# Incompleteness in Knowledge Bases

Hector J. Levesque

Dept. of Computer Science
University of Toronto
Toronto, Ontario

The first topic for discussion at the High Level Abstraction Workshop was

What should be modelled?

that is, what aspects of the world (slice of reality / enterprise) must be dealt with in a high level conceptual model. I would like to address this question from the point of view of incomplete knowledge bases. Although my remarks pertain mainly to AI applications, I suspect that any application using an information system of some sort will eventually have to face similar issues.

The knowledge bases required in many applications can be characterized by a lack of complete information about the world of interest. For example, a knowledge base used by a problem solver attempting to discover the identity of some individual might contain facts such as

(1) Mary's husband is a 29 year old doctor.
(2) Joe is under 29 years old.
(3) Either John or Jim is a teacher.
(4) Some student lives in Montreal.

Note that these facts do not say which doctor Mary is married to, what Joe's age actually is, which of John or Jim is the teacher or what student lives in Montreal. A similar kind of knowledge base can arise in medical applications when attempting to diagnose a disease from a set of vague symptoms, or in scene analysis where the goal is to recognize an object from a few visual clues. The key property of a knowledge base in these and other applications is that it may not have all the pertinent information about some relevant aspect of the world in question. More precisely, given some language L used to talk about the world, a knowledge base is incomplete if there is a relevant sentence of L whose truth cannot be determined solely on the basis of what is represented in the knowledge base.



To say that a knowledge base is incomplete is quite different from the somewhat trivial observation that there are aspects of the world that are irrelevant to the intended application and, consequently, not dealt with in the knowledge base. In the above examples, it is not that Joe's age, John's occupation and Mary's husband are irrelevant and therefore not known. They are relevant and something is indeed known about them. The knowledge base is incomplete precisely because these aspects of the world are relevant but not yet completely specified.

The reason incomplete knowledge bases are so important is that, in many applications, the knowledge base undergoes a continual evolution. At each stage, information can be acquired by direct observation (as in some recognition applications) or by the interaction with some outside agent (man or machine). This information is potentially very vague or indefinite in nature. More importantly, the user of the knowledge base cannot simply wait for it to stabilize in some final and complete form since this may never happen. So there has to be a systematic acknowledgement of the fact that any knowledge base may be only a partial model of the world it is intended to represent. In what follows, I will examine what steps should be taken to adequately cope with incomplete knowledge bases.

A fundamental requirement for dealing with incomplete knowledge bases is that the representation language used to construct or update a knowledge base must be prepared to accept very vague additions to what is already known. For example, it is one thing to be able to tell a knowledge base that

(5) George and Mary are teachers.

but quite a different matter to handle an addition like (3) above. Most representation languages in AI (and database management, for that matter) have no trouble interpreting (5) as something like

insert George into the class TEACHER
insert Mary into the class TEACHER

but this does not work for (3). Similarly, an addition of

(6) Bill lives in Montreal.

might be interpreted as

insert Bill into the class MONTREAL_CITIZEN

or

update Bill.home_city = Montreal

or

insert <Bill,Montreal> into the relation LIVES_IN_CITY

This does not help in the interpretation of (4), however, unless one is prepared to do something like

insert <Student,Montreal> into the relation ONE_OF_THE_CLASS_LIVES_IN_CITY

and have similar relations for every potential vague addition to the knowledge base.

Problems like this will arise with any representation language that changes a knowledge base in terms of insert/delete/update operations on classes and relations. These languages require that all additions to the knowledge base express relationships among certain objects known to the knowledge base (or create such objects). On the other hand, additions like (1), (2), (3) and (4) do not express such a relationship even though they name certain objects. Of course, this problem does not apply to representation languages based on logic since an explicit disjunction and existential quantification operator is provided.

The requirements on a representation language for incomplete knowledge bases do not stop here, however. Since a knowledge base need not know of all the relevant individuals in the world being modelled, it should be capable of being told whether or not there are individuals of a certain type that it does not yet know about.

Consider, for example, a knowledge base that has been told only that (5) is true. If the knowledge base is complete, then George and Mary are the only relevant teachers. If not, then there may or may not be other relevant teachers not yet known. An incomplete knowledge base can be told which is the case by the addition of something like

(7) There is a teacher other than George and Mary.

or

(8) George and Mary are the only teachers.

But suppose the knowledge base had been told about a million teachers. Now to be able to tell it whether or not it has complete information regarding this class would require replacing the phrase

George and Mary

in (7) and (8) by a list with a million elements! This is clearly unsatisfactory as a method of informing the knowledge base of the limits of what it knows. Moreover, it simply does not work for an infinite class. What is required instead of the impossibly long versions of (7) and (8) is something like

(7') There is a teacher other than the known teachers.

or

(8') The only teachers are the known teachers.

where the "known teachers" are to be understood as the ones that the knowledge base has been told about (explicitly or implicitly).

To be able to handle additions like (7') and (8'), a representation language must make the distinction between what is known about the world and what is true in this world. For example, in a knowledge base where both (3) and (5) are known, the distinction is between

the teachers: George, Mary, one of John or Jim and maybe others

and

the known teachers: George and Mary

In other words, the knowledge base must realize that there are at least three teachers but that only two of them are as yet known to be teachers. If the language does not differentiate between the two cases, then (7') simply says that some teacher is not a teacher (which is never true), while (8') says that every teacher is a teacher (which is always true).

The distinction between what is true in a world being modelled and what is known in a knowledge base is a property of the incompleteness of the knowledge base. This distinction should, therefore, be reflected in the language used to query the knowledge base as well as the language used to change the knowledge base. In fact, the "known teachers", in the above example, can be thought of as the individuals for which the knowledge base answers "yes" when asked if the individual is a teacher. In this sense, neither John nor Jim are known teachers. So the answer to the question

(9) Are there any teachers other than the known teachers?

should be "yes" because of the situation with John and Jim. Any other reply (including "unknown") would be misleading.

Even if a representation language does not allow additions like (7') or (8'), it should still allow questions like (9) so that a user can find out the limits of what is known. The only way questions like (9) can be ignored is when the knowledge base is complete, in which case, the answer would always be "no". Otherwise, there will be a difference between what is known and what is true and the query language must allow questions that are able to capture the distinction. For example, if the answer to the question

(10) Does any student live in Montreal?

is "yes", this might be because the knowledge base has been told (4), that Montreal has a non-zero student population. To find out if anyone has been represented as both a student and a citizen of Montreal, the stronger question

(11) Is anyone known to be a student living in Montreal?

must be used. The answer here will be "yes" only when the knowledge base has a specific individual in mind. In fact, there is a third question that could be considered which is

(12) Is there a student among the known citizens of Montreal?

In this case, the answer is "yes" only when there are people known to live in Montreal and it is known that one of them is a student, though it need not be known which. As far as I know, no existing query language is able to capture the differences in meaning between these three questions even though it is quite easy to imagine knowledge bases where the questions should be answered differently. To the extent that a query language should provide a complete and accurate picture of an incomplete knowledge base, all existing languages are inadequate in this regard.

To be able to interpret additions and questions that use "know" in this sense, a knowledge base will have to know not only about the world but about itself as well. Specifically, it will have to know what it does and does not know. Such a knowledge base is then a model of both a world and the knowledge base. So, to return to the first question of the workshop, I contend that a high level conceptual model must include not only aspects of the world, but also aspects of the evolving models of the world and how these two are related. This can only be avoided by assuming that what is in a model is precisely what is relevant in the world being modelled and, therefore, ignoring completely the possibility of incompleteness.

In my doctoral thesis (in preparation), I examine the semantics of a formal language that talks both about a world and an incomplete model of a world simultaneously. This language will provide a formal framework for both a knowledge base definition language and query language. The end result of this work should be a better understanding of the requirements for an adequate treatment of incomplete knowledge bases.

Acknowledgements I am indebted to Alex Borgida for having helped clarify some of the ideas in this document. This research was sponsored in part by the National Science and Engineering Council of Canada and by the Dept. of Computer Science, University of Toronto.