Danmarks
Tekniske
Universitet

DTU

02180
Introduction to Artificial Intelligence

Report for Assignment 2

Belief Revision Assignment

Authors

Patrick Martins – s204748
Samy Haffoudhi – s222887
João Silva – s222961
Andrea Bosisio – s226668

May 1, 2023

Preface

This is the report for the Belief Revision Assignment of the course *02180 - Introduction to Artificial Intelligence* for the academic year 2022-23, Spring semester.

Contents

1 Introduction

In this report, we present the development of a Belief Revision Engine.

Belief revision is crucial in Artificial Intelligence (AI) systems since it enables agents to adapt their beliefs based on new information and changes in their environment. An AI agent's beliefs are represented as an abstract set of all propositional formulas believed by the agent. Belief revision aims to modify these beliefs in a consistent and systematic manner. Belief revision is essential for ensuring that AI agents make informed decisions as it helps them maintain an accurate and up-to-date representation of their knowledge, free of logical contradictions.

Our engine is designed to handle belief revision tasks by implementing methods for logical entailment checking, contraction, and expansion of the belief base. Through AGM postulates our engine ensures that the belief revision process is coherent and well-founded.

2 Methodology

Exploiting the Levi Identity (eq. 1), we split the task of performing *revision* of a set of formulas B with another formula φ (formally denoted with $B * \varphi$) into two sub-task: the *contraction* (denoted with $\div$) and *expansion* (denoted with $+$) operations.

$$B * \varphi = (B \div \neg\varphi) + \varphi \tag{1}$$

The contraction operation is needed to avoid inconsistency and it is done by removing $\neg\varphi$ from B, ensuring a safe addition of φ to B. The contraction operation consists in generating a new subset of B, B', that does not imply $\neg\varphi$ by also limiting the removals from B. Since there could be multiple subsets, this operation is the most laborious one because one has to choose some reasonable criteria to rank those subsets. This problem is discussed in detail in Section 2.3.

2.1 Belief Base Design and Implementation

The belief revision engine utilizes a class `Agent` to represent an AI agent with a Knowledge Base (KB). The knowledge base is implemented as a list of propositional formulas, which allows us to store those formulas along with their temporal order of addition. The `Agent` class provides methods for adding new beliefs through belief revision (`revision(phi)`) and an `ask(phi)` method for querying the belief base (i.e., verifying if a formula `phi` is entailed by the belief base).

2.2 Logical Entailment

To check for logical entailment, a resolution-based method is employed, that is showing that $KB \models \varphi$ by showing that $KB \wedge \neg\varphi$ is unsatisfiable. Indeed, the `resolution` method in the `Agent` class converts the knowledge base KB and the conjunction with the negation of the query φ (i.e., $KB \wedge \neg\varphi$) into Conjunctive Normal Form (CNF). It then iteratively resolves pairs of clauses from the CNF representation until either an empty set (a contradiction) is generated or no new resolvents can be derived. If a contradiction is found it means that $KB \wedge \neg\varphi$ is unsatisfiable and so the method returns `True`, indicating that the query is entailed by the knowledge base. If no new resolvents can be generated the method returns `False`, indicating that the query is not entailed by the knowledge base.

2.3 Contraction of Belief Base

The contraction of the belief base is implemented using the remainders set method. The idea is to find the minimal sets of formulas that when removed from the belief base make it consistent with the addition of the new belief ϕ. As already mentioned, since that could be multiple possible remainders, a criteria to choose one needs to be implemented.

2.3.1 Priority Order of Formulas

Contraction makes use of a priority order over formulas in the knowledge base for deciding which formulas should get removed from the set. We chose to prioritize the newest knowledge, meaning the most recently added formulas have the highest plausibility order. This order is maintained implicitly through the use of a list for representing the knowledge base. We are prioritizing the newest formulas by iteratively trying to remove first the oldest formulas and then the combinations of them (always respecting the order).

This choice was motivated to simulate an agent which could adapt to its changing environment by prioritizing new incoming knowledge over its older knowledge, which could be useful in a variety of scenarios such as a robot sensing its environment through sensors or a new move made by an opponent in a game. In particular, this would also be the case for the Mastermind game.

Other choices could, for example, be the opposite, an agent prioritizing older knowledge, or priorities based on the formulas themselves rather than their insertion order.

2.4 Expansion of Belief Base

Expansion of the belief base is achieved using the `tell` method in the `Agent` class. This method simply adds a new propositional formula (belief) to the knowledge base. Since the knowledge base is implemented using a list, the formula is appended only if it is not already present in the list.

2.5 Revision of Belief Base

Accordingly to what is said above, to perform the revision of the belief base with the formula `query`, the `revision` method simply

1. calls the method `contraction` passing the negation of the `query` as argument, that is `Not(query)`;

2. calls the method `tell` with the `query` as argument.

3 Results

The implemented AGM postulates are a set of rationality constraints for belief revision. These postulates guide how an agent revises its beliefs when faced with new evidence. They ensure that the revision process is coherent and reasonable. The implemented AGM postulates are as follows:

1. **Success Postulate**: If an agent revises its knowledge base with a new belief, φ, then φ must be present in the revised knowledge base. This postulate ensures that the new information is accepted by the agent.

$$\varphi \in B * \varphi$$

2. **Inclusion Postulate**: The knowledge base revised with a new belief, phi, should be a subset of the knowledge base expanded with phi. This postulate guarantees that the revised knowledge base retains as much information as possible from the original knowledge base, given the new evidence.

$$B * \varphi \subseteq B + \varphi$$

   The success and inclusion postulates can easily be tested by revising a given knowledge base with any new belief φ and verifying that the revised knowledge base both contains φ and is a subset of the expansion of the original knowledge base with φ.

3. **Vacuity Postulate**: If the negation of φ is not in the knowledge base then the knowledge base revised with φ should be the same as the knowledge base expanded with φ. This postulate ensures that the agent does not discard information arbitrarily when revising its beliefs, only when there is a conflict with the new evidence.

$$\text{If } \neg\varphi \notin B, \text{ then } B * \varphi = B + \varphi$$

   The vacuity postulate was tested with the following simple knowledge base B: $B = Cn(\{p, q\})$

   By introducing a new belief r, and noticing that $\neg r \notin B$, we verified that the vacuity postulate holds: $B * r = B + r = Cn(\{p, q, r\})$

   This test could be extended to larger and more complex knowledge bases, as long as the condition $\neg\varphi \notin B$ is satisfied, and the results should be similar.

4. **Consistency Postulate**: If the new belief, φ, is consistent, the knowledge base revised with φ should also be consistent. This postulate guarantees that the agent maintains a consistent set of beliefs when revising its knowledge base, as long as the new evidence is consistent.

$$B * \varphi \text{ is consistent if } \varphi \text{ is consistent}$$

   The consistency postulate was tested on the knowledge base B, where $B = Cn(\{p, q\})$, when revising with a consistent belief and an inconsistent belief. When revising B with an inconsistent belief φ, such that $\varphi = p \wedge \neg p$, we get that the revised knowledge base is not consistent. On the other hand, if we revise B with a consistent belief φ, $\varphi = s \wedge \neg r$, the revised knowledge base is also consistent.

5. **Extensionality Postulate**: If two beliefs, φ and ψ, are equivalent, then the knowledge base revised with φ should be the same as the knowledge base revised with ψ. This postulate ensures that the revision process treats logically equivalent beliefs as identical, preserving the consistency and coherence of the agent's beliefs.

$$\text{If } (\varphi \leftrightarrow \psi) \in Cn(\emptyset), \text{ then } B * \varphi = B * \psi$$

   The extensionality postulate was tested for a knowledge base $B = Cn(\{p, q\})$, given two equivalent beliefs $\varphi = (p \wedge q) \implies r$ and $\psi = \neg r \implies (\neg p \vee \neg q)$. In the implementation, the two knowledge bases derived from the revision with the equivalent beliefs are converted to CNF and compared at the end.

These AGM postulates provide a foundation for the belief revision process, ensuring that the agent maintains a rational and coherent set of beliefs as it encounters new information, and this can be used to benchmark the implemented belief revision agent.

3.1 Examples of Revision of Knowledge Base

We tested our engine with all the examples provided in the lecture and we had coherent results for all of them. Moreover, while performing the revision, we are also asserting that all the AGM postulates implemented are fulfilled[1].

The following images illustrate the output from the console after revising beliefs on different knowledge bases.

Example 1 Revising $Cn(\{p, q\})$ with $\neg q$

```
Agent is revising [p, q] with ¬q
Agent 's new knowledge base after contraction: [p]
Agent 's updated knowledge base: [p, ¬q]
```

Example 2 Revising $Cn(\{p, q, r\})$ with $\neg(q \vee r)$

```
Agent is revising [p, q, r] with ¬(q | r)
Agent 's new knowledge base after contraction: [p]
Agent 's updated knowledge base: [p, ¬(q | r)]
```

Example 3 Revising $Cn(\{p, q, p \implies q\})$ with $\neg q$

```
Agent is revising [p, q, Implies (p, q)] with ¬q
Agent 's new knowledge base after contraction: [q | ¬p]
Agent 's updated knowledge base: [q | ¬p, ¬q]
```

[1]For the extensionality postulate we generate randomly either a formula ψ equal to φ or the negation of φ such that $\psi = \neg\varphi$.

Example 4 Revising $Cn(\{p, q, q \vee \neg p\})$ with $\neg q$ given different formula ordering

```
Agent1 is revising [p, q, q | ¬p] with ¬q
Agent1 's new knowledge base after contraction: [q | ¬p]
Agent1 's updated knowledge base: [q | ¬p, ¬q]

Agent2 is revising [q | ¬p, p, q] with ¬q
Agent2 's new knowledge base after contraction: [p]
Agent2 's updated knowledge base: [p, ¬q]
```

Note the difference in the order of formulas in the two initial knowledge bases: in the first case p is discarded because it's the oldest formula added, while in the second case, $p \implies q$ is discarded for the same reason.

4 Discussion

This project enabled all the group members of the project to better understand the concepts of contraction and revision of belief methods from a general knowledge base. We learned that the contraction of a belief is not as trivial as it might appear, since different sets of beliefs could be derived from the contraction of a belief. It became clear how an agent can methodically update its beliefs when presented with new information, while adhering to the AGM postulates to ensure rationality. Additionally, we became acquainted with the process by which a logical agent derives conclusions from a set of beliefs using the resolution method.

Initially, the group was in doubt on the application of plausibility order in the project, since it was thought that the general knowledge base would just have one knowledge base and revise knowledge accordingly. The greatest challenge was the implementation of the contraction algorithm, more specifically the method for collecting the remainders.

5 Conclusion

Belief revision plays a vital role in the development of intelligent agents enabling them to adapt to new information and maintain a consistent and up-to-date representation of their knowledge. This belief revision engine based on propositional logic and resolution provides a practical and efficient approach to address the challenges of belief revision in AI systems.

Future works on this project could be to optimize the resolution algorithm implementation, to perform the search of the remainders set by exploiting the search methods studied in the first part of the course, and to restructure the code in a more general way in order to handle multiple plausibility order criteria.

Another future work on this project could be to implement the game Mastermind. To this extent, a suitable representation for the game state would need to be developed that allowed for easy reasoning about the game state. The agent's reasoning process would leverage the AGM belief revision framework to revise its beliefs about the possible secret codes after each guess and feedback as well as strategies like depth-first-search, breadth-first-search and Monte Carlo Tree Search to explore the state space effectively. Finally, based on the performance evaluation, the agent could be enhanced by incorporating additional strategies, heuristics, or learning mechanisms to improve its code-breaking abilities.

References

[1] Stuart Russel and Peter Norvig. *Artificial Intelligence A Modern Approach*. Pearson Series, 4th edition, 2021.

[2] Edward N. Zalta, Uri Nodelman, Colin Allen, and R. Lanier Anderson. Logic of belief revision. *Winter 2017 Edition of the Stanford Encyclopedia of Philosophy*, 10 2017.