ADDITIONAL PAPERS

Improving Firing Control via Explicit Enabling Rules in Petri Nets

R. H. GALLARD, *Facultad de Ciencias Fisico, Matematicas y Naturales, Universidad Nacional de San Luis, Loçal 106, Ejrecito de los Andes 950 – lo. P. – 5700 San Luis, Republica Argentina*

When modelling partial aspects of operating systems, basic Petri nets have proved useful in studying concurrency. However, they are insufficient to determine some performance measures of the system under design. This problem arises, essentially, due to the following limitations; tokens are not distinguishable from one another and transition firings are ungovernable. This paper introduces controlled Petri nets as an extension to overcome those limitations. The extension allows the insertion of entity attributes and provides different, explicit, enabling rules for transition. While the natural expressiveness of Petri nets is preserved, they become suitable to interact with external environments. An example is given to show the flexibility and usefulness of controlled Petri nets to model the behaviour of a simple process manager. (pp. A513–A517)

An Efficient Strategy for the Bottom-up Evaluation of Datalog Queries

N. LEONE AND P. RULLO, *CRAI, 87036 Rende, Italy*

Rewriting methods, such as those based on magic sets, are used to improve the efficiency of the bottom-up evaluation of Datalog queries. The basic idea behind such methods is that of "pushing down" possible constants appearing in query goals, thus restricting the computation of the query answer to only relevant facts. However, it may happen that the rewriting of a stratified Datalog program generates an unstratified program, ie a program where negation occurs "within" recursion. Under such a condition, the convergence of the bottom-up evaluation process is not guaranteed any more.

In this paper we present an algorithm for the efficient bottom-up computation of the stratified semantics of Datalog queries. This algorithm, called the SNQS Algorithm, is based on a combination of the semi-naive method with the query/subquery strategy. Actually it is a simple variant of the "pure" bottom-up approach but it has a larger applicability domain. Indeed, it allows the evaluation of unstratified programs deriving from stratified ones when rewriting techniques are applied, thus supporting a full exploitation of query bindings. (pp. A519–A527)

Errata

It has been drawn to the attention of the Editors that there were a large number of mistakes in the paper 'On the Meaning of Safety and Security' by A. Burns, J. McDermid and J. Dobson, **35** (1), 3–15.

On page 4, column 2, line 4
'out' *should be* 'our'.

On page 4, column 2, section 2.1, lines 16–19

- 'a *safety critical* system is one whose failure could do us immediate, direct harm;
- a *security critical* system is one whose failure could enable, or increase the ability of, others to harm us.'

should be

- 'a **safe** system is one that cannot do us immediate, direct harm *even if it fails*;
- a **secure** system is one that cannot enable others to harm us *even if it fails*.'

On page 4, column 2, lines 22 and 25
'is' (at the end of the lines) *should be* 'us'.

On page 6, column 2, line 29
'when in' *should be* 'when'.

On page 6, column 2, line 30
'upon' *should be* 'open'.

On page 7, column 1, line 31
'even' *should be* 'event'.

On page 7, column 1, line 51
'the' *should be* 'that'.

On page 7, column 1, line 55
'even' *should be* 'event'.

On page 8, column 1, lines 14–15
'these being either ordinary components in an error state is detected.*'
should be
'these being either ordinary components in an error state or components only invoked when an error state is detected.*'.

On page 8, column 1, line 17
'or or' *should be* 'or'.

On page 8, column 2, line 13
'all' *should be omitted.*

On page 8, column 2, line 50
'object' *should be* 'objects'.

On page 8, column 2, line 54
'the' *should be omitted.*

On page 11, column 2, line 24
'to' *should be* 'is'.