Hugues Evrard, Alastair Donaldson, and Tyler Sorensen
h.evrard@imperial.ac.uk

This form is a summary description of the model entitled "DLCflexbar" 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

The DLC compiler [2,3,4,5] has been developed to automatically generate a distributed implementation of a concurrent system described using the LNT language. The implementation generated by DLC consists of processes (in the C language) executing in parallel and connected with POSIX sockets. These processes synchronize together and communicate using a distributed protocol for value-passing multiway rendezvous. Besides generating a distributed implementation, the DLC compiler can also produce an LNT model of this implementation by combining the source LNT description of the system with the protocol itself [1]. This implementation model can then be used to check the correctness of the distributed implementation using the CADP toolbox.

This collection of P/T nets was obtained by using DLC to generate implementation models to various instances of the *FlexibleBarrier* model introduced for the 2017 edition of the MCC. The flexible barrier enables application-wide thread synchronisation in the context of GPU *cooperative kernels* [1]. Each generated LNT model was translated automatically to LOTOS, and then to an interpreted Petri net using the CADP toolbox. Finally, a P/T net was 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.

Each instance of the model is parameterized by the maximal number N of concurrent processes that synchronize on the barrier.

Each instance is also parameterized by its version V, which specifies how the NUPN has been produced from the LOTOS specification. V is either equal to "a" if the NUPN has been generated *after* applying all the structural and data-flow optimizations of the CÆSAR compiler for LOTOS, or to "b" if the NUPN has been generated *before* these optimizations.

References

[1] Tyler Sorensen, Hugues Evrard, and Alastair F. Donaldson. *Cooperative Kernels: GPU Multitasking for Blocking Algorithms*. Proceedings of,the 11th Joint Meeting on Foundations of Software Engineering (ESEC/FSE'17), Paderborn, Germany, pages 431–441. Sept. 2017. Available from http://multicore.doc.ic.ac.uk/publications/fse-17.html.

[2] Hugues Evrard and Frédéric Lang. *Automatic Distributed Code Generation from Formal Models of Asynchronous Concurrent Processes*. Proceedings of the 23rd Euromicro International Conference on Parallel, Distributed and Network-based Processing, Special Session on Formal Approaches to Parallel and Distributed Systems (PDP/4PAD'2015), Turku, Finland. IEEE, 2015. Available from http://cadp.inria.fr/publications/Evrard-Lang-15.html.

[3] Hugues Evrard. *DLC: Compiling a Concurrent System Formal Specification to a Distributed Implementation*. Proceedings of the 22nd International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'2016), Eindhoven, Netherlands. Springer, 2016. Available from http://cadp.inria.fr/publications/Evrard-16.html.

[4] Hugues Evrard and Frédéric Lang. *Automatic Distributed Code Generation from Formal Models of Asynchronous Processes Interacting by Multiway Rendezvous*. Journal of Logical and Algebraic Methods in Programming, vol. 88, pages 121–153, Elsevier, 2017. Available from http://cadp.inria.fr/publications/Evrard-Lang-17.html.

[5] http://hevrard.org/DLC

Scaling parameter

Parameter name	Parameter description	Chosen parameter values
(N, V)	N is the number of processes and V is the version defined above	$\{2, 3, 4, 5, 6, 7, 8\} \times \{a, b\}$

Hugues Evrard, Alastair Donaldson, and Tyler Sorensen
h.evrard@imperial.ac.uk

Size of the model

Parameter	Number of places	Number of transitions	Number of arcs	Number of units	HWB code
$N = 2, V = a$	353	2169	8205	197	2–196–249
$N = 2, V = b$	4456	6272	16411	391	173–196–1051
$N = 3, V = a$	581	3891	14956	356	2–355–431
$N = 3, V = b$	7245	10555	28284	709	321–355–1879
$N = 4, V = a$	927	6615	25705	609	2–608–715
$N = 4, V = b$	11440	17128	46731	1215	559–608–3189
$N = 5, V = a$	1415	10593	41484	980	2–979–1125
$N = 5, V = b$	17305	26483	73264	1957	911–979–5101
$N = 6, V = a$	2069	16077	63325	1493	2–1492–1685
$N = 6, V = b$	25104	39112	109395	2983	1401–1492–7735
$N = 7, V = a$	2913	23319	92260	2172	2–2171–2419
$N = 7, V = b$	35101	55507	156636	4341	2053–2171–11211
$N = 8, V = a$	3971	32571	129321	3041	2–3040–3351
$N = 8, V = b$	47560	76160	216499	6079	2891–3040–15649

Structural properties

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

[a] stated by CÆSAR.BDD version 2.7 on all 14 instances (7 values of N × 2 values of V).
[b] stated by CÆSAR.BDD version 2.7 on all 14 instances (7 values of N × 2 values of V).
[c] stated by CÆSAR.BDD version 2.7 on all 14 instances (7 values of N × 2 values of V).
[d] stated by CÆSAR.BDD version 2.7 on all 14 instances (7 values of N × 2 values of V).
[e] stated by CÆSAR.BDD version 2.7 on all 14 instances (7 values of N × 2 values of V).
[f] from place 1 one cannot reach place 0.
[g] place 0 is a source place.
[h] stated by CÆSAR.BDD version 2.7 on all 14 instances (7 values of N × 2 values of V).
[i] stated by CÆSAR.BDD version 2.7 on all 14 instances (7 values of N × 2 values of V).
[j] stated by CÆSAR.BDD version 2.7 on all 14 instances (7 values of N × 2 values of V).
[k] stated by CÆSAR.BDD version 2.7 to be true on 7 instance(s) out of 14, and false on the remaining 7 instance(s).
[l] stated by CÆSAR.BDD version 2.7 on all 14 instances (7 values of N × 2 values of V).
[m] stated by CÆSAR.BDD version 2.7 on all 14 instances (7 values of N × 2 values of V).
[n] the definition of Nested-Unit Petri Nets (NUPN) is available from `http://mcc.lip6.fr/nupn.php`

Hugues Evrard, Alastair Donaldson, and Tyler Sorensen
h.evrard@imperial.ac.uk

Behavioural properties

safe — *in every reachable marking, there is no more than one token on a place* ✔ [o]
deadlock — *there exists a reachable marking from which no transition can be fired* ? [p]
reversible — *from every reachable marking, there is a transition path going back to the initial marking* ?
quasi-live — *for every transition t, there exists a reachable marking in which t can fire* ? [q]
live — *for every transition t, from every reachable marking, one can reach a marking in which t can fire* ?

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
$N = 2, V = a$	7.97923e+20 [r]	?	1	196
$N = 2, V = b$	?	?	1 [s]	$\leq$ 196 [t]
$N = 3, V = a$	2.25393e+30 [u]	?	1	355
$N = 3, V = b$	?	?	1 [v]	$\leq$ 355 [w]
$N = 4, V = a$	1.52867e+43 [x]	?	1	608
$N = 4, V = b$	?	?	1 [y]	$\leq$ 608 [z]
$N = 5, V = a$	?	?	1	979
$N = 5, V = b$	?	?	1 [aa]	$\leq$ 979 [ab]
$N = 6, V = a$	?	?	1	1492
$N = 6, V = b$	?	?	1 [ac]	$\leq$ 1492 [ad]
$N = 7, V = a$	?	?	1	2171
$N = 7, V = b$	?	?	1 [ae]	$\leq$ 2171 [af]
$N = 8, V = a$	?	?	1 [ag]	3040
$N = 8, V = b$	?	?	1 [ah]	$\leq$ 3040 [ai]

(o) safe by construction – stated by the CÆSAR compiler.
(p) stated by CÆSAR.BDD version 2.7 to be false on 6 instance(s) out of 14, and unknown on the remaining 8 instance(s).
(q) stated by CÆSAR.BDD version 2.7 to be true on 6 instance(s) out of 14, and unknown on the remaining 8 instance(s).
(r) stated by CÆSAR.BDD version 2.7.
(s) stated by the CÆSAR compiler.
(t) upper bound given by the number of leaf units.
(u) stated by CÆSAR.BDD version 2.7.
(v) stated by the CÆSAR compiler.
(w) upper bound given by the number of leaf units.
(x) stated by CÆSAR.BDD version 2.7.
(y) stated by the CÆSAR compiler.
(z) upper bound given by the number of leaf units.
(aa) stated by the CÆSAR compiler.
(ab) upper bound given by the number of leaf units.
(ac) stated by the CÆSAR compiler.
(ad) upper bound given by the number of leaf units.
(ae) stated by the CÆSAR compiler.
(af) upper bound given by the number of leaf units.
(ag) stated by the CÆSAR compiler.
(ah) stated by the CÆSAR compiler.
(ai) upper bound given by the number of leaf units.