Adiar 1.1 : Zero-suppressed Decision Diagrams in External Memory

---

Steffan Christ Sølvsten and Jaco van de Pol

18th of May, 2023

AARHUS
UNIVERSITY

Adiar

Binary Decision Diagrams
in External Memory

github.com/ssoelvsten/adiar

Adiar

Multi-terminal Decision Diagrams
in External Memory

github.com/ssoelvsten/adiar

Adiar

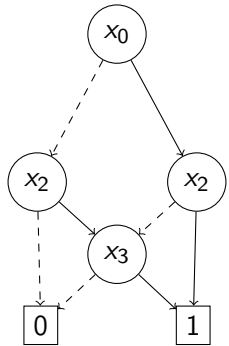
Quantum Multi-valued Decision Diagrams
in External Memory

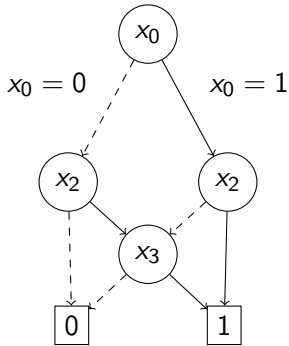
github.com/ssoelvsten/adiar

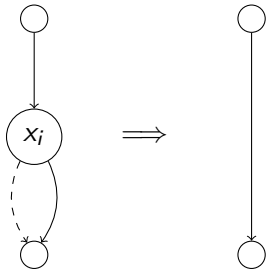
Adiar

Zero-suppressed Decision Diagrams
in External Memory

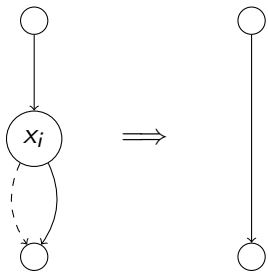
github.com/ssoelvsten/adiar


x_0
x_2
x_2
x_3
0
1

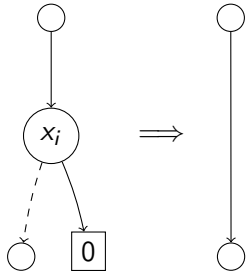

x_0
$x_0 = 0$
$x_0 = 1$
x_2
x_2
x_3
0
1



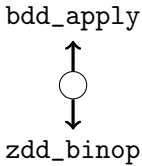
BDD: $f : \mathbb{B}^n \to \mathbb{B}$

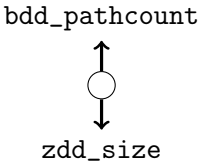


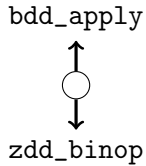
BDD: $f : \mathbb{B}^n \to \mathbb{B}$

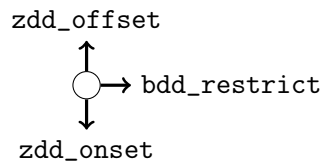


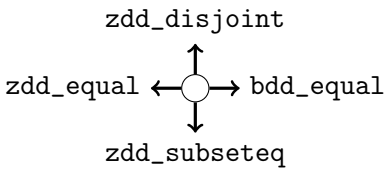
ZDD: $A \subseteq \mathcal{P}(\mathbb{B}^n)$

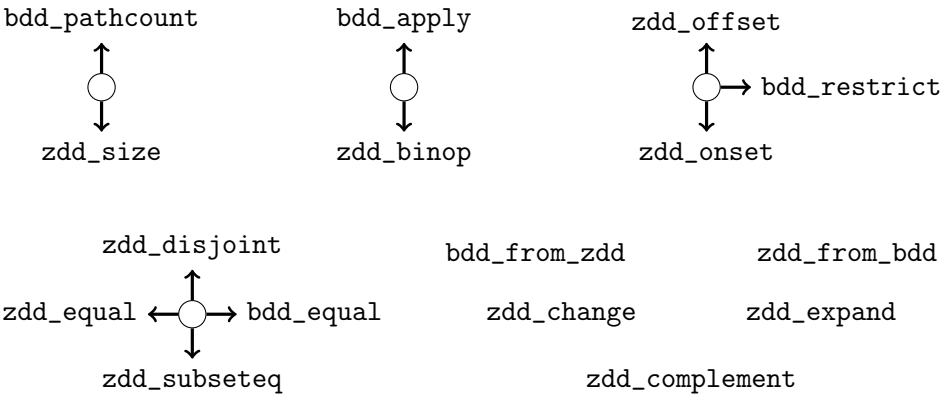

bdd_apply
zdd_binop

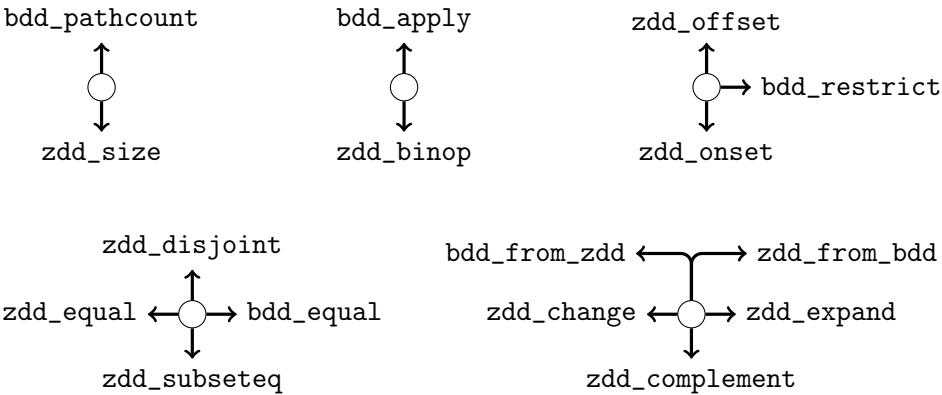

bdd_pathcount
zdd_size


bdd_apply
zdd_binop


zdd_offset
bdd_restrict
zdd_onset

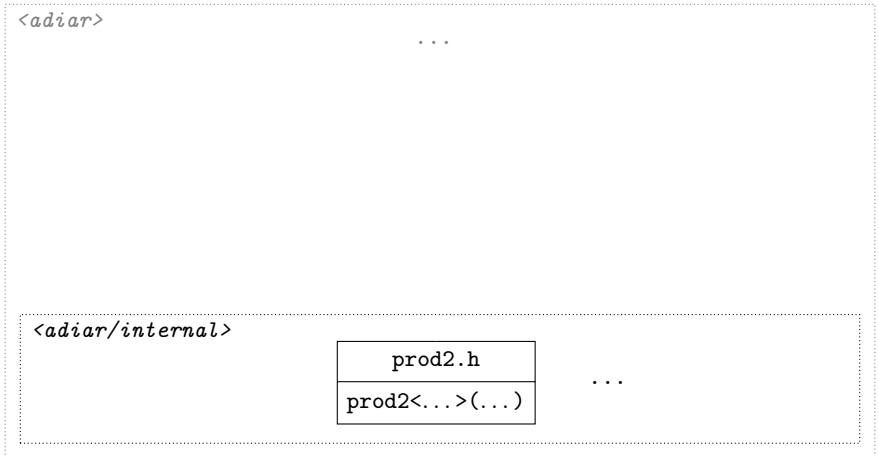

zdd_disjoint
zdd_equal
bdd_equal
zdd_subseteq

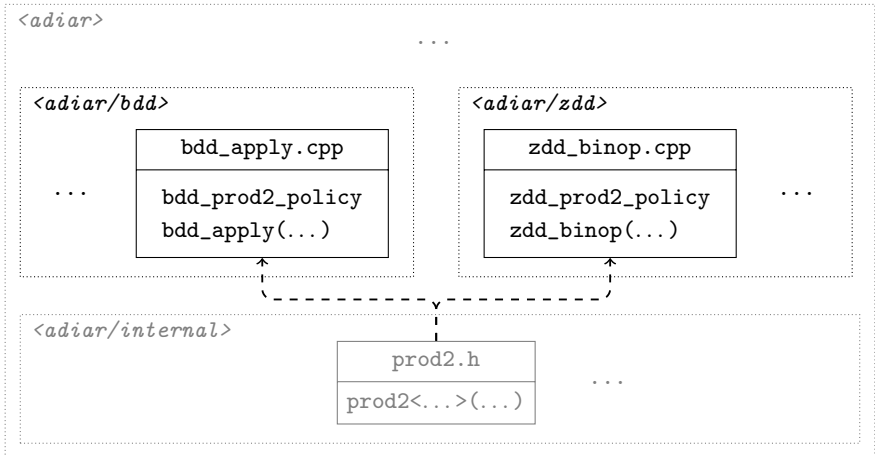

bdd_pathcount
zdd_size
bdd_apply
zdd_binop
zdd_offset
bdd_restrict
zdd_onset
zdd_disjoint
zdd_equal
bdd_equal
zdd_subseteq
bdd_from_zdd
zdd_from_bdd
zdd_change
zdd_expand
zdd_complement

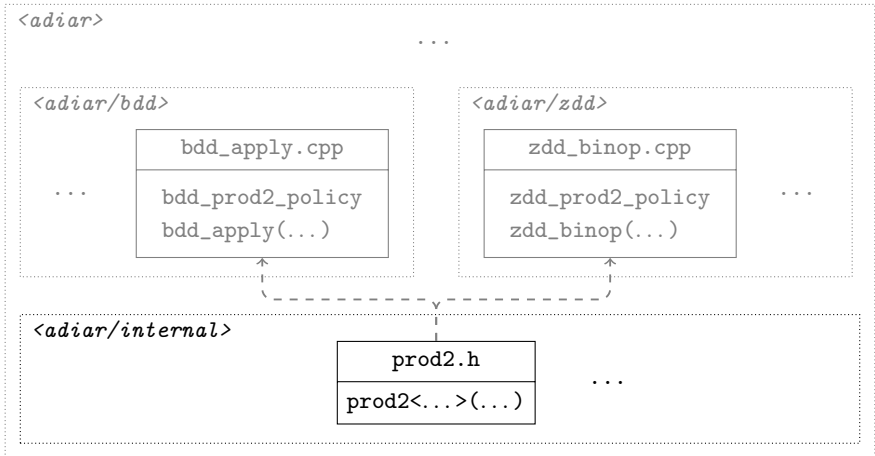

bdd_pathcount
zdd_size
bdd_apply
zdd_binop
zdd_offset
bdd_restrict
zdd_onset
zdd_disjoint
zdd_equal
bdd_equal
zdd_subseteq
bdd_from_zdd
zdd_from_bdd
zdd_change
zdd_expand
zdd_complement

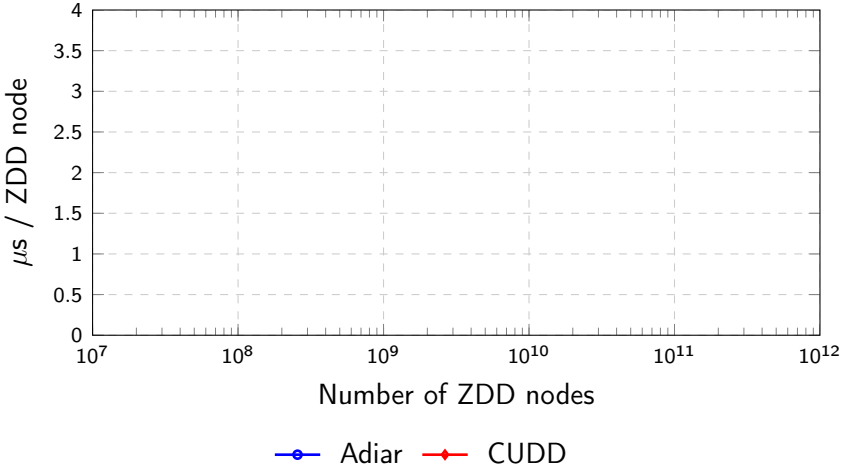
$\texttt{zdd\_binop}(A,\ B,\ \otimes) = \texttt{prod2}(A,\ B,\ \otimes,\ \texttt{zdd\_strategy})$

$\texttt{zdd\_binop}(A,\ B,\ \otimes) = \texttt{prod2<zdd\_policy>}(A,\ B,\ \otimes)$

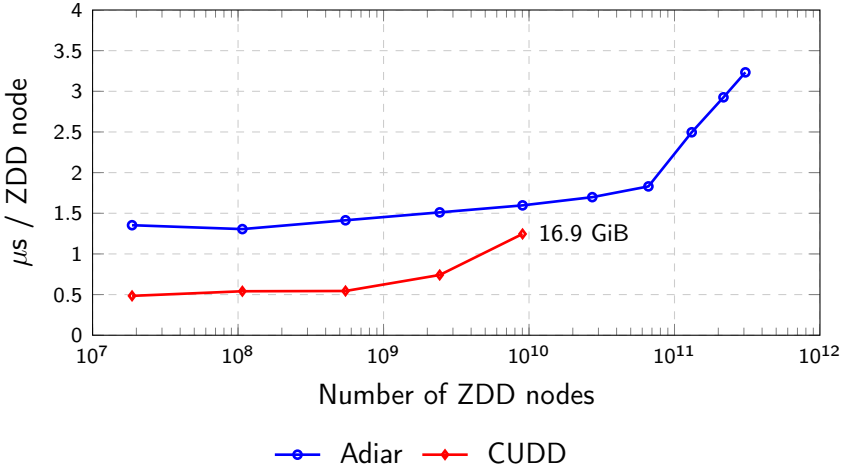

<adiar>
...
<adiar/internal>
prod2.h
prod2<...>(...)
...


<adiar>
...
<adiar/bdd>
...
bdd_apply.cpp
bdd_prod2_policy
bdd_apply(...)
<adiar/zdd>
zdd_binop.cpp
zdd_prod2_policy
zdd_binop(...)
...
<adiar/internal>
prod2.h
prod2<...>(...)
...

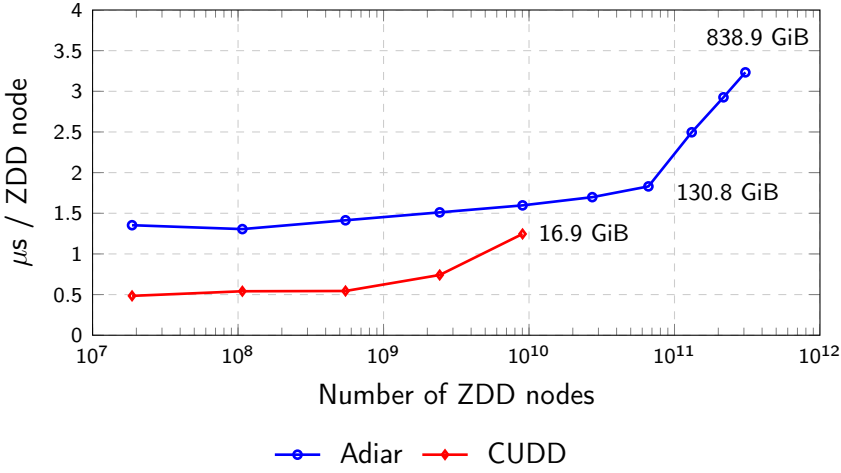

<adiar>
...
<adiar/bdd>
bdd_apply.cpp
bdd_prod2_policy
bdd_apply(...)
...
<adiar/zdd>
zdd_binop.cpp
zdd_prod2_policy
zdd_binop(...)
...
<adiar/internal>
prod2.h
prod2<...>(...)
...



Running time for *Tic-Tac-Toe* with 300 GiB of RAM.



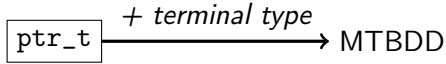
Running time for *Tic-Tac-Toe* with 300 GiB of RAM.



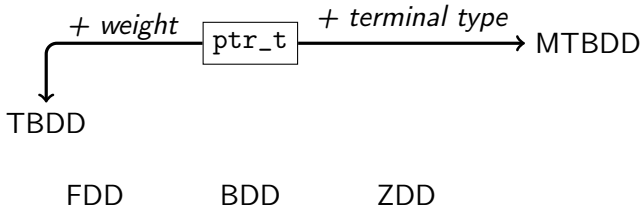
Running time for *Tic-Tac-Toe* with 300 GiB of RAM.

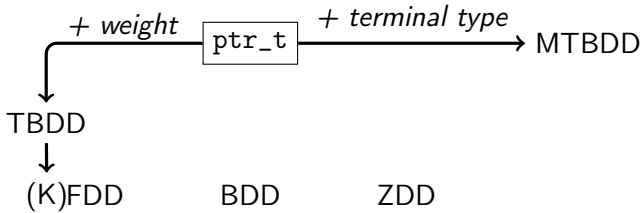
BDD ZDD

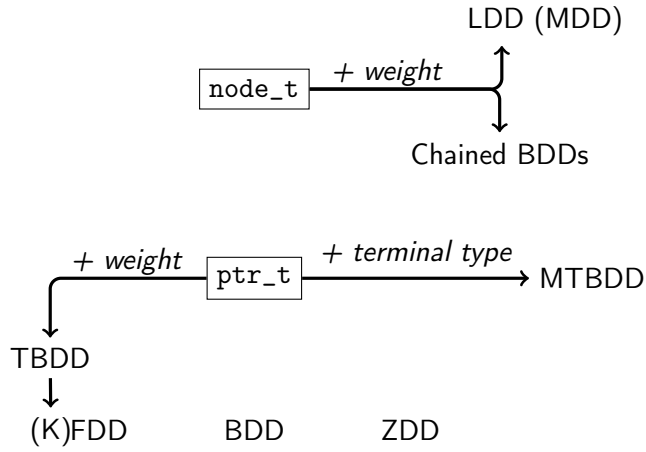
FDD BDD ZDD

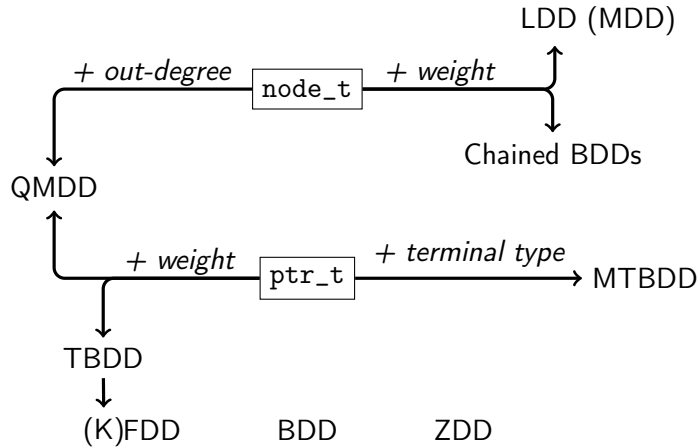


FDD BDD ZDD


+ weight
ptr_t
+ terminal type
MTBDD
TBDD
FDD
BDD
ZDD


+ weight
ptr_t
+ terminal type
MTBDD
TBDD
(K)FDD
BDD
ZDD


LDD (MDD)
node_t
+ weight
Chained BDDs
+ weight
ptr_t
+ terminal type
MTBDD
TBDD
(K)FDD
BDD
ZDD


LDD (MDD)
+ out-degree
node_t
+ weight
Chained BDDs
QMDD
+ weight
ptr_t
+ terminal type
MTBDD
TBDD
(K)FDD
BDD
ZDD

Steffan Christ Sølvsten

soelvsten@cs.au.dk

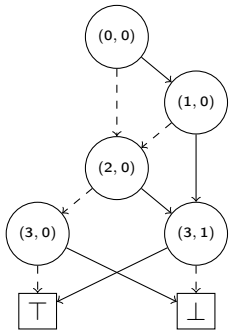
@ssoelvsten

Adiar

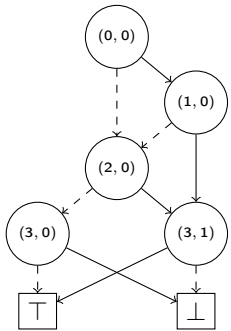
github.com/ssoelvsten/adiar

ssoelvsten.github.io/adiar

AARHUS
UNIVERSITY



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

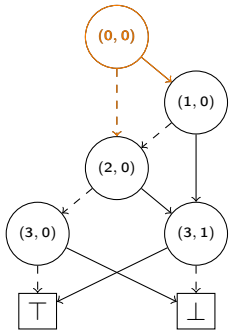


(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$

Priority Queue: Q_{count}:

[

]

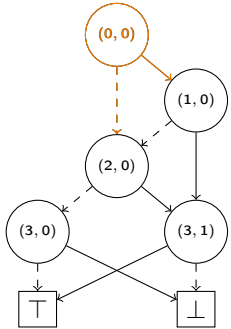


(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$

Priority Queue: Q_{count}:

[

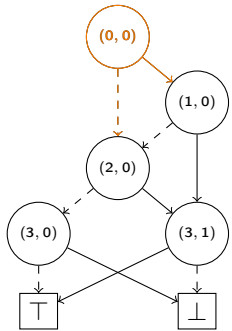
]



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

Priority Queue: Q_{count}:

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



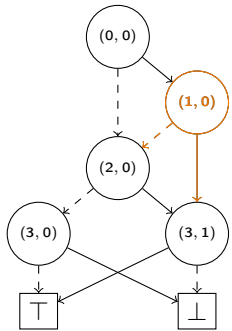
(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$

Seek	Sum	Result
$(1, 0)$	0	0

Priority Queue: Q_{count}:

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

$]$



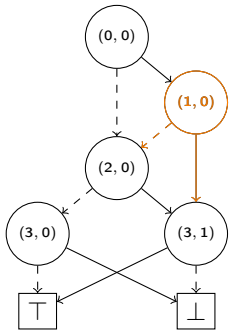
(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$

Seek	Sum	Result
$(1, 0)$	0	0

Priority Queue: Q_{count}:

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

$]$



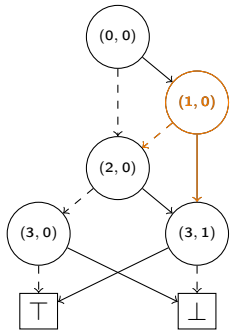
(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$

Seek	Sum	Result
$(1, 0)$	1	0

Priority Queue: Q_{count}:

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

$$]$$



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

Seek	Sum	Result
$(1,0)$	1	0

Priority Queue: Q_{count}:

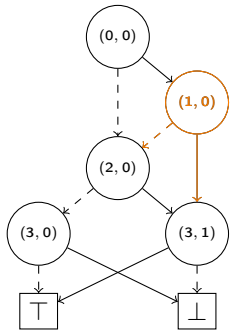
[

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

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

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

]



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

Seek	Sum	Result
$(2, 0)$	0	0

Priority Queue: Q_{count}:

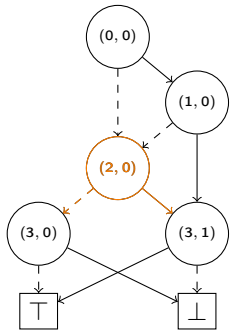
[

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

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

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

]

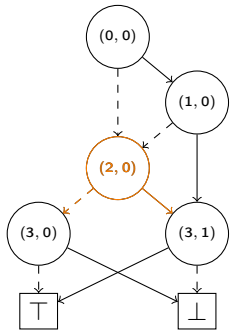


(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$

Seek	Sum	Result
$(2, 0)$	0	0

Priority Queue: Q_{count}:

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



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

Seek	Sum	Result
$(2, 0)$	1	0

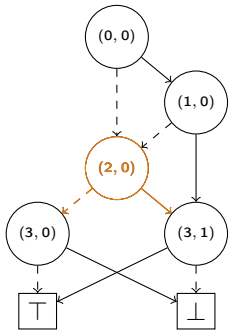
Priority Queue: Q_{count}:

[

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

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

]



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

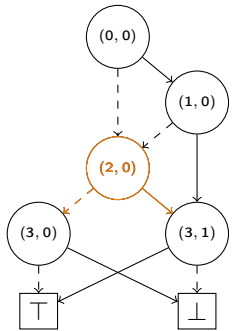
Seek	Sum	Result
$(2, 0)$	2	0

Priority Queue: Q_{count}:

[

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

]



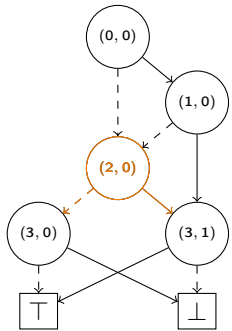
(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$

Seek	Sum	Result
$(2, 0)$	2	0

Priority Queue: Q_{count}:

[

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



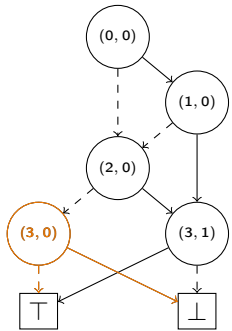
(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$

Seek	Sum	Result
$(3, 0)$	0	0

Priority Queue: Q_{count}:

[

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



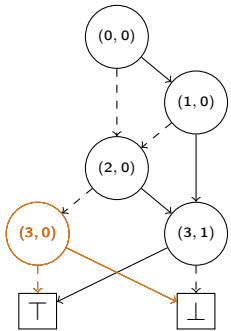
(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$

Seek	Sum	Result
$(3,0)$	0	0

Priority Queue: Q_{count}:

[

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



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

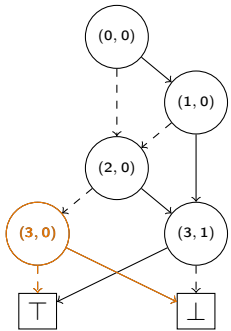
Seek	Sum	Result
$(3, 0)$	2	0

Priority Queue: Q_{count}:

[

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

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



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

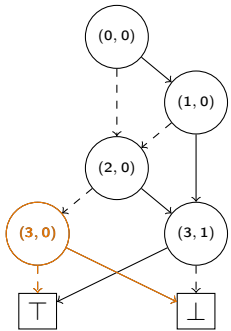
Seek	Sum	Result
$(3,0)$	2	2

Priority Queue: Q_{count}:

[

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

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



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

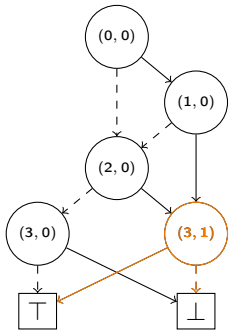
Seek	Sum	Result
$(3,1)$	0	2

Priority Queue: Q_{count}:

[

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

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



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

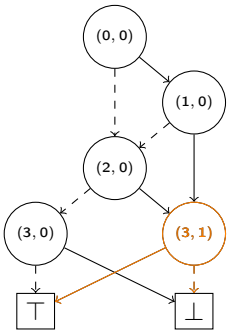
Seek	Sum	Result
$(3,1)$	0	2

Priority Queue: Q_{count}:

[

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

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



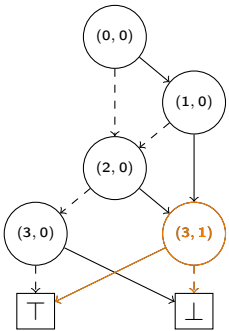
(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$

Seek	Sum	Result
$(3,1)$	1	2

Priority Queue: Q_{count}:

[

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



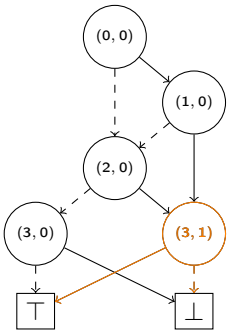
(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$

Seek	Sum	Result
$(3, 1)$	3	2

Priority Queue: Q_{count}:

[

]



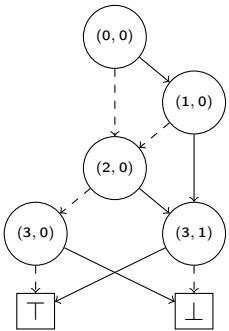
(a) $(x_0 \wedge x_1 \wedge x_3) \vee (x_2 \oplus x_3)$

Seek	Sum	Result
$(3, 1)$	3	5

Priority Queue: Q_{count}:

[

]



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

Result

5

Priority Queue: Q_{count}:

[

]

Steffan Christ Sølvsten

soelvsten@cs.au.dk

@ssoelvsten

Adiar

github.com/ssoelvsten/adiar

ssoelvsten.github.io/adiar

AARHUS
UNIVERSITY