Mobile Netw Appl (2016) 21:3–9
DOI 10.1007/s11036-015-0615-2

# Algebraically Autonomic Computing

Phan Cong Vinh[1]

Published online: 26 May 2015

**Abstract** Autonomic computing (AC) is characterized by self-* such as self-configuration, self-healing, self-optimization, self-protection and more which run simultaneously in autonomic systems (ASs). Hence, self-* is a set of self-_'s. Each self-_ in self-* is called self-* action. Our way to interpret self-* is to say that self-* actions are running on ASs. In this paper, algebraic objects called monoids are tasked with encoding the self-* action's perspective in all this, i.e. what the self-* action can do, and what happens when different self-* actions are done in succession.

**Keywords** Autonomic computing · Cycle monoid · Free monoid · Monoid · Monoid homomorphism · Presented monoid · Self-* · Self-* action

## 1 Introduction

A common way to interpret self-* in autonomic systems (ASs) is to say that self-* actions are running on ASs. Triggers of self-* actions from self-* can be performed concurrently to transform one AS state into another. A first rule for self-* actions is this: the performance of a sequence of several self-* actions is itself the performance of a self-* action - a more complex self-* action, but a self-* action nonetheless. Algebraic objects called monoids are tasked with encoding the self-* action's perspective in all this, i.e. what the self-* action can do, and what happens when different self-* actions are done in succession. A monoid can be construed as a set of self-* actions, together with a formula that encodes how a sequence of self-* actions is itself considered a self-* action. In this paper we concentrate on monoids.

## 2 Outline

The paper is a reference material for readers who already have a basic understanding of ASs and are now ready to know the novel approach for formalizing self-* in ASs using algebraic language.

Formalization is presented in a straightforward fashion by discussing in detail the necessary components and briefly touching on the more advanced components. Several notes explaining how to use the algebraic structures, including justifications needed in order to achieve the particular results, are presented.

We attempt to make the presentation as self-contained as possible, although familiarity with the notions of self-* in ASs is assumed. Acquaintance with the algebra and the associated notion of algebraic structures is useful for recognizing the results, but is almost everywhere not strictly necessary.

The rest of this paper is organized as follows: Section 3 includes some major work related directly to the content of the paper. In Section 4, AC is described as self-*. Section 5 presents algebraic objects called monoids to be tasked with

✉ Phan Cong Vinh
pcvinh@ntt.edu.vn

[1] Faculty of Information Technology, Nguyen Tat Thanh University, 300A Nguyen Tat Thanh street, Ward 13, District 4, Ho Chi Minh City, Vietnam

encoding the self-* action's perspective. In Section 6, we specify monoid self-* actions. Section 7 presents monoid homomorphisms. Finally, a short summary is given in Section 8.

## 3 Related work

The topic of AC has seen a number of developments through various research investigations following the IBM initiative such as AC paradigm in [1–5]; different approaches and infrastructures in [6–10] for enabling autonomic behaviors [11–14]; core enabling systems, technologies, and services in [15–20] to support the realization of self-* properties in autonomic systems and applications; specific realizations of self-* properties in autonomic systems and applications in [21–27]; architectures and modeling strategies of autonomic networks in [28–30]; middleware and service infrastructure as facilitators of autonomic communications in [31–33]; approaches in [34–36] to equipping current networks with autonomic functionality for migrating this type of networks to autonomic networks.

Moreover, AC has also been intensely studied by various areas of engineering including artificial intelligence, control systems and human orientated systems [37–40]. Autonomic computing has been set as an important requirement for systems devised to work in new generation global networked and distributed environments like wireless networks, P2P networks, Web systems, multi-agent systems, grids, and so on [41–45]. Such systems pose new challenges for the development and application of autonomic computing techniques, due to their special characteristics including: *nondeterminism, context-awareness and goal- and inference-driven adaptability* [37, 46–48].

Finally, the choice of the underlying formalization requires a close look at models for AC. Hence, our interest centers on formal approach to AC taking advantage of category theory [46]. In fact, categories were first described by Samuel Eilenberg and Saunders Mac Lane in 1945 [49], but have since grown substantially to become a branch of modern mathematics. Category theory spreads its influence over the development of both mathematics and theoretical computer science. The categorical structures themselves are still the subject of active research, including work to increase their range of practical applicability.

## 4 Autonomic computing as self-*

Autonomic computing (AC) imitates and simulates the natural intelligence possessed by the human autonomic nervous system using generic computers. This indicates that the nature of software in AC is the simulation and embodiment of human behaviors, and the extension of human capability, reachability, persistency, memory, and information processing speed [50]. AC was first proposed by IBM in 2001 where it is defined as

> "*Autonomic computing is an approach to self-managed computing systems with a minimum of human interference. The term derives from the body's autonomic nervous system, which controls key functions without conscious awareness or involvement*" [51].

AC is generally described as self-*. Formally, let self-* be the set of self-_'s. Each self-_ to be an element in self-* is called a *self-* action*. That is,

$$\text{self-}* = \{\text{self-\_} \parallel \text{self-\_ is a self-* action}\} \tag{1}$$

Note that the symbol $\parallel$ is read as "so that" in the paper. We see that self-CHOP is composed of four self-* actions of self-configuration, self-healing, self-optimization and self-protection. Hence, self-CHOP is a subset of self-*. That is, self-CHOP = {self-configuration, self-healing, self-optimization, self-protection} $\subset$ self-*. Every self-* action must satisfy some certain criteria, so-called *self-* properties*. In [6], T.D. Wolf and T. Holvoet classified the self-* properties in autonomic networks.

In its AC manifesto, IBM proposed eight actions setting forth an AS known as *self-awareness, self-configuration, self-optimization, self-maintenance, self-protection (security and integrity), self-adaptation, self-resource- allocation* and *open-standard-based* [51]. Kinsner pointed out that these actions indicate that IBM perceives AC is a mimicry of human nervous systems [52]. In other words, self-awareness (consciousness) and non-imperative (goal-driven) behaviors are the main features of ASs [50].

## 5 Monoids of self-*

A monoid of self-* is a sequence $(SELF\text{-}*, skip, |)$, where $SELF\text{-}*$ is a set of self-* actions, $skip \in SELF\text{-}*$ is an action, and $|$: $SELF\text{-}* \times SELF\text{-}* \to SELF\text{-}*$ is a concurrence, such that the following conditions hold for all $se\text{-}m, se\text{-}n, se\text{-}p \in SELF\text{-}*$:

- $se\text{-}m \mid skip = se\text{-}m$,
- $skip \mid se\text{-}m = se\text{-}m$, and
- $(se\text{-}m \mid se\text{-}n) \mid se\text{-}p = se\text{-}m \mid (se\text{-}n \mid se\text{-}p)$

The way they are written here is called *infix notation*. We refer to $skip$ as the identity action and to $|$ as the concurrence formula for the monoid. We call the first two rules *identity laws* and the third rule the *associativity law* for monoids.

Alternatively, the rules of identity and associativity can be stated

- $\mid(se\text{-}m, skip) = se\text{-}m$,
- $\mid(skip, se\text{-}m) = se\text{-}m$, and
- $\mid(\mid(se\text{-}m, se\text{-}n), se\text{-}p) = \mid(se\text{-}m, \mid(se\text{-}n, se\text{-}p))$

The way they are written above is called *prefix notation*. Note that we often use infix notation without mentioning it. That is, given a concurrence $\mid : SELF\text{-}* \times SELF\text{-}* \to SELF\text{-}*$, we may write $se\text{-}a \mid se\text{-}b$ rather than $\mid(se\text{-}a, se\text{-}b)$.

There is a monoid with only one action, $(\{skip\}, skip, \mid)$ where $\mid: \{skip\} \times \{skip\} \to \{skip\}$ is the unique concurrence. We call this monoid the *trivial monoid*, and sometimes denote it $\underline{1}$.

In monoid $(SELF\text{-}*, skip, \mid)$, given actions $se\text{-}m_1, se\text{-}m_2, se\text{-}m_3, se\text{-}m_4$ there are five different ways to parenthesize the concurrence $se\text{-}m_1 \mid se\text{-}m_2 \mid se\text{-}m_3 \mid se\text{-}m_4$, and the associativity law for monoids will show them all to be the same. We have

$$\begin{aligned}((se\text{-}m_1 \mid se\text{-}m_2) \mid se\text{-}m_3) \mid se\text{-}m_4 &= (se\text{-}m_1 \mid se\text{-}m_2) \mid (se\text{-}m_3 \mid se\text{-}m_4)\\ &= (se\text{-}m_1 \mid (se\text{-}m_2 \mid se\text{-}m_3)) \mid se\text{-}m_4\\ &= se\text{-}m_1 \mid (se\text{-}m_2 \mid (se\text{-}m_3 \mid se\text{-}m_4))\\ &= se\text{-}m_1 \mid ((se\text{-}m_2 \mid se\text{-}m_3) \mid se\text{-}m_4)\end{aligned}$$

In fact, the concurrence of any list of monoid self-* actions is the same, regardless of parenthesization. Therefore, we can unambiguously write $se\text{-}m_1 \mid se\text{-}m_2 \mid se\text{-}m_3 \mid se\text{-}m_4$ rather than any given parenthesization of it. This is known as the *coherence theorem* and can be found in [53].

## 5.1 Free monoids of self-*

Let $SELF\text{-}*$ be a set of self-* actions. A list in $SELF\text{-}*$ is a pair $(n, f)$ where $n \in \mathbb{N}$ is a natural number (called the length of the list) and $f : \underline{n} \to SELF\text{-}*$ is a function, where $\underline{n} = \{1, 2, \ldots, n\}$. We may denote such a list by

$(n, f) = [f(1), f(2), \ldots, f(n)]$

The empty list is the unique list in which $n = 0$; we may denote it by []. Given a self-* action $se\text{-}x \in SELF\text{-}*$ the singleton list on $se\text{-}x$ is the list $[se\text{-}x]$. Given a list $L = (n, f)$ and a number $i \in \mathbb{N}$ with $i \leqslant n$, the $ith$ entry of $L$ is the self-* action $f(i) \in SELF\text{-}*$.

Given two lists $L = (n, f)$ and $L' = (n', f')$, define the concatenation of $L$ and $L'$, denoted $L \ddagger L'$, to be the list $(n + n', f \ddagger f')$, where $f \ddagger f' : \underline{n + n'} \to SELF\text{-}*$ is given on $i \leqslant n + n'$ by

$$(f \ddagger f')(i) = \begin{cases} f(i) & \text{if } i \leqslant n \\ f'(i-n) & \text{if } i \geqslant n+1 \end{cases} \tag{2}$$

Let $SELF\text{-}* = \{se\text{-}a, se\text{-}b, se\text{-}c, \ldots, se\text{-}z\}$. The following are self-* actions of $List(SELF\text{-}*)$:

$[se\text{-}a, se\text{-}b], [se\text{-}p], [se\text{-}p, se\text{-}a, se\text{-}a], \ldots$

The concatenation of $[se\text{-}a, se\text{-}b]$ and $[se\text{-}p, se\text{-}a, se\text{-}a]$ is $[se\text{-}a, se\text{-}b, se\text{-}p, se\text{-}a, se\text{-}a]$. The concatenation of any list with [] is just itself.

A free monoid generated by $SELF\text{-}*$ is the sequence $(List(SELF\text{-}*), [], \ddagger)$, where $List(SELF\text{-}*)$ is the set of lists of self-* actions in $SELF\text{-}*$, where $[] \in List(SELF\text{-}*)$ is the empty list, and where $\ddagger$ is the operation of list concatenation. We refer to $SELF\text{-}*$ as the set of generators for the free monoid $(List(SELF\text{-}*), [], \ddagger)$.

A free monoid generated by $\varnothing$ is the sequence $(List(\varnothing), [], \ddagger)$ where $List(\varnothing)$ consists only of the empty list. It is the trivial free monoid.

In the section below, we will define the monoid $(List(SELF\text{-}*), [], \ddagger)$ by specifying some generators and some relations. Lists of generators provide us all the possible ways to write self-* actions of $(List(SELF\text{-}*), [], \ddagger)$. The relations allow us to have two ways of writing the same self-* actions.

## 5.2 Presented monoids of self-*

Let $SELF\text{-}*$ be a finite set of self-* actions, let $n \in \mathbb{N}$ be the number of relations we declare, and for each $1 \leqslant i \leqslant n$, let $m_i$ and $m'_i$ be self-* actions of $List(SELF\text{-}*)$. The *monoid presented by generators $SELF\text{-}*$ and relations* $\{(m_i, m'_i)$ where $1 \leqslant i \leqslant n\}$ is the monoid $(List(SELF\text{-}*)/\sim, [], \ddagger)$ defined fully when $\sim$ denotes the equivalence relation on $(List(SELF\text{-}*)$ generated by $\{xm_iy \sim xm'_iy$ where $x, y \in List(SELF\text{-}*), 1 \leqslant i \leqslant n\}$.

Every free monoid $(List(SELF\text{-}*), [], \ddagger)$ is a presented monoid, because we can just take the set of relations to be empty.

Let $SELF\text{-}* = \{se\text{-}a, se\text{-}b, se\text{-}c, se\text{-}d\}$. The idea of presented monoids is that you notice that list of self-* actions $[se\text{-}a, se\text{-}a, se\text{-}c]$ always gives the same result as list of self-* actions $[se\text{-}d, se\text{-}d]$. You also notice that list of self-* actions $[se\text{-}c, se\text{-}a, se\text{-}c, se\text{-}a]$ is the same thing as doing nothing. In this case, we have $m_1 = [se\text{-}a, se\text{-}a, se\text{-}c]$, $m'_1 = [se\text{-}d, se\text{-}d]$, and $m_2 = [se\text{-}c, se\text{-}a, se\text{-}c, se\text{-}a]$, $m'_2 = []$ and relations $\{(m_1, m'_1), (m_2, m'_2)\}$. Really this means that we are equating $m_1$ with $m'_1$ and $m_2$ with $m'_2$, which for convenience we will write out $[se\text{-}a, se\text{-}a, se\text{-}c] = [se\text{-}d, se\text{-}d]$ and $[se\text{-}c, se\text{-}a, se\text{-}c, se\text{-}a] = []$. To see how this plays out, we give an example of a calculation in $List(SELF\text{-}*)/\!\sim$. Namely,

$$\begin{aligned}[se\text{-}b, \underline{se\text{-}d, se\text{-}d}, se\text{-}a, se\text{-}c, se\text{-}a, se\text{-}a, se\text{-}c, se\text{-}d] &= [se\text{-}b, se\text{-}a, \underline{se\text{-}a, se\text{-}c, se\text{-}a, se\text{-}c},\\ &\quad se\text{-}a, se\text{-}a, se\text{-}c, se\text{-}d]\\ &= [se\text{-}b, se\text{-}a, \underline{se\text{-}a, se\text{-}a, se\text{-}c}, se\text{-}d]\\ &= [se\text{-}b, se\text{-}a, se\text{-}d, se\text{-}d, se\text{-}d]\end{aligned}$$

## 5.3 Cyclic monoids of self-*

A monoid is called cyclic if it has a presentation involving only one generator. Let $se\text{-}a$ be a self-* action; we look at some cyclic monoids generated by $\{se\text{-}a\}$.

With no relations the monoid will be the free monoid on one generator, and will have underlying set $\{[], [se\text{-}a], [se\text{-}a, se\text{-}a], [se\text{-}a, se\text{-}a, se\text{-}a], \ldots\}$, with identity list $[]$ and concatenation such that $[se\text{-}a, se\text{-}a] \ddagger [se\text{-}a] = [se\text{-}a, se\text{-}a, se\text{-}a, se\text{-}a] = se\text{-}a^4$. Note that $se\text{-}a^4$ is shorthand for $[se\text{-}a, se\text{-}a, se\text{-}a, se\text{-}a]$.

With the relation $se\text{-}a \sim []$ we will get the trivial monoid, a monoid having only one action. Consider the cyclic monoid with generator $se\text{-}a$ and relation $se\text{-}a^7 = se\text{-}a^4$. This monoid has seven actions

$$\{[] = se\text{-}a^0, se\text{-}a = se\text{-}a^1, se\text{-}a^2, se\text{-}a^3, se\text{-}a^4, se\text{-}a^5, se\text{-}a^6\}$$

and we know that

$$se\text{-}a^6 \ddagger se\text{-}a^5 = se\text{-}a^7 \ddagger se\text{-}a^4 = se\text{-}a^4 \ddagger se\text{-}a^4 = se\text{-}a^7 \ddagger se\text{-}a = se\text{-}a^5$$

We can depict this monoid as follows

$$\begin{tikzcd} skip \arrow[r] & se\text{-}a \arrow[r] & se\text{-}a^2 \arrow[r] & se\text{-}a^3 \arrow[r] & se\text{-}a^4 \arrow[r] & se\text{-}a^5 \arrow[r] & se\text{-}a^6 \arrow[ll, bend left] \end{tikzcd}$$

## 6 Actions of monoid of self-*

Let $AS$ be a set of autonomic system states. A self-* action of monoid $(SELF\text{-}*, skip, |)$ on $AS$, or simply a self-* action of $SELF\text{-}*$ on $AS$ or a $SELF\text{-}*$ action on $AS$, is a function

$$\circlearrowleft : SELF\text{-}* \times AS \to AS$$

such that the following conditions hold for all $se\text{-}m, se\text{-}n \in SELF\text{-}*$ and all $s \in AS$:

- $skip \circlearrowleft s = s$
- $se\text{-}n \circlearrowleft (se\text{-}n \circlearrowleft s) = (se\text{-}n \mid se\text{-}n) \circlearrowleft s$

Alternately, it is sometimes useful, we can rewrite $\circlearrowleft$ as $\alpha : SELF\text{-}* \times AS \to AS$ and restate the above conditions as

- $\alpha(skip, s) = s$
- $\alpha(se\text{-}n, \alpha(se\text{-}n, s)) = \alpha(se\text{-}n \mid se\text{-}n, s)$

The following proposition expresses the notion of autonomic system in terms of free monoids and their actions on finite sets.

**Proposition 1** *Let $SELF\text{-}*$ and $AS$ be finite non-empty sets of self-* actions and autonomic system states, respectively. Giving a function $\alpha : SELF\text{-}* \times AS \to AS$ is equivalent to giving an action of the free monoid $List(SELF\text{-}*)$ on $AS$.*

*Proof* We know that function $\delta : List(SELF\text{-}*) \times AS \to AS$ constitutes an action of the monoid $List(SELF\text{-}*)$ on the set $AS$ if and only if, for all $s \in AS$ we have $\delta([], s) = s$, and for any two actions $m, m' \in List(SELF\text{-}*)$ we have $\delta(m, \delta(m', s)) = \delta(m \ddagger m', s)$. Let

$$A = \{\delta : List(SELF\text{-}*) \times AS \to AS \,\|\, \delta \text{ constitutes an action}\}$$

We need to prove that there is an isomorphism of sets

$$\phi : A \xrightarrow{\cong} \mathrm{Hom}_{\mathbf{Set}}(SELF\text{-}* \times AS, AS)$$

Given an element $\delta : List(SELF\text{-}*) \times AS \to AS$ in $A$, define $\phi(\delta)$ on an element $(se\text{-}a, s) \in SELF\text{-}* \times AS$ by $\phi(\delta)(se\text{-}a, s) \overset{def}{=} \delta([se\text{-}a], s)$, where $[se\text{-}a]$ is the one-element list.

We now define $\psi : \mathrm{Hom}_{\mathbf{Set}}(SELF\text{-}* \times AS, AS) \to A$. Given an element $f \in \mathrm{Hom}_{\mathbf{Set}}(SELF\text{-}* \times AS, AS)$ define $\psi(f) : List(SELF\text{-}*) \times AS \to AS$ on a pair $(L, s) \in List(SELF\text{-}*) \times AS$, where $L = [\delta_1, \ldots, \delta_n]$ as follows. By induction, if $n = 0$, put $\psi(f)(L, s) = s$; if $n \geqslant 1$, let $L' = [\delta_1, \ldots, \delta_{n-1}]$ and put $\psi(f)(L, s) = \psi(f)(L', f(\delta_n, s))$.

We checks easily that $\psi(f)$ satisfies the two rules of action above, making it an action of $List(SELF\text{-}*)$ on $AS$. It is also easy to check that $\phi$ and $\psi$ are mutually inverse, completing the proof. $\square$

It follows that *an autonomic system is an action of a free monoid on a finite set.*

## 7 Monoid homomorphisms

Let $\mathcal{M} : (SELF\text{-}*, skip, |)$ and $\mathcal{M}' : (SELF\text{-}*', skip', |')$ be monoids. A monoid homomorphism $f$ from $\mathcal{M}$ to $\mathcal{M}'$, denoted $f : \mathcal{M} \to \mathcal{M}'$, is a function $f : SELF\text{-}* \to SELF\text{-}*'$ satisfying two conditions:

- $f(skip) = skip'$
- $f(se\text{-}a \mid se\text{-}b) = f(se\text{-}a) \mid' f(se\text{-}b)$, for all $se\text{-}a, se\text{-}b \in SELF\text{-}*$

The set of monoid homomorphisms from $\mathcal{M}$ to $\mathcal{M}'$ is denoted $\mathrm{Hom}_{\mathbf{Mon}}(\mathcal{M}, \mathcal{M}')$.

Let $SELF\text{-}* = \{se\text{-}a, se\text{-}c, se\text{-}g, se\text{-}u\}$ and let $SELF\text{-}*' = SELF\text{-}*^3$, the set of triplets in $SELF\text{-}*$. Let $\mathcal{M} = List(SELF\text{-}*)$ be the free monoid on $SELF\text{-}*$ and let $\mathcal{M}' = List(SELF\text{-}*')$ denote the free monoid on $SELF\text{-}*'$. There is a monoid homomorphism $F : \mathcal{M}' \to \mathcal{M}$ given by sending $m = (se\text{-}a, se\text{-}b, se\text{-}c)$ to the list $[se\text{-}a, se\text{-}b, se\text{-}c]$.

Given any monoids $List(SELF\text{-}*)$ there is a unique monoid homomorphism from $List(SELF\text{-}*)$ to the trivial monoid $\underline{1}$. There is also a unique homomorphism $\underline{1} \to List(SELF\text{-}*)$. These facts together have an upshot: between any two monoids $List(SELF\text{-}*)$ and $List(SELF\text{-}*')$ we can always construct a homomorphism

$$List(SELF\text{-}*) \to \underline{1} \to List(SELF\text{-}*')$$

which we call the trivial homomorphism $List(SELF\text{-}*) \to List(SELF\text{-}*')$. A morphism $List(SELF\text{-}*) \to List(SELF\text{-}*')$ that is not trivial is called a nontrivial homomorphism.

**Proposition 2** *let $F(SELF\text{-}*) : (List(SELF\text{-}*), [], \ddagger)$ be the free monoid on $SELF\text{-}*$, and let $\mathcal{M} : (M, skip, |)$ be any monoid. There is a natural bijection*

$$\mathrm{Hom}_{\mathrm{Mon}}(F(SELF\text{-}*), \mathcal{M}) \xrightarrow{\cong} \mathrm{Hom}_{\mathrm{Set}}(SELF\text{-}*, M)$$

*Proof* We provide a function $\phi : \mathrm{Hom}_{\mathbf{Mon}}(F(SELF\text{-}*), \mathcal{M}) \longrightarrow \mathrm{Hom}_{\mathbf{Set}}(SELF\text{-}*, M)$ and a function $\psi : \mathrm{Hom}_{\mathbf{Set}}(SELF\text{-}*, M) \longrightarrow \mathrm{Hom}_{\mathbf{Mon}}(F(SELF\text{-}*), \mathcal{M})$ and show that they are mutually inverse. Let us first construct $\phi$. Given a monoid homomorphism $f : F(SELF\text{-}*) \longrightarrow \mathcal{M}$, we need to provide $\phi(f) : SELF\text{-}* \longrightarrow M$. Given any $se\text{-}g \in SELF\text{-}*$ we define $\phi(f)(se\text{-}g) \overset{def}{=} f[se\text{-}g]$.

Now let us construct $\psi$. Given $p : SELF\text{-}* \longrightarrow M$, we need to provide $\psi(p) : List(SELF\text{-}*) \longrightarrow \mathcal{M}$ such that $\psi(p)$ is a monoid homomorphism. For a list $L = [se\text{-}g_1, \ldots, se\text{-}g_n] \in List(SELF\text{-}*)$, define $\psi(p)(L) : p(se\text{-}g_1) \mid \ldots \mid p(se\text{-}g_n) \in M$. In particular, $\psi(p)([]) = skip$. It is not hard to see that this is a monoid homomorphism. It is also easy to see that $\phi; \psi(p) = p$ for all $p \in \mathrm{Hom}_{\mathbf{Set}}(SELF\text{-}*, M)$. We show that $\psi; \phi(f) = f$ for all $f \in \mathrm{Hom}_{\mathbf{Mon}}(F(SELF\text{-}*), \mathcal{M})$. Choose $L = [se\text{-}g_1, \ldots, se\text{-}g_n] \in List(SELF\text{-}*)$. Then

$$\begin{aligned}\psi(\phi f)(L) &= (\phi f)(se\text{-}g_1) \mid \ldots \mid (\phi f)(se\text{-}g_n)\\ &= f[se\text{-}g_1] \mid \ldots \mid f[se\text{-}g_n]\\ &= f([se\text{-}g_1, \ldots, se\text{-}g_n])\\ &= f(L)\end{aligned}$$

□

## 8 Conclusions

In this paper, based on algebraic objects called monoids, we have algebraically specified to autonomic computing (AC) from which some useful properties of AC emerge. Monoids are tasked with encoding the perspective of self-* action in all this, i.e. what the self-* action can do, and what happens when different self-* actions are done in succession. A monoid is construed as a set of self-* actions, together with a formula that encodes how a sequence of self-* actions is itself considered a self-* action.

**Acknowledgments** Thank you to NTTU (Nguyen Tat Thanh University, Vietnam) for the constant support of our work which culminated in the publication of this paper. As always, we are deeply indebted to the anonymous reviewers for their helpful comments and valuable suggestions which have contributed to the final preparation of the paper.

## References

1. Ganek A (2006) Autonomic computing: concepts, infrastructure and applications. In: Overview of autonomic computing: origins, evolution, direction, 1st edn. CRC Press, pp 3–18
2. Bustard DW, Sterritt R (2006) Autonomic computing: concepts, infrastructure and applications. In: A requirements engineering perspective on autonomic systems development, 1st edn. CRC Press, pp 19–34
3. Renesse RV, Birman KP (2006) Autonomic computing: concepts, infrastructure and applications. In: Autonomic computing: a system-wide perspective, 1st edn. CRC Press, pp 35–48
4. Parashar M (2006) Autonomic computing: concepts, infrastructure and applications. In: Autonomic grid computing: concepts, requirements, and infrastructure, 1st edn. CRC Press, pp 49–70
5. Sweitzer JW, Draper C (2006) Autonomic computing: concepts, infrastructure and applications. In: Architecture overview for autonomic computing, 1st edn. CRC Press, pp 71–98
6. Wolf TD, Holvoet T (2006) Autonomic computing: concepts, infrastructure and applications. In: A taxonomy for self-* properties in decentralized autonomic computing, 1st edn. CRC Press, pp 101–120

7. Anthony R, Butler A, Ibrahim M (2006) Autonomic computing: concepts, infrastructure and applications. In: Exploiting emergence in autonomic systems, 1st edn. CRC Press, pp 121–148
8. Abdelwahed S, Kandasamy N (2006) Autonomic computing: concepts, infrastructure and applications. In: A control-based approach to autonomic performance management in computing systems, 1st edn. CRC Press, pp 149–168
9. Sadjadi SM, McKinley PK (2006) Autonomic computing: concepts, infrastructure and applications. In: Transparent autonomization in composite systems, 1st edn. CRC Press, pp 169–188
10. Steenkiste P, Huang AC (2006) Autonomic computing: concepts, infrastructure and applications. In: Recipe-based service configuration and adaptation, 1st edn. CRC Press, pp 189–208
11. Vinh PC (2006) Formal aspects of dynamic reconfigurability in reconfigurable computing systems. PhD thesis, London South Bank University, London
12. Vinh PC, Bowen JP (2007) A formal approach to aspect-oriented modular reconfigurable computing. In: Proceedings of 1st IEEE & IFIP international symposium on theoretical aspects of software engineering (TASE). Shanghai, China, 6–8 June. IEEE Computer Society Press, pp 369–378
13. Vinh PC, Bowen JP (2008) Formalization of data flow computing and a coinductive approach to verifying flowware synthesis. LNCS Trans Comput Sci 1(4750):1–36
14. Vinh PC (2007) Homomorphism between AOMRC and Hoare model of deterministic reconfiguration processes in reconfigurable computing systems. Sci Ann Comput Sci XVII:113–145
15. Liu H, Parashar M (2006) Autonomic computing: concepts, infrastructure and applications. In: A programming system for autonomic self-managing applications, 1st edn. CRC Press, pp 211–236
16. Heinis T, Pautasso C, Alonso G (2006) Autonomic computing: concepts, infrastructure and applications. In: A self-configuring service composition engine, 1st edn. CRC Press, pp 237–252
17. Chess DM, Hanson JE, Kephart JO, Whalley I, White SR (2006) Autonomic computing: concepts, infrastructure and applications. In: Dynamic collaboration in autonomic computing, 1st edn. CRC Press, pp 253–274
18. Schwan K et al (2006) Autonomic computing: concepts, infrastructure and applications. In: AutoFlow: autonomic information flows for critical information systems, 1st edn. CRC Press, pp 275–304
19. Adams R et al (2006) Autonomic Computing: concepts, infrastructure and applications. In: Scalable management – technologies for management of large-scale, distributed systems, 1st edn. CRC Press, pp 305–328
20. Durham L, Milenkovic M, Cayton P, Yousif M (2006) Autonomic computing: concepts, infrastructure and applications. In: Platform support for autonomic computing: a research vehicle, 1st edn. CRC Press, pp 329–350
21. Menascé DA, Bennani MN (2006) Autonomic computing: concepts, infrastructure and applications. In: Dynamic server allocation for autonomic service centers in the presence of failures, 1st edn. CRC Press, pp 353–368
22. Griffith R, Valetto G, Kaiser G (2006) Autonomic computing: concepts, infrastructure and applications. In: Effecting runtime reconfiguration in managed execution environments, 1st edn. CRC Press, pp 369–388
23. Chakravarti A, Baumgartner G, Lauria M (2006) Autonomic computing: concepts, infrastructure and applications. In: Self-organizing scheduling on the organic grid, 1st edn. CRC Press, pp 389–412
24. Bhat V, Parashar M, Kandasamy N (2006) Autonomic computing: concepts, infrastructure and applications. In: Autonomic data streaming for high-performance scientific applications, 1st edn. CRC Press, pp 413–434
25. Khargharia B, Hariri S (2006) Autonomic computing: concepts, infrastructure and applications. In: Autonomic power and performance management of internet data, 1st edn. CRC Press, pp 435–470
26. Jiang G et al (2006) Autonomic computing: concepts, infrastructure and applications. In: Trace analysis for fault detection in application servers, 1st edn. CRC Press, pp 471–492
27. Qu G, Hariri S (2006) Autonomic computing: concepts, infrastructure and applications. In: Anomaly-based self protection against network attacks, 1st edn. CRC Press, pp 493–522
28. Meer SVD et al (2008) Advanced autonomic networking and communication. In: Technology neutral principles and concepts for autonomic networking, 1st edn. Whitestein series in software agent technologies and autonomic computing. Springer, pp 1–25
29. López JAL, Munoz JMG, Padial JM (2008) Advanced autonomic networking and communication. In: A Telco approach to autonomic infrastructure management, 1st edn. Whitestein series in software agent technologies and autonomic computing. Springer, pp 27–42
30. Fahy C et al (2008) Advanced autonomic networking and communication. In: Modelling behaviour and distribution for the management of next generation networks, 1st edn. Whitestein series in software agent technologies and autonomic computing. Springer, pp 43–62
31. Greenwood D, Ghizzioli R (2008) Advanced autonomic networking and communication. In: Autonomic communication with RASCAL hybrid connectivity management, 1st edn. Whitestein series in software agent technologies and autonomic computing. Springer, pp 63–80
32. Nguengang G et al (2008) Advanced autonomic networking and communication. In: Autonomic resource regulation in IP military networks: a situatedness based knowledge plane, 1st edn. Whitestein series in software agent technologies and autonomic computing. Springer, pp 81–100
33. Calisti M, Ghizzioli R, Greenwood D (2008) Advanced autonomic networking and communication. In: Autonomic service access management for next generation converged networks, 1st edn. Whitestein series in software agent technologies and autonomic computing. Springer, pp 101–126
34. Razzaque MA, Dobson S, Nixon P (2008) Advanced autonomic networking and communication. In: Cross-layer optimisations for autonomic networks, 1st edn. Whitestein series in software agent technologies and autonomic computing. Springer, pp 127–148
35. Amoud RR et al (2008) Advanced autonomic networking and communication. In: An autonomic MPLS DiffServ-TE domain, 1st edn. Whitestein series in software agent technologies and autonomic computing. Springer, pp 149–168
36. Chen J et al (2008) Advanced autonomic networking and communication. In: Game theoretic framework for autonomic spectrum management in heterogeneous wireless networks, 1st edn. Whitestein series in software agent technologies and autonomic computing. Springer, pp 169–190
37. Wang Y (2007) Toward theoretical foundations of autonomic computing. Int J Cogn Inf Nat Intell (IJCiNi) 1(3):1–16
38. Jin X, Liu J (2004) From individual based modeling to autonomy oriented computation. In: Nickles M, Rovatsos M, Weiss G (eds) Agents and computational autonomy: potential, risks, and solutions, volume 2969 of Lecture Notes in Computer Science. Springer, Berlin, pp 151–169
39. Pacheco O (2004) Autonomy in an organizational context. In: Nickles M, Rovatsos M, Weiss G (eds) Agents and computational autonomy: potential, risks, and solutions, volume 2969 of Lecture Notes in Computer Science. Springer, Berlin, pp 195–208

40. Witkowski M, Stathis K (2004) A dialectic architecture for computational autonomy. In: Nickles M, Rovatsos M, Weiss G (eds) Agents and computational autonomy: potential, risks, and solutions, volume 2969 of Lecture Notes in Computer Science. Springer, Berlin, pp 261–273
41. Parashar M, Hariri S (eds) (2006) Autonomic computing: concepts, infrastructure and applications, 1st edn. CRC Press. 568 pp
42. Calisti M, Meer SVD, Strassner J (eds) (2008) Advanced autonomic networking and communication. Whitestein series in software agent technologies and autonomic computing. Springer, 190 pp
43. Ko S, Gupta I, Jo Y (2007) Novel mathematics-inspired algorithms for self-adaptive peer-to-peer computing. In: Serugendo GDM, Flatin JPM, Jelasity M (eds) Proceedings of 1st international conference on self-adaptive and self-organizing systems (SASO'07). IEEE Computer Society Press, Boston, pp 3–12
44. Yang B, Liu J (2007) An Autonomy Oriented Computing (AOC) approach to distributed network community mining. In: Serugendo GDM, Flatin JPM, Jelasity M (eds) Proceedings of 1st international conference on self-adaptive and self-organizing systems (SASO'07). IEEE Computer Society Press, Boston, pp 151–160
45. Butera W (2007) Text display and graphics control on a paintable computer. In: Serugendo GDM, Flatin JPM, Jelasity M (eds) Proceedings of 1st international conference on self-adaptive and self-organizing systems (SASO'07). IEEE Computer Society Press, Boston, pp 45–54
46. Vinh PC (2009) Autonomic computing and networking. In: Formal aspects of self-* in autonomic networked computing systems. Springer, pp 381–410
47. Vinh PC (2014) Toward formalized autonomic networking. Mob Netw Appl 19(5):598–607
48. Vinh PC (2014) Self-adaptation in collective adaptive systems. Mob Netw Appl 19(5):626–633
49. Lawvere FW, Schanuel SH (1997) Conceptual mathematics: a first introduction to categories, 1st edn. Cambridge University Press, Cambridge
50. Wang Y (2007) Exploring machine cognition mechanisms for autonomic computing. Int J Cogn Inf Nat Intell (IJCINI) 1(2):i–v
51. IBM (2001) Autonomic computing manifesto. Retrieved from http://www.research.ibm.com/autonomic/
52. Kinsner W (2007) Towards cognitive machines: multiscale measures and analysis. Int J Cogn Inf Nat Intell (IJCINI) 1(1):28–38
53. van Oosten J (2002) Basic category theory. Department of Mathematics, Utrecht University, The Netherlands