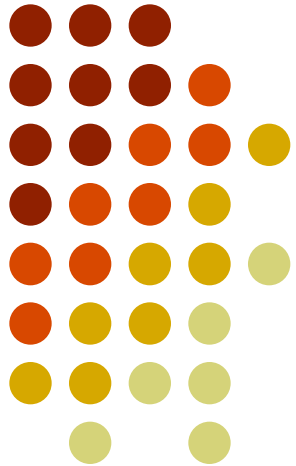
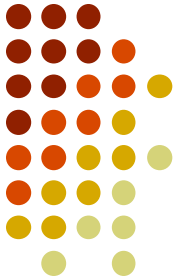
Reducing Interpolant Circuit Size by Ad-Hoc Logic Synthesis and SAT-Based Weakening

Gianpiero Cabodi
Paolo Camurati
Paolo Pasini
Marco Palena
Danilo Vendraminetto

Politecnico di Torino
Torino, Italy



Outline

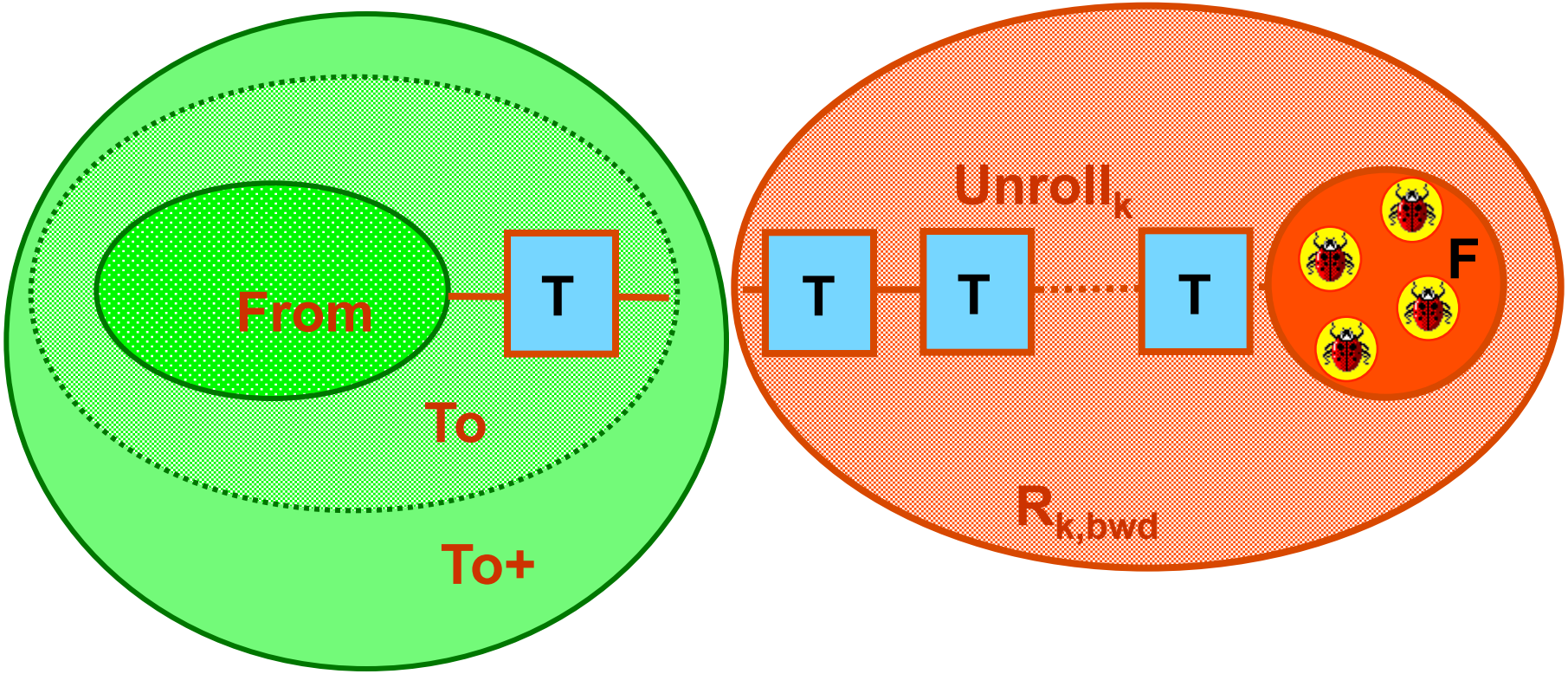


- Motivations & background
  - Craig interpolation in MC => Size bottleneck (>10^5 gates)
  - Highly redundant circuits, missing ad hoc reduction
- Contribution
  - Ad HOC (fast) logic synthesis (based on known techniques)
  - SAT-based Weakening (and strenthening) with high compaction potential (strength/time for size)
- Experimental results & Conclusions

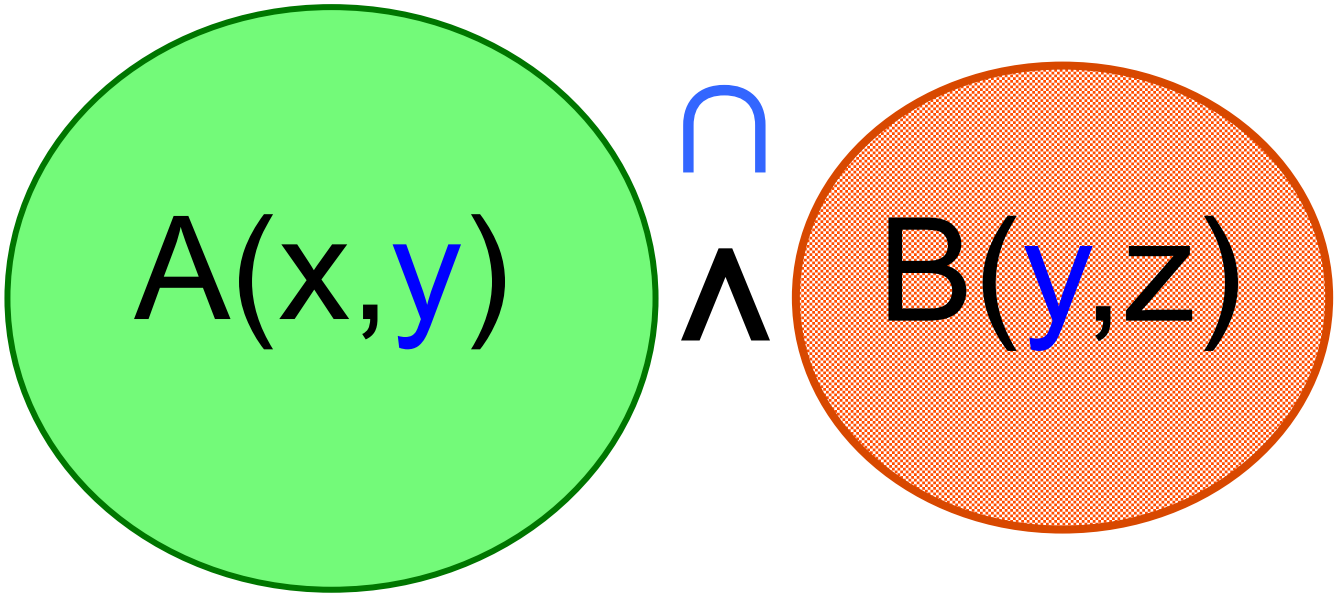
Motivations & related works

- Interpolation (ITP) still a major engine in UMC portfolio [McMillan CAV'03]
- Other ITP usages
  - Formula/constraint synthesis in predicate abstr.
- Scalability problem
  - ITPs are large and highly redundant
- Previous efforts [Marques-Silva CHARME'05, D'Silva et al. VMCAI'08, Cabodi et al. FMSD'15, Rollini et al LPAR'13 (PeRIPLO)]
  - Proof reduction
  - Interpolant compaction

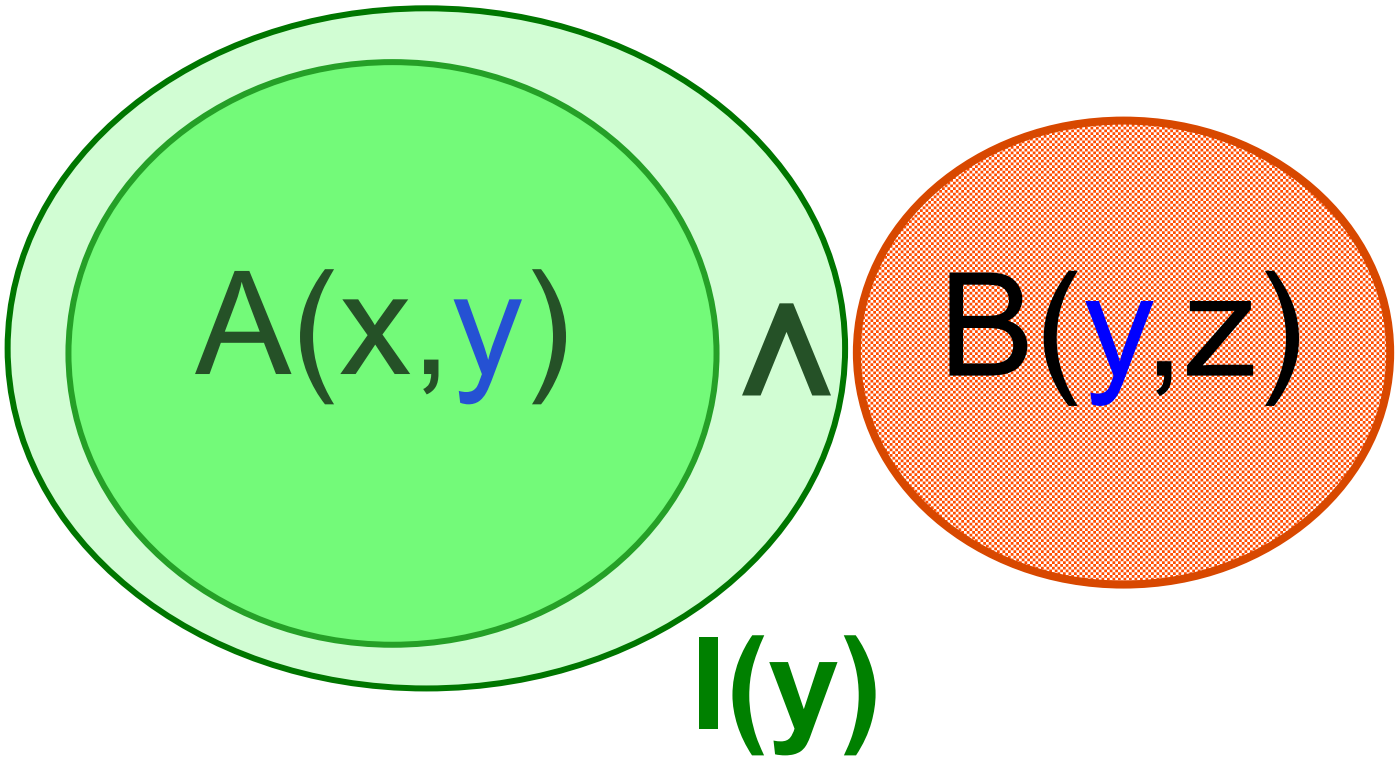
Background: Craig Interpolant for IMG in Unbounded MC



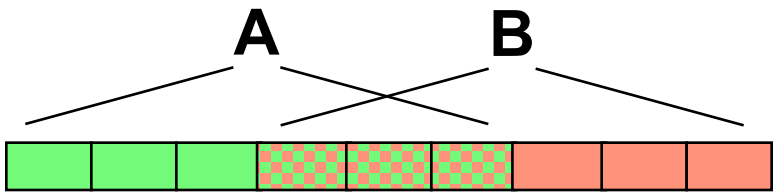
Interpolant: set view



Interpolant: set view



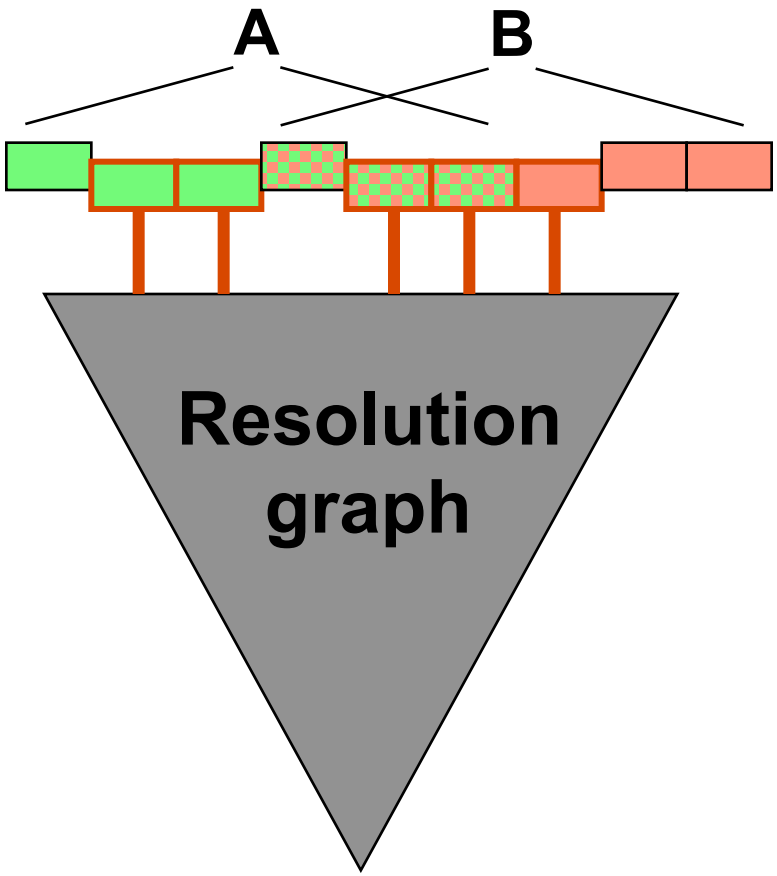
Craig Interpolant from refutation proof



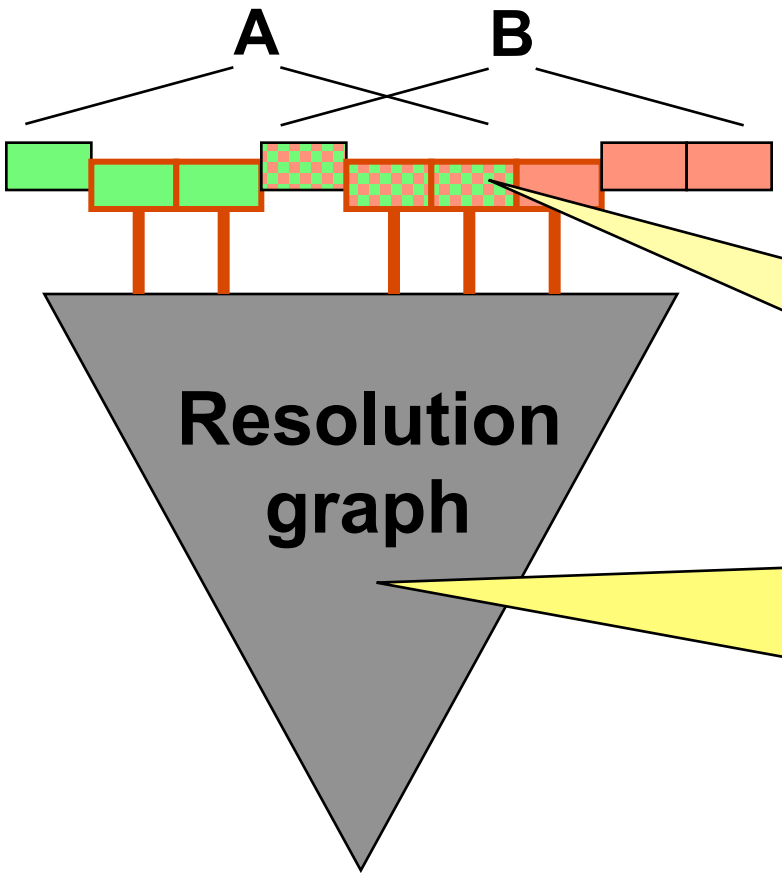
CNF clauses

UNSAT problem ($A \wedge B = 0$)

Craig Interpolant from refutation proof



Craig Interpolant from refutation proof

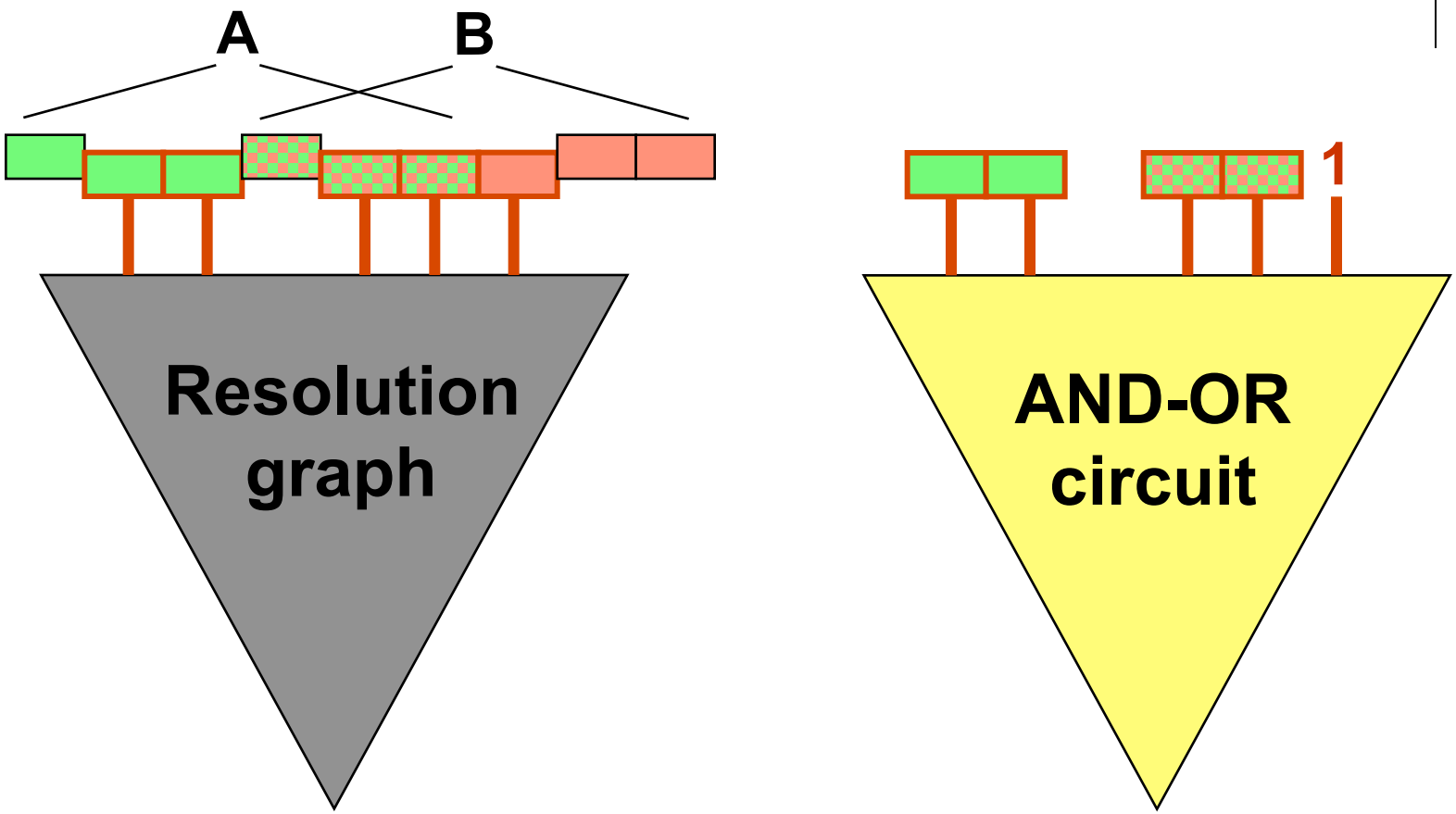


Unsatisfiable core

resolution

$$\frac{(A \vee p) \quad (\neg p \vee B)}{(A \vee B)}$$

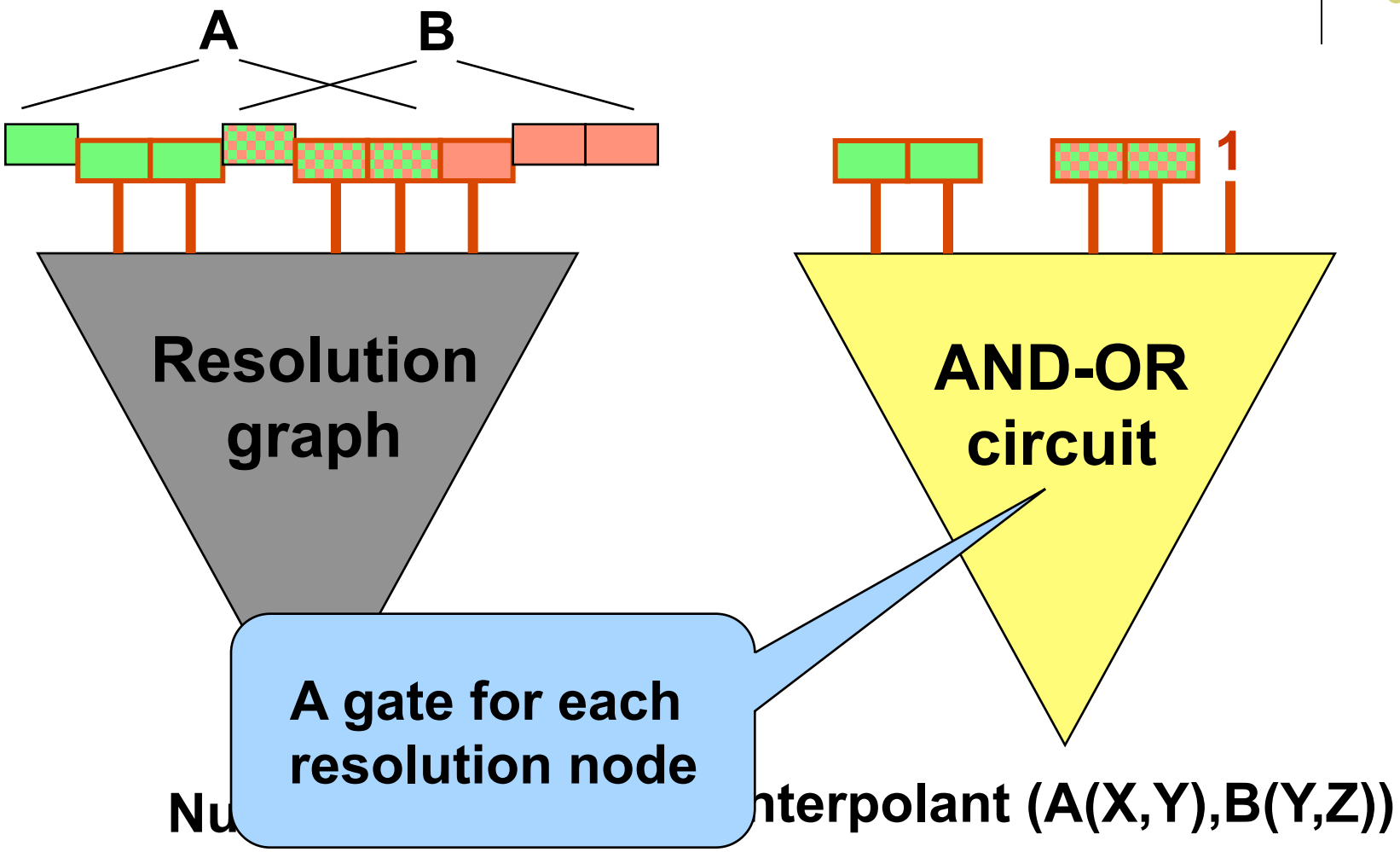
Craig Interpolant from refutation proof



Null clause

I(Y) = Interpolant (A(X,Y),B(Y,Z))

Craig Interpolant from refutation proof



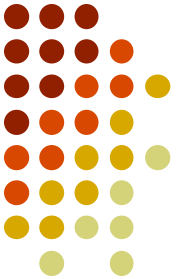
Redundancy / Compaction

|Interpolant| = O(|proof|)

highly redundant => compaction

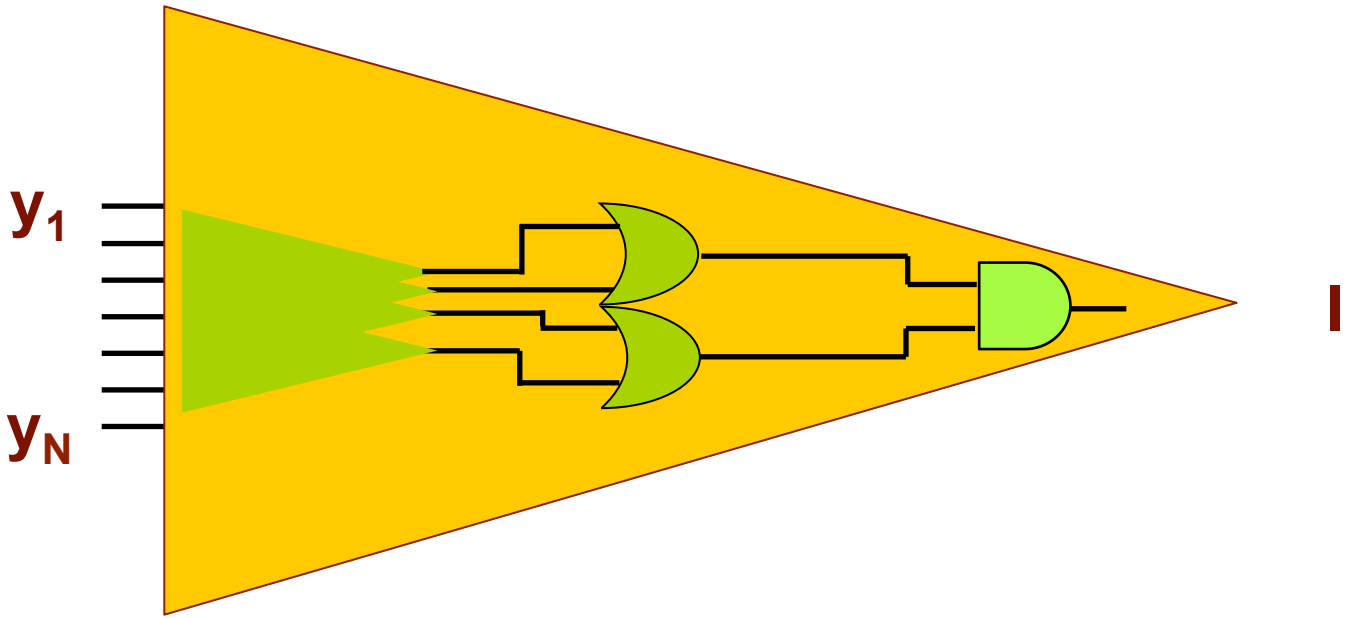
- Reduce proof
  - Recycle-pivots, local transformations, proof restructure
- Combinational logic synthesis
  - BDD/SAT-sweeping
  - Rewriting
  - Refactoring
- Ad hoc logic synthesis
  - Logic synthesis using proof graph
  - ITE-based decompose & compact

Contributions

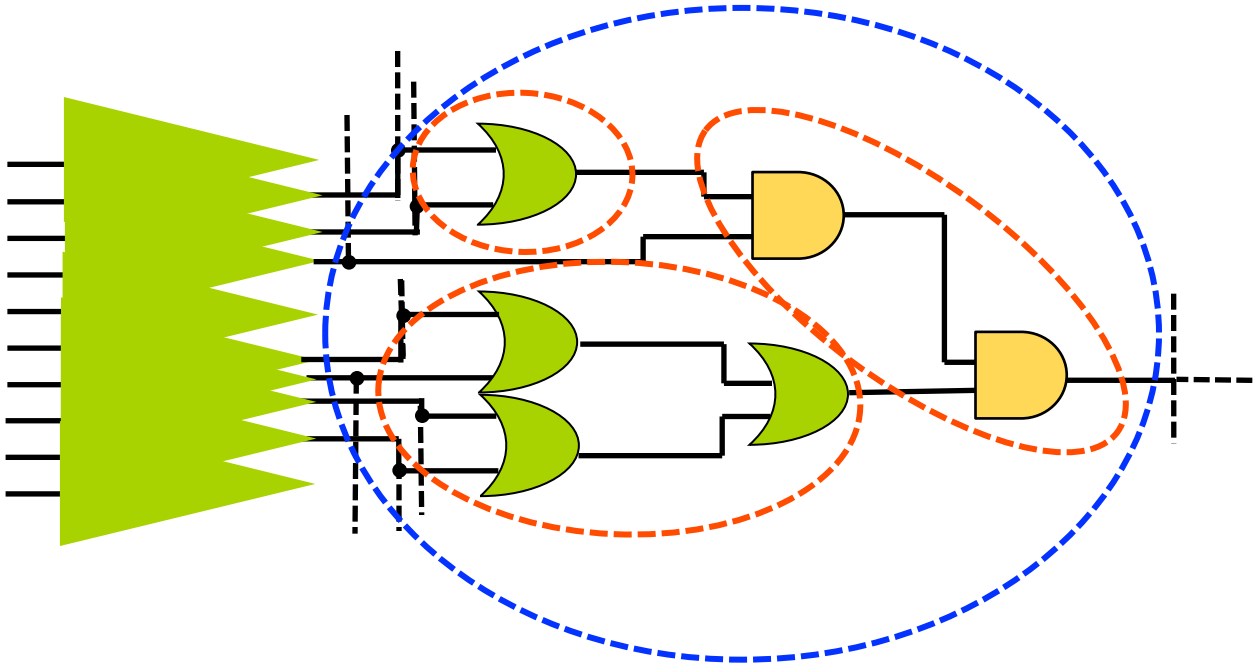


- AD-hoc logic synthesis (rewrite/refactor)
  - Implemented/tuned for interpolants
  - Interpolant strength unchanged
  - Trade-off speed for optimality
- Compaction by strenghtening/weakening
  - Gate Level Abstraction
  - Proof (core) based
  - Expensive (SAT needed)
  - Trade-off strength for size

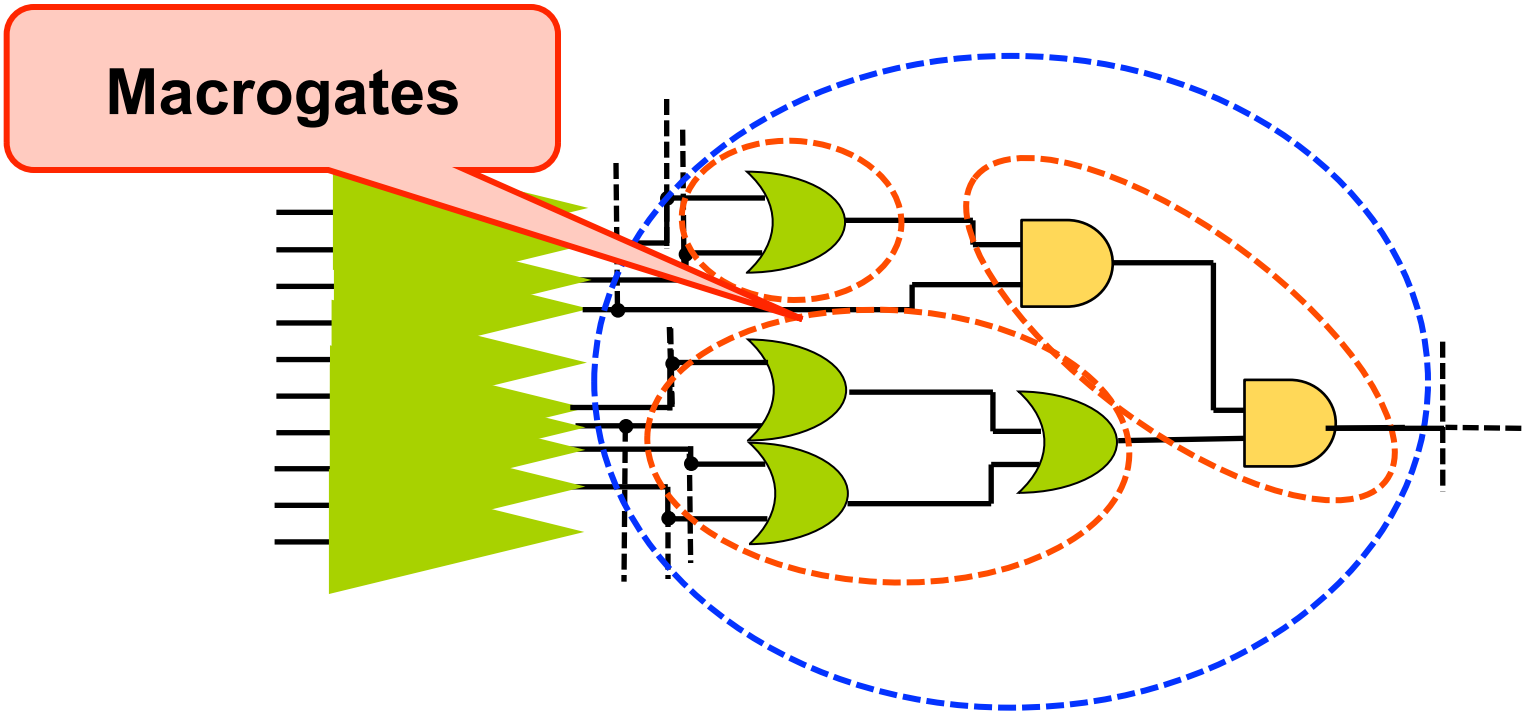
Circuit graph decomposition



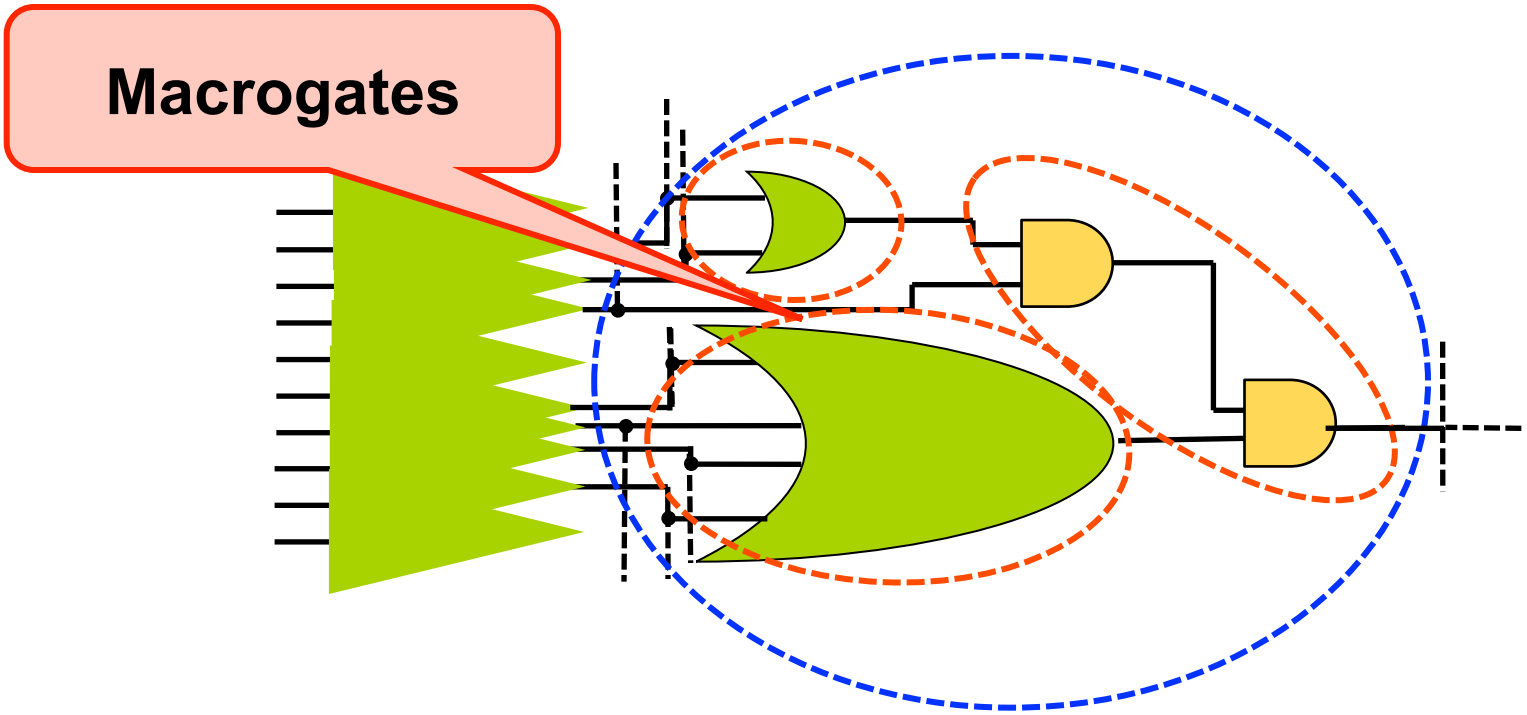
Circuit graph decomposition



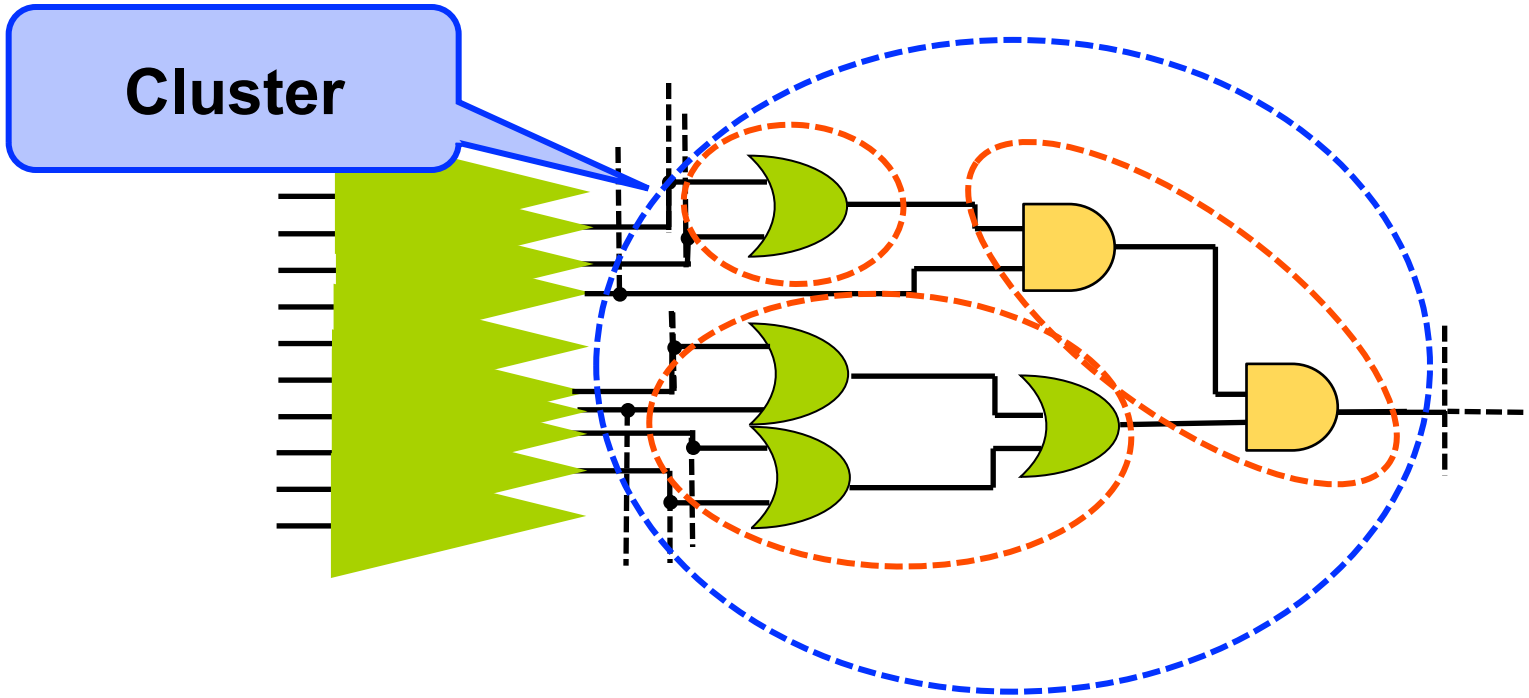
Circuit graph decomposition



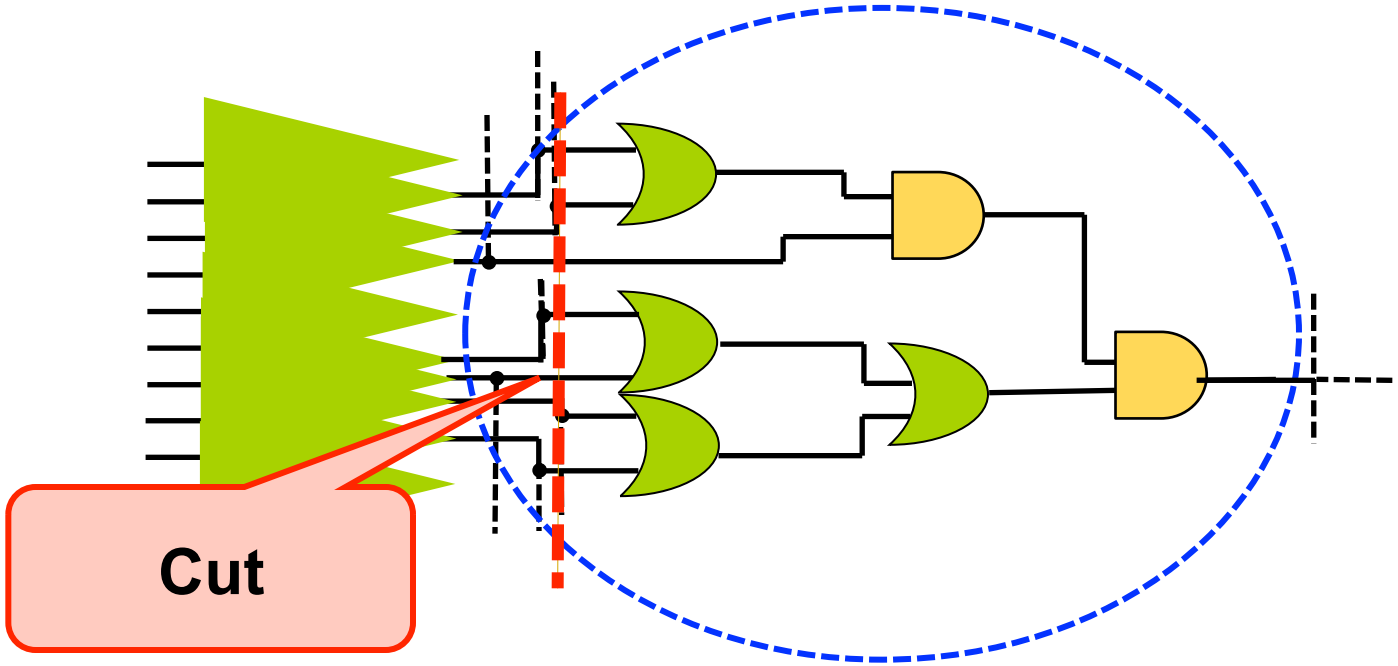
Circuit graph decomposition



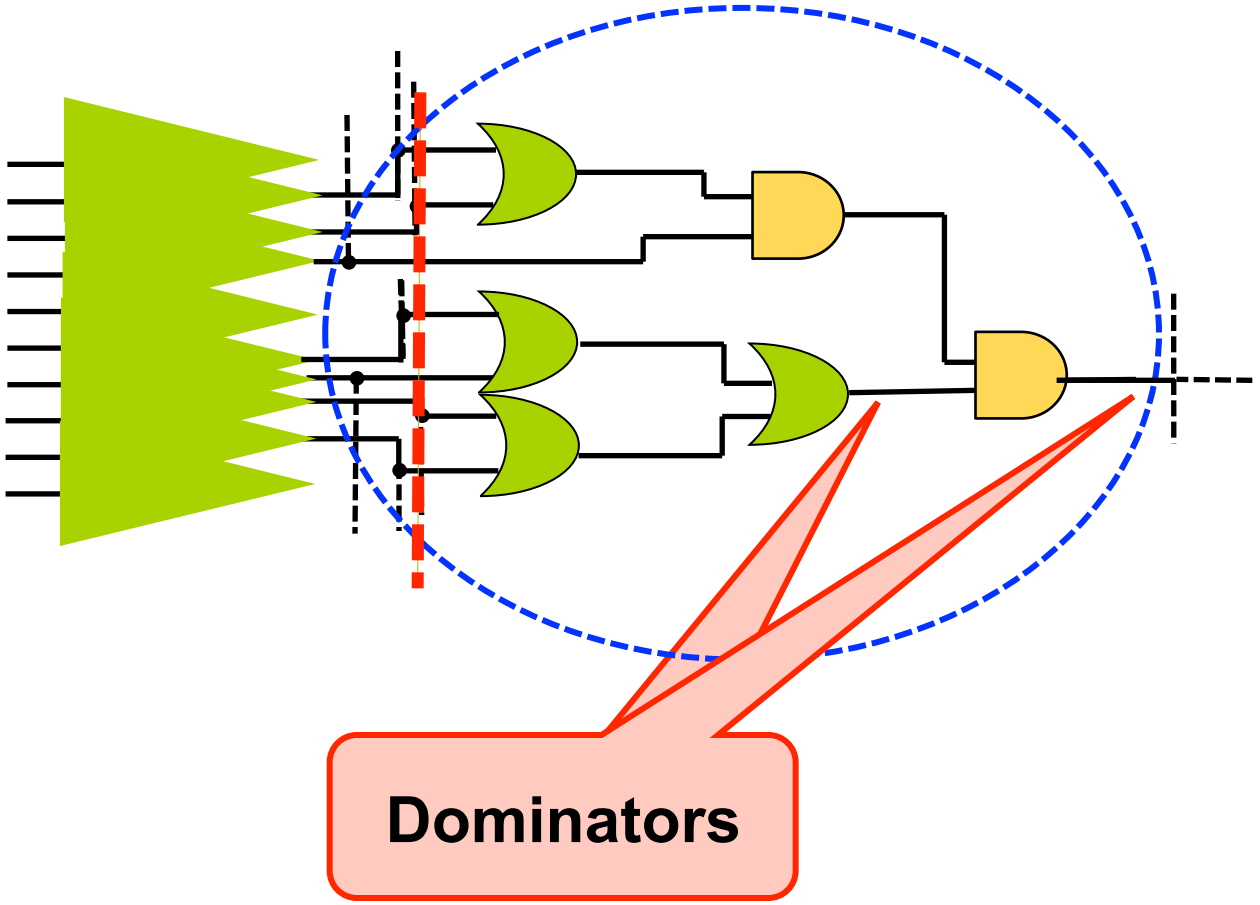
Circuit graph decomposition



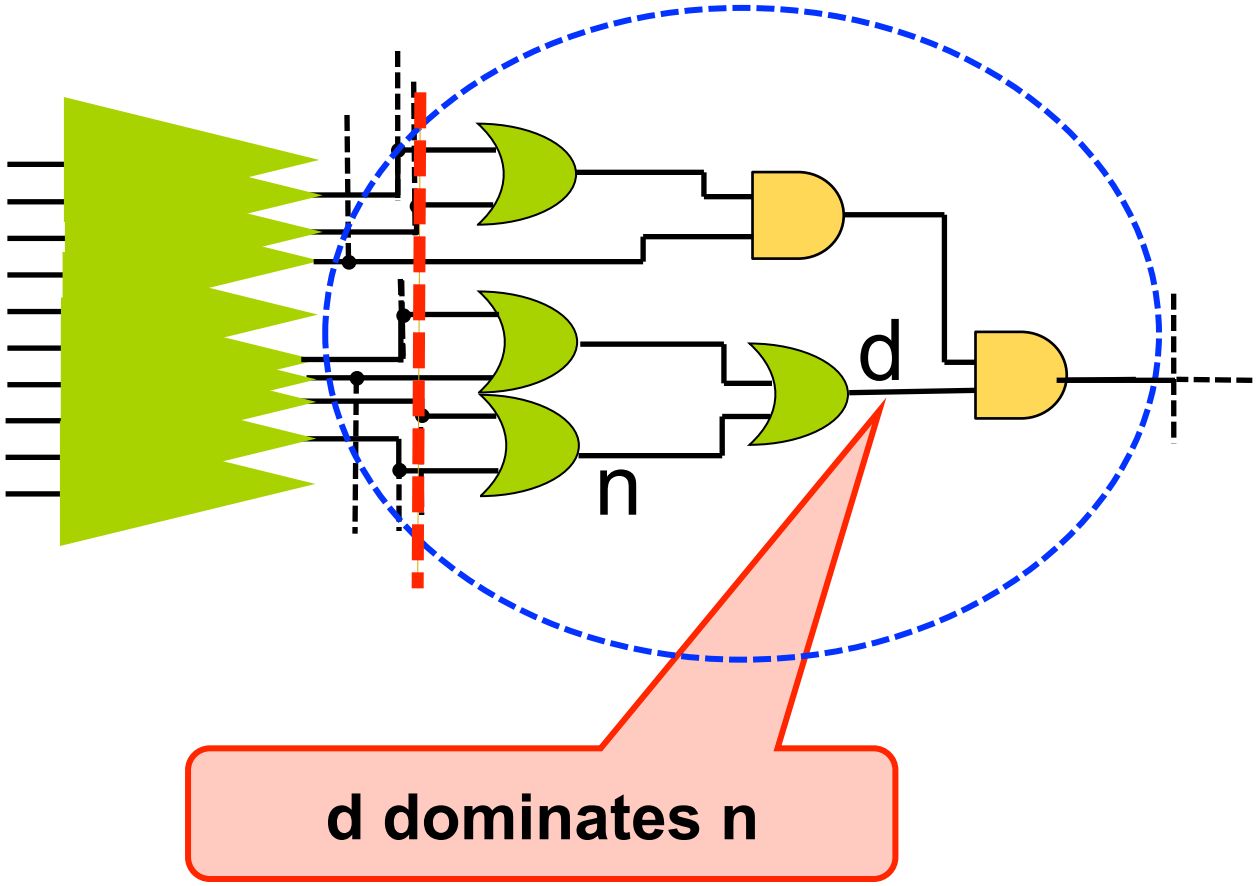
Circuit graph decomposition



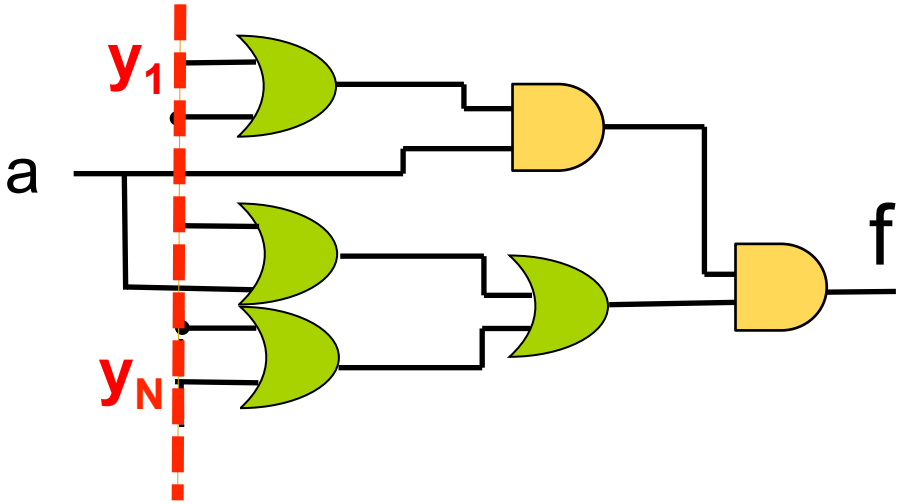
Circuit graph decomposition



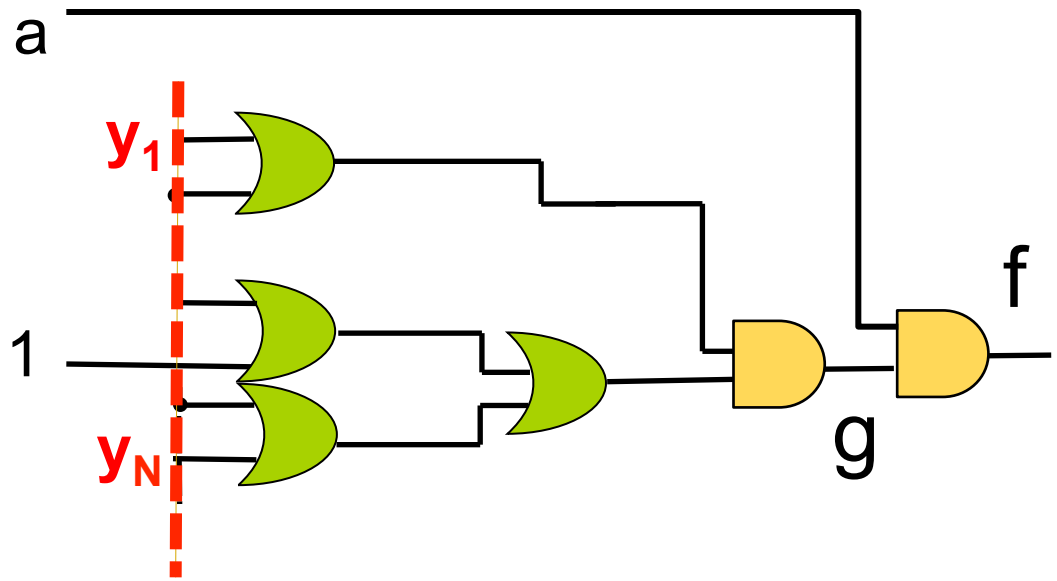
Circuit graph decomposition



Rewrite: direct ODC removal

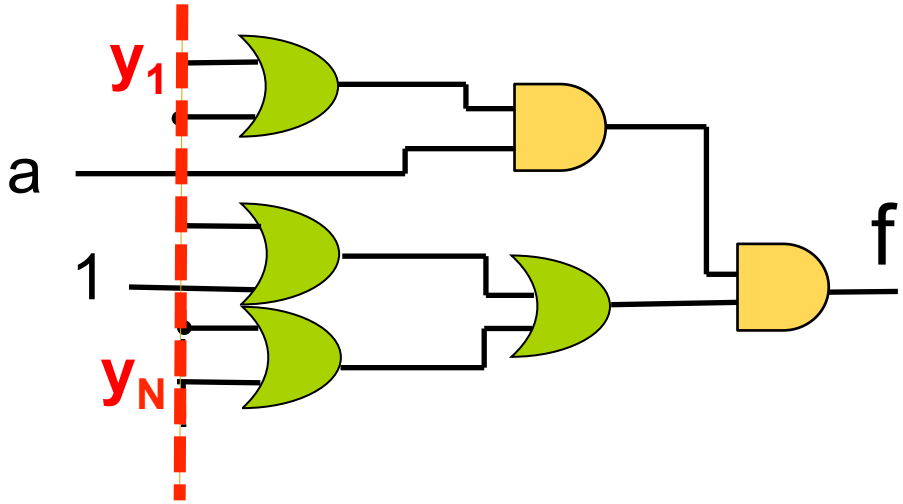


Rewrite: direct ODC removal



$$F(Y,a) = a \wedge g(Y,a) = a \wedge g(Y,1)$$

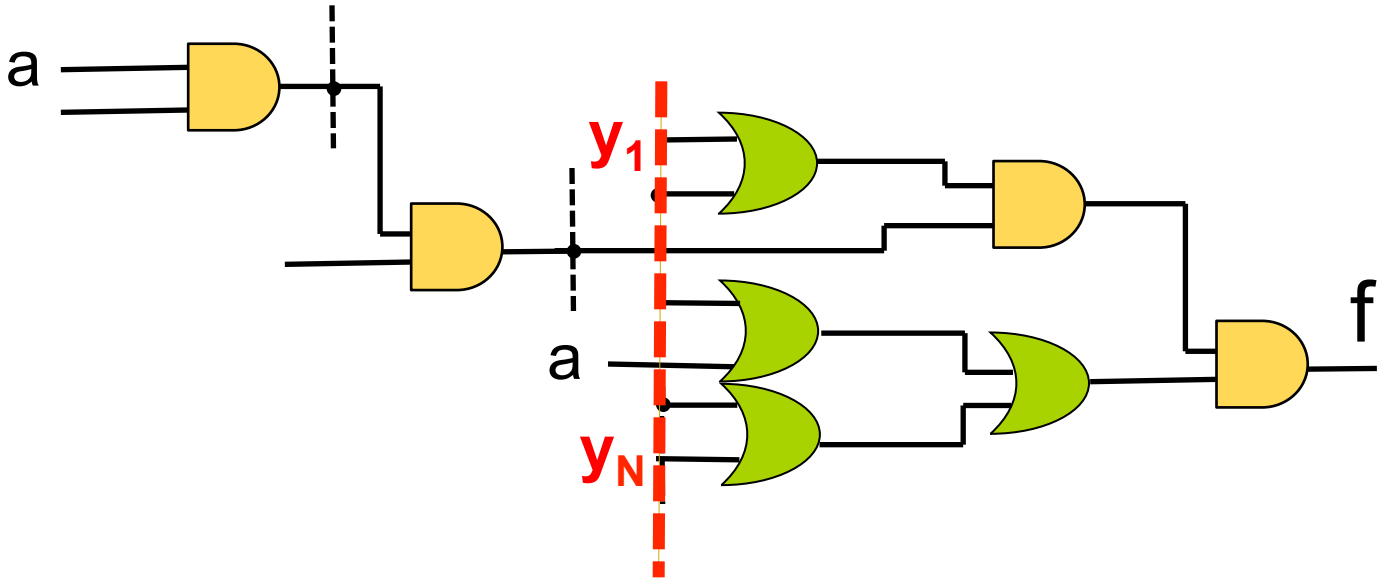
Rewrite: direct ODC removal



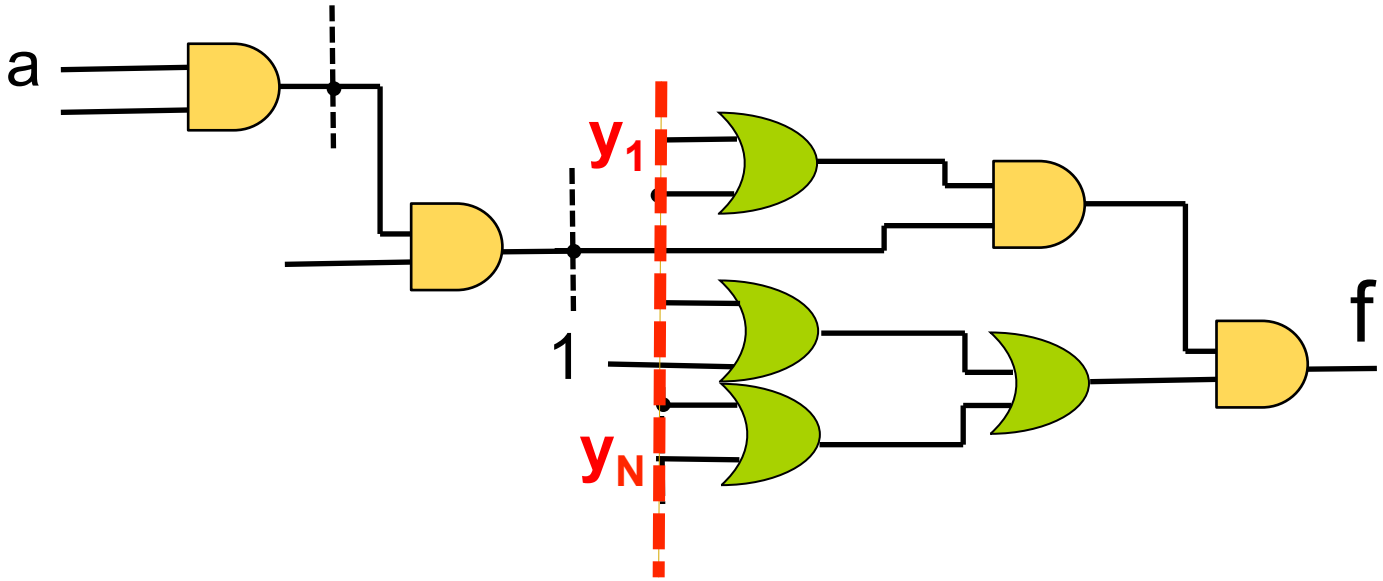
Algorithm

```
DirectOdcSimplify()
  forall clusters CL
    find node v ∈  cut(CL) with fanout(v) >1 in CL
    take u,t ∈ fanout(v) ∩ CL
    if (domG(t) dominates u)
      u <= constant // u is redundant
```

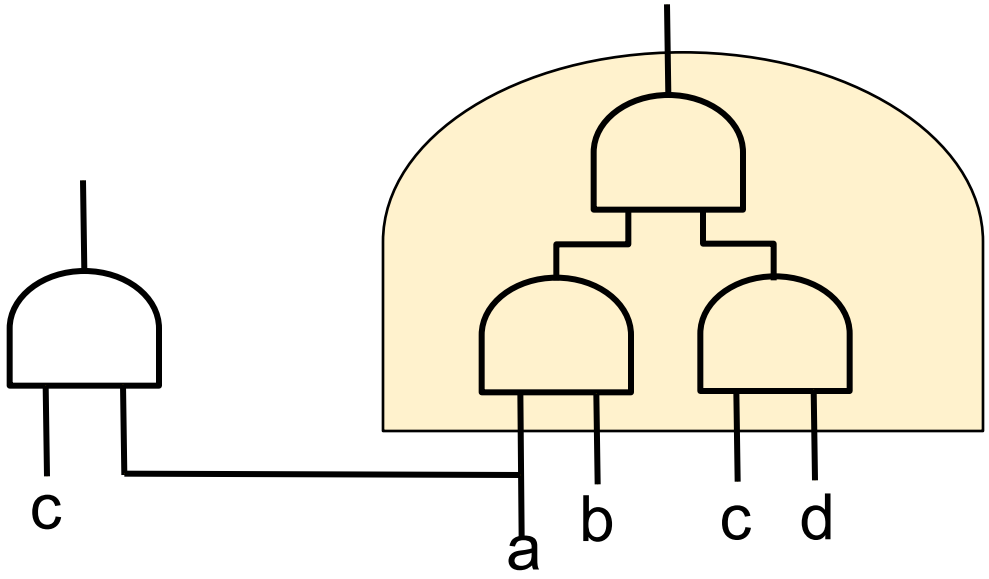
Rewrite: transitive ODC



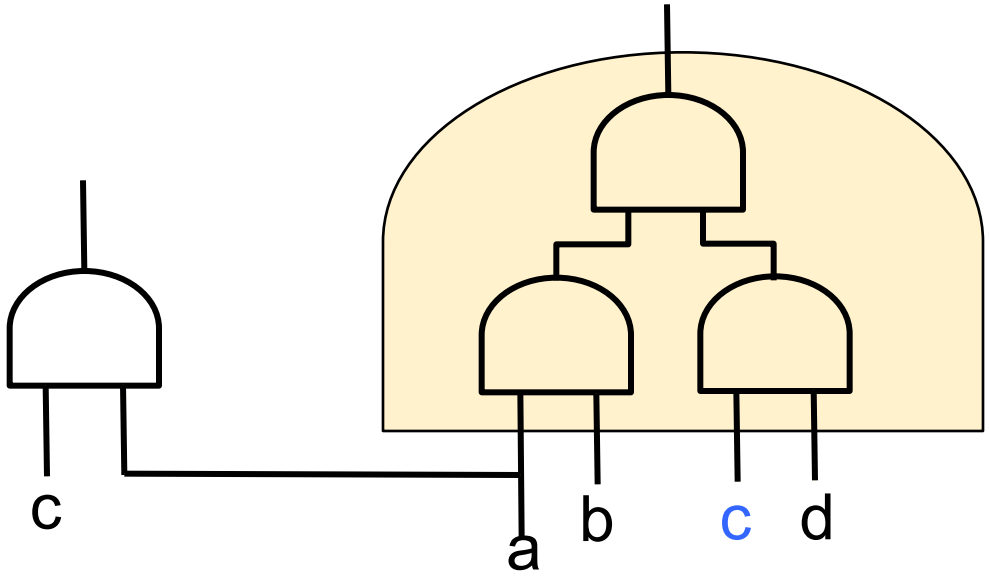
Rewrite: transitive ODC



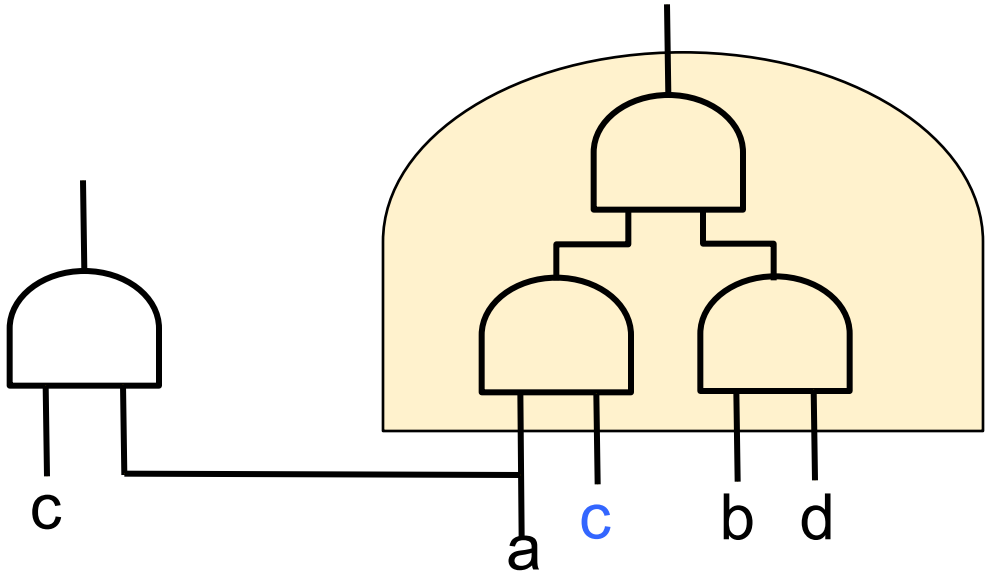
Refactor: search sharable term



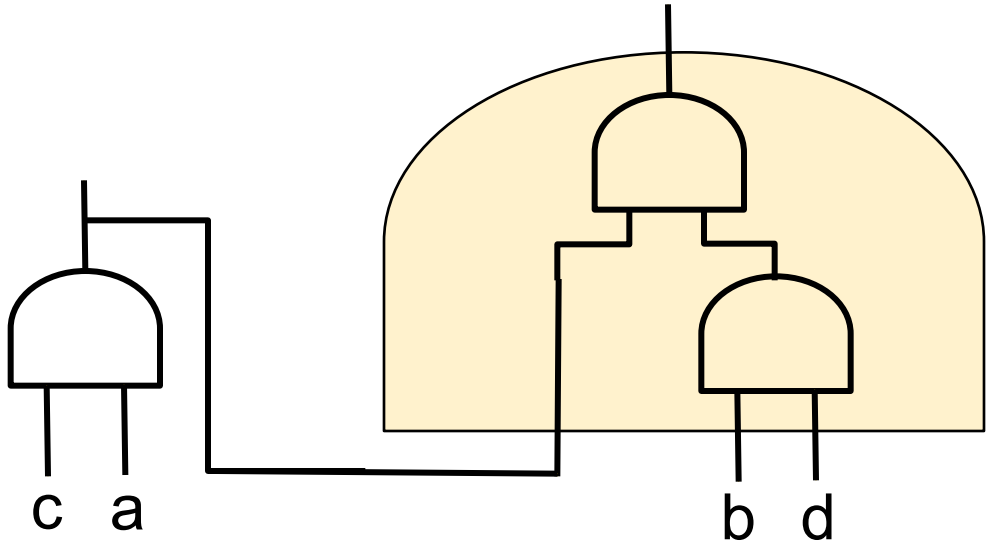
Refactor: search sharable term



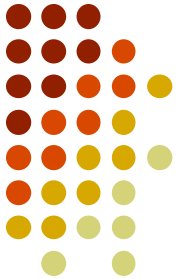
Refactor: search sharable term



Refactor: search sharable term



Algorithm



MacrogateRefactor()

forall macrogates G

forall nodes v $\in$ cut(G)

find AND(u,v) $\notin$ G such that u $\in$ cut(G)

refactor G using AND(u,v)

Experimental results

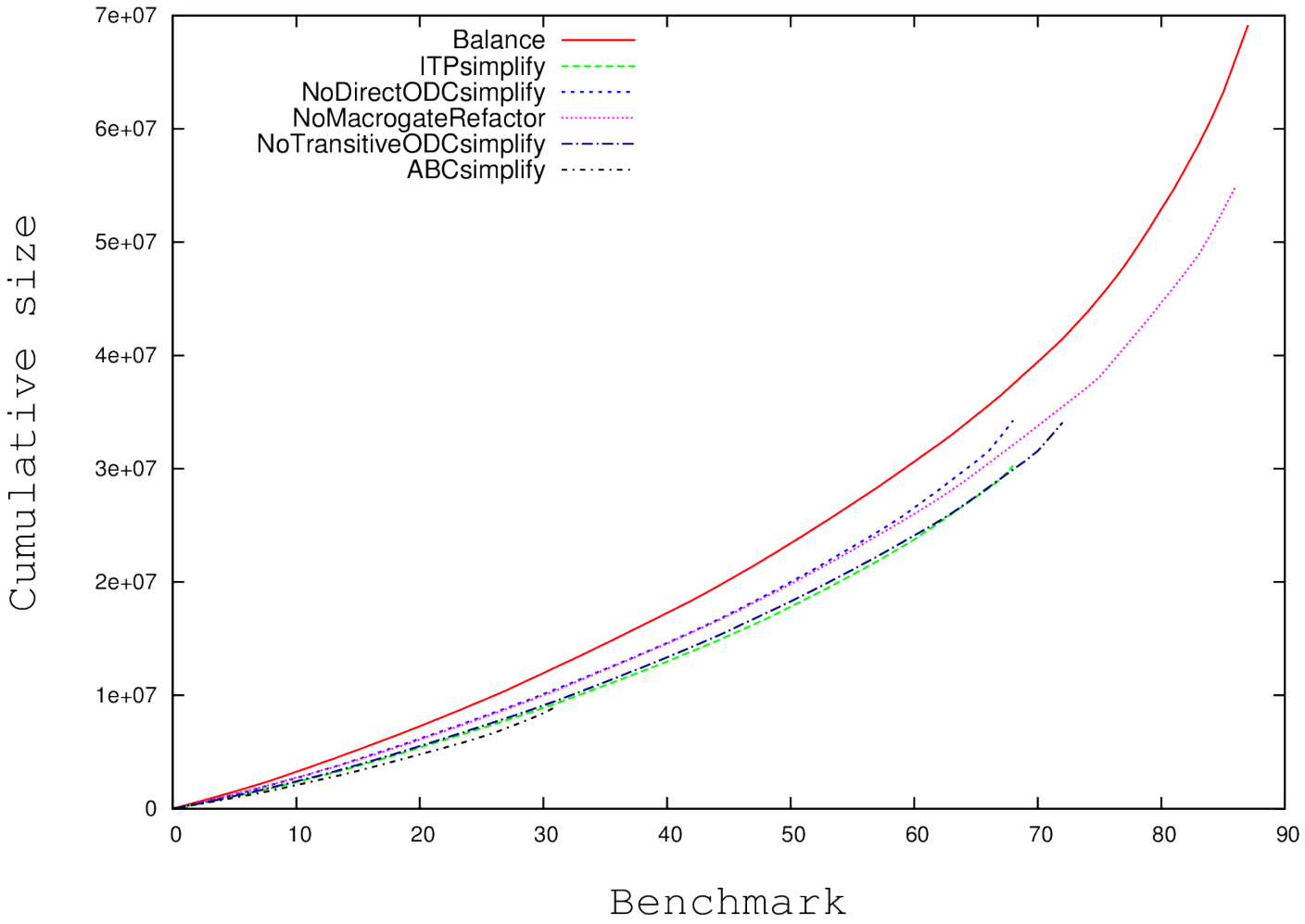
- Implementation on PdTrav tool
  - HWMCC ’07 to ‘15
- ITPs stored on files from MC runs
- Picked 87 instances
- *experiments also on ITPs from PeRIPLO (Sharygina’s group, Lugano)*

Logic synthesis

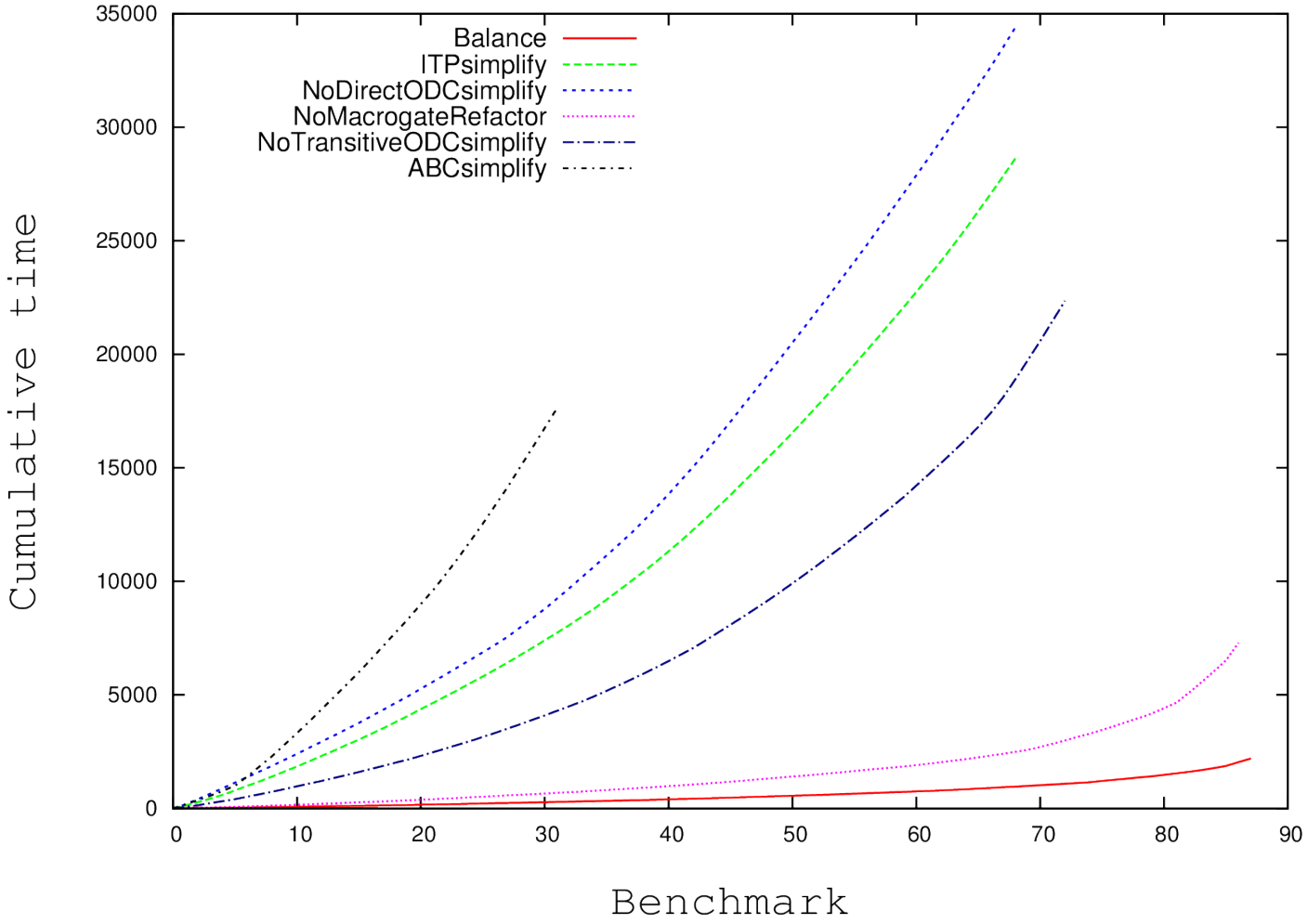
- Benchmark set (87 ITPs)
  - Size range: $4 \cdot 10^5$ – $8{,}5 \cdot 10^6$ gates
  - Average size: $2 \cdot 10^6$ gates
- Experimental set-up
  - Time limit: 900 seconds
  - Memory limit: 8 GB

	Completed Benchmarks	Average compaction rate	Average execution time
Balance	87	61.31 %	25.27 s
ITP Simplify	68	83.06 %	421.06 s
No Direct ODC	68	80.83 %	505.89 s
No Refactoring	86	69.28 %	84.80 s
No Transitive ODC	72	80.93 %	310.46 s
ABC	31	94.96 %	568.98 s

Cumulative size



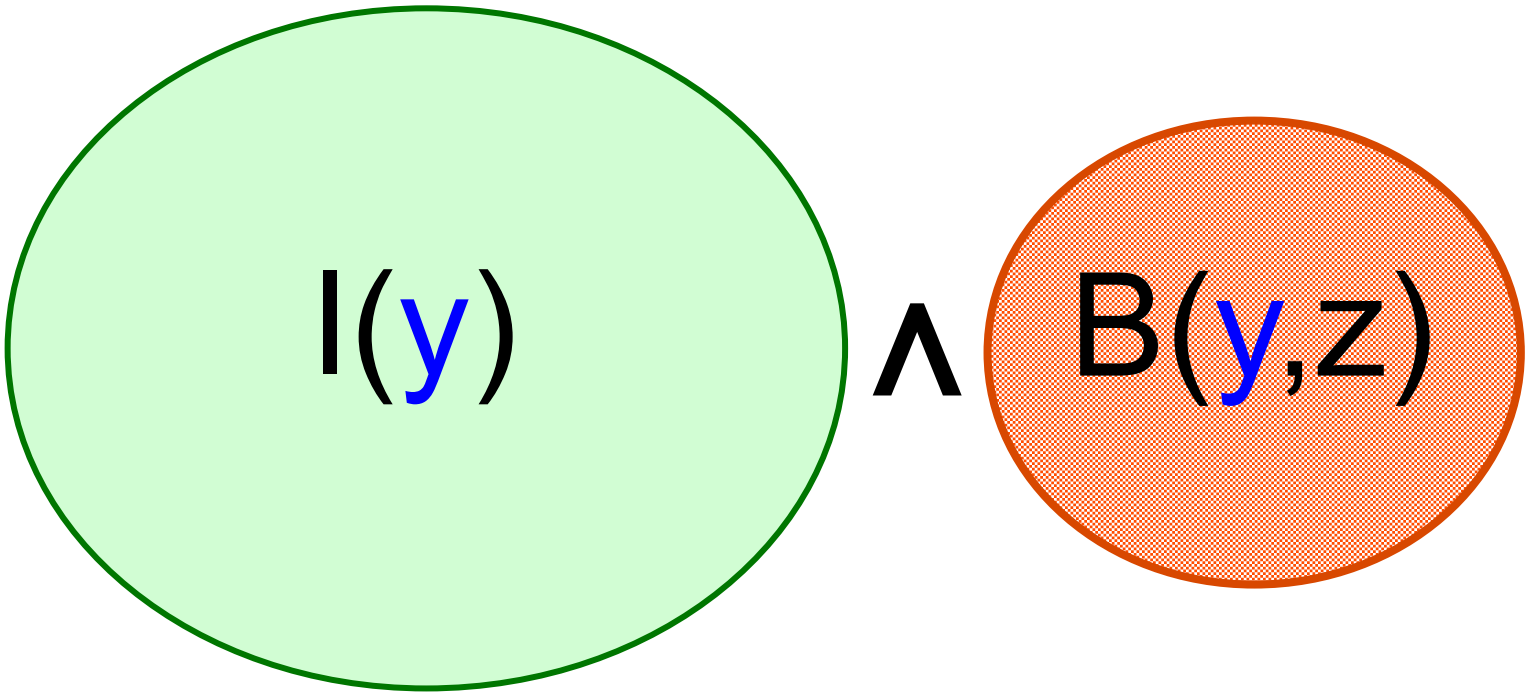
Cumulative time



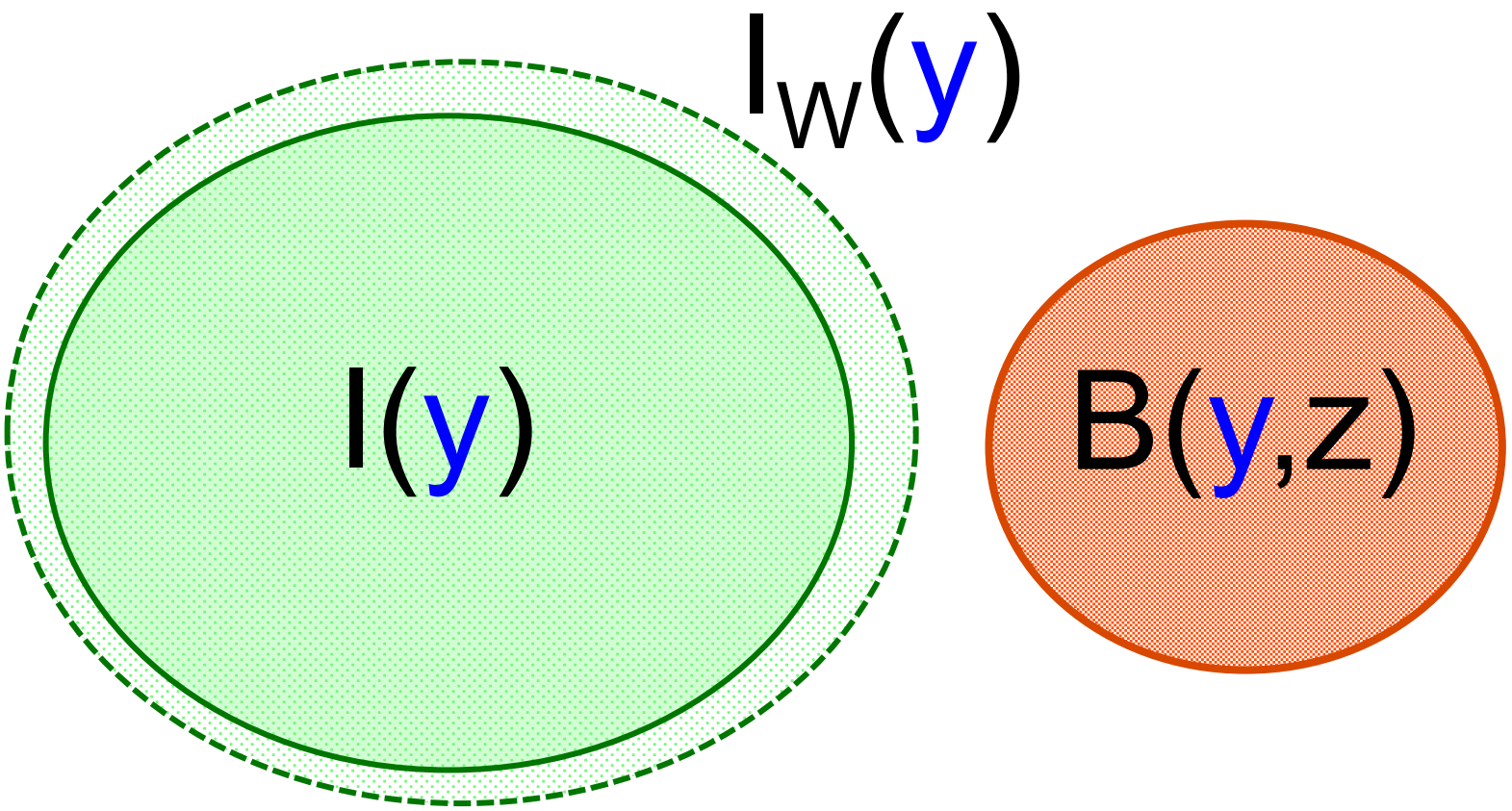
ITP weakening by SAT GLA

- GLA: Gate Level Abstraction
  - used here to abstract combinational circuit
- SAT-based abstraction (PBA)
  - SAT(I∧B) => UNSAT CORE => abstract
- Given cut var, monotonicity => existential quantification by constant substitution
- NNF encoding
  - monotone w.r.t. all circuit nodes except Pis.

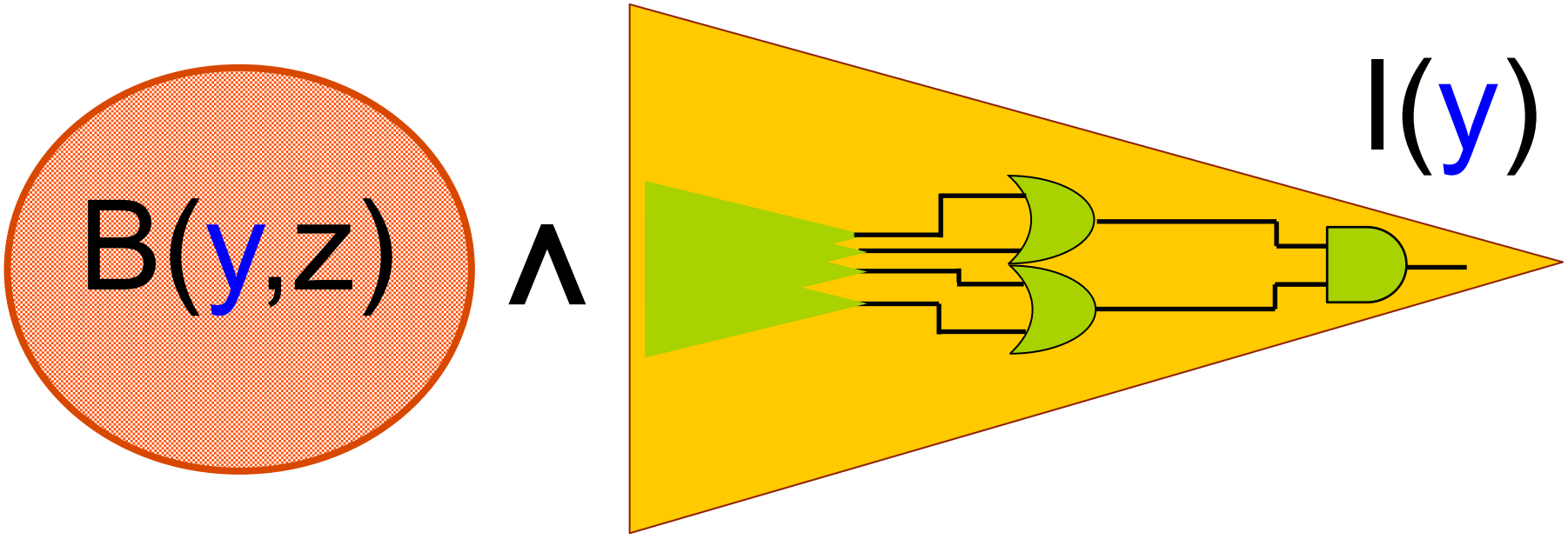
Interpolant weakening



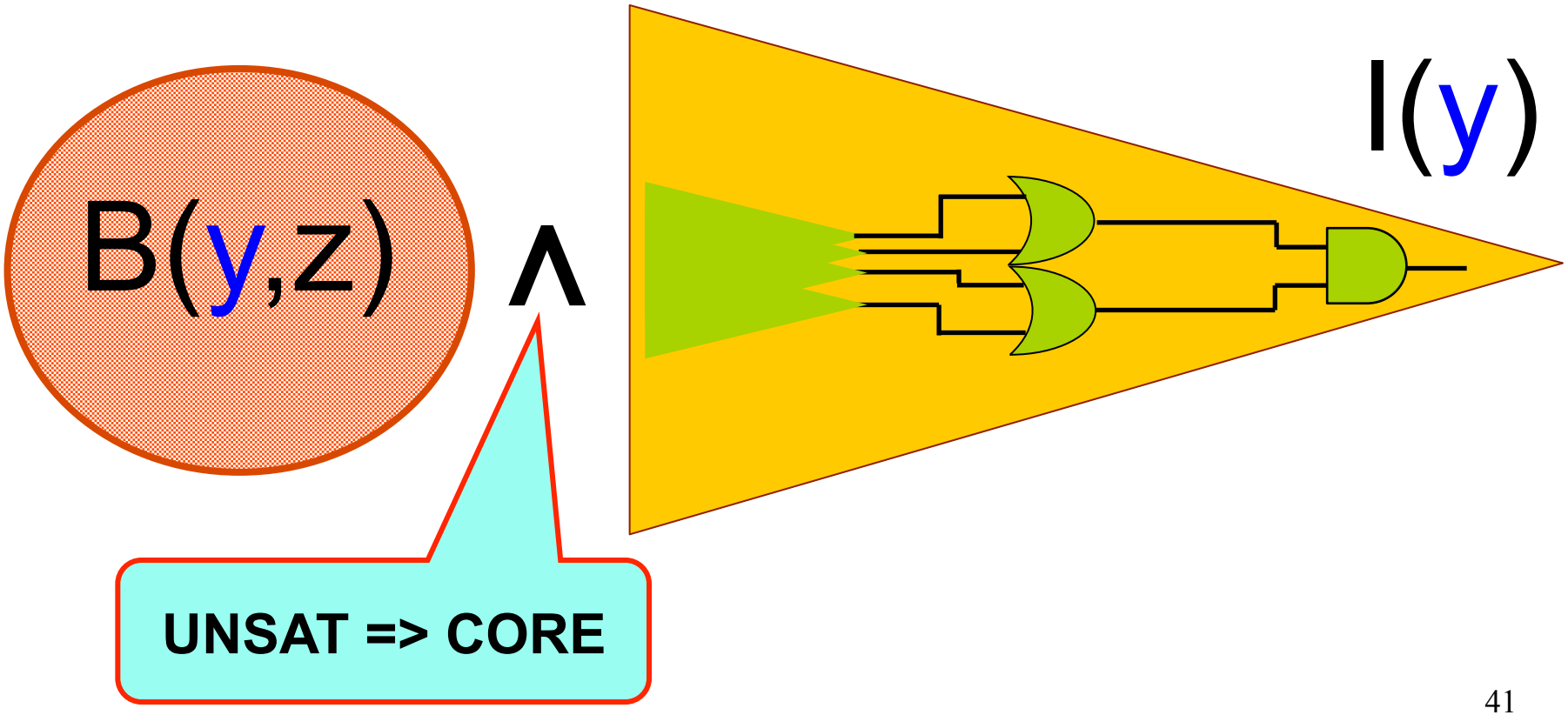
Interpolant weakening



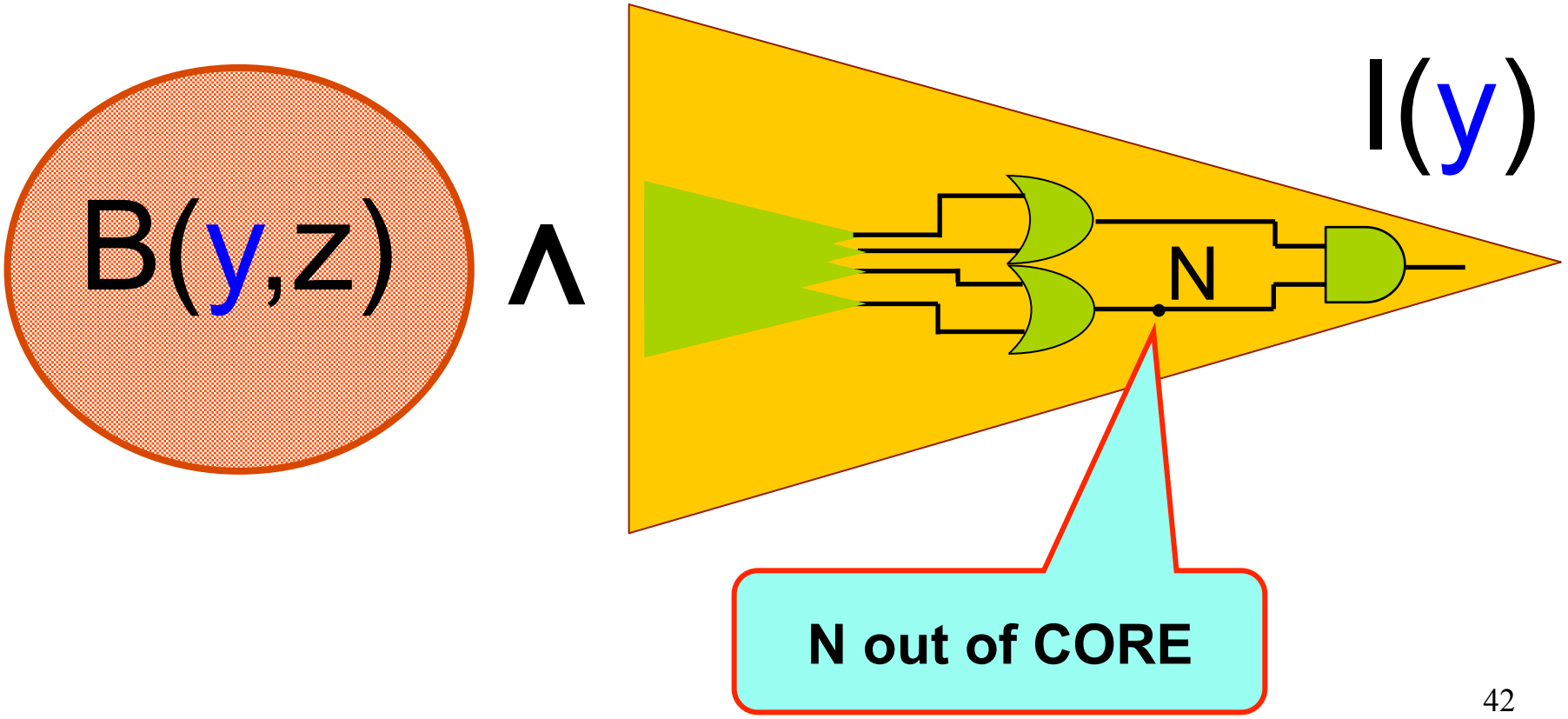
Interpolant weakening



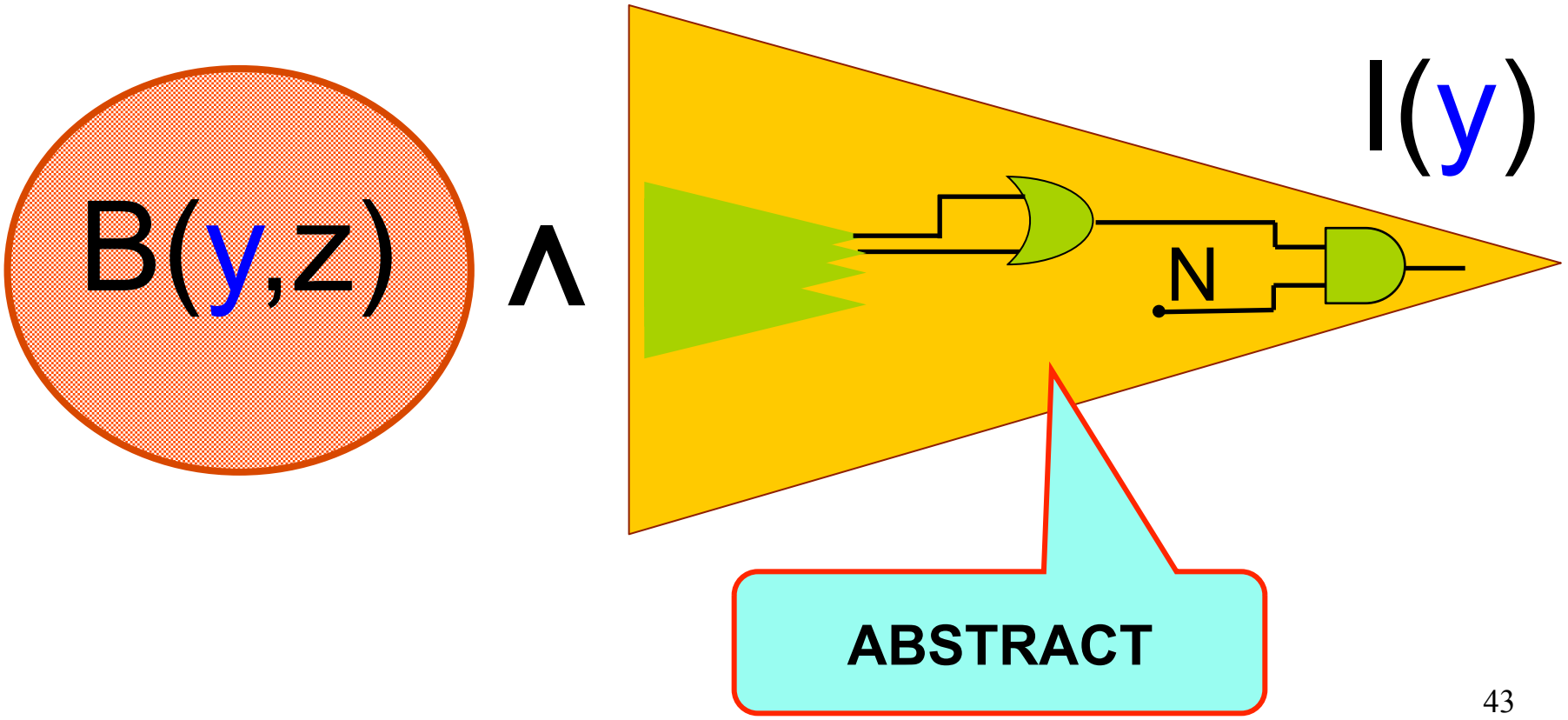
Interpolant weakening



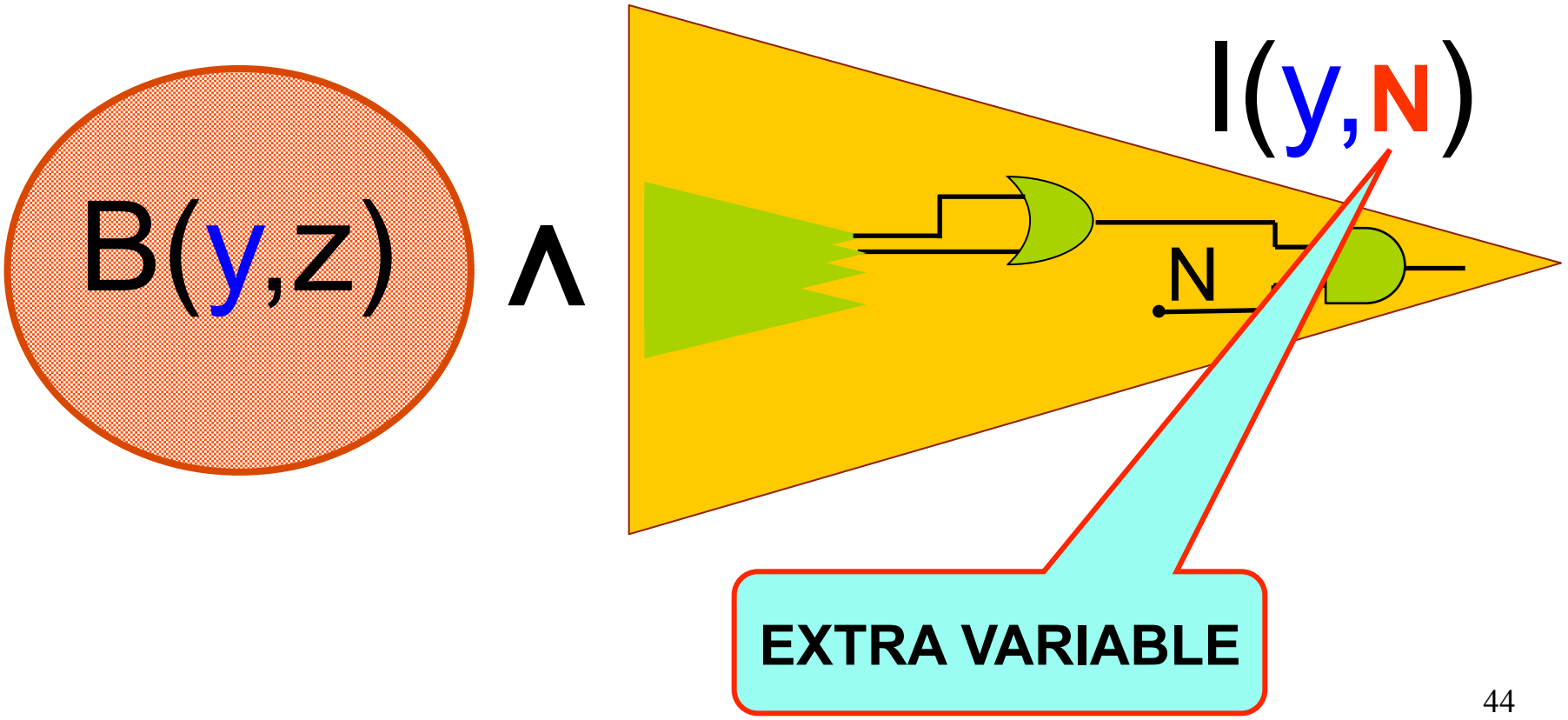
Interpolant weakening



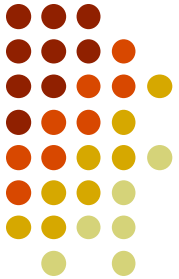
Interpolant weakening



Interpolant weakening



Existential quantification



NNF encoding

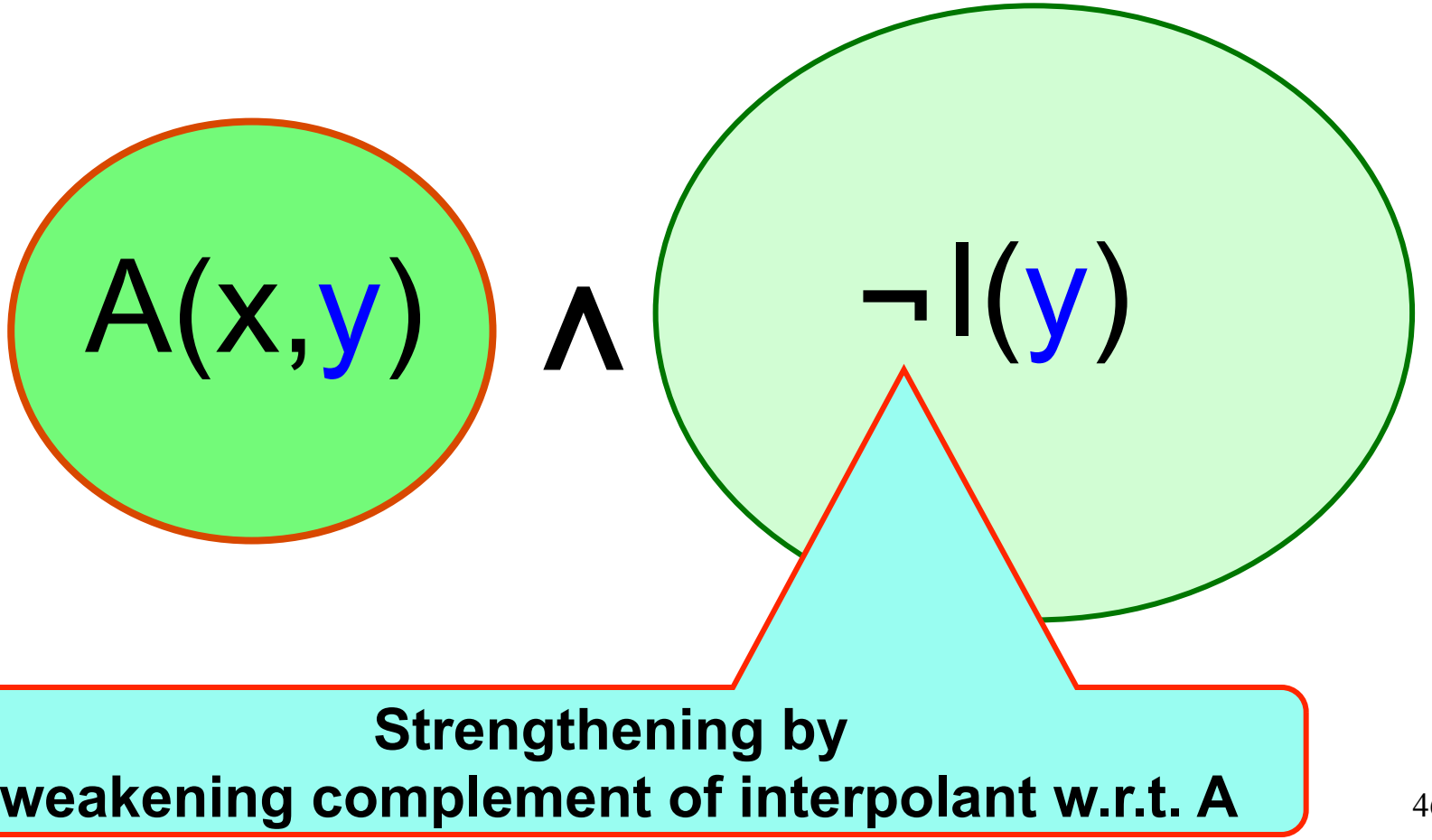
NNF circuit monotone w.r.t. internal nodes

Quantification by constant substitution

$$\exists_N I(Y,N) = I(Y,1)$$

1. $I(Y) \Rightarrow I_{NNF}(Y)$
2. $SAT(I_{NNF}(Y) \wedge B(Y,Z))$
3. $I_{NNF} \Rightarrow$ (uns. core, abstract, inject 1) $\Rightarrow I_{NNF,weak}$
4. $I_{NNF,weak} \Rightarrow I_{weak}$

Interpolant strengthening

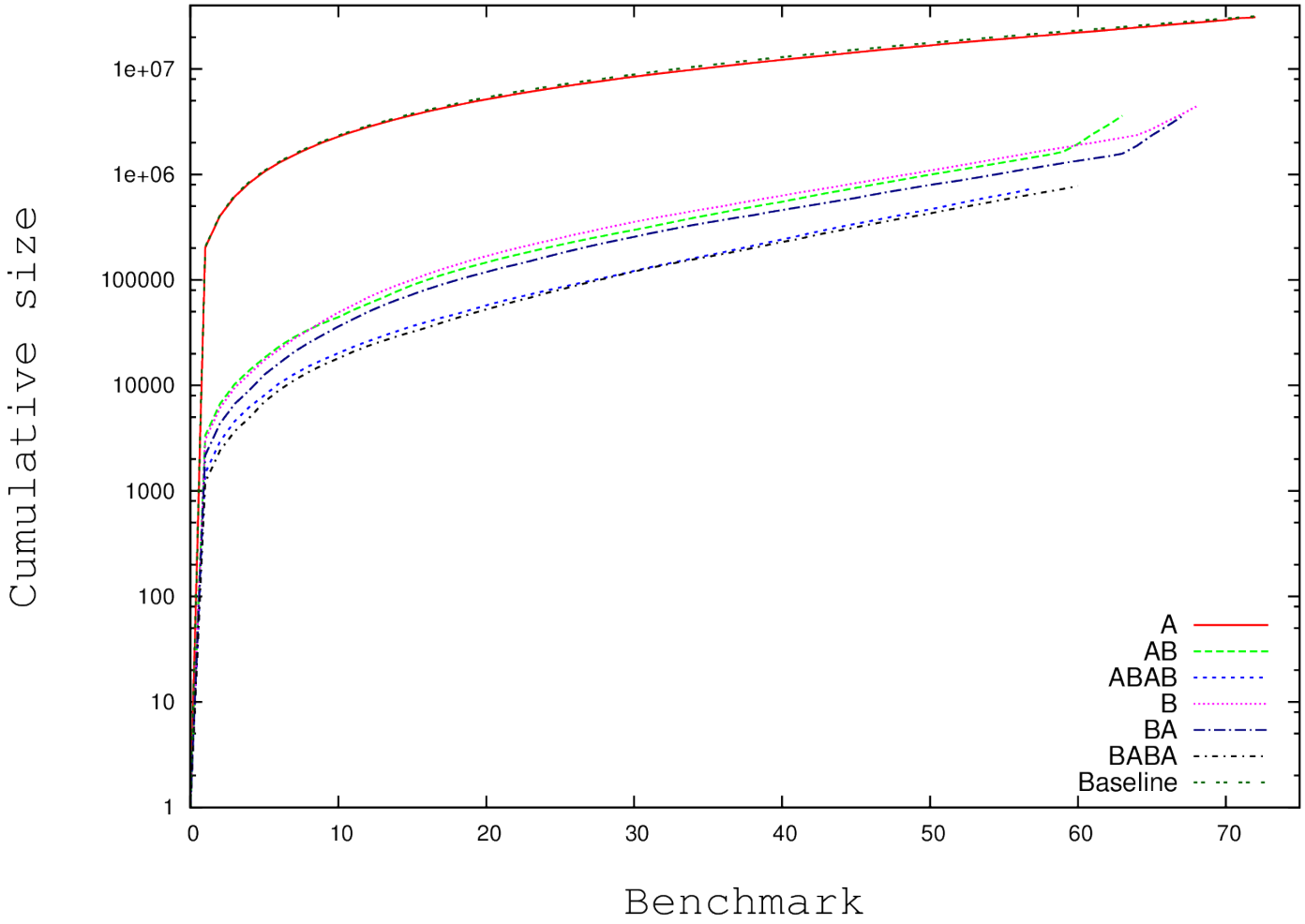


SAT-based weakening

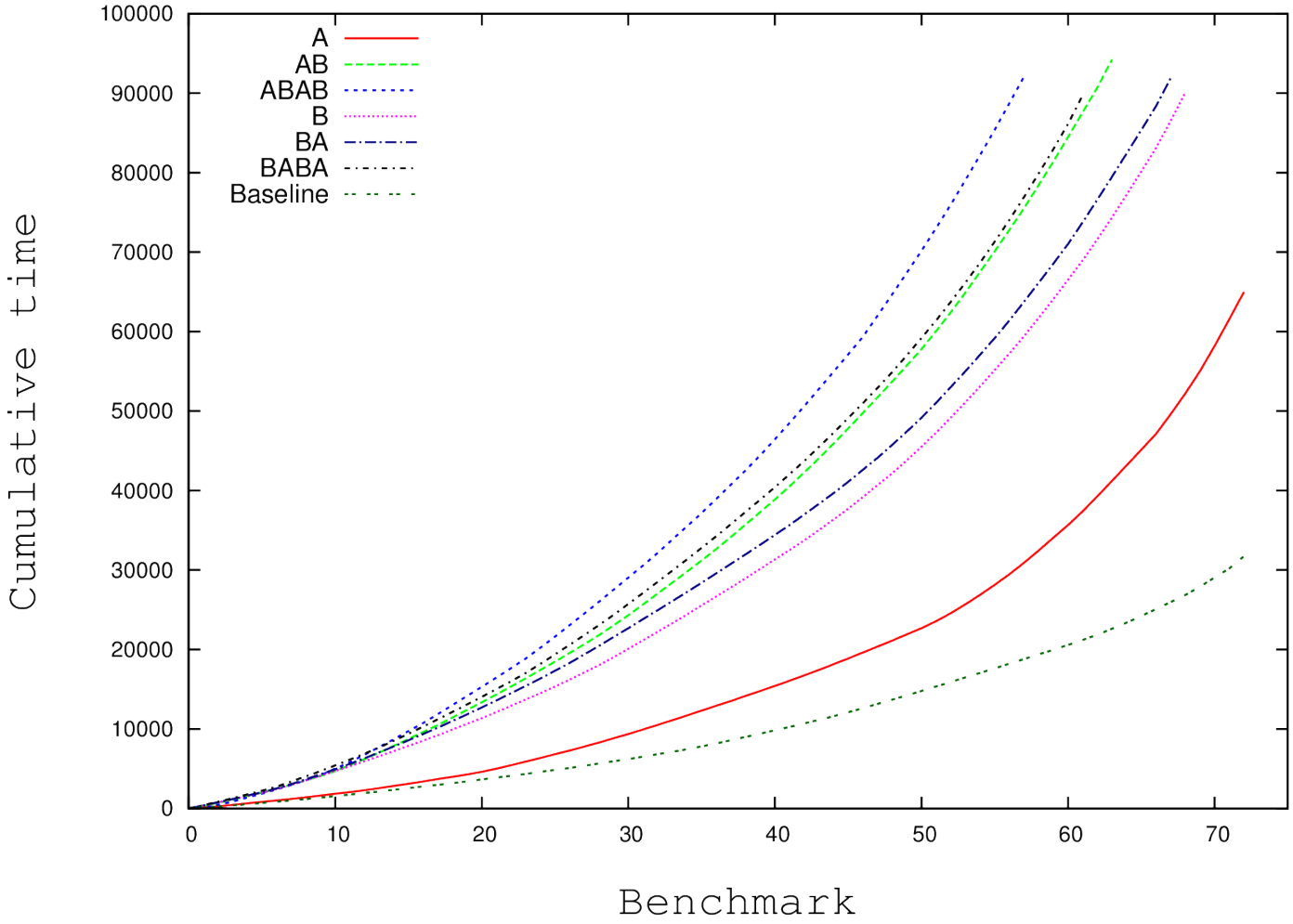
- Benchmark set (87 circuits)
  - Size range: $4 \cdot 10^5$ – $8{,}5 \cdot 10^6$ gates
  - Average size: $2 \cdot 10^6$ gates
- Experimental set-up
  - Time limit: 3600 seconds
  - Memory limit: 8 GB

	Completed Benchmarks	Average compaction rate	Average execution time
A	72	82.24 %	902.33 s
AB	63	97.99 %	1495.78 s
ABAB	57	99.59 %	1649.79 s
B	68	97.53 %	1324.51 s
BA	67	98.03 %	1371.30 s
BABA	60	99.56 %	1470.84 s

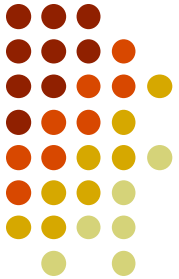
Cumulative size



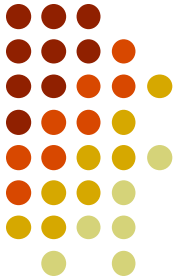
Cumulative time



Conclusions



- Contributions:
  - Adapting logic synthesis for fast/scalable ITP compaction
  - More expensive SAT-based compaction (weakening/strengthening)
- Evaluation
  - Fast synthesis always used
  - Expensive weakening/strengthening
    - Avoid memory explosion (or time not critical)
    - Strength may impact on quality of result



Thank you!