Compilation of Query-Rewriting Problems into Tractable Fragments of Propositional Logic

Yolifé Arvelo Blai Bonet María Esther Vidal
Departamento de Computación
Universidad Simón Bolívar
Caracas, Venezuela

Introduction

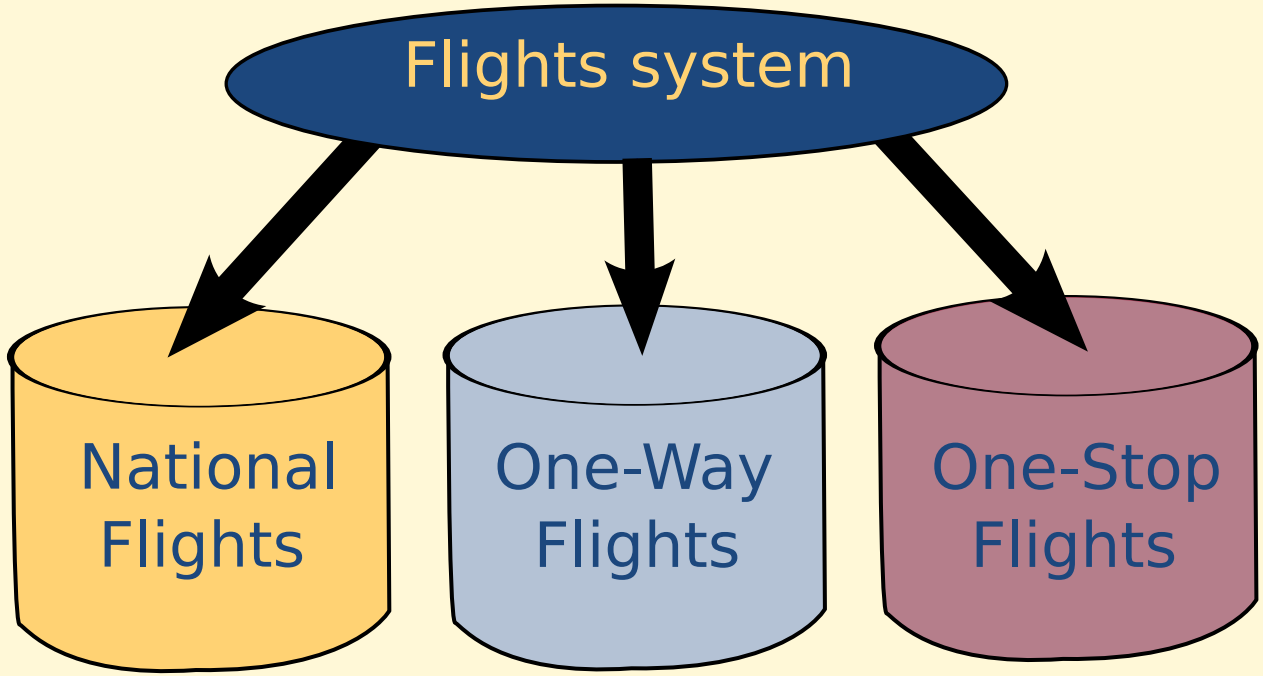
- We consider the problem of rewriting a query using materialized views
- This problem appears frequently in the context of Data Integration, Web Infrastructures and Query Optimization:
  - [Duschka & Genesereth 1997; Kwok & Weld 1996; Lambrecht, Kambhampati & Gnanaprakasam 1999]
  - [Levy, Rajaraman & Ordille 1996; Zaharioudakis et al. 2000; Mitra 2001]
- The problem is in general intractable and existing algorithms do not scale well even in simple cases

Data Integration

- **OBJECTIVE:** Given a query Q, retrieve all tuples **obtainable from the data sources** that satisfy Q
- Data sources are assumed to be:
  - **Independent** (i.e. maintained in a distributed manner)
  - **Described as views** (i.e. the **Local As View** model)
  - **Incomplete**

Data Integration: Example

QUERY: Find round-trip flights that start in the US



Query Rewriting Problem: Example

QUERY: Find round-trip flights that start in the US

$$Q(x, y) \text{ :- } \texttt{flight}(x, y), \texttt{flight}(y, x), \texttt{uscity}(x)$$

Data sources modelled as views:

$$\texttt{national}(x_1, y_1) \text{ :- } \texttt{flight}(x_1, y_1), \texttt{uscity}(x_1), \texttt{uscity}(y_1)$$
$$\texttt{oneway}(x_2, y_2) \text{ :- } \texttt{flight}(x_2, y_2)$$
$$\texttt{onestop}(x_3, z_3) \text{ :- } \texttt{flight}(x_3, y_3), \texttt{flight}(y_3, z_3)$$

Query Rewriting Problem: Solution

- **ASSUMPTION:** Views may be incomplete
- Then, the solution is the **collection** of rewritings:

$$R_1(x,y) \text{ :- } \texttt{oneway}(x,y), \texttt{oneway}(y,x), \texttt{national}(x,w)$$
$$R_2(x,y) \text{ :- } \texttt{oneway}(x,y), \texttt{oneway}(y,x), \texttt{national}(w,x)$$
$$R_3(x,y) \text{ :- } \texttt{national}(x,y), \texttt{national}(y,x)$$
$$R_4(x,y) \text{ :- } \texttt{oneway}(x,y), \texttt{national}(y,x)$$
$$R_5(x,y) \text{ :- } \texttt{national}(x,y), \texttt{oneway}(y,x)$$

- Observe that there is no rewriting using $\texttt{onestop}(x,y)$

Query Rewriting Problem: Formal

- **INPUT:** A query Q and set of views $\mathcal{V} = \{V_1, V_2, \ldots, V_n\}$
- **TASK:** Find a maximal-contained set of rewritings of Q using the views
- A rewriting is a query-like expression that refers only to the views
- **ASSUMPTION:** Q and V_i are **conjunctive** queries without arithmetic predicates

Related Work: Algorithms

- Bucket algorithm [Levy & Rajaraman & Ullman 1996]
- Inverse rules algorithm [Duscka & Genesereth 1997]
- MiniCon algorithm [Pottinger & Halevy 2001]

The MiniCon Algorithm [Pottinger & Halevy 2001]

- Exploit independences to decompose into smaller subproblems and then combine solutions
- Solutions to subproblems are called MCDs

MCD	View	Mapping	Covered subgoals
M_1	`national`	$\{X \rightarrow X_1, Y \rightarrow Y_1\}$	$\{0\}$
M_2	`national`	$\{X \rightarrow Y_1, Y \rightarrow X_1\}$	$\{1\}$
M_3	`national`	$\{X \rightarrow X_1\}$	$\{2\}$
M_4	`national`	$\{X \rightarrow Y_1\}$	$\{2\}$
M_5	`oneway`	$\{X \rightarrow X_2, Y \rightarrow Y_2\}$	$\{0\}$
M_6	`oneway`	$\{X \rightarrow Y_2, Y \rightarrow X_2\}$	$\{1\}$

The MiniCon Algorithm: How does it work?

- Generate all MCDs (very expensive since performs **blind search**)
- Rewritings generated **greedily** as combination of MCDs such that:
  - Cover disjoint subsets of subgoals in the query
  - Cover all subgoals in the query
- In the example, combining M_3, M_5, M_6 produces the rewriting:

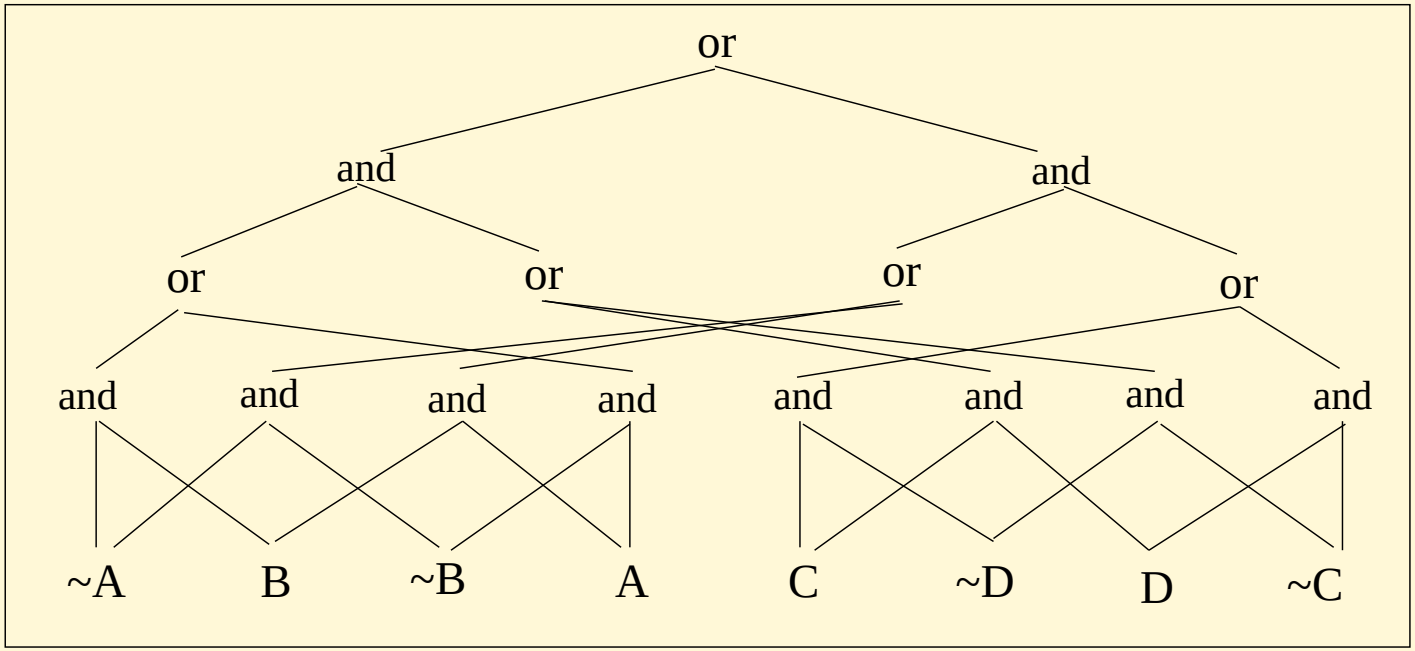
$$R_1(x, y) \text{ :- } \texttt{oneway}(x, y), \texttt{oneway}(y, x), \texttt{national}(x, w)$$

Our Approach: MCDSAT

- Given a query Q and a set of views $\mathcal{V}$
- Build a **propositional theory** such that its models are in correspondence with the MCDs
- Generating MCDs is now a problem of **model enumeration**
- Model enumeration can be done with modern SAT techniques that implement:
  - **Non-chronological backtracking via clause learning**
  - **Caching of common subproblems**
  - **Heuristics**
- We also extend propositional theory such that its models are in correspondence with the rewritings
- **We call our approach MCDSAT!!**

Negation Normal Forms (NNF)

- A formula is in Negation Normal Form (NNF) if constructed from literals using only conjunctions and disjunctions [Barwise 1977]
- It can be represented as a rooted DAG whose leaves are literals and internal nodes are labeled with conjunction or disjunction

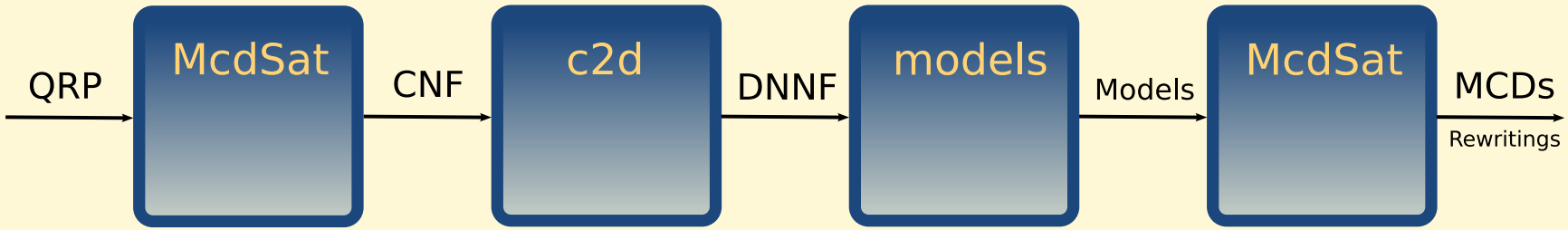


Deterministic and Decomposable NNFs (d-DNNFs)

- Introduced by [Darwiche 2001]
- A NNF is **decomposable** if each variable appears at most once below each conjunct
- A NNF is **deterministic** if disjuncts are pairwise logically inconsistent
- A d-DNNF supports a number of operations in **linear time**:
  - satisfiability
  - clause entailment
  - model counting
  - **model enumeration (output linear time)**
  - ...
- Transformation into d-DNNF is **intractable in the worst case**, but **not necessarily so on average**

Implementation

- MCDSAT translates QRP into a propositional theory T
- T is compiled into d-DNNF using Darwiche's `c2d` compiler
- Models are **obtained** from the d-DNNF and transformed into MCDs or rewritings



- `c2d` and `models` are off-the-shelf components
- MCDSAT written in scripting language

Experimental Study

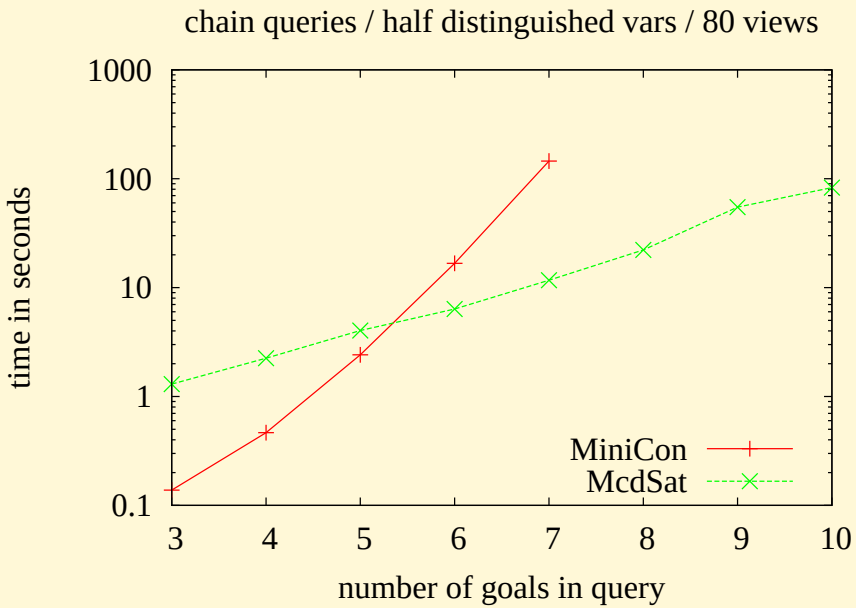
OBJECTIVE: To study the effect of the query sizes and number of views in the performance of MCDSAT and MiniCon

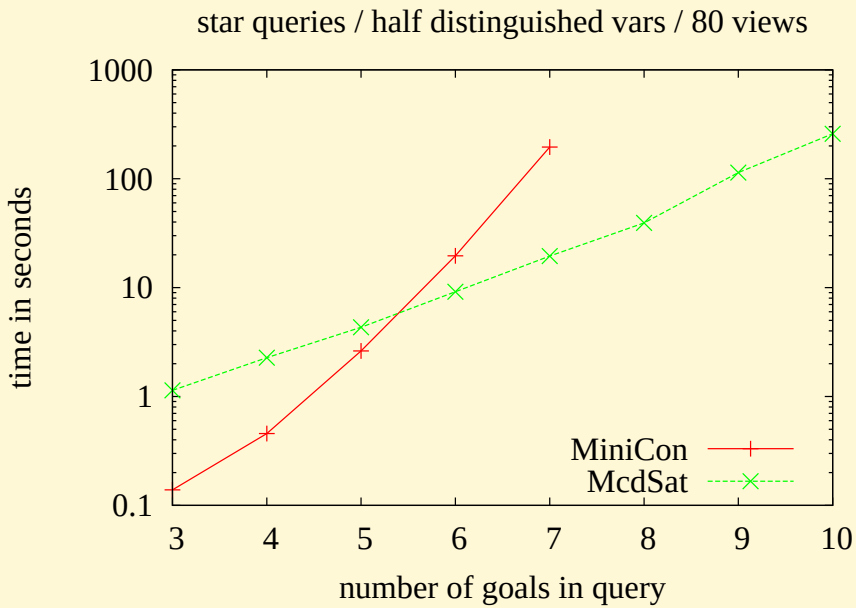
- Large benchmark with problems of different sizes and structures
- Comparison metric: **time**
- For lack of space, we only report few instances

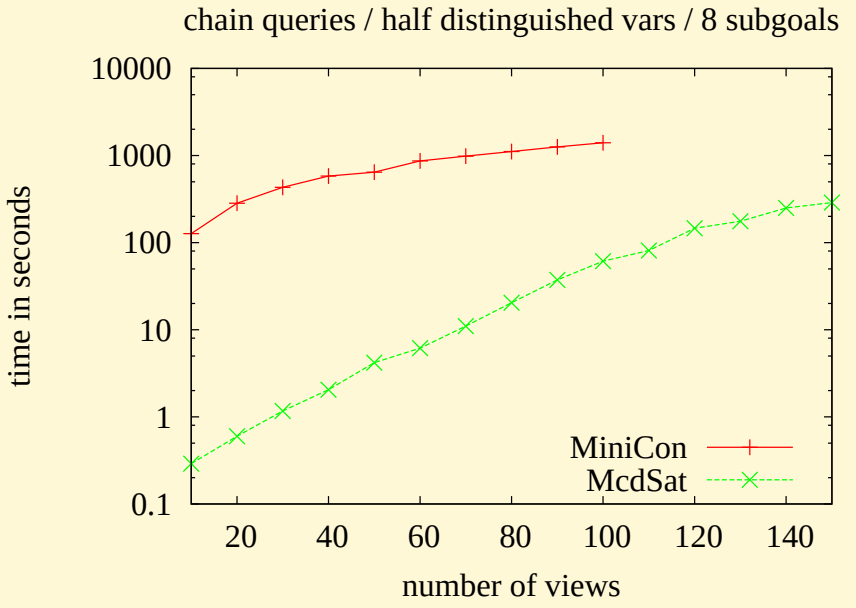
Experimental Results

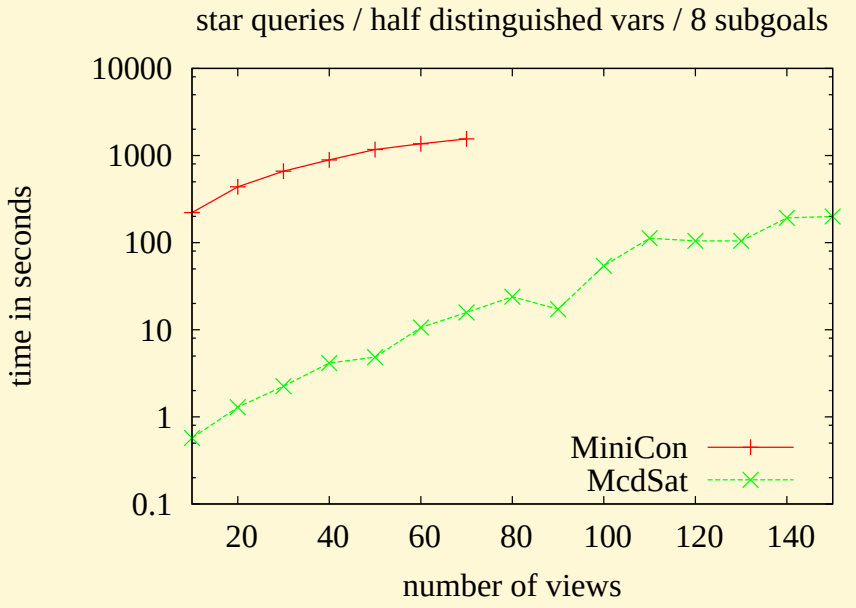
- **MCD Theory: time to generate MCDs (no combination)**
- **Extended Theory: time to generate rewritings**
- Structure: Chain and Star
- Half distinguished variables
- Queries of different length
- Different number of views
- Each point is average over 10 instances
- Random instances created with generator of [Afrati, Li & Ullman 2001]

Experimental Results: MCD Theories

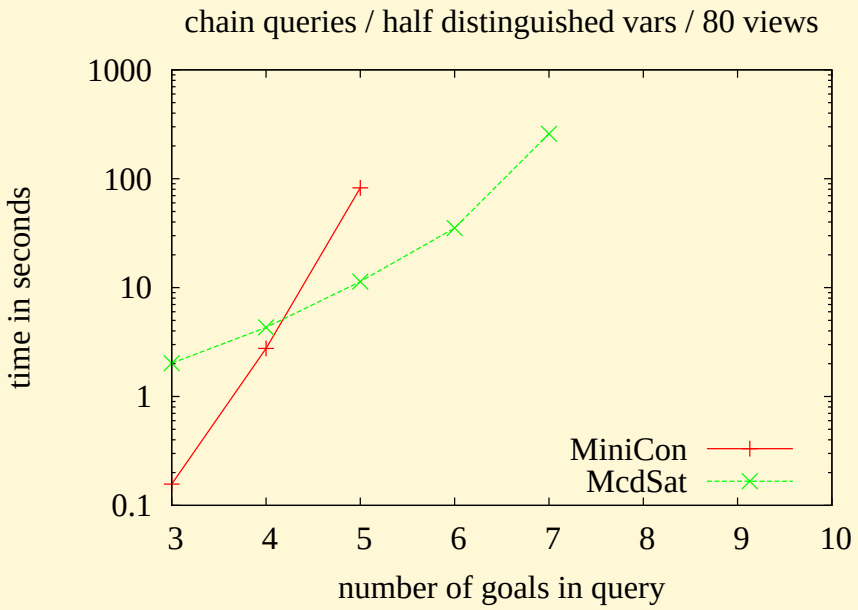


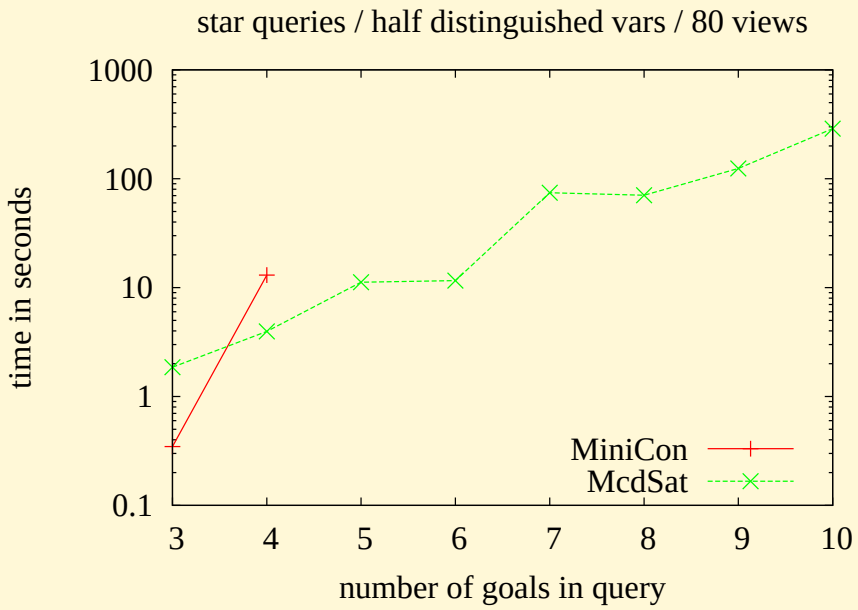


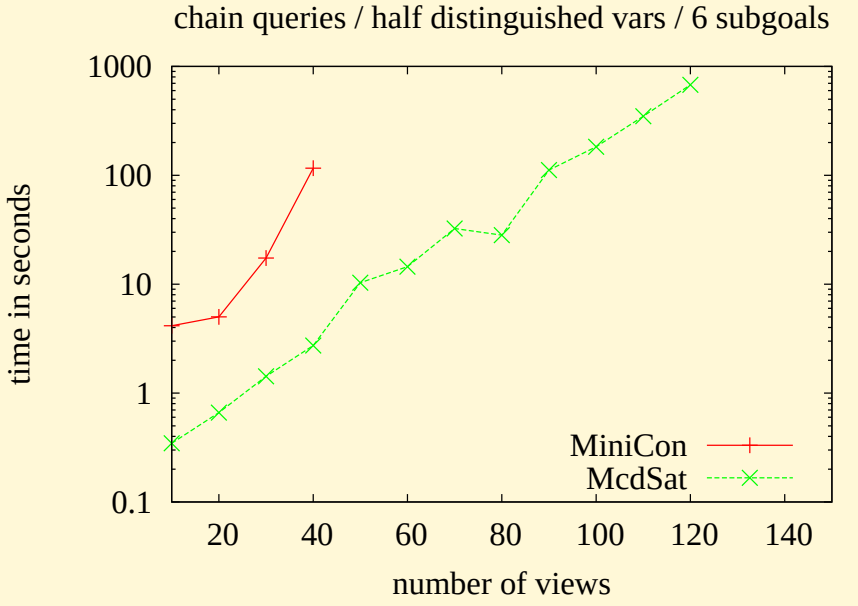


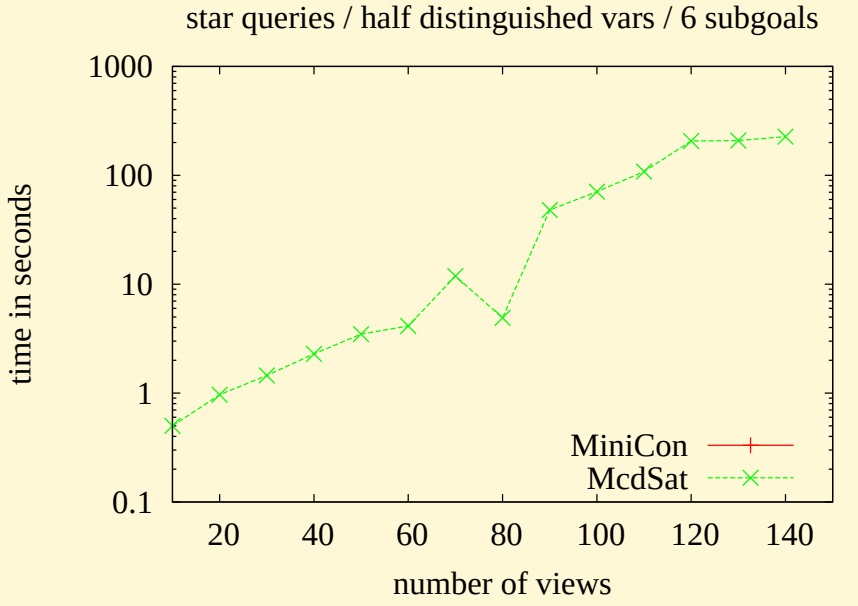


Experimental Results: Extended Theories









Conclusions

- Proposed a novel method for QRPs using propositional logic which:
  - Uses off-the-shelf propositional components
  - It's easy to implement
  - Shows improved performance over other methods
- Thus, the logical approach is **not only of scientific interest** but **practical too!**
- **Similar ideas can be applied to other problems!**