Labeled Event Structure Semantics of Linear Logic Planning

Ozan Kahramanoğulları*

Concurrency and planning are two fields of computer science that evolved independently, aiming at solving tasks that are similar in nature, but different in perspective: while planning formalisms focus on finding a plan (process), if there *exists* such a plan, that solves a given planning problem; the focus in concurrency theory is on universally quantified queries for proving properties of a concurrent, interacting system. Such a task requires an arsenal of mathematical methods, e.g., bisimulation, which respects the parallel behavior of actions and the non-determinism inherent in the system, and this way allows for an analysis of equivalence of processes.

We further elaborate on the linear logic approach to planning [7], aiming at providing a common language for planning and concurrency. Such a language will make it possible to import methods of concurrency to planning, which will then lead to a structural analysis of plans. Linear logic approach to planning, based on multiset rewriting, is a natural candidate for this task: linear logic allows explicit handling of resources due to controlled weakening and contraction, i.e., the multiplicative conjunction $\otimes$ is not idempotent ("$a \otimes a \vdash a$" is not provable), whereas conjunction $\wedge$ in classical logic is idempotent. The explicit treatment of resources does not only provide a natural solution to the famous frame problem, but is also crucial to express the dependency between actions that compete for resources. Furthermore, linear logic is widely recognized as a logic of concurrency (see, e.g., [8, 2]).

We establish an explicit correspondence between partial order plans and proofs of the linear logic planning problems which leads to a labeled event structure (LES) semantics of plans. Labeled Event Structures [9] is a behavioral model of concurrency which captures the causality between actions in terms of their dependencies in a partial order. Apart from the causality which is expressed in a partial order, in a LES, the non-determinism in the computation is captured by a conflict relation, which is a symmetric irreflexive relation of events. In a planning perspective, this corresponds to actions that are applicable at a state which conflict with each other. This conflict is in the sense that deciding for one over the other determines a different state space ahead. This is a concurrent model of the possible computations.

We associate to every planning problem a LES which represents the independence and causality of all actions performable in all different derivations produced by the search for a proof of the planning problem. By capturing the operational semantics of planning problems in terms of inference rules within the underlying logical framework, we associate a transition system to each proof of a planning problem. For this purpose, we adapt some ideas from [3] where LES semantics for a class of linear logic proofs has been studied. Relying on the notion of independence

*Computer Science Institute, University of Leipzig & International Center for Computational Logic, TU Dresden, ozan@informatik.uni-leipzig.de

among actions provided by the explicit handling of the resources, we then adapt the standard techniques in the literature to obtain LES's from transition systems. The LES provided leads to an abstraction from a transition system which allows to analyze these systems, observe recurring events and use standard methods of concurrency for a notion of plan equivalence.

An other contribution of this work is the establishment of the explicit correspondence between partial order plans and a certain class of linear logic proofs, which addresses the question of identity of proofs since a unique partial order corresponds to a class of proofs.

As the underlying formalism we employ the calculus of structures [4] presentation of linear logic [10]: The calculus of structures is a proof theoretical formalism, which is a generalization of the one-sided sequent calculus with the gain of interesting proof theoretical properties. In the calculus of structures, the notion of main connective of sequent calculus disappears, and this way rules become applicable deep inside a formula(*deep inference*), allowing a formula to move into another formula in a way determined by their local structure. This results in proof theoretical properties that are not available in the sequent calculus, and that are interesting from the view of computation as proof search.

We have implemented the proof search for the systems in the calculus of structures, and also a planner which implements the above ideas in the lines of [5, 6]. These implementations, mainly in system Maude [1], are available[1] for download. However, at the moment they are plausible only for planning problems of toy size.

References

[1] M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, and C. Talcott. The Maude 2.0 system. In Robert Nieuwenhuis, editor, *Rewriting Techniques and Applications, Proceedings of the 14th International Conference,*, volume 2706. Springer, 2003.

[2] U. Engberg and G. Winskel. Completeness results for linear logic on Petri nets. In A. Borzyszkowski and S. Sokolowski, editors, *Proc. of the Conf. on Mathematical Foundations of Computer Science*, pp 442–452, Gdańsk, Poland, 1993. Springer, LNCS 711.

[3] Alessio Guglielmi. *Abstract Logic Programming in Linear Logic Independence and Causality in a First Order Calculus.* PhD thesis, Universita di Pisa – Genova, 1996.

[4] Alessio Guglielmi. A system of interaction and structure. Technical Report WV-02-10, TU Dresden, 2002. to appear in ACM Transactions on Computational Logic.

[5] Ozan Kahramanoğulları. Implementing system BV of the calculus of structures in maude. In Laura Alonso i Alemany and Paul Égré, editors, *Proceedings of the ESSLLI-2004 Student Session*, pages 117–127, Université Henri Poincaré, Nancy, France, 2004.

[6] Ozan Kahramanoğulları. System BV without the equalities for unit. In C. Aykanat, T. Dayar, and I. Körpeoğlu, editors, *Proceedings of the 19th International Symposium on Computer and Information Sciences, ISCIS'04*, volume 3280 of *LNCS*. Springer, 2004.

[7] M. Masseron, C. Tollu, and J. Vauzeilles. Generating plans in linear logic. In *Foundations of Software Technology and Theoretical Computer Science*, volume 472 of *Lecture Notes in Computer Science*, pages 63–75. Springer-Verlag, 1990.

[8] Dale Miller. The π-calculus as a theory in linear logic: Preliminary results. In E. Lamma and P. Mello, editors, *Proceedings of the 1992 Workshop on Extensions to Logic Programming*, number 660 in LNCS, pages 242–265. Springer-Verlag, 1993.

[9] Vladimiro Sassone, Morgens Nielsen, and Glynn Winskel. Models for concurrency: Towards a classification. In *Theoretical Computer Science*, volume 170 (1–2), pages 297–348. 1996.

[10] Lutz Straßburger. MELL in the calculus of structures. *Theoretical Computer Science*, 309:213–285, 2003.

[1] http://www.informatik.uni-leipzig.de/ ˜ozan/maude_cos.html