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

**Yolifé Arvelo** and **Blai Bonet** and **María Esther Vidal**
Departamento de Computación
Universidad Simón Bolívar
Caracas, Venezuela
{yarvelo,bonet,mvidal}@ldc.usb.ve

## Abstract

We consider the problem of rewriting a query efficiently using materialized views. In the context of information integration, this problem has received significant attention in the scope of emerging infrastructures such as WWW, Semantic Web, Grid, and P2P which require efficient algorithms. The problem is in general intractable, and the current algorithms do not scale well when the number of views or the size of the query grow. We show however that this problem can be encoded as a propositional theory in CNF such that its models are in correspondence with the rewritings of the query. The theory is then compiled into a normal form, that is called d-DNNF and supports several operations like model counting and enumeration in polynomial time (in the size of the compiled theory), for computing the rewritings. Although this method is also intractable in the general case, it is not necessarily so in all cases. We have developed, along these lines and from off-the-shelf propositional engines, novel algorithms for finding maximally-contained rewritings of the query given the set of accessible resources (views). The algorithms scale much better than the current state-of-the-art algorithm, the MiniCon algorithm, over a large number of benchmarks and show in some cases improvements in performance of a couple orders-of-magnitude.

## Introduction

Emerging Web infrastructures are making available an enormous amount of resources and demand effi cient techniques to query them. To achieve this goal, tasks as locating sources, query rewriting, optimization, and evaluation need to scale up to large spaces of alternatives. In this paper, we consider the problem of rewriting a query using materialized views (Levy *et al.* 1995; Halevy 2001) which consists on the reformulation of the query from the views. In general, this problem is intractable in the worst case and existing algorithms do not scale well even for restricted classes of queries. This problem has received recent attention in the context of data integration (Duschka & Genesereth 1997b; Kwok & Weld 1996; Lambrecht, Kambhampati & Gnanaprakasam 1999), and query optimization and data maintenance (Levy, Rajaraman & Ordille 1996; Zaharioudakis *et al.* 2000; Mitra 2001).

We approach this problem from a logical perspective with the aim to exploit the recent developments in knowledge compilation and SAT which are currently able to handle logical theories with thousands of variables and clauses.

The main idea of our approach is to cast the query rewriting problem into a propositional theory such that the models of the theory constitute the solution for the problem. This theory is then *compiled* into a normal form called d-DNNF that implements model enumeration in polynomial time in the size of the compiled theory. From a complexity point of view, the proposal doesn't appear to be an overkill of the input problem since the number of rewritings is exponential in the worst case, and deciding if one candidate rewriting is valid is an NP-complete problem. Similarly, although model enumeration is intractable in the worst case, it doesn't need to be so for specifi c cases.

Our experimental results show that the logical approach is up to two orders of magnitude faster than the current state-of-the-art algorithms over benchmark problems from the literature. These results makes us believe that our formulation isn't only of theoretical interest but practical too, and encourage us to continue our line of research.

The paper is organized as follows. The next section covers preliminaries defi nitions of the rewriting and containment problems, and the MiniCon algorithm. We then describe the propositional theory and the d-DNNF normal form. Later, we show the experimental results and conclude with a brief discussion that includes a summary and future work.

## Preliminaries

Let $P$ be a set of predicates where each $p \in P$ is of arity $a_p$. We consider databases over $P$ of the form $D = \langle P, T \rangle$ where $T = \{T_p\}_{p \in P}$ is a set of tables, each $T_p$ with $a_p$ columns, that represents the predicates in extensional form. A conjunctive query $Q$ over $P$ is of the form

$$Q(\mathbf{x}) \;:-\; p_1(\mathbf{x}_1),\, p_1(\mathbf{x}_2),\, \ldots,\, p_m(\mathbf{x}_m)\,,$$

where $p_i \in P$, $\mathbf{x}$ is a vector of variables, and each $\mathbf{x}_i$ is a vector of length $a_{p_i}$ made of variables and constants. The result of $Q$ over $D$, denoted as $Q(D)$, is the table with $|\mathbf{x}|$ columns that result of the projection of the relational join $\bowtie \{T_{p_i}\}_{i=1}^m$ over $\mathbf{x}$; i.e. $Q(D) \overset{\text{def}}{=} \sigma_{\mathbf{x}}(\bowtie \{T_{p_1}\}_{i=1}^m)$. The head and body of $Q$ are $Q(\mathbf{x})$ and $\{p_i(\mathbf{x}_i) : 1 \leq i \leq m\}$ respectively, and $Vars(Q)$ are all the variables that appear in the query. A variable $x \in Vars(Q)$ that doesn't appear in the head is called an existential variable. The atoms in the body of $Q$ are also called the (sub)goals of $Q$.

A view $V$ over $D$ is a query over $P$. In the context of data integration, the database $D$ is an idealized description of the information contained across multiple data sources:

each data source $E$ is a table of tuples that is described as a view. Since the data sources are assumed to be maintained independently, they are *incomplete* in the sense that the table $E$ is a subset of the table $V(D)$. Moreover, two identically views $V$ and $V'$, describing different data sources, are typically such that $E_V \neq E_{V'}$ even though $V(D) = V'(D)$.

Thus, given a database $D$, a query $Q$ and a collection of (incomplete) views $E = \langle \{V_i\}_i, \{E_i\}_i \rangle$, we are required to find all the tuples in $Q(D)$ *obtainable* from the views in $E$. Formally, we need to find all the *rewritings*

$$R(\mathbf{x}) \;:\!-\; V_{i_1}(\mathbf{x}_1),\, V_{i_2}(\mathbf{x}_2),\, \ldots,\, V_{i_n}(\mathbf{x}_n)$$

such that $R(E) \subseteq Q(D)$,[1] or equivalently all the maximal-contained rewritings. A rewriting $R$ is contained in $R'$ if $R(E) \subseteq R'(E)$. If $\mathcal{R}$ is a collection of rewritings, the result $\mathcal{R}(E)$ of $\mathcal{R}$ over $E$ is the collection of tuples $\cup_{R\in\mathcal{R}} R(E)$.

A query rewriting problem (QRP) is a tuple $\langle P, Q, \{V_i\}\rangle$ where $P$ is a set of predicates, $Q$ is a query over $P$ and $\{V_i\}$ is a collection of views. We assume *safe* problems in the sense that all variables mentioned in the head of the query (resp. in the head of each view) appear in the body of the query (resp. in the body of each view). Further, we only deal with QRPs without constant symbols nor with arithmetic predicates inside the query and/or the views; e.g. predicates like '$x \leq y$' are forbidden.[2]

A rewriting $R$ is a *valid* rewriting, with respect to a given QRP, if for all databases $D = \langle P, T\rangle$ and (incomplete) extensions $\{E_i\}$, $R(E) \subseteq Q(D)$. A collection $\mathcal{R}$ of valid rewritings is a solution of the QRP if for all databases $D = \langle P, T\rangle$ and extensions $\{E_i\}$, there is no another $\mathcal{R}'$ such that $\mathcal{R}(E) \subset \mathcal{R}'(E) \subseteq Q(D)$. We are interested in finding the minimal rewriting $\mathcal{R}$ for a QRP (see below).

For example, consider a single binary predicate 'arc' used to talk about digraphs such that $\text{arc}(x, y)$ holds iff there is an arc from $x$ to $y$, and the QRP (Ullman 2000):

$$\begin{aligned}
Q(x,z) &\;:\!-\; \text{arc}(x,y),\, \text{arc}(y,z)\,, \\
V_1(x_1,y_1) &\;:\!-\; \text{arc}(x_1,x_1),\, \text{arc}(x_1,y_1)\,, \\
V_2(x_2,y_2) &\;:\!-\; \text{arc}(x_2,y_2)\,.
\end{aligned}$$

It can be shown that the collection of rewritings:

$$\begin{aligned}
R_1(x,z) &\;:\!-\; V_1(x,z)\,, \\
R_2(x,z) &\;:\!-\; V_2(x,y),\, V_2(y,z)\,, \\
R_3(x,z) &\;:\!-\; V_1(x,y),\, V_2(x,z)\,, \\
R_4(x,z) &\;:\!-\; V_1(x,y),\, V_2(y,z)\,, \\
R_5(x,z) &\;:\!-\; V_2(x,z),\, V_1(z,y)\,, \\
R_6(x,z) &\;:\!-\; V_2(x,y),\, V_1(y,z)
\end{aligned}$$

is the minimal rewriting solution of the QRP.

## The Query Containment Problem

An important related problem is to determine whether for two queries $Q$ and $Q'$, $Q(D) \subseteq Q'(D)$ for all databases $D$; the so-called query containment problem. In such case, we write $Q \subseteq Q'$, $Q \subset Q'$ if in addition there is a database $D$ such that $Q(D) \subset Q'(D)$, and finally $Q \cong Q'$ iff $Q \subseteq Q'$ and $Q' \subseteq Q$.

Several methods have been proposed to answer the query containment problem (Chandra & Merlin 1977; Ullman 2000) from which the first one, using containment mappings, is relevant to our work.

Given two queries $Q$ and $Q'$ over $P$, of the form

$$\begin{aligned}
Q(\mathbf{x}) &\;:\!-\; p_1(\mathbf{x}_1),\, p_2(\mathbf{x}_2),\, \ldots,\, p_m(\mathbf{x}_m)\,, \\
Q'(\mathbf{y}) &\;:\!-\; p'_1(\mathbf{y}_1),\, p'_2(\mathbf{y}_2),\, \ldots,\, p'_{m'}(\mathbf{y}_m)\,,
\end{aligned}$$

a *containment mapping* from $Q$ to $Q'$ is a homomorphism $\phi : Vars(Q) \to Vars(Q')$ such that every atom in the body of $\phi(Q)$ appears in the body of $Q'$, and the head of $\phi(Q)$ matches the head of $Q'$. That is, $p_i(\phi(\mathbf{x}_i)) \in Q'$ for all $1 \leq i \leq m$ and $\phi(\mathbf{x}) = \mathbf{y}$. It is known that

**Theorem 1 (Chandra & Merlin 1977)** *$Q' \subseteq Q$ if and only if there is a containment mapping from $Q$ to $Q'$.*

For a given QRP, we say that a valid rewriting $R$ is maximal if there is no another valid rewriting $R'$ such that $R \subset R'$ (when $R$ and $R'$ are considered as queries over $\{V_i\}$). It is then easy to show that the solutions to the QRP are sets of maximal valid rewritings. However, such sets are not unique since for any query $Q$ there are queries $Q'$ equivalent to it (in fact, a infinite number of them). For example, if $R(x) :\!- V(x)$ is a maximal valid rewriting then $R'(x) :\!- V(x), V(x)$ is also maximal. Yet, this is the only type of example, since with incomplete views, if $R \cong R'$ and $R' \neq R$ then either $body(R) \subset body(R')$ or vice versa.

Therefore, we can partially order the class of all solutions as follows: for solutions $\mathcal{R}$ and $\mathcal{R}'$, define $\mathcal{R} \leq \mathcal{R}'$ iff for every $R \in \mathcal{R}$ there is an $R' \in \mathcal{R}'$ such that $body(R) \subseteq body(R')$. Then, it is not hard to show that this partial order has a unique minimal element $\mathcal{R}^*$ called the *minimal solution* in which each $R \in \mathcal{R}^*$ is called a *minimal rewriting*.

The aim of QRP algorithms is to find $\mathcal{R}^*$ by constructing all minimal rewritings. This task however is very difficult to achieve since not much is known about minimal rewritings. Perhaps the best-known general results are

**Theorem 2 (Levy *et al.* 1995)** *If the query and views are such that no constant symbols neither arithmetic predicates appear in the bodies, any minimal rewriting has at most the same number of goals of the query.*

**Theorem 3 (Levy *et al.* 1995)** *To check whether there is a valid rewriting $R$ of $Q$ with at most the same number of goals as $Q$ is an NP-complete problem.*

Above theorems suggest a simple non-deterministic algorithm for finding solutions: generate all possible rewriting candidates $R$ of length less than or equal to the length of the query $Q$, and keep those contained. This algorithm is highly inefficient since there is an exponential number of candidates and testing containment is an expensive task.

## The MiniCon Algorithm

The MiniCon algorithm attempts to find a solution to the QRP by exploiting certain independences in the problem (Pottinger & Halevy 2001). In the example, it can be argued that the first goal of $V_1$ can be used to "cover" the goal $\text{arc}(x, y)$ while $V_2$ can be used to cover the second goal of the query yielding the rewriting $R_3$. In this case, the variables $x, y$ of $Q$ are both mapped into variable $x_1$ of $V_1$, and $y, z$ to $x_2, y_2$ of $V_2$, yielding $x = y = x_1 = x_2$ and $z = y_2$.

[1] Observe that the collection of views and their extensions can be thought as a database and thus $R(E)$ is well defined.

[2] Although constant symbols can be easily accommodated, interpreted predicates are more challenging.

The general idea can be formalized as follows (Pottinger & Halevy 2001). A view $V$ *covers* a subset of goals $C \subseteq body(Q)$ iff there is a pair $\langle h, \phi \rangle$ where $h : Vars(V) \to Vars(V)$ is a *head homomorphism* that is identity on existential variables and satisfies $h(x) = h(h(x))$ for head variables,[3] and $\phi : Vars(Q) \to h(Vars(V))$ is a partial homomorphism such that $\phi(Vars(head(Q))) \subseteq h(Vars(head(V)))$ and if $\phi(x)$ is an existential variable, then all goals $p_i \in body(Q)$ that mention $x$ are in $C$ and $Vars(p_i) \subseteq Dom(\phi)$.[4]

Such coverings, described by tuples $M = \langle V, \phi, h, C \rangle$, are called MiniCon Descriptions (MCDs).[5] For example, the MCD that covers the first goal of $Q$ with the first goal of $V_1$ is $M = \langle V_1, \phi, h, \{\text{arc}(x, y)\} \rangle$ where $h = \{x_1 \to x_1, y_1 \to y_1\}$ and $\phi = \{x \to x_1, y \to x_1\}$. An MCD defines an equivalence relation between the variables in $Q$ given by the non-empty subsets $\phi^{-1}(\{x\})$ for $x \in Ran(h)$. In the example, $\phi^{-1}(\{x_1\}) = \{x, y\}$ and thus $x$ is equivalent to $y$.

MiniCon computes a set $\mathcal{M}$ of MCDs and combine them in order to build valid rewritings. The valid rewritings are associated with combinations $S = \{M_1, \ldots, M_l\} \subseteq \mathcal{M}$ such that $\cup\{C_i : 1 \leq i \leq l\} = body(Q)$ and $C_i \cap C_j = \emptyset$ for all $1 \leq i < j \leq l$.

To describe the combination process, consider one such combination $S$ and the *least-restrictive* equivalence relation that agrees with all the equivalence relations defined by the MCDs in $S$. Define now the mapping $\psi(x)$ equal to one (fixed) representative of the class $[x]$ for $x \in Vars(Q)$, one (fixed) representative of the class containing $\phi_i^{-1}(\{x\})$ if non-empty and $x \in Vars(V_i)$, and $x$ if $x \in Vars(V_i)$ and $\phi^{-1}(\{x\}) = \emptyset$. The rewriting is then

$$R(\psi(\mathbf{x})) \ :- \ V_1(\psi(h(\mathbf{x}_1))), \ldots, V_l(\psi(h(\mathbf{x}_l))) \,.$$

For example, consider the MCD $M$ from above, and $M' = \langle V_2, h', \phi', \{\text{arc}(y, z)\} \rangle$ where $h' = \{x_2 \to x_2, y_2 \to y_2\}$ and $\phi' = \{y \to x_2, z \to y_2\}$. These MCDs induce the equivalence relation $\{\{x, y\}, \{z\}\}$ and thus $\psi$ is:

$$\{x \to x, y \to x, z \to z, x_1 \to x, y_1 \to y_1, x_2 \to x, y_2 \to z\}$$

that yields $R(x, z) :- V_1(x, y_1), V_2(x, z)$ equal to $R_3$.

The MiniCon algorithm is, up to our knowledge, the current state-of-the-art algorithm for computing maximally contained rewritings. The algorithm works in two phases: the first phase computes a set $\mathcal{M}$ of minimal MCDs,[6] and the second phase forms all possible combinations of MCDs into valid rewritings. The second phase, that works greedily by picking up MCDs to cover $Q$, is easily implemented. The first phase, on the other hand, is difficult to implement and we are not aware of any good implementation of it, or even sure on how to implement it efficiently.

[3]This condition makes up a *normal form* for head homomorphism, see the reference.

[4]This is Property 1 from (Pottinger & Halevy 2001).

[5]An MCD is just a description of a containment mapping. We use the term MCD whether such description is minimal, or not. See below for more on minimality.

[6]An MCD $M = \langle V, h, \phi, C \rangle$ is minimal if there is no another MCD $M' = \langle V, h', \phi', C' \rangle$ such that $C' \subset C$, and $h'$ and $\phi'$ are restrictions of $h$ and $\phi$ respectively. The set of minimal MCDs is enough to generate a valid solution for the QRP.

It should be emphasized that although MiniCon aims to generate the minimal solution $\mathcal{R}^*$, there are no guarantees for that. Indeed, it is not hard to design problems where MiniCon generates non-minimal rewritings.

## Propositional Encodings

Our approach is similar to MiniCon's in the sense of first building a set of MCDs and then combining them. The main contribution of this paper is however that we deal with both tasks from a logical perspective. The idea is to build a propositional theory, called the MCD theory, such that its models are in one-to-one correspondence with a set of MCDs *sufficient* to build a solution. The MCDs can then be recovered from the theory and combined into valid rewritings as in MiniCon or, as we will see, the MCD theory can be replicated into an extended theory such that its models are in correspondence with valid rewritings.

We first present the construction of the MCD theories and then the extended theories. For the rest of this section, let us consider a fixed QRP $\mathcal{Q} = \langle P, Q, \{V_i\}_{i=1}^n \rangle$.

### MCD Theories

Let us assume that the query $Q$ has $m$ goals and consider an MCD $\langle V, \phi, h, C \rangle$ for $\mathcal{Q}$. In order to simplify the logical theory, the information contained in $\langle \phi, h \rangle$ can be summarized with a relation $\tau \subseteq Vars(Q) \times Vars(V)$ with the sole restriction that the variables $\tau_x = \{y \in Vars(V) : (x, y) \in \tau\}$ are distinguished.[7] We thus deal only with $\tau$ relations.

The MCD theory for $\mathcal{Q}$, denoted by $T(\mathcal{Q})$, has the following variables:

1. $\{v_0, \ldots, v_n\}$ to indicate that the MCD uses view $V_i$; $v_0$ is used to indicate the *null* MCD,
2. $\{g_1, \ldots, g_m\}$ to indicate the goals in $C$,
3. $\{z_{j,k,i}\}$ to indicate that the $j$th goal in $Q$ is covered by the $k$th goal in $V_i$, and
4. $\{t_{x,y}\}$ to indicate that $(x, y) \in \tau$.

The range of indices for the $z$ variables depend on $\mathcal{Q}$; e.g. if $K$ are the goals in $V_i$ of the same type as goal $g_j$, then $k$ ranges in $K$ for $z_{j,k,i}$. For $t$ variables, $x$ ranges in the number of variables in $Q$ and $y$ ranges in the maximum number of variables in the views.

The theory $T(\mathcal{Q})$ contains the following clauses:

#### Covering Clauses

C1. (At least of view is used): $\bigvee_{i=0}^{n} v_i$

C2. (At most one view is used): $v_i \Rightarrow \neg v_j$ for $0 \leq i, j \leq n$,

C3. (Null view equals null): $v_0 \Rightarrow \neg g_j$ for $1 \leq j \leq m$,

C4. (Views are useful): $v_i \Rightarrow \bigvee_{j=1}^{m} g_j$ for $1 \leq i \leq n$,

C5. (Subgoals are covered at most once): $z_{j,k,i} \Rightarrow \neg z_{j,l,i}$ for appropriate $i, j, k, l$ with $k \neq l$,

C6. (Scope of views): $v_i \Rightarrow \neg g_j$ for such goals that can't be covered by $V_i$,

C7. (Consistency): $v_i \wedge g_j \Leftrightarrow \bigvee z_{j,k,i}$ for appropriate $i, j, k$.

[7]Indeed, from $\langle \phi, h \rangle$ construct $\tau$ by letting $(x, y) \in \tau$ iff $\phi(x) = h(y)$. On the other hand, from such $\tau$ form the least-restrictive equivalence relation such that for all $x$, $\tau_x$ is contained in an equivalence class. Then, choose a representative from each class and define $h(y)$ equal to it, and $\phi(x)$ equal to the representative of the class containing $\tau_x$.

These clauses follow directly from the definition of the variables. Note however that C7 is a double implication and thus there isn't a model satisfying $z_{j,k,i}$ and $z_{j',k',i'}$ with $i \neq i'$ since then $v_i$ and $v_{i'}$ must hold in contradiction with C2.

**Mapping Clauses**

C8. (Dead variables): $v_i \Rightarrow \neg t_{x,y}$ for all $x, y$ with $y \notin V_i$,

C9. (Head homomorphism): $v_i \wedge t_{x,y} \Rightarrow \neg t_{x,y'}$ for all existential variables $y, y' \in V_i$,

C10. (Distinguished): $v_i \Rightarrow \neg t_{x,y}$ for all distinguished $x \in Q$ and existential $y \in V_i$,

C11. (Existential): $v_i \wedge t_{x,y} \Rightarrow g_j$ for all existential $y \in V_i$ and goals $j$ that contain existential $x \in Q$,

C12. (Matching): $v_i \wedge z_{j,k,i} \Rightarrow t_{x,y}$ for all $x$ and $y$ that must match if goal $j$ in $Q$ is covered by goal $k$ in $V_i$.

Clauses C10 say that $\tau_x$ must be a subset of distinguished variables, and C11 that if $x$ is mapped into an existential variable, then all goals containing $x$ must also be mapped. Clauses C12, on the other hand, make the mapping.

**Minimization Clauses**

C13. (1-1 on $\exists$ vars): $v_i \wedge t_{x,y} \Rightarrow \neg t_{x',y}$ for $x, x' \in Q, y \in V_i$ and $x$ is existential,

C14. (If the view has no existential variables, the MCD covers at most one goal): $v_i \wedge g_j \Rightarrow \neg g_k$.

Strictly speaking, clauses C13–C14 are not necessary for the soundness and completeness of the theory, yet they often reduce the number of MCDs generated. This reduction is due to the fact that if $\langle V, h, \phi, C\rangle$ is a minimal MCD such that $V$ is a view with only distinguished variables, then $|C| = 1$.

The theory $T(\mathcal{Q})$ is the collection of all clauses generated by rules C1–C14. A model $\omega$ for $T(\mathcal{Q})$ either satisfies $v_0$ and is called *null*, or defines an MCD $M_\omega = \langle V, \tau, C\rangle$ where $V = V_i$ if $\omega \models v_i$, $C = \{g_j : \omega \models g_j\}$ and $\tau = \{(x, y) : \omega \models t_{x,y}\}$. It is not hard to show

**Theorem 4** *Let $\mathcal{Q}$ be a QRP. If $M$ is a minimal MCD for $\mathcal{Q}$, then there is $\omega \in Models(T(\mathcal{Q}))$ such that $M = M_\omega$. If $\omega \in Models(T(\mathcal{Q}))$, then $\omega$ is null or $M_\omega$ is an MCD.*

The structure of a query is an undirected graph with nodes equal to the goals of the query and edges between goals that share common variables. Chain (resp. star) queries are queries with a chain (resp. star) structure. A chain query is called a 2-chain if the query has exactly two distinguished variables that appear at the extremes of the chain. A 2-chain (resp. chain/star) QRP is a QRP where the query and all views are 2-chain (resp. chain/star).

**Theorem 5** *Let $\mathcal{Q}$ be a 2-chain QRP. If $T(\mathcal{Q}) \models \omega$ then $\omega$ is null or $M_\omega$ is a minimal MCD for $\mathcal{Q}$. Thus, for 2-chains, the set of MCDs associated to $T(\mathcal{Q})$ coincides with those computed by MiniCon.*

## Extended Theories

Theorem 2 tells us that a minimal rewriting has at most the same number of goals as the query and then, since each MCD covers at least one goal, only rewritings with at most that number of MCDs should be considered. Therefore, the extended theory is built by conjoining $m$ copies of the MCD theory, indexed with superscripts $t$ denoting the $t$-th copy, with the clauses:

C15. (Cover all goals): $\bigvee_{t=1}^{m} g_j^t$ for $1 \leq j \leq m$,

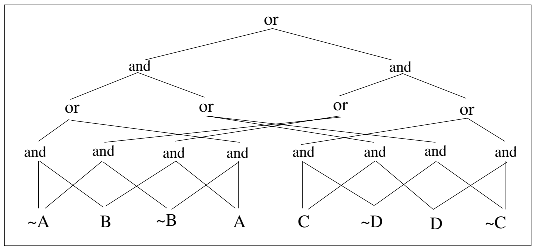

Figure 1: A decomposable and deterministic NNF.

C16. (Disjunctive covering): $g_j^t \Rightarrow \neg g_j^s$ for $1 \leq s \neq t \leq m$,

C17. (Symmetries): $g_i^t \Rightarrow \bigvee_{j=1}^{i-1} g_j^{t-1}$ for $1 \leq i, t \leq m$.

Clauses C15 say that all goals must be covered, C16 that each goal is covered at most once, and C17 break symmetries by enforcing an order upon the copies of the theories.

## Decomposable Negation Normal Form

Knowledge compilation is the area in AI concerned with the problem of mapping logical theories into suitable fragments that make certain desired operations tractable (Selman & Kautz 1996; Cadoli & Donini 1997). For example, propositional theories can be compiled into Ordered Binary Decision Diagrams (OBDDs) making a large number of operations tractable (Bryant 1992). A more recent compilation language is Decomposable Negation Normal Form (DNNF) (Darwiche 2001). DNNFs support a rich set of polynomial-time operations, are more succinct than OBDDs (Darwiche & Marquis 2002), and have been recently used in non-deterministic planning (Palacios *et al.* 2005).

A propositional theory is in Negation Normal Form (NNF) if it is constructed from literals using only conjunctions and disjunctions (Barwise 1977). An NNF can be represented as a directed acyclic graph in which leaves are labeled with literals and internal nodes are labeled with $\wedge$ and $\vee$; see Fig. 1 for an example.

An NNF is decomposable (DNNF) (Darwiche 2001) if for each conjunction $\bigwedge_i \phi_i$, the set of variables in each conjunct are pairwise disjoint; i.e. $Vars(\phi_i) \cap Vars(\phi_j) = \emptyset$ for $i < j$. A DNNF is deterministic (d-DNNF) (Darwiche 2001) if for each disjunction $\bigvee_i \phi_i$, the disjuncts are pairwise logically contradictory; i.e. $\phi_i \Rightarrow \neg\phi_j$ for $i < j$. The NNF in Fig. 1 is decomposable and deterministic. DNNFs support a number of operations, including model counting, in polynomial time; d-DNNFs support additional operations.

We use d-DNNFs to compute the MCDs and rewritings associated to the propositional theories. The d-DNNFs are obtained from the CNF theories using a general and publicly-available CNF to d-DNNF compiler. The compiler's algorithm is similar to the DPLL algorithm for SAT but enhanced with advanced techniques such as clause learning and caching.

## Experiments

We evaluated the proposed method, called MCDSAT, and the MiniCon algorithm over a large benchmark that includes problems of different sizes and structures. We didn't consider other algorithms like the bucket algorithm (Levy, Rajaraman & Ordille 1996) or the inverse-rules algorithm

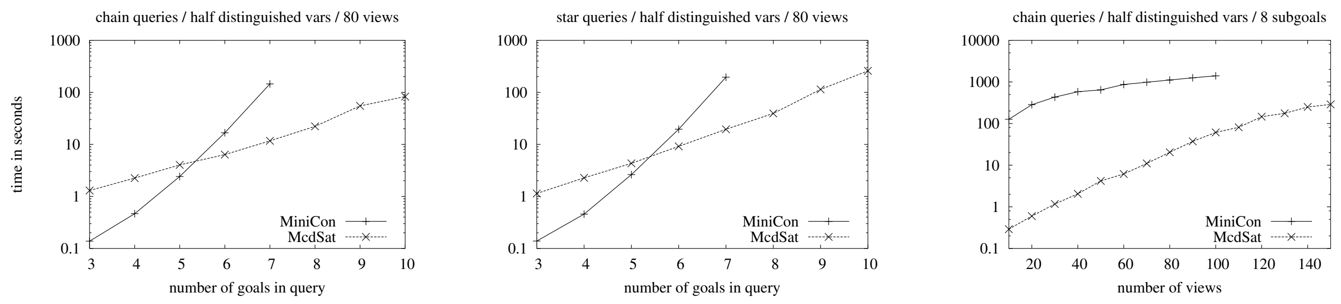

Figure 2: Compilation times of MCD theories for MCDSAT versus the time to generate MCDs for MiniCon. Each data point is the average over 10 runs. Time is in seconds. Missing points refer to early termination due the resource bounds.

| | CNF | | d-DNNF | |
|---|---|---|---|---|
| # goals | # vars | # clauses | # nodes | # edges |
| 3 | 185.2 | 4,575.3 | 286.0 | 677.8 |
| 5 | 369.8 | 6,818.5 | 651.6 | 3,884.3 |
| 7 | 643.7 | 10,377.8 | 1,165.4 | 12,021.0 |
| 3 | 555.6 | 13,742.9 | 534.3 | 1,518.2 |
| 5 | 1,849.0 | 34,164.5 | 1,881.7 | 13,980.3 |
| 7 | 4,505.9 | 72,835.6 | 5,841.9 | 116,858.9 |

Table 1: Average sizes of MCD (top) and extended (bottom) theories for chain queries with half variables distinguished.

(Qian 1996; Duschka & Genesereth 1997a) since (Pottinger & Halevy 2001) shows MiniCon to be superior.

Two experiments were run for each benchmark. The first one measured the time to compile the MCD theory and the time to generate the MCDs for MCDSAT, and the time to generate the MCDs for MiniCon. The second one measured the time to compile the extended theory for MCDSAT, and the time to compute the rewritings for MiniCon.

We considered problems with different structures: 2-chains, chains with all/half variables distinguished, stars with half/all variables distinguished, and random.

Some of these problems have exponential number of MCDs, rewritings or both. One clear advantage of the logical approach over MiniCon is that, once the theory is compiled, one can *repeatedly* ask for a fixed number of MCDs or rewritings in linear time, whereas MiniCon must start from scratch each time. Thus, the logical theories can be thought as *compact repositories* of MCDs or rewritings.

## Implementation

MCDSAT translates QRPs, specified in a suitable format, into a propositional formula in CNF which is then compiled into d-DNNF using the publicly-available c2d compiler (Darwiche 2004).[8] For MiniCon, we made an implementation in Java following the description in (Pottinger & Halevy 2001) and some advice from R. Pottinger. Additionally, we used the implementation of the first phase of MiniCon of (Afrati, Li & Ullman 2001).

## Overview of Results

For lack of space, we focus on chain and star problems either with 80 views and different number of subgoals, or with a query with 8/6 subgoals and a different number of views. These are good representative of other results; the complete benchmark contains others type of problems with different number of subgoals and views. The experiments were run in a cluster of 2GHz AMD processors each with 1Gb of RAM. Resource bounds of 30 minutes of time and 1Gb of memory were enforced in each experiment.

Fig. 2 shows the results for the first experiment. The curves are plotted in logarithmic scale where each point stands for the average compilation time of 10 experiments. As shown, MiniCon outperforms MCDSAT in the smaller instances but, as the number of goals in the query increases, MCDSAT shows an improved performance over MiniCon. Some of the improvement can be attributed to a better implementation of the SAT engines, yet that couldn't explain the exponential gap depicted. Also note that for bigger instances, MiniCon isn't able to finish within the given bounds. Table 1 shows the sizes of the CNF and compiled formulae for the propositional theories for chain queries with half variables distinguished.

The results for the second experiments are shown in Fig. 3. The results are similar to the first experiment where MCDSAT outperforms MiniCon on the bigger instances. The results for other structures and number of views are consistently similar to these curves.

Finally, we did an experiment aimed to harm MCDSAT with a problem that have an exponential number of non-minimal MCDs, all that appear as models of the theory. The QRP is:

$$\begin{aligned} Q(x,y) &:- \ p_1(x,z_1),\ q_1(z_1,y),\ldots,p_k(x,z_k),\ q_k(z_k,y)\,, \\ V_1(x,y) &:- \ p_1(x,z_1),\ q_1(z_1,y)\,, \\ &\ \vdots \\ V_k(x,y) &:- \ p_1(x,z_1),\ q_1(z_1,y),\ldots,p_k(x,z_k),\ q_k(z_k,y)\,. \end{aligned}$$

It can be shown that MiniCon computes only minimal MCDs whose number is $k(k+1)/2$ while MCDSAT computes all MCDs whose number is $2^{k+1}-k-2$. However, even in this extreme example, MCDSAT is able to generate the MCDs faster than MiniCon; see Table 2.

## Conclusions and Future Work

We have proposed a novel method for solving QRPs using propositional theories. The approach, grounded on the idea of MCDs from MiniCon, consists of building a theory such

[8]MCDSAT will be available in our Web page; c2d can be found at `http://reasoning.cs.ucla.edu/c2d`.

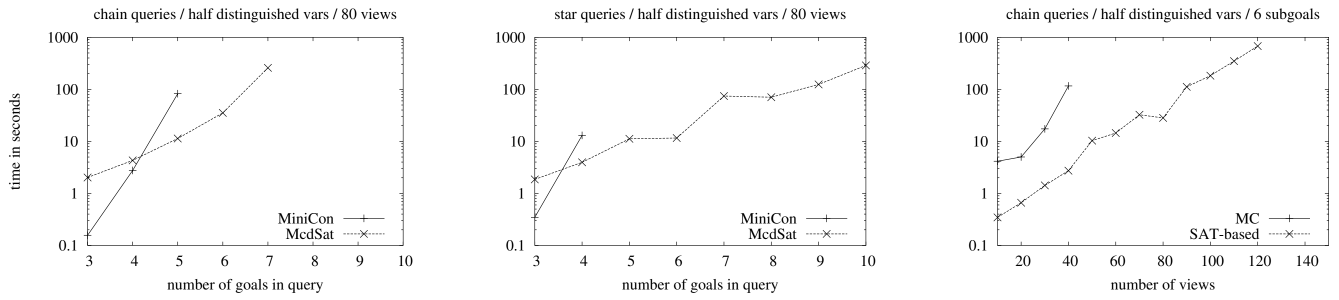

Figure 3: Compilation times of extended theories for MCDSAT versus the time to generate valid rewritings for MiniCon. Each data point is the average over 10 runs. Time is in seconds. Missing points refer to early termination due the resource bounds.

| | MiniCon | | MCDSAT | |
|---|---|---|---|---|
| $k$ | # MCDs | time | # MCDs | time |
| 4 | 10 | 0.6 | 26 | 0.0 |
| 6 | 21 | 1.6 | 120 | 0.1 |
| 8 | 36 | 11.2 | 502 | 0.2 |
| 10 | 55 | 80.1 | 2,036 | 0.4 |

Table 2: Problem with exponential # of non-minimal MCDs.

that its models are in correspondence with the MCDs from which a solution can be built.

The experimental results show a couple of order-of-magnitude improvement on time for the generation of the MCDs over MiniCon. These MCDs can be combined as in MiniCon, or with an extended logical theory for which its models are in correspondence with the rewritings.

We think that these theories can be further extended into templates or schemata for answering queries of certain form. For example, by using additional propositional variables, a general logical theory for answering chain queries of length at most $N$ can be constructed and compiled. Thus, in presence of such a query, certain propositional variables are instantiated so that the resulting models constitute a solution. Such general theories would be bigger and thus harder to compile, yet once compiled they would provide rewritings in polynomial time in the size of the compiled theory, and thus the preprocessing (initial compilation) can be amortized over multiple runs. We also think that other Databases and Data Integration problems can be addressed from a logical perspective. We plan to work along these lines.

**Acknowledgements:** We thank C. Li for providing his MiniCon code, R. Pottinger for her advice on the implementation of MiniCon, A. Darwiche for the c2d compiler, and the Laboratorio de Computación de Alto Rendimiento for the use of the CAR cluster. B. Bonet and M. E. Vidal are partially funded by USB-DID grants.

## References

Afrati, F.; Li, C.; and Ullman, J. Generating efficient plans for queries using views. In *SIGMOD Conference*, 2001.

Barwise, J., ed. 1977. *Handbook of Mathematical Logic*. North-Holland.

Bryant, R. E. 1992. Symbolic boolean manipulation with ordered binary-decision diagrams. *ACM Comp. Surveys* 24(3):293–318.

Cadoli, M., and Donini, F. 1997. A survey on knowledge compilation. *AI Communications* 10(3-4):137–150.

Chandra, A. K., and Merlin P. M. 1977. Optimal implementation of conjunctive queries in relational databases. In *Proc. ACM STOCS*, 77–90.

Darwiche, A., and Marquis, P. 2002. A knowledge compilation map. *J. Artif. Intell. Res.* 17:229–264.

Darwiche, A. 2001. Decomposable negation normal form. *J. of the ACM* 48(4):608–647.

Darwiche, A. 2001. On the tractable counting of theory models and its applications to belief revision and truth maintenance. *J. of App. Non-Classical Logics* 11:11–34.

Darwiche, A. 2004. New advances in compiling cnf into decomposable negation normal form. In *Proc. ECAI*, 328–332.

Duschka, O., and Genesereth M. 1997. Answering recursive queries using views. In *Proc. PODS*, 109–116.

Duschka, O., and Genesereth M. 1997. Query planning in infomaster. In *Proc. ACM Symp. on App. Comp.*, 109–111.

Halevy, A. 2001. Answering queries using views: A survey. *VLDB Journal* 10(4):270–294.

Kwok, C., and Weld, D. 1996. Planning to gather information. In *AAAI*, 32–39.

Lambrecht, E.; Kambhampati S.; Gnanaprakasam S. 1999. Optimizing recursive information gathering plans. In *IJCAI*, 1204–1211.

Levy, A.; Mendelzon, A.; Sagiv, Y.; and Srivastava, D. 1995. Answering queries using views. In *Proc. of PODS*, 95–104.

Levy, A.; Rajaraman, A.; and Ordille, J. 1996. Query-answering algorithms for information agents. In *Proc. AAAI*, 40–47.

Mitra,P. 2001. An Algorithm for Answering Queries Efficiently Using Views. In *Proc. Australasian Database Conf.*, 92–99.

Qian, X. 1996. Query folding. In *Proc. ICDE*, 48–55.

Palacios, H.; Bonet, B.; Darwiche, A.; and Geffner, H. 2005. Pruning conformant plans by counting models on compiled d-DNNF representations. In *Proc. ICAPS*, 141–150.

Pottinger, R., and Halevy, A. MiniCon: A scalable algorithm for answering queries using views. *VLDB Journal* 10:182–198, 2001.

Selman, B., and Kautz, H. 1996. Knowledge compilation and theory approximation. *J. of the ACM* 43:193–224.

Ullman, J. 2000. Information integration using logical views. *Theoretical Computer Science*, 239(2):189–210.

Zaharioudakis, M.; Cochrane, R.; Lapis, G.; Pirahesh, H.; and Urata, M. 2000. Answering complex SQL queries using automatic summary tables. *SIGMOD*, 105–116.