CS-E3220 Declarative Programming

OBDD-Based Reachability Methods

Jussi Rintanen

Department of Computer Science
Aalto University

October 9, 2019

Reachability Analysis

- Computer-Aided Verification: Is a bad state reachable?
  - Deadlock: a state in which no progress can be made
  - Other consistency conditions: facts *a* and *b* true simultaneously
  - A correct system: no bad states are reachable
- Automated decision making: How to reach a good state? (a goal)
  - planning, problem solving, decision-making, control
  - A correct system: good states are (generally) reachable

Reachability by Breadth-First Traversal

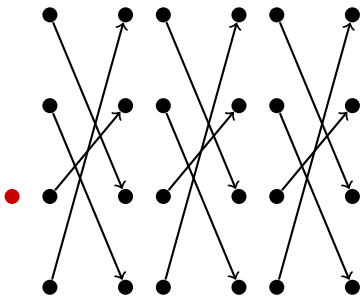
Compute set of all states reachable from S_0, when one-step transition relation is R:

1. $i := 0$
2. $S_0 :=$ initial states
3. $i := i + 1$
4. $S_i := S_{i-1} \cup \pi_2(S_{i-1} \bowtie R)$
5. if $S_i \not\subseteq S_{i-1}$, go to 3.

We first do this with relations, and then with logic!

Reachability by Breadth-First Traversal: Example

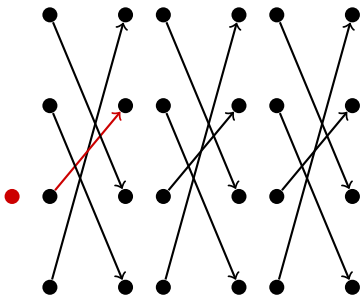
View reachability in terms of relations:



3-step reachability: initial state(s), 3 copies of the transition relation

Reachability by Breadth-First Traversal: Example

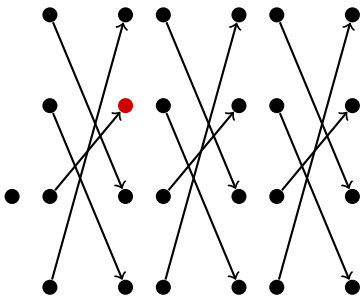
View reachability in terms of relations:



states 1 step away: limit the relation to initial states (relational join)

Reachability by Breadth-First Traversal: Example

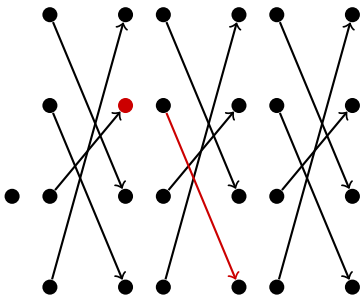
View reachability in terms of relations:



states 1 step away: follow the arcs to states at distance 1 (projection)

Reachability by Breadth-First Traversal: Example

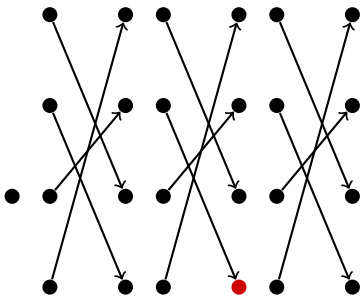
View reachability in terms of relations:



Repeat.

Reachability by Breadth-First Traversal: Example

View reachability in terms of relations:



Repeat.

Image of a Set w.r.t A Relation

To compute $\{s' | s \in S, sRs'\}$, first do $S \bowtie R = \{(s, s') \in R | s \in S\}$:

$$\begin{array}{c} 0 \\ \hline 000 \\ 010 \\ 111 \end{array} \bowtie \begin{array}{cc} 0 & 1 \\ \hline 000 & 011 \\ 001 & 010 \\ 010 & 001 \\ 011 & 000 \end{array} = \begin{array}{cc} 0 & 1 \\ \hline 000 & 011 \\ 010 & 001 \end{array}$$

Then drop the first element in each pair $(s, s') \in S \bowtie R$:

$$\Pi_1 \left(\begin{array}{cc} 0 & 1 \\ \hline 000 & 011 \\ 010 & 001 \end{array} \right) = \begin{array}{c} 1 \\ \hline 011 \\ 001 \end{array}$$

The result is $img_R(S) = \pi_1(S \bowtie R) = \{001, 011\}$.

Sets as Formulas

A formula represents the set of all of its satisfying valuations.

Example

$a \vee b$ over $X = \{a, b, c\}$ is the set $\{010, 011, 100, 101, 110, 111\}$.

a	b	c	$a \vee b$
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Relations as Formulas

A formula is a set of pairs. Valuation 110101 as a pair $(110, 101)$.

Example

Relation $\{(00, 00), (00, 01), (00, 11), (01, 11), (11, 11)\}$ as a formula:

a_0	b_0	a_1	$b1$	$(a_0 \to b_0) \wedge (b_0 \to a_1) \wedge (a_1 \to b_1)$
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Natural Join as Conjunction

0	1
01	10
10	01
11	11

$\bowtie$

1	2
00	01
01	10
10	11

$=$

0	1	2
01	10	11
10	01	10

a_0b_0	a_1b_1	a_2b_2	ϕ_1
00	00	?	0
00	01	?	0
00	10	?	0
00	11	?	0
01	00	?	0
01	01	?	0
01	10	?	1
01	11	?	0
10	00	?	0
10	01	?	1
10	10	?	0
10	11	?	0
11	00	?	0
11	01	?	0
11	10	?	0
11	11	?	1

a_0b_0	a_1b_1	a_2b_2	ϕ_2
?	00	00	0
?	00	01	1
?	00	10	0
?	00	11	0
?	01	00	0
?	01	01	0
?	01	10	1
?	01	11	0
?	10	00	0
?	10	01	0
?	10	10	0
?	10	11	1
?	11	00	0
?	11	01	0
?	11	10	0
?	11	11	0

a_0b_0	a_1b_1	a_2b_2	$\phi_1 \wedge \phi_2$
00	00	00	0
00	00	01	0
00	00	10	0
00	00	11	0
⋮	⋮	⋮	⋮
10	01	10	1
⋮	⋮	⋮	⋮
01	10	11	1
⋮	⋮	⋮	⋮
11	11	00	0
11	11	01	0
11	11	10	0
11	11	11	0

Projection as Existential Abstraction

Make columns a_0, b_0 *don't-care*s:

$\exists a_0 \exists b_0 . (a_1 \leftrightarrow a_0) \wedge (b_1 \leftrightarrow a_0) \equiv (a_1 \leftrightarrow b_1)$

a_0	b_0	a_1	b_1	$(a_1 \leftrightarrow a_0) \wedge (b_1 \leftrightarrow a_0)$
0	0	0	0	1
0	0	**0**	**1**	**0**
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	**0**	**1**	**0**
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	**0**	**1**	**0**
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	**0**	**1**	**0**
1	1	1	0	0
1	1	1	1	1

a_0	b_0	a_1	b_1	$a_1 \leftrightarrow b_1$
?	?	0	0	1
?	?	**0**	**1**	**0**
?	?	1	0	0
?	?	1	1	1

Universal and Existential Abstraction

$\exists$ and $\forall$ abstraction for c:

a	b	c	$a \vee (b \wedge c)$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

a	b	$\exists c.(a \vee (b \wedge c))$
0	0	0
0	1	1
1	0	1
1	1	1

a	b	$\forall c.(a \vee (b \wedge c))$
0	0	0
0	1	0
1	0	1
1	1	1

Image and Pre-Image Operations

$$img_\rho(\phi) = \text{subst}_{X/X'}(\exists X.(\phi \wedge \rho))) \quad (1)$$

$$preimg_\rho(\phi) = (\exists X'.((\text{subst}_{X'/X}(\phi)) \wedge \rho))) \quad (2)$$

These correspond to the relational operations

$$img_R(S) = \{s' | s \in S, sRs'\} \quad (3)$$

$$preimg_R(S) = \{s | s' \in S, sRs'\} \quad (4)$$

Symbolic Breadth-First Search

Compute all states reachable from S_0 with transition relation R:

1. $i := 0$
2. $S_0 :=$ initial states
3. $i := i + 1$
4. $S_i := S_{i-1} \cup \pi_2(S_{i-1} \bowtie R)$
5. if $S_i \not\subseteq S_{i-1}$, go to 3.

Symbolic Breadth-First Search

Compute all states reachable from S_0 with transition relation R:

1. $i := 0$
2. $S_0 :=$ formula describing initial state(s)
3. $i := i + 1$
4. $S_i := S_{i-1} \vee (\text{subst}_{X/X'}(\exists X.(S_{i-1} \wedge R)))$
5. if $S_i \not\models S_{i-1}$, go to 3.

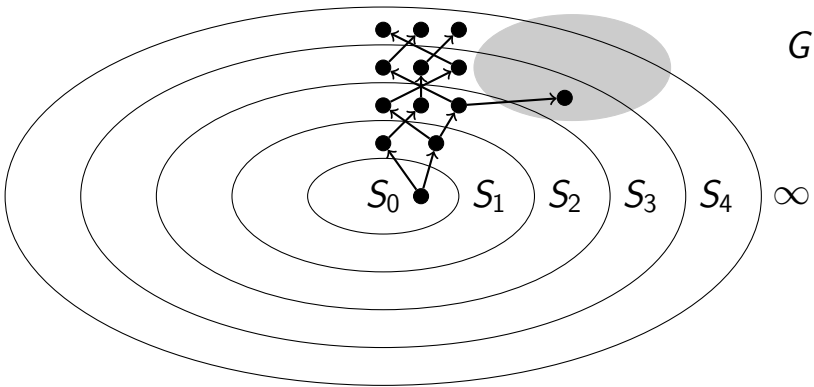
Symbolic Breadth-First Search

Test if some state in G is reachable from S_0 with transition relation R:

1. $i := 0$
2. $S_0 :=$ formula describing initial state(s)
3. $i := i + 1$
4. $S_i := S_{i-1} \vee (\text{subst}_{X/X'}(\exists X.(S_{i-1} \wedge R)))$
5. if $S_i \wedge G$ is not in SAT and $S_i \not\models S_{i-1}$, go to 3.
6. If $S_i \wedge G$ is in SAT then return YES, else return NO.

Distances from $S_0, S_1, \ldots$

Distance of one states $s \in G$ from initial states is 3 because $s \in S_3 \setminus S_2$.



Recovery of the Transition Sequence

When $S_i \wedge G$ is in SAT for some $i \geq 0$: find transition sequence.

1. $B_i := S_i \wedge G$
2. if $i = 0$, terminate.
3. $i := i - 1$
4. for some transition t such that $B_{i+1} \wedge R_t$ is in SAT:
   $B_i := S_i \cap \exists X'.(B_{i+1} \wedge R_t)$
   output the name of transition t
5. go to 2.

This procedure outputs a transition sequence in reverse order.

Representation of Transition Systems in Logic

Example

Move from A to B: $isAtA \wedge \begin{pmatrix} isAtA' \leftrightarrow \bot \\ \wedge isAtB' \leftrightarrow \top \\ \wedge isAtC' \leftrightarrow isAtC \\ \wedge isAtD' \leftrightarrow isAtD \\ \wedge isAtE' \leftrightarrow isAtE \end{pmatrix}$

Deterministic change

For all state variables $x \in X = \{x_1, \ldots, x_n\}$:
$P(x_1, \ldots, x_n) \wedge \bigwedge_{i=1}^{n} x_i' \leftrightarrow F_i(x_1, \ldots, x_n)$

- Change possible if $P(X)$ holds
- $F_i(X)$ is condition for x_i being *true* in the next state.

Representation of Transition Systems in Logic

Choice between deterministic changes

All possible changes: $C_1, C_2, \ldots, C_m$

Transition relation with all these changes: $C_1 \vee C_2 \vee \cdots \vee C_m$

Representation of Transition Systems in Logic

Where do the functions F_i for $x_i' \leftrightarrow F(X)$ come from?

- $F_i(X) = \bot$, if $x_i := 0$
- $F_i(X) = \top$, if $x_i := 1$
- $F_i(X) = x_i$, if x_i does not change

Representation of Transition Systems in Logic

Change described by programs with assignments and conditionals:

- $x_i := F(x_1, \ldots, x_n)$ for $x_i \in X = \{x_1, \ldots, x_n\}$
- eITE(ϕ, e_1, e_2) (for IF-THEN-ELSE)
- $e_1; e_2$

A (deterministic) transition is described by

- a Boolean condition: When is transition possible?
- Program that changes (some) state variables

Weakest Preconditions

Weakest precondition

If $wp_e(\phi)$ holds in a state, then ϕ will hold after executing p.
$wp_e(\phi)$ is the weakest (most general) formula with this property.

$$wp_\epsilon(\phi) = \phi \quad (5)$$

$$wp_{x:=b}(\phi) = \phi \text{ with } x \text{ replaced by expression } b \quad (6)$$

$$wp_{\mathsf{elITE}(\theta,e_1,e_2)}(\phi) = (\theta \wedge wp_{e_1}(\phi)) \vee (\neg\theta \wedge wp_{e_2}(\phi)) \quad (7)$$

$$wp_{e_1;e_2}(\phi) = wp_{e_1}(wp_{e_2}(\phi)) \quad (8)$$

Weakest Preconditions

Example

Let $\phi = a \vee b$ and $e = (a := c)$.

$$wp_e(a \vee b) \quad = \quad c \vee b$$

Weakest Preconditions

Example

Let $\phi = a \vee b$ and $e = \mathsf{eITE}(c, a := 1, c := 1)$.

$$\begin{aligned} wp_e(a \vee b) &= (c \wedge wp_{a:=1}(a \vee b)) \vee (\neg c \wedge wp_{c:=1}(a \vee b)) \\ &= (c \wedge (\top \vee b)) \vee (\neg c \wedge wp_{c:=1}(a \vee b)) \\ &= (c \wedge (\top \vee b)) \vee (\neg c \wedge (a \vee b)) \\ &= c \vee (\neg c \wedge (a \vee b)) \end{aligned}$$

and with $e_2 = \mathsf{eITE}(c, a := 1, b := 1)$,

$$\begin{aligned} wp_{e_2}(a \vee b) &= (c \wedge wp_{a:=1}(a \vee b)) \vee (\neg c \wedge wp_{b:=1}(a \vee b)) \\ &= (c \wedge (\top \vee b)) \vee (\neg c \wedge wp_{c:=1}(a \vee b)) \\ &= (c \wedge (\top \vee b)) \vee (\neg c \wedge (a \vee \top)) \\ &= c \vee \neg c \\ &= \top \end{aligned}$$

Translation of Programs into Logic

Let X be the set of all state variables.

A transition is a pair (p, e) where

- p is a formula over X
- e is a program over X

Corresponding formula is

$$p \wedge \bigwedge_{i=1}^{n} (x_i' \leftrightarrow wp_e(x_i)))$$