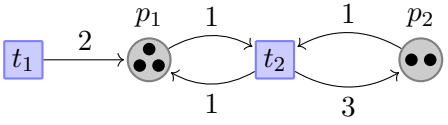
Abstract (for the general public):

Automatic analysis of concurrent systems

Principal Investigator: Sławomir Lasota

Objectives. We are going to improve on methods of automatic analysis of models of concurrent systems, and to enhance its practical applicability. We will specifically concentrate on the model of Petri nets, and investigate questions like *coverability* or *reachability* of a given configuration.

Illustrating example. As an example, consider a Petri net with two places $P = \{p_1, p_2\}$ and two transitions $T = \{t_1, t_2\}$, and its configurations that stores three tokens on place p_1 and two tokens on place p_2:



Every execution of transition t_1 puts additional two tokens on place p_1, while every execution of t_2 takes one token from p_1 and one from p_2, and then puts back one token on place p_1 and three tokens on place p_2. Here is an example *coverability* question: is there a run (i.e., a sequence of executions of transitions) starting in the given configuration which puts *at least* four tokens on p_1 andat least six tokens on p_2? Here is an example *reachability* question: is there a run that puts *exactly* four tokens on p_1 and exactly six tokens on p_2? In our example, coverability holds (in fact, both places p_1 and p_2 are *unbounded*, i.e., there is no finite bound on the number of tokens on these places) but reachability does not; indeed, the parity of the number of tokens on both places is preserves by both transitions.

Motivation. Motivation for this research is twofold. One one hand, there is a number of extremely challenging open theoretical problems, like exact computational complexity of the reachability problem for Petri nets, or decidability status of Petri nets with stack. We hope to successfully solve at least some (special cases) of these theoretical problems. On the other hand, recently observed progress in (often heuristic) algorithms solving the coverability problem calls for checking its applicability; we would like to investigate applicability of coverability solving in formal verification of concurrent programs, a task which is inherently difficult to achieve.