Practical Partition-Based Theorem Proving for Large Knowledge Bases

Bill MacCartney (Stanford KSL)
Sheila A. McIlraith (Stanford KSL)
Eyal Amir (UC Berkeley)
Tomas Uribe (SRI)

with thanks to
Mark Stickel (SRI)

Motivation

- Goal: to enable automated reasoners to exploit the implicit structure of large knowledge bases
- Reasoners in big KBs face combinatorial explosion
  - Making headway often requires KB-specific manual tuning
- But, large commonsense KBs contain structure
  - Loosely-coupled clusters of domain knowledge
- Partitioning aims to speed reasoning by:
  - Decomposing graph structure of KB into a tree of partitions
  - Propagating results between partitions using message-passing
  - Thereby, focusing proof search and ignoring the irrelevant

Outline

- Background: partition-based reasoning
  - Algorithms for automatic partitioning of large KBs
  - The MP algorithm for reasoning with partitions
- Experimental evaluation of MP
- Partition-derived ordering (PDO)
  - Automatic alternative to hand-crafted symbol orderings
- MP with focused support (MFS)
  - Enhancing vanilla MP with a smart within-partition strategy
- Combinations of strategies
  - Can outperform set-of-support by 10x or more

The espresso machine theory

A simple KB of propositional logic

(we normally use first-order logic)

(1) $\textit{ok-pump} \land \textit{on-pump} \rightarrow \textit{water}$

(2) $\textit{man-fill} \rightarrow \textit{water}$

(3) $\textit{man-fill} \rightarrow \neg\textit{on-pump}$

(4) $\neg\textit{man-fill} \rightarrow \textit{on-pump}$

(5) $\textit{water} \land \textit{ok-boiler} \land \textit{on-boiler} \rightarrow \textit{steam}$

(6) $\neg\textit{water} \rightarrow \neg\textit{steam}$

(7) $\neg\textit{on-boiler} \rightarrow \neg\textit{steam}$

(8) $\neg\textit{ok-boiler} \rightarrow \neg\textit{steam}$

(9) $\textit{steam} \land \textit{coffee} \rightarrow \textit{hot-drink}$

(10) $\textit{steam} \land \textit{tea} \rightarrow \textit{hot-drink}$

(11) $\textit{coffee} \lor \textit{tea}$

Outline

- Background: partition-based reasoning
  - Algorithms for automatic partitioning of large KBs
  - The MP algorithm for reasoning with partitions
- Experimental evaluation of MP
- Partition-derived ordering (PDO)
  - Automatic alternative to hand-crafted symbol orderings
- MP with focused support (MFS)
  - Enhancing vanilla MP with a smart within-partition strategy
- Combinations of strategies
  - Can outperform set-of-support by 10x or more

Automatic partitioning

Step 1: construct symbol graph

- Nodes are symbols in KB
- Edges connect nodes which appear together in an axiom
- Symbol graph captures structure of KB

(1) $\mathit{ok\text{-}pump} \wedge \mathit{on\text{-}pump} \rightarrow \mathit{water}$

(2) $\mathit{man\text{-}fill} \rightarrow \mathit{water}$

(3) $\mathit{man\text{-}fill} \rightarrow \neg \mathit{on\text{-}pump}$

(4) $\neg \mathit{man\text{-}fill} \rightarrow \mathit{on\text{-}pump}$

(5) $\mathit{water} \wedge \mathit{ok\text{-}boiler} \wedge \mathit{on\text{-}boiler} \rightarrow \mathit{steam}$

(6) $\neg \mathit{water} \rightarrow \neg \mathit{steam}$

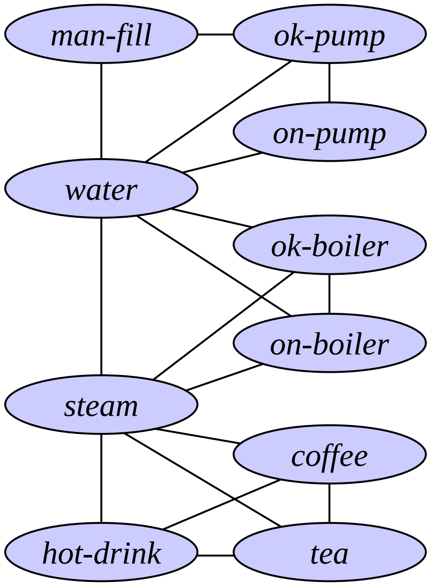
(7) $\neg \mathit{on\text{-}boiler} \rightarrow \neg \mathit{steam}$

(8) $\neg \mathit{ok\text{-}boiler} \rightarrow \neg \mathit{steam}$

(9) $\mathit{steam} \wedge \mathit{coffee} \rightarrow \mathit{hot\text{-}drink}$

(10) $\mathit{steam} \wedge \mathit{tea} \rightarrow \mathit{hot\text{-}drink}$

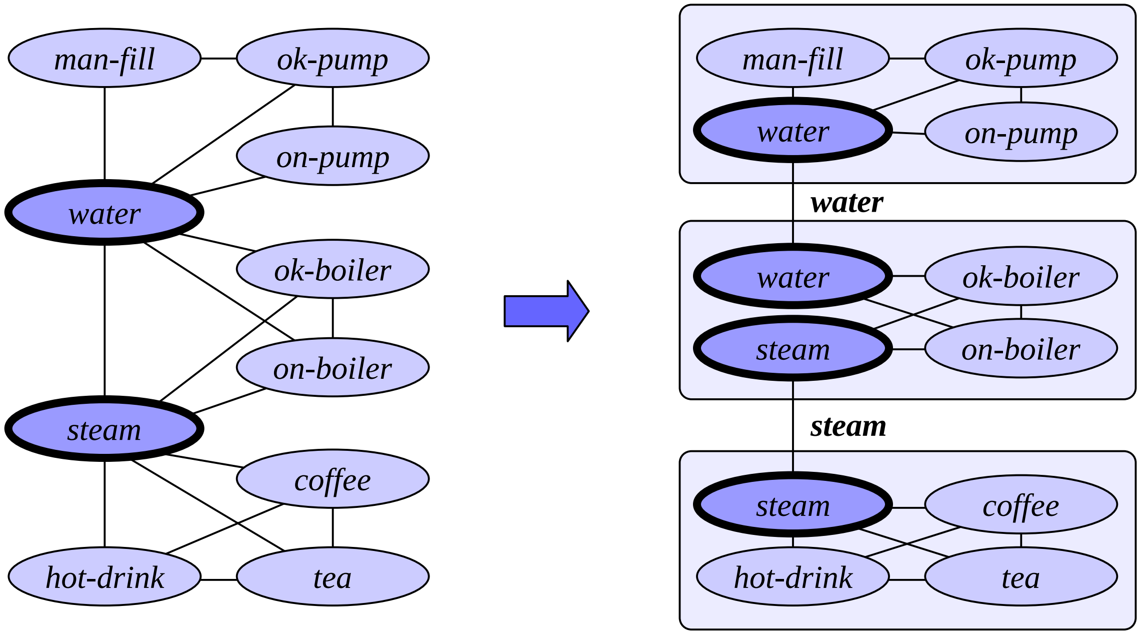
(11) $\mathit{coffee} \vee \mathit{tea}$



Automatic partitioning

Step 2: construct tree decomposition

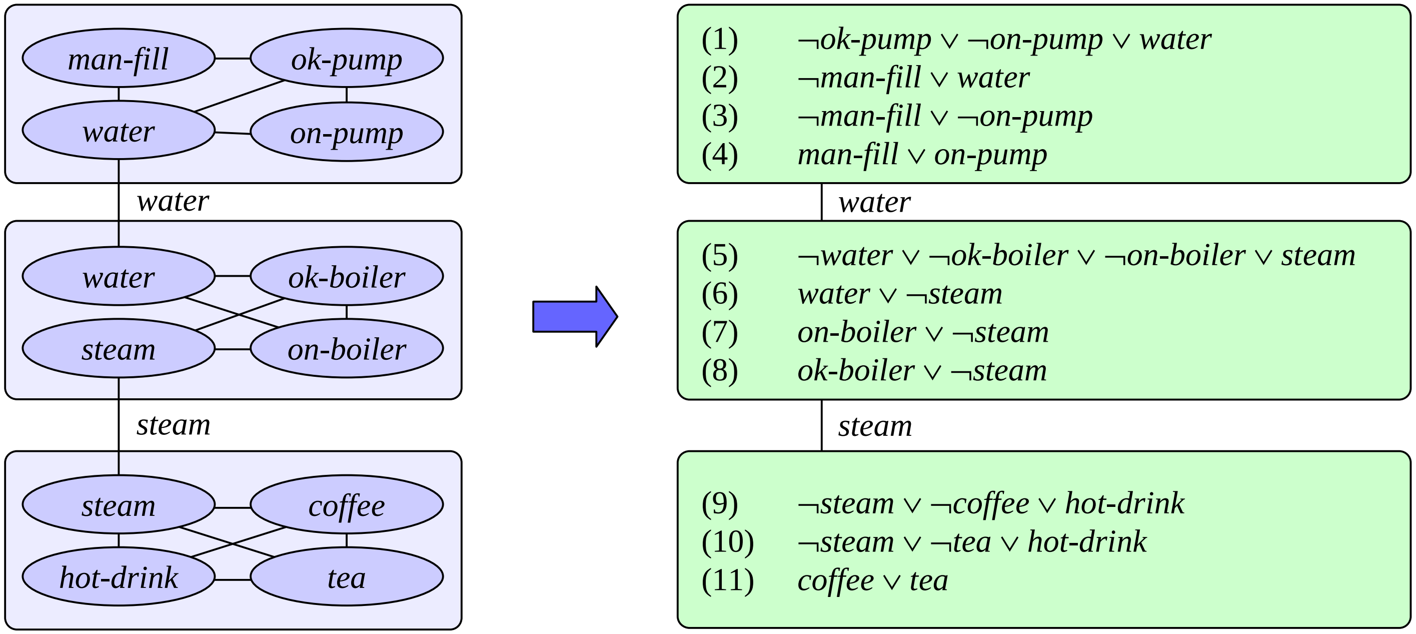
- Each node in tree decomposition corresponds to a tightly-connected cluster of symbols → a partition
- [Amir 2001] gives algorithm which approximates the optimal decomposition by a factor *O*(log *t*)



Automatic partitioning

Step 3: generate partition graph

- Allocate axioms to partitions according to vocabulary
- “Link languages” are defined by shared vocabularies
- Efficient reasoning depends on keeping link vocabularies small



(1) $\neg$*ok-pump* $\vee$ $\neg$*on-pump* $\vee$ *water*
(2) $\neg$*man-fill* $\vee$ *water*
(3) $\neg$*man-fill* $\vee$ $\neg$*on-pump*
(4) *man-fill* $\vee$ *on-pump*

water

(5) $\neg$*water* $\vee$ $\neg$*ok-boiler* $\vee$ $\neg$*on-boiler* $\vee$ *steam*
(6) *water* $\vee$ $\neg$*steam*
(7) *on-boiler* $\vee$ $\neg$*steam*
(8) *ok-boiler* $\vee$ $\neg$*steam*

steam

(9) $\neg$*steam* $\vee$ $\neg$*coffee* $\vee$ *hot-drink*
(10) $\neg$*steam* $\vee$ $\neg$*tea* $\vee$ *hot-drink*
(11) *coffee* $\vee$ *tea*

Outline

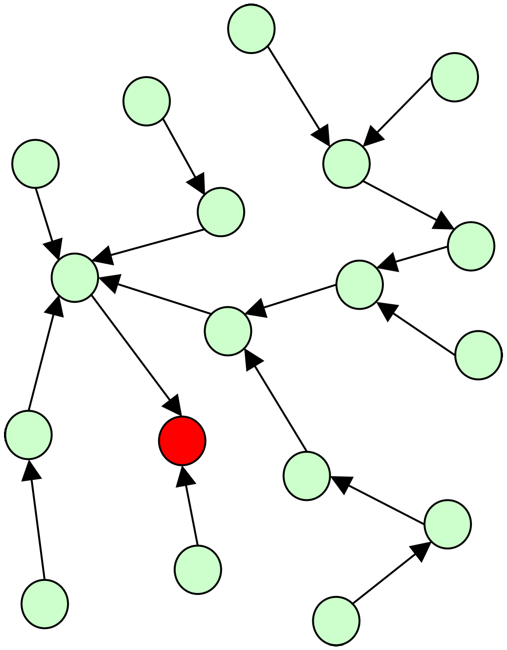
- Background: partition-based reasoning
  - Algorithms for automatic partitioning of large KBs
  - The MP algorithm for reasoning with partitions
- Experimental evaluation of MP
- Partition-derived ordering (PDO)
  - Automatic alternative to hand-crafted symbol orderings
- MP with focused support (MFS)
  - Enhancing vanilla MP with a smart within-partition strategy
- Combinations of strategies
  - Can outperform set-of-support by 10x or more

Reasoning with MP

MP Algorithm

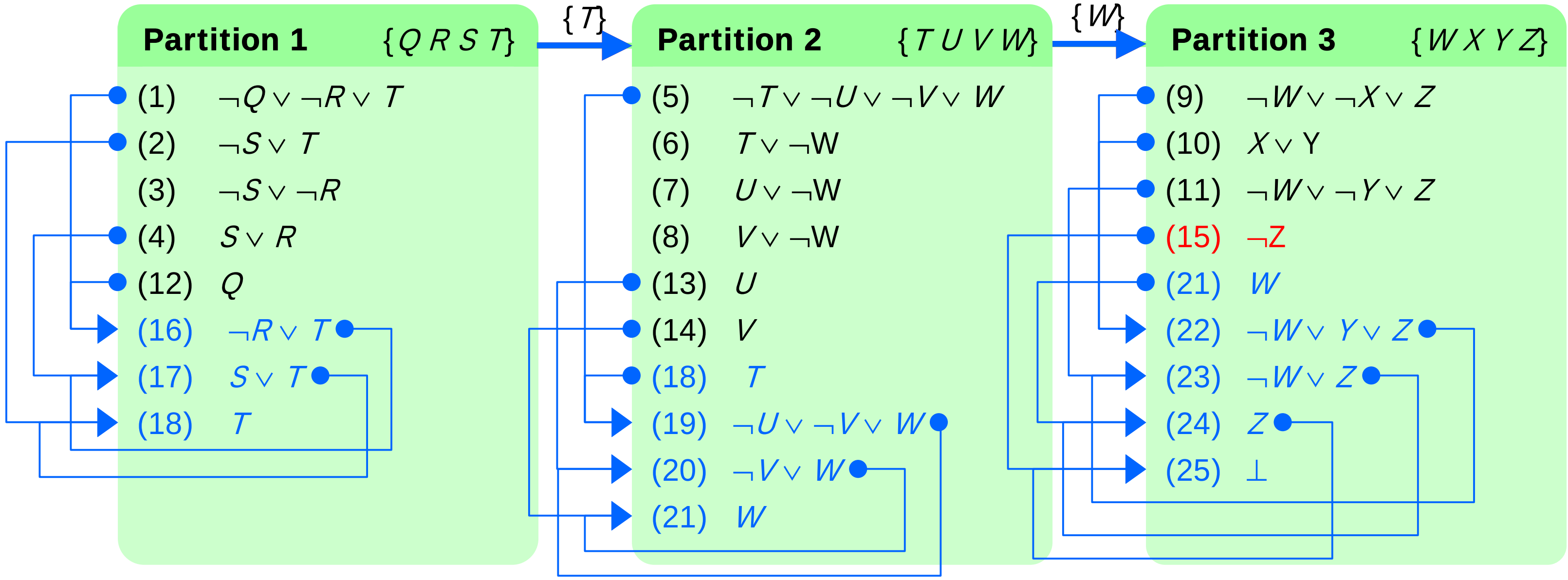
[Amir & McIlraith 2000]

- Start with a tree-structured partition graph
- Identify goal partition
  (based on matching vocabulary)
- Direct edges toward goal
  (fixing outbound link language L_i for each partition)
- Concurrently, in each partition:
  - Generate consequences in L_i
  - Pass messages in L_i toward goal



MP in action

Query: $Q \wedge U \wedge V \rightarrow Z$?



Partition 1 $\{Q\ R\ S\ T\}$

(1) $\neg Q \vee \neg R \vee T$

(2) $\neg S \vee T$

(3) $\neg S \vee \neg R$

(4) $S \vee R$

(12) Q

(16) $\neg R \vee T$

(17) $S \vee T$

(18) T

$\{T\}$

Partition 2 $\{T\ U\ V\ W\}$

(5) $\neg T \vee \neg U \vee \neg V \vee W$

(6) $T \vee \neg W$

(7) $U \vee \neg W$

(8) $V \vee \neg W$

(13) U

(14) V

(18) T

(19) $\neg U \vee \neg V \vee W$

(20) $\neg V \vee W$

(21) W

$\{W\}$

Partition 3 $\{W\ X\ Y\ Z\}$

(9) $\neg W \vee \neg X \vee Z$

(10) $X \vee Y$

(11) $\neg W \vee \neg Y \vee Z$

(15) $\neg Z$

(21) W

(22) $\neg W \vee Y \vee Z$

(23) $\neg W \vee Z$

(24) Z

(25) $\perp$

Using partitioning, this query took just 10 resolution steps.
Using set-of-support, the same query can take 28 steps.

Characteristics of MP

- Reasoning is performed locally in each partition
- Relevant results propagate toward goal partition
- Globally sound & complete

  ... provided each local reasoner is sound & complete for L_i-consequence finding

  [Amir & McIlraith 2000]
- Performance is worst-case exponential within partitions, but linear in tree structure

Minimizes *between-partition* deduction

Focuses *within-partition* deduction

Supports parallel processing

Different reasoners in different partitions

Outline

- Background: partition-based reasoning
  - Algorithms for automatic partitioning of large KBs
  - The MP algorithm for reasoning with partitions
- Experimental evaluation of MP
- Partition-derived ordering (PDO)
  - Automatic alternative to hand-crafted symbol orderings
- MP with focused support (MFS)
  - Enhancing vanilla MP with a smart within-partition strategy
- Combinations of strategies
  - Can outperform set-of-support by 10x or more

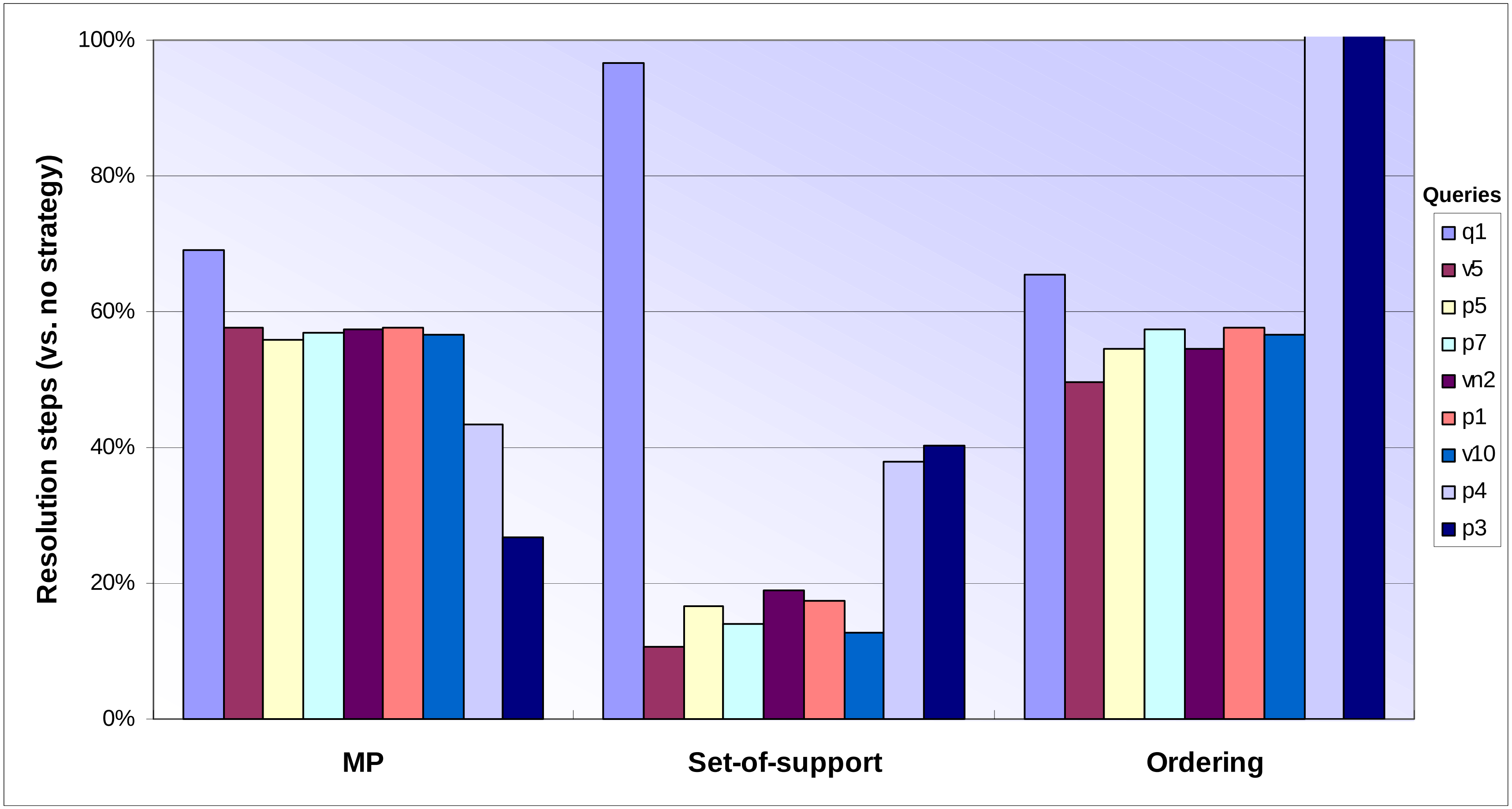
Experimental Evaluation of MP

- Do “real world” KBs exhibit inherent structure?
  - Do they have good tree decompositions (partition graphs)?
  - Can partition-based reasoning outperform other strategies?
- Experimental testbed
  - Theorem prover: SNARK
  - KB: Cyc
    - A subset on spatial relationships, ~750 axioms, ~150 symbols
    - We’re working on adding SUMO, others
  - Queries from outside source
  - Number of resolution steps used as chief performance metric
  - Normalized to number of steps required using no strategy

Comparison to conventional strategies

- Restriction strategies focus proof search
  - Disallow some resolution steps to speed search
  - Completeness issues are critical
- Set-of-support restriction
  - Place the negated query into a designated “set of support”
  - Allow only resolutions involving a clause from the set of support
  - Add newly-derived clauses to set of support
- Ordering restriction
  - Define a global ordering among predicates
  - Resolve on predicates in order from greatest to least
  - (SNARK provides a default ordering, which is arbitrary)

Experimental results: “vanilla” MP



Outline

- Background: partition-based reasoning
  - Algorithms for automatic partitioning of large KBs
  - The MP algorithm for reasoning with partitions
- Experimental evaluation of MP
- Partition-derived ordering (PDO)
  - Automatic alternative to hand-crafted symbol orderings
- MP with focused support (MFS)
  - Enhancing vanilla MP with a smart within-partition strategy
- Combinations of strategies
  - Can outperform set-of-support by 10x or more

Motivation for PDO

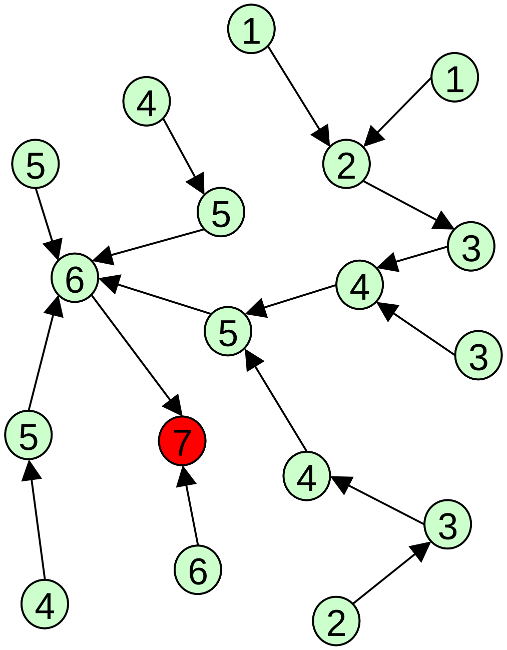
- Ordered resolution can be highly efficient
- Voronkov: best modern resolution provers use ordering to reduce search space
- But success depends on having the right ordering
- Until now, successful orderings have been
  - Laboriously hand-crafted
  - Tailored to a specific KB
  - Poorly understood
- Insight: partitioning can induce a good ordering

How PDO works

- Generate a partition-derived ordering

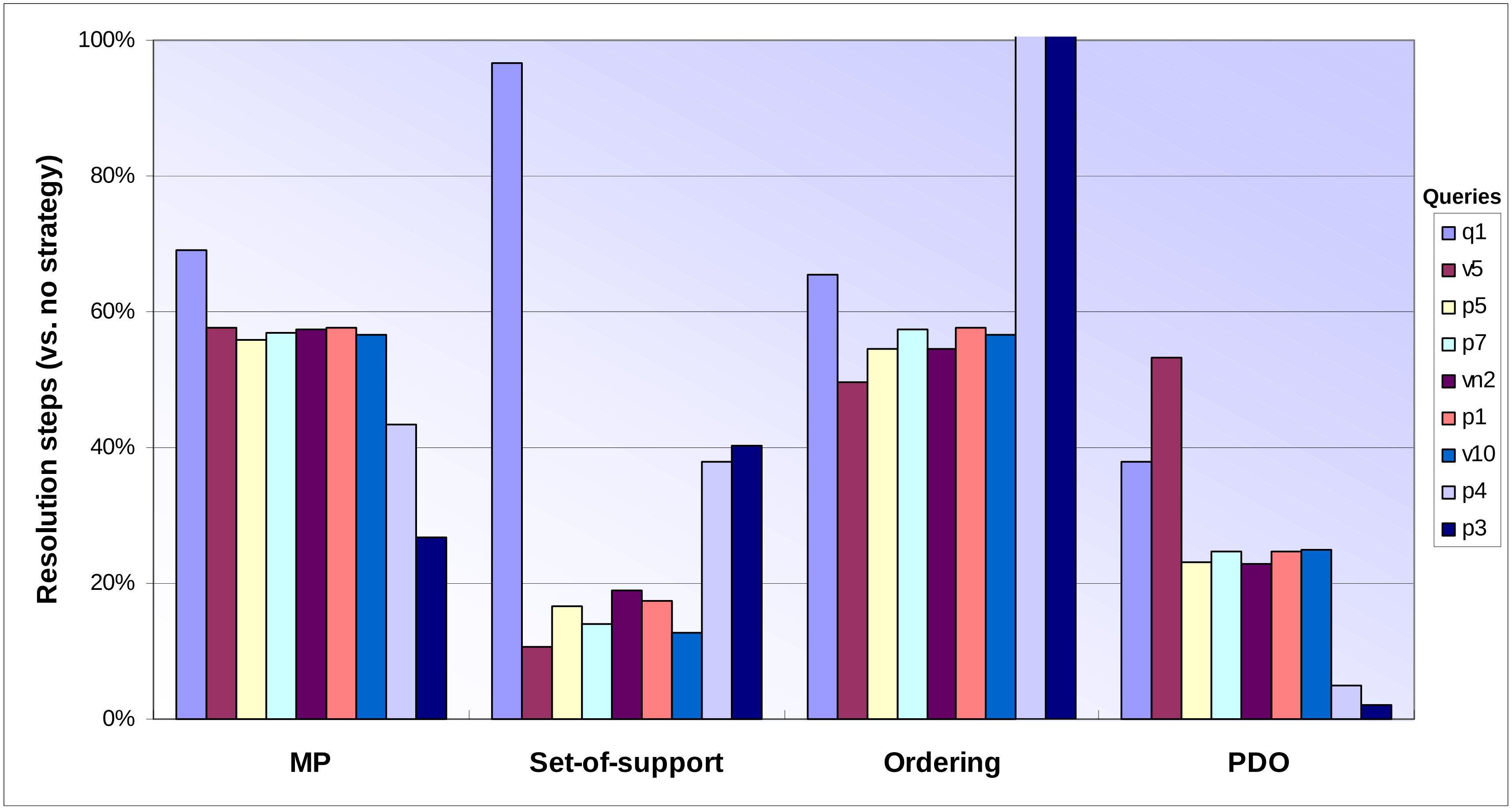
1. Direct edges of partition graph toward goal partition
2. Perform topological sort on partitions
3. Beginning with partitions furthest from goal, progressively append symbols from each partition to ordering

- Use result as input for ordered resolution
  - (Partition graph can now be discarded)
  - Sound & complete



- PDO roughly simulates MP

Experimental results: PDO



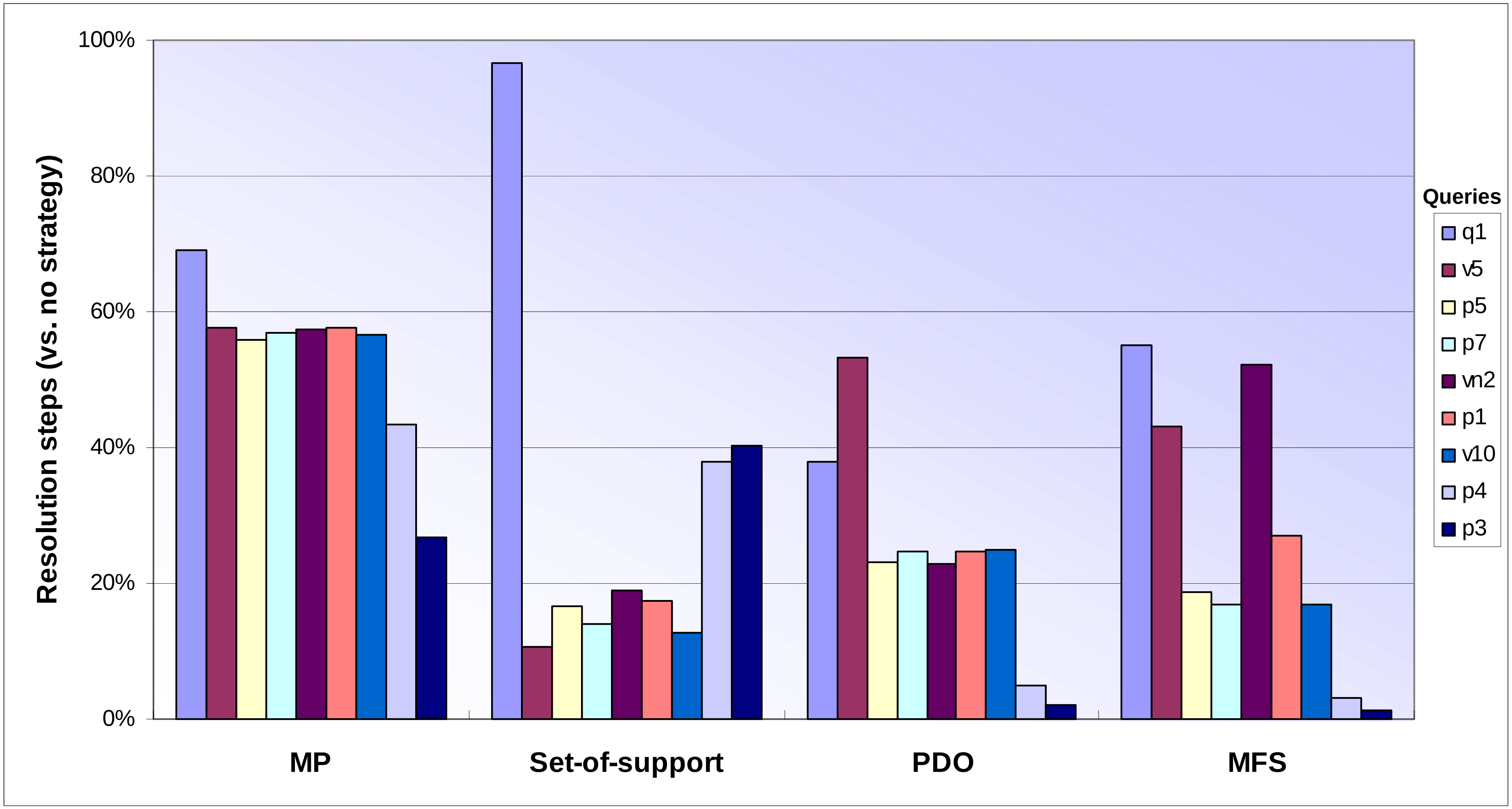
Outline

- Background: partition-based reasoning
  - Algorithms for automatic partitioning of large KBs
  - The MP algorithm for reasoning with partitions
- Experimental evaluation of MP
- Partition-derived ordering (PDO)
  - Automatic alternative to hand-crafted symbol orderings
- MP with focused support (MFS)
  - Enhancing vanilla MP with a smart within-partition strategy
- Combinations of strategies
  - Can outperform set-of-support by 10x or more

MP with focused support (MFS)

- Motivating intuition
  - Only results in the outbound link vocabulary can be propagated
  - So, focus within-partition reasoning on generating such results
- The "focused support" restriction
  - Initialize set *S* to contain any clause in the partition that includes a symbol in outbound link language.
  - Resolve two clauses only if one is in *S* and the resolved predicate is not in outbound link language. Add the resolvent to *S*.
- MFS is globally sound & complete [see paper for proof]

Experimental results: MFS



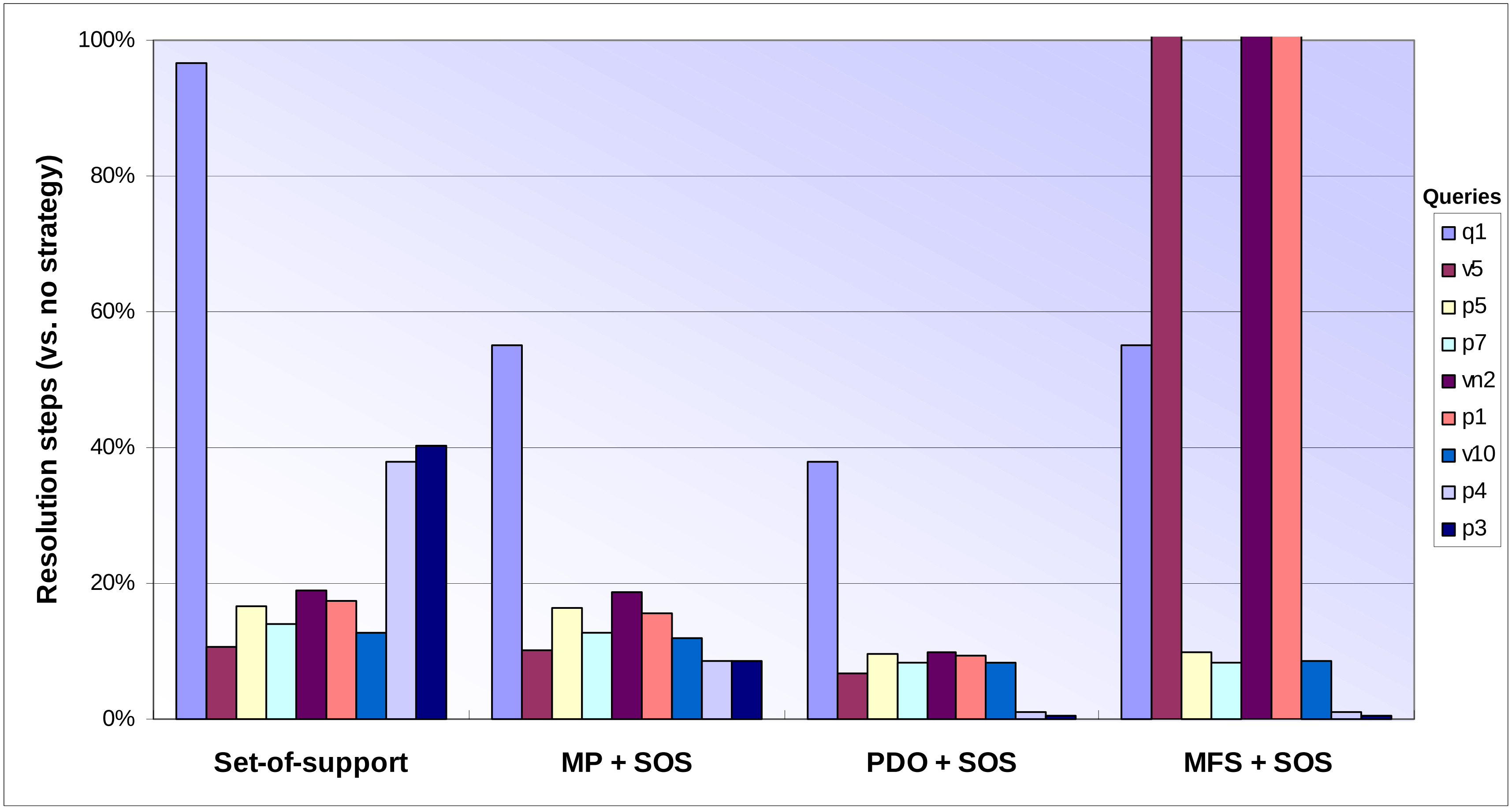
Outline

- Background: partition-based reasoning
  - Algorithms for automatic partitioning of large KBs
  - The MP algorithm for reasoning with partitions
- Experimental evaluation of MP
- Partition-derived ordering (PDO)
  - Automatic alternative to hand-crafted symbol orderings
- MP with focused support (MFS)
  - Enhancing vanilla MP with a smart within-partition strategy
- Combinations of strategies
  - Can outperform set-of-support by 10x or more

Strategy combinations

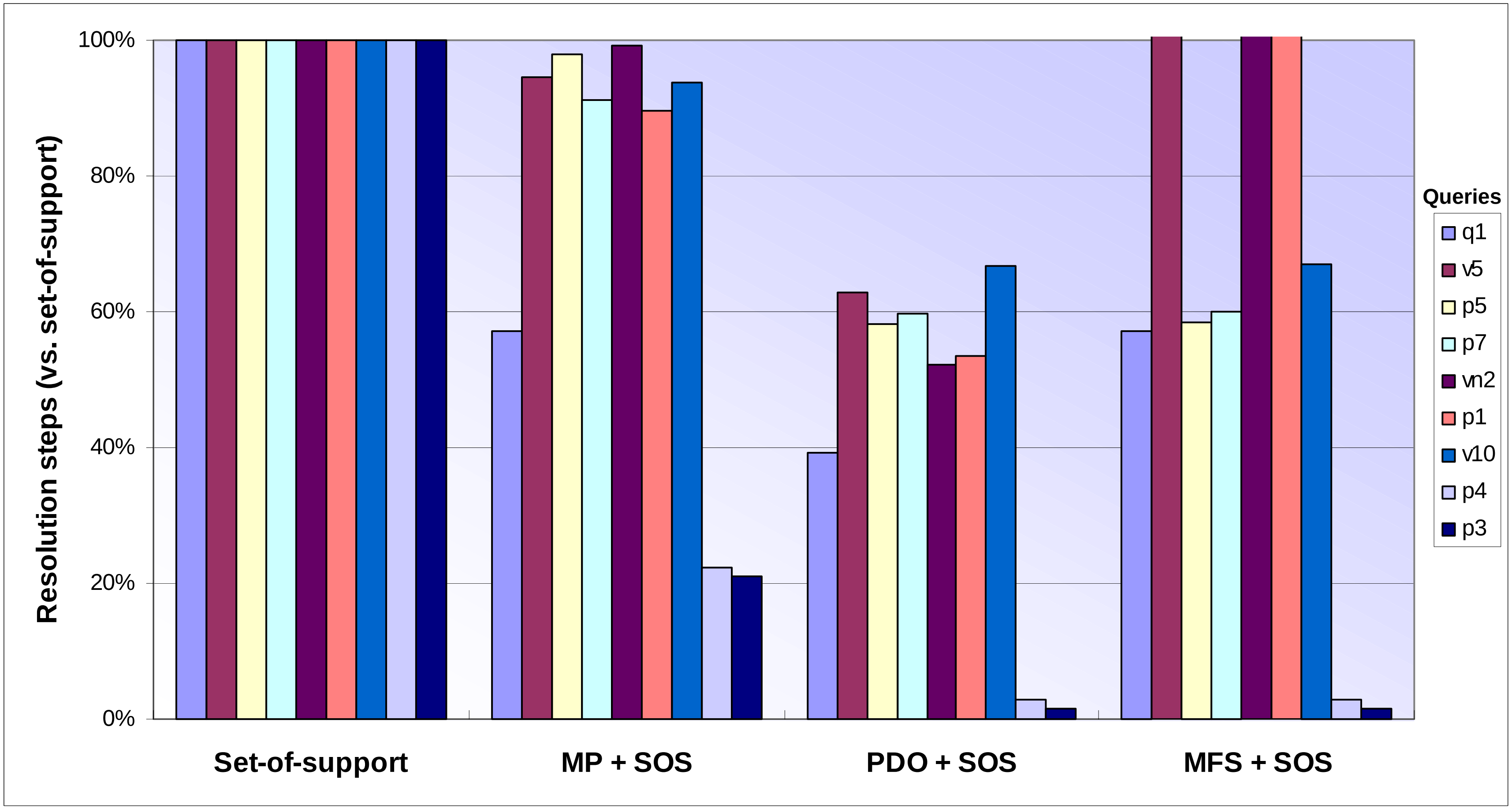
- Combine MP, PDO, or MFS with set-of-support
  - Maintain a set of support at global level
  - Allow resolution between two clauses only if they are in the same partition and at least one of them is in the support
- Completeness
  - These combinations are in general not complete
  - Incompleteness sometimes revealed in practice
- Performance
  - However, combinations outperform any single strategy

Experimental results: strategy combos



Experimental results: strategy combos

(same results, re-normalized vs. set-of-support)



Conclusions and Future Work

- Partitioning can speed up reasoning
  - Exploits implicit structure of large commonsense KBs
  - Reasoning becomes significantly more focused and efficient
  - MFS does even better by focusing reasoning *within* partitions
- Partition-derived ordering is surprisingly effective
  - Especially when combined with set-of-support
  - Automatic alternative to hand-crafted orderings
- Future work
  - Greater diversity of experimental results
    - Obstacle: scarcity of large KBs usable with generic FOL prover
  - Assessing the potential benefit of parallelization

References

Web

www.ksl.stanford.edu/projects/RKF/Partitioning/

Papers

- MacCartney, B., McIlraith, S., Amir, E. and Uribe, T., “Practical Partition-Based Theorem Proving for Large Knowledge Bases,” *18th International Joint Conference on Artificial Intelligence* (IJCAI-03), 2003.
- Amir, E. and McIlraith, S., “Partition-Based Logical Reasoning for First-Order and Propositional Theories,” accepted for publication in *Artificial Intelligence*.
- McIlraith, S. and Amir, E., “Theorem Proving with Structured Theories,” *17th International Joint Conference on Artificial Intelligence* (IJCAI-01), 2001.
- Amir, E., “Efficient Approximation for Triangulation of Minimum Treewidth,” *17th Conference on Uncertainty in Artificial Intelligence* (UAI ’01), 2001.
- Amir, E. and McIlraith, S., “Solving Satisfiability using Decomposition and the Most Constrained Subproblem.” *Proceedings of SAT 2001*, 2001.
- Amir, E. and McIlraith, S., “Partition-Based Logical Reasoning,” *7th International Conference on Principles of Knowledge Representation and Reasoning* (KR ’2000), 2000.

Thanks!

Results: automatic partitioning

- Partition graph is largely independent of query
  - But edges may need to be redirected
- We’re experimenting with multiple algorithms

	Alg 5	Alg 6
Number of partitions	124	40
Max symbols/partition	16	19
Max symbols/link	14	17
Max axioms/partition	80	95
Max partitions/axiom	25	28
Axioms in multiple partitions	152	152

Queries

hd-q1 If the pump is OK and the boiler is OK and the boiler is on, do we get a hot drink?

cyc-p5 If A and B are inside C, can C be inside A?

cyc-p7 If A and B are part of C and C is at D, where is A?

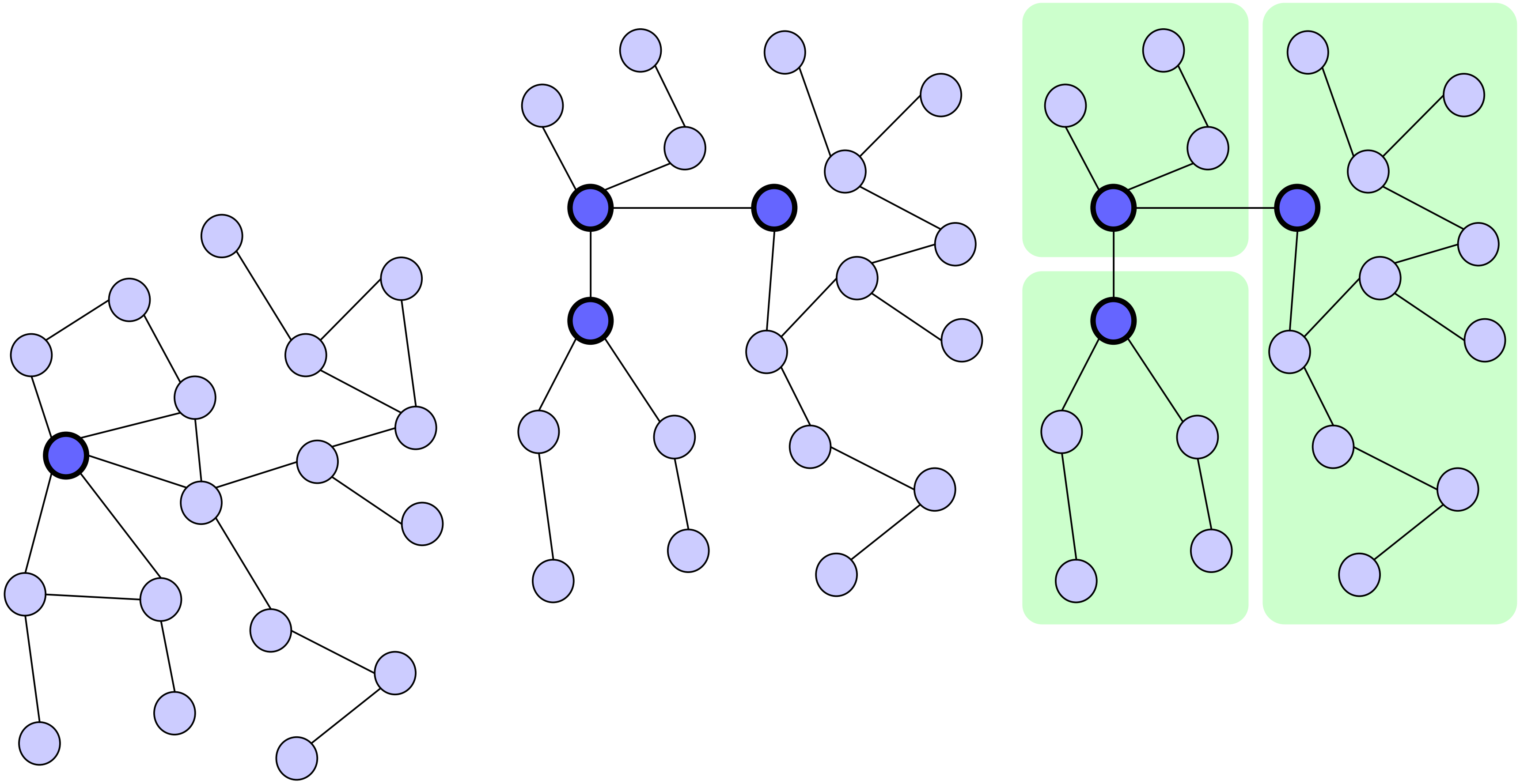
cyc-p1 Suppose that A is touching B and B is inside C and C is at D. Is A at D?

cyc-v5 A has parts B, C, and D. B has parts E, and F. Is F near A?

cyc-p3 If C is between A and B, and both A and B are inside D, and D is at E, is C at E?

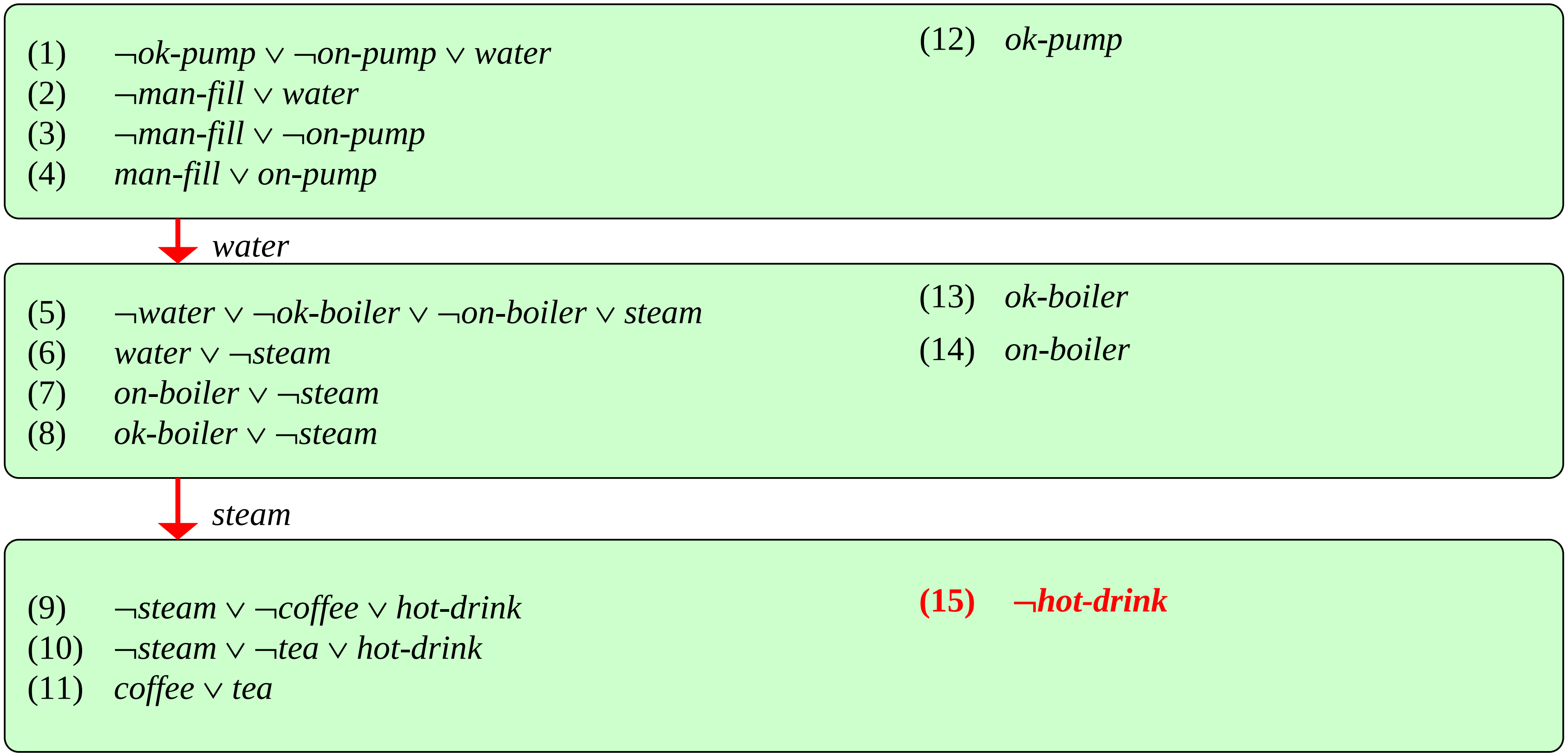
cyc-p4 If C is between A and B, and both A and B are at D, is C also at D?

Automatic partitioning



MP in action

Query: If the pump is OK and the boiler is OK and the boiler is on, do we get a hot drink?



(1) ¬*ok-pump* ∨ ¬*on-pump* ∨ *water*
(2) ¬*man-fill* ∨ *water*
(3) ¬*man-fill* ∨ ¬*on-pump*
(4) *man-fill* ∨ *on-pump*

(12) *ok-pump*

water

(5) ¬*water* ∨ ¬*ok-boiler* ∨ ¬*on-boiler* ∨ *steam*
(6) *water* ∨ ¬*steam*
(7) *on-boiler* ∨ ¬*steam*
(8) *ok-boiler* ∨ ¬*steam*

(13) *ok-boiler*

(14) *on-boiler*

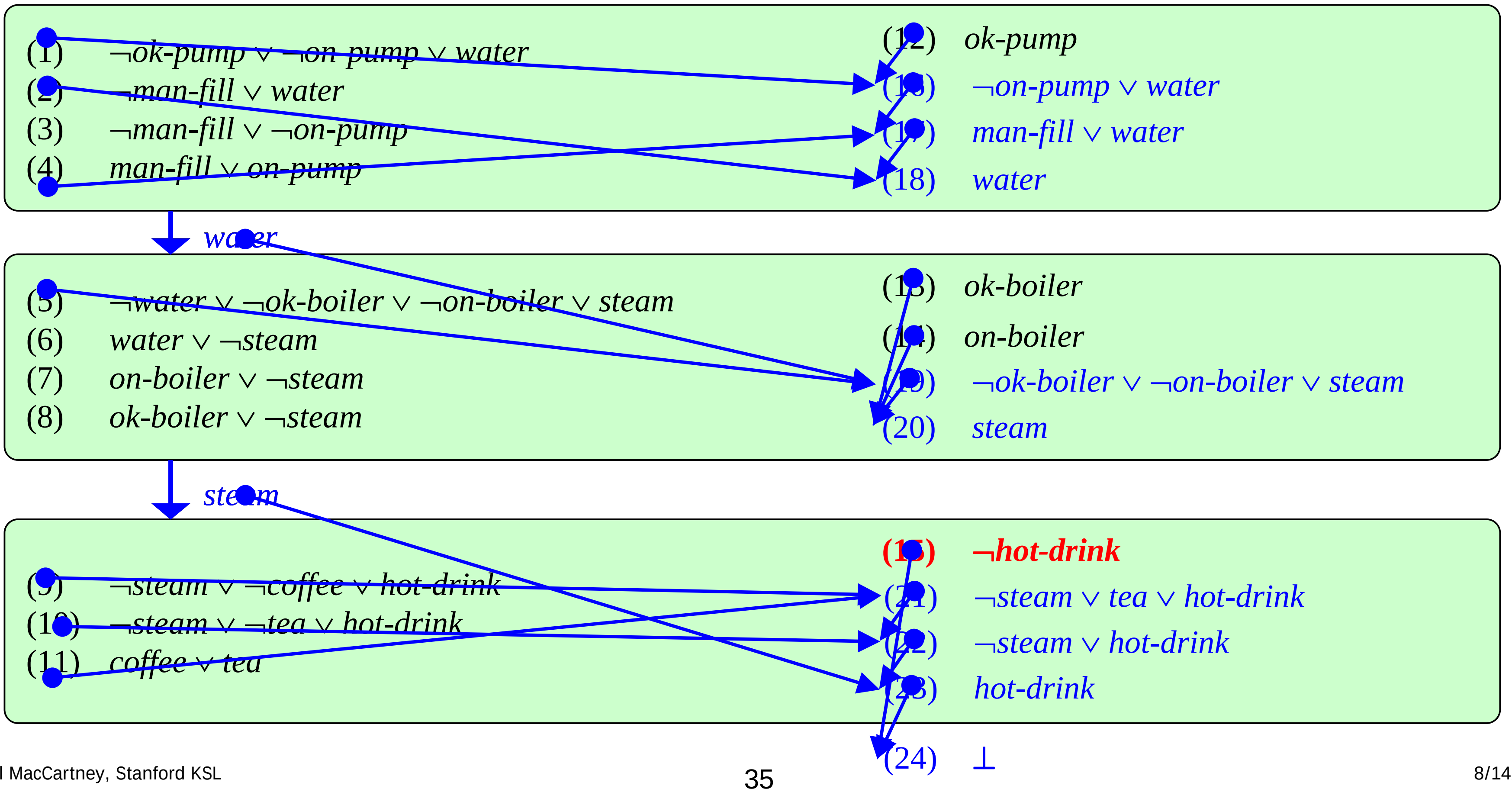
steam

(9) ¬*steam* ∨ ¬*coffee* ∨ *hot-drink*
(10) ¬*steam* ∨ ¬*tea* ∨ *hot-drink*
(11) *coffee* ∨ *tea*

(15) ¬*hot-drink*

MP in action

Using set-of-support, SNARK took 28 steps to prove this.
Using partitioning, SNARK took just 11 steps.



(1) ¬ok-pump ∨ ¬on-pump ∨ water
(2) ¬man-fill ∨ water
(3) ¬man-fill ∨ ¬on-pump
(4) man-fill ∨ on-pump

(12) ok-pump
(16) ¬on-pump ∨ water
(17) man-fill ∨ water
(18) water

water

(5) ¬water ∨ ¬ok-boiler ∨ ¬on-boiler ∨ steam
(6) water ∨ ¬steam
(7) on-boiler ∨ ¬steam
(8) ok-boiler ∨ ¬steam

(13) ok-boiler
(14) on-boiler
(19) ¬ok-boiler ∨ ¬on-boiler ∨ steam
(20) steam

steam

(9) ¬steam ∨ ¬coffee ∨ hot-drink
(10) ¬steam ∨ ¬tea ∨ hot-drink
(11) coffee ∨ tea

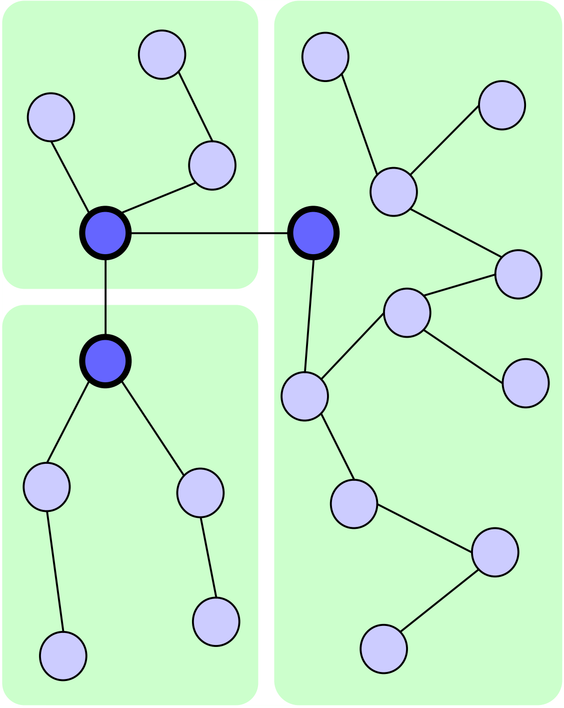
(15) ¬hot-drink
(21) ¬steam ∨ tea ∨ hot-drink
(22) ¬steam ∨ hot-drink
(23) hot-drink
(24) ⊥

Ongoing research

- Testing on more KBs
  - Finding good test data is a real challenge
- Characterizing the queries for which MP and its extensions work especially well
- Assessing the potential benefit of parallelization
  - Current implementation is serial
  - But reasoning within partitions can happen concurrently
- Distributed implementations
  - Demonstrating integration of heterogeneous reasoners

Recap: automatic partitioning

- Begin with a KB in PL or FOL
- Construct symbol graph
  - Edges join symbols which appear together in an axiom
- Apply tree decomposition algorithm
  - We use an adaptation of min-fill
- Partition axioms correspondingly
  - Each partition has its own vocabulary
  - “Link languages” are defined by shared vocabulary



Efficient reasoning depends on keeping partition sizes and link sizes small