I/O-efficient Symbolic Model Checking

Steffan Christ Sølvsten, Jaco van de Pol

31st of August, 2022

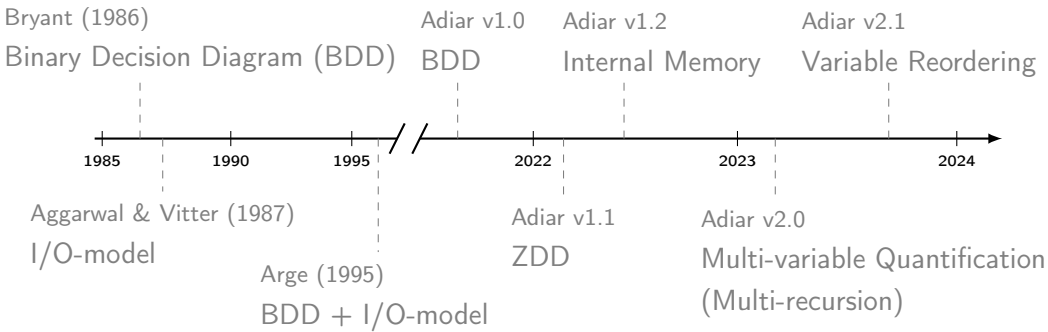
AARHUS
UNIVERSITY

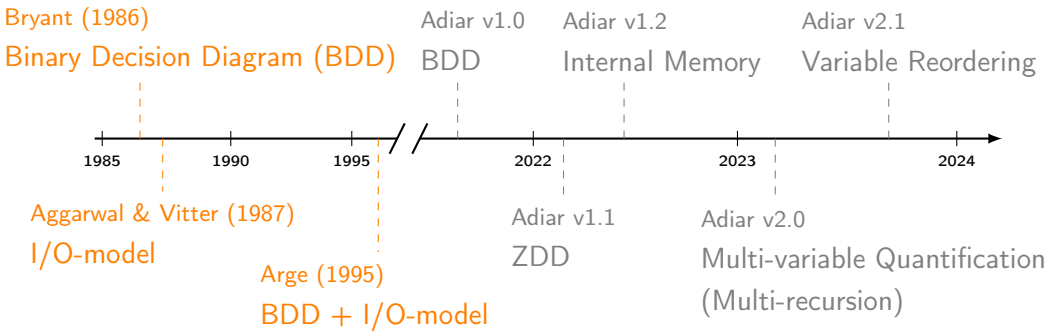
Adiar

I/O-efficient Decision Diagrams

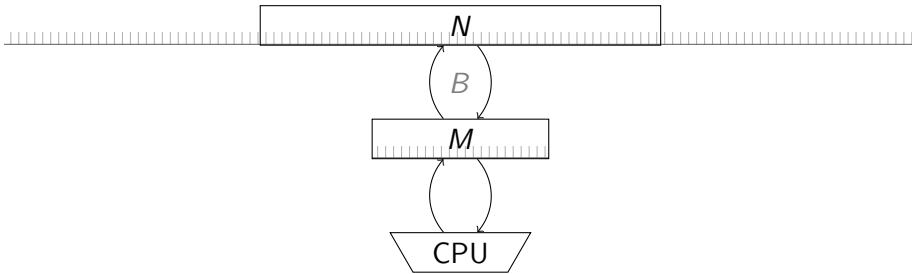
---

github.com/ssoelvsten/adiar


Bryant (1986)
Binary Decision Diagram (BDD)
Adiar v1.0
BDD
Adiar v1.2
Internal Memory
Adiar v2.1
Variable Reordering
1985
1990
1995
2022
2023
2024
Aggarwal & Vitter (1987)
I/O-model
Arge (1995)
BDD + I/O-model
Adiar v1.1
ZDD
Adiar v2.0
Multi-variable Quantification
(Multi-recursion)


Bryant (1986)
Binary Decision Diagram (BDD)
Adiar v1.0
BDD
Adiar v1.2
Internal Memory
Adiar v2.1
Variable Reordering
1985
1990
1995
2022
2023
2024
Aggarwal & Vitter (1987)
I/O-model
Arge (1995)
BDD + I/O-model
Adiar v1.1
ZDD
Adiar v2.0
Multi-variable Quantification
(Multi-recursion)

Aggarwal and Vitter '87 : I/O-model



The I/O-model by Aggarwal and Vitter '87

Aggarwal and Vitter '87 : I/O-model

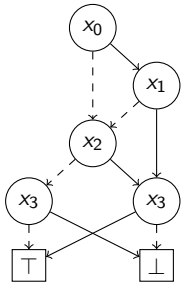
For any realistic values of N, M, and B we have that

$$N/B \quad < \quad \text{sort}(N) \triangleq N/B \cdot \log_{M/B} N/B \quad \ll \quad N \ ,$$

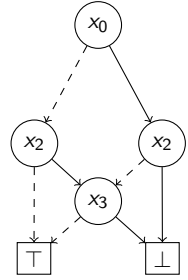
Theorem (Aggarwal and Vitter '87)
N elements can be sorted in $\Theta(sort(N))$ I/Os.

Theorem (Arge '95)
N elements can be inserted in and extracted from a Priority Queue in $\Theta(sort(N))$ I/Os.

Bryant '86 : Binary Decision Diagram



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$

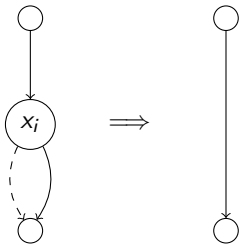


(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

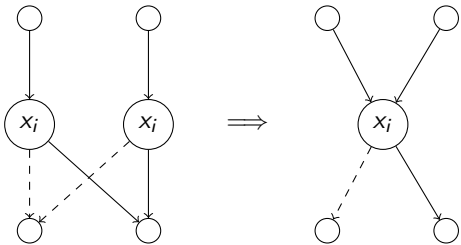
Examples of (Reduced Ordered) Binary Decision Diagrams.

Bryant '86 : Binary Decision Diagram

Theorem
For a fixed variable order, if one exhaustively applies the two rules below, then one obtains the Reduced OBDD, which is a unique canonical form of the function.

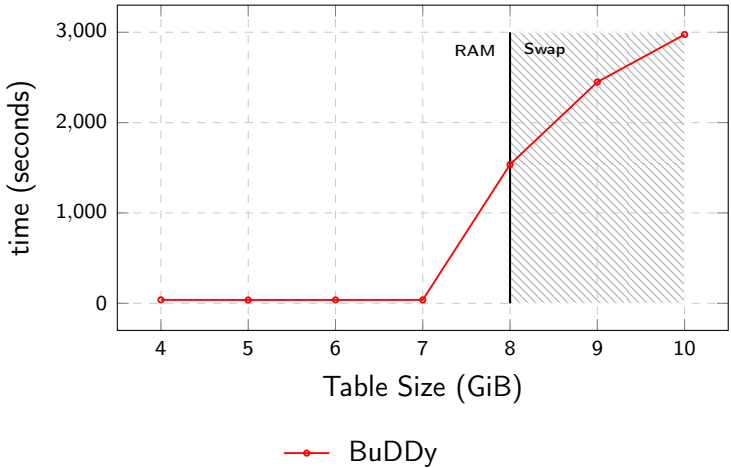


(1) Remove redundant nodes

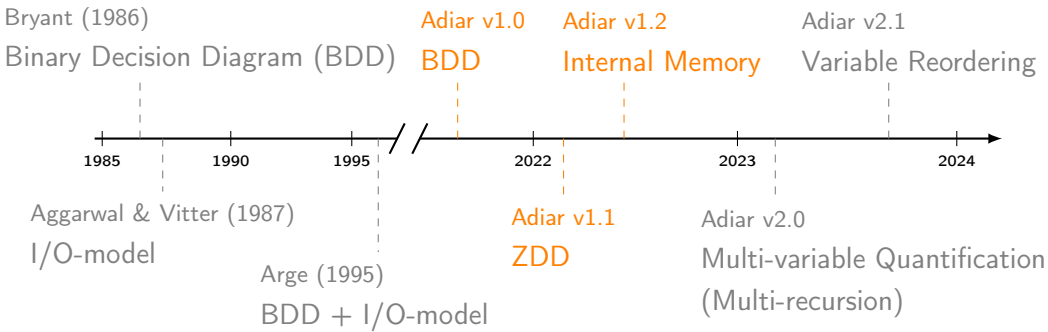


(2) Merge duplicate nodes

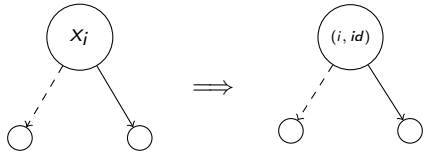
Arge '95 : BDD + I/O-model



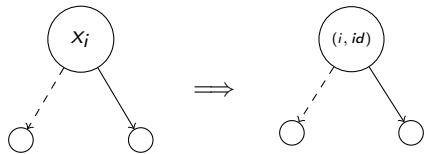
Running time for solving a problem that does not need more than 3 GiB.


Bryant (1986)
Binary Decision Diagram (BDD)
Adiar v1.0
BDD
Adiar v1.2
Internal Memory
Adiar v2.1
Variable Reordering
1985
1990
1995
2022
2023
2024
Aggarwal & Vitter (1987)
I/O-model
Arge (1995)
BDD + I/O-model
Adiar v1.1
ZDD
Adiar v2.0
Multi-variable Quantification
(Multi-recursion)

Adiar v1.0 : BDD

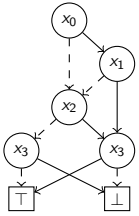


Adiar v1.0 : BDD



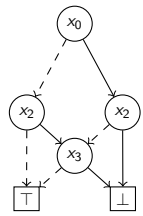
$$(i_1, id_1) < (i_2, id_2) \equiv i_1 < i_2 \vee (i_1 = i_2 \wedge id_i < id_j)$$

Adiar v1.0 : BDD



$$
\begin{bmatrix}
((0,0),(2,0),(1,0)) & , \\
((1,0),(2,0),(3,1)) & , \\
((2,0),(3,0),(3,1)) & , \\
((3,0),\bot,\top) & , \\
((3,1),\top,\bot) &]
\end{bmatrix}
$$

(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$

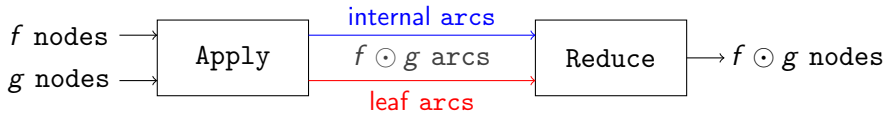


$$
\begin{bmatrix}
((0,0),(2,0),(2,1)) & , \\
((2,0),\top,(3,0)) & , \\
((2,1),(3,0),\bot) & , \\
((3,0),\top,\bot) &]
\end{bmatrix}
$$

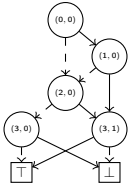
(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Node-based representation of prior shown BDDs

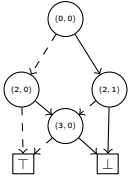
Adiar v1.0 : BDD



Adiar v1.0 : BDD



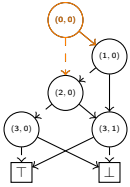
(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



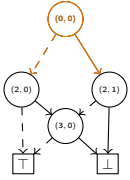
(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



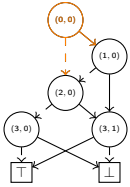
(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



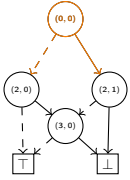
(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \ ? \ x_2 \vee x_3 \ : \ x_2 \wedge x_3)$

Priority Queue: $Q_{\mathbf{app}:1}$:

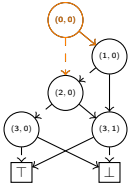
[$(0,0) \xrightarrow{\top} ((1,0),(2,1))$,

$(0,0) \xrightarrow{\bot} ((2,0),(2,0))$,

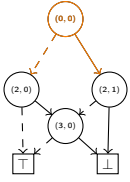
]

(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\min((1,0),(2,1))$

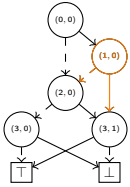
Priority Queue: $Q_{\mathbf{app}:\mathbf{1}}$:

$[\quad (0,0) \xrightarrow{\top} ((1,0),(2,1)) \quad ,$
$\quad\;\; (0,0) \xrightarrow{\bot} ((2,0),(2,0)) \quad ,$

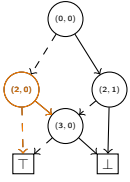
$]$

(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\min((1, 0), (2, 1))$

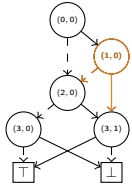
Priority Queue: $Q_{\mathbf{app}:\mathbf{1}}$:

[$(0, 0) \xrightarrow{\top} ((1, 0), (2, 1))$,
$(0, 0) \xrightarrow{\bot} ((2, 0), (2, 0))$,

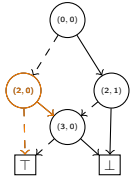
]

(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



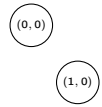
(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\min((1,0),(2,1))$

Priority Queue: $Q_{\mathbf{app}:\mathbf{1}}$:

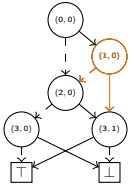
$$\left[\begin{array}{ll} (0,0) \xrightarrow{\top} ((1,0),(2,1)) & , \\ (0,0) \xrightarrow{\perp} ((2,0),(2,0)) & , \\ (1,0) \xrightarrow{\perp} ((2,0),(2,1)) & , \\ (1,0) \xrightarrow{\top} ((3,1),(2,1)) & , \end{array} \right.$$

]

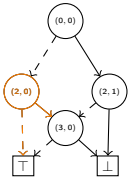


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\min((1,0),(2,1))$

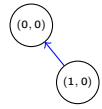
Priority Queue: $Q_{\textbf{app}:1}$:

[

$(0,0) \xrightarrow{\perp} ((2,0),(2,0))$,
$(1,0) \xrightarrow{\perp} ((2,0),(2,1))$,
$(1,0) \xrightarrow{\top} ((3,1),(2,1))$,

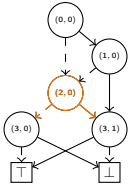
]

Output:
$(0,0) \xrightarrow{\top} (1,0)$

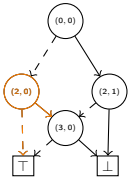


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:

$\min((2,0),(2,0))$

Priority Queue: $Q_{\mathbf{app}:\mathbf{1}}$**:**

[

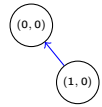
$(0,0) \xrightarrow{\perp} ((2,0),(2,0))$,

$(1,0) \xrightarrow{\perp} ((2,0),(2,1))$,

$(1,0) \xrightarrow{\top} ((3,1),(2,1))$,

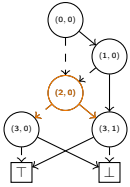
]

Output:

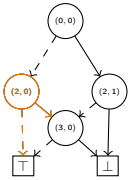


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\min((2,0),(2,0))$

Priority Queue: $Q_{app:1}$:

[

$(0,0) \xrightarrow{\perp} ((2,0),(2,0))$,

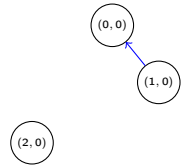
$(1,0) \xrightarrow{\perp} ((2,0),(2,1))$,

$(1,0) \xrightarrow{\top} ((3,1),(2,1))$,

$(2,0) \xrightarrow{\top} ((3,1),(3,0))$,

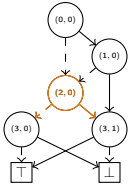
$(2,0) \xrightarrow{\perp} ((3,0),\top)$]

Output:

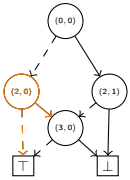


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\min((2,0),(2,0))$

Priority Queue: $Q_{\mathit{app}:1}$:

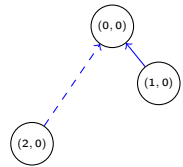
[

$(1,0) \xrightarrow{\perp} ((2,0),(2,1))$,
$(1,0) \xrightarrow{\top} ((3,1),(2,1))$,

$(2,0) \xrightarrow{\top} ((3,1),(3,0))$,

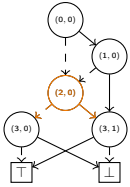
$(2,0) \xrightarrow{\perp} ((3,0),\top)$]

Output:
$(0,0) \xrightarrow{\perp} (2,0)$

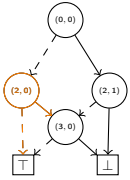


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:

$\min((2, 0), (2, 1))$

Priority Queue: $Q_{\mathbf{app}:\mathbf{1}}$:

[

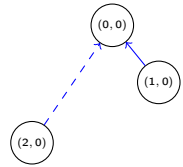
$(1, 0) \xrightarrow{\perp} ((2, 0), (2, 1))$,

$(1, 0) \xrightarrow{\top} ((3, 1), (2, 1))$,

$(2, 0) \xrightarrow{\top} ((3, 1), (3, 0))$,

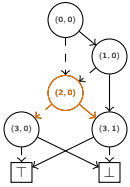
$(2, 0) \xrightarrow{\perp} ((3, 0), \top)$]

Output:

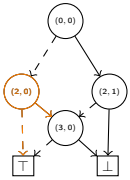


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:

$\min((2, 0), (2, 1))$

Priority Queue: $Q_{app:1}$:

[

$(1, 0) \xrightarrow{\top} ((3, 1), (2, 1))$,

$(2, 0) \xrightarrow{\top} ((3, 1), (3, 0))$,

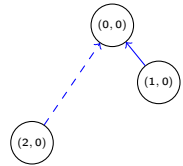
$(2, 0) \xrightarrow{\bot} ((3, 0), \top)$]

Priority Queue: $Q_{app:2}$:

[$(1, 0) \xrightarrow{\bot} ((2, 0), (2, 1)) \quad ((3, 0), (3, 1))$,

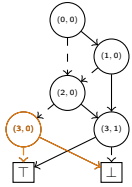
]

Output:

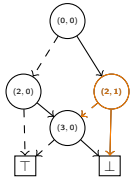


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\max((2,0),(2,1))$

Priority Queue: $Q_{app:1}$:

[

$(1,0) \xrightarrow{\top} ((3,1),(2,1))$,

$(2,0) \xrightarrow{\top} ((3,1),(3,0))$,

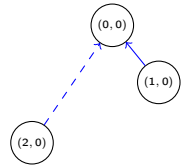
$(2,0) \xrightarrow{\bot} ((3,0),\top)$]

Priority Queue: $Q_{app:2}$:

[$(1,0) \xrightarrow{\bot} ((2,0),(2,1)) \quad ((3,0),(3,1))$,

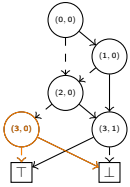
]

Output:

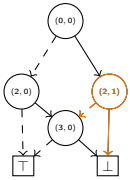


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\max((2,0),(2,1))$

Priority Queue: $Q_{app:1}$:

[

$(1,0) \xrightarrow{\top} ((3,1),(2,1))$,
$(2,1) \xrightarrow{\bot} ((3,0),(3,0))$,
$(2,0) \xrightarrow{\top} ((3,1),(3,0))$,

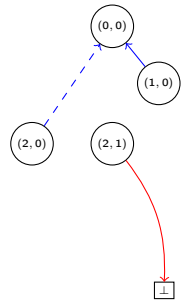
$(2,0) \xrightarrow{\bot} ((3,0),\top)$]

Priority Queue: $Q_{app:2}$:

[$(1,0) \xrightarrow{\bot} ((2,0),(2,1))$ $((3,0),(3,1))$,

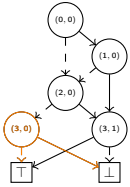
]

Output:
$(2,1) \xrightarrow{\top} \bot$

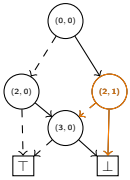


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\max((2,0),(2,1))$

Priority Queue: $Q_{app:1}$:

[

$(1,0) \xrightarrow{\top} ((3,1),(2,1))$,
$(2,1) \xrightarrow{\bot} ((3,0),(3,0))$,
$(2,0) \xrightarrow{\top} ((3,1),(3,0))$,

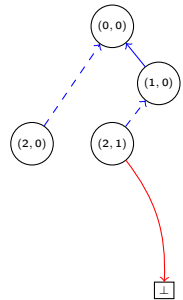
$(2,0) \xrightarrow{\bot} ((3,0),\top)$]

Priority Queue: $Q_{app:2}$:

[

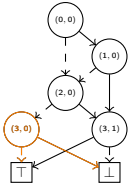
]

Output:
$(1,0) \xrightarrow{\bot} (2,1)$

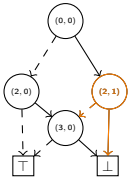


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\min((3, 1), (2, 1))$

Priority Queue: $Q_{app:1}$:

[

$(1, 0) \xrightarrow{\top} ((3, 1), (2, 1))$,
$(2, 1) \xrightarrow{\bot} ((3, 0), (3, 0))$,
$(2, 0) \xrightarrow{\top} ((3, 1), (3, 0))$,

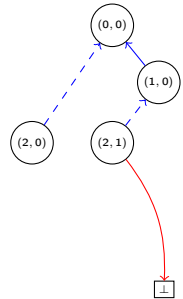
$(2, 0) \xrightarrow{\bot} ((3, 0), \top)$]

Priority Queue: $Q_{app:2}$:

[

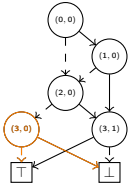
]

Output:

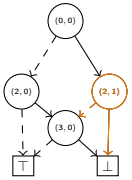


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\min((3, 1), (2, 1))$

Priority Queue: $Q_{app:1}$:

[

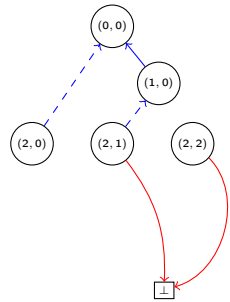
$(1, 0) \xrightarrow{\top} ((3, 1), (2, 1))$,
$(2, 1) \xrightarrow{\bot} ((3, 0), (3, 0))$,
$(2, 0) \xrightarrow{\top} ((3, 1), (3, 0))$,
$(2, 2) \xrightarrow{\bot} ((3, 1), (3, 0))$,
$(2, 0) \xrightarrow{\bot} ((3, 0), \top)$]

Priority Queue: $Q_{app:2}$:

[

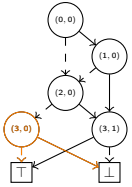
]

Output:
$(2, 2) \xrightarrow{\top} \bot$

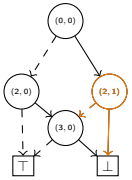


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\min((3,1),(2,1))$

Priority Queue: $Q_{app:1}$:

[

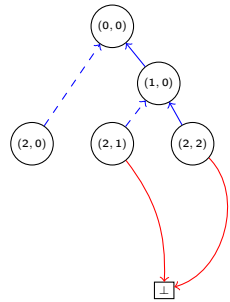
$(2,1) \xrightarrow{\bot} ((3,0),(3,0))$,
$(2,0) \xrightarrow{\top} ((3,1),(3,0))$,
$(2,2) \xrightarrow{\bot} ((3,1),(3,0))$,
$(2,0) \xrightarrow{\bot} ((3,0),\top)$]

Priority Queue: $Q_{app:2}$:

[

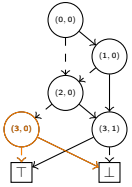
]

Output:
$(1,0) \xrightarrow{\top} (2,2)$

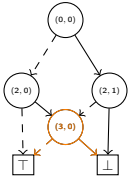


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\min((3,0),(3,0))$

Priority Queue: $Q_{app:1}$:

[

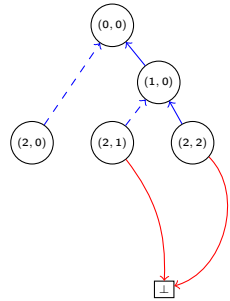
$(2,1) \xrightarrow{\perp} ((3,0),(3,0))$,
$(2,0) \xrightarrow{\top} ((3,1),(3,0))$,
$(2,2) \xrightarrow{\perp} ((3,1),(3,0))$,
$(2,0) \xrightarrow{\perp} ((3,0),\top)$]

Priority Queue: $Q_{app:2}$:

[

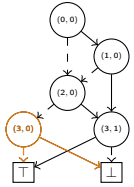
]

Output:

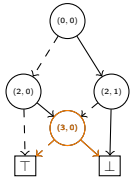


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\min((3,0),(3,0))$

Priority Queue: $Q_{app:1}$:

[

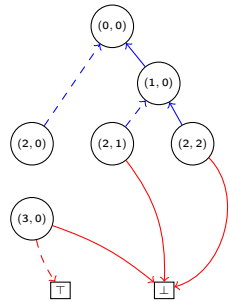
$(2,1) \xrightarrow{\perp} ((3,0),(3,0))$,
$(2,0) \xrightarrow{\top} ((3,1),(3,0))$,
$(2,2) \xrightarrow{\perp} ((3,1),(3,0))$,
$(2,0) \xrightarrow{\perp} ((3,0),\top)$]

Priority Queue: $Q_{app:2}$:

[

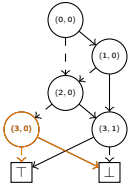
]

Output:
$(3,0) \xrightarrow{\perp} \top, (3,0) \xrightarrow{\top} \perp$

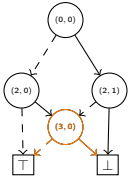


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\min((3,0),(3,0))$

Priority Queue: $Q_{app:1}$:

[

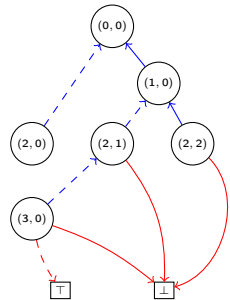
$(2,0) \xrightarrow{\top} ((3,1),(3,0))$,
$(2,2) \xrightarrow{\bot} ((3,1),(3,0))$,
$(2,0) \xrightarrow{\bot} ((3,0),\top)$]

Priority Queue: $Q_{app:2}$:

[

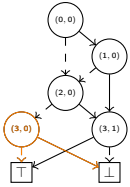
]

Output:
$(2,1) \xrightarrow{\bot} (3,0)$

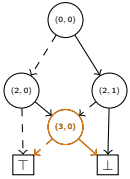


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\min((3, 1), (3, 0))$

Priority Queue: $Q_{app:1}$:

[

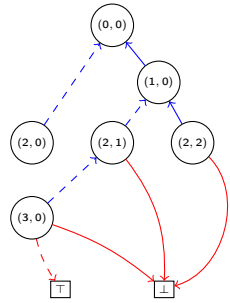
$(2, 0) \xrightarrow{\top} ((3, 1), (3, 0))$,
$(2, 2) \xrightarrow{\bot} ((3, 1), (3, 0))$,
$(2, 0) \xrightarrow{\bot} ((3, 0), \top)$]

Priority Queue: $Q_{app:2}$:

[

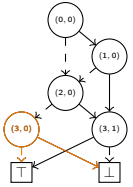
]

Output:

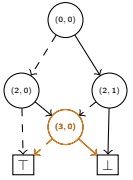


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\min((3, 1), (3, 0))$

Priority Queue: $Q_{app:1}$:

[

$(2, 0) \xrightarrow{\perp} ((3, 0), \top)$]

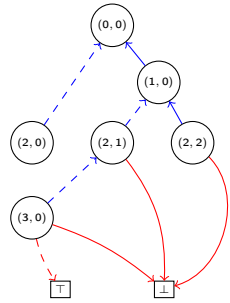
Priority Queue: $Q_{app:2}$:

[

$(2, 0) \xrightarrow{\top} ((3, 1), (3, 0)) \quad (\top, \perp)$,

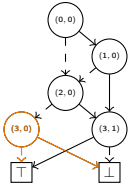
$(2, 2) \xrightarrow{\perp} ((3, 1), (3, 0)) \quad (\top, \perp)$]

Output:

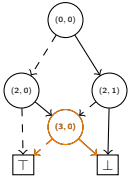


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:

$\min((3, 0), \top)$

Priority Queue: $Q_{app:1}$:

[

$(2, 0) \xrightarrow{\perp} ((3, 0), \top)$]

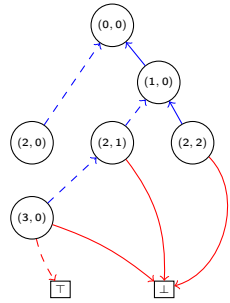
Priority Queue: $Q_{app:2}$:

[

$(2, 0) \xrightarrow{\top} ((3, 1), (3, 0)) \quad (\top, \perp)$,

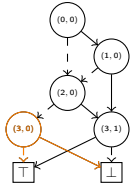
$(2, 2) \xrightarrow{\perp} ((3, 1), (3, 0)) \quad (\top, \perp)$]

Output:

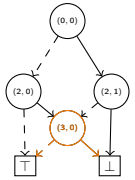


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\min((3,0), \top)$

Priority Queue: $Q_{app:1}$:

[

$(2,0) \xrightarrow{\perp} ((3,0), \top)$]

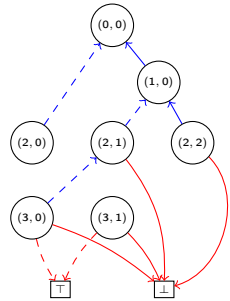
Priority Queue: $Q_{app:2}$:

[

$(2,0) \xrightarrow{\top} ((3,1),(3,0)) \quad (\top, \perp)$,

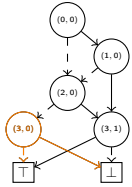
$(2,2) \xrightarrow{\perp} ((3,1),(3,0)) \quad (\top, \perp)$]

Output:
$(3,1) \xrightarrow{\perp} \top, (3,1) \xrightarrow{\top} \perp$

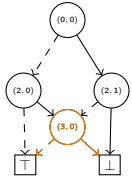


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\min((3, 0), \top)$

Priority Queue: $Q_{app:1}$:

[

]

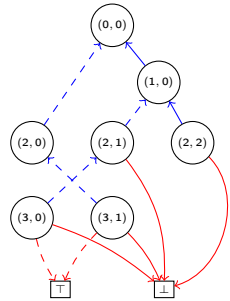
Priority Queue: $Q_{app:2}$:

[

$(2, 0) \xrightarrow{\top} ((3, 1), (3, 0)) \quad (\top, \bot)$,

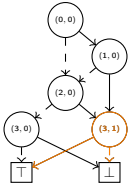
$(2, 2) \xrightarrow{\bot} ((3, 1), (3, 0)) \quad (\top, \bot)$]

Output:
$(2, 0) \xrightarrow{\bot} (3, 1)$

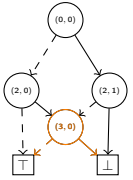


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \ ? \ x_2 \vee x_3 \ : \ x_2 \wedge x_3)$

Seek:

$\max((3, 1), (3, 0))$

Priority Queue: $Q_{app:1}$:

[

]

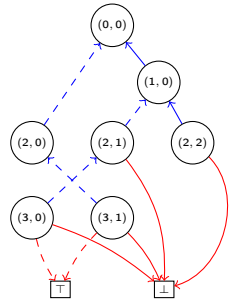
Priority Queue: $Q_{app:2}$:

[

$(2, 0) \xrightarrow{\top} ((3, 1), (3, 0)) \quad (\top, \bot)$,

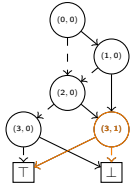
$(2, 2) \xrightarrow{\bot} ((3, 1), (3, 0)) \quad (\top, \bot)$]

Output:

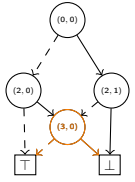


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\max((3,1),(3,0))$

Priority Queue: $Q_{app:1}$:

[

]

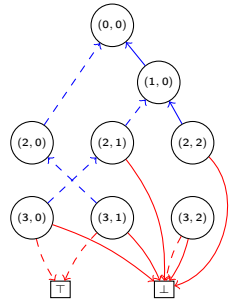
Priority Queue: $Q_{app:2}$:

[

$(2,0) \xrightarrow{\top} ((3,1),(3,0)) \quad (\top, \bot)$,

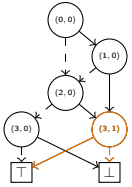
$(2,2) \xrightarrow{\bot} ((3,1),(3,0)) \quad (\top, \bot)$]

Output:
$(3,2) \xrightarrow{\bot} \bot, (3,2) \xrightarrow{\top} \bot$

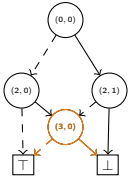


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Seek:
$\max((3, 1), (3, 0))$

Priority Queue: $Q_{app:1}$:

[

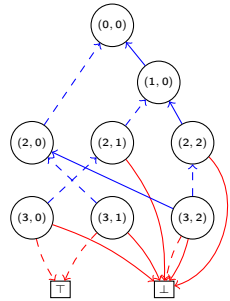
]

Priority Queue: $Q_{app:2}$:

[

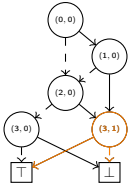
]

Output:
$(2, 0) \xrightarrow{\top} (3, 2), (2, 2) \xrightarrow{\bot} (3, 2)$

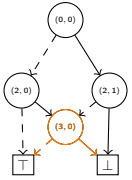


(c) $(a) \wedge (b)$

Adiar v1.0 : BDD



(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$



(b) $\neg(x_0 \; ? \; x_2 \vee x_3 \; : \; x_2 \wedge x_3)$

Priority Queue: $Q_{app:1}$:

[

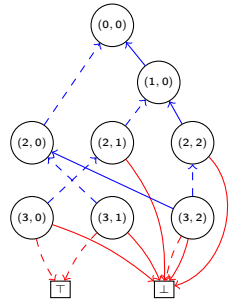
]

Priority Queue: $Q_{app:2}$:

[

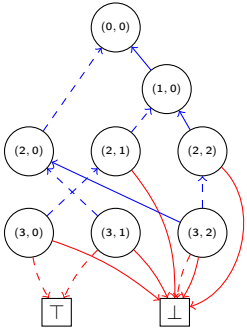
]

Output:



(c) $(a) \wedge (b)$

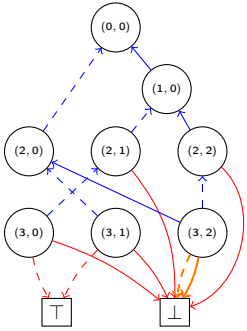
Adiar v1.0 : BDD



(c) $(a) \wedge (b)$

(d) $(a) \wedge (b)$ **reduced**

Adiar v1.0 : BDD



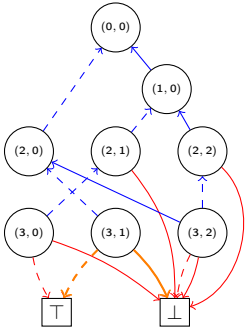
(c) $(a) \wedge (b)$

Level: 3

[$[(3, 2) \mapsto \bot]$]

(d) $(a) \wedge (b)$ **reduced**

Adiar v1.0 : BDD



(c) $(a) \wedge (b)$

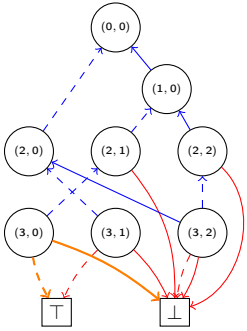
Level: 3

[[(3, 2) ↦ ⊥]]

[((3, 1), ⊤, ⊥) ,]

(d) $(a) \wedge (b)$ reduced

Adiar v1.0 : BDD



(c) $(a) \wedge (b)$

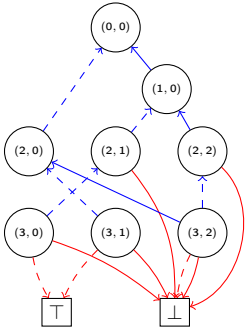
Level: 3

[$[(3, 2) \mapsto \bot]$]

[$((3, 1), \top, \bot)$,
$((3, 0), \top, \bot)$]

(d) $(a) \wedge (b)$ **reduced**

Adiar v1.0 : BDD



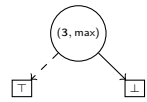
(c) $(a) \wedge (b)$

Level: 3

[$[(3, 2) \mapsto \bot]$]

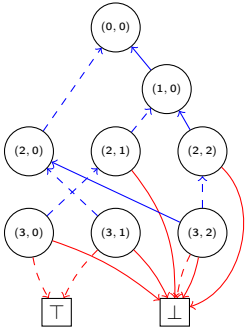
[$[(3, 1) \mapsto (3, \max)]$, $((3, 0), \top, \bot)$]

Output:
$((3, \max), \top, \bot)$



(d) $(a) \wedge (b)$ **reduced**

Adiar v1.0 : BDD



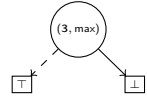
(c) $(a) \wedge (b)$

Level: 3

[$[(3,2) \mapsto \bot]$]

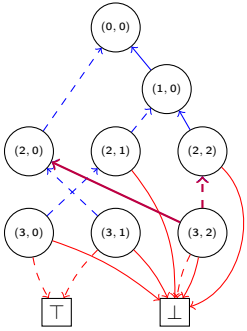
[$[(3,1) \mapsto (3, \max)]$,
$[(3,0) \mapsto (3, \max)]$]

Output:



(d) $(a) \wedge (b)$ reduced

Adiar v1.0 : BDD



(c) $(a) \wedge (b)$

Priority Queue: Q_{red}:

[$(2, 2) \xrightarrow{\perp} \perp$,

$(2, 0) \xrightarrow{\top} \perp$,

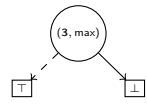
]

Level: 3

[ ]

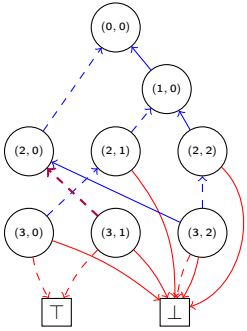
[$[(3, 1) \mapsto (3, \max)]$,
$[(3, 0) \mapsto (3, \max)]$]

Output:



(d) $(a) \wedge (b)$ reduced

Adiar v1.0 : BDD



(c) $(a) \wedge (b)$

Priority Queue: Q_{red}:

[$(2, 2) \xrightarrow{\bot} \bot$,

$(2, 0) \xrightarrow{\top} \bot$,

$(2, 0) \xrightarrow{\bot} (3, \max)$,

]

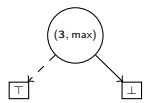
Level: 3

[ ]

[,

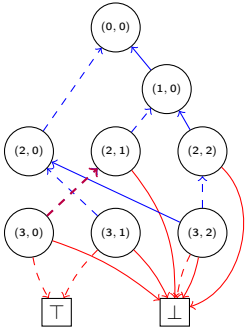
$[(3, 0) \mapsto (3, \max)]$]

Output:



(d) $(a) \wedge (b)$ reduced

Adiar v1.0 : BDD



(c) $(a) \wedge (b)$

Priority Queue: Q_{red}:

[$(2, 2) \xrightarrow{\perp} \perp$,
$(2, 1) \xrightarrow{\perp} (3, \max)$,
$(2, 0) \xrightarrow{\top} \perp$,
$(2, 0) \xrightarrow{\perp} (3, \max)$,

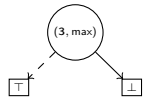
]

Level: 3

[ ]

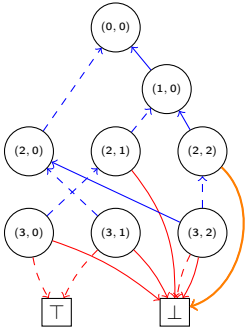
[,
]

Output:



(d) $(a) \wedge (b)$ reduced

Adiar v1.0 : BDD



(c) $(a) \wedge (b)$

Priority Queue: Q_{red}:

[

$(2,1) \xrightarrow{\bot} (3, \max)$,

$(2,0) \xrightarrow{\top} \bot$,

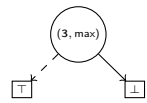
$(2,0) \xrightarrow{\bot} (3, \max)$,

]

Level: 2

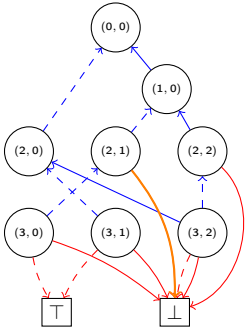
[$[(2,2) \mapsto \bot]$]

Output:



(d) $(a) \wedge (b)$ reduced

Adiar v1.0 : BDD



(c) $(a) \wedge (b)$

Priority Queue: Q_{red}:

[

$(2, 0) \xrightarrow{\top} \bot$,

$(2, 0) \xrightarrow{\bot} (3, \max)$,

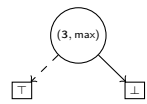
]

Level: 2

[$[(2, 2) \mapsto \bot]$]

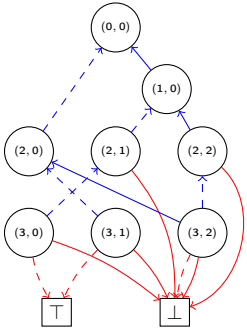
[$((2, 1), (3, \max), \bot)$,
]

Output:



(d) $(a) \wedge (b)$ reduced

Adiar v1.0 : BDD



(c) $(a) \wedge (b)$

Priority Queue: Q_{red}:

[

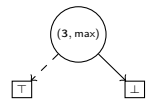
]

Level: 2

[$[(2, 2) \mapsto \bot]$]

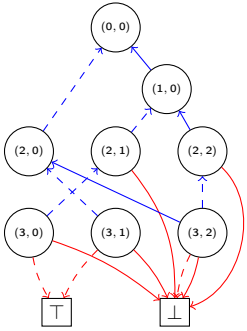
[$((2, 1), (3, \text{max}), \bot)$,
$((2, 0), (3, \text{max}), \bot)$]

Output:



(d) $(a) \wedge (b)$ reduced

Adiar v1.0 : BDD



(c) $(a) \wedge (b)$

Priority Queue: Q_{red}:

[

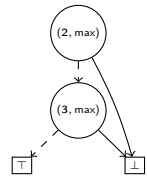
]

Level: 2

[$[(2, 2) \mapsto \bot]$]

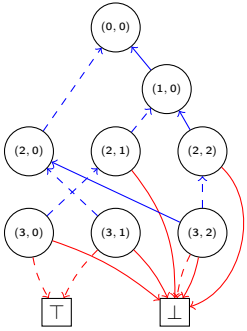
[$[(2, 1) \mapsto (2, \max)]$,
$((2, 0), (3, \max), \bot)$]

Output:
$((2, \max), (3, \max), \bot)$



(d) $(a) \wedge (b)$ reduced

Adiar v1.0 : BDD



(c) $(a) \wedge (b)$

Priority Queue: Q_{red}:

[

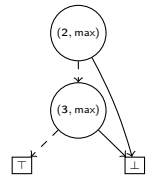
]

Level: 2

[$[(2, 2) \mapsto \bot]$]

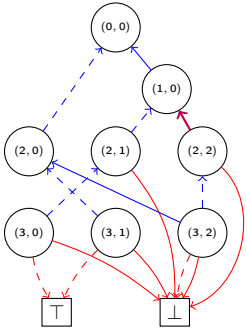
[$[(2, 1) \mapsto (2, \max)]$,
$[(2, 0) \mapsto (2, \max)]$]

Output:



(d) $(a) \wedge (b)$ reduced

Adiar v1.0 : BDD



(c) $(a) \wedge (b)$

Priority Queue: Q_{red}:

[

$(1, 0) \xrightarrow{\top} \bot$,

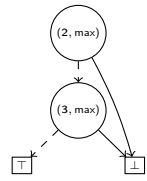
]

Level: 2

[ ]

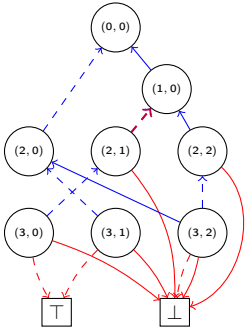
[$[(2, 1) \mapsto (2, \max)]$,
$[(2, 0) \mapsto (2, \max)]$]

Output:



(d) $(a) \wedge (b)$ reduced

Adiar v1.0 : BDD



(c) $(a) \wedge (b)$

Priority Queue: Q_{red}:

[

$(1, 0) \xrightarrow{\top} \bot$,

$(1, 0) \xrightarrow{\bot} (2, \text{max})$,

]

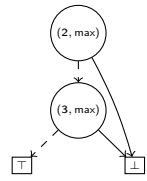
Level: 2

[ ]

[,

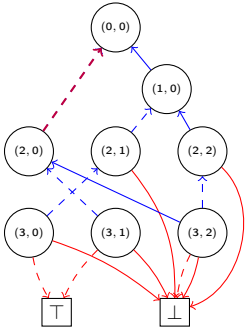
$[(2, 0) \mapsto (2, \text{max})]$]

Output:



(d) $(a) \wedge (b)$ reduced

Adiar v1.0 : BDD



(c) $(a) \wedge (b)$

Priority Queue: Q_{red}:

[

$(1, 0) \xrightarrow{\top} \bot$,

$(1, 0) \xrightarrow{\bot} (2, \max)$,

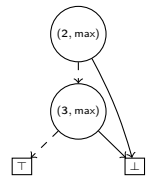
$(0, 0) \xrightarrow{\bot} (2, \max)$]

Level: 2

[ ]

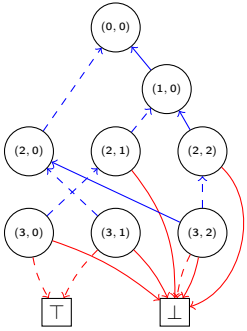
[,
]

Output:



(d) $(a) \wedge (b)$ reduced

Adiar v1.0 : BDD



(c) $(a) \wedge (b)$

Priority Queue: Q_{red}:

[

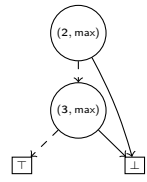
$(0, 0) \xrightarrow{\perp} (2, \text{max})$]

Level: 1

[ ]

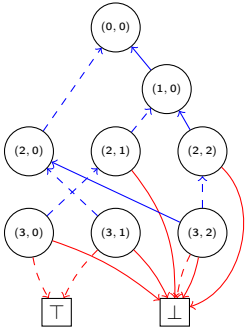
[$((1, 0), (2, \text{max}), \perp)$]

Output:



(d) $(a) \wedge (b)$ reduced

Adiar v1.0 : BDD



(c) $(a) \wedge (b)$

Priority Queue: Q_{red}:

[

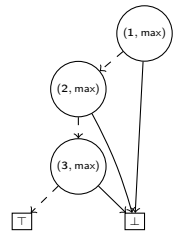
$(0, 0) \xrightarrow{\perp} (2, \text{max})$]

Level: 1

[ ]

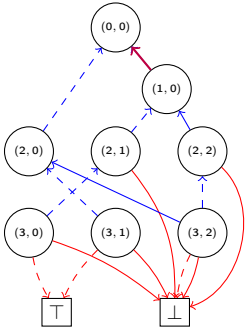
[$[(1, 0) \mapsto (1, \text{max})]$]

Output:
$((1, \text{max}), (2, \text{max}), \perp)$



(d) $(a) \wedge (b)$ reduced

Adiar v1.0 : BDD



(c) $(a) \wedge (b)$

Priority Queue: Q_{red}:

[

$(0, 0) \xrightarrow{\top} (1, \text{max})$,

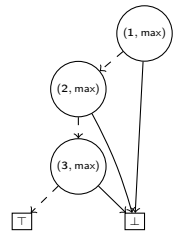
$(0, 0) \xrightarrow{\bot} (2, \text{max})$]

Level: 1

[ ]

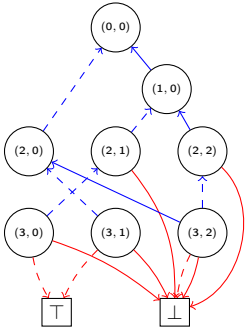
[ ]

Output:



(d) $(a) \wedge (b)$ reduced

Adiar v1.0 : BDD



(c) $(a) \wedge (b)$

Priority Queue: Q_{red}:

[

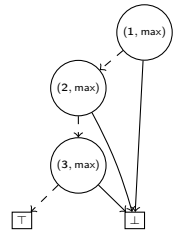
]

Level: 0

[ ]

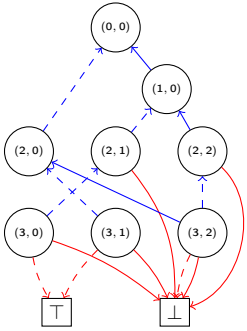
[((0, 0), (2, max), (1, max))]

Output:



(d) $(a) \wedge (b)$ reduced

Adiar v1.0 : BDD



(c) $(a) \wedge (b)$

Priority Queue: Q_{red}:

[

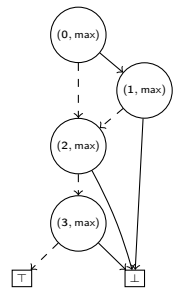
]

Level: 0

[ ]

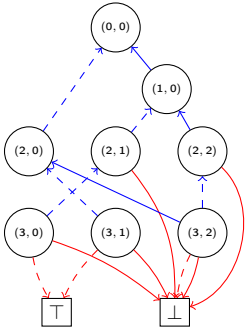
[[(0, 0) ↦ (0, max)]]

Output:
((0, max), (2, max), (1, max))



(d) $(a) \wedge (b)$ reduced

Adiar v1.0 : BDD



(c) $(a) \wedge (b)$

Priority Queue: Q_{red}:

[

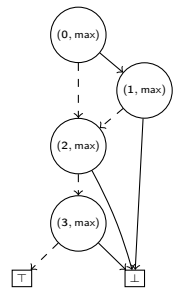
]

Level: 0

[ ]

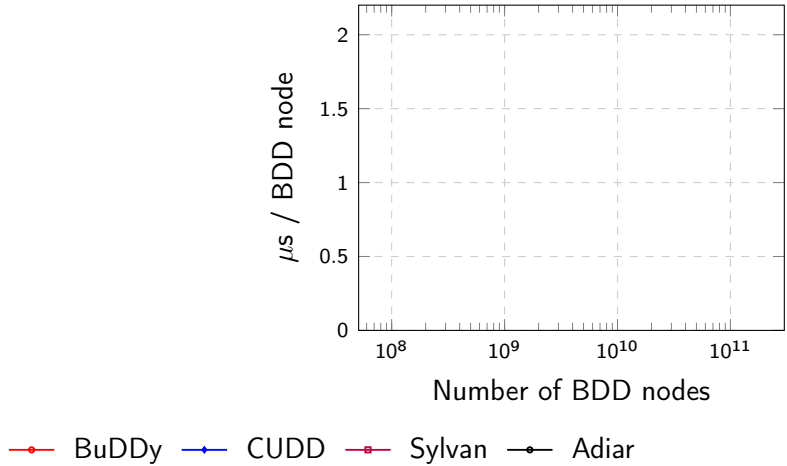
[ ]

Output:



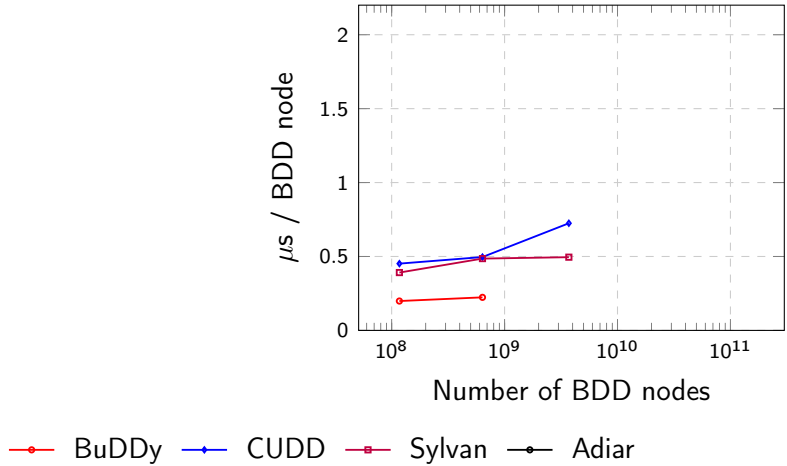
(d) $(a) \wedge (b)$ reduced

Adiar v1.0 : BDD



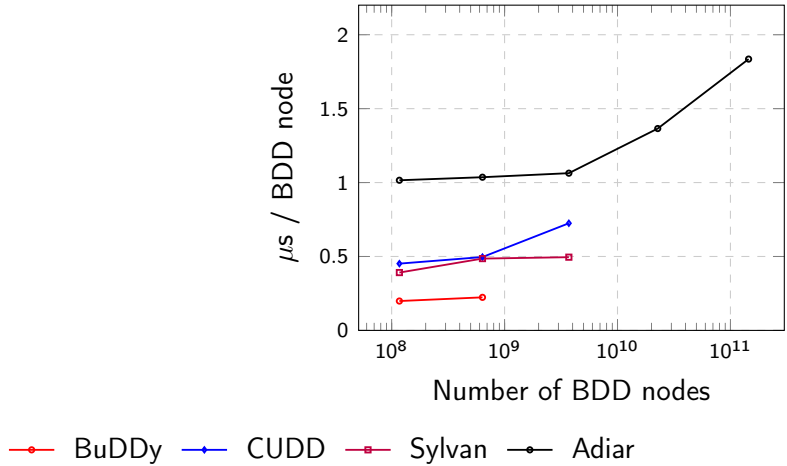
Minimal running time for the *Queens* problems.

Adiar v1.0 : BDD



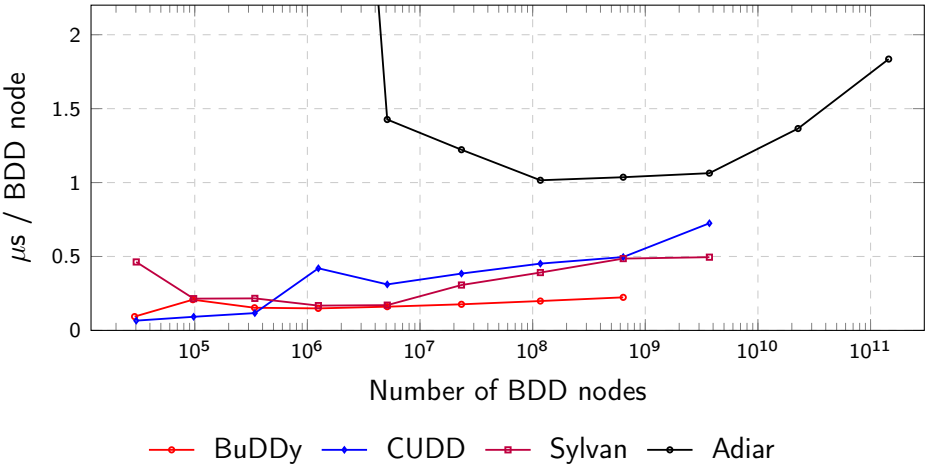
Minimal running time for the *Queens* problems.

Adiar v1.0 : BDD



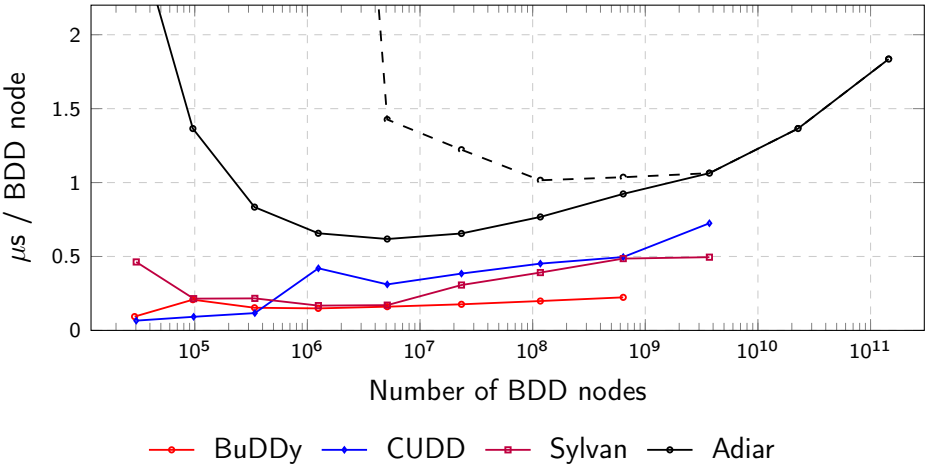
Minimal running time for the *Queens* problems.

Adiar v1.0 : BDD



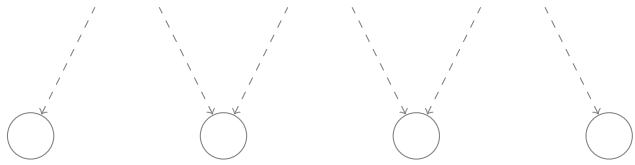
Minimal running time for the *Queens* problems.

Adiar v1.2 : Internal Memory

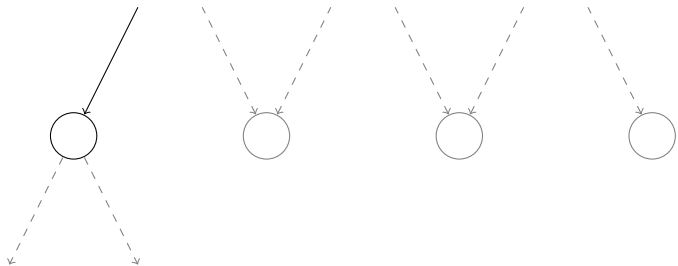


Minimal running time for the *Queens* problems.

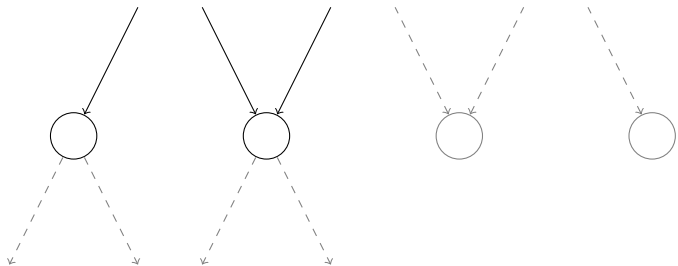
Adiar v1.2 : Internal Memory



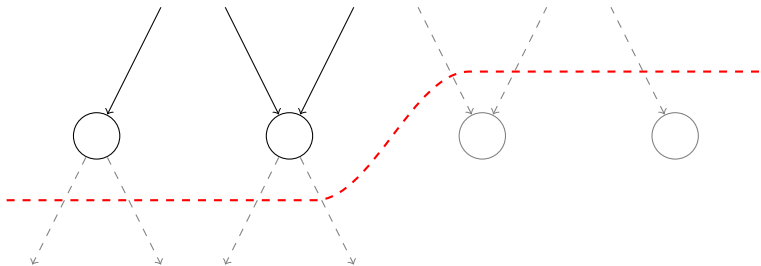
Adiar v1.2 : Internal Memory



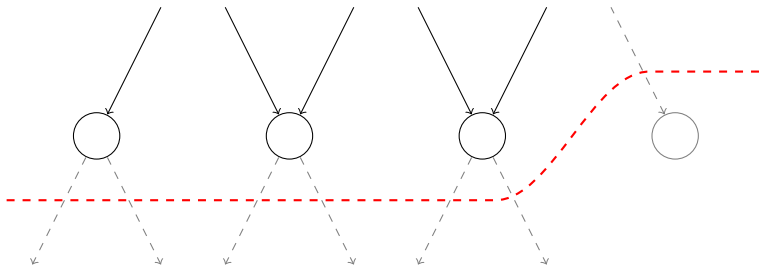
Adiar v1.2 : Internal Memory



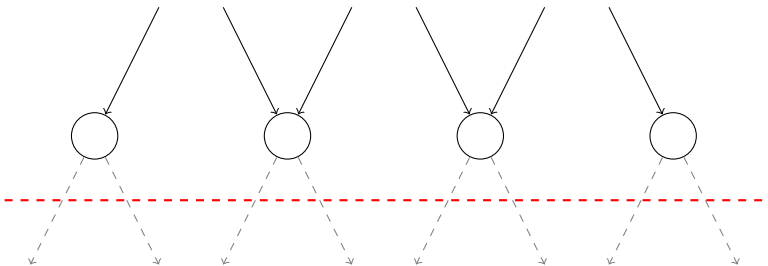
Adiar v1.2 : Internal Memory



Adiar v1.2 : Internal Memory

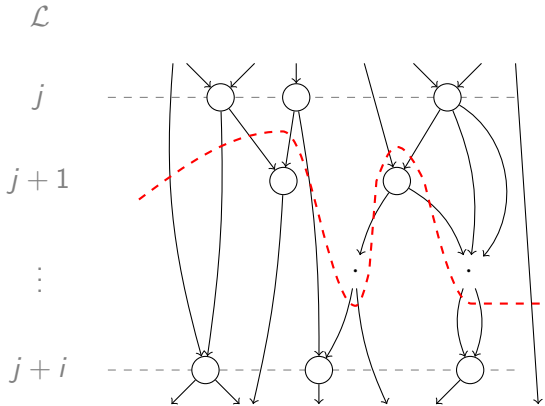


Adiar v1.2 : Internal Memory



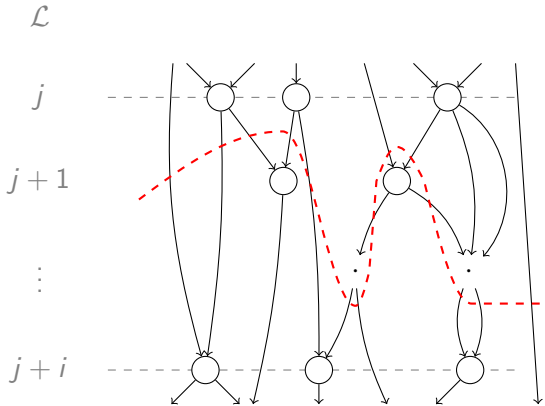
Adiar v1.2 : Internal Memory

Definition (i-level cut)



Adiar v1.2 : Internal Memory

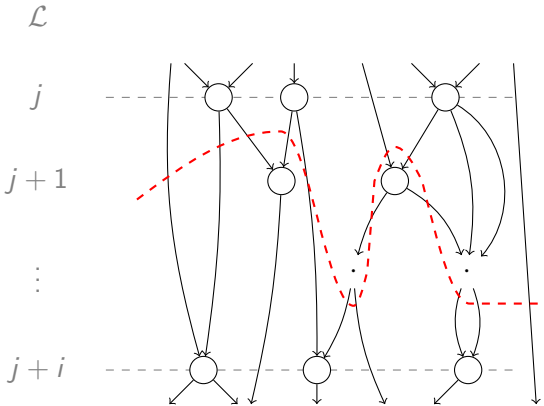
Definition (i-level cut)



Lemma
The maximum i-level cut problem is in P for $i \in \{1, 2\}$.

Adiar v1.2 : Internal Memory

Definition (i-level cut)



Lemma
The maximum i-level cut problem is in P for $i \in \{1, 2\}$.

Theorem (Lampis, Kaouri, Mitsou 2011)
The maximum i-level cut problem is NP-complete for $i \geq 4$.

Adiar v1.2 : Internal Memory

Adiar v1.2 : Internal Memory

Theorem
The maximum (i-level) cut of a BDD with N internal nodes is $N + 1$.

Adiar v1.2 : Internal Memory

Theorem
The maximum (i-level) cut of a BDD with N internal nodes is $N + 1$.

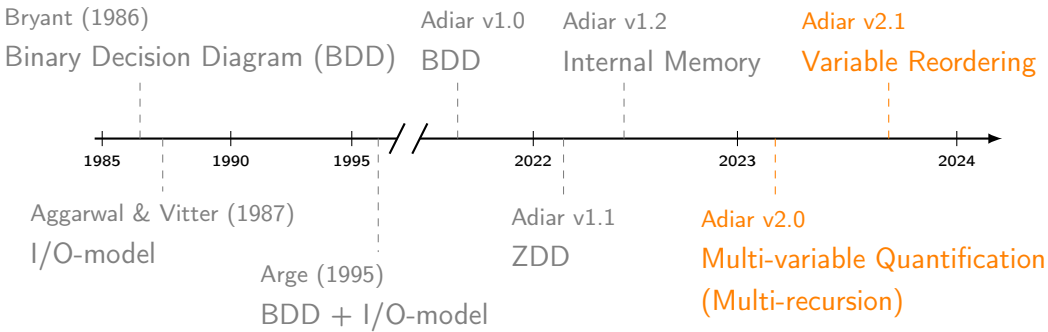
Theorem
For $i \in \{1, 2\}$, *the maximum i-level cut of the (unreduced) output of Apply is upper bounded by the product of the inputs' corresponding i-level cuts.*

Adiar v1.2 : Internal Memory

Theorem
The maximum (i-level) cut of a BDD with N internal nodes is $N + 1$.

Theorem
For $i \in \{1, 2\}$, *the maximum i-level cut of the (unreduced) output of Apply is upper bounded by the product of the inputs' corresponding i-level cuts.*

Lemma
The maximum 2*-level cut of a BDD is upper bounded by* $\frac{3}{2}$ *its* 1*-level cut.*


Bryant (1986)
Binary Decision Diagram (BDD)
Adiar v1.0
BDD
Adiar v1.2
Internal Memory
Adiar v2.1
Variable Reordering
1985
1990
1995
2022
2023
2024
Aggarwal & Vitter (1987)
I/O-model
Arge (1995)
BDD + I/O-model
Adiar v1.1
ZDD
Adiar v2.0
Multi-variable Quantification
(Multi-recursion)

Adiar v2.0 : Multi-variable Quantification

$$RelProd(S, T) \equiv (\exists \vec{x}.\ S(\vec{x}) \wedge T(\vec{x}, \vec{x'}))[\vec{x'}/\vec{x}]$$

Adiar v2.0 : Multi-variable Quantification

$$RelProd(S, T) \equiv (\exists \vec{x}.\ S(\vec{x}) \wedge T(\vec{x}, \vec{x'}))[\vec{x'}/\vec{x}]$$

Adiar v2.0 : Multi-variable Quantification

```
1  exists(f, V)
2    if f = ⊥ ∨ f = ⊤
3         then f
4    else if V ∩ {i ∈ ℕ | i ≥ top(f)} = ∅
5         then f
6    else if top(f) ∉ V
7         then Node { top(f), exists(f.low, V), exists(f.high, V) }
8    else let low  = exists(f.low, V)
9             high = exists(f.high, V)
10        in or(low, high)
```

A recursive multi-variable **exists** operation.

Adiar v2.0 : Multi-variable Quantification

```
1  exists(f, V)
2    if f = ⊥ ∨ f = ⊤
3         then f
4    else if V ∩ {i ∈ ℕ | i ≥ top(f)} = ∅
5         then f
6    else if top(f) ∉ V
7         then Node { top(f), exists(f.low, V), exists(f.high, V) }
8    else let low  = exists(f.low, V)
9             high = exists(f.high, V)
10        in or(low, high)
```

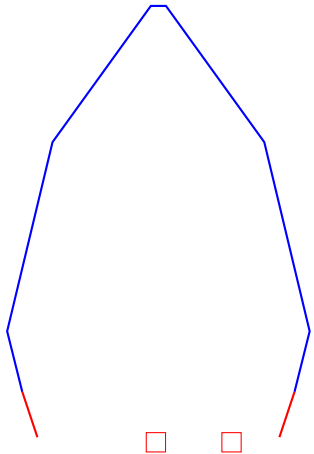
A recursive multi-variable **exists** operation.

Adiar v2.0 : Multi-variable Quantification

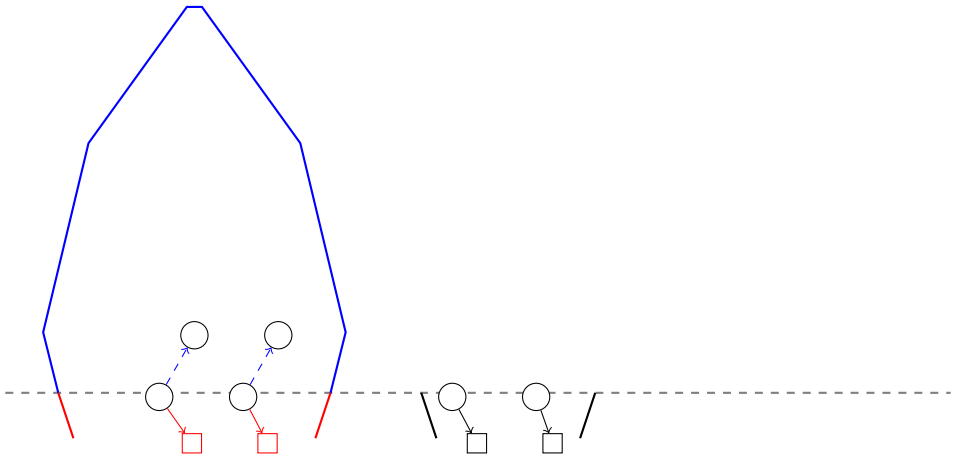
```
1  exists(f, V)
2    if f = ⊥ ∨ f = ⊤
3         then f
4    else if V ∩ {i ∈ ℕ | i ≥ top(f)} = ∅
5         then f
6    else if top(f) ∉ V
7         then Node { top(f), exists(f.low, V), exists(f.high, V) }
8    else let low  = exists(f.low, V)
9             high = exists(f.high, V)
10        in or(low, high)
```

A recursive multi-variable **exists** operation.

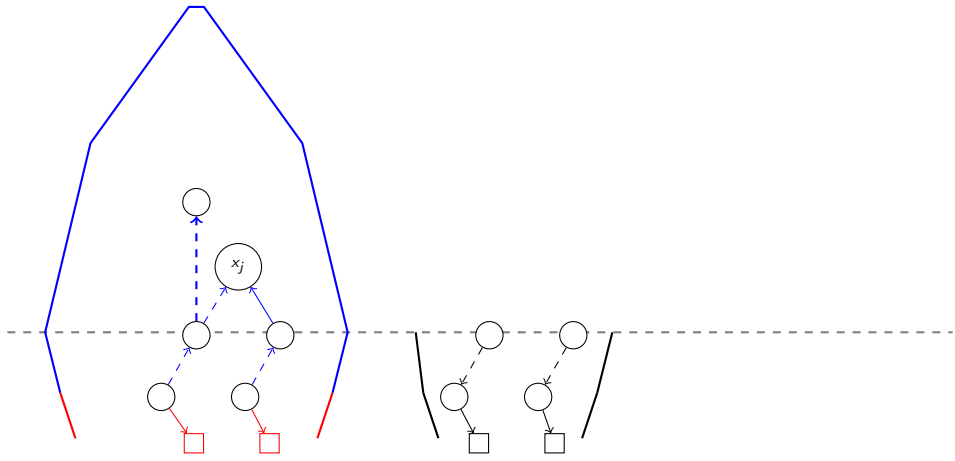
Adiar v2.0 : Multi-variable Quantification



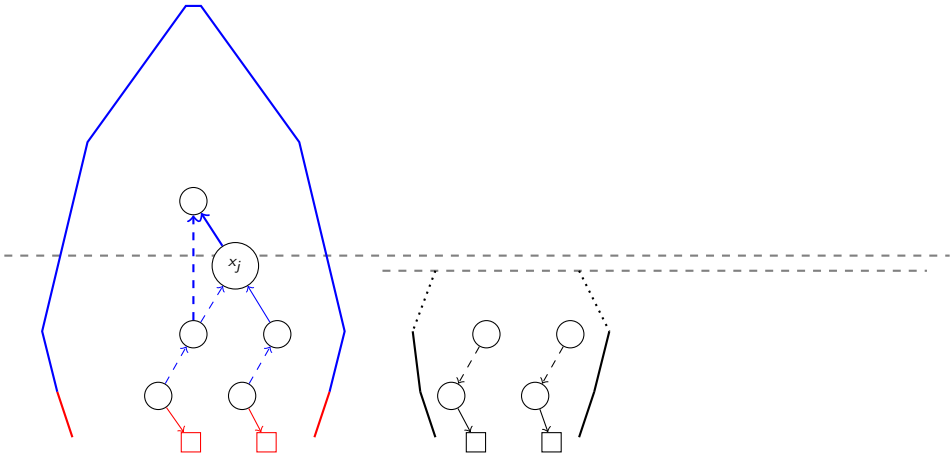
Adiar v2.0 : Multi-variable Quantification



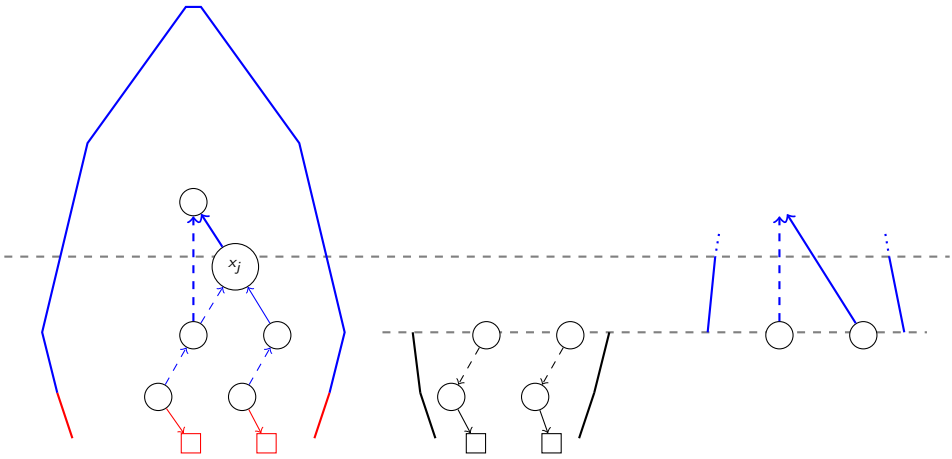
Adiar v2.0 : Multi-variable Quantification



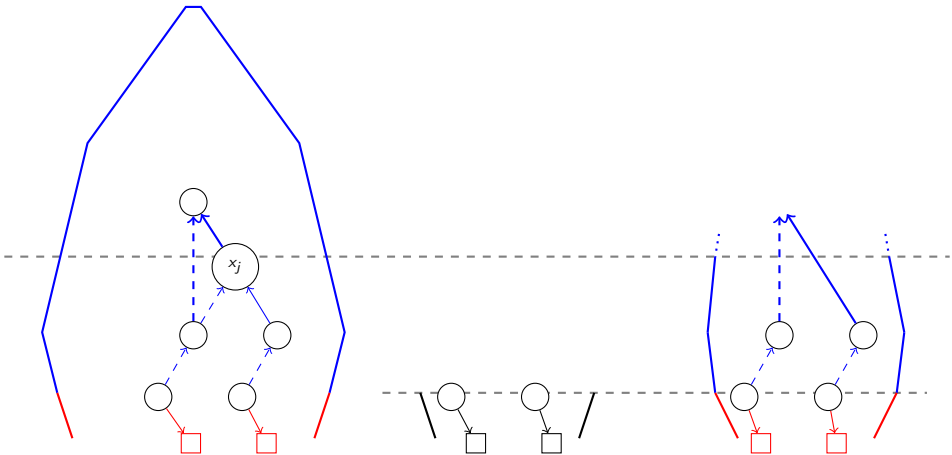
Adiar v2.0 : Multi-variable Quantification



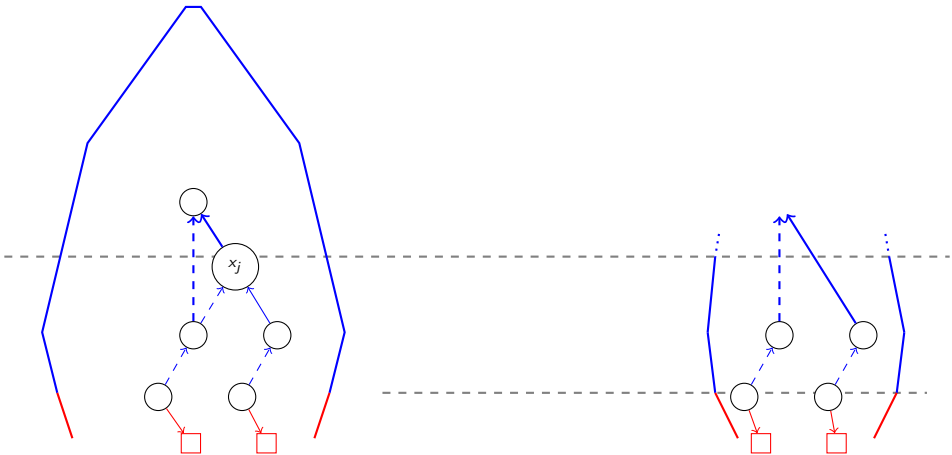
Adiar v2.0 : Multi-variable Quantification



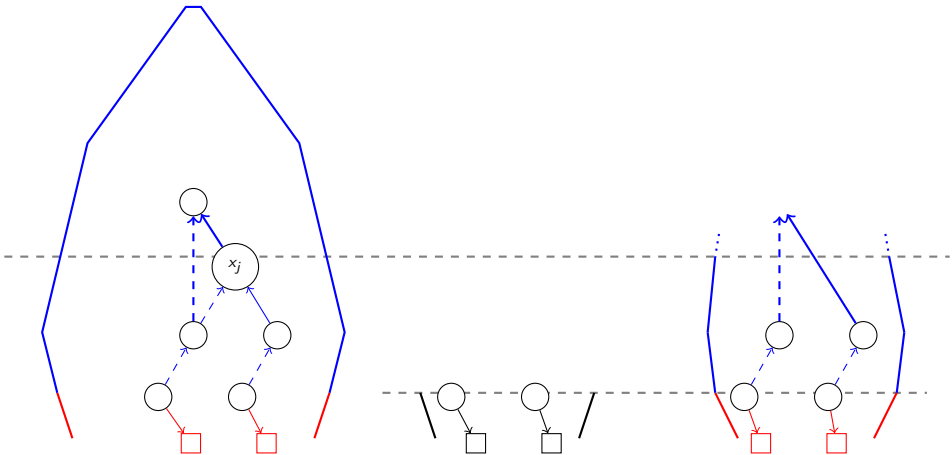
Adiar v2.0 : Multi-variable Quantification



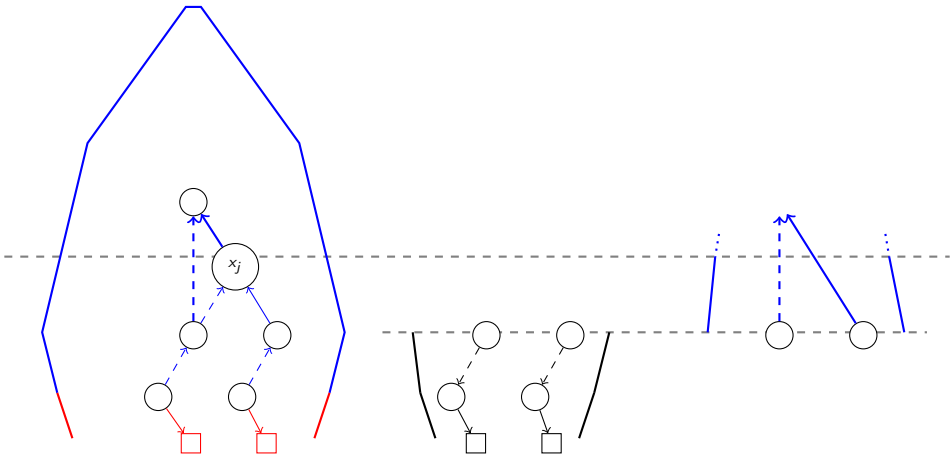
Adiar v2.0 : Multi-variable Quantification



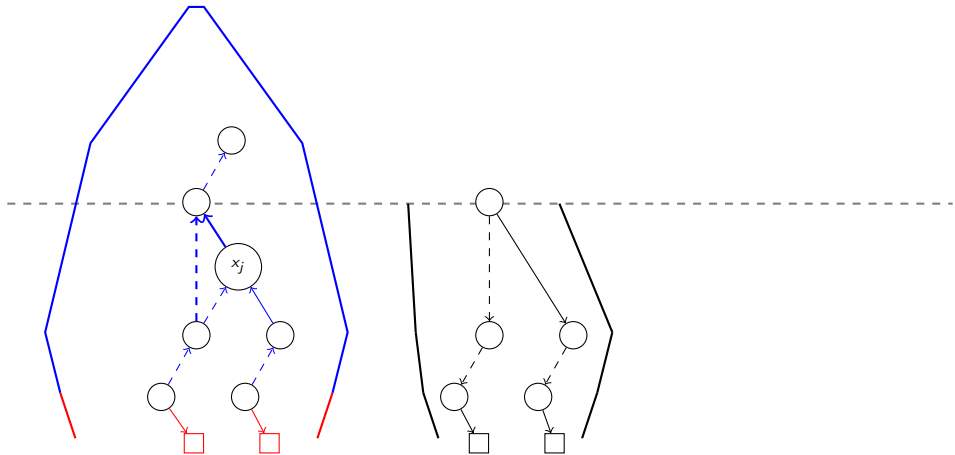
Adiar v2.0 : Multi-variable Quantification



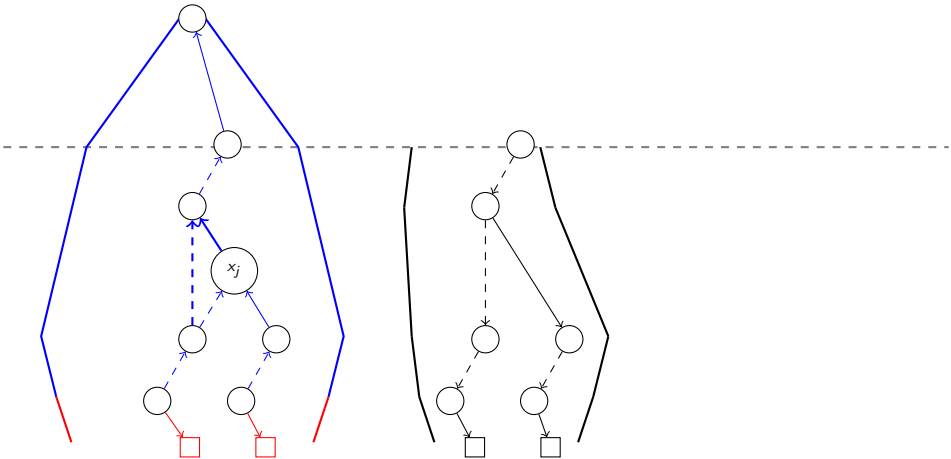
Adiar v2.0 : Multi-variable Quantification



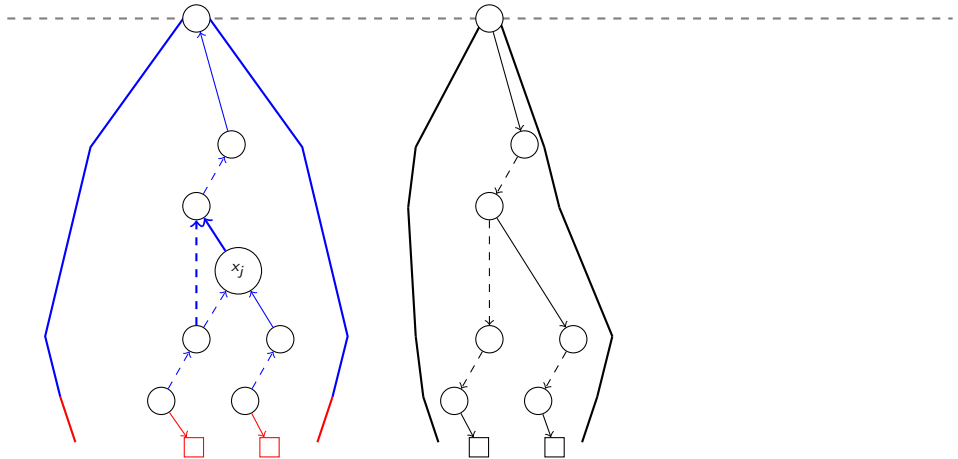
Adiar v2.0 : Multi-variable Quantification



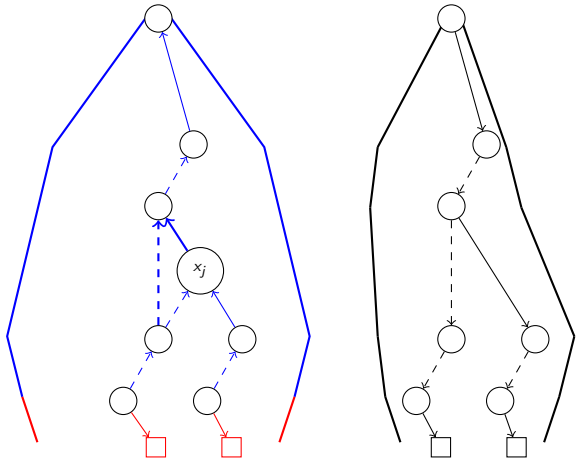
Adiar v2.0 : Multi-variable Quantification



Adiar v2.0 : Multi-variable Quantification



Adiar v2.0 : Multi-variable Quantification



Adiar v2.1 : Variable Reordering

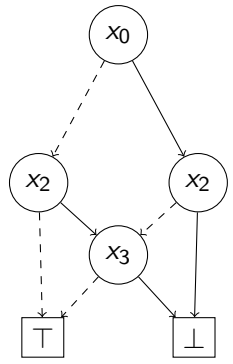
$$RelProd(S, T) \equiv (\exists \vec{x}.\ S(\vec{x}) \wedge T(\vec{x}, \vec{x'}))[\vec{x'}/\vec{x}]$$

Adiar v2.1 : Variable Reordering

```
1 substitute(f, i_map)
2   let low  = substitute(f.low)
3       high = substitute(f.high)
4       i'   = i_map[top(f)]
5   in bubble(i', low, high)
```

A recursive **substitute** operation.

$[0 \mapsto 3, 1 \mapsto 0, 2 \mapsto 2, 3 \mapsto 1]$



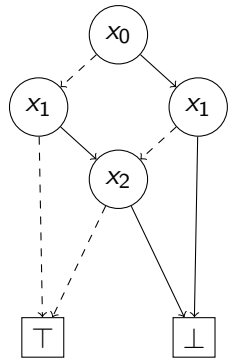
$\neg(x_0 \ ? \ x_2 \vee x_3 \ : \ x_2 \wedge x_3)$

Adiar v2.1 : Variable Reordering

```
1 substitute(f, i_map)
2   let low  = substitute(f.low)
3       high = substitute(f.high)
4       i'   = i_map[top(f)]
5   in bubble(i', low, high)
```

A recursive **substitute** operation.

$[0 \mapsto 3, 1 \mapsto 0, 2 \mapsto 2, 3 \mapsto 1]$



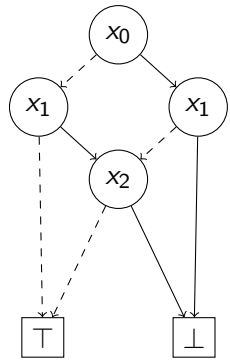
$\neg(x_0 \ ? \ x_2 \vee x_1 \ : \ x_2 \wedge x_1)$

Adiar v2.1 : Variable Reordering

```
1 substitute(f, i_map)
2   let low  = substitute(f.low)
3       high = substitute(f.high)
4       i'   = i_map[top(f)]
5   in bubble(i', low, high)
```

A recursive **substitute** operation.

$[0 \mapsto 3, 1 \mapsto 0, 2 \mapsto 2, 3 \mapsto 1]$



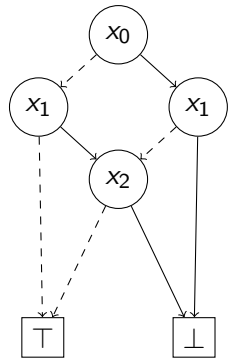
$\neg(x_0 \ ? \ x_2 \vee x_1 \ : \ x_2 \wedge x_1)$

Adiar v2.1 : Variable Reordering

```
1 substitute(f, i_map)
2   let low  = substitute(f.low)
3       high = substitute(f.high)
4       i'   = i_map[top(f)]
5   in bubble(i', low, high)
```

A recursive **substitute** operation.

$[0 \mapsto 3, 1 \mapsto 0, 2 \mapsto 2, 3 \mapsto 1]$



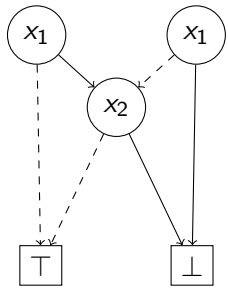
$\neg(x_0 \ ? \ x_2 \vee x_1 \ : \ x_2 \wedge x_1)$

Adiar v2.1 : Variable Reordering

$[0 \mapsto 3, 1 \mapsto 0, 2 \mapsto 2, 3 \mapsto 1]$

```
1 substitute(f, i_map)
2   let low  = substitute(f.low)
3       high = substitute(f.high)
4       i'   = i_map[top(f)]
5   in bubble(i', low, high)
```

A recursive **substitute** operation.



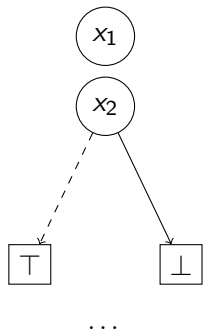
...

Adiar v2.1 : Variable Reordering

$[0 \mapsto 3, 1 \mapsto 0, 2 \mapsto 2, 3 \mapsto 1]$

```
1 substitute(f, i_map)
2   let low  = substitute(f.low)
3       high = substitute(f.high)
4       i'   = i_map[top(f)]
5   in bubble(i', low, high)
```

A recursive **substitute** operation.

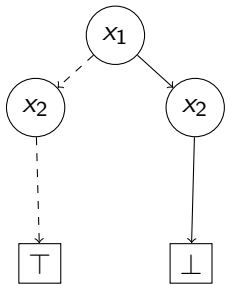


Adiar v2.1 : Variable Reordering

$[0 \mapsto 3, 1 \mapsto 0, 2 \mapsto 2, 3 \mapsto 1]$

```
1 substitute(f, i_map)
2   let low  = substitute(f.low)
3       high = substitute(f.high)
4       i'   = i_map[top(f)]
5   in bubble(i', low, high)
```

A recursive **substitute** operation.



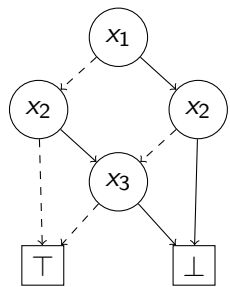
...

Adiar v2.1 : Variable Reordering

$[0 \mapsto 3, 1 \mapsto 0, 2 \mapsto 2, 3 \mapsto 1]$

```
1 substitute(f, i_map)
2   let low  = substitute(f.low)
3       high = substitute(f.high)
4       i'   = i_map[top(f)]
5   in bubble(i', low, high)
```

A recursive **substitute** operation.



$\neg(x_3 \ ? \ x_2 \vee x_1 \ : \ x_2 \wedge x_1)$

Adiar v2.1 : Variable Reordering

```
1 substitute(f, i_map)
2   let low  = substitute(f.low)
3       high = substitute(f.high)
4       i'   = i_map[top(f)]
5   in bubble(i', low, high)
```

A recursive **substitute** operation.

Time	Space	I/O
$O(NT)$	$O(NT)$	$O(NT)$

Complexity of depth-first **substitute**

Adiar v2.1 : Variable Reordering

```
1 substitute(f, i_map)
2   let low  = substitute(f.low)
3       high = substitute(f.high)
4       i'   = i_map[top(f)]
5   in bubble(i', low, high)
```

A recursive **substitute** operation.

Time	Space	I/O
$O(NT)$	$O(NT)$	$O(NT)$

Complexity of depth-first **substitute**

Time	Space	I/O
$O(NT \log T)$	$O(N + T)$	$O(N \cdot \text{sort}(T))$

Complexity of level-by-level **substitute**

Adiar v2.1 : Variable Reordering

Problem (Variable Replacement)
Given BDD f_π with variable ordering π and remapping of variables $m : \mathbb{N} \to \mathbb{N}$, construct $f'_\pi \equiv f_\pi[x/m(x)]$.

Adiar v2.1 : Variable Reordering

Problem (Variable Replacement)
Given BDD f_π with variable ordering π and remapping of variables $m : \mathbb{N} \to \mathbb{N}$, construct $f'_\pi \equiv f_\pi[x/m(x)]$.

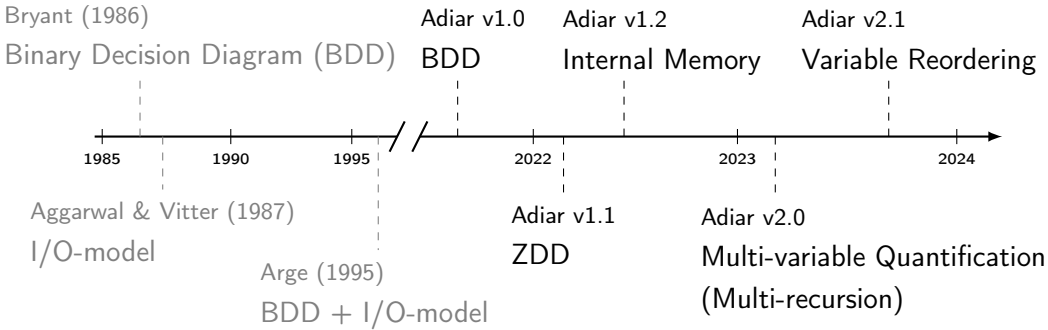
Problem (Static Variable Reordering)
Given BDD f_π with variable ordering π and another variable ordering π', construct $f_{\pi'} \equiv f_\pi$.

Adiar v2.1 : Variable Reordering

Problem (Variable Replacement)
Given BDD f_π with variable ordering π and remapping of variables $m : \mathbb{N} \to \mathbb{N}$,
construct $f'_\pi \equiv f_\pi[x/m(x)]$.

Problem (Static Variable Reordering)
Given BDD f_π with variable ordering π and another variable ordering π',
construct $f_{\pi'} \equiv f_\pi$.

Problem (Dynamic Variable Reordering)
Given BDD f_π with variable ordering π,
find π' and construct $f_{\pi'} \equiv f_\pi$ such that $|f_{\pi'}|$ is minimal.


Bryant (1986)
Binary Decision Diagram (BDD)
Adiar v1.0
BDD
Adiar v1.2
Internal Memory
Adiar v2.1
Variable Reordering
1985
1990
1995
2022
2023
2024
Aggarwal & Vitter (1987)
I/O-model
Arge (1995)
BDD + I/O-model
Adiar v1.1
ZDD
Adiar v2.0
Multi-variable Quantification
(Multi-recursion)