Spartacus

A Tableau Prover for Hybrid Logic

Daniel Götzmann

Graduate Seminar

Programming Systems Lab

2009-01-23

Advisor: Mark Kaminski

Responsible Professor: Gert Smolka

What is Spartacus?

- reasoner for hybrid logic
  - + global modalities
  - + reflexivity and transitivity
- pattern-based blocking
  - technique to ensure termination
  - different from traditional (chain-based) blocking
  - see whether it is a useful optimization technique
- optimization techniques

Overview

- basic modal logic
- basic architecture of Spartacus
- extensions to the architecture for
    - + nominals
    - + global modalities
- pattern-based blocking
- optimization techniques
- evaluation

Basic Modal Logic

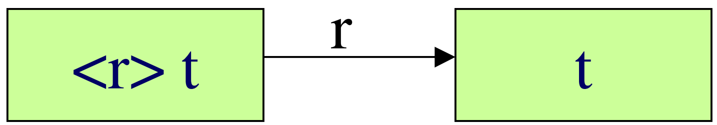
$t ::= p \mid \neg t \mid t \wedge t \mid t \vee t$ propositional logic

$\mid \langle r \rangle\, t \mid [r]\, t$ + modal operators

$t : S \rightarrow B$ predicates on states

$t\ x$ t holds in state x

$\langle r \rangle\, t\ x := \exists\, y.\ r\ x\ y \wedge t\ y$



$[r]\ t\ x := \forall\, y.\ r\ x\ y \Rightarrow t\ y$

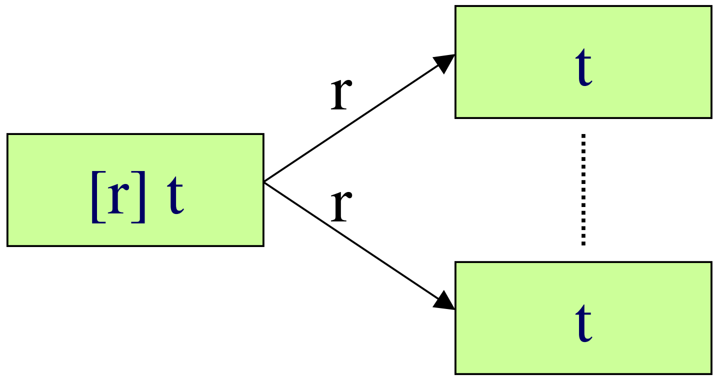


Tableau Algorithm

- decide satisfiability of a term t
- start with *branch* $\Gamma = \{tx\}$ (x not free in t)
- infer new constraints by application of tableau rules:

disjunction	$\dfrac{(s \vee t)\ x}{s\ x \mid t\ x}$	conjunction	$\dfrac{(s \wedge t)\ x}{s\ x,\ t\ x}$
diamond	$\dfrac{\langle r \rangle\ t\ x}{r\ x\ y,\ t\ y}$ y fresh	box	$\dfrac{[r]\ t\ x \qquad r\ x\ y}{t\ y}$

- a branch Γ is *closed* if $\{sx, \neg sx\} \subseteq \Gamma$
- a term is unsatisfiable if all branches are closed

Architecture

Node Store

for each state:
maintain a node that stores
inferred constraints

Backtracking Search

keep track of alternative branches
backtracking after conflict

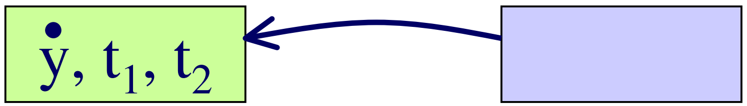
Agenda

store pending rule applications
different ordering heuristics

Nominals

$\dot{y}\ x$ $:= x \doteq y$ state x is named y

- state equivalence implemented as disjoint set forest
  - one node becomes representative, contains all constraints
  - other nodes are replaced by forward pointers to the representative



$@_y\ t\ x$ $:= t\ y$ t holds in the state named y $\dfrac{@_y\ t\ x}{t\ y}$

- implementation: add t to the representative of y

Global Modalities

E t x $:= \exists$ y.t y t holds in some state

$$\frac{E\ t\ x}{t\ y}\ \text{y fresh}$$

- implementation: create a new node and add t to it
- remember that a node for t has been created

A t x $:= \forall$ y.t y t holds in all states

$$\frac{A\ t\ x}{t\ y}\ \text{y on branch}$$

- implementation: add t to all nodes
- requires blocking for termination

Termination

A (<r> p)

$$\frac{\langle r \rangle\ t\ x}{r\ x\ y,\ t\ y}\ \text{y fresh}$$

$$\frac{A\ t\ x}{t\ y}\ \text{y on branch}$$

$$\frac{[r]\ t\ x \quad r\ x\ y}{t\ y}$$

Termination

A (<r> p) → universal modalities:

$$\frac{\langle r \rangle\ t\ x}{r\ x\ y,\ t\ y}\ \text{y fresh}$$

$$\frac{A\ t\ x}{t\ y}\ \text{y on branch}$$

$$\frac{[r]\ t\ x \quad r\ x\ y}{t\ y}$$

Termination

universal modalities:

<r> p

<r> t x / r x y, t y	y fresh
A t x / t y	y on branch
[r] t x r x y / t y	

Termination

<r> p

universal modalities:

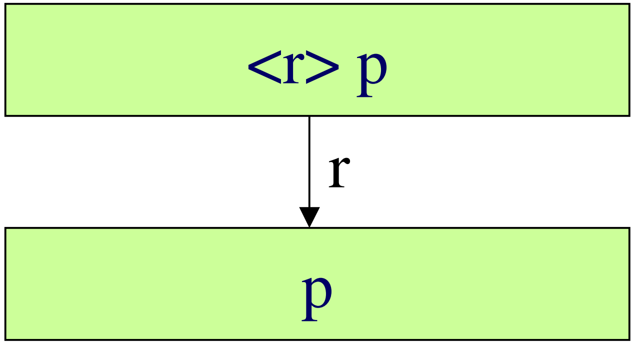
<r> p

$$\frac{\langle r\rangle\ t\ x}{r\ x\ y,\ t\ y}\ \text{y fresh}$$

$$\frac{A\ t\ x}{t\ y}\ \text{y on branch}$$

$$\frac{[r]\ t\ x \quad r\ x\ y}{t\ y}$$

Termination



universal modalities:

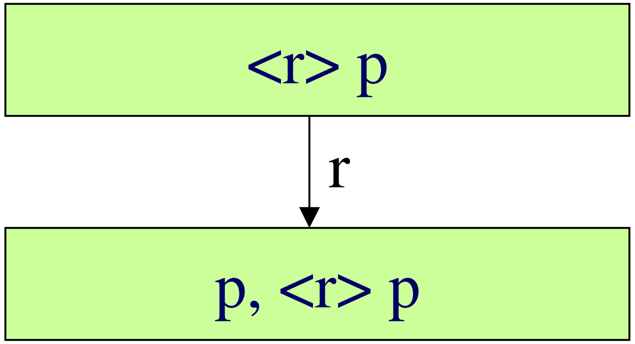
<r> p

$$\frac{\langle r \rangle\ t\ x}{r\ x\ y,\ t\ y}\ \text{y fresh}$$

$$\frac{A\ t\ x}{t\ y}\ \text{y on branch}$$

$$\frac{[r]\ t\ x \quad r\ x\ y}{t\ y}$$

Termination



universal modalities:

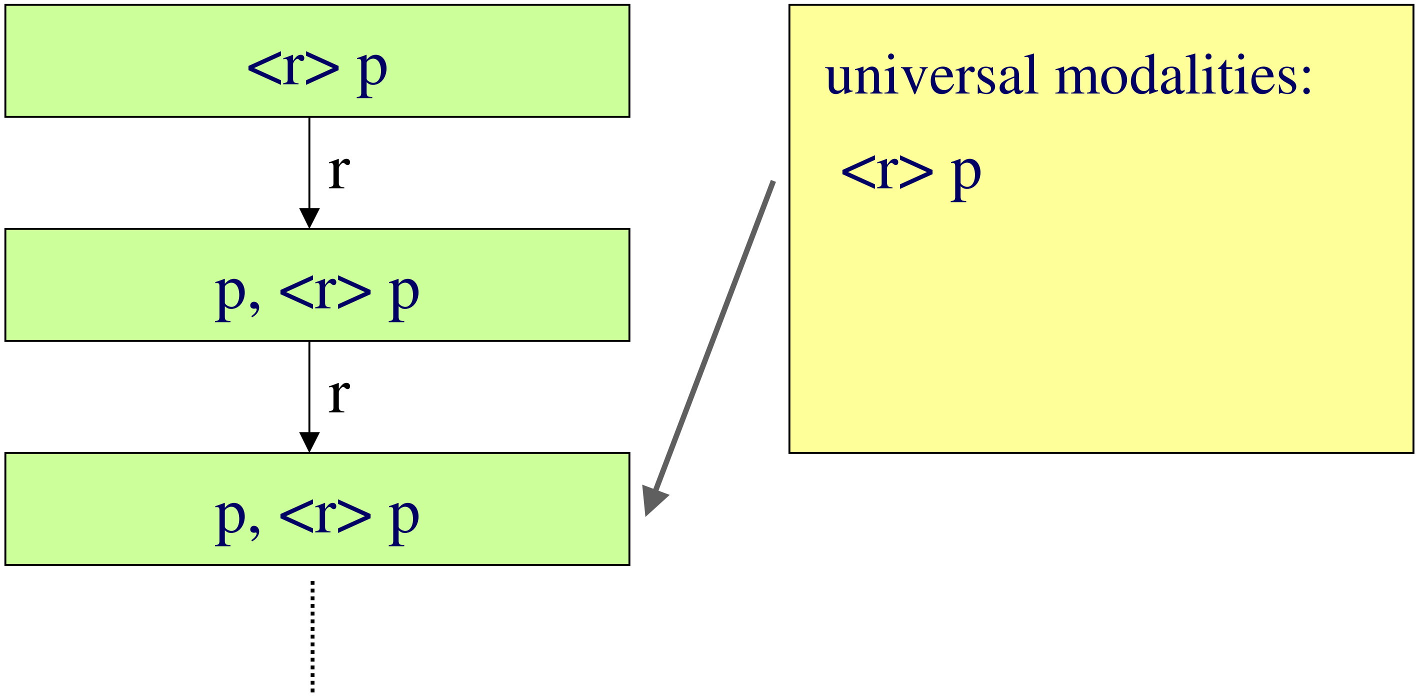
<r> p

$$\frac{\langle r\rangle\ t\ x}{r\ x\ y,\ t\ y}$$ y fresh

$$\frac{A\ t\ x}{t\ y}$$ y on branch

$$\frac{[r]\ t\ x \quad r\ x\ y}{t\ y}$$

Termination



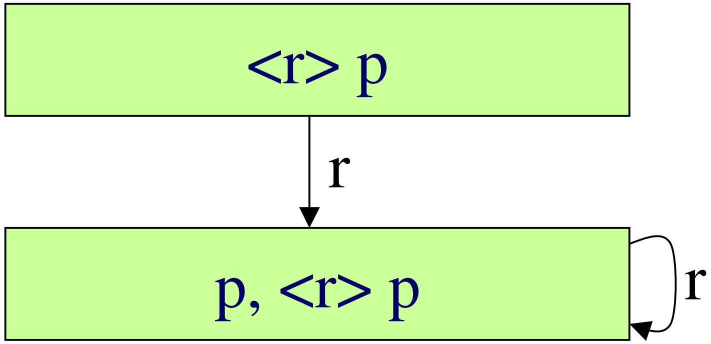
$$\frac{\langle r \rangle\ t\ x}{r\ x\ y,\ t\ y}\ \text{y fresh}$$

$$\frac{A\ t\ x}{t\ y}\ \text{y on branch}$$

$$\frac{[r]\ t\ x \quad r\ x\ y}{t\ y}$$

naïve approach:
not terminating!

Termination



self loop:
terminating!

universal modalities:

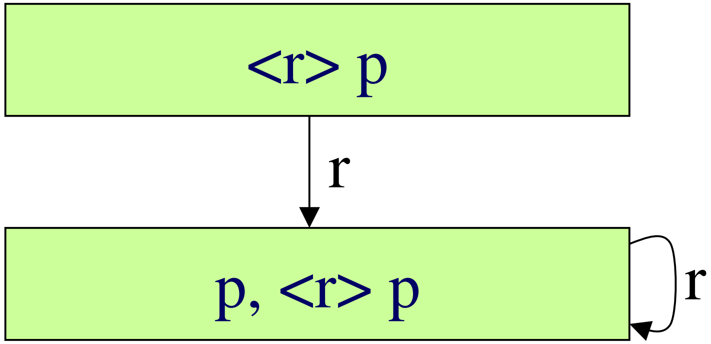
<r> p

$$\frac{\langle r\rangle\ t\ x}{r\ x\ y,\ t\ y}\ \text{y fresh}$$

$$\frac{A\ t\ x}{t\ y}\ \text{y on branch}$$

$$\frac{[r]\ t\ x \quad r\ x\ y}{t\ y}$$

Termination



self loop:
terminating!

general idea:
satisfy diamonds by
adding “safe” edges to
existing nodes

universal modalities:

<r> p

$$\frac{\langle r\rangle\ t\ x}{r\ x\ y,\ t\ y}\ \text{y fresh}$$

$$\frac{A\ t\ x}{t\ y}\quad \text{y on branch}$$

$$\frac{[r]\ t\ x \quad r\ x\ y}{t\ y}$$

Pattern-Based Blocking

$\langle r\rangle\ p,\ [r]\ q_1,\ [r]\ q_2$

universal modalities:

$\langle r\rangle\ p$

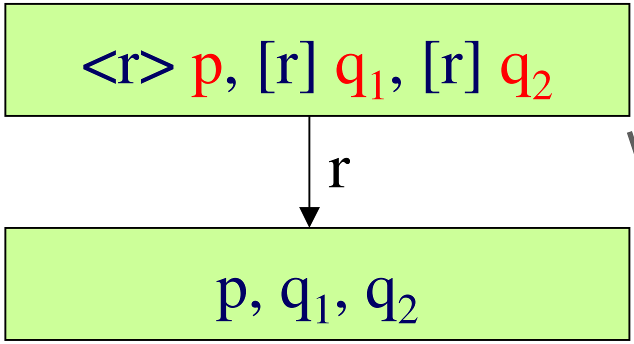
blocked patterns:

$$\frac{\langle r\rangle\ t\ x}{r\ x\ y,\ t\ y}\ \text{y fresh}$$

$$\frac{A\ t\ x}{t\ y}\ \text{y on branch}$$

$$\frac{[r]\ t\ x \quad r\ x\ y}{t\ y}$$

Pattern-Based Blocking



universal modalities:

$\langle r\rangle\, p$

blocked patterns:

$\{p, q_1, q_2\}$

$$\frac{\langle r\rangle\, t\ x}{r\ x\ y,\ t\ y}\ \text{y fresh}$$

$$\frac{A\ t\ x}{t\ y}\quad \text{y on branch}$$

$$\frac{[r]\ t\ x \quad r\ x\ y}{t\ y}$$

Pattern-Based Blocking

$\langle r \rangle p, [r] q_1, [r] q_2$

$\downarrow$ r

$p, q_1, q_2, \langle r \rangle p$

universal modalities:

$\langle r \rangle p$

blocked patterns:

$\{p, q_1, q_2\}$

$$\frac{\langle r \rangle\, t\; x}{r\; x\; y,\; t\; y}\quad y \text{ fresh}$$

$$\frac{A\; t\; x}{t\; y}\quad y \text{ on branch}$$

$$\frac{[r]\, t\; x \qquad r\; x\; y}{t\; y}$$

Pattern-Based Blocking

<r> p, [r] q_1, [r] q_2

↓ r

p, q_1, q_2, <r> p

<r> p is blocked, hence not expanded

universal modalities:

<r> p

blocked patterns:

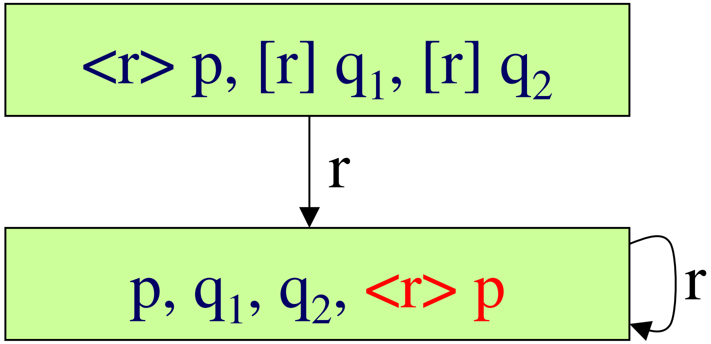
{p, q_1, q_2}

$$\frac{\langle r\rangle\, t\ x}{r\ x\ y,\ t\ y}\ \text{y fresh}$$

$$\frac{A\ t\ x}{t\ y}\ \text{y on branch}$$

$$\frac{[r]\, t\ x \quad r\ x\ y}{t\ y}$$

Pattern-Based Blocking



<r> p is blocked,
hence not expanded

universal modalities:

<r> p

blocked patterns:

{p, q_1, q_2}

$$\frac{\langle r\rangle\ t\ x}{r\ x\ y,\ t\ y}\quad y \text{ fresh}$$

$$\frac{A\ t\ x}{t\ y}\quad y \text{ on branch}$$

$$\frac{[r]\ t\ x \quad r\ x\ y}{t\ y}$$

Pattern-Based Blocking

- necessary to guarantee termination
- useful optimization
  - simple variant: store pattern only when node is created
  - eager variant: update stored pattern when new box is propagated

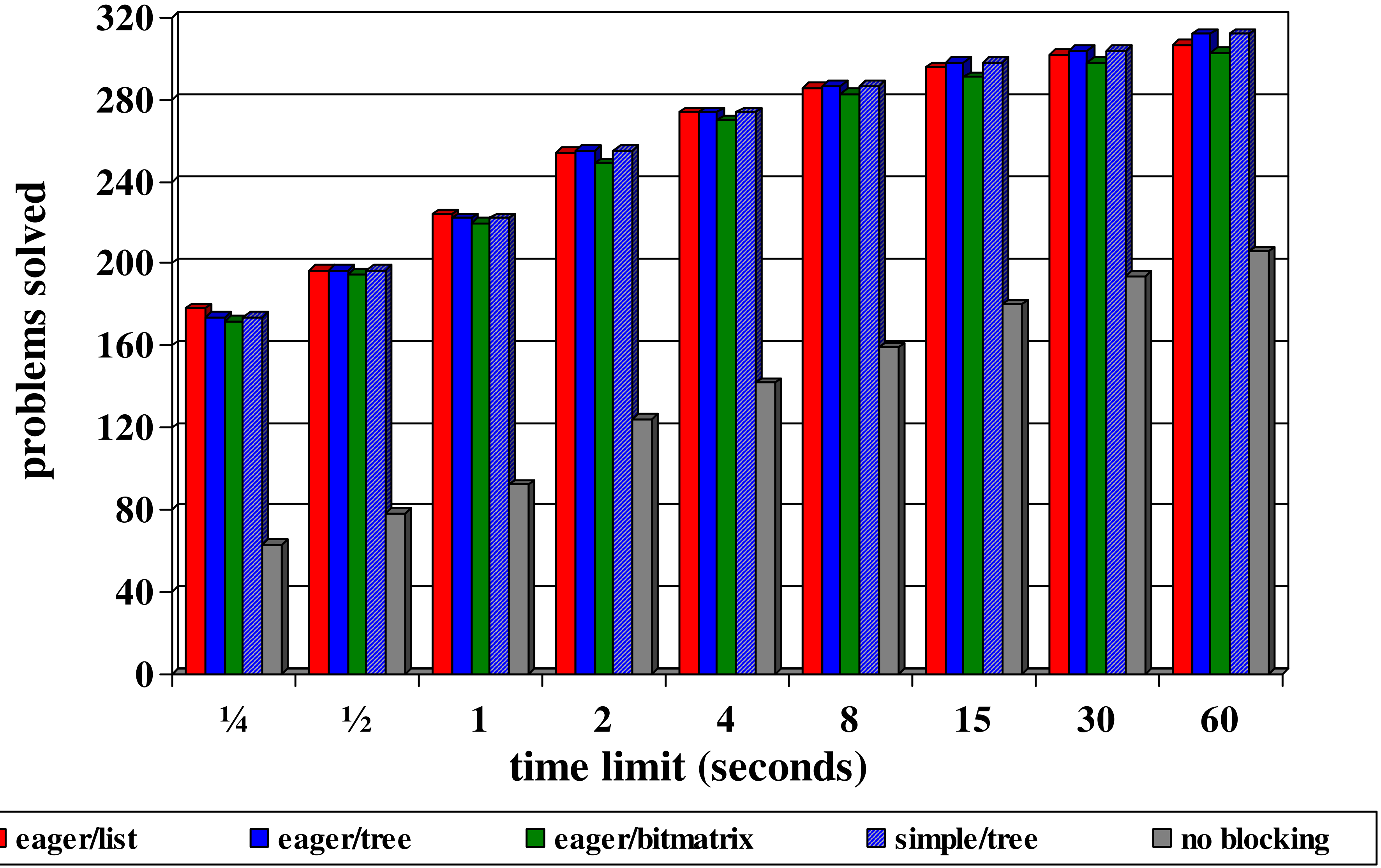
- needed: efficient data structure to store and query for patterns
- three data structures
  - tree-based (Hoffmann, Koehler, 1999)
  - bitvector-based (Giunchiglia, Tacchella, 2000)
  - based on array of lists (based on bitvector approach, compact representation)

Evaluation

- subset of TANCS-2000 benchmarks
  - randomly generated quantified boolean formulas translated into terms of basic modal logic
- "modalized" benchmarks
  - created similarly to TANCS-2000 benchmarks
  - each propositional variable is replaced by a modal term containing only one propositional variable
- terms generated randomly by K-CNF-generator
  - http://www.mrg.dist.unige.it/~tac/StarSAT/Sources4610712832/K-CNF-generator.tar.gz

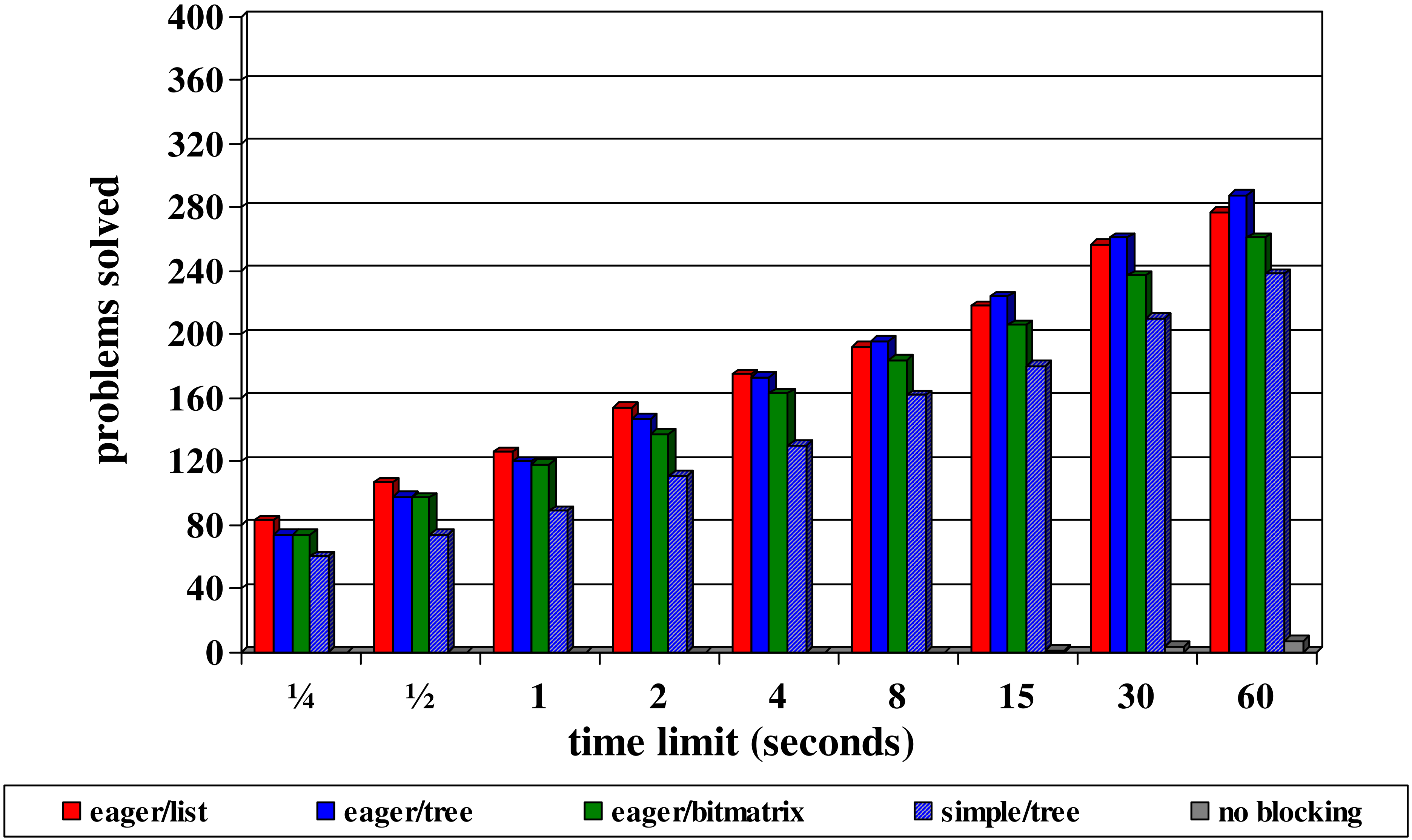
Pattern-Based Blocking

TANCS-2000 benchmarks



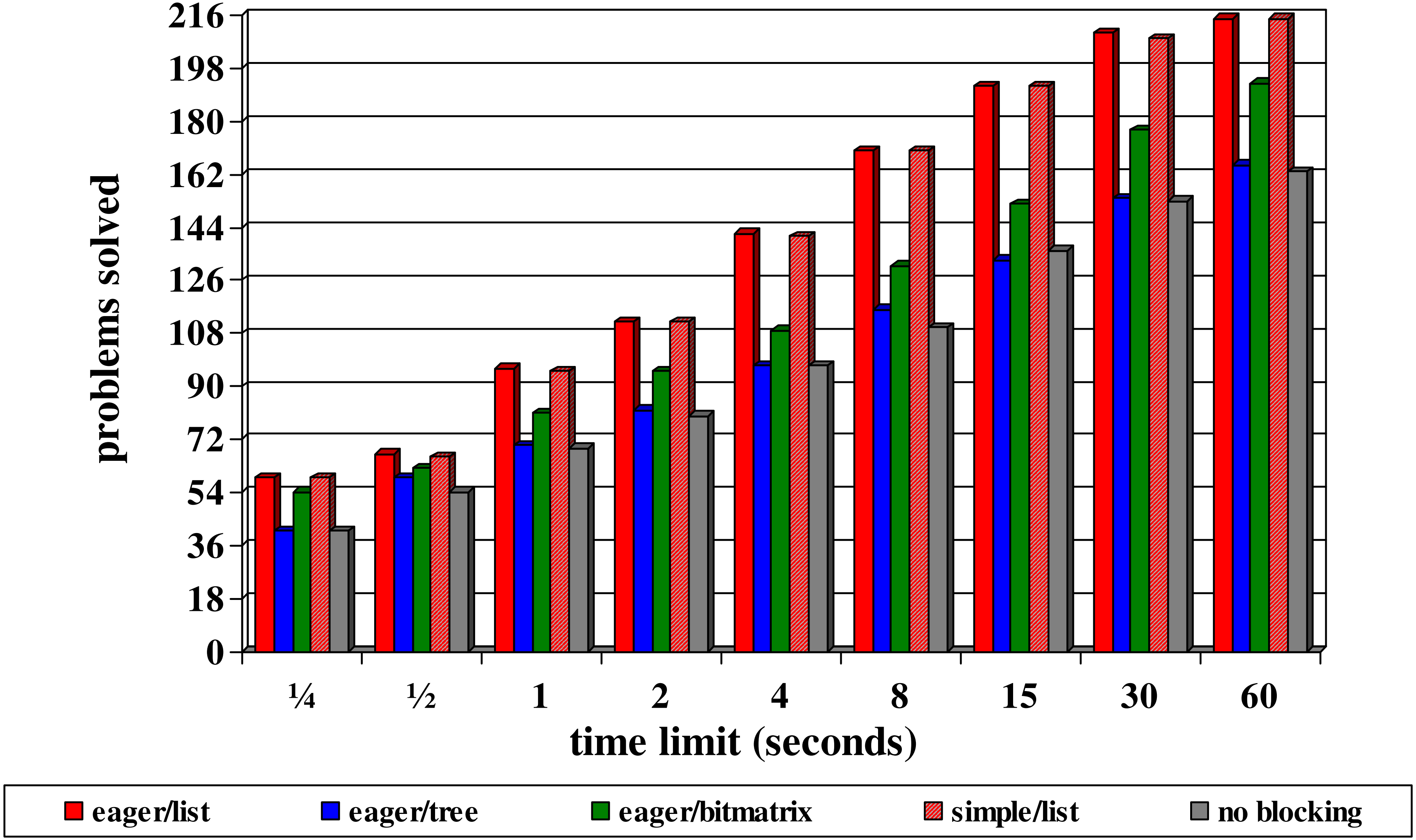
Pattern-Based Blocking

"modalized" benchmarks



Pattern-Based Blocking

terms generated by K-CNF-generator



Optimizations

- Term Normalization
- Backjumping
- Boolean Constraint Propagation
- Disjoint Branching
- Lazy Branching
- Caching of Unsatisfiable Sets of Terms

Lazy Branching

- extension of lazy unfolding (Horrocks, 1997)
- related to “pure literal elimination” in DPLL (Davis et. al., 1962)

- delay branching on disjunctions containing propositional literals

- example:

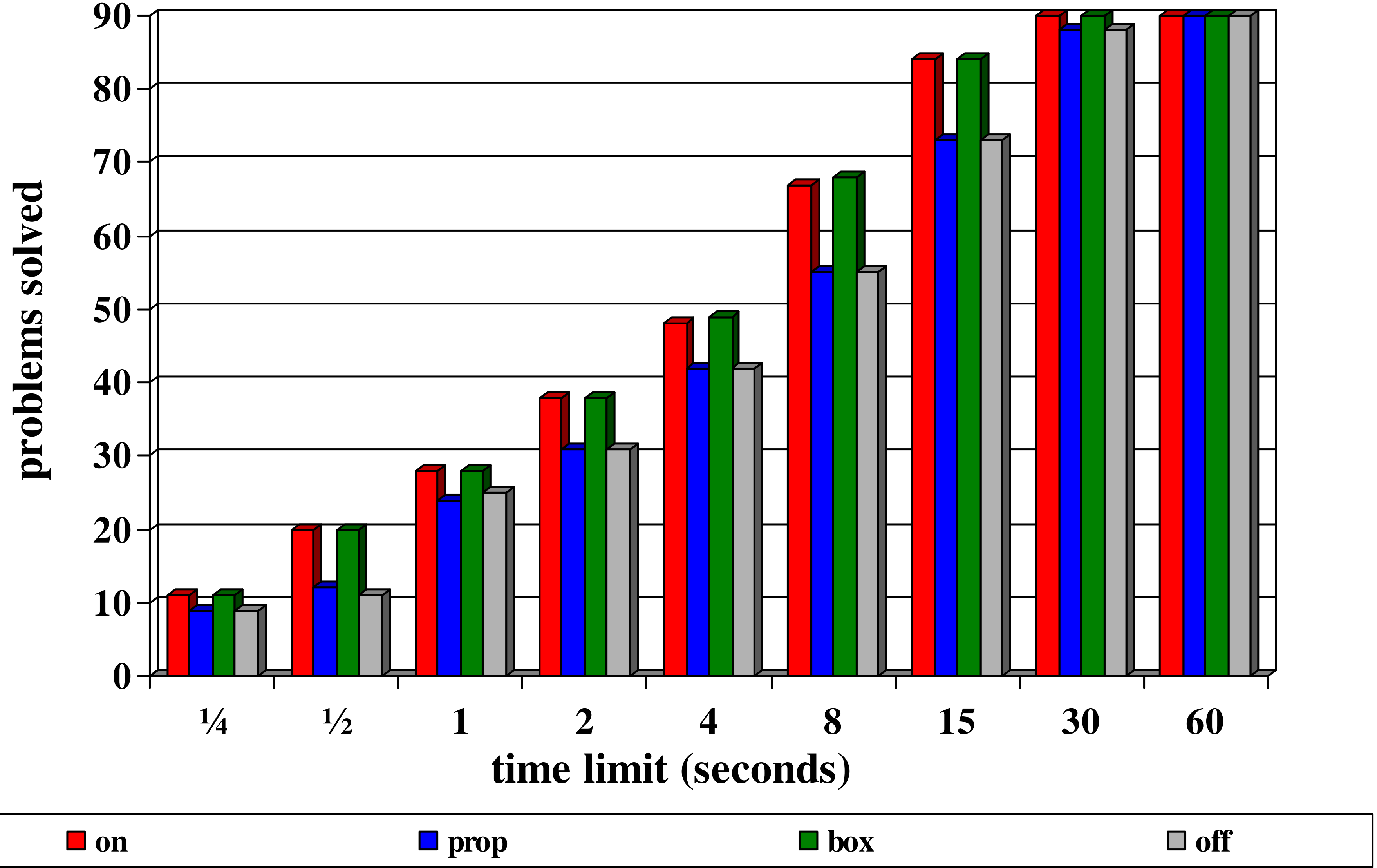
  > $(p \vee <r>q)x$ can be delayed if
  > - $\neg px$ is not on the branch, and
  > - $(\neg p \vee t)x$ is not being delayed

- only add disjunctions that cannot be delayed to the agenda

- lazy branching on boxes:
- only add disjunction $([r]s \vee ...)x$ to agenda when $<r>tx$ on the branch

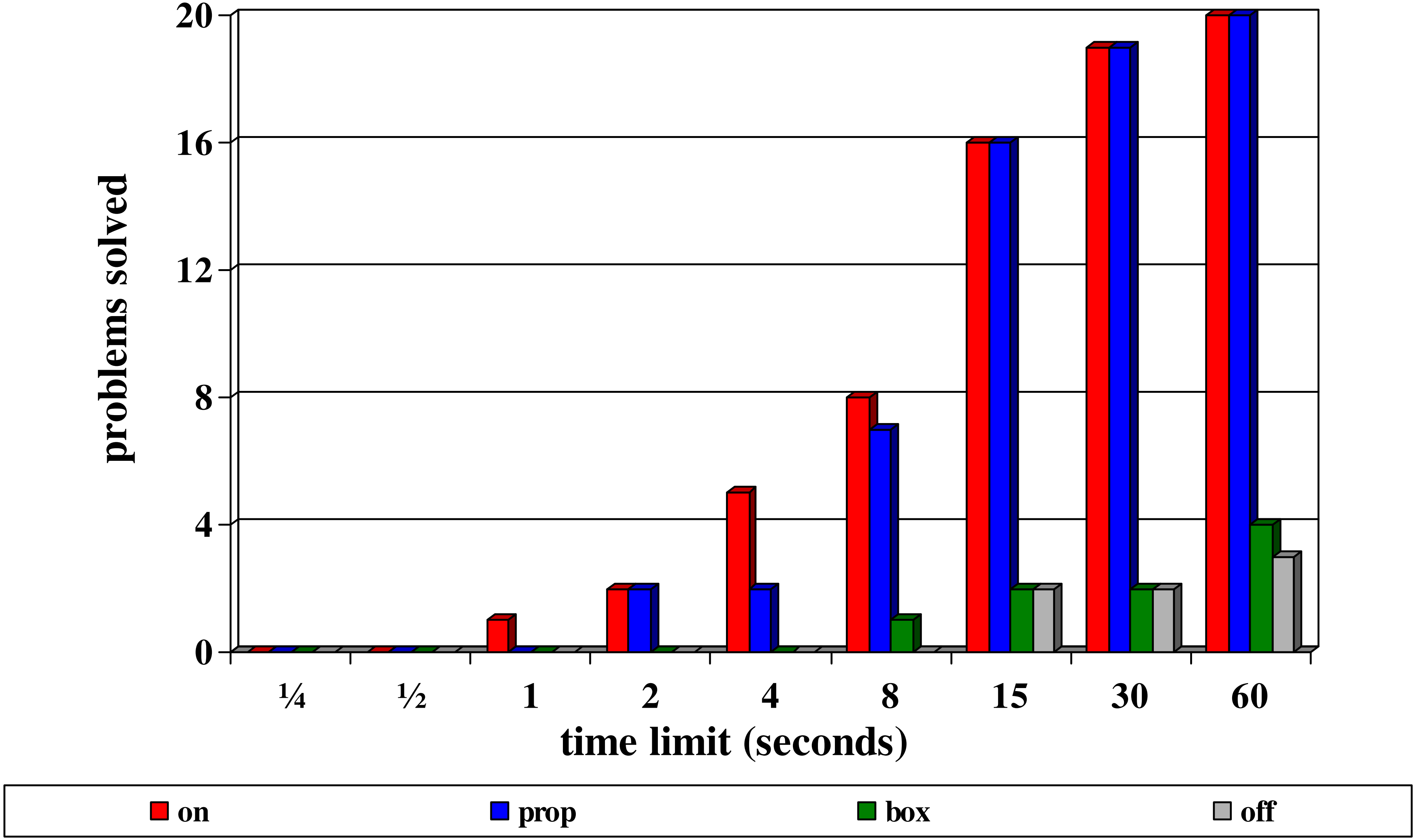
Lazy Branching

terms with large modal depth (K-CNF)



Lazy Branching

terms with hard propositional subproblems (K-CNF)

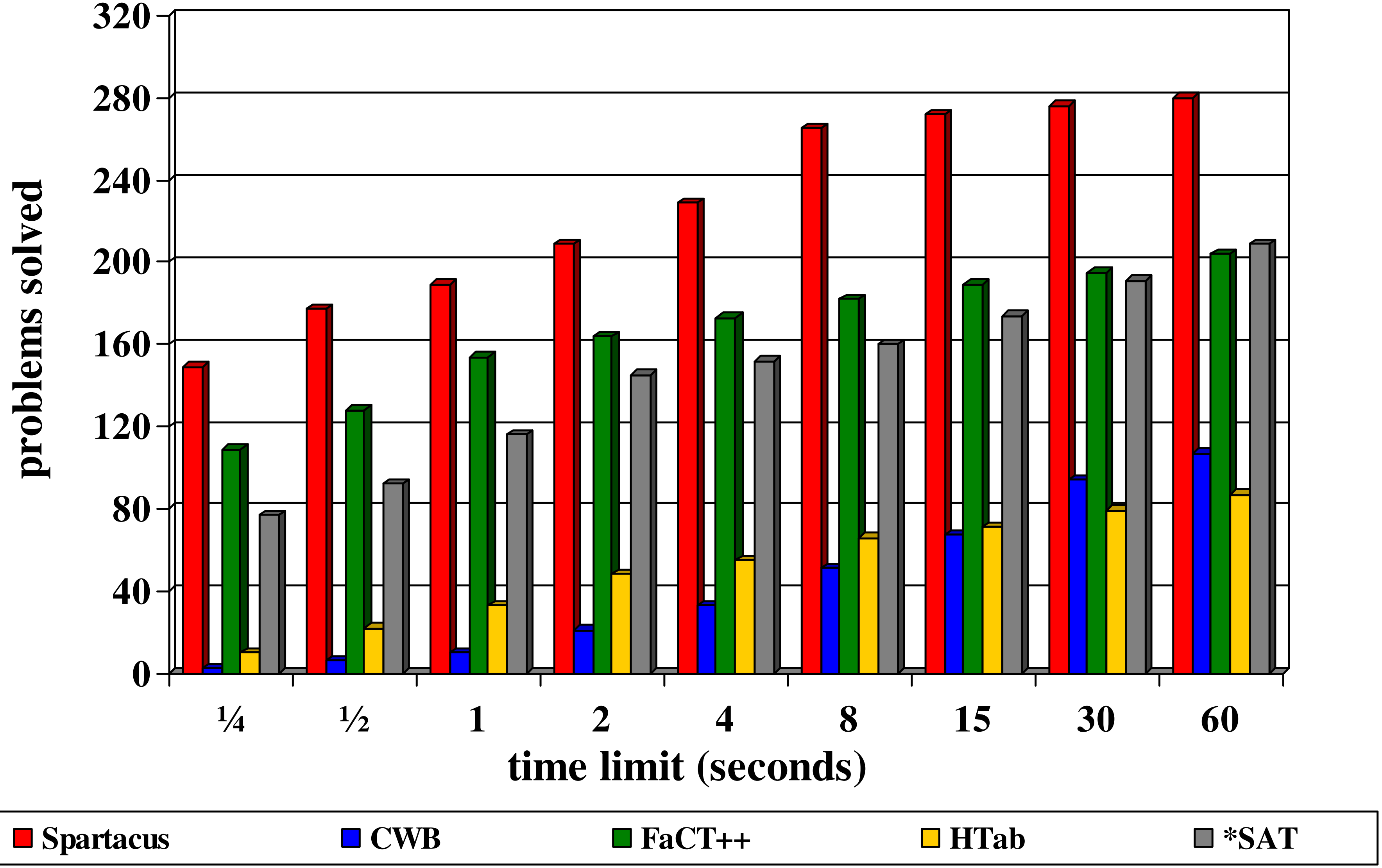


Comparison

- CWB
  - prototype reasoner for basic modal logic
  - http://users.rsise.anu.edu.au/~linda/CWB.html
- FaCT++
  - reasoner for the rich description logic SROIQ(D)
  - http://code.google.com/p/factplusplus/
- HTab
  - reasoner for hybrid logic
  - http://www.glyc.dc.uba.ar/intohylo/htab.php
- *SAT
  - reasoner for basic modal logic
  - http://www.mrg.dist.unige.it/~tac/StarSAT.html

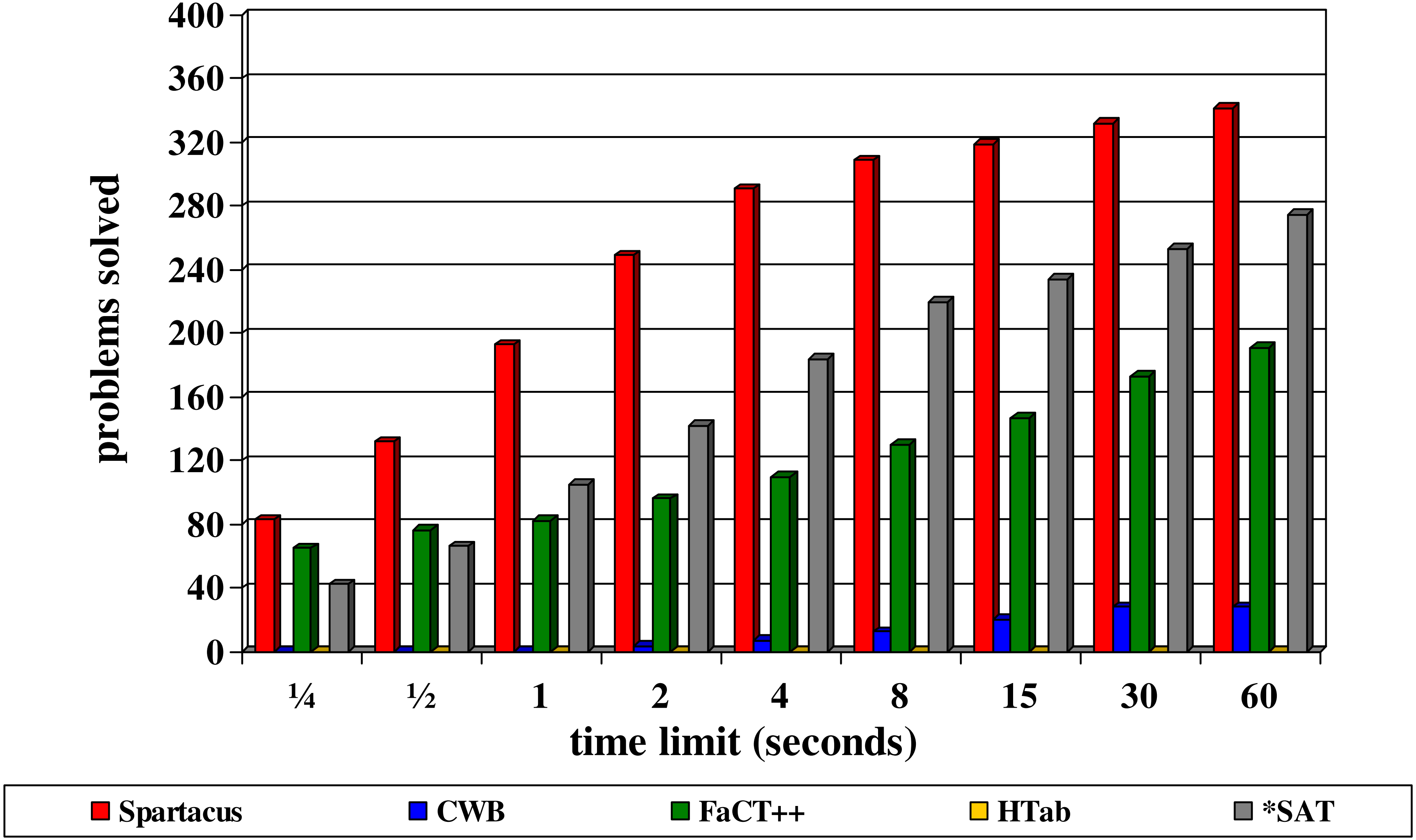
Comparison

TANCS-2000 benchmarks



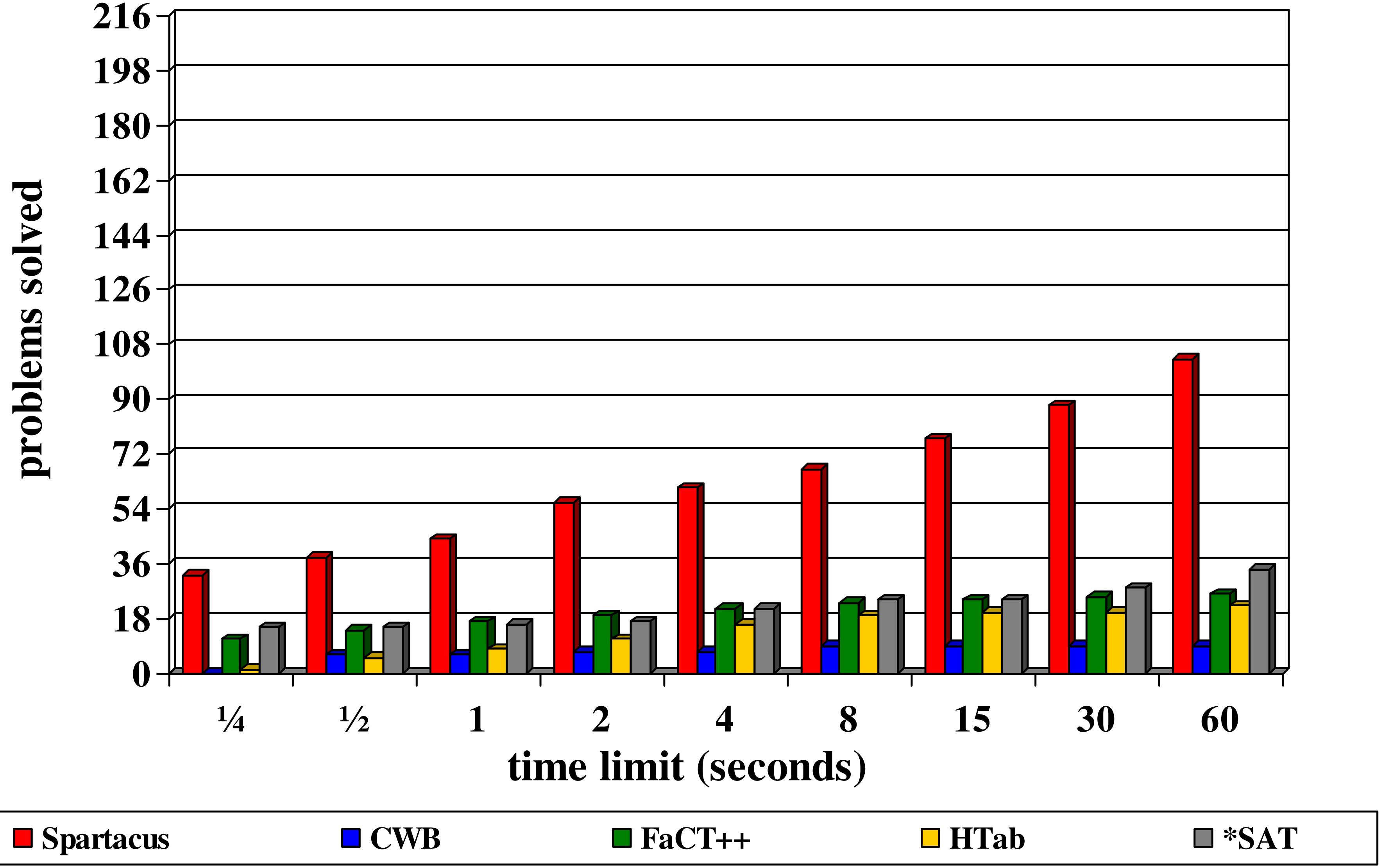
Comparison

“modalized” benchmarks



Comparison

terms generated by K-CNF-generator



Contributions

- Spartacus is a competitive reasoner for hybrid logic
- the first implementation of pattern-based blocking
- evaluation of data structures for storing patterns
- novel optimization technique: lazy branching

not presented here but in the thesis:

- evaluation of
  - optimization techniques
  - rule application strategies

Conclusion

- modal reasoning successful but
  - still room for improvement
  - optimizations important
- pattern-based blocking
  - technique to achieve termination (global modalities, transitivity)
  - promising optimization technique (as shown by evaluation)

Outlook

- improve rule application heuristics
- more features from description logic
  - role hierarchies
  - graded modalities
- PDL, μ-calculus?
- caching (learning from failure)?
- converse modalities?

References

- hybrid logic
  - C. Areces and B. ten Cate. Hybrid Logics. In P. Blackburn, J. van Benthem, and F. Wolter, editors, Handbook of Modal Logic. Elsevier, 2007.

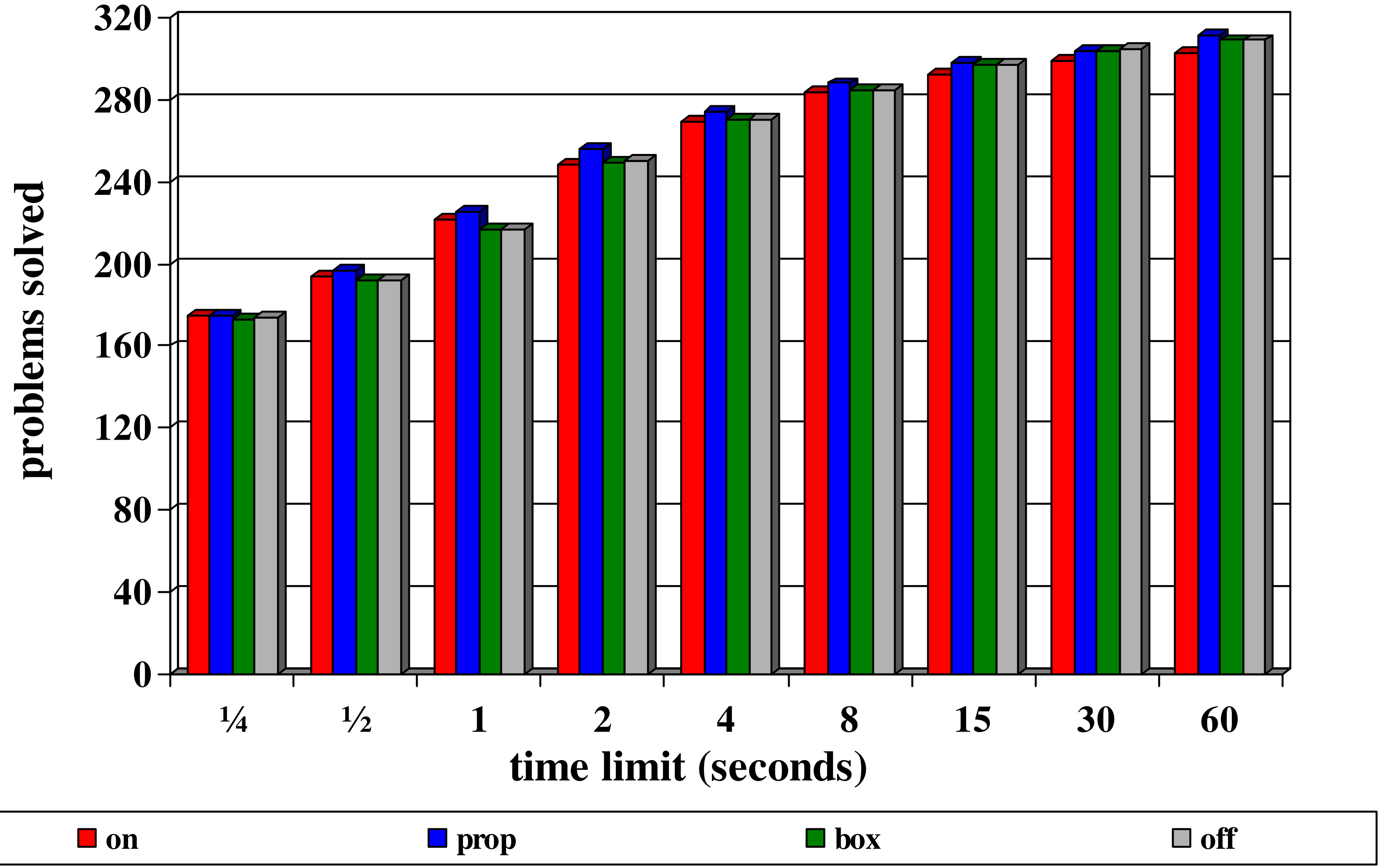
- pattern-based blocking
  - M. Kaminski and G. Smolka. Hybrid Tableaux for the Difference Modality. In Proc. 5th Workshop on Methods for Modalities (M4M-5), pp. 269-284, Cachan, France, November 2007. To Appear in ENTCS
  - M. Kaminski and G. Smolka. Terminating tableau systems for hybrid logic with difference and converse. Technical report, Saarland University, 2008.

- data structures for pattern-based blocking
  - J. Hoffmann and J. Koehler. A New Method to Index and Query Sets. In Proceedings of the 16th International Joint Conference on Artificial Intelligence, pp. 462-467, 1999.
  - E. Giunchiglia and O. Tacchella. A subset-matching size-bounded cache for satisfiability of modal logics. In Proceedings International Conference Tableaux'2000, pp. 237-251. 2000.

- optimizations
  - Marcello D'Agostino. Are tableaux an improvement on truth-tables? Cut-free proofs and bivalence. Journal of Logic, Language, and Information, 1(3):235–252, 1992.
  - M. Davis, G. Logemann, and D. Loveland. A machine program for theorem-proving. Communications of the ACM, 5:394–397, 1962.
  - J. W. Freeman. Improvements to Propositional Satisfiability Search Algorithms. PhD thesis, Departement of computer and Information science, University of Pennsylvania, Philadelphia, 1995.
  - I. Horrocks. Optimising Tableaux Decision Procedures for Description Logics. PhD thesis, University of Manchester, 1997.
  - I. Horrocks, U. Hustadt, U. Sattler, and R. Schmidt. Computational modal logic. In P. Blackburn, J. van Benthem, and F. Wolter, editors, Handbook of Modal Logic, chapter 4, pages 181–245. Elsevier, 2006.
  - D. Tsarkov, I. Horrocks, and P. F. Patel-Schneider. Optimizing terminological reasoning for expressive description logics. J. of Automated Reasoning, 39 (3):277–316, 2007.

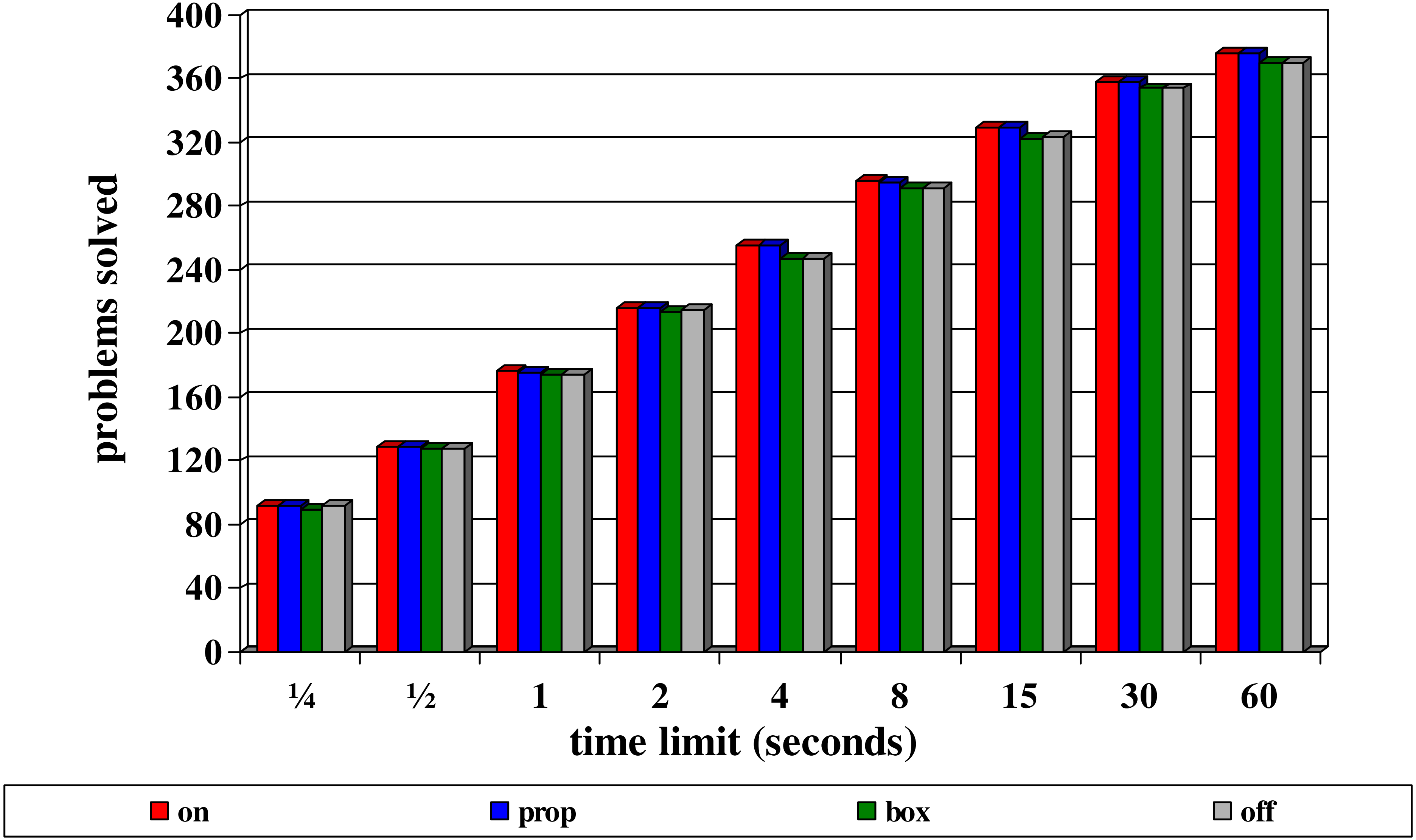
Lazy Branching

TANCS-2000 benchmarks



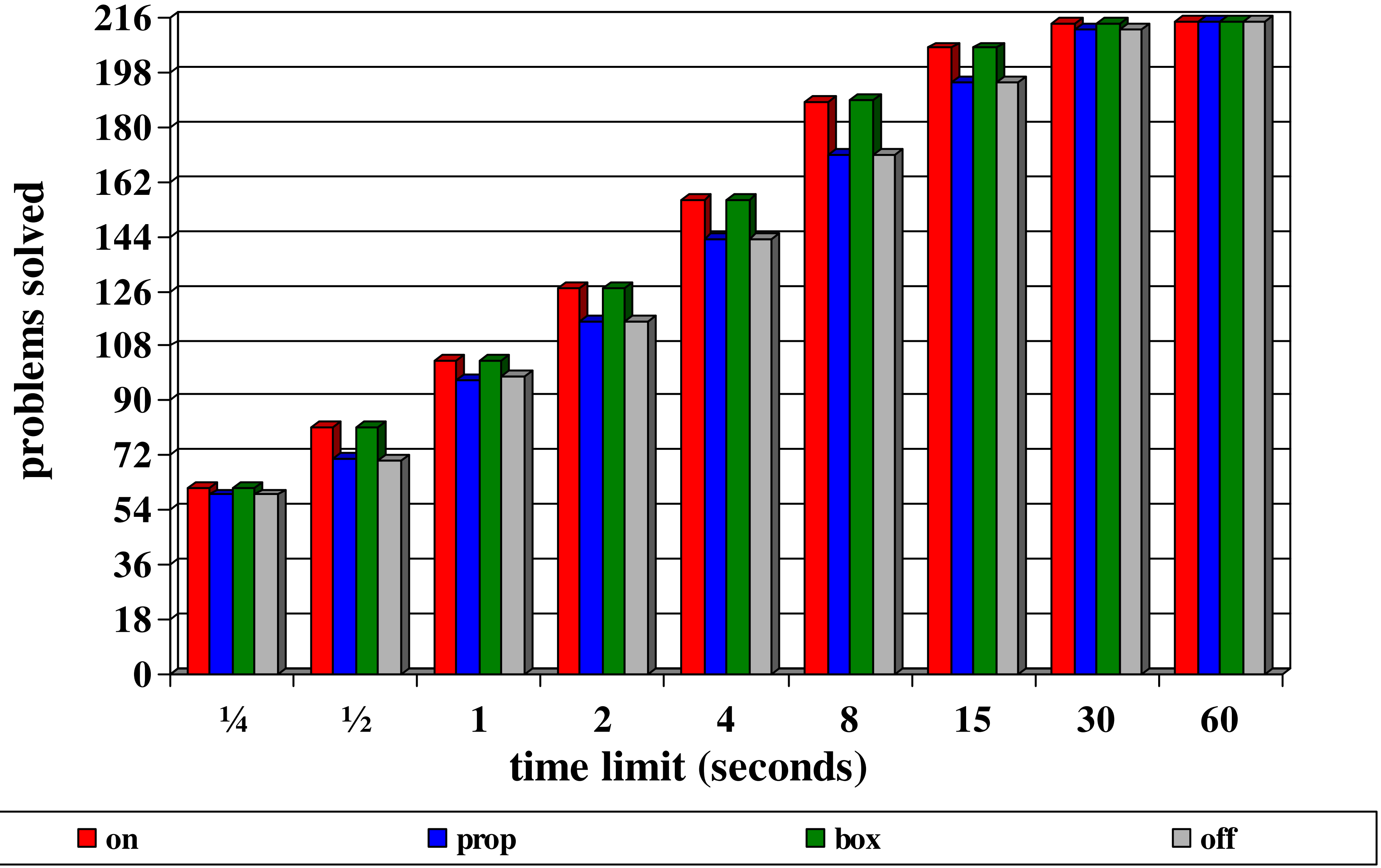
Lazy Branching

"modalized" benchmarks



Lazy Branching

terms generated by K-CNF-generator



Reflexivity and Transitivity

Reflexivity

$$\frac{[r]\ t\ x}{t\ x}\ \text{r reflexive}$$

Transitivity

$$\frac{[r]\ t\ x \qquad r\ x\ y}{[r]\ t\ y}\ \text{r transitive}$$

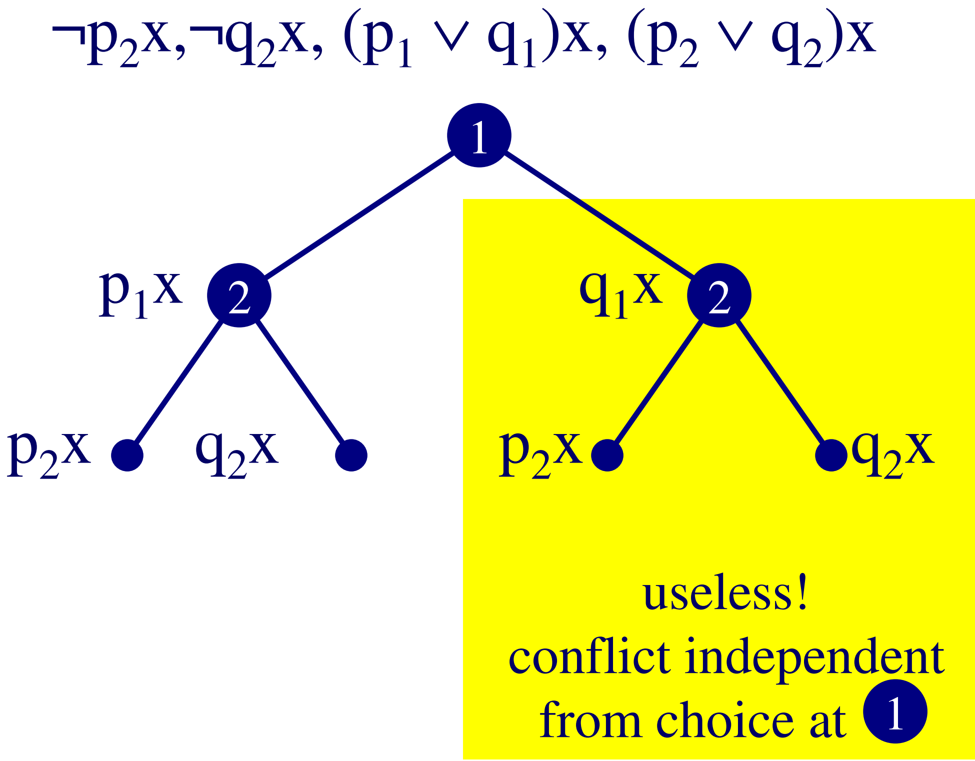
- requires blocking for termination

Term Normalization

- each subterm of input represented by index (integer)
- obviously equivalent terms have same index
  - e.g., $(p \wedge q)$ and $(q \wedge p)$
- indices of t and $\neg$t differ only in least significant bit
- obviously unsatisfiable terms mapped to 0
  - e.g, $(p \wedge \neg p)$
- obviously valid terms mapped to 1
  - e.g, $(p \vee \neg p)$

see also: Horrocks (1997)

Backjumping



- idea: for each term t: store branching points, on which t depends
- jump back to closest participating branching point

see also: Horrocks (1997)

Boolean Constraint Propagation

- before branching: eliminate disjuncts that obviously lead to conflict

- can eliminate all but one disjunct: add it deterministically
- can eliminate all disjuncts: backtrack immediately

- simple version: look at first disjunction on agenda only

- eager version: find disjunction on agenda that can be simplified
  - currently: search the entire agenda
  - better: watched literals

see also: Freeman (1995)

Disjoint Branching

- replace branching rule $\dfrac{(s \vee t)\ x}{sx \mid tx}$ by $\dfrac{(s \vee t)\ x}{sx \mid \neg sx,\ tx}$

- enfores semantically disjoint branches

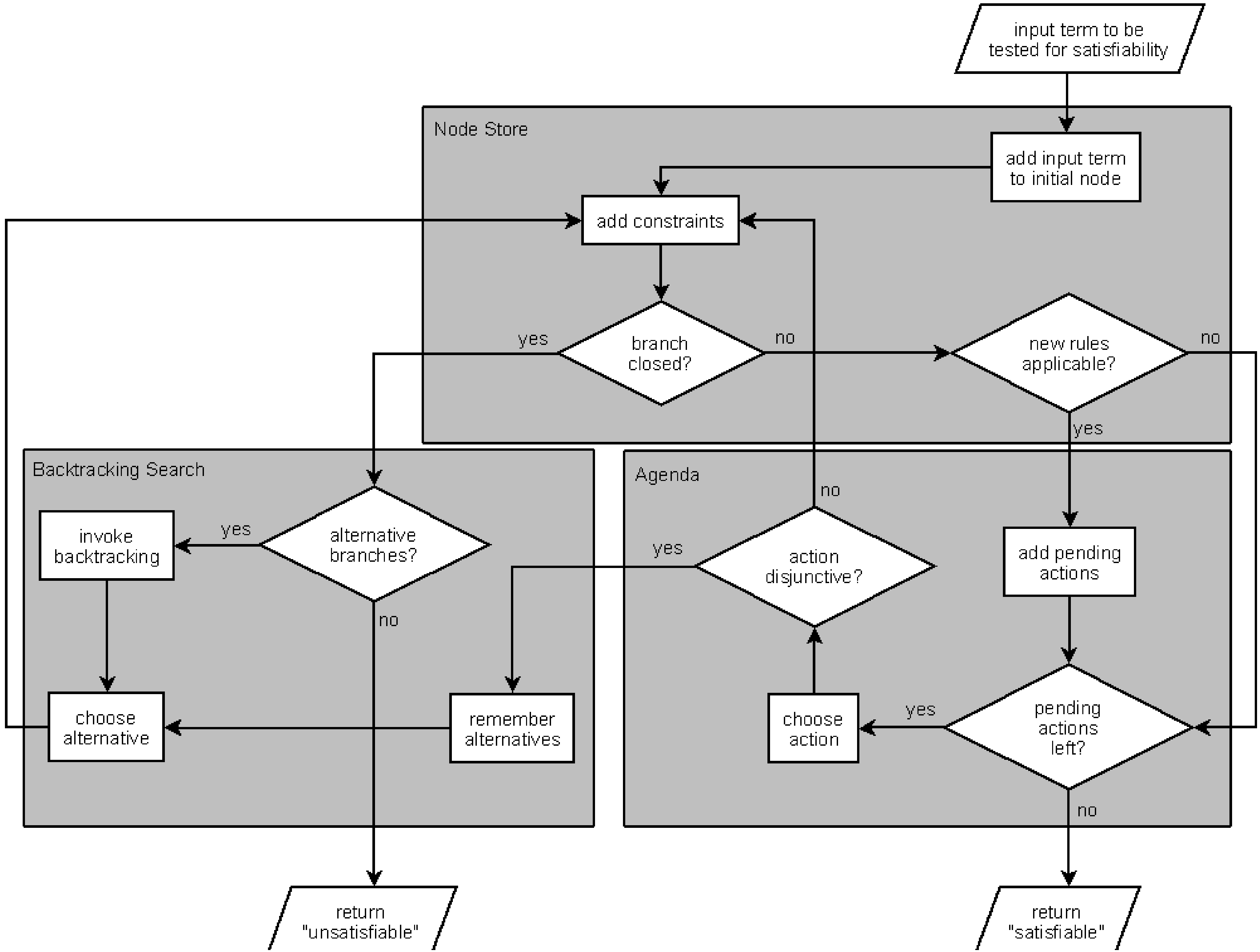
- potential drawback: adding ¬sx may require solving hard subproblem

- (weaker) alternative: no-good lists
    - remember failed alternatives

see also: D'Agostino (1992), Horrocks et. al. (2006), Tsarkov et. al. (2007)

Lines of Code

• total	4500
•core	1700
• tableau algorithm	300
• agenda	400
• node store	850
• backtracking search	150
• blocking	300
• core	150
• pattern stores	150
• additional infrastructure	600
• data structures	950
• preprocessing	950


input term to be tested for satisfiability
Node Store
add input term to initial node
add constraints
branch closed?
yes
no
new rules applicable?
no
yes
Backtracking Search
alternative branches?
yes
no
invoke backtracking
choose alternative
remember alternatives
Agenda
action disjunctive?
yes
no
add pending actions
pending actions left?
yes
no
choose action
return "unsatisfiable"
return "satisfiable"