$(\mathcal{C}/\mathcal{C}+)^{++}$: An Action Language for Representing Norms and Institutions

Marek Sergot Rob Craven

Department of Computing
Imperial College London
London SW7 2BZ
{mjs,rac101}@doc.ic.ac.uk

The language $\mathcal{C}/\mathcal{C}+$ of Giunchiglia, Lifschitz, McCain, Turner, and colleagues [5, 4, 3] is a formalism for specifying and reasoning about the effects of actions and the persistence ('inertia') of facts over time. An 'action description' in $\mathcal{C}/\mathcal{C}+$ is a set of $\mathcal{C}/\mathcal{C}+$ rules which define a transition system of a certain kind. A separate language is used for making assertions about this transition system (what is true when) and querying it. Implementations supporting a wide range of querying and planning tasks are available, notably in the form of the 'Causal Calculator' CCALC developed at the University of Texas[1].

We have been developing an extended form of the language $\mathcal{C}/\mathcal{C}+$, labelled $(\mathcal{C}/\mathcal{C}+)^{++}$ for now, designed for representing norms of behaviour and institutional aspects of (human or computer) societies. There are two main extensions. The first is a means of expressing 'counts as' relations between actions, also referred to as 'conventional generation'. In certain circumstances, the execution/occurrence of an action α, because of the existence of a set of norms or conventions, counts as the execution/occurrence of an action β. An agent raises its hand, and in certain contexts this act, by convention, counts as an act of making a bid. According to the account given in [6], a treatment of 'counts as' relations provides also a representation of the concept of institutionalised *power*, by which is meant that characteristic feature of institutions whereby agents, often when acting in specified roles, are empowered by an institution to perform certain specified acts and thereby create or modify normative/institutional states of affairs and relations (such as when a priest performs a marriage, or a manager assigns an employee to a project, or an agent signs a contract, or a programme chairperson declares a submitted paper as rejected).

The second extension of $(\mathcal{C}/\mathcal{C}+)^{++}$ is a way of specifying the permitted (acceptable, legal) states of a transition system and its permitted (acceptable, legal) transitions and histories ('runs').

The intended applications of $(\mathcal{C}/\mathcal{C}+)^{++}$ are to the specification and representation of norms and institutions in human and artificial (computational) societies. In particular, we want to be able to represent and reason about procedures and protocols—what actions are possible, permitted, valid at each stage, what rights, powers, obligations are created and destroyed as the protocol/procedure progresses, what normative or institutional relations are created when the protocol/procedure reaches completion. We want to be able to determine from a

[1] `http://www.cs.utexas.edu/users/tag/cc`

recorded narrative of what actions have been performed what actions are now possible, permitted, valid, and what their effects will be; and we should also like to be able to prove general properties of these protocols and procedures (termination, 'fairness', that such and such a protocol has a particular property, that two protocols have identical effects, and so on). Typical examples include auction protocols, rules of procedure, argumentation/dialogue games, negotiation protocols, contract formation, business rules and procedures, procedural law. An example of an application (of $\mathcal{C}/\mathcal{C}+$) to the representation of the 'contract net' protocol in multi-agent systems is presented in [2]. An alternative version using the event calculus is described in [1].

The current implementation of $(\mathcal{C}/\mathcal{C}+)^{++}$ uses the 'Causal Calculator' CCalc. An action description in $(\mathcal{C}/\mathcal{C}+)^{++}$ can be translated into a set of formulas of (classical) propositional logic (its 'literal completion'), which can then be passed to a standard propositional satisfaction solver. This is the method used by CCalc. It requires only minor modifications to adapt it for use with $(\mathcal{C}/\mathcal{C}+)^{++}$.

Besides some further language extensions, which include explicit support for agents, for roles, and for multiple institutions, we have two main lines of development. First, for application to larger-scale problems, we are trying to develop an alternative implementation, optimised for narrative assimilation rather than 'postdiction' or planning, which produces a computation similar to that of the event calculus. We have a prototype implementation though it works only on a restricted subset of the language at present. Second, we are investigating the applicability of $(\mathcal{C}/\mathcal{C}+)^{++}$ (and $\mathcal{C}/\mathcal{C}+$) as an alternative input language for standard (temporal) model checkers.

References

[1] A. Artikis, J. Pitt, and M. Sergot. Animated specifications of computational societies. In *Proceedings of AAMAS*. ACM, 2002.

[2] A. Artikis, M. Sergot, and J. Pitt. Specifying electronic societies with the Causal Calculator. In *Proceedings of Workshop on Agent-Oriented Software Engineering III (AOSE)*, LNCS 2585. Springer, 2003.

[3] Enrico Giunchiglia, Joohyung Lee, Vladimir Lifschitz, Norman McCain, and Hudson Turner. Nonmonotonic causal theories. *Artificial Intelligence*, To appear in 2003. `http://www.cs.utexas.edu/users/vl/papers/nmct.ps`.

[4] Enrico Giunchiglia, Joohyung Lee, Vladimir Lifschitz, and Hudson Turner. Causal laws and multi-valued fluents. In *Proc. of the Fourth Workshop on Nonmonotonic Reasoning, Action, and Change, Seattle*, August 2001.

[5] Enrico Giunchiglia and Vladimir Lifschitz. An action language based on causal explanation: Preliminary report. In *Proc. AAAI-98*, pages 623–630. AAAI Press, 1998.

[6] A. J. I. Jones and M. J. Sergot. A formal characterisation of institutionalised power. *Journal of the IGPL*, 4(3):429–445, 1996.