On Structured Belief Bases - Preliminary Report

Renata Wassermann

Institute of Logic, Language and Computation*

University of Amsterdam

email: renata@wins.uva.nl

Abstract

In this paper some ideas for adding structure to belief bases are presented. Structured belief bases can be seen as graphs, where each node is a belief and two nodes are adjacent if and only if they are related. Some notions of relatedness are defined. We then show how this extra structure can be used for implementing local change as defined in [HW98].

1 Introduction

Intelligent agents must have some way of revising their beliefs. Even though the AGM paradigm [AGM85] provides us a very elegant and powerful framework for reasoning about how a rational agent should change his beliefs when confronted with new information, it tells us very little about how such agent could really perform such belief changes.

Real agents (natural or artificial) cannot, and even stronger, are not supposed to calculate all consequence of their beliefs. Even if we consider approaches to belief revision that use belief bases [Fuh91, Han89, Neb92] instead of logically closed belief sets, the operations defined always depend of finding

*Plantage Muidergracht 24, 1018-TV, Amsterdam, The Netherlands

maximal subsets not implying some formula α, or minimal subsets implying α. This means that indirectly, they are using all the consequences of the belief base. Exploring all subsets of a reasonably sized base is a very expensive operation.

One way to attack the problem is to try to reduce the size of the set to be explored. Intuitively, not all of an agent's beliefs are relevant for deciding what to do with new information. There should be a way of isolating the subset of a belief base that contains the relevant beliefs for an operation of belief change. In [HW98], this approach is followed. A notion of compartment is presented for retrieving the relevant part of a belief base and then local operations of belief change are defined that act on the relevant part of the beliefs only. The problem is that the way the compartments are defined uses the notion of minimal subsets of the base implying some formula, that is, finding a compartment is as expensive as performing a traditional operation of belief change. But the representation results obtained for the local versions of the belief change operations are very general and do not depend on the particular way the compartment is defined, but only on certain properties of the result.

In this paper we present more efficient ways of retrieving the relevant part of a belief base. The retrieved set can then be "plugged" into the constructions of local operations. Even though the operations also rely on finding minimal or maximal subsets implying or not implying certain formulas, the set to be explored is reduced.

The paper proceeds as follows: in the next section the basic definitions and results of [HW98] are summarized. In section 3 a computer system is presented which has a very efficient mechanism for retrieving relevant beliefs from a base structured as a graph. The ideas of the system inspired some considerations on more general ways of structuring the base and on formalizing the notion of retrieving the most relevant beliefs for a certain input. These considerations are presented in section 4. In section 5, examples of relatedness relations are presented. Then some computational aspects are examined in section 6 and conclusions presented in section 7.

On the rest of the paper we will be working on a propositional framework, with a language $\mathcal{L}$ containing the basic boolean connectives.

Since we do not want to consider differences on the syntactical form of formulas in this paper, we can assume that (i) logical approach: when we say "p occurs in φ" we actually mean "for all ψ such that $\vdash \varphi \leftrightarrow \psi$, p occurs in ψ"; or (ii) computational approach: the formulas are all represented in some normal form, equivalent formulas have equal representation.

We will use the terms *belief bases* and *databases* indistinctly to refer to a set of formulas.

2 Local Change

In [HW98], we presented a framework for performing local reasoning. The key idea is that not all of an agent's beliefs are relevant for an operation of belief change. As an example, suppose I wake up and see that the sun is shining in Amsterdam. This contradicts my previous beliefs about the weather, like the one that it always rains in Holland, and leads to revision. However, the revision process does not have to take my beliefs about Mathematics into account. These beliefs are completely irrelevant for the change taking place and should not play a role in it.

How do we determine which of the beliefs from a belief base are relevant for a certain operation? We are interested in the concept of a compartment around a formula or a set of formulas. In [HW98], we define the relevant formulas in B for A as those that contribute to proving or disproving any formula of A. We formalize this using the notion of kernel sets - minimal subsets implying a given sentence:

Definition 2.1 *Let C be an inference operation on $\mathcal{L}$. Then the kernel operation $\perp\!\!\!\perp_C$ is the operation such that for all subsets B and elements α of L, $X \in B \perp\!\!\!\perp_C \alpha$ if and only if:*

1. $X \subseteq B$
2. $\alpha \in C(X)$
3. *for all Y, if $Y \subset X$ then $\alpha \notin C(Y)$*

The elements of $B \perp\!\!\!\perp_C \alpha$ are called kernels. We write $\perp\!\!\!\perp$ as an abbreviation of $\perp\!\!\!\perp_{Cn}$, where Cn is a classical consequence operation.

Definition 2.2 *The A-compartment of B, where A and B are sets of sentences is defined as:*

$c(A, B) = \bigcup_{\alpha \in A}(\bigcup((B \perp\!\!\!\perp \alpha) \cup (B \perp\!\!\!\perp \neg\alpha) \setminus (B \perp\!\!\!\perp \perp))$

We call c a compartmentalization function.

Given the notion of compartment around a set A, we can define the restriction of a given inference operation C to A as:

Definition 2.3 *Let C be an inference operation on $\mathcal{L}$ and let c be a compartmentalization function as in definition 2.2. Then for any set A, the A-restriction of C is the inference operation C_A such that for all sets B of sentences: $C_A(B) = C(c(A, B))$.*

A set B is A-restrictedly consistent if and only if $\perp \notin C_A(B)$

$B \vdash_A \alpha$ is an abbreviation of $\alpha \in C_A(B)$, and $B \vdash_\beta \alpha$ of $\alpha \in C_{\{\beta\}}(B)$.

If the underlying logic is monotonic, then the local restriction of the inference operator satisfies monotonicity and compactness.

We also showed how local versions of the operations of contraction, consolidation, revision and semi-revision can be constructed and characterized by means of postulates. I reproduce below one of the results obtained, namely the characterization of local kernel contraction, that can be used for deriving the others.

The idea behind kernel contraction is that, if we remove from the belief base at least one element of each α-kernel (minimal subset of the base that implies α), we obtain a belief base that does not imply α. In the case of the local operation, we are interested in obtaining a belief base that does not A-restrictedly imply α, where A is a set of sentences. We consider then only those kernel sets that A-restrictedly imply α and use an incision function to select the formulas from the kernel sets to be removed. An incision function is a function defined from sets of sets of sentences into sets of sentences, selecting at least one sentence from each set of the argument.

Definition 2.4 *An incision function is any function $\sigma : \mathcal{P}(\mathcal{P}(\mathcal{L})) \rightarrow \mathcal{P}(\mathcal{L})$ such that for any subset S of $\mathcal{P}(\mathcal{L})$:*

1. $\sigma(S) \subseteq \bigcup S$
2. *If $\emptyset \neq X \in S$, then $X \cap \sigma(S) \neq \emptyset$*

Definition 2.5 *Let C be an inference operation on $\mathcal{L}$ and σ an incision function. The local kernel contraction of B determined by C and σ is the operation $\dot{-}_{C,\sigma}$ such that for all sets of sentences B:*

$B \dot{-}_{C,\sigma} \alpha = B \setminus \sigma(B \perp\!\!\!\perp_C \alpha)$

The following theorem characterizes the operation of local kernel contraction. The intended interpretation is that C should be a local restriction C_A of a consequence operator Cn, as defined above. However, the formal result is more general and is therefore stated in terms of an operation C of a more general type.

Theorem 2.6 *Let C be an inference operation satisfying monotonicity and compactness. An operation $\dot{-}$ is an operation of local kernel contraction determined by C and some incision function if and only if for all sets of sentences B:*

- *If $\alpha \notin C(\emptyset)$, then $\alpha \notin C(B \dot{-} \alpha)$ (success)*
- *$B \dot{-} \alpha \subseteq B$ (inclusion)*

- *If $x \in B \setminus B \dot{-} \alpha$, then there is some $B' \subseteq B$ such that $\alpha \notin C(B')$ and $\alpha \in C(B' \cup \{x\})$ (core-retainment)*

- *If for all subsets B' of B $\alpha \in C(B')$ if and only if $\beta \in C(B')$, then $B \dot{-} \alpha = B \dot{-} \beta$ (uniformity)*

The problem with this approach is that the way the compartments are calculated is extremely inefficient. On the whole, the local change operations are as costly as the original versions, although intuitively more appropriate.

However, the representation results proved do not rely on the way the compartments are defined, but only on properties of the local inference operation obtained, namely, monotonicity and compactness. The compartment construction in definition 2.2 can be seen as an example, as one way of constructing a compartment such that the restriction of the consequence operator to the compartment satisfies the requirements. This means that if we find another way of retrieving the relevant beliefs from a base such that the associated local inference operation is also monotonic and compact, all the results go through.

In the rest of the paper, some ideas for efficient ways of retrieving relevant information are presented.

Note that even if we apply traditional constructions for belief change on the set of relevant beliefs, since this set can be kept much smaller than the belief base as a whole, the operations become more feasible.

3 RABIT

In this section I will briefly discuss the system RABIT (**R**easoning **A**bout **B**eliefs **I**n **T**ime). For details, the reader is referred to [Gar93, GED96].

The system consists of four modules, each one representing another sort of memory: LTM (Long-Term Memory), STM (Short-Term Memory), ITM (Intermediate-Term Memory) and RTM (Relevant-Term Memory).

The idea is that all the information (the beliefs) is stored in the LTM component, but the reasoning is performed in a small subset of the beliefs in LTM, the STM. The ITM component stores the history of the reasoning process and the RTM works as a kind of context, storing the relevant concepts (in RABIT concepts are any symbol in the language, like predicate names and constants). The idea of working in a small subset of the belief base is psychologically as well as computationally motivated.

RABIT has a very efficient algorithm for retrieving the relevant information from the LTM. The LTM is organized as a bipartite graph where the nodes are either formulas or concepts. Each formula is linked to all the concepts that occur in it. So, (their example) the formula *penguin(tweety)* is linked to the nodes *penguin* and *tweety*. If one wants to retrieve every formula that mentions Tweety, one only needs to take all nodes adjacent to Tweety in the graph.

A method of marker-passing is used for determining which formulas of the LTM go to STM. The system chooses the relevant concepts to start with (for example, the concepts occurring in a new belief acquired) and "spreads" the activation through the edges of the graph. The activation level decreases every time it passes a new edge. Every belief with an activation level above a certain threshold is copied into the STM. By regulating the different parameters, like the initial activation, the decrease function and the threshold of activation, one can keep the STM small.

The LTM may contain contradictions, that are only solved when they become evident in the STM. They are solved in the STM by means of a precedence order that tells the system which belief should be given up, but remain in the LTM. This means that a contradiction may be retrieved over and over again due to the same input. There is an "adaptive behaviour" that increases the distance between the formula that "lost" in the contradiction solving procedure and any node, so that it becomes less and less likely that the contradiction will be retrieved again.

The RABIT architecture is intuitively very appealing, but there are some problems:

1. The retrieval mechanism is very efficient, but what happens after the relevant part of the LTM has been retrieved is that the system starts inferring everything it can from the beliefs in the STM, that is, it is not goal-directed.

2. The LTM only grows, new beliefs are added, but nothing is deleted. Besides the fact that this presents obvious disadvantages from the computational point of view, it does not seem very intuitive that an agent, after concluding that a belief is false, continues to hold it for the same reasons. The belief could certainly be reinferred by another line of reasoning but we do not want a system that follows always the same (doomed to failure) line of reasoning.

3. The way in which the LTM is organized is purely syntactical. There are no links between concepts, for example. If we know that Paul is a lecturer, that a lecturer is a member of the staff and that members of the staff are people, there is no "short path" between beliefs about people in general and beliefs about Paul.

4 Structuring the belief base

In this section we add extra structure to belief bases in order to make the retrieval of the relevant beliefs more efficient.

We begin by assuming that a relatedness relation between formulas of the language is given, with the intended meaning as follows:

$\mathcal{R}(\varphi, \psi)$ if and only if φ and ψ are directly related.

Some desired properties of $\mathcal{R}$ are:

1. reflexivity - $\mathcal{R}(\varphi, \varphi)$
2. symmetry - $\mathcal{R}(\varphi, \psi) \Leftrightarrow \mathcal{R}(\psi, \varphi)$
3. $\mathcal{R}(\varphi, \neg\varphi)$

Transitivity is not assumed, for two reasons: first, we want to be able to model agents with limited resources, who may be unable to calculate the transitive closure of the relatedness function. Second, we want to be able to talk about degrees of relatedness.

Given a relatedness relation, we can represent a belief base as a (possibly disconnected) graph where each node is a formula and there is an edge between φ and ψ if and only if $\mathcal{R}(\varphi, \psi)$. This graph representation gives us immediately a notion of degrees of relatedness: the shorter the path between two formulas of the base is, the closer related they are. Another notion made clear is that of connectedness: the connected components of the graphs give a partition into unrelated "topics" or "subjects".

Formally:

Definition 4.1 *Let B be a belief base and $\mathcal{R}$ be a relation between formulas. A $\mathcal{R}$-path between two formulas φ and ψ in a belief base B is a sequence $P = (\varphi_0, \varphi_1, ..., \varphi_n)$ of formulas such that:*

1. *$\varphi_0 = \varphi$ and $\varphi_n = \psi$*
2. *$\{\varphi_1, ..., \varphi_{n-1}\} \subseteq B$*
3. *$\mathcal{R}(\varphi, \varphi_1)$, $\mathcal{R}(\varphi_1, \varphi_2)$, ..., and $\mathcal{R}(\varphi_n, \psi)$.*

If it is clear from the context to which relation we refer we will talk simply about a path in B.

We represent the fact that P is a path between φ and ψ by $\varphi \overset{P}{\rightsquigarrow} \psi$.

The length of a path $P = (\varphi_0, \varphi_1, ..., \varphi_n)$ is $l(P) = n$

Definition 4.2 *Let B be a belief base and $\mathcal{R}$ a relation between formulas of the language. We say that two formulas φ and ψ are related in B if and only if there is a path P such that $\varphi \stackrel{P}{\leadsto} \psi$.*

Definition 4.3 *Let B be a belief base, $\mathcal{R}$ a relation between formulas of the language and φ and ψ formulas. The unrelatedness degree of φ and ψ in B is given by:*

$$u(\varphi, \psi) = \begin{cases} 0 & \textit{if } \varphi = \psi \textit{ and } \varphi \in B \\ \textit{inf}\{l(P) | \varphi \stackrel{P}{\leadsto} \psi\} & \textit{if } \varphi \textit{ and } \psi \textit{ are related in } B \\ \infty & \textit{otherwise} \end{cases}$$

Observation 4.4 *A distance function on a set U is a function $d : U \times U \to$ IR such that: (i) $d(x, y) \geq 0$; (ii) $d(x, y) = 0$ iff $x = y$; (iii) $d(x, y) = d(y, x)$; and (iv) $d(x, y) \leq d(x, z) + d(z, y)$. The unrelatedness degree u restricted to the elements of B is a distance measure.*

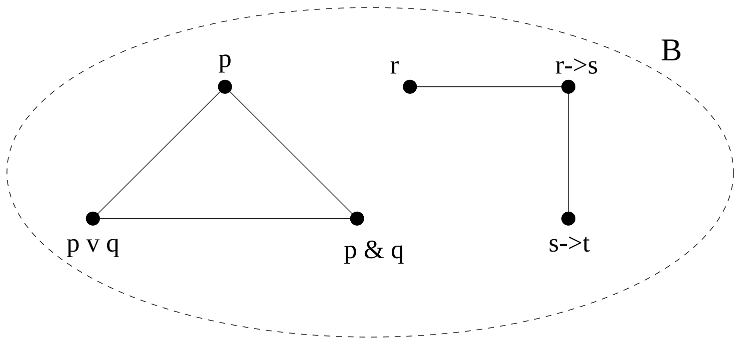

Figure 1: Structured belief base

In figure 1 we see an example of a belief base structured by a relation $\mathcal{R}$ defined by:

$\mathcal{R}(\varphi, \psi)$ if and only if φ and ψ share an atom

The edges in the figure represent the relation $\mathcal{R}$ between the elements of the belief base. The graph representing the structured base has two disconnected components. The formulas $p \wedge q$ and $r \to s$ are not related in B. The formula $q \wedge r$ is related to all formulas in the belief base, since it is possible to find a path between it and all formulas in the base. Note that the initial and final nodes of the path do not have to be elements of the base. This means that if one adds this formula to the base, all the other elements become related to each other in B. This can be interpreted on the following way: the two disconnected components of the original base represent beliefs about unrelated subjects. As soon as one introduces a belief relating the subjects of both components, all beliefs in the two componnents become related to each other.

We now show how, given the structure of a belief base, to retrieve the set of formulas relevant for a given formula α:

Definition 4.5 *The set of formulas of B which is relevant for a formula α with degree i is given by:*

$\Delta^i(\alpha, B) = \{\varphi \in B | u(\alpha, \varphi) = i\}$ *for* $i \geq 0$

Definition 4.6 *The set of formulas of B which are relevant for a formula α up to degree n is given by:*

$\Delta^{\leq n}(\alpha, B) = \bigcup_{0 \leq i \leq n} \Delta^i(\alpha, B)$ *for* $n \geq 0$

We say that $\Delta^{\leq \omega}(\alpha, B) = \bigcup_{i \geq 0} \Delta^i(\alpha, B)$ is the set of relevant formulas for α.

In section 2 we defined a notion of compartment around a sentence. In definition 2.2 the α-compartment of a belief base B was defined as the set of formulas of B that contribute for proving α or its negation. These were the formulas of B that were relevant for an operation of belief change. We have seen that for the representation results obtained in [HW98] the particular construction of the compartments did not matter, but only the properties of the inference relation obtained. We now show some properties of the inference operation obtained using $\Delta^{\leq n}$ instead of c (definition 2.2) as a compartmentalization function.

Proposition 4.7 *Let B be a belief base and α a formula. The local inference operations defined as $C^n_\alpha(B) = Cn(\Delta^{\leq n}(\alpha, B))$ are compact and monotonic.*

Proof: Since all sets considered are finite, compactness follows trivially. For the monotonicity, it suffices to note that if $B \subseteq D$, then $\Delta^{\leq n}(\alpha, B) \subseteq \Delta^{\leq n}(\alpha, D)$. □

This means that for any n, $\Delta^{\leq n}$ can be used as a compartmentalization function to define local operations characterized by the results presented in [HW98] and summarized in section 2. In the case of an ideal agent, we retrieve all of the agent's relevant beliefs, that is, $\Delta^{\leq \omega}(\alpha, B)$. Otherwise, we can limit the size of the set retrieved by the choice of n.

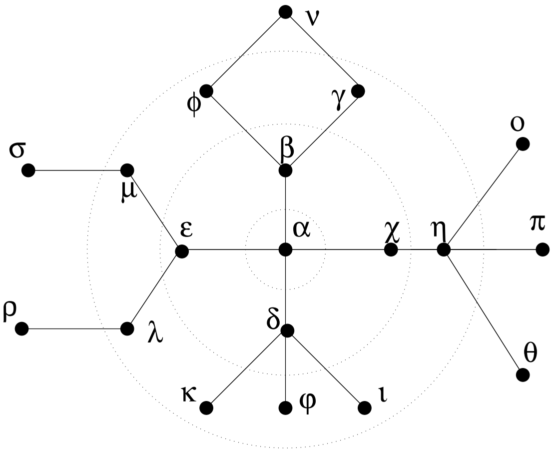

Figure 2: Degrees of relevance

In figure 2, we see a structured belief base $B = \{\alpha,\ \beta,\ \gamma,\ \delta,\ \varepsilon,\ \eta,\ \theta,\ \iota,\ \kappa,\ \lambda,\ \mu,\ \nu,\ o,\ \pi,\ \rho,\ \sigma,\ \phi,\ \varphi,\ \chi\}$. The dotted circles represent different levels of relevance for α. We have:

$\Delta^0(\alpha, B) = \{\alpha\}$
$\Delta^1(\alpha, B) = \{\beta, \chi, \delta, \varepsilon\}$
$\Delta^2(\alpha, B) = \{\gamma, \eta, \iota, \varphi, \kappa, \lambda, \mu, \phi\}$
$\Delta^3(\alpha, B) = \{\nu, o, \pi, \theta, \rho, \sigma\}$
$\Delta^{\leq\omega}(\alpha, B) = \Delta^0(\alpha, B) \cup \Delta^1(\alpha, B) \cup \Delta^2(\alpha, B) \cup \Delta^3(\alpha, B) = B$

5 Where does the structure come from?

In the previous section we assumed that a relation of relatedness between elements of the language was given. In this section we will present some ways of deriving such a relation from the belief base.

Epstein [Eps90] takes the topic or subject matter of a proposition as a primitive and says two propositions are related if they share a subject matter. He considers a set of topics $\mathcal{T}$ and supposes that each propositional variable has a non-empty set of topics associated. The set of topics associated with a proposition is simply the union of the sets of topics associated to each variable appearing in it. Epstein shows that one can also take the relatedness relation as a primitive and derive a topic assignment from it.

In many real applications, there is a part of the database dedicated to the definition of concepts which can be used to generate a relatedness relation. Let A be the subset of B where all the definitions are made. We can say that atoms appearing in the same formula of A are related. We can say that two formulas φ and ψ are related if they share an atom or if there are atoms p in φ and q in ψ such that p and q are related.

Formally:

Definition 5.1 *Let A be a set of formulas. We define a relation R_A on atoms by:*

$R_A(p,q)$ if and only if $p = q$ or p and q occur in the same formula of A.

The relation R_A can be extended to a relation on the language:

Definition 5.2 *Let $\mathcal{R}_A$ be a relation on formulas defined as:*

$\mathcal{R}_A(\varphi,\psi)$ iff there are atoms p in φ and q in ψ such that $R_A(p,q)$.

It is easy to see that $\mathcal{R}_A$ is symmetric and reflexive and that for every φ, $\mathcal{R}_A(\varphi,\neg\varphi)$.

The transitive closure of the relation $\mathcal{R}_A$ gives an equivalence relation that determines a partition of the belief base. Each member of the partition is completely independent from the others with respect to the definitions in A.

We can also have $A = B$, that is, use the whole belief base for deriving the relatedness relation. This is equivalent to the approach followed by [Rod97].

Rodrigues [Rod97] claims that to ask for a primitive relation between propositional variables or a primitive subject matter assignment may be to strong a requirement. He proposes instead to use directly the relation

$\mathcal{R}(\varphi,\psi)$ if and only if φ and ψ share an atom

He shows that this is actually the smallest relation satisfying the conditions given in [Eps90].

Another way of deriving the relatedness structure from the given data is to consider the knowledge base as a whole and create a graph that has as nodes formulas and atoms. Each formula is then linked to the atoms that occur in it. This is the approach followed by the RABIT system. The derived structure is that of a (possibly disconnected) bipartite graph. Atoms can be related to each other only via some formula in the database that contains them. In the same way, formulas are only related to other formulas via some atom they share. This has the advantage that it is very easy to insert a new formula in the graph, one only has to link it to the atoms it contains. On the other hand, since there is no link between atoms, even if the atoms p and q are intuitively directly related, the formulas p and q will only be related if there is any formula in the belief base containing both. In a first-order framework one could add some formula *related(p,q)* to the belief base if the agent believes that p and q are related. But if we are restricting ourselves to propositional languages, this is a serious drawback[1].

Epstein's and RABIT's approaches have the property that the relatedness relation is defined on the whole language and is independent from the particular belief base. This means that adding beliefs to the structure consists in adding the input formula α to the set of nodes and adding an edge between α and every node β such that $\mathcal{R}(\alpha, \beta)$. Deleting a belief α from the structure consists in removing α from the set of nodes and removing all the edges leaving from it. After adding or deleting a belief, the result is another structured base. The relation $\mathcal{R}$ does not have to be recalculated.

A third alternative is to combine the equivalence relation approach with the logical one. If the database is such that it contains several (small) independent modules, this can be a good strategy. One can first apply the first notion of relatedness and find a partition of the database. Then one only has to consider the partition members related to the input formula and look for the relevant formulas inside them using the logic formulation of compartments given in section 2. It can be shown that the compartment obtained is the same that would be obtained checking the whole set.

We can define a relatedness relation $\mathcal{R}$ that captures a notion closer to the logical compartments in section 2 as:

$\mathcal{R}(\varphi, \psi)$ if and only if there is a set of formulas $A \subseteq B$ such that $A \not\vdash \psi$, $A \cup \{\varphi\} \vdash \psi$ and $A \cup \{\varphi\} \not\vdash \bot$

or:

$\mathcal{R}(\varphi, \psi)$ if and only if $\varphi \in c(\psi, B)$

where c is a compartmentalization function as in definition 2.2.

Note that this relation is not symmetric. The definitions of path, unrelatedness degree and relevant set of formulas (Δ^ω) in section 4 can be mantained. The obvious problem of this approach is that it may be very hard to find the related pairs, but if the relation is given, no further problems arise.

One way that the relation can be given is as follows: if the language is finite, all sets of formulas can be listed. For each pair of formulas φ and ψ, one can select all sets where φ occurs, check whether they imply ψ and if so, check their subsets to see whether they also imply ψ. If not, then $\mathcal{R}(\varphi, \psi)$. This is of course very costly, but if the same language is going to be used several times, one can pre-compute this relation and use it for all belief bases.

[1] In a propositional framework, one could use some trick like adding a tautology like $(p \vee \neg p) \vee (q \vee \neg q)$ to the belief base in order to relate p and q

6 Computational Aspects

In this section we discuss the complexity of the operation described for retrieving the set of relevant beliefs from a base.

An anytime algorithm is one that whenever it is interrupted has built an approximate solution for a problem, and the longer it runs, the better the approximation gets.

The good thing about the method for retrieving the relevant beliefs is that it is an anytime method, that is, whenever it is interrupted, it has retrieved the most relevant beliefs, and the longer it runs, the closer it gets to retrieving all the relevant beliefs (the maximal connected subgraph). This is a very desirable property for modeling agents that may not have time (or memory) enough to find all the related beliefs. In the ideal case, if there is no resource limitation, the method succeeds in retrieving a maximal connected subgraph.

Below we present a sketch of an algorithm that takes as input a formula α and a belief base and returns the set of formulas of the base that are relevant for α. The algorithm can be stopped at anytime, always returning the set of most relevant beliefs for α. The algorithm is a modification of the algorithm BFS for breadth first search in [CLR90].

The belief base is represented by a vector of formulas, each one with a list of pointers to the adjacent nodes in the graph. The nodes adjacent to a formula α are given by Adjacent(α).

Retrieve(α,B,Relevant):
1. If $\alpha \in B$, then mark(α)
2. $\Delta^1(\alpha, B)$:= Adjacent(α)
3. $i := 1$; stop := false
4. While not stop do
    4.1. For all $\beta \in \Delta^i(\alpha, B)$ mark(β)
    4.2. i := i+1
    4.3 For all $\beta \in \Delta^{i-1}(\alpha, B)$,
        $\Delta^i(\alpha, B) := \Delta^i(\alpha, B) \cup \{\varphi \in$Adjacent$(\beta)$ s.t. not marked $\varphi\}$
    4.4 If $\beta \in \Delta^i(\alpha, B) = \emptyset$, then stop := true
5. Relevant := $\{\beta \in B$ s.t. marked$(\beta)\}$

At each step, the algorithm looks at the set retrieved at the previous step and gets all the neighbours that have not been visited yet. Depending on how the information is encoded, this can be done in linear time.

After retrieving the relevant beliefs, traditional belief change constructions can be applied to it, provided the set is small enough (one can stop the algorithm once the relevant set gets bigger than a certain limit).

7 Conclusions and Further Work

We have presented a method for retrieving the most relevant beliefs from a belief base structured by a relatedness relation. This method can be combined with the constructions of local belief change operations defined in [HW98].

The idea of using graphs representing dependency relations for finding the relevant part of a database is well-known in the field of logic programming (see, for example, [Dix95]). In the case of logic programs, the dependency relation is not symmetric and the graph obtained is a directed one.

Further work includes investigating how to use the structure of the belief base also for the revision of the relevant set.

Acknowledgments: I would like to thank Hans Rott and Frans Voorbraak for comments on an earlier draft. This research is sponsored by a grant from the Brazilian funding agency CAPES and the Dutch Research Council PIONIER-project Reasoning With Uncertainty.

References

[AGM85] Carlos Alchourrón, Peter Gärdenfors, and David Makinson. On the logic of theory change. *Journal of Symbolic Logic*, 50, 1985.

[CLR90] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. MIT Press, 1990.

[Dix95] Jürgen Dix. A classification theory of semantics of normal logic programs. *Fundamenta Informaticae*, XXII(3), 1995.

[Eps90] Richard L. Epstein. *The semantic foundations of logic, volume 1: Propositional Logic*. Nijhoff International Philosophy Series. Kluwer Academic Publishers, 1990.

[Fuh91] André Fuhrmann. Theory contraction through base contraction. *Journal of Philosophical Logic*, 20:175–203, 1991.

[Gar93] Kevin Gary. RABIT - A spreading activation approach to real-time commonsense reasoning. Master's thesis, Arizona State University, Tempe, Arizona, 1993. Available at http://www.public.asu.edu/ kgary/.

[GED96] Kevin Gary and Jennifer Elgot-Drapkin. RABIT- Bridging formal and implementational approaches to commonsense reasoning.

Technical report, Arizona State University, 1996. Available at http://www.public.asu.edu/ kgary/.

[Han89] Sven Ove Hansson. New operators for theory change. *Theoria*, 55:114–132, 1989.

[HW98] Sven Ove Hansson and Renata Wassermann. Local change. In preparation (a preliminary version appeared in the Fourth Symposium on Logical Formalizations of Commonsense Reasoning), 1998.

[Neb92] Bernhard Nebel. Syntax based approaches to belief revision. In Peter Gärdenfors, editor, *Belief Revision*. Cambridge University Press, 1992.

[Rod97] Odinaldo T. Rodrigues. *A Methodology for Iterated Information Change*. PhD thesis, Imperial College, University of London, 1997.