CMU CS 83 126

The Ins and Outs of Reason Maintenance

510.7808
C28r
83-126
c.3

Jon Doyle

Computer Science Department
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213
U.S.A.

Abstract: To progress significantly beyond current reason maintenance systems, we must formulate their structure and intended behavior precisely enough to analyze computational complexities and tradeoffs independently of the current set of limited implementation proposals. This paper summarizes one such formulation, and indicates some unsolved practical problems for investigation in future reason maintenance systems.

This paper will be presented at IJCAI-83.

This research was supported by the Defense Advanced Research Projects Agency (DOD), ARPA Order No. 3597, monitored by the Air Force Avionics Laboratory under Contract F33615-81-K-1539. The views and conclusions contained in this document are those of the author, and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the Government of the United States of America.

§**1.** Reason maintenance systems (a less deceptive name than the original "truth maintenance systems") have been studied by a variety of authors and have gained currency in artificial intelligence in spite of rather unwieldy descriptions in terms of complex procedures. (See [DOYLE 1979], [STALLMAN AND SUSSMAN 1977], [LONDON 1978], [MCALLESTER 1980], [CHARNIAK, RIESBECK, AND MCDERMOTT 1980], [THOMPSON 1979], [GÄRDENFORS 1980], [STEELE 1980], [DE KLEER AND DOYLE 1982], [MCDERMOTT 1982], [GOODWIN 1982], and [MARTINS 1983].) There is little hope for improving on existing RMS implementations without clearer statements of their intended behaviors and better analyses of their performance (see [MARTINS 1983]). These goals require mathematical formulations that clearly capture our intuitions, formulations that enable calculation of complexity-theoretic tradeoffs to guide and justify choices of implementation. [DOYLE 1982] develops exact theories and initial analyses of several issues arising in reason maintenance systems and in "non-monotonic logic." This paper summarizes portions of those theories and some of the questions they raise.

§**2.** Reason maintenance systems revise database states using records of inferences or computations, records called *reasons* or *justifications*, to trace the consequences of initial changes. By keeping track of what information has been computed from what, such a system can reconstruct the information "derivable" from given information. Although it is often convenient to think of such bits of information as beliefs and such derivations as arguments, the notion is much more general, applicable instead to all sorts of mental structures. For concreteness, we focus on RMS, the particular reason maintenance system developed by the author [DOYLE 1979]. The following treatment does not require intimate familiarity with any reason maintenance system, although in the interest of brevity we omit motivating discussion and examples, so passing acquaintance with the basic idea stated above is valuable. Here we only mention that, contrary to the impression held by some, RMS does not maintain consistency of beliefs in any important sense. RMS ensures that the set of "beliefs" held are acceptable with respect to the justifications held, or in other, looser terminology, that the assumptions made by the system are consistent with the justifications guiding their adoption. Logical notions of consistency play no role here, as the following treatment illustrates. The mistaken impression may stem from the contrasting importance of logical consistency in "non-monotonic logic" [MCDERMOTT AND DOYLE 1980]. (See [DOYLE 1982] for further discussion.)

§**3.** States of RMS contain a variety of elements or components. We let $\mathcal{D}$ denote the domain of all possible elements of states, so that states of RMS are sets $S \subseteq \mathcal{D}$. Not every subset of $\mathcal{D}$ is "admissible" as a state of RMS. We define which sets are so in a moment.

Let $\mathcal{N}$ be a finite set of elements called *nodes*. These are the fundamental components of states of RMS. Nodes are usually used to represent (within RMS) the database elements (beliefs, desires, rules, procedures, etc.) of significance to the external system using RMS, but we ignore those external meanings here since they have no bearing on the operation of RMS. Each set $\mathcal{N}$ generates a state-domain $\mathcal{D}(\mathcal{N})$ as follows. We define

$$\begin{aligned} SL(\mathcal{N}) &= \mathbf{P}\,\mathcal{N} \times \mathbf{P}\,\mathcal{N} \times \mathcal{N} \\ CP(\mathcal{N}) &= SL(\mathcal{N}) \times \mathcal{N} \\ \mathcal{D}(\mathcal{N}) &= \mathcal{N} \cup SL(\mathcal{N}) \cup CP(\mathcal{N}). \end{aligned}$$

(**P** means power set.) The elements of $SL(\mathcal{N})$ are called *SL-justifications* (for "support list"), and are written $A \parallel B \Vdash c$ for $A, B \subseteq \mathcal{N}$ and $c \in \mathcal{N}$. The elements of $CP(\mathcal{N})$ are called *CP-justifications* (for "conditional proof"), and are written $(A \parallel B \Vdash c) \Vdash d$ for $A, B \subseteq \mathcal{N}$ and $c, d \in \mathcal{N}$. SL-justifications will be interpreted as rules for making "non-monotonic" inferences, and CP-justifications as arguments based on "conditional-proofs" of nodes. The formal definitions of these terms follow. Note that if $\mathcal{N} \subseteq \mathcal{N}'$, then $\mathcal{D}(\mathcal{N}) \subseteq \mathcal{D}(\mathcal{N}')$. In fact, all of our definitions will be conservative in the sense that additions to the set of nodes do not change the meaning of previous conclusions or representations in important ways. Henceforth we hold the set of nodes constant and ignore the dependence of the domain on the set of nodes, writing simply $\mathcal{D}$ instead of $\mathcal{D}(\mathcal{N})$.

§**4.** Each state component is interpreted as a restriction on the states in which it may admissibly occur. Formally, we define an interpretation function $\mathcal{I} : \mathcal{D} \rightarrow \mathbf{P}\,\mathbf{P}\,\mathcal{D}$, where if $d \in \mathcal{D}$, then $\mathcal{I}(d) \subseteq \mathbf{P}\,\mathcal{D}$ is the set of potential states sanctioned by d. The set $\mathcal{S}$ of *admissible states* of RMS is defined to contain just those sets of components which are completely self-satisfying, specifically,

$$\mathcal{S} = \{S \subseteq \mathcal{D} \mid S \in \bigcap_{d \in S} \mathcal{I}(d)\}.$$

We define interpretations for nodes, SL-justifications, and CP-justifications as follows.

There are two sorts of nodes: *ordinary* nodes, and *contradiction* nodes. The user of RMS stipulates the sort of each node explicitly by labelling contradiction nodes as such. Ordinary nodes have no special meaning to RMS, except that they are distinct from other nodes. Contradiction nodes, in contrast, do have special meaning: RMS avoids including them in states. We name these two sets of nodes $\mathcal{N}^{\top}$ and $\mathcal{N}^{\perp}$ respectively, so that $\mathcal{N} = \mathcal{N}^{\top} \cup \mathcal{N}^{\perp}$. If $n \in \mathcal{N}^{\top}$, then $\mathcal{I}(n) = \mathbf{P}\,\mathcal{D}$, the "trivial" specification on states which rules out no potential states. If $n \in \mathcal{N}^{\perp}$, then $\mathcal{I}(n) = \emptyset$, the "contradictory" specification which accepts no states.

If $e = A \,\|\, B \,\|\!\!- c$ is a SL-justification, then

$$\mathcal{I}(e) = \{S \subseteq \mathcal{D} \mid A \subseteq S \subseteq B^c \supset c \in S\}.$$

($B^c = \mathcal{D} - B$.) That is, e specifies that if S contains every element of A and contains no element of B, then S must also contain c in order to be admissible. In RMS parlance, we say that if every element of A is *in* and every element of B is *out*, then c must be *in* as well.

If $e = (A \,\|\, B \,\|\!\!- c) \,\|\!\!- d$ is a CP-justification, then intuitively, the CP-justification is satisfied if d is in S whenever c is in all admissible states as close as possible to S that contain all elements of A and none of B. We express this formally by defining

$$\mathcal{I}(e) = \{S \subseteq \mathcal{D} \mid \; [\forall S' \in \eta(S, A, B) \quad c \in S'] \supset d \in S\}$$

where

$$\eta(S, A, B) = \nu(S, \{S' \in \mathcal{S} \mid A \subseteq S' \subseteq B^c\}),$$

and

$$\nu(S, X) = \{S' \in X \mid \forall S'' \in X \quad [S \bigtriangleup S'' \subseteq S \bigtriangleup S'] \supset [S'' = S']\}.$$

For this definition, we measure closeness of a state to S in terms of the set of elements by which the two differ, namely the symmetric difference $S \bigtriangleup S' = (S - S') \cup (S' - S)$. Other notions of closeness may be used instead if desired, but we cannot pursue them here.

§**5.** Given a set of nodes and justifications $S \subseteq \mathcal{D}$, the action of RMS is to derive an admissible state E containing S by adding additional nodes to S. Since E is admissible, it satisfies all the specifications represented by the elements of S. However, we do not accept every admissible superset of S as a reasonable solution to the requirements posed by S, since some of these supersets may introduce nodes and justifications completely unrelated to those mentioned in S. (Analogously, in logic the set of theorems of a set of axioms is the deductive closure of the axioms. The theorems form a deductively closed set, but there may be larger deductively closed sets containing the axioms whose extra elements have nothing to do with the axioms.) To avoid unwarranted additions to the initial set S, we define the *admissible extensions* of S, written $AExts(S)$, to be the admissible sets $E \supseteq S$ finitely grounded with respect to S, where E is finitely grounded with respect to S if and only if for every $e \in E$ there is a finite sequence $\langle g_0, \ldots, g_n \rangle$ of elements of E such that $e = g_n$ and for each $i \leq n$, either

(1) $g_i \in S$, or
(2) there is some $j < i$ such that
  (a) $g_j = A \parallel B \Vdash g_i$,
  (b) for each $a \in A$, $a = g_k$ for some $k < j$, and
  (c) $b \notin E$ for each $b \in B$, or
(3) there is some $j < i$ such that
  (a) $g_j = (A \parallel B \Vdash c) \Vdash g_i$ and
  (b) $c \in E'$ for each $E' \in \eta(E, A, B)$.

In other words, E is finitely grounded with respect to S if every element of E has a non-circular argument from S in terms of *valid* justifications in E. In general a set $S \subseteq \mathcal{D}$ may have any number (0 or more) of admissible extensions. Note that while the antecedents of SL-justifications must occur in such arguments, valid CP-justifications are simply looked on as "oracles" about other admissible states. RMS actually employs an approximation scheme instead of oracles, but it is too complicated to present here.

§6. In [DOYLE 1982] we develop at length the theory of this formulation of nodes and SL-justifications, leaving aside CP-justifications. That treatment is too long to reproduce here, but we can sketch a few of the principal results. First, one can stratify by construction every admissible extension into a series of levels $\Lambda_i(S, E)$ corresponding to the the lengths of the shortest arguments for elements of E from S, with $\Lambda_0(S, E) = S$ and $\Lambda_\omega(S, E) = \bigcup_{i=0}^{\infty} \Lambda_i(S, E)$. The first main result turns the same construction around to show that any set constructed in this way must be an admissible extension. This fact is important in proof of correctness of RMS algorithms.

(6.1) THEOREM (FIXED POINT). *$E \in AExts(S)$ iff $E = \Lambda_\omega(S, E)$.*

A corollary of this is that admissible extensions are set-inclusion minimal.

(6.2) COROLLARY (MINIMALITY). *If $E, E' \in AExts(S)$ and $E \subseteq E'$, then $E = E'$.*

A related result is that distinct admissible extensions must share some SL-justification that supports conclusions in one extension but not in the other, and so represent incompatible interpretations of the common justifications.

(6.3) THEOREM (STRONG VALIDITY-OPTIMALITY). *If $E, E' \in AExts(S)$ and $E \neq E'$, then there is some $e \in E \cap E'$ such that $e = A \parallel B \Vdash c$ and either $A \subseteq E \subseteq B^c$ but not $A \subseteq E' \subseteq B^c$, or $A \subseteq E' \subseteq B^c$ but not $A \subseteq E \subseteq B^c$.*

In the longer treatment we also show that $E \in AExts(S)$ can be checked in time $O(|\mathcal{D}|^3)$. I do not know if one can construct admissible extensions in polynomial time. Admissibility of states can also be checked in polynomial time. I do not yet know how bad the complexities become when CP-justifications are considered.

§7. RMS not only constructs admissible extensions of sets of nodes and justifications, it also updates the state whenever reasons are added to or deleted from a "kernel" set. After RMS leaves the database in state S, the external program using RMS computes a new kernel set $\partial(S)$ of justifications from S.

We write $\Delta(S)$ to mean the set of *admissible transitions from* (or successors of) S, the intent being that RMS revise the state to be an admissible extension of the new kernel. If we allow any revision, then $\Delta(S) = AExts(\partial(S))$, so the complexity questions mentioned earlier about constructing admissible extensions have considerable practical importance. A more interesting possibility is to require Δ to be *conservative*, that is, to allow only transitions to those states in $AExts(\partial(S))$ which are as close as possible to S. Here we again draw on the symmetric difference comparison of states introduced earlier, and define the conservative transition table Δ^{ν} by $\Delta^{\nu}(S) = \nu(S, AExts(\partial(S)))$. It would be very valuable to have conservative versions of RMS, since then successor states would look as much like their predecessors as possible. Unfortunately, the efficient mechanizability of conservative transitions is an open question. Because of this, RMS was implemented to "approximate" conservative revisions in the sense of probabilistic algorithms. These "approximations" are always admissible extensions, but sometimes may fail to be minimal transitions due to the limited information used in the "local" choices made by the revision algorithm.

§8. If $\Delta(S) = \emptyset$ because the external program includes in $\partial(S)$ a contradiction node or a justification to support a contradiction node, then RMS must perform "backtracking" to find a new state not containing any contradiction node. (One limitation of RMS corrected in some of its relatives is abnormal failure if $\Delta(S) = \emptyset$ without S containing any contradictions.) RMS backtracks by adding some new justifications A to the kernel $\partial(S)$ in hopes that $\partial(S) \cup A$ will have admissible extensions. RMS chooses new justifications to add so as to find a previously bypassed admissible transition as close as possible to the current state. Specifically, if $\langle S_i \rangle_{i=0}^{n}$ is the sequence of previous states, we can treat the intended behavior of RMS as that of retreating to some state in $\nu(S, \bigcup_{i=0}^{n}(\Delta(S_i) - \{S_{i+1}\}))$. However, I cannot exactly characterize the "nearness" relation actually realized by RMS as ν, because RMS only uses a heuristic choice based on the structure of the arguments which support contradiction nodes. Can RMS be improved to employ the same conservation principles in both backtracking and ordinary state transitions? It may be that MCALLESTER'S and GOODWIN'S improvements to RMS do so, but I have not yet been able to perform the necessary analysis. See [DOYLE 1982] for a discussion of a variety of possible backtracking schemes from this point of view.

§9. The preceding presents an exact specification for many aspects of RMS, and specifications of ideal behavior for other aspects, such as interpretation of CP-justifications, where RMS employs half-measures. Unfortunately, unanswered questions of computational feasibility and RMS's informal historical development prevent the actual program from living up to the full set of ideal specifications. With these exact specifications, can we now do better? And if the particular characterizations of conservative transitions and CP-justification satisfiability are provably intractable (or more likely, provably NP-hard), are there efficiently computable relations that approximate these well in some (perhaps probabilistic) sense?

REFERENCES

Charniak, E., Riesbeck, C. K., and McDermott, D. V., 1980. *Artificial Intelligence Programming*, Hillsdale, New Jersey: Lawrence Erlbaum Associates.

de Kleer, J., and Doyle, J., 1982. Dependencies and assumptions, *Handbook of Artificial Intelligence* (A. Barr and E. A. Feigenbaum, eds.) Los Altos: William Kaufmann, 72-76.

Doyle, J., 1979. A truth maintenance system, *Artificial Intelligence* **12**, 231-272.

Doyle, J., 1982. Some theories of reasoned assumptions: an essay in rational psychology, Pittsburgh: Department of Computer Science, Carnegie-Mellon University.

Gärdenfors, P., 1980. Epistemic importance and minimal changes of belief, Lund: Department of Philosophy, Lund University.

Goodwin, J. W., 1982. An improved algorithm for non-monotonic dependency-net update, Linköping: Software Systems Research Center, Linköping University.

London, P. E., 1978. Dependency networks as a representation for modelling in general problem solvers, College Park: Department of Computer Science, University of Maryland, TR-698.

Martins, J. P., 1983. Reasoning in multiple belief spaces, Ph.D. thesis, Buffalo: Department of Computer Science, State University of New York.

McAllester, D. A., 1980. An outlook on truth maintenance, Cambridge: Artificial Intelligence Laboratory, Massachusetts Institute of Technology, AI Memo 551.

McDermott, D., 1982. Contexts and data-dependencies: a synthesis, New Haven: Department of Computer Science, Yale University.

McDermott, D., and Doyle, J., 1980. Non-monotonic logic—I, *Artificial Intelligence* **13**, 41-72.

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

Steele, G. L. Jr, 1980. The definition and implementation of a computer programming language based on constraints, Cambridge: Artificial Intelligence Laboratory, Massachusetts Institute of Technology, TR-595.

Thompson, A., 1979. Network truth maintenance for deduction and modelling, *Proc. Fifth International Joint Conference on Artificial Intelligence*, 877-879.