appears in $J$, or $C$ must also appear in $P$. The Nixon-diamond example can be represented in default logic with one fact and two default rules:

$$
\text{Republican(Nixon)} \land \text{Quaker(Nixon)}.
$$

$$
\text{Republican}(x) : \neg \text{Pacifist}(x) \lor \neg \text{Pacifist}(x).
$$

$$
\text{Quaker}(x) : \text{Pacifist}(x) \land \text{Pacifist}(x).
$$

To interpret what the default rules mean, we define the notion of an extension of a default theory to be a maximal set of consequences of the theory. That is, an extension $S$ consists of the original known facts and a set of conclusions from the default rules, such that no additional conclusions can be drawn from $S$ and the justifications of every default conclusion in $S$ are consistent with $S$. As in the case of the preferred models in circumscription, we have two possible extensions for the Nixon diamond: one wherein he is a pacifist and one wherein he is not. Prioritized schemes exist in which some default rules can be given precedence over others, allowing some ambiguities to be resolved.

Since 1980, when nonmonotonic logics were first proposed, a great deal of progress has been made in understanding their mathematical properties. There are still unresolved questions, however. For example, if “Cars have four wheels” is false, what does it mean to have it in one’s knowledge base? What is a good set of default rules to have? If we cannot decide, for each rule separately, whether it belongs in our knowledge base, then we have a serious problem of nonmodularity. Finally, how can beliefs that have default status be used to make decisions? This is probably the hardest issue for default reasoning. Decisions often involve tradeoffs, and one therefore needs to compare the strengths of belief in the outcomes of different actions, and the costs of making a wrong decision. In cases where the same kinds of decisions are being made repeatedly, it is possible to interpret default rules as “threshold probability” statements. For example, the default rule “My brakes are always OK” really means “The probability that my brakes are OK, given no other information, is sufficiently high that the optimal decision is for me to drive without checking them.” When the decision context changes—for example, when one is driving a heavily laden truck down a steep mountain road—the default rule suddenly becomes inappropriate, even though there is no new evidence of faulty brakes. These considerations have led some researchers to consider how to embed default reasoning within probability theory or utility theory.

### 12.6.2 Truth maintenance systems

We have seen that many of the inferences drawn by a knowledge representation system will have only default status, rather than being absolutely certain. Inevitably, some of these inferred facts will turn out to be wrong and will have to be retracted in the face of new information. This process is called belief revision. Suppose that a knowledge base $KB$ contains a sentence $P$—perhaps a default conclusion recorded by a forward-chaining algorithm, or perhaps just an incorrect assertion—and we want to execute TELL($KB, \neg P$). To avoid creating a contradiction, we must first execute RETRACT($KB, P$). This sounds easy enough.

---

Footnotes: Belief revision is often contrasted with belief update, which occurs when a knowledge base is revised to reflect a change in the world rather than new information about a fixed world. Belief update combines belief revision with reasoning about time and change; it is also related to the process of filtering described in Chapter 15.
Problems arise, however, if any additional sentences were inferred from $P$ and asserted in the KB. For example, the implication $P = Q$ might have been used to add $Q$. The obvious "solution"—retracting all sentences inferred from $P$—fails because such sentences may have other justifications besides $P$. For example, if $R$ and $R = Q$ are also in the KB, then $Q$ does not have to be removed after all. Truth maintenance systems, or TMSs, are designed to handle exactly these kinds of complications.

One simple approach to truth maintenance is to keep track of the order in which sentences are told to the knowledge base by numbering them from $P_1$ to $P_n$. When the call $\text{RETRACT}(KB, P)$ is made, the system reverts to the state just before $P$ was added, thereby removing both $P$, and any inferences that were derived from $P$. The sentences $P_{i+1}$ through $P_r$ can then be added again. This is simple, and it guarantees that the knowledge base will be consistent, but retracting $P$ requires retracting and reasserting $n-i$ sentences as well as undoing and redoing all the inferences drawn from those sentences. For systems to which many facts are being added—such as large commercial databases—this is impractical.

A more efficient approach is the justification-based truth maintenance system, or JTMS. In a JTMS, each sentence in the knowledge base is annotated with a justification consisting of the set of sentences from which it was inferred. For example, if the knowledge base already contains $P$ and $Q$, then $\text{TELL}(P)$ will cause $Q$ to be added with the justification $\{P, Q\}$. In general, a sentence can have any number of justifications. Justifications make retraction efficient. Given the call $\text{RETRACT}(P)$, the JTMS will delete exactly those sentences for which $P$ is a member of every justification. So, if a sentence $Q$ had the single justification $\{P, P \vee Q\}$, it would be removed; if it had the additional justification $\{R, P \vee R = Q\}$, it would still be removed; but if it also had the justification $\{R, P \vee R = Q\}$, then it would be spared. In this way, the time required for retraction of $P$ depends only on the number of sentences derived from $P$ rather than on the number of other sentences added since $P$ entered the knowledge base.

The JTMS assumes that sentences that are considered once will probably be considered again, so rather than deleting a sentence from the knowledge base entirely when it loses all justifications, we merely mark the sentence as being out of the knowledge base. If a subsequent assertion restores one of the justifications, then we mark the sentence as being back in. In this way, the JTMS retains all the inference chains that it uses and need not rederive sentences when a justification becomes valid again.

In addition to handling the retraction of incorrect information, TMSs can be used to speed up the analysis of multiple hypothetical situations. Suppose, for example, that the Romanian Olympic Committee is choosing sites for the swimming, athletics, and equestrian events at the 2048 Games to be held in Romania. For example, let the first hypothesis be $\text{Site(Swimming, Pitesti)}$, $\text{Site(Athletics, Bucharest)}$, and $\text{Site(Equestrian, Arad)}$. A great deal of reasoning must then be done to work out the logistical consequences and hence the desirability of this selection. If we want to consider $\text{Site(Athletics, Sibiu)}$ instead, the TMS avoids the need to start again from scratch. Instead, we simply retract $\text{Site(Athletics, Bucharest)}$ and assert $\text{Site(Athletics, Sibiu)}$ and the TMS takes care of the necessary revisions. Inference chains generated from the choice of Bucharest can be reused with Sibiu, provided that the conclusions are the same.