The SOCS Computational Logic Approach to the Specification and Verification of Agent Societies

Marco Alberti[1], Federico Chesani[2], Marco Gavanelli[1], Evelina Lamma[1], Paola Mello[2], and Paolo Torroni[2]

[1] Dip. di Ingegneria - Università di Ferrara - Via Saragat, 1 - 44100 Ferrara, Italy
`{malberti, mgavanelli, elamma}@ing.unife.it`
[2] DEIS - Università di Bologna - Viale Risorgimento, 2 - 40136 Bologna, Italy
`{fchesani, pmello, ptorroni}@deis.unibo.it`

Abstract. This article summarises part of the work done during the first two years of the SOCS project, with respect to the task of modelling interaction amongst CL-based agents. It describes the SOCS social model: an agent interaction specification and verification framework equipped with a declarative and operational semantics, expressed in terms of abduction. The operational counterpart of the proposed framework has been implemented and integrated in SOCS-SI, a tool that can be used for on-the-fly verification of agent compliance with respect to specified protocols.

1 Introduction

Computees are Computational Logic-based entities interacting in the context of global and open computing systems [1]. They are abstractions of the entities that populate Global Computing (GC) environments [2]. These entities can form complex organizations, that we call *Societies Of ComputeeS* (SOCS, for short) [3]. The main objective of Global Computing, rephrased in terms of SOCS, is to provide a solid scientific foundation for the design of societies of computees, and to lay the groundwork for achieving effective principles for building and analyzing such systems. Between January 2002 and March 2004, the project developed a society formal model to satisfy the high-level objectives derived directly from the GC vision of an *open* and *changing* environment.

In this context, by "open" environment we mean, following Hewitt's work [4] about information systems and then Artikis et al.'s [5] about computational societies, an environment or society where the following properties hold:

(*i*) the behavior of members and their interactions are unpredictable (i.e., the evolution of the society is non-deterministic);
(*ii*) the internal architecture of each member is neither publicly known nor observable (thus, members may have heterogeneous architectures);
(*iii*) members of the society do not necessarily share common goals, desires or intentions (i.e., each member may conflict with others when trying to reach its own purposes).

This definition of openness is based on externally observable features within the society. It caters for heterogeneous and possibly non-cooperative members. Therefore, our model of society will not constrain the ways computees join or leave a society, it will emphasize the presence of heterogeneous computees in the same society, and it will assume that the internal structure of computees is not guaranteed to be observable, or their social behaviour predictable.

The SOCS social model specifies a social knowledge which interprets and gives a social meaning to the members' social behavior. It supports the notion of *social goal*, allowing for both goal-directed and non-goal-directed societies.

In our approach, we believe that the knowledge and technologies acquired so far in the area of Computational Logic provide a solid ground to build upon. At the society level, the role of Computational Logic is to provide both a declarative and an operational semantics to interactions. The advantages of such an approach are to be found:

(*i*) in the design and specification of societies of computees, based on a formalism which is declarative and easily understandable by the user;
(*ii*) in the possibility to detect undesirable behavior, through *on the fly* control of the system based on the computees' observable behavior (communication exchanges) and dynamic conformance check of such behaviour with the constraints posed by the society. Interestingly, as we will see, this can be achieved by exploiting a suitable proof procedure which is the operational counterpart of the mentioned formalism;
(*iii*) in the possibility to (formally) prove properties of protocols and societies.

Therefore, in our approach, we define the (semantics of) protocols and communication languages as logic-based integrity constraints over social events (e.g., communicative acts), called Social Integrity Constraints (ic_S) [6].

The ideal "correct" behaviour of a society is modelled as expectations about events. ic_S define the expectations stemming from a certain history of events and possibly a set of goals. Expectations and ic_S are the formalism used to define the "social semantics" of agent communication languages and interaction protocol: a semantics which is verifiable without having any knowledge about the agents' internals.

The syntax of ic_S and of the society in general are those of a suitably extended logic program. In fact, we define the "social knowledge" by assimilating it to abductive logic programs [7], and we define a notion of *expected social events*, by expressing them as abducible predicates, while using ic_S to constrain the "socially admissible" communication patterns of computees (i.e., those who match the expectations).

The society infrastructure is devoted to checking the compliance of the society members' behaviour, with respect to its expectations.

The compliance check is based on a proof-procedure called $\mathcal{S}$CIFF. $\mathcal{S}$CIFF, standing for "IFF, augmented with Constraints, for handling agent $\mathcal{S}$ocieties", is an extension of the well known IFF abductive logic programming proof-procedure, defined by Fung and Kowalski [8]. The $\mathcal{S}$CIFF extends the IFF in a number of directions: it provides both a richer syntax of abductive theories