

Universal Literal Deletion

Consider a quantifier prefix Q and a conjunction of clauses S .

- ▷ A variable p is **quantified before** a variable q if p occurs **before** q in Q .
- ▷ p is **universal in** Q , if Q contains $\forall p$.
- ▷ p is **existential in** Q , if Q contains $\exists p$.

Example: If Q is $\forall p \exists q \forall r$ then p is quantified before both q and r ; and q is quantified before r .

Theorem. Let C be a clause in S . If a variable p in C is universal in Q and all existential variables in C are quantified before p then we can removing the literal containing p from C does not change change the truth value of QS .

Example

$$\exists p \exists q \forall r \exists s ((p \vee \neg r) \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)).$$

QBF and OBDD

We know how to apply boolean operations to OBDDs. Can we also apply quantification to OBDDs in a straightforward way?

Quantification Problem:

- ▶ Given an OBDD representing a formula F , find an OBDD representing $\exists \forall_1 p_1 \dots \exists \forall_n p_n F$.

We cannot find a straightforward quantification algorithm since any such algorithm would solve the satisfiability-checking problem for quantified boolean formulas.

Quantification for OBDDs

However, we can design a straightforward algorithm when all quantifiers $\exists V_1, \dots, \exists V_n$ are the same.

Lemma.

(1) If $p \neq q$, then

$\exists q (\text{if } p \text{ then } F \text{ else } G) \equiv \text{if } p \text{ then } \exists q F \text{ else } \exists q G$ and

$\forall q (\text{if } p \text{ then } F \text{ else } G) \equiv \text{if } p \text{ then } \forall q F \text{ else } \forall q G.$

(2) $\exists p (\text{if } p \text{ then } F \text{ else } G) \equiv F \vee G$

$\forall p (\text{if } p \text{ then } F \text{ else } G) \equiv F \wedge G.$

Existential quantification algorithm for OBDDs

procedure $ex_elim(\{p_1, \dots, p_k\}, \{n_1, \dots, n_m\})$
parameters: global dag D
input: nodes n_1, \dots, n_m representing F_1, \dots, F_m in D
output: a node n representing $\exists p_1 \dots \exists p_k (F_1 \vee \dots \vee F_m)$ in (modified) D
begin
 if $m = 0$ then return $\boxed{0}$
 if some n_i is $\boxed{1}$ then return $\boxed{1}$
 if some n_i is $\boxed{0}$ then
 return $ex_elim(\{p_1, \dots, p_k\}, \{n_1, \dots, n_{i-1}, n_{i+1}, \dots, n_m\})$
 $p := max_atom(n_1, \dots, n_m)$
 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 $p \in \{p_1, \dots, p_k\}$
 then return $ex_elim(\{p_1, \dots, p_k\} - \{p\}, \{l_1, \dots, l_m, r_1, \dots, r_m\})$
 else
 $k_1 := ex_elim(\{p_1, \dots, p_k\}, \{l_1, \dots, l_m\})$
 $k_2 := ex_elim(\{p_1, \dots, p_k\}, \{r_1, \dots, r_m\})$
 return $integrate(k_1, p, k_2, D)$
end

Example

$$\exists r \forall q \exists p (p \leftrightarrow ((p \rightarrow r) \leftrightarrow q)).$$

Propositional Logic of Finite Domains

- ▶ Similar to propositional logic, but instead of two-valued boolean variables uses variables ranging over a **finite domain** of elements.
- ▶ Motivation: **system modeling**.
- ▶ **Definition and examples**.
- ▶ **Reasoning** using a tableau system.
- ▶ **Propositional logic** as an instance of **PLFD**.
- ▶ Translation of **PLFD** into **propositional logic**.

State-changing systems

Informally	Formally
At each time moment, the system is in a particular state .	This state can be characterized by values of some variables, called the state variables .
The system state is changing in time. There are actions (controlled or not) that change the state.	Actions change values of some state variables.

Examples

1. **Reactive systems**. These are the systems that 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.

Microwave Reasoning

variable	domain of values
selected_button	{ <i>none, micro, grill, defrost</i> }
is_on	{ <i>0, 1</i> }
content	{ <i>none, burger, pizza, cabbage</i> }
user	{ <i>nobody, student, veggie, mcdonald</i> }
state	{ <i>still, cooking, defrost</i> }

Propositional Logic of Finite Domains (PLFD)

PLFD is a **family of logics**. Each instance of PLFD is characterized by

- ▶ a set X of **variables**;
- ▶ a mapping dom , such that for every $x \in X$, $dom(x)$ is a non-empty finite set, called the **domain for x** .

Syntax of PLFD

Formulas

- ▶ If x is a variable and $v \in \text{dom}(x)$ is a value in the domain of x , then $x = v$ is a formula, also called **atomic formula**, or simply **atom**.
- ▶ Other formulas are built from atomic formulas **as in propositional logic**, using the connectives \top , \perp , \wedge , \vee , \neg , \rightarrow , and \leftrightarrow .

Semantics

- ▶ **Interpretation** for a set of variables X is a mapping I from X to the set of values such that for all $x \in X$ we have $I(x) \in \text{dom}(x)$.
- ▶ Extend interpretations to mappings from formulas to boolean values.
 1. $I(x = v) = 1$ if and only if $I(x) = v$.
 2. If A is not atomic, then as for propositional formulas.
- ▶ The definitions of **truth**, **models**, **validity**, **satisfiability**, and **equivalence** are defined exactly as in propositional logic.

Example

Let a variable x range over the domain $\{a, b, c\}$, that is $dom(x) = \{a, b, c\}$. Then the following formula is valid:

$$\neg x = a \rightarrow x = b \vee x = c.$$

Propositional Logic as PLFD

The domain for each variable is $\{0, 1\}$.

Instead of atoms p use $p = 1$.

In general, when p is a boolean variable, that is, $dom(p) = \{0, 1\}$, in PLFD we will write p instead of $p = 1$.