I.J. Intelligent Systems and Applications, **2017, 3, 1-7**
Published Online March 2017 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2017.03.01

Cognitive Agents and Learning Problems

Goran Zaharija
University of Split, Faculty of science, 21000 Split Croatia
E-mail: goran.zaharija@pmfst.hr

Saša Mladenović and Stefan Dunić
University of Split, Faculty of science, 21000 Split Croatia
E-mail: smladen@pmfst.hr, stefan.dunic@pmfst.hr

Abstract—Goals, Operators, Methods, and Selection rules (GOMS) model is a widely recognised concept in Human-Computer Interaction (HCI). Since the initial idea, several GOMS techniques were developed that were used for analysis, differing in their form defined by the logical structure and prediction power. Through defined operators and methods and following the certain rules, the user can reach a specific goal. This work represents an effort to apply GOMS method in the field of artificial intelligence, specifically on a state-space search problems. Card, Morgan, Newman GOMS (CMN-GOMS) model has been chosen, since it represents ground-floor of the GOMS idea that solves the given task through a sequence of operators. Compared with the informed search algorithms for solving the given task, CMN-GOMS model gave better results. Moreover, it was shown that this model could be used in any other space motion problem in the natural environment. LEGO® MINDSTORMS® EV3 robot was used to demonstrate the application of GOMS model in real world pathfinding problems and as a test-bed for comparing proposed model with well-known search algorithms.

Index Terms—Cognitive processes, search algorithms, GOMS model, robots, artificial intelligence.

I. Introduction

Goals, Operators, Methods, and Selections rules (GOMS) model is a widely recognised concept for Human-Computer Interaction (HCI). Its core lies in the assumption of the presence of an intelligent user who through goals, operators, methods and selections rules, can efficiently solve a given problem, which makes the method itself interactive.

From the cognitive learning perspective, learning involves the transformation of information from the environment into knowledge stored in the mind. Learning occurs when new knowledge is acquired or existing knowledge is modified by experience. Cognitive learning theories are used to explain simple tasks such as remembering the name of a new friend as well as the complex ones such as interpreting an abstract drawing. This learning approach focuses on how children process information through attention, memory, thinking, and other cognitive processes. Social cognitive learning perspective examines the process involved as people learn from observing others and gradually acquire control over their own behaviour. In other words, social cognitivists believe that people learn a new behaviour simply by watching what other people do.

On the other hand, Informed Searched Algorithms (ISA) that do not have external influence cannot always give the optimal solution [1].

GOMS model was developed by S.K. Card, T.P. Moran and A. Newell [2]. Depending on a given task, the type of the desirable feedback and the method implementation (sequential or parallel) there are several GOMS models that can be used: Keystroke-Level Model [3], Card, Moran, Newell GOMS [4], Natural GOMS Language [5] and Cognitive-Perceptual Motor GOMS [6]. GOMS model has a broad range of usage in computer science due to its ability to quantitatively and qualitatively analyse ways of using a particular system (forecasting the execution time, recognition and prediction of error influence [7]).

Problem solving in Artificial Intelligence (AI) can be presented as a systematical search of possible outcomes with a task of finding some predefined goal or solution. One of the most used approaches to problem solving in AI is applying search algorithms [8]. In Computer Science, a search algorithm is considered an algorithm that is used for evaluating a set of all possible states and selecting an appropriate solution.

As opposed to the GOMS model, to solve a given task search algorithms are relying on heuristic technique, i.e. they are designed through given set of rules based on general experience to find the most appropriate solution for a given task [9].

There are many different problems applicable to AI and search algorithms, but most of them can be placed in one of three main categories [10]: path-finding problems, two-player games and constraint-satisfaction problems. In real world situations, especially those involving robots, path-finding problems are the ones most occurring [11, 12]. Furthermore, path-finding problems are closest to the state-based model of the world, with each position representing a single state.

One of the most common problems used for testing robots and their AI are maze traversal problems [13]. A maze is, in most cases, two-dimensional lattice-like structure, consisting of a finite number of identically sized cells [14]. A robot is placed in one empty cell

(initial state) called starting cell, with the task of finding the way to the target cell in the maze (goal state). Not all cells are adjacent so robot must traverse through multiple cells in order to reach its goal. Some cells can contain obstacles, thus preventing a robot from passing through that cell. There can be multiple passageways to the target cell.

In this paper, we will analyse the LEGO® MINDSTORMS® EV3 robot [15] movement through a given space with obstacles. For this paper, CMN-GOMS model will be used since it was the first designed GOMS model that provides the proper GOMS idea. Our intention is to show the quality of the GOMS model in the learning process by comparing it to the ISA. Section 2 contains the basic theoretical description of the methods and material used in our research, while Section 3 covers the results. Discussion and conclusions are provided in Section 4.

II. Methods and Materials

A. GOMS model

GOMS model consists of four components which define cognitive structure used for data processing:

1. Set of goals which define state of a given task,
2. Set of operators that include perceptive (recognition of the surroundings), motoric (physical movement) and cognitive (data interpretation) data processing used in defining the framework,
3. Set of methods made of subgoals and operators usually sequentially used for achieving a particular goal,
4. Set of selections rules that define control structure needed for choosing the optimal method to solve a given task.

To reach a final goal (usually on the highest level of abstraction), the model performs a finite number of steps which includes usage of defined operators representing the lowest level of abstraction. Combination of a given number of actions together produces a single method. In general, the method that requires the minimum amount of time and steps for reaching the final goal is considered to be the most efficient one. This metric corresponds to the time complexity metrics of search algorithms [16]. Fig. 1 shows a simplified structure of the GOMS model that consists of initial state, selections rules, operators, methods, and final goal.

Execution time is usually experimentally measured for each of the components separately. The total execution time is calculated simply by adding the execution time of each component. In our case, execution time depends on physical characteristics of used robot and its construction.

The most used type of GOMS model is CMN-GOMS, which strictly follows the sequential logical structure of the framework. It assumes that user possesses all required knowledge and doesn't need to search the environment previous to the analysis [17]. For that reason, CNM-GOMS also can forecast sequence of the operators required to achieve a particular goal. Card et al. (1983) successfully demonstrated its forecasting abilities, as well as the estimation time needed to complete a given task on the examples of textual editing and operating systems. Even though the model has defined methods and goal structure, there is no predefined manual for set-up and execution of the method itself. Consequently, it is easy to create and implement CNM-GOMS framework which allows users to set up and complement methods inside the framework.

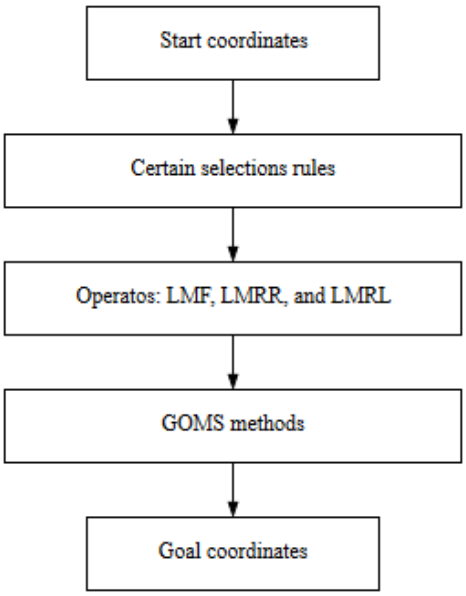

Fig.1. Simplified structure of the GOMS model for reaching the final goal from the initial state through defined selections rules, operators, and methods.

Table 1. GOMS methods used in the simple movement problem.

GOMS method #1	GOMS method #2	GOMS method #3
2 x LMRL	2 x LMRL	2 x LMRL
LMF	LMF	LMF
LMRL	LMRR	LMRL
3 x LMF	2 x LMF	3 x LMF
LMRL	LMRR	LMRL
4 x LMF	6 x LMF	2 x LMF
LMRL	LMRR	LMRL
3 x LMF	2 x LMF	LMF
	LMRR	LMRR
	2 x LMF	LMF
		LMRL
		LMF
		LMRR
		LMF
		LMRL
		LMF

In our research, we used three basic movement operators: Large Motors Forward (LMF), Large Motors

Rotate Left (LMRL), and Large Motors Rotate Right (LMRR). These operators correspond to the basic low-level robot actions [18]. Three different GOMS methods used in our analysis of the simple movement problem are listed in Table 1.

B. Informed search algorithms

Directed search strategies use heuristic information while performing a search in order to determine which solution is more promising than the other. Heuristic information is problem dependant (domain specific information) and must be known in advance. Informed search algorithms are designed to solve a given task as fast and efficiently as possible. The heuristic function evaluates every possible action and returns an estimated cost of performing that action. In this way, the process of achieving the desired goal is accelerated. It provides a way to inform the search about the direction to a goal. Using heuristics reduces problem dimension from exponential to polynomial. However, there is no guarantee that goal will be found or that solution would be optimal [19, 20].

In this paper two types of ISA are used; Best-First Search (BFS), and A* search algorithm.

BFS is one of the most popular heuristic-based algorithm [21] that represents the basis for the most developed ISA. The algorithm is used in the contest of path-searching problem that solves various combinatorial problems (path determination, planning, speech recognition, analysis and exploration of space, [22]). Key elements of the BFS algorithm are closed and open list. BFS pseudocode is shown in Fig.2

```
Best-first search {

closed list = [ ]
open list = [start node]

    do {
        if open list is empty then{
            return no solution
        }
        n = heuristic best node
        if n == final node then {
            return path from start to goal node
        }
        foreach direct available node do{
            if node not in open and not in closed list do {
                add node to open list
                set n as his parent node
        }
        delete n from open list
        add n to closed list
    } while (open list is not empty)
}
```

Fig.2. Pseudocode for Best-First Search algorithm.

A* search algorithm is one variant of the BFS algorithm that is broadened in a heuristic sense to make it more complete and optimal than the BSF. While searching the shortest path, in addition to the basic heuristic restrictions, a limit for the minimal cost is also included. Precisely, while searching for an optimal solution, A* search algorithm through each step inspects whether chosen path was the shortest one. This type of verification was optional in BFS, whereas in A* is obligated. Fig.3 shows A* search pseudocode. Implementation of A* search algorithm often requires adaptation to a particular problem due to the temporal and space complexity, memory management and various other factors [23].

```
A* search {

closed list = [ ]
open list = [start node]

    do {
        if open list is empty then {
            return no solution
        }
        n = heuristic best node
        if n == final node then {
            return path from start to goal node
        }
        foreach direct available node do{
            if current node not in open and not in closed list do {
                add current node to open list and calculate heuristic
                set n as his parent node
            }
            else{
                check if path from star node to current node is
                better;
                if it is better calculate heuristics and transfer
                current node from closed list to open list
                set n as his parrent node
            }
        delete n from open list
        add n to closed list
    } while (open list is not empty)

}
```

Fig.3. Pseudocode for A* search algorithm.

C. LEGO® MINDSTORMS® EV3

With technology improvement, LEGO® company developed series of LEGO® MINDSTORMS® robots with possibilities to move, communicate and “think”. For the purpose of our research EV3 model was used. With the help of computer programming, EV3 can solve various tasks, such as systematic space exploration, recognition and bypassing obstacles, and others. Main components of EV3 are (Fig.4): EV3 Brick, EV3 Motors (Large and Medium Motor), Infrared Sensor, Remote Infrared Beacon, Touch Sensor, and Colour Sensor.

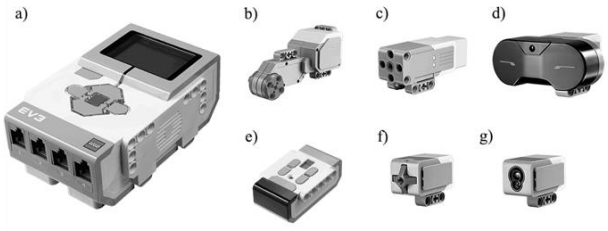

Fig.4. Main components of EV3: (a) EV3 Brick, (b) Large Motor, (c) Medium Motor, (d) Infrared Sensor, (e) Remote Infrared Beacon, (f) Touch Sensor, and (g) Colour Sensor.

D. EV3 virtual environment application

A framework designed for solving simple movement problem of EV3 robot by using BFS and A* search algorithms, as well as the GOMS methods is called "EV3 virtual environment" application. The application was developed using integrated development environment

(Microsoft Visual Studio) and C# programming language. Fig.5 shows the main screen of the application consisting of four sections:

- "Virtual environment" section – place where user chooses environment dimensions and sets up the initial conditions of the movement problem,
- "GOMS" section - the place where a user can visually see the implementation of GOMS model. This section allows a user to create methods using three types of available operators (LMF, LMRR, and LMRL) or to choose one of the two predefined algorithms of ISA. After defining all potential methods, a user can, depending on the chosen selections rule, find the optimal method among the possible ones.
- "EV3 connection" section - enables Bluetooth communication with the EV3 robot. Information about the method is sent through a sequence of operators (steps).
- "Results" section – displays the outputs of the analysis.

This environment serves as test-bed [24] for comparing different ISA and GOMS models. Although current (presented) version of the virtual environment includes only two different search algorithms (not counting GOMS models), additional algorithms can easily be implemented and used for comparison.

The performance of each method is analysed trough three separate parameters: travelled distance, number of used operators and total execution time. Distance is calculated using Manhattan metrics [25] with Von Neumann neighbourhood [26]. In practice, that means that every cell in a maze has only four neighbouring cells (north, east, south, and west).

E. Simple movement problem

Every agent, simulated or physical, is situated inside some particular environment and interacts with that environment. Type and intensity of interaction depend on agent's architecture and capabilities. Regardless, agent's environment must always be taken in consideration, as it cannot be observed isolated from its environment. For the purpose of this paper, environment properties were defined based on properties defined by Russel and Norvig [27]. Proposed agent architecture is intended for fully observable (and consequently deterministic), static, sequential and discrete environments.

Identifying the environment in which the agent (robot) operates is important step in valid problem representation. To demonstrate natural movement we designed a situation shown in Fig.6 The simple pathfinding problem consists of the start position, randomly placed obstacles and the goal position. After setting the problem in the natural environment, we recreated the same problem inside the virtual environment. The objective of our task was to find the optimal method to reach the goal from the start position. The method was later sent to EV3 robot to execute the actions in the real world.

Fig.5. Main screen of the „EV3 virtual environment" application consisting of 4 sections ("virtual environment", "GOMS", "EV3 connection", and "results" section).

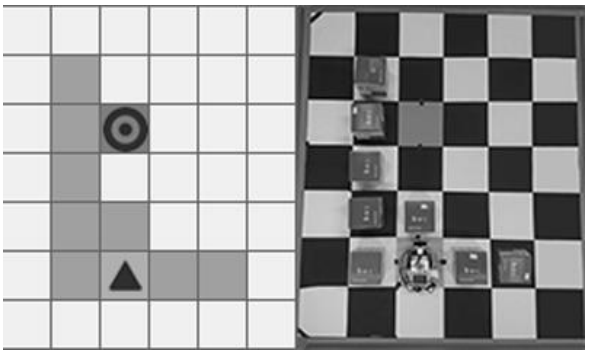

Fig.6. Environment set up of a movement problem. Virtual implementation of designed situation is shown on the left, while the same situation in the natural environment is shown on the right.

III. Setting the Movement Problem and the Results

In the EV3 virtual environment application, we set our movement problem illustrated in Fig.6 The application was designed in a way that user can either create his desirable methods using three defined operators (Fig.7) or simply by choosing two of the previously defined ISA. The user can test and save all the implemented methods.

We intentionally created three different GOMS methods listed in Table 1 to demonstrate their efficiency compared to the methods given by the ISA.

Therefore, two of the GOMS methods were designed specifically to recreate the same moving paths as they were given by the BFS and A* search algorithms, and the third one was intended to be the optimal one according to the expert or "super-user". Fig.8 shows the virtual setup of our pathfinding problem with calculated solutions (paths) coming from 5 methods mentioned above.

The analysis is performed by comparing all of the saved methods. Two selections rules were implemented inside the application considering the cost of the natural movement:

1. Restriction to minimum number of used operators, and
2. Restriction to minimum execution time.

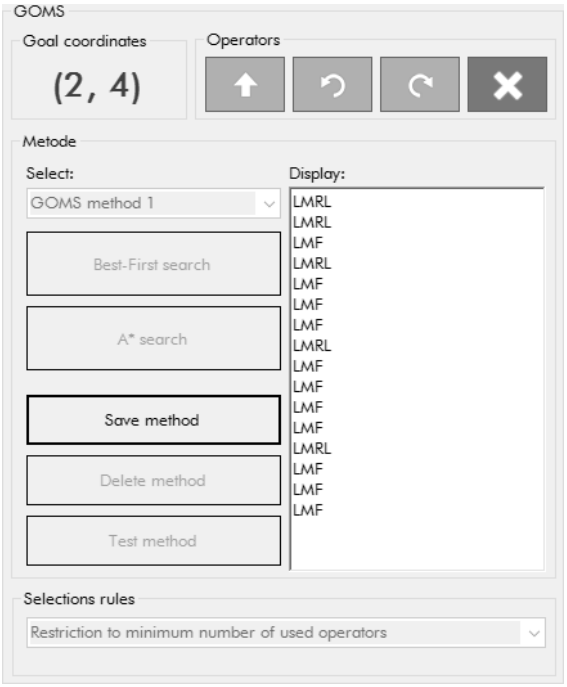

Fig.7. Creation of GOMS method #1 inside the EV3 virtual environment application.

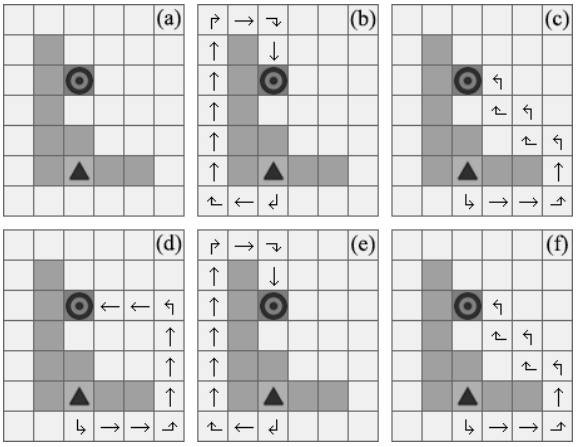

Fig.8. Virtual set up of the movement problem (a), and solutions of the BFS algorithm (b), A* algorithm (c), and three GOMS methods listed in Table 1 (d, e, f) sorted respectively.

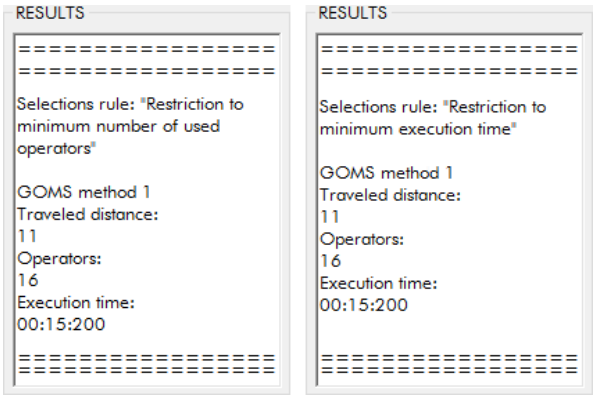

Fig.9. Analytic report of the five tested methods. In both cases of selections rules, the analysis gives the GOMS method #1 as the optimal one.

After choosing the selections rule, the cost of all methods is calculated, and the solution of the optimal method is provided. Total execution time for ISA methods was calculated by adding the time which algorithms need to explore the given space to find its optimal solution and the time required to execute the sequence of operators. Operators that EV3 robot can use have already defined execution time, calculated by measuring the execution time of each action in the real world using the same robot construction; LMF – 1.2 s, LMRR and LMRL – 0.4 s. Total execution time of GOMS methods is determined in a similar way. Instead of time required to explore the space and find the solution, we created a simulation that assumes the presence of a "super-user" (an expert) who already possesses all the required knowledge and can "imagine" the solution (path) in front. In this way, "thinking" time is significantly shorter compared to the ISA exploration time. This approach belongs to the field of “Expert systems”, which are used for embedding the human expert’s domain knowledge within a particular system [28].

Analysis of all methods proposed to the GOMS model, in both cases of selections rules, i.e. comparing the number of used operators or total execution time, gives the GOMS method #1 as the optimal one (Fig.9).

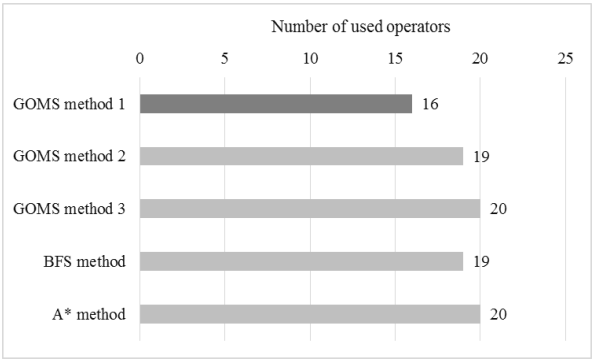

Fig.10. Number of used operators to reach the final goal for five methods used in the analysis.

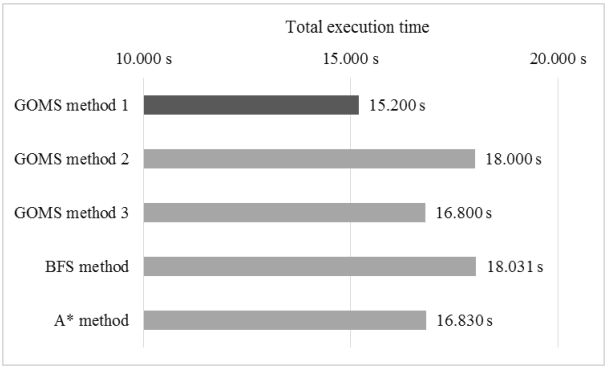

Fig.11. Total execution time required to find the final goal for five methods used in the study.

Comparing the methods, GOMS method #1 uses only 16 operators, whereas GOMS method #3, as well as the A* method, used 20 operators to reach the final goal (Fig. 10). Shortest total execution time was achieved in the case of GOMS method #1 being 15.200 s, while BFS

method needed 18.031 s for reaching the goal (Fig. 11). As said before, GOMS method #2 and #3 were created to give same solutions (paths) as BFS and A* methods, respectively, which can be noticed in Fig. 8 and Fig. 10. GOMS method #2 and BFS methods required 19 steps, whereas GOMS method #3 and A* methods required 20 steps in total. It is interesting to notice that, even though the pairs of methods are almost identical (consisting of the same number of operators), total execution time was different. In both cases, GOMS methods resulted in somewhat shorter execution time (~0.03 s shorter, Fig. 11).

IV. Discussion and Conclusions

The aim of this paper was to analyse and solve the simple movement problem simulated by the LEGO® MINDSTORMS® EV3 robot. For that purpose, we used two different approaches; ISA and GOMS model. Moreover, we demonstrated the advantages of the GOMS model usage over ISA.

In our work, we created three different GOMS methods and two ISA methods consisting of three basic operators (LMF, LMRR, and LMRL). Depending on the selections rules that restrict total execution time and number of used operators to a minimum value, in both cases, the analysis gave GOMS method #1 as the optimal one since it requires the smallest number of used operators and the shortest execution time (Fig.10 and Fig.11). It can be noticed that the path (result) given by the A* method (Fig.8c) has the same travelling distance as the GOMS method #1 (11 coordinate displacements). However, A* method includes four more unnecessary rotations increasing the number of used operators that adds up to the total execution time (4 x 0.4 s = 1.6 s, Fig. 10 and Fig. 11). Moreover, comparing created GOMS methods with the similar ISA methods (BFS and A* methods) GOMS methods resulted in somewhat shorter execution time. This is due to the assumption of the presence of an expert user with expert knowledge inside the GOMS methods who has all the information upfront and therefore requires less amount of time to calculate the solution. Nevertheless, for the purpose of our research we set up the movement problem in a simple way. For more complex situations, temporal gap between ISA methods and GOMS replications would probably increase.

Defining the selection rules is the crucial part of the GOMS model. In highly complex environmental situations which can include other types of obstacles rather than just some simple barriers it is vital to set the rules accurately to get the most efficient solution. We propose an artificial neural network to be imposed as a decision maker.

As opposed to the ISA methods GOMS model has an advantage that lies in the "super-user" hypothesis. However, this advantage can also be its disadvantage because the model itself cannot forecast all possible defects of the final user. In our case, EV3 robot was the final user with learning abilities and the application was the tool used to simulate and implement cognitive thinking of the expert user inside the robot. During the test demonstrations, EV3 robot had problems with physical rotation by a 90 ° that was a part of the two primary operators (LMRR and LMRL). There was no way for our application to predict this defect and therefore multiple repetitions of the experiment were necessary. Nevertheless, assuming that all perceptive, cognitive and motoric abilities function properly, GOMS model was proven to be adequate for the movement problem and also more efficient than ISA. In the end, it is better and faster to receive cognitive knowledge than learn it by yourself, especially in the situations with more complexity.

Application developed for the purpose of our research (EV3 virtual environment) can also be used for cognitive development in the sense of a fundamental logic and abstract thinking. Through the process of solving the wanted movement problem, a user can set-up virtual environment, create different methods, analyse and correct potential errors, and finally reach the desired goal.

For future work, there are several possibilities to extend the work presented in this paper. Our first intention is to implement additional search algorithms inside the virtual environment application. That would allow us to carry out a more detailed comparison between different approaches in pathfinding problems. Also, considering the problems with the physical robot, it is obvious that motor functions can be improved. Finally, our aim is to expand the research by testing the existing and future search algorithms on increased number of mazes, thus obtaining more accurate results.

References

[1] Gendreau, Michel, and Jean-Yves Potvin. "Tabu search." *Search methodologies*. Springer US, 2014. 243-263.

[2] Tauber, M. G., & Ackermann, D. (Eds.). (2013). *Mental models and human-computer interaction* (Vol. 7). Elsevier

[3] Rice, Andrew D., and Jonathan W. Lartigue. "Touch-level model (TLM): evolving KLM-GOMS for touchscreen and mobile devices. "*Proceedings of the 2014 ACM Southeast Regional Conference*. ACM, 2014.

[4] Guo, Hua, Diem Tran, and David H. Laidlaw. "Incorporating GOMS analysis into the design of an EEG data visual analysis tool. "*Visual Analytics Science and Technology (VAST), 2012 IEEE Conference on*. IEEE, 2012.

[5] Ritter, Frank E., Gordon D. Baxter, and Elizabeth F. Churchill. "Methodology I: Task Analysis. "*Foundations for Designing User-Centered Systems*. Springer London, 2014. 309-333.

[6] Patton, E. W., Gra, W. D., & John, B. E. (2012, September). Automated CPM-GOMS modeling from human data. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* (Vol. 56, No. 1, pp. 1005-1009). SAGE Publications.

[7] Ramkumar, A., Stappers, P. J., Niessen, W. J., Adebahr, S., Schimek-Jasch, T., Nestle, U., & Song, Y. (2016). Using GOMS and NASA-TLX to Evaluate Human-Computer Interaction Process in Interactive Segmentation. *International Journal of Human–Computer Interaction*, 1-12.

[8] Landau, R. H., PÃ, M. J., & Bordeianu, C. C. (2015). *Computational Physics: Problem Solving with Python*. John Wiley & Sons.
[9] Hansson, Othar, and Andy Mayer. "Heuristic search as evidential reasoning." *arXiv preprint arXiv: 1304. 1509* (2013).
[10] Korf, R. E. (2010). *Artificial intelligence search algorithms* (pp. 22-22). Chapman & Hall/CRC.
[11] Nahm, Y. E., & Ishikawa, H. (2005). A hybrid multi-agent system architecture for enterprise integration using computer networks. *Robotics and Computer-Integrated Manufacturing*, *21*(3), 217-234.
[12] Ni, Z., & He, H. (2013). Heuristic dynamic programming with internal goal representation. *Soft computing*, *17*(11), 2101-2108.
[13] Yadav, S., Verma, K. K., & Mahanta, S. (2012). The Maze problem solved by Micro mouse. *International Journal of Engineering and Advanced Technology (IJEAT) ISSN*, 2249-8958.
[14] Foltin, M. (2011). Automated Maze Generation and Human Interaction. *Brno: Masaryk University Faculty Of Informatics*.
[15] LEGO® MINDSTORMS® EV3, Manual, http://www.lego.com/en-us/mindstorms/, 26. 08. 2016.
[16] Woeginger Gerhard J. "Space and time complexity of exact algorithms: Some open problems." *International Workshop on Parameterized and Exact Computation*. Springer Berlin Heidelberg, 2004.
[17] John, B. E. and Kieras, D. E. The GOMS Family of User Interface Analysis Techniques: Comparison and Contrast, ACM Transactions on Computer-Human Interaction, 3 (1996), 320-351.
[18] Argall, Brenna D., et al. "A survey of robot learning from demonstration." *Robotics and autonomous systems* 57.5 (2009): 469-483.
[19] Ai-bing, N. I. N. G., Liang, M. A., & Xiao-hua, X. (2007, August). Solving degree-constrained minimum spanning tree with a new algorithm. In 2007 International Conference on Management Science and Engineering (pp. 381-386). IEEE.
[20] Kotthoff, L. (2014). Algorithm selection for combinatorial search problems: A survey. *AI Magazine*, *35*(3), 48-60.
[21] Dechter, R. and Pearl, J. Generalized best-first search strategies and the optimality of A*, Journal of the ACM, 32 (1985), 505-536.
[22] Pearl, J. Heuristics: Intelligent Search Strategies for Computer Problem Solving. Addison-Wesley, Reading, Mass., 1984.
[23] Liu, Xiang, and Daoxiong Gong. "A comparative study of A-star algorithms for search and rescue in perfect maze." *Electric Information and Control Engineering (ICEICE), 2011 International Conference on*. IEEE, 2011
[24] Hanks, Steve, Martha E. Pollack, and Paul R. Cohen. "Benchmarks, test beds, controlled experimentation, and the design of agent architectures." *AI magazine*14.4 (1993): 17
[25] Zille, H., & Mostaghim, S. (2015, May). Properties of scalable distance minimization problems using the Manhattan metric. In 2015 IEEE Congress on Evolutionary Computation (CEC) (pp. 2875-2882). IEEE.
[26] Min, X., Xu, X., & Wang, Z. (2014, June). Combining von Neumann neighborhood topology with approximate-mapping local search for ABC-based service composition. In Services Computing (SCC), 2014 IEEE International Conference on (pp. 187-194). IEEE.
[27] Russell, S. J., Norvig, P., Canny, J. F., Malik, J. M., & Edwards, D. D. (2003). *Artificial intelligence: a modern approach* (Vol. 2). Upper Saddle River: Prentice hall.
[28] Seraji Homayoun, and Ayanna Howard. "Behavior-based robot navigation on challenging terrain: A fuzzy logic approach." IEEE Transactions on Robotics and Automation 18.3 (2002): 308-321.

Authors' Profiles

Goran Zaharija was born in Split, Croatia on August 13, 1985. He received the B.Sc. and M.Sc. degrees from University of Split, Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture, in 2008 and 2010, respectively.

He is a Junior Researcher at the Department of Informatics at the Faculty of Science, University of Split. He teaches several undergraduate courses in programming and computer architecture. He is currently working on his PhD research in the field of Artificial Intelligence. His other scientific interests include machine learning, artificial neural networks and multi-agent systems.

Saša Mladenović was born in Split, Croatia. He received his PhD in computer science at the Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture in Split in 2010. He is an Assistant Professor at the Department of Informatics at the Faculty of Science, University of Split

From 1999 to 2006 he was acting as Technical manager of the Toll collection system department of Ecsat, Croatia – the company responsible for software development at the Transportation department of CS group Designer, integrator and operator of mission-critical systems, France. His research interests include: problems in teaching programming, interoperability, intelligent technologies like ontology and multi-agent systems, especially engineering applications of intelligent technologies. He is IEEE member since 1996.

Stefan Dunić was born in Croatia. He received the B.Sc. degree from University of Split, Faculty of Science, in 2013 and is working on his Master's thesis.

He is a student at the Faculty of Science, University of Split. He also works as a Sales Specialist at Vipnet – Telecommunications company in Croatia, where he also worked as a Business Sales Assistant. His research interests include software development, data analysis, web development, databases and learning methods.

How to cite this paper: Goran Zaharija, Saša Mladenović, Stefan Dunić, "Cognitive Agents and Learning Problems", International Journal of Intelligent Systems and Applications (IJISA), Vol.9, No.3, pp.1-7, 2017. DOI: 10.5815/ijisa.2017.03.01

