

Towards the Analysis & Prediction of Complex System Behaviour in SAPERE

Marco Alberti Andrea Omicini

DISI, ALMA MATER STUDIORUM – Università di Bologna, Italy

Email: marco.alberti3@studio.unibo.it, andrea.omicini@unibo.it

Abstract—Complex computational systems are often intrinsically unpredictable even for the engineers designing them: understanding their possible evolution over time is typically required, for instance in order to match some application requirements. In this paper, we first survey some of the most promising and effective techniques for the analysis and prediction of complex computational systems, then we focus on SAPERE-coordinated pervasive ecosystems. We identify the most essential elements of the SAPERE coordination model, select two techniques – model checking and data mining –, and derive two possible approaches to the analysis and prediction of the behaviour of SAPERE systems.

Keywords—Pervasive ecosystems, SAPERE coordination model, model checking, data mining, behaviour prediction

I. INTRODUCTION

The behaviour of complex systems like multiagent, intelligent, pervasive, and self-* organising systems is often unpredictable—either for practical or theoretical reasons, such as the desired occurrence of emergent behaviours. This is particularly problematic in the engineering of complex computational systems, where at least *some* degree of predictability is often mandatory to ensure the satisfaction of some application requirements.

This is the main reason why a number of different approaches in the last years have tried to find out some systematic way to address the issue—however, with limited or just partial success. In order to understand how to possibly improve over the current state-of-the-art, we focus on a specific class of complex computational systems, and try to conceptually apply the fittest approaches to such a class. In particular, in this paper we focus on the SAPERE coordination model [1], and on the SAPERE-coordinated pervasive ecosystems [2] as meaningful cases of pervasive, self-organising systems of services.

Along this line, in this position paper we first survey some of the most promising and effective approaches to the analysis and prediction of complex computational systems, then we focus on SAPERE systems—that is, pervasive service ecosystems built around the SAPERE coordination model. We identify the most essential elements of the SAPERE coordination model, then we identify the two most promising techniques – model checking and data mining –, and derive two potentially-effective approaches to the analysis and prediction of the behaviour of SAPERE pervasive ecosystems, foreseeing their application to the case of a SAPERE-based crowd-steering scenario.

II. BACKGROUND

While facing the problem of analysing the behaviour of complex computational system such as multiagent (MAS) and pervasive systems, we come across many approaches. However, even the most effective among them deal with very narrow portions of the system behaviour, perform quite limited sorts of analysis, and are typically applicable to quite specific domains. By looking at the existing literature, we can classify the approaches to the analysis of complex MAS behaviour according to two main classes: those addressing the issue of the compliance to the rules steering system evolution, and those aimed at predicting system behaviour focusing on specific system features.

In the remainder of this section we survey the most significant examples of the two classes.

A. Refining system evolution

Model-checking-based approaches essentially enumerate all admissible states of a system, and verify that some specific properties – often expressed in LTL (*Linear Temporal Logic*), less often in CTL (*Computational Tree Logic*) – hold in given states a system goes across. On the one hand, this technique is highly effective when used to verify *safety* or *liveness* properties in specific domains. At the same time, it turns to be impractical when dealing with systems featuring huge state spaces, and also when adopted in order to verify generic system properties. Besides, in a MAS context, verifying single entity properties is often not useful—in particular when dealing with essential properties of complex MAS like emergent behaviours.

In order to address the above issue, many extensions to the aforementioned logics were defined. For instance, [3] extends LTL by defining DLTL (*Dynamic Linear Temporal Logic*), which enforce the *until* operator to address global pre- and post-conditions. In [4] LTL is also extended towards GLTL (*Generalized Linear Temporal Logic*) in order to support a more global property-verification system.

Some specific activity was devoted to model-check BDI agents built after the AgentSpeak model. In [5], [6], [7], a restriction of AgentSpeak(L) addressing finite state systems only (called AgentSpeak(F)) is presented. Using this language, properties expressed in BDI logic can be easily converted to LTL specifications and verified with a compatible model-checking tool.

All model-checking approaches are coupled to a representation of the system in term of a LTS (Labelled Transition System). The most common LTS are of course Petri nets. Modelling a multiagent system using Petri nets is not an easy task: in order to help the translation, extension to Petri nets – such as SLPN (*Simple Logic Petri Nets*) – were defined. These sorts of Petri nets allow the specification of pre- and post-conditions enriching the transitions of a normal Petri net, thus making it easier the mapping of MAS—in particular in the case of MAS built using *AgentSpeak*, as discussed in [8].

Another example involving the use of Petri nets is the model-checking approach to *swarm systems*—as in [9]. The main peculiarity of the systems of such a sort is the huge number of entities involved, which makes the use of a standard model-checking approach quite problematic indeed. However, as suggested by [9], [10], the problem can be avoided by exploiting the fact that components of a swarm system are typically uniform, and not individually complex: so, a finite state space for every and all individuals can be defined. Along this line, a single LTS can be used to represent all the entities, by assigning a number to each admissible state for the individual entities, which represents at any instant of time the number of the system entities currently in that particular state. Such a number is computed by using probabilistic analysis, so it represents an estimation of the real number of entities that will be in the state at runtime: however, noise tolerance of swarm systems ensures the reliability of the analysis.

B. Predicting behaviour

Behaviour prediction is a relatively-new technique, compared to model-checking. For the past years, the two terms were often used as synonymous: nowadays, with the relentless rise of MAS as complex computational systems, the two techniques are becoming more and more distinct. There are systems, in fact, which cannot be analysed in a satisfactory way by using model-checking only.

True behaviour prediction is not so frequent in the literature. Most of the work on this particular analysis was performed in the context of RoboCup systems. There, the most satisfactory technique involves the use of data mining algorithms to analyse past behaviours of the system, in order to possibly to understand the actual rules governing the system evolution. A technique aimed at reducing the complexity of the behaviour of a team of robots down to atomic actions is presented in [11]. Good results based on the same sort of approach are reported in [12], where the authors show how to predict the strategy that football teams of robots will put in place.

Moving away from both robotic and football contexts, another interesting example is presented in [13], where a similar approach was used to identify the different phases of the behaviour of a *swarm system* modelling the interactions

between chickens inside their cages. The approach was meant to avoid cannibalistic attacks, identifying the best moment to intervene to prevent the gathering phase typically preceding aggressive actions.

III. THE SAPERE MODEL: AN OVERVIEW

In the last years, the emergence of an ever-growing number of pervasive computing technologies, mostly caused by the easy accessibility of distributed infrastructures, has led to the definition of the notion of *pervasive ecosystem* [1]. A pervasive ecosystem is a distributed computational net structure intertwined with an everyday environment, and modelled after a real-life natural ecosystem. In the same way as natural ecosystems, computational ecosystems have components and actors which abide the laws of the system and act to achieve some kind of goal.

The SAPERE coordination model [2], [1] is defined based on a notion of pervasive ecosystems ruled by *eco-laws* resembling chemical reaction laws. Instead of chemical components, eco-laws manipulate LSA (*Live Semantic Annotations*) which are stored locally in every node of the infrastructure representing a virtual counterpart of the ecosystem. Like chemical reactions, eco-laws act accordingly to specific stochastic rates. Using such a chemical approach, eco-laws can put into effect such processes as LSA diffusion, interaction, composition and disposal.

In short, the SAPERE coordination model deals with three major features: *situatedness*, *adaptivity* and *diversity tolerance*. Pervasive ecosystems deal with both spatially and socially situated activities of users, hence the need of being able to interact with the environment and adapt to it in place of contingency situations even without human intervention or global supervision. Furthermore, the structure of the system promotes openness, meaning that services supported by the system should not be limited *a priori*, and the addition of new ones should be promoted when possible to improve and integrate the existing system functionalities.

SAPERE is built on top of a set of core components capable to handle the above-mentioned features:

- because of the heterogeneous nature of the entities in the system, a uniform representation is needed; LSA tackle specifically this problem, being able to represent an up-to-date state of the entities, related to the specific context in which the information is produced and used
- the infrastructure where LSA are reified is called *LSA-space*: that is, a distributed net structure, where the context in which entities live and act is defined by the LSA in the current node
- in order to allow an entity to act rightfully, a mechanism to define what an entity can perceives and change is needed; this is achieved by *LSA-bonding*, which allows an LSA to link to other LSA in the same space, and SAPERE entities to inspect – via bonds – the state of their peers and act accordingly

- system rules – the eco-laws – are instead in charge of managing the global behaviour of the whole system, by manipulating – deleting, updating, moving, bonding – LSA in the system, in the way of chemical reactions

In [1] a more detailed description of the SAPERE model is provided.

The reference example adopted here as a conceptual reference for SAPERE is a museum-located system, equipped with public displays interacting with the personal handheld devices of the visitors [14]. Devices deposit LSA inside the LSA-space, and the eco-laws steer visitors using a computational field, – in this particular example, a gradient – guiding them through the rooms to their destination, or to possible exhibitions of interest, understood according to the information on their personal preferences in their LSAs. Another feature is the ability of the system to perceive the number of visitors in the areas of the museum in terms of a *crowd factor* to better steer movements and avoid crowd gathering in high-traffic areas.

IV. MODEL-CHECKING: A POSSIBLE APPROACH

Taking a bird-eye view over model-checking approaches, the pattern arising is apparently the need to verify global properties of the system, aside from the ones of the single entities. This is quite understandable wherever – holistically – “the whole is more than the sum of its parts”—as in the case of emergent behaviour in complex computational systems, for instance. Even knowing a way to map individual entity properties onto global system properties related to emergent behaviours – which would be anyway a problem, giving the typical non-compositionality of emergent properties –, it would be far easier to have a way to specify and verify global properties straightforwardly. This especially holds when dealing with self-organising systems—which is basically the reason why so much interest has been caught by the MABLE tool [15]. For this and other peculiar features, MABLE is seemingly quite a promising approach for modelling a SAPERE-coordinated system.

In short, MABLE is a Promela-compatible model-checking tool with a C-like language structure, enriched with BDI-friendly constructs and actions specifically aimed at making agent representation easy. MABLE allows the definition of *agents* as the standard building blocks of complex intelligent systems built as MAS. It supports a mechanism of *claims*, which – unlike other model-checking tools that support only single entity property specifications – can represent individual entity as well as global system LTL properties, to be verified during execution.

In the context of a SAPERE application – where emergent behaviours are a direct consequence of the manipulation of the LSA by the eco-laws, and not of single features (property compliance) of the entities – MABLE *claims* can be used to verify that the emergent behaviour evolves as expected, defining global properties – such as, for instance, crowd

levels or energy consumption – to be maintained below a specific threshold. A MABLE model can also be coupled to *semantic specifications* for specific communications [16]. A semantic specification allows the definition of pre- and post-conditions for a given communication, quite obviously meaning that for that the communication to happen, the pre-conditions need to be met, otherwise the execution is halted; when the communication has occurred, the post-conditions are put into effect. The mechanism of semantic specifications is pretty similar – at the due level of abstraction – to the mechanism of eco-laws described in [1], where the applicability of an eco-law with a specific rate depends on a certain combination of LSA reagents in the context (pre-conditions), and LSA products are created in the same context or in its neighbourhood (post-conditions).

Another feature of MABLE is the capability of an agent to make assumptions on the internal state of other agents—on their beliefs, for instance. Assumptions can also be expressed inside *claims*, which is particularly useful in MAS contexts, where entities are essentially blind to the internal state of others. By abstracting away from its intended meaning, this could be used to model the SAPERE mechanism of LSA-bonding SAPERE, which would change an external piece of information (LSA) into a internal property of the agent (belief). Along this line, a bond between LSA would simply be represented as a connection of the beliefs of the agents which would have otherwise produced the LSAs into question.

The application of the above approach to the crowd steering application example allows for two possible ways to model the system. The first would amount at considering every LSA as representative of some kind of entity, and modelling everything represented by a LSA as a MABLE agent. The second would require modeling only nodes and user handheld devices as agents, and other LSA as internal beliefs of the *node agent*. The latter approach is somewhat simpler, whereas the former one allows for a smaller interaction granularity—which would translate into more chances of property specification. Fields of the LSA would become beliefs of the corresponding agent, and bonds between LSA would become beliefs on the internal state of the coupled agents.

While in the structure of the former approach the creation of a new LSA corresponds to a spawn of a MABLE agent, in the latter the creation of new LSA corresponds to a communication act towards the node agent, followed by a change of its internal beliefs, where a new one is added. In both ways, the eco-laws would be modelled as MABLE communication acts, where the presence of specific LSA and/or rate of the eco-laws are the pre-conditions of the communication, and the LSA produced/consumed are the post-conditions of the communication. Thus, it would be possible to use *claims* to check whether the *crowd level* in the nodes, specified by an internal belief of the agent

corresponding to the node, evolves in a satisfactory way or not—for instance, if it ever goes beyond a specific threshold.

Finally, a SAPERE model could be represented in MABLE, then translated into a Promela specification¹, and ideally analysed using SPIN²—or, any other model checker supporting Promela specifications. Also, work is currently ongoing to create a translation mechanism into Java, with the aim to use JPF2³ as model-checking tool.

V. DATA MINING: A POSSIBLE APPROACH

After discussing a possible approach to the modelling and verification of a SAPERE-coordinated model to verify structural properties, in this section we sketch another possible approach. Putting together the techniques presented in [11] with the results described in [12], one could easily see that data mining analysis led to very good results, not only in predicting a specific behaviour, but in producing a probabilistic model of all expectable behaviours. Instead, in [13] the analysis was used successfully to identify various phases of the same behaviour.

Based on this, one could reasonably suppose that a similar approach, when correctly applied to a SAPERE-coordinated system, could be able to solve the problem (still being debated) of how the parameters of a simulation could be tuned up, and how to set the relation between the values of these parameters and the corresponding emergent behaviour. Till now, the only way to tune the system parameters is to observe of a set of simulations, and to deduce from there the way in which different values would affect the emergent behaviour. A first attempt to solve this problem was made by suggesting the possibility of providing the simulation with an optimisation mechanism through meta-heuristic algorithms – as in [17] –, but in this way it would be only possible to achieve a particularly-desirable behaviour, without identifying the real correlation between parameter values and the emergent behaviour itself. Besides, nothing ensures that the behaviour achieved would be the best possible result, being this kind of algorithms vulnerable to local maximum (minimum). The key-issue in data mining approaches are primarily the identification of the values that define the emergent behaviour, and the creation of a significant and adequately representative data set. Once the above issues are set, depending on the quality and effectiveness of the algorithm used, the possible analysis of a system – in particular, of a SAPERE-coordinated system – with data mining techniques should be meant to put in evidence:

- 1) the correlation between parameter values and emergent behaviour—or, in other words, the very goal of a data mining analysis, and the main problem that is being

addressed; data mining analysis is far more accurate than the current approaches

- 2) there could be counter-intuitive correlations between some parameters and some aspects of the emergent behaviour, correlations that would otherwise go unnoticed, and which data mining analysis should be able to find and properly highlight
- 3) once the correlation between the parameters and the behaviour is defined, it should be possible to identify the ideal assignment policy of the values, being capable, in this way, to either enforce desired behaviours, or discourage undesired ones
- 4) in addition to highlighting the correlation between micro- and macro-events of the simulation, such a sort of analysis would be able to identify correlations even between macro-events themselves, such as the identification of recurrent patterns of the emergent behaviour

Apart from this, data mining would be an off-line analysis, and, assuming the choice of an appropriate data set and identification of significant values, an exhaustive one, as opposed to a meta-heuristic optimisation.

As far as the crowd steering application is concerned, the first step would be to identify the parameters and values to be analysed—for instance, the crowd factor that penalises the calculation of distances, the rate of the eco-laws, and the effect of changing those values on the crowd levels in every node. In order to build a meaningful training set, a variety of simulation should be executed, varying both the values of the parameters and the topology of the museum, so to prevent pathological cases to affect the analysis. In this particular application, parameters are heavily intertwined to each other: however, it could be useful to analyse the effect of each parameter on the system behaviour by using two scatter plots besides a single three-dimensional one. It would be appropriate to build different groups of training sets subjecting each of these to analysis using different kind of algorithms. It is not easy to predict the kind of clustering that data sets would have, so it makes sense to use different kinds of algorithms, either partitional (*K-means*), if the number of clusters is known beforehand, hierarchical (*single linkage*), if it is not, bayesian, density based (*DBSCAN*), neural networks, or SVM (*Support Vector Machine*). As obvious, the more different algorithms are used, the better the analysis should result.

VI. CONCLUSIONS

In this position paper we face the issue of analysing and predicting the behaviour of complex pervasive, self-organising systems like SAPERE coordinated systems. We shortly survey the literature on the general subject along with the available tools, then we select two promising techniques and sketch their application to the analysis and prediction of the behaviour of SAPERE pervasive ecosystems.

¹Promela verification language: <http://www.dai-arc.polito.it/dai-arc/manual/tools/jcat/main/node168.html>

²The SPIN model checker: <http://spinroot.com/spin/whatispin.html>

³Java PathFinder 2: <http://babelfish.arc.nasa.gov/trac/jpf>

Our future activity will obviously include actually experimenting the two approaches here devised upon selected SAPERE applications – such as crowd steering and pervasive display applications – and also trying to integrate different techniques into a single articulated yet coherent approach capable of providing reliable prediction of the most meaningful and useful features of a SAPERE system.

ACKNOWLEDGMENTS

This work has been supported by the EU-FP7-FET Proactive project SAPERE Self-aware Pervasive Service Ecosystems, under contract no. 256873.

REFERENCES

- [1] M. Viroli, D. Pianini, S. Montagna, and G. Stevenson, “Pervasive ecosystems: a coordination model based on semantic chemistry,” in *27th Annual ACM Symposium on Applied Computing (SAC 2012)*, S. Ossowski, P. Lecca, C.-C. Hung, and J. Hong, Eds. Riva del Garda, TN, Italy: ACM, 26–30 Mar. 2012. [Online]. Available: <http://dl.acm.org/citation.cfm?doid=2245276.2245336>
- [2] F. Zambonelli, G. Castelli, L. Ferrari, M. Mamei, A. Rosi, G. Di Marzo, M. Risoldi, A.-E. Tchao, S. Dobson, G. Stevenson, Y. Ye, E. Nardini, A. Omicini, S. Montagna, M. Viroli, A. Ferscha, S. Maschek, and B. Wally, “Self-aware pervasive service ecosystems,” *Procedia Computer Science*, vol. 7, pp. 197–199, Dec. 2011, proceedings of the 2nd European Future Technologies Conference and Exhibition 2011 (FET 11). [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1877050911005667>
- [3] L. Giordano, A. Martelli, and C. Schwind, “Verifying communicating agents by model checking in a temporal action logic,” in *Logics in Artificial Intelligence*, ser. LNCS, J. J. Alferes and J. Leite, Eds., vol. 3229. Springer, 2004, pp. 57–69. [Online]. Available: http://rd.springer.com/chapter/10.1007/978-3-540-30227-8_8
- [4] L. R. Pokorny and C. R. Ramakrishnan, “Modeling and verification of distributed autonomous agents using logic programming,” in *Declarative Agent Languages and Technologies II*, ser. LNAI, J. A. Leite, A. Omicini, P. Torroni, and P. Yolum, Eds. Springer, Jun. 2006, vol. 3476, pp. 313–319, 2nd International Workshop (DALI 2004), New York City, NY, USA, 19 Jul. 2004. Revised Selected Papers. [Online]. Available: <http://www.springerlink.com/content/h0lphv2fvm3bwrea/>
- [5] R. H. Bordini, M. Fisher, C. Pardavila, and M. J. Wooldridge, “Model checking AgentSpeak,” in *2nd International Joint Conference on Autonomous Agents and Multi-Agents Systems (AAMAS 2003)*, J. S. Rosenschein, M. J. Wooldridge, T. Sandholm, and M. Yokoo, Eds. New York, NY, USA: ACM Press, 11–14 Jul. 2003, pp. 409–416. [Online]. Available: <http://dl.acm.org/citation.cfm?doid=860575.860641>
- [6] R. H. Bordini, M. Fisher, W. Visser, and M. J. Wooldridge, “Verifiable multi-agent programs,” in *Programming Multi-agent Systems*, ser. LNCS, M. M. Dastani, J. Dix, and A. El Fallah-Seghrouchni, Eds. Springer, 2004, vol. 3067, pp. 72–89, 1st International Workshop (PROMAS 2003), Melbourne, Australia, 15 Jul. 2003. Selected Revised and Invited Papers. [Online]. Available: <http://www.springerlink.com/content/f4hx00p4lxbxby2c3/>
- [7] —, “Verifying multi-agent programs by model checking,” *Autonomous Agents and Multi-Agent Systems*, vol. 12, no. 2, pp. 239–256, 2006, EUMAS’04 Special Issue. [Online]. Available: <http://www.springerlink.com/content/9130gm1003u756t3/>
- [8] T. M. Behrens and J. Dix, “Model checking multi-agent systems with logic based Petri nets,” *Annals of Mathematics and Artificial Intelligence*, vol. 51, no. 2-4, pp. 81–121, 2007, from the issue entitled “Special Issue in honor of Professor Jack Minker’s 80th birthday”. [Online]. Available: <http://www.springerlink.com/content/3q71261838q73u26/>
- [9] S. Konur, C. Dixon, and M. Fisher, “Formal verification of probabilistic swarm behaviours,” in *Swarm Intelligence*, ser. LNCS, M. Dorigo, M. Birattari, G. A. Di Caro, R. Doursat, A. P. Engelbrecht, D. Floreano, L. M. Gambardella, R. Groß, E. Sahin, H. Sayama, and T. Stützle, Eds. Springer, 2010, vol. 6234, pp. 440–447, 7th International Conference (ANTS 2010), Brussels, Belgium, 8–10 Sep. 2010. Proceedings. [Online]. Available: <http://www.springerlink.com/content/t8415hx2001g2041/>
- [10] C. Dixon, A. Winfield, and M. Fisher, “Towards temporal verification of emergent behaviours in swarm robotic systems,” in *Towards Autonomous Robotic Systems*, ser. LNCS, R. Groß, L. Alboul, C. Melhuish, M. Witkowski, T. J. Prescott, and J. Penders, Eds. Springer, 2011, vol. 6856, pp. 336–347, 12th Annual Conference (TAROS 2011), Sheffield, UK, 31 Aug. – 2 Sep. 2011. Proceedings. [Online]. Available: <http://www.springerlink.com/content/m2826416165wnn84/>
- [11] G. A. Kaminka, M. Fidanboylyu, A. Chang, and M. M. Veloso, “Learning the sequential coordinated behavior of teams from observations,” in *RoboCup 2002: Robot Soccer World Cup VI*, ser. LNCS, G. A. Kaminka, P. U. Lima, and R. Rojas, Eds., vol. 2752. Springer, 2003, pp. 111–125. [Online]. Available: <http://www.springerlink.com/content/hy66lxap25mr9lmc/>
- [12] A. D. Lattner, A. Miene, U. Visser, and O. Herzog, “Sequential pattern mining for situation and behaviour prediction in simulated robotic soccer,” in *RoboCup 2005: Robot Soccer World Cup IX*, ser. LNCS, A. Bredenfeld, A. Jacoff, I. Noda, and Y. Takahashi, Eds. Springer, 2006, vol. 4020, pp. 118–129. [Online]. Available: <http://www.springerlink.com/content/582485v139421n22/>
- [13] M. Mnif, U. Richter, J. Branke, H. Schmeck, and C. Müller-Schloer, “Measurement and control of self-organised behaviour in robot swarms,” in *Architecture of Computing Systems — ARCS 2007*, ser. LNCS, P. Lukowicz, L. Thiele, and G. Tröster, Eds. Springer, 2007, vol. 4415, pp. 209–223, 20th International Conference, Zurich, Switzerland, 12–15 Mar. 2007. Proceedings. [Online]. Available: <http://www.springerlink.com/content/n4035134561jk037/>

- [14] S. Montagna, M. Viroli, M. Risoldi, D. Pianini, and G. Di Marzo Serugendo, "Self-organising pervasive ecosystems: A crowd evacuation example," in *3rd International Workshop on Software Engineering for Resilient Systems*, ser. Lecture Notes in Computer Science. Geneva, Switzerland: Springer, 29–30 Sep. 2011, vol. 6968, pp. 115–129. [Online]. Available: <http://www.springerlink.com/content/096hh324410xj820>
- [15] M. J. Wooldridge, M. Fisher, M.-P. Huget, and S. Parsons, "Model checking multi-agent systems with MABLE," in *1st International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2002)*, C. Castelfranchi and W. L. Johnson, Eds., vol. 2. New York, NY, USA: ACM, 15–19 Jul. 2002, pp. 952–959. [Online]. Available: <http://dl.acm.org/citation.cfm?doid=544862.544965>
- [16] M.-P. Huget and M. J. Wooldridge, "Model checking for ACL compliance verification," in *Advances in Agent Communication*, F. Dignum, Ed. Springer, 2004, vol. 2922, pp. 75–90, international Workshop on Agent Communication Languages (ACL 2003), Melbourne, Australia, 14 Jul. 2003. Revised and Invited Papers. [Online]. Available: <http://www.springerlink.com/content/8d2rqhdyfjj4klnd/>
- [17] G. J. Mejtsky, "A metaheuristic algorithm for simultaneous simulation optimization and applications to traveling salesman and job shop scheduling with due dates," in *Winter Simulation Conference*, 9–12 Dec. 2007, pp. 1835–1843. [Online]. Available: <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4419810>