A Banker's Solution for Deadlock Avoidance in FMS With Flexible Routing and Multiresource States : Summary Report

The paper presents a newly defined class of Petri nets called S*PR able to model manufacturing systems. This class is an extension of more conventional petri nets used over the years. These sets model resource allocation systems with routing flexibility in the processing of parts of the nets and a multiset of resources at each processing step. Moreover, it introduces an algorithm based on the Banker's algorithm.

<u>Conventional Petri Nets Explained:</u>

Basically a, Petri net can be formally defined as an ordered triple.

PN = (P, T, F),
where: P is a nonempty set of places;
T is a nonempty set of transitions;
F is a nonempty set of arcs: from a place to a transition or from a transition to a place.

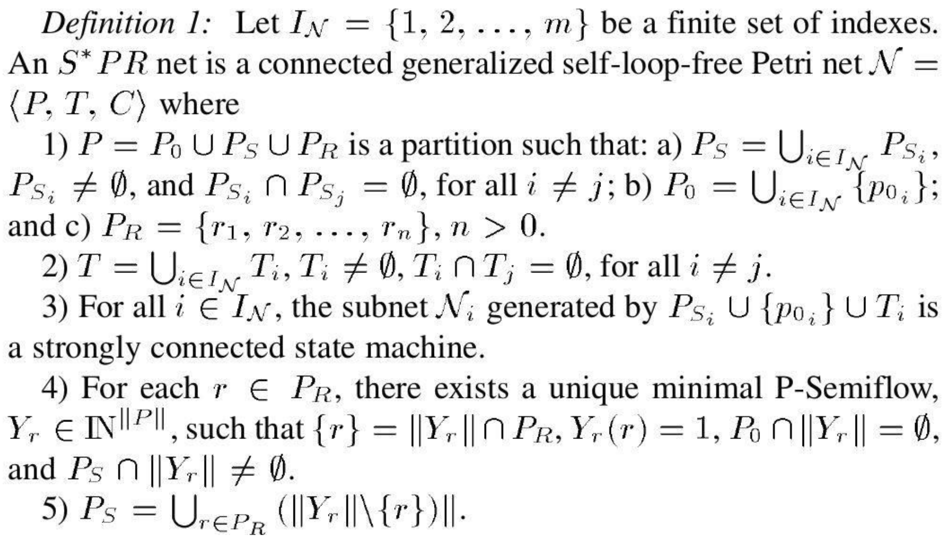
A marking of a net defines the state of the Petri net (the state of the system). A marking is defined as a function M: P{0, 1, 2,…}. It can be considered as a number of tokens situated in the net places. A place containing a token is a marked place. Number of tokens in a place p for marking M is denoted as M(p). Initial marking is denoted as M_0.

<u>Reachable Marking Explained:</u>

A marking can be changed by transition firing. A transition t is enabled and can fire (be executed) if every input place of it (a place from which an arc leads to t) contains a token. Transition firing removes one token from each input place and adds one token to each output place of it (a place to which an arc leads from t). Any marking that can be obtained by continuous firing of these transitions is called a reachable matrix.

S*PQ Petri Nets:

The paper formally defines S*PQ as follows:



Definition 1: Let $I_{\mathcal{N}} = \{1, 2, \ldots, m\}$ be a finite set of indexes. An S^*PR net is a connected generalized self-loop-free Petri net $\mathcal{N} = \langle P, T, C \rangle$ where

1) $P = P_0 \cup P_S \cup P_R$ is a partition such that: a) $P_S = \bigcup_{i \in I_{\mathcal{N}}} P_{S_i}$, $P_{S_i} \neq \emptyset$, and $P_{S_i} \cap P_{S_j} = \emptyset$, for all $i \neq j$; b) $P_0 = \bigcup_{i \in I_{\mathcal{N}}} \{p_{0_i}\}$; and c) $P_R = \{r_1, r_2, \ldots, r_n\}$, $n > 0$.

2) $T = \bigcup_{i \in I_{\mathcal{N}}} T_i$, $T_i \neq \emptyset$, $T_i \cap T_j = \emptyset$, for all $i \neq j$.

3) For all $i \in I_{\mathcal{N}}$, the subnet $\mathcal{N}_i$ generated by $P_{S_i} \cup \{p_{0_i}\} \cup T_i$ is a strongly connected state machine.

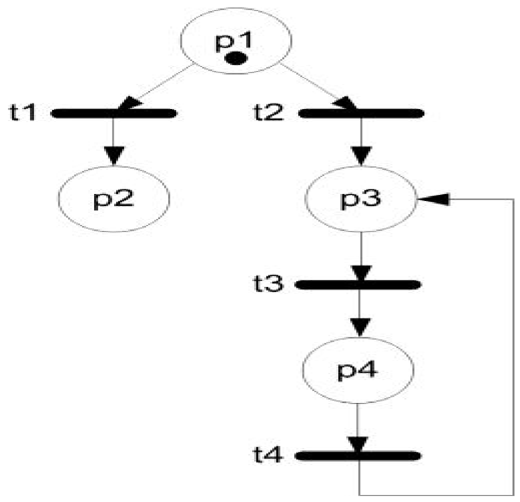
4) For each $r \in P_R$, there exists a unique minimal P-Semiflow, $Y_r \in \mathbb{N}^{||P||}$, such that $\{r\} = ||Y_r|| \cap P_R$, $Y_r(r) = 1$, $P_0 \cap ||Y_r|| = \emptyset$, and $P_S \cap ||Y_r|| \neq \emptyset$.

5) $P_S = \bigcup_{r \in P_R} (||Y_r|| \backslash \{r\})||$.

Mapping it to the previous section, each active process is a token at a state place.
by |M| the number of active processes at marking M: |M| = SUM_p (M(p)). For each reachable marking M, we define A_M to be the set of tokens (set of processes) while $A_M \rightarrow P_S$ is the mapping of processes to the process states.

For a given p ε P_s ,Y_R (p) = k means that k copies of resource r are used by each process (token) in the process state p . Consider a marking M and the case of two state places p, q such that p ∩ q = { t } and M (p) > 0 which means t is process enabled. T is resource enabled at M if and onlyif $M_R + Y_R (p) - Y_R (q) \geq 0$ .This is basically saying that since process need Y_R (q) resource units of R in state q, the sum of units of R process has i.e. Y_R (p) and sum of
free units of R in current marking i.e M_R should be greater than or equal to Y_R (q).

Deadlock in Conventional Petri Nets:



In the above, if t1 is taken instead of t2, we encounter a deadlock, since no transitions can be explored anymore. Vice-versa, no deadlock will ever occur.

Sequential Ordering and Termination Explained:

Sequential termination order means that an ordering in the set of
active processes exists, such that the first process can terminate using its granted resources plus the free ones, the second process can terminate using the resources it holds plus the onesfreed upon termination of the first process, and so on. An active process a is said to be M_R -terminable if and only if there exists a path F a =($q_0 t_1 q_1 \ldots q_{k-1} t_k q_k$) belonging to $T_S(p)$, ($q_0 = Pi_M(a)$; q_k belongs to P_0) , such that
$M[t_1 t_2 \ldots t_k > M^a$.

Unsafe/Safe states in S*PR Nets:

A system is safe when

- There exists an active process able to terminate using the resources it holds plus the available resources. (If no such process exists, the state is considered to be non-safe.)
- If it exists, free its resources and remove it from active processes, and iterate the method. If every process can be removed from the set of active processes, the state is safe.

This is only a **sufficient** condition. It is possible that some safe states exist for which such an ordering as explained in the previous section does not exist.

Algorithm for IsTerminable:

The algorithm verifies whether an active process corresponding to a reachable marking M is M_R-terminable. To repeat in simple English, a process is M_R -terminable (at marking M) if there exists a path joining the state place where the process stays and the corresponding idle state, in such a way that the path can be followed by the process using the free resources plus those it is currently holding.

Algorithm for IsBankerAdmissible:

The algorithm searches for a process (p) in the set of active processes (A_M) in a marking M and applies isTerminable on it. If nothing is found, then the state is unsafe, as discussed previously. If any process is found, restore its resources and remove it from the set of active processes and repeat it. It exploits the fact that if at a given state a process can terminate, all the processes in the same state place can terminate, one after another.

Complexity:

Worst case complexity is polynomial in size of petri-size.