



Linear-time Reductions of Resolution Proofs



Omer Bar-Ilan
Shlomo Hoory

Oded Fuhrmann
Ohad Shacham



Technion

Ofer Strichman

Resolution

- Resolution:

$$\frac{(l \vee l_1) \quad (\neg l \vee l'_1)}{(l_1 \vee l'_1)}$$

- Modern SAT solvers are implicit resolution engines
 - Learn new clauses through resolution.
 - Upon request, they produce a resolution proof.

Uses of the resolution proofs

- Extraction of unsatisfiable core
 - The subset of original clauses that were used in the proof
- Computing Interpolants
 - For unbounded SAT-based model checking
- Incremental satisfiability
 - Which learned clauses can be reused in the next instance

The smaller the better

- Many techniques for shrinking the proof / core
 - Minimum core: Σ_2 -complete
 - Minimal core: [B03,H05,LM04,OMASM04,..]
 - Small core: [ZM03, GKS06]
 - All exponential
 - Most popular: run-till-fix
- Smaller proofs $\overset{?}{\rightarrow}$ shorter verification time
- A good criterion:

By how much can you shrink the core in the first T sec?

In this work we investigate...

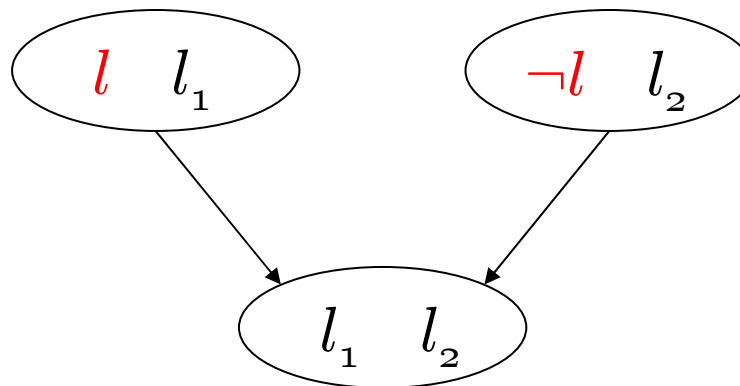
Linear-time Reductions of Resolution Proofs

- (linear in the size of the proof graph)
- We propose two techniques:
 1. Recycle – units
 2. Recycle – pivots

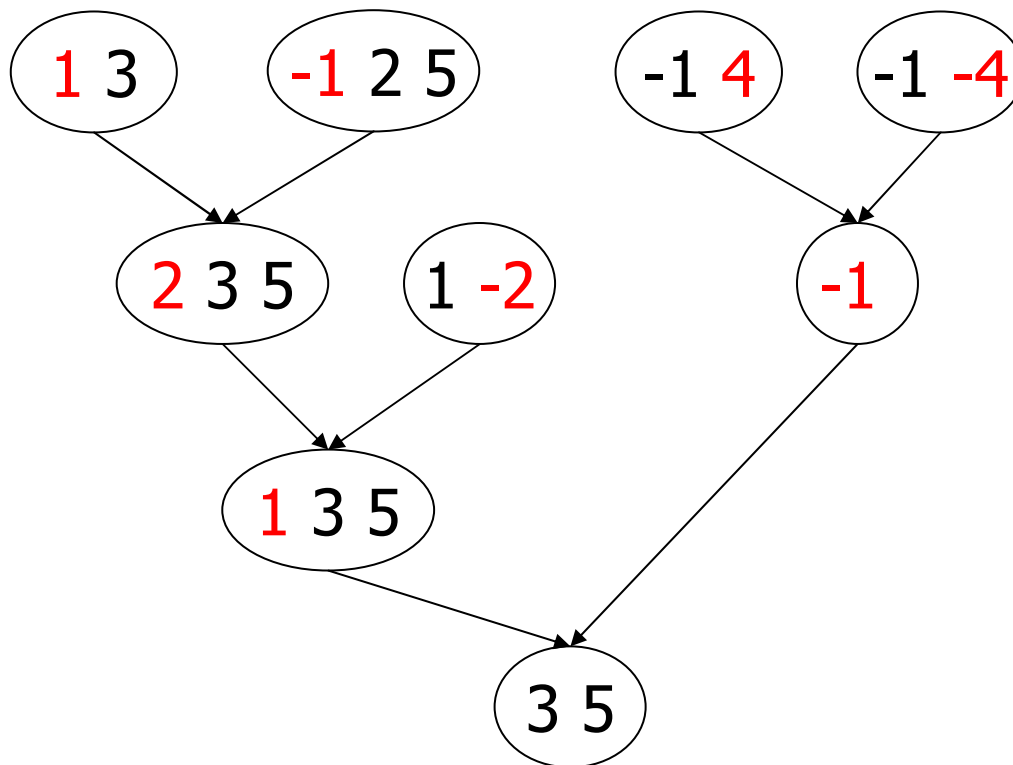


1. Recycle-units / observation

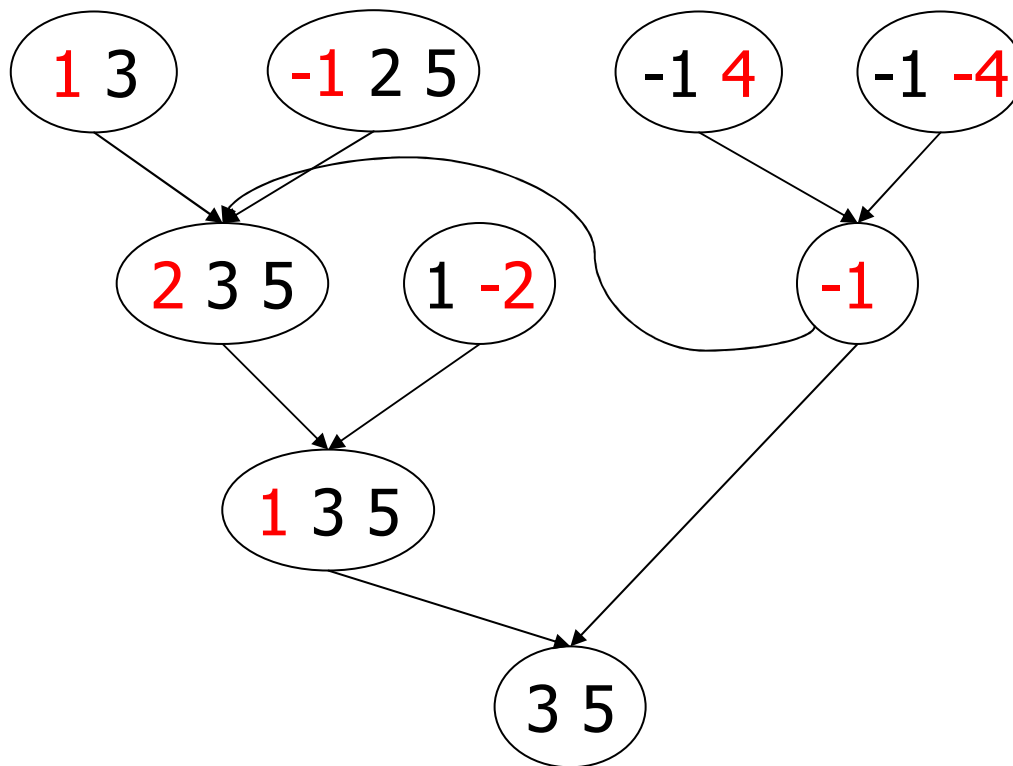
- When learning (resolving) a new clause in SAT,
 - The resolving clauses are not satisfied
 - Hence, the resolution-variable is unassigned
 - What if we learn it later on ?



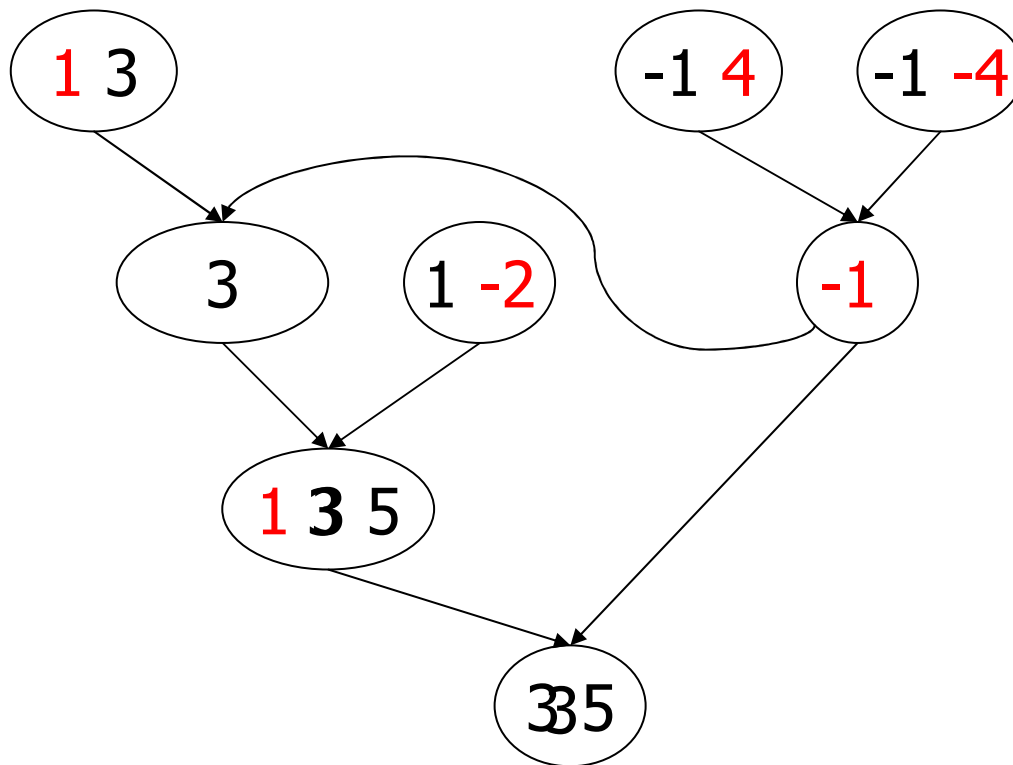
1. Recycle-units / easy case



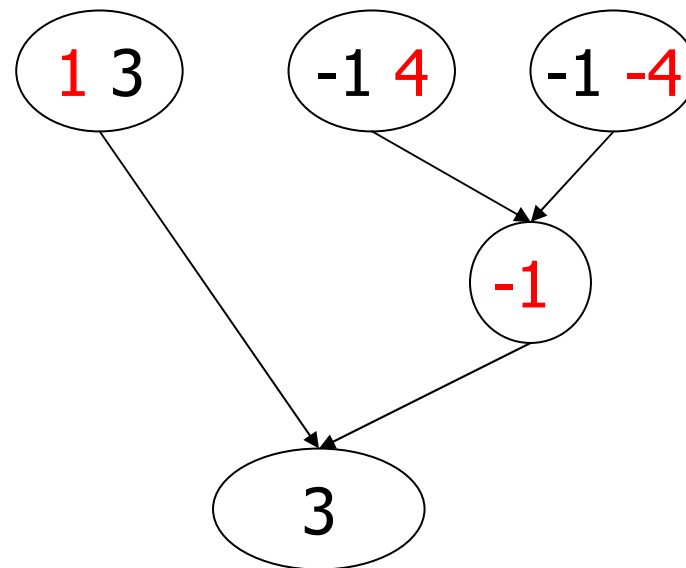
1. Recycle-units / easy case



1. Recycle-units

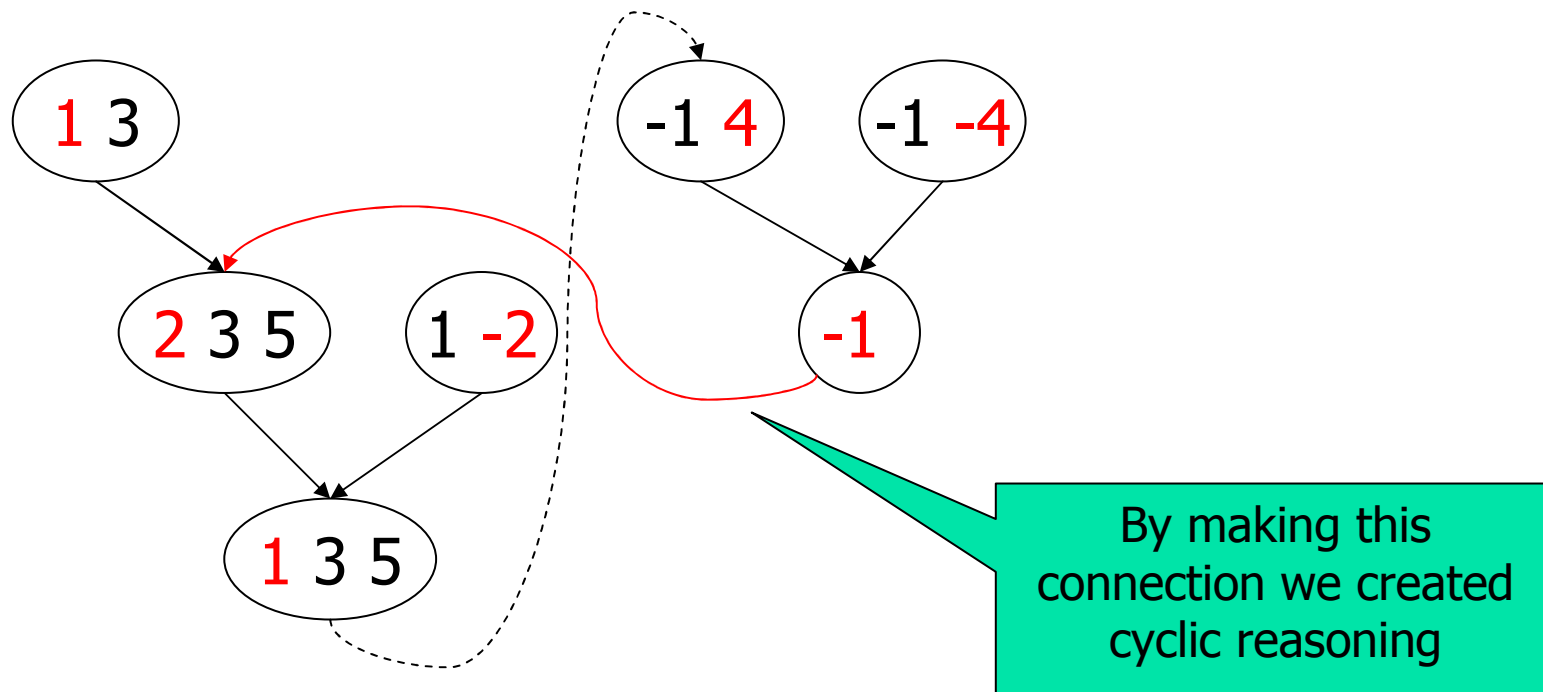


1. Recycle-units



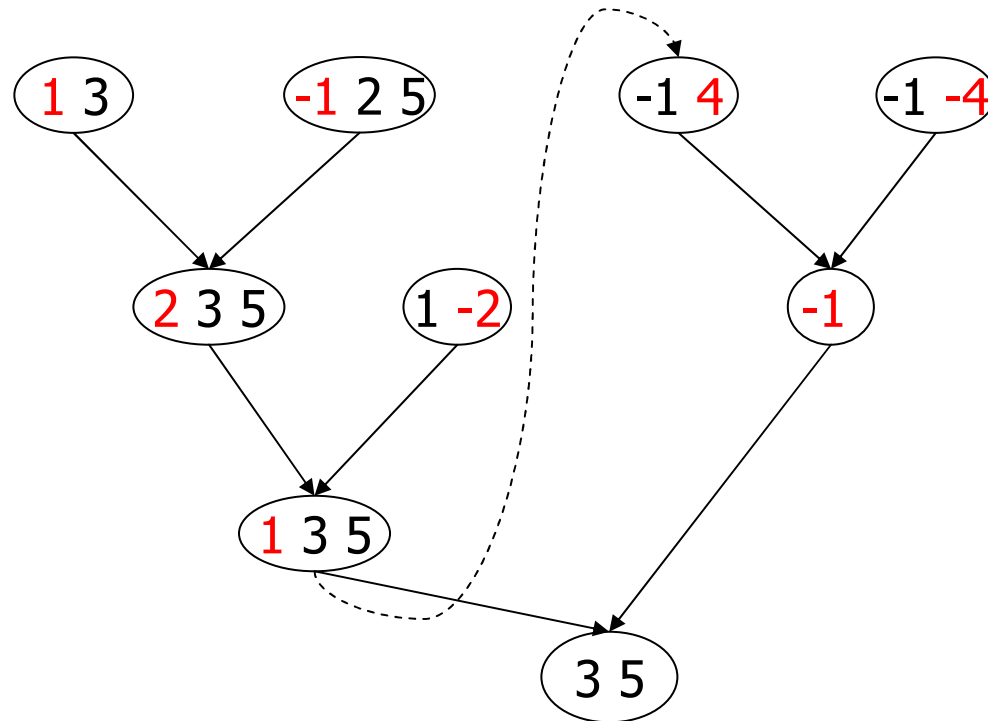
- Reduced proof by 4 clauses
- Reduced core by 1 clause

1. Recycle-units / beware of cycles



1. Recycle-units / beware of cycles

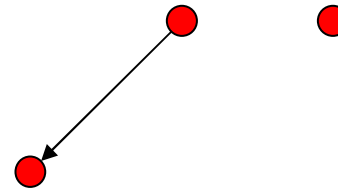
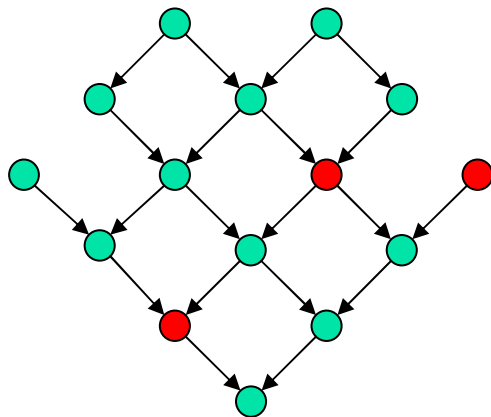
- Solution:
 - mark antecedents of units
 - apply only to marked nodes



1. Recycle-units / beware of cycles

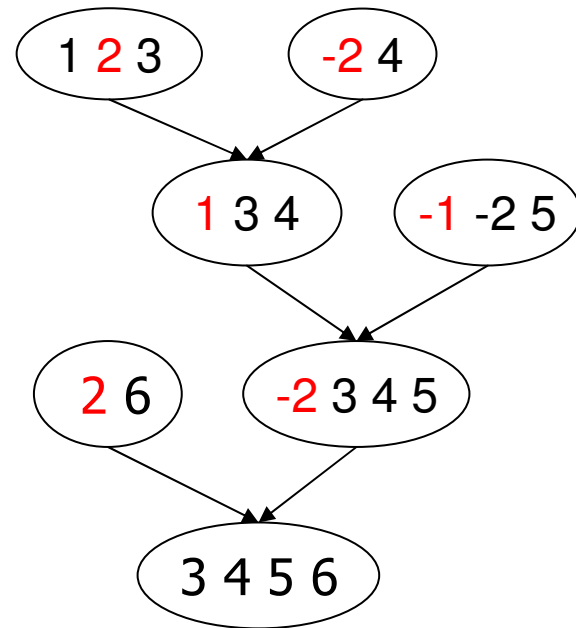
- A little tricky to make efficient.
 - The graph changes all the time.
 - Inefficient to update antecedents relations.
- Strategy
 - Maintain a projection of the graph to units.

1. Recycle-units / maintain a units graph

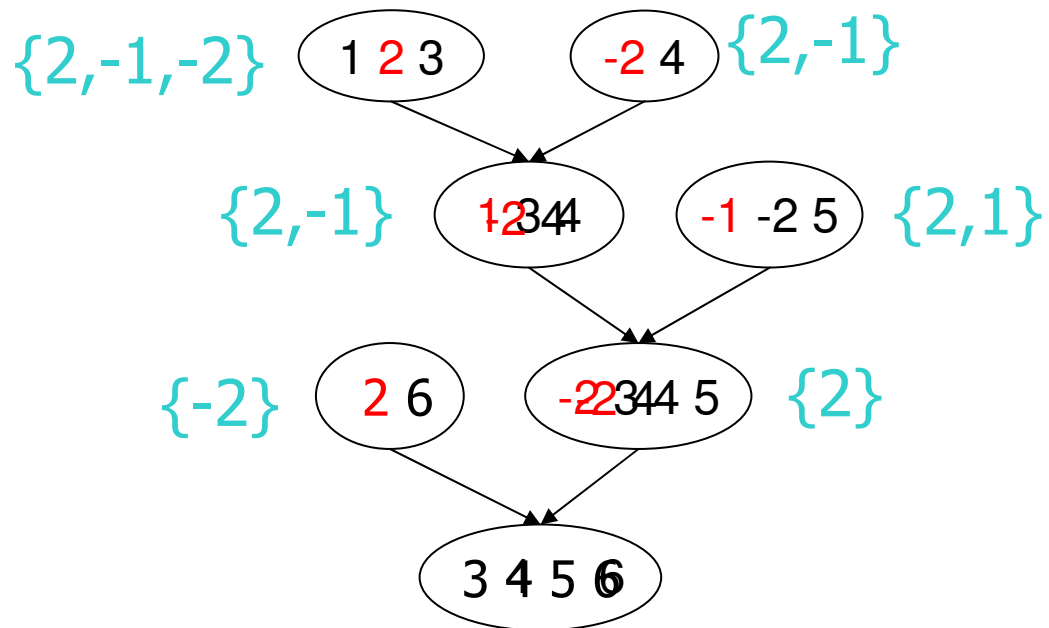


- unit
- other

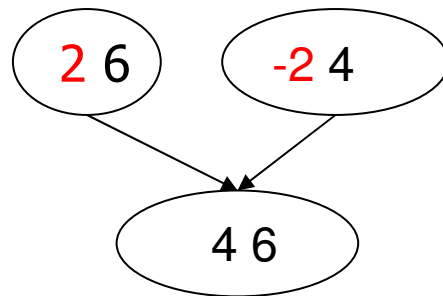
2. Recycle-pivots / Example (tree)



2. Recycle-pivots / Example (tree)



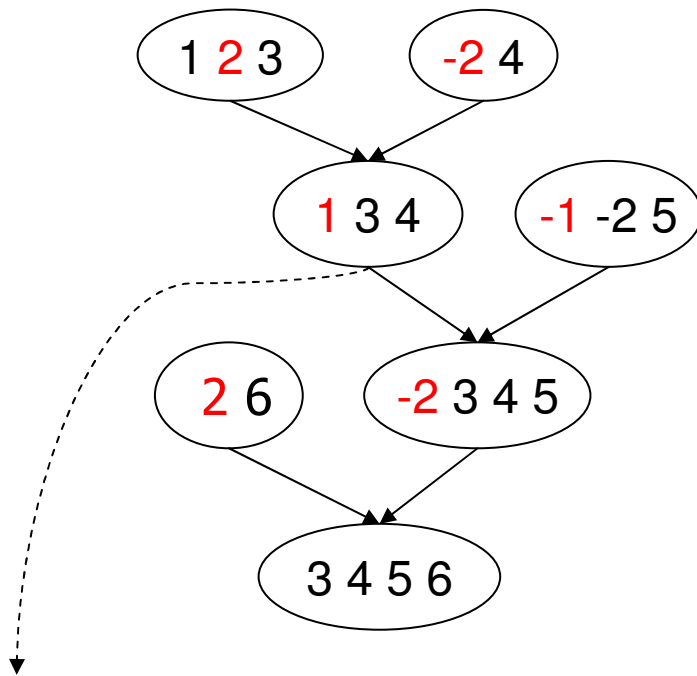
2. Recycle-pivots / Example (tree)



Reduced proof by 4 clauses
Reduced core by 2 clauses

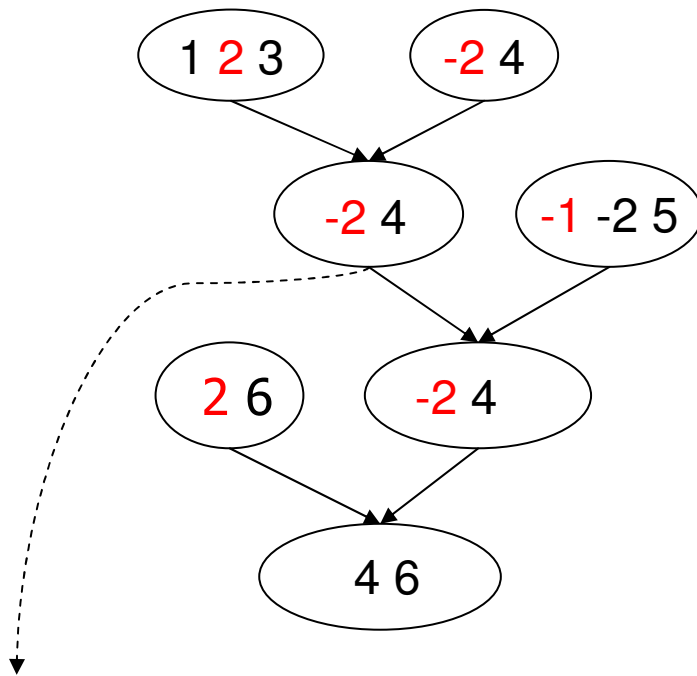
2. Recycle-pivots / DAGs

- Resolution graphs are DAGs
 - So, a node is on more than one path to the empty clause

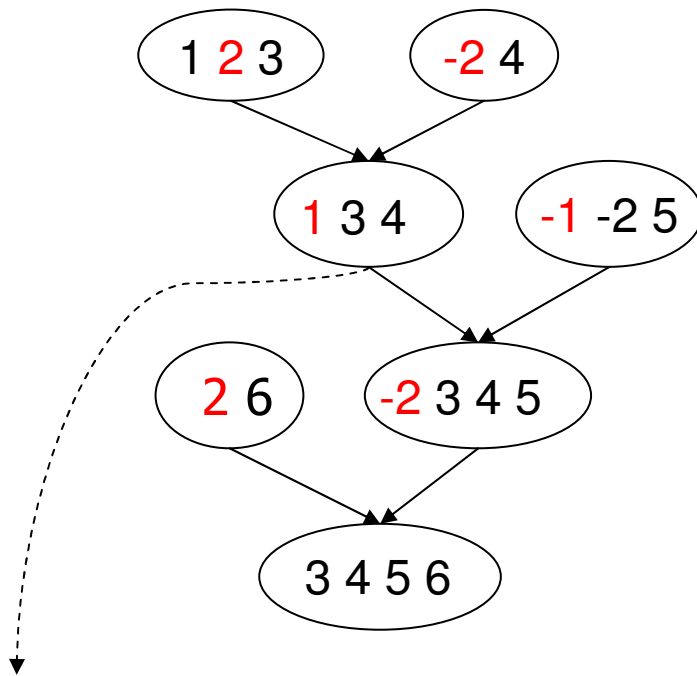


2. Recycle-pivots / DAGs

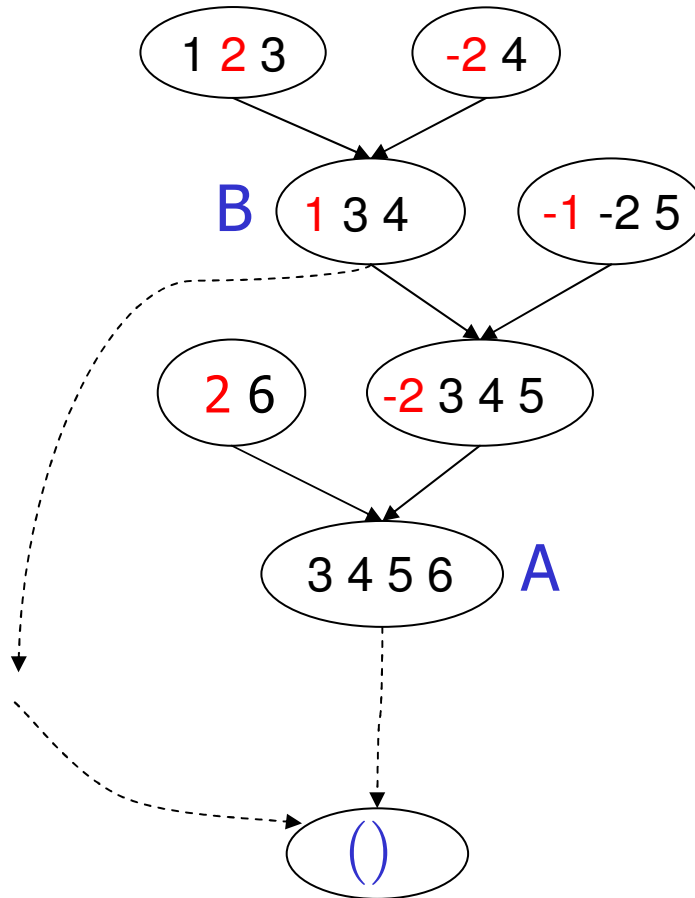
- Resolution graphs are DAGs
 - So, a node is on more than one path to the empty clause



2. Recycle-pivots / DAGs



2. Recycle-pivots / DAGs



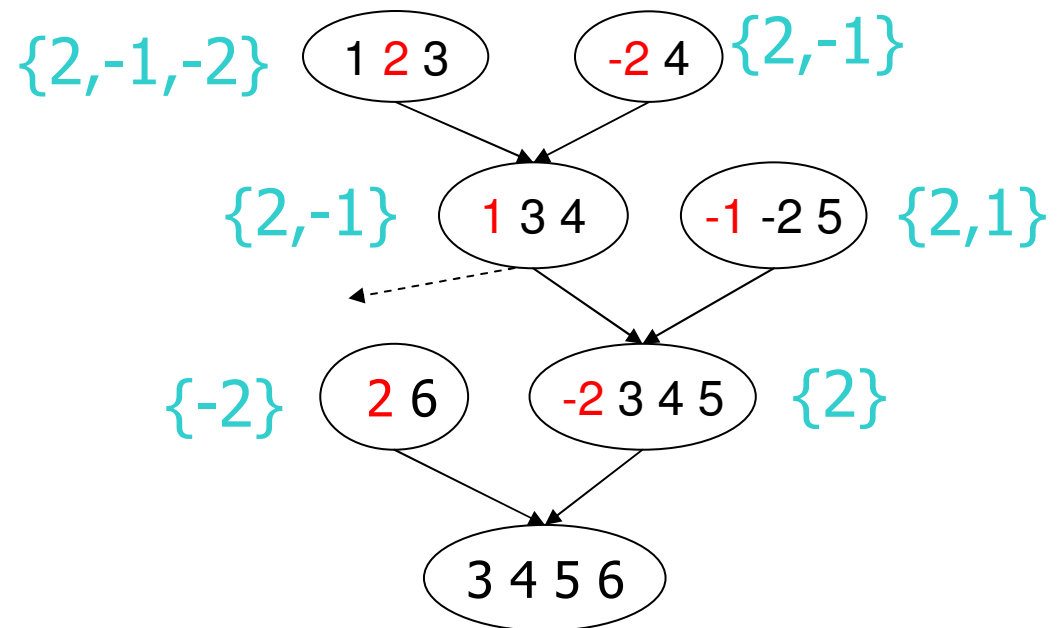
Does A dominate B ?

Dominance relation
can be found in
 $O(|E| \log |V|)$

Problem: need to be
updated each time.

2. Recycle-pivots / DAGs

- Our current implementation:
 - Stop propagating information across nodes with more than one child.




2. Recycle-pivots / Regular resolution

- Regular resolution (Tseitin 1970): no pivot is used twice along a path.
- Computationally weaker than general resolution
 - There are formulas in which regular proof \gg general proof
 - Because sometimes this forces a tree resolution
- We make the graph regular as long as it does not require splitting nodes

Experiments / Core-size

- 67 unsat instances from the public IBM benchmarks that took run-till-fix more than 10 sec.



		Leaves			
		$(\times 10^6)$			
Reduction	Time	Before	After	Per sec	Ratio
run-till-fix	8095	1.00	0.53	57.9	0.53
units	1002.5	1.00	0.99	5.2	0.99
pivots	32.5	1.00	0.95	1518.6	0.95
units + pivots	1235.8	1.00	0.94	43.4	0.95

Experiments / Proof-size

- 67 unsat instances from the public IBM benchmarks that took run-till-fix more than 10 sec.

		Nodes			
		$(\times 10^6)$			
Reduction	Time	Before	After	per sec	Ratio
run-till-fix	8095	11.83	17.67	-722.2	1.49
units	1002.5	11.83	11.51	316.9	0.97
pivots	32.5	11.83	10.46	42059.2	0.88
units + pivots	1235.8	11.83	10.24	1281.3	0.87

Summary

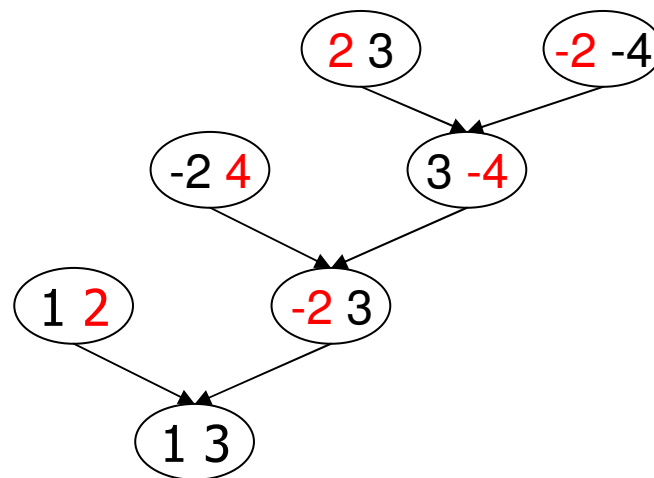
- Linear reductions are less effective than exponential ones
 - ...but worth it in the realm of short time outs.
- Double-pivot reduces proof size by 12% in no time.
- In use internally in IBM.
- Left to check:
 - Measure influence on interpolant-based prover.
 - Check combination with other minimization techniques.
 - Try to make efficient for incremental satisfiability.

SAT and resolution proofs

- Resolution is sound and complete for CNF formulas
- There exists a decision procedure that deduces the empty clause if and only if the input formula is unsatisfiable.
- Modern SAT solvers are implicit resolution engines
 - Learn new clauses through resolution.
 - Upon request, they produce a resolution proof.

2. Recycle pivots / Theory

- A restriction on general resolution: Regular resolution
 - no pivot is used twice along a path.



2 is used twice

Not Regular