Linear Analysis of Deadlock-Freeness of Petri Net Models

E. Teruel J.M. Colom M. Silva

Dpto. Ing. Eléctrica e Informática
María de Luna 3, E-50015 Zaragoza (Spain)
Tel: + 34 76 51 72 74 Fax: + 34 76 51 29 32
e-mail: eteruel@cc.unizar.es

Abstract

Competition for limited resources in concurrent systems can lead to deadlock states, a situation that is usually ruled out by the specification of reactive systems. This paper presents some new results concerning the structural analysis of deadlock-freeness using Petri net models. Our approach considers the state equation as the basic description of the behaviour of the net system. Actually, this description is a linear relaxation, so, in general, it leads to necessary *or* sufficient conditions on certain behavioural properties of the system. Nevertheless, the method is useful to uncover possible design errors leading to deadlock states, or to prove absence of such problems, and it can greatly reduce the space and time cost associated with the classical techniques based on reachability or even on the computation of the set of minimal invariants.

Keywords: Discrete event systems, manufacturing systems, deadlock-freeness, Petri nets, structural analysis.

1 Introduction

The interest in parallel and distributed systems grows constantly, as their use is extended. *Verification* of properties of good behaviour during the design process, prior to implementation, supported by *efficient tools* becomes crucial due to the overwhelming complexity of these systems. In particular, proving that a concurrent system cannot reach a deadlock condition (i.e. no activity because every process is waiting for some other to continue) may be difficult when the size of the system becomes large and its structure intricate due to complex and perhaps subtle interactions. (Flexible) manufacturing systems typically consist of several active agents competing for resources, so it is indeed a characterisitc domain where the study of deadlocks is specially relevant [BK 90, VNJ 90, ZDD 92]. (Partial) simulation is not an adequate technique to study this problem because it guarantees neither the detection of a possible deadlock nor the proof of its absence. Nevertheless, it is proposed by some authors as a simple and practical approach to foresee the future course of events to try to avoid reaching a deadlock condition [VNJ 90]. On the other hand, exhaustive simulation (or reachability analysis, or model-checking) allows to decide, but soon becomes unfeasible due to the state space explosion problem. Many efforts are directed to palliate this problem, trying to avoid generating the full state space while preserving the ability to decide the property [Valmari 91]. A different approach to deal with the problem is to restrict the class of systems considered and derive *ad hoc* techniques either to design deadlock-free systems [ZDD 92], or to obtain a control policy avoiding, or preventing, the deadlock [BK 90].

We apply here a technique to analyse the existence/absence of deadlocks in general concurrent systems. It gives a sufficient condition for deadlock-freeness (or a necessary one for existence of a deadlock) without generating the state space at all, but reasoning on the structure of the system. The basic technique is simple but still may require a big computational effort, so we introduce several techniques to improve its performance, often reducing the problem to checking a single system of linear inequalities for existence of solutions.

In order to precise what does the proposed technique consist in, let us briefly discuss the model we use and some of the analysis techniques based on it. Parallel and distributed systems can be conveniently modelled by *Petri nets*, which have also a well founded and mature theory of analysis [Peterson 81, BRAMS 83, Silva 85, Murata 89] that allows to check a great number of properties of the system. We assume the reader is familiar with the structure, firing rules, and basic properties of net models. The main concepts are recalled in section 2. One of the indigeneous techniques for the analysis of Petri net models is the so called *structural analysis*. This analysis method focuses on the relationship between the structure and the behaviour of the system/model. In general, structural techniques present a good trade-off between computational efficiency and decision power. Therefore, they can be used as initial decision procedures and eventually avoid resorting to reachability analysis. We consider structural analysis techniques based on linear algebra. The behaviour, which is non-linear in general, is *linearly relaxed* by the *state equation*. In order to study certain properties, they are expressed as a (set of) linear condition(s) and then it is checked if the solutions of the state equation fulfil it. To do so, tipically Integer Linear Programming [NW 88] techniques are used Unfortunately, the state equation has integer solutions that do not correspond to any state reachable on the net system. They are called *spurious* solutions. Due to the existence of spurious solutions, only *necessary* or *sufficient* conditions are obtained when this relaxation is used to study behavioural properties like reachability, boundedness or deadlock-freeness. Spurious solutions can be removed using different approaches [CS 91b]. Moreover, removing the integrality constraints leads, in general, to some more solutions of the state equation which do not correspond to reachable markings, diminishing the decision power, as will be illustrated. Nevertheless this further relaxation could be interesting because it permits using Linear instead of the more complex Integer Programming. There exist some classical structural analysis techniques based on linear algebra using *invariants* on the behaviour of a system instead of the state equation. Traditional invariants-based analysis methods require, in general, the computation of the elementary ones, which is an exponential space hard problem [CS 91a]. In [SC 88] it is shown that a structural analysis based on the state equation: (1) avoids the computation of the set of minimal invariants, because to prove a property the method essentially works making a selective search of the invariant allowing to decide the property; and (2) has greater decision power than the method based on invariants, because from the net state equation all invariants can be generated and in some nets the state equation contains more information than just the invariants.

The goal of this paper is to improve the analysis of deadlock-freeness in general nets by means of linear algebraic techniques. The basic technique (presented in section 3) requires the formulation of a large number of systems of linear inequalities [Colom 89]. Here, several techniques to cut down this computation effort are introduced (in section 4), and finally they are illustrated by means of a couple of examples.

2 Basic Concepts and Notation

In this section we recall the basic concepts and notation that are going to be used. For the sake of readability, whenever a net or system is defined it "inherits" the definition of all the characteristic sets, functions, parameters,... with names conveniently marked to identify whose is which.

A *Place/Transition net* is a triple $\mathcal{N} = (P, T, W)$ where P and T are disjoint sets of *places* and *transitions*, and $W : (P \times T) \cup (T \times P) \mapsto \mathbb{N}$ defines the *weighted flow relation*, which $(\forall p \in P, \forall t \in T)$ can alternatively be defined by:

$$Pre(p,t) \stackrel{\text{def}}{=} W((p,t)) \quad Post(p,t) \stackrel{\text{def}}{=} W((t,p))$$

These functions can be represented by matrices[1]. If $\mathcal{N}$ is *pure* (i.e. $\forall p \in P, \forall t \in T : W((p,t)) \cdot W((t,p)) = 0$; we assume all nets are pure) then the weighted flow relation is represented by the *incidence matrix* $C \stackrel{\text{def}}{=} Post - Pre$.

Let $v \in P \cup T$. Its *preset*, ${}^\bullet v$, and *postset*, $v^\bullet$, are given by ${}^\bullet v \stackrel{\text{def}}{=} \{u \in P \cup T \mid W((u,v)) > 0\}$ and $v^\bullet \stackrel{\text{def}}{=} \{u \in P \cup T \mid W((v,u)) > 0\}$. The preset (postset) of a set of nodes is the union of presets (postsets) of its elements.

A function $M : P \mapsto \mathbb{N}$ is called *marking*, and can be represented by a vector. A *Place/Transition system* (P/T-Σ) is a pair $(\mathcal{N}, M_0)$ where $\mathcal{N}$ is a P/T net and M_0 is the *initial* marking. A transition t is *enabled* at M iff $M \geq Pre[P,t]$. Being enabled, t may *occur* (or *fire*) yielding a new marking $M' = M + C[P,t]$, and this is denoted by $M \xrightarrow{t} M'$ and $M' = M \xrightarrow{t}$. An *occurence sequence* from M is a sequence $\sigma = t_1 \cdots t_k \in T^*$ such that $M \xrightarrow{t_1} M_1 \ldots M_{k-1} \xrightarrow{t_k} M'$, denoted by $M \xrightarrow{\sigma} M'$ and $M' = M \xrightarrow{\sigma}$. The set of all the occurence sequences from M_0 (the *language*) is denoted by $L(\mathcal{N}, M_0)$. The *reachability set* is $R(\mathcal{N}, M_0) \stackrel{\text{def}}{=} \{M_0 \xrightarrow{\sigma} \mid \sigma \in L(\mathcal{N}, M_0)\}$.

$(\mathcal{N}, M_0)$ is *bounded* iff the token content of every place is bounded in every reachable marking. This *safety* property is required if the system has to be implemented. Its structural version, *structural boundedness*, tells about its fulfilment for *any* initial marking. $(\mathcal{N}, M_0)$ is *live* iff every transition can ultimately occur from any reachable marking. $(\mathcal{N}, M_0)$ is *deadlock-free* iff at least one transition is enabled at every reachable marking.

Let $(\mathcal{N}, M_0)$ be a P/T-Σ with incidence matrix C. The *state equation* is an algebraic equation that gives a *necessary* condition for a marking to be reachable: a vector $M \in \mathbb{N}^n$ such that $\exists \vec{\sigma} \in \mathbb{N}^m : M = M_0 + C \cdot \vec{\sigma}$ is said to be *potentially reachable*, denoted by $M \in PR(\mathcal{N}, M_0) \supseteq R(\mathcal{N}, M_0)$. A further relaxation is often done, disregarding the integrality constraints, in order to apply more efficient algorithms.

3 A Sufficient Condition for Deadlock-Freeness

Deadlock-freeness concerns the existence of some activity from any reachable state of the system, and it is often part of the specification of concurrent systems. Moreover it is a necessary condition for liveness, although in general not sufficient. That is, the system can be deadlock-free but not live. When no part of the system can evolve, it is said that the system has reached a state of total deadlock (or *deadlock*, for short). In net system terms, a deadlock corresponds to a marking from which no transition is firable. We give below a basic general sufficient

[1]The rows and columns are indexed by the places and transitions respectively. $A[\Pi, \Theta]$ denotes the submatrix of A corresponding to rows of places in $\Pi \subseteq P$ and columns of transitions in $\Theta \subseteq T$.

condition for deadlock-freeness based on the state equation of the net system. It also contains $|T|$ "complex conditions", each one concerning the non-firability of a transition of the net ($\bigvee_{p \in {}^\bullet t} M[p] < Pre[p,t]$).

Proposition 3.1 [Colom 89] *Let $(\mathcal{N}, M_0)$ be a* P/T-Σ. *If there doesn't exist any solution $(M, \vec{\sigma})$, for the system*

$$\begin{array}{l} M = M_0 + C \cdot \vec{\sigma} \\ M \geq 0, \vec{\sigma} \geq 0 \\ \bigvee_{p \in {}^\bullet t} M[p] < Pre[p,t] \quad \forall t \in T \end{array} \tag{1}$$

then $(\mathcal{N}, M_0)$ is deadlock-free.

Obviously, the "complex conditions" are non linear, because they are expressed using the "$\vee$" connective. Anyway we can express the above condition by means of a set of linear systems as follows. Let $\alpha : T \mapsto P$ be a mapping that assigns to each transition one of its input places. If there doesn't exist α such that the system

$$\begin{array}{l} M = M_0 + C \cdot \vec{\sigma} \\ M \geq 0, \vec{\sigma} \geq 0 \\ M[\alpha(t)] < Pre[\alpha(t), t] \quad \forall t \in T \end{array} \tag{2}$$

has a solution, then $(\mathcal{N}, M_0)$ is deadlock-free. The problem is that we have to check it for *every* mapping α of input places to transitions so we have to check $\prod_{t \in T} |{}^\bullet t|$ systems of linear inequalities. If every transition has exactly one input place (e.g. State Machines) then only one system needs to be checked, but in general the number might be large.

4 Reduction of the Number of Systems to Solve

We aim at reducing as much as we can the large number of systems to be checked, preserving the set of *integer* solutions. For this purpose, we introduce four rules and a transformation. All of them are very simple and easily applicable. The first two rules (1 and 2) allow to disregard some transitions in the model, either because the condition they introduce is known to be redundant or because the transitions are known to be dead, respectively. These rules preserve not only the integer but also the real solutions. The second two rules (3 and 4) can be used to rewrite some "complex conditions" in a simpler way, usually producing the most dramatic reduction in the number of systems to solve, eventually leading to a unique system. The transformation of the net (definition 4.1) is intended to improve the performance of the last two rules in some transitions.

The first rule is net-based, so it can always be applied. The remaining ones require that some places be structurally bounded, and they use the *structural bound* of these places [SC 88], which will be most probably available at this stage of the analysis of the model or can be computed "when needed" if it is not. Let's recall the definition of the structural bound of a place:

$$\begin{array}{rll} SB(p) = & \text{maximize} & M[p] \\ & \text{subject to} & M = M_0 + C \cdot \vec{\sigma} \\ & & M \geq 0, \vec{\sigma} \geq 0 \end{array} \tag{3}$$

The proofs and other details and discussions can be found in [TCS 92].

Rule 1 *Let $\preceq$ be a partial order relation on T defined as: $t_1 \preceq t_2$ if and only if $Pre[P, t_1] \leq Pre[P, t_2]$. Let $\tau = \{t_i \in T \; (i \in \{1, \ldots, k\}) \,|\, i < j \Rightarrow t_i \preceq t_j\}$ be a totally ordered set by the relation $\preceq$. Removing the "complex conditions" corresponding to all transitions in τ but the* least *element, t_1, from system (1) the set of* real *solutions is preserved. The number of linear systems to be solved is divided by $\prod_{j=2}^{k} |{}^\bullet t_j|$. Moreover the new systems have $k-1$ inequalities less than the original ones.*

Rule 2 *Let $\tau = \{t \in T \,|\, t \text{ is dead}\}$. Removing all "complex conditions" corresponding to transitions in τ from system (1) the set of* real *solutions is preserved. The number of linear systems to be solved is divided by $\prod_{t \in \tau} |{}^\bullet t|$. Moreover the new systems have $|\tau|$ inequalities less than the original ones.*

(A simple sufficient condition for a transition t to be dead is $\exists p \in P : SB(p) < Pre[p,t]$. In [TCS 92] a couple of more elaborate sufficient conditions are given.)

Rule 3 *Let t be a transition such that $\forall p \in \pi \subseteq {}^\bullet t : SB(p) = Pre[p,t]$. Replacing in system (1) for the "complex condition" corresponding of transition t the following (less complex) condition:*

$$\left(\sum_{p \in \pi} M[p] < \sum_{p \in \pi} Pre[p,t] \right) \vee \left(\bigvee_{p \in {}^\bullet t \setminus \pi} M[p] < Pre[p,t] \right) \tag{4}$$

the set of integer *solutions is preserved. The number of linear systems to be solved is divided by $\frac{|{}^\bullet t|}{|{}^\bullet t| - |\pi| + 1}$, and their size doesn't change.*

In the *particular case* where $\pi = {}^\bullet t$, the "complex condition" is reduced to a single linear inequality. Therefore only one system of linear inequalities is needed for a structurally 1-bounded system, for instance.

Also when *all but one* of the input places of a transition are such that their structural bound equals to the weight of the arc, the "complex condition" of the transition can be reduced to a single linear inequality, using the following rule:

Rule 4 *Let t be a transition such that ${}^\bullet t = \pi \cup p' \wedge SB(p') > 0 \wedge \forall p \in \pi : SB(p) = Pre[p,t]$. Replacing in system (1) for the "complex condition" corresponding of transition t the following (less complex) condition:*

$$SB(p') \sum_{p \in \pi} M[p] + M[p'] < SB(p') \sum_{p \in \pi} Pre[p,t] + Pre[p',t] \quad (5)$$

the set of integer *solutions is preserved. The number of linear systems to be solved is divided by* $|{}^\bullet t|$*, and their size doesn't change.*

Up to now, the rules allow disregarding or simplifying to a single term (linear condition) the "complex condition" for all transitions t satisfying at least one of the following conditions:

- $\exists t' \in T : Pre[P,t] \geq Pre[P,t']$ (rule 1)
- t is a dead transition (rule 2)
- $\exists \pi \subseteq {}^\bullet t : SB(p \in \pi) = Pre[p,t] \wedge |\pi| \geq |{}^\bullet t| - 1$ (rule 4 and particular case of rule 3)

Rule 3 can also be applied to reduce the number of terms when the subset $\pi \subseteq {}^\bullet t$ does not satisfy the above condition (and has more than one element), although in this case it is not reduced to a single term.

We can still improve the performance of the rules on some transitions not satysfing the above conditions if we preapply a *transformation* to the system which preserves deadlock-freeness (actually it preserves the language modulo a projection). The idea is to substitute in the "difficult transitions" some input places that do not satisfy the equality of the structural bound and the weight by a construct that contains a transition where rule 4 can be applied, and two places, one of them being input of the original transition but now satisfying the equality (see definition 4.1 and figure 1). If it is done, then the number of systems to solve might be dramatically reduced, while their size grows slightly. Naturally we could apply this transformation wherever it was possible right at the beginning, and this would lead to the smallest number of systems wrt. the rules; this "blind" approach can be used to show that for some subclasses of P/T-Σ only one system is required, but of course it is computationally more efficient to apply the transformation just when it is needed, and this is why we present it after the rules.

Definition 4.1 *Let* $(\mathcal{N}, M_0)$ *be a* P/T-Σ*,* $t \in T$ *and* $p \in {}^\bullet t : p^\bullet = \{t\}$*. We define the* (p,t)-expanded system $(\mathcal{N}', M_0')$*:*

- $P' = P \cup \{p_a, p_b\}$
- $T' = T \cup \{t_p\}$
- $W'((p,t)) = 0$, $W'((p,t_p)) = W((p,t))$, $W'((t_p,p_a)) = W'((p_a,t)) = W'((t,p_b)) = W'((p_b,t_p)) = 1$ *and everything else remains unchanged*
- ${M_0}'[p_b] = 1$ *and everything else remains unchanged*

It is clear that $L(\mathcal{N}, M_0) = L(\mathcal{N}', M_0')|_T$, that is, every occurence sequence in $(\mathcal{N}, M_0)$ is firable in $(\mathcal{N}', M_0')$, and viceversa, disregarding the occurences of transition t_p. In particular, it follows immediately that $(\mathcal{N}, M_0)$ is deadlock-free if and only if $(\mathcal{N}', M_0')$ is.

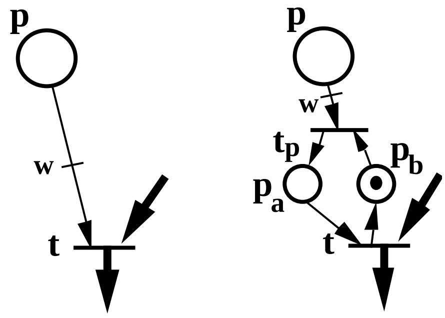

Figure 1: Transformation of the net

Proposition 4.1 *Let* $t \in T$ *and* $\pi \subseteq {}^\bullet t : \forall p \in \pi : p^\bullet = \{t\}$*. If the transformation of definition 4.1 is applied to all places in* π *then the number of systems might be divided (applying rules 3 and 4) by* $|\pi|$*, while* $|\pi|$ *inequalities and* $3|\pi|$ *new variables are added to each of the systems.*

The presented set of rules leads to a *unique* linear system in a wide variety of nets. We have already mentioned structurally 1-bounded systems. Other well-known subclasses that only require one system are Extended Free Choice systems, and Asymmetric Choice systems, also called *Simple* Petri nets. This is because we could apply the transformation of definition 4.1 to every place whose only output transition had more than one input place and after doing so *all* the transitions would enable rule 1, the particular case of rule 3, or rule 4.

In general, the existence of a solution is only necessary for the existence of an actual deadlock, so it needs to be checked later that the deadlock is reachable. Anyway, in some cases the condition is also *sufficient*, if we know that no spurious solution can be a deadlock, as it happens in Equal Conflict systems (which strictly include Extended Free Choice systems). Moreover, in this case liveness is equivalent to deadlock-freeness, provided that the system is bounded, so it is possible to characterize liveness by means of a unique linear system in the integer domain [TS 92]. (For Marked Graphs, since they are a subclass of Free Choice systems, the condition also consists in a single system of inequalities, and it is also necessary and sufficient. Nevertheless, in this particular case there exist more efficient algorithms to analyze deadlock-freeness, namely checking absence of unmarked circuits — is the unmarked subnet acyclic? — or unmarked P-semiflows — a linear programming problem.)

5 Examples

To illustrate the application of the above techniques and ideas to the study of deadlocks and deadlock-freeness, we consider a toy manufacturing system consisting of two

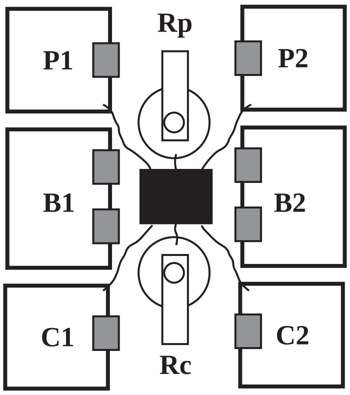

Figure 2: The lay-out of a manufacturing system.

producers, two consumers, two robots and two buffers. Each producer gives parts to its consumer through a buffer (with capacity 2 and 3, respectively). Both producers share a robot, and so do both consumers. The lay-out of the system is depicted in figure 2. We want to design a control program for the robots, so we model their behaviour. In the leftmost net in figure 3 the subnet generated by places $p1$, $p2$ and $p3$ models Rp, that generated by $p8$, $p9$ and $p10$ models Rc, that generated by $p4$ and $p5$ models $B1$ and that generated by $p6$ and $p7$ models $B2$. Robots are supposed to serve their "clients" as requested.

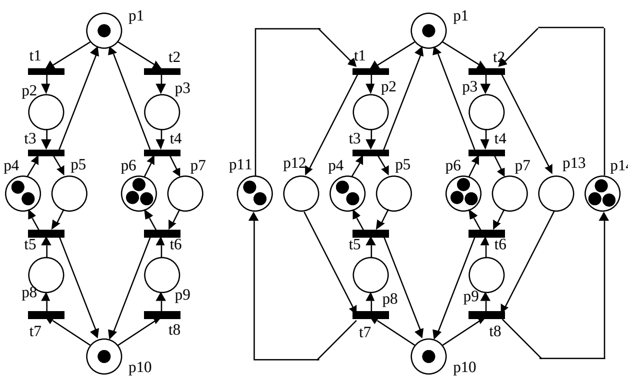

Figure 3: Petri net models of the manufacturing system.

The structural bounds of places are 1 for $p1$, $p2$, $p3$, $p8$, $p9$ and $p10$, 2 for $p4$ and $p5$ and 3 for $p6$ and $p7$. Transitions $t2$ and $t8$ can be disregarded according to rule 1, $t1$ and $t7$ have just one input place each so their "complex condition" is already linear, and rule 4 can be applied to the remaining transitions. Therefore, only one system of inequalities needs to be solved, namely:

$$
\begin{array}{ll}
M = M_0 + C \cdot \vec{\sigma} & \text{(state eq.)} \\
M \geq 0; \vec{\sigma} \geq 0 & \\
M[p1] \leq 0 & (t1) \\
2M[p2] + M[p4] \leq 2 & (t3) \\
3M[p3] + M[p6] \leq 3 & (t4) \\
2M[p8] + M[p5] \leq 2 & (t5) \\
3M[p9] + M[p7] \leq 3 & (t6) \\
M[p10] \leq 0 & (t7)
\end{array}
$$

This system has solutions. For instance, the first phase of the simplex algorithm gives $M = [0\ 1\ 0\ 0\ 2\ 3\ 0\ 0\ 1\ 0]^T$, and $\vec{\sigma} = [3\ 0\ 2\ 0\ 0\ 0\ 0\ 1]^T$, which corresponds to a reachable deadlock (fire $\sigma = t1\ t3\ t1\ t3\ t1\ t8$). The designer would realise that some information on the status of the buffers is needed to allocate the robots. For instance, two status variables can be used, as modelled in the rightmost net in figure 3. Now rule 4 can be applied to every transition. The resulting system has no integer solutions, so it is proven that the model is deadlock-free. It is interesting to point out that, in this case, using Linear Programming wouldn't decide because there are non-integer solutions. In fact, if we hadn't applied rule 4 to transitions $t5$ and $t6$, for instance, then none of the four resulting systems would have had solution, even in the real domain. This illustrates the fact that rules 3 and 4 *may* spoil the decision power when Linear Programming is used to solve.

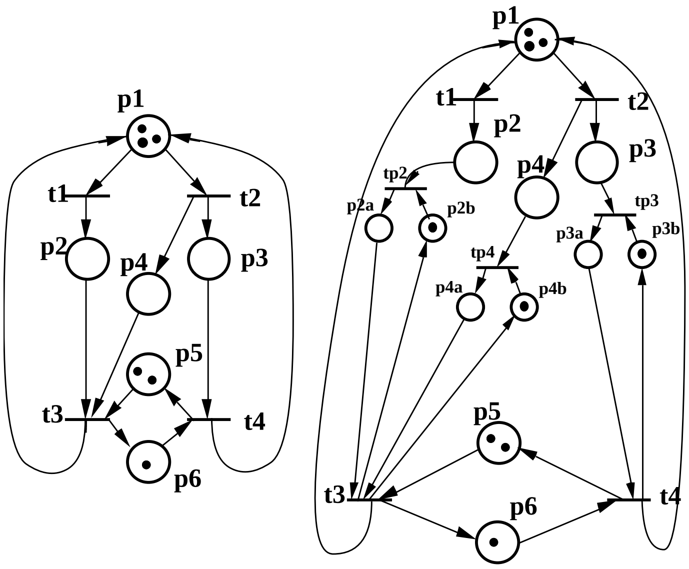

Figure 4: Illustration of the transformation.

Let us further illustrate the transformation with another example. Consider the leftmost system in figure 4. Rule 1 can be applied to transitions $t1$ and $t2$, so in the sequel only $t1$ will be considered. The number of systems has not been reduced because $|{}^\bullet t2| = 1$, but anyway we have removed one constraint. All the structural bounds of places are 3 but that of $p4$ which is 4, so none of the rules is applicable. But, being a Free Choice system, we know that a unique system is required. We apply the transformation to some of the arcs, as shown in figure 4. In the expanded system, rule 4 is applicable to transitions $t3$ and $t4$, and of course also to $tp2$, $tp3$, and $tp4$, so every non-firability condition is made linear. Solving the resulting system with the simplex algorithm the following solution is obtained: $M[p3] = 2$, $M[p3a] = 1$, $M[p4] = 4$, $M[p4a] = 1$, $M[p5] = 3$, $\vec{\sigma}[t2] = 4$, $\vec{\sigma}[tp3] = 2$, $\vec{\sigma}[tp4] = 1$, $\vec{\sigma}[t4] = 1$, and everything else equal to zero. This solution, in terms of the expanded net, leads to the corresponding solution in the original system applying the following trivial steps: (1) Fire backwards all transitions t_p such that place p_b is marked; and (2) Project the marking and the firing count vector on the original nodes. After firing back $tp3$ and $tp4$ we get: $M[p3] = 3$, $M[p4] = 4$, $M[p5] = 3$, $\vec{\sigma}[t2] = 4$, $\vec{\sigma}[t4] = 1$, and everything else equal to zero, which in this case corresponds to an actual deadlock (fire the sequence $\sigma = t2\ t2\ t2\ t4\ t2$).

6 Conclusions

The use of linear algebraic techniques to analyze deadlock-freeness has been presented and illustrated. In order to improve efficiency some information is lost, namely when the behaviour is described by means of the state equation, which is, in general, a linear relaxation. Nevertheless the method considers the greatest amount of structural information; in particular, it considers more information than traditional invariant techniques, and it is still more efficient. The result is a general structural sufficient condition for deadlock-freeness, that can be used to uncover design mistakes or to prove correctness without requiring any simulation or reachability analysis. The basic technique involves checking for solution a large number of systems of linear inequations. The main contribution of this work consists in the presentation of several simple techniques to reduce this number of systems, leading to a unique system in many cases. The presented tecniques can be complemented by others. For instance, the model could be reduced [Silva 85, Berthelot 86] beforehand. To do so, the rules must preserve not only the property to check but also the resolution of the method. For instance, removing implicit places does not affect the behaviour of the net system but could produce spurious solutions, eventually being deadlocks [CS 91b]. Also, the resulting systems could be further reduced applying standard linear algebra techniques (e.g. removing redundant equations) or logical simplifications.

These results can be applied in many different domains. Manufacturing systems are a typical kind of systems where the study of deadlocks is specially relevant. Another example is the tasking behaviour of Ada programs [MSS 89].

Acknowledgements

This work has been partially supported by the projects P IT-6/91 of the Aragonese CONAI (DGA), CICYT ROB-91-0949 of the Spanish Plan Nacional e Investigación, and Esprit W.G. 6067 (CALIBAN).

References

[Berthelot 86] G. Berthelot. Checking Properties of Nets Using Transformations. In *APN 85, LNCS 222*. Springer-Verlag. Berlin.

[BK 90] Z.A. Banaszak, B.H. Krogh. Deadlock Avoidance in Flexible Manufacturing Systems with Concurrently Competing Process Flows. *IEEE Trans. on Robotics and Automation 6-6.*

[BRAMS 83] G.W. BRAMS. *Réseaux de Petri: Théorie et Pratique.* Masson. Paris.

[CCS 90] J.M. Colom, J. Campos, M. Silva. On Liveness Analysis Through Linear Algebraic Techniques. In *Deliverables Covering the Period June 1989 to June 1990, Esprit Basic Research Action 3148 (DEMON), W.G.6.* Paris.

[Colom 89] J.M. Colom. *Análisis Estructural de Redes de Petri. Programación Lineal y Geometría Convexa.* PhD Thesis, Dpto. Ing. Eléctrica e Informática, Universidad de Zaragoza.

[CS 91a] J.M. Colom, M. Silva. Convex geometry and semiflows in P/T nets. A comparative study of algorithms for computation of minimal p-semiflows. In *APN 90, LNCS 483.* Springer-Verlag. Berlin.

[CS 91b] J.M. Colom, M. Silva. Improving the linearly based characterization of P/T nets. In *APN 90, LNCS 483.* Springer-Verlag. Berlin.

[MSS 89] T. Murata, B. Shenker, S.M. Shatz. Detection of Ada Static Deadlocks Using Petri Net Invariants. *IEEE Tr. Software Engineering 15-3.*

[Murata 89] T. Murata. Petri Nets: Properties, Analysis and Applications. *Procs. IEEE 77-4.*

[NW 88] G.L. Nemhauser, L.A. Wolsey. *Integer and Combinatorial Optimization.* Wiley. New York.

[Peterson 81] J.L. Peterson. *Petri Net Theory and the Modeling of Systems.* Prentice-Hall. Englewood Cliffs, N.J.

[SC 88] M. Silva, J.M. Colom. On the Computation of Structural Synchronic Invariants in P/T Nets. In *APN 88, LNCS 340.* Springer-Verlag. Berlin.

[Silva 85] M. Silva. *Las Redes de Petri: en la Automática y la Informática.* AC. Madrid.

[TCS 92] E. Teruel, J.M. Colom, M. Silva. Linear Analysis of Deadlock-Freeness of Petri Net Models. *Research Report GISI-RR-92-29. Dpto. Ing. Eléctrica e Informática. Univ. Zaragoza.* (13 pages)

[TS 92] E. Teruel, M. Silva. Liveness and Home States in Equal Conflict Systems. *Research Report GISI-RR-92-31. Dpto. Ing. Eléctrica e Informática. Univ. Zaragoza.* (To appear in *Application and Theory of Petri Nets 1993.*)

[Valmari 91] A. Valmari. Stubborn Sets for Reduced State Space Generation. In *APN 90, LNCS 483.* Springer-Verlag. Berlin.

[VNJ 90] N. Viswanadham, Y. Narahari, T.L. Johnson. Deadlock Prevention and Deadlock Avoidance in Flexible Manufacturing Systems Using Petri Net Models. *IEEE Trans. on Robotics and Automation 6-6.*

[ZDD 92] M. Zhou, F. DiCesare, A.A. Desrochers. A Hybrid Methodology for Synthesis of Petri Net Models for Manufacturing Systems. *IEEE Trans. on Robotics and Automation 8-3.*