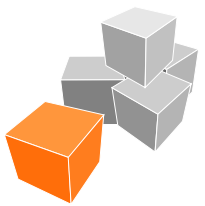
# Redundancy-free Residual Dispatch

## Using Ordered Binary Decision Diagrams for Efficient Dispatch

Andreas Sewe Christoph Bockisch Mira Mezini

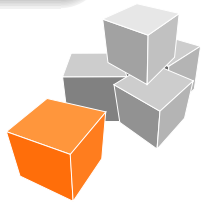
TECHNISCHE
UNIVERSITÄT
DARMSTADT

Seventh International Workshop on Foundations of Aspect-Oriented Languages, 2008



## Example Aspect

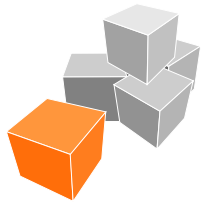
```
aspect SecurityMonitoring
{
   before(): call(void Connection.transmit(Text))
      && ((target(LocalConnection) && args(PlainText)
           && !cflow(call(void Log.debug())))
         || (!target(LocalConnection) && args(PlainText)))
   {
       throw new PolicyViolation("Cipher text required.");
   }
}
```



## Example Pointcut

```
call(void Connection.transmit(Text))
&& ((target(LocalConnection) && args(PlainText)
    && !cflow(call(void Log.debug())))
  || (!target(LocalConnection) && args(PlainText)))
```

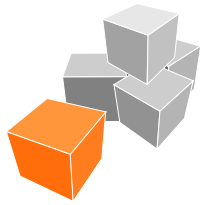
```
...
Connection connection =
  new RemoteConnection("www.st.informatik.tu-darmstadt.de");
Text text = new PlainText("Encrypt me!");
connection.transmit(text);
...
```



## Example Pointcut

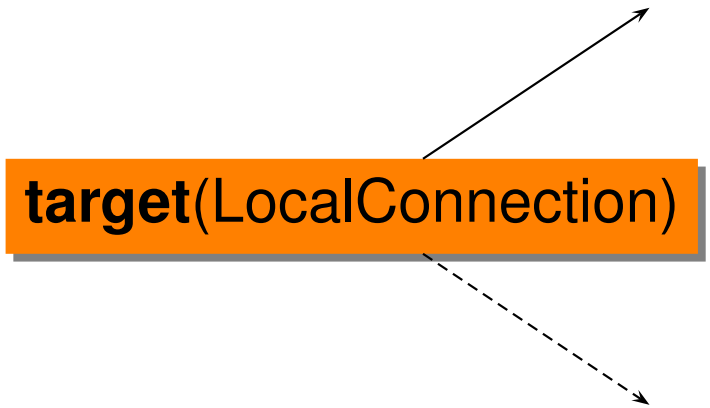
```
call(void Connection.transmit(Text))
&& ((target(LocalConnection) && args(PlainText)
    && !cflow(call(void Log.debug())))
  || (!target(LocalConnection) && args(PlainText)))
```

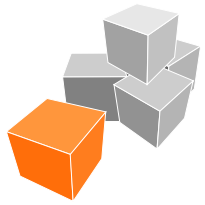
```
...
Connection connection =
  new RemoteConnection("www.st.informatik.tu-darmstadt.de");
Text text = new PlainText("Encrypt me!");
connection.transmit(text);
...
```



## Example Pointcut

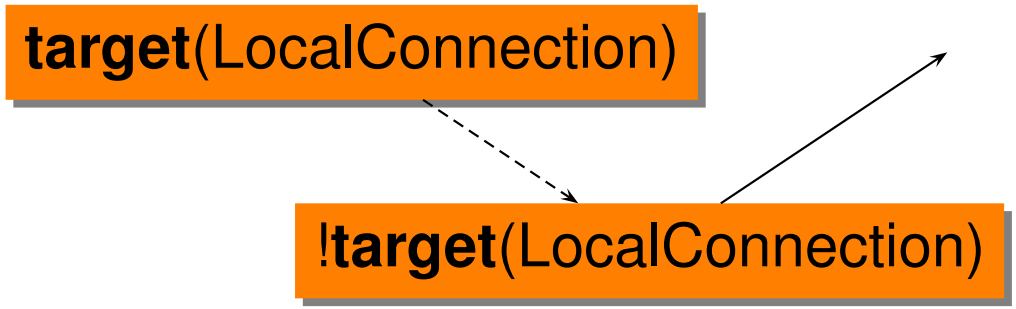
```
call(void Connection.transmit(Text))
&& ((target(LocalConnection) && args(PlainText)
    && !cflow(call(void Log.debug())))
  || (!target(LocalConnection) && args(PlainText)))
```

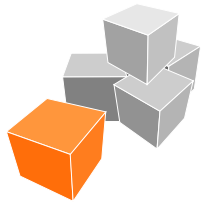


## Example Pointcut

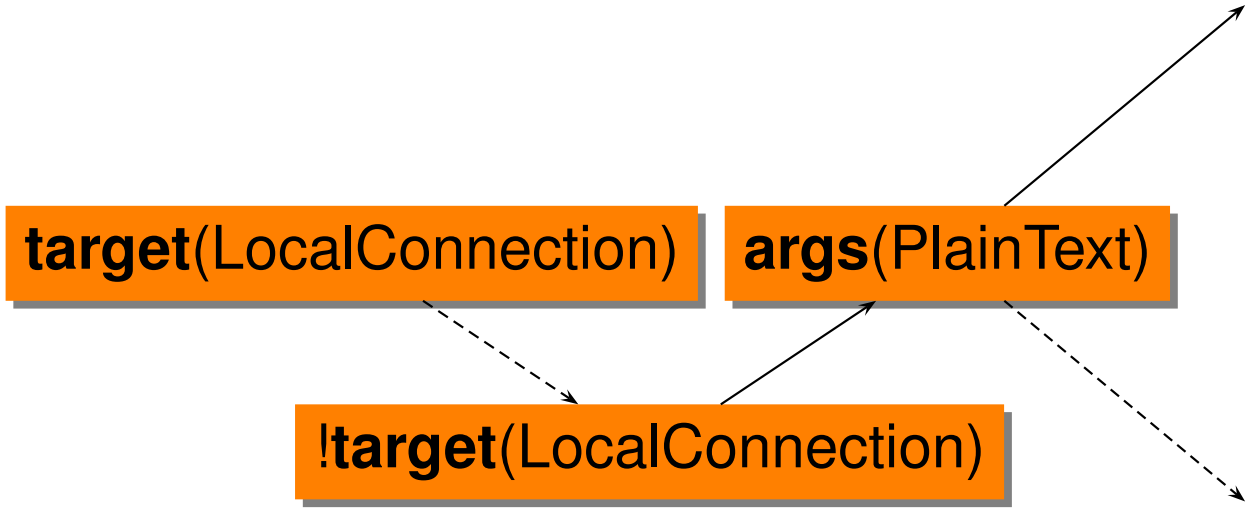
```
call(void Connection.transmit(Text))
&& ((target(LocalConnection) && args(PlainText)
    && !cflow(call(void Log.debug())))
  || (!target(LocalConnection) && args(PlainText)))
```

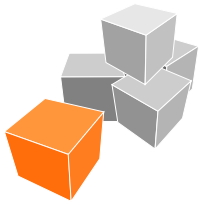


## Example Pointcut

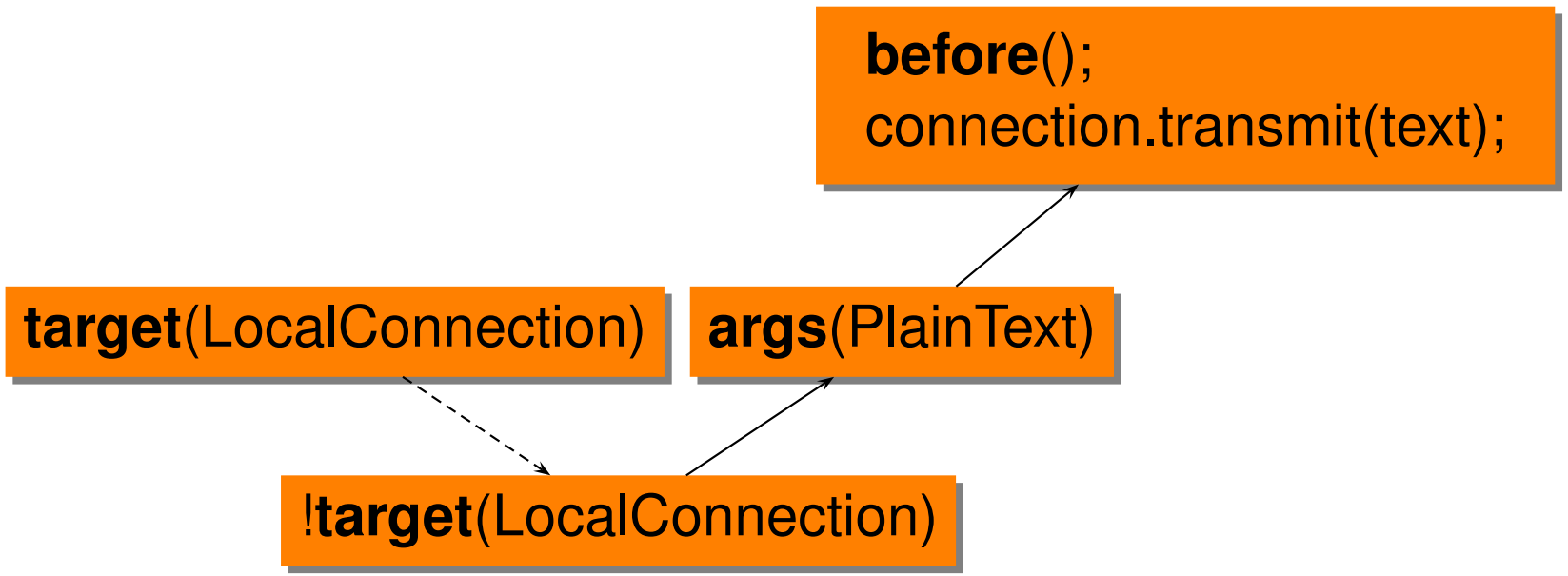
```
call(void Connection.transmit(Text))
&& ((target(LocalConnection) && args(PlainText)
    && !cflow(call(void Log.debug())))
  || (!target(LocalConnection) && args(PlainText)))
```

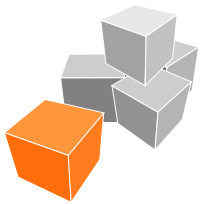


## Example Pointcut

```
call(void Connection.transmit(Text))
&& ((target(LocalConnection) && args(PlainText)
    && !cflow(call(void Log.debug())))
  || (!target(LocalConnection) && args(PlainText)))
```

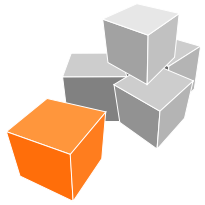


# From Pointcuts to Formulas

## Example Pointcut

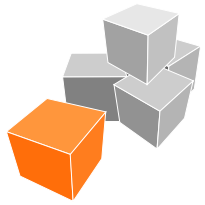
```
call(void Connection.transmit(Text))
&& ((target(LocalConnection) && args(PlainText)
    && !cflow(call(void Log.debug())))
  || (!target(LocalConnection) && args(PlainText)))
```



# From Pointcuts to Formulas

## Example Pointcut

```
call(void Connection.transmit(Text))
&& ((target(LocalConnection) && args(PlainText)
    && !cflow(call(void Log.debug())))
  || (!target(LocalConnection) && args(PlainText)))
```



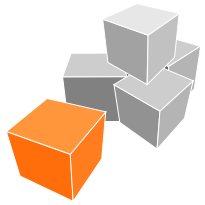
# From Pointcuts to Formulas

## Example Pointcut

```
call(void Connection.transmit(Text))
&& ((target(LocalConnection) && args(PlainText)
    && !cflow(call(void Log.debug())))
  || (!target(LocalConnection) && args(PlainText)))
```

## Example Formula

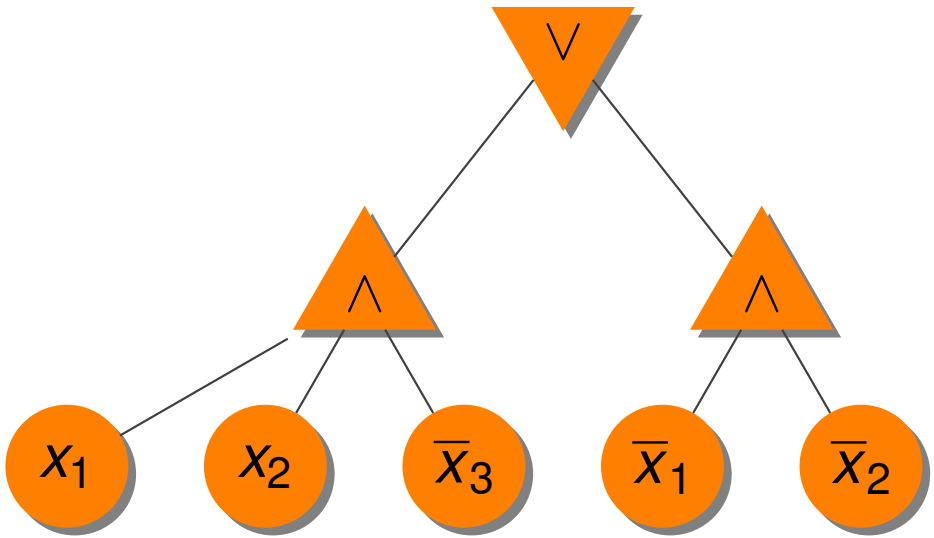
$$\phi = (x_1 \wedge x_2 \wedge \overline{x}_3) \vee (\overline{x}_1 \wedge x_2)$$

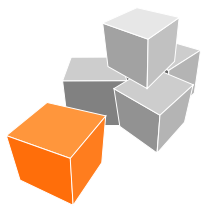


# From Formulas to Strategies

## Example Formula

$\phi = (x_1 \wedge x_2 \wedge \overline{x}_3) \vee (\overline{x}_1 \wedge x_2)$

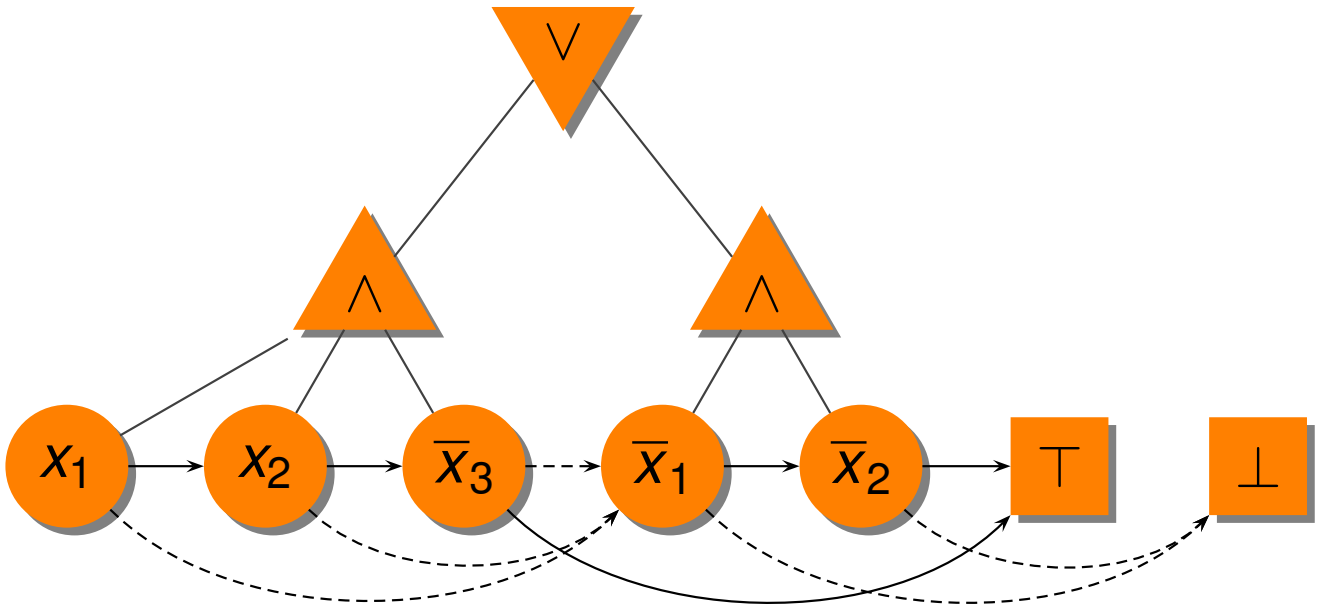


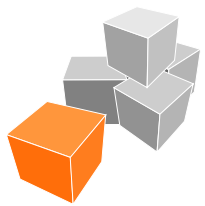



# From Formulas to Strategies

## Example Formula

$\phi = (x_1 \wedge x_2 \wedge \overline{x}_3) \vee (\overline{x}_1 \wedge x_2)$

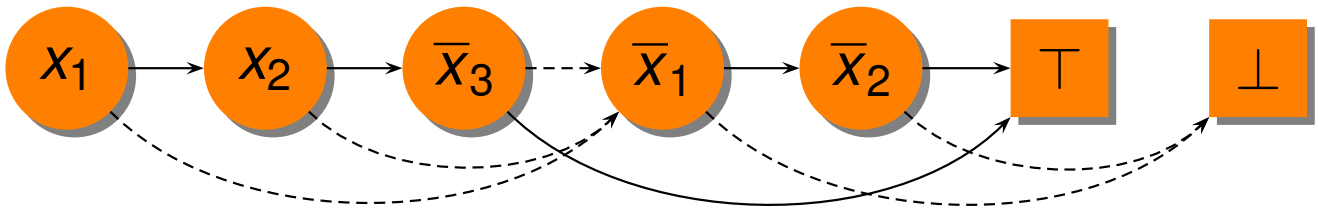


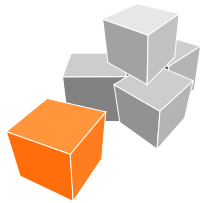



# From Formulas to Strategies

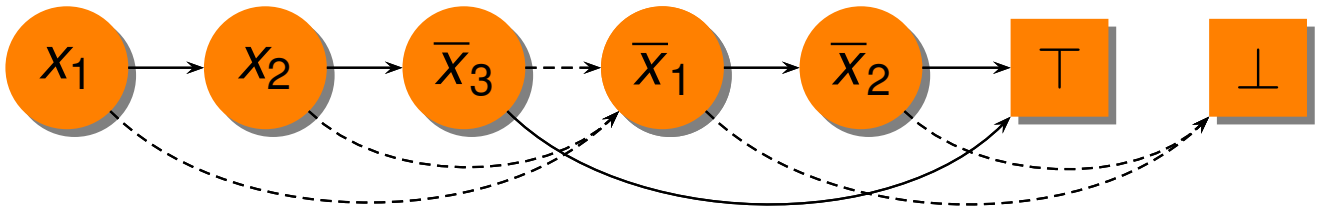
## Example Formula

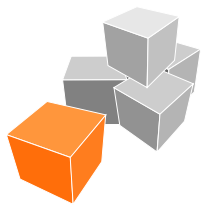
$\phi = (x_1 \wedge x_2 \wedge \overline{x}_3) \vee (\overline{x}_1 \wedge x_2)$



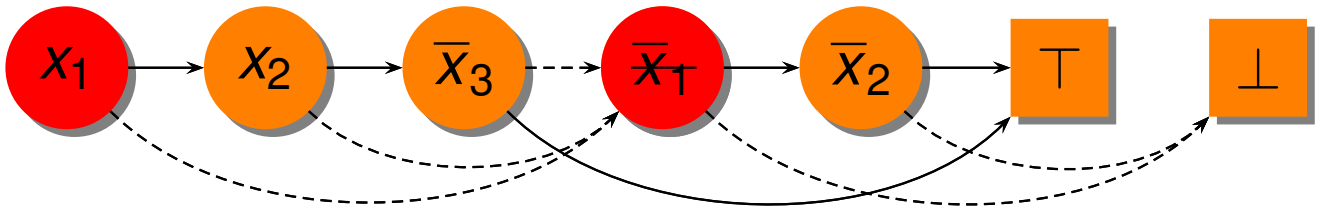



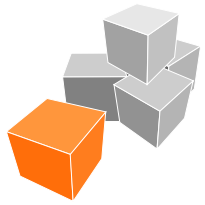
# Partial Redundancy Elimination



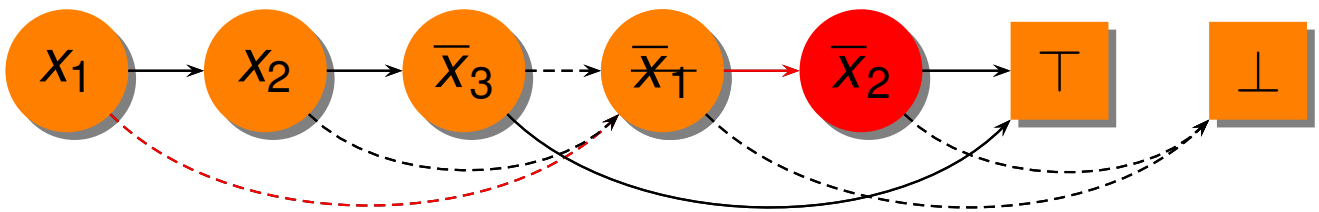



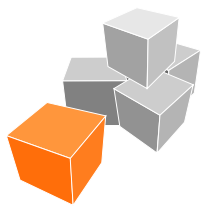
# Partial Redundancy Elimination



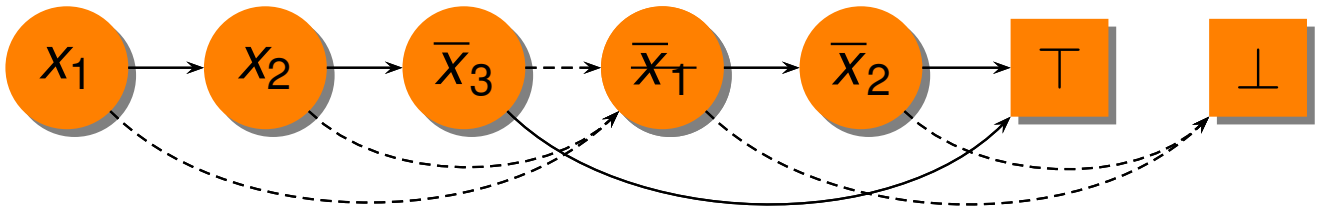



# Partial Redundancy Elimination

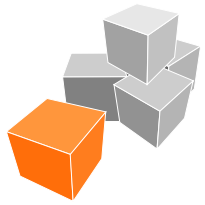


# Partial Redundancy Elimination




Formulas may prevent complete redundancy elimination.

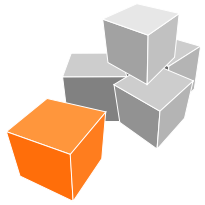


# Dispatch Functions

Residual dispatch at a join point shadow can be viewed as the evaluation of a Boolean function.

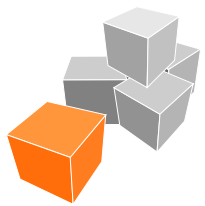
$$f_\phi : \{0,1\}^n \rightarrow \{0,1\}$$

Whether the advice is applicable depends on the $n$ atomic pointcuts $x_1, \ldots, x_n$ occurring in the residue $\phi$.



# Two Assumptions on Advice Dispatch

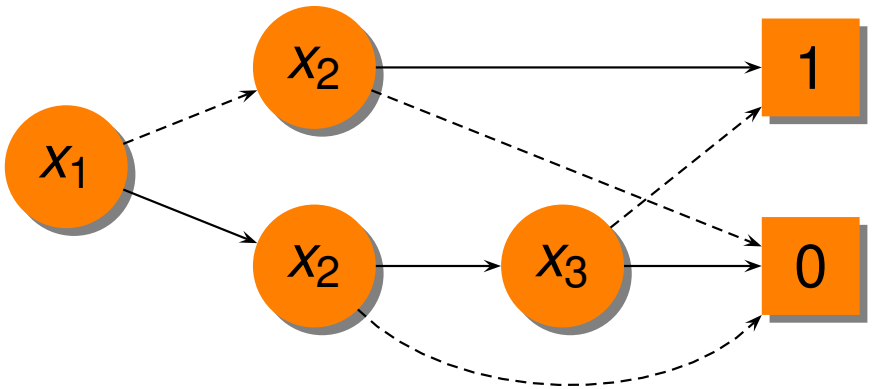
1. Evaluation is side-effect free.
2. Binding of parameters is not a side-effect.

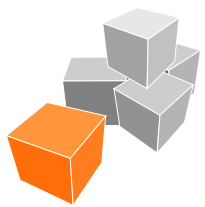


# From Formulas to BDDs to Strategies

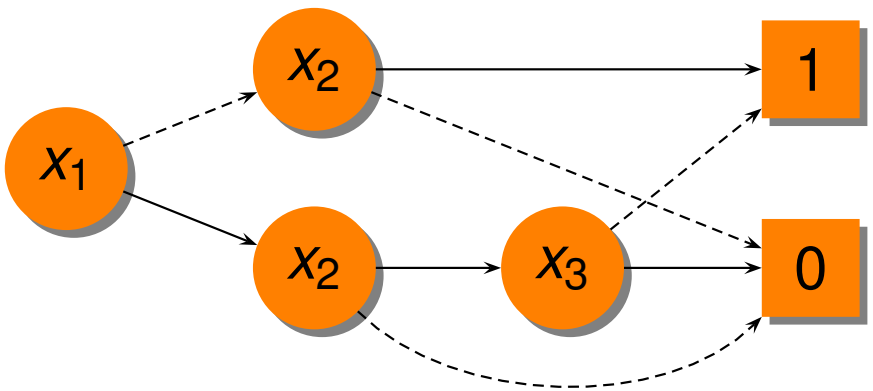
## Example Formula

$\phi = (x_1 \wedge x_2 \wedge \overline{x}_3) \vee (\overline{x}_1 \wedge x_2)$

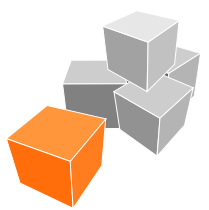


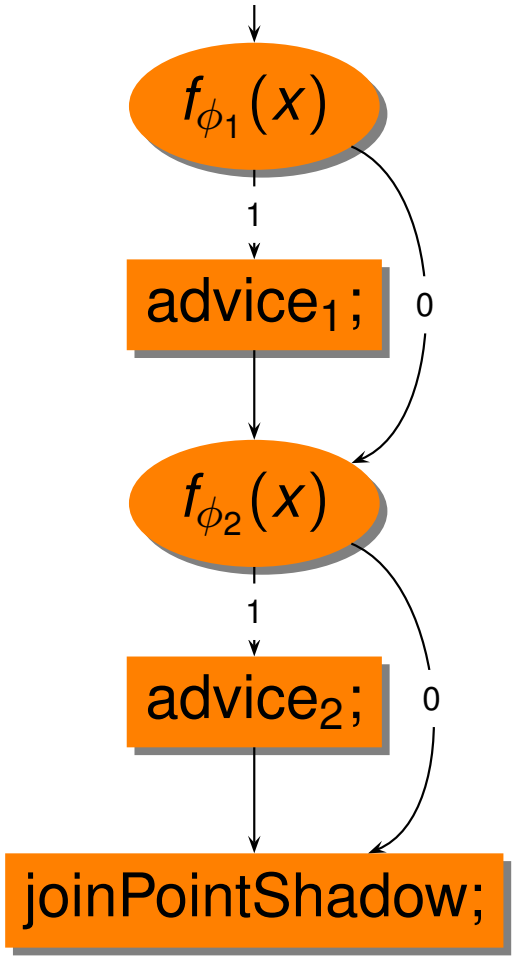



# Full Redundancy Elimination



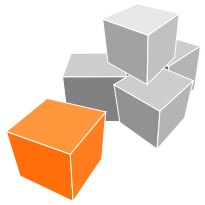

Reduced ordered binary decision diagrams offer complete redundancy elimination.






## Example

**before**() : joinPointShadow
&& $\phi_1$ { advice$_1$; }
**before**() : joinPointShadow
&& $\phi_2$ { advice$_2$; }

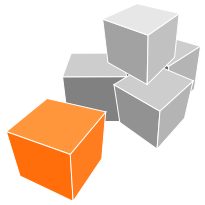


# Extended Dispatch Functions

Residual dispatch at a shared shadow can be viewed as the evaluation of an extended Boolean function.

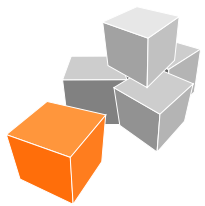
$$f_\Phi : \{0,1\}^n \rightarrow \{0,1\}^m$$

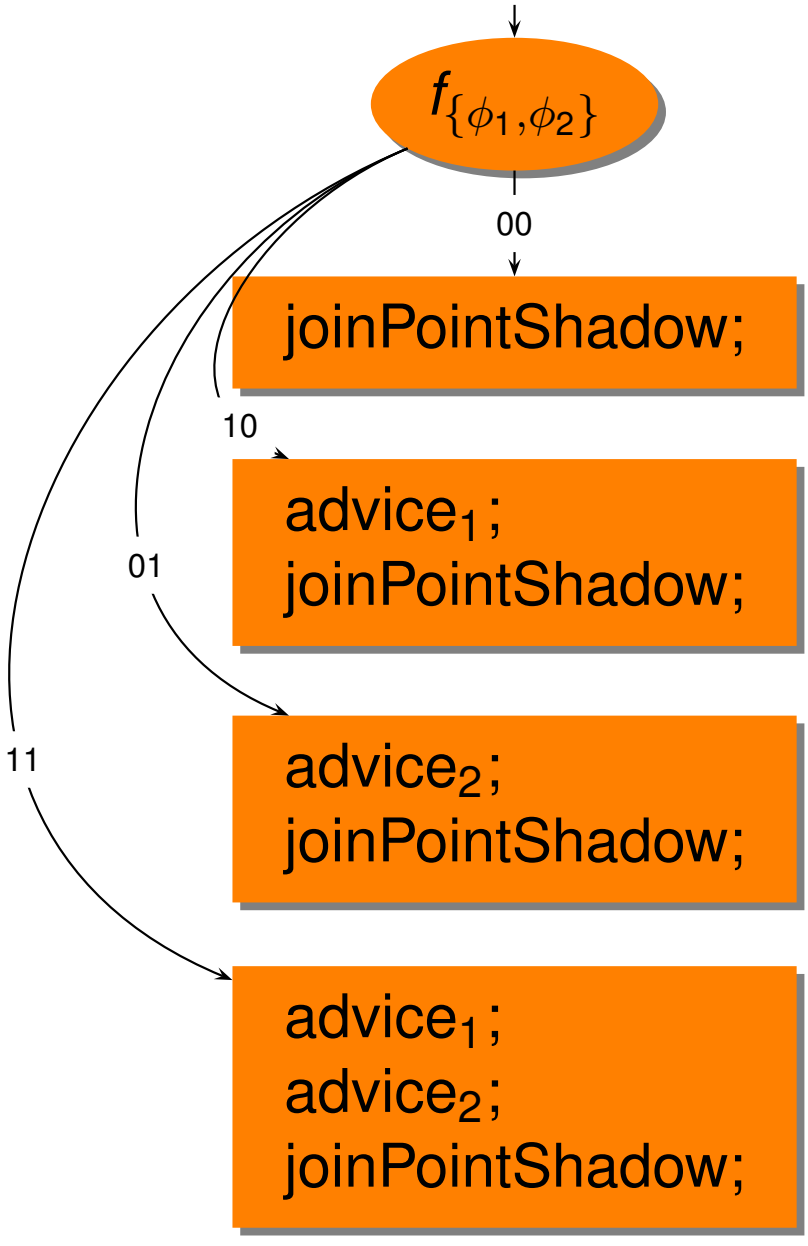
Which combination of the $m$ advice is applicable depends on the $n$ atomic pointcuts jointly occurring in the residues $\phi_1, \ldots, \phi_m$.



# Three Assumptions on Advice Dispatch

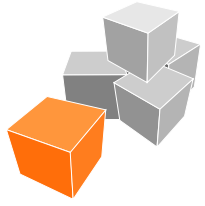
1. Evaluation is side-effect free.
2. Binding of parameters is not a side-effect.
3. Execution of an advice does not affect evaluation.






## Example

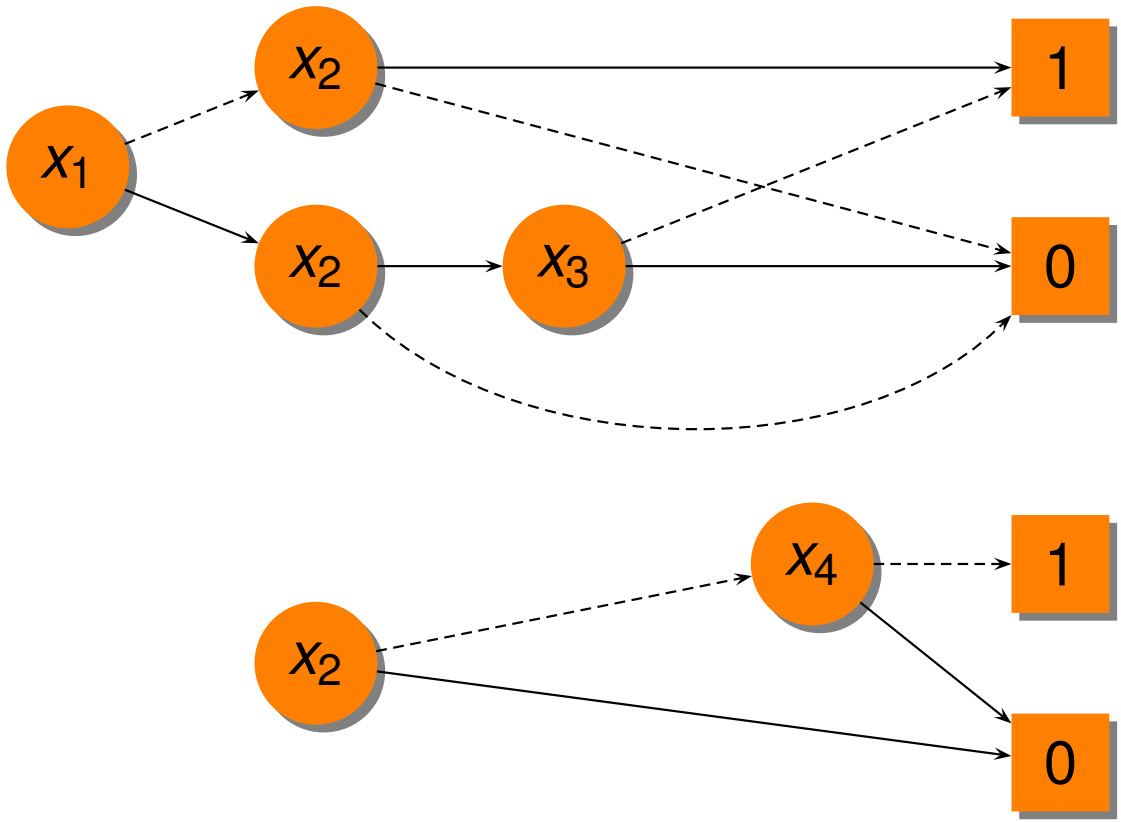
**before**() : joinPointShadow
&& $\phi_1$ { advice$_1$; }
**before**() : joinPointShadow
&& $\phi_2$ { advice$_2$; }

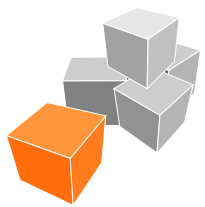


## Example Formulas

$\phi_1 = (x_1 \wedge x_2 \wedge \overline{x}_3) \vee (\overline{x}_1 \wedge x_2)$

$\phi_2 = \overline{x}_2 \wedge \overline{x}_4$

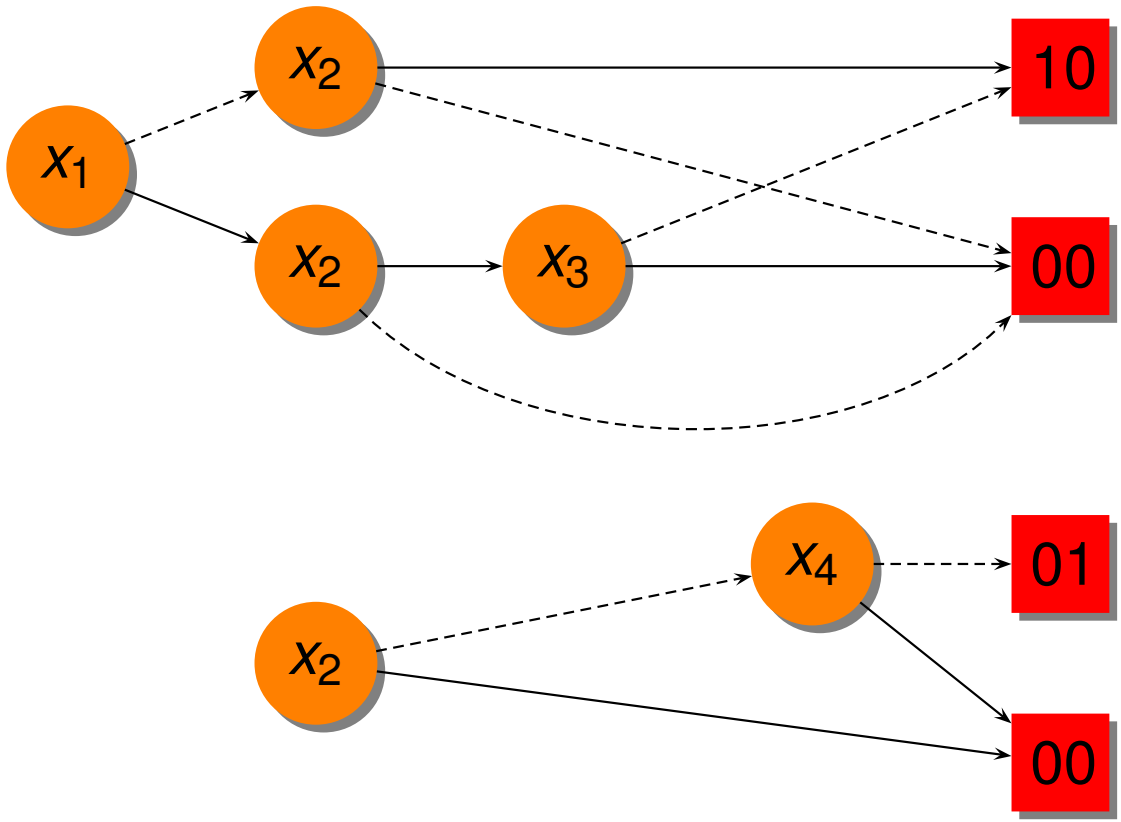


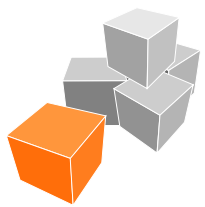



## Example Formulas

$\phi_1 = (x_1 \wedge x_2 \wedge \overline{x}_3) \vee (\overline{x}_1 \wedge x_2)$

$\phi_2 = \overline{x}_2 \wedge \overline{x}_4$

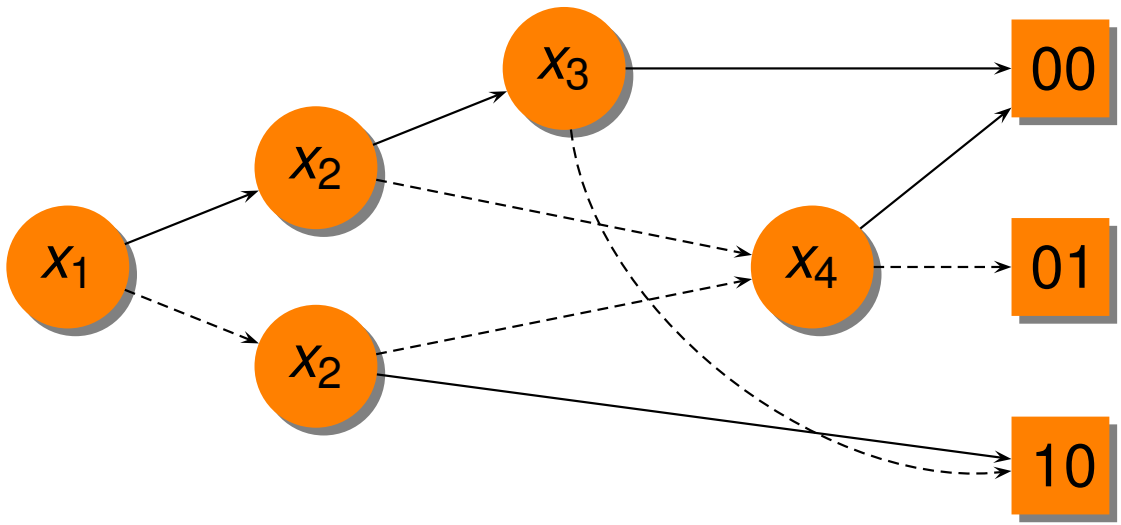


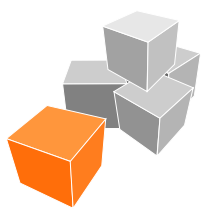



## Example Formulas

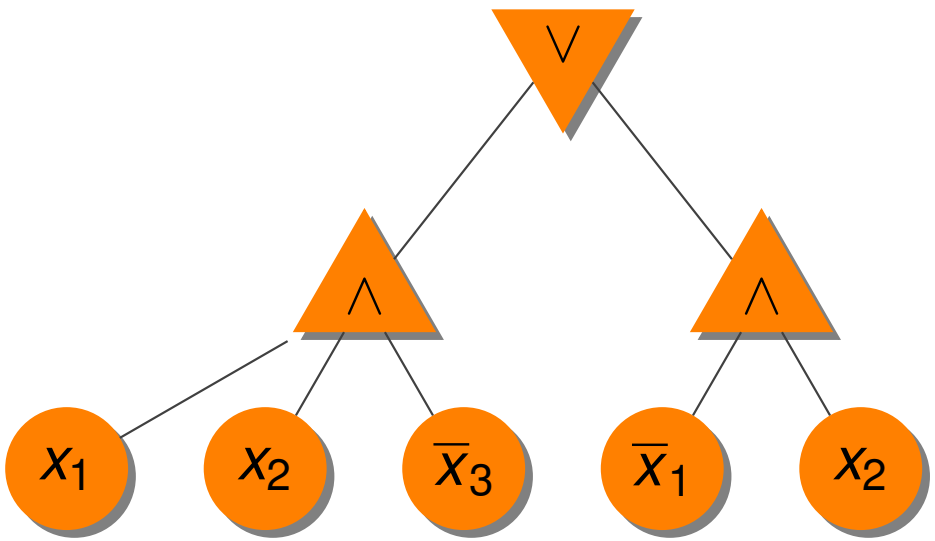
$\phi_1 = (x_1 \wedge x_2 \wedge \overline{x}_3) \vee (\overline{x}_1 \wedge x_2)$

$\phi_2 = \overline{x}_2 \wedge \overline{x}_4$

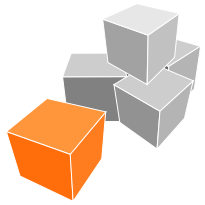


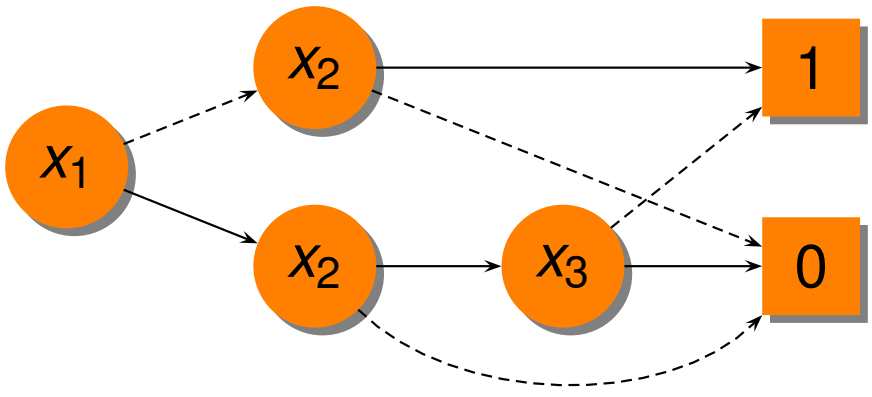
# Experimental Setup

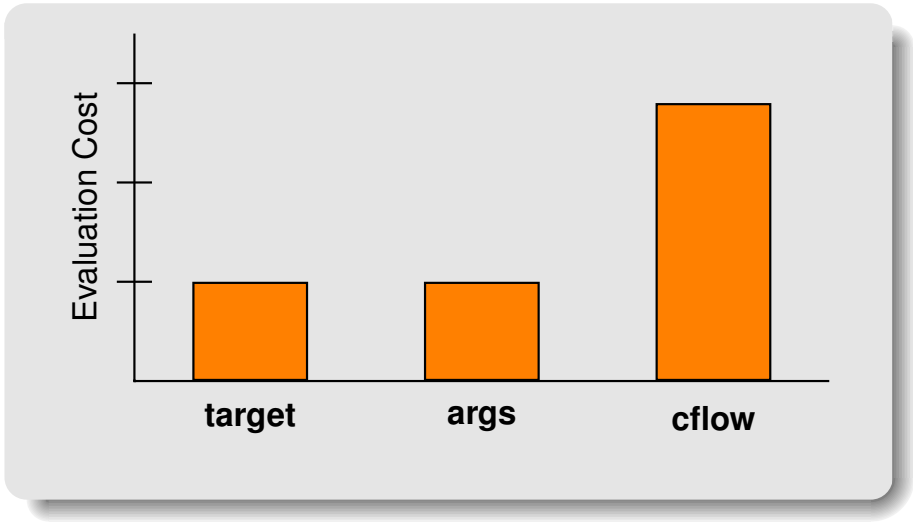



- Formulas of signature $\langle x_1, \ldots, x_5 \rangle$
- At most 6 propositional operators ($\wedge, \vee, \neg$)
- Non-trivial, i.e., not equivalent to $\bot$ or $\top$
- Simple, i.e., the laws of idempotence or boundedness are not applicable

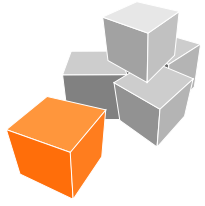


# Experimental Setup (cont'd)

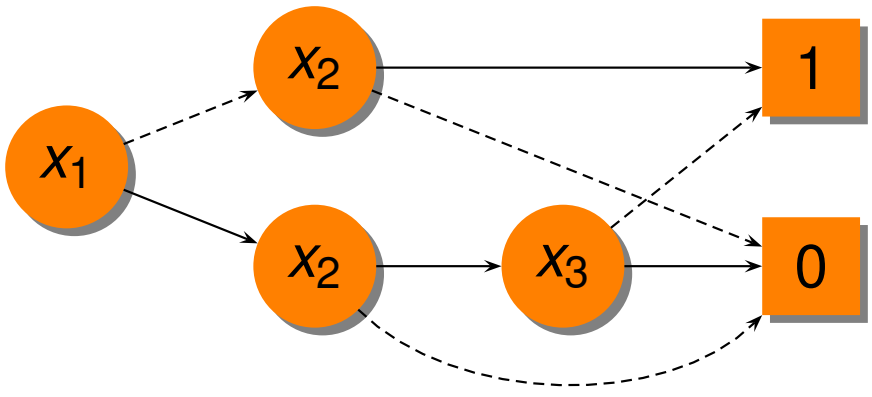


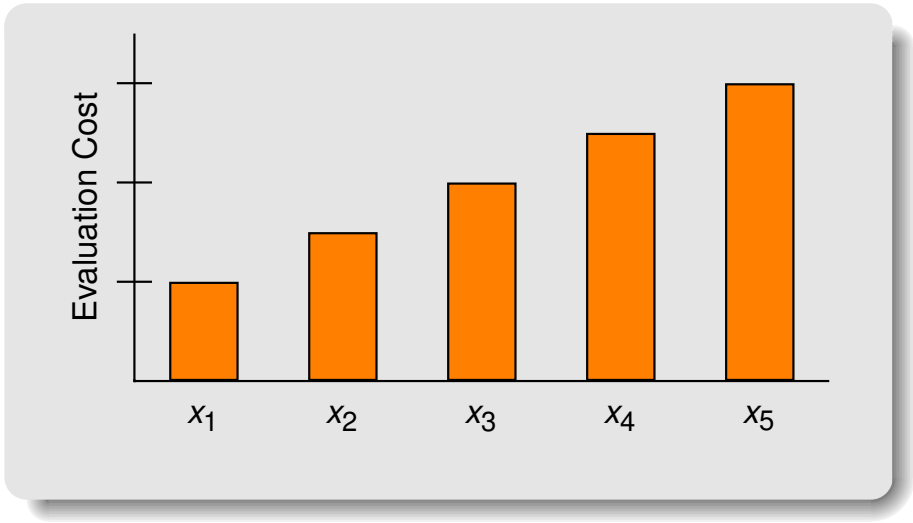




- Compared by average evaluation cost
- BDDs $\langle x_1, \ldots, x_5 \rangle$-ordered

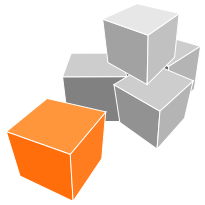


# Experimental Setup (cont'd)

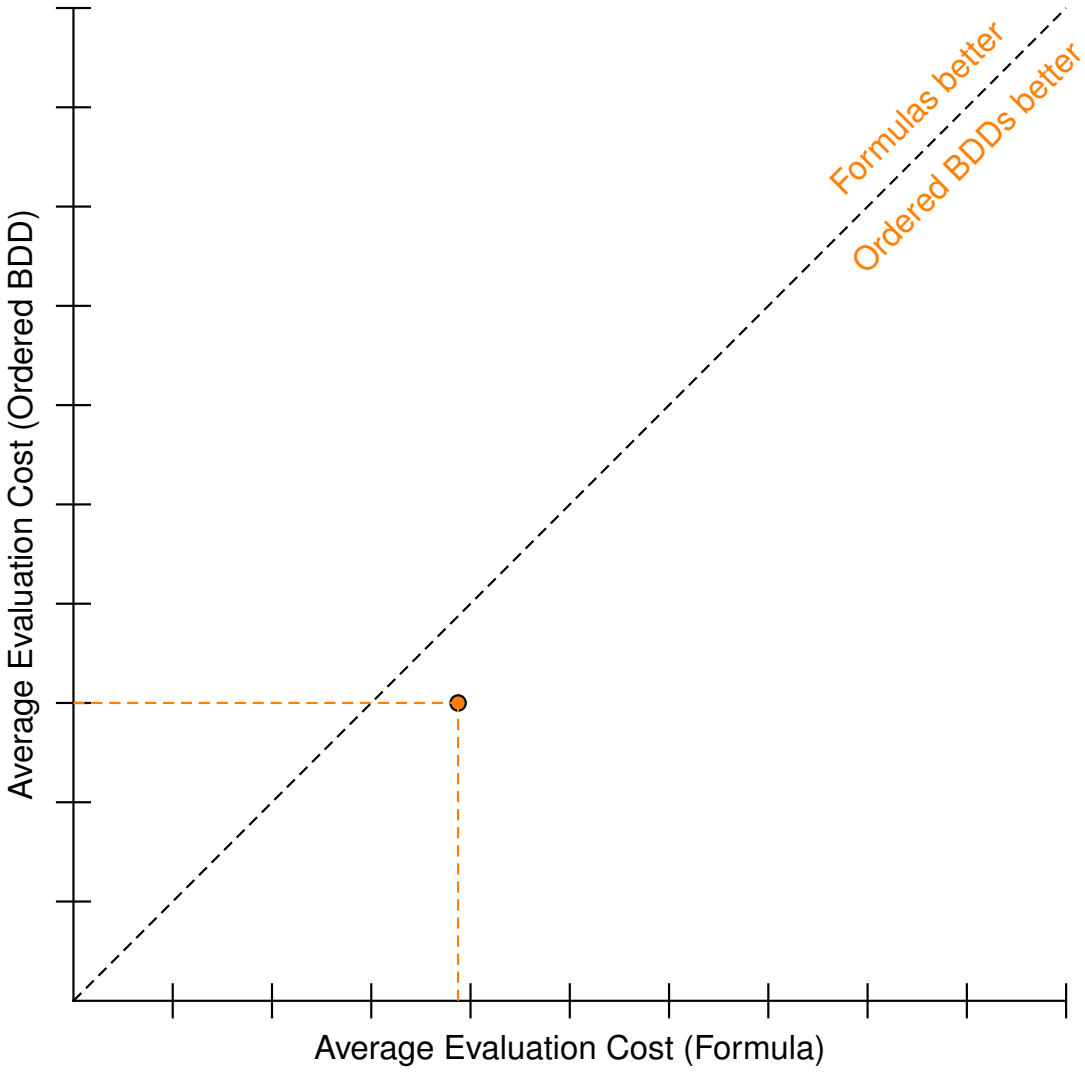


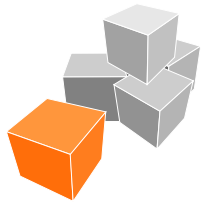




- Compared by average evaluation cost
- BDDs $\langle x_1, \ldots, x_5 \rangle$-ordered

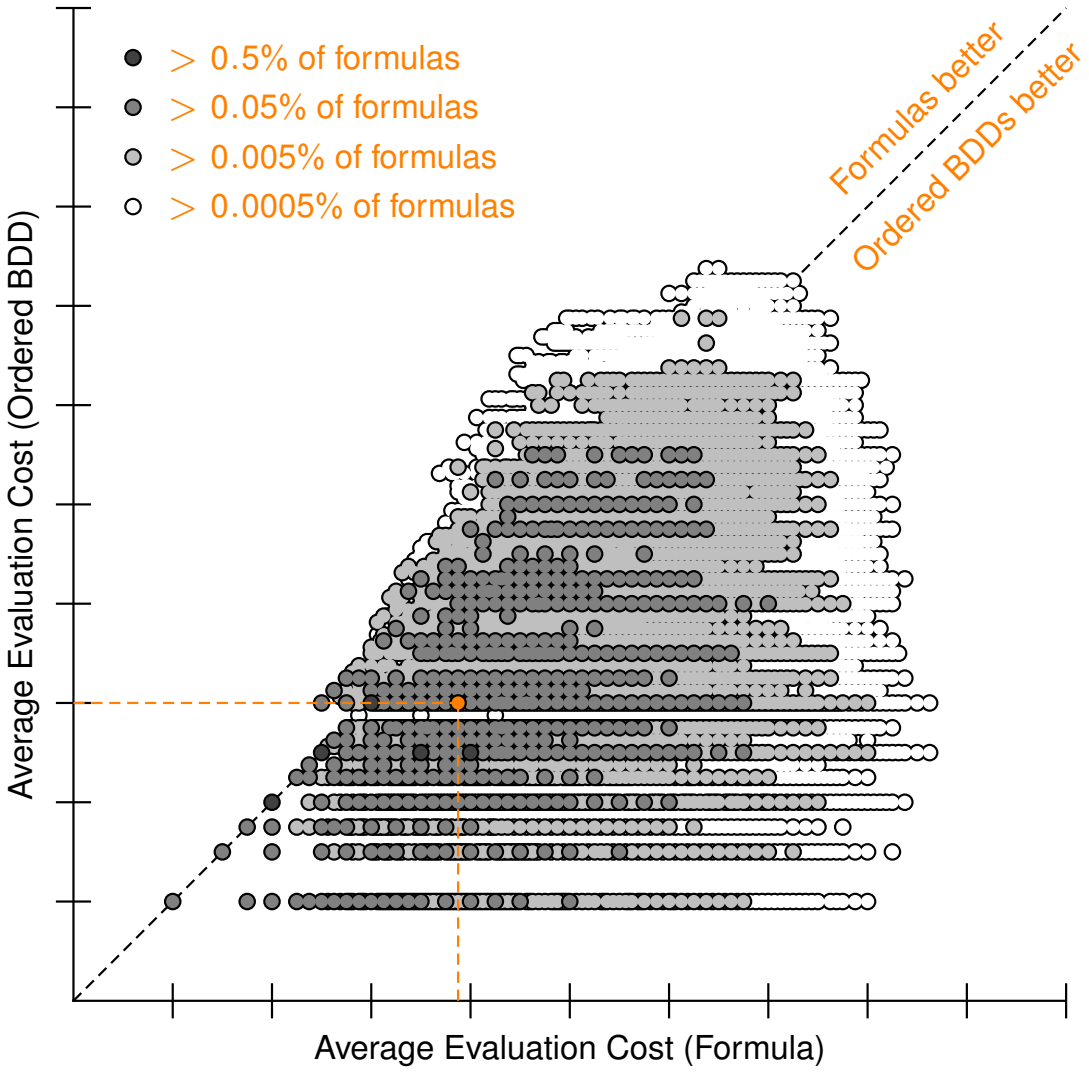


# Results

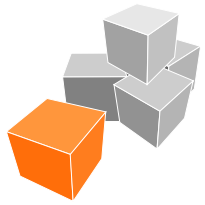

Formulas better
Ordered BDDs better
Average Evaluation Cost (Ordered BDD)
Average Evaluation Cost (Formula)



# Results

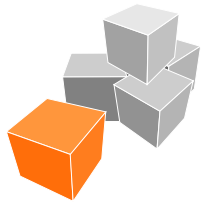



BDD-based dispatch functions outperform formula-based ones 80% of the time. (Further 7.7% are tied.)



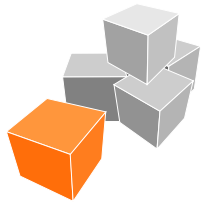
# Conclusion

- Extended dispatch functions are a useful concept.
- BDD-based dispatch functions offer complete redundancy elimination.
- They clearly outperform formula-based dispatch functions.

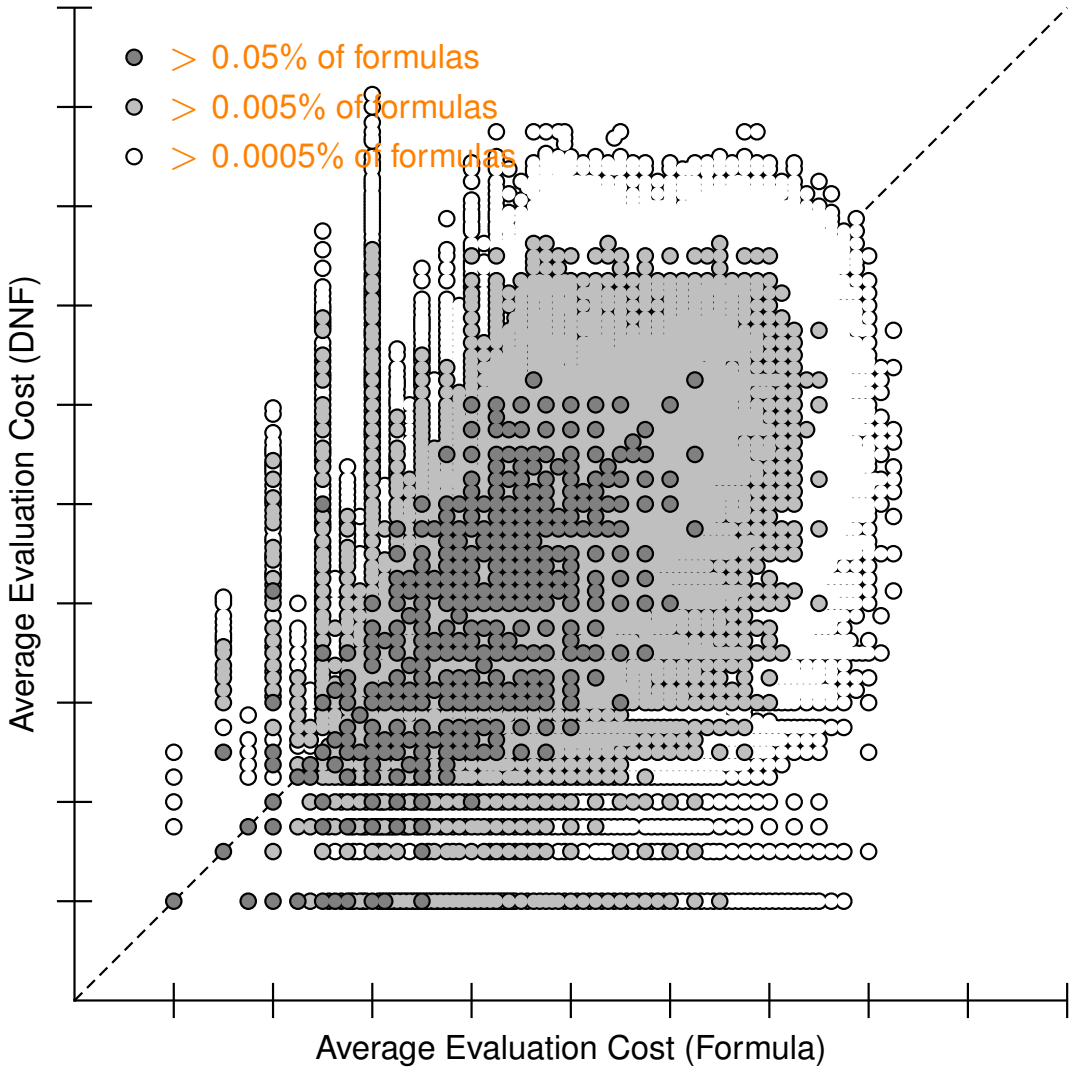


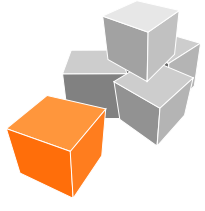
# Recommended Reading


- I. Wegener. *Branching Programs and Binary Decision Diagrams: Theory and Applications*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.
- R. E. Bryant. Graph-based algorithms for Boolean function manipulation. *IEEE Transactions on Computers*, 35(8), 1986.
- C. Chambers, W. Chen. Efficient multiple and predicated dispatching. *ACM SIGPLAN Notices*, 34(10), 1999.

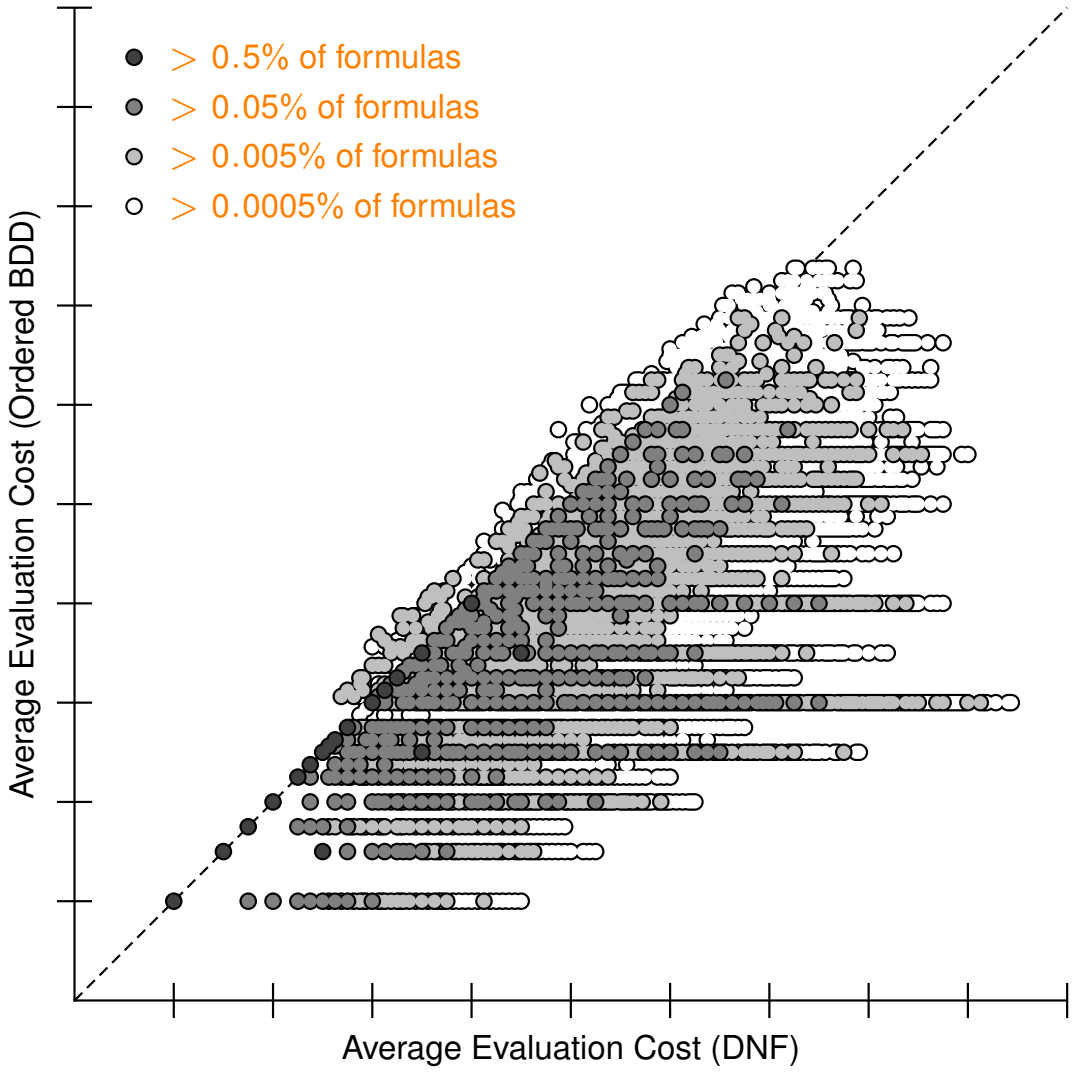



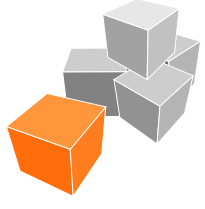
# Further Results



# Further Results (cont'd)



# Further Results (cont'd)