Constructing Optimal Fuzzy Metric Trees for Agent Performance Evaluation

Christos Dimou, Manolis Falelakis, Andreas L. Symeonidis,
Anastasios Delopoulos and Pericles A. Mitkas
Dept. of Electrical and Computer Engineering, Aristotle University of Thessaloniki, Greece

Abstract

The field of multi-agent systems has reached a significant degree of maturity with respect to frameworks, standards and infrastructures. Focus is now shifted to performance evaluation of real-world applications, in order to quantify the practical benefits and drawbacks of agent systems. Our approach extends current work on generic evaluation methodologies for agents by employing fuzzy weighted trees for organizing evaluation-specific concepts/metrics and linguistic terms to intuitively represent and aggregate measurement information. Furthermore, we introduce meta-metrics that measure the validity and complexity of the contribution of each metric in the overall performance evaluation. These are all incorporated for selecting optimal subsets of metrics and designing the evaluation process in compliance with the demands/restrictions of various evaluation setups, thus minimizing intervention by domain experts. The applicability of the proposed methodology is demonstrated through the evaluation of a real-world test case.

1 Introduction

Although agents and Multi-agent systems (MAS) have gained their position as an attractive programming paradigm for a broad range of application domains, software practitioners are still reluctant in incorporating agent solutions to solve real-world problems [6], thus delaying or even impeding the development of engineering methodologies that tackle all stages of the agent software development life cycle. In order to prove agent "competence", focus is now given on the evaluation of the quantitative advantages and drawbacks that agents and MAS exhibit at runtime. Indeed, in the field of Intelligent Systems (agents and MAS being a special case of), there has been a significant activity towards defining methods and metrics for evaluating aspects of emerging and intelligent behavior [1].

This paper extends the Agent Performance Evaluation (APE) methodology [3], a generic methodology that provides a comprehensive structure for organizing and using measurement-related information. The contribution of this work is twofold. First, all crisp, numeric representations of measurement information are replaced with fuzzy sets and fuzzy operations, making the performance characterizations more intuitive to human evaluators. Furthermore, meta-metrics are introduced in order to measure the validity and complexity of the contribution of each metric in the overall performance evaluation (Section 2). The presented methodology is demonstrated and validated through the Trading Agent Competition - Supply Chain Management (TAC/SCM) [2] game (Section 3).

2 Fuzzy Metrics Representation Trees

A Metrics Representation Tree (MRT) is a structured representation for the organization of metrics and measurement information. The tree is organized in layers that consist of *Simple Metrics* and *Composite Metrics* that correspond to the directly measurable aspects of the system and higher level evaluation concepts, respectively. An overview of the structure of a MRT is depicted in Figure 1.

2.1 Simple and Composite Metrics

A Simple Metric $y_i(m) \in [0, 1]$ is defined as a fuzzy set on the measurable aspect *m*. For example the Simple Metric "*high avg lead time*" is defined on the measurable aspect "*avg lead time*", expressed in *days*. It is derived that the membership value $\mu_{Y_i} = y_i(m)$ corresponds to the degree to which that particular agent assumes property $Y_i = y_i(m)$. Finally, we assume a finite set $Y = \{Y_i\}$, to denote the entire collection of Simple Metrics.

A Composite Metric E_k describes a performance aspect of the system that corresponds to a higher level concept that cannot be directly assigned a measured value collected through experiments. For all Composite Metric applies that $E_k \in E$, where E is the set of all Composite Metrics. Each

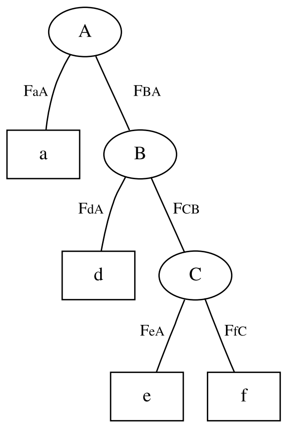

Figure 1. A Metrics Representation Tree

Composite Metric is composed of other lower level Simple or Composite Metrics.

Using the above descriptions, we can associate hierarchically Simple and Composite Metrics. To do so, we incorporate weights $F_{S_i E_k} \in [0,1]$ that signify the degree of contribution of metric S_i to the definition of E_k, which can be formulated as a discrete fuzzy set of the form:

$$E_k = F_{1k}/S_1 + F_{2k}/S_2 + \ldots + F_{nk}/S_n \tag{1}$$

where $F_{ik} = F_{S_i E_k}$ and $S_i \in S = Y \cup E$.

Composite Metrics that consist exclusively of Simple Metrics follow the detailed definition:

$$E_k = F_{1k}/Y_1 + F_{2k}/Y_2 + ... + F_{nk}/Y_n \tag{2}$$

where $Y_i \in Y$.

2.2 Meta-metrics

Three meta-metrics are defined in order to quantify the evaluation procedure: *Certainty*, *Validity* and *Complexity*.

Certainty is defined as the degree to which a Composite Metric exists in the MAS system under evaluation. Given the detailed definition of Eq. 2 and the membership degrees μ_{Y_i} of the Simple Metrics Y_i, *Certainty* is defined as:

$$\mu_{E_k} \triangleq \underset{i}{\mathcal{U}}(\mathcal{I}(F_{Y_i E_k}, \mu_{Y_i})), \tag{3}$$

where $\mathcal{U}$ and $\mathcal{I}$ are the operators for fuzzy union and fuzzy intersection respectively.

Validity of a Complex Metric E_k is defined as:

$$\mathcal{V}(E_k) \triangleq \underset{i}{\mathcal{U}}(F_{Y_i E_k}) \tag{4}$$

Validity describes the maximum possible value of μ_{E_k} and thus the maximum amount of information that a definition can provide.

Finally, *Complexity* describes the amount of time required to acquire the measurement value for a Metric E_k. It is defined as:

$$\mathcal{C}(E_k) = \sum_i c(t_i) \tag{5}$$

Validity and Complexity are independent from the measurement results and are computed prior to the identification. In contrast, Certainty depends on the specific measurement values, assigned to Simple Metrics and is, therefore, be computed after the execution of the experiments.

2.3 Inference

Once the measured values for all Simple Metrics have been collected, inference occurs in order to calculate the values of higher level Composite Metrics, by combining lower level metrics using union operations.

The first step is to transform the tree into a *detailed definition*, ie, a single-level tree which defines a composite entity based solely on simple metrics. This process is essentially the computation of the weights of the new tree. This is done by gradually substituting each composite metric with it's sub-nodes using fuzzy unions and intersections. Consider the example of Figure 1. The corresponding detailed definition of composite metric A is

$$A = F'_{aA}/a + F'_{dA}/d + F_{eA}/e + F_{fA}/f$$

where the new weights are computed as $F'_{aA} = F_{aA}$, $F'_{dA} = \mathcal{I}(F_{dB}, F_{BA})$, $F_{eA} = \mathcal{I}(F_{eC}, F_{CB}, F_{BA})$ and $F_{fA} = \mathcal{I}(F_{fC}, F_{CB}, F_{BA})$.

After computing the detailed definition weigths, we use Equation 3 to fully evaluate the Composite Metric. This inference procedure can be conveniently modeled using fuzzy relations (see [4]).

2.4 Partial Evaluation

In order to provide the evaluator designer with tools to select optimal subsets of Simple Metrics for saving computational cost, we define *Partial Certainty, Partial Validity* and *Partial Complexity*.

For a subset $\mathbf{A}$ of the set of Simple Metrics that contribute to the definition E_k, *Partial Certainty* denotes the confidence we have acquired that E_k exists in a data set by evaluating only the properties in $\mathbf{A}$. It is defined as:

$$\mu_{E_k}(\mathbf{A}) \triangleq \underset{Y_i \in \mathbf{A}}{\mathcal{U}}(\mathcal{I}(F_{Y_i E_k}, \mu_{Y_i})) \tag{6}$$

Similarly, measures the "quality" of the set $\mathbf{A}$ in the identification of E_k *Partial Validity* and is defined as

$$\mathcal{V}(E_k/\mathbf{A}) \triangleq \underset{Y_i \in \mathbf{A}}{\mathcal{U}}(F_{Y_i E_k}). \tag{7}$$

Finally, *Partial Complexity* is defined as:

$$\mathcal{C}(E_k/\mathbf{A}) \triangleq \sum_{i \in \mathbf{A}} c(t_i) \tag{8}$$

2.5 Optimal Subset Selection

Given a Complexity threshold C_T for the identification of a Composite Metric E_k, we select those subsets of $\mathbf{S}_{E_k}$, labeled $\mathbf{A}_i$ that satisfy the complexity criterion:

$$\mathcal{C}(E_k/\mathbf{A}_i) \leq C_T \tag{9}$$

The optimal set $\mathbf{A}^*_T \in \mathbf{A}$ maximizes the Validity meta-metric:

$$\mathcal{V}(E_k/\mathbf{A}^*_T) = max(\mathcal{V}(E_k/\mathbf{A}_i)) \tag{10}$$

Trading-off between Complexity and Validity is a decision problem that is solved based on time limitations put by the designer. The optimal subset must identified within a total of 2^k subsets, for a MRT with k Simple Metrics. The calculation of such a number of subset would be inefficient given time constraints mentioned earlier. However, the problem can be solved in pseudo-polynomial time using dynamic programming, as explained in [5].

3 Demonstration

3.1 Experimental setup

Within the TAC/SCM environment [2], agents act as Personal Computer (PC) manufacturers, willing to make profit in the simulated market, by competing with others on customer and supplier contracts, by means of auctioning. In this context, a proper set of performance attributes were selected by a domain expert and classified as Simple and Complex metrics in a TAC/SCM-specific Fuzzy-MRT. Figure 3.1 depicts the resulting MRT. Note that the actual weight values are calculated by applying a genetic algorithm on training data, as described in Section 3.2. The weight notation F_{i_j} in Figure 4 corresponds to the contribution of child metric i to its parent j, according to the metric enumeration provided in Table 1. Figures **??** and **??** depict the selected membership functions for the *FU* and *CompInv* Metrics, respectively.

Agents run their own factory unit, which has limited production capacity. Each day the game server sends requests for quotes (RFQs) to all agents on behalf of customers. Agents may make their offers to the customers based on their ability to satisfy delivery dates and reserve prices by sending a quote before the end of the day. The next day, if an agent's quote is a winning bid, the customer orders from that agent and, to get paid, the manufacturing agent must deliver the ordered PCs on-time. The manufacturing

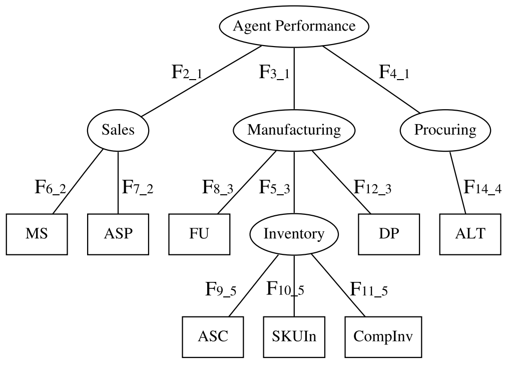

Figure 2. The Metrics Representation Tree for the TAC/SCM game

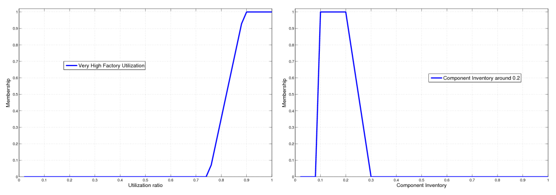

Figure 3. Sample Membership Functions for a) Factory Utilization, and b) Component Inventory

agent may either assemble the ordered PCs at that time or it can use PCs previously assembled and stocked in its inventory. Different types of PCs may be assembled, each requiring a different component compilation. Agents may procure components from eight different suppliers by sending RFQs and issuing orders to the suppliers. If an agent defaults in delivering customer orders, it is billed with a penalty that is determined by the customer in its initial RFQ. Figure 3 provides a schematic representation of the game. One may easily identify four game phases: (i) component procurement, (ii) inventory management, (iii) production and delivery scheduling, and (iv) computer sales.

A series of experiments was conducted in order to: a) calculate the optimal weights for the branches of the MRT, b) prove the validity of the MRT for evaluating the performance of actual TAC agents, and c) apply complexity control for partial evaluation. The experiments were performed on a collection of log files, produced during the execution of 530 games that involved the participation of a total of 2400 agent instances. For each agent, the dataset contains the crisp values of all Simple Metrics.

Table 1. Metrics for the TAC/SCM game

#	Name	Abbreviation	Type	Complexity	DetDef	Weight
1.	Agent Performance	-	Composite	-	-	-
2.	Sales	-	Composite	-	0.5512	-
3.	Manufacturing	-	Composite	-	0.9055	-
4.	Procuring	-	Composite	-	0.0819	-
5.	Inventory	-	Composite	-	0.7716	-
6.	Market Share	MS	Simple	4000	0.4973	0.4253
7.	Avg. Selling Price	ASP	Simple	4000	0.0769	0.2741
8.	Factory Units	FU	Simple	220	0.9990	0.0696
9.	Avg. Storage Cost	ASC	Simple	660	0.1451	0.7808
10.	Stock Keeping Unit Inventory	SKUInv	Simple	440	0.8631	0.8124
11.	Component Inventory	CompInv	Simple	220	0.8981	0.7524
12.	Delivery Performance	DP	Simple	220	0.8318	0.1314
13.	Avg. Lead Time	ALT	Simple	300	0.0866	0.0071

3.2 Training

In order to model the performance of an agent i on a game we use the metric $\mu(i)$, as defined by a domain expert:

$$\mu(i) = \begin{cases} w(BA(i)) \times BA(i) & \textit{if BA(i)} \geq 0 \\ (1 - w(BA(i)) \times BA(i) & \textit{if BA(i)} < 0 \end{cases}$$

where $BA(i)$ is the *Bank Account* of agent i after each game.

Since the absolute value of BA is always related to the correlated to the state of the simulated market in each game, the weighted term $w(BA(i))$ is employed. $w(BA(i))$ is defined as a function of attributes that remain constant throughout each game, namely the *Number of Agents (NoA)* participating in each game, the *Avg Purchase Price (APP)* and the *Customer Demand (CD)*:

$$w(BA(i)) = 0.3 \times f + 0.45 \times g + 0.25 \times h$$

where $f = 0.03 * NoA^2$, $g = 0.125 * (0.5 - ln(CD))$ and $h = e^{APP}/3$.

Note that $\mu(i)$ is a posterior metric, meaning that it uses information (Bank Account) not modelled by the MRT. To calculate the optimal fuzzy weights we use the posterior values $\mu(i)$ as ground-truth and minimize the error function:

$$err = \sum_{i=1}^{N} |\hat{\mu}(i) - \mu(i)|$$

where N is the total number of agents in the training set and $\hat{\mu}(i)$ is the inferred value of the performance of agent i, based on the currently calculated fuzzy weights.

To do so, we have employed a genetic algorithm, with its chromosome comprising the 12 corresponding MRT weights. The training process was applied to a total of 100 of the 2400 agents of the initial dataset and took less than a minute on a typical Core 2 machine. The produced fuzzy weights are presented in Column 7 (Weight) of Table 1.

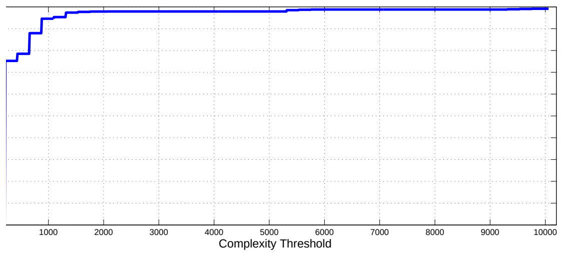

Figure 4. Partial Validity for increasing computational complexity

3.3 Testing

Using the weights computed in the previous section, we tested the produced Fuzzy MRT on the remaining 2300 agents of the dataset. The average error:

$$\frac{1}{M} \sum_{i=1}^{N} |\hat{\mu}(i) - \mu(i)|$$

where M the total number of testing agents, was 0.0742.

3.4 Complexity Control

Finally, using the process of Section 2.3, we produced the detailed definition of each Simple Metric. Using the corresponding computation complexities (as depicted in Column 5 of Table 1), we designed the evaluation procedure for various setups. Figure 3.4 shows the attained partial validity using to increasing complexity thresholds.

It is evident from this graph that, after a certain point, increasing complexity threshold does not have significant impact on the validity of the evaluation. We can, therefore achieve satisfactory levels of validity for low computational costs by properly designing the MRT.

4 Conclusions

In this work, we have presented an efficient scheme for constructing and employing optimal fuzzy trees for evaluating agent performance. A formal definition of the fuzzy trees is provided, as well as three meta-metrics for the quantitative assessment of the soundness of the selected metrics. The presented scheme was applied to the TAC/SCM domain. The produced MRT, containing optimal weights calculated by a genetic algorithm, was applied to performance data from previously conducted TAC/SCM games. The results showed that the proposed approach can be employed by agents in real-time. Moreover, under limited resources, the MRT can be significantly simplified, yet yielding satisfactory levels of evaluation accuracy.

5 Acknowledgements

This paper is part of the 03ED735 research project, implemented within the framework of the Reinforcement Programme of Human Research Manpower (PENED) and cofinanced by National and Community Funds (25% from the Greek Ministry of Development-General Secretariat of Research and Technology and 75% from E.U.-European Social Funding).

References

[1] J. Albus, E. R. Messina, and J. M. Evans. Performance metrics for intelliget systems - white paper. In *Proc. PERMIS00*, 2000.

[2] R. Arunachalam and N. M. Sadeh. The supply chain trading agent competition. *Electronic Commerce Research and Applications*, 4:63–81, 2005.

[3] C. Dimou, A. L. Symeonidis, and P. A. Mitkas. *Soft Computing for Knowledge Discovery and Data Mining*, chapter Data Mining and Agent Technology: a fruitful symbiosis. Springer, 2007.

[4] M. Falelakis, C. Diou, A. Valsamidis, and A. Delopoulos. Complexity control in semantic identification. In *Proc. FUZZ-IEEE05*, pages 102–107, 2005.

[5] M. Falelakis, C. Diou, A. Valsamidis, and A. Delopoulos. Dynamic semantic identification with complexity constraints as a knapsack problem. In *Proc. FUZZ-IEEE05*, pages 567 – 572, 2005.

[6] B. A. Kitchenham. Evaluating software engineering methods and tools part 12: evaluating desmet. *SIGSOFT Softw. Eng. Notes*, 23(5):21–24, 1998.