Neural Networks and the Estimation of Hands' Strength in Contract Bridge

Krzysztof Mossakowski and Jacek Mańdziuk*

Faculty of Mathematics and Information Science, Warsaw University of Technology,
Plac Politechniki 1, 00-661 Warsaw, POLAND
{mossakow, mandziuk}@mini.pw.edu.pl

Abstract. This paper is focused on a *Double Dummy Bridge Problem* (DDBP) which consists in answering the question of how many tricks are to be taken by a pair of players assuming perfect play of all four sides with all cards being revealed. Several experiments are also presented in a variant of DDBP in which the information about to whom of the two players in a given pair a particular card belongs to is hidden. In contrast to our previous works, which were devoted to no trump contracts, here we concentrate on suit contracts. Several interesting conclusions are drawn from comparison of weight patterns of networks trained on no trump contracts only vs. those trained exclusively on suit contracts. The ultimate goal of this research is to construct a computer program playing the game of contract bridge using neural networks and other CI techniques with the basic assumption of using zero-human-knowledge approach and to learn purely on examples.

1 Introduction

The game of bridge has attracted attention of many AI and CI researchers, e.g. [1–5]. In particular, some interest was also devoted to DDBP [5, 6] which is regarded as an interesting benchmark problem at initial stage of bridge playing program's development.

In this paper we continue our research efforts devoted to DDBP having in mind the ultimate goal - construction of a bridge playing system without explicit presentation of any human expert knowledge concerning the game.

It must be emphasized that the rules of the game were not presented in any form. In all experiments the training data contained only deals (i.e. information about which player each card belongs to) with target information about the number of tricks to be taken by one pair of players.

2 Previous Work - No Trump Contracts

In this section we briefly describe previous results obtained for no trump contracts, published in [7–9].

* This work was supported by the Warsaw University of Technology grant no. 504G 1120 0008 000

2.1 Various Approaches to DDBP

In [7] several neural network architectures were tested in the DDBP (from NS viewpoint) for no trump contracts. The data was taken from GIB Library [10] which includes $717,102$ deals with revealed all hands. Additionally the library provides a number of tricks taken by the pair NS in each contract under the assumption of a perfect play of all sides.

Several feed-forward perceptron neural networks with logistic activation function were created, trained using RProp algorithm and tested in JNNS (Java Neural Network Simulator) environment. In most cases logistic (unipolar sigmoid) activation function was used for all neurons except for the case of representation of data using negative numbers, where the hyperbolic tangent (bipolar sigmoid) activation function was applied.

Two main approaches to coding a deal suitable for neural network input representation were applied. In the first approach each card of each hand was represented by two real numbers: the value (2, 3,..., *king*, *ace*) and the suit - S (*Spades*), H (*Hearts*), D (*Diamonds*), C (*Clubs*). Both real numbers were calculated using a uniform linear transformation to the range $[0.1, 0.9]$ (see [7, 8] for details).

In the other deal representation - which was superior to the above described one - the positions of cards in the input layer were fixed, i.e. from the leftmost input to the rightmost one the following cards were represented: 2 of Spades, 3 of Spades, ..., *king* of Clubs, *ace* of Clubs. An input value denoted the hand to which a given card belonged: 1.0 for N, 0.8 for S, -1.0 for W, and -0.8 for E. The simplest network $52 - 1$ accomplished the result $(94.15 \mid 76.15 \mid 31.29)$[1], and the network with one hidden layer: $52 - 25 - 1$ improved this score to $(95.81 \mid 79.95 \mid 34.02)$. A slight modification of the above way of coding a deal consisted in extending the input representation to 104 neurons. The first 52 neurons represented assignment to a pair (value 1.0 for NS pair and -1.0 for WE), and the other 52 ones represented a hand in a pair (1.0 for N or W and -1.0 for S or E). The simplest network: 104-1 accomplished the result $(94.76 \mid 77.52 \mid 32.19)$, and two-hidden layer network $104 - 30 - 4 - 1$, yielded the result $(95.64 \mid 79.63 \mid 33.74)$.

The number of iterations required to learn the training set without overfitting depended mostly on the way of coding a deal. The networks with the first type of coding needed a few tens of thousands iterations, and networks with coding by cards' assignment only several hundred ones.

A few more ways of coding a deal were also tested, but regardless of the problem representation **it was concluded that with the proposed approach exceeding the level of** $(96 \mid 80 \mid 34)$ **is a challenging task.**

As a point of reference simple estimator based on Work point count (*ace* - 4 points, *king* - 3, *queen* - 2, *jack* - 1) was proposed. The number of tricks to be

[1] Each of the three values denotes the fraction *in percent* of test deals for which the network was mistaken, respectively by no more than 2 tricks (94.15%), no more than 1 trick (76.15%), and was perfectly right (31.29%). This notation is used in the whole paper.

taken by NS was estimated as $(13/40) * points\_of\_NS$. This estimator achieved the result of (86.19 | 61.32 | 22.52) which was significantly inferior to the ones achieved by neural architectures. This result suggests that networks learnt some additional information besides simple estimation of Work points.

2.2 Analysis of Trained Networks

Except for numerical results it is interesting to explore what is the problem representation in connection weights of trained networks. A closer look at this data obtained for $52-25-1$ architectures revealed some interesting observations [8, 9].

Firstly, weights of connections between input neurons representing *aces* and *kings* had always the biggest absolute values. This feature was simple to explain (for humans) - these cards were the most important in the game of bridge, especially in no trump contracts.

Secondly, in each trained network there were exactly 4 connections from input to hidden neurons which had absolute values noticeably bigger than all others (about 25.0 vs less than 7.0). Not surprisingly these favored connections started from 4 input neurons assigned to *aces*.

Thirdly, in all networks it was also possible to point out hidden neurons focused on one particular suit, one neuron per suit. Such neuron had much bigger absolute values of connections' weights from inputs representing the suit than weights' values from the rest of inputs. These connections are marked using long-chain lines in Fig. 1.

Another very interesting feature which appeared in all trained networks with 25 hidden neurons was the presence of four hidden neurons each specialized in five cards from one suit: *ten*, *jack*, *queen*, *king*, and *ace* (in Fig. 1 the respective connections are marked using the dotted line). In all these groups the most important were *queens* and *kings*, *jacks* were less important, but still much more relevant than *aces* and *tens*. The hypothesis is that these hidden neurons are responsible for very important aspect of the play of bridge - *the finesses*.

Finally, there existed hidden neurons with values of connections to the output close to zero. The authors were unable to find any pattern in their weights of connections from the inputs. The number of such neurons increased in case of more complicated networks.

3 Current Experiment - Suit Contracts

The new research results presented in this paper still concern the solution of DDBP, but, unlike in the previous research, the focus is now on suit contracts. For comparison with previous results in most cases the same network architectures and similar ways of coding a deal are used. These results are presented and discussed in section 3.2.

Another interesting issue is to check whether information about the exact hand location of a given card is really important in the training data, or maybe

Table 1. Results (in %) obtained for the test sets for $52-25-1$ networks. **NT** denotes *No Trump contracts*.

Description	Results
NT; input values: $N:1.0$, $S:0.8$, $W:-1.0$, $E:-0.8$	(95.81 | 79.95 | 34.02)
NT; input values: $N:1.0$, $S:0.8$, $W:-1.0$, $E:-1.0$	(95.97 | 80.46 | 34.35)
NT; input values: $NS:1.0$, $WE:-1.0$	(96.07 | 80.88 | 34.66)
Suit contracts; input values: $NS:0.5$, $WE:-0.5$	(98.68 | 87.88 | 40.11)
the above network tested on **NT**	(91.64 | 69.21 | 26.06)
NT and **suit contracts**; input values: $NS:0.5$, $WE:-0.5$	(97.72 | 84.90 | 37.56)
the above network tested on **suit contracts** only	(98.57 | 87.24 | 39.43)
the above network tested on **NT** only	(94.30 | 75.50 | 30.09)
Spades contracts; input values: $NS:1.0$, $WE:-1.0$	(98.77 | 88.00 | 40.13)
the above network tested on **Hearts contracts**	(59.18 | 39.09 | 14.12)
the above network tested on **Diamonds contracts**	(58.89 | 38.67 | 13.51)
the above network tested on **Clubs contracts**	(58.86 | 38.90 | 13.77)
Hearts contracts; input values: $NS:1.0$, $WE:-1.0$	(98.65 | 87.81 | 40.18)
Diamonds contracts; input values: $NS:1.0$, $WE:-1.0$	(98.66 | 87.68 | 39.96)
Clubs contracts; input values: $NS:1.0$, $WE:-1.0$	(98.73 | 87.90 | 40.02)

it would be enough to locate cards as belonging to either NS or WE pairs. This issue is discussed in section 3.1.

Finally, it seems worth investigating whether the results are repeatable, i.e. whether within the ensemble of neural nets the results for a given contract would be the same. This question is considered in section 3.3.

All results are summarized in Table. 1.

3.1 Hiding Opponents' Cards in NT Contracts

In previous work with NT contracts and all 4 hands revealed each of 52 input neurons pointed out the hand to which a given card was assigned, namely input equal to 1.0 denoted N hand, 0.8: S hand, -1.0: W, and -0.8: E. Let us recall that $52-25-1$ network trained using this way of coding achieved the result of (95.81 | 79.95 | 34.02).

Hiding opponent's cards was carried out by applying the following values for hands' description: $N:1.0$, $S:0.8$, W and $E:-1.0$. Results yielded by a network with the same architecture (52-25-1) were slightly improved: (95.97 | 80.46 | 34.35).

Surprisingly, hiding also information about cards' assignment in the pair NS, i.e. using input values: N and $S:1.0$, W and $E:-1.0$, yielded another slight improvement: (96.07 | 80.88 | 34.66).

These results prove that for data representation chosen in the experiments information about exact hand location of each card is not helpful and sometimes even misleading. The explanation of this phenomenon is one of our current research goals.

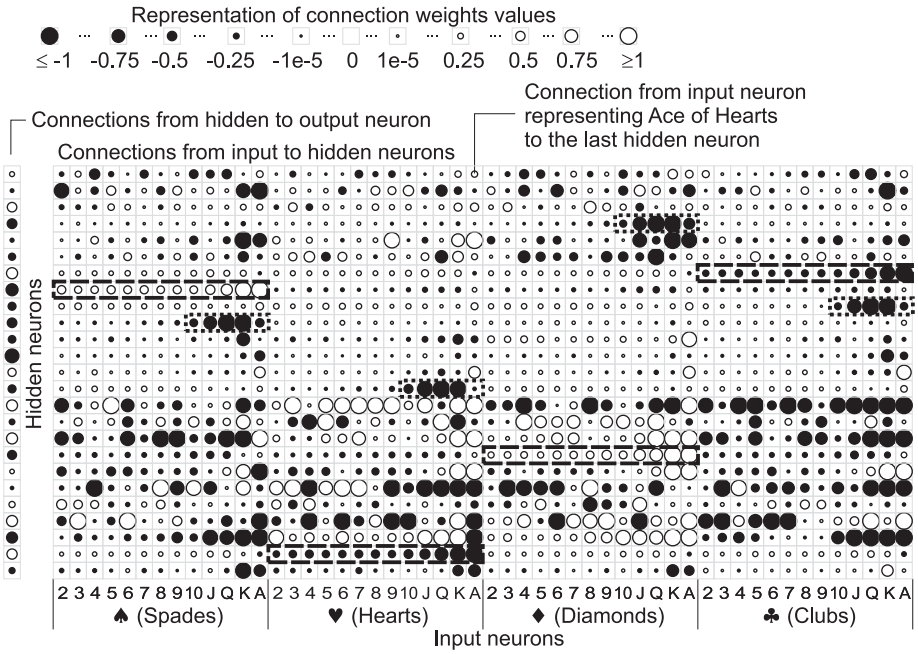

Fig. 1. NT contracts. Weights of connections of trained neural network with 25 hidden neurons ($52 - 25 - 1$). Each circle represents the weight of one connection. If the circle is placed in the leftmost column, it represents the weight of connection from hidden to output neuron, otherwise - from input to hidden neuron. The radius of the circle represents the absolute value of connection weight. Black circles denote negative and white ones positive weights.

3.2 Suit Contracts

In the experiments with suit contracts a deal was coded in the following way. Cards of the trump suit had input values equal to 1.0 for players N, S or -1.0 for players W, E. Cards of other suits: 0.5 for N, S and -0.5 for W, E. When no trump contract was presented to the network, all cards had input values equal to 0.5 or -0.5, resp. for NS and WE pairs.

Neural network with 25 hidden neurons trained on $400,000$ examples ($100,000$ deals repeated 4 times - once for each trump suit) accomplished the results (98.68 | 87.88 | 40.11). Testing this network on no trump contracts (not present in the training set) led to poorer score: (91.64 | 69.21 | 26.06).

When no trump contracts were added to the training and test sets, network with the same architecture yielded the result of (97.72 | 84.90 | 37.56). Results of testing this network separately on suit contracts were significantly better (98.57 | 87.24 | 39.43) than results of tests on no trump contracts only (94.30 | 75.50 | 30.09).

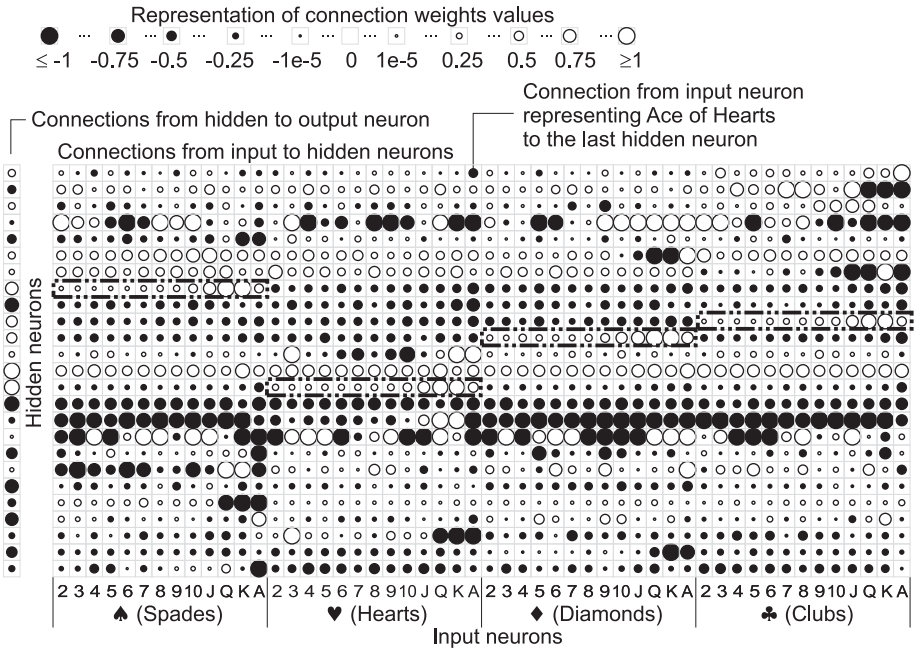

Fig. 2. Suit contracts. Weights of connections of trained network with 25 hidden neurons ($52 - 25 - 1$). See description of Fig. 1.

The network trained only on Spades contracts achieved the result of (98.77 | 88.00 | 40.13). Results of testing this network based exclusively on Hearts contracts: (59.18 | 39.09 | 14.12), Diamonds contracts: (58.89 | 38.67 | 13.51), and Clubs ones: (58.86 | 38.90 | 13.77) confirmed the effect of "specialization". Furthermore, the networks trained exclusively on other suits yielded similar results, i.e. (98.65 | 87.81 | 40.18) for training on Hearts contracts only, (98.66 | 87.68 | 39.96) for Diamonds contracts, and (98.73 | 87.90 | 40.02) for Clubs contracts. It is an interesting observation that results of specialized networks presented above are on the same level as results of networks trained on all suit contracts simultaneously. Closer investigation of this counterintuitive effect is planned in the near future.

The above results imply that **no trump contracts and suit contracts should be treated separately.** Connection weights of a network trained using suit contracts only are presented in Fig. 2. The main difference between this figure and Fig. 1 representing connection weights of a network trained exclusively on no trump contracts is visibly bigger relative importance of the lowest cards in a deal. This observation is also confirmed by Table. 2, where connection weights of a network without hidden neurons (52-1) are compared. This conclusion is in line with human bridge players' experience. Moreover, it is clear from the table that no significant differences between suits exist which again is a desirable effect.

Table 2. Values of connections between neurons of two networks without hidden neurons $52-1$ trained respectively on no trump and suit contracts. Values were linearly scaled to interval (0,4).

	NT Contracts				Suit Contracts			
Card's value	S	H	D	C	S	H	D	C
2	0.342	0.327	0.329	0.342	1.660	1.670	1.668	1.667
3	0.340	0.334	0.328	0.353	1.664	1.667	1.663	1.660
4	0.347	0.314	0.351	0.345	1.669	1.655	1.669	1.685
5	0.341	0.332	0.341	0.344	1.660	1.673	1.676	1.663
6	0.356	0.349	0.339	0.329	1.684	1.685	1.680	1.688
7	0.380	0.331	0.354	0.356	1.680	1.684	1.687	1.697
8	0.358	0.361	0.375	0.400	1.709	1.719	1.718	1.723
9	0.496	0.469	0.461	0.473	1.782	1.791	1.780	1.783
10	0.660	0.663	0.671	0.684	1.921	1.916	1.918	1.938
J	1.047	1.032	1.056	1.030	2.174	2.167	2.177	2.172
Q	1.676	1.688	1.675	1.656	2.569	2.569	2.572	2.565
K	2.643	2.643	2.677	2.655	3.207	3.210	3.220	3.216
A	3.975	3.971	3.966	3.989	3.982	3.984	3.973	3.995

Coming back to comparisons between Fig. 1 and Fig. 2 there can also be identified some common patterns in both figures, e.g. cards from *aces* to *tens* are the most important - connections from input neurons representing them to hidden neurons have the biggest absolute values. It is also possible to point out hidden neurons specialized in one suit, additionally with connections from *kings* and *queens* being the most important.

In summary, it should be emphasized that the above described weight patterns were observed also when other training sets had been used.

3.3 Reliability of Results

In order to check the reliability of results, 4 networks with one hidden layer of 25 neurons each, differing only by initial, randomly chosen connection weights, were trained based on the same set of deals. This experiment was aimed at checking the number of deals from the training set for which all 4 networks would learn the same number of tricks to be taken by NS.

For no trump contracts all 4 networks estimated the same number of tricks in 61.23% of contracts. In 37.93% of contracts estimated numbers of tricks differed by 1 trick, in 0.81% by 2 tricks and in 0.03% by 3 tricks. The same experiment for suit contracts output the following results: for 63.40% of contracts all networks were unanimous, for 36.56% of contracts there was a 1 trick difference, and for 0.04% of them the difference was equal to 2 tricks.

In 98.13% of testing deals for no trump contracts, and in 99.53% for suit contracts, real output values of all 4 trained networks differed by no more than 0.06. In these experiments target number of tricks was calculated using a uniform

Table 3. Results for subsets of a testing set achieved by a network $52-25-1$ trained on Spades contracts.

Target number of tricks	Number of deals	Results
0	1,138	(93.32 | 66.61 | 12.30)
1	2,725	(97.39 | 81.21 | 34.53)
2	5,156	(98.10 | 86.66 | 40.73)
3	8,043	(98.93 | 88.96 | 41.41)
4	10,447	(98.94 | 89.04 | 40.36)
5	12,201	(98.85 | 88.67 | 40.80)
6	12,927	(99.03 | 88.75 | 41.32)
7	12,709	(99.10 | 88.99 | 40.50)
8	11,467	(99.28 | 89.29 | 40.46)
9	9,618	(99.14 | 89.19 | 42.14)
10	6,866	(98.89 | 88.45 | 40.58)
11	4,225	(97.94 | 85.87 | 42.32)
12	1,935	(97.57 | 81.71 | 31.94)
13	543	(94.66 | 73.85 | 9.39)
All	100,000	(98.77 | 88.00 | 40.13)

linear transformation to the range $[0.1, 0.9]$, so value 0.06 was the range of real output values of networks for each number of tricks.

The results prove that the confidence in the learning process is high, and the training results are repeatable.

3.4 Results by the Target Number of Tricks

Results of a network $52-25-1$ trained on Spades contracts were investigated in detail in order to test whether the efficacy of the system varies for different numbers of target tricks. The test set containing 100,000 deals was divided into subsets which were then tested individually. Results are presented in Table. 3.

Only the results for 0, 1, 12, and 13 tricks are significantly worse than the result attained for the whole test set (the last row of the Table). Results for the other subsets are on the similar level, in spite of considerable differences in the number of deals in these subsets (e.g. 4,225 deals with 11 tricks vs. 12,927 ones with 6 tricks).

3.5 Sample Deals

Two sample deals are presented in Fig. 3. The first deal (Fig. 3(a)) was included in the reliability test described in previous section. Each of 4 networks estimated different number of tricks to be taken by the pair NS, i.e. $5, 6, 7$, and 8. A closer analysis of this deal revealed that the number of tricks for NS in no trump contract depends on information who makes defender's lead. Defender's lead from N or S hand enables to take 8 or 7 tricks, resp. On the other hand, defender's lead from W or E limits the number of tricks for NS to 6 or 5, resp. Information

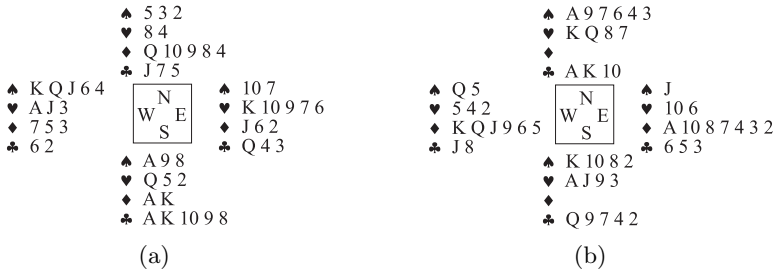

Fig. 3. Sample deals.

about defender's lead wasn't included in the input data (desired output values were specified for defender's lead from W hand). Hence the behavior of neural networks can be "justified".

The second deal (Fig. 3(b)) is a successful example of learning suit contracts - the network predicted grand slam of spades for NS with only 26 points (Work point count). Please note that also grand slam of Hearts for NS is possible, but in this case the network yielded only 12 tricks. **It is an interesting observation that the network estimated higher longer suit (Spades) than stronger one (Hearts)**.

4 Conclusions and Future Research

Based on the results, it can be concluded that in the DDBP it is advisable to train neural networks separately for no trump contracts and suit ones.

Reliability tests show that the confidence in training process is high and the results are repeatable.

Interesting patterns found in figures presenting networks connection weights' values (Fig. 1 and Fig. 2), and their reasonable explanation based on human experience in the game of bridge, look very promising and suggest taking into consideration the possibility of automatic, unguided discovering of knowledge hidden in connection weights.

Currently we are focused on defining an automatic input data preprocessing system capable to find functional similarities in deals and based on that allowing to either preprocess (transform) training data or divide it into subsets suitable for specialized networks. Similarity of results achieved for subsets of deals with the same number of target tricks (Table. 3) implies that dividing training set according to the number of target tricks and then applying specialized networks to these subsets seems to be a promising direction.

The next "big" step is to advance this research into play phase.

References

1. Kohle, M., Schonbauer, F.: Experience gained with a neural network that learns to play bridge. In: 5th Austrian Artificial Intelligence Meeting. (1995) 224–229
2. Sarkar, M., Yegnanarayana, B., Khemani, D.: Application of neural network in contract bridge bidding. In: Proc. of National Conf. on Neural Networks and Fuzzy Systems, Anna University, Madras (1995) 144–151
3. Smith, S., Nau, D., Throop, T.: Computer bridge: A big win for AI planning. AI Magazine **19** (1998) 93–106
4. Frank, I., Basin, D.A.: Search in games with incomplete information: A case study using bridge card play. Artificial Intelligence **100**(1-2) (1998) 87–123
5. Ginsberg, M.: Gib: Imperfect information in a computationally challenging game. Journal of Artificial Intelligence Research **14** (2001) 303–358
6. Wheen, R.: Brute force programming for solving double dummy bridge problems. In Levy, D., Beal, D., eds.: Heuristic Programming in Artificial Intelligence: the first computer olympiad. Ellis Horwood, Chichester (1989) 88–94
7. Mossakowski, K., Mańdziuk, J.: Artificial neural networks for solving double dummy bridge problems. In Rutkowski, L., Siekmann, J.H., Tadeusiewicz, R., Zadeh, L.A., eds.: Artificial Intelligence and Soft Computing - ICAISC 2004, 7th International Conference, Zakopane, Poland, June 7-11, 2004, Proceedings. Volume 3070 of Lecture Notes in Computer Science., Springer (2004) 915–921
8. Mańdziuk, J., Mossakowski, K.: Looking inside neural networks trained to solve double-dummy bridge problems. In: 5th Game-On International Conference on Computer Games: Artificial Intelligence, Design and Education (CGAIDE 2004), Reading, UK (2004) 182–186
9. Mossakowski, K., Mańdziuk, J.: Weight patterns in the networks trained to solve double dummy bridge problem. In Hryniewicz, O., et al., eds.: Issues in Intelligent Systems. Paradigms. Exit (2005) 155–165
10. Ginsberg, M. (http://www.cirl.uoregon.edu/ginsberg/gibresearch.html)