

Adiar 1.1 : Zero-suppressed Decision Diagrams in External Memory

Steffan Christ Sølvesten and Jaco van de Pol

NFM 2023









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`

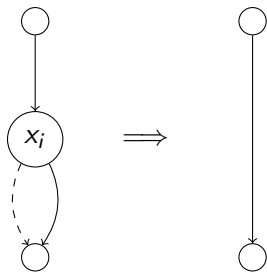
Adiar

Zero-suppressed Decision Diagrams
in External Memory

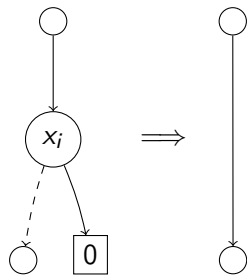
`github.com/ssoelvsten/adiar`







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



ZDD: $A \subseteq \mathbb{B}^n$

```
bdd bdd_apply(bdd f, bdd g, bool_op o)
```

```
bdd bdd_apply(bdd f, bdd g, bool_op o)
```

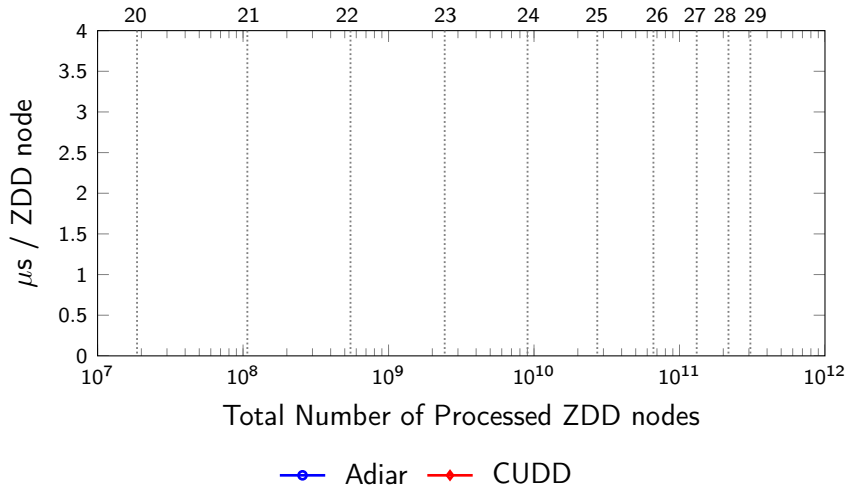
```
zdd zdd_binop(zdd A, zdd B, bool_op o)
```

```
bdd bdd_apply(bdd f, bdd g, bool_op o)  {  
    return prod2<bdd_policy>(f, g, o);  
}
```

```
zdd zdd_binop(zdd A, zdd B, bool_op o)  {  
    return prod2<zdd_policy>(A, B, o);  
}
```

```
bdd bdd_apply(bdd f, bdd g, bool_op o)  {  
    return prod2<bdd_policy>(f, g, o);  
}
```

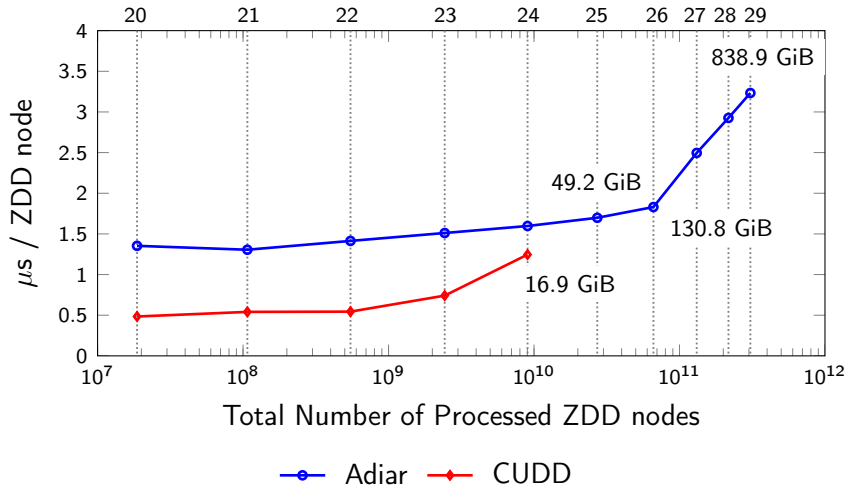
```
zdd zdd_binop(zdd A, zdd B, bool_op o)  {  
    return prod2<zdd_policy>(A, B, o);  
}
```

Running time for *3D Tic-Tac-Toe* with 300 GiB of RAM.



Running time for *3D Tic-Tac-Toe* with 300 GiB of RAM.



Running time for *3D Tic-Tac-Toe* with 300 GiB of RAM.

Done

BDD ZDD

Doable

MTBDD

LDD

QMDD

Done

BDD

ZDD

(K)FDD

Tagged/Chained BDD

Open

Clock DD

MDD

Doable

MTBDD

LDD

QMDD

Done

BDD

ZDD

(K)FDD

Tagged/Chained BDD

Steffan Christ Sølvsten

✉ soelvsten@cs.au.dk

🌐 ssoelvsten.github.io

Adiar

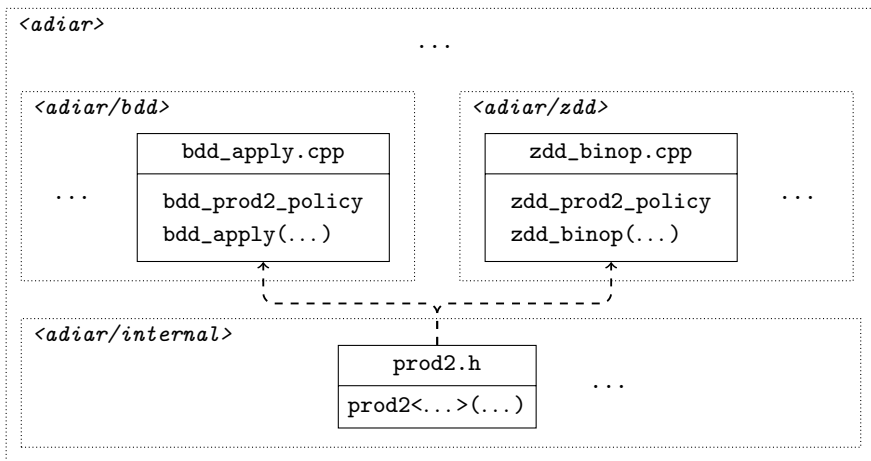
🔗 github.com/ssoelvsten/adiar

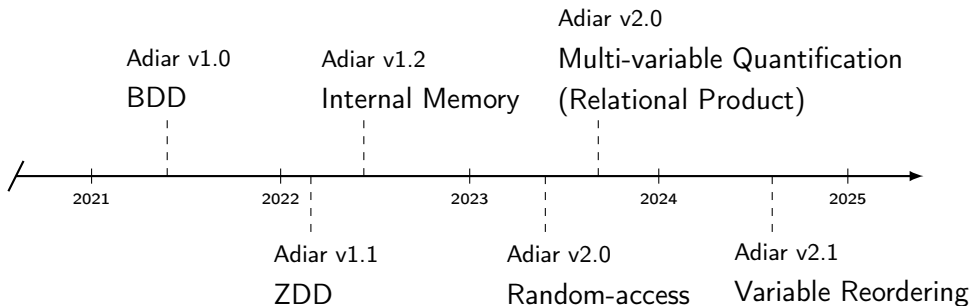
📖 ssoelvsten.github.io/adiar

Function	Operation Semantics
ZDD Constructors	
zdd_empty()	\emptyset
zdd_null()	$\{\emptyset\}$
zdd_singleton(var)	$\{x_{var}\}$
zdd_vars(vars)	$\{\bigcup_{i \in vars} \{x_i\}\}$
zdd_singletons(vars)	$\{\{x_i\} \mid i \in vars\}$
zdd_powerset(vars)	$\mathcal{P}(vars)$
zdd_sized_set(vars, k, \odot)	$\{s \in \mathcal{P}(vars) \mid s \odot k\}$
ZDD Manipulation	
zdd_binop(A, B, \otimes)	$\{x \mid x \in A \otimes x \in B\}$
zdd_change($A, vars$)	$\{(a \setminus vars) \cup (vars \setminus a) \mid a \in A\}$
zdd_complement(A, dom)	$\mathcal{P}(dom) \setminus A$
zdd_expand($A, vars$)	$\bigcup_{a \in A} \{a \cup v \mid v \in \mathcal{P}(vars)\}$
zdd_offset($A, vars$)	$\{a \in A \mid vars \cap a = \emptyset\}$
zdd_onset($A, vars$)	$\{a \in A \mid vars \subseteq a\}$
zdd_project($A, vars$)	$\bigcup_{a \in A} \{a \cap vars\}$

Function	Operation Semantics
Counting	
zdd_size(A)	$ A $
zdd_nodecount(A)	# ZDD Nodes in A
zdd_varcount(A)	# Non-empty Levels in A
Predicates	
zdd_equal(A, B)	$A = B$
zdd_unequal(A, B)	$A \neq B$
zdd_subseteq(A, B)	$A \subseteq B$
zdd_disjoint(A, B)	$A \cap B = \emptyset$
Set elements	
zdd_contains(A, a)	$a \in A$
zdd_minelem(A)	$a \in A \text{ s.t. } \forall a' \in A. a \leq a'$
zdd_maxelem(A)	$a \in A \text{ s.t. } \forall a' \in A. a' \leq a$
Conversion	
zdd_from(f, dom)	$\{x \in \mathcal{P}(dom) \mid f(x) = \top\}$
bdd_from(A, dom)	$\vec{x} : \mathcal{P}(dom) \mapsto \vec{x} \in A$

Operations provided by Adiar in <adiar/zdd.h>.







(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} :

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

]



(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} :

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

]



(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} :

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

]



(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} :

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



(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} :

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



(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} :

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



(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)	1	0

Priority Queue: Q_{count} :

[

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

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

]



(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} :

[

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



(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} :

[

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



(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} :

[

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

[

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

[

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



(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} :

[

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



(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} :

[

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



(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} :

[

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



(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} :

[

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



(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.github.io

Adiar

🔗 github.com/ssoelvsten/adiar

📖 ssoelvsten.github.io/adiar