A Rational Approach to Kriegspiel

P. Ciancarini
Dept. of Comp. Science
Univ. of Bologna

F. Dalla Libera
Dept. of Appl.Math.
Univ. of Venezia

F. Maran
School of Economics
Univ. of Venezia

ABSTRACT

Kriegspiel is an heterodox Chess variation in which the players have incomplete information because they are not informed of their opponent's moves. In fact, each player knows the position of his own pieces, but he can only guess the position of enemy pieces. Each player tries to guess the position of the opponent's pieces as the game progresses by trying moves that can be either legal or illegal with respect to the real situation: a referee accepts legal moves and rejects illegal ones; the latter are useful to gain insight about the situation. This means that players have to play in a context of uncertainty and partial information.

This paper describes the design of a Kriegspiel-playing program based on a "rational approach" (we know of no past attempts to build Kriegspiel playing programs). The program we have developed integrates two different notions of rationality introduced by Simon: the *substantive* and the *procedural* rationality [Sim76,Sim78]. The interesting part of such an experience is how the procedural rational approach can incorporate results obtained with substantive rationality, whereas the two approaches are usually considered alternative.

1. Kriegspiel

Heterodox Chess variations are chessboard games obtained changing the usual rules of Chess. There are thousands of Chess variations [Pri94], some of them derived simply changing the rules governing the movement of the pieces or the properties of the chessboard. Some variations change the nature of the Chess as a complete information game: one of these is Kriegspiel, a game invented to make Chess more similar to real warfare.

A Kriegspiel game involves three persons: two players and a referee. Each person has a chessboard. Only the referee can see the position of pieces of both players. Each player can only see the position of his own pieces and tries to gain information about the position of opponent's pieces as the game progresses. The game is ideal to be played on a network, using an automatic referee [WBB72].

We state the most important rules of Kriegspiel (for a more detailed description of the modern rules see [Li94]):

1. The starting position, the colors and movements of pieces, and the goal of giving checkmate are the same as in Chess.
2. The player P who has to move communicates his move to the referee only; the opponent O will not have such an information.
3. If the move is illegal, the referee answers to P that the move is illegal; then P has to try another move going back to point 2. P usually gains information on the state of O's pieces from such a failed try.
4. If the move is legal, the referee announces to both players that a move has been played legally and adds these pieces of information:

   If a piece has been captured (and which kind);

   If a check has been given (and on which row, column, or diagonal);

   If the game is terminated because of mate, stalemate, or insufficient material.

There are some local variants of Kriegspiel, which differ mainly in which information can be released by the referee. For instance, in [And58] a variant is discussed where each player can ask before each move if there are any captures by pawns. A famous set of rules was developed by players in RAND Corporation in the '50 [Fer92].

Fig. 1 shows the main control loop of a Kriegspiel game.

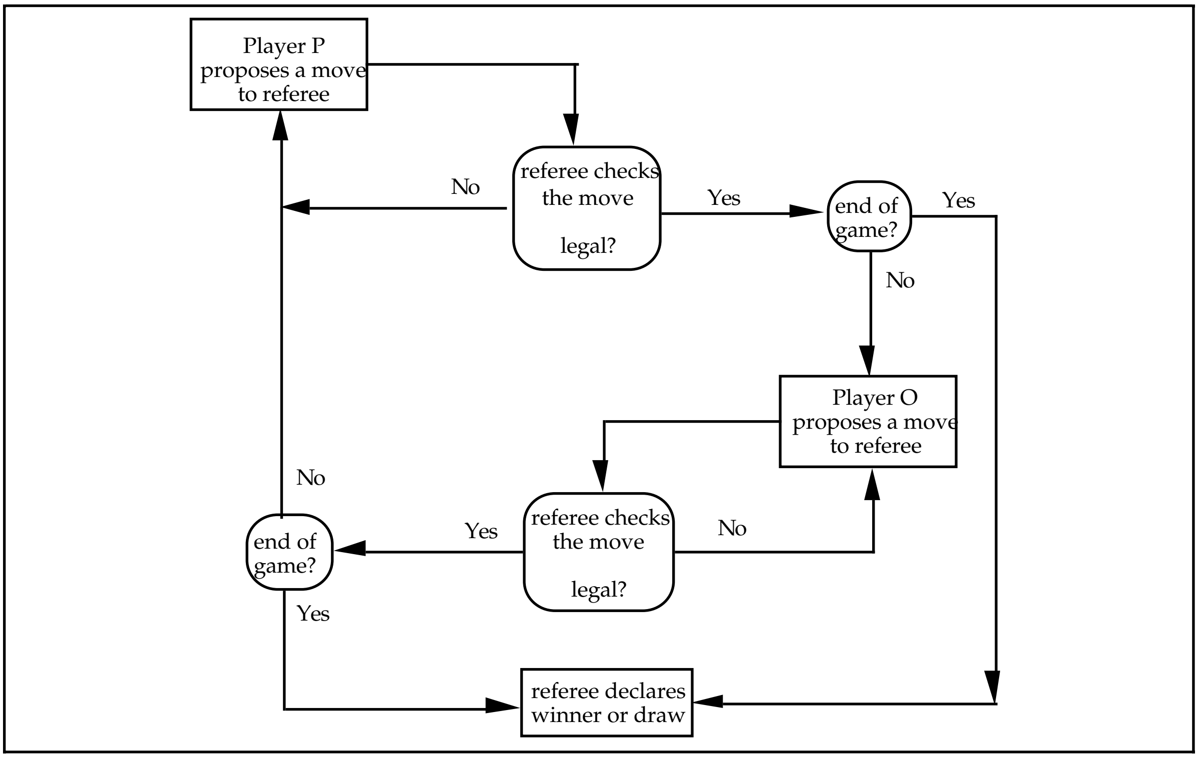


Fig. 1 *The coordination of a Kriegspiel game.*

The following Fig.2 shows an example which reports how evolves and terminates a Kriegspiel game. Players P and O only see their pieces; the referee sees he real situation. The figure must be read from top to bottom. Under the chessboard of each party there is the sentence he says.

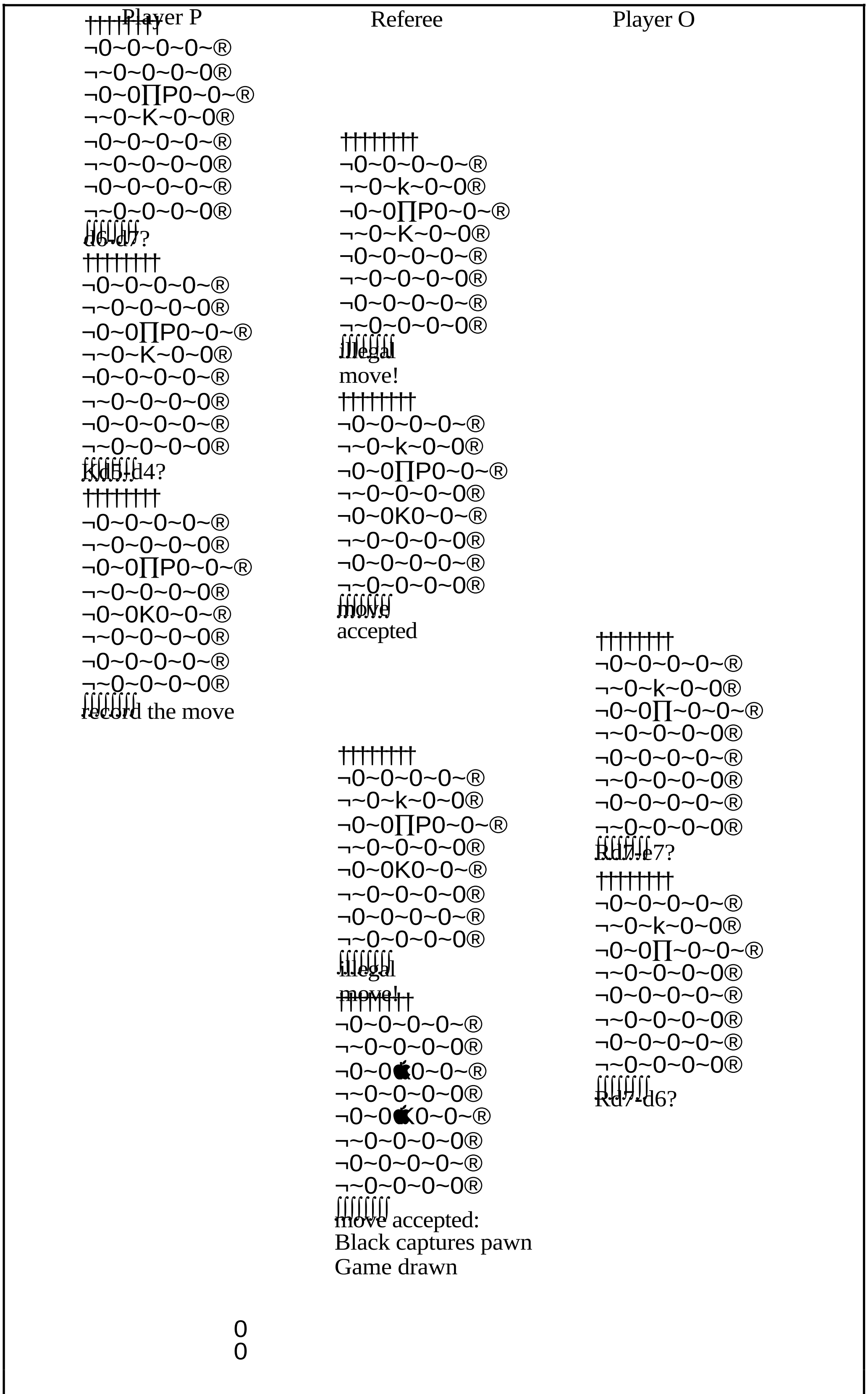


Fig.2 *Ending of a Kriegspiel game.*

2. Kriegspiel as a metaphor of decision making

The game of Chess has been widely studied because it is a microcosm that mirrors decision making situations in the real world. There is no reason to believe that the basic human faculties that a Chess professional uses to take his decisions are fundamentally different from the faculties used by an experienced businessman [Sim76].

However, a basic limit of Chess as a metaphor of decision making in the real world is that decisions taken by players have nothing to do with uncertainty in the sense in which the term is used in Game and Decision Theory, since the goal and the best strategy for each player can be easily and fully computed: if we could explore the whole game tree, Chess could be trivially solved by Minimax [VNeMor4]. We say that the kind of uncertainty faced by a Chess player is not caused by the contingencies of a strategic interaction.

Some business strategy theories suggest a distinction between *complicate* and *complex* situations [Fac89]: the former are similar to Chess situations, where the difficulty in decision making is determined by a lack of computational power; the latter instead involve incomplete information: the consequences of a decision are partially unknown, like the goals of the opponent.

In this technical sense Kriegspiel can be considered a *complex* game characterized by incomplete information, because of the asymmetry in the information available to the players as the game progresses. In fact, when a player makes an illegal move, from his "failure" he infers data that cannot be inferred by his opponent as well. Each player knows what he knows, but he does not know what his opponent knows about his knowledge.

A recent paper [SimSch92] points out how economic analysis and classical game theory have been developed accepting two main assumptions. The first assumption is that an economic actor (which in the Kriegspiel microcosm is the player) has a particular goal as the utility or profit maximization (in a game, to beat his opponent). The second assumption is that the actor is *substantively rational*. Substantive rationality is concerned with finding the correct or best action, given the goal in the specified situation.

Given these two assumptions, and given a description of a game-playing problem, a *descriptive* or *normative* economic analysis can be performed using standard tools like Game Theory models.

A descriptive analysis classifies the features of the game to be studied, and suggests a way to find the solution. For example, Chess is a *zero sum* game with *perfect information* and its solution can be obtained applying the

Minimax algorithm to the *game tree*. However, for games like Chess, Game Theory is not able to produce normative results in a substantively rational way, as it does in other games (e.g. the game played by two people which sum their fingers to see if the result is even or odd, where the best strategy for both players is to play an even or odd number of fingers with probability P=1/2).

According to Simon, another kind of rationality is the *procedural* rationality, which is concerned with procedures for finding correct actions taking into account not only the goal and the real situation, but also the knowledge and the computational capabilities and limits of the decision maker [Sim76,Sim78]. A good example of procedural rationality applied in Game Theory is the research on Chess-playing programs, which are built aiming at enabling them to make "reasonable" moves, because it is computationally impossible to fully determine which is the best move, i.e. they are based on a theory of "bounded rationality".

Imagine two economic competitors who make their decisions in an uncertainty context: their main goal is "to beat" the opponent exploiting a possible advantage. The players do not know exactly which is the real situation and if they have a competitive advantage, thus they need to collect as much information as they can. In Kriegspiel the owning of information is a real competitive advantage, and in order to gain information the player can try to purposely play illegal or risky moves. Each player has to face a trade off between the information and his cost, just like in a real economic situation.

3. Solving a Kriegspiel problem with substantive rationality

To approach Kriegspiel with substantive rationality means that we try to obtain normative and descriptive results using the tools provided by classical Game Theory.

Game Theory is an indispensable tool for descriptive analysis of games, because it allows to build an abstract model of the game to be studied and to classify the interactive strategic situation in the set of already known models.

According to Owen's taxonomy [Owe82], Kriegspiel is a multistage game with imperfect information, and the correspondent game tree is so large that it is impossible to be fully explored (as for Chess). The main difference between Kriegspiel and Chess is that the latter is a game of *complete information* while Kriegspiel is a game of *incomplete information*. The Zermelo theorem, which proves that complete information games like Chess are always solved with a *pure strategy*, is not valid for Kriegspiel. Moreover, it does not make sense to apply the minimax algorithm in order to find the best strategy in a given position. In most Kriegspiel problems there is an asymmetry between the information available to the players and in most stages of the game the players do not know which are the possible strategies of their opponent.

This does not mean that usual Chess-like analysis of a game tree is impossible. For instance, in a particular position of the endgame King+Pawn vs. King, Game Theory provides very useful tools in order to gain not only descriptive results but even normative results.

Suppose that the position of Fig.3, where Black has to move, is completely known to both players. Thus, before Black moves, White knows where the black king is (d7). On the other side, Black knows where the white king and pawn are. Apparently this problem is not different from a Chess problem, but there are two big differences.

The first one is that after the first move of Black, White will not know the black king location (it could be in c8, d8 or e8). The second is that already at the beginning of the problem White does not know what Black knows about white's pieces location, and at the same time Black does not know what White knows about black's king location.

```
††††††††
¬0~0~0~0~®
¬~0~k~0~0®
¬0~0∏P0~0~®
¬~0~K~0~0®
¬0~0~0~0~®
¬~0~0~0~0®
¬0~0~0~0~®
¬~0~0~0~0®
ſſſſſſſſ
```

Fig 3. *Black to move.*

We model the players' knowledge by the concept of *information set*.

Definition: The information set is the set of possible positions compatible with the knowledge which a player got from past moves. We introduce a special notation: U_1^{black} denotes the Information set 1 for black player.

An information set can be composed by one element only, that is the only possible real situation, or by several alternative element because after an unknown move of the opponent, a player must take into account several different possible situations.

We are ready to build the following game tree (Fig. 4) whose nodes are information sets. From each information set one or more arcs start, each of them representing a possible strategy. Arcs labeled “try” represent moves that dominate every other one since they produce information at no cost. For instance, given the information sets U_2^{white} and U_3^{white} it is always convenient to play Kc7 and Ke7 respectively, because if the move is legal White wins and if the move is not legal White can play another move gaining further information.

Note that we pruned the dominated strategies. This means, for instance, that we do not consider the strategy Pe7 in the information set U_1^{white} because it would be a free “gift” to the opponent: it is a dominated strategy.

The tree leaves indicate the results of the strategic interaction between the two players: victory for White, draw, or reiteration of the game.

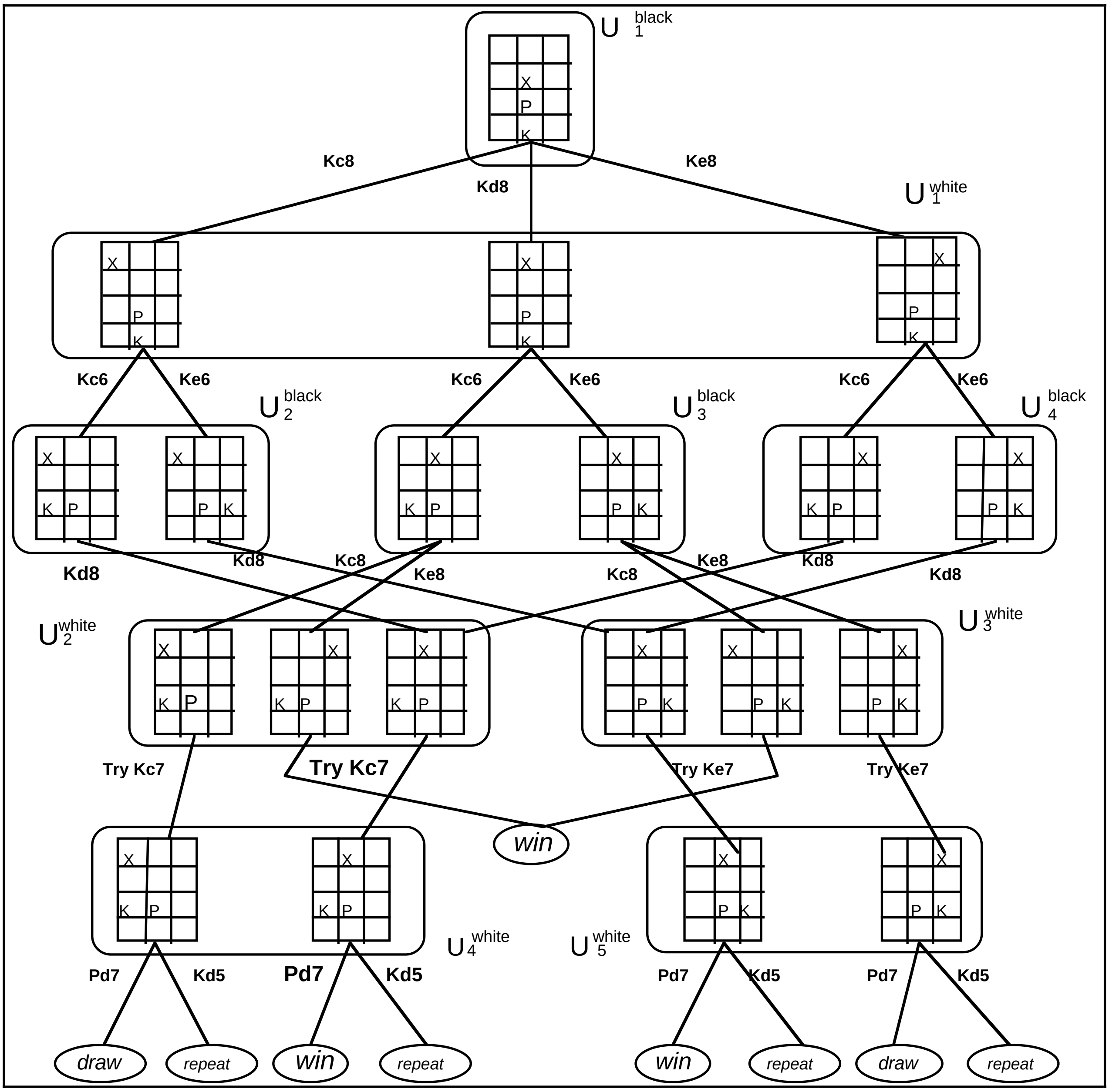


Fig.4 *A tree of information sets*

To simplify the analysis we transpose the extended form representation in the normal tabular form resuming the following strategies (α and β) available for the opponents (the symbol § means: "wait for the opponent move, and then play...")

Black :	White :
β1 = **Kd8**.§.**Kc8**	α1 = **Kc6**.§.(try **Kc7**) if not *win* then **Pd7**
β2 = **Kd8**.§.**Ke8**	α2 = **Ke6**.§.(try **Kc7**) if not *win* then **Pd7**
β3 = **Kc8**.§.**Kd8**	α3 = **Kc6**.§.(try **Kc7**) if not *win* then **Kd5**
β4 = **Ke8**.§.**Kd8**	α4 = **Ke6**.§.(try **Kc7**) if not *win* then **Kd5**

	β1	β2	β3	β4
α1	0	1	1	1
α2	1	0	1	1
α3	Γ	1	Γ	Γ
α4	1	Γ	Γ	Γ

It is possible to eliminate a column, since strategies β3 and β4 are equivalent. The result is

	β1	β2	β3
α1	0	1	1
α2	1	0	1
α3	Γ	1	Γ
α4	1	Γ	Γ

The payoffs are :
- 0 The game is draw.
- 1 White wins.
- Γ The game iterates.

This is the representation in normal form of a recursive game. Such a game is very similar to a classic problem in the literature of recursive games: the problem of Colonel Blotto [Owe82].

> *With three units, Col. Blotto must capture an enemy outpost defended by two units. He must, however, be careful: while he attacks the enemy outpost, his adversary should not capture his own camp. An attacker needs one more unit than the defending forces to be*

successful; if the attacking force is not large enough, it simply retreats to its own camp again, and the game starts again the following day.

The payoff is +1 if Blotto captures the enemy outpost without losing his own camp, -1 if the enemy captures Blotto's camp, whereas the payoff Γ means that the game restarts. The strategies in the game simply correspond to a division into attacking and defending forces; thus Blotto has four strategies, corresponding to 0, 1, 2, and 3 attacking units, respectively, while his opponent has three strategies. The corresponding matrix is :

	α_1	α_2	α_3	α_4
β_1	Γ	Γ	Γ	1
β_2	Γ	Γ	1	-1
β_3	Γ	1	-1	-1

The value of this game can be seen to be +1; Blotto's "ε-optimal" strategy will be of the form of playing strategies α_1, α_2, α_3, α_4, respectively with probability 0, 1-δ-δ2, δ, δ2. Now the smaller is δ the larger will be the probability of victory for Blotto, and the larger the expected length of the game will be. Thus, it seems that patience on behalf of Blotto is important here, because the optimal solution is correlated with the patience of the Colonel for waiting the final attack.

In the Kriegspiel endgame the final attack is the pawn push. The game can be solved giving to White an ε-optimal mixed strategy which consists of playing strategies α_3 or α_4 with probability respectively (1-ε/2) , (1-ε/2) and α1, or α2 with probability ε/2 or ε/2. The smaller is ε the closer to 1 will be the probability of a White win.

Such a solution can be explained as follows : if White played α_3 or α_4 with probability 1, he could not lose because he never risks the capture of his pawn, but the opponent, realizing this, would play systematically β_3 forcing the infinite reiteration. In order to break this loop, White should once in a while risk playing α_1 or α_2.

The critical position solved with substantive rationality is just a small part of a much more complex problem for White: *win* (if it is possible) *whatever the starting position is*. Solving such a problem with substantive rationality would require a huge computational effort, and even if we could make such an effort, the resources spent would not be compensated by the results, because in Kriegspiel uncertainty plays such a big role that any long range planning seems impossible. It would be like a company in a competitive market, which, waiting for its competitors move, should prepare hundreds of alternative strategic plans in order to use only the one it will fit the next competitor's move. Even if having a already ready strategic plan is a good thing, the cost of such a preparation does not pay for the huge effort needed to compute all the other plans that will be never used.

4. A Kriegspiel playing program

Our goal is to build an artificial player which acts consistently in a procedural rational way in order to reach its goal (in the problem in Fig.1 the goal is the promotion of the pawn) using a knowledge base.

We will see that thanks to procedural rationality we can effectively use a knowledge-based approach instead of a brute force approach to Kriegspiel playing. Most knowledge comes with experience and learning (in the program we have built, we have used rules for playing chess pawn endgames [Bra86]), but part of the knowledge base comes from the substantively rational approach. In our case the Game Theory model of recursive games belongs to the knowledge base of the artificial player. The interesting part of such an experience is how the procedural rational approach can incorporate results obtained with substantive rationality, while the two approaches are usually considered mutually exclusive.

4.1 The architecture of the artificial player

We have built in Prolog a Kriegspiel program able to play any ending KPK, namely King and Pawn versus King [Mar93]. Two agents cohabit in the same program: the artificial player and the referee.

The *artificial player* is an expert system able to deliberate a strategy in order to win (if it is possible). It is able to choose a move consistently with the goal of pushing the pawn to the eight row without being captured.

The *referee* is a module whose aim is to verify the legality of the moves proposed by both the artificial player and the user. If the move proposed by the player is illegal or it provokes a check the referee will inform the player. Another important role of the referee is to inform when the game is finished because of draw, stalemate, or checkmate.

The *user* is the opponent of the artificial player. He moves the black king and observes on a display the behavior of White.

All information used by the program is contained in the following data structure.

`(Whomoves, WK, WP, Wave, Taboo, BK)`

Whomoves indicates who has to make the next move.

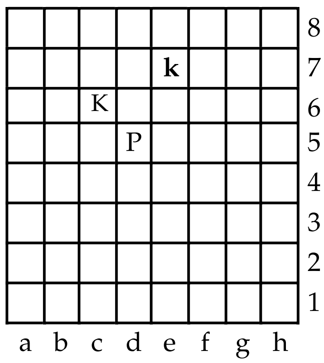
WK, *WP* and *BK* are respectively the coordinates of the White King, the White Pawn, and the Black King.

Wave and *Taboo* are lists consulted only by the artificial player which has access also to all the information but *BK* (if he knew *BK* he would play Chess instead of Kriegspiel).

Wave are the hypothetical coordinates of the squares where the black opponent could be located.

Taboo are the coordinates of the squares where the white king cannot move; such a list is updated whenever the referee gives new information. Once the artificial player knows the list *Taboo*, he can update the list *Wave* through a logic inference procedure. (e.g.: if the white king is in e6 and I try to play e6-e7 and the referee says *move illegal*, then the square e7 is the first element of the list Taboo.)

Fig.5a is an example of the codification of a position for the needs of the referee.

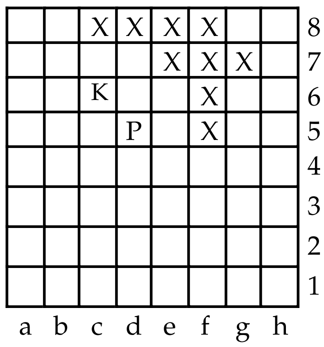


Whomoves = black
WK = c6
WP = d5
BK = e7

Information for Referee

Fig 5.a

Fig.5.b depicts what is the state information used by the artificial player, who has White (in the starting position Black has to move).



Whomoves = black
WK = c6
WP = d5
Wave = c8,d8,e8,f8,e7,f7,f6,f5,g7
Taboo = [ ]

Information for White

Fig 5.b

The list Taboo is empty because White has not tried to move yet, so he did not gain information from the referee.

Fig.6 shows the general flowchart of the program: actions in the circle are executed by the user, actions in rectangles are executed by the artificial player, questions are answered by the referee.

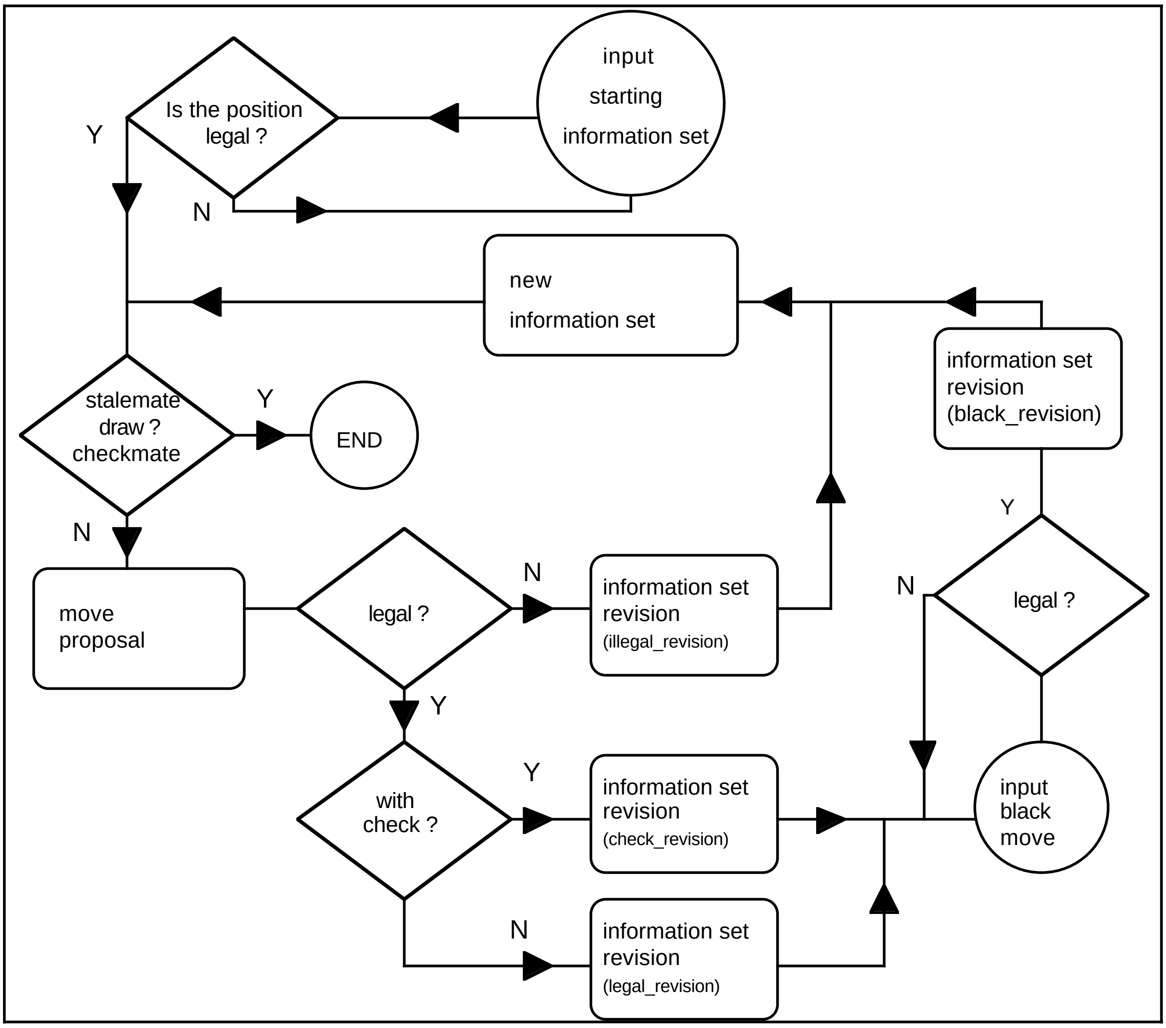


Fig.6 *General flowchart of the Kriegspiel playing program*

After the user inserts the starting position and the referee has checked its legality, the artificial player evaluates his information set and proposes his move.

The proposed move is evaluated by the referee who has to declare if it is illegal, legal or legal with check. In each case the referee suggests to the artificial player a revision of its information set, with the peculiarity that if the move is illegal, the artificial player can try a new move.

If the move is legal (with or without check), after the information set revision, the program asks the user for his move. The program terminates when the artificial player queens the pawn by pushing it to the eight row, or when the pawn is captured by the black king, or by stalemate.

4.2 A knowledge base for Kriegspiel

The artificial player (AP), instead of building the branches of the game tree, tries to match the features of the position recognized in his information set with the patterns that are embedded in his knowledge base. Every pattern is linked with a kind of move or with a list of moves (if the first move of the list is illegal AP will try the second one, and so on).

The implementation of the algorithm was quite simple in Prolog, since the choice of each move is ruled by the satisfaction of a sequence of goals.

White information set consist of 4 variables WK, WP, $Wave$ and $Taboo$, the combination of which generates the universe of possible situations.

The problem space can be classified by 5 possible patterns; AP has to match the patterns with the position. The following scheme introduces the five patterns, and shows the kind of move linked with it. Then we have a brief description of each pattern.

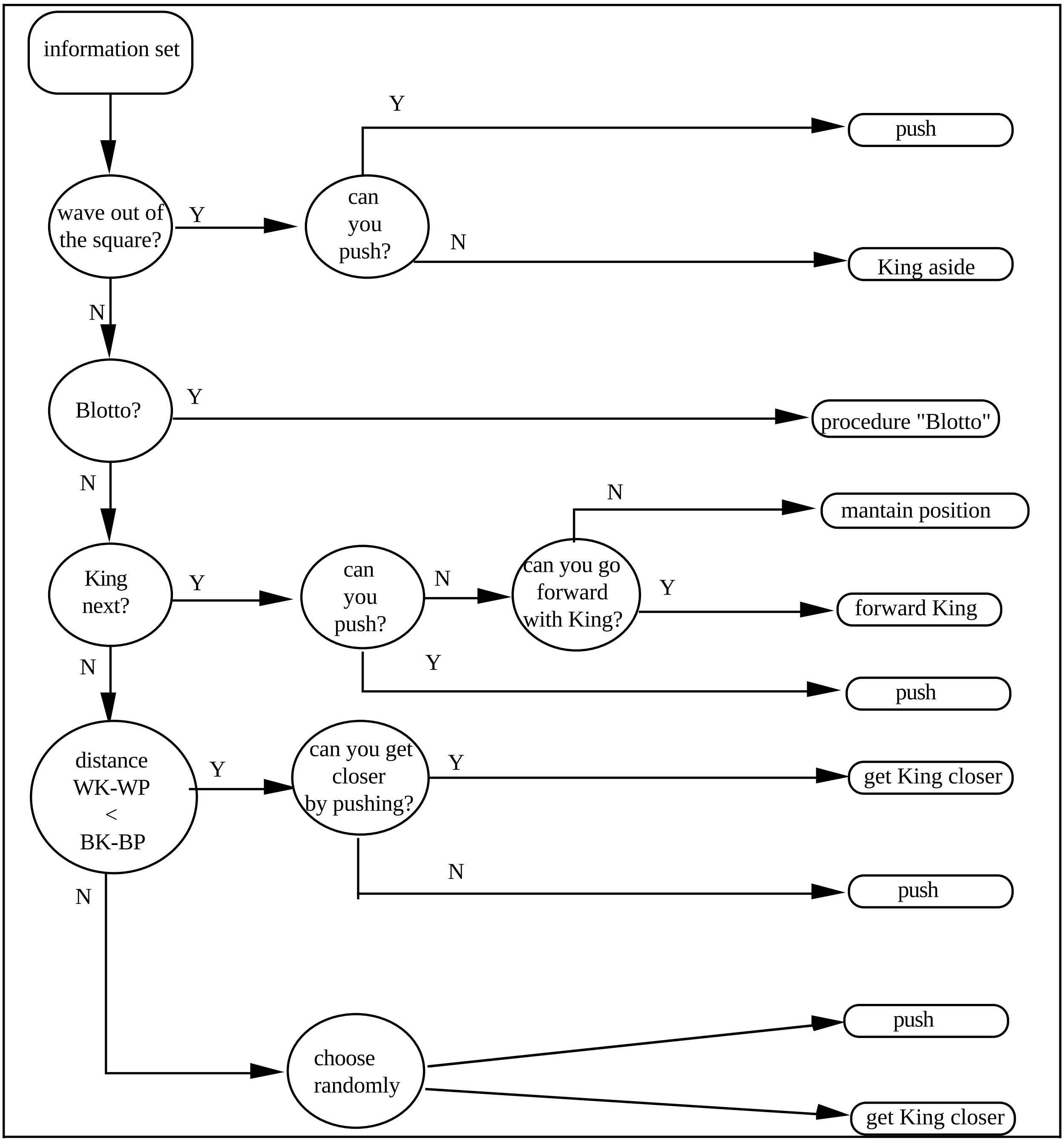
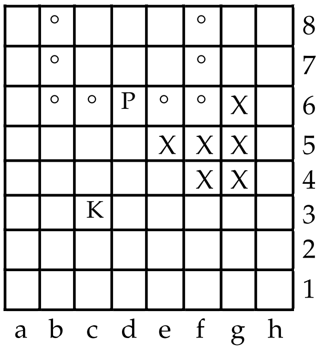


Fig.7 *Structure of the knowledge base*

Wave out of square.

This is the first test; it checks if the position in the information set allows the pawn to reach the eighth row without being threatened by the black king. The test is based on the well known chess endgame rule called "rule of the square", the only difference being that each element of the set [*Wave*] has to be out of the square. If the test is positive, the only thing to do is to push toward the victory. If the way is obstructed by the white king, the program will try to move the king aside.

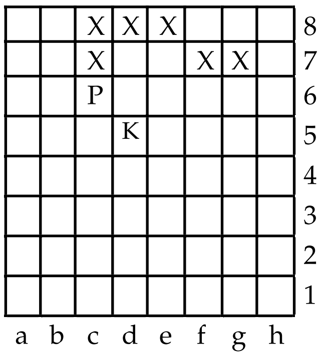


The symbol ° traces the edges of the square: if any element of the set [*Wave*] is included in the square, the pawn can be pushed toward the 8th line.

Fig. 8

Blotto:

The program calls test "Blotto" if the first test gives a negative response. The positions handled by this pattern are analog to that studied in the Sect. 3 with substantive rationality.



The position satisfies the test Blotto since

1. The white pawn is in the 6th line
2. The white king is next and behind the pawn
3. Part of the list *Wave* is included in the square

Fig 9

The solution obtained with Game Theory produces the best strategy in order to win. This advice has been inserted in the knowledge base of the artificial player and it is used when test Blotto is successful. We remark that the results obtained with substantive rationality are embedded in a procedural approach, in order to improve the final result.

With reference to the position illustrated above, the probabilistic automaton below describes which is the substantive strategy to follow. Arcs indicates the move to play as first attempt, the hatched arc indicates the move to play if the first attempt was illegal. If two different arcs start from the same point, a probability distribution is given to them.

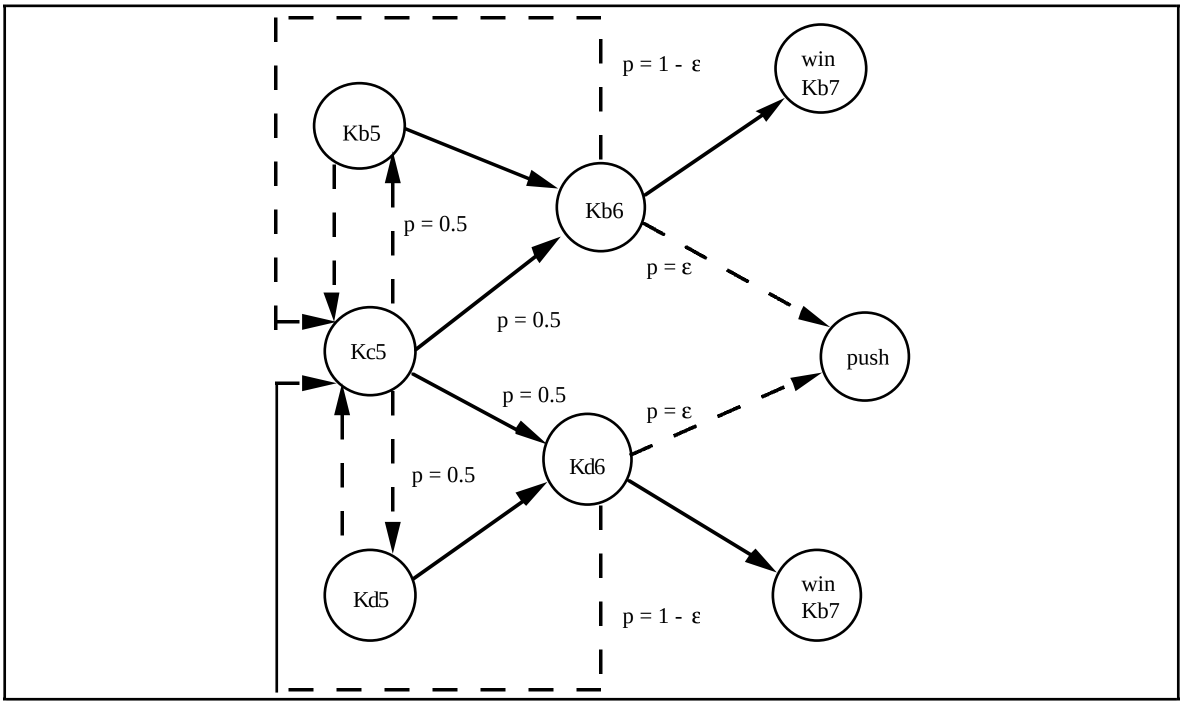


Fig.10 *Probabilistic automaton representing the Blotto test*

King_next:

If the first two tests fail, "king_next" verifies if the white king is located next to the pawn, if yes the artificial player will try to satisfy the list of goals (Push, Forward_king, Maintain_Position).

Goals are ordered: first the player will try to satisfy “push”; if this is not possible then it tries “forward_king”, and if this is impossible “maintain_position” is always satisfiable.

The list *Taboo* is used by the artificial player in order to understand if a goal is satisfiable or not. It can find the set of squares that are inhibited to the white pieces. *Taboo* is a dynamic list and it changes in the course of the game with the information given by the referee. The satisfiability of the move is subordinated to a primary goal that is to protect the pawn.

Distance *WK-WP < BK-WP*

This test verifies if the distance between the white king and the pawn is smaller than that the distance between the black king and the pawn (the “rule of the square”). Since we do not know exactly where is the black king, but we have a list of squares where he could be, we have to check as follows :

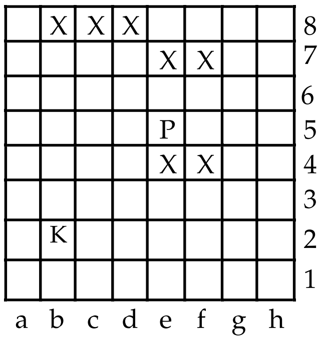
$$\text{Dist}(WK, WP) < \min_{X \in \text{Wave}} \text{Dist}(X, WP)$$

If the test is positive, the strategy to follow is to get closer with the king in order to gain the position described in the previous test.

Distance *WK-WP > BK-WP*

If all tests fail, the position will be handled by this pattern.

While in the previous situations the artificial player is able to win with a probability close to 1, in this kind of situation is possible that the black king captures the pawn and the game finishes with a draw.



If the black king were in e7 or f7, and Black knows where the pawn is the white king cannot protect the pawn.

If the black king were in b8,c8 or d8, the pawn could be protected by king. However, the black king could also be in e4 or f4 : in this case the best thing to do is pushing.

Fig. 11

The situation illustrated above is of uncertainty. If we had reasons to think that black king is in e4 or f4, we should push, but since we have no hint we choose randomly between the strategy *push* or *get king closer*.

4.3 The revision of the information set

The revision of the information set is another important operation executed by the artificial player, which tries to understand the position with the help of the information given by the referee.

The revision process includes four procedures:

If the referee declares the move is legal: legal_revision.

If the referee declares the move is legal with check: check_revision.

If the referee declares the move is illegal: illegal_revision.

After black move: black_revision.

legal_revision:

After a white legal move the structure of the information set is modified. The variables *WK* or *WP* will change with the new position of the moved piece.

We can also gain information to change the list *Wave*. For example, in Fig.12a the list *Wave* contains the squares [d8,e8,f8,e7,f7,d6,d5].

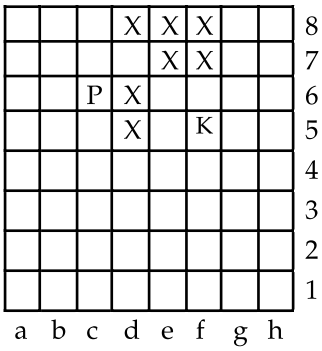


fig 12.a

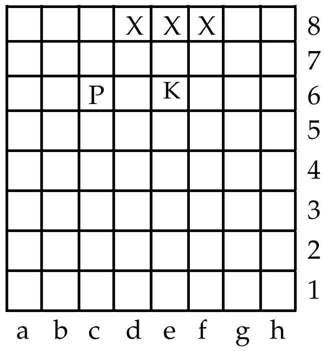


fig 12.b

If the move Ke6 is legal, the new *Wave* will be the one in Fig.12.b, since the set of squares [e7,f7,d6,d5] is incompatible with the allocation of the white king in e6, hence the new *Wave* will be the list [d8,e8,f8].

check_revision:

This is similar to the test described above: if the referee declares that the move is legal with check, the black king is certainly located in one of the two squares threatened by the pawn.

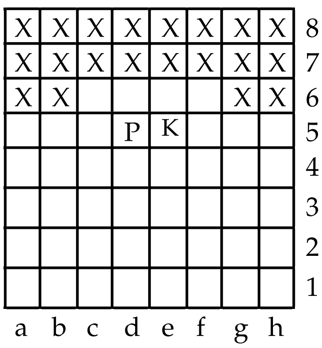


Fig 13.a

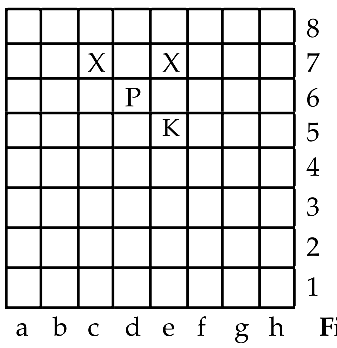


Fig 13.b

illegal_revision: The revision after an illegal move does not change the variable *WK* or *WP* in the structure, but only the lists *Wave* and *Taboo*.

Let Fig.14 be our information set, and the move Ke6 have just been declared illegal. The square e6 is not accessible for White, so we add it to the list *Taboo* (we remind *that Taboo* is a list where White stores all the squares he cannot reach)

	X	X	X		X	X	X	8
			X			X	X	7
			P			X	X	6
				K				5
								4
								3
								2
								1
a	b	c	d	e	f	g	h	

Fig 14

The procedure *illegal_revision* also updates the list *Wave*, on the basis of the logic implication with the squares in *Taboo*.

With reference to Fig.14 : if e6 belongs to the list *Taboo*, then the black king must be in a square next to e6. Since d7 is the only square in *Wave* and next to e6, for sure the black king is located in d7 which will be the only component of the new list *Wave* [d7] (see Fig.15).

								8
			X					7
			P					6
				K				5
								4
								3
								2
								1
a	b	c	d	e	f	g	h	

Fig 15

black_revision : This procedure updates 3 variables: *BK* is the location of the black king, that is known by the referee only, the list *Taboo* is cleaned, and the list *Wave* is updated according to the fact that the black king has been moved. See Fig 16.a and Fig.16.b for the contents of the list *Wave* before and after the move of the black king.

								8
	X					X		7
			P					6
					K			5
								4
								3
								2
								1
a	b	c	d	e	f	g	h	

fig 16.a

X	X	X			X	X	X	8
X					X		X	7
X	X	X	P				X	6
					K			5
								4
								3
								2
								1
a	b	c	d	e	f	g	h	

Fig 16.b

5. Conclusions

We have built a program based on knowledge alone: it makes no search at all in the problem space. Its main task consists of associating every possible information set to its knowledge base. If a match is found, the move to play is found in the list associated to the pattern found in the knowledge base. This approach is simple and powerful because the problem we faced (King+Pawn vs King) is very simple.

However, more complex Kriegspiel problems can be solved in our approach, for instance adding more pawns, or playing simple endings including pieces only as in [Fer92].

Our overall goal is to build a fully fledged kriegspiel playing program, able to both play and solve problems as those contained in [And58,Li95].

REFERENCES

[And58] G.Anderson, *Are There Any?*, England, 1958.

[Bra86] I.Bratko, *Prolog Programming for Artificial Intelligence*, AddisonWesley, 1986.

[Fac89] Faccipieri, "L'analisi strategica", in Rispoli (ed.) *L'impresa industriale* , Il Mulino 1989.

[Fer92] Ferguson, "Mate with bishop and knight in Kriegspiel", *Theoretical Computer Science*, 96, 389-403, 1992.

[LeoMag78] M.Leoncini, R.Magari, *Manuale di scacchi eterodossi,* Siena, Tipografia Senese (1978).

[Li94] D.Li, "Kriegspiel: Chess Under Uncertainty", Premier Publishing, 1994.

[Li95] D.Li, "Chess detective: Kriegspiel strategies", Premier Publishing, 1995.

[Mag85] R.Magari, "Giochi matematici ad informazione parziale", *Sapere*, April 1985.

[Mar93] F.Maran, "Razionalità sostanziale e procedurale in una variante eterodossa del gioco degli scacchi", Tesi di Laurea, Fac. Economia Venezia, 1993.

[Owe82] Owen, *Game theory*, Academic press, 1982.

[Pri94] D.Pritchard, *The Encyclopedia of Chess Variants*, Games and Puzzles Publications, 1994.

[Sim76] H.Simon, "From substantive to procedural rationality", in *Method and appraisal in economics,* Cambridge Univ Press, 1976.

[Sim78] H.Simon, "On how to decide what to do", *The Bell Journal of Economics*, 9:2, 1978.

[SimSha92] H.Simon and J.Schaeffer, "The game of Chess", in Aumann, Hart (eds.), *Handbook of Game Theory with economic applications*, North Holland, 1992.

[VNeMor47] J.Von Neumann, O.Morgenstern, *Theory of games and economic behaviour*. Princeton Univ. Press, 1947.

[WBB72] C.Wetherell, T.Buckholtz, and K.Booth, "A Director for Kriegspiel: a variant of Chess", *The Computer Journal*, 15:1, 66-70, February 1972.