THE STABLE ROOMMATES PROBLEM AND CHESS TOURNAMENT PAIRINGS

Eija Kujansuu, Tuukka Lindberg and
Erkki Mäkinen

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF TAMPERE

REPORT A-1995-2

UNIVERSITY OF TAMPERE
DEPARTMENT OF COMPUTER SCIENCE
SERIES OF PUBLICATIONS A
A-1995-2, APRIL 1995

THE STABLE ROOMMATES PROBLEM AND CHESS TOURNAMENT PAIRINGS

Eija Kujansuu, Tuukka Lindberg and Erkki Mäkinen

University of Tampere
Department of Computer Science
P.O. Box 607
FIN-33101 Tampere, Finland

ISBN 951-44-3746-2
ISSN 0783-6910

THE STABLE ROOMMATES PROBLEM AND CHESS TOURNAMENT PAIRINGS

Eija Kujansuu, Tuukka Lindberg, and Erkki Mäkinen#
Department of Computer Science, University of Tampere,
P.O. Box 607
FIN-33101 Tampere, Finland

Abstract. In many chess tournaments the number of players is much larger than the number of rounds to be played. In such tournaments the Swiss pairing system is usually used. This means that players with equal or almost equal scores so far are played against each others. Moreover, each player should alternately have, if possible, white and black pieces, and every pair of two players is allowed to play at most once against each others. This paper shows how the well-known stable roommates algorithm can be used to determine the pairs in a pairing system similar to the Swiss system.

Key Words and Phrases: chess, pairing, stable pairing, the stable roommates problem.
CR Categories: F.2.2., G.2.1.

1. Introduction

If in a chess tournament the number of players is one bigger than the number of rounds to be played, the all-play-all system (also called round-robin system) is used, i.e. each player plays against every other exactly once. If there are more players then some other pairing system must be applied. The most popular of these pairing systems is the so called *Swiss system*. In chapter 1.1. we shortly describe this system. A more detailed description of the Swiss pairing system is given in [5] and the complete set of pairing rules can be found in various FIDE (Fédération Internationale des Echecs) documents. Readers unfamiliar with the chess terminology used in this paper may consult e.g. [3].

The purpose of this paper is to suggest a new pairing system in which we formalize the pairing process as an instance of the stable roommates problem. In chapter 1.2. we recall the basic properties of the stable marriage and the stable roommates problems. For further details concerning these problems and algorithms solving them, see [2].

Corresponding author. E-mail: em@cs.uta.fi

1.1. The Swiss pairing system

For the sake of simplicity, we suppose that chess tournaments have an even number n of players. Players are numbered from 1 to n according to their ratings (by using the international ELO rating or some of its national variants) so that the player with the highest rating gets number 1 and the player with the lowest rating gets number n. In the first round the pairs are (1, n/2), (2, n/2 + 1),..., (n/2 - 1, n). The winner of a game scores one point, the loser scores nil, and in the case of a draw both players score half a point. In the second round and in all rounds to come players with equal scores should play against each others. This is not always possible. An obvious reason is that there can be *score groups* (i. e. groups of players with equal number of points) with odd number of players. Second, the players in a score group may have already played against each others. Third, in addition to scores we must also take care of the colours (playing with white or black pieces). As far as possible, at the end of each even round, all players should have had an equal number of whites and blacks. The *colour history* tells the colours with which the player has played in the previous rounds. An alternating colour history (e.g. WBWBW) is the optimal one. There is a trade-off between the demands concerning scores and colours: is it better to allow a pair with non-equal scores if it equalizes the colour histories or should we stick with equal scores although it might mean repetition of colours? The FIDE rules give us some advice in this trade-off situation. For example, the *colour difference*, i.e. the difference of games played with whites and blacks, should not exceed one. Moreover, there are regulations concerning *'floaters'*, i.e. players who are moved outside their own score groups in order to get reasonable pairs. Of course, a floater should be chosen such that the choice gives the best possible balance to the colour histories in question.

The pairing process is supposed to be done manually; FIDE rules say that computers can only assist the human tournament director who has the final decision in all matters concerning pairings. In order to keep the pairing algorithm manually feasible, pairing must be done score group by score group starting at the topmost group and continuing just above the middle group (the one containing the median player), then going to the lowest group and continuing upwards. Finally we handle the middle group with all the problematic cases pushed forward when the other groups were handled. The price paid for keeping the pairing system manually feasible is that we do not have a global view of the pairing process. This is recognized in the FIDE rules by adopting an elitistic policy of favouring top ranking players and players in the topmost score groups. This policy is shown e.g. in the order in which score groups are handled: the present order guarantees that the players in the topmost score group are handled in the best possible way. Similarly, when only one of two colour histories can be equalized, the rules demand to equalize the colour history of the higher ranked player.

We know only one earlier attempt to formalize the pairing process. Ólafsson [5] has proposed a method using weighted matchings. His method follows the FIDE rules with respect to the score group handling order. Thus, Ólafsson's method formalizes the trade-off situation between score proximity and colour equalization separately in each score group. Contrary to Ólafsson's proposal, we reject the assumption that a pairing system should imitate the manual system described in the FIDE rules. Hence, our pairing system produces pairs following the ultimate goals of the Swiss system (players with equal scores should play against each others, and colours alternate, if possible) without using the handling order based on score groups.

1.2. The stable marriage and stable roommates problems

An instance of the *stable marriage problem* consists of two equal-sized sets of participants, the men and the women. Associate with each person there is a strictly ordered *preference list* containing all members of the opposite sex. Person p *prefers* q to r if and only if q precedes r on p's preference list. A *matching* is a bijective mapping between the sets of men and women. If man m and woman w are matched in a matching M, then m and w are called *partners* in M; this is denoted by $m = p_M(w)$ and $w = p_M(m)$. A man m and a woman w are said to *block* the matching M, if m and w are not partners in M, but m prefers w to $p_M(m)$ and w prefers m to $p_M(w)$. If a matching has at least one pair of blocking persons it is *unstable*; otherwise it is *stable*.

Given an instance of the stable marriage problem, the *Gale-Shapley algorithm* finds a stable matching in time $O(n^2)$, where n is the common number of men and women. For each instance there exists at least one stable matching and the maximum number of stable matchings grows exponentially when n grows.

The Gale-Shapley algorithm solving the stable marriage problem could be used in solving the chess pairing problem as follows. Divide first the set of players into two equal-sized disjoint sets, the players having white pieces in the the round and the players having blacks. The preference lists are formed according to the score differences: players having the same score are in the beginning of the list and the score difference increases towards the rear of the list. An obvious drawback of this method is that the decision concerning colours (i.e. the division of the players into the two sets) becomes too dominating factor in the pairing process. A better balance between scores and colours is obtained by using the stable roommates problem which is to be described next.

The *stable roommates problem* is a variant of the stable marriage problem in which each person in a set (of even cardinality) puts all the other persons to his preference list. A matching is now a partition of the set into disjoint pairs. A matching is unstable if there are two persons who prefer each others to their partners in the matching. As above, such persons are said to block the matching. If no blocking pairs exist, the matching is stable. Contrary to the stable marriage problem, the stable roommates problem has instances which do not admit stable matchings at all. An instance is *solvable* if it admits a stable matching; otherwise it is *unsolvable*.

If an instance admits a stable matching it can be found in time $O(n^2)$ [2, 4]. The stable roommates algorithm deletes the entries from the preference lists until either some list becomes empty or every list is reduced to a single entry. The former case indicates that no stable matching exists for the instance in question, and in the latter case entries left in the lists constitute a stable matching.

In chapters 2 and 3 we consider instances of the stable roommates problem in which the preference lists are not complete, i.e. all persons do not necessarily put all other persons to their preference lists. Naturally, this increases the possibility to have an unsolvable instance.

Before going into the technical details of the new pairing system we give some motivation for the use of stable matchings. As we have already mentioned, the current FIDE pairing rules demand a somewhat artificial handling order for the score groups. By using the new pairing system, we can guarantee a fair treatment for players in all score groups. Moreover, the following usual and unconvenient (from the tournament director's point of view) occurrence is impossible: after seeing the pairing for the next round a player complains that he has wrong colour or that he is paired in a wrong score group, and what is more annoying, points out another player who would be a more suitable opponent for him according to the pairing rules and who would also benefit from the rearrangement of the pairs. Such a situation is impossible if the pairing for the next round is stable. Of course, there can be unsuitable colours and score groups, but since no blocking pairs are possible, a player cannot point out another single player who would also benefit from the rearrangement.

2. Making up the preference lists

The crux of our pairing system is of course the method used in making up the preference lists. When the lists are completed the outcome of the pairing system is determined by the normal roommates algorithm. In what follows we describe the main principles used in ordering the players to the lists. This simplified description disregards some nuances and fine tunings related to the complete set of FIDE rules. The principles given here for the main factors (score group and colour) of the pairing process are applicable also for all possible specialities found from the FIDE rules but these specialities are overlooked here.

Suppose we are preparing the pairings of the next round in a tournament. We know the scores and colour histories of every player. In order to formulate the rules used in making up the lists and to be able to compare different pairing systems we need some notations. The *colour difference* of player p, denoted by cd(p), is defined as the the difference of the times p has so far played with white and black pieces. Notice that after each odd round the absolute value of colour difference cannot be less than 1. If p and r are players, we define the *score difference* of p and r, denoted by sd(p,r), as the absolute value of the difference of their points so far scored in the tournament.

It is desirable that the resulting pairing does not contain a pair of players with a score difference much larger than the average score difference. Hence, an additional goal in the pairing process is to minimize the value

$$\max_{p+ r} \mathrm{sd}(p,r),$$

hereafter denoted as max-sd. In order to minimize max-sd in the new pairing system, we perform the algorithm so that we first fill in the preference lists only with players in the same score group. If a stable matching is not found, we gradually increase the allowed score difference until a stable matching is found. Note that this is by no means in contradiction with the general policy of handling all players simultaneously and not in score-group-wise: we do pair all players at the same time but neglect, if possible, pairs with a great score difference.

We devide the players into five classes depending on their colour differences. The possible colour differences are 2, 1, 0, -1, -2. As defined above, positive values indicate that a player has played more often with white pieces and negative values are correspondlingly related with black pieces. The matrix X shown in Table 1 gives the penalty related to colour differences used in a formula to be described below.

X	2	1	0	-1	-2
2	-	4	3	1	0
1	4	6	4	2	1
0	3	4	5	4	3
-1	1	2	4	6	4
-2	0	1	3	4	-

Table 1. The matrix X gives the penalties related to colour differences.

The missing values (denoted by '-') in Table 1 incidate that our system does not allow two players with colour difference 2 (or -2, respectively) to play against each others. Such a game would make the colour difference of one of the players in question to be 3 (or -3, resp.). If p and r are players, we simply write X[p,r] for the corresponding penalty found in X (although the rows and columns of X are named after the corresponding colour difference classes and not directly after the players).

Our system remembers colour histories two rounds backwards. Hence, it is sufficient to consider the following colour histories: BB, WW, BW, WB, B, and W. Colour histories B and W are used when finding the pairs of the second round. The matrix Y in Table 2 shows the cases where an additional small penalty based on colour histories is used. As with X, we use the notation Y[p,r] for players p and r. If Y[p,r] = s, a small additional penalty is added to the value used in determining the position of player r in p's list. The missing values in Y indicate that no additional penalty is used in those cases. The value s used in our tests is s = 0.0001. Y-values are used for breaking ties between otherwise equal players.

Y	BB	WW	BW	WB	B	W
BB	s			s	s	
WW		s	s			s
BW		s	s			s
WB	s			s	s	
B	s			s	s	
W		s	s			s

Table 2. Matrix Y gives an additional penalty based on colour histories.

For each player p we order the other players in the list of p in ascending order by the values

$$f(p,r) = \frac{10 - c}{6} * X[p,r] + c * sd(p,r) / ms + Y[p,r]$$

where

c is a coefficient by which we can tune the mutual influence of colours and scores,

and

ms is the maximum score difference allowed when making up p's list; as described earlier ms increases gradually if stable pairing is not found. If ms = 0, the term c * sd(p,r) / ms is disregarded.

In our tests we used the value c = 8.7. Decreasing c means that more emphasis is given to colours, while increasing c means that we stress the effect of scores.

3. The tests

The new pairing system was tested in a series of virtual tournaments against Protos [6], a commercial software certificated by FIDE. The number of players in the tournaments varied from 16 to 30, and the number of rounds was 5.

The pairings obtained in virtual tournaments were first analysed with two measures, which were to give us information on the general level. One of these measures was the sum of score differences of a round Σ sd(p,r). The other measure was the sum of colour differences Σ cd(p). With both measures it holds that the lower value the better result. The selection of the measures was guided by the twofold nature of the pairing process. Both of the measures should be minimized but in most cases reducing one of them rises the other one. This trade-off is already discussed in section 1.1. In the analysis, only the rounds from two to five were taken into account. This was due to the use of an identical pairing method in both programs in the first round.

	2nd round	3rd round	4th round	5th round
16	1	2	1	3
18	1	1	3	3
20	0	1	1	3
22	1	0	1	2
24	1	1	1	4
26	1	0	4	4
28	0	2	3	2
30	1	2	2	3

Table 3. The sums of score differences per round in Stable

Table 3 illustrates the sums of score differences obtained with Stable. The left column indicates the number of players of a round. The four other columns consist of the values gained with these different numbers of players in the last four rounds. The sum of score differences varied from 0 to 4. The average sum was 1.7. In four cases all players played against a player from their own score group, which is an ideal result.

Table 4 illustrates the same measure in Protos. Even with Protos the range of values was from 0 to 4. The average sum for Protos was 1.6. The number of ideal results was the same 4 obtained also with Stable. In both Tables 3 and 4 there is a clear trend of obtaining higher values in the last two rounds. This is the natural consequence of the uneven distribution of

	2nd round	3rd round	4th round	5th round
16	1	1	1	3
18	1	0	3	2
20	0	1	1	4
22	1	2	2	3
24	1	2	2	3
26	0	1	2	2
28	0	2	2	2
30	1	1	2	2

Table 4. The sums of score differences per round in Protos

scores among the players. The smaller the score groups become the higher is the likelihood for a player to get an opponent from an other score group than his own.

The sums of colour differences are illustrated by Tables 5 and 6, for Stable and Protos respectively. Here only rounds 2 and 4 were analysed because in odd rounds each player's colour difference is 1. In even rounds the value is either 0 or 2. So, these values reveal the number of players who must have an other colour in the following round. Simultaneously this value indicates how well the system can alternate the colours of the players. It does not, however, tell us the total number of players who have played twice with white or black pieces in the last two rounds. For this we would need to add information of colour histories. This indicator works anyway quite well also on its own. The results obtained with Stable and Protos differed from each other in many individual tournaments. The lower values, however, alternated between the two programs. From the generalized point of view a difference was found: Stable's average sum was 5.25 and the corresponding value for Protos was 5.75.

Further, the pairings can be analysed more precisely by concentrating on single pairs instead of the total pairings of a round. Some of these results even give us deeper understanding on the pairing result itself. For example, the distribution of the sum of score differences over single pairs may change the view of the situation radically. As mentioned earlier, one of the most important measures then is the maximal score difference of a round denoted by max sd(p,r). Further, the number of pairs having players from two different score groups completes the picture from the scores point of view. Other measures possible would be an individual colour difference cd(p) and a shortened colour history of two rounds' ch(p). The latter would be only a secondary measure used to complete the information given by cd(p).

	2nd round	4th round
16	0	8
18	4	4
20	4	4
22	0	4
24	8	4
26	4	8
28	4	8
30	12	8

Table 5 The sums of colour differences per round in Stable.

	2nd round	4th round
16	0	8
18	4	4
20	4	4
22	0	4
24	8	4
26	4	8
28	4	8
30	12	8

Table 6 The sums of colour differences per round in Protos.

For example with a cd(p)=1 a player p may have played twice with white pieces in successive rounds but the only way to be sure of the fact is to check the value of ch(p).

In Stable's pairings the values of max sd(p,r) varied from 0 to 3. The highest value 3 was obtained only once during the virtual tournaments. A value of the range from 1.5 to 2.5 was scored 8 times. The desirable situation of having a maximal score difference between 0 and 1 was reached in the remaining 23 rounds but only 4 of these admitted the ideal value 0. The number of pairs suffering from uneven players varied from 0 to 4. The higher the number of suffering pairs was, the lower difference was obtained in single pairs excluding the rounds with the value 0.

In Protos the score differences were more evenly distributed, the highest value obtained was 1. Of the 32 pairings only 4 were ideally paired, admitting no score differences. Due to the even distribution the number of suffering pairs was higher in relation to the Stable results. The maximum was nevertheless only 6 pairs.

4. Conclusions

The results of the two test programs differed from each other slightly but not significantly. Particularly the measures used to illustrate the total situations were promising. It seemed that a few little improvements in making up the preference lists would give us pairings at least as well as those obtained by using Protos.

However, the results on the pair level were less positive. Mainly the Stable's values of max sd(p,r) were too high compared with the FIDE rules in too many occasions. The question raised is how to minimize these values without shortening the preference lists so much that the instance would have no solution. Further research should be done in making up the preference lists and in reordering them when no solution is found.

A given instance of the stable roommates problem may have more than one solution (stable matching). All stable matchings can be found in time $O(n^3 \log n + n^2 r)$ where r stands for the number of stable matchings [1, 2]. Since we may well allow even 2-3 minutes our algorithm to find the pairs for the next round, we can also check whether any of the other possible stable matchings is better than the first one with respect to our measures. (The order in which the stable matchings are found depends on the structure of a poset formed by certain operations (called rotations) defined in the preference lists.) However, our tests indicated that no significant improvment (actually hardly any what so ever) can be obtained by checking all the stable pairings.

References

1. Dan Gusfield, The structure of the stable roommate problem: efficient representation and enumeration of all stable matchings, *SIAM J. Comput.* **17**, 4 (1988), 742-769.
2. Dan Gusfield and Robert W. Irving, *The Stable Marriage Problem. Structure and Algorithms*, The MIT Press, 1989.
3. David Hooper and Kenneth Whyld, *The Oxford Companion to Chess*, Oxford University Press, 1984.
4. Robert W. Irving, An efficient algorithm for the "stable roommaates" problem, *J. Algorithms* **6** (1985), 577-595.
5. Snjólfur Ólafsson, Weighted matchings in chess tournaments, *J. Opl. Res. Soc.* **41**, 1 (1990), 17-24.
6. Protos, Version 6.ENG. A computer program for the Swiss pairing system by Christian Krause, 1994.