Model Checking Strategies Provided by Petri Nets

Model checking (MC) techniques attract attention of computer science for more than 30 years as a way to increase confidence in software system. These techniques rely on Formal Methods which allow one to reason mathematically strictly about system properties. MC has become the popular verification technique partially due to Petri nets (PN) [GV03], a methodology with broad area of applications. Formal language approach gives a number of positive properties of PN which make them interesting for both linguistic and complexity theory [BBF01]. The demands from application areas have resulted in PN extensions: Colored PN, Stochastic PN, Fuzzy PN, Timed PN etc. Their generative capacity varies from recursively enumerable languages to a subset of regular languages and forms an inclusion hierarchy which is relevant to complexity hierarchy [AC99]. MC based on PN in contrast to classical Kripke structure can match to extended logics [BBF01] and give to developer much facilities.

It is encouraging fact that varying PN extensions in many cases one can achieve more flexibility, clear reasoning, conciseness. This is a crucial point of our research to investigate possibility of translation between PN extensions and from object-schemas (OS) [BBF01] into PN with respect to MC. The choice of strategies of MC is a way to decompose system into one or more PN and to check consistency and validity of desired properties. We aim to focus our work in specifications of OS as 1) it has semantic adequate to modern software architecture and 2) it restricts our research in reasonable bounds.

We want to adapt current PN decomposition techniques to OS. To do so we intent to try backward approach i.e. to translate OS from a fixed set into PN, so, provide PN with "fixed semantics".

As result of the research we expect to:
1) Construct a methodology of MC strategies based on PN which well fit to OS.
2) Investigate translation techniques between PN classes and OS (within theory of fixed parameterized tractability).
3) Fulfill the class hierarchy over inclusion for PN languages. To find new results in the computational NP-completeness of PN extensions [AC99].
4) Present the results in international workshops and conferences in formal languages and automates, and, so to retrieve back responses from the theoretical computer science community.

References