Analysis of Boolean Equation Systems through Structure Graphs

Michel A. Reniers   Tim A.C. Willemse   Jeroen J.A. Keiren

Department of Mathematics and Computer Science
Technische Universiteit Eindhoven

1. Introduction
2. Preliminaries
3. Structure Graphs for Boolean Equation Systems
   - Structure Graphs
   - Structure graphs for non-empty closed equation systems
4. Preservation and Reflection of Solution under Bisimilarity
   - true/false-Elimination
   - Normalisation
5. Application
6. Future work

Model Checking

μ-Calculus model checking problem: answer $L \models f$

- L is a *Labelled Transition System*;
- f is a μ-calculus formula;

- Model checking problem $=$ solving Boolean equation systems
- A BES is a sequence of fixed point equations
- Size BES $\mathcal{E}$ encoding $L \models f$ . . . . $\mathcal{O}(|L| \cdot |f|)$
- Solving BES $=$ finding the winner in Parity Games
- Algorithm solving BES $=$ algorithm for computing winner PG
- Computing winner in PG is in UP $\cap$ co-UP . . . . . . [Jurdziński'98]

Observations in practice

Minimisation pays .... [Keiren & Willemse'09]

- Bisimulation minimisations of underlying dependency graph;
- reductions for closed BESs in standard recursive form;
- Solving $\mathcal{E}$ greatly exceeds minimising $\mathcal{E}$ + solving minimised $\mathcal{E}$;

- Solving BES/Computing winners in PG:
  - Small Prog. Meas. [Jurdziński'00] .... $\mathcal{O}(\mathsf{ah}(\mathcal{E}) \cdot m \cdot (\frac{n}{2})^{\frac{\mathsf{ah}(\mathcal{E})}{2}})$
  - Bigstep [Schewe'07] .... $\mathcal{O}(n \cdot m^{\frac{\mathsf{ah}(\mathcal{E})}{3}})$
  - Strat. Impr. [Jurdziński & Vöge'00] .... $\mathcal{O}(2^{\mathsf{ah}(\mathcal{E})} \cdot n \cdot m)$
- Minimisation efficiently and substantially reduces n and m

Contributions of this work

- Graph model for BESs through SOS;
- Bisimulation minimisation for all closed BESs (not just SRF);
- Identified impact of transformation to SRF on minimisation;

1 Introduction

2 Preliminaries

3 Structure Graphs for Boolean Equation Systems
- Structure Graphs
- Structure graphs for non-empty closed equation systems

4 Preservation and Reflection of Solution under Bisimilarity
- true/false-Elimination
- Normalisation

5 Application

6 Future work

Definition (Boolean Equation System (BES))

A BES $\mathcal{E}$ is defined by the following grammar:

$$\begin{array}{ll} \mathcal{E} & ::= \epsilon \mid (\mu X = f)\ \mathcal{E} \mid (\nu X = f)\ \mathcal{E} \\ f, g & ::= c \mid X \mid f \vee g \mid f \wedge g \end{array}$$

- ϵ is the empty BES;
- μ, ν are least, resp., greatest fixed point signs;
- X is a proposition variable from a set $\mathcal{X}$;
- c is a constant from the set $\{\text{false}, \text{true}\}$;

A BES is in SRF if the formulae are given by the following grammar:

$$f \quad ::= X \mid \bigvee F \mid \bigwedge F$$

- where F is a non-empty set of proposition variables

bound/occurring proposition variables

$$\mathsf{bnd}(\epsilon) = \emptyset \qquad \mathsf{bnd}((\sigma X = f)\ \mathcal{E}) = \mathsf{bnd}(\mathcal{E}) \cup \{X\}$$

$$\mathsf{occ}(\epsilon) = \emptyset \qquad \mathsf{occ}((\sigma X = f)\ \mathcal{E}) = \mathsf{occ}(\mathcal{E}) \cup \mathsf{occ}(f)$$

$$\mathsf{occ}(c) = \emptyset \qquad \mathsf{occ}(f \vee g) = \mathsf{occ}(f) \cup \mathsf{occ}(g)$$
$$\mathsf{occ}(X) = \{X\} \qquad \mathsf{occ}(f \wedge g) = \mathsf{occ}(f) \cup \mathsf{occ}(g)$$

- $\mathcal{E}$ is *closed*: $\mathsf{occ}(\mathcal{E}) \subseteq \mathsf{bnd}(\mathcal{E})$
- $X \trianglelefteq Y$: the equation for X precedes Y's in $\mathcal{E}$
- $\mathcal{E}$ is well-formed: $\trianglelefteq$ is acyclic;

Proposition formulae are interpreted in the context of an environment $\eta{:}\mathcal{X} \to \mathbb{B}$, assigning Boolean values to proposition variables;

Definition (Interpretation of proposition formulae)

Let $\eta{:}\mathcal{X} \to \mathbb{B}$ be an environment. Interpretation $[\![f]\!]\eta$ maps a proposition formula f to true or false:

$$[\![c]\!]\eta = c \qquad [\![X]\!]\eta = \eta(X)$$

$$[\![f \vee g]\!]\eta = [\![f]\!]\eta \vee [\![g]\!]\eta \qquad [\![f \wedge g]\!]\eta = [\![f]\!]\eta \wedge [\![g]\!]\eta$$

The solution to a BES is defined in context of environment $\eta{:}\mathcal{X} \to \mathbb{B}$.
Peculiarities:

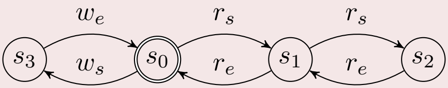
- Solution to closed BES independent of environment:

$$\forall X \in \mathsf{bnd}(\mathcal{E}) : \quad [\![\mathcal{E}]\!]\eta(X) = [\![\mathcal{E}]\!]\eta'(X) \text{ for all } \eta, \eta'$$

- Solution is order-sensitive:

$$[\![(\mu X = Y)\ (\nu Y = X)]\!] \neq [\![(\nu Y = X)\ (\mu X = Y)]\!]$$

Example (Mutual exclusion)



On some path a reader can infinitely often start reading:

$$\nu X.\mu Y.\ \langle r_s \rangle X \vee \langle \overline{r_s} \rangle Y$$

Corresponding BES (translation of [Mader'97]):

$$(\nu X_{s_0} = Y_{s_0})\ (\nu X_{s_1} = Y_{s_1})\ (\nu X_{s_2} = Y_{s_2})\ (\nu X_{s_3} = Y_{s_3})$$
$$(\mu Y_{s_0} = X_{s_1} \vee Y_{s_1})\ (\mu Y_{s_1} = X_{s_2} \vee Y_{s_0})\ (\mu Y_{s_2} = Y_{s_1})\ (\mu Y_{s_3} = Y_{s_0})$$

- rank(X) indicates in which block of like-signed equations X occurs.
- rank(X) is odd iff X is defined in a μ-equation.
- rank(X) inductively defined on structure of BES.

Example (Rank)

$$\begin{array}{llcl} (1) & \mu X & = & (X \wedge Y) \vee Z \\ (2) & \nu Y & = & W \vee (X \wedge Y) \\ (3) & \mu Z & = & Z \\ (3) & \mu W & = & Z \vee (Z \vee W) \end{array}$$

1. Introduction
2. Preliminaries
3. Structure Graphs for Boolean Equation Systems
   - Structure Graphs
   - Structure graphs for non-empty closed equation systems
4. Preservation and Reflection of Solution under Bisimilarity
   - true/false-Elimination
   - Normalisation
5. Application
6. Future work

Aim

Identify a graph structure capturing all information needed for solving equation systems.

It should (at least):

- capture ranks of bound variables;
- reflect the dependencies between bound variables and proposition formulae;
- reflect the dependencies between proposition formulae and its subformulae;
- indicate the top logical operators/constants of proposition formulae;

Definition (Structure Graph)

A structure graph is a graph $\mathcal{G} = \langle T, t, \rightarrow, d \rangle$, where:

- T is a finite set of formulae, and $t \in T$ is the initial formula;
- $\rightarrow \subseteq T \times T$ is a dependency relation;
- $d{:}T \rightarrow (2^{D_\blacktriangle} \cup 2^{D_\blacktriangledown} \cup 2^{D_\top} \cup 2^{D_\bot})$, where, for $e \in \{\blacktriangle, \blacktriangledown, \top, \bot\}$, $D_e = \mathbb{N} \cup \{e\}$, is a term decoration mapping;

$\mathcal{G}_i = \langle T_i, t_i, \rightarrow_i, d_i \rangle$; $R \subseteq T_0 \times T_1$ is a bisimulation if for all $u_0 \; R \; u_1$:

- $d_0(u_0) = d_1(u_1)$;
- for all $v_0 \in T_0$, if $u_0 \rightarrow_0 v_0$, then $u_1 \rightarrow_1 v_1$ for some $v_1 \in T_1$ satisfying $v_0 \; R \; v_1$; (similarly for $u_1 \rightarrow_1 v_1$)

$\mathcal{G}_0 \underline{\leftrightarrow} \mathcal{G}_1$ iff $t_0 \; R \; t_1$ for some bisimulation R

Definition

$\mathcal{G} = \langle T, t, \rightarrow, d \rangle$ is called BESsy if:

- a node t decorated by $\top$ or $\bot$ has no $\rightarrow$ successor;
- a node is decorated by ▲ or ▼ or a rank iff it has a $\rightarrow$ successor;
- a node with multiple $\rightarrow$ successors, is decorated with ▲ or ▼
- a node with rank 0 or 1 is reachable from t,
- the ranks of all reachable nodes form a closed interval

Observation:

- BESsyness is preserved under bisimilarity.

Definition (Transformation BESsy structure graph to BES)

Let $\mathcal{G} = \langle T, t, \rightarrow, d \rangle$ be a BESsy structure graph.

- For each node $u \in T$ with $d(u) \cap \mathbb{N} \neq \emptyset$ introduce an equation:

$$\sigma X_u = rhs(u) \quad \sigma = \begin{cases} \mu & \text{if rank}(X_u) \text{ is odd} \\ \nu & \text{otherwise} \end{cases}$$

$$\text{rank}(X_u) = \min(d(u) \cap \mathbb{N})$$

- The equation system $\mathcal{E}$ is obtained by ordering the equations from left-to-right based on the ranks of the variables;
- The equation system associated to $\mathcal{G}$ is denoted $\mathcal{E}_{\mathcal{G}}$;
- The formula associated to $\mathcal{G}$ is the formula $term(t)$ in the context of the equation system $\mathcal{E}_{\mathcal{G}}$.

For BESsy Structure Graphs, $term$ and rhs are defined as follows:

$$term(u) = \begin{cases} \bigsqcap\{term(u') \mid u \to u'\} & \text{if } d(u) = \{\blacktriangle\}, \\ \bigsqcup\{term(u') \mid u \to u'\} & \text{if } d(u) = \{\blacktriangledown\}, \\ \text{true} & \text{if } \top \in d(u), \\ \text{false} & \text{if } \bot \in d(u), \\ X_u & \text{otherwise,} \end{cases}$$

$$rhs(u) = \begin{cases} \bigsqcap\{term(u') \mid u \to u'\} & \text{if } \blacktriangle \in d(u), \\ \bigsqcup\{term(u') \mid u \to u'\} & \text{if } \blacktriangledown \in d(u), \\ term(u') & \text{otherwise, } u' \text{ s.t. } u \to u'. \end{cases}$$

$\bigsqcap\{t\} = t$ if $t \lessdot \min(T), T \neq \emptyset$, then $\bigsqcap(\{t\} \cup T) = t \wedge \left(\bigsqcap T\right)$

Mapping proposition formulae onto a Structure Graph via SOS

- Flatten nesting hierarchy of the same connective:

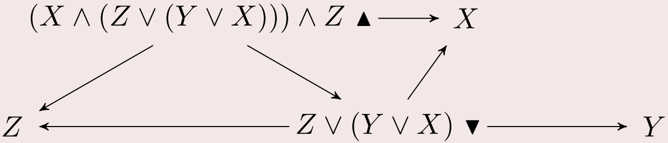
$$\frac{}{\textsf{true}\,\top} \qquad \frac{}{\textsf{false}\ \bot} \qquad \frac{}{(t \wedge t')\blacktriangle} \qquad \frac{}{(t \vee t')\blacktriangledown}$$

$$\frac{t\blacktriangle \quad t \rightarrow u}{t \wedge t' \rightarrow u} \qquad \frac{t\blacktriangledown \quad t \rightarrow u}{t \vee t' \rightarrow u}$$

- Dependencies on variables and constants:

$$\frac{\neg t\blacktriangle}{t \wedge t' \rightarrow t} \qquad \frac{\neg t\blacktriangledown}{t \vee t' \rightarrow t}$$

Example $((X \wedge (Z \vee (Y \vee X))) \wedge Z)$



Mapping equations onto a Structure Graph via SOS

- Ranking the variables:

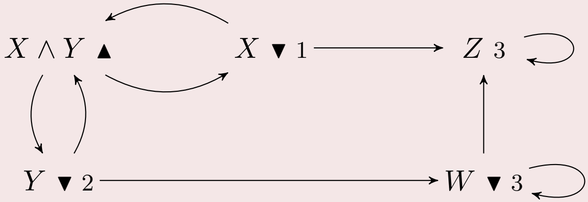
$$\frac{X \in \mathsf{bnd}(\mathcal{E}) \quad \mathsf{rank}(X) = n}{X \pitchfork n}$$

- The structure of a node representing a variable is derived from the right-hand side of the corresponding equation:

$$\frac{\sigma X = t \in \mathcal{E} \quad \neg t\blacktriangle \quad \neg t\blacktriangledown}{X \to t} \qquad \frac{\sigma X = t \in \mathcal{E} \quad t \to u}{X \to u}$$

Example (Equation system and associated structure graph)

$$
\begin{array}{lcl}
\mu X & = & (X \wedge Y) \vee Z \\
\nu Y & = & W \vee (X \wedge Y) \\
\mu Z & = & Z \\
\mu W & = & Z \vee (Z \vee W)
\end{array}
$$



Notation:

- $\mathcal{G}_{\mathcal{E},t}$ denotes structure graph associated to a formula t in the context of an equation system $\mathcal{E}$
- $\mathcal{G}_{\mathcal{E}} = \mathcal{G}_{\mathcal{E},X}$, where $X \in \mathsf{bnd}(\mathcal{E})$ is the least element w.r.t. $\trianglelefteq$

Lemma

Let $\mathcal{E}$ be a non-empty closed equation system. Let t, u, v be proposition formulae such that $\mathsf{occ}(t) \cup \mathsf{occ}(u) \cup \mathsf{occ}(v) \subseteq \mathsf{bnd}(\mathcal{E})$.

$$\mathcal{G}_{\mathcal{E},(t \wedge u) \wedge v} \leftrightarroweq \mathcal{G}_{\mathcal{E},t \wedge (u \wedge v)} \qquad \mathcal{G}_{\mathcal{E},t \wedge u} \leftrightarroweq \mathcal{G}_{\mathcal{E},u \wedge t}$$

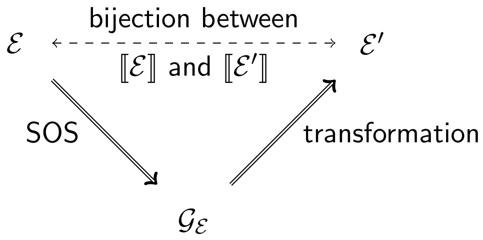
- Idempotency of $\wedge$ is not captured by bisimilarity

Theorem (Preservation of Solution)

- *$\mathcal{E}$ is a non-empty closed BES and $\mathcal{G}_\mathcal{E}$ is its Structure Graph;*
- *$\mathcal{E}'$ is obtained by transforming $\mathcal{G}_\mathcal{E}$ to BES;*

There is a total bijective mapping $h : \mathsf{bnd}(\mathcal{E}) \rightarrow \mathsf{bnd}(\mathcal{E}')$ such that:

$$\forall X \in \mathsf{bnd}(\mathcal{E}) : [\![\mathcal{E}]\!](X) = [\![\mathcal{E}']\!](h(X))$$



1 Introduction

2 Preliminaries

3 Structure Graphs for Boolean Equation Systems
- Structure Graphs
- Structure graphs for non-empty closed equation systems

4 Preservation and Reflection of Solution under Bisimilarity
- true/false-Elimination
- Normalisation

5 Application

6 Future work

Aim

Bisimilarity preserves and reflects solutions for BESsy structure graphs

Strategy:

$$\begin{array}{ccccc} \mathcal{G}_0 & \xrightarrow{\text{reduce}} & \mathcal{G}_0' & \xrightarrow{\text{normalise}} & \mathcal{G}_0'' \\ \updownarrow\!| & & \updownarrow\!| & & \updownarrow\!| \\ \mathcal{G}_1 & \xrightarrow{\text{reduce}} & \mathcal{G}_1' & \xrightarrow{\text{normalise}} & \mathcal{G}_1'' \end{array}$$

- reduce removes all constant decorations;
- normalise removes all nestings of $\wedge$ and $\vee$;
- normalised structure graph is bisimilar to graph for BES in SRF;

Theorem (Keiren & Willemse'09)

Bisimilarity preserves and reflects solution for BES in SRF

Eliminate false and true nodes

- false node maps to (fresh) node X with $\mu X = X$;
- true node maps to (fresh) node X with $\nu X = X$;
- ranks 0, resp. 1 assigned to new nodes;

Operator reduce is defined by the following deduction rules

$$\frac{t\blacktriangle}{\mathsf{reduce}(t)\blacktriangle} \qquad \frac{t\blacktriangledown}{\mathsf{reduce}(t)\blacktriangledown} \qquad \frac{\neg t\top \quad \neg t\perp \quad t \to u}{\mathsf{reduce}(t) \to \mathsf{reduce}(u)} \qquad \frac{\neg t\top \quad \neg t\perp \quad t \pitchfork n}{\mathsf{reduce}(t) \pitchfork n}$$

$$\frac{t\top}{\mathsf{reduce}(t) \to \mathsf{reduce}(t)} \qquad \frac{t\top}{\mathsf{reduce}(t) \pitchfork 0} \qquad \frac{t\perp}{\mathsf{reduce}(t) \to \mathsf{reduce}(t)} \qquad \frac{t\perp}{\mathsf{reduce}(t) \pitchfork 1}$$

Theorem (Congruence)

Bisimilarity is a congruence for reduce

Removing nesting of $\wedge$ and $\vee$

- Only non-ranked terms occur as subterms in right-hand sides of equations with nested occurrences of $\wedge$ and $\vee$;
- Rank non-ranked nodes based on ranks assigned to successors;

Operator normalise is defined by the following deduction rules

$$\frac{t\blacktriangle}{\mathsf{norm}(t)\blacktriangle} \qquad \frac{t\blacktriangledown}{\mathsf{norm}(t)\blacktriangledown} \qquad \frac{t \longrightarrow u}{\mathsf{norm}(t) \longrightarrow \mathsf{norm}(u)} \qquad \frac{t \pitchfork n}{\mathsf{norm}(t) \pitchfork n}$$

$$\frac{t \not\pitchfork \quad t \longrightarrow u \quad \mathsf{norm}(u) \pitchfork n \quad \forall_v\ t \longrightarrow v \Rightarrow \mathsf{norm}(v) \pitchfork m \wedge m \leq n}{\mathsf{norm}(t) \pitchfork n}$$

Theorem (Congruence)

Bisimilarity is a congruence for normalise

Notation:

- $\mathcal{G}_{\mathsf{norm}(\mathcal{E})}$ denotes the structure graph $\mathcal{G}_{\mathcal{E},\mathsf{norm}(X)}$ where X is the least variable w.r.t. $\trianglelefteq$

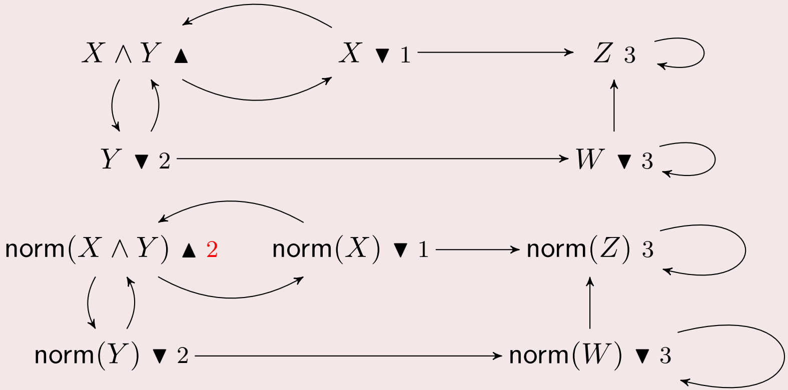
Lemma (Normalisation preserves and reflects solution)

- *$\mathcal{E}$ is a non-empty, closed equation system;*
- *$\mathcal{E}_{norm}$ is the equation system obtained by transforming $\mathcal{G}_{\mathsf{norm}(\mathcal{E})}$ into an equation system;*

There is a total injective mapping $h : \mathsf{bnd}(\mathcal{E}) \to \mathsf{bnd}(\mathcal{E}_{norm})$ such that:

$$\forall X \in \mathsf{bnd}(\mathcal{E}) : [\![\mathcal{E}]\!](X) = [\![\mathcal{E}_{norm}]\!](h(X)).$$

Example (Normalisation)



Theorem

- $\mathcal{E}$ *and* $\mathcal{E}'$ *are non-empty, closed equation systems;*
- f *w.r.t.* $\mathcal{G}_{\mathcal{E}}$ *and* f' *w.r.t.* $\mathcal{G}_{\mathcal{E}'}$ *are bisimilar;*

$$[\![f]\!][\![\mathcal{E}]\!] = [\![f']\!][\![\mathcal{E}']\!]$$

1 Introduction

2 Preliminaries

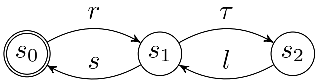
3 Structure Graphs for Boolean Equation Systems
- Structure Graphs
- Structure graphs for non-empty closed equation systems

4 Preservation and Reflection of Solution under Bisimilarity
- true/false-Elimination
- Normalisation

5 Application

6 Future work

Academic example: LTS modelling an unreliable channel:



- Read a message from the environment . . . . action r;
- Send a message to the environment . . . . action s;
- Lose a message . . . . action l;

Model checking problem

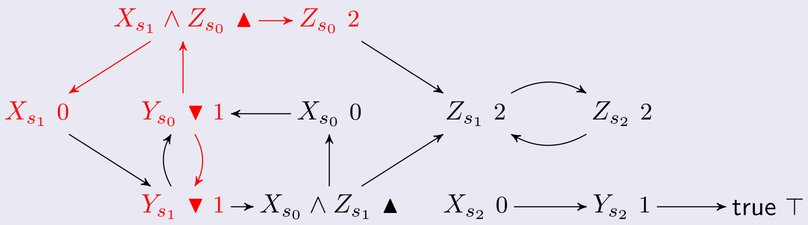
Determine all s_i that can infinitely often never send a message on all reading/sending paths:

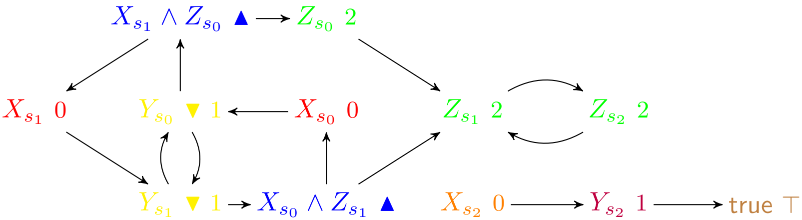
$$s_i \models \nu X.\mu Y.(([r]X \wedge [s]X \wedge (\nu Z.\langle \overline{s} \rangle Z)) \vee ([r]Y \wedge [s]Y))$$

The solution to X_{s_i} (in the BES below) answers whether $s_i \models \phi$

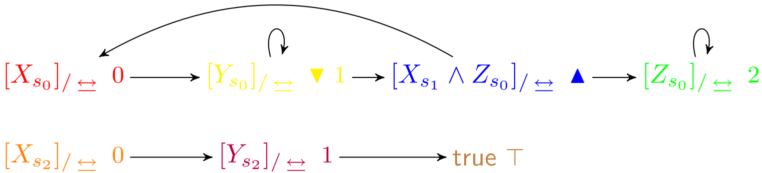
$$(\nu X_{s_0} = Y_{s_0})\ (\nu X_{s_1} = Y_{s_1})\ (\nu X_{s_2} = Y_{s_2})$$
$$(\mu Y_{s_0} = (X_{s_1} \wedge Z_{s_0}) \vee Y_{s_1})\ (\mu Y_{s_1} = (X_{s_0} \wedge Z_{s_1}) \vee Y_{s_0})\ (\mu Y_{s_2} = \mathsf{true})$$
$$(\nu Z_{s_0} = Z_{s_1})\ (\nu Z_{s_1} = Z_{s_2})\ (\nu Z_{s_2} = Z_{s_1})$$

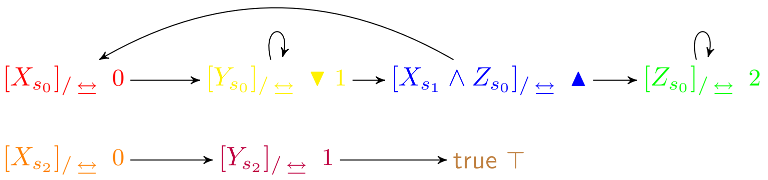
Associated structure graph





Bisimulation minimised structure graph:





Minimised BES

$$
\begin{array}{l}
(\nu X_{[X_{s_0}]_{/\underline{\leftrightarrow}}} = X_{[Y_{s_0}]_{/\underline{\leftrightarrow}}})\ (\nu X_{[X_{s_2}]_{/\underline{\leftrightarrow}}} = X_{[Y_{s_2}]_{/\underline{\leftrightarrow}}})\\
(\mu X_{[Y_{s_0}]_{/\underline{\leftrightarrow}}} = (X_{[X_{s_0}]_{/\underline{\leftrightarrow}}} \wedge X_{[Z_{s_0}]_{/\underline{\leftrightarrow}}}) \vee X_{[Y_{s_0}]_{/\underline{\leftrightarrow}}})\ (\mu X_{[Y_{s_2}]_{/\underline{\leftrightarrow}}} = \mathsf{true})\\
(\nu X_{[Z_{s_0}]_{/\underline{\leftrightarrow}}} = X_{[Z_{s_0}]_{/\underline{\leftrightarrow}}})
\end{array}
$$

Minimised BES:

$$(\nu X_{[X_{s_0}]/\underline{\leftrightarrow}} = X_{[Y_{s_0}]/\underline{\leftrightarrow}})\ (\nu X_{[X_{s_2}]/\underline{\leftrightarrow}} = X_{[Y_{s_2}]/\underline{\leftrightarrow}})$$
$$(\mu X_{[Y_{s_0}]/\underline{\leftrightarrow}} = (X_{[X_{s_0}]/\underline{\leftrightarrow}} \wedge X_{[Z_{s_0}]/\underline{\leftrightarrow}}) \vee X_{[Y_{s_0}]/\underline{\leftrightarrow}})\ (\mu X_{[Y_{s_2}]/\underline{\leftrightarrow}} = \mathsf{true})$$
$$(\nu X_{[Z_{s_0}]/\underline{\leftrightarrow}} = X_{[Z_{s_0}]/\underline{\leftrightarrow}})$$

Observations:

- Answering $s_i \models \phi$ requires solving 5 equations instead of 9. All equations have true as their solution;
- Original LTS: unreducable using strong bisimulation;
- BES size reduction: roughly 50%;

1. Introduction
2. Preliminaries
3. Structure Graphs for Boolean Equation Systems
   - Structure Graphs
   - Structure graphs for non-empty closed equation systems
4. Preservation and Reflection of Solution under Bisimilarity
   - true/false-Elimination
   - Normalisation
5. Application
6. Future work

Future work:

- Extension to open equation systems and concatenation of such equation systems;
- Extension to other coarser notion of equivalence (stuttering equivalence);