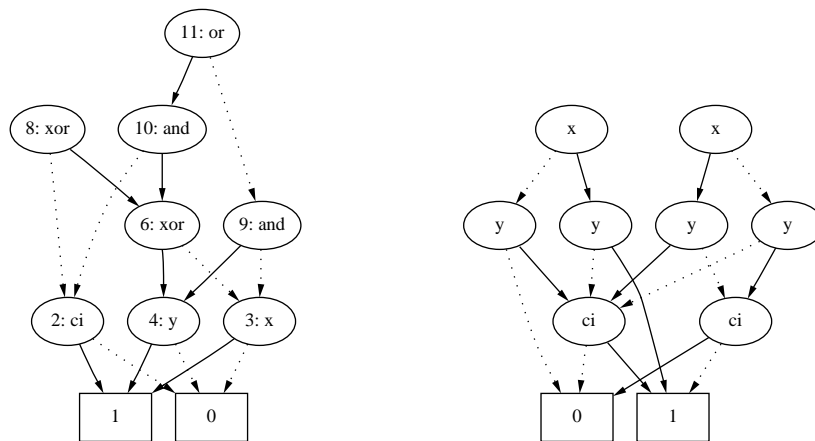


# An Introduction to Binary Decision Diagrams

## Addendum and Erratum

Henrik Reif Andersen



An (RO)BED and a corresponding (RO)BDD.

Lecture notes for 49285 Advanced Algorithms E98, November 1998.

E-mail: [hra@it.dtu.dk](mailto:hra@it.dtu.dk). Web: <http://www.it.dtu.dk/~hra>

Department of Information Technology, Technical University of Denmark  
Building 344, DK-2800 Lyngby, Denmark.

## 9 Constructing ROBDDs from Boolean Expressions

The algorithms from section 4 can be combined into a single algorithm UPALL for converting Boolean expressions into ROBDDs. This algorithm is given in figure 1. It is called *upall*, since viewing the Boolean expression as a directed acyclic graph of Boolean operators over a range of variables, it can be seen as moving these variables *up* past the operators. Viewing a Boolean expression as a directed acyclic graph of operators, is a special case of what is called a *Boolean Expression Diagram*. A Boolean Expression Diagram is a BDD in which apart from variable vertices also operator vertices are allowed. The figure on the front page shows an example of an (RO)BED and the corresponding (RO)BDD after conversion with UPALL. More details on BEDs can be found in [AH97] or from the Boolean Expression entry on the URL: <http://www.it.edu/people/hra/>.

```
UPALL[T, H](t) =
1: case
2:   t = 0 :           return 0
3:   t = 1 :           return 1
4:   t = xi :         return MK(i, 0, 1)
5:   t = ¬t' :        return APPLY(not, UPALL(t'), 0)
6:   t = t1 op t2 : return APPLY(op, UPALL(t1), UPALL(t2))
7:   t = ∃xi.t' :     return APPLY(∨, RESTRICT(UPALL(t'), i, 0),
                                     RESTRICT(UPALL(t'), i, 1))
8:   t = t'[xj := b] : return RESTRICT(UPALL(t'), j, b)
```

Figure 1: The algorithm UPALL combining the algorithms from section 4 for converting a Boolean expression into an ROBDD.

## Erratum

- P. 22, figure 13: In line 4 and 5 the call to *res* should be removed such that the lines now read:

```
4:   else (* var(u) = j *) if b = 0 then return low(u)
5:   else (* var(u) = j, b = 1 *) return high(u)
```

- P. 22 l. 11: *varu* must be changed to *var(u)* twice in this line and again in line 14.
- The numbers in the figure on p. 21 is a bit misleading. Below is an improved version of the figure.

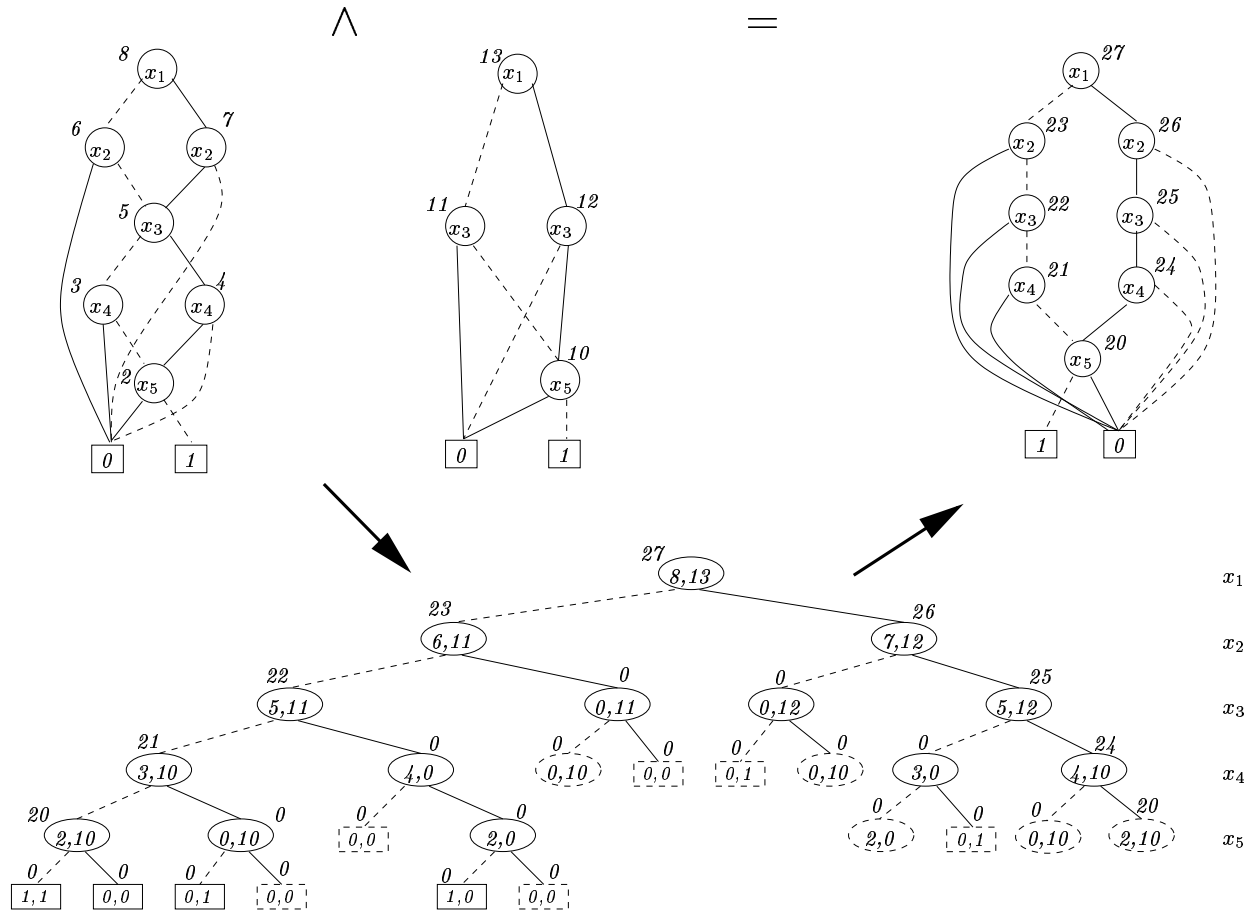


Figure 12: An example of applying the algorithm APPLY for computing the conjunction of the two ROBDDs shown at the top left. The result is shown to the right. Below the tree of arguments to the recursive calls of APP. Dashed nodes indicate that the value of the node has previously been computed and is not recomputed due to the use of dynamic programming. The solid ellipses show calls that finishes by a call to MK with the variable index indicated by the variables to the right of the tree. The identity of the vertices returned by the calls to MK are shown next to the nodes.

## Acknowledgement

Thanks to Jakob Lichtenberg for constructive comments.

## References

- [AH97] Henrik Reif Andersen and Henrik Hulgaard. Boolean Expression Diagrams. In *Proceedings, Twelfth Annual IEEE Symposium on Logic in Computer Science*, pages 88–98, Warsaw, Poland, June 29–July 2 1997. IEEE Computer Society.