

# I/O-efficient Symbolic Model Checking

---

**Steffan Christ Sølvesten**, Jaco van de Pol

31<sup>st</sup> of August, 2022

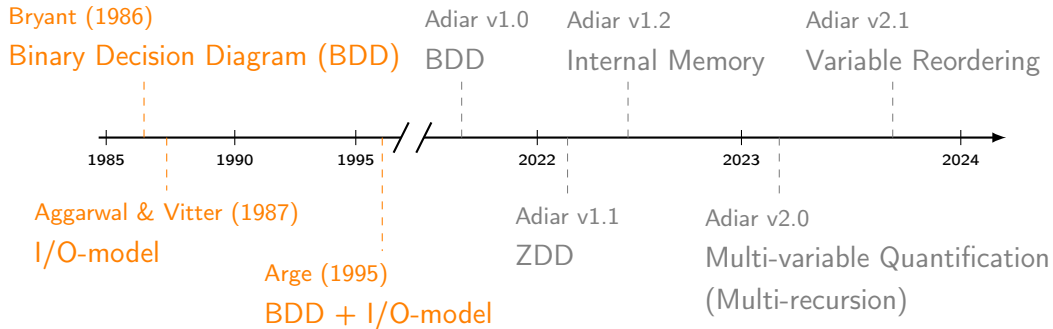




# Adiar

[github.com/ssoelvsten/adiar](https://github.com/ssoelvsten/adiar)







## 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

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

**Theorem (Aggarwal and Vitter '87)**

*$N$  elements can be sorted in  $\Theta(\text{sort}(N))$  I/Os.*

**Theorem (Arge '95)**

*$N$  elements can be inserted in and extracted from a Priority Queue in  $\Theta(\text{sort}(N))$  I/Os.*





# Bryant '86 : Binary Decision Diagram



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



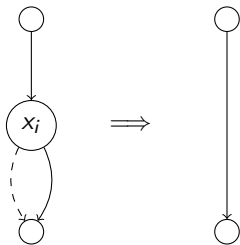
**(b)**  $\neg(x_0 \oplus x_2 \vee x_3) : x_2 \wedge x_3$

Examples of (Reduced Ordered) Binary Decision Diagrams.

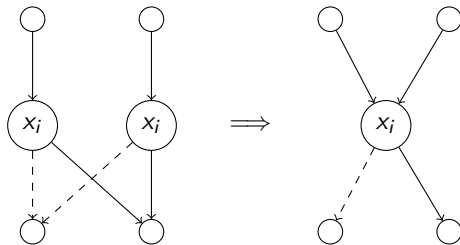
# Bryant '86 : Binary Decision Diagram

## 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.



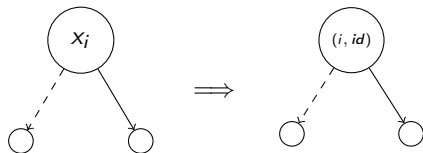


## Adiar v1.0 : BDD





## 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



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



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

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:  $Q_{app:1}$ :

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



**(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:  $Q_{app:1}$ :

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



]

(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:  $Q_{app:1}$ :

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



**(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:  $Q_{app:1}$ :

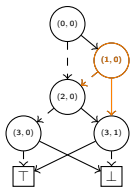
[  $(0, 0) \xrightarrow{\top} ((1, 0), (2, 1))$  ,  
 $(0, 0) \xrightarrow{\perp} ((2, 0), (2, 0))$  ,  
 $(1, 0) \xrightarrow{\perp} ((2, 0), (2, 1))$  ,  
 $(1, 0) \xrightarrow{\top} ((3, 1), (2, 1))$  ,

]



(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:  $Q_{app:1}$ :

[

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

]

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

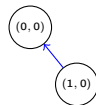
Priority Queue:  $Q_{app:1}$ :

[

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

]

Output:



(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:  $Q_{app:1}$ :

[

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

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

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

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

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:  $Q_{app:1}$ :

[

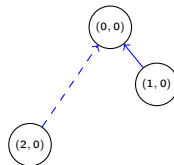
$(1, 0) \xrightarrow{\perp} ((2, 0), (2, 1))$  ,

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

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

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

Output:



(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:  $Q_{app:1}$ :

[

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

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

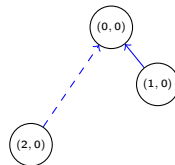
$(2, 0) \xrightarrow{\perp} ((3, 0), \top)$  ]

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

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

]

Output:

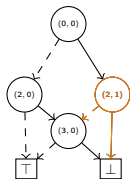


(c)  $(a) \wedge (b)$

# 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:  $Q_{app:1}$ :

[

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

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

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

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

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

]

Output:



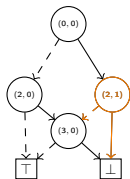
(c)  $(a) \wedge (b)$



# 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:  $Q_{app:1}$ :

[

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

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

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

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

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

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

]

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



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

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

[

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

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

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

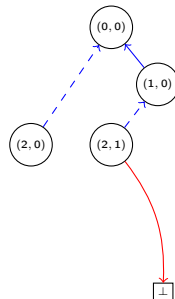
$(2, 0) \xrightarrow{\perp} ((3, 0), \top)$  ]

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

[

]

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



(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:  $Q_{app:1}$ :

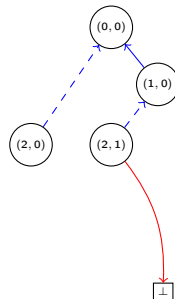
[

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

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

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

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

[

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

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

[

]

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



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

# Adiar v1.0 : BDD



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



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

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

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



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

# Adiar v1.0 : BDD



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



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

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

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

[

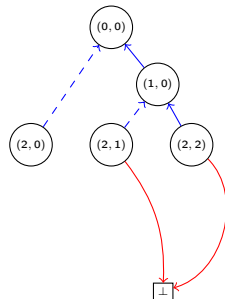
$(2, 1) \xrightarrow{\perp} ((3, 0), (3, 0))$  ,  
 $(2, 0) \xrightarrow{\top} ((3, 1), (3, 0))$  ,  
 $(2, 2) \xrightarrow{\perp} ((3, 1), (3, 0))$  ,  
 $(2, 0) \xrightarrow{\perp} ((3, 0), \top)$  ]

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

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

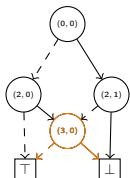


(c)  $(a) \wedge (b)$

# Adiar v1.0 : BDD



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



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

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

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

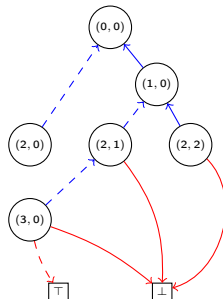
[

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

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

[

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



(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:  $Q_{app:1}$ :  
 [

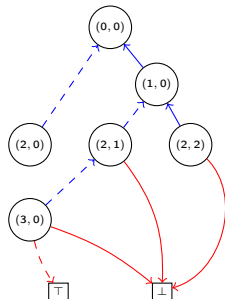
$(2, 0) \xrightarrow{\top} ((3, 1), (3, 0))$  ,  
 $(2, 2) \xrightarrow{\perp} ((3, 1), (3, 0))$  ,  
 $(2, 0) \xrightarrow{\perp} ((3, 0), \top)$  ]

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

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:

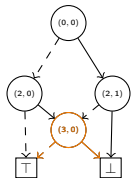


(c)  $(a) \wedge (b)$

# 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:  $Q_{app:2}$ :

[

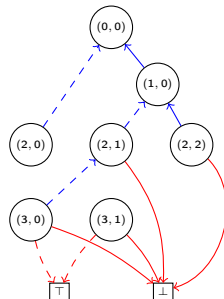
$(2, 0) \xrightarrow{\top} ((3, 1), (3, 0)) \quad (\top, \perp)$

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

,

]

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:  $Q_{app:2}$ :

[

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

,

]

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



(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:  $Q_{app:2}$ :

[

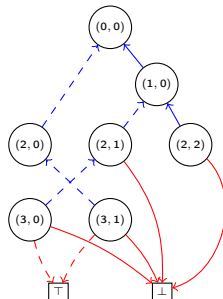
$(2, 0) \xrightarrow{\top} ((3, 1), (3, 0)) \quad (\top, \perp)$

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

,

]

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

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

[

]

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

[

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

,

]

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



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

# Adiar v1.0 : BDD



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



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

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

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

[

]

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

[

]

Output:

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



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



# Adiar v1.0 : BDD



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



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

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

[

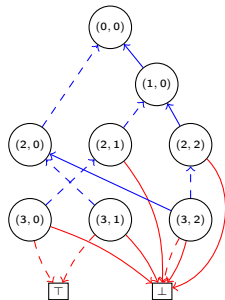
Priority Queue:  $Q_{app:2}$ :

[

]

]

Output:



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

## Adiar v1.0 : BDD



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

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

# Adiar v1.0 : BDD



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

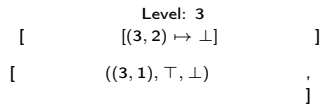
[ Level: 3  
 $[(3, 2) \mapsto \perp]$  ]

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

# Adiar v1.0 : BDD



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

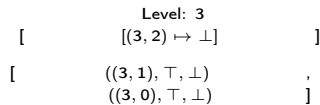


(d)  $(a) \wedge (b)$  reduced

# Adiar v1.0 : BDD



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



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

# Adiar v1.0 : BDD



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

Level: 3

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

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



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

# Adiar v1.0 : BDD



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

Level: 3  
 $[ \quad [(3, 2) \mapsto \perp] \quad ]$   
 $[ \quad [ (3, 1) \mapsto (3, \max) ] \quad , \quad [ (3, 0) \mapsto (3, \max) ] \quad ]$

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:  $Q_{red}$ :

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

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

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

]

Level: 3

[

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

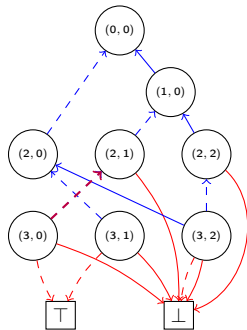
]

Output:



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

# Adiar v1.0 : BDD



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

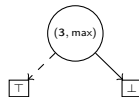
Priority Queue:  $Q_{red}$ :

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

Level: 3

[  
 [  
 ,  
 ]  
 ]

Output:



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

# Adiar v1.0 : BDD



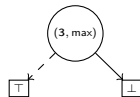
(c)  $(a) \wedge (b)$

Priority Queue:  $Q_{red}$ :

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

[  
 Level: 2  
 $[(2, 2) \mapsto \perp]$   
 ]

Output:



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

# Adiar v1.0 : BDD



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

Priority Queue:  $Q_{red}$ :

[

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

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

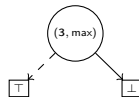
]

Level: 2

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

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

Output:



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

# Adiar v1.0 : BDD



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

Priority Queue:  $Q_{red}$ :

[

]

Level: 2

[

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

]

[

$((2, 1), (3, \max), \perp)$

,

$((2, 0), (3, \max), \perp)$

]

Output:



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

# Adiar v1.0 : BDD



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

Priority Queue:  $Q_{red}$ :

[

]

Level: 2

[

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

]

[

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

,

$((2, 0), (3, \max), \perp)$

]

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



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

# Adiar v1.0 : BDD



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

Priority Queue:  $Q_{red}$ :

[

]

Level: 2

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

[

]

[

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

,

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

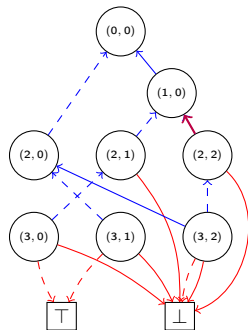
]

Output:



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

# Adiar v1.0 : BDD



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

Priority Queue:  $Q_{red}$ :

[

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

]

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:  $Q_{red}$ :

[

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

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

]

Level: 2

[

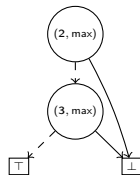
]

[

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

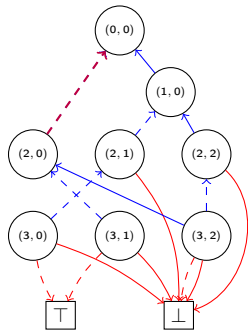
]

Output:



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

# Adiar v1.0 : BDD



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

Priority Queue:  $Q_{red}$ :

[

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

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

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

Level: 2

[

[

]

,  
]

Output:



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

# Adiar v1.0 : BDD



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

Priority Queue:  $Q_{red}$ :

[

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

Level: 1

[

]

[

$((1, 0), (2, \max), \perp)$

]

Output:



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

# Adiar v1.0 : BDD



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

Priority Queue:  $Q_{red}$ :

[

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

Level: 1

[

]

[

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

]

Output:  
 $((1, \max), (2, \max), \perp)$



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

# Adiar v1.0 : BDD



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

Priority Queue:  $Q_{red}$ :

[

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

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

Level: 1

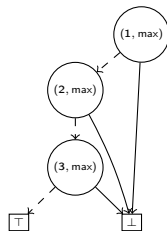
[

]

[

]

Output:



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

# Adiar v1.0 : BDD



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

Priority Queue:  $Q_{red}$ :

[

]

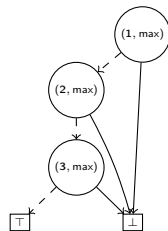
Level: 0

[

]

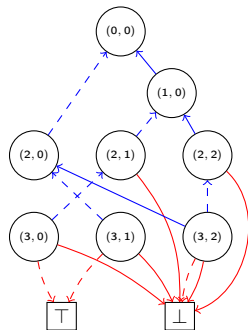
[  $((0, 0), (2, \max), (1, \max))$  ]

Output:



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

# Adiar v1.0 : BDD



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

Priority Queue:  $Q_{red}$ :

[

]

Level: 0

[

]

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

]

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



—●— BuDDy —●— CUDD —■— Sylvan —●— Adiar

Minimal running time for the *Queens* problems.

## Adiar v1.0 : BDD



—●— BuDDy —◆— CUDD —■— Sylvan —●— Adiar

Minimal running time for the *Queens* problems.

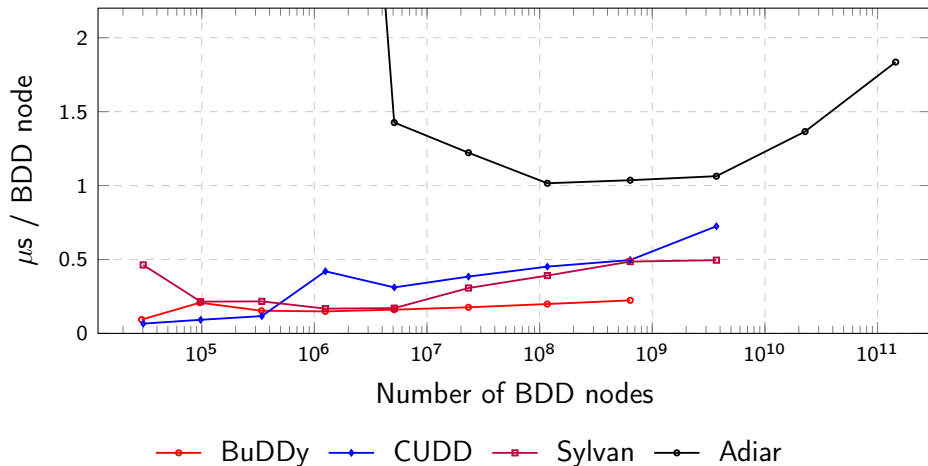
## Adiar v1.0 : BDD



—●— BuDDy    —◆— CUDD    —■— Sylvan    —●— Adiar

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



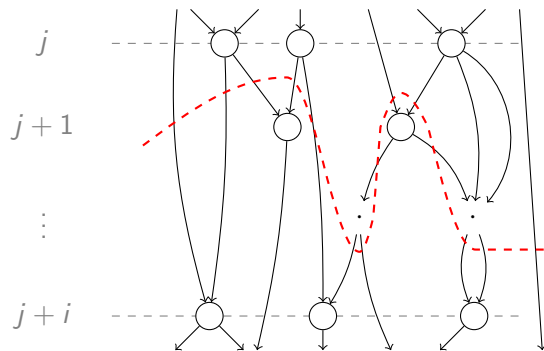
## Adiar v1.2 : Internal Memory



## Adiar v1.2 : Internal Memory

### Definition (i-level cut)

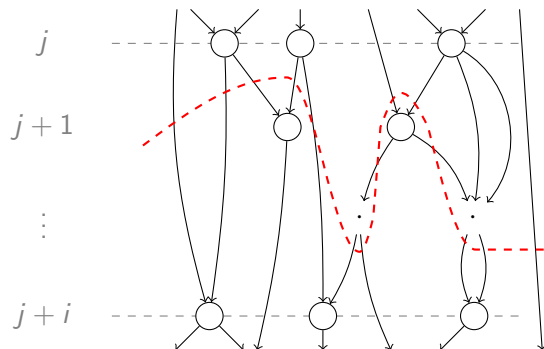
$\mathcal{L}$



# Adiar v1.2 : Internal Memory

## Definition (i-level cut)

$\mathcal{L}$

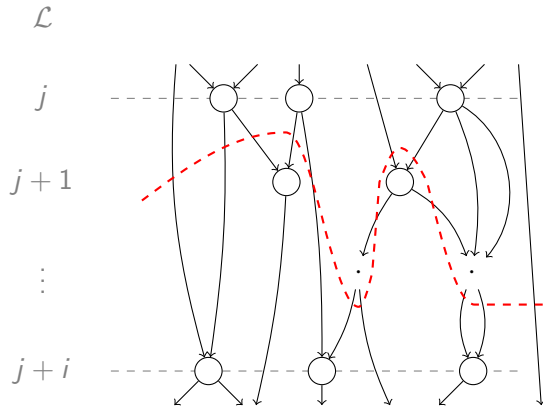


## Lemma

*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.*



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

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

## Adiar v2.0 : Multi-variable Quantification

```
1 exists(f, V)
2   if f =  $\perp$   $\vee$  f =  $\top$ 
3       then f
4   else if  $V \cap \{i \in \mathbb{N} \mid i \geq \text{top}(f)\} = \emptyset$ 
5       then f
6   else if top(f)  $\notin V$ 
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  $f = \perp \vee f = \top$ 
3       then f
4   else if  $V \cap \{i \in \mathbb{N} \mid i \geq \text{top}(f)\} = \emptyset$ 
5       then f
6   else if  $\text{top}(f) \notin V$ 
7       then Node {  $\text{top}(f)$ , 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 =  $\perp$   $\vee$  f =  $\top$ 
3     then f
4   else if  $V \cap \{i \in \mathbb{N} \mid i \geq \text{top}(f)\} = \emptyset$ 
5     then f
6   else if  $\text{top}(f) \notin V$ 
7     then Node {  $\text{top}(f)$ , 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



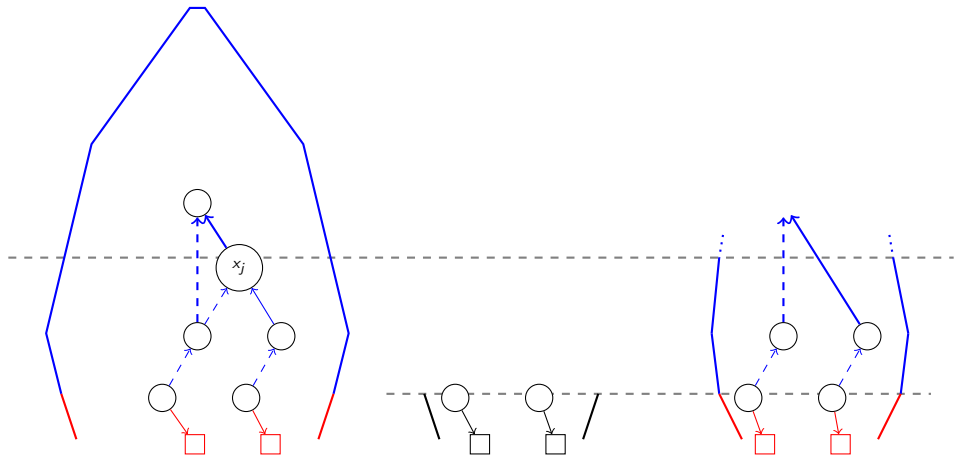
## Adiar v2.0 : Multi-variable Quantification



## Adiar v2.0 : Multi-variable Quantification



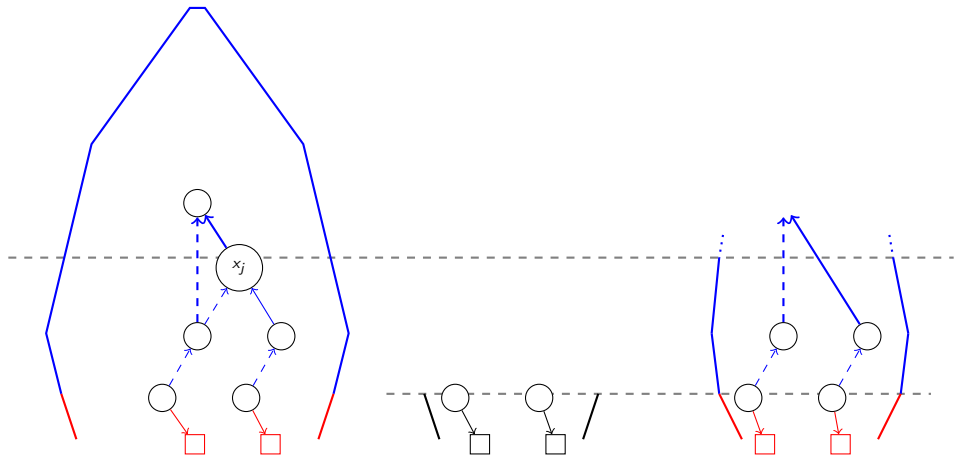
## Adiar v2.0 : Multi-variable Quantification



## Adiar v2.0 : Multi-variable Quantification

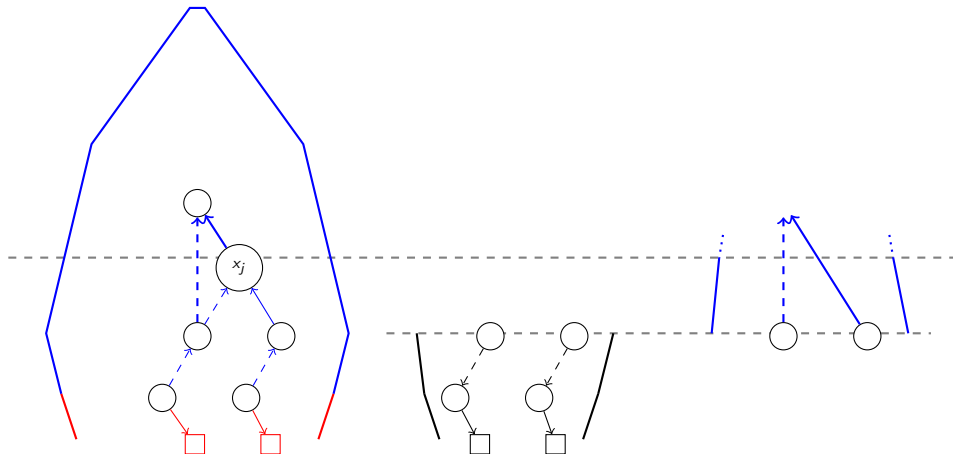


## Adiar v2.0 : Multi-variable Quantification





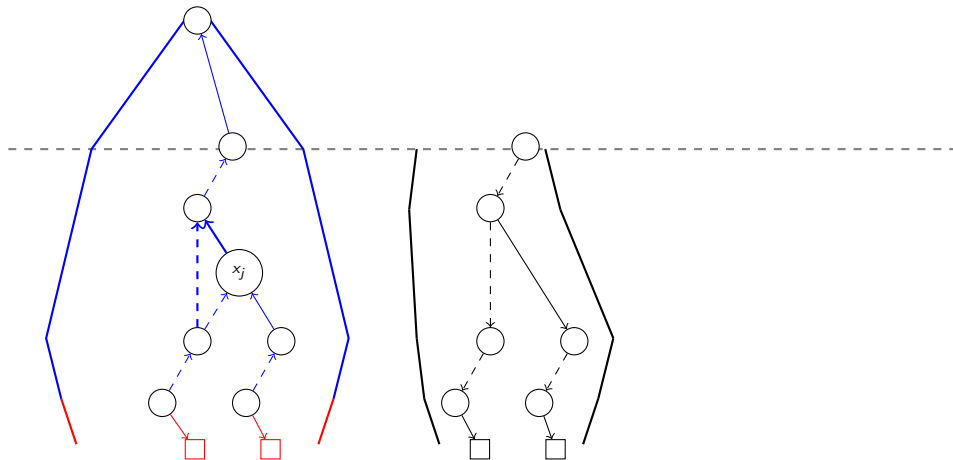
## Adiar v2.0 : Multi-variable Quantification



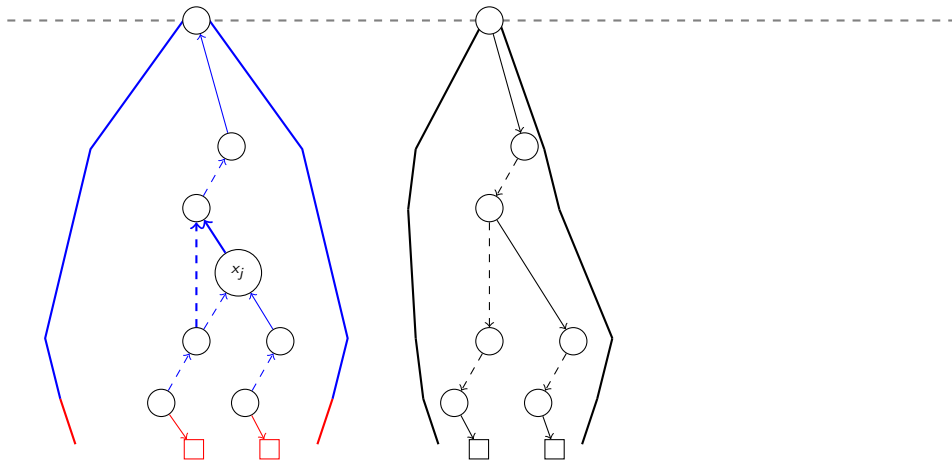
## Adiar v2.0 : Multi-variable Quantification



## Adiar v2.0 : Multi-variable Quantification



## Adiar v2.0 : Multi-variable Quantification



## Adiar v2.0 : Multi-variable Quantification





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

## 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.

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



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

## 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.

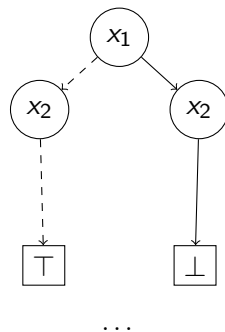


## Adiar v2.1 : Variable Reordering

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

```
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.



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

## 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  $f_\pi$  with variable ordering  $\pi$  and remapping of variables  $m : \mathbb{N} \rightarrow \mathbb{N}$ , construct  $f'_\pi \equiv f_\pi[x/m(x)]$ .*

## Adiar v2.1 : Variable Reordering

### **Problem (Variable Replacement)**

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

### **Problem (Static Variable Reordering)**

*Given BDD  $f_\pi$  with variable ordering  $\pi$  and another variable ordering  $\pi'$ , construct  $f_{\pi'} \equiv f_\pi$ .*

## Adiar v2.1 : Variable Reordering

### Problem (Variable Replacement)

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

### Problem (Static Variable Reordering)

Given BDD  $f_\pi$  with variable ordering  $\pi$  and another variable ordering  $\pi'$ ,  
construct  $f_{\pi'} \equiv f_\pi$ .

### Problem (Dynamic Variable Reordering)

Given BDD  $f_\pi$  with variable ordering  $\pi$ ,  
find  $\pi'$  and construct  $f_{\pi'} \equiv f_\pi$  such that  $|f_{\pi'}|$  is minimal.



