Adaptive Finite Automaton: A New Algebraic Approach

Reginaldo Inojosa Silva Filho and Ricardo Luis de Azevedo da Rocha

Computing Engineering Department
Engineering School of the University of São Paulo,
Av. Luciano Gualberto s/n Trav 3, n.158 São Paulo - SP, Brazil
reginaldo.uspoli@gmail.com, rlarocha@usp.br

Abstract. The purpose is to present a new and better representation for the adaptive finite automaton and to also show that both formulations – the original and the newly created – have the same computational power. Adaptive finite automaton original formulation was explored and a way to overcome some difficulties found by [7] in its representation and proofs about its computational power were sought. Afterwards both formulations show to be equivalent in representation and in computational power, but the new one has a highly simplified algebraic notation. The use of the new formulation actually allows simpler theorem proofs and generalizations, as can be verified in the last section of the paper.

Keywords: Adaptivity, Automata Theory, Algebraic Formulation.

1 Introduction

The principal idea of automata theory led to the creation of more complex types of automata, with more general automaton behaviors [13]. In the classical formal languages theory, only automata models with invariable internal structure were considered. In the adaptive automata model, the highest computational power is achieved when this assumption is relaxed.

The term adaptive has a well-defined sense within the self-modifying computational models. In this field, the adaptive formalisms may be divided into two main categories: adaptive grammars [3],[1] and adaptive machines [11]. Adaptive automata belong to the second category and their major characteristic is ability, without the interference of any external agent, to decide to modify its own structure in response to some external input. There are many applications for an adaptive automata model [2],[5] and its computational power is Turing Machine equivalent [9]. Despite these facts, the adaptive automata model is not fully formalized and the purpose here is to present a new and better formalization for this machine model. For this, the paper is structured as follows. The second section presents the notations and theoretical preliminaries as well as a brief description of the classical (original) adaptive finite automaton model. The third section describes the new formulation and the principal equivalence results in relation to the classical formulation. The last section presents the conclusion and further work to be developed.

A. Dobnikar, U. Lotrič, and B. Šter (Eds.): ICANNGA 2011, Part II, LNCS 6594, pp. 275–284, 2011.

2 Notations and Technical Preliminaries

Several notations are required for the adequate discussion of the model presented. Let $\mathbb{N} = \{0, 1, 2, \dots\}$ be the set of natural numbers. For a finite arbitrary indexed set $I = \{i_0, i_1, \dots, i_m\}$ the two functions, *remove function* and the *expand function* are defined as

$$rem(I, x) = \{I - \{x\} : x \in I\} \tag{1}$$

$$insert(I, x) = \{i_0, i_1, \dots, i_{m+1}\} \tag{2}$$

with $(x \notin I)$ and $i_{m+1} = x$

If I is a typed set, the element inserted is of the same type as I. The Axiom of Choice [4] is assumed to hold. Thus, there is a choice function for every arbitrary st $\mathcal{U}$, in which a choice function is defined as $c(U) \in \mathcal{U}$ for $U \in \mathcal{U}$. A non-empty sequence φ is an ordered list $(a_k, \dots, a_{n-1}, a_n)$ of abstract objects with $[k, n] \subseteq \mathbb{N}$. The sequence is always nominated by the latest letters of the Greek alphabet. Let Σ be a non-empty fixed, but arbitrary, finite set of atomic symbols called alphabet. Any generic symbol of Σ is represented by the first letters of the Greek alphabet, for example, $\alpha \in \Sigma$. A string t over Σ is a finite concatenation of alphabet symbols in which the length of t is denoted by $|t| \in \mathbb{N}$. The *empty word*, denoted by ε, has its length equal to zero. The set of all possible strings over Σ is denoted Σ^*, while the set of all nonempty strings over Σ is denoted Σ^+. Any subset $L \subseteq \Sigma^*$ is called a *language* over Σ. A *non-deterministic finite state automaton without outputs* over Σ is defined by the quintuple $M = (Q, q_0, E, \Sigma, \partial)$, in which Q is a finite, non-empty set of states, $\partial \subseteq Q \times \{\Sigma \cup \{\varepsilon\}\} \times Q$ is the automaton state-transition partial function, which can be expressed by the set $\partial = \{\delta_1, \dots, \delta_n\}$, in which $\delta_i = (q', \alpha, q'')$ for $1 \leq i \leq n$, $\{q', q''\} \subseteq Q$ and $\alpha \in \Sigma \cup \{\epsilon\}$. The set $E \subseteq Q$ is the accepting states set, while q_0 is the initial state. A *scalar hierarchical structure*[12] is indicated by the notation $\langle a_n \langle a_{n-1} \dots \langle a_1 \langle a_0 \rangle \rangle \rangle \rangle$, in which a_i is part of a_{i+1} for $0 \leq i \leq (n-1)$.

2.1 Adaptive Automata

To initiate the discussion about the structural redefinition of the adaptive automata model, it is first necessary to summarize its classical formulation. In the Adaptive Automaton [7], there is a set of (optional) adaptive actions which change the behavior of a non-deterministic finite state automaton by modifying the set of rules defining it. This modification is performed by an adaptive function, which is executed by removing or inserting new elements to the automaton transition function. The adaptive mechanism of the adaptive automaton is defined by attaching a pair of adaptive functions, $\mathcal{B}$ (before) and $\mathcal{A}$ (after). This pair of adaptive functions is attached to the subjacent non-adaptive automaton transitions, one to be performed before the transition takes place, and another to be performed after executing the transition. This altered automaton transition is called adaptive transition and its notation is summarized below:

$$(q, \alpha) : \mathcal{B} \rightarrow q' : \mathcal{A} \tag{3}$$

in which $q \in Q; \alpha \in \Sigma \cup \{\epsilon\}$; $\mathcal{B}$ and $\mathcal{A}$ are the adaptive functions.

In the general case, adaptive functions $\mathcal{A}$ and $\mathcal{B}$ are symbolically declared apart from the finite state automaton, and they comprehend a header (the name and the formal parameters of the adaptive function) and a body (declaration of names and elementary adaptive actions), which have the general form:

$\eta(\Omega)$ {
  declaration of names (optional)
  declaration of elementary adaptive actions (optional)
}

in which : $\Omega = \{\varphi_1, \ldots, \varphi_n\}$ is the n-parameters set of arguments φ_i (for $1 \leq i \leq n$) passed by a call-by-value strategy to the adaptive function named η. The arguments may assumed to be any coherent value within the function body. The body part is formed by the declaration of names and a set of elementary adaptive actions, responsible for the modifications to be performed. Declaration of names is a list of elements chosen to represent objects in the scope of the function body. Each declaration of names assumes the following form: $g_1^*, g_2^*, \ldots, g_i^*; v_1, v_2, \ldots, v_j$ in which the names followed by an asterisk denote generators (new names to objects that do not belong to the underlying finite automaton), and the remaining names denote variables. Elementary adaptive actions specify the actual modifications to be imposed on the automaton.

Any elementary adaptive action potentially has local variables that are filled once by elementary inspection actions (defined below) and generators that are filled once with new values (different from all values or variables defined in the model) when the adaptive action starts. Values are assigned only once to variables by elementary adaptive inspection and elimination-type actions (explained below), then those objects become read-only. Generators receive unique values also once at the start of function execution and remain read-only. Elementary adaptive actions can be of three types:

1. $?[(q, \alpha) : \mathcal{B} \rightarrow q' : \mathcal{A}]$: Inspection-type actions (introduced by a question mark in usual notation) search the current set of transitions for those the shapes of which match the given pattern. The elements to be inspected are replaced by variable names. These variables must be unique, being filled by the inspection mechanism with the current value of the corresponding inspected unknown elements, and they become read-only thereafter.
2. $-[(q, \alpha) : \mathcal{B} \rightarrow q' : \mathcal{A}]$: Elimination-type adaptive actions (introduced by a minus sign in usual notation), which eliminate all transitions matching the given shape from the current set of transitions in the automaton.
3. $+[(q, \alpha) : \mathcal{B} \rightarrow q' : \mathcal{A}]$: Insertion-type adaptive actions (introduced by a plus sign in usual notation), which add a new transition to the set of current transitions, according to the specified shape. The adaptive mechanism turns a finite automaton into an adaptive one by allowing its set of rules to change dynamically.

The adaptive function is performed following the sequence below [7]: a) the only values available for variables are those that came from parameters in function

header, b) the generators are filled, c) the set of elementary adaptive actions of inspection is performed, d) the set of elementary adaptive actions of deletion is performed, e) the set of elementary adaptive actions of insertion is performed. In the classical formulation, there is no particular restriction to the use of multiple variables in inspecting and eliminating elementary adaptive actions [7]. Additionally, there is no restriction to the types that variables can take. Variables can assume the role of states or symbols in the transition pattern of an elementary adaptive action. Thus, the use of the variable concept is not clear.

Adaptive functions were defined informally, using a pseudocode notation to describe the adaptive functions behavior. This fact implies that the adaptive automata model, despite its computational power and innovative paradigm, has ambiguous concepts, such as the nature and number of parameters allowed for adaptive functions and the use of variables [8]. As can be seen in the next section, in order to deal with these problems, it is necessary to formalize those definitions.

3 The Adaptive Automaton New Formulation

The task of rewriting the model of the adaptive automaton goes beyond finding the solutions for the problems mentioned above. First, as the name suggests, the use of the algebraic theoretical approach can be viewed as an attempt to understand certain properties (such as types of possible modifications, function composition, etc.) of the automaton transformations [6]. In this context, the concept of adaptive automata was inserted in a more general concept of adaptive algebra. Second, in turn, an algebraic extension of the automaton concept implies a new expression for it. Hence, the traditional elements of the automata theory (automata configuration, step function, etc.) were brought into this reformulation.

The following concepts start this idea and provide the basis for the embedding of the adaptive automata.

Definition 1 ($\mathscr{M}^0$- algebra). *Given the set $\mathscr{M}^0$ of all non-deterministic finite state automaton under an alphabet Σ, a $\mathscr{M}^0$- algebra ís defined as $\mathfrak{M} = (\mathscr{M}^0, F)$ in which $F = (f_1, f_2, ...f_n)$ for $f_i : \mathscr{M}^0 \longrightarrow \mathscr{M}^0$.*

Given a non-deterministic finite state automata $M^0 \in \mathscr{M}^0$ and its state-transition function ∂, any transition $\delta = (q', \alpha, q'')$ in which $\delta \in \partial$ is called a *proper transition.* When $\delta \notin \partial$, it is called a *foreign transition.* For any element $M^0 \in \mathscr{M}^0$, the functions defined below:

$$S(q_i, a, q_j, M^0) = \{(q_i, a, q_j) : (q_i, a, q_j) \in \partial\} \tag{4}$$

$$S(q_i, q_j, M^0) = \{(q_i, \alpha, q_j) : (q_i, \alpha, q_j) \in \partial\} \tag{5}$$

$$S(q_i, M^0) = \{(q_i, \alpha, q'') : (q_i, \alpha, q'') \in \partial\} \tag{6}$$

$$S(q_j, M^0) = \{(q', \alpha, q_j) : (q', \alpha, q_j) \in \partial\} \tag{7}$$

$$S(q_i, a, M^0) = \{(q_i, a, q'') : q_i, a, q'') \in \partial\} \tag{8}$$

$$S(q_j, a, M^0) = \{(q', a, q_j) : (q', a, q_j) \in \partial\} \tag{9}$$

$$S(a, M^0) = \{(q', a, q'') : (q', a, q'') \in \partial\} \tag{10}$$

are called proper-search operations. For any element $M^0 \in \mathscr{M}^0$, the function defined as:

$$N(\delta, M^0) = \begin{cases} \delta & \Leftrightarrow \ \delta \notin \partial \\ \varnothing & \Leftrightarrow \ \delta \in \partial \end{cases} \tag{11}$$

is the non-pertinence identification operation.

A sequence $\delta_{pro} = (\delta_{pro_1}, \ldots, \delta_{pro_m})$ of proper transitions for an automaton M^0 is called positive sequence. Any positive sequence of M^0 in which all elements are any of proper-search operations is called a positive pattern sequence and is designated by $\hat{\delta}_{pro}$. On the other hand, a transitions sequence $\delta_{for} = (\delta_{for_1}, \ldots, \delta_{for_n})$, of foreign transition for M^0 is called a negative sequence. Analogously to positive pattern sequence, the negative sequence which has all the elements as non-pertinence identification operations $N(\delta, M^0)$ is called a *negative pattern sequence* and is designated by $\hat{\delta}_{for}$. Given a negative and a positive pattern sequences for an element $M^0 \in \mathscr{M}^0$, the sequence $\phi = (\hat{\delta}_{for}, \hat{\delta}_{pro})$ is called a *transformation pair*.

3.1 Adaptive Algebra

Utilizing the proper transition and foreign transition concept, as well the pattern sequences and the definition of (1) and (2), for any element $M^0 \in \mathscr{M}^0$, the *δ-remove operation* and *δ-insertion operation* are defined, respectively, by the operators:

$$f^-_{\delta_{pro_k}} M^0 = f^-(\delta_{pro_k}, M^0) = (Q, q_0, E, \Sigma, rem(\partial, \delta_{pro_k})) \text{ with } \delta_{pro_k} \in \hat{\delta}_{pro} \tag{12}$$

$$f^+_{\delta_{for_k}} M^0 = f^+(\delta_{for_k}, M^0) = (insert(insert(Q, q'), q''), q_0, E, \Sigma, insert(\partial, \delta_{for_k})) \tag{13}$$

with $\delta_{for_k} \in \hat{\delta}_{for}$

Now, it is possible to introduce the concept of adaptive algebra.

Definition 2 (Adaptive Algebra). *Adaptive Algebra is the $\mathscr{M}^0$- algebra in which $F = (f^-, f^+)$.*

3.2 Adaptive Transformations

Let an element $M^0 \in \mathscr{M}^0$ and its transformation pair $\phi = (\hat{\delta}_{for}, \hat{\delta}_{pro})$ formed by positive pattern sequence $\hat{\delta}_{pro}$ and a negative pattern sequence $\hat{\delta}_{for}$, in which $\hat{\delta}_{pro} = (\delta_{pro_1}, \ldots, \delta_{pro_m})$ and $\hat{\delta}_{for} = (\delta_{for_1}, \ldots, \delta_{for_n})$. The *Adaptive Function* is defined as:

$$\mathbb{F}_\phi M^0 \triangleq \mathbb{F}(\phi, M^0) = F^-_{\hat{\delta}_{pro}} F^+_{\hat{\delta}_{for}} M^0 \tag{14}$$

in which

$$F^{-}_{\hat{\delta}_{pro}} M^0 \triangleq F^{-}(\hat{\delta}_{pro}, M^0) = (f^{-}_{\delta_{pro_m}} \circ f^{-}_{\delta_{pro_{m-1}}} \circ \ldots \circ f^{-}_{\delta_{pro_2}} \circ f^{-}_{\delta_{pro_1}}) M^0 \quad (15)$$

$$F^{+}_{\hat{\delta}_{for}} M^0 \triangleq F^{+}(\hat{\delta}_{for}, M^0) = (f^{+}_{\delta_{for_n}} \circ f^{+}_{\delta_{for_{n-1}}} \circ \ldots \circ f^{+}_{\delta_{for_{(2)}}} \circ f^{+}_{\delta_{for_1}}) M^0 \quad (16)$$

are the *remove transformation* and *insertion transformation*, respectively.

The adaptive algebra operations provide the basis for building more complex operators. These operators will be used in the redefinition of adaptive automata. There are some assumptions to be considered:

- For a transition $\delta = (q', \alpha, q'')$, it is possible to have $q' \in Q$ and/or $q'' \in Q$ for $Q \in M^0$ and still keeping the condition $\delta = (q', \alpha, q'') \notin \partial$ for $\partial \in M^0$.
- $\mathbb{F}_{\phi} M^0 = F^{+}_{\hat{\delta}_{for}} M^0$ when $\hat{\delta}_{pro} \in \phi$ is an empty sequence.
- $\mathbb{F}_{\phi} M^0 = F^{-}_{\hat{\delta}_{pro}} M^0$ when $\hat{\delta}_{for} \in \phi$ is an empty sequence.
- In the particular situation in which both sequences, $\hat{\delta}_{pro}$ and $\hat{\delta}_{for}$ are empty, the first-order transformation pair is called *void* and is represented by $\phi^{\emptyset}$. In this case, $\mathbb{F}_{\phi^{\emptyset}} M^0 = M^0$.

3.3 Adaptive Automata

The new formulation for the adaptive automata model is more than a simple algebraic reinterpretation of the old definition. The purpose of this section is to show the most important elements of the automata theory within an adaptive model.

Definition 3 (Adaptive Automata). *A first-order adaptive automata is the quadruple* $M^1 = (M^0, \Phi, \phi^{\emptyset}, \partial^1)$*, in which* $M^0 \in \mathscr{M}^0$ *is called subjacent device. Set* Φ *of the transformation pairs is called adaptive behavior set. The element* $\phi^{\emptyset} \in \Phi$ *is a void transformation pair called null behavior. Set* ∂^1 *is the adaptive transition function. Each element of* ∂^1 *takes the form* $\delta^1_{i,k} = (\delta_i, \langle M^1 \langle \mathbb{F}_{\phi_k}(M^0) \rangle \rangle)$*, for* $\phi_k \in \Phi$ *and* $\delta_i \in \partial$ *in which* ∂ *is the subjacent device state-transition function.*

The *configuration* of a first-order adaptive automaton M^1 is the duple $(q', t) \in Q \times \Sigma^*$ in which Q belongs to a subjacent device of M^1. The *initial configuration* is represented by (q_0, t) in which q_0 is the initial state of the subjacent device before the execution of any adaptive transition of M^1.

For the configuration $(q', t) \in Q \times \Sigma^*$, the *one step function* shows how the adaptive automata changes from one configuration to another:

$$(q', t) \vdash_{[\mathbb{F}_{\phi_k} M^0]} (q'', w) \Leftrightarrow \exists \alpha \in \Sigma : \alpha w = t \quad (17)$$

in which q'' is a state of $\mathbb{F}_{\phi_k} M^0$ and $((q', \alpha, q''), \langle M^1 \langle \mathbb{F}_{\phi_k}(M^0) \rangle \rangle) \in \partial^1$ for $\phi_k \in \Phi$.

For a $j \in \mathbb{N}$, the closure of the one move function for an adaptive automaton is defined as:

$$(q', t) \vdash^{*}_{[\mathbb{F}_{\phi_{k_j}} \ldots \mathbb{F}_{\phi_{k_2}} \mathbb{F}_{\phi_{k_1}} M^0]} (q'', w) \quad (18)$$

iff $(q' = q'')$ and $(w = t)$ or

1. $t = a_0 a_1 \ldots a_j w$ with $a_i \in \Sigma$ for $0 \leq i \leq j$
2. $\exists(\phi_{k_1}, \phi_{k_2}, \ldots \phi_{k_{j+1}})$ with $\phi_{k_i} \in \Phi$ for $0 \leq i \leq j$
3. $\exists\ p_1, p_2 \ldots p_j \in Q$ such that:

$$
\begin{aligned}
(q', t) &\vdash_{[\mathbb{F}_{\phi_{k_1}}(M^0)]} (p_1, a_1 a_2 a_3 \ldots a_j w) \\
&\vdash_{[\mathbb{F}_{\phi_{k_2}} \mathbb{F}_{\phi_{k_1}}(M^0)]} (p_2, a_2 a_3 \ldots a_j w) \vdash_{[\mathbb{F}_{\phi_{k_3}} \mathbb{F}_{\phi_{k_2}} \mathbb{F}_{\phi_{k_1}}(M^0)]} \cdots \\
&\vdash_{[\mathbb{F}_{\phi_{k_j}} \ldots \mathbb{F}_{\phi_{k_2}} \mathbb{F}_{\phi_{k_1}}(M^0)]} (p_j, a_j w) \\
&\vdash_{[\mathbb{F}_{\phi_{k_{j+1}}} \mathbb{F}_{\phi_{k_j}} \ldots \mathbb{F}_{\phi_{k_2}} \mathbb{F}_{\phi_{k_1}}(M^0]} (q'', w)
\end{aligned}
$$

4 New Formulation Equivalence

Finally, the equivalence between the two presentations (the classic and the algebraic) will be proved. Proposition 1 from [10] shows that any adaptive transition that uses adaptive actions before and after the underlying device transition can be replaced by two consecutive transitions using only before (or after) adaptive actions. Thus, each transition has a unique adaptive function and simplifies the notation. The Lemma below states this result.

Lemma 1. *The adaptive actions may be placed in order to change an automaton* (*a*) *before the rule without using after-rule adaptive actions, or* (*b*) *after the rule (transition) without using before-rule adaptive actions.*

Lemma 1, in conjunction with Lemmas 2 and 3, shows that any transition of an adaptive finite-state automaton constructed using the original (classical) way [6] can be mapped to a transition using the new representation.

Lemma 2. *Any finite state automaton modification performed by the classical elementary adaptive actions (inspection, elimination or insertion) can, alternatively, be performed by proper-search operations, δ-remove operation or δ-insertion operation, respectively.*

Proof (SKETCH). The proof of Lemma 2 is not difficult but lengthy. For space reasons, only the proof sketch will be shown here. All inspection actions can be replaced with proper-search functions that, by definition, perform exactly the same type of search in automaton structure. The equivalence of the elimination action with the δ-remove operation is immediate from definitions, but the equivalence between insertion action and δ-insertion operation is more complex: if the insertion action actually inserts an internal transition (without an adaptive action) then the equivalence is immediate from the definitions, too; however, if the added transition is an adaptive one, the insertion action needs two δ-insertion operations for an equivalent modification effect. Without loss of generality, consider that all the classical adaptive actions must be defined *a priori*, then, in the new model, they must also be available as empty transitions disconnected from the initial model. Then two δ-insertion operations connect the adaptive action to the proper added transition. Thus, the order of execution of the new automaton transition and the adaptive action are preserved by the adaptive automaton transition function. □

Lemma 3. *Each adaptive action A_i, defined using the original formulation, may be represented by the (new) adaptive function $\mathbb{F}_{\phi_i} M^0$.*

Proof (by the use of Lemmas 1 and 2).

By Lemma 1, it is possible to transform any adaptive finite automaton into another one, which makes use of adaptive actions only after the rule has occurred. It is clear from the definitions that any adaptive action A_i is composed by lists of elementary adaptive actions.

Let A_i be an adaptive action described by the three sets of elementary adaptive actions (inspection elementary actions, deletion elementary actions, and insertion elementary actions sets). Using Lemma 2, all the elementary actions can be mapped to the proper-search operations, δ-remove operation and δ-insertion operation, respectively, of the new formulation. Thus, there exists an adaptive function $\mathbb{F}_{\phi_i} M^0$ that, using Lemma 2, maps the same tasks performed by the classical adaptive action A_i.

Hence, the (new) adaptive function $\mathbb{F}_{\phi_i} M^0$ is computationally equivalent to the adaptive action A_i. □

Now it is still to be proven that both representational formalisms, the original one and this newer one are, in fact, equivalent and, thus, the new formalism can replace the original (traditional) one without causing any harm to the theory.

Theorem 1. *Any adaptive finite-state automaton described in the original form can be replaced by the new form without destroying its properties.*

Proof (by induction on the number of adaptive transitions).

Base of the induction: By Lemma 3, for the classical adaptive transition $(q', \alpha) \rightarrow q'' : \mathcal{A}_i$, there is an adaptive function $\mathbb{F}_{\phi_i}$ such that $(q', \alpha t) \vdash_{[\mathbb{F}_{\phi_i} M^0]} (q'', t)$ and action $\mathcal{A}$ are mapped directly.

Inductive hypothesis: Suppose that there is a sequence of n transitions, with $0 \leq n \leq \ell$:

$$\begin{aligned} &(q^k, \alpha^k) \rightarrow q^{k+1} : \mathcal{A}_j \\ &\ldots \\ &(q^{k+n}, \alpha^{k+n}) \rightarrow q^{k+(n+1)} : \mathcal{A}_{j+n} \end{aligned}$$

in the classical formulation, then there is a sequence $\mathbb{F}_{\phi_{j+n}} \ldots \mathbb{F}_{\phi_j} M^0$ of the new formulation adaptive functions such that each $\mathbb{F}_{\phi_i}$, for $1 \leq i \leq n$, is equivalent to $\mathcal{A}_i$ and $(q, \alpha^k \ldots \alpha^{(k+n)} t) \vdash^*_{[\mathbb{F}_{\phi_{j+n}} \ldots \mathbb{F}_{\phi_j} M^0]} (q^{k+(n+1)}, t)$.

Inductive step: Suppose that there is a sequence of $n+1$ classical formulation adaptive transitions:

$$\begin{aligned} &(q, \alpha) \rightarrow q' : \mathcal{A}_1 \\ &(q', \alpha') \rightarrow q'' : \mathcal{A}_2 \\ &\ldots \\ &(q^{(n-1)}, \alpha^{(n-1)}) \rightarrow q^{(n)} : \mathcal{A}_n \\ &(q^{(n)}, \alpha^{(n)}) \rightarrow q^{(n+1)} : \mathcal{A}_{n+1} \end{aligned}$$

so, by the inductive hypothesis, there is a sequence of n transitions $(q, \alpha'\alpha'' \ldots \alpha^{(n-1)}t) \vdash^*_{[\mathbb{F}_{\phi_n} \ldots \mathbb{F}_{\phi_2} \mathbb{F}_{\phi_1} M^0]} (q^{(n)}, t)$ which performs the original n-transitions sequence. Based on this sequence, it is possible to build $(q^{(n)}, \alpha w) \vdash_{[\mathbb{F}_{\phi_{n+1}} M^0]} (q^{(n+1)}, w)$, that performs the same task as the original $\mathcal{A}_{n+1}$. Thus, combining those results, it is clear that the original $(n+1)$-transitions sequence is properly mapped by a $(n+1)$-transitions sequence of the new formulation. □

5 Conclusion

The adaptive function itself, expressed by transformation $\mathbb{F}_\phi M^0$, restricted the nature and number of parameters allowed for adaptive actions to only two: the underlying device and the transformation pair. Thus, using the Adaptive Algebra to redefine the adaptive function concept allowed the resolution of the two problems mentioned in subsection 2.1 (and in [7]) and was the major objective of this work. As collateral results, the adaptive automaton notation became more compact and sharper. The inclusion of an adaptive algebra opened an entire new branch for the study of the adaptive technology. Now, there are many interesting questions to be analyzed, ranging from a more extensive study of the algebraic properties of the adaptive operations, up to the topological characteristics of the adaptive automata space, as well as the connections to other areas, such as the computational learning theory.

5.1 Future Work

With respect to the essential aspects of the adaptive automata space, there are various topics to be explored. The first one, which is going to appear in a forthcoming paper, consists in verifying the possibility to define a higher order adaptive automaton. In this case, the subjacent device is itself an adaptive automaton.

Acknowledgment

The work reported here received support through FAPESP grant 2010/09586-0.

References

1. Carmi, A.: Adapser: An lalr(1) adaptive parser. In: The Israeli Workshop on Programming Languages & Development Environments, Haifa, Israel (July 2002)
2. de Sousa, M., Hirakawa, A.: Robotic mapping and navigation in unknown environments using adaptive automata. In: Adaptive and Natural Computing Algorithms, Part III, pp. 345–348. Springer, Heidelberg (2005)
3. Jackson, Q.T.: Adapting to Babel — Adaptivity and Context-Sensitivity in Parsing: From $a^n b^n c^n$ to RNA. Ibis Publications (2006)
4. Jech, T.J.: About the Axiom of Choice. In: Handbook of Mathematical Logic, pp. 345–370. North-Holland, Amsterdam (1977)

5. Neto, J.J., Silva, P.S.M.: An adaptive framework for the design of software specification languages. In: Adaptive and Natural Computing Algorithms, Part III, pp. 349–352. Springer, Heidelberg (2005)
6. Mikolajczak, B. (ed.): Algebraic and Structural Automata Theory. North-Holland, Amsterdam (1991)
7. Neto, J.J., Pariente, C.A.B.: Adaptive Automata - A Revisited Proposal. In: Champarnaud, J.-M., Maurel, D. (eds.) CIAA 2002. LNCS, vol. 2608, pp. 158–168. Springer, Heidelberg (2003)
8. de Azevedo da Rocha, R.L.: An Attempt to Express the Semantics of the Adaptive Devices. In: Advances in Technological Applications of Logical and Intelligent Systems - Selected Papers from the Sixth Congress on Logic Applied to Technology. Frontiers in Artificial Intelligence and Applications, vol. 186, pp. 13–27. IOS Press, Amsterdam (2009)
9. de Azevedo da Rocha, R.L., Neto, J.J.: Adaptive automaton, limits and complexity compared to the Turing machine - in Portuguese. In: Proceedings of the I LAPTEC, pp. 33–48 (October 2000)
10. de Azevedo da Rocha, R.L., Neto, J.J.: An adaptive finite-state automata application to the problem of reducing the number of states in approximate string matching. In: XI Congreso Argentino de Ciencias de la Computación - CACIC 2005, Concordia, Entre Ríos, Argentina, pp. 17–21 (October 2005)
11. Rubinstein, R.S., Shutt, J.N.: Self-modifying finite automata. In: IFIP Congress, vol. 1, pp. 493–498 (1994)
12. Salthe, S., Matsuno, K.: Self-organization in hierarchical systems. Journal of Social and Evolutionary Systems 18(4), 327–338 (1995)
13. Sragovich, V.G.: Mathematical Theory of Adaptive Control. World Scientific, Singapore (2006)