(Coloured)GSPN Models of the UTO1 Example

Rossano Gaeta
Dipartimento di Informatica
Università degli Studi di Torino

1 Multiprocessor description

The parallel architecture contains P processors. Each processor of the multiprocessor system is provided with two cache memories for instructions and data that are not shared with the other processors. Shared data are stored in a single global cache memory, connected to the processors with a multiple bus high speed interconnection network (B is the number of such busses). In order to support the many requests for shared data, the shared cache memory is assumed to be a multiport interleaved memory.

A main memory is connected to the cache via a single main memory bus. Accesses to the main memory take place only when requests for instructions, data, and shared data are not satisfied within the caches. The locality properties of data and programs insure that such memory misses occur only in less than 10% of the references.

Processors execute for random amounts of time of average lengths Z before issuing memory requests. Π represents the probability of accessing instructions, while $1 - \Pi$ is the probability of accessing data. The degree of data sharing is assumed to be S and R is the probability that a memory access is a read (obviously, $1 - R$ represents the probability of a write).

Finally, H is the probability of a cache hit. Cache access times last in the average 1 time unit. When a cache miss occurs, the cache controller selects one cache block to be replaced to make room for the incoming block. The policy for this replacement could be random. A block (private or shared) must be written back in main memory when it has been modified in any cache. Assume that R represents also the probability of replacing non-modified blocks; with probability R a memory access (that keeps busy both the cache and the main memory) lasts K time units; with probability $(1 - R)$ $2K$ time units.

2 Assumptions

In the following we assume that the cache policy is of the copy-back type without buffering that is, a modified block held in a cache is written back to the main memory only upon its replacing (determined by the block replacing algorithm implemented by the cache controller). For the performance evaluation of this model we assume to have a system comprizing ten processors (since the hit ratio is 90%) and two busses for the access to the shared data cache. We will study the performance indices we define in Section 5 varying the degree of sharing of data S among processors.

3 Modeling the Architecture

The SWN model of the multiprocessor architecture is depicted in Figure 1. We use two non ordered colour classes to identify the different objects composing the system: class PROC for the processors which contains P basic objects, and class BUS for the busses which contains B basic objects. All places have colour domain PROC but place AVAIL_BUS whose colour domain is BUS and places SHD_M_H,WAIT_BUS_SH, SH_W_R_HIT,SH_W_R_MISS,S_R_HIT,S_W_HIT,S_R_MS,S_W_MS whose colour domain is PROC,BUS (the Cartesian product of the two colour classes). Place THINK and transition start_acc model the workload of the system (the rate of transition start_acc is the inverse of the average length of execution time).

A probabilistic choice is modeled using an immediate transition for each possibility with an associated weight equal to the probability of the choice; the transitions share the same input place.

Immediate transition instr (with associated weight Π) and immediate transition data (whose weight is $1 - \Pi$) model the randomness in accessing data or instructions with a given probability. Immediate transition private and immediate transition shared model the possible access to a data cache.

Immediate transition pr_hit (sh_hit and in_hit) and immediate transition pr_miss (sh_miss and in_miss) model the possibility of having a hit or a miss in accessing the cache (for instructions, private or shared data). Transition p_rd_h (s_rd_h) and transition p_wr_h (s_wr_h) model the probability of having a read or a write hit; the corresponding timed transitions p_r_hit (s_r_hit) and p_w_hit (s_w_hit) model the time required to perform these operations. Since the cache policy is of the copy back type a read hit and a write hit take the same amount of time (here assumed to be 1 unit long). Immediate transition get_bus_pr (get_bus_sh and get_bus_inst) models the acquisition of the global bus connecting the caches to the main memory. A token in place MEMORY_BUS represents the availability of the bus. Immediate transition p_rd_m (s_rd_m) and timed transition p_r_ms (s_r_ms) model a miss in the cache which involves only one access to the main memory since the block chosen by the replacing algorithm is not modified. Transitions p_wr_m (s_wr_m) and p_w_ms (s_w_ms) model a miss which requires two accesses to the main memory to write back a modified block.

In the case of an access to the shared cache, transition get_bus models the acquisition of one of the B high speed busses which will be released upon the firing of one among the transitions s_r_hit , s_w_hit, s_r_ms and s_w_ms. Since an access to the instruction cache is always a read only operation, the time required for a hit is one unit long (timed transition in_ht), while a miss involves only one access to the main memory which takes K units of time to be performed (timed transition in_ms).

4 Reduction of the SWN model

We can apply to the SWN model described in the previous section several reduction techniques devised to reduce the complexity of the analysis of coloured models. In particular we can apply the *decolourization* technique presented in [4]. All transitions of the SWN model in Figure 1 except get_bus have one input arc labelled with a projection function (x and b) while get_bus has two input arcs labelled with different projection functions. This is a sufficient condition for the application of the decolour-

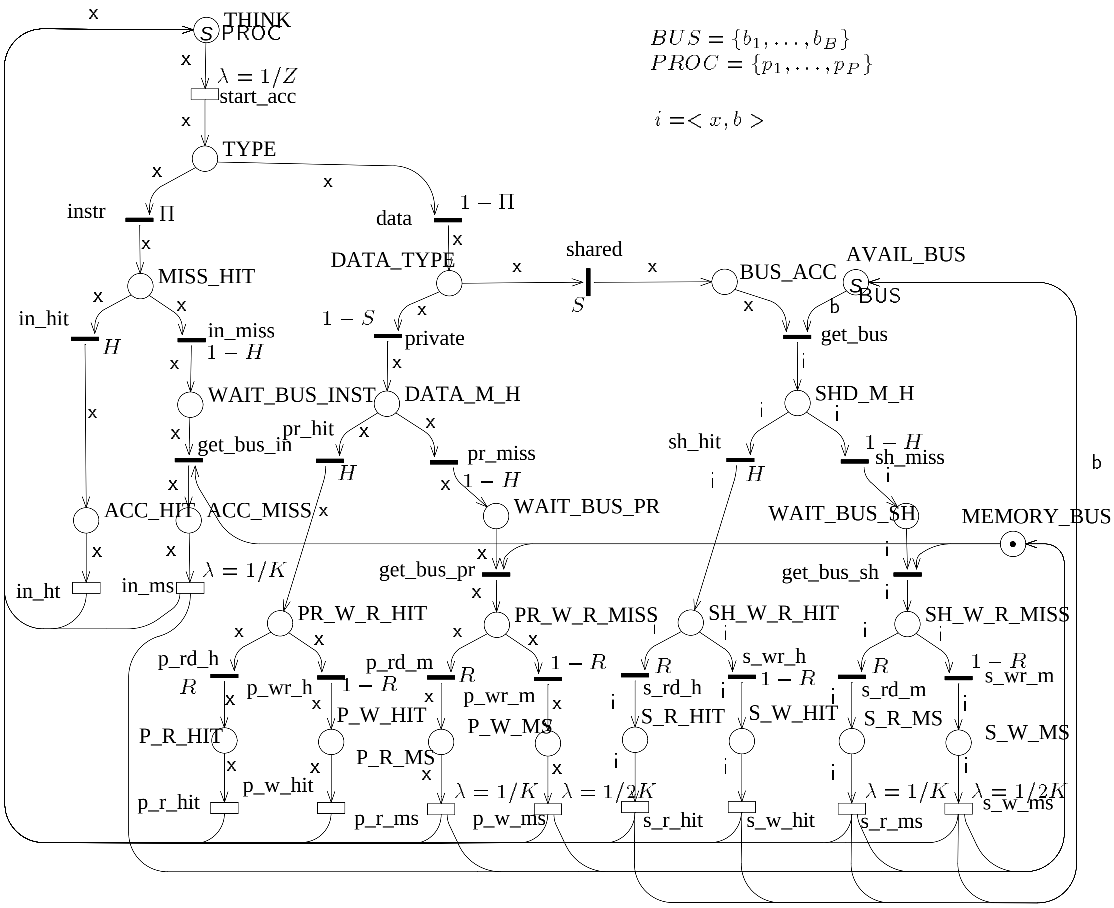

Figure 1: SWN model of the multiprocessor architecture

ization technique. Intuitively, decolourization means stripping away from the model all the informations about redundant colour classes. In this case the classes of the processors and busses that are useful from a modeling point of view are instead redundant from the analysis point of view. The only relevant information about this type of entities is their number. Applying decolourization to the model shown in Figure 1 yields a non coloured (P/T) Petri net model whose structure is unchanged, where the initial marking of place THINK is P (non coloured) tokens and the initial marking of place AVAIL_BUS is B (non coloured) tokens.

Some classical structural reduction techniques (as those described in [1]) can be applied to the decolourized model obtaining the more compact model of Figure 2. In this model the timed transitions with the same rate and the same output places have been folded, thus reducing the size of the model.

Another reduction technique (not shown here) concerning the reduction of immediate transitions described in [3] can be applied to the set of transitions {instr,data,private,shared,in_hit,in_miss} in order to obtain a single free choice conflict among immediate transitions thus reducing the size of the vanishing state space.

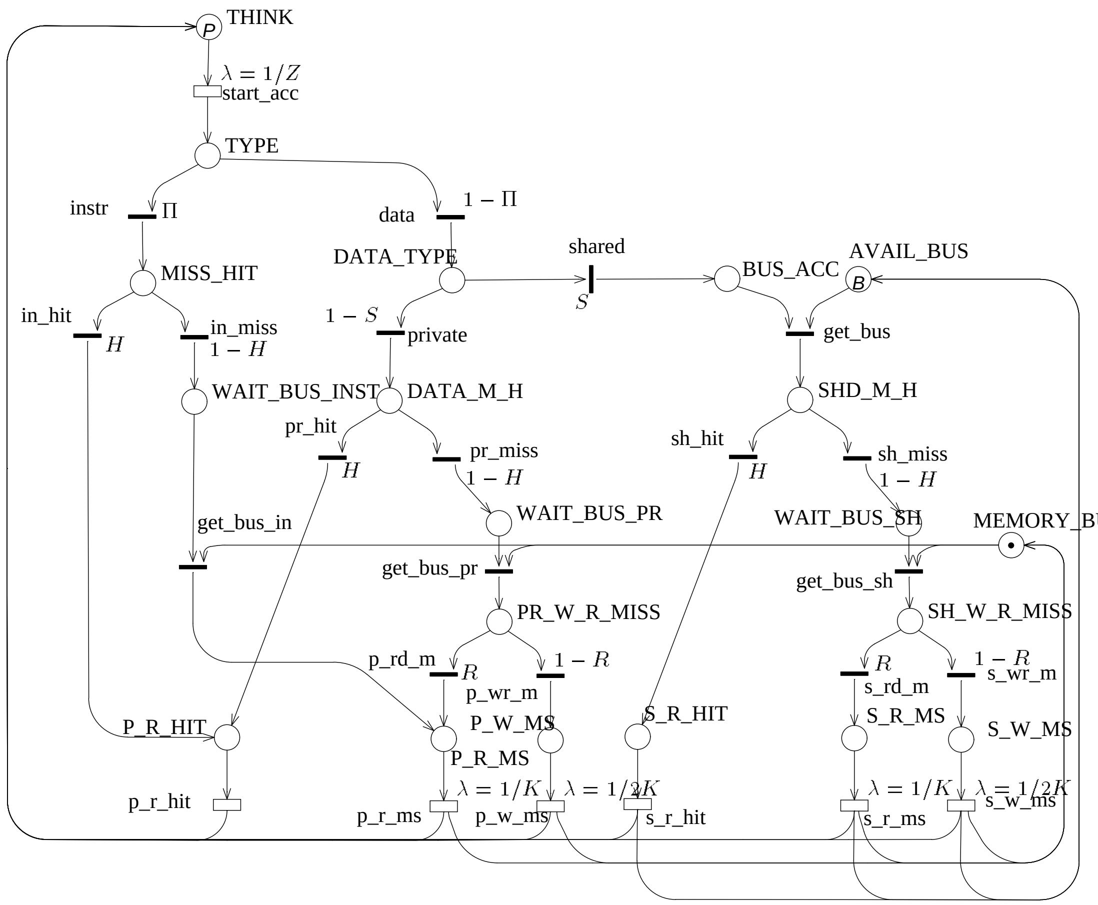

Figure 2: Cache system model(reduced)

5 Performance Evaluation

In this section we show how to define some interesting performance indexes in terms of information that can be obtained by the analysis of Petri net model of Figure 2.

The first performance index we define is the utilization of the global bus connecting the caches to the main memory i.e.:

$$1 - \overline{M}(\mathsf{MEMORY\_BUS}) \tag{1}$$

where $\overline{M}(\mathsf{MEMORY\_BUS})$ is the mean number of tokens in place $\mathsf{MEMORY\_BUS}$. The mean number of busses used to access the shared data cache can be written as

$$B - \overline{M}(\mathsf{AVAIL\_BUS}) \tag{2}$$

while the utilization of each bus is

$$\frac{B - \overline{M}(\mathsf{AVAIL\_BUS})}{B}. \tag{3}$$

We can also define the mean number of processors slowed down by cache misses as

$$\overline{M}(\mathsf{WAIT\_BUS\_INST}) + \overline{M}(\mathsf{WAIT\_BUS\_PR}) + \overline{M}(\mathsf{WAIT\_BUS\_SH}) + (1 - H)\overline{M}(\mathsf{BUS\_ACC}) \tag{4}$$

and the waiting time to access the shared data as

$$\frac{\overline{M}(\mathsf{BUS\_ACC}) + \overline{M}(\mathsf{WAIT\_BUS\_SH}) + \overline{M}(\mathsf{S\_R\_HIT}) + \overline{M}(\mathsf{S\_R\_MS}) + \overline{M}(\mathsf{S\_W\_MS})}{X(\mathsf{shared})} \tag{5}$$

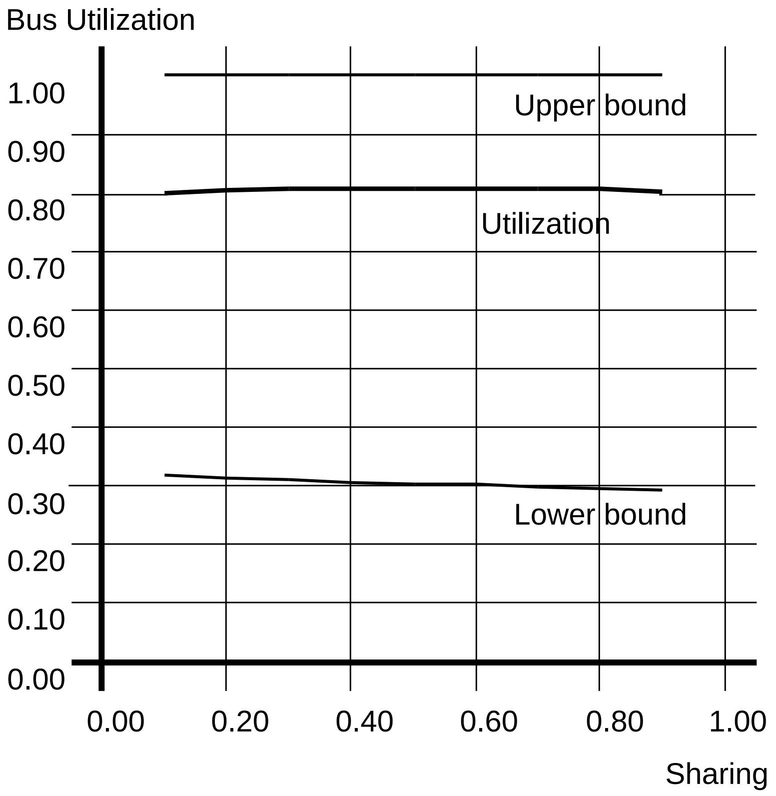

Figure 3: Utilization of the global bus VS. degree of sharing

It is important to note that the definition of these performance indexes is preserved by the decoulorization technique we have applied.

The exact results for the performance indexes 1, 2, 5 are shown in the diagrams in Figures 3, 4, and 5; they have been obtained using the *GreatSPN1.5* package [5] taking 6 minutes (40 seconds for each point) of CPU time on a SUN SPARC ELC with 16 MByte of main memory. The diagram in Figure 3 shows also the lower and upper bound (whose computation technique will be shortly outlined below).

From these diagrams we see that both the mean number of busses used to access the shared data cache and the waiting time to access shared data grow linearly as the degree of sharing of data among processors increases. On the contrary, the utilization of the global bus si slightly affected by S (it directly depends on H) and reaches a maximum value for $S = 0.5$.

Another technique that can be applied when the size of the system increases is the computation of *bounds* [2] for the performance indexes such as the throughput of transitions or the mean number of tokens of places. The solution is obtained by solving a linear programming problem involving relations and constraints about markings, P-invariants, firing rates and some additional constraints induced by the model structure. The set of relations and constraints used for our model are shown in Table 1. The upper

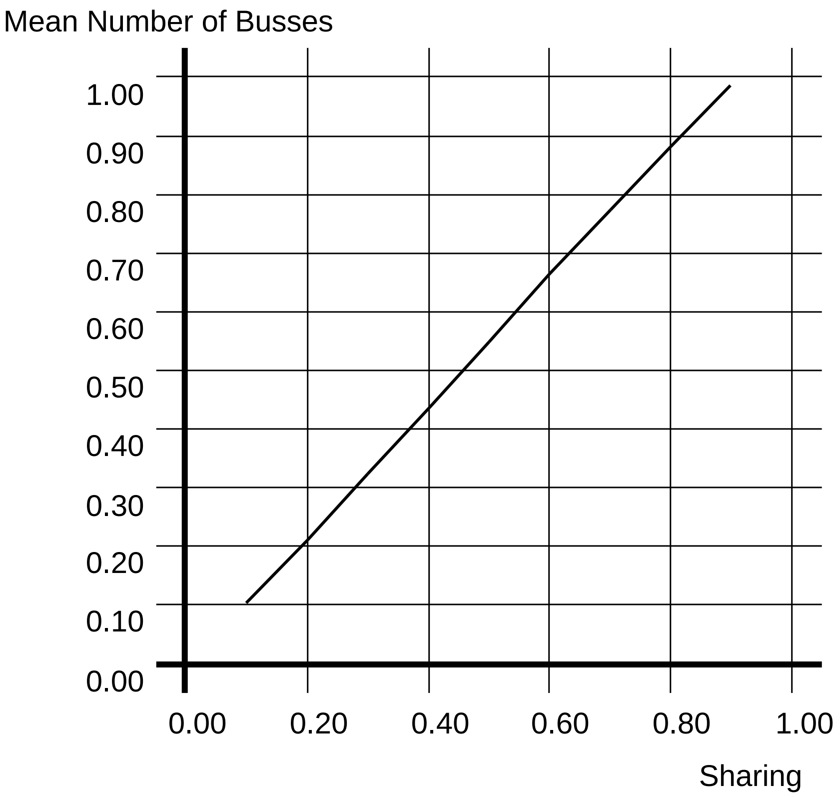

Figure 4: Mean number of shared busses VS. degree of sharing

bound for the global bus utilization (in the case of a system composed of 30 processors, 4 busses and $S = 0.5$) is 1 while the lower bound is 0.3193. The computation of such bounds has taken 3.8 seconds while the exact solution could not be computed due to the exponential growth of the size of the reachability graph. Two evident advantages of this technique are the possibility of dealing with arbitrary distributions for the transition firing times and the low (and independent of the initial marking) computational cost to get the solutions.

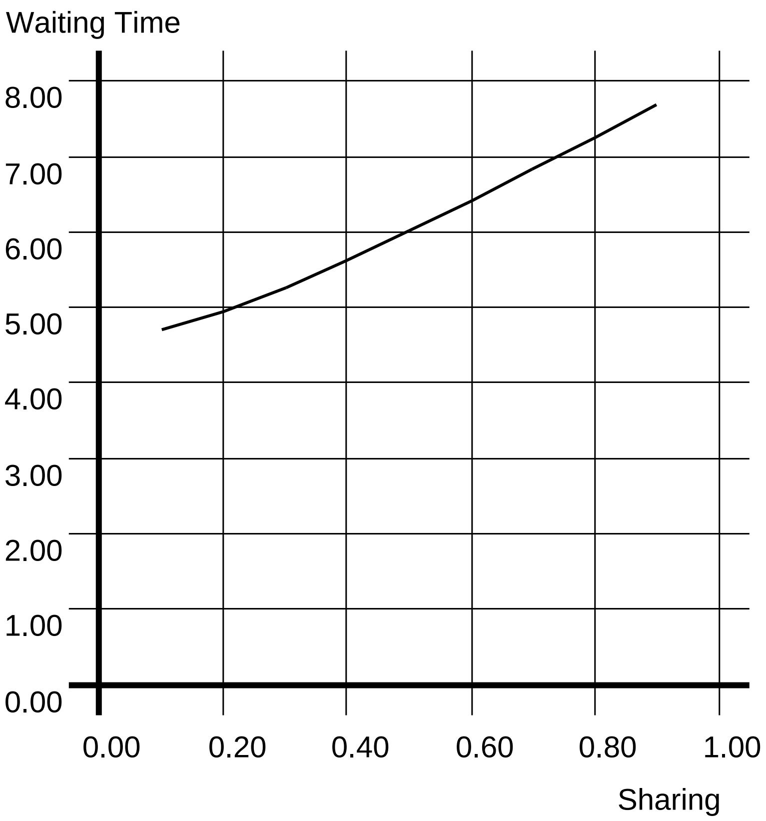

Figure 5: Waiting time to access shared data VS. degree of sharing

$\bullet M(\mathsf{THINK}) + M(\mathsf{P\_R\_HIT}) + M(\mathsf{WAIT\_BUS\_INST}) + M(\mathsf{WAIT\_BUS\_PR}) + M(\mathsf{BUS\_ACC}) +$
$+ M(\mathsf{WAIT\_BUS\_SH}) + M(\mathsf{P\_R\_MS}) + M(\mathsf{P\_W\_MS}) + M(\mathsf{S\_R\_HIT}) + M(\mathsf{S\_R\_MS}) +$
$+ M(\mathsf{S\_W\_MS}) = P$
$\bullet M(\mathsf{AVAIL\_BUS}) + M(\mathsf{WAIT\_BUS\_SH}) + M(\mathsf{S\_R\_MS}) + M(\mathsf{S\_W\_MS}) + M(\mathsf{S\_R\_HIT}) = B$
$\bullet M(\mathsf{MEMORY\_BUS}) + M(\mathsf{P\_W\_MS}) + M(\mathsf{P\_R\_MS}) + M(\mathsf{S\_R\_MS}) + M(\mathsf{S\_W\_MS}) = 1$
$\bullet M(\mathsf{BUS\_ACC}) \leq P - B$
$\bullet M(\mathsf{WAIT\_BUS\_INST}) \leq P - 1$
$\bullet M(\mathsf{WAIT\_BUS\_PR}) \leq P - 1$
$\bullet X(\mathsf{p\_r\_hit}) = H \cdot (\Pi + (1-\Pi)(1-S)) \cdot X(\mathsf{start\_acc})$
$\bullet X(\mathsf{p\_r\_ms}) = (1-H) \cdot (\Pi + (1-\Pi)(1-S) \cdot R) \cdot X(\mathsf{start\_acc})$
$\bullet X(\mathsf{p\_w\_ms}) = (1-H)(1-\Pi)(1-S)(1-R) \cdot X(\mathsf{start\_acc})$
$\bullet X(\mathsf{s\_r\_hit}) = H \cdot (1-\Pi) \cdot S \cdot X(\mathsf{start\_acc})$
$\bullet X(\mathsf{s\_r\_ms}) = (1-H)(1-\Pi) \cdot S \cdot R \cdot X(\mathsf{start\_acc})$
$\bullet X(\mathsf{s\_w\_ms}) = (1-H)(1-\Pi) \cdot S \cdot (1-R) \cdot X(\mathsf{start\_acc})$
$\bullet X(\mathsf{start\_acc}) \cdot Z = M(\mathsf{THINK})$
$\bullet X(\mathsf{p\_r\_hit}) = M(\mathsf{P\_R\_HIT})$
$\bullet X(\mathsf{p\_r\_ms}) \cdot K = M(\mathsf{P\_R\_MS})$
$\bullet X(\mathsf{p\_w\_ms}) \cdot 2 \cdot K = M(\mathsf{P\_W\_MS})$
$\bullet X(\mathsf{s\_r\_hit}) \cdot 2 = M(\mathsf{S\_R\_HIT})$
$\bullet X(\mathsf{s\_r\_ms}) \cdot K = M(\mathsf{S\_R\_MS})$
$\bullet X(\mathsf{s\_w\_ms}) \cdot 2 \cdot K = M(\mathsf{S\_W\_MS})$
$\bullet M(\mathsf{AVAIL\_BUS}) - B + \frac{B}{P-B} \cdot M(\mathsf{BUS\_ACC}) \leq 0$
$\bullet M(\mathsf{MEMORY\_BUS}) - 1 + \frac{1}{P-1} \cdot M(\mathsf{WAIT\_BUS\_INST}) \leq 0$
$\bullet M(\mathsf{MEMORY\_BUS}) - 1 + \frac{1}{P-1} \cdot M(\mathsf{WAIT\_BUS\_PR}) \leq 0$
$\bullet M(\mathsf{MEMORY\_BUS}) - 1 + \frac{1}{B} \cdot M(\mathsf{WAIT\_BUS\_SH}) \leq 0$

Table 1: Relations and constraints for the bounds computation

References

[1] G. Berthelot. Transformations and decompositions of nets. In W. Brawer, W. Reisig, and G. Rozenberg, editors, *Advances on Petri Nets '86 - Part I*, volume 254 of *LNCS*, pages 359–376. Springer Verlag, Bad Honnef, West Germany, February 1987.

[2] G. Chiola, J. Campos, J.M. Colom, and M. Silva. Operational analysis of timed Petri nets and application to the computation of performance bounds. submitted for publication, 1993.

[3] G. Chiola, S. Donatelli, and G. Franceschinis. GSPN versus SPN: what is the actual role of immediate transitions? In *Proc. 4th Intern. Workshop on Petri Nets and Performance Models*, pages 20–31, Melbourne, Australia, December 1991. IEEE-CS Press.

[4] G. Chiola and G. Franceschinis. A structural colour simplification in Well-Formed coloured nets. In *Proc. 4th Intern. Workshop on Petri Nets and Performance Models*, pages 144–153, Melbourne, Australia, December 1991. IEEE-CS Press.

[5] Giovanni Chiola. *GreatSPN* 1.5 software architecture. In *Proc.* 5^{th} *Int. Conf. Modeling Techniques and Tools for Computer Performance Evaluation*, Torino, Italy, February 1991.