Evaluating Dynamic Protocols for Open Agent Systems

Michael Apostolou[1] and Alexander Artikis[2 3]
[1]Department of Informatics, University of Piraeus, Athens, 18534, Greece
[2]Institute of Informatics & Telecommunications, NCSR "Demokritos", Athens, 15310, Greece
[3]Electrical & Electronic Engineering Department, Imperial College London, SW7 2BT, UK
p04008@students.cs.unipi.gr, a.artikis@acm.org

ABSTRACT

We present a software system for evaluating 'dynamic' protocol specifications for open multi-agent systems, that is, specifications that are developed at design-time but may be modified at run-time by the protocol participants.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence

General Terms

Design, Experimentation

Keywords

organised adaptation, dynamic specification

1. INTRODUCTION

A particular kind of Multi-Agent System (MAS) is one where the member agents are developed by different parties, and where there is no access to an agent's internal state. MAS of this type are often classified as 'open'. In some open MAS, environmental, social or other conditions may favour, or even require, specifications modifiable during the system execution. Consider, for instance, the case of a malfunction of a large number of sensors in a sensor network, or the case of manipulation of a voting procedure due to strategic voting, or when an organisation conducts its business in an inefficient manner. We have been developing an infrastructure for 'dynamic' protocol specifications for open MAS, that is, specifications that are developed at design-time but may be modified at run-time by the protocol participants [1]. The infrastructure consists of well-defined procedures for proposing a modification of the 'rules of the game' of a protocol as well as decision-making over and enactment of proposed modifications.

The protocol rules that may be modified at run-time are called *Degrees of Freedom* (*DoF*). Consider, for example, the rules that express the conditions in which an agent accepts a proposition as a premise, as a DoF of an argumentation protocol. Agents may decide, at run-time, to change the value of this DoF: it may be decided that silence implies consent, or that illegally obtained evidence is/is not accepted, etc.

Cite as: Evaluating Dynamic Protocols for Open Agent Systems, Michael Apostolou, Alexander Artikis, *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009),* Decker, Sichman, Sierra and Castelfranchi (eds.), May, 10–15, 2009, Budapest, Hungary, pp. 1419 – 1420

We present a software system for evaluating dynamic protocol specifications for open MAS. A protocol specification with l DoF creates an l-dimensional specification space where each dimension corresponds to a DoF. A point in the l-dimensional specification space, or *specification point*, represents a complete protocol specification, and is denoted by an l-tuple where each element of the tuple expresses a value of a DoF. Our software system allows for the evaluation of a dynamic protocol specification by modelling the corresponding specification space as a *metric space* [2]. More precisely, given the set of specification points of a protocol, the protocol evaluator (hereafter 'user') denotes the 'desired' specification point, and the system computes, among other things, the 'distance' between the desired point and the specification point current at each time. In the following sections we present in detail the functionality and operation of our system, including the way we compute the distance between two specification points.

2. SYSTEM FUNCTIONALITY

In order to evaluate a dynamic protocol specification the user has to follow two steps. First, she has to denote the desired value of each DoF — the desired DoF values constitute the desired specification point. Second, she has to rank all DoF values, that is, associate each DoF value with a non-negative integer. Ranking the DoF values allows for the use of a standard *metric* (or *distance function*) to calculate the distance between two (ranked) specification points.

After the completion of the above steps, the system allows the user to view the following types of information, for any time-point of the protocol execution under consideration:

- The current specification point, that is, the current value of each DoF.
- The distance between the desired specification point and the current specification point. The distance is computed with the use of several metrics, such the (weighted) Manhattan metric.

The user may specify the maximum (threshold) distance that a specification point may have from the desired one. The threshold is metric-specific. The system will notify the user when the distance between the desired point and the current one exceeds the specified threshold.

Apart from information relating to specific time-points, the system generates a report concerning the complete protocol execution, recording the intervals in which:

- The value of a DoF is the desired one. (The desired specification point is reached when the value of *every* DoF is the desired one.)
- The distance between the desired specification point

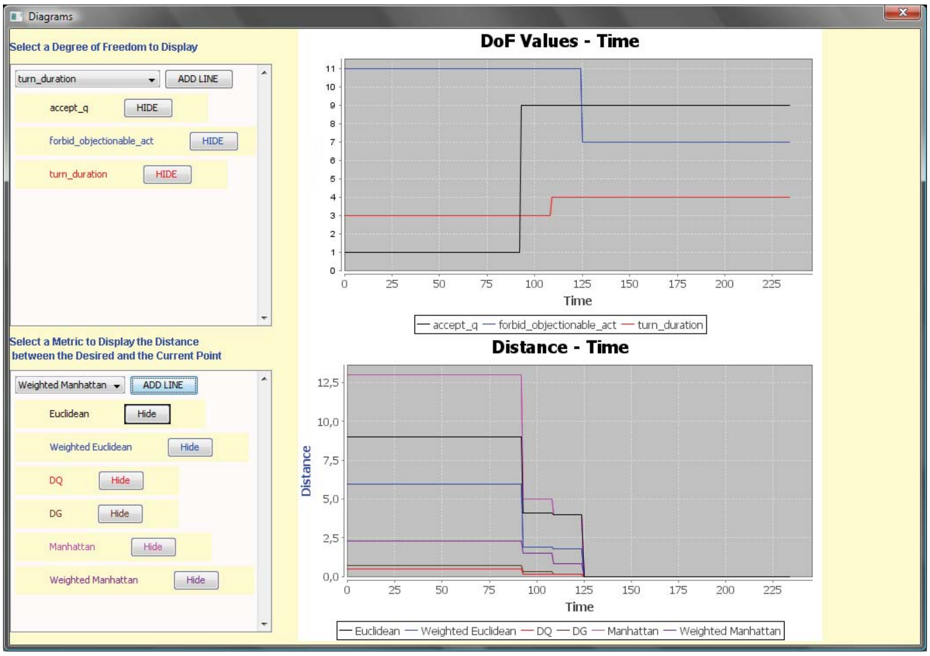

Figure 1: Evaluating a Dynamic Specification of an Argumentation Protocol.

and the point current at each time is the longest/ shortest.

- The distance between the desired specification point and the point current at each time is greater/less than the user-specified threshold distance.

Additionally, the average distance between the desired specification point and the point current at each time is recorded.

To aid the evaluation of a protocol specification, the system visualises a subset of the aforementioned types of information by means of two types of graph. Figure 1 shows a system screenshot displaying the two graph types; the screenshot concerns the evaluation of a dynamic argumentation protocol for open MAS. The first graph type, displayed on top of Figure 1, shows the values of a protocol's DoF over time (in this example there are three DoF). The user may choose to display in this graph the desired values of each DoF (in addition to the actual DoF values). To avoid clutter the user may select to view a subset of the actual/desired DoF values. The second type of graph, displayed on the bottom of Figure 1, shows the distance between the desired specification point and the point current at each time. Six lines are shown in the bottom graph indicating that the user chose to compute the distance with the use of six different metrics — the names of the chosen metrics are shown on the bottom-left panel. The user may choose to additionally display in this graph each metric-specific threshold distance — in this way the graph would explicitly show the intervals in which the distance between the desired specification point and the point current at each time is greater/less than the threshold distance. The two types of graph are displayed on the same panel in order to help the user identify whether a change of a DoF value reduces or increases the distance between the actual specification point and the desired one (in the example shown in Figure 1 every DoF value modification reduces the distance between the actual specification point and the desired one — at time 125 the desired specification point is reached).

3. SYSTEM OPERATION

The system consists of two main components. The first is a Prolog implementation of an Event Calculus formalisation, called EClp, expressing a dynamic protocol specification. The Event Calculus [4] is a formalism for representing and reasoning about events and their effects. EClp expresses, among other things, the conditions in which an agent may successfully propose a modification of the protocol rules (DoF values), and the conditions in which a proposal is accepted and, consequently, the protocol rules (DoF values) are modified. (See [1] for details about formalising dynamic protocol specifications.) Given a narrative of events that have taken place during a protocol execution, EClp computes, among other things, the specification point (the value of every DoF) current at each time.

The second system component is a Java implementation offering the system front-end, enabling the interaction with the user. The Java implementation queries EClp about a protocol's DoF and their possible values, as well as the specification point current at each time. Given the information retrieved from EClp and that entered by the user (desired specification point, ranking of DoF values), the system offers the functionality presented in the previous section.

The system is an implementation of the ideas presented in [3].

4. REFERENCES

[1] A. Artikis. Dynamic protocols for open agent systems. In *Proceedings of AAMAS*, 2009.

[2] V. Bryant. *Metric Spaces*. Cambridge University Press, 1985.

[3] D. Kaponis and J. Pitt. Dynamic specifications in norm-governed open computational societies. In *Proceedings of ESAW Workshop*, LNCS 4457, pages 265–283. Springer, 2007.

[4] R. Kowalski and M. Sergot. A logic-based calculus of events. *New Generation Computing*, 4(1):67–96, 1986.