Hubert Garavel and Quentin Nivon
{hubert.garavel, quentin.nivon}@inria.fr

This form is a summary description of the model entitled "Erlangen Mainframe V0" proposed for the Model Checking Contest @ Petri Nets. Models can be given in several instances parameterized by scaling parameters. Colored nets can be accompanied by one or many equivalent, unfolded P/T nets. Models are given together with property files (possibly, one per model instance) giving a set of properties to be checked on the model.

Description

This model represents the Erlangen mainframe [1] [2] [3], a multiprocessor system designed to serve two purposes: (i) it maintains an important database and therefore has to process transactions submitted by many *users*, and (ii) it is used for program development and has to provide computing capacity to *programmers* for compiling and testing their programs. In addition, two interesting features are present:

- *Failures* may cause system downtimes, by making the mainframe become unavailable until it is repaired.
- Two types of *priorities* are built into the system. Database users need immediate reaction, so they *explicitly* have priority over the jobs issued by programmers. Failures cannot be preempted, which implies that they are neither buffered nor delayed; thus, they *implicitly* have the highest priority and take down the system immediately, until repair.

The description of the mainframe is highly modular and hierarchical (see Fig. 1). On the topmost level, the system is the parallel composition of three parts, the *Loads*, the *Queues*, and the *Processors*:

- *Loads*: There are three different arrival streams that put load on the system, namely the database *users*, the *programmers*, and *failures*. Each of these arrival streams produces events according to a given arrival rate. This rate however is not constant, but is instead modelled to vary according to a so-called Markov Modulated Poisson Process e. This means that each arrival stream has multiple phases (as in morning-afternoon-evening-night), and changing the phase comes with a change in arrival rate. The phase changes are governed by yet another rate, and happen synchronously across the streams; this is achieved by synchronising the three load processes, that otherwise run independently in parallel, on the phase change.
- *Queues*: The mediation between the events arriving from the loads and the processors is handled by three queues. The *user_queue* buffers the jobs generated by database users; the *prog_queue* buffers the jobs generated by programmers; the *fail_queue* reacts to failure events by triggering repairs. The priority mechanisms discussed above are implemented using clever synchronisations, ensuring that programmer jobs are only served if no user jobs are pending. Both types of jobs are, of course, processed only if the mainframe is not in a failure state.
- *Processors*: This part of the mainframe represents a multiprocessor consisting of four identical processors that run in parallel. The processors synchronise altogether on failures and repairs, meaning that failures affect the entire system, halting all processors, until repair. The various interactions between the components described above (which will be represented by *processes* in our models) are shown in Fig. 1. Transitions performed simultaneously by multiple processes are modelled using *synchronisation*. In Fig. 1, the black bullets denote synchronisations, annotated by the number of processes participating and by an associated *action* (e.g., *prog_job* or *get_prog_job*). Furthermore, white triangles indicate situations where there is competition between several processes (here, always those modelling the four processors) to participate as a process in such a synchronisation.

The Erlangen mainframe was originally described using the TIPP process algebra [1] [2] and, more recently, using both the PRISM/Storm input language and LNT [3]. All these descriptions, which we refer to as version V0 of the Erlangen mainframe, use a "compound" modelling semantics in which transitions carry both an event and a stochastic rate.

We provide 12 different instances of the Erlangen mainframe version V0, by varying the number P of processors between 4 (which was the default value used in papers [1] [2] [3]) and 13. When P is greater than 9, the size of the corresponding PNML files gets too large and exceeds the limits stated by the MCC rules. To avoid this issue, we reduce the number of transitions by replacing certain processors by simpler versions that only accept high priority *user_job* requests, while rejecting lower priority *prog_job* requests. Each instance is denoted by a pair (P, C), where C (with $C \leq P$) is the number of "complete" processors that handle both requests, while $P - C$ is the number of simplified processors in each instance.

All these instances were originally expressed in LNT, a modern language that can be translated to LOTOS automatically. Each generated LOTOS specification was then translated to an interpreted Petri net using the CADP toolbox. A P/T net was then obtained by stripping out all data-related information (variables, types, assignments, guards, etc.) from the interpreted Petri net, leading to a NUPN (Nested-Unit Petri Net) model translated to PNML using the CÆSAR.BDD tool.

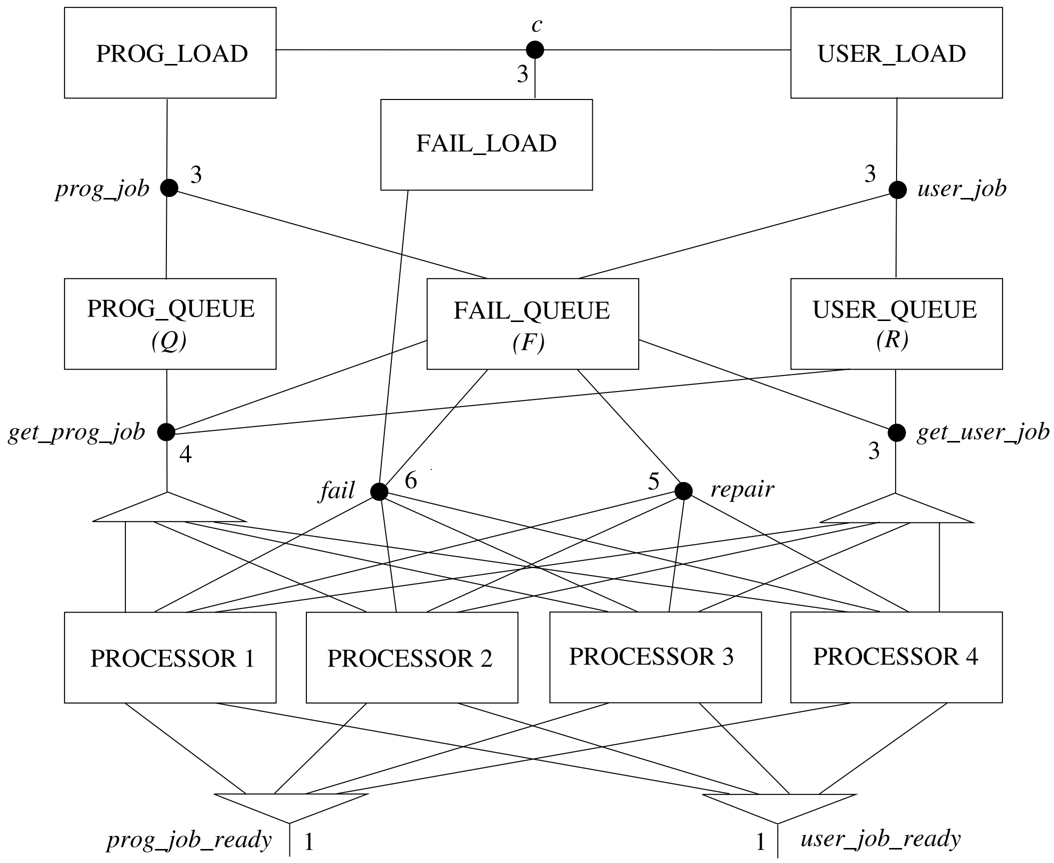

Figure 1: Representation of the original Erlangen mainframe (i.e., with $P = 4$ processors.)

References

1. U. Herzog, V. Merksiotakis. *"Stochastic Process Algebras Applied to Failure Modelling."* Proceedings of the 2nd Workshop on Process Algebras and Performance Modelling (PAPM'94), Regensberg/Erlangen, Germany. pp. 107–126, 1994.

2. H. Hermanns, U. Herzog, V. Merksiotakis. *"Stochastic Process Algebras as a Tool for Performance and Dependability Modelling"* Proceedings of the International Computer Performance and Dependability Symposium (IPDS'95), Erlangen, Germany. pp. 102–111, 1995.

3. H. Garavel, H. Hermanns, D. Parker. *"Revisiting a Pioneering Concurrent Stochastic Problem: The Erlangen Mainframe."* Principles of Verification: Cycling the Probabilistic Landscape. LNCS vol. 15261, 2024. https://doi.org/10.1007/978-3-031-75775-4_3

Scaling parameter

Parameter name	Parameter description	Chosen parameter values
(P, C)	P is the total number of processors and C is the number of "complete" processors able to execute both types of jobs.	$\{4...14\} \times \{0...9\}$

Hubert Garavel and Quentin Nivon
{hubert.garavel, quentin.nivon}@inria.fr

Size of the model

Parameter	Number of places	Number of transitions	Number of arcs	Number of units	HWB code
$P = 4, C = 4$	183	445	3331	19	6–10–56
$P = 5, C = 5$	203	949	10344	21	7–11–62
$P = 6, C = 6$	223	2425	34205	23	8–12–68
$P = 7, C = 7$	243	6817	114442	25	9–13–74
$P = 8, C = 8$	263	19957	381303	27	10–14–80
$P = 9, C = 9$	283	59341	1260524	29	11–15–86
$P = 10, C = 5$	273	23608	545089	31	12–16–87
$P = 10, C = 6$	279	35278	817267	31	12–16–88
$P = 10, C = 7$	285	52780	1225525	31	12–16–89
$P = 10, C = 8$	291	79030	1837903	31	12–16–90
$P = 11, C = 4$	281	31390	788748	33	13–17–91
$P = 11, C = 5$	287	46948	1182750	33	13–17–92
$P = 11, C = 6$	293	70282	1773744	33	13–17–93
$P = 12, C = 1$	277	18712	504563	35	14–18–93
$P = 12, C = 2$	283	27934	756485	35	14–18–94
$P = 12, C = 3$	289	41764	1134359	35	14–18–95
$P = 12, C = 4$	295	62506	1701161	35	14–18–96
$P = 13, C = 1$	291	37156	1082128	37	15–19–98
$P = 13, C = 2$	297	55594	1622818	37	15–19–99
$P = 14, C = 0$	299	49450	1540891	39	16–20–102

Structural properties

ordinary — *all arcs have multiplicity one* .... yes
simple free choice — *all transitions sharing a common input place have no other input place* .... no [a]
extended free choice — *all transitions sharing a common input place have the same input places* .... no [b]
state machine — *every transition has exactly one input place and exactly one output place* .... no [c]
marked graph — *every place has exactly one input transition and exactly one output transition* .... no [d]
connected — *there is an undirected path between every two nodes (places or transitions)* .... yes [e]
strongly connected — *there is a directed path between every two nodes (places or transitions)* .... no [f]
source place(s) — *one or more places have no input transitions* .... yes [g]
sink place(s) — *one or more places have no output transitions* .... ? [h]
source transition(s) — *one or more transitions have no input places* .... no [i]
sink transitions(s) — *one or more transitions have no output places* .... no [j]
loop-free — *no transition has an input place that is also an output place* .... yes [k]
conservative — *for each transition, the number of input arcs equals the number of output arcs* .... no [l]
subconservative — *for each transition, the number of input arcs equals or exceeds the number of output arcs* .... no [m]
nested units — *places are structured into hierarchically nested sequential units* [n] .... yes

[a] stated by CÆSAR.BDD version 3.7 on all 20 instances (11 values of P × {1,2} values of C).
[b] stated by CÆSAR.BDD version 3.7 on all 20 instances (11 values of P × {1,2} values of C).
[c] stated by CÆSAR.BDD version 3.7 on all 20 instances (11 values of P × {1,2} values of C).
[d] stated by CÆSAR.BDD version 3.7 on all 20 instances (11 values of P × {1,2} values of C).
[e] stated by CÆSAR.BDD version 3.7 on all 20 instances (11 values of P × {1,2} values of C).
[f] stated by CÆSAR.BDD version 3.7 on all 20 instances (11 values of P × {1,2} values of C).
[g] stated by CÆSAR.BDD version 3.7 on all 20 instances (11 values of P × {1,2} values of C).
[h] stated by CÆSAR.BDD version 3.7 to be true on 1 instance(s) out of 20, and false on the remaining 19 instance(s).
[i] stated by CÆSAR.BDD version 3.7 on all 20 instances (11 values of P × {1,2} values of C).
[j] stated by CÆSAR.BDD version 3.7 on all 20 instances (11 values of P × {1,2} values of C).
[k] stated by CÆSAR.BDD version 3.7 on all 20 instances (11 values of P × {1,2} values of C).
[l] stated by CÆSAR.BDD version 3.7 on all 20 instances (11 values of P × {1,2} values of C).
[m] stated by CÆSAR.BDD version 3.7 on all 20 instances (11 values of P × {1,2} values of C).
[n] the definition of Nested-Unit Petri Nets (NUPN) is available from `http://mcc.lip6.fr/nupn.php`

Hubert Garavel and Quentin Nivon
{hubert.garavel, quentin.nivon}@inria.fr

Behavioural properties

safe — *in every reachable marking, there is no more than one token on a place* .... yes (o)
dead place(s) — *one or more places have no token in any reachable marking* .... ? (p)
dead transition(s) — *one or more transitions cannot fire from any reachable marking* .... ? (q)
deadlock — *there exists a reachable marking from which no transition can be fired* .... ? (r)
reversible — *from every reachable marking, there is a transition path going back to the initial marking* .... ?
live — *for every transition t, from every reachable marking, one can reach a marking in which t can fire* .... ? (s)

Size of the marking graphs

Parameter	Number of reachable markings	Number of transition firings	Max. number of tokens per place	Max. number of tokens per marking
$P = 4, C = 4$	9.08745e+10 (t)	?	1	10
$P = 5, C = 5$	1.23047e+12 (u)	?	1	11
$P = 6, C = 6$	1.67542e+13 (v)	?	1	12
$P = 7, C = 7$	2.29325e+14 (w)	?	1	13
$P = 8, C = 8$	3.15375e+15 (x)	?	1	14
$P = 9, C = 9$	4.35458e+16 (y)	?	1	15
$P = 10, C = 5$	6.84695e+16 (z)	?	1	16
$P = 10, C = 6$	1.05579e+17 (aa)	?	1	16
$P = 10, C = 7$	$\geq$ 1.62849e+17 (ab)	?	1 (ac)	16
$P = 10, C = 8$	$\geq$ 2.51152e+17 (ad)	?	1 (ae)	16
$P = 11, C = 4$	3.95742e+17 (af)	?	1	17
$P = 11, C = 5$	6.09707e+17 (ag)	?	1	17
$P = 11, C = 6$	$\geq$ 9.35892e+17 (ah)	?	1 (ai)	17
$P = 12, C = 1$	9.80244e+17 (aj)	?	1	18
$P = 12, C = 2$	1.49671e+18 (ak)	?	1	18
$P = 12, C = 3$	2.29369e+18 (al)	?	1	18
$P = 12, C = 4$	3.52753e+18 (am)	?	1	18
$P = 13, C = 1$	8.73564e+18 (an)	?	1	19
$P = 13, C = 2$	1.33459e+19 (ao)	?	1	19
$P = 14, C = 0$	4.34363e+19 (ap)	?	1	20

(o) safe by construction – stated by the CÆSAR compiler.
(p) stated by CÆSAR.BDD version 3.7 to be true on 1 instance(s) out of 20, false on the remaining 16 instance(s), and unknown on the remaining 3 instance(s).
(q) stated by CÆSAR.BDD version 3.7 to be true on 1 instance(s) out of 20, false on the remaining 16 instance(s), and unknown on the remaining 3 instance(s).
(r) stated by CÆSAR.BDD version 3.7 to be false on 17 instance(s) out of 20, and unknown on the remaining 3 instance(s).
(s) stated by CÆSAR.BDD version 3.7 to be false on 1 instance(s) out of 20, and unknown on the remaining 19 instance(s).
(t) stated by CÆSAR.BDD version 3.7.
(u) stated by CÆSAR.BDD version 3.7.
(v) stated by CÆSAR.BDD version 3.7.
(w) stated by CÆSAR.BDD version 3.7.
(x) stated by CÆSAR.BDD version 3.7.
(y) stated by CÆSAR.BDD version 3.7.
(z) stated by CÆSAR.BDD version 3.7.
(aa) stated by CÆSAR.BDD version 3.7.
(ab) stated by CÆSAR.BDD version 3.7.
(ac) stated by the CÆSAR compiler.
(ad) stated by CÆSAR.BDD version 3.7.
(ae) stated by the CÆSAR compiler.
(af) stated by CÆSAR.BDD version 3.7.
(ag) stated by CÆSAR.BDD version 3.7.
(ah) stated by CÆSAR.BDD version 3.7.
(ai) stated by the CÆSAR compiler.
(aj) stated by CÆSAR.BDD version 3.7.
(ak) stated by CÆSAR.BDD version 3.7.

Hubert Garavel and Quentin Nivon
{hubert.garavel, quentin.nivon}@inria.fr

(al) stated by CÆSAR.BDD version 3.7.
(am) stated by CÆSAR.BDD version 3.7.
(an) stated by CÆSAR.BDD version 3.7.
(ao) stated by CÆSAR.BDD version 3.7.
(ap) stated by CÆSAR.BDD version 3.7.