Adelson - Velskiy 1 -

3

Coll	SC340	Title	
Box	54	Series	1986-052
Fol	3	Fol. Title	

chess playing programs
sim cog proc
game playing
USSR cybernetics

adel 1

G.M. Adelson-Velskiy, V.L.Arlasarov,
A.G. Uskov

PROGRAMME PLAYING CHESS

A report on the sympsoium "Theory and Computing Methods in the Upper Mantle Problem"

§ 1. Stating the Problem of Composing a "thinking" programme

The problem of the nature of human thinking and of the possibility to model it is stated long ago. There are no reasons to have doubts of principal possibility to model the thinking because it is a type of motion of matter and hence is cognizable as all such processes. However, an approach to this problem and its scientific stating appeared only after the creation of automatic computers with the programme control and after the wide study of their possibilities.

Of course, at present there can be no reason of solving the problem to model thinking. By the authors' opinion this work is a preliminary study which can be useful in solving this problem.

From the cybernetic viewpoint the thinking is a process of working on information and the main obstacle to construct computers or to compose programmes for existing computers which could solve problems requiring the creative human thinking is that just the character of such a working on information is not available.

When studying the processes of working on information in thinking it is almost impossible to use recent scientific achievements. Psychology and physiology of brain do not state this problem yet. As to the cybernetics, its part devoted to specific forms of working on information is practically not developed. Actually, this part of cybernetics is worked out just in the study of the character of working on information in the thinking.

This study starting practically in the blank space goes on experimentally.

Computers are constructed or programmes for existing computers with programme control are composed to perform operations which by the opinion of their authors associate with thinking. In these investigations the main tool is programming because it displays the most resourcefulness. Besides just the experience in such a programming defines which new qualities an automatic computer must have to increase its possibilities. The authors consider it possible to call the above mentioned programmes as "thinking". This work is devoted to a description of such a "thinking" programme - the programme playing chess.

The choice of the problem for which the "thinking" programme is composed is determined by the author's hypothesis on the character of the creative thinking.

This hypothesis naturally arose from self-observation in solving mathematical problems (such a self-observation is very difficult to carry out and it always occurs to be incomplete). This observation shows that the result of thinking process is an argument, i.e. a chain of conclusions based logically upon assumptions which are considered as true before solving the problem and upon intermediate results obtained in the process of this solving. It can be observed also that in the process of solving one has to operate with unstrict arguments in which not all assumptions are proved. However, it is very difficult to determine how assumptions are selected in the solving. Particular assumptions arising in solving process are fixed in the memory. Some of them actually appear in the solution, another ones occur to be untrue and other bear no relation with the solution. But all fixed assumptions are a drop in the ocean amidst all possible assumptions which can associate with the solution. This leads to the viewpoint that the assumptions fix in the memory in result of a preliminary selection carried out mainly in the subconsciousness.

It is necessary to note also that elements of the solution of the stated problem are searched associatively (these elements are searched by association with known solutions of problems or particular methods of their solving, which are in some sense similar to the given problem). Besides, most lucky ideas arise in result of intuition (a sudden dawning).

The authors proceed from a conjecture that there does not exist the intuition as a particular type of thinking, i.e. all unawaited new ideas come in result of a subconscious brain work which is governed by the same rules as its conscious work available for the self-observation.

Thus the problem of composing a "thinking" programme is formulated as follows.

To compose a programme for an automatic computor to solve a problem from a definite field of problems. The solution of the problem must arise in result of analysing chains of elementary occurrences. These occurrences must be selected from a great number of possible occurrences.

It is necessary to note that such a character of a "thinking" programme is based upon the authors' assumptions about the character of the thinking. In general, a programme must be called "thinking" if it realizes an algorithm associated by the composer's opinion with the thinking or devoted to the solving of a problem which by the composer's opinion cannot be solved without the creative thinking.

§ 2. Requirements for the problem to solve which a "thinking" programme is composed

It is easy to see that the above determination of the "thinking" character of a programme has to the considerable extent a subjective character. Thus, starting the work on the specific "thinking" programme we must not forget that this work may have nothing to do with the study of the charachter of the human thinking. Besides, methods to analyse the composed programme and results of its work for the sake of drawing conclusions about the character of thinking are not available.

So the necessity arises to have objective criteria to evaluate the performed work and its results. By the authors' opinion this leads to the following requirements for the problem to solve which a "thinking" programme is composed.

1. The strict algorithm for solving the problem, if it exists, must not be realizable because of the great amount of memory or too great number of operations necessary to realize such an algorithm (in the last case the programme does not pass in time). Otherwisd it would be possible to compose an ordinary computing programme solving the given problem. Such a programme to all appearance will have no elements related to process of the thinking. At the same time, if a programmer will compose a programme founded on the other principles which by his opinion associate with the thinking, it will occur to be impossible to determine to which extent elements of the known precise algorithm are used in such a programme.

Besides, by the authors' opinion, in this case the evaluation of results achieved in such an "unautomatic" programme will be too subjective.

2. Thus, the considered programme may deliver the solution of the given problem withopt the complete foundation of its choice and this solution may prove to be untrue. In this connection objective criteria of the success must exist in solving the given problem. The composed programme must be evaluated on the basis of these criteria. In result of such an evaluation possibilities arise to improve the programme which can lead, for example, to finding new methods of working on information or its selection.

In particular it follows from this requirement that the programme must actually work and deliver results which can be evaluated by an objective criterion. Thus, an algorithm realized by a "thinking" programme will not require too great an amount of memory.

The problem must not be "hopeless", i.e. there must be possibilities to obtain from the programme more or less reasonable solution. Otherwise it will be difficult to make use of the objective criterion of the success.

to state the problem of composing a programme

From this viewpoint it is obviously not yet time/to prove one of known unproved theorems.

An objective success criterion for such a programme can be only the strict prooof of the theorem or construction of a contradiction. However the hope to obtain such a result is as yet too small. Any other result delivered by such a programme will be too difficult for an objective evaluation.

The problem to compose a chess programme, i.e. the programme chosing a move in a given position satisfies all these requirements. Indeed, the only known algorithm of finding the best move - the analysis of all possible variants and chess positions - require for its realization on the most high-speed computers an unconceivable time or an unconceivable amount of memory. Thus programmes realizing this algorithm practically does not pass. It is necessary to note here that there is a viewpoint about the existence of quick algorithms modelling subconscious processes in brain in solving such types of problems. Finding of such an algorithm, if it exists, assuredly associates with the study of the nature of the thinking. The authors do not agree with such a viewpoint; however, there are not enough data to disprove it. The scheme of chess programmes to which this work is devoted can be used to search and to realize such algorithms; however in accordance with authors' assumptions all the investigations are performed in the other field. Further, there exists a good success criterion for the chess programme; the evaluation of strongness of chosen moves in various positions can be used as this criterion. It is possible to carry out, and they actually were, games of chess between the chess programme and a man and between the chess programme and itself; on the grounds of these chess-games conclusions can be drawn about the chess strongness of the programme and about its weak points and possibilities to improve this programme. At last the examples of chess-games given in this work show that the problem is not "hopeless".. In these chess-games the programme on the whole approached to the strongness of the 4th class in the chess-opening and in the beginning of the middle game. Besides, the programme can be improved.

§ 3. Problems of improving the programme and of time to choose a move.

The simplest way to improve the programme is to increase the number of variants to be considered and the analysis of these variants for greater number of moves.

To achieve this it is sufficient to change several constants in the programme. However, such a change will lead to a strong increase of the time which the computer spends to choose a move. As it was mentioned above giving up the limitations on the time to choose a move it is possible to compose a programme realizing the method of complete analysis (though such a programme cannot work). That is why the authors consider as realizable programmes spending normally several minutes to choose a move and in more complicated positions not more than 1.5 hours for this choice. Thus together with the problem of improving the working programme the problem arises to speed up this programme. This problem directly relates with the problem of improving. Indeed, if such a speeding is achieved the constants mentioned above can be changed. It is necessary to note that there exist other methods to strengthen the programme without an essential increasing of the computing time; these methods are mainly connected with more precise evaluation of positions.

However, the most perspective ways to strengthen the programme require speeding of the work of its existing parts.

The authors consider the methods of such a speeding which appeared after the analysis of the performed work to be one of the most important results. The first of these methods associates with improving the programming technique. It is necessary to analyse all the parts of the existing programme and to replace subprograms realizing their tasks not in the most quick way by better ones. This apparently will lead to increase in the speed of work 1.5-2 times. The second way of the speeding is given by N.I. Bessonov. He had shown[7] that the main elementary operation of a chess programme-finding all squares where the given chess-man can move - can be realized as an elementary operation in the computer. Some other operations which often appear in a chess programme can also be realized as elementary computer operations. It is important to note that such a method of speeding to all appearance exists for any "thinking" programme.

By the authors' opinion, this method of speeding is in some way realized in human thinking. In studying the chess play or some other specific field of intellectual work complexes of cells in brain are created; these complexes are the instruments which perform such elementary operations.

Authors suppose these complexes to be created in the brain by coupling between cells prepared beforehand but not switched on. However, the character of such a coupling is not available at present.

It is possible to project of the possibilities of speeding associated with the consideration not all the chess-board and not all the chess-men on the chess-board in some variants.

Let a chess-position be given, for example, as shown on the diagram 1.

In the 3-semi-moves analysis (i.e. a move, an opponent's answer and one more move is considered) one of the working programs constructs an analysis tree for the given position shown on Scheme 1.

Thus the analysing scheme does not contain all the possible variants corresponding to the rules of the game of chess.

In realization of above-mentioned methods of speeding the programme the analysis tree does not change.

That is why the speeding resulting from these methods does not depend upon the depth of computing variants.

The algorithm which determines the tree of variants analysed in the given programme will be called the chess position analysing scheme.

Let n be the depth of the analysis of the variants (i.e. the number of semimoves for which the computation is carried out), t is the time of choosing a move. For the schemes considered in this work t depends exponentially upon n:

$$t \quad C^n \qquad (1)$$

Here the coefficient C characterizing the given scheme depends upon the number of branches in ~~the~~ every move. Thus when the number of branches decreases the time to choose a move decreases experimentally, and the speeding increases with the increase of n. A preliminary selection of moves in the analysing scheme and earlier cutting some of the variants leads to a decrease in a number of branches. Thus a scheme must be obtained composed of a comparatively small number of weakly branched variants with great computation depth. A corresponding work is performed by a chess-player mainly subconsciously which leads to a difficulty in constructing such a scheme. In the consciousness traces of productive work - a search of a needed move - are left. Similarly the chess theory is in the main composed of instructions having a productive but not selective character.

In the working programme specific methods to select moves in the analyzing scheme are realized. However in this programme only a preliminary selection works which is founded upon a specific analysis of appearing positions.

In ananlysing the variants in the chess position analysing scheme not the complete information is extracted from these variants. In fact only evaluations of positions appearing in these variants are used in the programme. At the same time the choice of moves for the chess position analysing scheme is associated with specific ideas (in the working programme they are mainly ideas related to a threat of a material gain); in result of computing a particular variant it is possible not only find out whether a given idea passes but as well to obtain an information about its specific properties being an obstactle to realize this idea. This information can be used to determing the analysing scheme more exactly.

Thus the questions arises about the recording, storage and use of information obtained in the process of programme working.

The complete storage of the information can be carried out in the form of a table of appeared positions with their evaluation as this was made in the work of A.S. Broodno and Landau "The programme of the play 'singlecoloured'".2)

However, the identical positions in different variants occur in the chess-play not too often to have a considerable effect in using these tables. At the same time positions often occur which are not identical but equivalent from the viewpoint of carrying out some idea. In using the table of occurred positions to implify analysing the equivalence of positions with respect to a specific idea the computing time cannot only decrease but even increase. It is necessary to develop methods to compare quickly the given position with the positions from the table.

It is also possible to outline other ways to use an information obtained in the programme computing process.

The possibilities are interesting to speed the program by use of several arithmetic units working simultaneously and of the parallel access. These possibilities cannot be realized because computers with several arithmetic units and the parallel access are not available at present. However some data of organizing an operation with such parallel working units can be obtained analysing the working chess programme.

Except the authors in composing particular parts of the programme A.S.Kronrod took part. In discussions associated with this programme E.M.Landis, A.L.Broodno and N,I.Bessonov also took part.

§ 4. Forced moves and forced game[*]

In solving the problem of constructing an algorithm which differs from a complete analysis attempts are natural to use results of the chess theory and observations of chess-players' experience. However, the chess theory gives algorithms to choose a move only in some simple types of positions (a mate to the single king, some types of the pawn endgame), it being necessary to make the descriptions of these algorithms more precise to compose the programme. More complicated types of positions in the endgame are considered to be studied in the chess theory; however, the principles of this investigation are not described in terms of an exact non-contradictory algorithm. However, by the author's opinion, most interesting are the parts of the chess theory devoted to more complicated positions, especially to positions of the middlegame. These parts are a summary of methods used in games of chess by strong chess-players, the complete classification of these methods and non-contradictory rules to use them being not available. An analysis of particular variants is a criterion of using these methods. However, this analysis is not complete analysis, so it is not possible to formulate its principles. Moreover, in a sufficiently complicated position two grandmasters consider different variants (though they often come to the same conclusion).

Thus in the chess theory there do not exist rules to choose a move in a sufficiently complicated position with strictly proved applicability. By the authors' opinion this shows that the choice of a true move in the chess position is the problem requiring for its solving the creative thinking. In accordance with this conclusion the correctness of algorithms used in the programme needs an experimental verification but not a deductive proof.

Algorithms realized in the programme choose a move analysing variants. However, this analysis is not complete; in positions of these variants not all possible moves are considered and also variants are not brought to final positions. At the same time analysis of variants is organized as a complete analysis of a game of two opponents with a complete information and moves by turn. This game can be called a model game defining the algorithm.

Each position of the model game is a particular chess position and every move is a move permitted by the chess-game rules. Final positions in the chess-game, i.e. positions in which one of the kings is mated or drawn positions, are final also in the model game with corresponding results. However not all the moves permissible by the chess game rules are permitted in the model game, the rules of determination of possible moves depending upon the place of the position in the game tree G. Many positions of

----

[*] Translater's note. By the forced game in the chess theory a game is called when a player is caught in a position where no matter what move he makes he cannot escape certain defeat; by the forced move a move is called which a player is compelled to make. These terms have nothing to do with the terms "forced move" and "forced game" used in this work.

the model game are final though they are not final as chess positions.

Each chess position can be considered as an initial position of a model game. The programme evaluates a position and chooses the best move in the model game.

The result of the game in final positions is determined by the evaluation function . Besides the model game the programme's work is determined by rules of the order of examining possible moves. Thus model game positions are the chess positions situated in specified junctions of game tree G.

The important notions in the algorithms described below are that ones of a forced move and a forced game. To define these notions it is necessary to dwell on the question of the evaluation function $f(A)$ structure. If a position is not a final chess position then for a position with the move of White's the value of this function is defined in the form

$$f(A) = f_M(A) + f_n(A) \qquad (2)$$

Here $f_M(A)$ is the material position evaluation:

$$f_M(A) = C + \sum_{W.C.} Z_i - \sum_{B.C.} Z_j \qquad (3)$$

To the difference of the sums of the material evaluations for white and black chess-men a positive value C is added to make $f_M(A)$ positive. Material evaluations of chess-men are as generally accepted; they are given in the Table 1.

Table 1

pawn	knight	bishop	rook	queen
2	7	7	10	20

The material evaluation of the king does not exist because white and black sides present in all chess- positions. A simple calculation shows that c may be chosen to have any value greater than 228.

The positional evaluation function is defined in the next paragraph. It is sufficient to note here that

$$0 < f_n(A) < 1 \qquad (4)$$

In result of this algorithms considered here do not permit positional offers if the algorithm does not lead to the complete computation of such an offer. Authors consider this to be necessary on the recent stage of development of chess programmes because first of all the programme is needed which will not make blunders.

In final chess positions the result of the game is defined as follows:

$$f(A)=\begin{cases} 2C+1 & \text{, if the black king is mated} \\ C+\frac{1}{2} & \text{, if a drawn position} \\ 0 & \text{, if the white king is mated} \end{cases} \qquad (2')$$

If A is a position with the move of Black's then the value of the evaluation function for Black's is given in the form

$$f_B(A)=2C+1-f(A) \qquad (5)$$

Thus the discussion below only positions with moves of White's can be considered.

Let A_n be a position in the model game considered in the tree G of the complete analysis of this game; ℓ is the rank of this position (i.e. the number of semimoves leading to the position A_ℓ from the position A_0), m_ℓ is the lower bound of the evaluation of the rank ℓ on the n-th step in the algorithm when the position A_n is considered (in § 7 the bounds for the evaluation of the algorithm are discussed). Note that m_ℓ is the bound for Whites if A_n is a position with a move of White's and it is the bound for Blacks if A_n is a position with a move of Black's.

By the forced move in the narrow sense any capture is called which is permitted by the chess rules and after which the position A_q appears with the material evaluation satisfying the following condition

$$f_M(A_q)\geq[m_\ell] \qquad (6)$$

($[m_\ell]$ denotes here the entire part of m_ℓ)

By the forced move in the broad sense we will call any forced move in the narrow sense, any check and any answer to the check permissible by chess rules under the condition that the material evaluation of the position appearing after this answer will satisfy (6).

Using the notion of the forced move it is easy to define the notion of the forced game of two opponents with a complete information and moves in turn; this game will be denoted below by F.

Let a position A_0 be given, together with evaluation function $f(A)$ and $f_n(A)$ and also upper and lower bounds m_0 and M_0 for evaluation of initial position A_0. For any position A_q of the game tree G with the initial position A_0 the following ~~values~~ are determined: ℓ - the rank of the position A_q, i.e. the number of semimoves in the branch of the game tree G beginning from the position A_0 and finishing at the position A_q, and also upper and lower bounds M_e and m_e for evaluation of the position A_q if it is a position with the move of White or otherwise M_{be} and m_{be} for Black.

Allowed moves from the considered position A_q in the game F are all the forced moves and a so-called "empty move", or "declining the forced variant" after which the final position A'_q arises, the result of game in this position being equal to the value of evaluation function in the position A_q:

$$\tau(A'_q) = f(A_q) \tag{7}$$

Three variants of the forced game were tried. In the first variant only forced moves in the narrow sense are allowed; in the one all forced moves in the broad sense are admissible; at last, in the third variant a control constant of number of checks ϑ_{ch} is given, each opponent being allowed to make this number of checks. If in the chain of positions A_o A_1 ... A_q the number of checks which a given side has made is less than ϑ_{ch} then the forced moves in the broad sense are admissible in the position A_q. Otherwise only forced moves in the narrow sense are admissible.

§ 5. Specific indices of the chess position and their accounting in the evaluation function

The evaluation function is defined by position indices which are taken into account by chess-players in practical chess-games. Chess theory and practice show that the same indices have different meaning depending also upon a game plan contemplated beforehand and particular tactical properties of a position. However, to compose a chess programme which does not make gross blunders and does not get quickly into hopeless positions it is sufficient to have an evaluation function which accounts for different indices independently one upon another. Such an evaluation function is defined in the form:

$$f_n(A) = \frac{1}{2} + \sum_{W.n} \Pi_i З_i - \sum_{б} \Pi_j З_j \tag{8}$$

Here the summation is carried out over all indices taken by the programme into account and are evaluations of these indices. They had different values in different chess games played by the programme. By the authors' opinion selection of these values cannot considerably strengthen the programme.

Often static and dynamic indices of positions are discerned. By static indices these one are called which do not change in all the positions of considered variants. There do not exist absolutely static indices because any index can appear or disappear in a chess game. However in computing to a bounded depth many positions have indices of the static character. Especially often the indices are static such as the stage of a chess-game (opening, endgame, middlegame), the type of a position (open or closed position in the middlegame, types of the endgame), a pawn structure of mutually blockaded pawn chains.

One must not put independent terms in the form (8) in correspondence to static indices. Indeed, accounting for a static index changes values of the evaluation function by a constant and does not influence upon choosing a move beacuse the static indcx is present or absent in all the positions of considered variants. However, as the staticity of any index is relative to one of them (castling or its loss) which do not require much time for computation was taken into account in working programs for the position evaluation function $f_n(A)$.

It is possible to account for values of static indices to determine evaluations 3_i of these indices in the evaluation function $f_n(A)$. The character of the corr sponding dependence may be complicated and the corresponding programme for determining these evaluations may require relatively much computing time because this programme works before the programme of the complete analysis and do not increase its computing time. In working programs such a preliminary adjusting of the evaluation function is not carried out.

As any index accounted by the position evaluation function $f_n(A)$ programme must be determined in the all final positions considered in the complete analysis the amount of time required to determine the indices is of ~~the~~ great importance. Hence it follows that, as a rule, it is not expedient to introduce the determination of indices requiring the analysis of variants (such as, for example, the index of chess-man overburdening by defence functions) into the programme for the position evaluation function $f_p(A)$.

We give below the list of indices which can be accounted for with a permissible computing time for the accepted system of information. It is necessary to note that the choice of this information is associated with conveniences to compose the programme for the position evaluation function. We will give the chess sense of these indices and their exact definition. The indices can be divided into several types.

I. Indices associated with the pawn structure of the position

1) Pawns in the center. For White the central squares are e4, d4, e5, d5, e6, d6; for Black - e5, d5, e4, d4, e3 and d3. To account for the index the number of pawns of the given side is multiplied by the index evaluation given in the Table 2. In this table as in all the paragraph the evaluation is given in the units of position evaluation d which is chosen to satisfy the form (4) for any chess position.

2) The phalanx. Two pawns on the same rank and neighbouring files are called the phalanx. To account for the corresponding index the number of phalanxes of a given side located in areas indicated on the Diagram 2 is multiplied by the index evaluation 3_{ph} (see Table 2). The collection of k pawns on the same rank and neighbouring files is accounted for as k-1 phalanxes.

3) Isolated pawns.

4) Double pawns. In the programme only double isolated pawns and isolated pawns on an open line have a non-zero evaluation. If an isolated pawn is located on a line open for the opponent the term 3_{dp} presents in the position evaluation function; if k

isolated pawns are located on this line then the corresponding term is equal to $K З_{dp}$. Value of $З_{dp}$ is given in Table 2.

5. Passed pawns. The evaluation of each passed pawn depends upon the distance between this pawn and the queening square. In the programme the evaluation is defined by the form:

$$З_p = 32 - 4S \qquad (9)$$

where S is the number of ranks between this pawn and the queening square. It would be necessary to take into account for pinned passed pawns and supported passed pawns but this is not realised in the programme.

6. Weak point in the narrow sense. A square attacked by an opponent's pawn which cannot be attacked by a given side's pawn is called the weak point in the narrow sense.

7. Weak point in the broad sense. A square which cannot be attacked by a given side's pawn is called the weak point in the broad sense.

8. Detained pawn. We call so a pawn with a weak point in the narrow sense before it located on a line open for the enemy.

Of course this does not settle all the position indices associated with the pawn structure. However the rest indices to all appearance cannot be well accounted for by the evaluation function in the form (8). It is necessary to note also that the indices 6,7,8 are not accounted for in the programme.

II. Indices associated with chess-men mobility

1. Chess-men possibilities. Each square protected by a given chess-man is called its possibility. Each possibility presents in the evaluation function with the evaluation depending upon the chess-man; in most of the games played by the programme this evaluation is equal to the given in Table 2.

2. Pinned chess-men. It is possible to define exactly the notion of a pinned chess-man including also an indirect pin According to this definition a chess-man is called pinned, if its every move leads to a position which the enemy can use to change the material position evaluation in his favour in the game where only forced moves are permitted (we will use below the terms forced game). However, such a definition of this notion is not possible to use in the programme for position evaluation function in a final position because to find a pin an additional analysis of variants is needed which will increase the computing time to the greater extent than increasing of the computation depth by one move. The narrow notion of the pin has a geometric character. If on a bishop, queen or rook line undefended opponent's chess-man or a chess-man with a greater material value is located (the king always has the greatest material value) then the enemy's chess-man located between these two considered chess-men (if this chess-man is single) is called pinned. The narrow notion of the pin was not used in working programmes for the position evaluation function because to find chess-men pinned in the narrow sense it is necessary to prolong lines of action of long-range chess-men which requires the same computing time as to determine their possibilities.

3. Masked attack. If unprotected or more valid opponent s chess-man is located on the line of a bishop, rook or the queen then a masked attack presents.

4. "Good" or "bad" bishop. A bishop is called "good" if central pawns are located on squares of the other colour and "bad" if central pawns are located on squares of the same colour. However, in many positions this index does not matter; besides this index is usually static.

Thus in working programs for the position evaluation function only possibilities are used among indices associated with the mobility of chess-men.

Table 2

Notations	Indication	Estimate
	Pawn in the centre	+10
	Doubled and isolated pawns	-12
	Phalanx	+ 4
	King possibilities	0
	Queen "	+ 1
	Rook "	+ 2
	Bishop "	+ 5
	Knight "	+ 5
	Pawn "	0

III. Indices associated with cooperation of chess-men

1. The castling and its loss. If a castling is performed in a position then the term $3_{oo}=11$ is present in the evaluation function; if a castling is lost then this term $3_{oo} = -11$.

To formulate other indices it is necessary to introduce the notion of the king's area. The collection of squares consisting of the king's square and all adjacent squares is called the king's area.

2. Pressing of the opponent's chess-men on the king. Lines opened for the opponent with non-vacuous intersection with the king's area are called the lines opened upon the king. The lines must be accounted for in the evaluation function if the enemy has

major chess-men; besides the evaluation of a completely opened line must be higher than for a semiopened one.

Working programmes use only castling and its loss among the indices associated with king's safety.

IV. Indices of this and the next type are not used in working programmes. The indices given below can be realized with a relatively not great computing time. It is necessary to note also that they do not settle all the indices of this type.

1. Major chess-man on an open line. Major chess-men located on open lines and lines of pressing upon isolated and detained pawns and upon the king.

2. Major chess-man on the 7-th and 8-th ranks. A chess-man must have an evaluation if this chess-man can can hold out on these ranks, i.e. the evaluation is equal to 0 if the chess-man appeared on one of these ranks by the last move. Besides one should account for threatens to the enemy from the 7th or 8th ranks but it is rather difficult to carry this out.

3. A chess-man located in the weak opponent's point. Here the evaluation also takes place if the chess-man can hold out the point. Besides, the evaluation must depend upon the chess-man itself and the squares of its pressing.

4. The battery* Several major chess-men located on the same file, on the 7-th or 8-th rank and also a bishop and the queen located on the same diagonal line are called the battery. It is necessary to account for squares upon which a battery presses.

V. Other indices.

1. The attack. This index has a great importance which is accounted for in a better way not in the evaluation function but in other parts of the programme. Its exact definition requires an analysis of variant just as in the case of the indirect pin. However it is possible to distinguish a direct attack on a senior or an unprotected chess-man.

2. The double attack. A double attack presents in a position with two or more attacks on a senior or unprotected chess-men. To all appearance, the presence of a double threaten must by no means lead to prolong a variant.

3. Delay of the bishop's development. This is a specific index associated with the location of chess-men. For example, it is a location of a chess-man on d6 if the bishop on C8 and pawns on b7 and d7 are located.

----

*Translator's note.

By the term "battery" a situation is usually called in the chess theory when the king is exposed to check after removal of the opponent's piece and opening a line along which his other piece is acting (discovered check). This term has nothing to do with the term "battery" used in this work.

The list of indices given here may be easily enlarged. By the authors' opinion the main situation which is not described by the given indices is the necessity of preliminary evolutions to achieve a given objection. For example, to occupy a weak point by a knight it is necessary to perform a preliminary evolution. A preliminary evolution may be necessary to occupy an open line or to undermine a pawn chain. Such actions associate with a specific plan accepted in a given game. However there are many obscurities in principles of designing a chess programme choosing a [illegible] in the initial position and playing in accordance with this plan.

§ 6. Algorithms of a move choice.

The simplest of the algorithms realized in working programmes is called by the authors the absolute scheme. Rules of possible moves determination depend in this algorithm upon a chess position, upper and lower bounds of evaluation and depth computation constant ϑ_n.

Let an initial position A_0 is given; if the rank of the given position A_0 is less than ϑ_n all moves permissible by chess rules are possible. If $\ell \geqslant \vartheta_n$ then game rules in the considered position coincide with that ones of the forced game with the same upper and lower bounds and the given evaluation function. Including forced variants in the analysis without limiting their depth makes it possible to avoid blunders for small values of depth computation constant ϑ_n. However, the play of the programme is not free from some relatively strong blunders for $\vartheta_n = 4$.

The problem of the strength of the chess programme in playing is closely connected with the problem of its computing time. In working programmes their parts not associating with the complete analysis practically do not require time. Hence the computing time of the programme is proportional to the number of appearing positions. The increasing of the depth computation constant ϑ_n by 1 leads to 6-time increase in the average time of a move choice, because in a middlegame chess position by chess rules near 40 moves are possible upon the average. Thus the working programme spends to choose a move near 1 min. for $\vartheta_n = 2$, near 40 min. for $\vartheta_n = 4$ and some 4 hours for $\vartheta_n = 5$ (the time of the move choice strongly depends upon the character of a position), so that for $\vartheta_n = 5$ the game by an absolute scheme is practically impossible.

To construct algorithms spending less time to choose a move the authors suggested to define wider rules to choose possible moves than in the forced game. We will formulate below the rules using which it is possible to construct an active game algorithm or non-absolute scheme.

First of all we will define the notions of an active move and not losing move.

Let G is a game tree and A_q is its junction of the rank ℓ. We suppose below that A_q is a position with a move of White; if A_q is a position with a move of Black then the corresponding definition is based upon the note on the p. . Let m_ℓ is the lower bound of the rank ℓ on the q-th step of the complete analysis when the position A_q is considered.

All the moves in the position A_q permissible by chess rules are considered with positions A_{n_i} appearing after these moves. The forced game F is considered in each position A_{n_i} with the given evaluation function $f(A)$ and upper and lower bounds $m_{\ell+1}$ and $M_{\ell+1}$ in the forms

$$m_{\ell+1} = [m_\ell] + \delta, \qquad (10.1)$$

$$M_{\ell+1} = [M_\ell] + 1 - \delta \qquad (10.2)$$

Here d is the unit of the position evaluation, i.e. the minimal difference between different evaluations.

If the position evaluation for A_{n_i} in this forced game appears to be less than $m_{\ell+1}$ then the move leading from the position A_q to the position A_{n_i} is called not losing.

To define the notion of the active move positions A'_n are considered together with positions A'_n ; in A'_{n_i} chess-men situated identically to A'_{n_i} but the move belongs to the other side (possibility of castling is the same as in the position A_q and enpassants are not permissible). In the each of positions A'_{π_i} the forced game F with the evaluation function $f(A)$ is considered and with upper and lower bounds $m_{\ell+2}$ and $M_{\ell+2}$ defined in the form:

$$m_{\ell+2} = [m_\ell] + 1 - \delta, \qquad (11.1)$$

$$M_{\ell+2} = [m_\ell] + 1 \qquad (11.2)$$

If the position evaluation in A'_{n_i} is more or equal to $M_{\ell+2}$ in this forced game then the corresponding move is called active. Thus a move is active if after it a threaten of a material winning appears.

It is possible to define the evaluation of not losing move in the first of the forced games described above and not active move in the second forced game. This evaluation is equal to the so-called pseudoevaluation of the corresponding positions $PEVA_{n_i}$ or $PEVA'_{n_i}$.

The pseudoevaluation is determined for the analysed tree positions together with the upper and lower bounds. This pseudoevaluation coincides with the game result for final positions.

Pseudoevaluations for other positions are defined as follows. Let A is a position with a move of White, then

$$\Pi EVA_q = \max_{A_{n_i} > A_q} \Pi EVA_{n_i} \tag{12.1}$$

In contradistinction to the corresponding form defining the position evaluation the maximum is taken in the form (12.1) with respect to only the positions presenting in the initial part G' of the game tree G considered in the evaluation of the initial position and not with respect to all positions A_{n_i} directly following from the position A_q.

Similarly, if the move belongs to Black in the position A_s then

$$\Pi EV_b A_s = \max_{A_{s_i} > A_s} \Pi EV_b A_{s_i} \tag{12.2}$$

To all appearance, one may consider the rules given to determine the pseudoevaluation as based upon an optimistic view on the validity of the discussed algorithm.

It is easy to prove the pseudoevaluation to coincide with the evaluation if in each of the considered positions the best move is analysed as the first one. Indeed, in the forced game the [illegible] pseudoevaluation may indicate a larger material loss than it in fact presents because by the rules of the forced game the incomplete winning back of the offered material is not permissible. Then it makes sense to consider the pseudoevaluation only for not losing and not winning moves.

Not active and not losing moves are put in order according to their evaluation. The evaluation for such moves can be determined both from the forced game to define the moves safety and from the forced game to define this move's activity. These evaluations can occur to be different. One of these evaluations is accepted in the each of the working programmes. Not losing and not active move with the largest evaluation will be called below the best passive move

Three variants of the active game algorithms were constructed. In all of them the depth computation constant ϑ_n is given. If the rank ℓ of the considered position A_q is more or equal to ϑ_n then - as in the absolute game - the same moves are possible in the position A_q as in the forced game F. If $\ell < \vartheta_n$ moves are checked by two described above forced games and active and not losing moves are determined. All moves which are active and not losing at the same time are included in the complete analysis scheme and besides one more (in some cases more than one) passive not losing move with the largest evaluation is considered in this scheme. For $\ell = \vartheta_n - 1$ all the moves permissible by the chess rules are allowed in the position Aq because in this case the foregoing checking moves for their activity and safety would lead to losses in the computing time.

Before the active game the forced game is given from the initial position of the move choice A_o and the upper and lower bound have the extremal values

$$m = 0 \tag{13.1}$$

$$M = 2C_M + 1 \tag{13.2}$$

The evaluation EVA_o of the position A_o in this game determines the lower and upper bounds for the active game,

$$m = [EVA_o], \tag{14.1}$$

$$M = [EVA_o] + 1 - d \tag{14.2}$$

Here as above d is the position evaluation unit. Thus the position evaluation EVA_o lies between the upper and lower bounds if a material winning or losing do not present in the position.

The determination of possible moves in positions with the zero or unity rank somewhat differs from the described above. If in these positions the lower (for the move of White) or the upper (for the move of Black) bounds still satisfy the corresponding form (13.1) or (13.2) after the analysis of all the active and the best passive moves then the analysis of moves continues in the order determined by evaluations of moves until the position evaluation will lie between the bounds given by the forms (13.1) and (13.2). If such a move does not exist then the second round of the active game with changed upper and lower bounds is given. The new bounds are determined by the pseudoevaluation of the initial position A_o defined by (11.1) (if a move belongs to White) or by (11.2) (if the move belongs to Black). Lower and upper bounds for the second active game round are equal to

$$m = [PEVA_o] \tag{15.1}$$

$$M = [PEVA_o] + 1 - \delta \tag{15.2}$$

If in result of the second round the position evaluation for A_o will not get between the bounds calculated by the new pseudoevaluation of the initial position A_o then the third round is given and, if it is necessary, the fourth round. If this is not yet sufficient then the programme stops not making a move.

We will stay on the differences in three variants of the active game algorithms. In the first variant not losing moves are found first of all and the evaluation of these moves is determined. After this not losing moves are examined for their activity. All the active moves and the best passive one (except the described above case for positions with the zero and unity ranks) are permissible by the game rules. In this variant two forced games are given for the each move because in a position the majority of moves are not losing. This is why the second variant of the game works more quickly; in this variant moves are first of all examined for their activity and in this examination the move evaluation is determined. The checking for safety is carried out only for active moves and for the best passive one and if the best passive move is losing then following movds are examined until not losing move will be found or all moves permissible by the chess rules will be exhausted. If the last opponent's move was active then moves from the given position are examined only for safety and three not losing moves with the largest evaluation are considered to be permissible. This variant occurred to be not sufficiently active, so the third variant was designed. If the foregoing opponent's move was not active then possible moves are determined in this variant just as in the second one. If the opponant's move was active then first of all active and not losing answers are searched. If such moves exist then the best passive move is considered also to be possible; if there do not exist active answers then three best passive moves are possible.

§ 7. The information obtained and used by the chess programme

The initial information for the main cycle of the programme - the cycle of the complete analysis - is the information about the position A_q from which a move is made and the information about the work performed before. As moves possible in a given position and indices presenting in position evaluation are determined by operations on the chess-board realizable by a comparatively small number of computing operations cells of the memory correspond to chess-men and bits of these cells - to chess-board squares. If the memory cell had 64 bits it would be possible to use one cell for one chess-man. As the number of bits is less two memory cells correspond to each of the chess-men. 32 bits in a cell correspond to chess-board squares (see Fig.3)

e2

a1	b1	c1	d1	e1	f1	g1	h1	a2	b2	c2	d2	f2	g2	h2	a3	b3	c3	d3	e3	f3	g3	h3	a4	b4	c4	d4	e4	f4	g4	h4	
a5	b5	c5	d5	e5	f5	g5	h5	a6	b6	c6	d6	e6	f6	g6	h6	a7	b7	c7	d7	e7	f7	g7	h7	a8	b8	c8	d8	e8	f8	g8	h8

fig.3.

In these bits chess-man possibilities are indicated. Bits corresponding to squares where the given chess-man can move are indicated by 1, other bits are indicated by 0. Besides this, in remaining bits (which are not occupied by the above information) the kind of a chess-man and the number of the squares where it stands is indicated. There are no specific cells for pawns. To the collection of white pawns 5 couples of cells correspond. In the first one by 1 bits are indicated corresponding to the squares where pawns stand; in the second one squares are similarly indicated where pawns can move; in the third one
[illegible] indicated where [illegible] can move; [illegible]

squares under the attack of the pawn to the right side and in the 4-th - to the left side of it are indicated; in the 5-th couple squares are indicated to which an initial move from the 2-nd to the 4-th rank is possible (for the lower half of the chess-board) and possibilities of en passant captures (for the upper half of the chess-board).

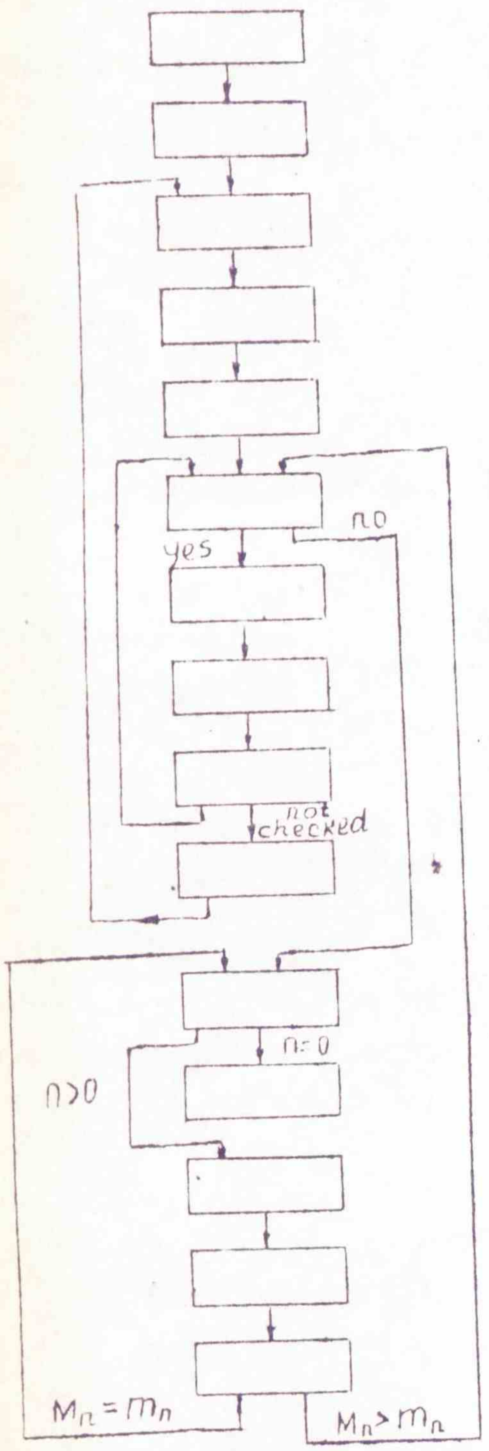
There is a similar information about black pawns. Besides there are couples of cells where squares are marked, occupied by white chess-men, black chess-men and all the chess-men. There are also cells with the information about the possibility or the loss of the castling (for White and for Black).

The programme of the complete analysis (the so-called general scheme) is composed so that it can be used for any game of two opponents with the complete information and moves in turn without changes. In essence these requirements are not principal. The main inner cycle in the programme of determination of the longest path with the units of the given length (programme PERT) is designed on the same principles. This programme differs from a usual cyclic programme by the property that, after a performed cycle the main cycle parameter - the analysis depth n (i.e. the number of semimoves from an initial position leading to the considered position) . can increase or decrease depending upon other parameters determined inside that cycle. The block-scheme of the complete analysis programme is shown on Fig.4. It is necessary to note that in fact this programme is realized somewhat differently from this block-scheme but these differences are of no principal importance.

It is possible to note "chess blocks" among blocks of the programme. These are the blocks of computing the evaluation function, of the next move, of defining the chess-men possibilities and partly of the determination of permissible moves. Other blocks are associated with the realization of the complete analysis scheme and do not depend upon the type of the game.

We will describe below several blocks of this programme.

Comparatively quick algorithms were found for the block "chess-men possibilities". They are easy to be shown for the case of the arithmetic unit operating with a 64-bit mantissa so that all the squares of the chess-board correspond to the bits of a single cell.



fiq 4

Exit rollback of the programme and n = o

Determination of chess-men possibilities in a given position.

Bounds transition from the rank n-1 to the rank n.

Computation of the evaluation function f(A)

Determination of pсssible moves and information retrieval about unmade moves

Change of the information about the next move and checking its permissibility

Next move

Chess-men possibilities

Checking: are not we checked?

Increasing n by 1

Checking: n > o

Exit

Reducing n by 1

Bounds transition from the rank n + 1 to the rank n

Checking $M_n > m_n$

Fig. 4

Then, for example, the possibilities of white pawns are defined by the following equality:

$$\Pi_m = 2^8 \Pi / A.C. \qquad (16.1)$$

$$\Pi_{tl} = 2^9 \Pi / h \qquad (16.2)$$

$$\Pi_{tr} = 2^7 \Pi / a \qquad (16.3)$$

Here Π is the cell of white pawns, Π_m - the cell "pawns move", (by one rank), Π_{tl} - the cell "pawnstake left", Π_{tr} - "pawns take right", A.C. - "all the chess-men", h - "the rank h" (i.e. 1 in bits corresponding to squares on the rank h and 0 in the other bits) a - "the rank a", - the notation of logic subtraction:

$$a / b = a - (a \wedge b)$$

N.I. Bessonov indicated that it is possible to construct specific computer instructions to determine chess-men possibilities. In this case the time to determine a chess-man possibility equals to the machine time to perform an instruction. The examples are constructed of non-arithmetic operations for a general-purpose computer with the programme control.

The specific property of the complete analysis programme is that all information necessary and sufficient for its work corresponds to a position situated on the tree branch passing from the initial position A_o to the considered position A_n. The following information is used in the programme.

1. Bounds for position evaluation M_k and m_k ($k = 0, 1, \ldots, n$).

2. Number of the first move in the given position and the last move from the given position ($a_o, a_1 \ldots, a_n$) and ($b_o, b_1 \ldots b_n$).

3. Positions $A_o, A_1, \ldots A_n$ and made moves $A_o \longrightarrow A_1$, $A_1 \longrightarrow A_2, \ldots\ldots, A_{n-1} - A_n$.

4. The information about active and passive moves in the active game algorithm.

As it can be seen from the block-scheme, a number of blocks operates with this information.

The block of the transition of the bounds from the rank n-1 to the rank n realizes the formula

$$T_{n-2} \to T_n \qquad (17)$$

Here Tn for moves of White is equal to the lower bound of m_B and for moves of Black - to the lower bound m_{B_n} for the Black (see (5)). The corresponding upper bounds are defined by the formula

$$M_{(B)n} = 2C_n + 1 - T_{n-1} \qquad (18)$$

The programme of determination of all the possible moves extracts all the moves from the given position and finds also a_n and b_n - addresses of the first and the last moves.

In the programme of the move to do a_n plays the part of a counter and b_n - its control constant. Thus, the programme checking whether it is possible to make a move increases a_n by 1 and examines if a_n is larger than b_n.

The programme of the move to do must make a move stored in the cell a_n-1 from the position A_n, i.e. to construct a new position All the information about this position is obtained except chess-men possibilities defined by the corresponding block. In the beginning of the work this programme stores the position A_n in the cells of an operation p ition A and stores the new operation position in the **cells of the position** A_{n+1} after making a move.

The work of the block "are not we checked?" is obvious from its appelation. The same can be said about increasing n by 1.

If a move cannot be made for n=o, the analysis is accomplished (all the moves from the initial position are looked over), and, as a rule, a needed move is determined. If $n > 1$ then it is necessary to pass to the consideration of the position A_{n-1} . If A_n was the final position in the model game its evaluation equal to the value of the evaluation function f**(A)** (whites) and f_B(A) (blacks) was stored in the cell T_n . After reducing n by 1 the transition of the bounds with the rank n + 1 to the rank n is made by the formula

$$max(T_n, 2C+1-T_{n-1}) \rightarrow T_n \qquad (19)$$

Thus the block checking $M_n > m_n$ examines whether the following inequality is satisfied:

$$2C+1-T_{n-1}-T_n > 0 \qquad (20)$$

List of Games

No.1	White:	the programme;	computation on 3 semimoves
	Black:	the programme;	computation on 3 semimoves
No.2	White:	the programme;	computation on 4 semimoves
	Black:	the programme;	computation on 3 semimoves
No.3	White:	the programme;	computation on 3 semimoves
	Black:	the programme;	amputation on 4 semimoves
No.4	White:	the programme;	computation on 4 semimoves
	Black:	the programme;	computation on 4 semimoves
No.5	White:	the programme;	computation on 5 semimoves
	Black:	a man	
No.6	White:	the programme;	computation on 3 semimoves
	Black:	a man	

Партия № I game 1

№ хода	Время белых (мин.)	Ход белых	Ход черных	Время черных (мин.)	№ хода	Время белых (мин.)	Ход белых	Ход черных	Время черных (мин.)
N of move	White's time	White's move	Black's move	Black's time	N of move	White's time	White's move	Black's move	Black's time
I	0,I	e2-e4	d7-d5	1,0	26	0,7	Fc5-d4	f7-f6	0,6
2	I,8	Cb1-c3	d5:e4	3,4	27	0,4	Fd4-e3	b5-b4	0,9
3	I,I	Cc3:e4	Cg8-f6	6,2	28	2,4	Td1-c1	b4:c3	I,7
4	3,4	Ce4:f6+	e7-f6	1,6	29	0,7	b2:c3	Ta8-b8	I,3
5	3,8	d2-d4	Cb1-c6	8,0	30	0,6	Tc1-d1	g7-g5	0,9
6	II,0	Cd1-f3	Fc8-d4	14,6	3I	3,8	Td1-d4	T18-e5	4,3
7	38,3	Fc1-e3	Ff8-b4+	17,6	32	4,0	Td4:c4	Te5:d5	4,5
8	8,9	c2-c3	Fd4f3	7,0	33	7,7	Rf3-g4	Tb8-b2	5,5
9	6,9	Dd1-f3	Fb4-d6	8,5	34	I0,5	Rb4-f3	Tb2-a2	5,2
I0	9,8	d4-d5	Cc6-e5	4,9	35	23,I	Tc4-c5	Td5:c5	5,7
II	4,8	Df3-f5	Dd8-d7	8,7	36	2,0	F13:c5	a5-a4	I,8
I2	I,2	Df5:d7	Ce5:d7	0,5	37	I,2	Fc5-d4	f6-f5	I,I
I3	0,7	Ff1-d3	Cd7-b6	5,6	38	0,7	Fd4-e3	Ta2-e2	0,6
I4	8,7	Fd3-b5	Cb6-d7	1,4	39	0,6	Fe3:g5	Tc2:c3	0,3
I5	0,8	0-0-0	D7-a6	1,9	40	0,8	Fg5-e3	a4-a3	0,2
I6	I,7	Fb7:d7	Re8:d7	0,2	4I	0,3	Rf3-e2	a3-a2	0,8
I7	0,3	f2-f3	Th8-e8	0,7	42	I,5	Re2-f3	Tc3-a3	2,8
I8	I,6	Rc1-d2	c7-c5	1,6	43	I,5	Rf3-e2	Da2-a1	6,I
I9	I,0	Rd2-e2	c5-c4	0,9	44	70,0	Re2-f3	Da1-c3	53,0
20	3,5	Td1-d4	b7-b5	1,8		White resigned			
2I	I,8	Td4-h4	Fd6-c5	2,3		белые сдались			
22	I,4	Th4-e4	f6-f5	1,7					
23	I,4	Fe3:c5	f5:e4	o,4					
24	0,9	Th1-d1	e4:f3+	0,8					
25	I,4	Re2:f3	a6-a5	1,0					

2

Партия № 2

Game 2

№ Хода	Время белых (мин.)	Ход белых	Ход черных	Время черных (мин.)	№ хода	Время белых (мин.)	Ход белых	Ход черных	Время черных (мин.)
No. of move	White's time	White's move	Black's move	Black's time	No. of move	White's time	White's move	Black's move	Black's time
I	0,3	e2-e4	Cg8-f6	0,5	26	2,4	d4:c5	bG:C5	0,9
2	4,9	e4-e5	Cf6-d5	I,0	27	II,7	Cc3-e4	Dd7-C6	5,0
3	6,5	c2-c4	Cd5-b6	I,0	28	6,2	Ce4-g3	Ta8-a4	I,6
4	3,I	d2-d4	e7-e6	2,3	29	4,I	h2-h3	Tf8-a8	2,0
5	3,6	c4-c5	Cb6-d5	0,6	30	2,4	Dd2-C3	Ta4-b4	2,6
6	3,9	Ff1-d3	b7-b6	3,7	3I	4,7	Te1-c1	Tb4-c4	3,0
7	52,9	c5:b6	Ff8-b4+	I,2	32	2,2	DC3-d2	Tc4:c1+	I,4
8	I2,0	Fc1-d2	a7:b6	2,3	33	I,7	Dd2:C1	Fg6-d3	I,4
9	I7,I	Fd2:b4	Cd5:b4	I,3	34	9,I	Dc1-d2	Dc6-b5	I,I
I0	I2,I	Cb1-c3	Cb4:d3	7,0	35	2,4	Dd2-c1	Db5-b4	3,8
II	2,3	Dd1-d3	Fc8-a6	I,9	36	9,9	Ra1-a2	Fd3-C4+	I0,9
I2	5,6	Dd3-e4	Cb8-c6	I,3	37	6,9	Ra2-b1	Ta8:a3	5,I
I3	7,6	Cg1-f3	d7-d5	2,7	38	7I,I	Cf3-e1	Db4-b3	9,7
I4	9,0	De4-g4	o-o	3,8	39	8,9	Dc1:c4	d5:c4	3,0
I5	3I,5	O-O-O	Cc6-b4	3,0	40	I5,3	Cg3-e2	Ta3-a2	8,I
I6	4,4	Rc1-b1	Fa6-d3+	4,0					
I7	I4,2	Rb1-a1	Fd3-f5	II,I					
I8	4,4	Dg4-f4	Cb4-c2+	4,2					
I9	I0,0	Rb1-c1	Cc2-e3	3,4					
20	I0,7	Rb1-a1	Ce3:d1	2,I					
2I	2,9	Th1:d1	Dd8-d7	2,I					
22	2,9	g2-g4	Ff5-g6	2,3					
23	0,6	Td1-11	Fg6-d3	I,4					
24	I4,6	Df4-d2	Fd3-g6	2,I					
25	I,0	a2-a3	c7-c5	I,4					

белые сдались

white resigned

Партия 3

Game 3

№ хода	Время белых (мин)	Ход белых	Ход черных	Время черных (мин)	№ хода	Время белых (мин)	Ход белых	Ход черных	Время черных (мин)
No. of move	White's time	White's move	Balck's move	Black's time	No. of move	White's time	White's move	Black's move	Bl. time
I	0,2	e2-e4	d7-d5	4,3	26	4,5	Tf1-e1	Td4-d3	I0,4
2	I,8	Cb1-c3	d5:e4	6,9	27	4,0	f3-f4	g5-f4	27,4
3	I,5	Cc3:e4	Dd8-d4	I3,9	28	6,0	g3:f4	Fb6-d4	32,7
4	4,7	Dd1-f3	Cg8-f6	33,4	29	7,6	Fe7-h4	Fd4:C3	8,2
5	II,7	Ce4:f6+	e7:f6	7,0	30	2,6	d2:c3	Fe6-d5+	4,3
6	9,6	Ff1-b5+	c7-c6	5,I	3I	I,0	Rh1-g1	Td3-f3	II,0
7	2,8	Cg1-e2	Dd4-g4	20,0	32	8,5	Te1-e7	Tf3:f4	9,7
8	2,5	Df3:g4	Fc8:g4	I,9	33	6,7	Fh4-g3	Tf4-g4	II,9
9	I,4	Fb5-d3	Ff8-d6	I,3	34	I,4	Te7-b7	a7-a5	8,9
I0	I,3	f2-f3	Fg4-e6	0,8	35	I,3	c3-c4	Fd5-f3	4,5
II	0,9	0-0	0-0	I,I	36	2,4	Td1-d3	Ff3-e2	7,9
I2	0,9	Ce2-c3	Fd6-c5+	I,7	37	3,6	Td3-d2	f5-f4	6,0
I3	0,2	Rg1-h1	Cb8-d7	0,3	38	2,7	Td2:e2	f4:g3	6,3
I4	0,4	b2-b3	Cdt-e5	3,0	39	3,9	h2:g3	Tg4:g3+	3,6
I5	3,7	Fc1-b2	Tf8-d8	9,3	40	3,2	Te2-g2	Tg3:g2	I,6
I6	4,3	Cc3-e4	Ce5:d3	6,I	4I	0,3	Rg1:g2	Ta8-d8	2,4
I7	5,2	c2+d3	Fc5-b6	26,4	42	I,3	Tb7-b6	Td8-d2+	4,6
I8	7,8	d3-d4	f6-f5	I,I	43	I,8	Rg1-f3	Td2-d3+	2,8
I9	I,8	Ce4-c3	Td8:d4	0,7	44	I,8	Rf3-e2	Td3-d6	I,I
20	0,6	Cc3-b1	Td4-h4	I,8	45	0,6	Tb6-b8	Rg8-g7	0,8
2I	0,7	g2-g3	Th4-b4	2,8	46	0,8	Tb8-a8	Td6-e6+	I,2
22	0,9	Fb2-a3	Tb4-g4	I,7	47	0,7	Te2-d1	Te6-e5	I,9
23	5,I	Cb1-c3	Tg4-d4	I,6	48	I,0	Ta8-c8	Te5-e6	I,0
24	0,7	Ta1-d1	g7-g5	2,5	49	0,5	Tc8-d8	Te6-e5	0,8
25	I,3	Ca3-e7	ht-h6	6,3	50	0,9	Td8-d6	Te5-e6	I,0

Партия 3 (продолжение)

Game 3 (continued)

№ хода	Время белых (мин)	Ход белых	Ход черных	Время черных (мин)
5I	0,3	Td6-e6	f7-e6	0,I
52	0,04	Rd1-e2	e6-e5	0,03
53	0,004	Re2-d1	e5-e4	0,003
54	0,04	Rd1-e2	Rg7-f6	0,8
55	0,2	Re2-e3	Rf6-e5	0,2
56	0,I	a2-a4	Re5-f5	0,2
57	0,I	Re3-d4	h6-h5	0,5
58	0,2	Rd4-c5	Rf5-e5	0,I
59	0,2	b3-b4	a5+b4	0,2
60	0,I	Rc5:c6	Re50d4	0,I
6I	0,I	Rc6-b5	e4-e3	0,5
62	0,6	Rb5:b4	e3-e2	0,7
63	0,6	Rb4-b5	h5-h4	I,8
64	0,7	Rb5-b4	Dc2-e1+	6,I
65	4,4	Rb4-b5	Dc1-b1+	38,2
66	3,0	Rb5-a5		

Партия не доиграна

The game was not finished

Партия № 4

Game No.4

№ хода	Время белых (мин.)	Ход белых	Ход чёрных	Время чёрных (мин.)	№ хода	Время белых (мин.)	Ход белых	Ход черных	Время черных (мин.)
No. of move	White's time	White's move	Black's move	Black's time	No. of move	White's time	White's move	Black's move	Black's time
I	0,3	e2-e4	d7-d5	4,3	26	I7,7	Ta4-g1	Tb8-d8	8,7
2	5,3	e4:d5	Dd8:d5	2,2	27	I0,3	Rd3-c3	Cg2-e3	8,2
3	3,4	Cb1-c3	Dd5-e6+	3,7	28	4,3	Tg1:g7	Ce3-d1+	3,4
4	I,4	Ff1-e2	De6-g6	4,6	29	4,3	Rc3-b4	Td8-d7	20,4
5	29,0	Cc3-d5	Dg6-d6	2,8	30	I,5	Tg7:h7	Rc7-b6	I,9
6	24,0	c2-c4	Cg8-f6	25,5	3I	3,4	Th7-h6	Td7-d2	3,9
7	I0,0	Cd5:f6+	e7:f6	3,2	32	5,5	Th6:f6	Td:a2	7,3
8	8,6	d2-d4	Cb8-c8	26,I	33	4,8	c4-c5+	Rb6-a6	8,I
9	7,3	d4-d5	Cc6-e5	I7,3	34	I,4	Tf6:c6+	Ra6-b7	2,2
I0	23,3	f2-f4	Ce5-g6	23,3	35	3,I	Tc6-f6	Ta2:h2	I,4
II	39,6	Dde-b3	Ff8-e7	225,0	36	I,5	Tf6:f7	Rb7-c6	2,0
I2	I9,9	Db3-b5+	c7-c6	I7,I	37	2,8	Tf7-f6+	Rc6-d5	3,4
I3	I0,7	d5:c6	b7:c6	9,8	38	5,7	Ch3-g5	Rd5-d4	8,I
I4	I6,8	Db5-a4	Re8-d7	79,I	39	I,2	Cg5-f3+	Rd4-e4	2,4
I5	33,6	Cg1-h3	Dd6-b4+	28,7	40	I,5	Cf3:h2	Cd1-e3	2,0
I6	5,2	Dq4:b4	Fe7:b4+	3,4	4I	I,8	Tf6-f7	Ce3-d5+	I,4
I7	2,6	Fc1-d2	Fb4:d2	9,3	42	2,5	Rb4-b5	Cd5:f4	8,0
I8	6,9	Re1:d2	Ta8-b8	33,2	43	I,9	Tf7:a7	Re4-d3	I,5
I9	32,7	b2-b3	Rd7-c7	25,8	44	4,I	c5-c6	Rd3-c2	I,7
20	I0,7	Th1-f1	Th8-d8+	I6,7	45	4,4	c6-c7	Rc2:b3	5,8
2I	I0,0	Fe2-d3	Fc8-f5	6I,0	46	6,I	c7-c8Q	Rb3-b2	7,0
22	7,2	Tf1-f3	Cg6-h9	I54,9	47	9,0	Dc8-c4	Cf4-g6	2,5
23	I6,7	Tf3-g3	Ff5:d3	I3,6	48	7,4	Ta7-a2+		
24	3,6	Tg3+:d3	Td8:d3+	2,2					
25	I,3	Rd2:d3	Ch4:g2	4,7					

черные сдались

Black resigned

Партия № 5

Game No.5

№ хода	Время белых (мин.)	Ход белых	Ход чёрных	Время черных (мин.)	№ хода	Время белых (мин.)	Ход белых	Ход черных (мин.)	Время черных (мин.)
No. of move	White's time	White's move	Black's move	Black's time	No. of move	White's time	White's move	Black's move	Bl. time
I	I0,0	Cb1-c3	e7-e6		26	2,0	Te1-f1	Tc1:f1+	
2	30,0	d2-d4	d7-d5		27	2,0	Rg1:f1	Tc8-c1+	
3	30,0	e2-e4	d5:e4		28	2,0	Rf1-e2	Tc1-b1	
4	30,0	Cc3:e4	Cg8-86		29		Cd7:b6	Tb1:b3	
5	I2,0	Ce4:f6	Dd8:f6		30		Cb6-c8	Tb3-a3	
6	30,0	Cg1-f3	h7-h6		3I		Cc8-b6	Rg8-f8	
7	33,0	Dd1-d3	a7-a6		32		Cb6-d7+	Rf8-e7	
8	40,0	Fc1-d2	Fc8-d7		33		Cd7-c5	a6-a5	
9	65,0	Dd3-b3	b7-b6		34		Re2-d1	Re7-d6	
I0	55,0	Ff1-d3	Ff8-d6		35		Cc5-b7+	Rd6-c7	
II	I30,0	Fd2-b4	Df6-e7		36		Cb7-c5	Rc7-c6	
I2	I3,0	Fb4:d6	De7:d6		37		Rd1-d2	e6-e5	
I3	I00,0	0-0	0-0		38		Rd2-e3	e5-d4+	
I4	75,0	Db3-a3	Cb8-c6		39		Re3:d4	Ta3-a2	
I5	I5,0	Da3:d6	C7:d6		40		f2-f4	Ta2:g2	
I6	75,0	Tf1-e1	Cc6-b4		4I		h2-h3	Tg2-b2	
I7	I3,0	Fd3-e4	d6-d5		42		Rd4-c3	Tb2-b4	
I8	I0,0	Fe4-d3	Tf8-c8		43		d3-d4	Tb4-c4+	
I9		Cf3-e5	Fd7-b5		44		Rc3-d3	Tc4:c5	
20	I3,0	a2-a4	Fb5:d3		45		d4:c5	Rc6:c5	
2I	I2,0	c2:d3	Cb4-c2		46		Rd3-c3	d5-dr+	
22	I0,0	Ta1-b1	Cc2:e1						
23	4,0	Tb1:e1	Tc8-c2						
24	I0,0	b2-b3	Ta8-c8						
25	I0,0	Ce5-d7	Tc2-c1						

партия не доигрывалась

the game was not finished

Партия № 6

Game No.6

№ хода	Время белых (мин.)	Ход белых	Ход черных	Время черных (мин.)	№ хода	Время белых (мин.)	Ход белых	Ход черных	Время черных (мин.)
No. of move	Wh. time	White's move	Black's move	Black's time	No. of move	Wh. time	White's move	Black's move	Black's time
I		e2-e4	c7-c5		20		Ff4-e5	f7-f6	
2		Dd1-h5	b7-b6		21		Fe5-g3	e6-e5	
3		Ff1-c4	e7-e6		22		Rc1-b1	Tc8-c5	
4		Cb1-c3	Cg8-f6		23		f3-f4	e5-e4	
5		Dh5-f3	Fc8-b7		24		Fd3-f1	Td8-c8	
6		Cg1-e2	Ff8-c7		25		Te1-e3	f6-f5	
7		d2-d4	0-0		26		Cg3-h4	a7-a5	
8		Fc1-f4	Cb8-c6		27		Ch4-e7	Tc5-c7	
9		0-0-0	c5:d4		28		Ce7-d6	Tc7-d7	
10		Ce2:d4	Cc6:d4		29		Cd6-e5	Rg8-f7	
11		Td1:d4	Fe7-c5		30		Te3-g3	g7-g6	
12		Td4-d2	Fc5-b4		31		Ff1-e2	Kf7-e6	
13		Ta1-e1	Cf6:e4		32		Fe5-d4	Td7-d6	
Cc3:e4		d7-d5			33		Tg3-h3	h7-h5	
15		Ce4-c3	Fb4:c3		34		Th3-g3	Re6-f7	
16		b2:c3	Ta8-c8		35		Fe2:h5	g6:h5	
17		Fc4-d3	Dd8-f6		36		Tg3-g7+	Rf7-e6	
18		Ff4-d6	Df6:f3		37		Tg7:bt		
19		g2:f3	Tf8-d8						

The game was not finished

Игра не окончена