Dependency-Directed Reconsideration
Belief Base Optimization for Truth Maintenance Systems

Frances L. Johnson and **Stuart C. Shapiro**
Department of Computer Science and Engineering; SNePS Research Group
Center for Multisource Information Fusion; Center for Cognitive Science
University at Buffalo, The State University of New York
201 Bell Hall, Buffalo, NY 14260-2000, USA
{*flj* | *shapiro*}*@cse.buffalo.edu*

Abstract

We define reconsideration, a non-prioritized belief change operation on a finite set of base beliefs. Reconsideration is a hindsight belief change repair that eliminates negative effects caused by the order of previously executed belief change operations. Beliefs that had previously been removed are returned to the base if there no longer are valid reasons for their removal. This might result in less preferred beliefs being removed, and additional beliefs being returned. The end product is an optimization of the belief base, converting the results of a series of revisions to the very base that would have resulted from a batch revision performed after all base beliefs were entered/added. Reconsideration can be done by examining the entire set of all base beliefs (both currently believed and retracted) — or, if the believed base is consistent, by examining all retracted beliefs for possible return. This, however, is computationally expensive. We present a more efficient, TMS-friendly algorithm, dependency-directed reconsideration (DDR), which can produce the same results by examining only a dynamically determined subset of base beliefs that are actually affected by changes made since the last base optimization process. DDR is an efficient, anytime, belief base optimizing algorithm that eliminates operation order effects.

Introduction and Motivation

As a knowledge representation and reasoning (KRR) system gathers information to reason about, it has to update its belief space. When performing belief change operations — whether belief change operations are performed on logically closed theories (Alchourrón, Gärdenfors, & Makinson 1985) or finite bases (Nebel 1989; Hansson 1991), using ideal or resource-bounded agents (Wassermann 1999; Williams 1997)) — there is no doubt that the order of the operations performed usually affects the makeup of the current belief base, because of the effects of consistency maintenance.

New information can enter the system and might conflict with existing information. This can happen when multiple sources give contradictory information or when a single source changes his mind or simply because we are modeling a changing world.

As we will show, maintaining consistency as information is gathered (belief change in series) can result in a base that is less optimal than if consistency maintenance were put off until all the information was gathered (a batch operation). But, how can we know when *all* the information is gathered? And how can the system reason before the batch consistency-restoring operation is performed?

Some real world implementations only reason about a known fixed domain of possible base beliefs. We focus, instead, on a reasoning system that can receive new information and that needs to maintain a current base from which to reason. We are concerned with improving this kind of system so that it can

1. maintain a consistent base from which to reason
2. re-optimize its base after a series of belief changes.

We define *reconsideration*, which was first introduced in (Johnson & Shapiro 2004b), as a belief change operation that optimizes a finite belief base by eliminating the effects of the order in which belief change operations have been performed. As a result of reconsideration, beliefs retracted earlier might be recovered[1], and some currently believed (weaker) beliefs might be retracted. We then present an efficient algorithm for implementing reconsideration called dependency-directed reconsideration (DDR) (Johnson & Shapiro 2004a; 2004b).

Preliminaries

Foundations Approach

We assume that the KRR systems that will use reconsideration will follow a *foundations* approach to belief revision (Hansson 1999). That is, they distinguish between core beliefs (i.e. base beliefs, also called hypotheses in (Martins & Shapiro 1988)), which have independent standing, and derived beliefs. The set of base beliefs is the belief base and is assumed to be finite. We also assume the systems will use non-prioritized revisions (Hansson 1999).

[1]This is very different from the recovery of "retracted" beliefs during either saturated kernel contractions (Hansson 1999) or the second part of Hybrid Adjustment (Williams & Sims 2000). The similarity to belief liberation (Booth *et al.* 2003), is discussed later in this paper.

Notation and Terminology

For this paper, we use a propositional language, $\mathcal{L}$, which is closed under the truth functional operators $\neg, \vee, \wedge, \rightarrow$, and $\leftrightarrow$. Formulas of the language $\mathcal{L}$ are denoted by lowercase letters $(p, q, r, \ldots)$. Sets and sequences are denoted by uppercase letters $(A, B, C, \ldots)$.

A derives p is denoted as $A \vdash p$. *Cn* is defined by $Cn(A) = \{p \mid A \vdash p\}$, and $Cn(A)$ is called the *closure* of A.

A belief base B is consistent iff for every $p \in Cn(B), \neg p \notin Cn(B)$. In other words: B is consistent iff $B \nvdash \bot$, where $\bot$ denotes logical contradiction. The closure of a belief base is referred to as its theory or belief *space*.

Given a finite belief base, B, the set of *p-kernels* of B is the set $\{A \mid A \subseteq B, A \vdash p$ and $(\forall A' \subsetneq A) A' \nvdash p\}$ (Hansson 1999). The p-kernels actually used to derive p are called p's *origin sets* in (Martins & Shapiro 1988).

A minimally inconsistent subset of a set A is some $S \subseteq A$ s.t. $S \vdash \bot$, but for all $S' \subsetneq S$, $S' \nvdash \bot$. These sets are referred to as nogoods in the ATMS literature (de Kleer 1986; Forbus & de Kleer 1993), and we will adopt that term in this paper.

Assuming a Linear Ordering

The Linear Ordering We assume that there is a linear preference ordering ($\succeq$) over the beliefs in the current base. This ordering might change when new information is added to the base. Unless otherwise stated, we assume this ordering is recency-independent.

When $a \succeq b$ and $b \not\succeq a$, then $a \succ b$; and $a \succ b$ implies $a \succeq b$. Any base can be represented as a unique sequence of beliefs in order of descending preference: $B = p_1, p_2, \ldots, p_n$, where if $i \neq j$, then $p_i \neq p_j$ and $p_i \succ p_{i+1}$, $1 \leq i < n$ (p_i is preferred over p_{i+1} ; p_{i+1} is *weaker* than p_i). Equality over sequences is defined as $p_1, p_2, \ldots, p_n = q_1, q_2, \ldots, q_m$ when $n = m$ and $(\forall i, 1 \leq i \leq n) : p_i = q_i$.

Non-equal beliefs that are logically equivalent may have different preferences. They will, nevertheless, survive belief change operations, or not, together.

Measuring Preference Given a set of base beliefs $B^{\cup} = p_1, ..., p_n$, (ordered according to $\succeq$), a numerical value for credibility can be calculated from the preference ordering:

- $\text{Cred}(p_i, B^{\cup}, \succeq) = 2^{n-i}$
- For $B \subseteq B^{\cup}$,
  if $B \nvdash \bot$, then $\text{Cred}(B, B^{\cup}, \succeq) = \sum_{q \in B} \text{Cred}(q, B^{\cup}, \succeq)$,
  otherwise, $B \vdash \bot$, and $\text{Cred}(B, B^{\cup}, \succeq) = -1$.

Essentially, the credibility of a base is equivalent to the bit vector of its elements. We can define $\succeq_{B^{\cup}}$ to be a linear ordering over bases: $B \succeq_{B^{\cup}} B'$ if and only if $\text{Cred}(B, B^{\cup}, \succeq) \geq \text{Cred}(B', B^{\cup}, \succeq)$. Likewise, $B \succ_{B^{\cup}} B'$ if and only if $\text{Cred}(B, B^{\cup}, \succeq) > \text{Cred}(B', B^{\cup}, \succeq)$.

Optimal Base Given a possibly inconsistent set of base beliefs, $B^{\cup} = p_1, p_2, ..., p_n$, ordered by $\succeq$, the base B is considered *optimal* w.r.t. $B^{\cup}$ and $\succeq$ if and only if $B \subseteq B^{\cup}$ and $(\forall B' \subseteq B^{\cup}) : B \succeq_{B^{\cup}} B'$. This favors a single strong belief over multiple weaker beliefs.

Reconsideration results in a base that is optimal w.r.t. the current $B^{\cup}$ and the *current* ordering. Although requiring a linear ordering is unrealistic for real world applications, we use it, because it simplifies the algorithm explanations and proofs. These explanations and proofs are fully detailed in (Johnson 2005) — including a proof to show that the algorithms work with a pre-order that determines a unique weakest element in each nogood.

Operations on a Base

Introduction For this paper, we use Hansson's belief base operations of expansion, kernel consolidation, and kernel semi-revision (Hansson 1991; 1997; 1999).

Expansion $B + a$ (the expansion of the belief base B by the belief a) is defined as $B \cup \{a\}$.

Kernel Consolidation Briefly, $B!$ (the kernel consolidation of B) is the removal of at least one element from each nogood in B s.t. $B! \subseteq B$ and $B! \nvdash \bot$.

For this paper, we refer to the weakest belief in a nogood as *that* nogood's culprit, and the operation of selecting the beliefs to be removed from a base B to resolve an inconsistency is called *culprit selection* — selecting *which* culprits to remove. We assume this selection is determined by a global incision function (Hansson 1999) maximizing the value of $\text{Cred}(B!, B^{\cup}, \succeq)$:
$(\forall B' \subseteq B) : B! \succeq_B B'$ — consolidation of a base is the optimal *subset* of that base (w.r.t. B and $\succeq$). **(C1)**

Kernel Semi-Revision The kernel semi-revision of the base B by the belief a (written as $B *^{?} a$) is expansion by a followed by consolidation, where the consolidation operation might actually *remove* a: $B *^{?} a =_{def} (B + a)!$.

Kernel semi-revision is a form of non-prioritized belief change (Hansson 1999). This is in contrast to *prioritized* revision of B by a (written as $B * a$), which must adhere to the (Success) postulate: $a \in (B * a)$.

Contraction for Consistency-Maintenance Only For this paper, we will define reconsideration for a system that performs expansion, kernel consolidation and kernel semi-revision, but *not* user-initiated contraction (e.g. removing some belief p from a belief base/space even if there was no inconsistency) — thus, any removal of a base belief from the base is done solely to eliminate a contradiction. As explained in (Chopra, Georgatos, & Parikh 2001) "Agents do not lose beliefs without a reason: to drop [i.e. retract] the belief $\ldots p$ is to *revise* by some information that changes our reasoning." Similarly, reconsideration supports the idea that beliefs should not *remain* lost (retracted) without a reason.

Example of Operation Order Side-Effects

The examples below illustrate the effect that operation order can have on a belief base.[2] Given the base $B0 = \{s, d\}$ (also represented as a sequence: $B0 = s, d$), we show the

[2] These examples are based on the Cleopatra's children example mentioned in (Hansson 1991; 1999).

results of adding two more beliefs using semi-revision. The beliefs to be added are $\neg(s \vee d)$ and $s \vee d$, with the preference ordering: $s \vee d \succ \neg(s \vee d) \succ s \succ d$.

Example1: $B0 *^? \neg(s \vee d) = \{\neg(s \vee d)\}$, where the retraction of *both* s and d were required to maintain consistency. Subsequently, $\{\neg(s \vee d)\} *^? (s \vee d)$ forces $\neg(s \vee d)$ to be retracted. The resultant base is $B1 = \{s \vee d\}$.

Example2: If we reverse the order, however, of the two additions — i.e. first add $s \vee d$, then add $\neg(s \vee d)$ — the addition of $s \vee d$ would not force any retractions for consistency maintenance, and the following addition of $\neg(s \vee d)$ would "fail". First, $B0 *^? (s \vee d) = \{s \vee d, s, d\}$. Then, $\{s \vee d, s, d\} *^? \neg(s \vee d) = \{s \vee d, s, d\}$. The belief $\neg(s \vee d)$ would not be added to the base, because it directly contradicts the more preferred $s \vee d$. The resultant base would be $B2 = \{s \vee d, s, d\} = s \vee d, s, d$.

The two examples differ only in the order that the additions take place, yet the resulting base in Example1 ($B1$) is a proper subset of the optimal $B2$. It is easy to see that $B1$ is not optimal, because the return of both s and d would expand the base without introducing any inconsistency.

Note that $B2$ is the base that would result if both beliefs had been added (in any order) *before* consistency maintenance was triggered: $B2 = ((B0 + (s \vee d)) + (\neg(s \vee d)))! = ((B0 + \neg(s \vee d)) + (s \vee d))!$.

When reading about reconsideration in the next section, keep in mind that performing reconsideration at the end of Example 1 would return both s and d to the belief base. Reconsideration would not alter the base in Example 2, because that base is already optimal.

Reconsideration

We present the theory of reconsideration (Johnson & Shapiro 2004b)) as a method for achieving the optimal belief base by reconsidering past belief change operations in light of more recently acquired information. This requires maintaining a set of *all* beliefs (currently believed or not).

Let B_0 be a finite belief base. Let B_n be the belief base that results from a series of expansion and consolidation operations on B_0 (and the subsequent resulting bases: $B_1, B_2, B_3, \ldots$). We define $B^\cup = \bigcup_{0 \le i \le n} B_i$, with a current linear ordering $\succeq$. X_n is the set of base beliefs removed (and currently dis-believed: $B_n \cap X_n = \emptyset$) from these bases during the course of the series of operations: $X_n =_{def} B^\cup \setminus B_n$.

For the remainder of this paper, all triples are assumed to be of the form $\langle B, B^\cup, \succeq\rangle$, where $\succeq$ is the linear ordering of $B^\cup$, $X = B^\cup \setminus B$ and $Cn(\langle B, B^\cup, \succeq\rangle) = Cn(B)$.

Reconsideration of $\langle B, B^\cup, \succeq\rangle$ is written $\langle B, B^\cup, \succeq\rangle!\!\Downarrow$ and defined as : $\langle B, B^\cup, \succeq\rangle!\!\Downarrow \; =_{def} \; \langle B^\cup!, B^\cup, \succeq\rangle$.

Theorem 1 *The base resulting from reconsideration is optimal w.r.t.* $B^\cup$ *and* $\succeq$.
Proof: $\langle B, B^\cup, \succeq\rangle!\!\Downarrow \; =_{def} \; \langle B^\cup!, B^\cup, \succeq\rangle$. $(\forall B' \subseteq B^\cup)$: $B^\cup! \succeq_{B^\cup} B'$. (*from* **C1**) □

We also note that the result of reconsideration is unaffected by the currently believed base:

Observation 1 *Given that* B_{opt} *is the optimal base w.r.t.* $B^\cup$ *and* $\succeq$, $(\forall B \subseteq B^\cup) : \langle B, B^\cup, \succeq\rangle!\!\Downarrow = \langle B_{opt}, B^\cup, \succeq\rangle$.

Dependency-Directed Reconsideration

A naive implementation of reconsideration, by looking at all of $B^\cup$ and determining what should be retracted to establish consistency, becomes computationally more expensive as $B^\cup$ increases in size. Similarly, the more efficient algorithm (used if B is consistent) of examining elements in X to see if any can be returned to the base also becomes computationally more expensive as X and $B^\cup$ increase in size.

The best way to improve the implementation is to reduce the number of retracted base beliefs whose removal is to be reconsidered. This can be done by the process of dependency-directed reconsideration (DDR, briefly described in (Johnson & Shapiro 2004a; 2004b)) which uses the nogoods in $B^\cup$.

The DDR Process

If consolidation results in removing some base belief, p, from the base, this might allow a currently disbelieved *weaker* base belief, q, to return to the base — e.g. q can return if the reasons for disbelieving it depended on p being believed. Specifically, a belief can return if it either (1) does *not* raise an inconsistency or (2) if any inconsistencies raised can be resolved by retracting *only* weaker beliefs. DDR is the process of:

1. due to a retraction — determining which beliefs should be examined for *possible* return to the base and inserting them onto a global priority queue
2. for a retracted belief q which might be able to return to the base — determining whether q can return, and, if so, whether there is a set R of weaker conflicting beliefs in the base that needs to be removed from the base to maintain consistency
3. if some q can return — executing the simultaneous exchange of q into the base and R out

Because a retracted belief points to the removed beliefs to be examined for possible return to the base, DDR is "dependency-directed" (a phrase first coined in (Stallman & Sussman 1977)).

Defining a Knowledge State

A knowledge state is a five-tuple: $\langle B \; , \; B^\cup \; , \; I \; , \; \succeq \; , \; Q\rangle$, where :

- B is a set of base beliefs that are currently believed (the current base)
- $B^\cup$ is the set of *all* base beliefs, believed or not
- X is the set of currently disbelieved base beliefs. It is defined as $X =_{def} B^\cup \setminus B$ and is called the exbase.
- I is a set of all the nogoods in $B^\cup$.
  $I = \{S \mid S \subseteq B^\cup, S \vdash \bot, \text{and } (\forall S' \subsetneq S) : S' \nvdash \bot\}$
- $\succeq$ is a linear preference ordering on the beliefs in $B^\cup$
- Q is a sequence, possibly empty, of tagged beliefs $(\langle p_1, \tau_1\rangle, \langle p_2, \tau_2\rangle, \ldots, \langle p_n, \tau_n\rangle)$ s.t.

– $p_i \in X$
– $i \neq j \;\Rightarrow\; p_i \neq p_j$
– $p_i \succ p_{i+1}\ ,\ 1 \leq i < n$
– $\tau \in \{justout, in?, both\}$

Unless otherwise stated, we assume the notational default that (for $i \in \{0, 1, 2, \ldots\}$) any knowledge state referred to as KS_i is the five-tuple $\langle\ B_i\ ,\ B_i^{\cup}\ ,\ I_i\ ,\ \succeq_i\ ,\ Q_i\ \rangle$ and that $X_i = B_i^{\cup} \setminus B_i$.[3] For a knowledge state referred to as KS , the five-tuple elements are B, $B^{\cup}$, I, $\succeq$, and Q , respectively, and $X = B^{\cup} \setminus B$. This will remain unchanged for the duration of this paper. We also allow the shorthand expression $p \in Q$ to stand for $\exists \tau$ s.t. $\langle p, \tau \rangle \in Q$. Likewise, the deductive closure of a knowledge state is defined as the closure of its base: $Cn(KS) =_{def} Cn(B)$.

The information in I can be represented in a graph connecting base beliefs to the nogoods they are in (for an example, see Figure 1).

Functions for Q (1) Empty(Q) = *true* if Q is empty, else *false*. (2) First(Q) returns the first element in the priority queue (non-destructively), unless Empty(Q), in which case it returns *false*. (3) Rest(Q) returns a priority queue just like Q but without its first pair (non-destructively). If Empty(Q), it returns *false*.

Knowledge State Preference Ordering Given a knowledge state, $KS = \langle B, B^{\cup}, I, \succeq, Q \rangle$, if First($Q$) = $\langle p, \tau \rangle$, then QImpact(KS) = Cred($p, B^{\cup}, \succeq$). If Empty(Q), QImpact(KS) = 0. If Cred($B, B^{\cup}, \succeq$)=-1, then $\nexists KS'$ s.t. $KS \succeq_{KS} KS'$. Otherwise, given the knowledge state $KS_1 = \langle B_1, B^{\cup}, I, \succeq, Q_1 \rangle$:

- If Cred($B, B^{\cup}, \succeq$) = Cred($B_1, B^{\cup}, \succeq$) and QImpact(KS) = QImpact(KS_1), then $KS \succeq_{KS} KS_1$
- $KS \succ_{KS} KS_1$ if either
  – Cred($B, B^{\cup}, \succeq$) > Cred($B_1, B^{\cup}, \succeq$) or
  – Cred($B, B^{\cup}, \succeq$) = Cred($B_1, B^{\cup}, \succeq$) and QImpact(KS) < QImpact(KS_1).
- If $KS \succ_{KS} KS_1$, then $KS \succeq_{KS} KS_1$.

Since QImpact(KS) is determined by First(Q), different queues with identical first elements will have the same impact. Therefore, $\succeq_{KS}$ is *not linear*.

A knowledge state KS is optimal iff Empty(Q) and B is optimal w.r.t. $B^{\cup}$ and $\succeq$.

Knowledge State Belief Change Operations

Introduction The operations that can be performed on (and change) a knowledge state, $KS = \langle B, B^{\cup}, I, \succeq, Q \rangle$, are addition, consolidation, and reconsideration.

Knowledge State Addition $KS +_! \langle p, \succeq_p \rangle$ is adding p to KS — where the belief p (along with necessary preference ordering information regarding that belief, $\succeq_p$) is added to the knowledge state (using semi-revision (Hansson 1997)) The result is a new knowledge state, $KS_1 = \langle B_1, B_1^{\cup}, I_1, \succeq_1, Q_1 \rangle$, where

- $B_1 = B *^? p$ Note:p is not *guaranteed* to be in B_1
- $B_1^{\cup} = B^{\cup} \cup \{p\}$
- $X_1 = B_1^{\cup} \setminus B_1$
- $I_1 = \{N \mid N \subseteq B_1^{\cup}$ and N is a nogood $\}$
- $\succeq_1$ is $\succeq$ adjusted to include the preference information $\succeq_p$ — which positions p relative to other beliefs in $B^{\cup}$, while leaving the order of other beliefs in $B^{\cup}$ unchanged. The resulting ordering is the transitive closure of these orderings.[4] If this re-ordering results in altering the culprit for a nogood, a corresponding change is reflected in Q_1, see below.
- Q_1 is similar to Q with the following changes: (1) if p was in Q and is now in B_1, then it is *not* in Q_1; (2) $(\forall b \in B)$ if $b \in X_1$, then $\langle b, justout \rangle \in Q_1$; (3) if $p \in B$ and, after the re-ordering, p is a *new* culprit for an existing nogood, then, if the previous culprit is in X, it is now in Q_1 with a tag of *in?* .

Knowledge State Consolidation $KS!$ is consolidation of KS which produces $KS_2 = \langle B!, B^{\cup}, I, \succeq, Q_2 \rangle$, where Q_2 is similar to Q adjusted so that: $(\forall p)$ If $p \in B$ and $p \notin B_2$, then $\langle p, justout \rangle \in Q_2$.

Reconsideration $KS_{\Downarrow}$ is reconsideration of KS which produces $KS_3 = \langle B^{\cup}!, B^{\cup}, I, \succeq, Q_3 \rangle$, where Q_3 is empty.

DDR Algorithms

In the following algorithms for implementing DDR, all changes are to local variables except for line 18 of DDR, where the current knowledge state gets updated with the latest improvement towards optimality.

DDR-Q-Insert($p, \succeq, Q$) inserts the belief p into the queue and adjusts its tag to indicate that it should be considered for possible return to a base. The resulting tag will depend on whether p is already in the queue Q and, if so, with what tag.

DDR-Q-Insert($p, \succeq, Q$) returns a priority-queue (Q') like Q, where:
if $\langle p, in? \rangle \in Q$, **then** $Q' = Q$
if $\langle p, both \rangle \in Q$, **then** $Q' = Q$
if $\langle p, justout \rangle \in Q$, **then** $Q' = Q$ with the tag for p is changed from $justout$ to $both$.
if $p \notin Q$, **then** $Q' = Q$ with $\langle p, in? \rangle$ inserted into it.

Safe-Return (p, KS) evaluates whether a belief, p, can be returned to the current belief base (B). A belief can return to the base if either (1) it does *not* raise an inconsistency or (2) every inconsistency its return would trigger can be resolved through the removal of weaker beliefs. This function returns a pair: a boolean and a set of beliefs (R). If p *cannot* return to the base, the boolean is **false** and $R = \emptyset$. If p *can* return to the base, the boolean is **true** and R contains the beliefs to be removed from the base for consistency maintenance — it is possible for R to be empty. Preconditions: $B \nvdash \bot$ and $p \notin B$. Postconditions if p cannot return: $(\forall B' \subseteq B)$:

[3] For example: $KS_1 = \langle\ B_1\ ,\ B_1^{\cup}\ ,\ I_1\ ,\ \succeq_1\ ,\ Q_1\ \rangle$.

[4] We assume that if $p \in B^{\cup}$, the location of p in the sequence might change — i.e. its old ordering information is removed before adding $\succeq_p$ and performing closure — but all other beliefs remain in their same relative order.

If $((B \cup \{p\}) \setminus B') \not\vdash \bot$, then $B \succ_{B^\cup} ((B \cup \{p\}) \setminus B')$. Postconditions if p can return: $((B \cup \{p\}) \setminus R) \not\vdash \bot$, $((B \cup \{p\}) \setminus R) \succ_{B^\cup} B$, and $(\forall r \in R) : r \in B,\ p \succ r$, and $(((B \cup \{p\}) \setminus R) \cup \{r\}) \vdash \bot$.

KS-Add-Remove(KS, p, R) returns a knowledge state KS_1 similar to KS but with belief p returned to the base, belief set R removed from the base, and the priority queue adjusted to reflect these changes. Preconditions: $B \not\vdash \bot, R \subseteq B, p \notin B, p \in Q$ and $(\forall r \in R) : p \succ r$. Postconditions: $p \in B_1,\ R \subseteq X_1,\ B_1 \not\vdash \bot,\ p \notin Q_1, KS_1 \succ_{KS} KS$ and $(\forall r \in R) : p \succ r, (B_1 \cup \{r\}) \vdash \bot$ and $\langle r, justout \rangle \in Q_1$.

Update-KS-Justout(KS, p) returns a knowledge state KS_1 like input KS with its queue updated s.t. $p \notin Q_1$ and all disbelieved culprit nogood-mates of p are inserted into the queue by DDR-Q-Insert. Preconditions: $B \not\vdash \bot, p \in X$, and $p \in Q$ with tag of *justout* or *both*. Postconditions: $p \notin Q_1$, $KS_1 \succ_{KS} KS$ and $(\forall q \in X) :$ If $p \succ q$ and $(\exists N \in I)$ s.t. $\{q, p\} \subseteq N$, then $q \in Q$ with a tag of *in?* (or *both*) as determined by DDR-Q-Insert.

DDR Algorithm

DDR(KS) takes the system knowledge state (KS) as input, and improves it incrementally (with each pass throught the loop), until it is optimal w.r.t. $B^\cup$ and $\succeq$. $B^\cup, I$,and $\succeq$ are unaffected by DDR. Preconditions: $B \not\vdash \bot$. Postconditions: Let KS_{fin} be the system knowledge state. $B_{fin} \not\vdash \bot$ and Empty(Q_{fin}). Loop conditions: At line 2: KS'_{top} equals the system knowledge state. After line 18 : KS'_{bot} equals the system knowledge state. For each pass: $KS'_{bot} \succ KS'_{top}$.

procedure DDR(KS)
1 $KS' \leftarrow \langle B, B^\cup, I, \succeq, Q \rangle \leftarrow KS$
2 **loop until** Empty(Q)
3 $\langle p, \tau \rangle \leftarrow$ First(Q)
4 **if**$_1$ (τ = *in?* or τ = *both*) , **then**
5 $\langle$ *can-return*, $R\rangle \leftarrow$ Safe-Return(p, KS')
6 **if**$_2$ *can-return* , **then**
7 $KS' \leftarrow$ KS-Add-Remove(KS', p, R)
8 **else**$_2$
9 **if**$_3$ τ = *both* , **then**
10 $KS' \leftarrow$ Update-KS-Justout(KS', p)
11 **else**$_3$
12 $Q \leftarrow$ Rest(Q)
13 **endif**$_3$
14 **endif**$_2$
15 **else**$_1$
16 $KS' \leftarrow$ Update-KS-Justout(KS', p)
17 **endif**$_1$
18 $KS \leftarrow KS'$;;; DESTRUCTIVE
19 **end loop**

Note that the loop in the DDR algorithm works its way through the dynamically changing elements of the priority queue. Each element is processed in turn and how it is processed depends of its tag, τ.

If $\tau = justout$, then the belief was retracted and the DDR process will insert onto the queue the appropriate beliefs to be considered for possible return to the base. If $\tau = in?$, then DDR determines whether the belief can return to the base and, if so, returns it (with appropriate consistency maintenance). If $\tau = both$, then the belief is first processed as if its tag is $in?$ — only if the belief cannot return to the base will it then be processed as if it has the tag $justout$.

Theorem 2 *Given a knowledge state KS and DDR(KS) run to completion results in the knowledge state KS_{fin}, KS_{fin} is optimal (w.r.t. $B^\cup$ and $\succeq$ of KS).*

The proof for this theorem is long (and is given in (Johnson 2005)), but it can be shown that: $\forall p \in X_{fin} : \exists N \in I_{fin}$ s.t. (1) $(\forall q \in N)\ q \succeq p$ and (2) $(N \setminus \{p\}) \subseteq B_{fin}$. This equates to optimality.

DDR Example

Example3 Using $B^\cup$ from Figure 1, consider a knowledge state KS_1 where $B_1^\cup = B^\cup \setminus \{\neg p\}$, $\succeq_1$ is like $\succeq$ without the preference information for $\neg p$, and Empty(Q_1). The optimal base w.r.t. $B_1^\cup$ and $\succeq_1$ would be $B_1 = p, p{\rightarrow}q, p{\rightarrow}r, m{\rightarrow}r, s, s{\rightarrow}t, w{\rightarrow}v, w{\rightarrow}k, p{\rightarrow}v, z{\rightarrow}v, n, w, m, z$. $KS_1 *^? \langle \neg p, \succeq_{\neg p} \rangle$ (where $\succeq_{\neg p} = \neg p \succ p$) now forces the retraction of p to form KS_2 whose base is $B_2 = \neg p, p{\rightarrow}q, p{\rightarrow}r, m{\rightarrow}r, s, s{\rightarrow}t, w{\rightarrow}v, w{\rightarrow}k, p{\rightarrow}v, z{\rightarrow}v, n, w, m, z$.

Most systems stop here (with a sub-optimal base) because they focus on maintaining consistency but do not review previous belief change operations to see if they could be improved using hindsight. The set of retracted base beliefs (X_2) is $\{p, \neg q, \neg r, \neg v, \neg t, \neg k\}$, $B_2^\cup = B^\cup$ (from Figure 1), and $\succeq_2 = \succeq$.

The optimal base w.r.t. $B^\cup$ and $\succeq$, would be: $B = \neg p, p \rightarrow q, p \rightarrow r, m \rightarrow r, s, s \rightarrow t, w \rightarrow v, w \rightarrow k, p \rightarrow v, z \rightarrow v, n, \neg q, \neg r, w, z$.

DDR can be performed on KS_2 ($KS_2{}^\cup_\downarrow$) to produce this optimal base. When $KS_1 *^? \langle \neg p, \succeq_{\neg p} \rangle$ produces KS_2, B_2 is as described above, and Q_2 contains the single pair $\langle p, justout \rangle$. DDR on KS_2 goes through the following steps:

1. Process $\langle p, justout \rangle$. Result: $KS_3 = \langle B_2, B_2^\cup, I, \succeq, Q_3 \rangle$, where $Q_3 = \langle \neg q, in? \rangle, \langle \neg r, in? \rangle, \langle \neg v, in? \rangle$
2. Process $\langle \neg q, in? \rangle$. Result: $KS_4 = \langle (B_2 \cup \{\neg q\}), B_3^\cup, I, \succeq, Q_4 = \langle \neg r, in? \rangle, \langle \neg v, in? \rangle \rangle$
3. Process $\langle \neg r, in? \rangle$. Result: $KS_5 = \langle (B_4 \cup \{\neg r\}) \setminus \{m\}, B_4^\cup, I, \succeq, Q_5 \rangle$, where $Q_5 = \langle \neg v, in? \rangle, \langle m, justout \rangle$. Note: $\neg r \succ m$.
4. Process $\langle \neg v, in? \rangle$. Result: $KS_6 = \langle B_5, B_5^\cup, I, \succeq, Q_6 = \langle m, justout \rangle \rangle$.
5. Process $\langle m, justout \rangle$. Result $KS_7 = \langle B_6, B_6^\cup, I, \succeq, Q_7 \rangle$, where Empty($Q_7$) and $B_7 = B$, which is optimal w.r.t. $B^\cup$ and $\succeq$. Note: $B^\cup = B_7^\cup$.

In step 3, $\neg r$ returns to the base with the simultaneous removal of m (for consistency), because $\neg r \succ m$ (Cred($B_5, B^\cup, \succeq$) >Cred($B_4, B^\cup, \succeq$)). In step 4, once DDR determines that $\neg v$ cannot return to the base (due to its being the culprit for the nogood $\{w \rightarrow v, w, \neg v\}$), it would prune off the examination of the nogood containing z, and the set containing $\neg k$ would also not be examined or affected. This is an example of how the nogoods examined

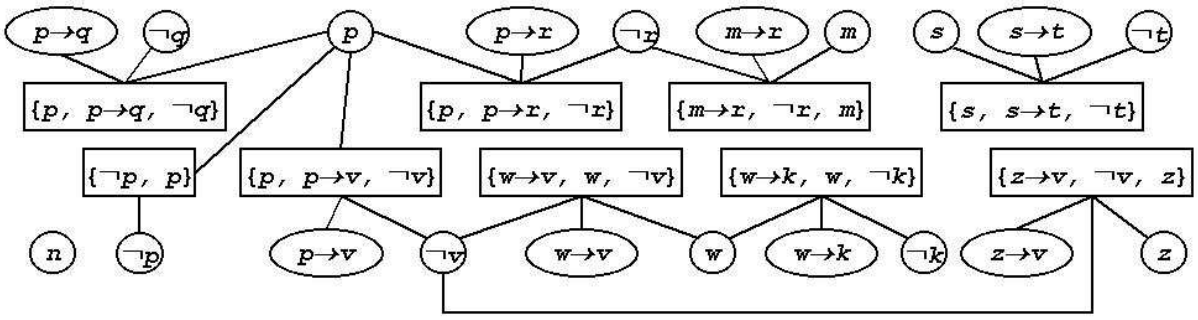

Figure 1: A graph showing the elements of $B^{\cup}$ (circles/ovals) of a knowledge state, $KS = \langle B, X, I, \succeq, Q \rangle$, connected to their nogoods (rectangles), where $B^{\cup} = \neg p, p, p{\rightarrow}q, p{\rightarrow}r, m{\rightarrow}r, s, s{\rightarrow}t, w{\rightarrow}v, w{\rightarrow}k, p{\rightarrow}v, z{\rightarrow}v, n, \neg q, \neg r, w, \neg v, m, z, \neg t, \neg k$ (in decreasing order of preference w.r.t. $\succeq$).

during DDR are determined dynamically. The nogood containing s would also be ignored by DDR, because it is not connected to p in any way. This last case is representative of the possibly thousands of unrelated nogoods for a typical belief base which *would* be checked during a naive operation of reconsideration, but are ignored by DDR.

Discussion[5]

Benefits of DDR

DDR Produces Optimality If run to completion, DDR(KS) makes the knowledge state optimal. (Th. 2)

DDR is an Anytime Algorithm The key elements of an anytime algorithm (Dean & Boddy 1988) are that (1) the answer is available at any point and (2) the quality of the answer must improve as a function of time.

DDR starts with the *current* knowledge state (with a consistent base). With each pass through the algorithm loop, the knowledge state for the system gets updated with any changes that the DDR algorithm makes (line 18). If DDR is halted, the system knowledge state is the same as it was at the start of that single loop pass — incorporating all the improvements of the previous passes through the DDR loop. Thus, the answer — the most recent answer — is available *at any point* (1).

Let $KS'_{top} = KS'$ at line 2. Let $KS'_{bot} = KS'$ at line 18. Changes to KS' occur at lines 7, 10, 12, or 16, exclusively. For each pass through the DDR loop: $KS'_{bot} \succ KS'_{top}$. Therefore, (2) is satisfied. Additionally, these incremental improvements towards optimality are both measurable and recognizable — desirable attributes for an anytime algorithm mentioned in (Zilberstein 1996).

DDR can be restarted at any time Whenever DDR is stopped/halted, the system is left with a knowledge state KS who's Q has been properly maintained. Performing DDR on KS will still result in an optimal knowledge state. NOTE: Even if semi-revision is performed on the knowledge state multiple times since DDR was stopped, recalling DDR and running it to completion will still result in an optimal state (optimal w.r.t. the *current* $B^{\cup}$ and ordering).

DDR offers diminishing returns with time With each pass through the DDR loop, the changes to the base/exbase involve less and less preferred beliefs, because any beliefs more credible than the first belief in Q will (for these $B^{\cup}$ and $\succeq$) remain in their current set (B or X). Diminishing returns is another attribute mentioned in (Zilberstein 1996).

Adding DDR to an Existing TMS Making an existing truth maintenance system[6] DDR-capable involves very little additional computational load. The reasoning system is unaffected. The contradiction handling system must add a few steps: maintaining the priority queue and the detected nogoods with links to and from their elements.

We assume the TMS already has some technique for determining the culprit in each nogood; we merely require that the technique is consistent (resulting in a partial order over the culprits). Any arbitrary linear ordering of all base beliefs that is consistent with this pre-established partial order will work with the DDR algorithms.

Recall that DDR can be started, stopped, and continued at any time. If DDR is performed only during "down times" (times when reasoning and belief change are not being performed) there will be no perceivable effect — except that of reasoning with a more optimal base.

When DDR *must* be performed prior to executing any further reasoning or belief change operations, then the cost of DDR is obviously preferred over that of an incorrect base, and DDR is computationally less expensive than a naive batch consolidation over *all* base beliefs in an effort to determine *the* optimal base.

Reconsideration for Non-TMS Implementations Implementing DDR in a system that does not already detect nogoods might be computationally expensive. These systems

[5] For full discussion and proofs, see (Johnson 2005).

[6] By truth maintenance system, we are referring to systems that already compute nogoods: such as JTMS, ATMS and LTMS as discussed in (Forbus & de Kleer 1993) and the system described in (Martins & Shapiro 1988).

choose not to compute nogoods for some reason that probably includes the computational load, and to add nogood detection and maintenance would probably be disadvantageous. These systems may operate with more restricted domains or logics, however, where a naive implementation of reconsideration would be helpful (provided the computational load is not prohibitive).

The advantage of DDR for default logic systems is still under exploration, but if a default logic system has a section of its base that contains monotonic beliefs, DDR could be helpful in maintaining the optimality of that section.

Related Work: Belief Liberation

Introduction Our $B^{\cup}$ is similar to the *belief sequence*, σ, used in the definition of a retraction operator, called σ-*liberation*, that can result in belief liberation (Booth *et al.* 2003). Belief liberation occurs when the retraction of a belief from a base removes the conflict that forced the prior retraction of some weaker belief. In this case, the weaker belief can return to the base — it is "liberated" — as a part of that retraction operation.

σ is a linear sequence of beliefs $(p_1, \ldots, p_n)$ ordered by recency,where p_n is the most recent information the agent has received (and has highest preference). The set $[[\sigma]]$ is the set of all the sentences appearing in σ. Two immediate differences between our $B^{\cup}$ and liberation's σ are (1) our ordering is recency-independent, whereas theirs *is* recency and (2) the two orderings are in reverse order: the last element of σ has highest preference, whereas our first element ranks highest. The second difference is superficial, because both sequences are traversed from strongest belief to weakest when determining the optimal base for that sequence.

Similarities Since both research groups assume a linear ordering, if we assume that $B^{\cup} = [[\sigma]]$ and we order $B^{\cup}$ by recency, our optimal base (w.r.t. $B^{\cup}$ and the ordering) would be equivalent to the base associated with σ. Likewise, the belief space that is associated with σ, which we call K_{σ}, is equivalent to $Cn(B^{\cup}!)$.

We define σ-addition (adding a belief to σ) as follows: $\sigma+p$ is adding the belief p to the sequence $\sigma = p_1, \ldots, p_n$ to produce the new sequence $\sigma_1 = p_1, \ldots, p_n, p$. This is also the technique described in (Chopra, Georgatos, & Parikh 2001). Now, given the knowledge state $KS = \langle B, [[\sigma]], I \succeq, Q\rangle$, where $\succeq$ is a linear ordering based on *recency*, we can say that $Cn(\ (KS + \langle p, \succeq_p\rangle)\!\downarrow\ ) = K_{\sigma+p}$.

Key Difference The research in belief liberation focuses on defining the retraction operation of σ-liberation for some belief set K relative to some *arbitrary* σ. The focus is on K and how it changes when a retraction is performed — i.e. if the retraction is equivalent to an operation of σ-liberation for *some* σ s.t. $K = K_{\sigma}$. The authors do not advocate maintaining any one specific base belief sequence σ.

We are focusing on optimizing the belief base (and corresponding belief space) following some series of belief change operations. We do not (in this paper) define any operation of retraction, but assume retraction is for consistency maintenance only (as a part of semi-revision) where the belief that is retracted is determined by the incision function during the consolidation operation.

DDR as a CSP

DDR can be framed as a boolean Constraint Satisfaction Problem (CSP) to obtain a knowledge base $KS = \langle B, B^{\cup}, I, \succeq, Q\rangle$ (Russell & Norvig 2003). The beliefs in $B^{\cup}$ are the *variables* of the CSP. Their *domain* of possible values consists of: *true* (in the base) and *false* (in the exbase). The nogoods represent *multiary hard constraints*, and Figure 1 is a *constraint hyper-graph*.

There are two ways to optimize the result:
(1) Add Cred$(B, B^{\cup}, \succeq)$ as an *objective function*, which must be maximized.
(2) Add a *hierarchy* of *unary constraints*: $(\forall p \in B^{\cup})$: $p = true$, where the ordering of these constraints is identical to the ordering of the beliefs in $B^{\cup}$—$(\forall p_i, p_j \in B^{\cup})$: $p_i = true \succ p_j = true$ iff $p_i \succ p_j$.

When new beliefs are added to the knowledge state, we then have an *iterated* CSP.

The Recovery Aspect of DDR

The Recovery postulate for belief theories states that $K \subseteq (K \sim p) + p$ where "$\sim$" is theory contraction and "$+$" is theory expansion (Alchourrón, Gärdenfors, & Makinson 1985). Recovery does not hold for bases, because returning a previously removed belief to a base does not guarantee that the kernels for that belief (which would also have been removed) will also be returned to the base (or its belief space).

Reconsideration (including DDR) adheres to the following recovery-like axiom:
If $\neg p \notin Cn(KS)$, then $B_{KS} \subseteq B_{KS2}$, where $KS2 = ((KS + \langle \neg p, \succeq_{\neg p}\rangle)\!\downarrow + \langle p, \succeq_p\rangle)\!\downarrow$ and $\succeq_p$ indicates $p \succ \neg p$.[7]

For a more detailed discussion of the Recovery aspect of reconsideration, refer to (Johnson & Shapiro 2005).

Conclusions and Future Work

Reconsideration is a belief base optimizing operation that eliminates operation order effects. Implementing reconsideration in any system offers a mechanism for optimizing both the belief base and the belief space. The computational costs need to be weighed against the other needs of the system.

DDR is an efficient, anytime, TMS-friendly algorithm that implements reconsideration. Implementing DDR in a TMS-style system allows optimization of the base in a computationally friendly way:

- optimization can yield to urgent reasoning demands
- DDR optimizes the most important parts of the base first
- DDR can be performed in multiple stages that can interleave with the addition of new information.

There no longer needs to be a trade-off between having a consistent base ready for reasoning and having an optimal base (by delaying consistency maintenance until all information is gathered). DDR also provides an easy test for when

[7] This is similar to (R3) in (Chopra, Ghose, & Meyer 2002).

optimization may be needed: when the priority queue is not empty.

Ongoing work involves dealing with conditions that are less than ideal:

- working with a non-linear ordering
- dealing with nogood sets that do not have a unique weakest element
- dealing with a non-ideal reasoning systems, specifically ones that are not guaranteed explicitly know every nogood in $B^{\cup}$.

We are also implementing DDR into an existing TMS system, SNePS (Shapiro & The SNePS Implementation Group 2004).

Acknowledgments

The authors are grateful for the insights and feedback of William J. Rapaport, Carl Alphonce, Ken Regan, David R. Pierce, Bharat Jayaraman, Doug Lenat, Samir Chopra, Jean-Pierre Koenig, Erwin Segal, Jan Chomicki, John Santore and the SNePS Research Group.

References

Alchourrón, C. E.; Gärdenfors, P.; and Makinson, D. 1985. On the logic of theory change: Partial meet contraction and revision functions. *The Journal of Symbolic Logic* 50(2):510–530.

Booth, R.; Chopra, S.; Ghose, A.; and Meyer, T. 2003. Belief liberation (and retraction). In *TARK '03: Proceedings of the 9th conference on Theoretical aspects of rationality and knowledge*, 159–172. ACM Press. Full version in *Studia Logica* 79(1):47-72, 2005.

Chopra, S.; Georgatos, K.; and Parikh, R. 2001. Relevance sensitive non-monotonic inference on belief sequences. *Journal of Applied Non-Classical Logics* 11(1-2):131–150.

Chopra, S.; Ghose, A.; and Meyer, T. 2002. Iterated revision and recovery: a unified treatment via epistemic states. In van Harmelen, F., ed., *ECAI 2002: 15th European Conference on Artificial Intelligence*, number 77 in Frontiers in Artificial Intelligence and Applications, 541–545. Amsterdam, The Netherlands: IOS Press.

de Kleer, J. 1986. An assumption-based truth maintenance system. *Artificial Intelligence* 28(2):127–162.

Dean, T., and Boddy, M. 1988. An analysis of time-dependent planning. In *Proc. of AAAI-88*, 49–54.

Forbus, K. D., and de Kleer, J. 1993. *Building Problem Solvers*. Cambridge, MA: MIT Press.

Hansson, S. O. 1991. *Belief Base Dynamics*. Ph.D. Dissertation, Uppsala University.

Hansson, S. O. 1997. Semi-revision. *Journal of Applied Non-Classical Logic* 7:151–175.

Hansson, S. O. 1999. *A Textbook of Belief Dynamics*, volume 11 of *Applied Logic*. Dordrecht, The Netherlands: Kluwer.

Johnson, F. L., and Shapiro, S. C. 2004a. Dependency-directed reconsideration. In K. Forbus, D. Gentner, T. R., ed., *Proceedings of the 26th Annual Meeting of the Cognitive Science Society, CogSci2004*, 1573. Mahwah, NJ: Lawrence Erlbaum Assoc.

Johnson, F. L., and Shapiro, S. C. 2004b. Knowledge state reconsideration: Hindsight belief revision. In *Proceedings of the Nineteenth National Conference on Artificial Intellicence (AAAI-2004) and Sixteenth Conference on Innovative Applications of Artificial Intelligence(IAAI-2004)*, 956–957. Menlo Park, CA: AAAI Press (http://www.aaai.org/).

Johnson, F. L., and Shapiro, S. C. 2005. Improving recovery for belief bases. In Morgenstern, L., and Pagnucco, M., eds., *Proceedings of the Sixth Workshop on Nonmonotonic Reasoning, Action and Change*, in press. Edinburgh, Scotland: IJCAII.

Johnson, F. L. 2005. *Belief Change in a Deductively Open Belief Space*. Ph.D. Dissertation, Department of Computer Science and Engineering, University at Buffalo, The State University of New York, Buffalo, NY. Forthcoming.

Martins, J. P., and Shapiro, S. C. 1988. A model for belief revision. *Artificial Intelligence* 35:25–79.

Nebel, B. 1989. A knowledge level analysis of belief revision. In Brachman, R. J.; Levesque, H. J.; and Reiter, R., eds., *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning (KR'89)*, 301–311.

Russell, S., and Norvig, P. 2003. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Englewood Cliffs, NJ, 2nd edition edition. http://aima.cs.berkeley.edu/.

Shapiro, S. C., and The SNePS Implementation Group. 2004. *SNePS 2.6.1 User's Manual*. Department of Computer Science and Engineering, State University of New York at Buffalo, Buffalo, NY. Available as http://www.cse.buffalo.edu/sneps/Manuals/manual261.ps.

Stallman, R., and Sussman, G. J. 1977. Forward reasoning and dependency-directed backtracking in a system for computer-aided circuit analysis. *Artificial Intelligence* 9(2):135–196.

Wassermann, R. 1999. *Resource-bounded Belief Revision*. Ph.D. Dissertation, University of Amsterdam, Amsterdam, The Netherlands.

Williams, M.-A., and Sims, A. 2000. SATEN: An object-oriented web-based revision and extraction engine. In Baral, C., and Truszyński, M., eds., *Proceedings of the 8th International Workshop on Non-Monotonic Reasoning NMR'2000*. CoRR article: cs.AI/0003059.

Williams, M.-A. 1997. Anytime belief revision. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*. San Mateo, CA: Morgan Kaufmann. 74–79.

Zilberstein, S. 1996. Using anytime algorithms in intelligent systems. *AI Magazine* 17(3):73–83.