Modelling the Dynamic of Intentions using a Temporal Propositional Logic

Student: Maarten Burger 6584772
First Supervisor: Dragan Doder
Second Supervisor: Maria Francisca Pessanha

2 July 2021
Bachelor Thesis
Artificial Intelligence, Utrecht University
7.5 ECTS

Abstract

Belief-Desire-Intention agents are often based upon logical models and a main part of the field of BDI research is finding logical models. In this thesis I answered whether a logical model could be made and used to perform belief and (sophisticated) intention revision while staying close to propositional logic. I did this by building a logical model, proposing and formalizing a system of checking consistency, proposing and formalizing a (sophisticated) system of intention revision which connected intention to automatic planning and adopting a system of belief revision. Doing so, I showed the value staying close to propositional logic for a logical model as existing tools for propositional logic can be utilized.

1 Introduction

Belief-Desire-Intention (henceforth abbreviated as BDI) models can be used for developing and programming intelligent agents [14]. Their main use is for planning in a dynamic environment where the state of the world and information available to the agent can change. In order to adjust to change the agent needs to be able to revise its internal state, its beliefs and intentions. One of the main aspects of BDI (software) research is to find logical models in which it is possible to define and reason about BDI agents. This work will focus on building such a logical model and using it to define procedures which BDI agents can use to help them navigate dynamic environments.

BDI agents have long played an important role in the field of AI, as they have been used in the automation of smart calendars, air traffic management, e-health applications, customer service, household robots, and much more [15]. Recently, it has even been used in the field of security to automate a defense against Distributed Denial-of-Service (DDoS) attacks [13], with good results. The full extent of uses for BDI agents is currently unknown, so any contribution to finding more uses for BDI agents would further the field. In the case of this work, the contribution will be focused on building a logical model which can be used as a foundation for a BDI agent.

In this work I study how to model the dynamic of belief and intention within a logic that is close to propositional logic. My first step would be to propose a logic based on the previous logics proposed by the papers of M. van Zee et al. [20], A. Herzig et al. [10], Rao & Georgeff [14] and Y. Shoham [16], there will be more details on these papers in the Related Work section. This previous work has already dealt with the question of whether it is possible for a logic to model belief and intention revision in a dynamic environment while using the Database Perspective of Shoham. These logics are very complex, while this made these logics very expressive, it complicated the consistency check and revision. My goal is to propose a simplified version of these logics while keeping enough expressive power to properly formalize temporal behaviour of actions and their effects. But make it close to standard propositional logic so that the existing tools for checking consistency and revision for propositional logic might be utilized with very little modification.

In this work I will try to answer the question: ‘’Can a logical model for a BDI agent be made using the Database Perspective while staying close enough to propositional logic that existing tools can be applied?”. As a secondary goal I will also try to add a connection between intention and automated planning within my logical model, this connection is something A. Herzig et al. argued should be more looked into [9]. I will also add a novel way of revising intentions by weighting intentions using a happiness-value.

This is an interesting subject to research because while belief revision has received a lot of attention, intention revision has not received too much attention yet. Especially considering that building BDI agents from the Database Perspective has only really caught on after the paper of Shoham in 2009 [16], making this a relatively new part of the field.

2 Related Work

2.1 From Philosophy to AI

Before 1987 intention and belief (which would form the basis of BDI agents) were considered to be purely philosophical concepts. As such, most of the work relating to the concept of intention and concept of belief was done by philosophers. Among the philosophers it was generally accepted that these can help in understanding the relation one has to the world. On the one hand there was knowledge and on the other hand, desires [8]. Where knowledge, or belief, captured what one knows or thinks to know about the world and desire, or intention, captured what one wanted to be true. It became the dominant view that an intention can be considered a complex form of belief, an intention to perform an action might simply mean that you believe you will perform that action in the future [11]. However, in 1985, a very influential paper [1] by E. Alchourrón, Peter Gärdenfors, and David Makinson was written. This paper detailed the AGM postulates (which were named after the authors), this paper provided a framework for logic that can revise propositions and the postulates provided a set of conditions that any revision of such propositions should adhere to. At the time this was not thought up for or used for BDI agents, but it would become very influential in the field. Even now, in 2021, many logics for revising belief still treat the AGM postulates as must have conditions a revision procedure should have.

The view of belief and intention as a purely philosophical concept changed in 1987 when M. Bratman released his book ”Intentions, plans and practical reason” [5] which really laid the groundwork for what we now know as BDI agents. In this work, Bratman argued that intentions cannot be reduced further and are a basic ’mental state’ in of itself. This means that intentions have their own rules and norms they must adhere to. He asserts that commitment is the important distinguishing factor between intention and desire. Later, Bratman himself (along others) proposed an architecture for a rational agent in [4] with suggestions for implementations. This work invited other researchers to research this architecture in a more formal and systematic way. Bratman’s architecture would lay the groundwork for nearly every BDI agent for the next 20 years.

Many researchers accepted Bratman's invitation, and Bratman's BDI architecture would quickly be formalized by Cohen & Levesque [6] who made a formal logical model which was able to model mental states such as belief and intention (called "having a goal" in [6]). The main goal of Cohen & Levesque in [6] was to properly formalize intention in a useful manner such that it could be used for years to come. However, many found the approach of Cohen & Levesque unnecessarily complex. Later, in 1995, Rao & Georgeff united the theoretical and practical side of BDI agents [14], going from an ideal theoretical perspective to a practical perspective. In the following years many BDI logics were made and the field truly came alive as theory and practice started to communicate more.

Much later, in 2009, a very influential paper would be written by Y. Shoham [16] which considered a new architecture for BDI agents and a new way of looking at and implementing these agents in systems. This is what Shoham aptly named "The Database Perspective" and we will go in much further detail about this perspective in the next section. Since this is a relatively recent work, there is still much to discover about the models and the potential of this perspective. In a 2016 article [10], A. Herzig et al. suggested that all future BDI agents should use the database architecture suggested by Shoham [16] as a key component. Which is what we will also be doing in this work.

Even later, in 2016, Herzig et al. [9] wrote an article detailing the findings, limits and perspectives of BDI agents throughout the years, which goes further in depth into the history of BDI agents. In this paper A. Herzig et al. argued for more research in considering the connection between intention and automated planning, which is also what we will be looking at.

2.2 The Database Perspective

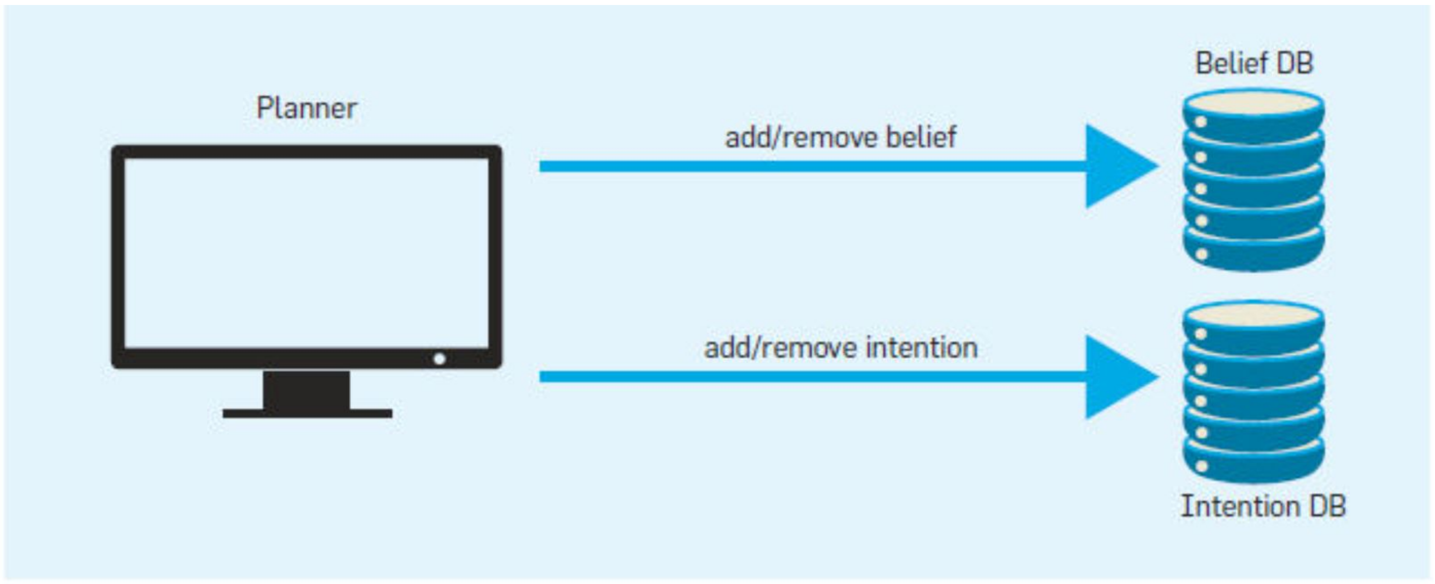

Figure 1: Schematic overview of the Database Perspective

In order to understand our approach to belief and intention it is important to understand the database perspective proposed by Y. Shoham [16]. This aptly named perspective is a way of looking at an agent as an Belief-Intention Database, which consists of a planner, an intention database and a belief database. This planner could be a multitude of functional planners that can perform some form of reasoning (as an example, this planner could be very similar to a STRIPS planner). This Belief-Intention Database can form an agent who should be able to keep track of the following:

- Beliefs about its current state.

- Beliefs about its future state.
- Beliefs about what actions are available in the current state.
- Beliefs about what actions will be available in future states.
- Beliefs about its plans in future states.

This Belief-Intention Database must also be consistent, which means that both the belief database and the intention database must be consistent internally and with each other. Shoham applies the following rules of consistency to this Belief-Intention Database:

- Beliefs must be internally consistent
- Intentions must be internally consistent. Considering we are using atomic actions we can say the following:
  - At most one action can be intended for any given moment
  - If two intended actions immediately follow one another, the earlier cannot have postconditions that are inconsistent with the preconditions of the latter.
- Intentions must be consistent with beliefs, which means:
  - If an agent intends to take an action the agent cannot believe that its preconditions do not hold.
  - if the agent intends to take an action, the agent believes that its postconditions hold.

This way of looking at belief and intention and the rules the database must adhere to form a strong basis to build upon. There have been many papers, such as [20], [10] and [9]. The last of those papers ([9]) even proposed to adopt Shoham's Database Perspective as an important key component for any BDI agents in the near future.

Shoham later used the ideas in his paper to build the Timeful application, a smart calendar [17]. Timeful has since been acquired by Google and its features have been (partially) integrated into Google Calendar. As David Kadavy (an associate of Shoham) put it: ”Timeful was intended to become an AI that tells you the best time to do the things that you intend to do”. The Timeful application shows that Shoham's Database Perspective has practical applications in AI.

2.3 This Work

In this work I seek to propose a logical model that can formally model a Belief-Intention Database such as the one described above and that can check consistency according to the consistency rules as Shoham described. I will base part of my logic upon the work of [20] where a logic is introduced to model such a Belief-Intention Database. This work will focus specifically on a new way of revising intentions, I will lightly touch upon belief revision since belief revision has already garnered a lot of attention and we will adopt an existing way of belief revision as proposed by [7] which satisfies the AGM postulates [1]. This new way of intention revision is interconnected with automated planning, which is something [9] suggested as a good connection to make (that is, the connection between intentions and automated planning).

The logical model I will propose in this work will be focused on atomic actions on a linear timeline. Herzig et al. [10] wrote about refining intentions on a linear timeline. This article only dealt with the refinement of intention for a linear timeline and not with the revision of these intentions. Recently, van Zee et al. [20] wrote a paper detailing a logical model and a system of belief and intention (including the revision of both) for a branching timeline using atomic actions, branching time in [20] complicates consistency. In this work I seek to make a new model combining the two, from Herzig et al. [10] I will adopt linear time and from van Zee et al. [20] I will adopt atomic actions. The result of this combination is a new logical model which would be my contribution. It is noteworthy that my model will not be more expressive than already existing models (in fact, the model in [20] is more expressive) but a part of the field of research in BDI agents, BDI-logics and BDI-software is discovering (or building, depending on whether you view logic as discovered or invented) logical models to model, define and reason about BDI systems.

Similarly to both [16] and [20], I would also like to separate strong and weak beliefs within the belief database. This separation is based upon the postconditions of intentions, where facts about the world are strong beliefs and the consequences of the actions an agent intends to do are weakly held beliefs. Both [16] and [20] treat postconditions as weak beliefs. This gives us the following property:
If an agent intends to take an action it must weakly believe that its postconditions will hold.

This distinction between strong and weak beliefs will be very useful when revising beliefs, since it gives us a way to cleanly revise without having leftover beliefs from another plan.

Similar to both [16] and [20] as well, I'd like to introduce the idea of treating preconditions as assumptions. This means that if an agent intends to take an action it will believe that its preconditions will hold somewhere in the future. This means that even if the preconditions for an action aren't true now, an agent can still intend to do them later if it believes the preconditions will be made true. This gives us the following property: *if an agent intends to take an action it cannot believe that its preconditions do not hold.*

I would now like to introduce our running example which we will come back to multiple times throughout this work:

Running Example. Imagine John, John loves swimming and he wants to go to the swimming pool. There are a few things John needs to take care of first though. John has grown quite a bit and does not fit into his swimsuit anymore, so he needs a new one before he can go swimming. John formulates a simple plan, first he goes to the store and then to the swimming pool. John believes that this plan will go swimmingly. What John does not know is that he might get a call from his grandma that she would really like him to visit for lunch or maybe someone else buys the last swimsuit in the store. If either of these events happen, John needs a way to properly reassess his options and decide what he would like to do next.

For the sake of argument let's say that the plan John made to go swimming does, in fact, go swimmingly. In this example John weakly believes that he will have a swimsuit (since having a swimsuit is a postcondition of buying one, but not a fact of the world). On the other hand John strongly believes that the weather is sunny, since that is a state of the world which does not depend on any action he intends to take. John also assumes that he will have a swimsuit by the time he goes swimming and he assumes that it will be sunny. Despite only one of these being dependent on his actions, he still assumes both.

3 Logical Model

In this section I will propose a logical model that can be used to properly represent an agent's beliefs and intentions and reason about them.

3.1 Syntax

I will define a language $\mathbb{L}$. This language should be able to express the beliefs of an agent.

Definition 1.

- Let the set $P = \{p, q, r...\}$ be a finite set of logical propositional letters as we know them from propositional logic.
- Let the set $P_l = \{\phi, \psi, \gamma...\}$ be any propositional formula that can be made from the propositional letters $p \in P$ and using the connectives $\wedge$ and $\neg$.
- Let the set $A = \{a, b, c...\}$ be a finite set of deterministic atomic actions. These actions have a *pre* and *post* which are an abbreviation for a formula $\phi \in P_l$. It is possible that the *pre* and *post* are empty, this action would then have no necessary preconditions and no postconditions (an action for doing nothing might have no requirements or consequences aside from time progressing).
- $do(a)$ is a function that will work as a transitional function to go from a time t to a time $t+1$. To perform $do(a)$ at time t means that an agent will take action a at time t.

Definition 2. Let $\mathbb{L}$ be inductively defined as follows:

- $p_t \in \mathbb{L}$
- for all ϕ, if $\phi \in \mathbb{L}$, then $\neg\phi \in \mathbb{L}$
- for all ϕ, ψ, if $\phi, \psi \in \mathbb{L}$, then $\psi \wedge \phi \in \mathbb{L}$

Where $p \in P$ and $t \in \mathbb{N}$

For example: if an agent holds the belief that p_t then it believes that p is true at time t, the agent can use this to hold beliefs of the future, if the agent believes at time t that p_{t+3} (by looking ahead) that means that the agent believes that in the future (in 3 non-specific time-units from time t) p will (still) be true.

3.2 Semantics

Every single atomic letter $p_t \in \phi$ where $\phi \in \mathbb{L}$ can be evaluated by a valuation function v. For every output of the function v it holds that the result must either be true or false, because of this we can consider the members of $\mathbb{L}$ booleans. For each time-unit t the members $\phi \in \mathbb{L}$ can have a different valuation. In other words, if p is true at time t (so p_t is true), it does not say anything about the truth-value of p at time $t+1$ (denoted by p_{t+1}).

A state S_t is a valuation of a subset of members of $\mathbb{L}$ at a certain time t. This model must be internally consistent. Meaning that if p_t holds, $\neg p_t$ cannot also hold. Every state can have one and only one member of the kind $do(a)_t$ it believes to be true for each value of t. This is the action that the agent would like to perform at this time t. Since we are using atomic actions it is not possible to perform multiple actions within a single time-unit. That leaves us with the following function definitions:

Definition 3.

- $v(S_t) \rightarrow P' \subseteq P$ where P' denotes a possible subset of P and $t \in \mathbb{N}$ is a valuation from a state to a set of propositions.

- $\mathit{Go}(S_t) \rightarrow \mathit{do}(a)$ where $a \in A$ is a function that assigns an action to a state.
- $\mathit{pre}(a) \rightarrow \phi$ where $a \in A$ and $\phi \in P_L$ is a function that assigns to a precondition of an action a propositional formula.
- $\mathit{post}(a) \rightarrow \phi$ where $a \in A$ and $\phi \in P_L$ is a function that assigns to a postcondition of an action a propositional formula.
- ϕ_t where $\phi \in P_l$ is an abbreviation for assigning the temporal value t to every propositional letter in ϕ. So $\forall p \in \phi, p \rightarrow p_t$ where $p \in P$. With this we can also assign temporal values to *pre* and *post*.

For example: a precondition of the action to buy a swimsuit might be to have money, represented by $pre(buyswimsuit$ or the formula $p \vee q$, where p represents having the cash on hand and q represents having a valid credit card. So if you wanted to buy a swimsuit in time t, and abbreviated it by $pre(buyswimsuit)_t$ this formula would change to $p_t \vee q_t$.

A timeline T is a set of states for all values of t in that model. So $T = \{S_0, S_1, S_2 ...\}$. These are akin to what is called a 'path' in [10]. The idea is that at time 0, everything that is defined by $v(S_0)$ is true, and the same goes for the other values of t. Simply put at time t the result of $v(S_t)$ is true.

We define a model as a tuple where $M = (T, v, \mathit{Go})$. Using this tuple we can start adding more conditions to our definition. So for M it must hold that: If $\mathit{Go}(S_t) = a$ then $\mathit{pre}(a) \in v(S_t)$ and $\mathit{post}(a) \in v(S_{t+1})$. This ensures that actions that have effects on future states actually affect them.

Now we will define truth functions for our models. It is important to note that these will go on a temporal basis as denoted by a subscript t.

Definition 4.

- $M \models p_t$ if and only if $p_t \in v(S_t)$ where $p \in P$ and $S_t \in T$.
- $M \models \mathit{do}(a)_t$ if and only if $\mathit{Go}(S_t) = \mathit{do}(a)$ and $M \models \mathit{pre}(a)_t$ and $M \models \mathit{post}(a)_{t+1}$ where $a \in A$ and $S_t \in T$.
- $M \models \neg\phi$ if and only if $M \not\models \phi$.
- $M \models \phi \wedge \psi$ if and only if $M \models \phi$ and $M \models \psi$.

Next we will define validity and satisfiability. We define the set of all possible models as $\mathcal{M}$. Valid formulas hold in every possible model $M \in \mathcal{M}$ and satisfiable formulas hold in some possible model $M \in \mathcal{M}$. So that gives us the definitions for the formulas $\phi \in \mathbb{L}$:

Definition 5.

- A formula ϕ is valid if and only if $\forall M \in \mathcal{M}$ it holds that $M \models \phi$.
- A formula ϕ is satisfiable if and only if $\exists M \in \mathcal{M}$ such that $M \models \phi$.
- A set of formulas Φ is valid if and only if $\forall M \in \mathcal{M}$ it holds that $\forall \phi \in \Phi, M \models \phi$.
- A set of formulas Φ is satisfiable if and only if $\exists M \in \mathcal{M}$ such that $\forall \phi \in \Phi, M \models \phi$.

4 Revision

In this section we will define the specifics of what our belief database and our intention database actually consist of, what it means for these databases to be consistent and a way to actually revise intentions and beliefs. I will also explore the connection between automatic planning and intentions.

4.1 Consistency

As stated in the related work section, we will adapt Shoham's Database Perspective [16]. This means our agent will consist of a planner, an Intention Database and a Belief Database. We will define our databases in this section. We will define an Intention Database $\mathbb{I}$ which consists of the intentions of the agent and a Belief Database $\mathbb{B}$ which contains the strong beliefs of the agent. This gives us the following definitions:

Definition 6.

- An intention I is a tuple which consists of an action and a time-unit such that $I = (a, t)$ where $a \in A$ and $t \in \mathbb{N}$. We will call the set of all intentions I_a
- An Intention Database $\mathbb{I} = \{(a_1, t_1), (a_2, t_2)...\}$ is a set of intentions where no two intentions can have the same time unit, so $\forall(a_i, t_i), \forall(a_j, t_j)$ if $t_i = t_j$ then $(a_i, t_i) = (a_j, t_j)$ where $(a_i, t_i), (a_j, t_j) \in \mathbb{I}$.
- A Belief Database $\mathbb{B} = \{\phi, \psi...\}$ consists of strong beliefs at a certain time point where $\phi, \psi \in \mathbb{L}$
- A Belief-Intention Database $\mathbb{BI} = (\mathbb{B}, \mathbb{I})$ is a tuple consisting of a Belief Database and an Intention Database.

Weak beliefs are determined by the postconditions of actions, as I will adopt the view on weak beliefs from both [16] and [20]. In this case, the agent will weakly believe in every postcondition associated with the intentions in the Intention Database $\mathbb{I}$, this means that $\forall(a, t) \in \mathbb{I}$ the agent weakly believes that $post(a)$ will hold in time $t + 1$. This is a good moment to reiterate that $pre(a)$ and $post(a)$ are abbreviations for formulas in P_l. In short: strong beliefs are facts of the world and weak beliefs are consequences of actions. We'll denote the set of weak beliefs as $\mathbb{W}$. So we get this definition for weak beliefs as:

Definition 7.

- $\mathbb{W} = \{post(a)_{t+1}, post(b)_{u+1}...\}$ where $(a, t), (b, u) \in \mathbb{I}$. The addition of 1 is here since you will believe the consequences of an action will hold in the time-unit after you've taken the action, not before.
- The set of strong and weak beliefs together $\mathbb{WB} = \mathbb{W} \cup \mathbb{B}$. This set consists of everything the agent believes to be true and everything the agent believes will be true.

Example: John wants to go swimming, for simplicity's sake John only deals with information which is relevant to go swimming. In order to go swimming he needs to have a swimsuit and it needs to be sunny outside. John knows it is sunny outside but he doesn't have a swimsuit. So John formulates a plan to buy a swimsuit at time 1 and go swimming at time 2. John weakly believes that he will have a swimsuit at time 2, but strongly believes that it will be sunny at time 2. So his Intention Database consists of two intentions, an intention to buy a swimsuit at time 1 and an intention to go swimming at time 2. His Belief Database consists of the preconditions of buying a swimsuit (such as having money or having a store nearby) and that it's sunny.

In formal terms John's databases look like this:

- $\mathbb{I}_{John} = \{(buyswimsuit, 1), (swim, 2)\ \}$

- $\mathbb{B}_{John} = \{pre(buyswimsuit)_1, sunny_1, sunny_2\}$
- $\mathbb{W}_{John} = \{post(buyswimsuit)_2, post(swim)_3\}$

In order to make sure our agent (and John) stay consistent we need to define consistency for these databases. This is very important for formulating a plan and will be very important for revising beliefs and intentions. For example: John wouldn't be able to go swimming if it wasn't sunny, so if John were to try to plan to go swimming when it is not sunny, the plan should come up as inconsistent. So we need to make sure we can see if the Belief-Intention Database is consistent, luckily Shoham [16] already has some conditions defined, we already mentioned them in Related Work but for clarity's sake they will be repeated here. We will also enumerate them so we can make them correspond to our formalized conditions.

1. Beliefs must be internally consistent
2. Intentions must be internally consistent.
3. At most one action can be intended for any given moment
4. If two intended actions immediately follow one another, the earlier cannot have postconditions that are inconsistent with the preconditions of the latter.
5. If an agent intends to take an action the agent cannot believe that its preconditions do not hold.
6. if the agent intends to take an action, the agent believes that its postconditions hold.

Now let's formulate these conditions for our Belief-Intention Database $\mathbb{BI}$. The numbers here correspond to one another, so our first formalization for our Belief-Intention Database corresponds to the first condition of Shoham (respectively for each condition):

1. $\mathbb{B}$ must be satisfiable, that is to say that the set of formulas that make up $\mathbb{B}$ must be satisfiable.
2. This one we don't need to formally define as our Intention Database $\mathbb{I}$ cannot be internally inconsistent since an intention I is a tuple, not a logical proposition (for example: a negation of our intention such as $\neg(a, t)$ doesn't exist). Thus, this condition automatically holds.
3. This was already defined for our Intention Database $\mathbb{I}$ but we will do so again: $\forall(a_i, t_i), \forall(a_j, t_j)$ if $t_i = t_j$ then $(a_i, t_i) = (a_j, t_j)$ where $(a_i, t_i), (a_j, t_j) \in \mathbb{I}$.
4. $\forall(a, t), pre(a)_t \cup \mathbb{WB}$ must be satisfiable where $(a, t) \in \mathbb{I}$.
5. In our model, condition 4 and 5 are logical equivalents, so as long as 4 holds, 5 holds.
6. $\forall(a, t), post(a)_{t+1} \in \mathbb{WB}$ where $(a, t) \in \mathbb{I}$

These conditions would be enough if $\mathbb{I}$ has to necessarily fill every empty time-unit, but it doesn't have to... So we can define an additional condition to make sure the gap over empty time-units can be bridged:

7. For each time-unit t that is smaller than the largest time unit in the Intention Database $\mathbb{I}$ which does not occur in the Intention Database, we will call this set of missing time-units T. Which gives us our condition: $\exists M \in \mathcal{M}$, such that $\forall t \in T$ it must hold that $\exists a \in A$ such that $M \models (pre(a)_t \cup post(a)_{t+1} \cup \mathbb{WB})$. This should almost always be the case since we have a do nothing action.

Our model should capture all these conditions. So we can check consistency by checking satisfiability. So $(\mathbb{B}, \mathbb{I})$ is consistent if and only if $\forall(a,t) \in \mathbb{I}, do(a) \cup \mathbb{B}$ is satisfiable. For this I would like to introduce a consistency function *cons*:

Definition 8.

- $cons(\mathbb{B}, \mathbb{I}) \rightarrow \{True, False\}$ is a function that checks consistency for a Belief-Intention Database $\mathbb{B}, \mathbb{I}$. If $\mathbb{B}, \mathbb{I}$ is consistent it will output $True$, if $\mathbb{B}, \mathbb{I}$ is inconsistent it will output $False$.

With this, we can already check consistency. But as this would be painstakingly slow for a human, we can also check consistency with an algorithm. A propositional logic SAT solver [3] can solve the Boolean Satisfiability Problem of any propositional logical formula. All we need to do is to encode our logic and conditions into propositional logic and then a standard SAT solver for propositional logic could check consistency for us. So we will make a (large) propositional formula and if the SAT solver tells us our formula is satisfiable then we can say it is consistent. Our formula F will be a conjunction with the following conjuncts:

- Every separate formula in $\mathbb{WB}$ becomes a conjunct in formula F.
- For every Intention $(a,t) \in \mathbb{I}$ with a value of t that we haven't seen before, $pre(a)_t$ becomes a conjunct in formula F. If we have seen a value of t before, the formula $p \wedge \neg p$ becomes a conjunct to guarantee inconsistency.
- For each time-unit t that is smaller than the largest time unit in the Intention Database $\mathbb{I}$ which does not occur in the Intention Database $\mathbb{I}$ a disjunction formula will become a conjunct. The formula consists of a disjunction D_t of conjunctions of the pre- and postconditions of all available actions. So $\forall a((pre(a)_t \wedge post(a)_{t+1})$ becomes a disjunct in the disjunction formula D for each missing time-unit, then formula D_t becomes a conjunct in formula F.

Now we have our (large) conjunctive formula F and it's almost ready to be put into a SAT solver. The only problem we still have is that the propositional letters have temporal properties (for example: the t in p_t). The solution to this is simple however, each unique combination of propositional letter and time becomes a separate propositional letter. So the formula $(p_1 \wedge p_2 \wedge p_1)$ becomes $(p \wedge r \wedge p)$. So if we do so we convert all temporal propositional letters into regular propositional letters, and now our formula F is ready to be plugged into a SAT solver. What we did is effectively encode our temporal logic, actions and conditions into propositional logic.

So let's assume that in our example John is the planner and the Belief-Intention Database is his smart calendar, if John attempts to plan an action that's not possible (such as going swimming while it's not sunny) the smart calendar will tell John that his plan is inconsistent and that he should readjust it. But what if we can (partially) automate this process?

4.2 Intention Revision

So now we have our Belief-Intention Database for which we can check consistency, we can finally move on to (partially) automating building an Intention Database $\mathbb{I}$ and revising it. First we will consider a simple case in which the Intention Database isn't automatically built but can revise automatically. And second we will consider a more sophisticated case in which an Intention Database can be automatically built and revised. Finally, we will consider a system for intention revision which is a combination of the two previous systems. It is good to note that while revising intentions, our strong Belief Database $\mathbb{B}$ should remain untouched, but our weak beliefs $\mathbb{W}$ can (and probably will) be affected.

The idea of intention revision is intuitive. If an agent tries to add an intention I to their Intention Database $\mathbb{I}$ we get a new Intention Database $\mathbb{I}' = \mathbb{I} \cup I$. If this new Belief-Intention Database $(\mathbb{B}, \mathbb{I}')$ is consistent, then there is no problem and the intention can be added. However, if it turns out $(\mathbb{B}, \mathbb{I}')$ is inconsistent we have a problem, and we're going to have to revise.

4.2.1 Maximal Consistent Subset Revision

A method that has been used a lot in other works (such as [16] and [20]) is revising $\mathbb{I}$ with the maximal consistent subset. For finding the maximal consistent subset of Intentions in $(\mathbb{B}, \mathbb{I} \cup I)$ we can simply use any MAX-SAT solver by utilizing repeated calls on our previously mentioned SAT-solver where we only change elements of the set $(\mathbb{B}, \mathbb{I}$ and leave the set of formulas $\mathbb{B}$ untouched, as has already been done in [2] (so we can simply adopt their MAX-SAT implementation). We can still formalize the maximal consistent subset by way of a function *mcs*, which we will define here:

Definition 9.

- $mcs(\mathbb{B}, \mathbb{I}, I) \rightarrow \mathbb{I}'$ is a function that assigns a subset $\mathbb{I}'$ of an Intention Database $\mathbb{I} \cup I$ such that $I \in \mathbb{I}'$ and such that there is no other possible subset of $\mathbb{I} \cup I$ that is consistent and has more elements than $\mathbb{I}'$. So $\neg \exists \mathbb{J}$ such that $((|\mathbb{J}| > |\mathbb{I}'|) \ \wedge \ (cons(\mathbb{B}, \mathbb{J}) = True) \ \wedge \ I \in \mathbb{J})$ where $\mathbb{J}, \mathbb{I}' \subseteq \mathbb{I} \cup I$. If multiple valid sets of I' exist, one will be chosen randomly.

So let's go back to John, for simplicity's sake we will assign the time-units as such: 1 corresponds to the morning, 2 corresponds to the afternoon and 3 corresponds to the evening. John wants to buy a swimsuit in time 1 (morning) and go swimming in time 2 (afternoon). His grandma might call him over for dinner, which means he would want to go to have dinner with Grandma in time 3 (evening). Assuming that there the preconditions for going to dinner hold, this should be no problem for John as he didn't have plans for that moment anyway. In order to be sure, we can simply check the consistency using the method described in the previous section. If the check comes out as consistent then there's no problem. John can simply add the new intention to his Intention Database on his smart calendar.

But what if Grandma doesn't call John over for dinner, but for lunch? Then John would need to add having lunch with Grandma in time 2 (afternoon) to his Intention Database, which would clash with going swimming, which is currently planned for time 2 (afternoon). So now John needs to revise his intentions. So John's smart calendar will look for the maximal consistent subset with a requirement that the new intention must be part of it, using our *mcs* function. So the smart calendar will remove swimming, but keep his other intentions since there is no reason to remove them. But if John can't go swimming, does he still want to buy a swimsuit? Maybe he would rather spend his morning reading a book now that he knows he can't go swimming. This simple system of revision has no way of knowing or revising it in such a manner, it can only remove intentions that are inconsistent with others while giving priority to the newly added intention. I would like to propose a different system, which is slightly closer to how humans revise their intentions when new information presents themselves.

4.2.2 Intention Revision and Automatic Planning

The system I would like to propose further explores the connection between automated planning and intention, which is a connection that A. Herzig et al. [9] argued should be explored further. This system was inspired by the dynamic programming approach to the zero-one knapsack problem as described in [19]. As there is a value for each item in the zero-one knapsack problem, there will be a happiness value for each action in this system. So John would give his smart calendar a Belief Database $\mathbb{B}$ and a set of all available actions A. John has happiness values assigned to each action, for example: going swimming would have a happiness value of 100, but buying a swimsuit has a happiness value of -10 and reading a book has a happiness value of 30. So, going back to just 3 time-units John could: Buy a swimsuit in the morning, go swimming in the afternoon and read in the evening, granting him a total happiness value of $-10 + 100 + 30 = 120$. John could also read all day granting him a happiness value of $30 + 30 + 30 = 90$. His smart calendar should choose the former option for him. So we should formalize a few things before we go onward:

Definition 10.

- $\mathit{happy}(a) \rightarrow z$ where $a \in A, z \in \mathbb{Z}$ is a function which assigns a happiness value to an action, the happiness value can be any integer.
- $\mathit{totalhappy}(\mathbb{I}) \rightarrow z$ where $\mathbb{I}$ is an Intention Database and $z \in \mathbb{Z}$ is a function that calculates the cumulative happiness value of all actions of intentions in the database.
- $\mathit{build}(A, \mathbb{B}, t) \rightarrow \mathbb{I}'$ is a function which constructs an Intention Database $\mathbb{I}'$ from a set of available actions A, a Belief Database $\mathbb{B}$ and a natural number t such that the resulting Belief-Intention Database $(\mathbb{B}, \mathbb{I}')$ is consistent and such that there is no other Intention Database whose total happiness value is greater, so that means that: $\neg\exists\mathbb{J}$ such that $((\mathit{totalhappy}(\mathbb{J}) > \mathit{totalhappy}(\mathbb{I}')) \wedge (\mathit{cons}(\mathbb{B}, \mathbb{J}) = \mathit{True}))$ where $\mathbb{J}$ is an Intention Database. It considers all possible permutations of $\mathbb{I}$ that can be made using the actions in A up to a certain time-unit t. If multiple valid sets of I' exist, one will be chosen randomly.

Now we can build an Intention Database using the build function. We can add more conditions to model it even more closely to how we as humans plan our time, we add a tie-breaker for the *build* function where Intention Databases with less total elements (thus, actions) would have priority since they give us more free time. So now our current plan Intention Database is always optimal given the available actions, so in order to revise, the set of available actions would have to change. This change can happen in two ways: we could discover a new action a and add it to the set of all actions A, which we then use in the *build* function. Another possibility is that our Belief Database $\mathbb{B}$ changes, which means that actions that we couldn't do before might become available now. Let's consider both options with our running example:

In the first case, John is a bit forgetful and doesn't realize he could go have lunch with Grandma. That is until Grandma calls John and John is reminded of the availability of the action and he adds it to his smart calendar and assigns it a happiness value of 200, overshadowing the happiness value of going swimming or reading. Now he simply adds it to his available actions A' and his smart calendar will simply use the *build* function to generate a new Intention-Database which has an optimal happiness value for John.

In the second case, a prerequisite to have lunch with Grandma would be getting a call for her, so John knows the action is available, but can't perform it because the preconditions don't hold, in formal terms $\mathit{pre}(\mathit{grandmalunch}) \notin \mathbb{WB}$. So John gets a call from Grandma that she is available for lunch, meaning that $\mathit{pre}(\mathit{grandmalunch}) \in \mathbb{B}$, and thus also $\mathit{pre}(\mathit{grandmalunch}) \in \mathbb{WB}$. For now, we will assume that $\mathbb{B}' = \mathit{pre}(\mathit{grandmalunch}) \cup \mathbb{B}$ is consistent, since we do not need to revise our beliefs as long as a new belief is consistent. So John updates the Belief Database $\mathbb{B}$ of his smart calendar and the smart calendar simply uses the *build* function again to generate a new Intention Database with optimal happiness value for John.

In both cases, considering our available actions with happiness values in round brackets $\mathit{read}(30)$, $\mathit{buyswimsuit}(-10), \mathit{swim}(100), \mathit{grandmalunch}(200)$, his new Intention Database $\mathbb{I}$ looks like $\mathbb{I} = \{(\mathit{read}, 1), (\mathit{grandmalunch}, 2), (\mathit{read}, 3)\}$.

Now we have a concrete connection between intentions and automatic planning. As we have defined a way to automatically plan a course of actions within our logical model using intentions. This is what [9] argued should be looked into further, and so we have, however small this contribution may be.

4.2.3 Intention Revision, Revisited

This new system is closer to planning than revision, as it builds (and rebuilds) the intention database whenever something changes. While it is interesting, it is also quite drastic, as John and his smart calendar do not care about any previous commitments, and as both [16] and [20] say, intention is a commitment toward time. So now we can introduce a new system which combines the maximal consistent subset with the new happiness planner, we shall call it: maximal happiness subset. As the name implies this system would determine the subset of intentions that generate maximal happiness with the new intention included. We shall call this function *mhs* (maximal happiness subset) which we can define as follows:

Definition 11.

- $mhs(\mathbb{I}, I) \rightarrow \mathbb{I}'$ is a function that assigns a subset $\mathbb{I}'$ of $\mathbb{I} \cup I$ such that $I \in \mathbb{I}'$ and such that there is no other possible subset of $\mathbb{I} \cup I$ that is consistent and has more happiness and more elements than $\mathbb{I}'$. So $\neg\exists\mathbb{J}$ such that $((|\mathbb{J}| \geq |\mathbb{I}'|) \wedge (totalhappy(\mathbb{J}) > totalhappy(\mathbb{I}')) \wedge (cons(\mathbb{B}, \mathbb{J}) = True) \wedge I \in \mathbb{J})$ where $\mathbb{J}, \mathbb{I}' \subseteq \mathbb{I} \cup I$. If multiple valid sets of I' exist, one will be chosen randomly.

Using *mhs* to revise, John's smart calendar would remove buying a swimsuit for him since it knows John doesn't want to buy a swimsuit if he can't go swimming after. This way of intention revision still keeps actions that an agent is committed to looking for ways to maximize your happiness. This gives it an edge over using standard maximal consistent subset as it can recognize intentions that are merely used as a means to an end, but not as goal in of itself and remove these intentions should their goal be removed and these do not add value on their own.

So now we have a way of revising intentions using the *mhs* function, but when John got a call from his grandma we assumed that the new belief was consistent with the old Belief Database. But what if it is inconsistent? Then we would require a way to revise beliefs. However, it is good that we defined intention revision first, since we just saw that intentions can change as a consequence of changed beliefs.

4.3 Belief Revision

As was said in the related work section, we adopt a way of revising beliefs that adheres to the six basic AGM postulates [1], which are conditions that a belief revision procedure should adhere to in order to be considered reasonable. We will use $\mathbb{B}$ to denote the original belief set, ϕ to denote the new belief that we add during our procedure and $\mathbb{B}'$ to denote the belief set after revision. The six basic postulates are as follows:

- Closure. Closure means that the result of a belief revision on a deductively closed belief set, should also be a deductively closed belief set so: If $\mathbb{B}$ is a deductively closed belief set then $\mathbb{B}'$ is a deductively closed belief set. In our Belief Database we leave deductive closure as implied, but since we follow the same rules as propositional logic, we can deductively close our set of formulas under standard propositional logic closure.

- Success. Success means that after revising the original belief set $\mathbb{B}$ with a belief ϕ, the new belief ϕ is actually a part of the new set $\mathbb{B}'$. So if $\mathbb{B}$ is revised with ϕ, then $\phi \in \mathbb{B}'$.

- Inclusion. Inclusion means that when revising $\mathbb{B}$ with ϕ no other beliefs that aren't already in the original belief set $\mathbb{B}$ get added. So $\mathbb{B}' \subseteq (\mathbb{B} \cup \boldsymbol{\phi})$, where $\boldsymbol{\phi}$ is the set of all formulas that are the logical consequence of ϕ.

- Vacuity. Vacuity means that beliefs that do not need to be removed, will not be removed. So if $\neg\phi \notin \mathbb{B}$ then $\mathbb{B}' = (\mathbb{B} \cup \boldsymbol{\phi})$, where $\boldsymbol{\phi}$ is the set of all formulas that are the logical consequence of ϕ.

- Consistency. Consistency means that consistency is retained. So if $\mathbb{B}$ is consistent and ϕ is consistent, then $\mathbb{B}'$ must also be consistent. Similarly, $\mathbb{B}'$ should only be inconsistent if ϕ is inconsistent.

- Extensionality. Extensionality means that if two logically equivalent formulas ϕ and ψ independently revise $\mathbb{B}$, the result should be the same.

Since my logical model is very close to propositional logic, I can adopt an existing form of Belief Revision, I shall adopt the iterated revision operator from A. Darwiche and J. Pearl [7]. The operator is based on a proposal by Spohn for revising ordinal conditional functions [18]. This operator can revise beliefs using a set of formulas $\mathbb{B}$ and a new belief ϕ while adhering to the AGM postulates (in fact, they believed the AGM postulates to be too weak and made their revision operator satisfy even stronger properties). As such, this revision operator is compatible with our Logical Model and Belief-Database $\mathbb{B}$. We shall define this way of revising beliefs as:

Definition 12.

- $rev(\mathbb{B}, \phi) \rightarrow \mathbb{B}'$ where $\mathbb{B}$ is a Belief Database, $\phi \in \mathbb{L}$ is a new belief and $\mathbb{B}'$ is the revised Belief Database.

We ought to keep in mind that whenever the Belief Database $\mathbb{B}$ is updated, we should also update our Intention Database $\mathbb{I}$ (using *mcs*, *build* or *mhs*). We must revise our Intention Database when the Belief Database is changed because it is possible that some actions that were possible are no longer possible and in the case of *build*, new actions might be possible. With this system, we can revise beliefs in such a way that it can be considered reasonable (as it follows the AGM postulates). With this system in place we have a system for both intention and belief revision. John can now fully rely on his smart calendar to help him out.

5 Conclusion

5.1 Findings and Contributions

We have successfully created a logical model and shown that it is capable of revision and that existing tools for propositional logic can be used, so to answer the question: 'Can a logical model for a BDI agent be made using the Database Perspective while staying close enough to propositional logic that existing tools can be applied?" is yes it can. By building a logical model we have contributed to the field of BDI (software) research since a major part of this field is finding new logical models which can be used to implement and reason about BDI agents. We have also shown that it can be valuable to stay close to propositional logic so that available tools (such as SAT) can be utilized for the logical model.

We have also accomplished our secondary goal of finding, showing and strengthening the connection between intentions and automatic planning, this is something A. Herzig et al. suggested in [9]. By combining a logical model designed for BDI agents with a procedure for automatic planning we have shown a way to use intentions in this logical model for the purpose of automatic planning. This system is a novel way of building and revising intentions which contributes to the BDI field as a whole.

5.2 Limits

This logical model and revision is currently limited to atomic actions and linear time. This makes it less expressive than other already existing logics such as the Parameterized-time Action Logic (PAL) from [20]. However, by limiting it in this way our logic is very close to propositional logic which allows us to utilize existing tools for propositional logic such as SAT with very little modification.

5.3 Extensions

As previously stated, this model is limited to a linear timeline. While the automated planning using the *build* function considers all permutations (and thus, all timelines) in the end it only ever really utilizes one. This is something I think has room for extension, as planning for multiple cases might be beneficial and would allow the model to work with degrees of uncertainty. For example, if an agent is not sure of whether the weather will be sunny or rainy tomorrow, it would be wise to have a plan for both occasions. A subroutine for this could already be implemented by using the *build* function multiple times, just with different Belief Databases $\mathbb{B}$. Although a more computationally optimized solution might be preferred for an actual artificial agent.

Another possibility would be to create a custom Belief Revision system, utilizing a refined happiness-value for belief, perhaps it is even possible to (partially) model confirmation bias as we observe it in humans into the belief revision procedure [12]. This wouldn't make for a very good BDI agent for planning, but mayhaps it could function as a Cognitive Model.

Further future work is possible by expanding this logical model, further exploring the link between intention and automatic planning and refining the assignment of happiness-value within our automatic planning to further automate it or to make it properly reflect desire to ensure the "Desire" is not lost in the name Belief-Desire-Intention. Currently the database perspective as proposed by Shoham [16] only has a Belief Database and Intention Database.

References

[1] Carlos E. Alchourrón, Peter Gärdenfors, and David Makinson. On the logic of theory change: Partial meet contraction and revision functions. *Journal of Symbolic Logic*, 50(2):510–530, 1985.

[2] Carlos Ansótegui, Maria Luisa Bonet, and Jordi Levy. Sat-based maxsat algorithms. *Artificial Intelligence*, 196:77–105, 2013.

[3] Mordechai Ben-Ari. *Propositional Logic: SAT Solvers*, pages 111–129. Springer London, London, 2012.

[4] M. E. Bratman, D. J. Israel, and M. E. Pollack. Plans and resource-bounded practical reasoning. *Computational Intelligence*, 4(4):349355, 1988.

[5] Michael Bratman. *Intention, Plans, and Practical Reason.* Cambridge: Cambridge, MA: Harvard University Press, 1987.

[6] Philip R. Cohen and Hector J. Levesque. Intention is choice with commitment. *Artificial Intelligence*, 42(2):213–261, 1990.

[7] Adnan Darwiche and Judea Pearl. On the logic of iterated belief revision. *Artificial Intelligence*, 89(1):1–29, 1997.

[8] Donald Davidson. Actions, reasons, and causes. *The Journal of Philosophy*, 60:685–700, 1963.

[9] Andreas Herzig, Emiliano Lorini, Laurent Perrussel, and Zhanhao Xiao. Bdi logics for bdi architectures: Old problems, new perspectives. *KI - Künstliche Intelligenz*, 31, 10 2016.

[10] Andreas Herzig, Laurent Perrussel, Zhanhao Xiao, and Dongmo Zhang. Refinement of intentions. In Loizos Michael and Antonis Kakas, editors, *Logics in Artificial Intelligence*, pages 558–563, Cham, 2016. Springer International Publishing.

[11] Richard Holton. Partial Belief, Partial Intention. *Mind*, 117(465):27–58, 01 2008.

[12] Saul M Kassin, Itiel E Dror, and Jeff Kukucka. The forensic confirmation bias: Problems, perspectives, and proposed solutions. *Journal of applied research in memory and cognition*, 2(1):42–52, 2013.

[13] Ingrid Nunes, Frederico Schardong, and Alberto Schaeffer-Filho. Bdi2dos: An application using collaborating bdi agents to combat ddos attacks. *Journal of Network and Computer Applications*, 84:14–24, 2017.

[14] Anand S Rao, Michael P Georgeff, et al. Bdi agents: From theory to practice. In *Icmas*, volume 95, pages 312–319, 1995.

[15] Smitha Rao and A.N. Jyosthsna. Bdi applications and architectures. In *IJERT*, volume 2, Issue 2, February 2013.

[16] Yoav Shoham. Logical theories of intention and the database perspective. *Journal of Philosophical Logic*, 38:633–647, 12 2009.

[17] Yoav Shoham. Why knowledge representation matters. *Communications of the ACM*, 59(1):47–49, 2015.

[18] Wolfgang Spohn. Ordinal conditional functions: A dynamic theory of epistemic states. In *Causation in decision, belief change, and statistics*, pages 105–134. Springer, 1988.

[19] P. Toth. Dynamic programming algorithms for the zero-one knapsack problem. *Computing*, 25(1):29–45, Mar 1980.

[20] Marc van Zee, Dragan Doder, Leendert van der Torre, Mehdi Dastani, Thomas Icard, and Eric Pacuit. Intention as commitment toward time. *Artificial Intelligence*, 283:103270, 2020.