Survival of the mutable: architecture of adaptive reactive agents

Renata Luiza Stange[1,2], Paulo Roberto Massa Cereda[2], and João José Neto[2]

[1] Universidade Tecnológica Federal do Paraná
Av. Prof[a]. Laura Pacheco Bastos, 800, Industrial, Guarapuava, PR – Brasil
[2] Escola Politécnica, Departamento de Engenharia de
Computação e Sistemas Digitais, Universidade de São Paulo
Av. Prof. Luciano Gualberto, s/n, Travessa 3, 158, São Paulo, SP – Brasil
rlstange@usp.br, paulo.cereda@usp.br, jjneto@usp.br

Abstract. An agent is defined as any device that perceives a certain environment through stimuli and acts upon it as to achieve a certain goal. There is a plethora of theories, architectures and languages in the literature aiming at how much an agent may be improved at performing a task. However, the majority of them focuses on the internal agent function itself instead of adopting a macroscopic, broader view of what the term "intelligent" means in the long run. In this paper we take a bio-inspired route and describe how the simplest reactive agent can be boosted towards improvements at performing complex tasks by making it mutable. We provide a mathematical framework to support such features. Conceptually, the addition of a mutability layer does not break the existing paradigms and allows hybrid approaches as a means to achieve better results.

1 Introduction

Like other concepts of computation, specially the ones targeted in different areas, the definition of what an agent is does not have universally accepted consensus amongst the scientific community [11,1]. We mainly focus on Artificial Intelligence and therefore adopt both definition and classification of agents as seen in [25]. Russell and Norvig [25] define an agent simply as "something that acts". According to this definition, it is possible to glimpse a computer program as an agent. However, there are scenarios in which a computational agent is expected to be rational and operate autonomously, perceiving its environment and eventually being susceptible to change [11].

Change, in this context, is a disputable concept when discussing behavioral actions [8,20]. In general, there is no clear indication of how such operation affects the agent behavior for subsequent interactions with the environment besides of a narrowed, error-proned local inspection [20]. A single misplaced action might mutilate the agent function and permanently compromise the entire task resolution. The more safety guards added to detect and avoid error propagation, the more complex and unfeasible the agent will be in terms of maintenance and

debugging [9]. Change might convey the idea of arbitrary actions taken from a confined, limited perspective.

The phrase "survival of the fittest" was coined by Herbert Spencer [27] as a way of describing the mechanism of natural selection, in which the biological concept of fitness is defined as reproductive success. Similarly, we pinpoint mutability as the key element for a bio-inspired agent model, such that the change concept is, from now on, bound to evolving the entire entity and not only its corresponding internal function [18].

Credited to theoretical physicist Stephen Hawking, the famous quote "intelligence is the ability to adapt to change" points us towards a better, stricter term to denote successive evolutions in an intelligent agent model: adaptation. However, it does not suffice being adaptable; as an agent inevitably has to operate autonomously, model adaptations must be triggered from within as opposed to explicit external calls [28]. Hence, the agent must be adaptive, in which the model spontaneously evolves without external interference, based solely on perceptions, stimuli and history. Biological entities are heavily based on such concept. In [3], Beer argues that the Artificial Intelligence area has overlooked the importance of adaptive behavior as a crucial substrate for intelligent behavior. Since the adaptive model is expected to occasionally evolve, we chose the simplest, most straightforward agent type from Russell and Norvig's classification [25] as a proof of concept to demonstrate that the complexity of an agent may be greatly reduced while retaining intelligent behavior. In this paper, we aim at providing an architecture for such agents.

As to provide a consistent, coherent architecture for adaptive reactive agents, we rely on a mathematical formalism proposed by José Neto [13,12]. This particular approach has the clear advantage of providing a macroscopic model view, omitting superfluous details and characteristics [6]. Similar formalisms, such as cellular automata and recursive adaptable grammars, are interesting alternatives for designing bio-inspired models and may be employed as well.

2 Mathematical background

This section formally introduces the mathematical formalism proposed by José Neto [13]. Observe that the theory relies on an adaptive mechanism enclosing a non-adaptive rule-driven device, such that the latter may be enhanced in order to accommodate an adaptive behavior while preserving its integrity and original properties [12].

Definition 1 (rule-driven device). *A rule-driven device is defined as* $ND = (C, NR, S, c_0, A, NA)$*, such that ND is a rule-driven device,* C *is the set of all possible configurations,* $c_0 \in C$ *is the initial configuration,* S *is the set of all possible input stimuli,* $\epsilon \in S$*,* $A \subseteq C$ *is the subset of all accepting configurations (respectively,* $F = C - A$ *is the subset of all rejecting configurations), NA is the set of all possible output stimuli of ND as a side effect of rule applications,* $\epsilon \in NA$*, and NR is the set of rules defining ND as a relation* $NR \subseteq C \times S \times C \times NA$*.*

Definition 2 (rule). *A rule $r \in NR$ is defined as $r = (c_i, s, c_j, z)$, $c_i, c_j \in C$, $s \in S$ and $z \in NA$, indicating that, as response to a stimulus s, r changes the current configuration c_i to c_j, processes s and generates z as output [13]. A rule $r = (c_i, s, c_j, z)$ is said to be compatible with the current configuration c if and only if $c_i = c$ and s is either empty or equals the current input stimulus; in this case, the application of a rule r moves the device to a configuration c_j, denoted by $c_i \Rightarrow^s c_j$, and adds z to the output stream.*

Definition 3 (acceptance of an input stimuli stream by a rule-driven device). *An input stimuli stream $w = w_1 w_2 \dots w_n$, $w_k \in S - \{\epsilon\}$, $k = 1, \dots, n$, $n \geq 0$, is accepted by a device ND when $c_0 \Rightarrow^{w_1} c_1 \Rightarrow^{w_2} \dots \Rightarrow^{w_n} c$ (in short, $c_0 \Rightarrow^w c$), and $c \in A$. Respectively, w is rejected by ND when $c \in F$. The language described by a rule-driven device ND is represented by $L(ND) = \{w \in S^* \mid c_0 \Rightarrow^w c, c \in A\}$.*

Definition 4 (adaptive rule-driven device). *A rule-driven device $AD = (ND_0, AM)$, such that ND_0 is a device and AM is an adaptive mechanism, is said to be adaptive when, for all operation steps $k \geq 0$ (k is the value of an internal counter T starting in zero and incremented by one each time a non-null adaptive action is executed), AD follows the behavior of an underlying device ND_k until the start of an operation step $k + 1$ triggered by a non-null adaptive action, modifying the current rule set; in short, the execution of a non-null adaptive action in an operation step $k \geq 0$ makes the adaptive device AD evolve from an underlying device ND_k to ND_{k+1}.*

Definition 5 (operation of an adaptive device). *An adaptive device AD starts its operation in configuration c_0, with the initial format defined as $AD_0 = (C_0, AR_0, S, c_0, A, NA, BA, AA)$. In step k, an input stimulus move AD to the next configuration and starts the operation step $k + 1$ if and only if a non-adaptive rule is executed; thus, being the device AD in step k, with $AD_k = (C_k, AR_k, S, c_k, A, NA, BA, AA)$, the execution of a non-null adaptive action leads to $AD_{k+1} = (C_{k+1}, AR_{k+1}, S, c_{k+1}, A, NA, BA, AA)$, in which $AD = (ND_0, AM)$ is an adaptive device with a starting underlying device ND_0 and an adaptive mechanism AM, ND_k is an underlying device of AD in an operation step k, NR_k is the set of non-adaptive rules of ND_k, C_k is the set of all possible configurations for ND in an operation step k, $c_k \in C_k$ is the starting configuration in an operation step k, S is the set of all possible input stimuli of AD, $A \subseteq C$ is the subset of accepting configurations (respectively, $F = C - A$ is the subset of rejecting configurations), BA and AA are sets of adaptive actions (both containing the null action, $\epsilon \in BA \cap AA$), NA, with $\epsilon \in NA$, is the set of all output stimuli of AD as side effect of rule applications, AR_k is the set of adaptive rules defined as a relation $AR_k \subseteq BA \times C \times S \times C \times NA \times AA$, with AR_0 defining the starting behavior of AD, AR is the set of all possible adaptive rules for AD, NR is the set of all possible underlying non-adaptive rules of AD, and AM is an adaptive mechanism, $AM \subseteq BA \times NR \times AA$, to be applied in an operation step k for each rule in $NR_k \subseteq NR$.*

Definition 6 (adaptive rules). *Adaptive rules* $ar \in AR_k$ *are defined as* $ar = (ba, c_i, s, c_j, z, aa)$ *indicating that, as response to an input stimulus* $s \in S$*, ar initially executes the prior adaptive action* $ba \in BA$*; the execution of ba is canceled if this action removes ar from* AR_k*; otherwise, the underlying non-adaptive rule* $nr = (c_i, s, c_j, z)$*,* $nr \in NR_k$ *is applied and, finally, the post adaptive action* $aa \in AA$ *is applied [13].*

Definition 7 (adaptive function). *Adaptive actions may be defined as abstractions named adaptive functions, similar to function calls in programming languages [13]. The specification of an adaptive function must include the following elements: (a) a symbolic name, (b) formal parameters which will refer to values supplied as arguments, (c) variables which will hold values of applications of elementary adaptive actions of inspection, (d) generators that refer to new value references on each usage, and (e) the body of the function itself.*

Definition 8 (elementary adaptive actions). *Three types of elementary actions are defined in order to perform tests on the rule set or modify existing rules, namely:*

(i) inspection: the elementary action does not modify the current rule set, but allows inspection on such set and querying rules that match a certain pattern. It employs the form $?\langle pattern \rangle$*.*
(ii) removal: the elementary action removes rules that match a certain pattern from the current rule set. It employs the form $-\langle pattern \rangle$*. If no rule matches the pattern, nothing is done.*
(iii) insertion: the elementary action adds a rule that match a certain pattern to the rule set. It employs the form $+\langle pattern \rangle$*. If the rule already exists in the rule set, nothing is done.*

Such elementary adaptive actions may be used in the body of an adaptive function, including rule patterns that use formal parameters, variables and generators available in the function scope.

According to Cereda and José Neto [6,5], adaptive devices offer conveniences of a more compact model representation and better organization, as each underlying device covers a specific context at a time.

3 Adaptive reactive agents

An agent is defined as a device that perceives its environment through sensors and acts on this environment through actuators [25]. Perception refers to the perceptual stimuli of the agent and a sequence of perceptions consists of the history of all perceptions so far. The choice of action may depend only on the actual perception or sequence of perceptions obtained up to a given instant in time [25]. Similarly, an adaptive reactive agent also perceives its environment through sensors and acts in the environment through actuators, according to a

function f that determines its internal behavior; however, the rule set of f may spontaneously adapt over time, in response to operating history and perceptions, such as a conventional biological entity. Function f maps a set of perceptions into an action [25], such that $f\colon 2^P \mapsto AC$, where P is the set of perceptions and AC is the set of actions. The rule set adaptation occurs through the adaptive mechanism, triggered by the execution of adaptive rules [28].

An ordinary reactive agent acts upon the current perception, ignoring history. Such approach has the clear advantage of providing an intelligible, modular and efficient representation through $\langle condition \mapsto action \rangle$ rules [25]. However, simplicity comes at the cost of having no explicit knowledge representation, besides the absence of symbolic internal environmental representation and action memory [28].

Definition 9 (reactive agent). *A reactive agent is defined as* $RA = (Q, P, AC, \Gamma, \delta)$*, such that* Q *is the set of internal states,* P *is the set of perceptions,* AC *is the set of actions,* Γ *is the mapping,* $\Gamma\colon Q \times 2^P \mapsto Q$*, and* δ *is a function that maps rules into actions,* $\delta\colon P \mapsto AC$*.*

Reactive agents have a limiting characteristic: decision-making acts only on the current perception, implying the environment must be fully observable, otherwise the agent will potentially fail [25]. The inclusion of an adaptive layer in a reactive agent allows handling predicted yet unexpected situations in the current context. Adaptive actions evolve the agent through time and accommodate new contexts from environment observations [6,7].

Definition 10 (adaptive reactive agent). *An adaptive reactive agent is defined as* $AA = (A_0, AM)$*, such that* A_0 *is a reactive agent and* AM *is an adaptive mechanism. For all operation steps* $k \geq 0$ *(*k *is the value of a built-in counter* T *started at zero and incremented by one unit every time a non-null adaptive action is executed),* AA *follows the behavior determined by the underlying reactive agent* A_k *until a non-null adaptive action is executed, which starts the* $k+1$ *operation step through adapting the rule set of the agent function.*

The architecture of an adaptive reactive agent presented in Figure 1 is heavily inspired by Russell and Norvig's model [25]. For convenience, we included an additional component containing learning modules in order to reduce complexity and verbosity. However, it is important to observe that the model in itself is powerful enough to handle any sort of computational task; the introduction of a new component is merely for convenience and expressiveness purposes [24].

According to Figure 1, the adaptive mechanism is represented as an adaptive action handler. Such handler intercepts requests from the underlying agent function and adapts the rule set according to patterns established in its action set and optionally in the learning modules. The underlying reactive agent evolves over time based on adaptations on its rule set, triggered by decision-making actions in the associated handler. Note that the adaptive layer enclosing the agent retains the original reactive property; however, the agent itself acts by context during its life cycle, one evolutionary reactive step at a time.

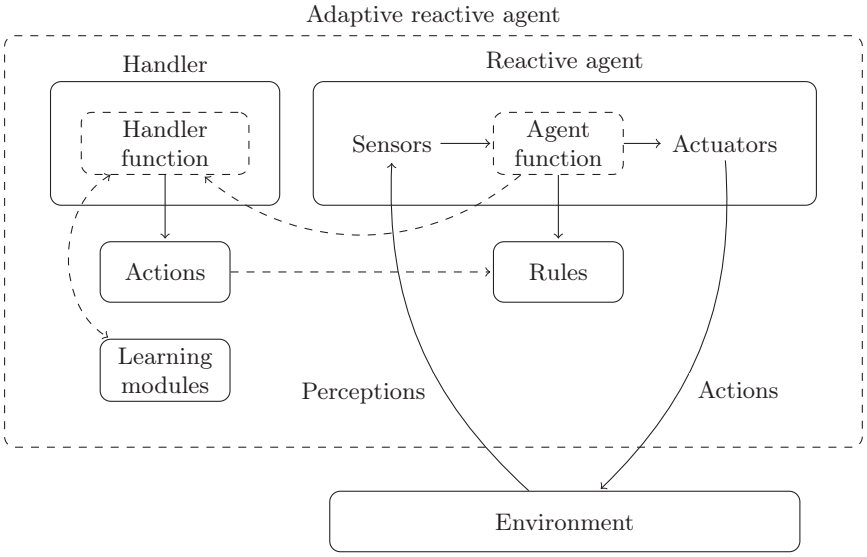


Fig. 1. Architecture of an adaptive reactive agent, heavily inspired by Russell and Norvig's model [25].

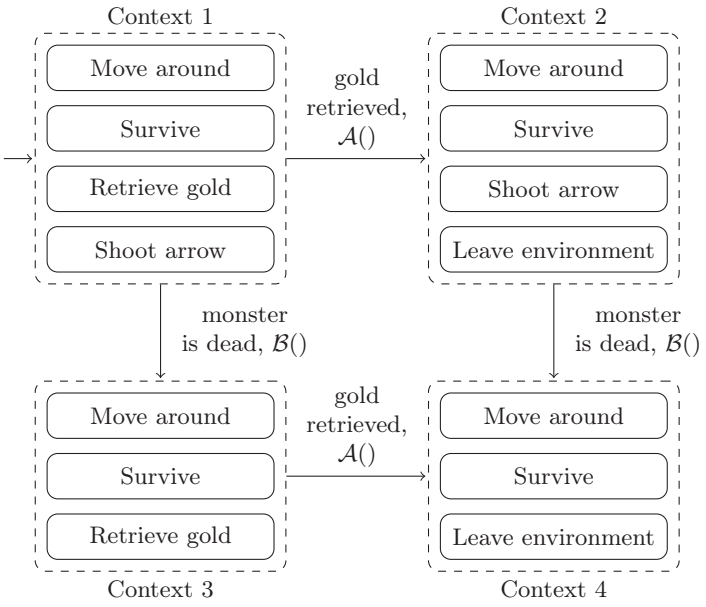
The first module employs associative learning, such that the agent is able to learn, memorize and reconstruct a pattern by associating the various parts of this pattern with one another [19,29]. This process involves the creation of a cognitive map. From a biological view, a cognitive map is the way certain beings memorize the information they gain about the spatial organization of their environment and how they make use of such information to navigate from one point to another [19]. Cognitive maps are usually represented as graphs with nodes that are computational elements. From this perspective, a cognitive map may be enhanced with higher level adaptation as well [26].

The second module employs reinforcement learning, such that the agent is able to recognize and favor certain behaviors that yield rewards rather than those that yield punishments [19,4]. Ethologists classify this characteristic as motivational system. The reinforcement process involves the production of goal seeking sequences through $\langle object \mapsto goal \rangle$ associations [4,10]. Booker [4] mentions the response (of his models) to regularities hidden behind the equivocal nature of sensory cues and use of information afforded by the environment with respect to goals.

New contexts provide incremental resolution by dividing a task into smaller subtasks [17,2]. Solving a task in a simpler abstraction space may be used to solve similar tasks at higher abstraction levels [15]. The process is then repeated until the original task is solved in its original abstraction space [23]. Such resolution strategy may produce significant search space reductions [21,22,16].

4 Experiments and discussions

In order to evaluate the concept of a reactive adaptive agent, we wrote a case study using the traditional *wumpus world* problem, a variant of a computer game [30] presented by Genesereth [25] as a testing environment for artificial intelligence techniques. The wumpus world consists of a two-dimensional grid containing a number of holes, a monster and an agent. The agent begins its iteration in grid position $(1, 1)$ and its task is set to avoid the monster and the holes, find the gold and leave the environment by reaching the same position it initiated. The agent can perceive a breeze in positions adjacent to the holes, a stench in positions adjacent to the monster, and a glow in positions adjacent to the gold. Variations of the wumpus world are available, in which the agent is armed with one or more arrows, being able to shoot the monster when found [14]. Figure 2 illustrates the possible contexts handled by the reactive adaptive agent, according to the execution of the corresponding adaptive actions.



adaptive action $\mathcal{A}()$
$-(r \in R \mid \phi(r) = \text{retrieve gold})$
$+(r \notin R \mid \phi(r) = \text{leave env.}\ldots)$
end

adaptive action $\mathcal{B}()$
$-(r \in R \mid \phi(r) = \text{shoot arrow})$
end

Fig. 2. Contexts handled by the reactive adaptive agent for the wumpus world problem.

The agent interaction in the wumpus world was simulated using an agent specification and simulation software developed by Jill Zimmerman[3]. We generated 14 random environments of size 4×4 with obstacles, gold and one monster. The simulation software measured the agent interaction with the environment and calculated a final score based on its overall performance [25]. We simulated the interaction of four types of agents – reactive, model-based, goal-based and adaptive reactive – in the generated environments and evaluated their global performances. We also introduced a second adaptive reactive agent enhanced with learning modules. The simulation was repeated 10 times for each ordered pair $\langle agent, environment \rangle$, totaling 700 executions. Results are shown in Figure 3.

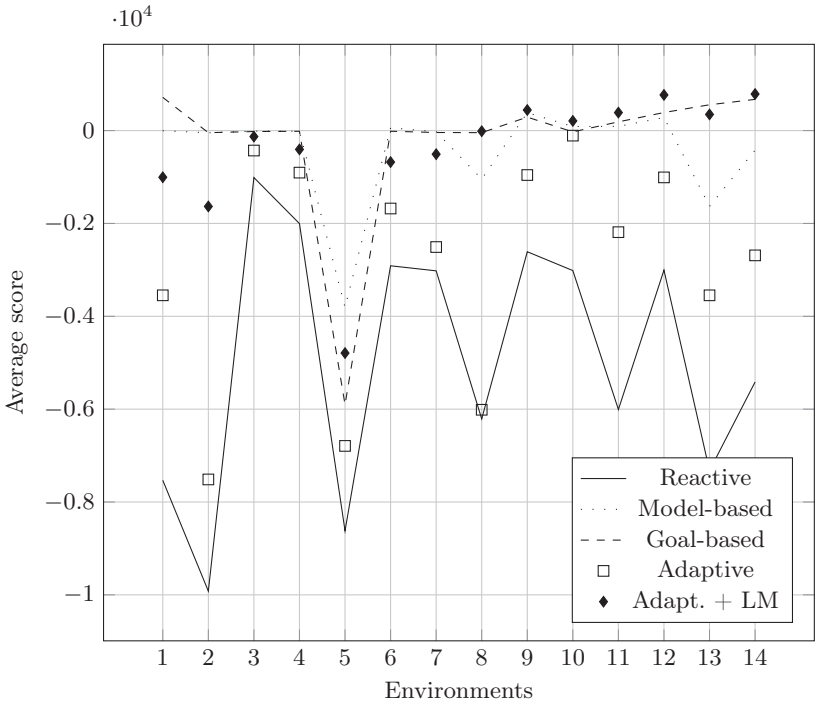


Fig. 3. Average score of 5 agents in 14 wumpus world environments.

According to Figure 3, agents had a similar overall execution for the same environments. In particular, the ones based on model and goals had significantly close performances. The reactive adaptive agent demonstrated a relatively better performance when compared to the ordinary reactive agent and, in some environments, had scores close to those based on model and goals. Most notably, when enhanced with learning modules, the adaptive reactive agent had a sig-

[3] Available at `http://phoenix.goucher.edu/~jillz/cs340/`.

nificant performance boost, even surpassing the classical approaches on certain environments.

5 Final remarks

In this paper, we presented a mutable approach to the classic reactive agent through a bio-inspired model, while preserving its original properties. The key element towards intelligence is adaptation. New contexts bring practical benefits such as division of work, abstraction, reuse and maintenance. From a biological point of view, conservation of energy is achieved through a compact cognitive representation, since the current context holds no superfluous knowledge.

Through mutability, the organization of rules by context grants simplification of the original task, dividing it into smaller subtasks. Additionally, execution becomes more efficient in the long run since a context will only be activated if it is actually needed. Existing techniques may be combined [6,28] in order to specifically address each subtask.

Adaptivity is an interesting approach towards a natural, bio-inspired design of agents, such that the underlying behavior is flexibly adjusted to contingencies that arises in its interaction with an unknown environment. Mutability is a crucial feature for survival, and the understanding of adaptive behavior in biological beings may yield interesting results in the computational domain.

References

1. Ahmad, A., Ahmad, M.S., Yusof, M.Z.: An exploratory review of software agents. International Symposium on Information Technology (2008)
2. Anzai, Y., Simon, H.A.: The theory of learning by doing. Psychological Review (86), 124–140 (1979)
3. Beer, R.D.: Intelligence as Adaptive Behavior: an Experiment in Computational Neuroethology. Academic Press Inc, Boston (1990)
4. Booker, L.B.: Instinct as an inductive bias for learning behavioral sequences. In: From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior. The MIT Press (1991)
5. Cereda, P.R.M., José Neto, J.: Towards performance-focused implementations of adaptive devices. Procedia Computer Science 109, 1164–1169 (2017)
6. Cereda, P.R.M., José Neto, J.: Adaptive data mining: Preliminary studies. IEEE Latin America Transactions 12(7), 1258–1270 (October 2014)
7. Cereda, P.R.M., José Neto, J.: Utilizando linguagens de programação orientadas a objetos para codificar programas adaptativos. In: Memórias do IX Workshop de Tecnologia Adaptativa – WTA 2015. pp. 2–9 (2015)
8. Chiel, H.J., Beer, R.D.: Simulation of adaptive behavior. Current Opinion in Neurobiology (1), 605–609 (1991)
9. Grillner, S., Buchanan, J.T., Lansner, A.: Simulation of the segmental burst generating network for locomotion in lamprey. Neuroscience Letters (89), 31–35 (1988)
10. Holland, J., Holyoak, K., Nisbett, R., Thagard, P.: Induction: Processes of Inference, Learning and Discovery. The MIT Press (1986)

11. Jennings, N.R., Sycara, K., Wooldridge, M.: A roadmap of agent research and development. Autonomous Agents and Multi-Agent Systems 1, 7–38 (1998)
12. José Neto, J.: Adaptive automata for context-dependent languages. SIGPLAN Notices 29(9), 115–124 (1994)
13. José Neto, J.: Adaptive rule-driven devices: general formulation and case study. In: International Conference on Implementation and Application of Automata (2001)
14. Khan, G.M., Miller, J.F., Halliday, D.M.: Advances in modeling adaptive and cognitive systems, chap. Intelligent agents capable of developing memory of their environment, pp. 77–114. UEFS (2010)
15. Knoblock, C.: Learning abstraction hierarchies for problem solving. In: Proceedings of the Eighth National Conference on Artificial Intelligence. pp. 923–928 (1990)
16. Knoblock, C.: Search reduction in hierarchical problem solving. In: AAAI'91 Proceedings. pp. 686–691 (1991)
17. Knoblock, C.: Generating Abstraction Hierarchies: An Automated Approach to Reducing Search in Planning. Springer US (1993)
18. Kristan, W.B., McGirr, S.J., Simpson, G.V.: Behavioural and mechanosensory neurone responses to skin stimulation in leeches. Journal of Experimental Biology (96), 143–160 (1982)
19. Matarić, M.: Navigating with a rat brain: A neurobiologically-inspired model for robot spatial representation. In: From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior. The MIT Press (1991)
20. Meyer, J.A., Guillot, A.: Simulation of adaptive behavior in animals: review and prospect. In: From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior. MIT Press (1991)
21. Minsky, M.: Computers and Thought, chap. Steps toward artificial intelligence, pp. 406–450. McGraw-Hill, New York, NY (1963)
22. Newell, A., Simon, H.A.: Contemporary Approach to Creative Thinking, chap. The processes of creative thinking, pp. 63–119. Atherton Press (1962)
23. Pólya, G.: How to Solve It. Princeton University Press (1945)
24. Rocha, R.L.A., José Neto, J.: Autômato adaptativo, limites e complexidade em comparação com a Máquina de Turing. In: Proceedings of the Second Congress of Logic Applied to Technology. pp. 33–48 (2000)
25. Russell, S.J., Norvig, P.: Artificial Intelligence: A Modern Approach. Pearson Education, 2 edn. (2003)
26. Silva Filho, R.I., Rocha, R.L.A.: Adaptive and Natural Computing Algorithms: 10th International Conference, ICANNGA 2011, Ljubljana, Slovenia, April 14-16, 2011, Proceedings, Part II, chap. Adaptive Finite Automaton: A New Algebraic Approach, pp. 275–284. Springer Berlin Heidelberg (2011)
27. Spencer, H.: The Principles of Biology, vol. 1. John Childs and Son, London (1864)
28. Stange, R.L., Cereda, P.R.M., José Neto, J.: Agentes adaptativos reativos: formalização e estudo de caso. In: Memórias do XI Workshop de Tecnologia Adaptativa – WTA 2017. pp. 63–71. São Paulo (2017)
29. Stange, R.L., José Neto, J.: Learning decision rules using adaptive technologies: a hybrid approach based on sequential covering. Procedia Computer Science 109, 1188–1193 (2017)
30. Yob, G.: Hunt the wumpus. People's Computer Company 2(1) (1973)