VDS Class Project

ROBDD Construction Example

Construction of Reduced Ordered BDD for the following function:

$$f = (a + b).c.d$$

We need different representation for BDD construction:

$$f = and\big(or(a,b), and(c,d)\big)$$

Variables: a, b, c, d

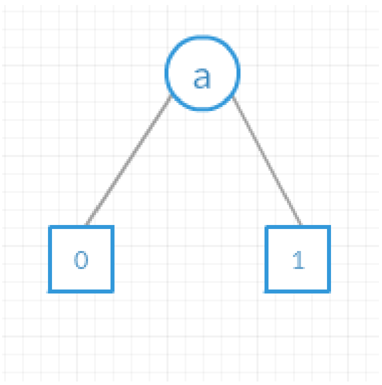
First of all, Order of variables should be defined added to the table.

$$a > b > c > d$$

1st Step: add $'0'$ and $'1'$ to the table as terminal nodes. (It should be done by constructor)

2nd Step: add variables to the table

For example for variable a => $ite(a, 1,0)$



3rd Step: add $or(a, b)$ to the table

$ite(a, 1, b) <=> ite(ID2, ID1, ID3)$

Algorithm: terminal case? NO

Find top variable: compare IDS of $a, b, '1'$

ID$_{'1'}$ is constant, ID$_a$=2, ID$_b$=3 => a is top var.

$T = ite(ID1, ID1, ID3)$ returns $ID1$

$F = ite(ID0, ID1, ID3)$ returns $ID3$

Add new node to unique table $\{a, ID1, ID3\}$

4th Step: add $and(c,d)$ to the table

$ite(c,d,0) <=> ite(ID4, ID5, ID0)$

Top Variable c, no terminal

$T = ite(ID1, ID5, ID0)$ returns $ID5$

$F = ite(ID0, ID5, ID0)$ returns $ID0$

Add new node to unique table $\{c, ID5, ID0\}$

Now, we know that $or(a,b)$ is ID6 and $and(c,d)$ is ID7

5th Step: add $and\big(or(a,b), and(c,d)\big)$ to the table. ($and(ID6, ID7)$)

Top Variable: $ID6 \to a\,, ID7 \to c => topVar \to a$

$ite(ID6, ID7, ID0)$

$T = ite(ID1, ID7, ID0)$ returns $ID7$

$F = ite(ID3, ID7, ID0)$ -> not a terminal case

Top Variable: b

$T = ite(ID1, ID7, ID0)$ returns $ID7$

$F = ite(ID0, ID7, ID0)$ returns $ID0$

Add $\{b, ID7, ID0\}$ to the table: $ID8$

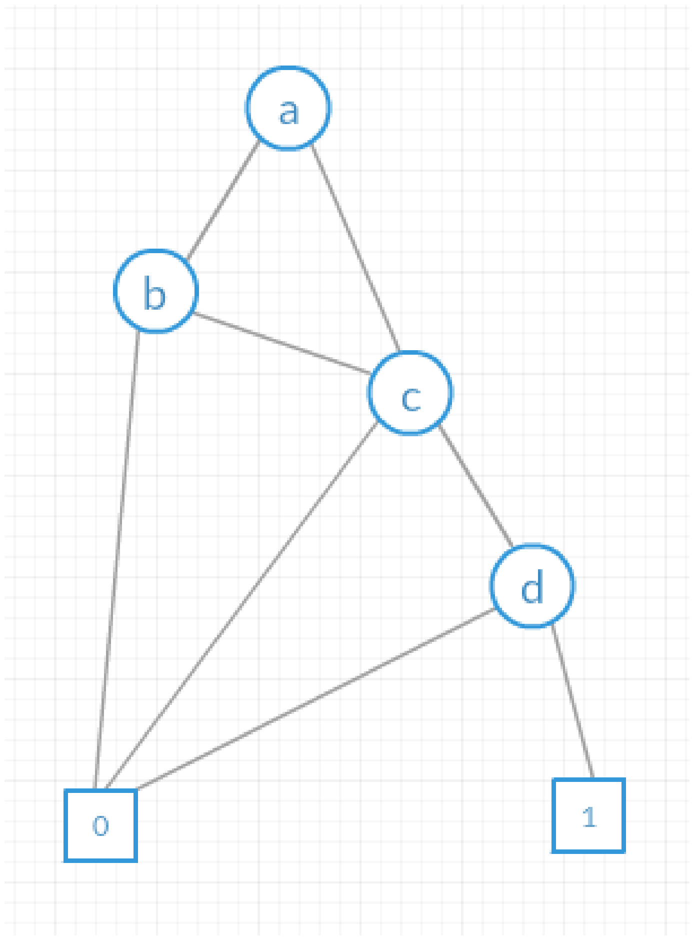
Add new node to unique table $\{a, ID7, ID8\}$

Final unique table should look like this:

BDD ID	Label	High	Low	Top Var
0	“0”	0	0	0
1	“1”	1	1	1
2	“a”	1	0	a
3	“b”	1	0	b
4	“c”	1	0	c
5	“d”	1	0	d
6	“or”	1	3	a
7	“and”	5	0	c
8		7	0	b
9	“f”	7	8	a

Final Unique Table

Final BDD Result:



Co-factoring Algorithm :

```
cofactorTrue(f,x)
    If(terminal)
        return $f$
    If(f.top == x)
        return $f$.high
    Else
        $T = cofactorTrue(f.high, x)$
        $F = cofactorTrue(f.low, x)$
        return $ite(f.topVar, T, F)$
```

Terminal case:

- f is constant.
- x is constant
- f.top > x

```
cofactorFalse(f,x)
    If(terminal)
        return $f$
    If(f.top == x)
        return $f$.low
    Else
        $T = cofactorFalse(f.high, x)$
        $F = cofactorFalse(f.low, x)$
        return $ite(f.topVar, T, F)$
```

Terminal case:

- f is constant.
- x is constant
- f.top > x