Using PROMPT Ontology-Comparison Tools in the EON Ontology Alignment Contest

Natalya F. Noy, Mark A. Musen

Stanford Medical Informatics, Stanford University,
251 Campus Drive, x-215, Stanford, CA 94305, USA
{noy, musen}@smi.stanford.edu

Abstract. Objective evaluation and comparison of knowledge-based tools has so far been mostly an elusive goal for researchers and developers. Objective experiments are difficult to perform and require substantial resources. The EON Ontology Alignment Contest attempts to overcome these problems in inviting tool developers to perform a series of experiments in ontology alignment and compare their results to the reference alignments produced by experiment authors. We used our PROMPT suite of tools in the experiment. We briefly describe PROMPT in the paper and present our results. Based on this experience, we share our thoughts on the experiment design, its positive and negative aspects, and talk about lessons learned and ideas for future such experiments and contests.

1 Introduction

Objective evaluation and comparison of knowledge-based tools has so far been mostly an elusive goal for researchers and developers. First, such evaluations require resources, both financial and human. Second, for an evaluation to be objective, it must be designed and run by someone other than tool developers themselves. Otherwise, the evaluation setup and comparison parameters are inevitably skewed (often subconsciously) to benefit the designers' own tools. Third, it is often hard to come up with realistic tasks and gold standards because all tools are designed for somewhat different purposes and trying to pigeonhole the tools into a single set of tasks for an evaluation often puts the tool designers in the untenable position of comparing their tool in circumstances for which it was not designed. Fourth, many of the criteria are, by their very nature subjective. For example, when evaluating the quality of an ontology, we often don't have a single correct answer for how certain concepts should be represented. When evaluating the quality of ontology alignment, we often cannot agree on the precise relationships between concepts in source ontologies.

Therefore, any experiment or contest to compare ontology-based tools will almost inevitably draw criticism. Nonetheless, given the extreme dearth of any comparative evaluation of ontology-based tools, any good evaluation is a significant step forward. Hence, we believe that the community will learn many lessons from the EON Ontology Alignment Contest[1] that is run as part of the 3d Workshop on the Evaluation of

[1] http://co4.inrialpes.fr/align/Contest/

Ontology-based Tools at the International Semantic Web Conference.[2] In the experiment, developers of ontology-alignment tools used their tools to compare concepts in a set of ontologies. The resulting alignment was then compared to the reference alignment provided by the experiment authors. Hopefully, the lessons from this experiment will enable the community to produce new experiments that do not suffer from some of the problems the EON experiment has. Indeed, in some of the materials it is called a "contest" and in others, an "experiment" that will help us to understand how to run such evaluations. We subscribe to the latter goal since the experiment design, being the first one of its kind, has considerable deficiencies that make a contest premature.

In the rest of this paper, we share our thoughts on the experiment design, its positive and negative aspects, describe the set of PROMPT tools that we used in the experiment and our results. We then talk about lessons learned and ideas for future such experiments and contests.

2 Ontology Comparison with PROMPT

PROMPT is a suite of tools for managing multiple ontologies. It is a plugin to the Protégé ontology-editing and knowledge-acquisition environment.[3] The open architecture of Protégé allows developers to extend it easily with plugins for specific tasks. We implemented PROMPT as a set of such plugins.

2.1 Components of the PROMPT Suite

The PROMPT suite includes tools for many of the tasks in multiple-ontology management: interactive ontology merging [1], graph-based mapping [3], creating views of an ontology [2], ontology versioning [4], and ontology-library maintenance. It is through development of these tools that we came to realize that many of these directions are indeed related and started integrating the approaches into a common framework. The tools in the PROMPT suite share user-interface components, internal data structures, some of the algorithms, logging facilities, and so on.

IPROMPT is an interactive ontology-merging tool. It leads users through the ontology-merging process, suggesting what should be merged, identifying inconsistencies and potential problems and suggesting strategies to resolve them. IPROMPT uses the structure of concepts in an ontology and relations among them as well as the information it gets from user's actions. For example, if IPROMPT's analysis identified that two classes from different ontologies may be similar and then the user merged some of their respective subclasses, IPROMPT will be even more certain that those classes are similar.

ANCHORPROMPT—another component in the suite—analyzes a graph representing ontologies on a larger scale, producing additional suggestions. It takes as input a set of pairs of matching terms in the source ontologies and produces new pairs of matching terms. ANCHORPROMPT's results could then be used in IPROMPT to present new suggestions to the user.

[2] http://km.aifb.uni-karlsruhe.de/ws/eon2004/
[3] http://protege.stanford.edu

Merging source ontologies to create a new one is not always what the user needs. If the user prefers to keep the source ontologies separately and to record only the alignment, iPROMPT saves the alignment as a side-effect of the merging process. The user can then discard the merged ontology and simply use the produced alignment.

PROMPTDIFF is a tool for comparing ontology versions. We observed that many of the heuristics we used in iPROMPT would be very useful in finding what has changed from one version of an ontology to another. These heuristics include analysis and comparison of concept names, properties that are attached to concepts, domains and ranges of properties and so on. At the same time, our level of confidence in the analysis could be much higher than in the case of ontology merging: If two concepts that came from versions of the same ontology look the same (e.g., have the same name and type), they probably *are* the same, whereas if two frames that came from independently developed ontologies look the same, they may or may not be the same. Consider a class `University` for example. In two different ontologies, this class may represent either a university campus, or a university as an organization, with its departments, faculty, and so on. If we encounter a class `University` in two versions of the same ontology, it is much more likely that it represents exactly the same concept.

The PROMPTDIFF algorithm for version comparison consists of two parts: (1) an extensible set of heuristic matchers and (2) a fixed-point algorithm to combine the results of the matchers to produce a structural diff between two versions. Each matcher employs a small number of *structural* and *lexical* properties of the ontologies to produce matches. The fixed-point step invokes the matchers repeatedly, feeding the results of one matcher into the others, until they produce no more changes in the diff.

One matcher, for example, looks for unmatched classes where all siblings of the class have been matched. If multiple siblings are unmatched, but their sets of properties differ, another matcher will pick up this case and try to match these classes to unmatched subclasses of the parent's image. Another matcher looks for unmatched properties of a class when all other properties of that class have been matched. There are matchers that look for lexical properties such as all unmatched siblings of a class acquiring the same suffix or prefix. The architecture is easily extensible to add new matchers. With the introduction of OWL, for example, we implemented additional matchers that compared anonymous classes.

2.2 Evaluating PROMPT in the EON Ontology-Alignment Contest

In the EON Ontology-Alignment Contest, the participants had to perform the following task: For each of the ontologies in the experiment, map its classes and properties to the classes and properties in a **reference ontology** and record the alignment. The alignment was then compared to the **reference alignment** provided by the authors of the experiment.

We originally planned to use iPROMPT and ANCHORPROMPT—our tools for ontology merging and alignment—in the experiment. However, most of the source ontologies in the experiment are not independently developed ontologies, but rather versions of the same ontology. Even in the three experiments where the reference ontology developed by the experiment authors was compared to ontologies developed elsewhere,

the comparison closely resembled comparison between ontology versions: All ontologies represented the structure of BibTex references and therefore were at the same time limited and driven by the BibTex data model and hence did not vary significantly in the concepts they represented. They varied only in coverage: some ontologies included for instance more detailed representation of people and projects but since these concepts were not part of the BibTex model, they were not mapped to concepts in the reference ontology anyway.

Given that the comparison was very close to version comparison, we decided to use only PROMPTDIFF, our version-comparison tool. PROMPTDIFF produces a mapping between concepts in one version and their corresponding concepts in another version. In addition, PROMPTDIFF produces a comparison between these corresponding concepts, presents the results to the user in an intuitive interface and enables users to accept and reject changes. For the experiment, we used only the first component of PROMPTDIFF, the one that produces correspondences between concepts (classes and properties) in two versions.

Adaptation of PROMPTDIFF **for the Experiment.** Figure 1 shows the process that we used to perform comparisons for the experiment. First, we ran PROMPTDIFF on the two ontologies to be mapped. PROMPTDIFF runs in a stand-alone completely automatic mode and produces a table of mappings between concepts (clases, properties, and instances) as one of its results. Second, we recorded the results in an RDF file following the format set in the experiment. The recording mechanism was our first modification of the tool for the experiment. Third, we used PROMPTDIFF *itself* again to compare the alignment to the benchmark alignment provided in the experiment and to compile the results. Because PROMPTDIFF compares not only classes and properties, but also individuals (by comparing their values for corresponding properties), we could easily perform this comparison of RDF data automatically. We present our comparison results in Section 3.

We have performed minimal adaptation of the tool itself. One of the key features of PROMPTDIFF is that it is an *extensible* collection of matchers. Hence, it is easy to add new matchers. In the process of the experiment, we found several interesting heuristics that we have added to the tool. These heuristics were conservative enough that they did not lower the resulting precision for all the tests, but they did help us increase the recall. For instance, we already had a heuristic that matched two properties P_1 and P_2, each of which had a single range, R_1 and R_2 respectively, if we already new that R_1 and R_2 matched. We did not have a corresponding heuristic for domains, however, and therefore added it. We found these heuristics to be generally useful for version comparison and plan to keep them in the tool. Overall, we added three new matchers to the 17 matchers that we already had.

Other Considerations. We would like to point out several other features of PROMPTDIFF that are relevant for the experiment.

As a plugin to Protégé which, through its extensible knowledge model and architecture, supports many different backends and ontology formats, PROMPTDIFF works with ontologies in OWL, RDFS, OKBC, and other languages. Therefore, we did not need to

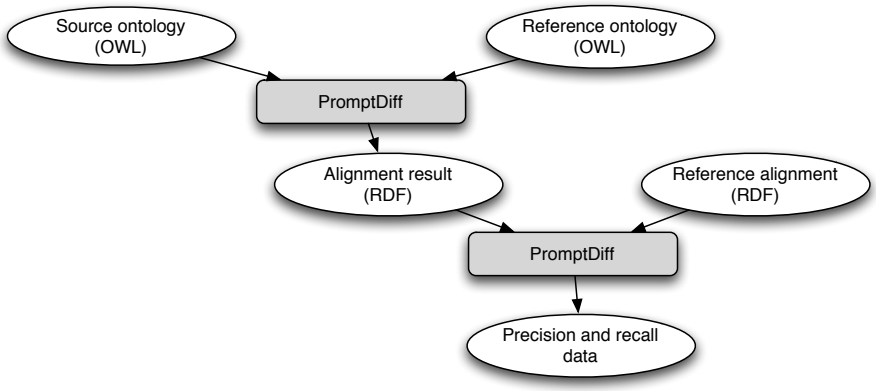

Fig. 1. Using PROMPTDIFF in the EON Ontology-alignment Contest. First, PROMPTDIFF automatically compares the source ontology to the reference ontology. Second, it produces the alignment results in the RDF Format specified in the contest rules. Third, it automatically compares the alignment result to the reference alignment to produce recall and precision data.

adapt it in any way either to compare the ontologies in the experiment (which were in OWL) or to compare the alignments that PROMPTDIFF produced to the reference alignment (which were in RDF).

PROMPTDIFF and other tools in the PROMPT suite produce only equivalence mappings. Some of the reference alignments in the experiment contained generalization and specialization mappings. In our results, we considered only the equivalence mappings from the reference alignment. In cases where PROMPTDIFF produced an equivalent mapping and the reference alignment contained only generalization or specialization, we considered it a positive result for PROMPTDIFF. In almost all of these cases, we believed that, from the common-sense point of view, the equivalence alignment was actually more correct (Section 3).

PROMPTDIFF does not use comments or instance information to align classes or properties (It uses instance information only to align between instances themselves). Therefore, tests that differed only in the presence of comments produced identical results.

3 Experiment Results

We performed 20 tests as part of the experiment, all the tests specified in the experiment. Table 1 contains the summary of the results. Note that in the discussion below we focus almost exclusively on recall. Because PROMPTDIFF uses a very conservative approach in creating matches, matches that it finds are almost always correct. Our previous experiments have put precision at 100%. in this experiment, it was above 99%.

3.1 Results of Specific Tests

In 10 of the 20 tests, the source ontology contained classes and properties with exactly the same names as those in the reference ontology. While in a general case of ontology alignment, when two classes from different ontologies have the same name, they may not necessarily match, in version alignment, they almost always do. In fact, in the experiment, they always matched. Furthermore, in these 10 cases, there were no other matches. Thus, any tool, which, just as PROMPTDIFF starts by comparing names of concepts (and, perhaps their types) would trivially produces the perfect alignment in these cases. These tests are: 101, 103, 104, 221, 222, 223, 224, 225, 228, 230. Test 102 had no matches, and PROMPTDIFF correctly found this result. We do not discuss these tests further in this section.

Tests 201 and 202: No names, no comments. In these tests, each class and property name was replaced by a random one. These tests were the most difficult of all tests because, except for the imported classes $foaf : Organisation$ and $foaf : Person$, there was no other name information for the tool to use. PROMPTDIFF was able to match 8 classes and 3 properties. Having the two imported classes was crucial to come up with any matches, since PROMPTDIFF could then use such clues as having single unmatched subclasses of a matched class; having a single property attached to matched classes; and so on.

We performed a variation of this experiment, where the class names all remained scrambled, but property names were all restored (tests 201a and 202a in Table 1). Having properties helped tremendously since now PROMPTDIFF was able to find correctly 90 of the 92 matches from the reference alignment. The only classes that it did not match were $MastersThesis$ and $PhDThesis$. In fact, after examining these classes in the ontology, we are convinced that these two classes are indistinguishable to any tool in the experiment: They are referenced in exactly identical ways in the reference ontology and having their names scrambled makes them indistinguishable.

Since PROMPTDIFF does not use comments in its analysis, results for the tests 201 and 202 (which differed only in the presence of comments) were identical.

Test 204: Naming conventions. Despite the use of different naming conventions in the source ontology, PROMPTDIFF found all matches.

Tests 205 and 206: Synonyms and Foreign Names. As the Table 1 shows, PROMPTDIFF matched approximately 50% of classes in each of these tests, and approximately 25% of properties. PROMPTDIFF does not have a specific matcher that uses a dictionary to look up word translations or a thesaurus to look up synonyms. Hence, it was left to using only structural clues to find the matches. Without the knowledge of synonyms, however, many of the classes and properties were hard or impossible to distinguish. Some of the classes that it did not match include $MastersThesis$ and $PhDThesis$ mentioned earlier, and $School$ and $Publisher$, which are structurally indistinguishable without instances (which PROMPTDIFF does not use to compare classes). Other classes could probably be distinguished structurally if we used less conservative heuristics, but this approach would have produced many false matches in other experiments. Clearly, tools

that use dictionaries and thesauri would perform better on these tests. PROMPTDIFF itself would also benefit from additional matchers that use these external sources.

Test 301, 302, 303: BibTex/MIT, BibTex/UMBC, BibTex/Karlsruhe. The comparisons in this series were the only ones involving ontologies developed at different institutions and, hence, were not true *version* comparisons. As we pointed out earlier, the ontologies were still sufficiently close and used a very similar terminology, and hence using PROMPTDIFF was still appropriate. However, this case was the first one where PROMPTDIFF produced some incorrect matches (PROMPTDIFF incorrectly matched the $Conference$ classes in the two ontologies, and two $address$ properties).

In the reference alignments for these tests, we believed that some of the equivalence matches were not correct. In test 301, the matches between properties $date$ and $hasYear$ and $book$ and $booktitle$ were not supported by the ontologies. There was a similar problem with the $book$–$booktitle$ match in the reference alignment for test 302. In tests 302 and 304, the reference alignment suggests that class $Chapter$ in the reference ontology matches class $InBook$ in the source ontology and the class $InBook$ in the reference ontology is actually a specialization of the class $InBook$ in the source ontology. We did not find any data in the ontologies to support this match and therefore considered the equivalence match between $Chapter$ and $InBook$ in the reference alignment to be incorrect and the equivalence match between the two classes $InBook$ to be correct. It was also unclear why the properties $proceedings$, $isPartOf$ and $booktitle$ match. We did not include these incorrect matches from the reference alignment in our results. We also did not consider matches to concepts in non-local namespaces and therefore discarded the match to the $dateTime$ XML Schema datatype from the reference alignment for test 302.

3.2 Overall Results

Table 1 presents a detailed look at the results of the experiments. We separate the results for classes and properties before finding the average value. We have included our modified tests 201a and 202a into consideration: In these tests, we kept the class names scrambled, but restored the property names. We show the average values for the original set of tests (the line marked "no mods"). The line marked "mods" shows the average values if we consider modified versions of tests 201 and 202 (with property names intact) instead of the original ones. Given that the recall number for tests 201 and 202 were clearly outliers (recall that because PROMPTDIFF does not look at comments, these two tests were identical), we also looked at the average values without these outliers (the last line in the table).

The average precision for both class and property matches was above 99%. The average recall is 62.4%. Note however, that his low recall figure is caused mainly by two tests, 201 and 202. If we replace these two tests with our modified tests, 201a and 202a (the "mods" line), the recall goes up to 94.1%. If we drop these two outlier tests altogether, the recall is 94.9%. In all cases, where there was any difference in recall between classes and properties, the recall for classes was higher than the recall for properties.

Test	Class matches in PROMPT	Property matches in PROMPT	Precision for class matches (%)	Precision for property matches (%)	Overall PROMPT precision (%)	Recall for class matches (%)	Recall for property matches (%)	Overall PROMPT recall (%)
101	33	59	100	100	100	100	100	100
102	0	0	100	100	100	100	100	100
103	33	59	100	100	100	100	100	100
104	33	59	100	100	100	100	100	100
201	8	3	100	100	100	24.2	5.1	14.7
201a	31	59	100	100	100	93.9	100.0	97.0
202	8	3	100	100	100	24.2	5.1	14.7
202a	31	59	100	100	100	93.9	100	97.0
204	3	59	100	100	100	100	100	100
205	20	21	100	100	100	60.6	35.6	48.1
206	23	21	100	100	100	69.7	35.6	52.6
221	33	59	100	100	100	100	100	100
222	33	59	100	100	100	100	100	100
223	33	59	100	100	100	100	100	100
224	33	59	100	100	100	100	100	100
225	33	59	100	100	100	100	100	100
228	33	59	100	100	100	100	100	100
230	33	59	100	100	100	100	100	100
301	14	15	93.3	100	96.7	93.3	41.7	67.5
302	12	19	100	90.5	95.2	100	86.4	93.2
303	16	28	100	93.3	96.7	88.9	100.0	94.4
304	30	46	100	100	100	100	100	100
no mods			99.8	99.6	**99.7**	67.6	57.3	**62.4**
mods			99.9	99.7	**99.8**	94.5	93.8	**94.1**
no outliers			99.8	99.4	**99.6**	96.9	92.8	**94.9**

Table 1. Experiment results for PROMPTDIFF. Test numbers correspond to the tests in the experiment description. Tests 201a and 202a are the same as tests 201 and 202 respectively, but with property names retained and only class names scrambled. The first line in the results ("no mods") represents the average result for all the original tests in the set; the line for "mods" result represent the average for all the tests when tests 201a and 202a (our own test modifications) are considered instead of 201 and 202; results for "no outliers" represent the average after dropping the two (identical from PROMPT's point of view) worst tests, 201 and 202.

3.3 Discussion of PROMPT Results

Generally, we were very satisfied with the performance results. The precision was almost perfect, averaging above 99.5%. In general, if there were some matches between class or property names, there was enough information in the structure of the ontology to find the rest of the matches.

PROMPT did not perform as well with finding all property matches. Indeed, after examining the ontologies, we found that many properties were hard to distinguish purely by the structure of the ontology. Many properties had the same domains and ranges, there were very few restrictions to differentiate them, and so on. We felt that any heuristics that would have allowed us to find these property matches, would have provided false positives in other tests, improving recall but lowering the precision.

In summary, PROMPTDIFF works very well, both in terms recall and precision, if at least some of the class and property names match. It is notable that it was able to find 25% of the class matches even when all property and class names were scrambled.

As we noted earlier, PROMPTDIFF provides only equivalence matches, and therefore if generalization or specialization matches are required, this tool will not be appropriate.

There is some information that PROMPTDIFF does not use but that could provide additional matches. We plan to consider using this information in the future, such as using dictionaries for comparing ontologies in different languages; using thesauri for synonym lookup; using instance information and property values to match classes and properties.

4 Discussion of the Experiment

Perhaps the most important result of the experiment are the insights that we gained in how these experiments can be conducted in the future. Many things about this experiment were very positive:

- It was a controlled experiment, with test cases and reference results published well in advance and in computable form.
- The ontologies were of reasonable size.
- Some of the alignments were to real ontologies produced by different groups independent from one another (although they all were modeled from the BibTex data model, which made this independence slightly less valuable than it could have been).

Many things can be improved and must be considered for future experiments, however.

Use ontologies developed independently. As we have noted earlier, this experiment was really about comparing ontology *versions* rather than about comparing independently developed ontologies. This experiment, while a perfect experiment for comparing ontology-versioning tools, does not provide any data on how well the same tools

will perform in the semantic-integration scenario when they must align truly independent ontologies. A more interesting and challenging experiment would be to use ontologies that are truly independent from one another—developed by different groups and in the domain that is less "pre-modeled" than BibTex references.

Provide consensus alignments. The more independent source ontologies are the more likely it is that alignments themselves would be a point of disagreement. Even in this experiment, we disagreed with some of the reference alignments provided by the experiment authors. Some of the alignments were completely unsubstantiated by any information in the ontologies or even by common sense. In the future, a panel of experts from different groups (ideally, the authors of the ontologies themselves) should produce a consensus alignment reference to minimize disagreement.

Include interactive tools. Many researchers working in the area of ontology alignment and mapping firmly believe that a fully automatic ontology alignment is not possible. Therefore, many tools include an interactive element in them. The current experiment assumed a fully automatic alignment. For more realistic ontologies and usage scenarios, we have to incorporate the interactive component as well. It is challenging to introduce such component in a controlled way. One possibility would be to specify the set of user-provided alignments as input. This approach would simulate tools that allow users to provide alignment information to the tool. It may be however that we can come up with a controlled test for interactive tools only after we perform a pilot experiment similar to the current one to understand better what types of inputs the tools require and at which stages.

Freeze systems before the test set is published. The results of the current experiment will inevitably be skewed because the tool developers had the opportunity to adjust their tools after the test data were published. This situation opens up a possibility that developers will produce versions of the tools optimized for this particular task and set of ontologies, but not for another set of ontologies and tasks. In our case, as noted earlier, we have added three new matchers to PROMPTDIFF after analyzing the ontologies. These matchers are not specific to the test, they are simply heuristics we have not thought of before and will remain part of the tool. However, to make the results completely fair, future experiments (and, even more so, "contests") should require that a version of the tool is frozen before the test data are published. We can follow the traditional TREC model where training data are published first, tools fine-tuned and instrumented to collect all the statistics, and then frozen before the test data are made available. Then the period to perform the tests can be very short (2 weeks or so) since no tool instrumentation will need to be done at this point.

Provide more automated tools to collect and compare results. One of the great positive points about the current experiment was the requirement to use the standard format to record the alignments. There is no reason (except time and effort, which should not be significant) not to go further and provide a suite of tools that will perform the analysis on the alignment files, producing the same statistics and in the same format for all the participants.

Require that tools be made available to the community. Ideally, the tools participating in the experiment should be available to the community under some open-source or free-use license. In this case, others will be able to reproduce the results and to use the tools on their own ontologies. If making tools available for free is not feasible, they must at least be available to the experiment organizers so that they can verify the results. We believe that this availability to the organizers should be a hard requirement for anyone participating in the experiment.

Finally, we believe that such experiments are extremely important in not only comparing different semantic-integration tools, but also in improving these tools and their performance.

Acknowledgments

This work was conducted using the Protégé resource, which is supported by grant LM007885 from the United States National Library of Medicine. This work was supported in part by a contract from the US National Cancer Institute. The Prompt suite is available as a plugin to the Protégé ontology-development environment at `http://protege.stanford.edu/plugins/prompt/prompt.html` The results of the experiment are available at `http://protege.stanford.edu/plugins/prompt/oacontest`.

References

1. N. Noy and M. Musen. PROMPT: Algorithm and tool for automated ontology merging and alignment. In *Seventeenth National Conference on Artificial Intelligence (AAAI-2000)*, Austin, TX, 2000.
2. N. Noy and M. A. Musen. Specifying ontology views by traversal. In *3rd International Semantic Web Conference (ISWC2004)*, Hiroshima:Japan, 2004.
3. N. F. Noy and M. A. Musen. Anchor-PROMPT: Using non-local context for semantic matching. In *Workshop on Ontologies and Information Sharing at the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-2001)*, Seattle, WA, 2001.
4. N. F. Noy and M. A. Musen. PromptDiff: A fixed-point algorithm for comparing ontology versions. In *Eighteenth National Conference on Artificial Intelligence (AAAI-2002)*, Edmonton, Alberta, 2002.