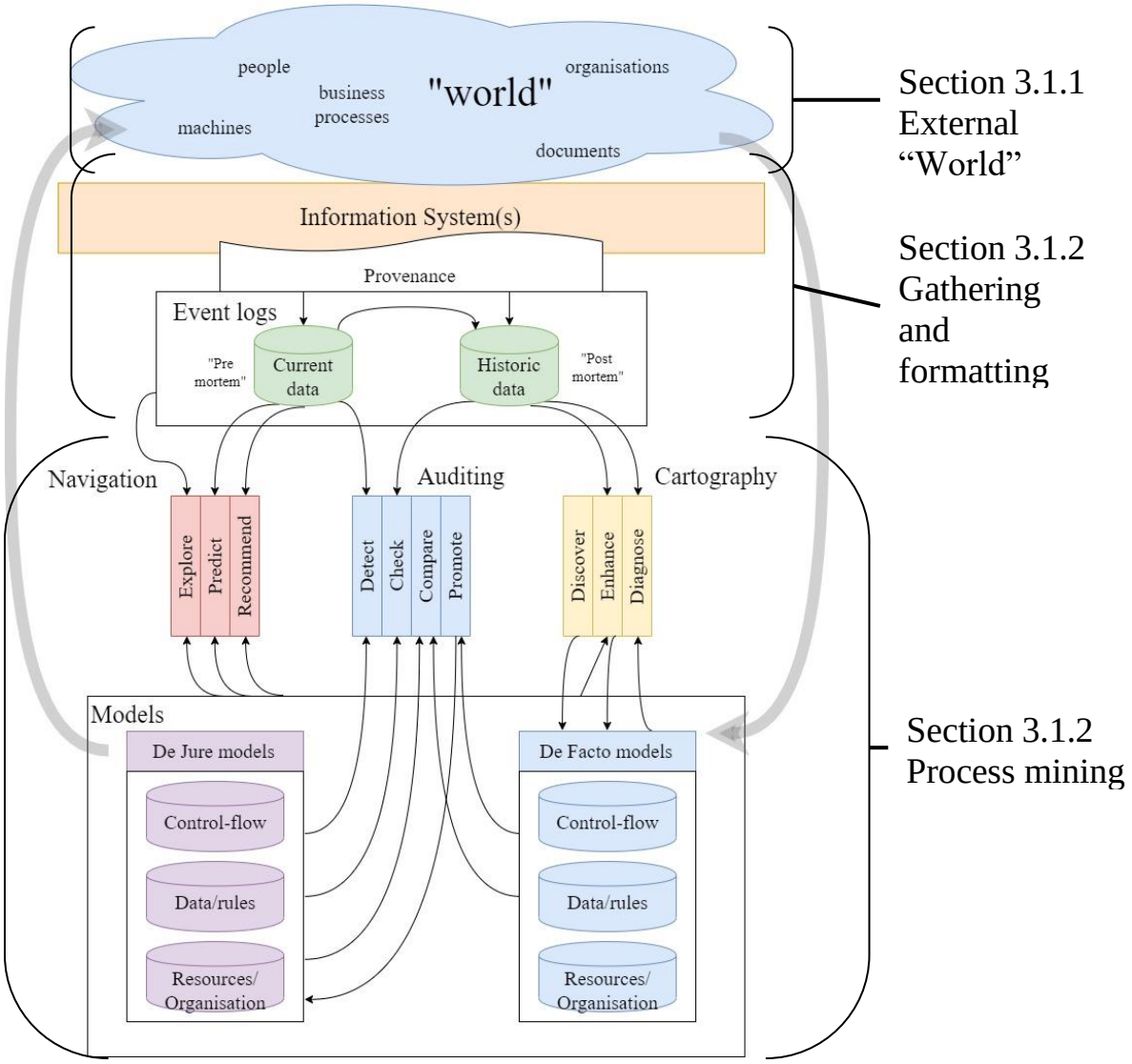


Figure 3.1: Process mining framework

Next, the diagram shows two types of event logs: current data (pre mortem) and historical data (post mortem). Current event data refers to cases that have not yet completed. Historical event data refers to cases that are completed. Event data from chess games used in this research is historical data since the games are completed and new event data is not simultaneously added.

The process mining framework distinguishes between two models: “de jure models” and “de facto models”. A de jure model indicates how certain events in a process should be done, i.e., it has influence on the workflow of a process. A de facto model is descriptive and identifies how the process works, it does not influence the workflow.

Between the event logs and models, the diagram depicts ten process mining related activities sorted in three categories: cartography, auditing, and navigation. The goal of the navigation category is improving a running process with the help of current data. The goal of the auditing category is to investigate a model, i.e., a de jure model is compared with event logs or a de facto model. Lastly, the goal of the cartography category is to make a process-map. A process-map could be a model that gives a clear overview of a process and its activities. In this research, the activities: compare and discover from the auditing and cartography categories respectively were used. The navigation category was not used since running processes were not used. Further on in this chapter, these activities will be thoroughly explained.

3.1.1 External “World”

Chess is one of the most well-known two player strategy games. The game is played on an 8x8 board with 6 different pieces that are able to move in different ways and are arranged as can be seen in Figure 3.2. The different pieces are:

King (K)	Can move one square in any direction,
Queen (Q)	Can move diagonally, horizontally, and vertically as many squares as possible,
Bishop (B)	Can move diagonally as many squares as possible,
Knight (N)	Can move two squares horizontally and one square vertically or two squares vertically and one square horizontally, it may also move through other pieces,
Rook (R)	Can move horizontally or vertically as many squares as possible,
Pawn (P)	Can move one square up from the players perspective, one square forwards diagonally when capturing another piece or two squares up when it is the first move of the pawn.

Figure 3.2: Basic chess board configuration

The goal of the game is to capture the opponent's king, called a checkmate. This is mainly done by winning material during the opening and the mid game to have an advantage in the end game. Capturing opponents' pieces can be done by placing your piece on a square that was occupied by an opponent's piece. Next to the movement of pieces, there are several rules in chess:

Check: When a king is under attack, it is called a "check". When the king is in check, the king must first be brought to safety, removing the check. When this is not possible, it is called "checkmate" and the player loses.
Castling: Casting is a principle to bring the king to safety. When the king and one of the rooks have not moved, the squares between these pieces are empty and the empty squares are not attack by the opponent, the king moves 2 (to the king's side) or 3 (to the queens' side) squares horizontally and the rook is placed one square next to the king, see Figure 3.3.

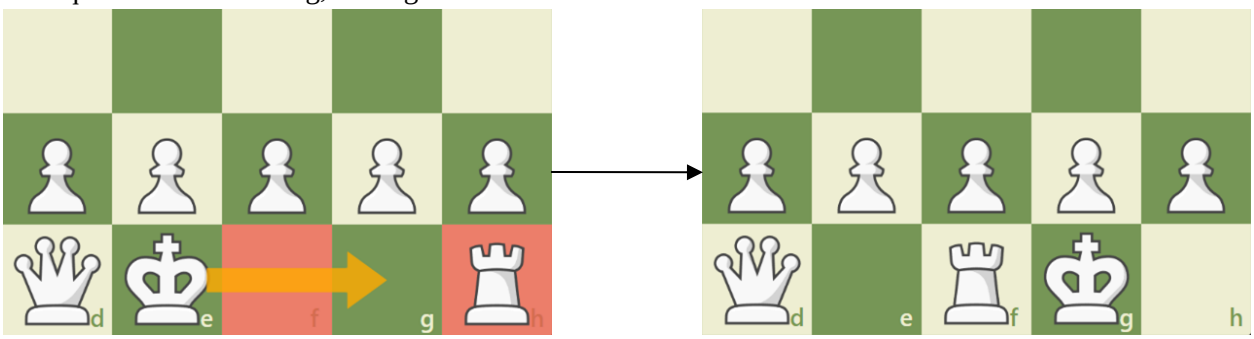


Figure 3.3: King side castling

Pawn promotion: When a pawn reaches the other side of the board, it will be promoted to another piece. The player who promotes the pawn may choose with which piece the pawn is replaced. Often, the player chooses to promote the pawn to a queen since this is the strongest piece, but scenarios exist where a knight is chosen. A pawn cannot be promoted to a king.
En Passant: Another important rule concerning pawns is "En passant". En passant is the only capturing move, where the capturing piece does not end on the captured piece's square. When a pawn moves two squares up, an opponent's pawn may capture the pawn as if the pawn moved only one square up, see Figure 3.4.
Dead position: A dead position is a board position in which neither player has any sequence of legal moves that leads to a win. For example, a king against a bishop and king is a dead position since neither player can win. A dead position results in a draw.

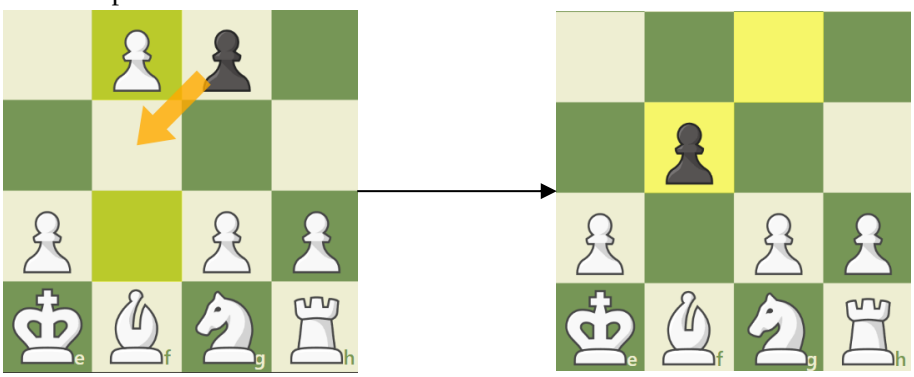


Figure 3.4: En passant

Stalemate: A stalemate is a position in which a player is not in check but does not have any legal moves. This position results in a draw.
Repetition of moves: If the same position occurs three times, a player may claim a draw.

3.1.2 Gathering and formatting data

In this research, many games of different players with different ratings were needed for process mining. These games were obtained via online chess platforms. Nowadays, many games are played online and are stored by the corresponding platform for the player to review, share or export. Currently, the biggest chess platforms are Chess.com and Lichess with approximately 30 million and 5 million players respectively. Every month, these platforms store millions of games, creating an immense amount of data to use. Lichess has an open database in which they post large files with PGN data open for any use. Chess.com does not have an open database in which monthly games can be downloaded. Instead, Chess.com allows a user to visit a player's profile, and download specific games from the player. Both platforms deliver the data in PGN format (portable game notation). PGN data contains one or more games with information about the players, the game and the moves. This research uses the games from Lichess of January 2014 which contains 697,600 games and is 100mb large. This file was chosen since it contained sufficient games and was not too large in size. The size of the file matters because of the extraction process, explained later in this section. Next to that, using games from 2014 does not change the end results since chess gameplay does not change significantly. The last big change in the chess gameplay occurred after the reigning world champion lost a chess match against a computer 20 years ago.

The PGN data from Lichess contained many games of players with various ratings. Only the high and low rated games had to be extracted to compare the two process models. PGN-Extract was used to extract games based on the players' ratings. To obtain the low rated games (LRGs) the rating cap was set at 975 and for the high rated games (HRGs) the floor was set at 2200. These values were chosen in order to get approximately the same number of games in the high and low rated files. Both files contained around 250 games.

To transform the gathered and sorted data into event logs, the data had to be converted from PGN format to CSV format. This conversion was done by Pgn2data, a Python library that converts chess PGN files to data tables. A shortened version of a CSV file generated by Pgn2data can be seen in Table 3.1. However, this CSV file contained a lot of unusable data and too many moves per game. Due to the complexity of chess, only the first twelve moves of every game were used in the process mining algorithms. To extract only the first twelve moves for both CSV files, a self-written script was used, which can be found in Appendix A. In addition, the script removed the unusable data and made a new CSV file containing: game_id, move_no, player, notation, move, from_square, to_square, piece and colour. Lastly, the CSV file produced by the self-written script was imported in ProM-lite. Using the plug-in, "Convert CSV to XES", the CSV file was converted to a XES file, a file type that is accessible for most process mining techniques.

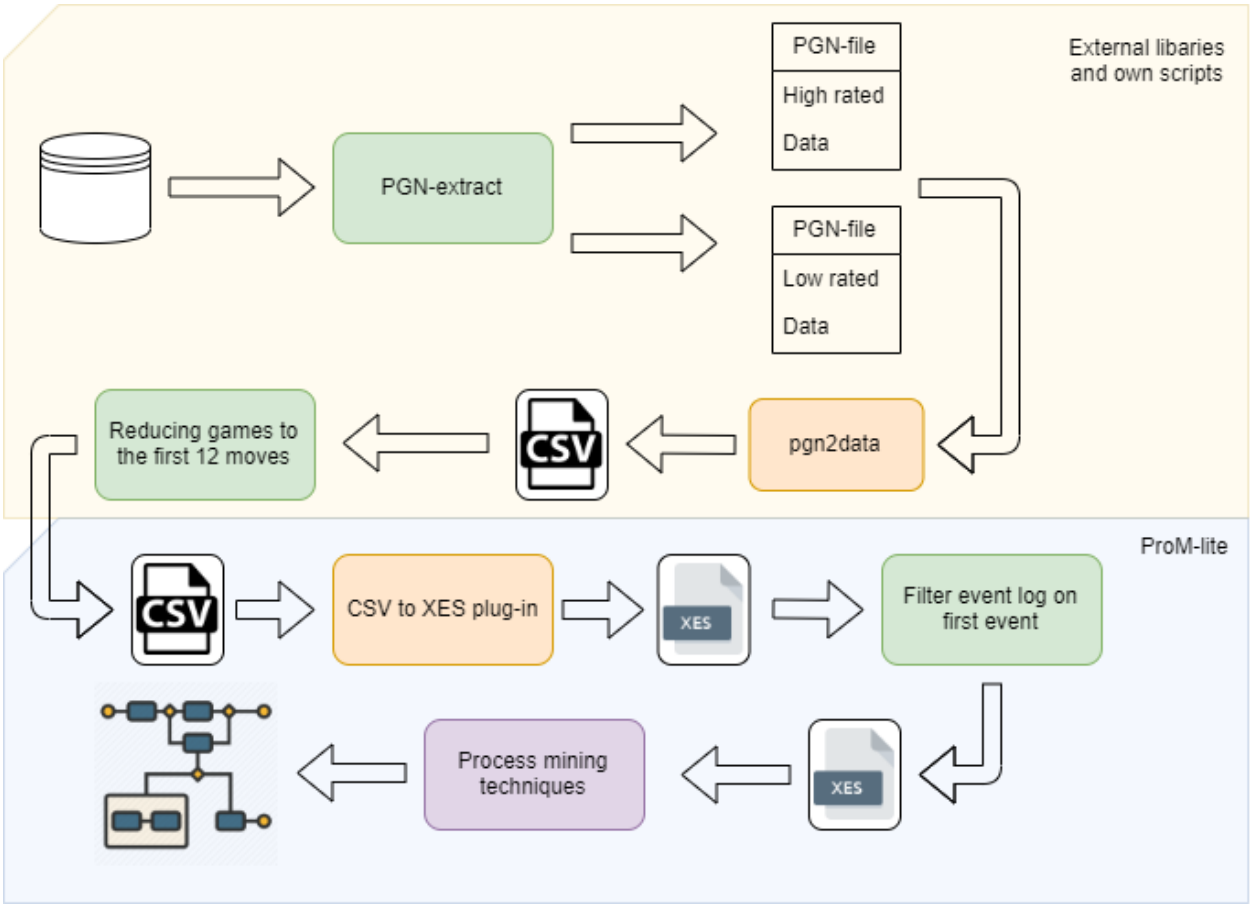


Figure 3.5: Data gathering and formatting model

game_id	move_no	player	notation	move	from_square	to_square	piece	color	fen
c3fee9a3-d381-4edc-ac8f-adbfada00343	1	P1	e4	e2e4	e2	e4	P	White	rnbqkbnr/pppppppp/8/8/4P3/8/PPPP1PPP/RNBQKBNR
c3fee9a3-d381-4edc-ac8f-adbfada00343	2	P2	d5	d7d5	d7	d5	P	Black	rnbqkbnr/ppp1pppp/8/3p4/4P3/8/PPPP1PPP/RNBQKBNR
c3fee9a3-d381-4edc-ac8f-adbfada00343	3	P1	exd5	e4d5	e4	d5	P	White	rnbqkbnr/ppp1pppp/8/3P4/8/8/PPPP1PPP/RNBQKBNR

c3fee9a3-d381-4edc-ac8f-adbfada00343	4	P2	Qxd5	d8d5	d8	d5	Q	Black	rnb1kbnr/ppp1pppp/8/3q4/8/8/PPPP1PPP/RNBQKBNR

Table 3.1: Shortened version of a chess event log

According to the Chess.com database[6], the two most popular first moves are e4 and d4. In the gathered data, these two moves were also the most popular first moves. To simplify the data once more, the event logs were filtered based on the first move. The event logs containing the high and low games were filtered separately, resulting in four event logs: HRGs starting with d4, HRGs starting with e4, LRGs starting with d4 and LRGs starting with e4.

Lastly, an approach to identifying chess expertise is by using conformance checking. Conformance checking requires a process model and event logs. The process models were created using the data which was gathered and formatted as explained earlier in this section. The event logs needed for conformance checking consisted of three games from high and low rated players. These games were extracted from another file in the Lichess database and were formatted using the same methods described earlier.

3.1.3 Process mining

As explained in the previous section, chess platforms store an immense amount of chess games and process mining techniques can utilize this data. In this research, process mining was used because of its capability to visualise patterns in large amounts of data. This unique feature of process mining is exactly needed to visualise the patterns in novice and experienced chess players.

The process mining was done with Prom-lite 1.3. Prom-lite is an extensive framework that supports a variety of plugins and process mining algorithms. ProM-lite is a derived version from ProM, as ProM-lite only included the most popular and typical packages from ProM in order to make the framework less consuming and clearer for new users. Before starting with process discovery, the event logs had to be in the correct format. The event logs consisted of a trace id, the game id and event parameters: from square, to square, piece and colour. In the following sections, the use of the three types of process mining is elaborated.

3.1.3.1 Process discovery

To identify the differences between the high and low rated gameplay, de facto models are made with techniques from the cartography and auditing categories. To begin, the discovery technique was used to discover process models from the event logs. Prom-lite has a lot of different discovery techniques as plug-ins. To experiment with the different techniques and examine their differences, artificial event logs were used. In the figures below, the different discovery techniques on the same data are depicted.

Figure 3.6 shows a Petri net generated with the Alpha miner plug-in. The Petri net clearly shows the sequence of events from left to right and where possible loops occur. This basic Petri net, however, does not show the frequency of certain events or transitions. Next to that, when discovering more complex processes, the Petri net becomes less clear due to the circles that indicate transitions. Overall, the Alpha

[6] https://www.chess.com/explorer

miner plug-in makes a model that is easy to understand but lacks information about frequency of events and the feature to modify certain parameters.

Figure 3.7 shows the fuzzy model generated by the “mine for a fuzzy model” plug-in. The model has two main ways to indicate the frequency of events and transitions. Every block contains an event name, state and a number indicating how many times this event occurred out of the total amount of traces. The significance and correlation metrics of the transitions of events are depicted with arrows of different sizes and shades of grey. In addition, the plug-in also provides various parameters for the user to experiment with. The node filter regulates how many events are shown based on the significance. When certain events have a significance below the cut-off, the events are clustered. The edge filter regulates which transitions between events are displayed based on the significance. When a transition's significance is below the cut-off, the transition is not shown. Lastly, the concurrency filter filters based on two parameters: preserve and ratio. Combining these filters enables many variations to make a model clearer or extract certain information from a model.

Figure 3.8 shows a model generated by the “mine with inductive visual miner” plug-in. The model has similar features to the fuzzy model to indicate frequency of events and transitions. The inductive visual model also shows the actual number of occurrences of the events or transitions. In addition, the plug-in has a visualisation feature in which the traces run through the model and bottlenecks can easily be detected. The plug-in provides two parameters that can be adjusted to show the right information in the model. The activities filter regulates how many events are shown in the model and the paths filter regulates the number of transitions. Both filters are based on the significance level of the corresponding parameter.

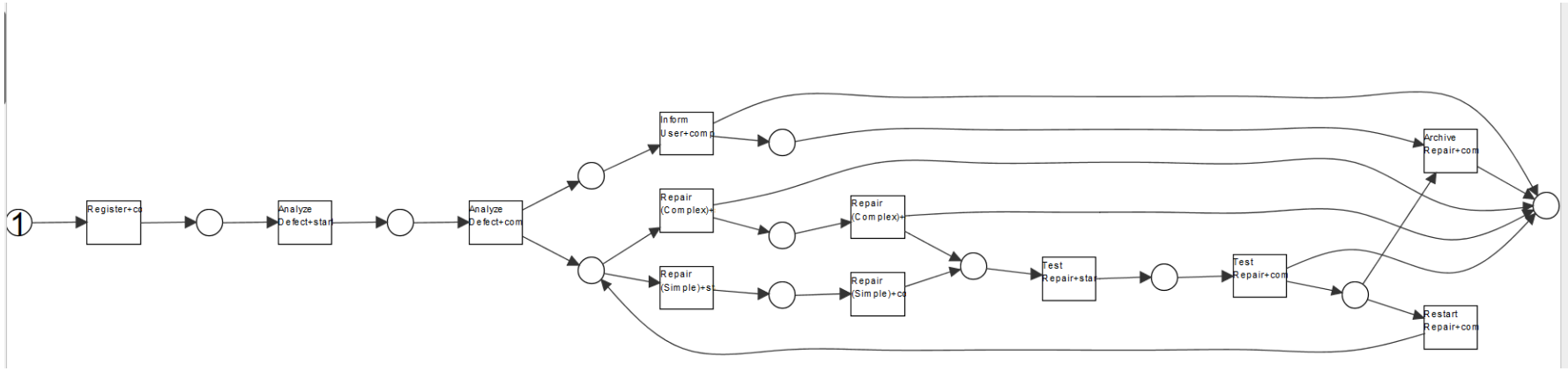


Figure 3.6: Simple Petri net generated by the alpha miner

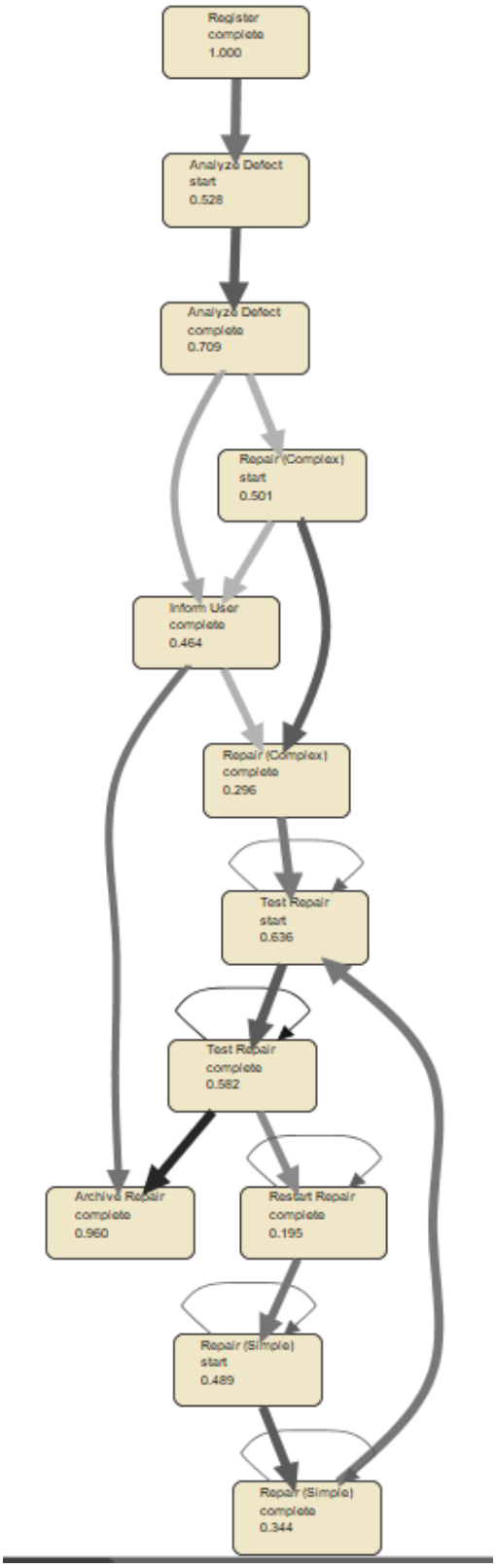


Figure 3.7: Fuzzy model generated by the fuzzy miner

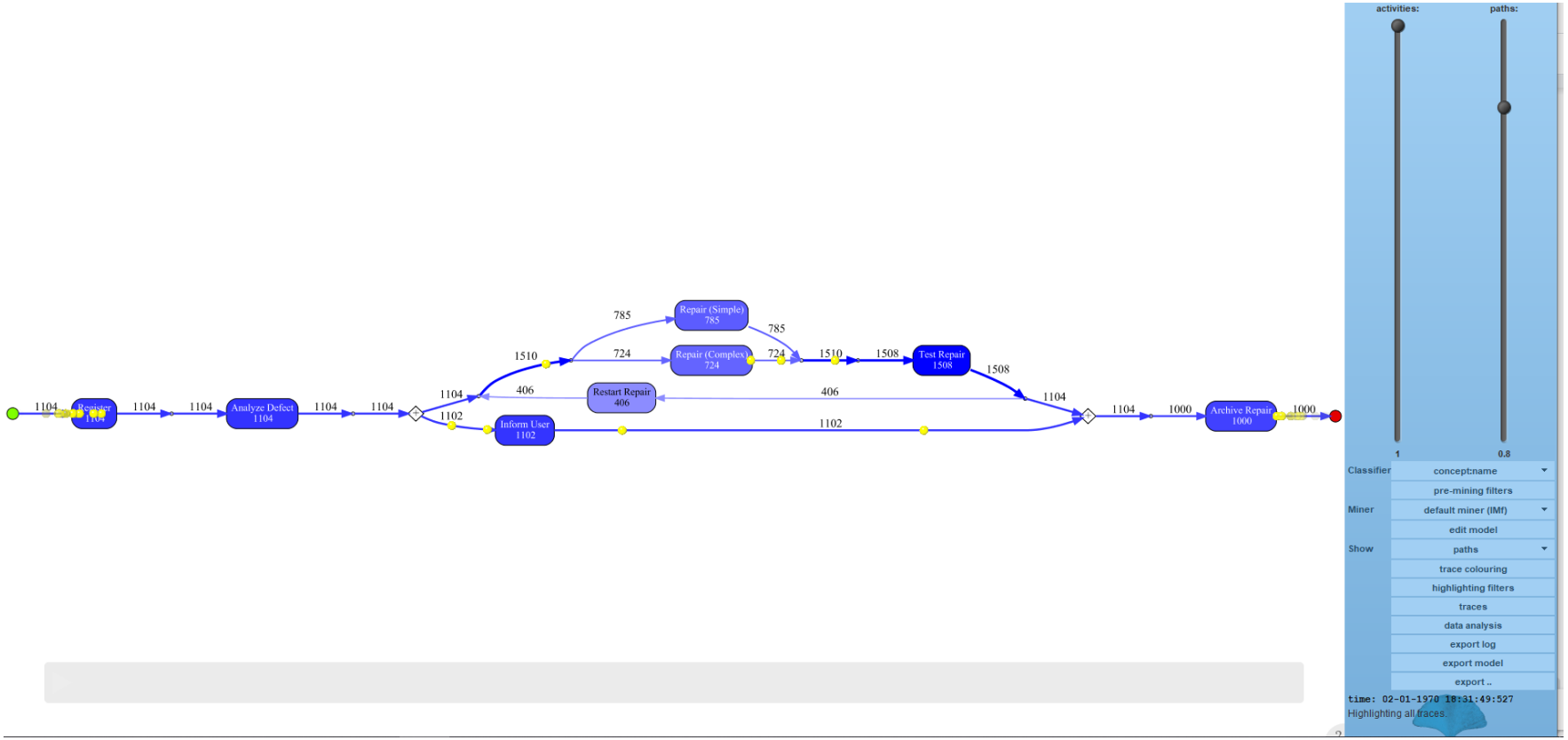


Figure 3.8: Visual model generated by the inductive miner

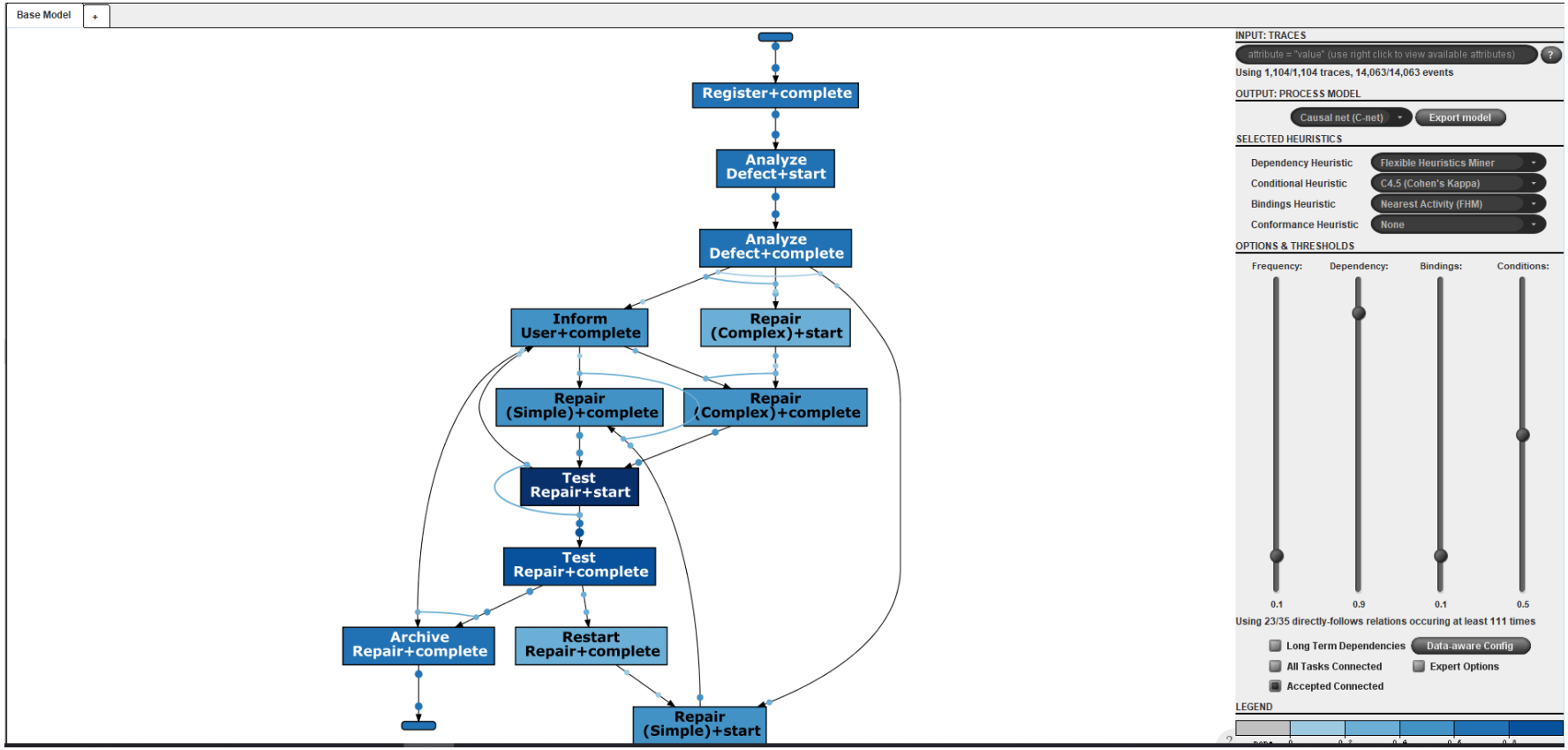


Figure 3.9: Causal net generated by the heuristic miner

Figure 3.9 shows a model generated by the “Interactive data-aware heuristic miner” plug-in. The model has different shades of blue to indicate the significance of the events. A darker shade of blue implies that the event occurs relatively often. Next to that, many transition arrows have lines connected to other arrows. This implies that there are more smaller transitions between the two events. This plug-in also provides a lot of options to visualise the right information in the model. In addition, the plug-in is able to output different process models that can be extracted and used in other plug-ins to analyse them further.

Lastly, the plug-in “process comparator” was used to compare two data sets. The plug-in generates a de facto model, but applies techniques from the categories auditing and cartography, diverging from the previous models as they only used cartography techniques. To specify, the model uses the compare technique from the auditing category and the discover technique from the cartography category. In the data gathering and extraction phase, two data sets were constructed, one containing the low rated games and one containing the high rated games. The process comparator develops a model from these event logs and indicates the differences between the two datasets with colour. The colour legend is based on Cohen's d, which is a type of effect size between two means. Effect size is a quantity of measure of the magnitude of the difference between two means. Cohen’s d takes the standardised mean difference. Since it is standardised, it is possible to compare values from groups with different variables and sizes. When events have a darker shade of a certain colour, the event occurs more frequently in one of the two event logs. The alpha significance level determines how much difference between the event logs is needed to get a certain shade. For example, with a high alpha value, only a small difference is needed to create a dark shade while models with a low alpha value need large differences before colours are added to the events. Lastly, the plug-in has a filter to reduce the number of events. The filter will remove infrequent events based on a threshold set by the user.

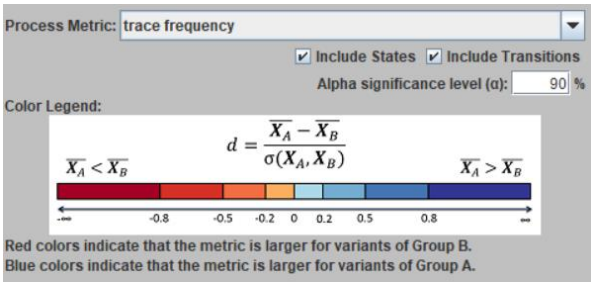


Figure 3.10: Legend of the process comparator

3.1.3.2 Conformance checking

The aim of conformance checking is testing the fitness of a process model. A set of traces is run through the model to see if the model is able to replay the traces. This technique was used to determine if a certain game conforms better with the model generated from HRGs or the model generated from the LRGs. The plugin “Precision of DPN” was used, which used a Data Petri Net (DPN) and an event log as input and outputted a precision result. The precision result consisted of the original DPN, a table with the transitions and the corresponding precision and various traces with observed and expected behaviour (see Figure 3.11). The colour coded DPN illustrates the precision of the transitions. Darker colours indicate a lower precision.

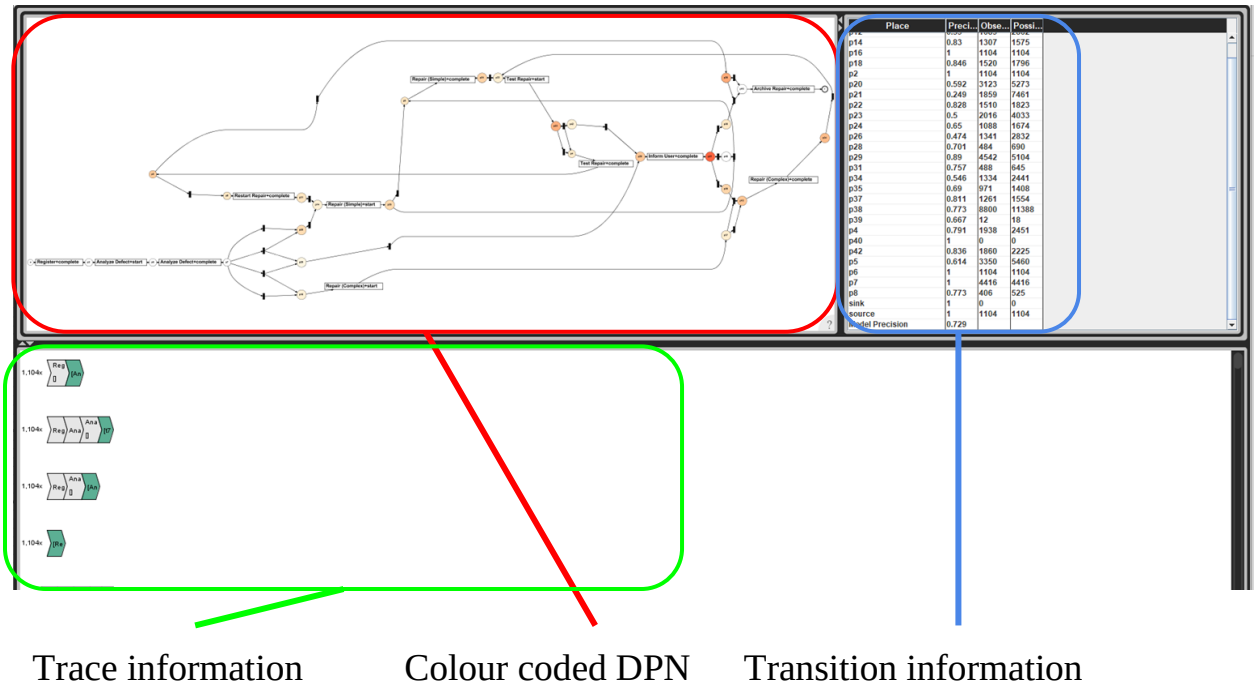


Figure 3.11: Example of the precision output

3.1.3.3 Process enhancement

Generated models by process discovery techniques can often be enhanced by utilizing additional attributes like timestamps, which is called process enhancement. During process enhancement, potential optimizations of a process model are explored with the help of the event logs. By optimizing the process model, ultimately the underlying process is optimised as well. Often the easiest method to enhance a process is to repair and optimize the event log. For example, event logs often contain missing data or identical values for different events which reduces the quality of a model. Process enhancement techniques analyse these inconsistencies in the event log and try to repair them.

ProM-lite provides plug-ins to repair event logs or add extra attributes to the event log. In this research, this type of process enhancement was not needed and therefore not used.

3.2 Concluding remarks

To summarise, the process mining framework developed by van der Aalst is used in [1]. The framework can be divided into the following three sections: the External “World”, Gathering and formatting of data and process mining. The external “World" section explains where the data originates from, i.e. the chess community, platforms and the game itself. The gathering and formatting of the data section contains the steps required to format the data in a proper way for process mining. Many iterations of simplifying and dividing the data were necessary to use the data in process mining algorithms. Lastly, the process mining section describes the three main process mining techniques.

Chapter 4 –Ideation

In this chapter, the ideation phase of this research is described. The ideation phase focuses on forming a creative idea to solve the problem. Accordingly, this chapter will show the process of forming an idea that would improve the chess Elo system, especially for new players. To begin with, the stakeholder analysis will be described, containing the identified stakeholders. Next, various techniques to identify chess expertise are explored. After that, a requirement analysis is described that shows the requirements of the research based on the technologies and stakeholders. To conclude, a final idea is stated as a solution to the problem.

4.1 Stakeholder analysis

A stakeholder analysis is performed to identify groups that have interest in or are influenced by this research. Also, the requirements of the stakeholders are identified and analysed to consider in the ideation process. The following main and sub stakeholders were identified:

The first main stakeholder is the *online chess player*. Online chess players play online on various chess platforms. Every player has an Elo rating indicating the player's strength. This research aims at augmenting the Elo system, therefore influencing the online chess players. Next to that, some of the players have interest in this research as well, because of unpleasant experiences with the current Elo system.

The second main stakeholder is the *online chess platform* itself. Improving the rating system of chess platforms initially influences the users. Consequently, the platform itself is influenced by the players. The platform wants to create the best experience for their users and therefore has a high interest in this research.

Assuming the success of this research, this technique can be applied in different disciplines to increase the user experience. As a result, *software developers and UX designers* also have an interest in this research.

Lastly, the *University of Twente* facilitates this research. This project is supervised by the University of Twente, they provide feedback, the assessment, and the tools to complete this research.

4.2 Acquisition of relevant information

In this section, various possible techniques to analyse chess expertise and how process mining techniques aid this are explained. It starts by stating three approaches to identifying chess expertise. Subsequently, the three process mining techniques are stated.

4.2.1 Analyse chess expertise techniques

There are multiple ways to analyse the chess expertise of a player during their game or with their game decisions. The various techniques are described below in this section, together with their benefits and limitations. The following techniques appear to be the most promising.

Chess engines are able to calculate more than 20 moves ahead, therefore being able to calculate the best move. The best move can be determined using heuristics such as material count and board position. A chess engine can provide every possible move with a heuristic score. A human player's move can be given the heuristic score, calculated by the chess engine. Based on the mean of the heuristic score of the player, their chess expertise can be estimated. As a result of the high accuracy of chess engines, this method also has a high accuracy. A disadvantage of this method is that it requires a lot of calculations for every move. However, the number of calculations can be decreased when the possible moves from the last calculation are stored. The engine already calculated many turns ahead, therefore not every move has to be calculated again.

Chess questionnaires provide great insight into the player's game knowledge. The questionnaire challenges a player to find the best move, see certain patterns or improve the board positions. This truly tests the player's insight in the game and therefore gives a great estimate of the player's expertise. Also, the player is presented with mostly unfamiliar positions and playstyles, hence removing the possibility for the player to be specialised in one playstyle. Chess questionnaires, however, are not applicable during a player's game itself. The player must do it separately and voluntarily, which is not desired by some players since it is time consuming. Next to that, chess questionnaires often do not have any time control, meaning that a player has a large amount of time to examine the question. A player who makes the best moves in a shorter time span is seen as a stronger chess player, the questionnaire does not consider this field of chess expertise.

Comparing chess data is a method that has not been tested yet. In theory, the moves from a player can be compared to two databases, one containing games of low rated players and one containing games from high rated players. Based on the resemblance with the two databases, a player's chess expertise can be estimated. For example, whenever a player plays moves that occur often in the high rated database, it can be assumed that the player is also high rated. Compared to the chess engine method, this method requires less calculations. This method, however, might become less accurate as the game proceeds. Chess has a tremendous amount of possible board positions, therefore after several turns specific board positions are most likely not in the database.

4.2.2 Process mining algorithms

Within the process mining domain, there are multiple groups of techniques that can be applied for similar purposes. In section 3.1.3, these techniques are described and explained. In this section, an overview is given of the plug-ins provided by Prom-Lite.

Process mining technique	Plug-in	Description
Process discovery	Alpha Miner	Algorithm that takes an event log as input and results in a Petri net with an initial marking
	Interactive Data-Aware Heuristic Miner	Tool that enables interactive exploration of heuristic process discovery
	Mine For A Fuzzy Model	Algorithm that takes an event log as input and results in a Fuzzy Model
	Mine With Inductive Visual Miner	Algorithm that takes an event log as input and results in a process tree model
	Process Comparator	Algorithm that compares a collection of event logs and results in a process model with a color scheme
Conformance Checking	Conformance Checking of DPN	Conformance checking of a Petri net with Data with regard to an event log
	Precision of DPN	Calculate the data-aware precision of a Petri net with Data, given an event log
	Replay a Log on Petri Net	Replay an event log on a Petri net to check conformance
Process Enhancement	Repair Log with Respect to Alignment	Repairs an event log by using the alignments of replay and the model itself

Table 4.1: Overview of process mining plug-ins

4.3 The concept

Regarding the previously mentioned techniques and stakeholders, two main concepts are created. Both concepts are based on two datasets, one consisting of HRGs and one of LRGs.

Process Comparator

To analyse the differences between high and low rated players, an event log with HRGs and an event log with LRGs will be produced. These will both be put in the process comparator, which outputs a process model. In this model, the differences between the two event logs are highlighted with a color code. The

model illustrates the good and bad sequences of moves, which can be compared to the moves of a player. When these moves correspond to the bad sequences, the player is more likely to be an inexperienced player and vice versa.

Conformance checking

Conformance checking can be used to estimate a player's chess expertise, by comparing his or her games with data. Two event logs will be made, one with HRGs and one with LRGs. From these event logs, separate models will be made. Ultimately, a player's games will be conformance checked with the models. The games of the player will correspond more to one of the models, which can be used to estimate the expertise of a player.

Chapter 5 Data analytics

In this chapter, the realisation of the research is stated. It starts with elaborating the decisions made during the specification of the data. Afterwards, the results of the process mining algorithms are shown. The models are presented and notable information in the models is indicated.

5.1 Pre-processing of data

For the experiments, chess games were downloaded from Lichess and filtered and converted to usable event logs. Filtering and converting the data was an iterative process where many decisions were made to get the best process mining results. As stated before, chess can be seen as a complex process. To extract useful information from complex processes, the data first needs to be simplified to eventually get a clearer process model. Figure 5.1 shows a process model generated by the fuzzy miner algorithm from event logs that are not filtered. Because of the large number of transitions, a spaghetti model is created that makes it hard to understand the process. Next to that, the algorithms took over 30 minutes to generate a model which made it hard and time consuming to adjust the model.

The first step to reduce the complexity was to reduce the number of games in the event logs. From the larger downloaded dataset, highest rated games and the lowest rated games were stored in separate datasets. The dataset with low rated games contained 230 games and the dataset with high rated games contained 256 games. This number of games per data set is mainly chosen because Ferelli and Angelastro used similar amounts and got significant results and were able to extract information about frequently played moves [26].

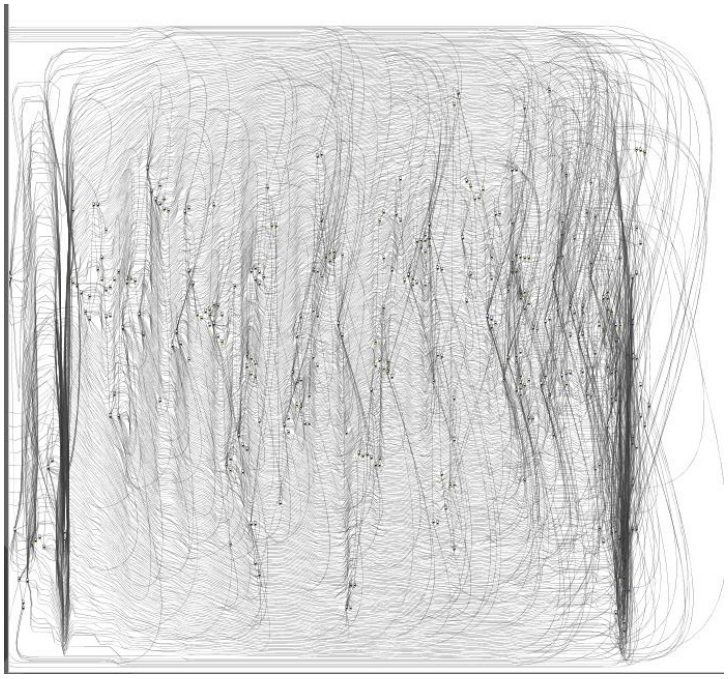


Figure 5.1: Fuzzy model from 50 unfiltered games

The event logs used during this research consisted of the game id as trace id and the event attributes: from square, to square, piece and player colour. The game id is chosen as a trace id to compare different games to each other. Every game is a trace with moves as events. The event attributes, from square and to square, are included into the event to show the movement of a piece. A common method to show the movement of pieces is with chess notation, i.e., knight moves to f3 is noted as Nf3. This method is not used since it does not show where the piece came from, and therefore many different moves can have the same notation. The colour of the piece is included in the events to indicate which player moved the piece. This creates a better overview of what responses players make on certain moves.

Another approach to simplify the event logs was to reduce the amount of turns per game. The length of chess games varies largely from 4 to over 100 turns. Considering the number of possible moves in chess and the large number of turns, the variation in events would be enormous. In these turns many of the same moves are made in completely different and unrelated positions. For example, a certain move might be good in the early stages of the game while bad in later stages of the game. To reduce the number of identical events in different positions and simultaneously reduce the complexity of the model, only the opening of the game is covered in the event logs. The first twelve moves are chosen since the opening often consists of moving the knights, bishops and two pawns from their original positions. This can be achieved in twelve moves. For these twelve moves, the opening is chosen since it always has the same starting position. While the end game has fewer pieces and is therefore easier to analyse, there is also a high diversity of endgames which makes it hard to find equivalences between different games. The mid game is not chosen because of the extraordinary number of possible positions, which would make the process model too complex.

Lastly, the event logs are filtered on the starting event. As stated before, moving a pawn to d4 or e4 are the most common first moves. Therefore, the event logs are filtered and divided based on the first move. Only two starting moves are chosen because adding more would not provide significant results, since these opening moves are played less frequently, less data is present in the datasets from these moves.

5.2 Process Mining results

In this section, the generated process models are illustrated and outstanding patterns or events are addressed. The process models were generated to analyse their capabilities in identifying chess expertise (research question 3). First, the process discovery results are presented. Second, the results of the conformance checking are shown. Lastly, the process comparator results are presented.

5.2.1 Process discovery results

The process models shown below are generated by the “Interactive data-aware heuristic miner” plug-in. In total, there are four models: LRGs starting with d4, LRGs starting with e4, HRGs starting with d4 and HRGs starting with e4. The first noticeable difference is the complexity of the LRG model compared to the HRG model. The LRG models have significantly more transitions and events. Notably, the LRG d4 model has the appearance of a spaghetti model, which makes it hard to analyse due to the lack of structure.

In Figure 5.2, the models with e4 are illustrated next to each other. The HRG model shows a high frequency of c7|c5|P|Black, while this event does not appear in the LRG model. Similarly, a frequent event following the start event is d7|d5|P|Black. This event does not appear consecutively to the start event in the HRG model. Both models contain different events and sequences of events. Lastly, the LRG model has

larger and more blue lines between the transition arrows, indicating a higher variance of transitions between events.

Something that can be seen in all four models is that certain events are absent in certain sequences. For example, in the HRG model in figure 5.2, the sequence: [e2, e4, P, White] - [c7-c6-P-Black] - [d2-d4-P-White] - [c5-d4-P-Black]. The last event indicates that the black pawn on c5 takes a white pawn on d4. However, when following the start of the sequence, the black pawn never entered the c5 square and therefore made this sequence impossible. As a result of certain events not occurring frequently enough in the event log, the event is omitted from the model. Similarly, because certain events are omitted, the model occasionally links two events from the same colour to each other. This again would not be possible since it is a turn-based game and only one move can be made in a turn.

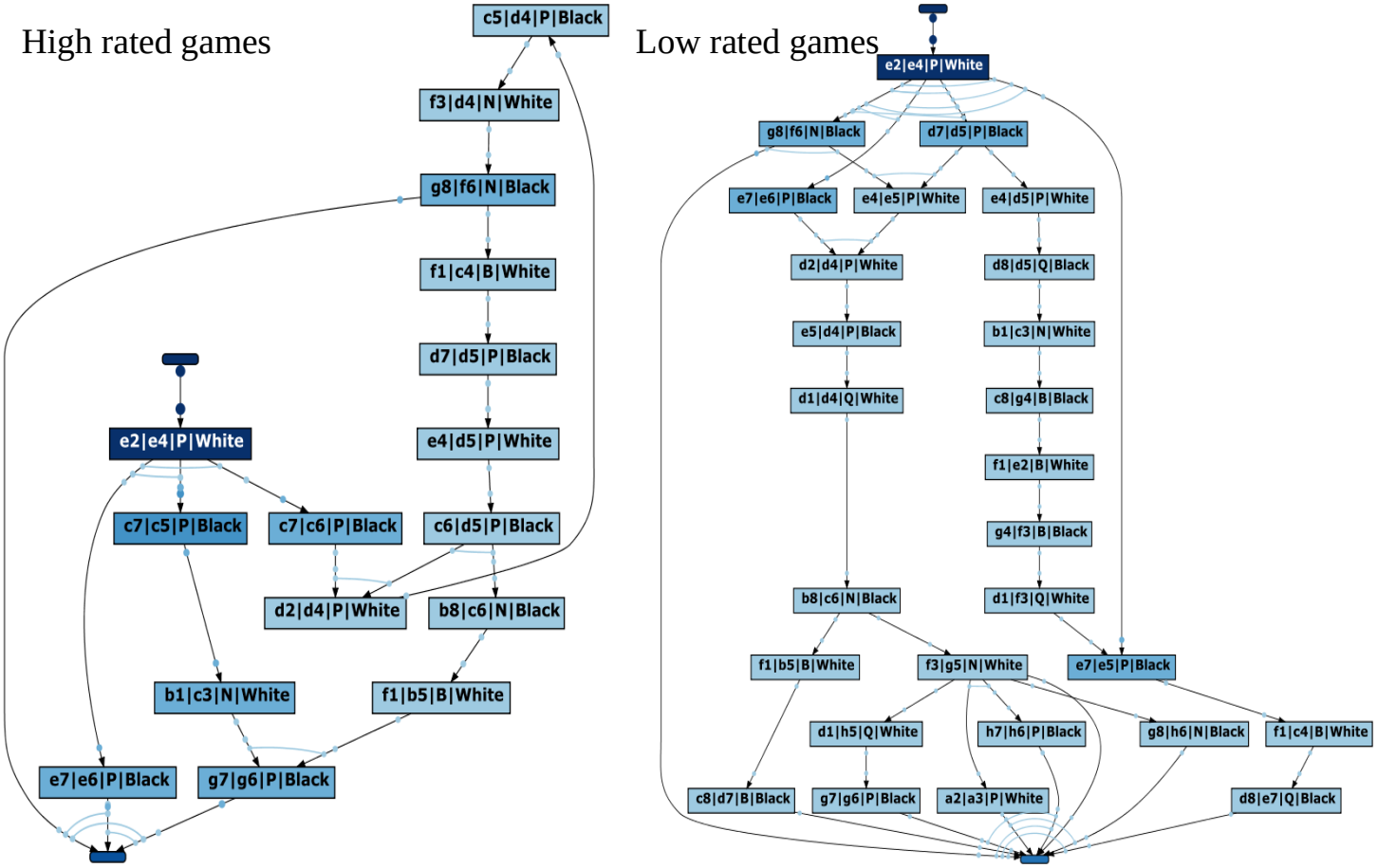


Figure 5.2: HRG model and LRG model next to each other

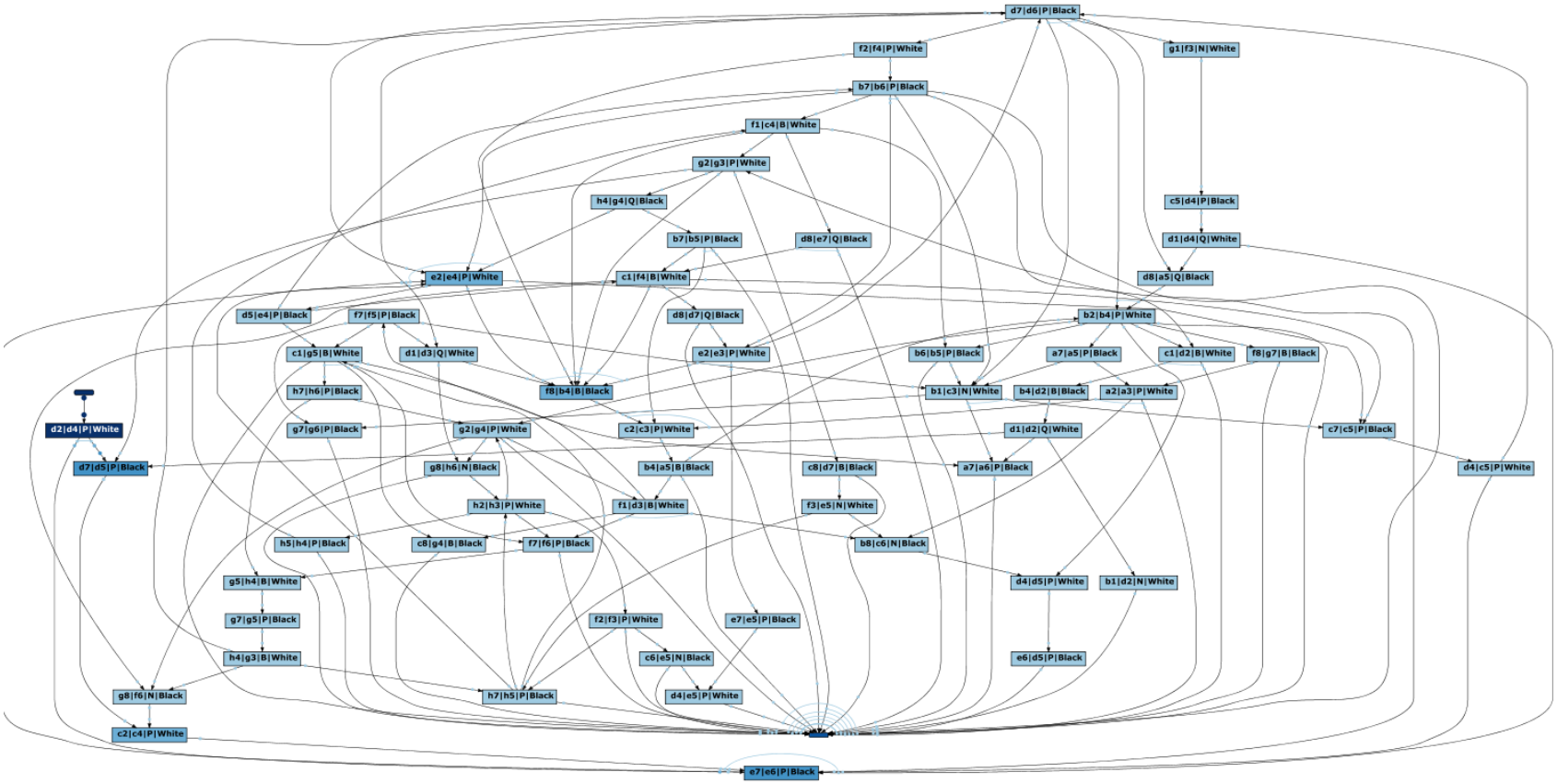


Figure 5.3: LRG model with the starting move d4

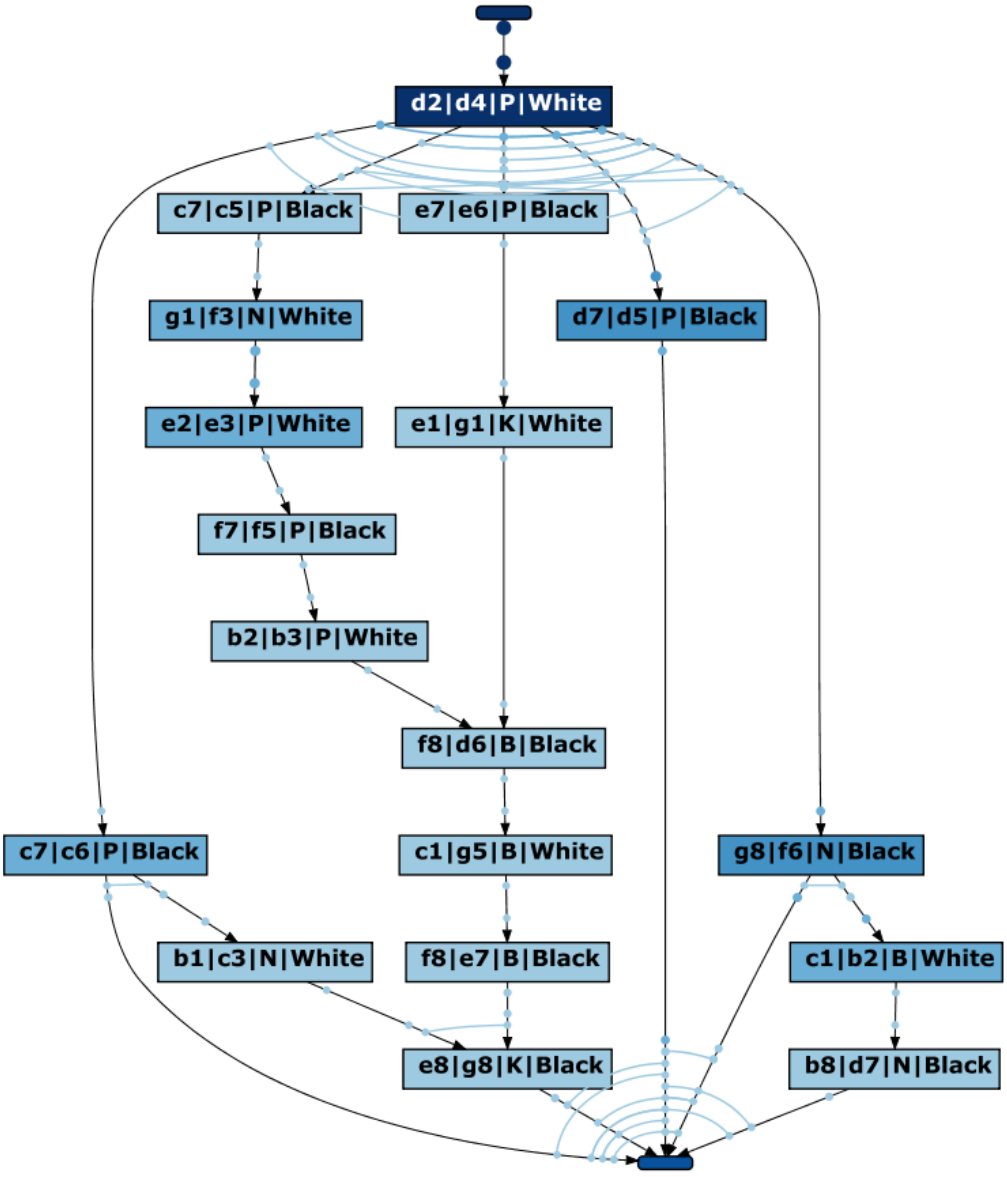


Figure 5.4: HRG model with the starting move d4

5.2.2 Conformance checking results

In the figures 5.5-5.8 below, the conformance checking results are illustrated. A HRG and LRG Petri Net are both checked with an event log containing three HRGs and an event log containing three LRGs. In Table 5.1, the precision scores of the conformance checking can be found. The conformance checking technique was used to explore to what extent this technique can aid in identifying a player's chess expertise.

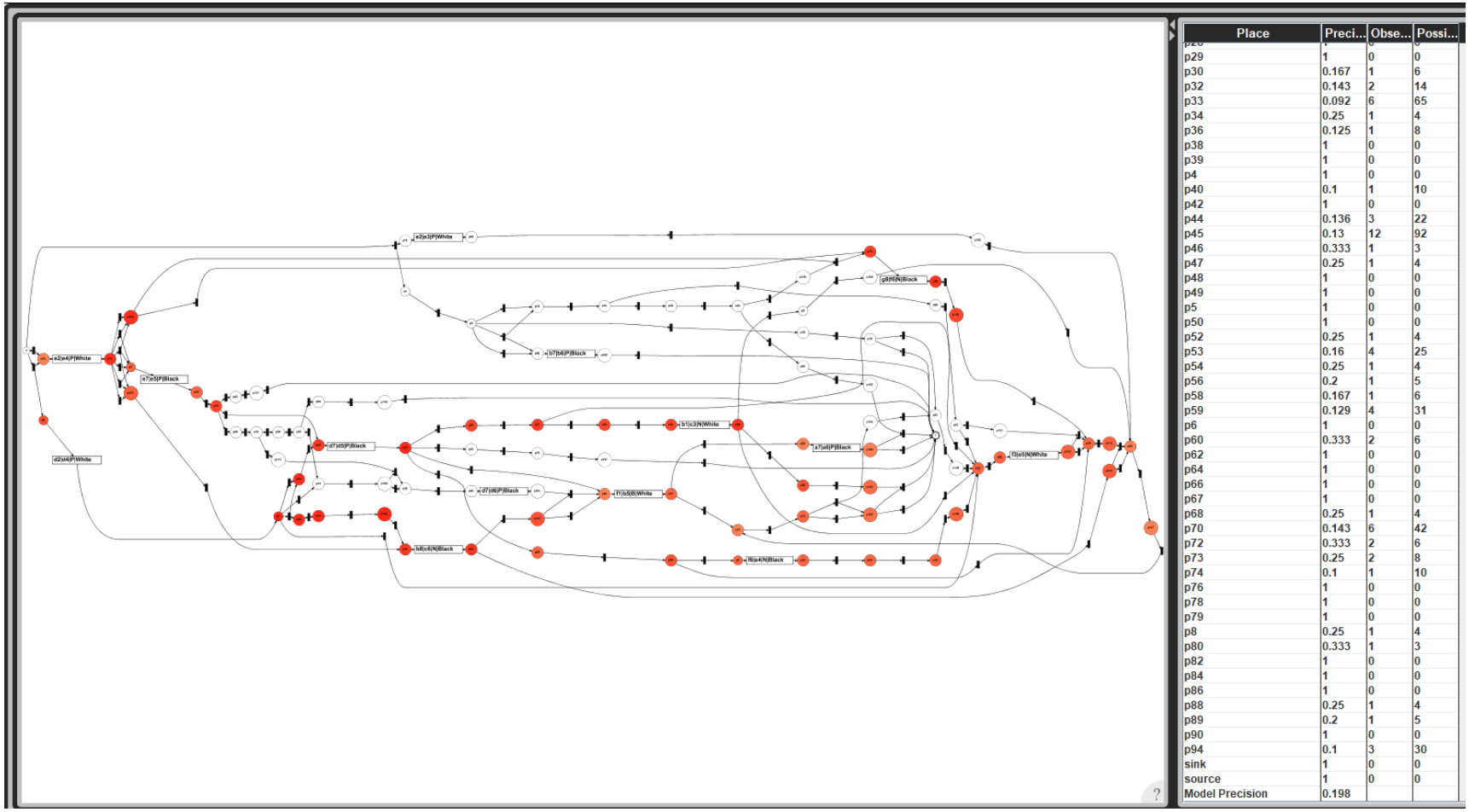
In the conformance checking models, the original Petri net is displayed with an additional colour code. This colour code indicates which transitions and events could not be replayed in the model. On the right side of the conformance checking results, a table is displayed which shows all the separate transitions and events with the appropriate precision score. These transition or event specific precision scores, however, are not important in this research.

The overall precision score of a model indicates how well the event logs could be replayed by the model. Therefore, a high precision score indicates that the events and transitions in the event log are similar to a certain sequence of events and transitions in the model. When running a HRG through a HRGs model, it is expected that the precision would be high and when running a LRG through a HRGs model, it is expected that the precision is low.

In figures 5.5-5.8, the generated conformance checking models are displayed. The LRGs model has a lower precision score in both cases than the HRGs models, also revealed by larger amounts of red shades in the LRGs model.

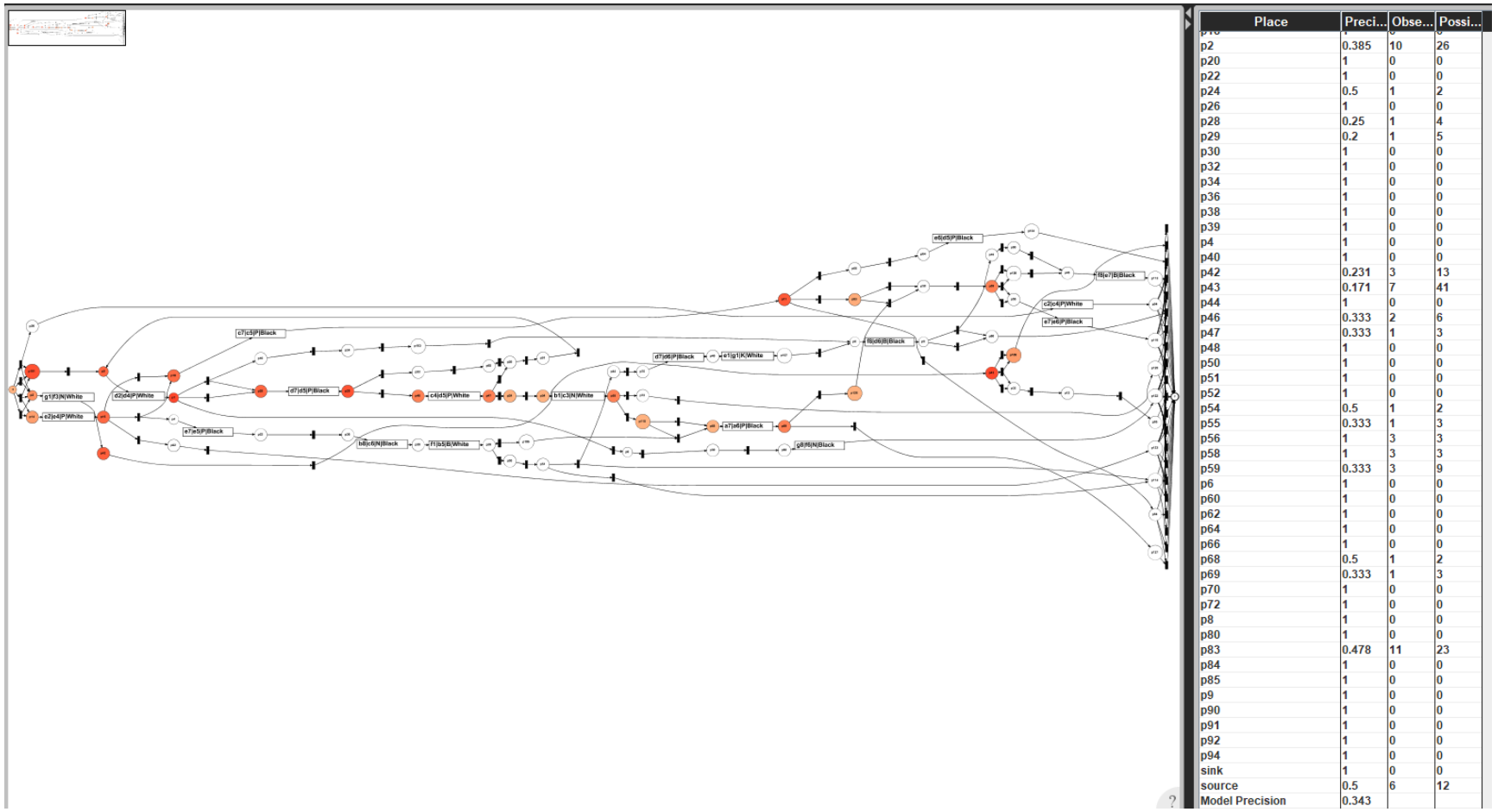
	HRG model	LRG model
High rated games	0.343	0.198
Low rated games	0.337	0.182

Table 5.1: Precision scores for the models and event log combinations



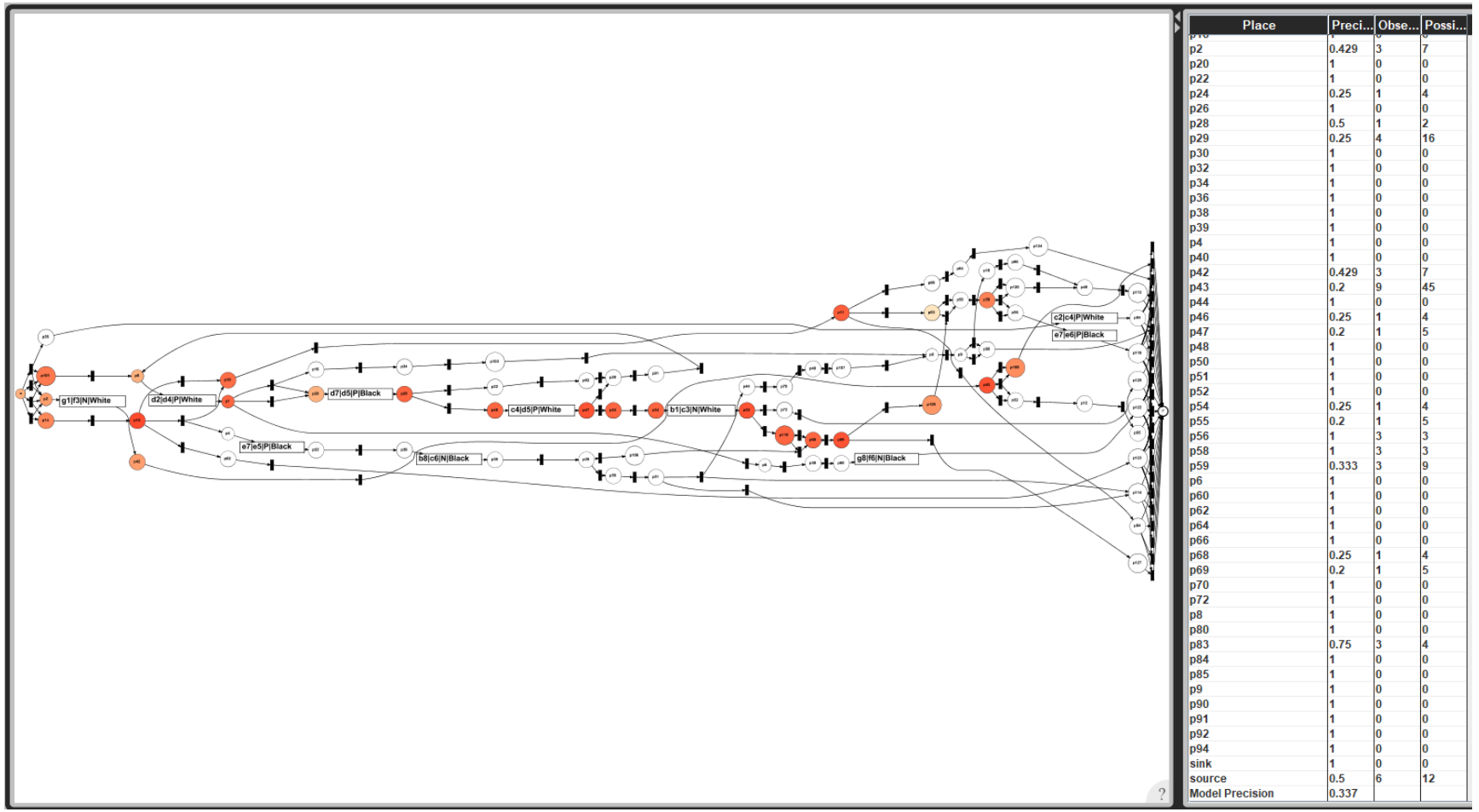
Place	Preci...	Obse...	Possi...
p29	1	0	0
p30	0.167	1	6
p32	0.143	2	14
p33	0.092	6	65
p34	0.25	1	4
p36	0.125	1	8
p38	1	0	0
p39	1	0	0
p4	1	0	0
p40	0.1	1	10
p42	1	0	0
p44	0.136	3	22
p45	0.13	12	92
p46	0.333	1	3
p47	0.25	1	4
p48	1	0	0
p49	1	0	0
p5	1	0	0
p50	1	0	0
p52	0.25	1	4
p53	0.16	4	25
p54	0.25	1	4
p56	0.2	1	5
p58	0.167	1	6
p59	0.129	4	31
p6	1	0	0
p60	0.333	2	6
p62	1	0	0
p64	1	0	0
p66	1	0	0
p67	1	0	0
p68	0.25	1	4
p70	0.143	6	42
p72	0.333	2	6
p73	0.25	2	8
p74	0.1	1	10
p76	1	0	0
p78	1	0	0
p79	1	0	0
p8	0.25	1	4
p80	0.333	1	3
p82	1	0	0
p84	1	0	0
p86	1	0	0
p88	0.25	1	4
p89	0.2	1	5
p90	1	0	0
p94	0.1	3	30
sink	1	0	0
source	1	0	0
Model Precision	0.198		

Figure 5.5: Conformance checking of the LRG model with high rated games, model precision: 0.198



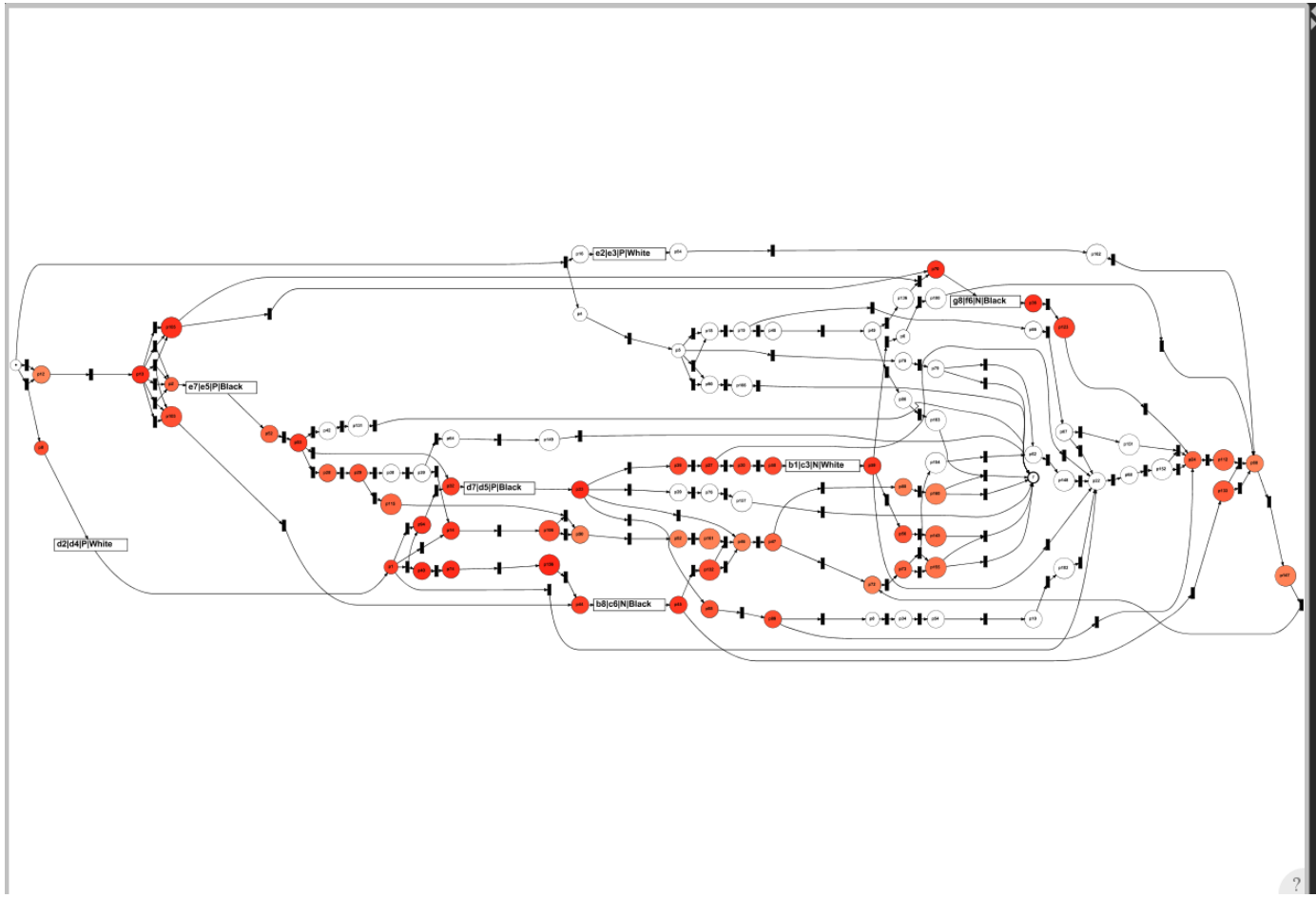
Place	Preci...	Obse...	Possi...
p2	0.385	10	26
p20	1	0	0
p22	1	0	0
p24	0.5	1	2
p26	1	0	0
p28	0.25	1	4
p29	0.2	1	5
p30	1	0	0
p32	1	0	0
p34	1	0	0
p36	1	0	0
p38	1	0	0
p39	1	0	0
p4	1	0	0
p40	1	0	0
p42	0.231	3	13
p43	0.171	7	41
p44	1	0	0
p46	0.333	2	6
p47	0.333	1	3
p48	1	0	0
p50	1	0	0
p51	1	0	0
p52	1	0	0
p54	0.5	1	2
p55	0.333	1	3
p56	1	3	3
p58	1	3	3
p59	0.333	3	9
p6	1	0	0
p60	1	0	0
p62	1	0	0
p64	1	0	0
p66	1	0	0
p68	0.5	1	2
p69	0.333	1	3
p70	1	0	0
p72	1	0	0
p8	1	0	0
p80	1	0	0
p83	0.478	11	23
p84	1	0	0
p85	1	0	0
p9	1	0	0
p90	1	0	0
p91	1	0	0
p92	1	0	0
p94	1	0	0
sink	1	0	0
source	0.5	6	12
Model Precision	0.343		

Figure 5.6: Conformance checking of the HRG model with high rated games, model precision: 0.343



Place	Preci...	Obse...	Possi...
p2	0.429	3	7
p20	1	0	0
p22	1	0	0
p24	0.25	1	4
p26	1	0	0
p28	0.5	1	2
p29	0.25	4	16
p30	1	0	0
p32	1	0	0
p34	1	0	0
p36	1	0	0
p38	1	0	0
p39	1	0	0
p4	1	0	0
p40	1	0	0
p42	0.429	3	7
p43	0.2	9	45
p44	1	0	0
p46	0.25	1	4
p47	0.2	1	5
p48	1	0	0
p50	1	0	0
p51	1	0	0
p52	1	0	0
p54	0.25	1	4
p55	0.2	1	5
p56	1	3	3
p58	1	3	3
p59	0.333	3	9
p6	1	0	0
p60	1	0	0
p62	1	0	0
p64	1	0	0
p66	1	0	0
p68	0.25	1	4
p69	0.2	1	5
p70	1	0	0
p72	1	0	0
p8	1	0	0
p80	1	0	0
p83	0.75	3	4
p84	1	0	0
p85	1	0	0
p9	1	0	0
p90	1	0	0
p91	1	0	0
p92	1	0	0
p94	1	0	0
sink	1	0	0
source	0.5	6	12
Model Precision	0.337		

Figure 5.7: Conformance checking of the HRG model with low rated games, model precision: 0.337



Place	Preci...	Obse...	Possi...
p29	0.2	1	5
p30	0.167	1	6
p32	0.143	2	14
p33	0.099	9	91
p34	1	0	0
p36	0.111	1	9
p38	1	0	0
p39	1	0	0
p4	1	0	0
p40	0.1	1	10
p42	1	0	0
p44	0.146	6	41
p45	0.125	17	136
p46	0.333	1	3
p47	0.25	1	4
p48	1	0	0
p49	1	0	0
p5	1	0	0
p50	0.333	1	3
p52	0.25	1	4
p53	0.135	7	52
p54	1	0	0
p56	0.2	1	5
p58	0.167	1	6
p59	0.133	2	15
p6	1	0	0
p60	0.333	2	6
p62	1	0	0
p64	1	0	0
p66	1	0	0
p67	1	0	0
p68	1	0	0
p70	0.111	1	9
p72	0.333	2	6
p73	0.25	2	8
p74	0.1	1	10
p76	1	0	0
p78	1	0	0
p79	1	0	0
p8	1	0	0
p80	0.333	1	3
p82	0.333	1	3
p84	1	0	0
p86	1	0	0
p88	0.2	1	5
p89	0.182	2	11
p90	1	0	0
p94	0.118	4	34
sink	1	0	0
source	1	0	0
Model Precision	0.182		

Figure 5.8: Conformance checking of the LRG model with low rated games, model precision: 0.182

5.2.3 Process comparator results

In Figures 5.10 and 5.11, the results of the process comparator are displayed. The process comparator compares the two inputted event logs by making a model and providing it with a certain colour code. Figure 5.9 shows the legend for this colour code, with the red shades indicating a higher frequency in the HRGs and blue shades a higher frequency in the LRGs. Both events and transitions are colour coded, therefore also providing information on which specific transitions occurred in which event logs. White events and black transition arrows indicate that the event or transition occurs for similar amounts in HRGs and LRGs event logs. Lastly, every event has a certain border width, which indicates the overall frequency of this event. A larger border width indicates a higher overall appearance of this event in the event logs.

The first notable thing is that the majority of events have a red shade, indicating that these events occur more frequently in the HRG event logs. Also, the model only contains one sequence of events with a blue shade, while there are many with a red shade and the majority of the transition arrows have a red shade. Moreover, all but one of the events with a knight move are coloured red, indicating that early knight moves are valuable among the high rated players. Besides the pawns, the knight is the most easy and flexible piece to move in the beginning of the game since it can jump over other pieces. Next to that, some events have a white colour. A white colour indicates that the event occurs a similar number of times in the HRGs and LRGs event log.

Another outstanding phenomenon regarding the events is that some events do not have any transition arrows connecting them to other events. For example, the event [c8, g4, B, Black] which can be seen at the top of Figure 5.10, has no apparent connection to the start event or any other event. This unconnected event occurs frequently enough in the event logs, but has many different transitions which are filtered out of the eventual model. Next to that, some events have transition arrows that go back and forth. In chess, this is impossible since the same move cannot be made two times in succession. This also shows that a lot of sequences in the model are not viable.

Figure 5.10 shows the process comparison model starting with the move e4. Noticeable, the event c7|c5|P|Black has a dark shade of red. This corresponds to the results from the process discovery section, in which the LRG model did not display that event. This event represents the move pawn to c5, which is the start of the Sicilian defence. The Sicilian defence is a common counter to the starting move e4, as it defends the centre of the board while not yet moving the centre pawns. Another interesting event is e1|g1|K|White, which is a castling move. This event frequently occurs in the HRG, but not in the LRG.

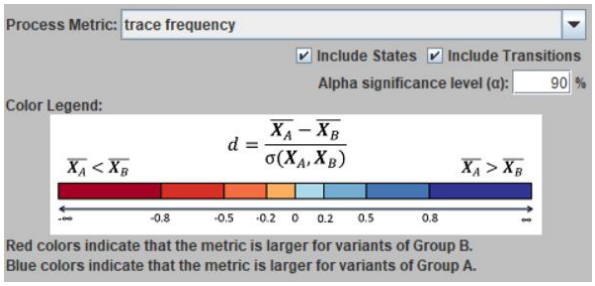


Figure 5.9: Legend of the process comparator

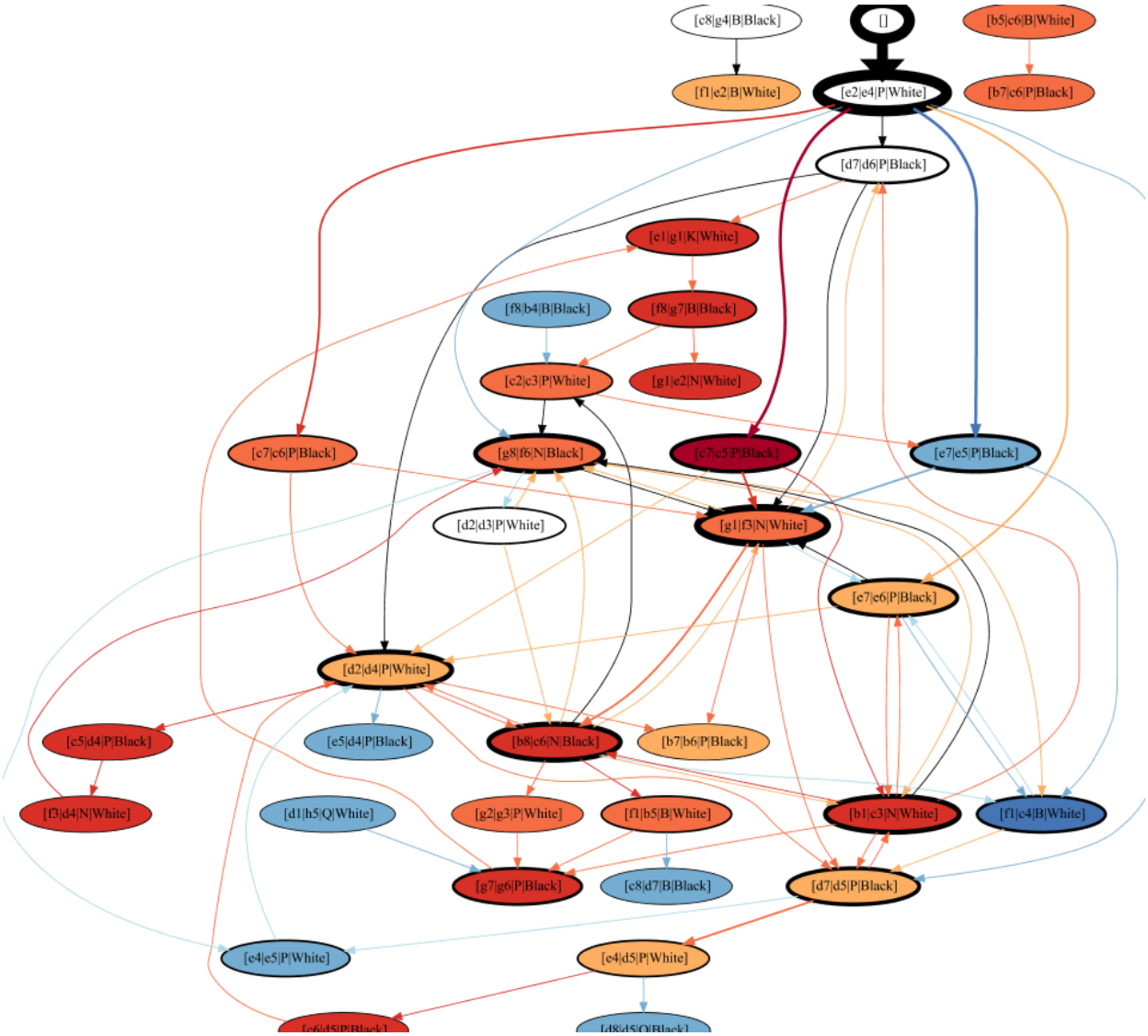


Figure 5.10: Comparison of the HRG and LRG models starting with e4

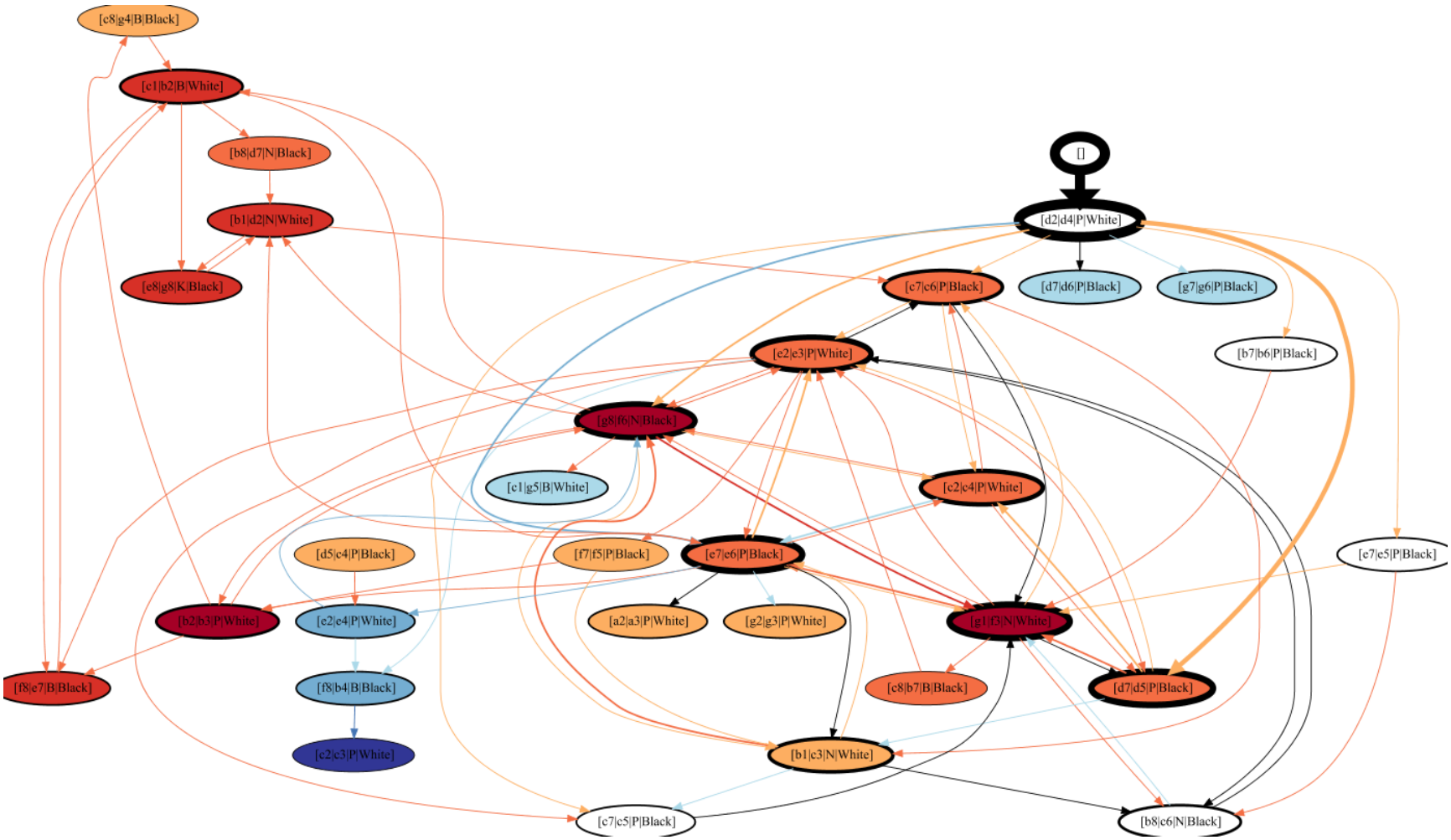


Figure 5.11: Comparison of the HRG and LRG models starting with e4

Chapter 6–Discussion

In this section, the specification choices and the results are elaborated and discussed. Possible explanations and implications of the results are stated, as well as limitations of the results.

In the sections 3.1.2 and 5.1, the decisions during the gathering and formatting data are stated and elaborated. Many steps in the formatting of the data were necessary to simplify the event logs, in order to use them in the process mining techniques. Some steps, however, were implemented to make the models more clear, but as a consequence, more information had to be filtered out. For example, only the first twelve moves of a game were used, to create a clearer model of the opening. The opening of a chess game, however, can also consist of 20 moves and therefore a lot of possible data of the opening is lost. Similarly, due to the relatively small amount of data and to make the analysis go faster, the event logs were filtered on the first move. Only the traces starting with the moves e4 and d4 were preserved, therefore many different opening moves were removed from the event log and are therefore not analysed. Lastly, to speed up the conversion of data and filtering of data, only 500 games were used. Online chess databases, however, contain many more games which are free to use. More data would ultimately increase the validity of the model, as it would represent the actual reality of the game even better and it would increase the reliability, as more noise would be filtered.

In section 5.2, the results process mining on chess event logs are stated and elaborated. To begin, the models generated by the heuristic miner are clear while still containing sufficient events. This confirms the usability of process mining techniques on filtered chess event logs. Nevertheless, the LRG models are more complex than the HRG models. This implies that the low rated players are less consistent and play a larger variation of openings. A larger number of different openings leads to more different moves, thus more events in the models. The inconsistency of the low rated players might be caused by the lack of opening knowledge from the players. There are many different openings and defences against these openings which lead to equal positions. Many lower rated players do not possess this knowledge and develop their own openings and defences. This also explains why the HRG model contains the event c7|c5|P|Black, while the LRG model does not. The move c5 is the start of the Sicilian defence, a popular defence against the move e4.

Next, the conformance checking on the models with event logs consisting of three HRGs and three LRGs generated precision scores. The HRG model had a higher precision score with both the HRGs and LRGs test event logs compared to the LRG model. The expected result was to see a high precision score with the HRG model checked with the HRGs and the LRG model with the LRGs, and a low precision score with the HRG model checked with the LRGs and the LRG model with the LRGs. These results were not observed, therefore regarding this approach of identifying the chess expertise invalid. Presumably, this is caused by the consistency of high rated players and inconsistency of low rated players. The HRG model contained many popular openings while the LRG model contained many other less common moves. High and low rated players both often play certain openings or parts of them, therefore fitting more in the HRG model than the LRG model.

Moreover, the process comparator models show the differences between the HRG and LRG process models. The majority of the events in these models has a red shade. This is most likely caused by the inconsistency from lower rated players, as explained earlier in this section. Moreover, almost all the

knight moves are red. This might be caused by the same effect, but it could also be the result of the higher complexity of knight moves. Knights move in complex patterns, which makes it more challenging to find the right move. Another noticeable difference is early castling, which often occurs in the HRG. Castling is a move to bring the king to safety. King safety is often overlooked by lower rated players, since opponents generally do not take advantage of the king's vulnerability. In games between higher rated players, this is more important and is therefore more frequently seen.

Overall, the process comparator is a great technique to visualise the differences between high and low rated games. The models contain many common openings and can be used to give indications for good moves to novel players. When two events are connected with a large arrow, it implies that the transition occurs frequently, thus the novel player can assume that it is a good sequence of moves. Similarly, when two events are not connected, this sequence of moves is not often played and is therefore very position specific or not a good move.

6.1 Limitations

This research is conducted in the field of process mining, which is still relatively young. This provided challenges and limitations in the available literature and practical knowledge. A large quantity of knowledge about the ProM-lite plug-ins, like how to interpret results, were obtained through forum pages instead of published articles. Also, ProM-lite is a free to use process mining framework, so there is a smaller quality assurance compared to commercial tools. Prom-lite is a minimized version of Prom, therefore this research was limited in the number of plug-ins available.

Beside the tools, there were multiple limitations in gathering and formatting the data. First, due to time constraints only a small amount of the available data was used. Also, formatting the data and generating models costs significantly more time when the amount of data increases.

Chapter 7–Evaluation

To evaluate this approach of identifying the differences between novice and expert chess players, the generated models were presented to two chess enthusiasts with many hours of experience in chess gameplay and the chess community. The models were first explained and afterwards the models were informally discussed to reveal their expert opinion. The two evaluations were performed separately. In this section, their view on the models explained, together with remarks, future possibilities and shortcomings of this approach. First, the ideas of expert 1 will be discussed, followed by the ideas of expert 2. Lastly, the ideas will be compared and discussed.

7.1 Expert 1

Expert 1 is approximately one year actively involved in the chess community and plays chess on a regular basis for over a year. He has a rating of 1200 on the Chess.com platform on the day of the evaluation. This rating indicates that he is an intermediate player.

First, the expert elaborated on his interpretation of the model, whether it was clear and what should be different for more clarity. Expert 1 first commented that the event should probably contain more than the move and the colour of the player. He stated that not only the specific move is important, but also the board state and the situation as a whole, “A specific move can be very good or very bad depending on the board state”. Since the event logs only contain a specific move, the board state should be extracted from the previous sequence of events. Using the previous sequence of events however is not always possible. Expert 1 stated that some events have no transition arrow connecting them to other events and therefore also no previous sequence of events. This makes the model hard to understand. Moreover, expert 1 specified that some events have arrows going back and forth, which is impossible in a chess game. Lastly, expert 1 explained that the colour scheme is counter intuitive. At the moment the events that are frequent in the HRGs are coloured red, while red is often associated with wrong or bad. Also, it was unclear what the white events were indicating. With a colour scheme with blue and red, Expert 1 would expect purple to be in between.

Secondly, to what extent the model is able to show the difference between novice and expert players was discussed. Expert 1 quickly stated that only looking at the opening of a game is often not enough. “Many players play a standard opening and are able to remember all the book moves of this opening”. Expert 1 explained that these book moves are always the best moves and would therefore not make a distinction between novice and expert players. The differences between novice and expert players mainly occur in the midgame when hard decisions are made. This would however be hard to visualise with such a model due to the extreme number of different possibilities. Expert 1 also described how the Chess.com analysis method can be used to identify a player’s expertise. During a game, Chess.com keeps track of the number of blunders and best moves made, which could give a good indication of how well a player plays.

Lastly, the potential to educate new players with a model was discussed. Expert 1 described that it can only be useful for the start of the game since the model cannot be specific enough for the midgame. Moreover, to make the model usable for educational purposes, it must be clearer and be more intuitive. Expert 1 explains that the analysis mode in Chess.com has a more intuitive user interface, clearly describing the best moves, which makes it clearer and easier to understand.

7.2 Expert 2

Expert 2 has been playing chess actively for approximately four years but got familiar with the game at a young age. He has a rating of 1450 which indicates that he is a high intermediate player. First, the clarity of the model was discussed. Expert 2 commented that some events did not have any transition arrows going towards that event, and the purpose of this was unclear. Similarly, it was unclear why some events had no transition arrows going out. Overall, he explained that the colour code is clear and that the flow of events in such a model is self-explanatory.

Secondly, the use and possibilities of the model to identify a player's chess expertise were discussed. To begin with, Expert 2 doubted the use of the model since it only uses the first 12 moves of a game. He explained that many players learn a certain opening by heart and are able to play this against any opposing opening move. Expert 2 stated, opening theory will stay the same between high and low rated games, it must simply be remembered and therefore does not contain as much information about expertise. However, Expert 2 also stated that often low rated players do not remember opening theory as well as higher rated players and therefore this model is still able to illustrate the differences between high and low rated players. Expert 2 explained that the endgame contains more information about the expertise of a player, since it often does not contain a standard sequence of moves. In the endgame, board analysis and knowing where to attack or defend is imperative to win the game. Many positions require the players to think ahead for many steps and make important decisions. All these factors in the endgame make it more interesting to determine a player's chess expertise. However, the end game is very diverse and does not have a set starting point or position and would therefore make it harder to analyse using process mining. Moreover, Expert 2 explained that using 500 games to make a model might not be enough due to the large number of possibilities in chess, especially in the mid and end game.

Lastly, the ability to educate new chess players was discussed. Expert 2 first explained that the model cannot be blindly followed, since it does not consider past events that well. For example, he explained that many events go to and from the event 'knight to f3', however the transition arrows cannot be blindly followed since sometimes a move is good or bad depending on the previous moves. Expert 2 did see possibilities in educating what possible moves are. He stated, "Many new players do not know which pieces to move or what to do after the first couple of moves, a model showing what popular moves are could help with giving indications for possible moves". The model shows that early knight moves are popular, and new players can use this information in their own games. However, to make all of this work, the model should first be clearer and more understandable for new players. At the moment, the model shows too many transitions and events which would make it difficult to understand for new players. To conclude, Expert 2 mainly saw possibilities in educating new chess players in what kind of moves are important in the early game, and by giving possible moves as hints but the model has to be clearer.

7.3 Evaluation conclusion

To conclude, both experts agreed that the model could be helpful for novice chess players to get indications for possible moves, granted that the model becomes clearer and more understandable and that the model gives a good insight into the main differences between novice and expert players. However, both experts also had doubts about the use of the opening, as it does not represent the expertise of a chess

player very well. This is mainly because the same openings can be played every game, and the opening can be learned by heart, hence removing the chess expertise. The experts explained that the endgame would offer more information about the expertise of a player, but this would become more difficult since there is no clear start of the endgame and there are many variations in endgames. The experts also agreed that the sequences of events should not be blindly followed because the board state is not truly considered, which is one of the most important things in chess.

Chapter 8–Conclusion

In this chapter, the research questions will be answered based on the findings of the literature research and process mining experiments. The research questions defined in the introduction of this thesis are:

- *RQ1: What are the differences in gameplay and mindset between novice and expert chess players and how can they be identified?*
- *RQ2: How can process mining techniques be applied on complex event logs?*
- *RQ3: How can process mining techniques be utilized to identify chess expertise?*

First, the main findings to these research questions will be described.

8.1 Key findings

8.1.1 What are the differences in gameplay and mindset between novice and expert chess players and how can they be identified?

There are a lot of gradations in chess gameplay, which can be identified in multiple ways. The literature review shows that players with a different chess expertise have different playstyles, analysing capabilities and most importantly experience. Commonly, novice players play “Hope chess”, a playstyle in which the players do not fully analyse the opponents’ opportunities when making moves. A hope chess player does not anticipate the opponent making threatening moves. This also relates to the lack of analysing capabilities of novice players. Novice players are often unable to tell the difference between good and bad positions, which leads to poor game decisions. Lastly, a major difference between novice and expert players is their knowledge of chess patterns. Especially the opening and end game contain many common patterns which lead to advantages for a certain player if it is not dealt with correctly. However, also the mid-game contains many chess tactics which are often missed by novice players.

The previously discussed features can be identified in order to estimate a player's chess expertise. The literature review shows that many methods to identify chess expertise have been explored and developed. To begin with, the most popular methods compare the player’s moves with the best moves calculated by a computer. Based on the amount of best moves a player makes, their chess expertise is estimated. Similarly, chess questionnaires look at to what degree a player is able to make the best moves, the more correct answers, the higher the estimated chess expertise will be. In this research a novel method to identify the differences between novice and expert chess players is proposed. Process mining or more specifically the process comparator is able to illustrate the difference in moves and sequences of moves.

8.1.2 How can process mining techniques be applied on complex event logs?

Chess is a complex game with many possible board positions and moves. This results in complex event logs containing many different events and transitions. The standard plug-ins of ProM-lite are not capable of discovering processes from complex event logs. Therefore, the complex event logs were simplified. First, only the first twelve moves of a game were used in the event logs. The first twelve moves contain a

lot of information about the opening, while significantly reducing the complexity of the event logs. Second, the event logs were filtered on the first move, making the event logs smaller and more specific. With divided data, the process mining techniques had to be applied multiple times, but the process mining algorithms were faster and the models became clearer. To finalise, the filters in the ProM plug-ins were used to simplify the models. The heuristic and fuzzy miner both had event and transition filters, which reduced the number of events and transitions based on the number of occurrences. The threshold was increased, removing infrequent events and transitions which ultimately simplified the process model.

8.1.3 How can process mining techniques be utilized to identify chess expertise?

ProM-lite provides a great variety of plug-ins capable of process discovery and process enhancement. The results of this research show that the process comparator is a great plug-in to compare event logs and is capable of identifying chess expertise. To identify chess expertise with process mining techniques, an event log with HRGs and an event log with LRGs need to be created. The process comparator develops a model from these event logs and illustrates the differences between the two event logs. The differences between HRGs and LRGs indicate the differences in gameplay between high and low rated players. To estimate a player's chess expertise, the moves can be compared to the moves in the model. Based on to what extent these moves coincide with the moves in the model, the chess expertise is estimated.

Nevertheless, the second concept to identify chess expertise using process enhancement did not prove to be an effective method. The precision scores produced by the process enhancement plug-in did not match the expected scores. The HRG model always produced a higher precision score even when it was checked with LRGs.

Chapter 9–Future Work

This research shows the promising possibilities of process mining on chess event logs. This however is only the first step in using it in applications to improve currently used systems. To implement this promising technique, more research and testing is needed. To begin, only a small amount of data is analysed. To increase the reliability of the technique, more games should be included in the event logs. With more games, noise would be filtered out which would lead to similar models for different databases. Next to that, various first moves should be examined. At the moment, only the two most popular first moves are included in the event log. However, chess has many different openings and possible opening moves. By separating the various first moves, the models would remain clearer and easier to understand, while also showing the imperative information of the opening. Moreover, this research only focused on the opening of the game while neglecting the mid and end game. Both the mid and end game contain a lot of information about the user's expertise. For example, in the mid game all the pieces are playing an important role in the game, defending, and attacking at the same time, therefore making it harder to see what the best moves are. The end game often contains less pieces but features common patterns to checkmate the opponent's king. Depending on if and how fast a player sees these patterns or the best moves are found, the expertise can theoretically be estimated. Future research should include more parts of the game to get a better understanding of the user's expertise.

Another attribute in which future research could expand, is the Elo ratings distribution in the event log. In this research, only the highest and lowest rated games are obtained from a large PGN file, to get two separate event logs. This was mainly to investigate the possibilities of analysing the differences between HRGs and LRGs. This could be expanded by creating a separate event log for every 100 points in Elo rating, i.e., a separate event log for 100-200 and 200-300 et cetera. This has potential to outline the more subtle differences between differently rated players. Also, it might discover certain thresholds for game tactics, like en passant.

To add more data to the event logs, thus making the event logs more complex, novel methods to mine complex processes need to be explored. The complex process models need to be simplified while still showing relevant information. In addition, future research should focus on timestamps in the event logs. The time that a chess player has to spend on thinking about a move or analysing a position, reveals a lot about the expertise of the player. This component is not utilized in this research due to the lack of time but can be added by using process enhancement. Moreover, process enhancement can be used to add other attributes to the model, like checks and captures. This would make the model more precise and a better representation of the real-world game.

Lastly, this research only focuses on process mining on chess event logs to identify a player's expertise. Similar techniques, however, can also be applied to other software to identify a user's expertise or experience. For example, software programs like movie editing often have a steep learning curve, meaning that a lot of time needs to be spent before experiencing the smallest benefits of the software. This might be because users are overwhelmed with the number of tools, settings and other possibilities. When a user's expertise can be identified with process mining, a software program can be personalized based on the expertise level. Novice users could be provided with a tour, only the basic settings and tooltips while expert users could get all the advanced features and tips about advanced settings. Ultimately, this could improve the user experience by lowering the learning curve.

Appendix A

```
import pandas as pd

df = pd.read_csv("input.csv")
output = pd.DataFrame(columns=['game_id',
'move_no','player','timestamp','notation','move','from_square','to_square','piece','color'])
counter = 1
threshold = 12
prevID = 0
for index, row in df.iterrows():
  if row["game_id"] == prevID and counter < threshold:

    tempdf = pd.DataFrame({'game_id': [row['game_id']],
                'move_no': [row['move_no']],
                'player': [row['player']],
                'timestamp': [row['timestamp']],
                'notation': [row['notation']],
                'move': [row['move']],
                'from_square': [row['from_square']],
                'to_square': [row['to_square']],
                'piece': [row['piece']],
                'color': [row['color']],
                })
    output = output.append(tempdf, ignore_index = True)
    counter += 1
  elif row["game_id"] != prevID:

    tempdf = pd.DataFrame({'game_id': [row['game_id']],
                'move_no': [row['move_no']],
                'player': [row['player']],
                'timestamp': [row['timestamp']],
                'notation': [row['notation']],
                'move': [row['move']],
                'from_square': [row['from_square']],
                'to_square': [row['to_square']],
                'piece': [row['piece']],
                'color': [row['color']],
                })
    output = output.append(tempdf, ignore_index = True)

    counter = 1
    prevID = row["game_id"]

output.to_csv('output.csv', index=False)
```

References

[1] Van der Aalst, W. “Process mining: Data science in action. In Process Mining: Data Science in Action.” (2016). https://doi.org/10.1007/978-3-662-49851-4

[2] Keith, B., & Vega, V. “Process mining applications in software engineering.” Advances in Intelligent Systems and Computing, 537, 47–120. (2017). https://doi.org/10.1007/978-3-319-48523-2_5

[3] Weerapong, S., Porouhan, P., & Premchaiswadi, W. “Process mining using α-algorithm as a tool (A case study of student registration).” International Conference on ICT and Knowledge Engineering, 213–220. (2012). https://doi.org/10.1109/ICTKE.2012.6408558

[4] Amelia Effendi, Y., & Sarno, R. “Conformance Checking Evaluation of Process Discovery Using Modified Alpha++ Miner Algorithm.” Proceedings - 2018 International Seminar on Application for Technology of Information and Communication: Creative Technology for Human Life, ISemantic 2018, 435–440 (2018). https://doi.org/10.1109/ISEMANTIC.2018.8549770

[5] Nuritha, I., & Mahendrawathi, E. R. “Structural Similarity Measurement of Business Process Model to Compare Heuristic and Inductive Miner Algorithms Performance in Dealing with Noise.” Procedia Computer Science, 124, 255–263 (2017). https://doi.org/10.1016/j.procs.2017.12.154

[6] R’Bigui, H., & Cho, C. “The state-of-the-art of business process mining challenges.” International Journal of Business Process Integration and Management, 8(4), 285–303 (2017). https://doi.org/10.1504/IJBPIM.2017.10009731

[7] Ferilli, S., & Angelastro, S. “Mining chess playing as a complex process.” Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 10312 LNCS, 248–262. (2017). https://doi.org/10.1007/978-3-319-61461-8_16

[8] Ferilli, S. “WoMan: Logic-based workflow learning and management.” IEEE Transactions on Systems, Man, and Cybernetics: Systems, 44(6), 744–756 (2014). https://doi.org/10.1109/TSMC.2013.2273310

[9] Ferilli, S., & Esposito, F. “A logic framework for incremental learning of process models.” Fundamenta Informaticae, 128(4), 413–443 (2013). https://doi.org/10.3233/FI-2013-951

[10] Ferilli, S., & Redavid, D. “A process mining approach to the identification of normal and suspect traffic behavior.” In Advances in Intelligent Systems and Computing (Vol. 728). Springer International Publishing (2018). https://doi.org/10.1007/978-3-319-75608-0_4

[11] Chapela-Campa, D., Mucientes, M., & Lama, M. “Understanding complex process models by abstracting infrequent behavior.” Future Generation Computer Systems, 113, 428–440 (2020). https://doi.org/10.1016/j.future.2020.07.030

[12] Chapela-Campa, D., Mucientes, M., & Lama, M. “Simplification of Complex Process Models by Abstracting Infrequent Behaviour.” In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Vol. 11895 LNCS. Springer International Publishing (2019). https://doi.org/10.1007/978-3-030-33702-5_32

[13] Sun, H. W., Liu, W., Qi, L., Du, Y. Y., Ren, X., & Liu, X. Y. “A process mining algorithm to mixed multiple-concurrency short-loop structures.” Information Sciences, 542, 453–475 (2021). https://doi.org/10.1016/j.ins.2020.07.003

[14] Elo, A. E. “The rating of chessplayers, past and present.” ARCO PUBLISHING. INC (1978).

[15] Grabner, R. H., Stern, E., & Neubauer, A. C. “Individual differences in chess expertise: A psychometric investigation. Acta Psychologica.” 124(3), 398–420 (2007). https://doi.org/10.1016/j.actpsy.2006.07.008

[16] Campitelli, G., & Gobet, F. “The role of practice in chess: A longitudinal study. “Learning and Individual Differences, 18(4), 446–458 (2008). https://doi.org/10.1016/j.lindif.2007.11.006
[17] Heisman, D. “The Improving Chess Thinker.” Mongoose Press (2009).
[18] Ferreira, D. R. “Determining the strength of chess players based on actual play.” ICGA Journal, 35(1), 3–19 (2012). https://doi.org/10.3233/icg-2012-35102
[19] Van Der Maas, H. L. J., & Wagenmakers, E. J. “A psychometric analysis of chess expertise.” American Journal of Psychology, 118(1), 29–60 (2005).
[20] Junior, L. R. S., & Thomaz, C. E. “Visual perception ranking of chess players.” In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Vol. 12131 LNCS. Springer International Publishing (2020). https://doi.org/10.1007/978-3-030-50347-5_27
[21] Scheible, C., & Schütze, H. “Picking the amateur’s mind - Predicting chess player strength from game annotations.” COLING 2014 - 25th International Conference on Computational Linguistics, Proceedings of COLING 2014: Technical Papers, 311–321 (2014).
[22] Van der Aalst, W. M. P., & Weijters, A. J. M. M. “Process mining: A research agenda.” Computers in Industry, 53(3), 231–244 (2004). https://doi.org/10.1016/j.compind.2003.10.001
[23] A. Mader and W. Eggink. “A Design Process For Creative Technology.” Tech. Rep. (2014).
[24] Vladimir A. Rubin, et al. “Process mining can be applied to software too!.” Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, New York, USA (2014). https://dl.acm.org/doi/pdf/10.1145/2652524.2652583
[25] Nikitin, K. “Educational Game Analysis Using Intention and Process Mining,” (2020).
[26] Ferilli, S., Angelastro, S. “Mining chess playing as a complex process,” Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) (2017). https://doi.org/10.1007/978-3-319-61461-8_16
[27] Carr. J,. “Chess is seeing a huge popularity spike on Twitch,”, Daily esports, January 11, 2021. https://www.dailyesports.gg/chess-is-seeing-a-huge-popularity-spike-on-twitch/#:~:text=One%20of%20the%20oldest%20games,the%20popularity%20increase%20of%20chess.
[28] Günther, C. W., & Van Der Aalst, W. M. P. “Fuzzy mining - Adaptive process simplification based on multi-perspective metrics.” Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 4714 LNCS, 328–343 (2007). https://doi.org/10.1007/978-3-540-75183-0_24