Difficulty Rating of Sokoban Puzzle[1]

Petr Jarušek and **Radek Pelánek** [2]

Abstract. Sokoban puzzle is very challenging problem for both humans and computers. It also illustrates differences between human and artificial intelligence – different problems are difficult for humans and for computers. Whereas algorithmic techniques for Sokoban solving have been intensively studied by previous research, factors determining difficulty for humans have not been sufficiently explained so far. We describe two methods for difficulty rating of Sokoban puzzle – a problem decomposition metric and a computational model which simulates human traversal of a state space. We evaluate these metrics on large scale data on human solving (2000 problems solved, 785 hour of problem solving activity).

1 INTRODUCTION

Human problem solving is influenced by many parameters, e.g., formulation of rules, number of items that need to be held in working memory, or familiarity with problem domain. But even when all of these intuitive parameters are fixed, there can still be large differences in problem difficulty. We study human behaviour during Sokoban problem solving and we show that even for a set of very similar problems the differences in difficulty are significant and not explained by any simple problem parameter. Why do these differences occur?

Understanding this phenomenon is important not just for providing insight into human cognition, but it also has several applications. Humans and computers solve problems in different ways and have complementary strengths [22]; to built tools for human-computer collaboration we need to understand what makes problems difficult for humans. Another application is for teaching and training (e.g., with intelligent tutoring systems [2]). Humans enjoy problem solving, but only when faced with problems of adequate difficulty, i.e., neither too boring nor too difficult. To recommend appropriate instance of problem we need to be able to evaluate its difficulty.

1.1 Sokoban

In this paper we report on experiments with the Sokoban puzzle. Sokoban is a logic puzzle which has simple rules and yet incorporates intricate dynamics and great complexity for both humans and computers. Example of the puzzle is in Figure 1. There is a simple maze with several boxes and one man. The goal is to get the boxes into the target squares by pushing one box at a time.

We have chosen Sokoban puzzle for several reasons. The first reason is interestingness. Sokoban has simple rules and intuitive visual setting; hence it is very simple to understand. Most people find it interesting and challenging and thus are willing to solve it voluntarily.

[1] Supported by GA ČR grant no. P202/10/0334.

[2] Department of Information Technology, Faculty of Informatics, Masaryk University Brno, Czech Republic

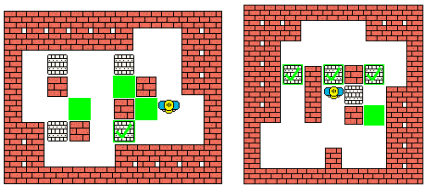


Figure 1. Example of two difficult Sokoban puzzles. The median solving time for the left problem (further denoted example 1) is 43 minutes, for the right one (denoted example 2) it is 49 minutes.

The second reason is availability of resources. There is a very large number of levels of the puzzle freely available on the Internet. These available levels span wide range of difficulty. Sokoban is also used by artificial intelligence community as a testbed for developing single agent search techniques, with many techniques and results are described in research literature (see e.g., [5, 6, 11, 12, 13]).

The third reason is complexity. Despite the simplicity of the rules, Sokoban can be very challenging for both humans and computers (Sokoban is PSPACE-complete problem [5]). The puzzle also illustrates differences between humans and computers. There exist small instances that can be quickly solved by computer (using a trivial brute force algorithm) but take humans hours to solve (see Figure 1). At the same time, there are also instances of the puzzle, which humans can solve but which are beyond capabilities of the state-of-the-art artificial intelligence solvers [11].

1.2 Human Problem Solving

Problem solving encompasses many different activities. The main classification is into well-structured problems and ill-structured problems [23]. Well-structured problems have clear boundaries and sharply defined situations, rules, and goals (e.g., proving mathematical statement, optimizing logistic operation). Ill-structured problems are defined more vaguely (e.g., writing a book).

A typical example of well-structured problem are logic puzzles. Puzzles contain all important information in the statement of the problem (and hence do not depend on knowledge), are amenable to automated analysis, and are also attractive for humans. The use of puzzles has a long tradition both in computer science (particularly artificial intelligence) [20] and cognitive psychology [21, 23].

Human problem solving has been studied for a long time, starting by a seminal work by Simon and Newell [21]; for a recent overview see [17]. Relevant to this work is particularly research concerned with puzzles which can be directly expressed as state space traversal, e.g., Tower of Hanoi puzzle [14], river crossing problems [9], Water jug puzzle [3], Fifteen puzzle [18], and Chinese ring puzzle [15].

This work is concerned with analysis of human problem solving

activity on the Sokoban puzzle with the special focus on the issue of problem difficulty. As opposed to previous research, which has been based only on small samples of human problem solving activity in laboratory setting, we employ large scale data collection through Internet. The data are collected via web portal which contains a simulator of the puzzle. People log into this portal and try to solve provided puzzles. All their actions (including their timing) are saved and stored into a database on the server.

This approach has certainly some disadvantages over the standard 'laboratory' approach to experiments with human problem solving, particularly we do not have a direct control over our subjects. Nevertheless, we believe that the advantages significantly outweigh these disadvantages. We have been able to collect large number of data about human problem solving activity: more then two thousand completed games. Almost three hundred people solved Sokoban on the portal and spent 785 hours with solving. This is much more than would be feasible with the classical laboratory approach. Moreover, the experimental setting is cheap and the data collection rather fast.

1.3 Problem Difficulty

We focus on the issue of problem difficulty. Previous research on the problem difficulty (e.g., [7, 9, 14, 15, 18]) was focused particularly on the following concepts:

- hill-climbing heuristic, which was studied for example for river crossing problems [9], Fifteen puzzle [18], and Water jug puzzle [3, 4],
- means-end analysis, which was proposed as a key concept in the "General Problem Solver" [16] and was studied for example for Tower of Hanoi puzzle,
- differences between comprehension of isomorphic problems, which focus on the difficulty of successor generation and were studied for example for Tower of Hanoi [14] and Chinese ring puzzle [14].

However, these concepts are not sufficient to explain differences in difficulty in Sokoban problems. In our experiment there are very similar problems with large difference in difficulty (more than 10-fold) – whereas the problems in Figure 1 took human on average nearly one hour, other problems were solved in within few minutes. Yet with respect to the above mentioned concepts the problems are nearly the same. Hill-climbing is not applicable for solving Sokoban problems (except very easy ones). Means-end analysis is applicable only in very limited sense and it is not clear how this concept could explain large differences in difficulty of different Sokoban problems. Differences between comprehension of isomorphic problems and successor generation also cannot be responsible for differences in difficulty, because all instances are stated in the same way.

1.4 Contributions

We describe a novel type of experiment with human problem solving using large scale data collection of human problem solving activity over the Internet. Using the collected data on Sokoban puzzle, we identify differences in problem difficulty that are not explained by previous research. To explain these differences, we develop an abstract computational model of human behaviour during search through a problem state space. We also describe a successful difficulty metric which is based on problem decomposition. Full version of the paper, which contains more detailed analysis of collected data, is available as a technical report [10].

2 PRELIMINARIES AND METHODOLOGY

In this section we describe the Sokoban problem, its state space, and methodology of our data collection on human problem solving. We also provide an analysis of the collected data.

2.1 Sokoban and Its State Space

Sokoban is a single player game that was created around 1980 in Japan[3]. Example of the puzzle is in Figure 1. There is a simple maze with several boxes and one man. The goal of the puzzle is to get the boxes into the target squares. The only allowed operation is a push by a man; man can push only one box. For more precise description of rules see, e.g., [1].

To analyze behavior of humans during problem solving, we explore their movement in the underlying problem state space. State space of the game is then formalized as a directed graph $G = (V, E)$ where V is the set of game states and E is the set of edges corresponding to a move of single box[4]. State of the game is thus given by a position of boxes in the maze and by area reachable by a man. We denote s_0 the vertex corresponding to the initial position of the game. In our discussion we consider only states reachable from the initial state.

There can be several states corresponding to a solved problem – all boxes have to be on given position in the final state, but there can be more final states due to the position of the man. Nevertheless, in nearly all cases there is just one final state; thus to simplify the discussion in the following we assume just one final state denoted s_f[5]. We say that a state s is "live" if there exists a path from s to s_f; otherwise we call the state s "dead".

2.2 Data Collection

To obtain data about human Sokoban solving, we used a web portal with a game simulator. Simulator was implemented in Javascript; all moves during the game were sent to the server, where they were stored into a database. We logged description of each move and time spent before the move.

For the experiment we used 35 problems, all of them had 4 boxes and similar size. To avoid bias from learning, we randomized order of problems for each player. Level were given non-explanatory names (fixed prefix + random looking three digit number). Moreover there were 4 sample training problems which players solved to gain the understanding of the problem; these training problems are not included in our analysis.

Participants were not paid and we did not have a direct control over them since the whole experiment run over the Internet. As a motivation to perform well there was a public results list – this is for most people sufficient motivation, and at the same time it is sufficiently weak so that there is not a tendency to cheat.

Summary statistics about the experiment are given in Table 1. In the next section we provide the analysis of data and show that they are sufficiently robust to be used for research purposes.

[3] Sokoban is Japanese for "ware-house keeper"

[4] A naive formulation of a state space is to consider as a move each step of a man. This formulation does not add any important information to the analysis and leads to unnecessary large state spaces.

[5] This simplification is only for sake of readability of the text. Implementations of all our techniques work correctly in the general case of multiple final states.

Table 1. Summary information about data collection.

participants who solved all 35	6
participants who solved at least 20	35
participants who solved at least 1 level	294
successful attempts to solve a problem	2071
all attempts to solve a problem	21511
total time spent solving	785 hours

2.3 Data Analysis

Our aim is to explain and predict difficulty of individual problems. As the first step it is necessary to specify a fair and robust measure of difficulty. There are several natural measures of difficulty: time taken to solve a problem, number of moves necessary to solve a problem, number of solvers who successfully solved a problem. It turns out that all these measures are highly correlated (see [10]), i.e., it seems plausible that any single of them sufficiently captures a concept of problem difficulty. In the rest of the paper we use as a difficulty measure the median solving time of successful attempts.

Figure 2 shows the distribution of solving time using the boxplot method; problems are sorted by the median time (our measure of difficulty). The solving time median varies from one minute to almost one hour. This result is interesting because we are working with very similar Sokoban problems – all levels have exactly four boxes and the size and topology of the underlying maze is in all cases also very similar.

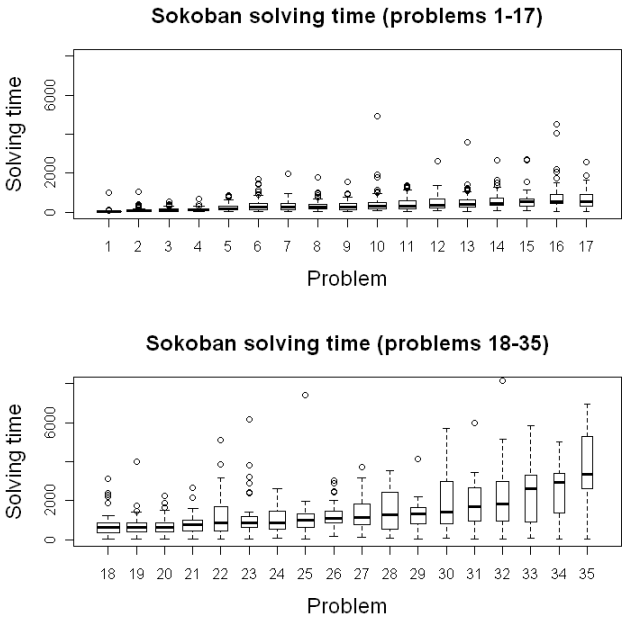


Figure 2. Boxplot of time to solve the Sokoban problems.

In our experiment we saved information about all moves performed by solvers (including time taken to make the move). Although 73.8% of game-configuration are dead (i.e. states from which is impossible to reach the goal), humans usually do not spent much time in dead states (14.5% on the average). They can relatively quickly discover that they are in bad configuration and restart the game. Humans spent more time far from goal position. Once humans get to one half of the distance between the start and goal state, they finish the problem rather quickly (see Figure 3).

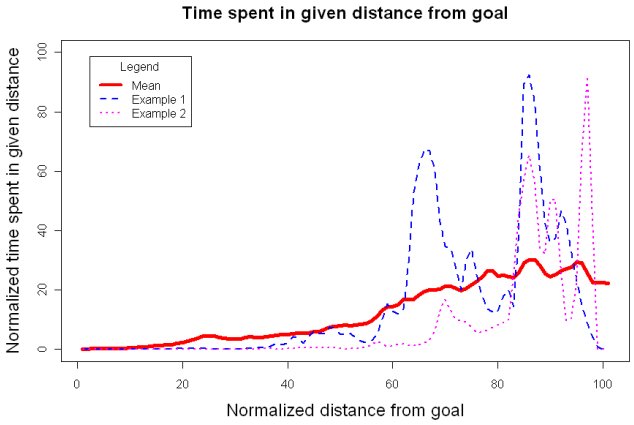


Figure 3. Normalized time spent in states with certain distance from goal. The bold line is a mean over all problems, two colored lines are problems from Figure 1.

3 COMPUTATIONAL MODEL OF HUMAN SOLVER

In this section we describe a simple computational model of human Sokoban solving. Our goal is to replicate only the human behavior in terms of state space traversal characteristics. We do not try to model the actual human cognitive processes while solving the problem, i.e., this is cognitive engineering model rather than cognitive science model [8]. Our model is very abstract and is based only on information about underlying problem state space, i.e., the model is not specific for Sokoban. In the next section we use the model for difficulty rating.

3.1 Basic Principle

Our model is based on the analysis of human behaviour as discussed in Section 2.3. At the beginning humans explore the state space rather randomly, later, as they get closer to the solution, they move more straightforwardly to the goal. Since humans spent most time at live states, the basic model works only with these states and completely avoids dead states.

The model starts at the initial state and then repeatedly selects a successor state. This selection is in the basic model local and very simple – it is a combination of two tendencies:

- random walk – select a random successor,
- optimal walk – select a successor which is closer to the goal state.

Human decisions are usually neither completely random, nor completely optimal. Nevertheless, the model assumes that a weighted combination of these two tendencies can provide a reasonable fit of human behaviour.

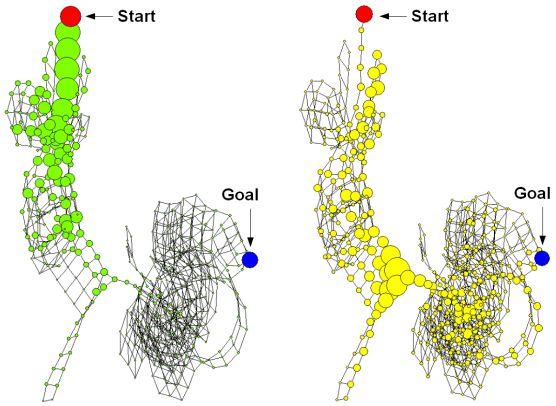


Figure 4. Comparison of human visits (left) and model visits (right) for the example 2. Vertexes represent states of the game, edges represents transitions between states. Size of each vertex is proportional to visits spent by human solvers / model.

3.2 Model Formalization

The general principle of our model is the following. In each step the model considers all successors s' of the current state s. Each successor s' is assigned a value $score(s')$, the sum of all $score$ values is denoted $SumScore$. The model moves to a successor which is selected randomly according to a probabilistic distribution:

$$P(s') = score(s')/SumScore$$

This general model is specified by a selection of a $score$ function. In this report we evaluate the basic version of the model which uses a simple function based on distance $d(s)$ of a state s from the goal state. The function is defined as follows (B is a single parameter of the model – 'optimality bonus'):

$$score(s') = \begin{cases} 0 & d(s') = \infty \\ d(s) & d(s') \neq \infty, d(s') \geq d(s) \\ d(s) + B & d(s') < d(s) \end{cases}$$

Let us discuss the intuition and rationale behind this formula. The first case means that dead states have zero probability of being visited, i.e., the model visits only live states . The second and third case means that successors that lead to the solution get an 'optimality bonus', i.e., they have higher chance of being selected. The use of distance from the goal in the formula has the consequence that the relative advantage of 'bonus' increases as the model gets closer to the goal, i.e., the model behaves less randomly when it is close to the goal (as do humans).

If $B = 0$ than the model behaves as a pure random walk (within live states). As B increases the behaviour of the model converges to the optimal path. Hence by tuning the parameter B the model captures continuous spectrum of behaviour between randomness and optimality.

Figure 4 gives an example of a comparison of human and model state space traversal.

3.3 Possible Extensions

In this work we discuss and evaluate just the simplest reasonable version of the model. Nevertheless, the model can be further extended in several ways. The extensions are quite natural and can be done simply by extending the scoring function:

- Hill climbing heuristic (specific for the particular problem, i.e., Sokoban). For Sokoban the natural heuristic is the total distance of boxes from goal positions.
- Use of memory (loop avoidance heuristic). The model would remember states that were already visited and in the scoring function would prefer unvisited states.
- Exploration of dead states (with lower, but non-zero, probability than live states).
- Penalization of long back edges. Humans can recognize not just moves which lead to dead states, but also moves which lead "backwards".

Each of these extensions incorporates at least one additional parameter into the model. With the size of our testing data (35 problems) it could be misleading to evaluate versions of the model with more parameters due to the potential overfitting of data [19].

The model can be easily used as a basis for a difficulty metric. We just run the model repeatedly and take the number of steps it took the model to reaching the goal state. The average number of steps is used as a difficulty metric.

4 METRICS BASED ON THE SHORTEST PATH

In this section we describe metrics based on the shortest path.

4.1 Basic Metrics

The most intuitive difficulty metric is the 'number of steps necessary to reach a solution', i.e., the length of the shortest path from initial to goal state in the state space. Other metrics can be obtained as variations on this basic principle.

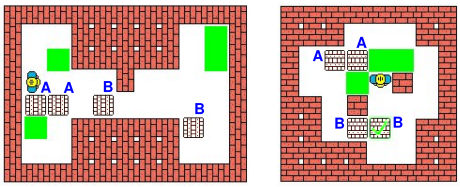
One of the concepts which was intensively studied in previous research on problem solving is the hill-climbing heuristics [3, 4, 18]. This concept can be quite directly applied as a difficulty metric. The straightforward hill-climbing heuristic for Sokoban is to minimize the total distance of boxes from their goal positions. Given this heuristic, we can define the metric 'counterintuitive moves' as a number of steps on the shortest path which go against this hill-climbing heuristic.

4.2 Problem Decomposition

Another intuitively important concept in problem solving is problem decomposition. Humans are not good at systematic search, but they are good at tasks such as abstraction, pattern recognition, and problem decomposition. If a problem can be decomposed into several subproblems it is usually much simpler (for humans) than a same type of problem which is highly interwoven and indecomposable (see example in Figure 5). The concept of problem decomposition is however more difficult to grasp than the hill-climbing heuristic.

We propose a way to formalize problem decomposition for a Sokoban puzzle. A natural unit of 'composition' is a single box. Thus we can consider decomposition of a problem into single boxes and than count as a single move any series of box pushes. We can also generalize this idea and decompose the problem into two pairs of boxes[6] and than count as a single move any series of box pushes within the group.

[6] Remember that all our problems contain 4 boxes.



problem	decomposition			
	ABCD	**AABB**	ABAB	ABBA
left problem	10	**2**	6	5
right problem	14	**7**	12	10

Figure 5. Example of two Sokoban puzzle; the first one can be easily decomposed into two subproblems and is thus easy (median solving time 3:02 minutes), the second one is rather indecomposable and thus very difficult (median solving time 53:49 minutes). The table gives the number of 'steps' for different decomposition (see text). The bold column corresponds to the decomposition provided in the figure.

Let D be a division of n boxes into several groups (at most n), in our case $n = 4$ and we denote the division by 4 letter string; e.g., 'ABAB' is a division in which the first and the third box are in group A, the second and the fourth box are in group B; 'ABCD' is a division in which each box is in a separate group. Each edge in the state space is labelled by identification of the group to which the moved box belongs. Let p be a path in a state space (a sequence of valid moves). We are interested in the number of label alternations $a(p, D)$ along the path.

Optimal solution of the problem with respect to a division D (denoted $s(D)$) is the minimum $a(p, D)$ along all paths from the initial to the goal state. This optimal solution can be computed by the Dijkstra algorithm over an augmented state space – vertices are tuples (s, g) where s is a state in the state space and g is an identification of a group and edges have weights 0 or 1. Figure 5 gives results for different decompositions of the two provided examples.

For our evaluation we use two metrics based on these concepts. At first, we use the 'box change' metric which is based on the division 'ABCD', i.e., each box is a single group. At second, we use the '2-decomposition' metric, which is the minimum number of steps over all possible division into two groups (division 'AABB', 'ABAB', 'ABBA'). We also tried other types decompositions (e.g., 3-1 decomposition such as 'AAAB'), but the results were similar to 2-decomposition and we do not report them explicitly.

5 Evaluation

Based on results from Section 2.3, we take as a 'real' measure of problem difficulty a median solving time of human solvers. Figure 6 provides scatter plots showing relation among several of the described metrics and the real difficulty.

To quantify the quality of a metric we use correlation coefficients. Except for the standard Pearson correlation coefficient, we also measure Spearman correlation coefficient, which gives the correlation with respect to ordering of values – for practical application of difficulty metrics the ordering is often more important than absolute values.

Table 2 provides overview of all results. The results shown for metric based on model correspond to the optimal choice of the bonus parameter B. We have done sensitivity analysis of the metric based on the computational model with respect to the parameter – Spearman coefficient is quite stable, but Pearson coefficient is not (for details see [10]).

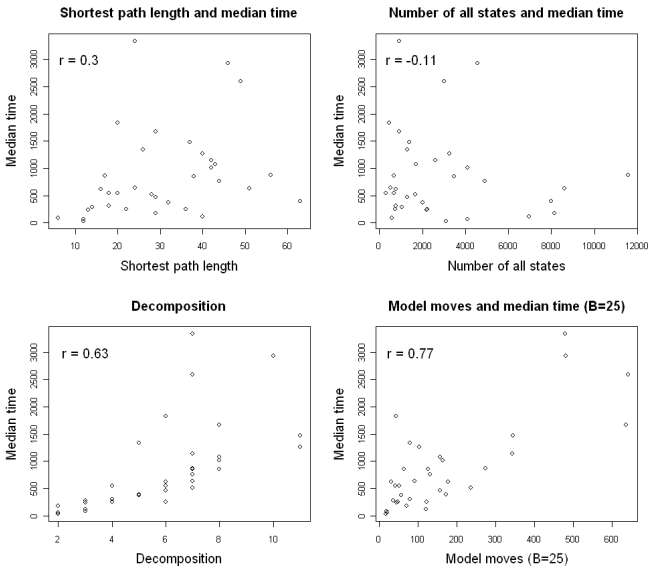


Figure 6. Scatter plots for 4 different difficulty metrics.

We also evaluated the most straightforward approach based on the size of state space, which was initially proposed by Newell and Simon [21]. This metric, however, does not achieve statistically significant correlation. Similar results were obtained for other metrics based on global state space parameters (see [10]).

The intuitively plausible metric based on length of the solution also provides rather poor results. An improvement is brought by Sokoban specific extension of shortest paths (counterintuitive moves) and the best results are obtained by the metric based on problem decompositions and by the metric based on computational model. We believe that the results of the computational model can be further improved by extensions discussed in Section 3.3, but at the moment we do not have enough data to fairly evaluate a model with several free parameters.

Table 2. Correlation coefficients for different difficulty metrics, results given in bold are statistically significant ($\alpha = 0.05$).

type	metric	Pearson	Spearman
state space	size	-0.11	-0.07
shortest paths	shortest path	0.30	**0.47**
	counterintuitive moves	**0.52**	**0.69**
problem decomposition	box change	**0.51**	**0.74**
	2-decomposition	**0.63**	**0.82**
model	avg. steps, $B = 25$	**0.76**	**0.66**

6 CONCLUSIONS AND FUTURE WORK

In this paper we study methods for difficulty rating of a Sokoban puzzle. We describe two successful methods of difficulty rating – computational model based on the problem state space and problem decomposition based on division of a given problem into several subproblems. Using these methods, we are able to partially predict the

difficulty of the given puzzles and thus to explain the large differences of the recorded Sokoban solving time. We believe that both techniques should be applicable to other problems as well.

There are many direction for future research:

- evaluation of the computational model on other transport puzzles (Rush hour, Replacement puzzle),
- development of the score function without whole state space knowledge,
- a more detailed study of data on human problem solving, particularly study of differences among individual solvers (comparison between experts and average solvers),
- application of studied metrics to improve recommender systems in intelligent tutoring systems (evaluation of unsolved puzzles),
- development of a tutor system devoted to problem solving skill acquisition.

Acknowledgement

We thank Ondřej Bouda for assistance with the web experiment and Aymeric du Peloux, David W. Skinner, M. Hiroshi, and Mart Homs Caussa for creating some of the Sokoban problems which were used in our experiments.

REFERENCES

[1] Sokoban wiki. `http://www.sokobano.de/wiki`.
[2] J.R. Anderson, C.F. Boyle, and B.J. Reiser, 'Intelligent tutoring systems', *Science*, **228**(4698), 456–462, (1985).
[3] M.E. Atwood and P.G. Polson, 'Further explorations with a process model for water jug problems', *Memory & Cognition*, **8**(2), 182–192, (1980).
[4] H.P. Carder, S.J. Handley, and T.J. Perfect, 'Counterintuitive and alternative moves choice in the Water Jug task', *Brain and Cognition*, **66**(1), 11–20, (2008).
[5] J. Culberson, 'Sokoban is PSPACE-complete, Technical Report', Department of Computing Science, The Univerzity of Alberta, 1997.
[6] D. Dor, U. Zwick, 'Sokoban and other motion planning problems', 1996.
[7] M. Dry, M.D. Lee, D. Vickers, and P. Hughes, 'Human performance on visually presented traveling salesperson problems with varying numbers of nodes', *Journal of Problem Solving*, **1**(1), 20, (2006).
[8] W. D. Gray, *The Cambridge Handbook of Computational Psychology*, chapter Cognitive Modeling for Cognitive Engineering, 565–588, Cambridge University Press, 2008.
[9] J.G. Greeno, 'Hobbits and orcs: Acquisition of a sequential concept', *Cognitive Psychology*, **6**(2), 270–292, (1974).
[10] P. Jarušek, R. Pelánek *Human Problem Solving: Sokoban Case Study*, Technical Report, Masaryk Univerzity, Faculty of Informatics, 2010.
[11] A. Junghanns, *Pushing the limits: New developments in single-agent search*, Ph.D. dissertation, University of Alberta, Department of Computing Science, 1999.
[12] A. Junghanns, S. Jonathan, 'Sokoban: A Challenging Single-Agent Search Problem', Department of Computing Science, The Univerzity of Alberta, 1997.
[13] A. Junghanns, J. Schaeffer, 'Sokoban: Enhancing general single-agent search methods using domain knowledge', Department of Computing Science, The Univerzity of Alberta, 2000.
[14] K. Kotovsky, J.R. Hayes, and H.A. Simon, 'Why are some problems hard? Evidence from tower of Hanoi', *Cognitive psychology*, **17**(2), 248–294, (1985).
[15] K. Kotovsky and H.A. Simon, 'What Makes Some Problems Really Hard: Explorations in the Problem Space of Difficulty.', *Cognitive Psychology*, **22**(2), 143–83, (1990).
[16] A. Newell and H.A. Simon, 'GPS, a program that simulates human thought', *Computers and thought*, 279–293, (1963).
[17] Z. Pizlo, 'Human Problem Solving in 2006', *The Journal of Problem Solving*, **1**(2), 3, (2007).
[18] Z. Pizlo and Z. Li, 'Solving combinatorial problems: The 15-puzzle', *Memory and Cognition*, **33**(6), 1069, (2005).
[19] S. Roberts and H. Pashler, 'How persuasive is a good fit? A comment on theory testing.', *Psychological Review*, **107**(2), 358–367, (2000).
[20] J. Schaeffer and H.J. Van den Herik, 'Games, computers, and artificial intelligence', *Artificial Intelligence*, **134**(1-2), 1–8, (2002).
[21] H.A. Simon and A. Newell, *Human problem solving*, Prentice Hall, 1972.
[22] L.G. Terveen, 'Overview of human-computer collaboration', *Knowledge Based Systems*, **8**(2), 67–81, (1995).
[23] R.A. Wilson and F.C. Keil, *The MIT encyclopedia of the cognitive sciences*, MIT Press, 1999.