Juggling Preferences in a World of Uncertainty

Conference Paper · September 2017

DOI: 10.1109/RE.2017.12

2 authors:

Luis-Hernán García-Paucar
Aston University
11 PUBLICATIONS **53** CITATIONS
SEE PROFILE

Nelly Bencomo
Aston University
117 PUBLICATIONS **3,024** CITATIONS
SEE PROFILE

Juggling Preferences in a World of Uncertainty

Luis H. Garcia Paucar, Nelly Bencomo
ALICE: Aston Lab for Intelligent Collectives
Aston University, UK
email: garciapl@aston.ac.uk, nelly@acm.org

Kevin Kam Fung Yuen
Research Institute of Big Data Analytics
Xi'an Jiaotong-Liverpool University, China
email: kevinkf.yuen@gmail.com

***Abstract*—[Context/Motivation] Decision-making for self-adaptive systems (SAS) requires the runtime trade-off of multiple non-functional requirements (NFRs) and the costs-benefits analysis of the alternative solutions. Usually, it requires the specification of weights for NFRs and decision-making strategies. Generally, these weights are defined at design-time with the support of previous experiences and domain experts. [Questions/Problems] Under some specific conditions detected at runtime, it can be the case that the weights assigned to the NFR at design time may not be suitable anymore at runtime. As a result, the system may not behave in the expected way and it may either execute unnecessary adaptations or miss crucial adaptations with a detrimental effect on the behaviour of the system. [New ideas/ early results] In this RE@Next! paper, we introduce a novel approach for automatic runtime reappraisal of the weights of NFRs given new evidence collected from the environment during the execution of the system. Our early results suggest, as expected, that the approach improves the decision-making process by allowing the reappraisal and update of the weights of the NFRs in accordance to the newly detected environmental context.**

***Index Terms*—Self-adaptation; decision making; AHP; P-CNP; non-functional requirements trade-off; uncertainty; runtime weights updating**

I. Introduction

In self-adaptive systems (SAS), the assumptions made at design time probably change at runtime causing changes on the initially defined utility weights (aka preferences or priorities). Furthermore, different weights may imply different decisions to be performed by the system. Different authors have approached these issues [1], [2], [3], [4], [5]. However, critical challenges need to be further explored. One of the issues is that current approaches focus only on the design-time activities and, even if effective, they are unlikely to be generalizable for other application domains [6], [7], [8], [9]. Further, the needs for uncovering relationships between NFRs and updating utility preferences during runtime have been neglected [10], [5]. We argue that tackling these challenges involve the role of preferences and the re-prioritization of NFRs due to new evidence found at runtime. We show in [11] that there have been important research efforts towards decision-making for SAS taking into account NFR weights. However, relevant results about dynamic reassessment and update of utility weights are still challenging. Some approaches use ad-hoc methods for collecting users' preferences, while others use techniques such as Multi Criteria Decision Analysis Methods (MCDA) [7], [6], [12]. Some MCDA based approaches, such as Primitive Cognitive Network Process (P-CNP) [13], [14], are used for the specification of NFR weights while some others such as Analytic Hierarchical Process (AHP) [15] are used for specifiying NFR weights and reasoning at runtime based on the prioritization of a set of alternative decisions [16]. In [6], [9], [17] the support for weight updating exists but it requires user intervention. For instance, authors of [9] suggested that if users do not agree with the final solution proposed by a model, they can revise the configuration values using an user interface. Authors in [17] suggested a user interface for manipulating threshold on preferences at runtime to modify the tactics suggested by the system. In [12], an autonomic preference tuning algorithm is used, while [18] proposed an approach for mining users' behaviour. However, this last approach is only applicable to one specific application domain and presents scalability issues. Given the identified research gap, the main contribution of this paper is ARRoW, a novel approach for the Automatic Runtime Re-appraisal and update of the Weights associated with NFRs. In this paper, ARRoW uses the decision-making process supported by conditional probabilities (Bayesian inference) based on Dynamic Decision Networks (DDNs) and Bayesian surprises [19]. For the reappraisal of weights, ARRoW uses Primitive Cognitive Network Process (P-CNP). P-CNP is an improved version of the Analytic Hierarchy Process (AHP) [15] that allows the update of NFRs weights during runtime and when the current weights are found not suitable anymore.

II. Background

This section overviews DDNs models, P-CNP and Bayesian Surprises. We briefly explain how these concepts are relevant to runtime decision-making in SASs while being able to reappraise weights.

A. DDN Models for Decision-Making in SAS

Dynamic Decision Networks (DDNs) are abstractions for reasoning over time about the world [20].

A DDN is a temporal model and is represented during several time slices. Fig. 1 shows a DDN where NFR denotes a set of state nodes, which are unobservable, and E denotes the observable evidence nodes. A DDN links decision maker preferences U (i.e. utility nodes), state and evidence nodes to make informed decisions D (i.e. decision nodes). In our approach, the state nodes will represent the satisfaction level of non-functional requirements. At design time, initial decision maker preferences (i.e. weights) are defined for each possible combination among state and decision nodes (See Table I).

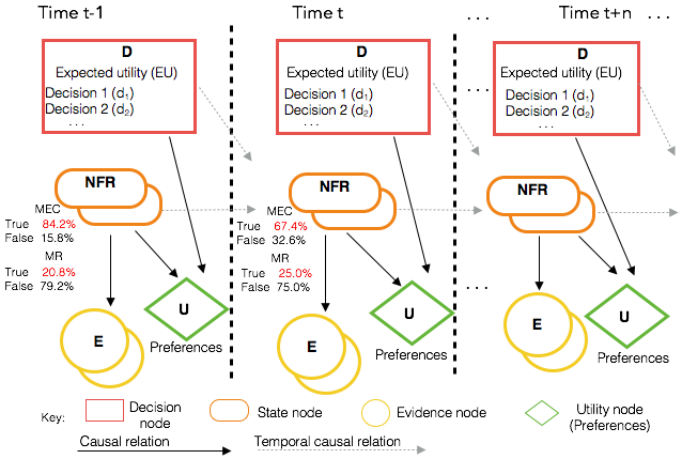

Fig. 1. Example of DDN Structure

These initial weights are the content of a utility node U and may need to be updated under some specific conditions detected at the runtime environment. To make a decision, a DDN computes the expected utility (EU) for each possible decision outcome $D = d_j$. The EU is computed using the equation 1 as follows:

$$EU(d_j|e) = \sum_{nfr_i \in NFR} U(nfr_i, d_j) \times P(nfr_i \mid e, d_{(t-1)}, nfr_{i(t-1)}) \quad (1)$$

The decision outcome $D = d_j$ with the highest expected utility is selected by the model for each time slice.

TABLE I
EXAMPLE OF UTILITY NODE WEIGHTS

A_i	d_j	MEC	MR	Weight
A1	d_1	T	T	0.1602
A2	d_1	T	F	0.1346
A3	d_1	F	T	0.1152
A4	d_1	F	F	0.0950
A5	d_2	T	T	0.1602
A6	d_2	T	F	0.1354
A7	d_2	F	T	0.1115
A8	d_2	F	F	0.0880

Key: T=True F=False d_j=Decision outcome j Ai=Alternative i

B. P-CNP for NFR preferences

P-CNP is a Multi-Criteria Decision Analysis (MCDA) method [21] to derive weights among several competing alternatives by using a comparison scale and paired comparisons [13].

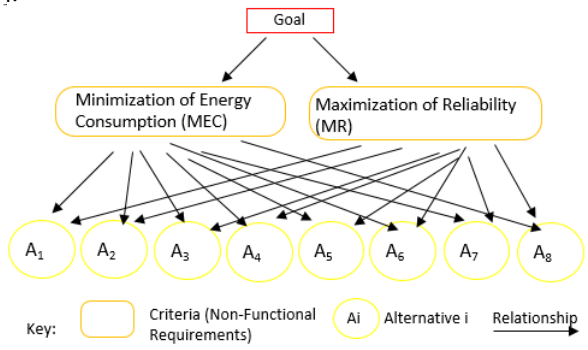

Fig. 2. Example of Structural Assessment Network (SAN)

P-CNP involves the following steps:

- Step 1: Problem cognition process. A decision problem is formulated in the form of a measurable Structural Assessment Network (SAN). A SAN is a model with the following elements: a *goal*, which in our context is the identification of weights for each possible combination of decision and state nodes, i.e., the content of a utility node U. A *criteria structure* (in our context the criteria are the NFRs of a SAS) and the *set of alternatives* $\{A_i\}$ to achieve the goal, i.e., the concrete entries of a utility node U. Fig. 2 shows an example of an SAN model.
- Step 2: Weights assessment of criteria. The comparison and prioritization of the criteria, which were identified in the previous step, are performed. *The weights assessment* among NFRs is performed by using paired comparisons. For the example in Fig. 2, the NFRs are *Minimization of Energy Consumption* (MEC) and *Mazimization of Reliability* (MR). As a result of comparisons, a Pairwise Opposite Matrix (POM) B is formed. A POM is a matrix that represents the relative importance between each pair of criteria. Fig. 3 shows a feasible POM after the comparison between the criteria MEC and MR.

$$B = [b_{ij}] = \begin{array}{c} \\ \text{MEC} \\ \text{MR} \end{array} \begin{array}{c} \begin{array}{cc} \text{MEC} & \text{MR} \end{array} \\ \begin{bmatrix} 0 & 3 \\ -3 & 0 \end{bmatrix} \end{array} \quad \begin{array}{c} \textbf{Weights} \\ 0.5938 \\ 0.4062 \end{array}$$

Fig. 3. POM for Weights assessment

The values b_{ij} correspond to the values of a measurement scale schema in P-CNP [22]. The value 3 represent that MEC is 'more important' than MR. When $i = j$, $b_{ij} = 0$. In a POM, the comparisons only fill the upper triangular matrix, the lower triangular matrix is given by the opposite of the upper triangular matrix such that $b_{ij} = -b_{ji}$.
- Step 3: Alternatives assessment with respect to criteria. *The alternatives assessment* is performed by using paired comparison between the alternatives A_i with respect to each criterion. In Fig. 2, eight alternatives A_i are shown. For each criterion a POM must be derived. The results of this step are POMs with the relative weights for the alternatives A_i in relation to each criterion (i.e., in our example: MEC and MR).
- Step 4: Information fusion and ranking. Taking the weights from steps 2 and 3, a new POM with the global weights for each alternative A_i is computed by using the weighted arithmetic mean [13], [14] . The values are usually re-escaled to [0,1]. The final weights for the alternatives in Fig. 2 are: $A_1 = 0.1602$, $A_2 = 0.1346$, $A_3 = 0.1152$, $A_4 = 0.0950$, $A_5 = 0.1602$, $A_6 = 0.1354$, $A_7 = 0.1115$ and $A_8 = 0.0880$.
These weights correspond to the entries of a utility node U in a DDN (see Table I) and may need to be updated at runtime, given new information collected from the environment.

C. Surprises to Identify Deviations from Expected Behaviour

A Bayesian surprise value means that the evidence provided from the environment leads to a difference between the prior and posterior probabilities of an event [23], [19]. A Bayesian surprise measures how observed data modify

assumptions of the world during runtime [19]. In Equation 2, the surprise S represents the divergence between the prior and posterior distributions of a NFR and is calculated by using the Kullback-Leibler divergence (KL) [19]. Therefore, given a non-functional requirement NFR, and E representing the evidence provided by the properties monitored as variables in the execution environment, P(NFR) is the prior probability distribution of the non-functional requirement NFR being partially satisficed and P($NFR|E$) is the posterior probability distribution of the NFR being partially satisficed given the evidence E.

$$S(NFR, E) = KL(P(NFR|E), P(NFR)) = \sum_{nfr_i \epsilon NFR} P(nfr_i|E) \log \frac{P(nfr_i|E)}{P(nfr_i)} \quad (2)$$

For instance, in Fig. 1, we observe a prior distribution of $84.2\% = True$ and $15.8\% = False$ for MEC, but after one time slice and an evidence E, our posterior distribution is $67.4\% = True$ and $32.6\% = False$. The divergence between this two distributions will represent a Bayesian Surprise.

III. ARRoW: Automatic Runtime Re-appraisal and Updating of NFR Weights

A. Upper and lower bounds for NFRs

On the basis of information provided by the sytem's experts, we have identified upper and lower bounds of the satisfaction level for each NFR (See Table II). The upper and lower bounds previously identified allow us to classify the satisfaction level for each NFR into three categories: *poor*, *suitable*, and *optimal*. The satisfaction level for each NFR is monitored at runtime and based on its monitored values is determined how important it is. NFRs with *poor* values are more important than NFRs with *suitable* values and NFRs with *suitable* values are more important than NFRs with *optimal* values.

TABLE II
Categories for the satisfaction of NFRs

NFR Ranges of Satisfaction (%)	[0%,lower bound%)	[lower bound%, upper bound%)	[upper bound%,100%)
Categories of Monitored Values	Poor	Suitable	Optimal

B. The proposed approach

ARRoW is applied over different time slices in a SAS to help the system to improve its behaviour with reappraising and updating unsuitable weights of NFRs and entries of a utility node U detected during its operation at runtime. The approach involves the following steps:

- Step 1: Detecting changes on the environment. Compute a Bayesian Surprise for each state variable (i.e. each NFR) of a DDN representing a SAS. This process is performed at runtime and for each time slice. If a surprise is detected and the satisfaction value of the NFR is within the *poor* category, the following step will be continued.
- Step 2: Balancing weights of NFRs. The monitored level of satisfaction of the NFRs, are compared with the categories: *poor*, *suitable* and *optimal* by using an scale of comparison derived from P-CNP (See Table III). The highest weights are assigned to NFRs with monitored values of satisfaction within the *poor* category. For instance, if the monitored values of satisfaction for the NFRs in Fig. 2 are: "MR within the *optimal* category and MEC within the *poor* category", after applying the scale of comparison in Table III, the results will show that MEC is "strongly over" MR given their runtime satisfaction values. In this example, b_{12} is 5 where b_1 is MEC and b_2 is MR.

TABLE III
Pairwise scale of comparison. Adapted from [13]

b_{ij}	Runtime scale of Comparison for NFRs
5	i is strongly over j
3	i is fairly over j
0	i and j are equal
-3	j is fairly over i
-5	j is strongly over i

B = [b i j] =

	MEC	MR	Weights
MEC	0	5	0.6562
MR	-5	0	0.3438

Fig. 4. POM for criteria: non-functional requirements MEC and MR

This step corresponds to step 2 of P-CNP and we obtain as a result a new POM POM (See Fig. 4). Next, we apply steps 3 and 4 of P-CNP to obtain the global weights of alternatives $\{A_i\}$, which are the entries of an utility node U of a DDN. Table I shows an example of these weights which are further used by a DDN model to compute the expected utility EU for each decision outcome $D = d_j$ and to choose the final decision for each time slice.

IV. Experiments

The experiments presented in this paper are based on the application of ARRoW to a real case study: a Remote Data Mirroring (RDM) system [11]. Nonetheless, an initial version of the approach has also been applied to an Ambient Assisted Living (AAL) application [23]. In both cases, the results confirm the validity of our initial arguments.

A. Introduction to RDM Case Study

An RDM is a technique with the goal of protecting data against inaccessibility and to provide further resistance to data loss [24]. An RDM can be configured in different ways, for instance, in terms of the network's topology, minimum spanning tree (MST) vs. redundant topology (RT). An RT topology offers a higher level of reliability than an MST topology. However, the costs of maintaining a non-stop redundant topology may be prohibitive according to given contexts. An assessment of the trade-off between these two choices need to be made at design-time and revisited at runtime under the light of new evidence found. In this paper, we will focus on the need of the reassessment of the initial weights at runtime.

B. Initial Setup

A DDN for the application of RDM was designed using two alternatives for network topologies: MST and RT as described above. Each configuration provides its own levels of reliability and energy costs which are taken into account in the NFRs

TABLE IV
UTILITY NODE INITIAL WEIGHTS - RDM CASE STUDY

Ai	Topology	MEC	MR	Weight
A1	MST	T	T	0.1602
A2	MST	T	F	0.1300
A3	MST	F	T	0.1191
A4	MST	F	F	0.0945
A5	RT	T	T	0.1606
A6	RT	T	F	0.1312
A7	RT	F	T	0.1161
A8	RT	F	F	0.0883

Key: Ai=Alternative i MR=Maximization of Reliability
T=True F=False MEC=Minimization of Energy Consumption

TABLE V
CATEGORIES OF NFRS - RDM CASE STUDY

Non Functional Requirements (NFRs)	Ranges of Satisfaction levels of NFRs (%)		
	Poor	Suitable	Optimal
Minimization of Energy Consumption (MEC)	[0%,53%)	[53%,70%)	[70%,100%]
Maximization of Reliability (MR)	[0%,55%)	[55%,72%)	[72%,100%]

Maximization of Reliability (MR) and Minimization of Energy Consumption (MEC).

1) Scenario and initial preferences: Based on information provided by the domain experts, the scenario is described as follows: the states of two monitored variables REC="Ranges of Energy Consumption" and NCC="Number of Connected Components" are monitored during runtime. Whilst REC can be directly observed by sensors, NCC must be computed with a reachability algorithm [25] that examines which RDM nodes can be reached by traversing active network links [24].

Initial preferences are represented by the utility node U of the DDN. The set of initial weights are given in Table IV.

2) Upper and lower bounds for NFRs: The information shown in Table V was provided by the domain experts, and corresponds to the identified values for the categories: *poor*, *suitable* and *optimal*. These ranges represent the satisfaction levels of the NFRs, Minimize Energy Consumption (MEC) and Maximize Reliability (MR) and are used to determine at runtime the importance of one NFR in relation to the others.

C. Experiments for RDM Case Study

Two specific situations where weights update is needed have been tackled in this case study. For each situation, surprises have been calculated between the prior and posterior distributions of the NFRs MEC and MR, reacting accordingly.

1) Situation 01: Surprises and unneeded adaptation: In time slice 3, we can observe two surprises (i.e., S1 and S2) and that an adaptation is suggested from RT to MST topology (See Table VII, column Topology without ARRoW). Studying the conditional probabilities provided by the DDN under the current conditions (Fig. 5, time slice 3):

- P($MEC = true | RECin[x, y)$) = 64.7% and
- P($MR = true | NCC < r$) = 33.6 %

We observe that while the probability for Maximization of Reliability is within the *poor* category, the probability for Minimization of Energy Consumption is within the *suitable* category. The selected choice, i.e. to adapt from RT to MST,

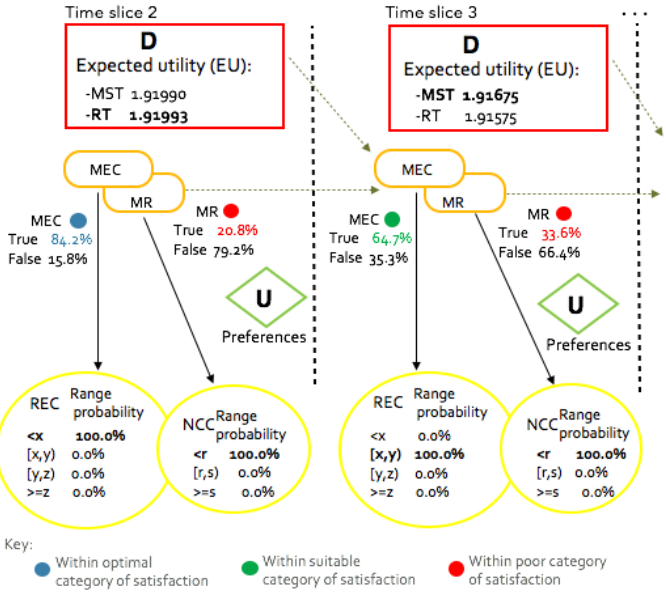

Fig. 5. Situation 01: Surprises and unneeded adaptation

TABLE VI
UTILITY NODE UPDATED WEIGHTS (SITUATION 01) - RDM

Ai	Topology	MEC	MR	Weight
A1	MST	T	T	0.1602
A2	MST	T	F	0.1161
A3	MST	F	T	0.1309
A4	MST	F	F	0.0930
A5	RT	T	T	0.1617
A6	RT	T	F	0.1188
A7	RT	F	T	0.1300
A8	RT	F	F	0.0894

Key: Ai=Alternative i MR=Maximization of Reliability
T=True F=False MEC=Minimization of Energy Consumption

certainly may not be the best choice: using MST topology would improve Minimization of Energy Consumption, which already is within the *suitable* category, at the expense of Maximization of Reliability, which is currently within the *poor* category and needs to be improved. ARRoW improves this behaviour. Given that we have a surprise and a satisfaction value within the *poor* category (first step of our approach), the next step is:

- Balancing weights of NFRs. Under the current context: MEC within the *suitable* category and MR within the *poor* category, we obtain the Pairwise Opposite Matrix (POM) shown in Fig. 6. As a result of applying the scale in Table III, we observe that b_{12} is -3 where b_1 is MEC and b_2 is MR. The results show that MR "is fairly over" MEC given their runtime values. This step corresponds to step 2 of P-CNP. Next, after applying steps 3 and 4 of P-CNP, a set of new weights for the utility node U of a DDN are computed and shown in Table VI.

$$B = [b_{ij}] = \begin{array}{c} \\ \text{MEC} \\ \text{MR} \end{array} \begin{array}{c} \begin{array}{cc} \text{MEC} & \text{MR} \end{array} \\ \begin{bmatrix} 0 & -3 \\ 3 & 0 \end{bmatrix} \end{array} \begin{array}{c} \text{Weights} \\ 0.4062 \\ 0.5938 \end{array}$$

Fig. 6. POM for criteria - RDM Case Study - Situation 01

TABLE VII
SURPRISES AND MONITORED VALUES - RDM CASE STUDY

Time Slice (TS)	Adaptation	S1 (MEC)	S2 (MR)	REC monitored values	NCC monitored values	Chosen Topology without ARRoW	Chosen Topology with ARRoW
1	-	0	0	REC in [x, y)	NCC>=s	MST	MST
2	Yes	0.04962	0.83396	REC<x	NCC<r	RT	RT
3	Yes	0.16349	0.06360	REC in [x, y)	NCC<r	MST	RT
4		0	0	REC in [x, y)	NCC<r	MST	RT
5		0	0	REC in [x, y)	NCC<r	MST	RT
6	-	0	0.88634	REC in [x, y)	NCC>=s	MST	MST
7		0	0	REC in [x, y)	NCC>=s	MST	MST
8		0	0	REC in [x, y)	NCC>=s	MST	MST
9		0	0	REC in [x, y)	NCC>=s	MST	MST
10	-	0	0.83396	REC in [x, y)	NCC<r	MST	RT
11		0	0	REC in [x, y)	NCC<r	MST	RT
12		0	0	REC in [x, y)	NCC<r	MST	RT
13		0	0	REC in [x, y)	NCC<r	MST	RT

Finally, using the new weights and the Equation 1 in section II-A, the DDN computes the expected utility EU for each possible decision: $d_1 = MST$ topology and $d_2 = RT$ topology. The obtained values are $MST = 1.91245$ and $RT = 1.91262$. These values will replace the observed values in Fig. 5, time slice 3. At this moment, the choice with the highest expected utility is RT topology, therefore the new suggested decision by the DDN for time slice 3 is not to be adapted, and RT Topology will continue to be used. (See Table VII, time slice 3, column Topology with ARRoW).

2) Situation 02: Surprises and needed adaptation: We can observe that in time slice 10 there is a surprise but the DDN has not suggested any adaptation (See Table VII). Given the conditional probabilities under the current context:

- P($MEC = true|RECin[x, y)$)=73.2% and
- P($MR = true|NCC < r$) = 20.8 %

We observe that the probability for Minimization of Energy Consumption is within the *optimal* category, and the probability for Maximization of Reliability is within the *poor* category (see Table V). The selected choice, i.e. not to be adapted, certainly may not be the best choice: continuing using MST topology would improve Minimization of Energy Consumption, which already is within the highest possible category, at the expense of Maximization of Reliability, which is currently within the *poor* category. Equivalent to the situation 01 and after applying ARRoW, the new suggested decision by the DDN for time slice 10 is to adapt from MST to RT topology (See Table VII, time slice 10, column Topology with ARRoW).

D. Analysis of Results

By updating original weights when necessary, we have shown how ARRoW can improve the decision making and behaviour of a self-adaptive application. The approach was successfully applied on two specific situations where there is need for reappraising the initial weights or preferences. Fig. 7 and 8 show the satisfaction level over time of the NFRs MR and MEC respectively. In Fig. 7, in time slice 3, in the case of ARRoW is applied, we observe further reduction of the satisfaction level of MR (i.e. from 33.6% to 20.8%) has been

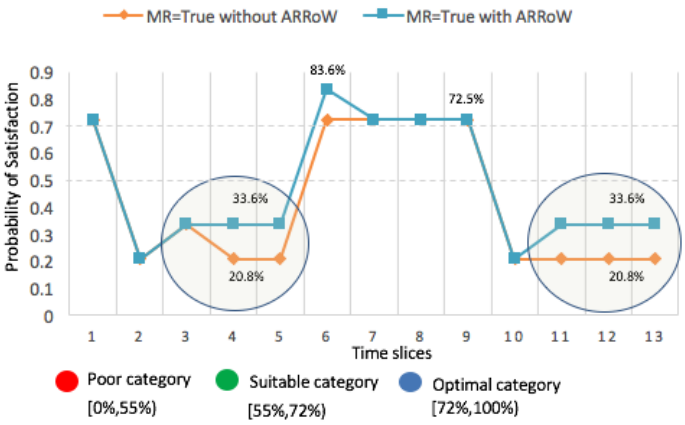

Fig. 7. Satisfaction level of Maximization of Reliability (MR)

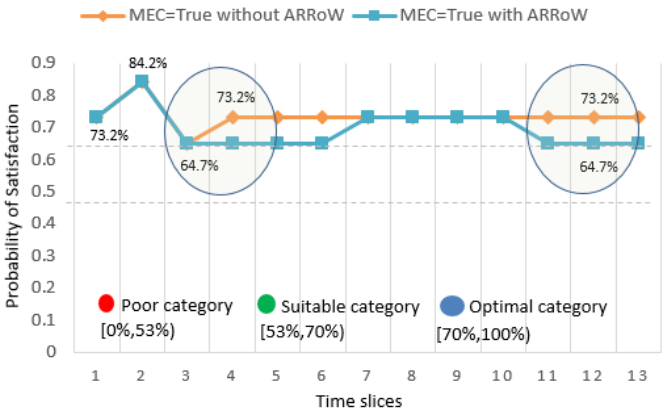

Fig. 8. Satisfaction level of Minimization of Energy Consumption (MEC)

prevented. Further, later at time slice 6, the satisfaction level was improved and was even taken to the optimal category (i.e. 83.6%) and maintained within until time slice 9. In time slice 10, changes in the environment, (dictated by the scenario described in Table VII), provoked a reduction of the satisfaction level (i.e. to 20.8%). However, when applying ARRoW, and different from when ARRoW is not applied, the satisfaction level is recovered (i.e. during time slices 11, 12 and 13). In the case of Fig. 8, in time slice 3, when ARRoW was applied, we observe the satisfaction level of MEC was maintained within a suitable category (i.e. 64.7%). Then, at time slice 6, the satisfaction level was improved to the optimal category (i.e. 73.2%) and maintained within it until time slice 9. Finally, in time slice 10, changes in the environment (see again Table VII) provoked a reduction of the satisfaction level (i.e. to 64.8%) but still within the suitable category (i.e. time slices 11, 12 and 13).

From the analysis above, we can conclude that ARRoW can improve the satisfaction levels of some NFRs (i.e from *poor* to *suitable* or to even *optimal*) in exchange of slight degradation of others (i.e from *optimal* to *suitable*) when there are trade-off among NFRs. However, an alternative and feasible result could be to improve the satisfaction level of some NFRs and to maintain the current satisfaction level of others. Ultimately, ARRoW would offer a better informed decision-making process.

V. Conclusions

We have argued that failing to re-appraise and update the weights of NFRs when there is runtime evidence that the original weights (which were determined at design-time) do not agree with the newly discovered contexts, can cause unexpected results or degrade the behaviour of the system. This paper has presented the novel approach ARRoW, for the re-appraisal and further update of the weights associated with NFRs according to evidence gathered from the operational environmental at runtime. The result is the facilitation of a better-informed decision-making process based on evidence collected at runtime. Currently, to our knowledge, there are not definitive research results related to the specific issue of dynamic update of weights; with some few exceptions such as [18], [12], [9] which, different from ARRoW, focus on specific domains. ARRoW, as used in this paper, inherits the scalability issues of the DDNs therefore, we plan to use other decision-making mechanisms which not necessarily include the use of DDN to therefore improve the scalability of the approach. Specifically, we are exploring Approximate POMDP Planning Toolkit (APPL) [1]. APPL is a set of C++ algorithms for solving Partially Observable Markov Decision Processes (POMDPs) [26], [27]. By using an efficient online POMDP solver, DESPOT [27], we are going to implement new case studies and experiments involving a bigger number of NFRs, monitoring variables and possible decisions.

Further, there is potential use of the proposed category $suitable$ to implement RELAX by RELAXing one NFR (accepting that the NFR goes from $optimal$ to $suitable$) for the benefit of another NFR during a given time. This can be done due to the fact that ARRoW allows the identification of situations where such a RELAXation ($optimal$ to $suitable$) is possible or available. We are also working on new real case studies. We are working with an IT International Peruvian Company (http://www.bitperfect.pe) to apply our approach as an extension of the case study presented in [23] given certain similarities. We believe that ARRoW is novel and has potentially significant implications for RE research in specific areas such as self-adaptive, autonomous and self-aware systems. It is critically important to solicit feedback from the RE community and create awareness of the importance of the research topic of the need of weights reassesment for future editions of the RE Conference. Finally, we hope that by promoting ARRoW at RE 2017, we may be able to identify collaborators for our future work in this area.

References

[1] M. Salehie and L. Tahvildari, "Self-adaptive software: Landscape and research challenges," *ACM Trans. Auton. Adapt. Syst.*, vol. 4, no. 2, pp. 14:1–14:42, May 2009.

[2] B. H. Cheng and et al., "Software engineering for self-adaptive systems." Berlin, Heidelberg: Springer-Verlag, 2009, ch. Software Engineering for Self-Adaptive Systems: A Research Roadmap, pp. 1–26.

[3] E. Yuan, N. Esfahani, and S. Malek, "A systematic survey of self-protecting software systems," *ACM Trans. Auton. Adapt. Syst.*, vol. 8, no. 4, pp. 17:1–17:41, Jan. 2014.

[4] M. Salama, R. Bahsoon, and N. Bencomo, "Managing trade-offs in self-adaptive software architectures: A systematic mapping study," in *Managing trade-offs in adaptable software architectures*, I. Mistrk, N. Ali, J. Grundy, R. Kazman, and B. Schmerl, Eds. Elsevier, 2016.

[5] C. Krupitzer, F. M. Roth, S. VanSyckel, G. Schiele, and C. Becker, "A survey on engineering approaches for self-adaptive systems," *Pervasive and Mobile Computing*, vol. 17, Part B, pp. 184 – 206, 2015.

[6] G. Elahi and E. Yu, "Requirements trade-offs analysis in the absence of quantitative measures: A heuristic method," ser. SAC '11. New York, NY, USA: ACM, 2011, pp. 651–658.

[7] E. Letier, D. Stefan, and E. T. Barr, "Uncertainty, risk, and information value in software requirements and architecture," in *Proceedings of ICSE*, ser. ICSE 2014. NY, USA: ACM, 2014, pp. 883–894.

[8] S. Liaskos, S. A. McIlraith, S. Sohrabi, and J. Mylopoulos, "Representing and reasoning about preferences in requirements engineering," *Requir. Eng.*, vol. 16, no. 3, pp. 227–249, Sep. 2011.

[9] H. Song, S. Barrett, A. Clarke, and S. Clarke, "Self-adaptation with end-user preference: Using run-time models and constraint solving," in *the Intrl. Conference MODELS*, USA, 2013.

[10] N. Bencomo, "Quantun: Quantification of uncertainty for the reassessment of requirements," in *23rd IEEE International Requirements Engineering Conference, RE*, 2015, pp. 236–240.

[11] L. H. G. Paucar and N. Bencomo, "The reassessment of preferences of non-functional requirements for better informed decision-making in self-adaptation," *AIRE - 3rd International Workshop*, 2016.

[12] X. Peng, B. Chen, Y. Yu, and W. Zhao, "Self-tuning of software systems through goal-based feedback loop control," in *Requirements Engineering Conference (RE)*, Sept 2010, pp. 104–107.

[13] K. Yuen, "Cognitive network process with fuzzy soft computing technique in collective decision aiding," *The Hong Kong Polytechnic University, PhD thesis*, 2009.

[14] Y. K.K.F., "The pairwise opposite matrix and its cognitive prioritization operators: the ideal alternatives of the pairwise reciprocal matrix and analytic prioritization operators," *Journal of the Operational Research Society*, 2012.

[15] T. Saaty, "Decision making with the analytic hierarchy process," *Inter. Journal of Services Sciences,*, 2008.

[16] L. Pimentel, D. Ros, and M. Seruffo, "Wired/Wireless Internet Communications," vol. 8458, pp. 122–135, 2014.

[17] J. P. Sousa, R. K. Balan, V. Poladian, D. Garlan, and M. Satyanarayanan, "User guidance of resource-adaptive systems," in *In Proc. of International Conference on Software and Data Technologies*, 2008.

[18] J. García-Galán, L. Pasquale, P. Trinidad, and A. Ruiz-Cortés, "User-centric adaptation of multi-tenant services: Preference-based analysis for service reconfiguration," in *SEAMS*, ser. SEAMS 2014, 2014.

[19] N. Bencomo and A. Belaggoun, "A world full of surprises: bayesian theory of surprise to quantify degrees of uncertainty," in *ICSE*, 2014, pp. 460–463.

[20] S. J. Russell and P. Norvig, *Artificial intelligence - a modern approach: the intelligent agent book*, ser. Prentice Hall series in artificial intelligence. Prentice Hall, 1995.

[21] J. Figueira, S. Greco, and M. Ehrogott, *Multiple Criteria Decision Analysis: State of the Art Surveys*. Springer, 2005.

[22] K. K. F. Yuen, "The Primitive Cognitive Network Process in healthcare and medical decision making: Comparisons with the Analytic Hierarchy Process," *Applied Soft Computing Journal*, vol. 14, no. PART A, pp. 109–119, 2014.

[23] L. H. G. Paucar and N. Bencomo, "Runtime Models Based on Dynamic Decision Networks : Enhancing the Decision-making in the Domain of Ambient Assisted Living Applications," *MRT, France*, 2016.

[24] A. Ramirez, B. Cheng, N. Bencomo, and P. Sawyer, "Relaxing claims: Coping with uncertainty while evaluating assumptions at run time," *MODELS*, 2012.

[25] R. Tarjan, "Depth-first search and linear graph algorithms." *Proceedings of the 12th Annual Symposium on Switching and Automata Theory. pp. 114121.*, 1971.

[26] D. H. H. Kurniawati and W. Lee, "Sarsop: Efficient point-based pomdp planning by approximating optimally reachable belief spaces," *In Proc. Robotics: Science and Systems*, 2008.

[27] N. Ye, A. Somani, D. Hsu, and W. S. Lee, "Despot: Online pomdp planning with regularization," *Journal of Artificial Intelligence Research 58 (2017) 231-266*, 2017.

[1] http://bigbird.comp.nus.edu.sg/pmwiki/farm/appl/index.php

.