Creating context for decision making in autonomic systems with collaborative reinforcement learning

MORANE GRUENPETER, Pierre-and-Marie-Curie University

Abstract: Collaborative reinforcement learning (CRL) is a new approach in autonomic systems for self-management decision making. CRL allows to share effort between autonomic elements. We will show several CRL algorithms that insure global optimization while learning locally.

Moreover, we will use a linguistic approach for learning communication and creating context in autonomic elements. A mapping of utility functions can be useful for reducing the state space of possible actions.

Additional Key Words and Phrases: Autonomic Systems, Collaborative Reinforcement learning, Q-learning, learning communication, Context, Pragmatics

ACM Reference Format:
Morane GRUENPETER, 2017. ALASCA report. V, N, Article A (January 2017), 11 pages.
DOI: 10.1145/0000000.0000000

1. INTRODUCTION

The autonomic computing system is self-configuring, self protecting, self-healing and self optimizing. It is a self-managing computer system that adapts itself according to an administrator's goals [Kephart and Chess 2003].

In 2010,Dobson et al.'s article state that IBM's original vision of autonomic computing remains unfilfilled. What is missing? A long range, overeaching strategy [Dobson et al. 2010]. This strategy can be completed with a decision making mechanism that acts and reacts to a multitude of factors in a given time frame.

We determine the viability of the mechanism by its response time and quality of response, therfore, building decision mechanisms is an important aspect of self management. A "good" global decision for the system is a well organized allocation of resources.

A state-of-the-art overview of autonomic decision making techniques is provided in [Maggio et al. 2011]. They describe the design, implementation and testing of heuristics solutions (greedy algorithms), standard control-based solutions (proportional integral and petri nets), advanced control-based solutions (adaptive control and model predictive control), model-based machine learning solutions (neural networks) and model-free machine learning solutions (reinforcement learning). All are tested for self-optimization.

Accordingly, reinforcement learning implementations demonstrate slow convergence and aren't viable by themselves. However, CRL techniques have been used as an improvement of regular reinforcement learning for faster reaction.

In this paper, I will present a linguistic approach to CRL decision making in autonomic systems by learning communication and reducing the state space of possible actions.

First we will present the decision making phase in autonomic systems. Then we will investigate CRL for self-management introduced in multi-agent systems. Then we will define context and connect it to communication learning as a way to collaborate in an autonomic systems. Finally we will propose possibilities for further investigation on this subject.

2. THE DECISION PHASE

2.1. Decision in autonomic systems

The decision phase in autonomic system is part of analyze and plan steps of the MAPE loop (monitor, analyze, plan, execute). In a self-managed system, where constant adaptation is the key to a well engineered system, the execution of an action depends on the state of the system. The decision is chosen in regards to a decision policy defined in the system. According to an artificial intelligence perspective [Kephart and Walsh 2004] we have three different types of policies: action based, goal based, and utility function based.

action based < goal based < utility function based

The last policy is the most complex but is more accurate in the decision process. An action based decision is an if [State] then [action]. A goal based policy determines a sequence of actions to achieve a certain defined goal. Finally the utility function policy specifies tradeoffs using rewards and penalties to gain utility. Utility function policies generalize goal based policies and the most desired state isn't specified in advance. It will be calculated recursively by the agent [Kephart and Walsh 2004]. That is to say that an autonomic element can learn it's own utility function and optimize it. When we use CRL, which we will define in the next section, the agents can communicate about their utility function modifications and optimizations.

— **What is a good decision?** In our case the decision is related to the ressources allocated to the auntonomic element and a good decision is relative to the system. Good isn't optimal, but most optimal decisions aren't attainable in the time frame. In comlpex systems, heuristic solutions are used to approach the optimum.
— **Can a self optimizing system find its optimum ?** It depends on the system's model. In simple models, the optimum can be found, but an autonomic system is more complex and an optimum at time t can't stay optimal for long. Furthermore, if the system has found its optimum and has adapted itself to this optimum, the state of the system changed so that the question remains: is the optimum still optimal?
— **Can a local "good" decision be the optimum for the global system?** We have seen in class that most of the time the contrary is more accurate, the so called "selfish" decision of one autonomic element is not likely to be globally optimal.

2.2. Decision techniques

Common techniques for the Decision phase are presented and compared in [Maggio et al. 2011]

— Heuristic solutions *(greedy algorithms)*
— Standard control-based solutions *(PID, Petri nets)*
— Advanced control-based solutions *(MDP)*
— Model based machine learning solutions *(neural networks)*
— **Model free machine learning solutions *(reinforcement learning)***

All these techniques have been presented and tested on an application called Heart-Beats framework where the autonomic system allocated resources for the sensor data. Their conclusion is that "the best decision method can vary depending on the specific application to be optimized; however, adaptive and model-predictive control systems tend to produce good performance and may work best when the applications to be controlled are a priori unknown." [Maggio et al. 2011]

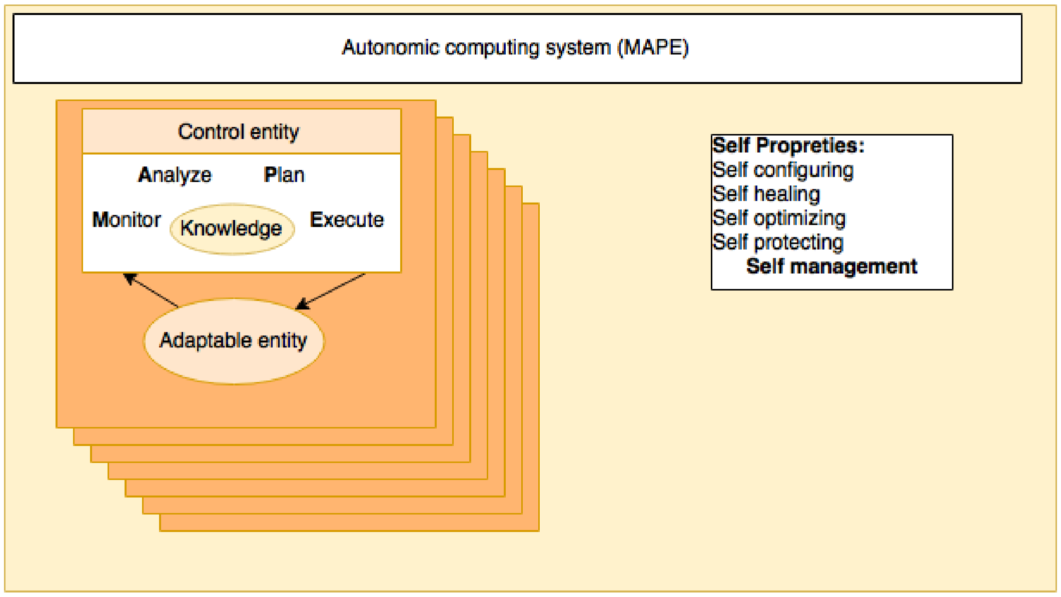


Fig. 1. Autonomic system with MAPE loop

2.3. Reinforcement learning in autonomic systems

2.3.1. Q-learning. Q-learning is a model free reinforcement learning technique. It can be used to find an optimal action-selection policy for any given (finite) Markov decision process (MDP[1]). In 2007,IBM researchers published a manifesto and case studies about Reinforcement Learning in Autonomic Computing was published [Tesauro 2007], where the RL technique was a promising methodology to a more effectivre real-time self-managing system.

To illustrate in figure 1 the knowledge module is the center of the MAPE loop and it is difficult to accuratly engineer it in a continuously evolving environment. The idea of the manifesto was to develop a model which would consider the influence that actions have on the system. Introducing SARSA algorithm into the management element. SARSA means *State-Action-Reward-State-Action* as Algorithm 1 computes a decision by choosing an action and evaluating it in regards of its reward:

$$Q_{S_k,a_k} \leftarrow Q_{S_k,a_k} + \overbrace{\alpha}^{\text{learning rate}} [\underbrace{r_{k+1}}_{reward} + \gamma Q_{(s_{k+1},a_{k+1})} - Q_{(s_k,a_k)}]$$

The learning rate, α determines how quickly the used information will be replaced with new information. Will the system prefer to choose a known state or a new state? The discount factor γ determines the importance of future rewards, when closer to 0, the agent will prefer short-term rewards and when closer to 1, the agent will prefer

[1]Markov decision processes can be solved with RL without explicit specification of the transition probabilities

long-term rewards. In [Tesauro 2007] a Hybrid RL method is also presented where learning is used to observe the policy in place and adapt accordingly.

ALGORITHM 1: SARSA algorithm

```
Input: $Q_{(s,a)}$ arbitrarily
repeat
    Initialize $s$ ;
    Choose $a$ from $s$ using policy derived from $Q$ ;
    for each step in episode do
        Take action $a$ ;
        Observe $r$, $s'$ ;
        Choose $a'$ from $s'$ using policy derived from $Q$ ;
        $Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma Q(s',a') - Q(s,a)]$ ;
        $s$ ←s' a←a' end
    until end of episode;
```

2.3.2. Limitations of RL. Even though RL can solve MDPs without having the explicit model of the system and require less initial knowledge, RL isn't enough and convergence is slow. In [Maggio et al. 2011], the SARSA algorithm is presented and tested alongside other self-optimization decision techniques and the clear result is, that RL isn't efficient in comparison to model based techniques. The main limitation is having a system with slow convergence, where timing is important. Even if the system presents better results in the long run, taking bad decision in the beginning before acquiring knowledge compromises the system's goals. Thus an autonomic system can't be deployed without simulation, which complexifies the developpement of the system.

How can we address this limitations? Collaborative Reinforcement is an option to share inforamtion about states that have already been observed by other autonomic elements.

3. COLLABORATION AND COORDINATION

Learning and optimizing in a sophisticated autonomic system forces the autonomic elements to continually adapt to their environment like agents in a multi agent system. The autonomic elements have the common goal of self-optimization of the system, therefore the decision making process can't be independent.

The major difference between coordination and collaboration is the point of view of the autonomic element. When coordinating, the autonomic element can, and in most systems will, stay independent with no regards to other elements while an arbiter or other meta-control layer will be in charge of coordination. On the other hand, when collaborating, the elements can't stay independent, the elements will need to *work together* for the global optimization.

We know that a local optimized decision doesn't necessarily leads to aglobal optimization. Also the autonomic system continuouslly changes, therefore coordination or collaboration are essential to seek and maintain global optimization.

In figure 2 we can see shared resources for different autonomic elements. Each autonomic element represents an application manager to which a different amount of users are connected. We can see that the system needs to divide resources relatively to user demand. The system needs to take a global decision. To this end we can use coordination or collaboration.

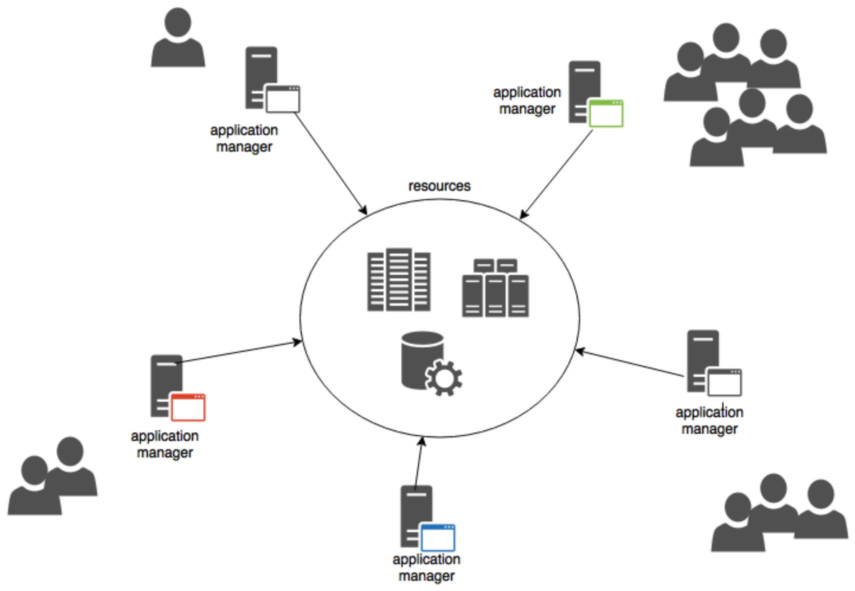


Fig. 2. Resources allocation in an autonomic system

3.1. Coordination

Different coordination mechanisms can be used in autonomic systems. First, we have implicit coordination: *a posteriori coordination*. However it is usually unefficient to coordinate after an action has already been taken by one element.

In [Melekhova and Malenfant], a decentralised token based scheme is proposed, where each token reperesents the possibility to access the shared resources, the autonomic manager needs to acquire the token in order to use the resources only when in posession of the token. Then the autonomic manager can release the token, when no longer needed. This approach is a way to determine a communication scheme without the need to communicate with every other autonmic element.The autonomic managers are linked through the Token system, and exchange their needs by acquiring and releasing tokens:

$$acquire \rightarrow keep \rightarrow release$$

Another approach for coordination are the coordination graphs (CG) in [Kok and Vlassis 2006] where the global Q-function is factored in the graph. Kok and Vlassis present variable elimination (VE) algorithm and max-plus algorithm (an approximate alternative to VE) as a coordination solution with payoff propagation. The VE algorithm doesn't scale well because of its important dependancy.

3.2. Collaboration

Collaborative reinforcement learning means to collectively adapt to a changing environment by local optimisation and global communication. In [Dowling et al. 2005],the advantages of adding collaboration techniques to RL are described as following: distributed systems can use distributed RL where the learning rate is increased due to the shared effort. CRL allows to handle larger state spaces by partitionning a given state space and giving each agent learning abilities.

In [Kok and Vlassis 2006] a cooperation approach is presented. Cooperation is similar to collaboration: the algorithm describes a multi-agents system represented on a

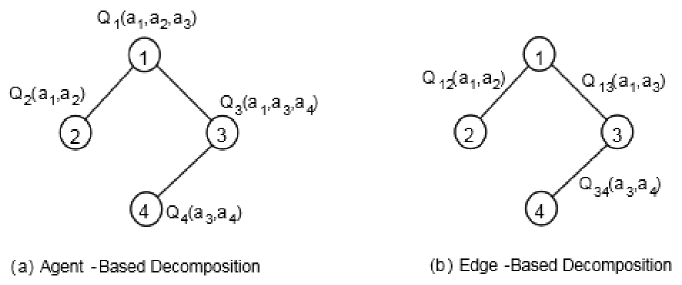


figure.3

Fig. 3. The global Q-function for a 4-agent problem can be modeled in a coordination graph [Kok and Vlassis 2006]

CG with an Agent-Based decomposition and an Edge-Based decomposition as we can see in figure 3.

Sparse cooperative Q-learning algorithm. The global Q-function equals the sum of the local Q-functions of all n-agents $Q_{(s,a)} = \sum_{i=1}^{n} Q_i(s_i, a_i)$ in [Kok and Vlassis 2006]:

$$\sum_{i=1}^{n} Q_i(s_i, a_i) := \sum_{i=1}^{n} Q_i(s_i, a_i) + [\sum_{i=1}^{n} R_i(s, a)] + \gamma max(s', a') - \sum_{i=1}^{n} Q_i(s_i, a_i)$$

The SCQL algorithm approximates the global action-value fonction using a Coordination graph (CG) framework with two different decompositions. First, an agent-based decomposition (SparseQ agent)and second an edge-based decomposition (SparseQ edge).In the SparseQ, each agent stores a Q-function based on its own action and the actions of its neighbours, the Q function is updated based on the local temporal-difference. The representation and computational complexity are similar to the coordination approach.

Kok and Vlassis resume, after testing the SparseQ variants, that they outperform other existing multiagent-learning methods. The combination of the edge-based decomposition and the max-plus algorithm results in a method which scales only linearly in the number of dependencies of the problem.

In [Dowling et al. 2005] CRL is defined as a decentralized reinforcement learning system with a set of discrete optimization problems (DOP) and a set of independent learners. Here similarly to Sparse cooperative Q-learning the system is modeled as a graph and the agents goal is to solve in a sequential manner the DOP for the global system with a cost model for network connections. Collaboration is possible when an agent executes an action and forwards the result to its neighbour. The neighbour returns a cost to the value received. The cost signal, $g_{-i}(s, a)$, is an environement calculation for an estimated network cost of delegating the DOP. The costs and state-transition probabilities are kept in a cache table for each agent.

3.3. Collaborate towards better resources allocation

In autonomic systems we are seeking methods to achieve a global optimized decision for the system, where resources are limited and distributed. When using reinforcement learning, without collaboration, the autonomic element is modelized with it's own Q-learning function, and the learning process stays independent from others progress. The question remains

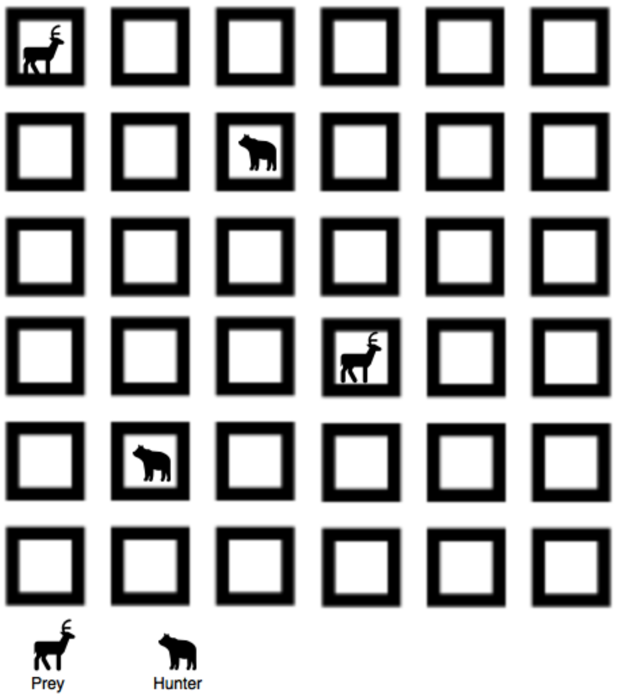


Fig. 4. Hunter and Prey problem in game theory

> Can N communicating agents outperform N non-communicating agents? [Tan 1998]

We can analyse the Hunter-Prey problem ilustrated in figure 4 where on a $n \cdot n$ grid there are x hunters and y preys. The goal for the hunters is to find a prey. The location of all agents changes each step in time. In this experiment, the hunters can be independent and have their own experience by exploring the grid. Would it be more efficient if the hunters played together and could communicate?

From [Melo and Veloso 2011], N-agent coordination is hard since the size of the state space grows exponentially in N. That's why they propose to limit the coordination scope by limiting interactions to local communication between agents.

The autonomic elements should communicate aboute the utility function to readjust it when the environment changes.

4. CREATING CONTEXT IN AUTONOMIC SYSTEMS

Context is "the circumstances that form the setting for an event, statement, or idea, and in terms of which it can be fully understood" (Oxford dictionary) Knowledge exists when data has meaning and *context* provides **meaning**.

> "The knowledge plane resides in a different space than the data and control space" [Dobson et al. 2010]

In this research, we use context as a tool to understand knowledge. The autonomic system's MAPE loop (see figure 1) uses knowledge as the core element of the loop, especially to analyze and plan. This way knowledge is kept as the autonomic element's experience. In RL the state space of all learned states is the data the autonomic element uses for better decision making. To reduce the state space we need to know what is relevent data by giving data meaning.

Context in Linguistics. From a pragmatic point of view, ”Context can be dynamically reshaped so it is also a generative principle” [Gold 2012]: it is the reaction to social phenomena and it should be defined and mapped.

Context in Computer science. Context is everything *but* the explicit input and output [Lieberman and Selker 2000]. In computer science, the context problem contradicts the black box idea, where output is completely determined by input.

Application’s history is part of context. Information about the computer system and connected networks can also be part of context. In addition users past actions are also part of context.

In autonomic systems the environment consists largely of other autonomic elements and events that affect each element separately and the system as a whole.

4.1. Creating Context by learning communication

How can autonomic elements create context ? One possibility presented in [Gal and McAllister 2014] for multi-agent systems is by creating a mapping that provides shared meaning to data. The mapping of communication is constructed as follows:

(1) an agent has a word map to features of an object
(2) agents learn each-other’s word-feature mappings
(3) the word-feature mapping is reinforced when both agents use the same word to identify a distinctive feature

Each autonomic element constructs its mapping and communicates with other elements in the system to continue the process of learning communication.

Learning to communicate- miscoordination can result from misinterpretation of messages exchanged [Goldman et al. 2007] that’s why it is important to continue to learn to communicate dynamically. A static map of communication features is likely to be limited and results in misunderstandings.

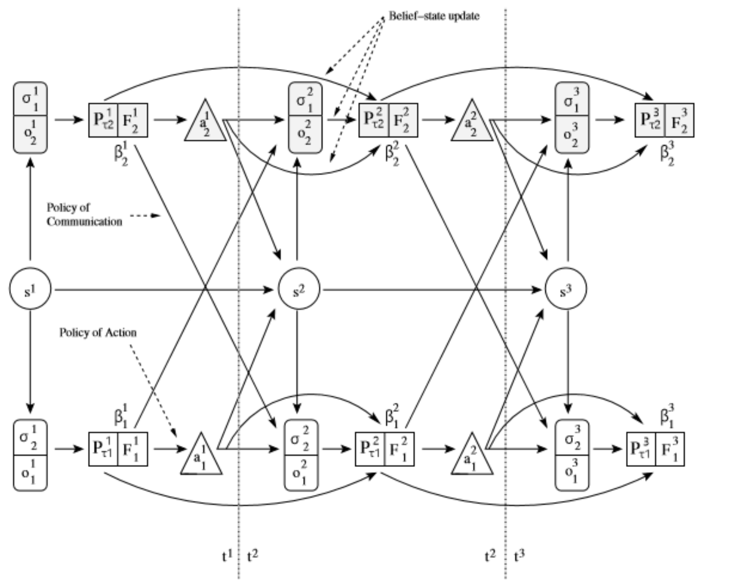
4.2. Communication policies

The general language-learning process from [Goldman et al. 2007] contains elementary parts as follows:

— State - global state of the environment at time t
— Observation - point of view of one autonomic element of the global state
— Message - communication exchanged between autonomic elements
— Belief-state :
    — Translation - understanding of messages (using meaning on data)
    — Belief-features - all other beliefs about the current state of the environement

To illustrate, figure 5 taken directly from the article, shows the transitions from state to state using policy of communication and policy of action. The important feature of [Goldman et al. 2007] is the notion of a translation. Learning communication is taking place in this system while acting, however, the action policy is a model-based MDP.

In the process the **state-action transitions** enables the system to transition from one state to another and will affect its environment. Furthermore, a **policy of action** and **policy of communication** will be used by each element in the system. The first for self-optimization and achieving the system goal and the latter for enabling better communication between autonomic elements.



s^i	State i	t^j	Time-step j
σ_i^j	Message j, sent by Agent i	o_i^j	Observation j of Agent i
a_i^j	Action j of Agent i	β_i^j	Belief-state j of Agent i
$P_{\tau i}^j$	Translation-distribution j of Agent i	F_i^j	Belief-features j of Agent i

figure.5

Fig. 5. A sketch of the language-learning process. [Goldman et al. 2007]

4.3. Communicate, collaborate, learn and optimize

An approach worth investigating is a parallel use of learning to communicate (collaborate) and learning to optimize (act).To illustrate, figure 6 shows an autonomic element in an autonomic system where it contains a local Q-function for learning to communicate and a local Q-function for learning best optimization action. It receives signals from peers for word mapping to continuously construct the context in which it acts. In addition, the autonomic elements receive messages about optimized actions found by its peers. After calculation of each Q-function, a new state is found. If the received signal was misunderstood or the optimization calculation yield a better result, the autonomic element advertises the information. Otherwise, each autonomic element continues learning independently till new signals and messages are received.

4.3.1. Why is learning to communicate and creating context difficult?

— Difficult to specify
— State space explosion
— Using RL in parallel is a challenge

Adaptation is a dynamic process. With this research we still have many unanswered questions and open paths. First we should implement this approach on a small system before scaling, to see how the state space is evolving.

An integral part of context is the element of 'out of context'. In this short article this subject will not be discussed, however 'out of context' can be translated in computer science to errors and exceptions that are not in the initial model.

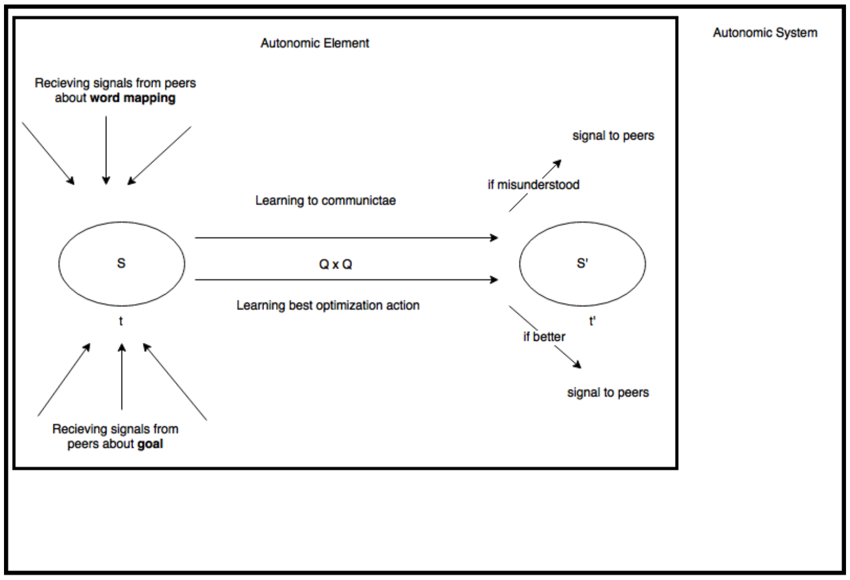

Autonomic System
Autonomic Element
Recieving signals from peers about **word mapping**
Learning to communictae
Q x Q
Learning best optimization action
S
t
S'
t'
if misunderstood
signal to peers
if better
signal to peers
Recieving signals from peers about **goal**

Fig. 6. Autonomic element using Q-function to learn and Q-function to optimize

5. FUTURE PERSPECTIVES AND CONCLUSION

In this article, we present a different approach to collaborative reinforcement learning in autonomic systems after describing the importance of the decision phase, existing implementation of RL algorithms and CRL algorithms. Creating context in autonomic systems, is a linguistic approach to decision making. RL isn't the best solution for self-optimization, but we can imagine CRL as a research opportunity.

(1) Does collaboration help the autonomic elements minimize independent effort?
(2) Can the system achieve convergence faster with CRL?
(3) Will learning communication affect decision quality? for better or for worse?

ACKNOWLEDGMENTS

The author would like to thank Dr. Jacques Malenfant of Pierre and Marie Curie University for providing articles and specifications about autonomic systems for this research paper and Leah Gruenpeter-Gold for all the insights about context and pragmatics.

REFERENCES

Simon Dobson, Roy Sterritt, Paddy Nixon, and Mike Hinchey. 2010. Fulfilling the Vision of Autonomic Computing. *Computer* 43, 1 (Jan. 2010), 35–41. DOI:http://dx.doi.org/10.1109/MC.2010.14

Jim Dowling, Raymond Cunningham, Anthony Harrington, Eoin Curran, and Vinny Cahill. 2005. Self-star Properties in Complex Information Systems. Springer-Verlag, Berlin, Heidelberg, Chapter Emergent Consensus in Decentralised Systems Using Collaborative Reinforcement Learning, 63–80. http://dl.acm.org/citation.cfm?id=2167575.2167582

Yarin Gal and Rowan McAllister. 2014. Emergent Communication for Collaborative Reinforcement Learning. *University of Cambridge, MLC RCC* (May 2014). http://mlg.eng.cam.ac.uk/rowan/files/EmergentCommunicationForCollaborativeReinforcementLearning.pdf

Leah Gruenpeter Gold. 2012. The Cobweb of Context ImagINe- (or out of)- Context. *Tel aviv University, Philosophy department* (2012).

Claudia V. Goldman, Martin Allen, and Shlomo Zilberstein. 2007. Learning to Communicate in a Decentralized Environment. *Autonomous Agents and Multi-Agent Systems* 15, 1 (2007), 47–90. http://rbr.cs.umass.edu/shlomo/papers/GAZjaamas07.html

Jeffrey O. Kephart and David M. Chess. 2003. The Vision of Autonomic Computing. *Computer* 36, 1 (Jan. 2003), 41–50. DOI:http://dx.doi.org/10.1109/MC.2003.1160055

Jeffrey O. Kephart and William E. Walsh. 2004. An Artificial Intelligence Perspective on Autonomic Computing Policies. In *POLICY*.

Jelle R. Kok and Nikos Vlassis. 2006. Collaborative Multiagent Reinforcement Learning by Payoff Propagation. *J. Mach. Learn. Res.* 7 (Dec. 2006), 1789–1828. http://dl.acm.org/citation.cfm?id=1248547.1248612

H. Lieberman and T. Selker. 2000. Out of Context: Computer Systems That Adapt to, and Learn from, Context. *IBM Syst. J.* 39, 3-4 (July 2000), 617–632. DOI:http://dx.doi.org/10.1147/sj.393.0617

Martina Maggio, Henry Hoffmann, Marco D. Santambrogio, Anant Agarwal, and Alberto Leva. 2011. Decision Making in Autonomic Computing Systems: Comparison of Approaches and Techniques. In *Proceedings of the 8th ACM International Conference on Autonomic Computing (ICAC '11)*. ACM, New York, NY, USA, 201–204. DOI:http://dx.doi.org/10.1145/1998582.1998629

Olga Melekhova and Jacques Malenfant. A Token-Based Scheme for Coordinating Decisions in Large-Scale Autonomic Systems. (????).

Francisco S. Melo and Manuela Veloso. 2011. Decentralized MDPs with Sparse Interactions. *Artif. Intell.* 175, 11 (July 2011), 1757–1789. DOI:http://dx.doi.org/10.1016/j.artint.2011.05.001

Ming Tan. 1998. Readings in Agents. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, Chapter Multi-agent Reinforcement Learning: Independent vs. Cooperative Agents, 487–494. http://dl.acm.org/citation.cfm?id=284860.284934

Gerald Tesauro. 2007. Reinforcement Learning in Autonomic Computing: A Manifesto and Case Studies. *IEEE Internet Computing* 11, 1 (Jan. 2007), 22–30. DOI:http://dx.doi.org/10.1109/MIC.2007.21

Received February 2017; revised ; accepted