Message-Passing algorithms for the Verification of Distributed Protocols

Loïg Jezequel and Javier Esparza

Technische Universität München

VMCAI 2014
January 19-21

About message-passing algorithms (MPA)

MPA encompass a broad set of algorithms [Fabre 07]

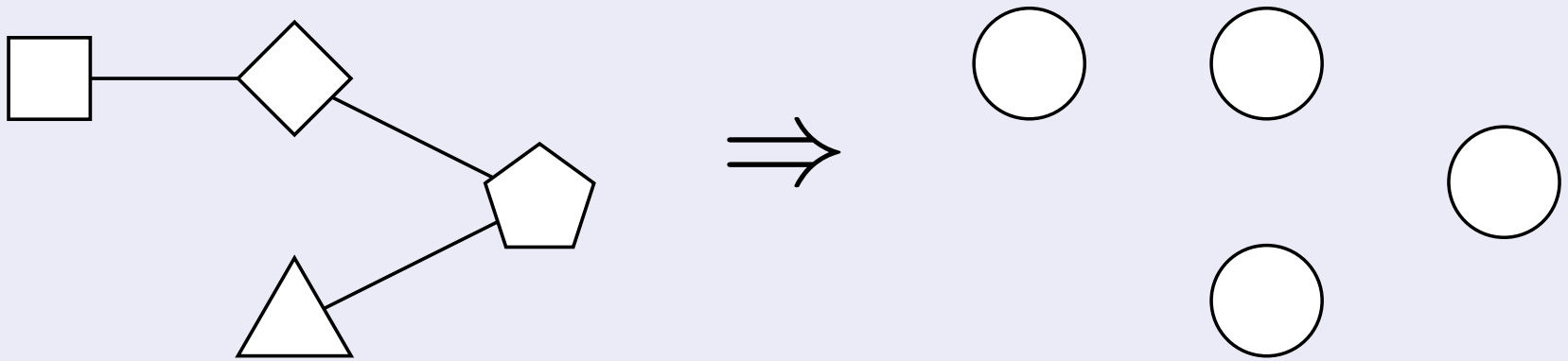
Viterbi algorithm, Kalman filtering, Belief propagation, ...

Informal description of MPA

Agents repeatedly exchange messages through the edges of a graph to compute local views of some global system

In other words: MPA solve the reduction problem on trees

Each node's view of the tree/behaviour of each node embedded in the tree



MPA for the verification of distributed systems

Distributed system

- A set of components (e.g agents, computers, programs)
- Communication/interaction (e.g shared memory, channels)

Verification

- Global properties (e.g *mutual exclusion*)
- Local properties (e.g *a given agent is live*)

Verification and the reduction problem

Local properties can be verified on a solution to the reduction problem corresponding to a distributed system

Some formalism

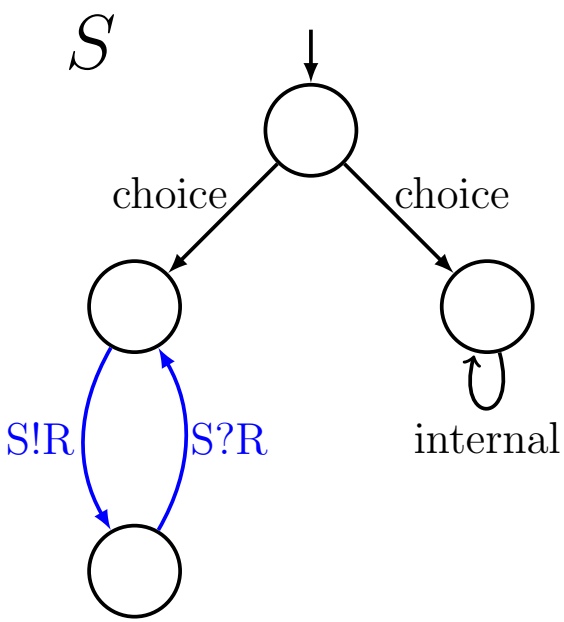
Formalism I: Distributed systems

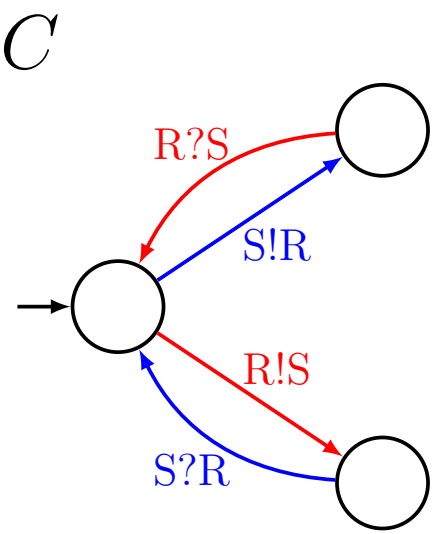
Components as LTSs

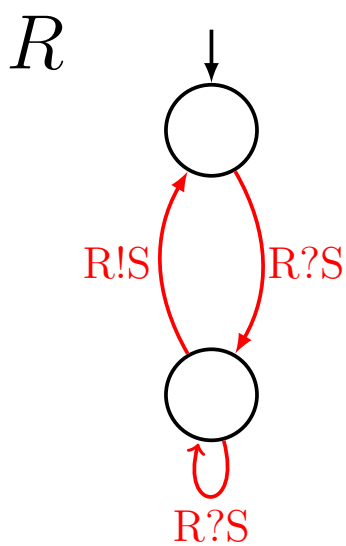
$\mathcal{L} = (\Sigma, S, T, s^0)$

Communications

By *rendez-vous* on shared labels

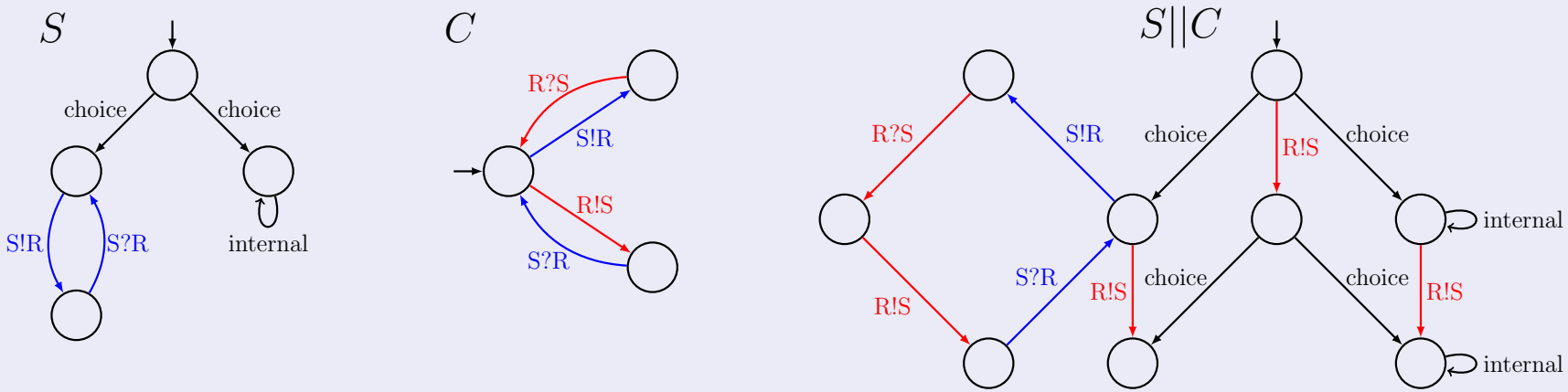






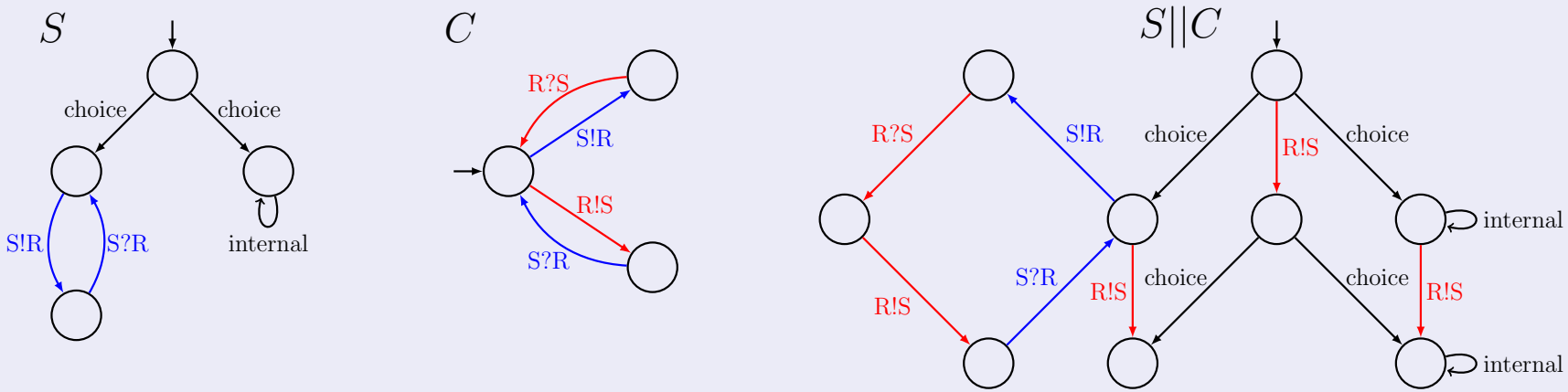
Formalism II: Behaviour of an embedded component

Parallel composition: Behaviour of the full system

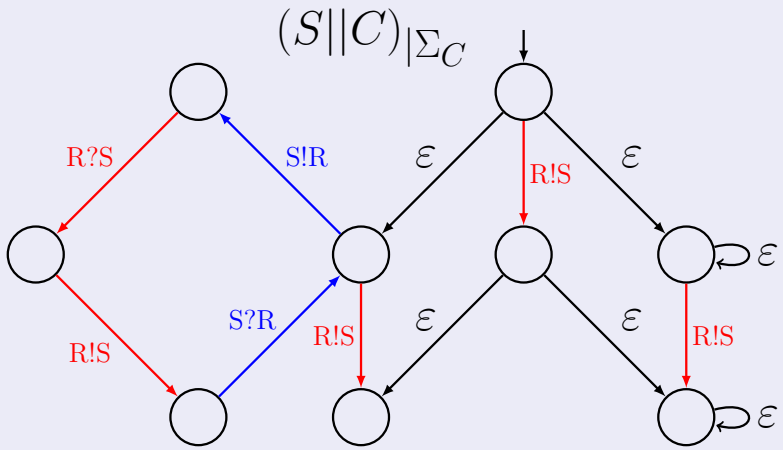


Formalism II: Behaviour of an embedded component

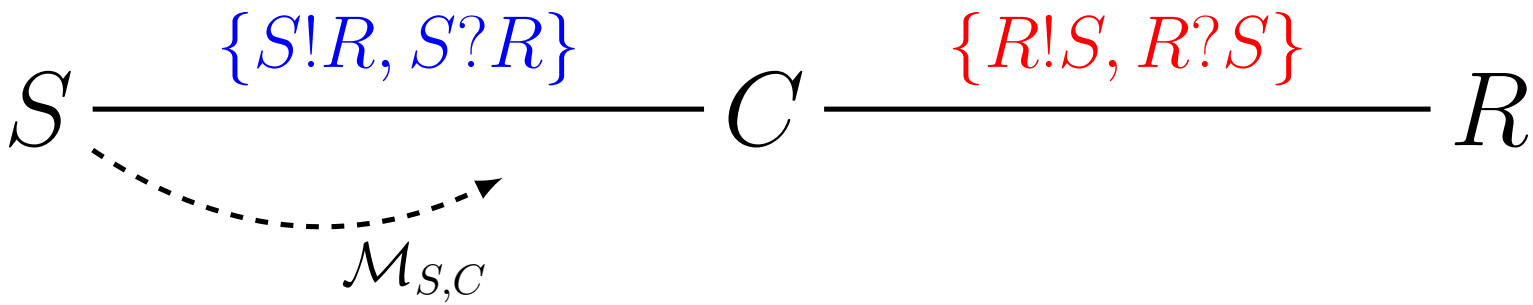
Parallel composition: Behaviour of the full system



Hiding: Behaviour of a single component



Formalism III: MPA with no optimization

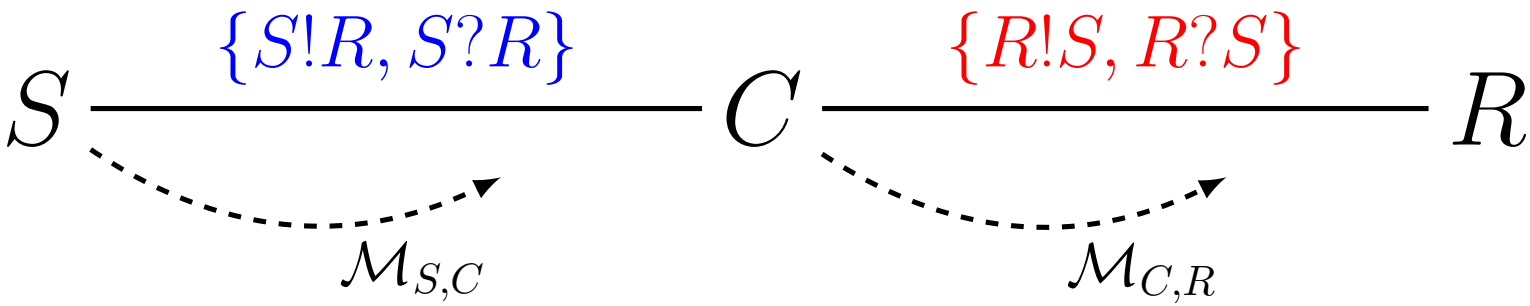


Messages

$\mathcal{M}_{S,C} = S_{|\Sigma_C}$ (= S as C views it)

Result

Formalism III: MPA with no optimization

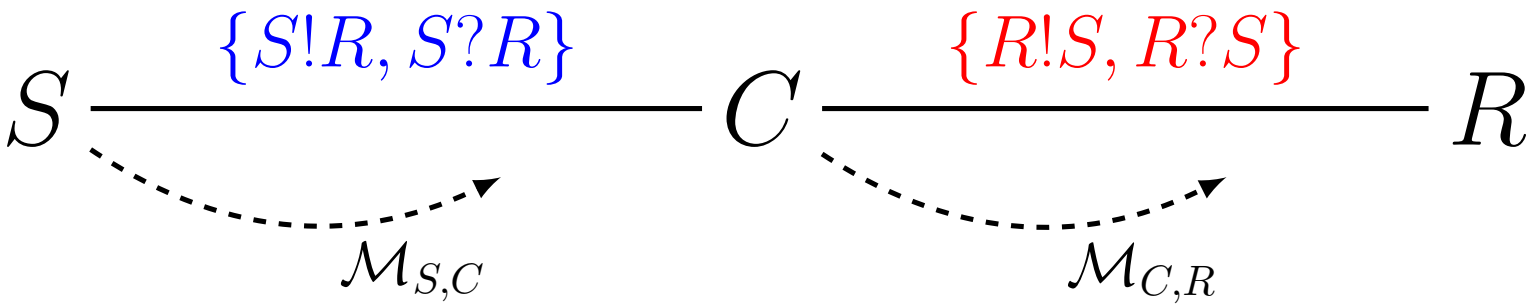


Messages

$\mathcal{M}_{S,C} = S_{|\Sigma_C}$, $\mathcal{M}_{C,R} = (\mathcal{M}_{S,C}||C)_{|\Sigma_R}$ $(= S||C$ as R views it$)$

Result

Formalism III: MPA with no optimization



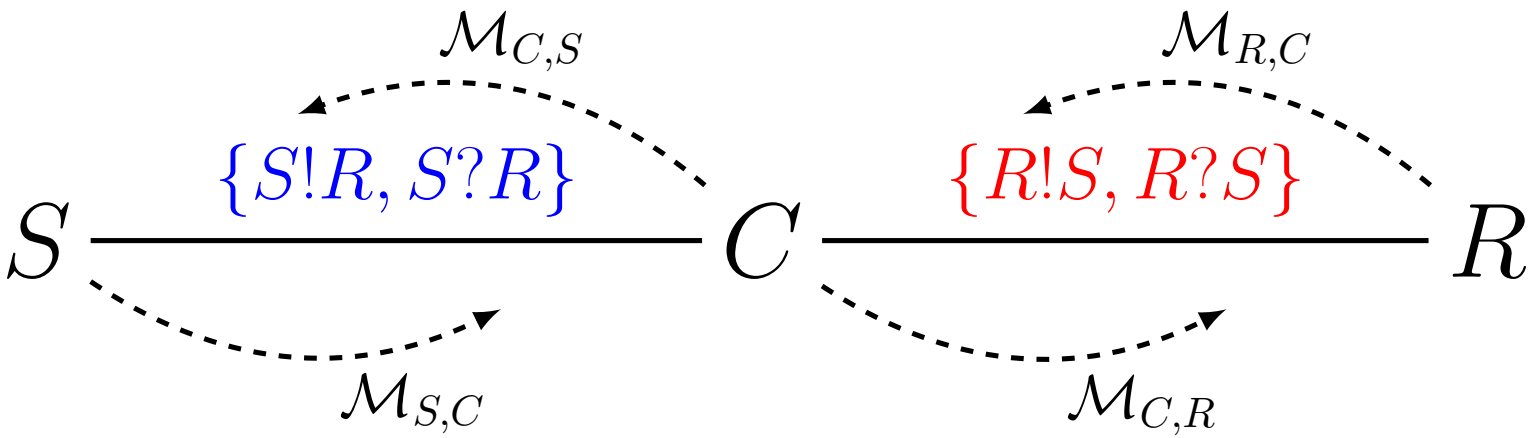
Messages

$\mathcal{M}_{S,C} = S_{|\Sigma_C}$, $\mathcal{M}_{C,R} = (\mathcal{M}_{S,C}||C)_{|\Sigma_R}$

Result

$R' = \mathcal{M}_{C,R}||R = (S||C||R)_{|\Sigma_R}$

Formalism III: remark



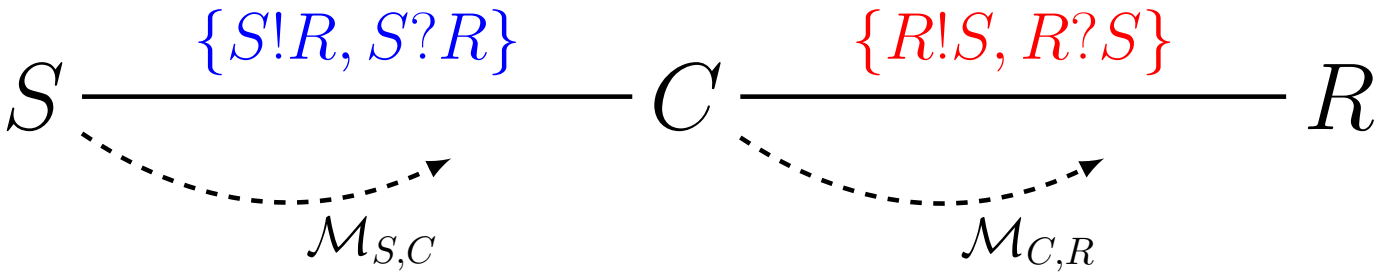
Messages

$\mathcal{M}_{S,C} = S_{|\Sigma_C}$, $\mathcal{M}_{C,R} = (\mathcal{M}_{S,C}||C)_{|\Sigma_R}$
$\mathcal{M}_{R,C} = R_{|\Sigma_C}$, $\mathcal{M}_{C,S} = (\mathcal{M}_{R,C}||C)_{|\Sigma_S}$

Result

$R' = \mathcal{M}_{C,R}||R = (S||C||R)_{|\Sigma_R}$
$S' = \mathcal{M}_{C,S}||S = (S||C||R)_{|\Sigma_S}$, $C' = \mathcal{M}_{S,C}||\mathcal{M}_{R,C}||C = (S||C||R)_{|\Sigma_C}$

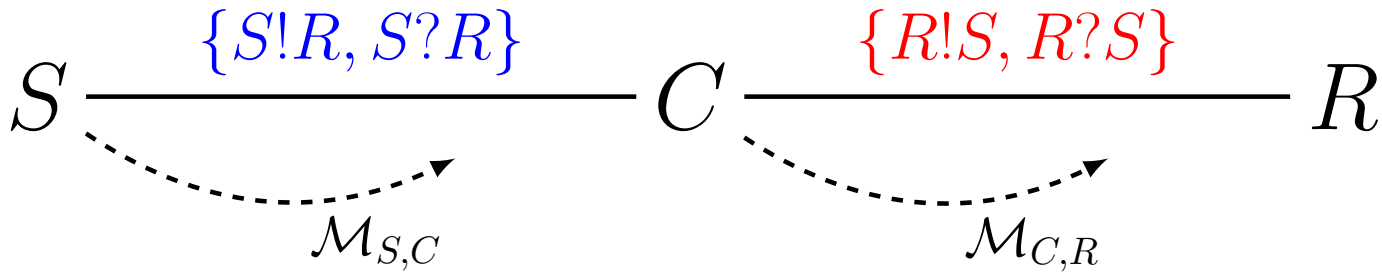
Formalism IV: Interest of MPA, step by step optimization



Intuition

- Equality is a too strong requirement in $R' = (S||C||R)_{|\Sigma_R}$
- Replace it by a weaker notion $\equiv$ of "same behaviour"

Formalism IV: Interest of MPA, step by step optimization



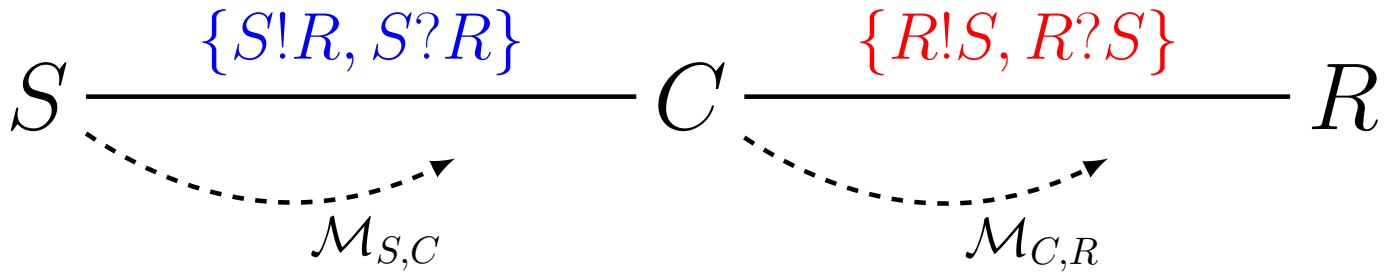
Intuition

- Equality is a too strong requirement in $R' = (S||C||R)_{|\Sigma_R}$
- Replace it by a weaker notion $\equiv$ of "same behaviour"

Theorem: well choosen behaviour

Let $\equiv$ be a congruence for LTSs,
taking $\mathcal{M}_{S,C} \equiv S_{|\Sigma_C}$, $\mathcal{M}_{C,R} \equiv (\mathcal{M}_{S,C}||C)_{|\Sigma_R}$, and $R' \equiv \mathcal{M}_{C,R}||R$
gives $R' \equiv (S||C||R)_{|\Sigma_R}$.

Formalism IV: Interest of MPA, step by step optimization



Intuition

- Equality is a too strong requirement in $R' = (S||C||R)_{|\Sigma_R}$
- Replace it by a weaker notion $\equiv$ of "same behaviour"

Theorem: well choosen behaviour

Let $\equiv$ be a congruence for LTSs,
taking $\mathcal{M}_{S,C} \equiv S_{|\Sigma_C}$, $\mathcal{M}_{C,R} \equiv (\mathcal{M}_{S,C}||C)_{|\Sigma_R}$, and $R' \equiv \mathcal{M}_{C,R}||R$
gives $R' \equiv (S||C||R)_{|\Sigma_R}$.

Interest

Allows for silent-transitions removal and size-reduction of LTSs

Local safety properties

Local safety properties

Definiton: Local property of L_i in $L_1||\ldots||L_n$

A property of the finite traces of $(L_1||\ldots||L_n)_{|\Sigma_i}$

Congruence: Trace equivalence

$L \equiv L'$ if and only if L and L' have the same sets of finite traces

Implementation of $L' :\equiv L$

$L' := L$

Local safety properties

Definiton: Local property of L_i in $L_1||\ldots||L_n$

A property of the finite traces of $(L_1||\ldots||L_n)_{|\Sigma_i}$

Congruence: Trace equivalence

$L \equiv L'$ if and only if L and L' have the same sets of finite traces

Implementation of $L' :\equiv L$

$L' := RED(L)$

Local safety properties

Definiton: Local property of L_i in $L_1||\ldots||L_n$

A property of the finite traces of $(L_1||\ldots||L_n)_{|\Sigma_i}$

Congruence: Trace equivalence

$L \equiv L'$ if and only if L and L' have the same sets of finite traces

Implementation of $L' :\equiv L$

$L' := DET(RED(L))$

Local safety properties

Definiton: Local property of L_i in $L_1||\ldots||L_n$

A property of the finite traces of $(L_1||\ldots||L_n)_{|\Sigma_i}$

Congruence: Trace equivalence

$L \equiv L'$ if and only if L and L' have the same sets of finite traces

Implementation of $L' :\equiv L$

$L' := MIN(DET(RED(L)))$

Local liveness properties

Local liveness properties I

Observable traces are not enough

- Consider an LTS $L_1 || \ldots || L_n$ with set of traces $\{ab^\omega, ac^\omega\}$
- Assume $\Sigma_i = \{a, b\}$
- The only observable infinite trace in $(L_1 || \ldots || L_n)_{|\Sigma_i}$ is ab^ω
- "After a enventually b" seems to hold

Local liveness properties I

Observable traces are not enough

- Consider an LTS $L_1||\ldots||L_n$ with set of traces $\{ab^\omega, ac^\omega\}$
- Assume $\Sigma_i = \{a, b\}$
- The only observable infinite trace in $(L_1||\ldots||L_n)_{|\Sigma_i}$ is ab^ω
- "After a enventually b" seems to hold

Notion of divergence

In an LTS $L_1||\ldots||L_n$, a divergence of L_i is a finite observable trace t_i of L_i such that there exists an infinite observable trace t of $L_1||\ldots||L_n$ satisfying $t_{|\Sigma_i} = t_i$

Local liveness properties II

Definiton: Local property of L_i in $L_1||\ldots||L_n$

A property of the finite traces of $(L_1||\ldots||L_n)_{|\Sigma_i}$ and of the divergences of L_i in $L_1||\ldots||L_n$

Congruence: Trace equivalence

$L \equiv_d L'$ if and only if L and L' have the same sets of finite traces and of divergences

Implementation of $L' :\equiv_d L$

$L' := L$

Local liveness properties II

Definiton: Local property of L_i in $L_1||\ldots||L_n$

A property of the finite traces of $(L_1||\ldots||L_n)_{|\Sigma_i}$ and of the divergences of L_i in $L_1||\ldots||L_n$

Congruence: Trace equivalence

$L \equiv_d L'$ if and only if L and L' have the same sets of finite traces and of divergences

Implementation of $L' :\equiv_d L$

$L' := DIV(L)$

Local liveness properties II

Definiton: Local property of L_i in $L_1||\ldots||L_n$

A property of the finite traces of $(L_1||\ldots||L_n)_{|\Sigma_i}$ and of the divergences of L_i in $L_1||\ldots||L_n$

Congruence: Trace equivalence

$L \equiv_d L'$ if and only if L and L' have the same sets of finite traces and of divergences

Implementation of $L' :\equiv_d L$

$L' := HID(DIV(L))$

Local liveness properties II

Definiton: Local property of L_i in $L_1||\ldots||L_n$

A property of the finite traces of $(L_1||\ldots||L_n)_{|\Sigma_i}$ and of the divergences of L_i in $L_1||\ldots||L_n$

Congruence: Trace equivalence

$L \equiv_d L'$ if and only if L and L' have the same sets of finite traces and of divergences

Implementation of $L' :\equiv_d L$

$L' := HID(MIN(DET(RED(DIV(L)))))$

Experimental evaluation

Implementation and experimental setting

Implementation

Extension of the planner DISTOPLAN [Fabre, J., Haslum, Thiébaux 10]

- Current version written in SCALA
- JAVA library for hiding: dk.brics.automaton

Experimental setting

- Intel Core i5 processor
- 4GB of memory
- No time limit

Comparison with SPIN

- Partial order reduction
- "Best possible" memory management

Raymond's mutual exclusion algorithm I

Overview of the algorithm

Token-based mutual exclusion in a tree of processes

Our setting

- Complete binary tree
- Verification of local properties of the root

A safety property

It is not possible to request the token twice without receiving it in between

A liveness property

The token is received in finite time after any request

Raymond's mutual exclusion algorithm II

Results obtained by DISTOPLAN (times in seconds)

Depth	Safety (traces)			Liveness (divergences)		
	MPA	OneWay	Verif.	MPA	OneWay	Verif.
2 (3)	0.12	0.15	<0.01	0.14	0.13	<0.01
3 (7)	1.41	1.23	<0.01	2.07	1.88	<0.01
4 (15)	2.36	2.20	<0.01	4.58	4.36	<0.01
5 (31)	5.29	4.67	<0.01	10.44	9.67	<0.01
6 (63)	10.62	9.63	<0.01	21.81	20.27	<0.01
7 (127)	21.94	19.70	<0.01	44.86	41.55	<0.01

Results obtained by SPIN

Depth 2: Better than DISTOPLAN ($< 0.01s$)

Greater depths: Out of memory (4GB)

Pragmatic general multicast protocol I

Overview of the algorithm

Protocol for distributing information between multiple agents, designed to minimize acknowledgements and retransmissions of messages

Our setting

- One sender and one/multiple receivers
- Verification of local properties of the sender
- Channels with bounded capacity, possibly losing messages
- Fixed number of data to send

A safety property

The last data can only be sent once

A liveness property

The first data is always sent at least once (in finite time)

Pragmatic general multicast protocol II

One receiver, two different data, channels of capacity one

Results obtained by DISTOPLAN (times in seconds)

Processes	Traces				Divergences		
	Basic	MPA	OneWay	Verif.	MPA	OneWay	Verif.
5	7.79	0.08	0.03	<0.01	0.11	0.08	<0.01
10	20.27	0.13	0.08	<0.01	0.16	0.13	<0.01
15	32.76	0.19	0.15	<0.01	0.22	0.20	<0.01
20	41.99	0.23	0.16	<0.01	0.26	0.20	<0.01
25	53.14	0.26	0.21	<0.01	0.31	0.24	<0.01
30	67.50	0.30	0.25	<0.01	0.37	0.27	<0.01
35	77.32	0.35	0.29	<0.01	0.43	0.34	<0.01
40	89.95	0.40	0.32	<0.01	0.49	0.36	<0.01
45	101.25	0.46	0.36	<0.01	0.57	0.40	<0.01
50	113.60	0.50	0.40	<0.01	0.60	0.44	<0.01

Pragmatic general multicast protocol III

One receiver, two/three different data, channels of capacity one/two

Results obtained by DISTOPLAN (times in seconds)

Processes	Traces		Divergences	
	d=2, c=2	d=3, c=1	d=2, c=2	d=3, c=1
5	10.71	10.63	15.37	13.26
10	19.19	12.60	28.94	18.00
15	27.24	14.56	41.77	22.21
20	35.53	16.46	55.77	26.80
25	43.66	18.24	68.40	30.95
30	52.14	20.66	81.43	35.36
35	60.16	22.64	95.39	39.82
40	68.78	24.80	109.17	44.49
45	77.00	26.66	122.57	48.56
50	85.12	29.01	136.60	53.27

Pragmatic general multicast protocol IV

Multiple receiver (leaves of a complete binary tree)

Results obtained by DISTOPLAN (times in seconds)

Depth	Traces			Divergences		
	d=2, c=1	**d=2, c=2**	**d=3, c=1**	**d=2, c=1**	**d=2, c=2**	**d=3, c=1**
3 (7)	0.85	26.26	59.30	1.41	33.96	93.02
4 (15)	1.58	56.10	114.05	1.60	72.89	156.93
5 (31)	2.48	113.82	235.32	2.93	153.47	316.63
6 (63)	5.06	231.27	472.19	5.73	310.28	641.13
7 (127)	10.24	474.57	979.85	12.10	625.61	1582.23

Conclusion

Conclusion

Summary

- From MPA for planning to MPA for verification
- Other equivalences than trace equivalence can be used
- Experimental analysis on two protocols

Remark on true/false properties

- Only true properties in our benchmarks
- SPIN is still far better on false properties

Futur work

- Equivalence relation for deadlocks
- MPA and global properties