Exploiting Irrelevance Reasoning to Guide Problem Solving

Alon Y. Levy*
Dept. of Computer Science
Stanford University
Stanford, California 94305
(levy@cs.stanford.edu)

Yehoshua Sagiv
Dept. of Computer Science
Hebrew University
Jerusalem, Israel
(sagiv@cs.huji.ac.il)

Abstract

Identifying that parts of a knowledge base (KB) are irrelevant to a specific query is a powerful method of controlling search during problem solving. However, finding methods of such *irrelevance reasoning* and analyzing their utility are open problems. We present a framework based on a proof-theoretic analysis of irrelevance that enables us to address these problems. Within the framework, we focus on a class of *strong-irrelevance* claims and show that they have several desirable properties. For example, in the context of Horn-rule theories, we show that strong-irrelevance claims can be derived efficiently either by examining the KB or as logical consequences of other strong-irrelevance claims. An important aspect is that our algorithms reason about irrelevance using only a small part of the KB. Consequently, the reasoning is efficient and the derived irrelevance claims are independent of changes to other parts of the KB.

1 Introduction

Control of reasoning is a major issue in scaling up problem solvers that use declarative representations, since inference is slowed down significantly as the size of the knowledge base (KB) is increased. A key factor for the slow down is the search of the inference engine through parts of the KB that are *irrelevant* to the query at hand. Moreover, since a KB is designed for a variety of tasks, it is often at a level of detail that is too refined for a specific query. Often, we have additional knowledge about the domain or about the KB that can be used to cut down drastically the space that the inference engine has to search. One important type of such knowledge consists of *irrelevance claims* stating that certain formulas are redundant with respect to, or will not be part of any derivation of a given class of queries, and consequently, those formulas can be removed. Such irrelevance claims can be either given by the user or derived automatically by the system.

* Supported by a fellowship from Shell Oil Company.

Effective use of irrelevance reasoning requires a formal understanding of the issues involved in such reasoning, as has been done in the context of probabilistic reasoning [Pearl, 1988]. The work of [Subramanian, 1989] presented a framework for stating irrelevance claims, and raised several issues concerning irrelevance reasoning. However, two issues remain largely open. The first is to find efficient methods for automatically deriving irrelevance claims. The second issue is to determine the utility of removing irrelevant knowledge, since removing irrelevant parts of a KB does not always improve efficiency. For example, redundant formulas (which may be considered irrelevant) can often speedup problem solvers

To address these issues, we first present a space of definitions of irrelevance, based on a proof-theoretic analysis of the notion. This space enables us to make finer dis Unctions than those possible in the framework of [Subramanian, 1989]. The main distinction we make is between *weak-irrelevance* claims and *strong-irrelevance* claims. Roughly, a formula is strongly irrelevant to a query if it cannot appear in any of its derivations, whereas it is weakly irrelevant if it does not appear in some of its derivations. Strong-irrelevance claims turn out to have some desirable properties. For example, in many cases it is possible to find efficiently formulas in the KB that are strongly irrelevant to a given query. In some cases it is even possible to find *all* strongly-irrelevant formulas. Furthermore, unlike weak irrelevance, removing strongly-irrelevant formulas from a KB may only improve the performance (and sometimes the improvement is by orders-of-magnitude, as we will show). We investigate strong irrelevance in detail for Horn-rule KBs and describe novel algorithms for efficiently deriving new strong-irrelevance claims from those given by the user.

Our algorithms consider, in addition to the rules of the KB, only constraints on the ground facts that may possibly appear (e.g., order constrr'nts, sorts), as opposed to looking at the ground facts themselves. Consequently, if the ground facts change, the irrelevance claims still hold and, therefore, the cost of irrelevance reasoning is amortized over many queries. The main difficulty in irrelevance reasoning is finding properties satisfied by all possible derivations of a given query. To do so, we use a powerful tool, the *query tree,* first introduced in [Levy and Sagiv, 1992]. The query tree encodes finitely all possible derivations of the query (even when rules are recursive). The query tree facilitates automatic derivation of irrelevance claims that follow from an examination of the KB and irrelevance claims supplied by the user.

2 Formalizing Irrelevance

2.1 Preliminaries

We consider a knowledge base (KB) of closed formulas Δ in first-order predicate calculus. We assume that the inference mechanism of the KB employs a set of inference rules $\mathcal{S}$. A derivation D of a closed formula ψ from Δ is a sequence of formulas, $\alpha_1, \ldots, \alpha_n$, such that $\alpha_n = \psi$ and for each i $(1 \leq i \leq n)$, either $\alpha_i \in \Delta$, α_i is a logical axiom, or α_i is the result of applying a rule in $\mathcal{S}$ to some elements $\alpha_{i_1}, \ldots, \alpha_{i_l}$ that appear prior to α_i. The formulas $\alpha_{i_1}, \ldots, \alpha_{i_l}$ are said to be *immediate subgoals* of α_i. The set of formulas in D that do not have any subgoal is called the *base* of the derivation, denoted by $Base(D)$. The set $Base(D)$ represents a "support set" for ψ. We consider only derivations in which every α_i is a subgoal of ψ (not necessarily an immediate subgoal).

A *query* is represented by a formula ψ. If ψ is a closed formula (i.e., has no free variables), then the answer is **true** if the the inference mechanism can find some derivation of ψ from Δ, and **false** otherwise.[1] If ψ contains free variables, the answer is the set of assignments for the free variables, such that the resulting closed formulas are derivable from Δ;[2] in this case, a derivation is a set containing a single derivation for each answer. A query may have several derivations from a given knowledge base, and we denote the set of those derivations by $\mathcal{D}_\Delta(\psi)$ (note that if ψ has free variables, then $\mathcal{D}_\Delta(\psi)$ is a set of sets of derivations).

2.2 Definitions of Irrelevance

Our goal is to express and reason with irrelevance claims of the form "*X is irrelevant to ψ with respect to the theory Δ*." and so, we need to give such claims a formal definition. X is called the *subject* of the irrelevance claim. In this paper, we discuss the case in which X is a formula or set of formulas. Other irrelevance subjects such as objects, predicates and distinctions between predicates are discussed in [Levy, 1993].

In order for a definition of irrelevance to be useful, it should give us information that could be used effectively. For example, we would like to know whether irrelevance claims can be derived, how the claims change when the KB changes, and what is the utility of removing irrelevant facts. Unfortunately, there is no single best definition of irrelevance that both matches our intuition and enables us to establish such properties. For example, we can define a formula ϕ to be irrelevant to ψ if there is some derivation of ψ that does not contain ϕ, or alternatively, we can require that *no* derivation of ψ contains ϕ. To make the needed distinctions, we present a space of definitions of irrelevance and investigate the different properties of various definitions within this space.

It should be noted that our analysis is not an attempt to formalize the common sense notion of irrelevance or argue for properties of such a notion (as done, for example, by [Gardenfors, 1978]). Our goal is to utilize the notion of irrelevance to speed inference and, therefore, we analyze the ways in which it can arise in inference. Specifically, we analyze irrelevance in a proof-theoretic setting by considering the possible derivations (or more generally, paths) that an inference mechanism can pursue in answering a query. In contrast, the analysis of [Subramanian, 1989] is *meta-theoretic,* i.e., it considers only the formulas in the KB, not the possible derivations of the query. Consequently, we are able to make finer distinctions than those made in Subramanian's framework.

Definitions in the space vary along two axes. The first considers different ways of defining irrelevance of a subject ϕ with respect to a *single* derivation D of the query ψ. We capture this by a definition of *derivation irrelevance*, denoted by $DI(\phi, D)$. The following are a few examples of how DI can be defined:

- $DI_1(\phi, D)$ iff $\phi \notin Base(D)$.
- $DI_2(\phi, D)$ iff $\phi \notin D$.
- $DI_3(\phi, D)$ iff $Base(D) \not\models \phi$.
- $DI_4(\phi, D)$ iff $Base(D) \not\models \phi, \neg\phi$.

Definition DI_1 requires ϕ not to be in the support set of D. Definition DI_2 is stronger and requires ϕ not to appear in D. Definition DI_3 is even stronger and requires that ϕ not be a logical consequence of the formulas in D, while DI_4 requires the same also for $\neg\phi$. Note that ϕ is not necessarily a formula in the KB.

Requiring that DI holds for all possible derivations of the query may be too restrictive. Therefore, in the second axis we examine different restrictions on the set of derivations we consider. For example, we can require DI to hold for the set of *minimal* derivations[3] of the query, or for the set all derivations bounded by some resource constraint. Alternatively, we can require DI to hold for *some* derivation in the chosen set. For example, we can require that DI hold for some minimal derivation of the query.[4] Given a predicate DI and a set of derivations, a definition of irrelevance in our space is as follows:

Definition 2.1: Let Δ be a KB, ϕ be a closed formula, ψ be a query, and $\mathcal{D}_0$ be a subset of the set $\mathcal{D}_\Delta(\psi)$ of all possible derivations of ψ. Let $DI(\tau, D)$ be a condition specifying when a formula τ is irrelevant with respect to a derivation D.
The formula ϕ is said to be *weakly irrelevant* to ψ with respect to Δ, DI and $\mathcal{D}_0$, denoted by $WI(\phi, \psi, \Delta, DI, \mathcal{D}_0)$, if $DI(\phi, D)$ holds for some $D \in \mathcal{D}_0$.

[1] We can return unknown if neither ϕ nor $\neg\phi$ are derivable. However that does not affect our discussion.

[2] An alternative definition often considered is finding *one* variable binding that satisfies the query formula. However, this distinction does not affect our discussion.

[3] Given some criteria of minimality for derivations. See Section 3 for a definition of minimality in the case of Horn-rule KBs.

[4] We can also consider other ways of quantifying over a set of derivations, such as requiring that DI holds for some percent of the derivations. In this paper, however, we consider only universal and existential quantifications.

The formula ϕ is said to be *strongly irrelevant* to ψ with respect to Δ, DI and $\mathcal{D}_0$, denoted by $SI(\phi, \psi, \Delta, DI, \mathcal{D}_0)$, if $DI(\phi, D)$ holds for every $D \in \mathcal{D}_0$. If $\mathcal{D}_\Delta(\psi)$ is empty (i.e., ψ is not derivable from Δ), then ϕ is both weakly and strongly irrelevant to ψ. ∎

We often refer to irrelevance of a set of formulas or to irrelevance of formulas with respect to a set of KBs. To define irrelevance of a set of formulas, we extend the definition of DI as follows. If Φ is a set of formulas, we define $DI(\Phi, D)$ to hold if $DI(\phi_i, D)$ holds for every $\phi_i \in \Phi$. To define irrelevance for a set of knowledge bases Σ, we define $WI(\phi, \psi, \Sigma, DI, \mathcal{D}_0)$ to hold if $WI(\phi, \psi, \Delta, DI, \mathcal{D}_0)$ holds for every $\Delta \in \Sigma$, and similarly for SI.[5] The following example illustrates the various definitions.

Example 2.2: Let $\psi = CanTA(Fred, 101)$ be the query and Δ_0 be the following knowledge base:

$r_1 : AttendClass(x, y) \Rightarrow Pass(x, y)$.
$r_2 : PassExam(x, y) \Rightarrow Pass(x, y)$.
$r_3 : Pass(x, y) \wedge TookGradCourse(x) \Rightarrow CanTA(x, y)$.
$r_4 : Pass(x, y) \wedge (y \geq 200) \Rightarrow TookGradCourse(x)$.
$g_1 : AttendClass(Fred, 101)$. $g_2 : PassExam(Fred, 101)$.
$g_3 : PassExam(Fred, 202)$. $g_4 : PassExam(Fred, 161)$.

Note that $WI(g_2, \psi, \Delta_0, DI_3, \mathcal{D}_\Delta(\psi))$ holds (i.e., g_2 is weakly irrelevant to ψ), because only one of g_1 and g_2 is needed to prove ψ. Atom g_4 is strongly irrelevant to ψ (i.e., $SI(g_4, \psi, \Delta_0, DI_3, \mathcal{D}_\Delta(\psi))$ holds), since it is not used in any derivation of ψ. $CanTA(Fred, 202)$ is strongly irrelevant to ψ if DI_2 is used for DI (i.e., $SI(CanTA(Fred, 202), \psi, \Delta_0, DI_2, \mathcal{D}_\Delta(\psi))$ holds), but not if DI_3 is used instead of DI_2. ∎

The space encompasses several definitions of irrelevance and related notions discussed in the past. The definitions of [Subramanian, 1989] are instances of weak irrelevance; in particular, although her definitions are not in terms of derivations, the main definition of [Subramanian, 1989] is equivalent to $WI(\phi, \psi, \Delta, DI_3, \mathcal{D}_\Delta(\psi))$.[6] The definition of update independence in [Elkan, 1990] is equivalent to $WI(\phi, \psi, \Delta, DI_1, \mathcal{D}_\Delta(\psi))$ (see [Levy and Sagiv, 1993]). The definition of irrelevance given in [Srivastava and Ramakrishnan, 1992] is equivalent to $SI(\phi, \psi, \Delta, DI_1, \mathcal{D}_\Delta(\psi))$. In [Levy and Sagiv, 1992], we discuss strong irrelevance with respect to the set $\mathcal{D}_0$ of minimal derivations.

Several properties can be shown for classes of definitions in the space, mostly distinguishing between weak and strong irrelevance. Here, we mention only two of them. For full details, see [Levy, 1993].

Observation 2.3: Closure under union. When a system needs to determine whether it can use all the irrelevance claims it has, or whether using certain ones will falsify others, it needs to know whether they are closed under union of their subjects. Weak-irrelevance claims do not add up in general, but for strong-irrelevance claims we have a sufficient condition for adding up that depends on DI. Specifically, if the following holds for any derivation D and sets Φ_1, Φ_2 (as, for example, in the case of DI_1–DI_4),

$$DI(\Phi_1, D) \wedge DI(\Phi_2, D) \Rightarrow DI(\{\Phi_1, \Phi_2\}, D)$$

then for any choice of $\mathcal{D}_0$ and Σ,
$SI(\Phi_1, \psi, \Sigma, DI, \mathcal{D}_0) \wedge SI(\Phi_2, \psi, \Sigma, DI, \mathcal{D}_0) \Rightarrow$
$SI(\{\Phi_1, \Phi_2\}, \psi, \Sigma, DI, \mathcal{D}_0)$.

Observation 2.4: Problem-solving payoff. The utility of removing irrelevant formulas is an important question. For weak irrelevance, this is a subtle issue. In fact, explanation-based learning systems do exactly the opposite by adding redundant rules, which are weakly irrelevant. In Example 2.2, adding the rule $AttendClass(x, y) \wedge (y \geq 200) \Rightarrow CanTA(x, y)$ may speed the solving of some queries. The utility of adding such rules is a subject of ongoing research (e.g., [Minton, 1988]). For strong irrelevance, savings are guaranteed for many cases. For example, when considering all derivations of the query (i.e., $\mathcal{D}_0 = \mathcal{D}_\Delta(\psi)$), deriving ψ from $\Delta - \Phi$ costs no more than deriving it from Δ if $SI(\Phi, \psi, \Delta, DI_2, \mathcal{D}_\Delta(\psi))$ holds. This property also holds if we consider a set of derivations $\mathcal{D}_0$, such that the inference engine is always guaranteed to find one of the derivations in $\mathcal{D}_0$ before it finds others. In the next section, we will illustrate that removing strongly-irrelevant formulas can result in significant time savings and not only in space savings.

3 Deriving Irrelevance Claims

A key issue in relevance reasoning is the ability to decide which formulas are irrelevant to a given query. Specifically, two questions are of interest. First, given a knowledge base and a query, which formulas in the knowledge base are irrelevant to the query? Second, if we are given an irrelevance claim by a user, can we derive other irrelevance claims that logically follow. This section considers these questions for the case of KBs consisting of a set V of Horn rules and a set G of ground atomic facts

We consider atomic queries that are either ground or contain free variables. A derivation of a query uses a single rule of inference: Given an instance of a rule from the KB, the consequent of the rule can be inferred if the antecedents were inferred previously. A derivation D is conveniently viewed as a tree, as shown in Figure 1 A derivation tree (and the corresponding derivation) is *minimal* if there is no pair of identical goal-nodes, n_1 and n_2, such that n_1 is an ancestor of n_2. A formula ϕ is said to be irrelevant to a derivation D if ϕ does not appear in D (i.e., we use $DI2$ from the previous section). We distinguish two sets of predicates in the KB. the *eztenstonal predicates* (EDB) that appear in G and the *intensional predicates* that appear in the consequents of the rules. As a syntactic convenience, we assume that base predicates do not appear in the consequents of any of the rules.

Deriving irrelevance claims requires that we establish properties of all possible derivations of a query, and that entails examining the whole KB. This is, of course, im-

[5] Note that for every $\Delta \in \Sigma$, the set $\mathcal{D}_0$ is actually different. We assume that $\mathcal{D}_0$ is a characterization of a set of derivations that can be instantiated for any given Δ (e.g., all minimal derivations).

[6] Assuming the inference engine is complete. A variant of this definition, which is given in [Subramanian and Genesereth, 1987], is obtained by using DI_4.

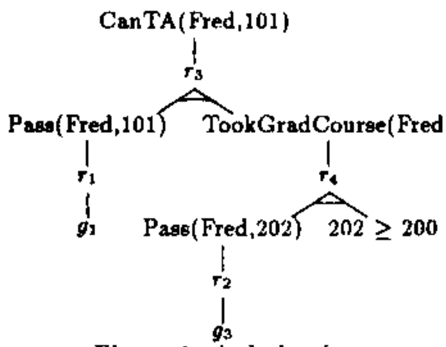

Figure 1: A derivation tree.

practical and defeats the original goal of relevance reasoning. Therefore, we would like to derive irrelevance claims that depend only on a small and stable part of the KB and are independent of changes to other parts of the KB. In many applications, the bulk of the KB consists of ground facts that are updated frequently, while the rules of the KB form a small and stable part. Therefore, we are not going to examine the ground facts of KB directly. Instead, we will use a set *C* of high-level constraints describing the ground facts that may possibly appear in the KB (e.g., $Age(x,y) \Rightarrow y \leq 150$).

Constraints may also appear in rules, in the form of *interpreted predicates* (such as the order predicates $=$, $\neq$, $\leq$ and $<$, or *sort* predicates). Many interactions between rules can be detected by analyzing the semantics of these interpreted predicates. Furthermore, interpreted predicates play an important role in many applications, and often, reasoning with them can be done efficiently. Note that in particular, the variable patterns in the rules can be viewed as constraints (using equality).

We would like our relevance reasoning to incorporate both the semantics of the interpreted predicates appearing in rules and the semantics of the constraints imposed on the possible ground facts. Formally, it means that we have to consider the following problem. Given a set *P* of rules, a set *C* of constraints on the ground facts, and a query *q,* find (some or all) rules and ground facts that are irrelevant to *q* in every KB of $\Sigma(\mathcal{P},\mathcal{C})$, where $\Sigma(\mathcal{P},\mathcal{C})$ denotes the set of all KBs consisting of *V* and a set *G* of ground facts, such that *G* satisfy the constraints of *C.*

The distinctions made in our space of definitions also correspond to different answers to the above problem. First, we observe that determining weak irrelevance is undecidable, even in very restricted cases:

Proposition 3.1: *Determining whether a formula 4> is weakly irrelevant to a query q, with respect to* DI_2 *and the set of all derivations* $\mathcal{D}_\Delta(q)$, *is undecidable even if the rules of V contain no function symbols, C is empty and there are no interpreted predicates in rules.*

This result is proved by a reduction from the rule-redundancy problem [Shmueli, 1987]. Algorithms that find some weakly-irrelevant formulas, but may fail to identify all of them, are described by [Sagiv and Yannakakis, 1981] and [Sagiv, 1988].

For strong irrelevance, the situation is much better. In [Levy and Sagiv, 1992], we have shown that strong irrelevance is decidable for function-free Horn rules and interpreted predicates. The result shows that strong irrelevance is decidable when considering either the set of all derivations $\mathcal{D}_\Delta(q)$ or only the set of all minimal derivations.[7] The algorithms cover a wide range of interpreted predicates, e.g., order and sort constraints.[8]

When function symbols are introduced, determining strong irrelevance is undecidable for KB's with recursion. However, the algorithms described remain sound, i.e., if they deem a formula irrelevant, then it is irrelevant.

Example 3.2 : Let *step, bigStep, badPoint* and *goodPoint* be the EDB predicates, where the rules are:

$r_1 : badPoint(x) \wedge path(x,y) \wedge goodPoint(y) \Rightarrow goodPath(x,y)$.
$r_2 : link(x,y) \Rightarrow path(x,y)$.
$r_3 : link(x,z) \wedge path(z,y) \Rightarrow path(x,y)$.
$r_4 : step(x,y) \Rightarrow link(x,y)$.
$r_5 : bigStep(x,y) \Rightarrow link(x,y)$.

The following constraints are known about the ground facts:
$badPoint(x) \Rightarrow 100 < x < 200$, $step(x,y) \Rightarrow x < y$.
$goodPoint(x) \Rightarrow 150 < x < 170$.
$bigStep(x,y) \Rightarrow x < 100 \wedge y > 200$.

Figure 2 shows that Rule r_5 is strongly irrelevant to the query *goodPath(x,y),* since the constraints on *big Step* contradict those of the nodes that might be unified with the consequent of r_5. Moreover, we can also deem many ground facts strongly irrelevant, such as all *badPoint(x),* where $x \geq 170$, all $step(x,y)$, where $x > 170$ or $y < 100$, and all the facts for *bigStep.* I

In the next section, we will describe an algorithm for deriving logical conclusions from irrelevance claims given by the user. The algorithm uses a powerful tool, called the *query tree,* first introduced in [Levy and Sagiv, 1992]. Below, we briefly describe some aspects of query trees.

3.1 The Query Tree

There are several difficulties in deriving irrelevance claims. First, we need to establish properties of the set of all derivations of the query without enumerating them. Second, we are given the rules of the KB, but have only a partial knowledge about the ground facts in the KB. Finally, we want to enforce the semantics of the interpreted predicates. The query tree provides a compact representation of *precisely* the set of all derivations of the query that satisfy the semantics of the interpreted predicates.

The query tree is a symbolic AND-OR tree consisting of goal nodes and rule nodes (see Figure 2). The root of the tree is a goal node labeled with the query. A goal-node *g* has a child for every rule whose consequent unifies with *g,* and the actual child is the rule resulting from the unification with *g.* A rule node has a goal-node child for

[7]The algorithms for these two cases differ in their complexity

[8]Formally, we require that the constraint language *C* satisfy several properties. We must be able to determine when two sentences in the constraint language are equivalent. Furthermore, there must be a finite number of non-equivalent sentences when the number of variables is fixed. For full details, see [Levy and Sagiv, 1992].

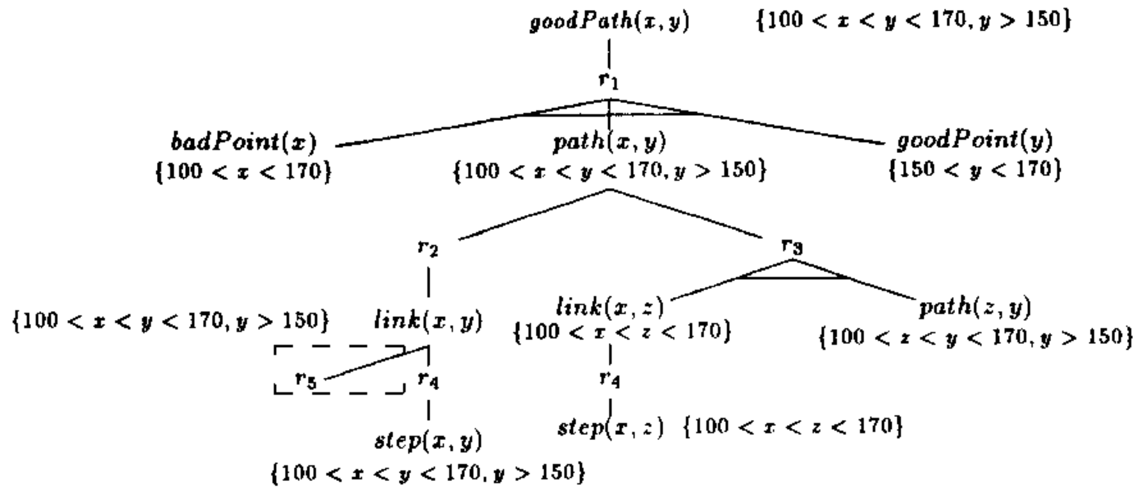

Figure 2: The query tree of Example 3.2. The labels of the rule nodes are omitted for clarity. The expanded equivalent of the node $path(z,y)$ is $path(x,y)$.

every conjunct in its antecedent. If a KB has recursive rules, such a simple minded construction of the tree will not terminate. In order to get a finite representation of all possible derivations, we attach a *label* to each node in the tree. The label of a node contains the tightest constraint that needs to be satisfied by tuples generated in that node. The label is inferred by the constraint literals appearing in the rules and the constraints known about the possible ground facts that may appear in the KB. A goal node will be further expanded only if there is no other expanded node in the tree that has an isomorphic label. Note that computing the labels of the nodes may require several phases of propagation through the nodes in the tree.

A query-tree *T* encodes a set of *symbolic derivations.* A symbolic derivation is like a derivation except that some constants are replaced by variables, and it represents the set of derivations that can be obtained by assigning constants to those variables. A symbolic derivation is *encoded* by the query tree if it can be constructed as follows. Start from the root and choose one rule-node child and its subgoal nodes. Inductively, let *t* be the tree created so far. If *n* is a leaf of *t* for an IDB predicate, let n' be the goal node in *T* that has a label isomorphic to *n* and was expanded (n' may be *n* itself). Expand *n* with one of the children of n'.

When the KB contains no function symbols, the query tree encodes precisely the set of derivations of the query that use formulas that satisfy the constraints. Specifically:

- For any derivation $\in \mathcal{D}_0$ that uses only ground facts that satisfy the constraints of C, there exists a symbolic derivation *d* and an assignment σ, such that $\sigma(d) = d'$ and *d* is encoded by the query tree.
- For every symbolic derivation *d* encoded by the tree, there is a variable assignment σ, such that $\sigma(d) \in \mathcal{D}_0$ and the leaves of $\sigma(d)$ satisfy $\mathcal{C}$.

Consequently, the query tree provides a sound and complete inference mechanism for strong irrelevance (with respect to $\mathcal{D}_\Delta(q)$). Specifically, a rule is strongly irrelevant to q if and only if an instance of it does not appear in the tree. A ground fact $p(a_1,\ldots,a_n)$ is strongly irrelevant if and only if $a_1,\ldots,a_n$ does not satisfy the label of any goal node of p in the query tree. When the KB contains function symbols (and recursion) the query tree encodes a superset of $\mathcal{D}_\Delta(q)$, and therefore, it provides only a sound inference mechanism for strong irrelevance.

By refining the labels of nodes in the tree, the query tree can also be made to encode precisely the set of minimal derivations of the query (see [Levy and Sagiv, 1992]) and the set of all derivations when some of the EDB literals appear negated (see [Levy *et al.*, 1993]).

The complexity of building the tree (and therefore of deriving *SI*) is linear in the number of rules in $\mathcal{P}$. It may be exponential in the arity of predicates. However, arities of predicates tend to be very small (e.g., frame systems usually employ mostly binary predicates). Furthermore, we believe that exponential running time is not likely to occur in practice (since finding examples with exponential running time requires careful crafting of the rules),[9] and so, building the query tree is not an expensive operation.

3.2 Irrelevance Facts from an External Source

Often a user may be able to supply the system with additional irrelevance claims based on his knowledge about the ground facts in the KB (or those that may appear in the KB). Specifically, the user may know that a set of formulas Φ is strongly irrelevant to the query q in the current state of the KB. This knowledge may not be expressible as explicit constraints on ground facts.[10] Clearly, if we are told that a formula ϕ is strongly irrelevant to q, we can remove ϕ from the KB. However, we may also be able to conclude that other formulas are irrelevant as well. This section describes an algorithm for deriving such conclusions using the query-tree.

Specifically, it requires rules that create in their consequents permutations of the variables from their antecedents

[10] For example, this knowledge may be based on the fact that the join of two relations is empty.

Suppose that $\mathcal{P}$ is a set of rules and $\mathcal{C}$ is a set of constraints on the EDB predicates. Let I be an irrelevance claim stating that a set of formulas Φ is strongly irrelevant to a query q; more precisely, I states that the set of possible KBs is some subset $\Sigma' \subseteq \Sigma(\mathcal{P}, \mathcal{C})$, such that $SI(\Phi, q, \Sigma', DI_2, \mathcal{D}_\Delta(q))$ holds. We assume that Φ is either a set of rules or a set of ground facts specified as $\{p(\bar{x}) \mid C(\bar{x})\}$, where p is some predicate and $C(\bar{x})$ is a formula with only interpreted predicates.

To derive logical conclusions from I, our strategy is to create a set of rules $\mathcal{P}_1$, such that when formulas from Φ are excluded, $\mathcal{P}_1$ and $\mathcal{P}$ produce the same derivations of the query q for every set of ground facts G satisfying $\mathcal{C}$. We then create a query tree for $\mathcal{P}_1$ and find all formulas that are strongly irrelevant to q. Clearly, if a formula ϕ is strongly irrelevant with respect to $\mathcal{P}_1$, then it is also strongly irrelevant with respect to $\mathcal{P}$ whenever I holds.

Formally, $\mathcal{P}_1$ is created as follows:

1. If Φ is a set of rules, then $\mathcal{P}_1 = \mathcal{P} - \Phi$.
2. If $\Phi = \{p(\bar{x}) \mid C(\bar{x})\}$, then let the negation of $C(\bar{x})$ be $d_1 \vee \ldots \vee d_m$, where each d_i is a conjunction of literals of interpreted predicates. The set $\mathcal{P}_1$ consists of all the rules of $\mathcal{P}$ whose consequent is not p and all rules of the form $q_1(\bar{x}_1) \wedge \ldots q_l(\bar{x}_l) \wedge d_i(\bar{x}) \Rightarrow p(\bar{x})$ $(1 \leq i \leq m)$, where $q_1(\bar{x}_1) \wedge \ldots q_l(\bar{x}_l) \Rightarrow p(\bar{x})$ is a rule of $\mathcal{P}$.

Example 3.3: Suppose that in Example 3.2 we are told that $\{path(x, y) \mid x < 110 \wedge y > 160\}$ are strongly irrelevant. The rules for the predicate $path$ in $\mathcal{P}_1$ would therefore be:
$link(x, y) \wedge x \geq 110 \Rightarrow path(x, y)$.
$link(x, z) \wedge x \geq 110 \wedge path(z, y) \Rightarrow path(x, y)$.
$link(x, y) \wedge y \leq 160 \Rightarrow path(x, y)$.
$link(x, z) \wedge path(z, y) \wedge y \leq 160 \Rightarrow path(x, y)$.
The query tree for $\mathcal{P}_1$ shows that $\{badPoint(x) \mid x < 110\}$ and $\{goodPoint(x) \mid x > 160\}$ are strongly irrelevant to $goodPath(x, y)$. ▮

Theorem 3.4: *Suppose that ϕ is strongly irrelevant to q with respect to $\Sigma(\mathcal{P}_1, \mathcal{C})$, or more precisely, suppose that $SI(\phi, q, \Sigma(\mathcal{P}_1, \mathcal{C}), DI_2, \mathcal{D}_\Delta(q))$ is true. Then $SI(\phi, q, \Sigma', DI_2, \mathcal{D}_\Delta(q))$ holds, where Σ' is the set of possible KBs, as implied by I.*

Proof sketch: The theorem follows from the observation that $\mathcal{P}_1$ produces the same derivations as $\mathcal{P}$, except for the ones that use formulas in Φ. If Φ is a set of rules, the observation follows trivially. If Φ includes ground facts, the added literals in the rules of $\mathcal{P}_1$ guarantee that formulas in Φ will not be used in derivations of $\mathcal{P}_1$. ▮

In general, as the theorem below shows, even if there are no function symbols, finding all the logical consequences of a strong-irrelevance claim is not possible.

Theorem 3.5: *Suppose that I states that the set of possible KBs is some subset $\Sigma' \subseteq \Sigma(\mathcal{P}, \mathcal{C})$, such that $SI(\Phi, q, \Sigma', DI_2, \mathcal{D}_\Delta(q))$ holds. Determining whether $SI(\phi, q, \Sigma', DI_2, \mathcal{D}_\Delta(q))$ holds is undecidable.*

The theorem is proved by a reduction from the rule-redundancy problem [Shmueli, 1987].

3.3 The Utility of Relevance Reasoning

Removing strongly-irrelevant formulas (i.e., rules and ground facts) effectively prunes many useless paths that a problem solver (such as a backward chainer) has to pursue. Removing a large number of ground facts can particularly impact the performance, since much of the cost of a problem solver is in doing database lookups. The savings will be especially significant when the lookup involves uninstantiated variables. For instance, in Example 3.2 we need to perform many lookups of the form *step(x,y),* where y is uninstantiated. Removing all the ground facts for which y > 170 will drastically reduce the search.

Identifying irrelevant facts also yield savings when updates are done. For example, if the KB is updated with a fact that is known to be irrelevant, then we need not recompute the answer to the query. Finally, identifying which facts are irrelevant to a query leads to space savings in storing the KB. This is especially significant when deciding which parts of a large KB should be brought into main memory.

We tested the impact of removing irrelevant facts for over 20 sets of queries taken from four domains. Space limitations preclude the presentation of the complete results. Table 1 presents a set of representative results. More detailed results can be found in [Levy, 1993] Rows 1 &: 2 use the rules given in Example 3.2. Rows 3-6 are taken from a travel KB (using real airline data) Row 7 uses a KB describing compatibilities between wines and dishes (gleaning some knowledge from [Rombauer and Rombauer-Becker, 1975]), while the last row uses a KB describing relationships between students, advisors and institutions (using a database of Ph.D. graduates in computer science).

In the table, *Filtering Time* is the time taken to build the query tree and to remove the irrelevant facts. *Percent irrelevant* is the percent of facts that were removed from the KB. BC1 is the time taken to find all solutions to the query using the original KB, and BC2 is the corresponding time using the filtered KB.[11] The results show significant speedups, usually in excess of 3, ranging up to 31 (mean: 10.8, median: 4.4), while the time taken to build the query tree and filter the KB are usually insignificant. The speedups grew significantly as the percent of irrelevant facts grew. For example, using the same query as in Row 3, the speedup was a factor of 280 when 90% of the ground facts were removed. Furthermore, the speedups grew as the size of the KB grew, even when the percent of irrelevant facts remained the same. Similar speedups were recorded when the number of nodes visited in the search was the performance measure (instead of the execution time) and when measuring the time taken to find just the first solution to the query. The experiments were performed on a TI Explorer II.

[11] The performance of our backward chainer compared favorably with that of Epikit (a commercial implementation of MRS [Russell, 1985)).

	KB size		Filtering	Percent	BC1	BC2
	Facts	Rules	time (sec)	irrelevant	(sec)	(sec)
1	350	6	1.7	63 %	2450	170
2	350	6	1.7	63 %	580	214
3	200	18	7.8	70 %	380	12
4	200	18	5.6	70 %	25	5.3
5	200	18	19.8	65 %	41	10.8
6	200	18	8.5	62 %	460	19
7	1300	47	8.7	70 %	24	18.5
8	150	17	0.8	59 %	33	7.5

Table 1: Experimental results.

4 Discussion

Relevance reasoning is a powerful method to control inference. We presented a general framework for stating knowledge about irrelevance and reasoning with it. We concentrated on the class of strong-irrelevance claims that has several desirable properties, such as existence of efficient algorithms for detecting irrelevant facts. Removing strongly-irrelevant formulas may only improve the performance, and our experiments have shown that these savings are significant. Furthermore, relevance reasoning is done with only partial knowledge about the contents of the KB and does not need to be repeated when certain changes occur in the KB. Consequently, our methods are especially effective for KBs that contain many ground facts.

Our analysis of irrelevance can be viewed as a refinement of the analysis in [Subramanian, 1989]. The specific definitions considered by Subramanian fall under weak irrelevance in our framework. Subramanian defined the class of computational-irrelevance claims to be claims that lead to computational savings. Our class of strong irrelevance is a prime example of computational irrelevance. It should be noted that [Subramanian and Genesereth, 1987) discusses a definition of strong irrelevance, but it is a variation on weak irrelevance and is not an instance of computational irrelevance.

The query tree encodes the space of possible derivations of the query. Recently, the question of finding optimal methods to search that space has received a lot of attention [Smith, 1986, Greiner, 199l]; a related issue is analyzing the utility of techniques in explanation based learning [Etzioni, 1990, Greiner and Jurisica, 1992]. Much of this work requires a graph-like representation of the search space under consideration. The query tree is such a representation that handles recursive theories in a principled manner and fully incorporates the interpreted constraints appearing in rules.

5 Acknowledgements

The authors would like to thank Richard Fikes, Nir Friedman, Karen Myers, Pandu Nayak and Jeff Van Baalen for valuable discussions and comments on earlier drafts of this paper.

References

[Elkan, 1990] Elkan, Charles 1990. Independence of logic database queries and updates. In Proceedings of the 9th ACM Symp. on Principles of Database Systems. 154-160.

[Etzioni, 1990] Etzioni, Oren 1990. Why prodigy/ebl works. In Proceedings of AAAI-90.

[Gardenfors, 1978] Gardenfors, Peter 1978. On the logic of relevance. In Sythese. (37) pp. 351-367.

[Greiner and JuriSica, 1992] Greiner, Russell and Jurisica, Igor 1992. A statistical approach to solving the ebl utility problem. In Proceedings of AAAI~92.

[Greiner, 1991] Greiner, Russell 1991. Finding optima] derivation strategies in a redundant knowledge base. In Artificial Intelligence. Vol. 50 (1) pp. 95 116.

[Levy and Sagiv, 1992] Levy, Alon Y. and Sagiv, Yehoshua 1992. Constraints and redundancy in datalog. In The Proceedings of the 11th ACM Symp. on Principles of Database Systems.

[Levy and Sagiv, 1993] Levy, Alon Y. and Sagiv, Yehoshua 1993. Queries independent of updates. Technical Report, KSL, Stanford University.

[Levy et a/., 1993] Levy, Alon Y.; Mumick, Inderpal Singh; Sagiv, Yehoshua; and Shmueli, Oded 1993. Equivalence, query-reachability and satisfiability in datalog extensions. In Proceedings of the 12th ACM Symp. on Principles of Database Systems.

[Levy, 1993] Levy, Alon Y. 1993. Irrelevance Reasoning in Knowledge Base Systems. Ph.D. Dissertation, Stanford University, Stanford, CA.

[Minton, 1988] Minton, Steve 1988. Quantitative results concerning the utility of explanation based learning. In Proceedings of AAAI-88.

[Pearl, 1988] Pearl, Judea 1988. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann Publishers, Inc., San Mateo, California.

[Rombauer and Rombauer-Becker, 1975] Rombauer, Irma S. and Rombauer-Becker, Marion 1975. Joy of Cooking. Bobbs Merrill Company Inc., N.Y.C., NY.

[Russell, 1985] Russell, Stuart 1985. The complete guide to MRS. Technical Report KSL-85-12, Department of Computer Science, Stanford University, Stanford, CA.

[Sagiv and Yannakakis, 198l] Sagiv, Y. and Yannakakis, M. 1981. Equivalence among relational expressions with the union and difference operators. In J. ACM 27:4 pp. 6SS-655.

[Sagiv, 1988] Sagiv, Yehoshua 1988. Optimizing datalog programs. In Minker, Jack, editor 1988, Foundations of Deductive Databases and Logic Programming. Morgan Kaufmann, Los Altos, CA. 659-698.

[Shmueli, 1987] Shmueli, Oded 1987. Decidability and expressiveness aspects of logic queries. In Proceedings of the 6th ACM Symposium on Principles of Database Systems.

[Smith, 1986] Smith, David 1986. Controlling Inference. Ph.D. Dissertation, Stanford Univesity, Stanford, CA.

[Srivastava and Ramakrishnan, 1992] Srivastava, Divesh and Ramakrishnan, Raghu 1992. Pushing constraint selections. In Proceedings of the 11th ACM Symp. on Principles of Database Systems, San Diego, CA.

[Subramanian and Genesereth, 1987] Subramanian, D. and Genesereth, M.R. 1987. The relevance of irrelevance. In Proceedings of IJCAI-87.

[Subramanian, 1989] Subramanian, Devika 1989. A Theory of Justified Reformulations. Ph.D. Dissertation, Stanford University, Stanford, CA.