State Space Reduction of Linear Processes using Control Flow Reconstruction

Jaco van de Pol, Mark Timmer
University of Twente

FMT Lunchmeeting
April 14, 2009

Contents

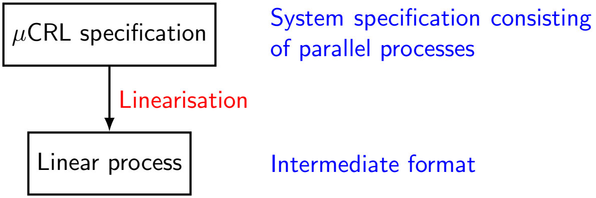
Contents

The μCRL toolset

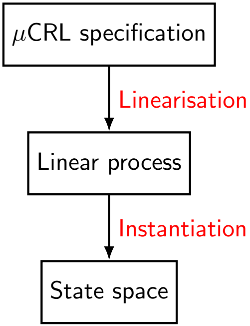
$\boxed{\mu\text{CRL specification}}$

System specification consisting of parallel processes

The μCRL toolset



The μCRL toolset

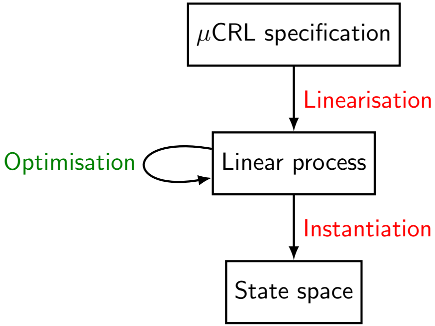


System specification consisting of parallel processes

Intermediate format

Very big graph

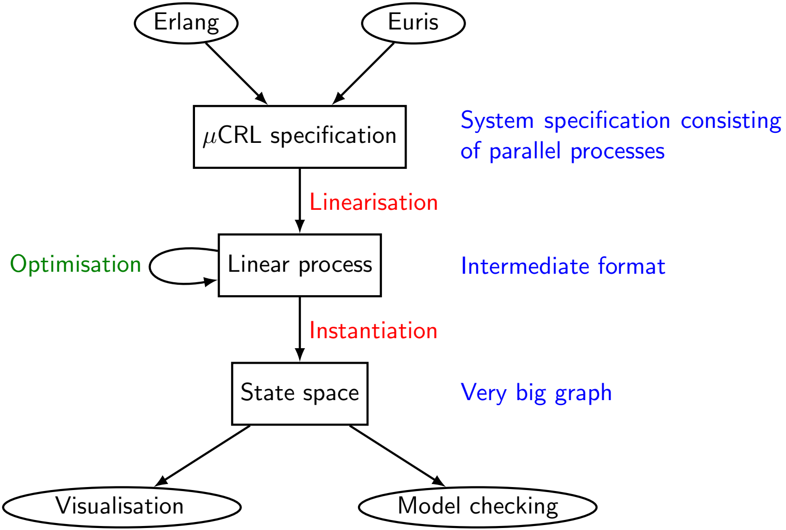
The μCRL toolset



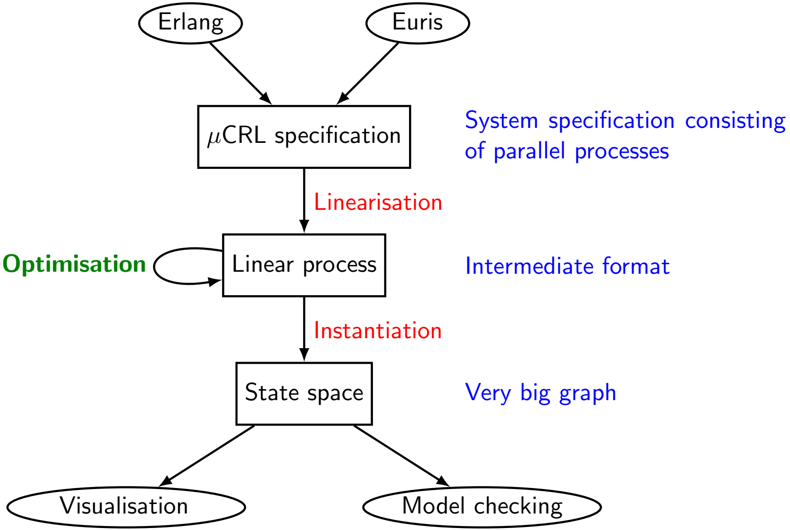
System specification consisting of parallel processes

Intermediate format

Very big graph


Erlang
Euris
μCRL specification
System specification consisting of parallel processes
Linearisation
Optimisation
Linear process
Intermediate format
Instantiation
State space
Very big graph
Visualisation
Model checking

The μCRL toolset



The linear process equation

The basic structure of an LPE

$$
\begin{aligned}
X(d\colon D) = & \sum_{e_1 : E_1} c_1(d, e_1) \Rightarrow a_1(d, e_1) \cdot X(g_1(d, e_1)) \\
+ & \quad \ldots \\
+ & \sum_{e_n : E_n} c_n(d, e_n) \Rightarrow a_n(d, e_n) \cdot X(g_n(d, e_n))
\end{aligned}
$$

The linear process equation

The basic structure of an LPE

$$\begin{aligned} X(\boldsymbol{d} \colon \boldsymbol{D}) = & \sum_{\boldsymbol{e_1} \,:\, \boldsymbol{E_1}} c_1(\boldsymbol{d}, \boldsymbol{e_1}) \Rightarrow a_1(\boldsymbol{d}, \boldsymbol{e_1}) \cdot X(g_1(\boldsymbol{d}, \boldsymbol{e_1})) \\ + & \quad \ldots \\ + & \sum_{\boldsymbol{e_n} \,:\, \boldsymbol{E_n}} c_n(\boldsymbol{d}, \boldsymbol{e_n}) \Rightarrow a_n(\boldsymbol{d}, \boldsymbol{e_n}) \cdot X(g_n(\boldsymbol{d}, \boldsymbol{e_n})) \end{aligned}$$

- d: a vector of state variables
- e_i: a vector of local variables for summand i
- c_i: the enabling condition for summand i
- a_i: the (parameterised) action for summand i (possibly τ)
- g_i: the next-state function for summand i

The linear process equation

The basic structure of an LPE

$$\begin{aligned} X(\boldsymbol{d} \colon \boldsymbol{D}) = & \sum_{\boldsymbol{e_1} : \boldsymbol{E_1}} c_1(\boldsymbol{d}, \boldsymbol{e_1}) \Rightarrow a_1(\boldsymbol{d}, \boldsymbol{e_1}) \cdot X(g_1(\boldsymbol{d}, \boldsymbol{e_1})) \\ + & \quad \ldots \\ + & \sum_{\boldsymbol{e_n} : \boldsymbol{E_n}} c_n(\boldsymbol{d}, \boldsymbol{e_n}) \Rightarrow a_n(\boldsymbol{d}, \boldsymbol{e_n}) \cdot X(g_n(\boldsymbol{d}, \boldsymbol{e_n})) \end{aligned}$$

- d: a vector of state variables
- e_i: a vector of local variables for summand i
- c_i: the enabling condition for summand i
- a_i: the (parameterised) action for summand i (possibly τ)
- g_i: the next-state function for summand i

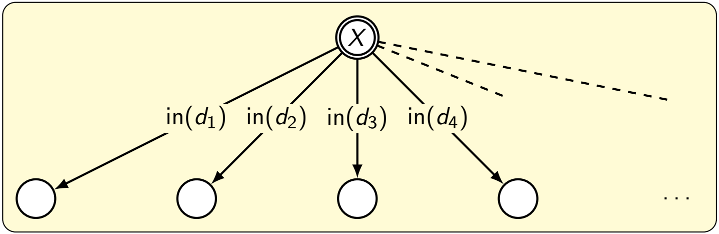
$$\boldsymbol{d} \xrightarrow{a(\boldsymbol{p})} \boldsymbol{d'} \Leftrightarrow \exists i . \exists \boldsymbol{e_i} . c_i(\boldsymbol{d}, \boldsymbol{e_i}) = \mathtt{true} \wedge a_i(\boldsymbol{d}, \boldsymbol{e_i}) = a(\boldsymbol{p}) \wedge g_i(\boldsymbol{d}, \boldsymbol{e_i}) = \boldsymbol{d'}$$

An example

$$X = \sum_{d\,:\,D} \mathrm{in}(d) \cdot (\tau \cdot \mathrm{loss} \cdot X + \tau \cdot \mathrm{out}(d) \cdot X)$$

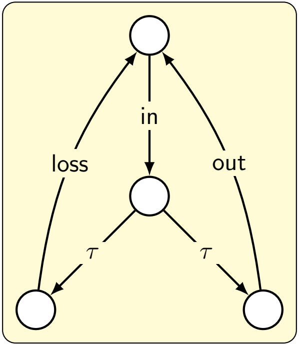
An example

$$X = \sum_{d\colon D} \mathrm{in}(d) \cdot (\tau \cdot \mathrm{loss} \cdot X + \tau \cdot \mathrm{out}(d) \cdot X)$$



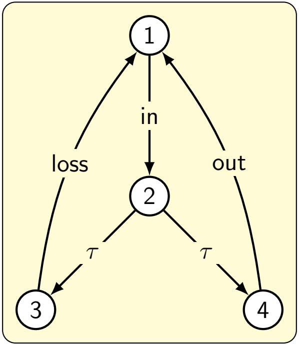
An example

$$X = \sum_{d:\,D} \mathsf{in}(d) \cdot (\tau \cdot \mathsf{loss} \cdot X + \tau \cdot \mathsf{out}(d) \cdot X)$$



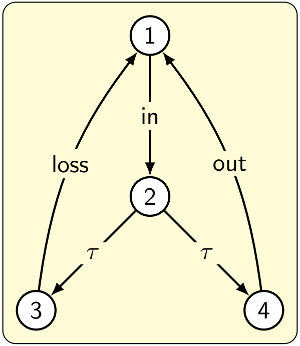
An example

$$X = \sum_{d:\,D} \mathsf{in}(d) \cdot (\tau \cdot \mathsf{loss} \cdot X + \tau \cdot \mathsf{out}(d) \cdot X)$$



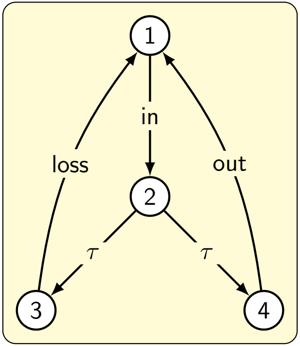
An example

$$X = \sum_{d:\,D} \mathsf{in}(d) \cdot (\tau \cdot \mathsf{loss} \cdot X + \tau \cdot \mathsf{out}(d) \cdot X)$$



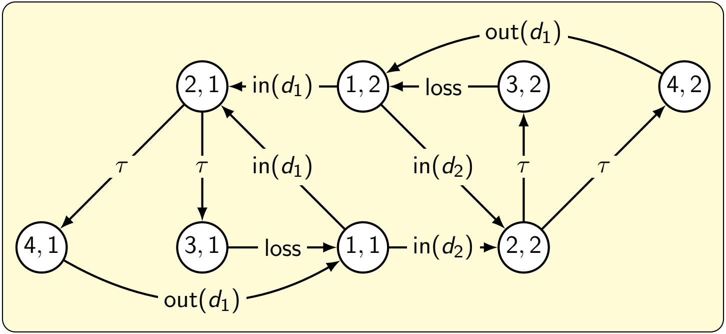
$$\begin{aligned}
& X(pc\colon \{1,2,3,4\}, x\colon D) = \\
& \quad \textstyle\sum_{d:\,D}\; pc = 1 \Rightarrow \mathsf{in}(d) \cdot X(2,d) \\
& + \quad pc = 2 \Rightarrow \tau \cdot X(3,x) \\
& + \quad pc = 2 \Rightarrow \tau \cdot X(4,x) \\
& + \quad pc = 3 \Rightarrow \mathsf{loss} \cdot X(1,x) \\
& + \quad pc = 4 \Rightarrow \mathsf{out}(x) \cdot X(1,x)
\end{aligned}$$

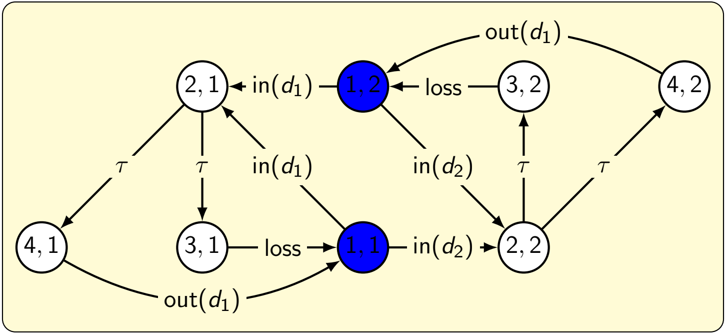
$$X = \sum_{d\colon D} \mathrm{in}(d) \cdot (\tau \cdot \mathrm{loss} \cdot X + \tau \cdot \mathrm{out}(d) \cdot X)$$

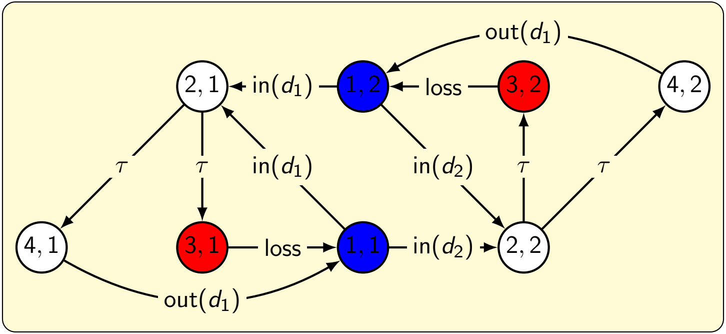


$$\begin{aligned}
X(pc\colon \{1,2,3,4\}, x\colon D) = & \\
\textstyle\sum_{d\colon D} \; & pc = 1 \Rightarrow \mathrm{in}(d) \cdot X(2,d) \\
+ \quad & pc = 2 \Rightarrow \tau \cdot X(3,x) \\
+ \quad & pc = 2 \Rightarrow \tau \cdot X(4,x) \\
+ \quad & pc = 3 \Rightarrow \mathrm{loss} \cdot X(1,x) \\
+ \quad & pc = 4 \Rightarrow \mathrm{out}(x) \cdot X(1,x)
\end{aligned}$$

Initial process: $X(1, d_1)$.


out(d_1)
2, 1
in(d_1)
1, 2
loss
3, 2
4, 2
τ
τ
in(d_1)
in(d_2)
τ
τ
4, 1
3, 1
loss
1, 1
in(d_2)
2, 2
out(d_1)


out(d_1)
2, 1
in(d_1)
1, 2
loss
3, 2
4, 2
τ
τ
in(d_1)
in(d_2)
τ
τ
4, 1
3, 1
loss
1, 1
in(d_2)
2, 2
out(d_1)


out(d_1)
2, 1
in(d_1)
1, 2
loss
3, 2
4, 2
τ
τ
in(d_1)
in(d_2)
τ
τ
4, 1
3, 1
loss
1, 1
in(d_2)
2, 2
out(d_1)

Problem: control flow is hidden in state parameters

Control Flow Reconstruction

Problem: control flow is hidden in state parameters

Solution:

1. Detect control flow parameters
2. Identify clusters of summands
3. Assign data parameters to clusters
4. Deduce when data parameters are relevant

Contents

Control flow parameters

Observation:
program counters (control flow parameters) are special.

Control flow parameters

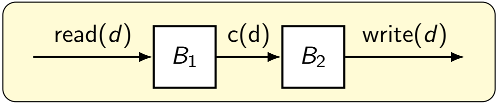
Observation:
program counters (control flow parameters) are special.

$$\boxed{B_1 = \sum_{d\,:\,D} read(d) \cdot w(d) \cdot B_1} \quad \boxed{B_2 = \sum_{d\,:\,D} r(d) \cdot write(d) \cdot B_2}$$

Observation:

program counters (control flow parameters) are special.

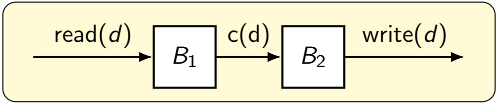
$\boxed{B_1 = \sum_{d\,:\,D} read(d) \cdot w(d) \cdot B_1}$ $\boxed{B_2 = \sum_{d\,:\,D} r(d) \cdot write(d) \cdot B_2}$



Control flow parameters

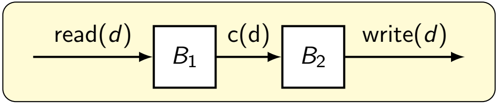
Observation:
program counters (control flow parameters) are special.

$$\boxed{B_1 = \sum_{d\colon D} read(d) \cdot w(d) \cdot B_1} \quad \boxed{B_2 = \sum_{d\colon D} r(d) \cdot write(d) \cdot B_2}$$



$$
\begin{array}{lll}
\multicolumn{3}{l}{X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) =} \\
\sum_{d\colon D} & a = 1 & \Rightarrow \mathrm{read}(d) \cdot X(2, b, d, y) \\
+ & b = 2 & \Rightarrow \mathrm{write}(y) \cdot X(a, 1, x, y) \\
+ & a = 2 \wedge b = 1 & \Rightarrow c(x) \cdot X(1, 2, x, x)
\end{array}
$$

Control flow parameters

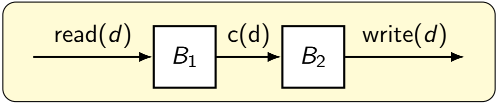


$$
\begin{array}{lll}
\multicolumn{3}{l}{X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) =} \\
\sum_{d\colon D} & a = 1 & \Rightarrow \mathrm{read}(d) \cdot X(2,b,d,y) \\
+ & b = 2 & \Rightarrow \mathrm{write}(y) \cdot X(a,1,x,y) \\
+ & a = 2 \wedge b = 1 & \Rightarrow c(x) \cdot X(1,2,x,x)
\end{array}
$$

In every summand, each control flow parameter

- is either left unchanged, or
- has a clear transition from a source value to a destination value

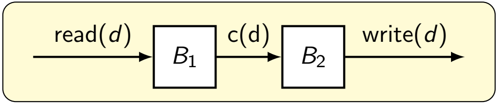
Control flow parameters



$$
\begin{array}{llll}
X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) = & & & \\
\quad \sum_{d\colon D} & a = 1 & \Rightarrow & \mathrm{read}(d) \cdot X(2, b, d, y) \\
+ & b = 2 & \Rightarrow & \mathrm{write}(y) \cdot X(a, 1, x, y) \\
+ & a = 2 \wedge b = 1 & \Rightarrow & c(x) \cdot X(1, 2, x, x)
\end{array}
$$

In every summand, each control flow parameter

- is either left unchanged, or
- has a clear transition from a source value to a destination value

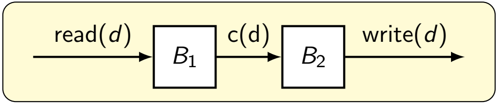


$$
\begin{array}{lllll}
X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) = & & & & \\
 & \sum_{d\colon D} & a = 1 & \Rightarrow & \mathrm{read}(d) \cdot X(2, b, d, y) \\
+ & & b = 2 & \Rightarrow & \mathrm{write}(y) \cdot X(a, 1, x, y) \\
+ & & a = 2 \wedge b = 1 & \Rightarrow & c(x) \cdot X(1, 2, x, x)
\end{array}
$$

In every summand, each control flow parameter

- is either left unchanged, or
- has a clear transition from a source value to a destination value

Control flow parameters

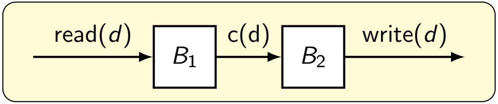


$$
\begin{aligned}
X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) = & \\
\textstyle\sum_{d\colon D} \quad & a = 1 && \Rightarrow \mathrm{read}(d) \cdot X(2, b, d, y) \\
+ \quad & b = 2 && \Rightarrow \mathrm{write}(y) \cdot X(a, 1, x, y) \\
+ \quad & a = {\color{red}2} \wedge b = {\color{red}1} && \Rightarrow c(x) \cdot X({\color{red}1}, {\color{red}2}, x, x)
\end{aligned}
$$

In every summand, each control flow parameter

- is either left unchanged, or
- has a clear transition from a source value to a destination value

Control flow parameters

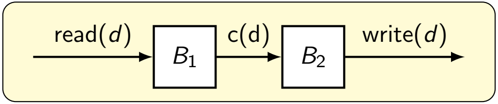


$$
\begin{array}{llll}
X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) = & & & \\
 & \sum_{d\colon D} & a = 1 & \Rightarrow \mathrm{read}(d) \cdot X(2, b, d, y) \\
+ & & b = 2 & \Rightarrow \mathrm{write}(y) \cdot X(a, 1, x, y) \\
+ & & a = 2 \wedge b = 1 & \Rightarrow c(x) \cdot X(1, 2, x, x)
\end{array}
$$

In every summand, each control flow parameter

- is either left unchanged, or
- has a clear transition from a source value to a destination value

Control flow parameters



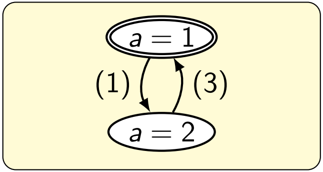
$$
\begin{array}{llll}
X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) = & & & \\
 & \sum_{d\colon D} & a = 1 & \Rightarrow \mathrm{read}(d) \cdot X(2, b, d, y) \\
 + & & b = 2 & \Rightarrow \mathrm{write}(y) \cdot X(a, 1, x, y) \\
 + & & a = 2 \wedge b = 1 & \Rightarrow c(x) \cdot X(1, 2, x, x)
\end{array}
$$

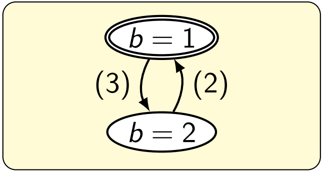
In every summand, each control flow parameter

- is either left unchanged, or
- has a clear transition from a source value to a destination value (it rules the summand)

Control Flow Graph

$$
\begin{array}{lllll}
X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) = & & & & \\
\quad \sum_{d\colon D} & a = 1 & \Rightarrow \mathrm{read}(d) \cdot X(2, b, d, y) & & (1) \\
+ & b = 2 & \Rightarrow \mathrm{write}(y) \cdot X(a, 1, x, y) & & (2) \\
+ & a = 2 \land b = 1 & \Rightarrow c(x) \cdot X(1, 2, x, x) & & (3)
\end{array}
$$





Contents

The *belongs to* relation

A data parameter k belongs to a CFP j if the cluster of j contains all summands that

- either change k, or
- make use of k (in an action, condition or next-state)

A data parameter k belongs to a CFP j if the cluster of j contains all summands that

- either change k, or
- make use of k (in an action, condition or next-state)

$$
\begin{array}{llll}
X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) = & & & \\
\quad \sum_{d\,:\,D} & a = 1 & \Rightarrow \mathrm{read}(d) \cdot X(2, b, \mathbf{d}, y) & (1) \\
+ & b = 2 & \Rightarrow \mathrm{write}(\mathbf{y}) \cdot X(a, 1, x, y) & (2) \\
+ & a = 2 \land b = 1 & \Rightarrow c(\mathbf{x}) \cdot X(1, 2, x, \mathbf{x}) & (3)
\end{array}
$$

The *belongs to* relation

A data parameter k belongs to a CFP j if the cluster of j contains all summands that

- either change k, or
- make use of k (in an action, condition or next-state)

$$
\begin{array}{lll}
\multicolumn{3}{l}{X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) =} \\
\quad \sum_{d\colon D} & a = 1 & \Rightarrow \mathrm{read}(d) \cdot X(2, b, \mathbf{d}, y) \quad (1) \\
+ & b = 2 & \Rightarrow \mathrm{write}(\mathbf{y}) \cdot X(a, 1, x, y) \quad (2) \\
+ & a = 2 \wedge b = 1 & \Rightarrow c(\mathbf{x}) \cdot X(1, 2, x, \mathbf{x}) \quad (3)
\end{array}
$$

So, x belongs to a and y belongs to b.

Thus, relevance of x can be decided by the control flow of a.

Relevance

$R(k, j, s)$: parameter k is relevant when CFP j is in state s

Relevance

$R(k, j, s)$: parameter k is relevant when CFP j is in state $s \iff$

There is a summand that can be taken when $d_j = s$, that either

- directly uses k for its condition or action, or
- indirectly uses k to determine the value of a parameter that is relevant after taking the summand

Relevance

$R(k, j, s)$: parameter k is relevant when CFP j is in state $s \Longleftrightarrow$

There is a summand that can be taken when $d_j = s$, that either

- directly uses k for its condition or action, or
- indirectly uses k to determine the value of a parameter that is relevant after taking the summand

$$
\begin{array}{llll}
X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) = & & & \\
\quad \sum_{d\colon D} & a = 1 & \Rightarrow \mathrm{read}(d) \cdot X(2, b, d, y) & (1) \\
+ & b = 2 & \Rightarrow \mathrm{write}(y) \cdot X(a, 1, x, y) & (2) \\
+ & a = 2 \wedge b = 1 & \Rightarrow \tau \cdot X(1, 2, x, x) & (3)
\end{array}
$$

Relevance

$R(k, j, s)$: parameter k is relevant when CFP j is in state $s \iff$

There is a summand that can be taken when $d_j = s$, that either

- directly uses k for its condition or action, or
- indirectly uses k to determine the value of a parameter that is relevant after taking the summand

$$
\begin{array}{llll}
X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) = & & & \\
\quad \sum_{d\colon D} & a = 1 & \Rightarrow \mathrm{read}(d) \cdot X(2, b, d, y) & (1) \\
+ & b = 2 & \Rightarrow \mathrm{write}(y) \cdot X(a, 1, x, y) & (2) \\
+ & a = 2 \wedge b = 1 & \Rightarrow \tau \cdot X(1, 2, x, x) & (3)
\end{array}
$$

Relevance

$R(k, j, s)$: parameter k is relevant when CFP j is in state $s \Longleftrightarrow$

There is a summand that can be taken when $d_j = s$, that either

- directly uses k for its condition or action, or
- indirectly uses k to determine the value of a parameter that is relevant after taking the summand

$$
\begin{array}{lllr}
X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) = & & & \\
\quad \sum_{d\colon D} & a = 1 & \Rightarrow \mathrm{read}(d) \cdot X(2, b, d, y) & (1) \\
+ & b = 2 & \Rightarrow \mathrm{write}(y) \cdot X(a, 1, x, y) & (2) \\
+ & a = 2 \wedge b = 1 & \Rightarrow \tau \cdot X(1, 2, x, x) & (3)
\end{array}
$$

So: $R(y, b, 2)$

Relevance

$R(k, j, s)$: parameter k is relevant when CFP j is in state $s \iff$

There is a summand that can be taken when $d_j = s$, that either

- directly uses k for its condition or action, or
- indirectly uses k to determine the value of a parameter that is relevant after taking the summand

$$
\begin{array}{llll}
X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) = & & & \\
\quad \sum_{d\colon D} & a = 1 & \Rightarrow \mathrm{read}(d) \cdot X(2, b, d, y) & (1) \\
\quad + & b = 2 & \Rightarrow \mathrm{write}(y) \cdot X(a, 1, x, y) & (2) \\
\quad + & a = 2 \wedge b = 1 & \Rightarrow \tau \cdot X(1, 2, x, x) & (3)
\end{array}
$$

So: $R(y, b, 2)$

Relevance

$R(k, j, s)$: parameter k is relevant when CFP j is in state $s \iff$

There is a summand that can be taken when $d_j = s$, that either

- directly uses k for its condition or action, or
- indirectly uses k to determine the value of a parameter that is relevant after taking the summand

$$
\begin{array}{llll}
X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) = & & & \\
\quad \sum_{d\colon D} & a = 1 & \Rightarrow \mathrm{read}(d) \cdot X(2, b, d, y) & (1) \\
+ & b = 2 & \Rightarrow \mathrm{write}(y) \cdot X(a, 1, x, y) & (2) \\
+ & a = 2 \wedge b = 1 & \Rightarrow \tau \cdot X(1, 2, x, x) & (3)
\end{array}
$$

So: $R(y, b, 2)$ and $R(x, a, 2)$.

Relevance

$R(k,j,s)$: parameter k is relevant when CFP j is in state $s \iff$

There is a summand that can be taken when $d_j = s$, that either

- directly uses k for its condition or action, or
- indirectly uses k to determine the value of a parameter that is relevant after taking the summand

$$
\begin{array}{llll}
X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) = & & & \\
\quad \sum_{d\colon D} & a = 1 & \Rightarrow \mathrm{read}(d) \cdot X(2,b,d,y) & (1) \\
+ & b = 2 & \Rightarrow \mathrm{write}(y) \cdot X(a,1,x,y) & (2) \\
+ & a = 2 \wedge b = 1 & \Rightarrow \tau \cdot X(1,2,x,x) & (3)
\end{array}
$$

So: $R(y,b,2)$ and $R(x,a,2)$.

If $\neg R(k,j,s)$, then k is irrelevant when j is in state s

Contents

Transformation

Based on data flow analysis, irrelevant parameters can be changed.

Transformation

Based on data flow analysis, irrelevant parameters can be changed.

▷ To never increase the state space, replace by their initial value.

Transformation

Based on data flow analysis, irrelevant parameters can be changed.
▷ To never increase the state space, replace by their initial value.

$$
\begin{array}{llll}
X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) = & & & \\
\quad \sum_{d\,:\,D} & a = 1 & \Rightarrow \mathsf{read}(d) \cdot X(2, b, d, y) & (1) \\
+ & b = 2 & \Rightarrow \mathsf{write}(y) \cdot X(a, 1, x, y) & (2) \\
+ & a = 2 \wedge b = 1 & \Rightarrow c(x) \cdot X(1, 2, x, x) & (3)
\end{array}
$$

We saw: $\neg R(x, a, 1)$ and $\neg R(y, b, 1)$.

Based on data flow analysis, irrelevant parameters can be changed.
▷ To never increase the state space, replace by their initial value.

$$
\begin{array}{llll}
X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) = & & & \\
\quad \sum_{d\colon D} & a = 1 & \Rightarrow \mathsf{read}(d) \cdot X(2, b, d, y) & (1) \\
+ & b = 2 & \Rightarrow \mathsf{write}(y) \cdot X(a, 1, x, y) & (2) \\
+ & a = 2 \wedge b = 1 & \Rightarrow c(x) \cdot X(1, 2, x, x) & (3)
\end{array}
$$

We saw: $\neg R(x, a, 1)$ and $\neg R(y, b, 1)$.
So, assuming initially $x = y = d_1$

Transformation

Based on data flow analysis, irrelevant parameters can be changed.
▷ To never increase the state space, replace by their initial value.

$$
\begin{array}{llll}
X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) = & & & \\
\quad \sum_{d\,:\,D} & a = 1 & \Rightarrow \mathsf{read}(d) \cdot X(2, b, d, y) & (1) \\
+ & b = 2 & \Rightarrow \mathsf{write}(y) \cdot X(a, 1, x, y) & (2) \\
+ & a = 2 \wedge b = 1 & \Rightarrow c(x) \cdot X(1, 2, x, x) & (3)
\end{array}
$$

We saw: $\neg R(x, a, 1)$ and $\neg R(y, b, 1)$.
So, assuming initially $x = y = d_1$

Transformation

Based on data flow analysis, irrelevant parameters can be changed.
▷ To never increase the state space, replace by their initial value.

$$
\begin{array}{llll}
\multicolumn{4}{l}{X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) =} \\
\sum_{d\,:\,D} & a = 1 & \Rightarrow \mathsf{read}(d)\cdot X(2,b,d,y) & (1) \\
+ & b = 2 & \Rightarrow \mathsf{write}(y)\cdot X(a,1,x,y) & (2) \\
+ & a = 2 \wedge b = 1 & \Rightarrow c(x)\cdot X(1,2,x,x) & (3)
\end{array}
$$

We saw: $\neg R(x,a,1)$ and $\neg R(y,b,1)$.
So, assuming initially $x = y = d_1$

Based on data flow analysis, irrelevant parameters can be changed.
▷ To never increase the state space, replace by their initial value.

$$
\begin{array}{llll}
\multicolumn{4}{l}{X(a\colon\{1,2\}, b\colon\{1,2\}, x\colon D, y\colon D) =} \\
\sum_{d\colon D} & a = 1 & \Rightarrow \mathsf{read}(d)\cdot X(2,b,d,y) & (1)\\
+ & b = 2 & \Rightarrow \mathsf{write}(y)\cdot X(a,1,x,y) & (2)\\
+ & a = 2 \wedge b = 1 & \Rightarrow c(x)\cdot X(1,2,x,x) & (3)
\end{array}
$$

We saw: $\neg R(x,a,1)$ and $\neg R(y,b,1)$.
So, assuming initially $x = y = d_1$

$$
\begin{array}{llll}
\multicolumn{4}{l}{X(a\colon\{1,2\}, b\colon\{1,2\}, x\colon D, y\colon D) =} \\
\sum_{d\colon D} & a = 1 & \Rightarrow \mathsf{read}(d)\cdot X(2,b,d,y) & (1)\\
+ & b = 2 & \Rightarrow \mathsf{write}(y)\cdot X(a,1,x,d_1) & (2)\\
+ & a = 2 \wedge b = 1 & \Rightarrow c(x)\cdot X(1,2,d_1,x) & (3)
\end{array}
$$

Example

$$
\begin{array}{llll}
X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) = & & & \\
\quad \sum_{d\,:\,D} & a = 1 & \Rightarrow \mathrm{read}(d) \cdot X(2, b, d, y) & (1) \\
+ & b = 2 & \Rightarrow \mathrm{write}(y) \cdot X(a, 1, x, d_1) & (2) \\
+ & a = 2 \wedge b = 1 & \Rightarrow c(x) \cdot X(1, 2, d_1, x) & (3)
\end{array}
$$

Example

$$
\begin{array}{llll}
X(a\colon \{1,2\}, b\colon \{1,2\}, x\colon D, y\colon D) = & & & \\
\quad \sum_{d\,:\,D} & a = 1 & \Rightarrow \mathrm{read}(d) \cdot X(2, b, d, y) & (1) \\
+ & b = 2 & \Rightarrow \mathrm{write}(y) \cdot X(a, 1, x, d_1) & (2) \\
+ & a = 2 \wedge b = 1 & \Rightarrow c(x) \cdot X(1, 2, d_1, x) & (3)
\end{array}
$$

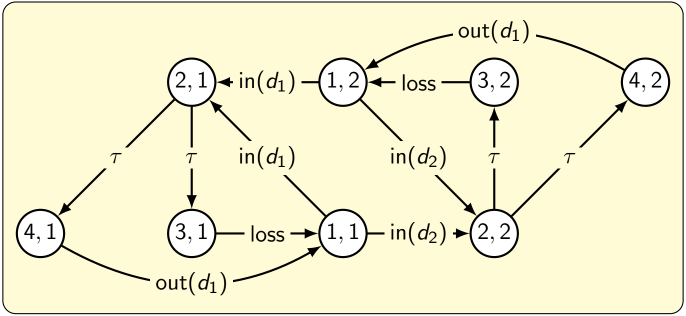
Now:

- $X(1, 2, x, y)$ are only reachable for $x = d_1$
- $X(2, 1, x, y)$ are only reachable for $y = d_1$

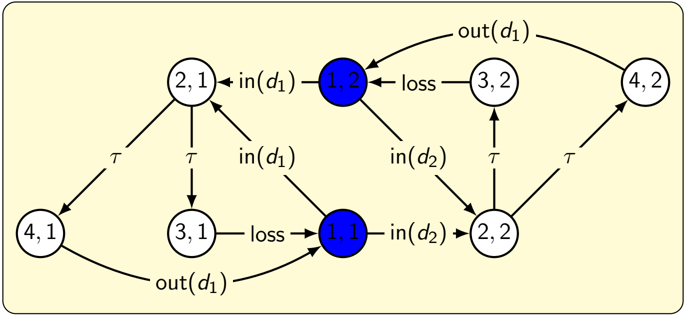
For $|D| = 5$, state space reduction of 60 to 36 states

For $|D| = n$, reduction of $2n^2 + 2n$ to $n^2 + 2n + 1$ states
(so a decrease of $n^2 - 1$ states)

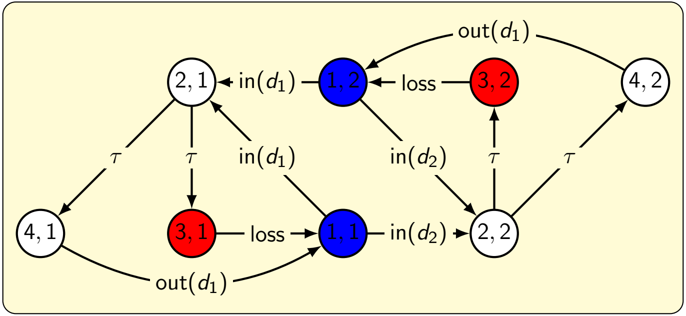
Example


out(d_1)
2, 1
in(d_1)
1, 2
loss
3, 2
4, 2
τ
τ
in(d_1)
in(d_2)
τ
τ
4, 1
3, 1
loss
1, 1
in(d_2)
2, 2
out(d_1)

Example

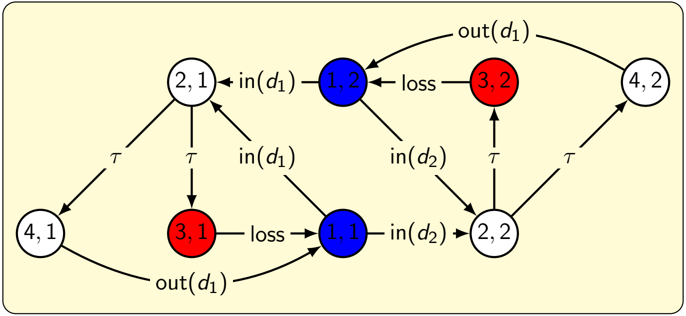

out(d_1)
2, 1
in(d_1)
1, 2
loss
3, 2
4, 2
τ
τ
in(d_1)
in(d_2)
τ
τ
4, 1
3, 1
loss
1, 1
in(d_2)
2, 2
out(d_1)

Example

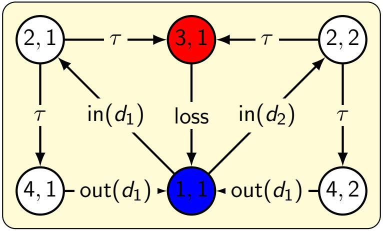


out(d_1)
2, 1
in(d_1)
1, 2
loss
3, 2
4, 2
τ
τ
in(d_1)
in(d_2)
τ
τ
4, 1
3, 1
loss
1, 1
in(d_2)
2, 2
out(d_1)

Example



out(d_1)
2, 1
in(d_1)
1, 2
loss
3, 2
4, 2
τ
τ
in(d_1)
in(d_2)
τ
τ
4, 1
3, 1
loss
1, 1
in(d_2)
2, 2
out(d_1)



2, 1
τ
3, 1
τ
2, 2
τ
in(d_1)
loss
in(d_2)
τ
4, 1
out(d_1)
1, 1
out(d_1)
4, 2

Correctness and effectiveness

Theorem: correctness

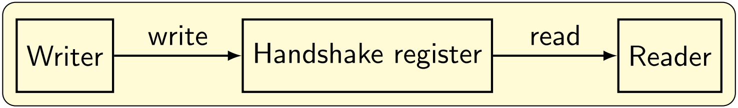
The transformed LPE is strongly bisimilar to the original

Theorem: effectiveness

The number of reachable states of the transformed LPE is at most as large as the number of reachable states in the original

Contents

A handshake register



- Recentness

  Any value read was at some point during the read action the last value written

- Sequentiality

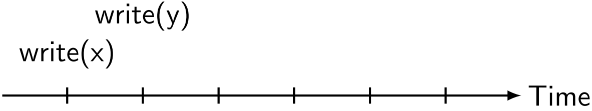
  The values of sequential reads occur in the same order as they were written

- Waitfree

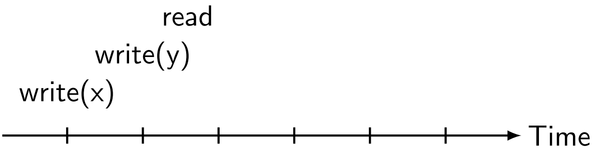
  Completion of reads/writes in a bounded number of steps

Building blocks for the handshake register

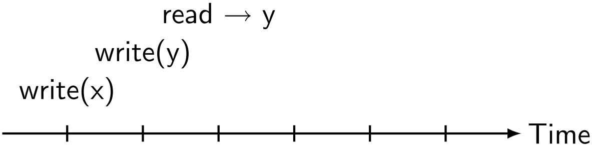
Building blocks for the handshake register



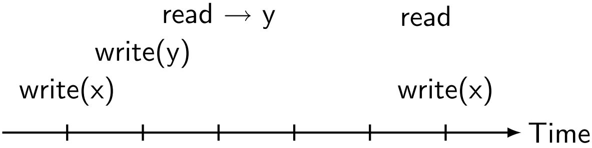
Building blocks for the handshake register



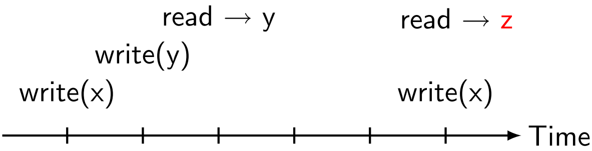
Building blocks for the handshake register



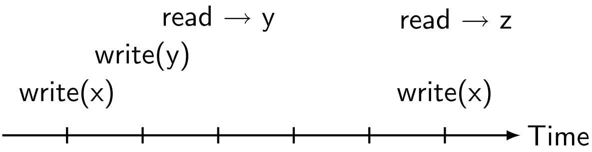
Building blocks for the handshake register




write
Safe register
read


read → y
read → z
write(y)
write(x)
write(x)
Time

Building blocks for the handshake register



- 4x safe register
- 4x atomic boolean register

Verifying the implementation

- Model the handshake register specification as a μCRL process
- Model the implementation as a μCRL process
- Generate their state spaces
- Minimise with respect to some equivalence ($\tau^* a$)
- Check for graph equivalence

Verifying the implementation

- Model the handshake register specification as a μCRL process
- Model the implementation as a μCRL process
- Generate their state spaces
- Minimise with respect to some equivalence ($\tau^* a$)
- Check for graph equivalence

Problem: state space explosion

Verifying the implementation

- Model the handshake register specification as a μCRL process
- Model the implementation as a μCRL process
- Generate their state spaces
- Minimise with respect to some equivalence ($\tau^* a$)
- Check for graph equivalence

Problem: state space explosion

Solution: Apply `stategraph`! (and compare to `parelm`)

Applying stategraph

	constelm | parelm | constelm		
	states	time (expl.)	time (symb.)
$\lvert D\rvert = 2$	540,736	0:23.0	0:04.5
$\lvert D\rvert = 3$	13,834,800	10:10.3	0:06.7
$\lvert D\rvert = 4$	142,081,536	–	0:09.0
$\lvert D\rvert = 5$	883,738,000	–	0:11.9
$\lvert D\rvert = 6$	3,991,840,704	–	0:15.4

	constelm | stategraph | constelm		
	states	time (expl.)	time (symb.)
$\lvert D\rvert = 2$	45,504	0:02.4	0:01.3
$\lvert D\rvert = 3$	290,736	0:12.7	0:01.4
$\lvert D\rvert = 4$	1,107,456	0:48.9	0:01.6
$\lvert D\rvert = 5$	3,162,000	2:20.3	0:01.8
$\lvert D\rvert = 6$	7,504,704	5:26.1	0:01.9

Why the reduction was obtained

$$
\begin{array}{llllr}
\multicolumn{5}{l}{Y(i\colon \mathrm{Bool}, j\ \colon \mathrm{Bool}, r\colon \{1,2,3\}, w\colon \{1,2,3\}, v\colon D, vw\colon D, vr\colon D) =} \\
 & & r = 1 & \Rightarrow \mathrm{beginRead}(i,j) \cdot Y(i,j,2,w,v,vw,vr) & (1) \\
+ & & r = 2 \wedge w = 1 & \Rightarrow \tau \cdot Y(i,j,3,w,v,vw,v) & (2) \\
+ & \sum_{x\colon D} & r = 2 \wedge w \neq 1 & \Rightarrow \tau \cdot Y(i,j,3,w,v,vw,x) & (3) \\
+ & & r = 3 & \Rightarrow \mathrm{endRead}(i,j,vr) \cdot Y(i,j,1,w,v,vw,vr) & (4) \\
+ & \sum_{x\colon D} & w = 1 & \Rightarrow \mathrm{beginWrite}(i,j,x) \cdot Y(i,j,r,2,v,x,vr) & (5) \\
+ & & w = 2 & \Rightarrow \tau \cdot Y(i,j,r,3,vw,vw,vr) & (6) \\
+ & & w = 3 & \Rightarrow \mathrm{endWrite}(i,j) \cdot Y(i,j,r,1,vw,vw,vr) & (7)
\end{array}
$$

Why the reduction was obtained

$$
\begin{array}{llll}
Y(i: \mathrm{Bool}, j & : \mathrm{Bool}, r: \{1,2,3\}, w: \{1,2,3\}, v: D, vw: D, \color{red}{vr: D}) = & & \\
 & r = 1 & \Rightarrow \mathrm{beginRead}(i,j) \cdot Y(i,j,2,w,v,vw,vr) & (1) \\
+ & r = 2 \wedge w = 1 & \Rightarrow \tau \cdot Y(i,j,3,w,v,vw,\color{blue}{v}) & (2) \\
+ \sum_{x:\,D} & r = 2 \wedge w \neq 1 & \Rightarrow \tau \cdot Y(i,j,3,w,v,vw,\color{blue}{x}) & (3) \\
+ & r = 3 & \Rightarrow \mathrm{endRead}(i,j,\color{red}{vr}) \cdot Y(i,j,1,w,v,vw,vr) & (4) \\
+ \sum_{x:\,D} & w = 1 & \Rightarrow \mathrm{beginWrite}(i,j,x) \cdot Y(i,j,r,2,v,x,vr) & (5) \\
+ & w = 2 & \Rightarrow \tau \cdot Y(i,j,r,3,vw,vw,vr) & (6) \\
+ & w = 3 & \Rightarrow \mathrm{endWrite}(i,j) \cdot Y(i,j,r,1,vw,vw,vr) & (7)
\end{array}
$$

Why the reduction was obtained

$$
\begin{array}{lllll}
Y(i\colon \mathrm{Bool}, j & : \mathrm{Bool}, r\colon \{1,2,3\}, w\colon \{1,2,3\}, v\colon D, vw\colon D, vr\colon D) = & & & \\
 & r = 1 & \Rightarrow \mathrm{beginRead}(i,j) \cdot Y(i,j,2,w,v,vw,vr) & & (1) \\
+ & r = 2 \wedge w = 1 & \Rightarrow \tau \cdot Y(i,j,3,w,v,vw,v) & & (2) \\
+ \sum_{x\colon D} & r = 2 \wedge w \neq 1 & \Rightarrow \tau \cdot Y(i,j,3,w,v,vw,x) & & (3) \\
+ & r = 3 & \Rightarrow \mathrm{endRead}(i,j,vr) \cdot Y(i,j,1,w,v,vw,d_1) & & (4) \\
+ \sum_{x\colon D} & w = 1 & \Rightarrow \mathrm{beginWrite}(i,j,x) \cdot Y(i,j,r,2,v,x,vr) & & (5) \\
+ & w = 2 & \Rightarrow \tau \cdot Y(i,j,r,3,vw,vw,vr) & & (6) \\
+ & w = 3 & \Rightarrow \mathrm{endWrite}(i,j) \cdot Y(i,j,r,1,vw,vw,vr) & & (7)
\end{array}
$$

Other case studies

Other specifications `stategraph` was applied to:

- An Automatic In-flight Data Acquisition unit for a helicopter
- A cache coherence protocol for a distributed JVM
- The sliding window protocol
- An automatic translation from Erlang to μCRL of a distributed resource locker in Ericsson's AXD 301 switch

Other case studies

Other specifications `stategraph` was applied to:

- An Automatic In-flight Data Acquisition unit for a helicopter
- A cache coherence protocol for a distributed JVM
- The sliding window protocol
- An automatic translation from Erlang to μCRL of a distributed resource locker in Ericsson's AXD 301 switch

Results:

- Reductions in the number of states (up to 20 percent)
- Reductions in the number of parameters (up to 75 percent)
- Reductions in the number of summands (up to 25 percent)

Contents

Conclusions

- Novel method for reconstructing control flow
  - Even control flow hiding in state parameters is found
- Data flow analysis based on this control flow
  - Resetting variables that are no longer relevant
  - Decreases in states, parameters and summands
  - Reductions obtained before generating the entire state space
- Precise proofs of correctness *and* decrease of state space
- Case studies show that impressive results are indeed obtained

Future work

- Investigate additional applications for the reconstructed control flow
    - Invariant generation
    - Visualisation (already implemented)
    - Improve confluence checking
- Use more precise abstractions based on control flow

Future work

- Investigate additional applications for the reconstructed control flow
  - Invariant generation
  - Visualisation (already implemented)
  - Improve confluence checking
- Use more precise abstractions based on control flow
- Apply these techniques to a probabilistic linear format (currently in development)

Questions

?