ELECTRONIC NEWSLETTER ON
REASONING ABOUT ACTIONS AND CHANGE

Issue 98048 Editor: Erik Sandewall 27.5.1998

Today

Debates

Today, we have a second invited comment to the article by Iliano Cervesato, Massimo Franceschet, and Angelo Montanari that has been submitted to the ETAI.

The discussion about the ontology of time has not been concluded yet, and additional contributions are foreseen in a forthcoming issue of this Newsletter.

ETAI Publications

Discussion about received articles

The Complexity of Model Checking in Modal Event Calculi with Quantifiers

by Iliano Cervesato, Massimo Franceschet, and Angelo Montanari

From: Peter Jonsson

Comments and questions to the authors:

Page 4, col 1, para 2: Please find a better name than "condition ()".

Page 6, col 1, para 3: You write "The best approximation of /formula/ we can achieve...". In what sense is it the "best" approximation?

Chapter 4 "Implementation". I agree that the implementation of QCMEC in λ-prolog is very elegant. However, due to my own experience of prolog-like languages, I wonder how efficient the implementation is? I suggest that you include a small empirical investigations as follows: First, implement a QCMEC algorithm in some fast, imperative language such as C. This is straightforward since you have already proved that QCMEC is in PSPACE and your way of showing this (by exhibiting a polynomial-time alternating TM) immediately suggests a recursive algorithm for the QCMEC problem. (Even though such a naive algorithm may not be the fastest algorithm, it is probably efficient enough.) Then, compare your implementation in λ-prolog with the C-implementation on a number of test examples (such as the diagnosis example). Even though this experiment does not prove anything, it can very well give us an idea of the (in)efficiency of the implementations. I will not be surprised if the C-implementation completely outperforms the λ-prolog-implementation. I will be extremely surprised if the λ-prolog-implementation outperforms the C-implementation.

Page 8, before Th. 4.1: You write "We only show the statement of our soundness and completeness result since a fully worked out proof would require a very detailed account of the semantics of λ-prolog". So why implement QCMEC in λ-prolog at all? By implementing a straightforward QCMEC algorithm using a language with a more intuitive semantics, the soundness/completeness proof would be more or less trivial (once again, compare with your PSPACE-membership proof of QCMEC).

Chapter 5 "Complexity Analysis" Why not rearrange the section as follows: Begin the section by proving that QCMEC is in PSPACE. Then there is no need for first showing that CMEC is in PSPACE, extending the result to EQCMEC and finally realize that the proof for EQCMEC also works for QCMEC.

Page 8, col 2, proof of Th. 5.3.: You write "...the algorithm enters in an AND (resp. OR) state. It nondeterministically choose one among α and β and evaluates it in w." This should be rephrased. If you are in an OR state, the sentence is correct but if you are in an AND state, no nondeterministical choice takes place. Instead, the ATM checks that all paths starting in the AND state is an accepting path.

Page 10, col 1, end of proof of Th. 5.3. Please give the proof of why $w \vdash ind(F, 1)$ iff G is true. Or, at least, give the reader a hint...

Page 11, col 1: You write "In Section 4 we have transliterated the definition of QCMEC and its subcalculi in the higher-order logic programming λ-prolog. The directness of the implementation allows checking easily that the complexity of the implemented algorithms coincide with the bounds proved in this section for the problems they implement". One should be extremely careful when justifying complexity results with implementations in logic programming languages. Such languages often introduce large overheads and it is difficult to know the exact complexity of the different operations (unification,...) that are performed by the systems. Your assertion may very well be correct, but it needs further justification.

This newsletter is issued whenever there is new news, and is sent by automatic E-mail and without charge to a list of subscribers, in plaintext format. The present postscript issue is a bona fide rendering of the same information in alternative typography. Instruction for contributors and other additional information is found at the following URL:

`http://www.ida.liu.se/ext/etai/actions/njl/`