Efficient Equality Checking for Non-Shared Binary Decision Diagrams

Steffan Christ Sølvsten, Jaco van de Pol

6th of September, 2021

AARHUS UNIVERSITY

$$f \leftrightarrow g \equiv \top$$

$$f \leftrightarrow g \equiv \top$$

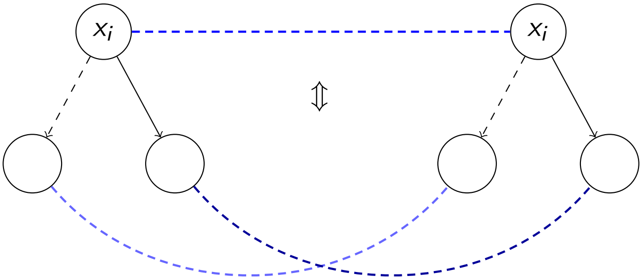
$$\underbrace{O(\mathsf{sort}(N^2))}_{\texttt{Apply}} + \underbrace{O(\mathsf{sort}(N^2))}_{\texttt{Reduce}} + \underbrace{O(1))}_{\text{check is } \top} = O(\mathsf{sort}(N^2))$$

Theorem (Bryant '86)
Let π *be a variable order and* $f : \mathbb{B}^n \to \mathbb{B}$ *then there exists a unique (up to isomorphism) Reduced Ordered Binary Decision Diagram representing* f *with ordering* π.

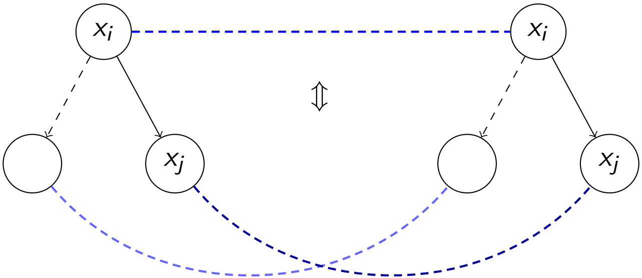
Theorem (Bryant '86)
Let π be a variable order and $f : \mathbb{B}^n \to \mathbb{B}$ then there exists a unique (up to isomorphism) Reduced Ordered Binary Decision Diagram representing f with ordering π.

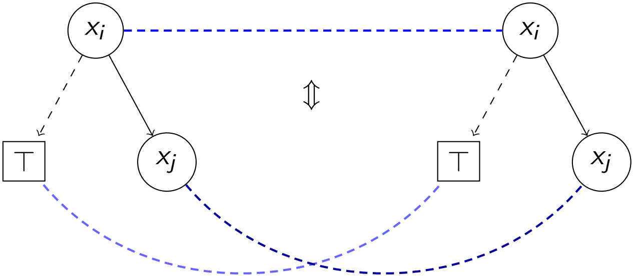
Trivial cases: $f \not\equiv g$ if there is a mismatch in

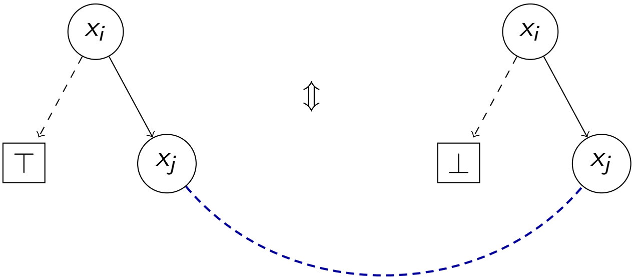
- $N_f \neq N_g$ Number of nodes $O(1)$ I/Os
- $L_f \neq L_g$ Number of levels $O(1)$ I/Os
- $N_{f,i} \neq N_{g,i}$ Number of nodes on a level $O(L/B)$ I/Os
- $L_{f,i} \neq L_{g,i}$ Label of an ith level $O(L/B)$ I/Os

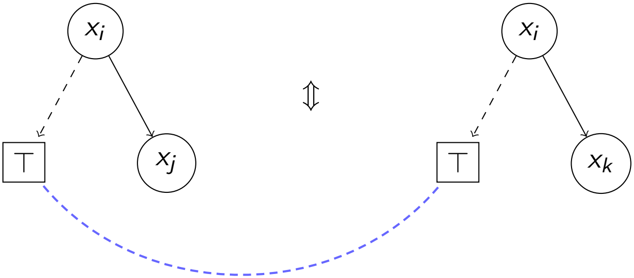


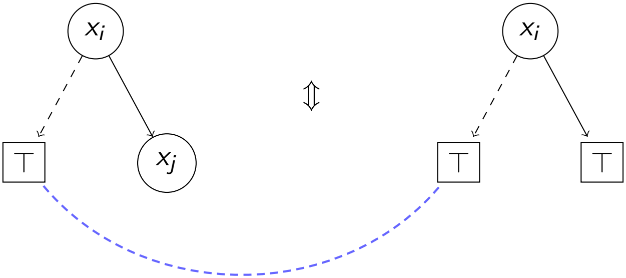
x_i
$\Updownarrow$
x_i


x_i
x_j
$\Updownarrow$
x_i
x_j


x_i
x_j
$\top$
$\Updownarrow$
x_i
$\top$
x_j


x_i
$\top$
x_j
$\Updownarrow$
x_i
$\perp$
x_j


x_i
$\top$
x_j
$\Updownarrow$
x_i
$\top$
x_k


x_i
x_j
$\top$
$\Updownarrow$
x_i
$\top$
$\top$

`IsIsomorphic(`f`, `g`)`

- Check whether root v_f of f and root v_g of g have a local violation.
- Check $low(v_f) \sim low(v_g)$ and $high(v_f) \sim high(v_g)$ “recursively”.

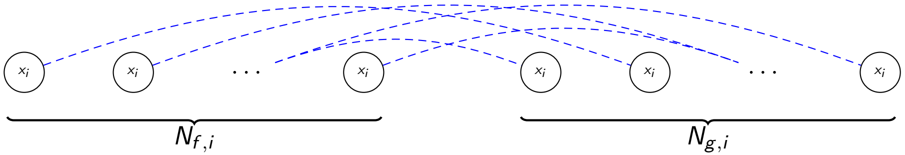
Return `false` on first violation. If there are no violations then return `true`.

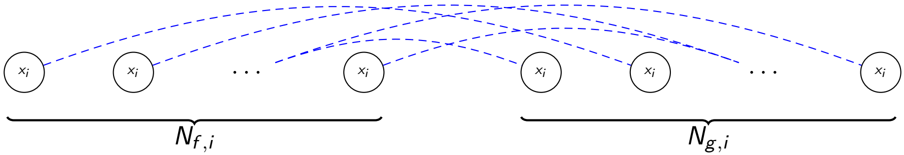
`IsIsomorphic(`f`, `g`)`

- Check whether root v_f of f and root v_g of g have a local violation.
- Check $low(v_f) \sim low(v_g)$ and $high(v_f) \sim high(v_g)$ "recursively".

Return `false` on first violation. If there are no violations then return `true`.

$$\underbrace{O(\mathrm{sort}(N^2))}_{\texttt{Apply}'} + \underbrace{\cancel{O(\mathrm{sort}(N^2))}}_{\texttt{Reduce}} + \underbrace{\cancel{O(1)}}_{\text{check is } \top} = O(\mathrm{sort}(N^2))$$


x_i
x_i
. . .
x_i
x_i
x_i
. . .
x_i
$N_{f,i}$
$N_{g,i}$



Return `false` if more than $N_{f,i} = N_{g,i}$ pairs of nodes (v_f, v_g) are checked on level i.

$$O(\text{sort}(N))$$

Observation
The output of `Reduce` has the following properties

- Nodes on level i are output sorted by their children

$$((i_1, id_1), low_1, high_1) <_{lex(i,low,high)} ((i_2, id_2), low_2, high_2) \ ,$$

  where

$$\forall (i, id) \ : \ (i, id) < \bot < \top \ ^1.$$

- Nodes on level i have their identifiers *consecutively* numbered

$$MAX - N_{f,i} + 1, \ldots, MAX - 1, MAX \ .$$

[1] Assuming the BDD is not negated. If that is the case then $(i, id) < \top < \bot$.

Theorem
If G_f and G_g are outputs of `Reduce`.

$$G_f \sim G_g \iff \textit{For all } i \in [0; N_f) \textit{ the node } G_f[i] \textit{ matches } G_g[i] \textit{ numerically.}$$

Proof.
$\Leftarrow$: Must describe the exact same graph.

$\Rightarrow$: Strong induction on BDD levels bottom-up ... $\square$

Theorem
If G_f and G_g are outputs of $\mathtt{Reduce}$.

$$G_f \sim G_g \iff \text{For all } i \in [0; N_f) \text{ the node } G_f[i] \text{ matches } G_g[i] \text{ numerically.}$$

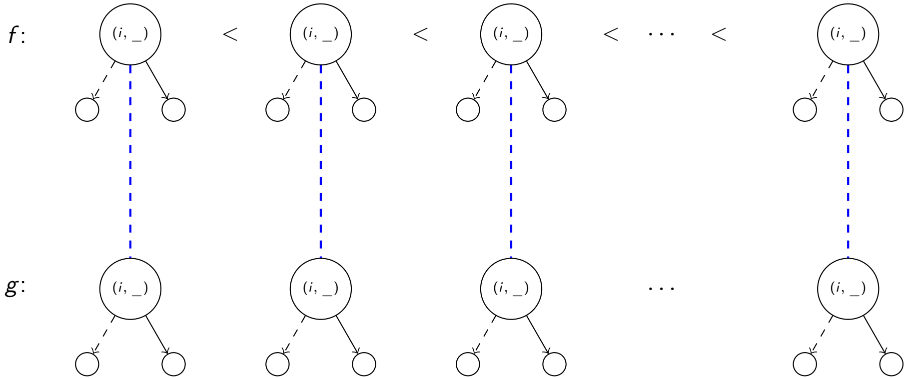
Proof.
$\Leftarrow$: Must describe the exact same graph.

$\Rightarrow$: Strong induction on BDD levels bottom-up ... $\square$

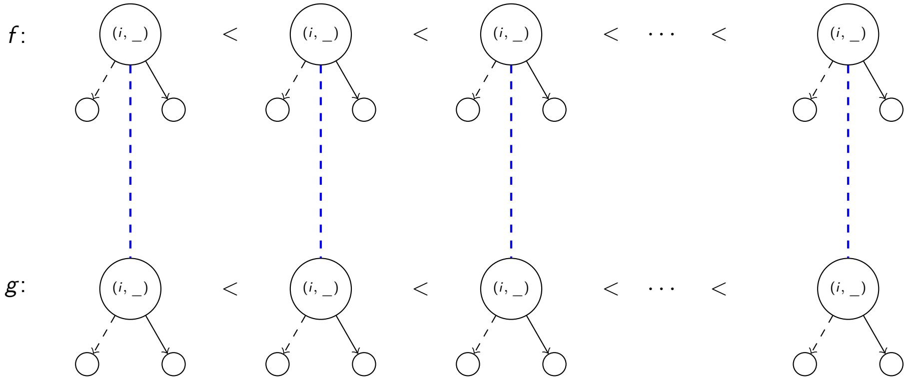
Corollary
If G_f and G_g are outputs of $\mathtt{Reduce}$ *then $f \equiv g$ is computable using $2 \cdot N/B$ I/Os.*[2]

[2]Assuming they are both unnegated (or both negated).

Base case: n



Induction case: $i < n$



Induction case: $i < n$

Algorithm	Time (s)
$f \leftrightarrow g \equiv \top$	0.38
$O(\text{sort}(N))$	0.058
$2 \cdot N/B$	0.006

Checking the (EPFL Benchmark) *voter* circuit's single output gate ($|N_f| = |N_g| = 5.76$ MiB).