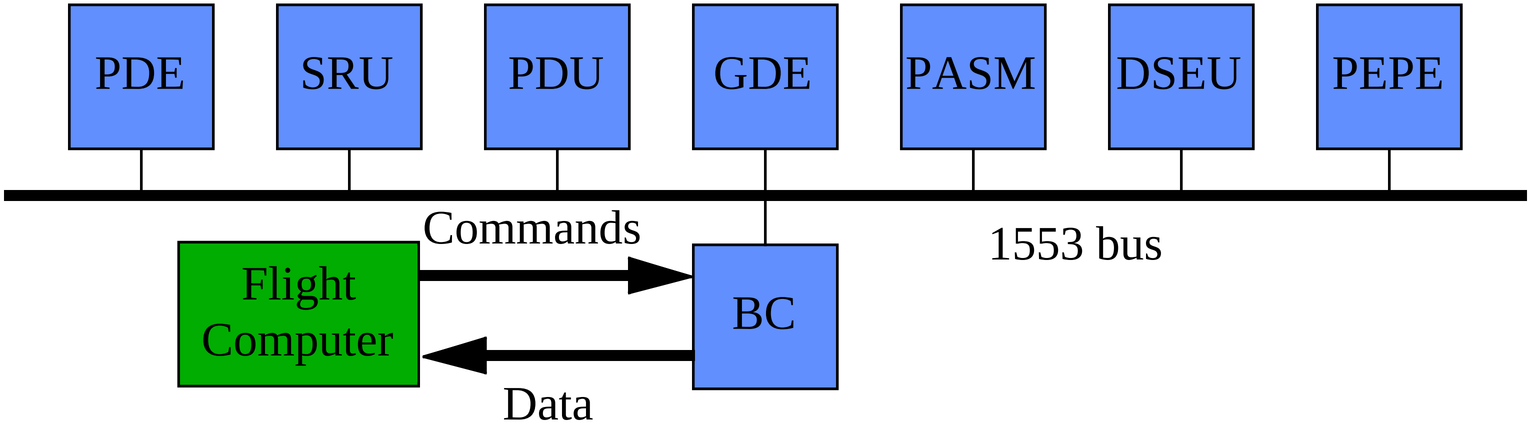
Example: DS-1 bus communication



- Some of the clauses describing bus communication

$C_1: \neg\, nci \quad\quad \neg\, a \quad\quad nco$ $\quad C_4: \neg\, rf \quad\quad ia$ $\quad C_7: \neg\, ok \quad\quad \neg\, uf$

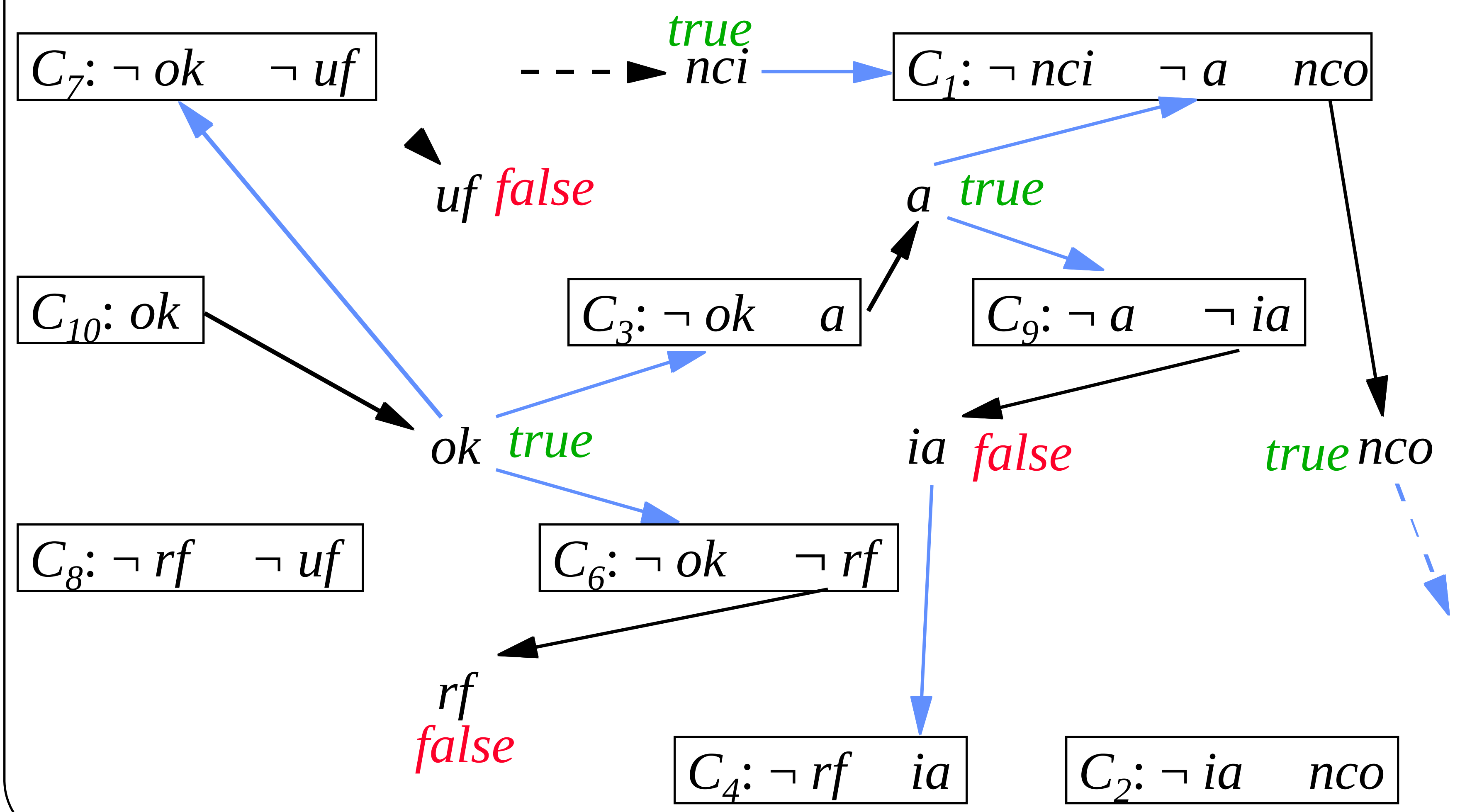
$C_2: \neg\, ia \quad\quad nco$ $\quad C_5: \neg\, uf \quad\quad ia$ $\quad C_8: \neg\, rf \quad\quad \neg\, uf$

$C_3: \neg\, ok \quad\quad a$ $\quad C_6: \neg\, ok \quad\quad \neg\, rf$ $\quad C_9: \neg\, a \quad\quad \neg\, ia$

BC health: *ok*, *rf*, *uf* BC activity: *a*, *ia*

No input cmd: *nci* No output cmd: *nco*

LTMS inference



C_7: ¬ *ok* ¬ *uf*

true
nci

C_1: ¬ *nci* ¬ *a* *nco*

uf *false*

a *true*

C_{10}: *ok*

C_3: ¬ *ok* *a*

C_9: ¬ *a* ¬ *ia*

ok *true*

ia *false*

true *nco*

C_8: ¬ *rf* ¬ *uf*

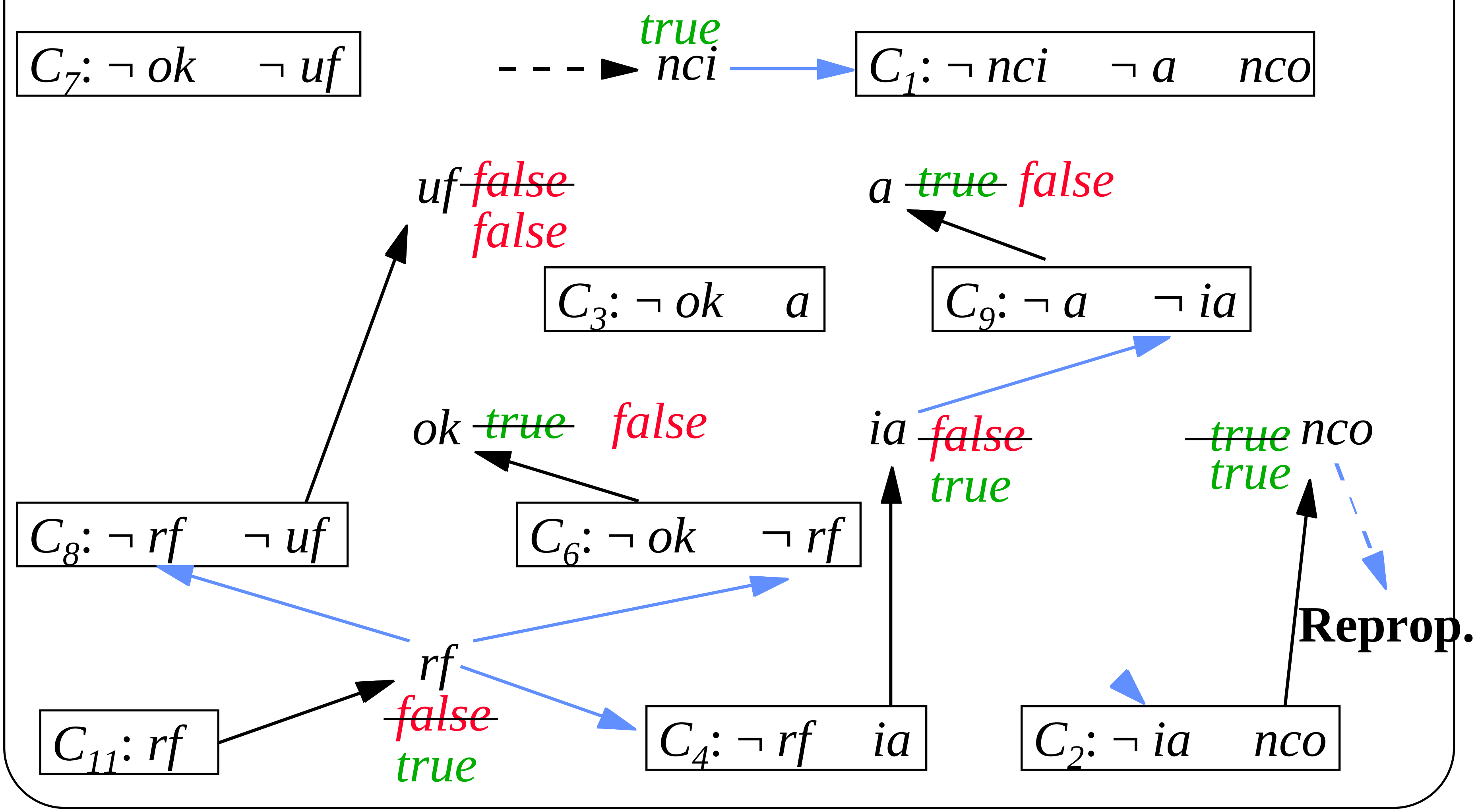
C_6: ¬ *ok* ¬ *rf*

rf
false

C_4: ¬ *rf* *ia*

C_2: ¬ *ia* *nco*

Deleting C_{10}, adding C_{11}



C_7: ¬ *ok* ¬ *uf*

true *nci*

C_1: ¬ *nci* ¬ *a* *nco*

uf ~~false~~ false

a ~~true~~ false

C_3: ¬ *ok* *a*

C_9: ¬ *a* ¬ *ia*

ok ~~true~~ false

ia ~~false~~ true

~~true~~ *nco* true

C_8: ¬ *rf* ¬ *uf*

C_6: ¬ *ok* ¬ *rf*

Reprop.

rf ~~false~~ true

C_{11}: *rf*

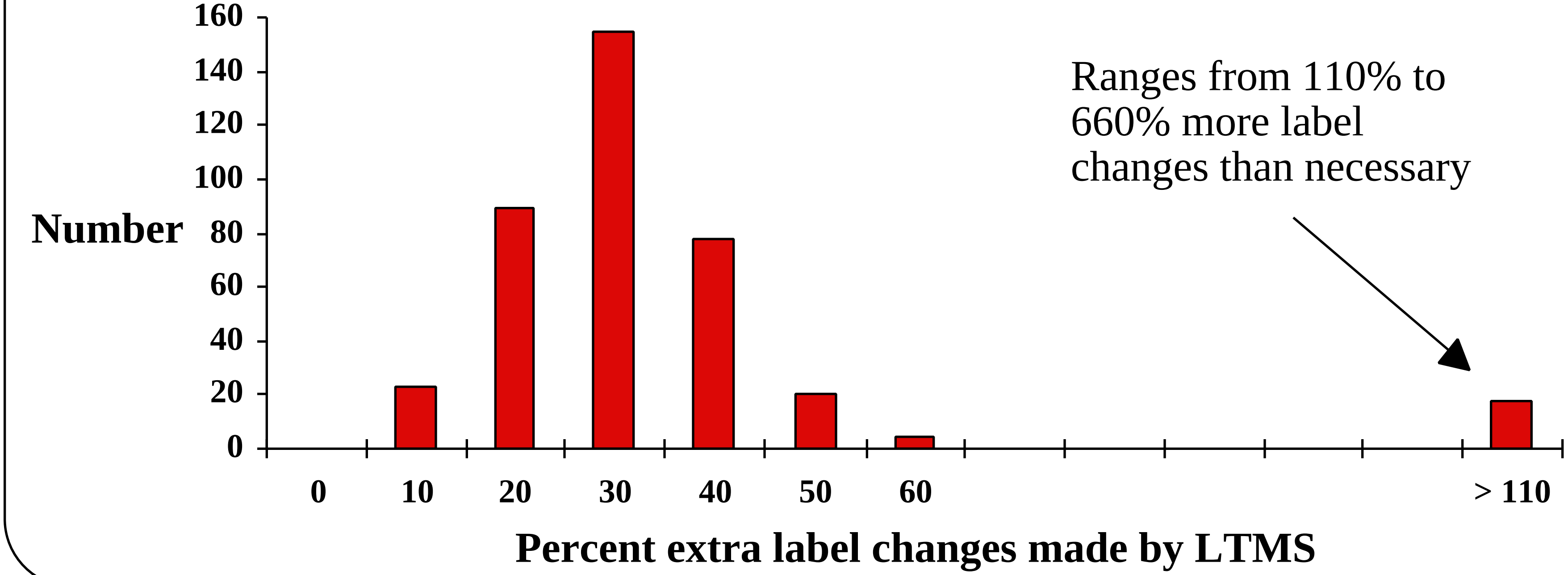
C_4: ¬ *rf* *ia*

C_2: ¬ *ia* *nco*

Summary of problem

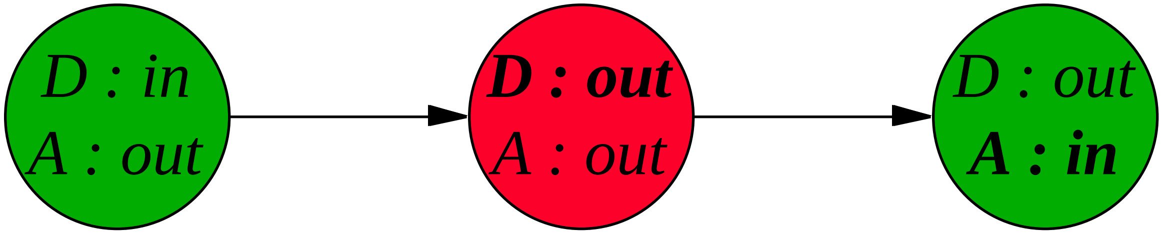
- **Problem:** Unnecessary repropagation during a context switch
- Excessive repropagation can be a sigificant problem in practice

Data from 387 distinct context switches on DS-1 theory containing 12,693 clauses



Cause of problem

- To guarantee *well-founded support*, the LTMS context switch algorithm is overly conservative
  - switching a context by deleting clause *D* and adding clause *A*



- **Previous solution:** Use an ATMS
  - context switch requires *no* label propagation
  - labeling algorithm is exponential in time and space
- **New solution:** Use an ITMS

ITMS algorithm intuitions

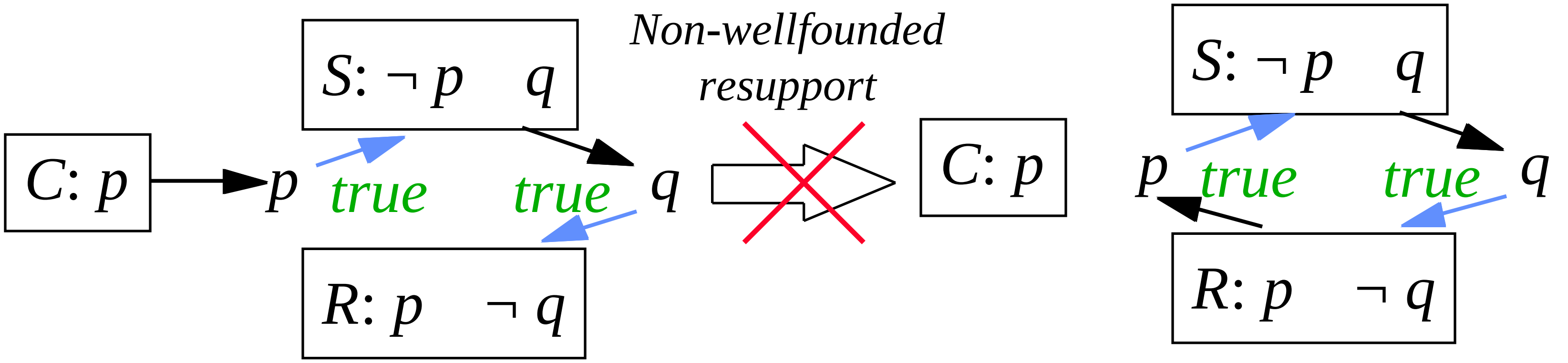
- *Resupport* propositions during clause deletion
  - if resupport is possible, proposition’s consequences are not touched
  - must guarantee *well-founded support*
- …but resupport available only after clause addition
  - so add new clause and propagate *before* deleting old clause
- …but added clause is often a conflict and propagation terminates
  - develop a new algorithm to *propagate through conflicts*

Top-level ITMS algorithm

```
procedure switch-context($D$, $A$, $\Sigma$)    // delete $D$, add $A$ to $\Sigma$
    Add $A$ to $\Sigma$ and propagate any unit clauses
    if conflicting clause detected then
        while there is a conflict that can be propagated do
            Propagate through the conflict
            Use propagated label to resupport propositions (if possible)
        endwhile
    endif
    Delete $D$ from $\Sigma$
    Propagate any unit clauses
end switch-context
```

Resupporting a proposition

- Clause *R* can resupport proposition *p* which is currently supported by clause *C* if
  - *p* occurs with the same sign in *R* and *C*
  - all other literals in *R* are *false*
  - resulting support is *well-founded*



C: *p*

S: ¬ *p* *q*

R: *p* ¬ *q*

p true true *q*

Non-wellfounded resupport

C: *p*

S: ¬ *p* *q*

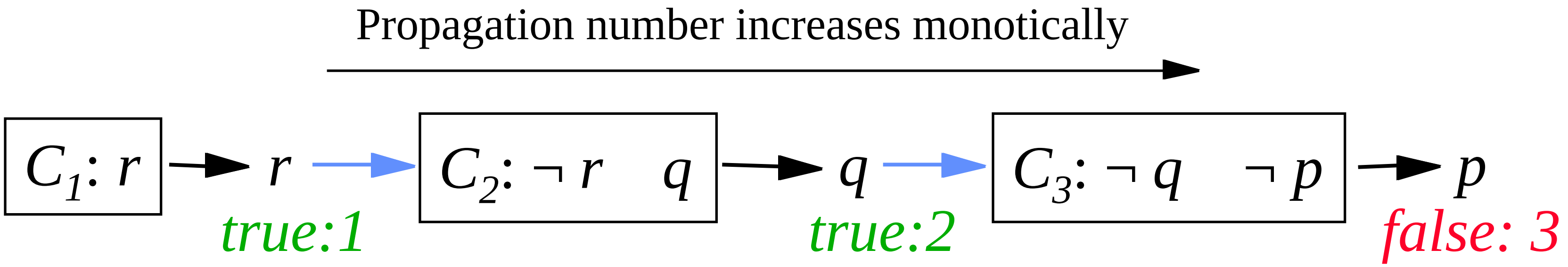
p true true *q*

R: *p* ¬ *q*

- Guaranteeing well-founded support is linear in the size of Σ
  - defeats the very purpose of using an ITMS

Propagation numbers

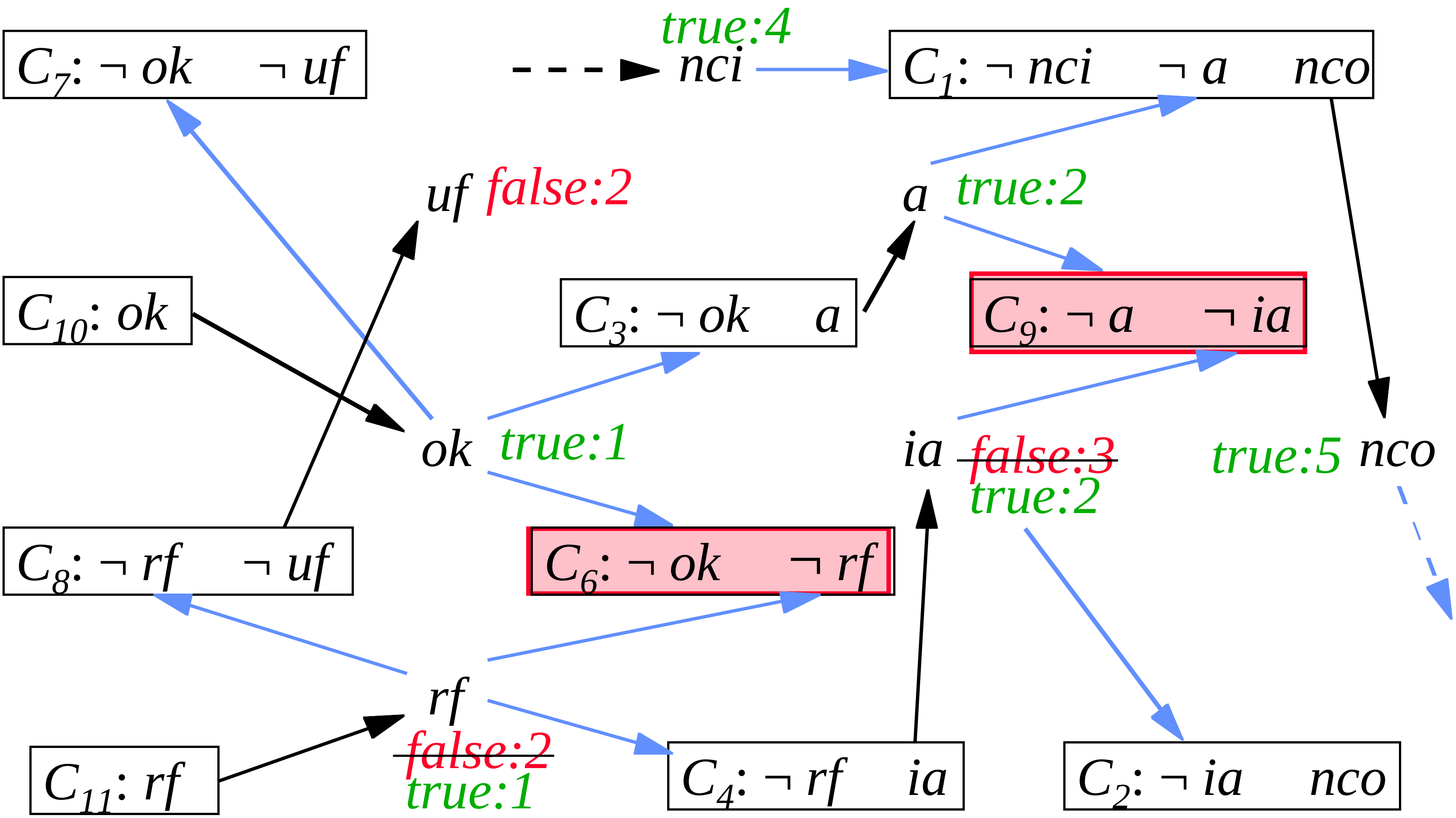
- Assign a *propagation number* to each supported proposition
  - proposition's propagation number is *greater than* propagation number of other propositions occurring in supporting clause



If *p*'s propagation number is greater than *q*'s propagation number, then *q* *cannot* depend on *p*

- Resupport proposition *p* with clause *R* only if *p*'s propagation number is *greater than* propagation number of other literals in *R*
  - sufficient, but not necessary, condition for resupport

Resupporting *nco*



C_7: ¬ *ok* ¬ *uf*

nci true:4

C_1: ¬ *nci* ¬ *a* *nco*

uf false:2

a true:2

C_{10}: *ok*

C_3: ¬ *ok* *a*

C_9: ¬ *a* ¬ *ia*

ok true:1

ia ~~false:3~~ true:2

true:5 *nco*

C_8: ¬ *rf* ¬ *uf*

C_6: ¬ *ok* ¬ *rf*

rf ~~false:2~~ true:1

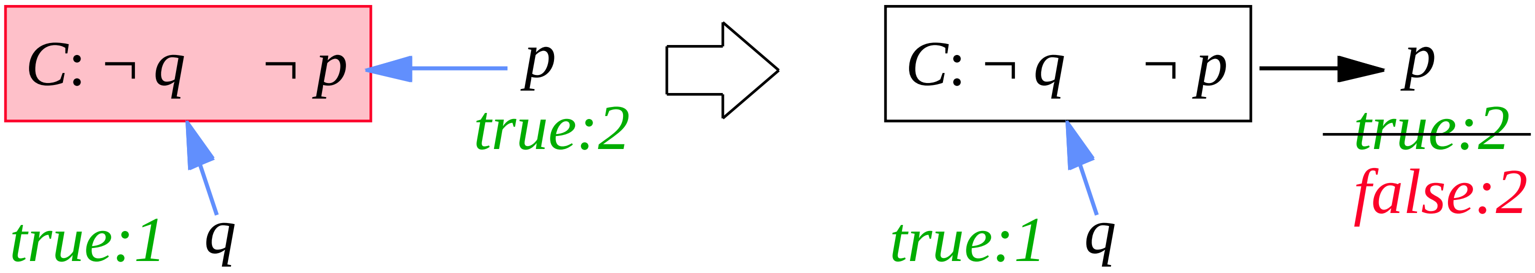
C_{11}: *rf*

C_4: ¬ *rf* *ia*

C_2: ¬ *ia* *nco*

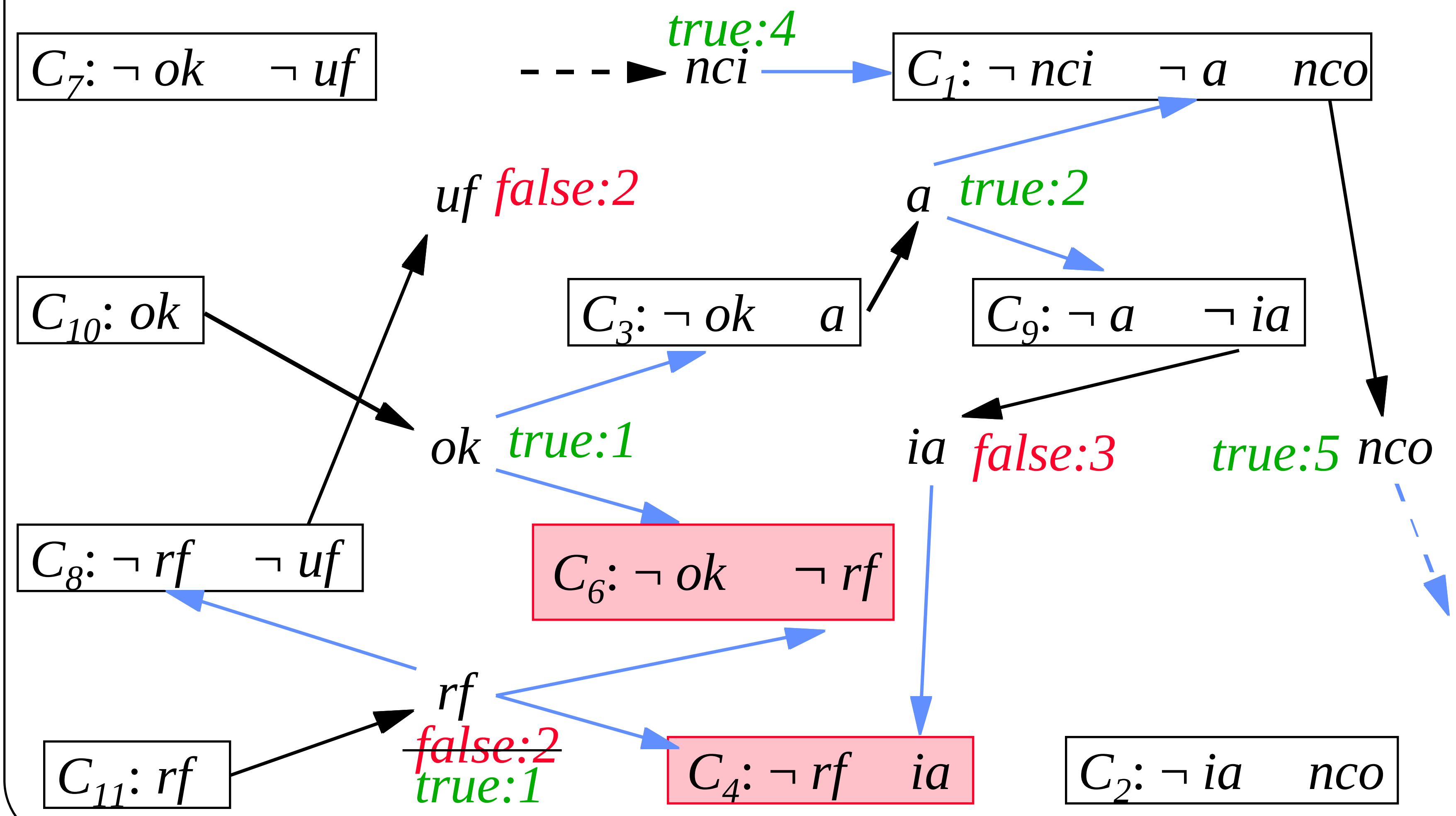
Propagating through a conflict

- Switch the label of a proposition *p* in a conflict *C*
  - and let *p*'s support be *C*



- *C* must provide *p* with a well-founded support
  - *p*'s propagation number must be *greater than or equal to* the propagation number of other literals in *C*
- Resupport other propositions using clauses in which *p* occurs
- Prevent infinite loops by changing a proposition's label at most once

Propagating through C_4



C_7: ¬ *ok* ¬ *uf*

true:4

nci

C_1: ¬ *nci* ¬ *a* *nco*

uf *false:2*

a *true:2*

C_{10}: *ok*

C_3: ¬ *ok* *a*

C_9: ¬ *a* ¬ *ia*

ok *true:1*

ia *false:3*

true:5 *nco*

C_8: ¬ *rf* ¬ *uf*

C_6: ¬ *ok* ¬ *rf*

rf

~~*false:2*~~

true:1

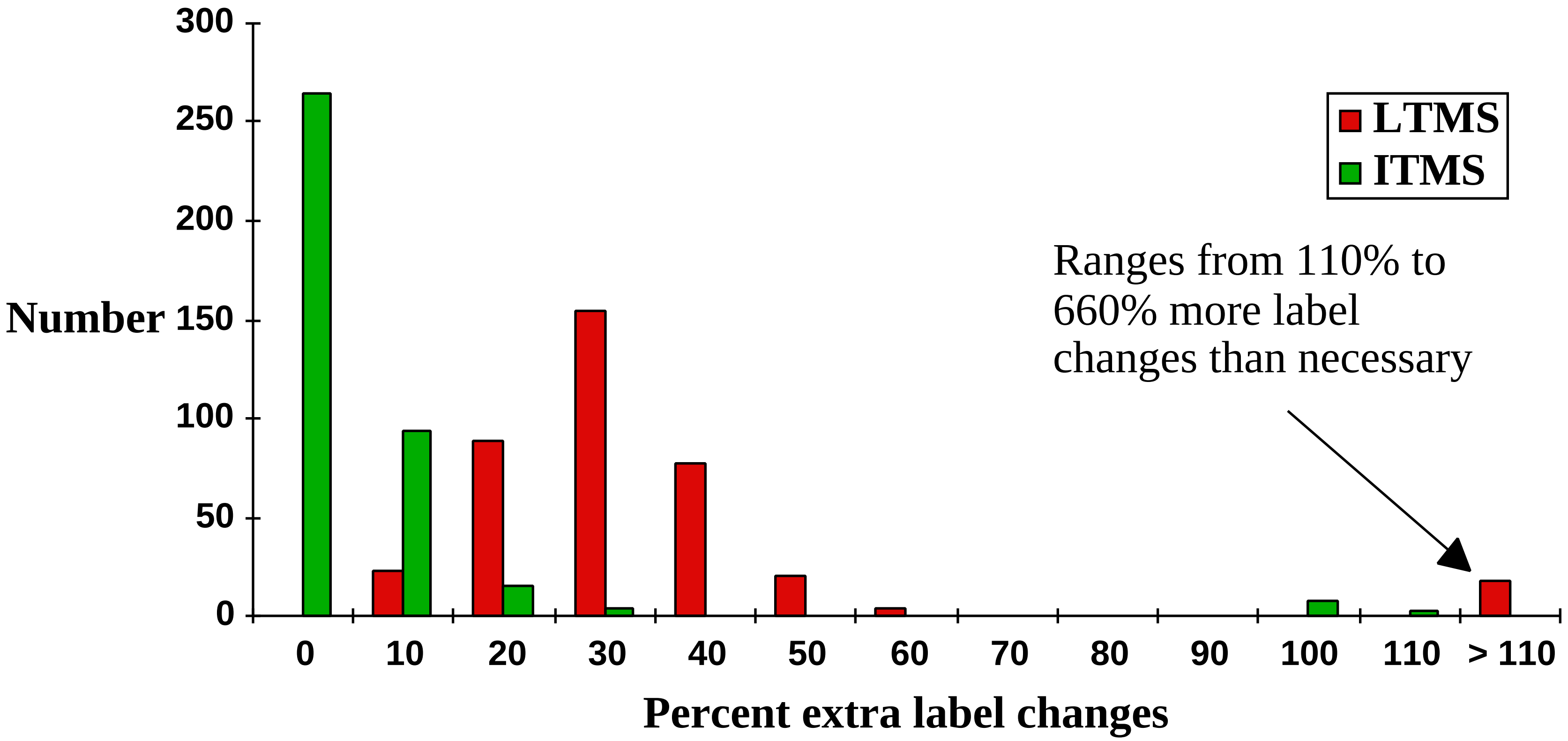
C_{11}: *rf*

C_4: ¬ *rf* *ia*

C_2: ¬ *ia* *nco*

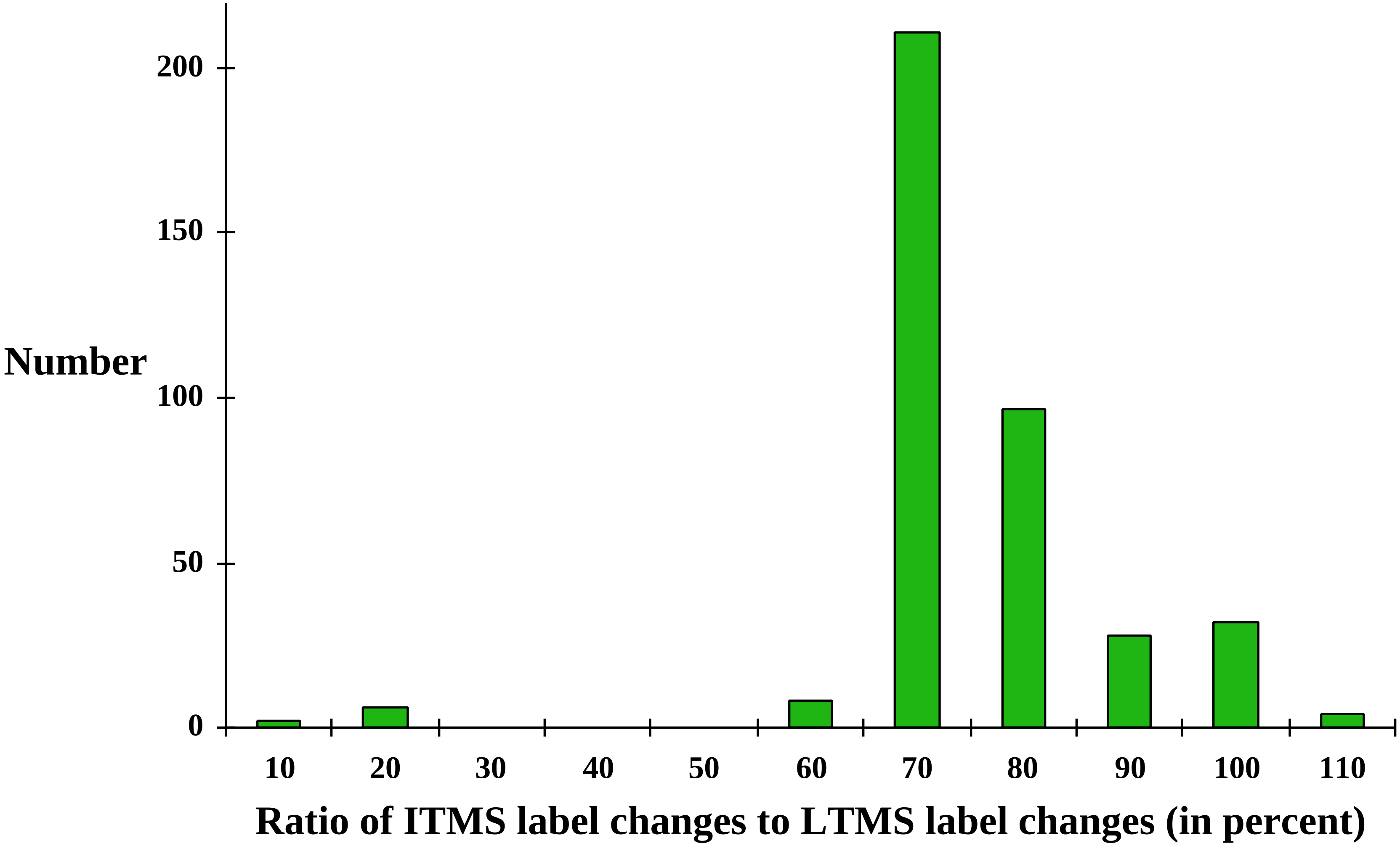
ITMS significantly decreases extra label changes

Data from 387 context switches on DS-1 theory containing 12,693 clauses



Ranges from 110% to 660% more label changes than necessary

Comparing the ITMS to the LTMS



Conclusions

- The ITMS is an aggressive incremental TMS that optimizes context switching
  - clause addition done before clause deletion
  - novel resupport algorithm using propagation numbers
  - novel algorithm to propagate through conflicts
- Dramatic reduction in worst-case performance compared to a traditional LTMS
- Critical for achieving adequate performance in Livingstone’s real-time propositional reasoning execution kernel