Model: Echo Algorithm
Type: P/T Net
Origin: Academic

since
MCC 2012

Niels Lohmann
niels.lohmann@uni-rostock.de

*This form is a summary description of the model entitled "Echo Algorithm" 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 file specifies the Echo Algorithm (see [Reisig98]) for grid like networks. Echo is a protocol for propagation of information with feedback in a network. The algorithm operates in an incomplete, but connected bidirectional network of agents. A distinguished agent (initiator), starts the distribution of a message by sending it to all its neighbors. On receiving some first message, every other agent forwards the message to all its neighbors, except the one it received its first message from. Then it awaits messages from all recipients of its forwards (regardless whether these messages had been intended as forwards or acknowledgments) and replies to the agent where it received its first message from. As soon as the initiator receives a message from all its neighbors, the protocol terminates.

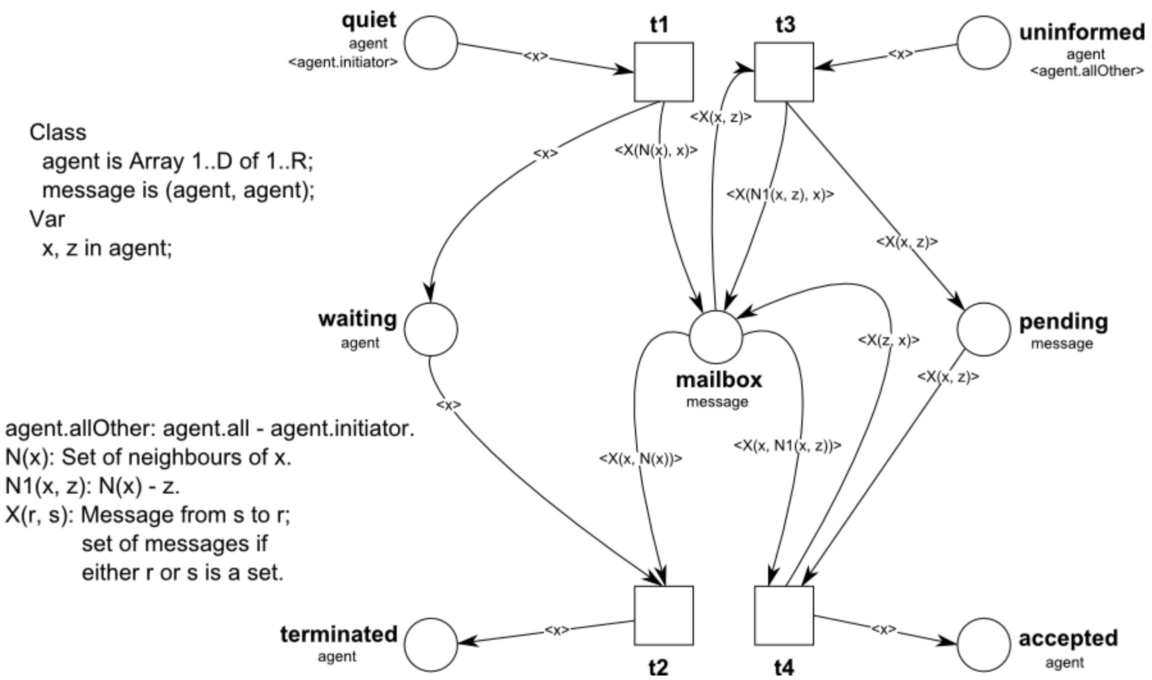
In this example, agents are arranged in a hypercube. The network can be scaled in two values: the number of *dimensions* and the number of *agents per dimensions*. For instance, a network with two dimensions and three agents per row would consist of $3^2 = 9$ agents while a network with three dimensions and 4 agents per row would consist of $4^3 = 64$ agents.

Regardless of the chosen values for these dimensions, we always connect agents that are immediate neighbors in one of the dimensions of the hypercube. We place the initiator into the center of the cube which means that the number of agents per row should be an odd number.

Unfolded versions of the Echo Algorithm are also provided for a variety of scaling parameters. These nets are given in LoLA low-level format and PNML.

The model is sketched in the figure. The two sorts $D$ and $R$ model the scaling factors dimensions and agents per row, respectively. Messages are modeled as pairs (receiver, sender).

*In April 2021, Pierre Bouvier provided a decomposition of all the instances of this model into networks of communicating automata. Each network is expressed as a Nested-Unit Petri Net (NUPN) that can be found, for each instance, in the "toolspecific" section of the corresponding PNML file.*




 *generated on January 2, 2024*

Niels Lohmann
niels.lohmann@uni-rostock.de

## References

Wolfgang Reisig. *Elements of Distributed Algorithms. Modeling and Analysis with Petri Nets.*, Springer, 1998.

The original algorithm has been modeled as an algebraic Petri net in LoLA high-level format, see http://service-technology.org/files/lola/lola.pdf

## Scaling parameter

| Parameter name | Parameter description | Chosen parameter values |
|---|---|---|
| (dimensions $d$, agents per row $a$) | see description | (2,9), (2,11), (2,15), (2,19), (3,3), (3,5), (3,7), (4,3), (5,3) |

## Size of the model

| Parameter | Number of places | Number of transitions | Number of arcs | Number of units | HWB code |
|---|---|---|---|---|---|
| $(d = 2, a = 9)$ | 735 | 570 | 3220 | 334 | 1–333–523 |
| $(d = 2, a = 11)$ | 1119 | 874 | 4996 | 518 | 1–517–794 |
| $(d = 2, a = 15)$ | 2127 | 1674 | 9700 | 1006 | 1–1005–1506 |
| $(d = 2, a = 19)$ | 3455 | 2730 | 15940 | 1654 | 1–1653–2443 |
| $(d = 3, a = 3)$ | 265 | 206 | 1252 | 136 | 1–135–188 |
| $(d = 3, a = 5)$ | 1445 | 1190 | 8260 | 726 | 1–725–974 |
| $(d = 3, a = 7)$ | 4209 | 3518 | 25540 | 2108 | 1–2107–2792 |
| $(d = 4, a = 3)$ | 1019 | 850 | 6340 | 514 | 1–513–682 |
| $(d = 5, a = 3)$ | 3717 | 3222 | 28404 | 1864 | 1–1863–2478 |

## 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)]

[(a)] stated by CÆSAR.BDD version 1.7 on all 9 instances ((2,9), (2,11), (2,15), (2,19), (3,3), (3,5), (3,7), (4,3), and (5,3)).
[(b)] stated by CÆSAR.BDD version 2.6 on all 9 instances ((2,9), (2,11), (2,15), (2,19), (3,3), (3,5), (3,7), (4,3), and (5,3)).
[(c)] stated by CÆSAR.BDD version 1.7 on all 9 instances ((2,9), (2,11), (2,15), (2,19), (3,3), (3,5), (3,7), (4,3), and (5,3)).
[(d)] stated by CÆSAR.BDD version 1.7 on all 9 instances ((2,9), (2,11), (2,15), (2,19), (3,3), (3,5), (3,7), (4,3), and (5,3)).
[(e)] stated by CÆSAR.BDD version 1.7 on all 9 instances ((2,9), (2,11), (2,15), (2,19), (3,3), (3,5), (3,7), (4,3), and (5,3)).
[(f)] stated by CÆSAR.BDD version 1.7 on all 9 instances ((2,9), (2,11), (2,15), (2,19), (3,3), (3,5), (3,7), (4,3), and (5,3)).
[(g)] stated by CÆSAR.BDD version 1.7 on all 9 instances ((2,9), (2,11), (2,15), (2,19), (3,3), (3,5), (3,7), (4,3), and (5,3)).
[(h)] stated by CÆSAR.BDD version 1.7 on all 9 instances ((2,9), (2,11), (2,15), (2,19), (3,3), (3,5), (3,7), (4,3), and (5,3)).
[(i)] stated by CÆSAR.BDD version 1.7 on all 9 instances ((2,9), (2,11), (2,15), (2,19), (3,3), (3,5), (3,7), (4,3), and (5,3)).
[(j)] stated by CÆSAR.BDD version 1.7 on all 9 instances ((2,9), (2,11), (2,15), (2,19), (3,3), (3,5), (3,7), (4,3), and (5,3)).
[(k)] stated by CÆSAR.BDD version 1.7 on all 9 instances ((2,9), (2,11), (2,15), (2,19), (3,3), (3,5), (3,7), (4,3), and (5,3)).
[(l)] stated by CÆSAR.BDD version 1.7 on all 9 instances ((2,9), (2,11), (2,15), (2,19), (3,3), (3,5), (3,7), (4,3), and (5,3)).
[(m)] stated by CÆSAR.BDD version 1.7 on all 9 instances ((2,9), (2,11), (2,15), (2,19), (3,3), (3,5), (3,7), (4,3), and (5,3)).

Niels Lohmann
niels.lohmann@uni-rostock.de

**nested units** — *places are structured into hierarchically nested sequential units* [(n)] . . . . . . . . . . ✔

## Behavioural properties

**safe** — *in every reachable marking, there is no more than one token on a place* . . . . . . . . . . ✔
**dead place(s)** — *one or more places have no token in any reachable marking* . . . . . . . . . . ?
**dead transition(s)** — *one or more transitions cannot fire from any reachable marking* . . . . . . . . . . ?
**deadlock** — *there exists a reachable marking from which no transition can be fired* . . . . . . . . . . ✔ [(o)]
**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* . . . . . . . . . . ?

## 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 |
|---|---|---|---|---|
| $(d = 2, a = 9)$ | ? | ? | 1 [(p)] | $\in$ [209, 333] [(q)] |
| $(d = 2, a = 11)$ | $\geq$ 5.2673e+09 [(r)] | ? | 1 [(s)] | $\in$ [321, 517] [(t)] |
| $(d = 2, a = 15)$ | $\geq$ 4 [(u)] | ? | 1 [(v)] | $\in$ [617, 1005] [(w)] |
| $(d = 2, a = 19)$ | ? | ? | 1 [(x)] | $\in$ [1009, 1653] [(y)] |
| $(d = 3, a = 3)$ | $\geq$ 26465 [(z)] | ? | 1 [(aa)] | $\in$ [83, 135] [(ab)] |
| $(d = 3, a = 5)$ | $\geq$ 2 [(ac)] | ? | 1 [(ad)] | $\in$ [477, 725] [(ae)] |
| $(d = 3, a = 7)$ | ? | ? | 1 [(af)] | $\in$ [1423, 2107] [(ag)] |
| $(d = 4, a = 3)$ | $\geq$ 4 [(ah)] | ? | 1 [(ai)] | $\in$ [353, 513] [(aj)] |
| $(d = 5, a = 3)$ | ? | ? | 1 [(ak)] | $\in$ [1379, 1863] [(al)] |

## Other properties

The intuitive description of the Echo Algorithm can be modeled as a CTL formula

$$(\mathbf{AF}\ \text{“initiator terminated”}) \land (\mathbf{A}\ \neg\text{“initiator terminated”}\ \mathbf{U}\ \text{“all other sites accepted”})$$

This formula is given for the unfolded low-level models.

(n) the definition of Nested-Unit Petri Nets (NUPN) is available from `http://mcc.lip6.fr/nupn.php`
(o) confirmed at MCC'2014 by GreatSPN, Lola, and Tapaal on 9 instances.
(p) This net is safe.
(q) upper bound given by the number of leaf units.
(r) stated by CÆSAR.BDD version 3.5.
(s) This net is safe.
(t) upper bound given by the number of leaf units.
(u) stated by CÆSAR.BDD version 3.5.
(v) This net is safe.
(w) upper bound given by the number of leaf units.
(x) This net is safe.
(y) upper bound given by the number of leaf units.
(z) stated by CÆSAR.BDD version 2.0.
(aa) This net is safe.
(ab) upper bound given by the number of leaf units.
(ac) stated by CÆSAR.BDD version 3.3.
(ad) This net is safe.
(ae) upper bound given by the number of leaf units.
(af) This net is safe.
(ag) upper bound given by the number of leaf units.
(ah) stated by CÆSAR.BDD version 2.0.
(ai) This net is safe.
(aj) upper bound given by the number of leaf units.
(ak) This net is safe.
(al) upper bound given by the number of leaf units.