

Quantifier elimination

procedure $ex_elim(P, \{n_1, \dots, n_m\})$

parameters: a dag D containing n_1, \dots, n_m and N_0, N_1 as nodes

input: nodes n_1, \dots, n_m representing A_1, \dots, A_m in D

output: a node n representing $\exists P(A_1 \vee \dots \vee A_m)$ in (modified) D

begin

if $m = 0$ then return N_0 ; if $m = 1$ and $P = \emptyset$ then return n_1

if some n_i is a leaf labelled 0 then

return $ex_elim(P, \{n_1, \dots, n_{i-1}, n_{i+1}, \dots, n_m\})$

if some n_i is a leaf labelled 1 then return N_1

$p := max_atom(n_1, \dots, n_m)$

forall $i = 1 \dots m$

if n_i is labelled by p then $(l_i, r_i) := (left(n_i), right(n_i))$

else $(l_i, r_i) := (n_i, n_i)$

if $p \in P$ then return $ex_elim(P - \{p\}, \{l_1, \dots, l_m, r_1, \dots, r_m\})$

else $(k_1, k_2) := (ex_elim(P, \{l_1, \dots, l_m\}), ex_elim(P, \{r_1, \dots, r_m\}))$

if $k_1 = k_2$ then return k_1

return $integrate(k_1, p, k_2, D)$

end

Transition systems and temporal properties

State-changing systems

1. At each particular time moment, the system is in a particular **state**;
2. This state can be characterized by values of some variables, called the **state variables**.
3. There is a notion of an action, and actions may result in a **state change** of the system, that is a change of the value of some variables.

Examples

1. **Reactive systems**. These are the systems which interact with their environment.
2. **Concurrent systems**. These are the systems which consist of a set of components functioning together. Usually, functioning of these components can be described independently, but they communicate through **shared variables** or some kind of **communication channels**, for example queues.

Reasoning about state-changing systems

1. Build a **formal model** of this state-changing system which describes, in particular, functioning of the system, or some abstraction thereof.
2. Using a **logic to specify and verify properties** of the system.

Vending machine example

Consider an example state-changing system: a **vending machine** which dispences drinks in a university department. The machine has several components, including at least the following: a **storage space** for storing and preparing drinks, a **dispencer box** for dispensing drinks and a **slot** for putting coins. When the machine is operating, is comes through several stages depending on the behaviour of the current **customer**.

Vending machine example

Each action undertaken by the customer or by the machine itself may **change the state** of the machine. For example, when the customer inserts a coin in the slot, the amount of money collected in the slot changes.

Actions which may change the state of the system are called **transitions**.

Modelling state-changing systems

To build a formal model of a particular state-changing system, we should define

1. What are the **state variables**.
2. What are the possible **values** of the state variables.
3. What are the **transitions** and how they change the values of the state variables.

A state can be identified with the set of pairs *(variable, value)*, or with a **function** from variables to values.

Transition systems

A **transition system** is a tuple $S = (V, D, dom, I, T)$, where

1. V is a finite set of **state variables**.
2. D is a non-empty set, called the **domain**. Elements of D are called **values**.
3. dom is a mapping from V to the set of non-empty subsets of D .
For each state variable $v \in V$, the set $dom(v)$ is called the **domain for v** .
4. I is a set of states, called **initial states**.
5. T is a set of transitions.

Transition systems

A transition t is **applicable** to a state s if there exists a state s' such that $(s, s') \in t$. The **transition relation of S** , denoted by Tr_S , is the set of pairs of states $\bigcup_{t \in T} t$, i.e., it is the union of all transitions in the system.

A transition system S is called **deterministic** if for every state s there exists at most one state s' such that (s, s') belongs to the transition relation of S , and **non-deterministic** otherwise.

A transition system S is said to be **finite-state** if its domain D is finite, and **infinite-state** otherwise.