Comparison of different Mental Model Theory implementations

Julia Mertesdorf, Christian Breu

August 2, 2018

Abstract

This paper shortly presents the Mental Model Theory introduced by Philip Johnson-Laird and Ruth M. J. Byrne and the Preferred Mental Model Theory, a variation of the Mental Model Theory which was invented by M. Ragni and M. Knauff [3]. Three different implementations are introduced and evaluated. The first two implementations written by P. Johnson-Laird cover different domains of the Mental Model Theory, in particular Spatial and Temporal reasoning. The third implementation written by A. Dietz [5] is based on the Weak Completion Semantics and attempts to model the Preferred Mental Model Theory for simple Spatial relations. To illustrate the relatively new WCS-approach, a full example for the model construction is given and directly compared to the according model-construction-processes of the similar Spatial Model. The example is followed by a cross-comparison between the two implementations by P. Johnson-Laird and afterwards a semantic comparison between all three implementations is drawn. At the end, we present a proposal as well as some extensions on how to translate the WCS-approach, which was originally implemented in Prolog, into a Python-program.

1 Introduction

1.1 The Mental Model Theory

The Mental Model Theory [2] was developed by Philip Johnson-Laird and Ruth M.J. Byrne and attempts to explain the thinking process of individuals behind drawing inferences. The key idea of the theory is that when individuals are confronted with a problem requiring deduction, they construct a mental model of the problem first and afterwards derive a solution based on this mental model. According to the Mental Model Theory, in case the given conclusion holds with the first constructed mental model of the problem, all possible mental models are constructed for a given deduction problem to check whether all potential models support the conclusion. Thus, the order of the premises was assumed to be irrelevant as all models are constructed anyway to check whether the conclusion holds.

1.2 The Preferred Mental Model Theory

The Preferred Mental Model Theory, developed by M. Ragni and M. Knauff [3], is based on the Mental Model Theory and the central assumption of it is that humans consider only certain mental models instead of all possible mental models. These are called the 'preferred mental models' and depend on the given presented order of premises of a deduction problem. The Preferred Mental Model Theory comprises three phases, which are model construction, model inspection and model variation. In the first phase, only one model is constructed, which is dependent on the order of the premises. In the second phase, it is verified whether the given conclusion holds in the preferred model. The third phase starts the variation of the initial preferred model with the least possible operations. This last phase is however often skipped by subjects, as most of them infer their answer already after verifying or falsifying the given conclusion with the first and preferred model.

1.3 Weak-Completion-Semantics

The Weak Completion Semantics are a mechanism of logical representation and inference and with that, a Logical Framework that can be used for Cognitive Modelling. The WCS are based on Logic Programs and the three-valued Łukasiewicz Logic. Thus, the Weak Completion Semantics are usually used as a tool in the Inference-rules-approach [4], which opposes the Mental Model Theory in its explanations on how human reasoning works. The Inference-rules-approach states that humans make use of logical rules of inference in order to infer a conclusion in a given deduction problem, instead of creating mental models to symbolize the problem. However, it is shown by A. Dietz (in [5], chapter 6.4) that Logic programs under the Weak Completion Semantics can also be used to model the Preferred Mental Model Theory for Spatial Reasoning problems. This approach is presented in a later chapter but is however limited to the first two phases mentioned in 1.2.

2 The Implementations

In the following section, all implementations and different approaches are generally explained to be compared in a later section.

2.1 Temporal Model

The Temporal Model, written in Lisp by Philip Johnson-Laird [1], is the implementation of the temporal domain of the Mental Model Theory he invented. This program solves temporal deduction problems, usually consisting of 4 premises like "Event a happens after / before / while event b" and a question at the end, for instance: "What is the relation between the events c and e?"
The characteristic of the implementation is that always all possible mental models for one given deduction problem are constructed. That means, in case a given deduction problem can lead to several different models, which all illustrate the given premises correctly, the program will construct all of these models and not just the first fitting model. The resulting amount of different models for one deduction problem can be interpreted as an indicator for how difficult a deduction problem is. For instance, if a deduction problem only has one potential mental model (called a one-model problem), then it is a rather easy problem. On the other hand, if a problem leads to many different models, which may also lead to different conclusions, it is more difficult to solve. These kind of problems are referred to as multiple-model problems.
Overall, the temporal model implementation is able to recognize indeterminacies as well as inconsistent last premises, combine different models into one, answer a given question about a deduction problem and verifying a given conclusion (which is the last premise in a problem), in case there are only premises but no question given. Moreover, the model incorporates a simple working memory, which counts the current amount of potential models for the problem. In case the problem contains a question and the amount of models exceed the capacity of the working memory, a backtracking process is started, trying to solve the problem with only the relevant premises for the two referred events in the question.

2.2 Spatial Model (Space-5)

The Spatial Model from Philip Johnson-Laird[1] is an implementation of the Mental Model Theory for spatial relations. The program constructs an initial model from a given set of premises. The given conclusion to be verified is always the last parsed premise in the premise-set of a spatial deduction problem. After constructing the initial mental model, the program checks whether the conclusion holds in this model.
If the conclusion holds in the initial model, the program will modify this model, trying to make the conclusion to not hold. In this phase the program tries to construct a model in which every premise holds except the conclusion-premise. In case there is a model that falsifies the conclusion (with the conclusion not

holding in the model), this model will be returned and the search for counterexamples stops.
To get the falsified premise, the program negates the relation between the two objects (for instance transforming a left- to a right-relation). If the conclusion is found to be false in the initial model, the program will search for a model where the conclusion holds. Therefore, the program will, analogous to when the conclusion did hold in the initial model, try to make the conclusion to hold while all other premises need to hold as well. In order to make the premise hold in the model, the program will swap the two objects of the premise in the model and check if the premise holds then. It will also change the position of one of the objects in the premise, that needs to hold in the model before trying to verify the model again.
In case it was possible to falsify an initially holding conclusion, or verify a conclusion that did not hold in the initial model, that means that the initial model might not have been correct, or that there are several different models leading to different conclusions.
It is worth mentioning that the initial model will only be modified in certain ways to make a premise hold. The program moves objects to a position where the premise holds, but these possible positions aren't tested exhaustively. Therefore at least some models will be left out of the search for alternative models.

2.3 Preferred Mental Model implemented in WCS

The Weak-Completion-Semantics-approach written by A. Dietz[5] is an implementation of the Preferred Mental Model Theory for simple spatial relations. The approach however covers only the first two phases of the Preferred Mental Model Theory, which is the model construction and model inspection. Both of these phases are processed at the same time in Logic Programs under Weak Completion Semantics. Moreover, the approach is only implemented for the two spatial relations "left" and "right", but can be extended to other spatial directions and more complex relations.
The Weak-Completion-Semantics approach processes all premises and creates a model in the form of a logic program. Since logic programs do not consider the order of the given premises, which is one of the key aspects that differentiate the Preferred Model from the overall Mental Model Theory, A. Dietz invented a mechanism to also consider the order. The order of the premises is considered with the help of a time index i, which determines when a certain premise should be processed. For instance, if a problem consists of three premises, they will be labeled with the time indices 1, 2 and 3 in their given order.

In the beginning, all given premises are added with their time index to the logic program. For instance, the first premise $left(car, house)$ would be added as the clause $l(car, house, 1) \leftarrow \top$ to the logic program. Moreover, a Closed World Assumption is made at time-step 1, meaning that all permutations of two objects referred to in the premises are added as a negative fact to the logic program. For instance, $l(car, house, 1) \leftarrow \bot$ is added as a fact. Since the logic

program is weakly completed afterwards, all relations that were given with the premises remain true in the Logic program, since $l(car, house, 1) \leftrightarrow \top \vee \bot$ equals $l(car, house, 1) \leftrightarrow \top$.
There are some more helper relations introduced in order to process the premises at their given order, like "$ln(object_1, object_2, i)$", meaning that $object_1$ is the direct left neighbour of $object_2$; "$ol(object, i)$" and "$or(object, i)$" meaning that at time-step i, the left / right space next to the object is empty. As in the beginning, no spaces next to objects are occupied, $ol(object, 1) \leftarrow \bot$ and $or(object, 1) \leftarrow \bot$ is added for all objects mentioned in the premises.
After all negative facts regarding the Closed-World-Assumption are set, the program can start to place objects. If in phase i, $object_1$ should be placed to the left of $object_2$, and both the space to the right of $object_1$ and to the left of $object_2$ are empty, $object_1$ is placed as the direct left neighbour of $object_2$. This relation holds until the preferred model is found, which means that the rule $ln(object_1, object_2, i+1) \leftarrow ln(object_1, object_2, i)$ always holds in the logic program.
In that way, more and more rules are added to the logic program which encode neighbouring-relations between two objects and empty / occupied spaces next to them. Additionally, the rule $left(object_1, object_2) \leftarrow ln(object_1, object_2, n)$, (n is the number of premises) is added to check whether the conclusion holds. Two more rules are needed to encode the relation between left and right and the transitivity of the left-relation: $right(object_1, object_2) \leftarrow left(object_2, object_1)$; $left(object_1, object_3) \leftarrow left(object_1, object_2) \wedge left(object_2, object_3)$. After applying and adding all possible rules in each phase until every premise was processed, the program is finished. The conclusion can be verified by checking whether the relation given in the last premise (the conclusion-premise) holds in the least fixed point of the logic program.

3 Comparison of Model construction and verification: Spatial Model and WCS-approach

This section discusses the similarities and differences of the Weak-Completion-Semantics-approach of A. Dietz and the Spatial Model (space-5) of P. Johnson-Laird with regard to the basic model creation steps and the verification. These two models both implement the Preferred Mental Model Theory and are therefore quite similar (the Spatial Model is originally an implementation of the Mental Model Theory, we however came to the conclusion that it would also fit the Preferred Mental Model Theory, as will be discussed in a later section). In the upcoming small examples for the WCS-approach, the logic programs will not be fully displayed, only the for the explanation necessary and positive clauses to actually represent the model are shown. For a full example, see the following chapter.

3.1 Creation of a new model with two objects

In space-5, a new model is constructed by creating a list of three dimensions. Both objects are added into the list according to their relation. For instance, after parsing the premise 'the x is left to the y', the program first adds 'y' to the model at the origin (0, 0, 0) and afterwards 'x' to the model at the coordinates of 'y' together with the relation, which is encoded as relative coordinates. The relation which encodes left is (-1, 0, 0) in the program, so 'x' will be added to the model at the coordinates (-1, 0, 0).
In the Weak-Completion-Semantics-approach, a new clause encoding the direct neighbourhood is added to the logic program, since the left and right space next to each object is initially empty. This clause encodes the relation, the two objects and the additional time component.
A short example to illustrate the process in the WCS-approach: The premise left(x, y) is added as l(x, y, 1) $\leftarrow \top$ to the logic program. Furthermore the clauses ol(y, 1) $\leftarrow \bot$ and or(x, 1)$\leftarrow \bot$, encoding that the left space next to y and the right space next to x are empty, are added and result in the clause ln(x, y, 1) $\leftarrow \top$ being added to the logic program in the next iteration.
Conclusively, the creation of a new model is very similar and semantically equivalent. The only difference is the way the model is represented. In space-5, the model is actually built while the WCS-approach only represents the relations between the objects of the model.

3.2 Adding new objects to a model

In the Spatial Model (space-5), the new object is inserted at the coordinates of the object which is already in the model, added with the relation-coordinates (since the relation in the Spatial and Temporal Model of P. Johnson-Laird is encoded as a tuple of three numbers). In the usual case, the Spatial Model tries to insert the item at these coordinates and checks whether the spot is already occupied. In case it is, the specified relation (the tuple of three numbers, where

two of the numbers are 0) is added again to the the initial coordinates where the item was supposed to be inserted. This step is repeated until the first free spot is found and the item can be inserted.
There is however an exception for when the relation equals (0, 0, 0) and when there already is another object at the specified coordinates. In this case, the Spatial Model will insert the item anyway and as a result, two objects are temporarily at the same place. In case that happens, this instance of the model will be discarded in almost every case in a later step, due to some premises not holding anymore.
For example, if there is a model consisting of x at coordinates (0, 0, 0) and y at coordinates (1, 0, 0), the new premise 'z is in front of y', will lead to the insertion of z at the coordinates (1, 1, 0). The coordinates of the new item z are calculated out of the coordinates of y(1, 0, 0), added with the tuple (0, 1, 0) which encodes the relation 'in front of'. Since this spot is not occupied by any other object, z will be directly placed at the given coordinates instead of adding this relation to the current coordinates until a free spot is found. If there would be already another object at the coordinates (1, 1, 0), the program would add the relation again to these coordinates and try to add 'z' at the coordinates (1, 2, 0) next.

In the WCS-approach, a new clause is added to the program according to the given relation of the two objects from the premise. If the place next to the already existing object in the model is occupied, the program will find the first free place (first-free-fit strategy) to add the new object with the help of the free-space rules $lo(object, i)$, $ro(object, i)$ and the transitivity of the relations.
As a quick example, the model which consists of ln(x, y, 1) and the new premise left(z, y) are given. The program will first add the clause l(z, y, 2) to the program, meaning that in time-phase 2, z should be added to the left of y. Since x is the left neighbour of y, ol(y, 2) $\leftarrow \top$ holds in the program and therefore z can't be inserted as the direct left neighbour of y. There is a rule in the WCS-approach implementing the first-free-fit strategy to add an object 'a' next to another object 'b' when the direct space next to the object 'b' is already occupied (analogous to the Spatial Model which will add the relation to the coordinates until a free spot is reached): l(a, c, i+1) $\leftarrow$ l(a, b, i+1) $\wedge$ ln(c, b, i). As a result of applying this rule, the clause l(z, x, 2) is added to the logic program so that all direct relations are represented. Since the space to the right of z and the space left to x are free, the clause ln(z, x, 3) is added to the logic program in the next iteration, meaning the two objects z and x are not only in a left relation, but also direct neighbours of each other.
To sum up, both implementations follow a first-free-fit strategy and try to insert the object at the next free spot, except the one case in the Spatial Model mentioned above. But since this exception only shows up in the model-variation phase, which is not implemented in the WCS-approach of A. Dietz, the step of adding new objects to a model is semantically equivalent under the first two phases model construction and model inspection.

3.3 Combining models

When two already existing models with no overlapping objects should be combined in order to satisfy a new premise that refers one object in each of the sub-models, the spatial model appends one of the models to the other model, fusing the two separate models into one big model.
For instance, we've given two models, the first containing object x at (0, 0, 0) and y at (1, 0, 0), and the second model containing object y at (0, 0, 0) and z at (1, 0, 0). These two models are combined into one bigger model with object x at coordinates (0, 0, 0), y at (1, 0, 0) and z at (2, 0, 0). The second model is added to the first one at the position (1, 0, 0), since the overlapping object y is placed at this position in the first model. In order to combine these two models, the program needs to calculate a new origin for all the objects in the second model, giving them all new coordinates with a certain offset, which is (1, 0, 0) in this case.
The WCS-approach will add at least one new clause in order to represent the relation between the two items in the different models. So for instance, if the logic program contains the two clauses $ln(car, dog, 2) \leftarrow \top$ and $ln(cat, house, 2) \leftarrow \top$, the new premise (the third premise) $left(dog, cat)$ would lead to the insertion of the new clause $l(dog, cat, 3) \leftarrow \top$ as well as $ln(dog, cat, 3) \leftarrow \top$ in the following iteration. In case the new premise which triggers the combination process refers two elements that cannot be inserted directly next to each other, more clauses are inserted to link these two objects. If for instance the new premise to insert would be $left(dog, house)$, then the program would first insert the clause $l(dog, cat, 3) \leftarrow \top$ and in a later step $l(dog, house, 3) \leftarrow \top$.
The combination process is in general semantically equivalent, but the combination of two separate models can need many steps in the WCS-approach, so that often the program needs to terminate until the relation between all pairs of 2 objects in the initially separate models are clear. The Spatial Model on the other hand finishes the combination process in one step before the next premise is processed.

3.4 Verification

The Spatial Model verifies a given conclusion or premise in its initial model by checking whether the given relation holds between the two objects in the conclusion-premise with the help of the coordinates of the objects in the model. The implementation first gets the coordinates of the two objects in the conclusion-premise as well as the encoded relation of this premise. Afterwards it is checked for each of the three axes whether this relation holds, given the coordinates of the two objects. For instance, the relation "left" (subject is to the left of the object) is encoded as the tuple (-1, 0, 0). The verification will then check for the x-coordinate, whether the x-coordinate of the subject is smaller than the x-coordinate of the object. If that is the case, the x-axis was successfully verified. Moreover, the verification-function checks whether for the two other coordinates (all coordinates that are '0' in the encoded relation, which would be y and z in

our example), the coordinates are the same in the subject and object, meaning they have to be placed on the same axis in order to be successfully verified. If a verification turned out to be False, the program will print that the verification was not successful since the relation did not hold in one of the axes.
For instance, the model with the objects 'a' at (0, 0, 0), 'b' at (1, 0, 0) and 'c' at (2, 0, 0) is given; the premise to be checked is 'c is on the right of a'. The program will check if the x-coordinate of 'c' is bigger than the x-coordinate of 'a'. Since 2 is bigger than 0 and the y- and z-coordinate of the objects 'a' and 'c' are equal, the premise to check does hold in the given model. It is also important to note that the objects need to be placed on the same axis, meaning that the y- and z-coordinate of both items to be checked need to be the same. This means that the relation is interpreted direct, so if 'a' is to the left of 'b', but also in front of it at the same time, the relation technically does not hold according to the Spatial Model, even though it would hold in reality.

In the WCS-approach, after the model construction has finished, all clauses are transformed to left or right clauses of the form $left(object_1, object_2) \leftarrow \top$, $right(object_1, object_2) \leftarrow \top$ without a time index. This way all relations between all the objects in the model are set to true, if they hold. The conclusion can then be verified easily by just checking if the conclusion-premise transformed into a clause is set to True in the model.
For instance, after constructing the model out of the two premises 'x is left to y' and 'z is left to y' (with the conclusion-premise 'z is left to x'), the model consists of the clauses ln(x, y, 2), ln(z, y, 2) and ln(z, x, 2). (note: only the positive and necessary clauses are represented, the negative clauses as well as the positive clauses calculated in an earlier step like ln(x, y, 1) and the clauses concerning the ol / or - relation are left out for readability). The program will then change all clauses that encode a left direct neighbourhood first to general left and afterwards to right clauses. After that step, all left and right relations in the model are explicitly represented: $left(x, y) \leftarrow \top$, $left(z, y) \leftarrow \top$, $left(z, x) \leftarrow \top$, $right(y, x) \leftarrow \top$, $right(y, z) \leftarrow \top$, $right(x, z) \leftarrow \top$. Since the conclusion premise 'left(z, x)' is contained and set to True in the model, the given conclusion holds and the model was successfully verified.

To sum up, the WCS-approach only stores the relations between all items in the model, while the Spatial Model actually creates a representation of the model in a spatial and easily imaginable way. In the WCS-approach, the actual position of the items is only implicitly given through the clauses, meaning the positions are only relative to each other, whereas in the Spatial Model, all objects have fixed coordinates.
In general, the two models follow a quite similar strategy to construct their preferred mental model.

3.5 Example: WCS-approach and Spatial Model

This chapter will discuss a bigger and concrete example taken from A. Dietz (in [5], Chapter 6.5) to compare the Spatial Model with the WCS-approach.
The example Problem consists of the premises

- left(porsche, hummer)
- left(dodge, hummer)
- left(dodge, porsche) (conclusion premise)

The Spatial Model first parses the first premise and then creates a new model with the two objects, since initially the mental model is empty. As described in 3.1 the program creates a model and puts the hummer at the origin (0, 0, 0). The left-relation between the two objects porsche and hummer is encoded as (-1, 0, 0), so the porsche will be placed at position (-1, 0, 0), which is the position of the hummer with the added relation vector. Since this position is not occupied, there is no need to further search for a fitting position.
In the next step, the second premise is parsed and the dodge will be added as a new object to the model with the given left-relation (-1, 0, 0), since the hummer already exists in a model. The first possible position to put the dodge is (-1, 0, 0), but since this position is already occupied by the porsche, it is added to the model at the next possible position (-2, 0, 0) with the relation added a second time to the target coordinates.
The model construction is already done after these two steps and the conclusion needs to be verified. The program extracts the two objects dodge and porsche from the conclusion premise and compares their coordinates in the model. The relation left, which is represented by the tuple (-1, 0, 0) has to hold between the two objects in the model in order to be successfully verified. Therefore, the x-coordinate of the dodge needs to be smaller than the x-coordinate of the porsche and since this is the case with $-2 < 0$ (and the y- and z- coordinate of 'dodge' and 'porsche' is equal), the conclusion is successfully verified. (if the model variation phase is also taken into account, the Spatial model would try to falsify the conclusion in the next step, and return with the statement that the initial conclusion could have been false since the program was able to successfully falsify the conclusion. The falsifying model would be achieved by simply swapping the porsche and the dodge).
The WCS-aproach takes the two premises as an input and adds the time-index 1 and 2 to the two premises to process them in the given order. The names of the objects in the clauses will be abbreviated in the following, h represents the hummer, d the dodge and p the porsche. In the first iteration, the clauses l(p, h, 1)$\leftarrow \top$ and l(d, h, 2)$\leftarrow \top$ are added to the logic program. The first clause leads to the insertion of the new clause ln(p, h, 1)$\leftarrow \top$ in the following iteration since both the right space next to porsche and the left space next to hummer is initially free. As a result of that, ol(h, 2)$\leftarrow \top$ and or(p, 2)$\leftarrow \top$ are added to the logic program in the next iteration, because both spots are occupied now. Moreover, the clause ln(p, h, 2)$\leftarrow \top$ is added due to the fact that all relations

have to hold until the model construction is finished. The clause l(d, p, 2)$\leftarrow \top$ is added as well by using the rule as explained in chapter 3.2, as a result of the clauses l(d, h, 2) $\leftarrow \top$ (of iteration 1) and ln(p, h, 1) $\leftarrow \top$ (of iteration 2).
In the fourth iteration, the clause ln(d, p, 2) $\leftarrow \top$ is added since l(d, p, 2) $\leftarrow \top$ holds and both the space to the right of d and the space to the left of p are free. Additionally, the clause left(p, h)$\leftarrow \top$ is added because all premises have been processed and all ln-relations that have the highest possible time-index can be transformed into left-relations without a time-index. In the remaining three iterations, all of these ln-relations will lead to the insertion of a corresponding left-relation without a time index and the inverse right-relation in the following. Since the clause left(d, p), which encodes the conclusion-premise, is also part of the positive clauses, the conclusion holds in this model.

The following table illustrates what happens in each iteration of the WCS-approach. Like in the original example of A. Dietz, only new created clauses are shown in each iteration in order to sustain the readability. Moreover, the right column shows how the Spatial model processes the premises analogously to the WCS-approach.

Iteration	$I^{\perp}$	$I^{\top}$	Space-5 Model
1	l(p, h, 1), l(d, h, 2)	l(d, h, 1), l(d, p, 1), l(h, d, 1), l(h, p, 1), l(p, d, 1), l(p, h, 1) ol(d, 1), ol(h, 1), ol(p, 1), or(d, 1), or(h, 1), or(p, 1)	First premise left(porsche, hummer) is parsed with the resulting relation $(-1, 0, 0)$
2	ln(p, h, 1)	ln(d, h, 1), ln(d, p, 1), ln(h, d, 1), ln(h, p, 1), ln(p, d, 1)	Creation of the first model, the object 'porsche' is to the left of 'hummer': porsche hummer
3	ln(p, h, 2), ol(h, 2), or(p, 2), l(d, p, 2)	ol(d, 2), ol(p, 2), or(d, 2), or(h, 2), l(h, p, 2), l(p, d, 2), l(p, h, 2)	Second premise left(dodge, hummer) is parsed with the resulting relation $(-1, 0, 0)$
4	ln(d, p, 2), left(p, h)	ln(d, h, 2), ln(h, p, 2), ln(p, d, 2), l(h, d, 2)	The object 'dodge' is added to the left of porsche, which results in the model: dodge porsche hummer
5	**left(d, p)**, right(h, p)	ln(h, d, 2)	Verification phase
6	left (d, h), right (p, d)		Verification phase
7	right (h, d)		Verification phase

Table 1: Example with comparison of the WCS-approach and space-5. The first two columns are the positive and negative clauses in the logic program from the WCS-model.

4 Comparison of all three Implementations

All three models are implementations of the Mental Model or the Preferred Mental Model Theory. There are however some differences, since the focus was set to different aspects of the Mental Model Theory. The Temporal Model by P. Johnson-Laird is modeling the temporal domain of the Mental Model Theory. The difference to the general idea of the Mental Model Theory is however that all possible models are constructed from the beginning, instead of constructing one model first and then varying it until all models were successfully verified. Therefore the verifying-part of the Temporal Model works a bit different to just constructing the models one after another and verifying the premise after each new model: The temporal model will construct all models first, and then search for all models that support the relation before, while and after between the two events given in the question. In case all models support only one of these relations, the program can give a definite answer about the relation between the two events. If the models support two or more relation, there is no definite relation, since the deduction problem is a multiple-model problem with no valid answer. The focus on the Temporal Model implementation was to give an indicator about the difficulty of solving a certain deduction problem. If a deduction problem only leads to one model, it is a so-called one-model-problem and rather easy to solve for subjects. If a deduction problem leads to different models, it is harder to solve, especially when the different models also lead to different conclusions. To sum up, the Temporal Model was implemented to estimate the difficulty of a deduction problem for subjects and differs from the two other Implementations since it models the Mental Model Theory and the order of the premises is irrelevant.

In contrast, the two other implementations model the Preferred Model Theory, which means they will not construct all potential models and the constructed models are dependent on the order of the premises. The Spatial Model by P. Johnson-Laird should actually implement the Mental Model Theory, however, the order of the premises does matter since other models are constructed when the order is changed and therefore we came to the conclusion that this program actually fits the Preferred Model Theory. However, the WCS-approach and the Spatial model could construct all possible mental models like the Temporal model, when they would take all permutations of the premises in a problem as an input. Since the Temporal Model doesn't have a model variation phase, the WCS-approach would lead to the same result as the result of the Temporal Model for a problem when the premises of the WCS-approach are permutated in all possible ways.

As explained in 3, the Spatial Model implementation of P. Johnson-Laird and A. Dietz are quite similar regarding the construction process of the first and preferred mental model. The Weak-Completion-Semantics-approach however covers only one direction instead of all three possible axes like in the Spatial Model by Johnson-Laird. Moreover, the third phase of the Preferred Model Theory, which is the model variation phase, is not implemented in the Weak-Completion-Semantics-approach. Therefore, only the Spatial Model is a full implementation of the Preferred Model Theory for Spatial relations. In the model variation phase, the Spatial Model will try to falsify or verify the first solution concerning the conclusion-premise after constructing the initial model. If the first solution based on the initial model was that the conclusion holds, the model tries to falsify this conclusion by finding a model that fits the premises but doesn't support the conclusion anymore. The same happens the other way around when the initial conclusion was found to be false. The first found model that falsifies or verifies the initial true / false solution is returned. In case no such model was found, the initial solution was true and couldn't be disproven.

Besides the missing model variation of the WCS-approach compared to the Spatial Model, another difference is the representation of the deduction problem in the model. The Spatial Model will construct a visible geometrical figure with absolute positions of the elements (meaning they all have certain coordinates) that represents the problem in a way that is easy to imagine. Additionally, the relations are only implicitly defined between the objects. The only point where they are explicitly defined for a short time is when they are parsed from the premises in order to decide how to handle the new object(s). The WCS-approach, on the other hand, defines its mental model of the deduction problem only over relations, which means the positions of the elements are only defined relative to each other with no fixed coordinates. Moreover, the WCS-approach does not define the objects itself in the logic programs, they are only implicitly defined over the relations encoded as clauses. To sum up, the two implementations differ in that the Spatial Model explicitly defines the objects while implicitly defining the relations, while the WCS-approach does the exact opposite of that.

5 Cross-Comparison between the Temporal and Spatial Model

The two Mental Model Theory - implementations by P. Johnson-Laird have a lot in common, arising interest in the question to which grade the two models behave similarly for a given problem-type. We therefore adapted the implementations in a way that both models are able to solve each others problem-sets, meaning to run a Spatial deduction problem on the Temporal Model and vice versa. Afterwards, we compared the results for Spatial Deduction problems, solved one time with the Spatial Model and the other time with the adapted Temporal Model and analogously the results for Temporal Deduction problems once solved with the Temporal and once solved with the adapted Spatial Model. By using the temporal parser for temporal deduction problems and the spatial parser for spatial deduction problems, the two programs can be compared in how they differ with the same input-problem.
However, we had to make some exceptions in order to successfully run spatial and temporal deduction problems on both models, since the underlying concept especially for the encoded relations differs. The main difference is that in a temporal deduction problem, the relation "while" is handled differently to the other two relations "before" and "after". For "before" and "after"-relations, the order is strict and relevant in contrast to the order within events in a "while"-relation. Two or more events can happen at the same time, but the order between these events is irrelevant for temporal deduction problems. On the other hand in spatial deduction problems, the order is always relevant for all three axes. Since the "while"-relation in the Temporal Model is encoded as the tuple (0, 1, 0), which matches the coding for the "in-front-of"-relation in the Spatial Model, this leads to problems in the model construction-phase. Therefore, we handled some exceptional cases in a different way to the general method of constructing models, like checking whether a given relation (0, 1, 0) is a temporal or spatial relation and handling it accordingly in different ways.

We compared the four problem-types 'combination-problems', 'deduction- problems', 'indeterminate-problems' and 'inconsistent-premises-problems', which are present in both implementations by P. Johnson-Laird with each other, each of the problems solved by both implementations.

5.1 Combination problems

We found that for combination-problems, the Spatial and Temporal models behaved alike which is not a big surprise since most of the functions required for the combination of two sub-models are identical. All combination-problems (both Spatial and Temporal problems), computed once with the Spatial and once with the Temporal Model leaded always to exactly one and the same model. The Temporal Model should always compute all possible mental models but doesn't however in terms of combination problems, since the combine-method doesn't handle indeterminacies and just follows one possible way of combining the sub-models.

5.2 Deduction problems

The spatial deduction problems were all handled the same way in both the Temporal and the Spatial Model. Deduction problem no. 5 required the combination of two sub-models and since the combination doesn't include handling indeterminacies in the Temporal Model, the only resulting model was similar to the resulting model of the Spatial Model. The only other difference was that in problem no. 1, the Temporal Model found three fitting models for the given deduction problem due to the indeterminacies, in all other problems, both implementations only had one model as a result and all returned that the last premise follows validly from the previous ones.
The temporal deduction problems were handled similarly as well by both models, there were however more differences compared to the spatial deduction problems. The main opposing result we found was that for problem no. 5, the Temporal Model returned the answer 'the last premise follows validly from the previous ones', the Spatial Model however returned that 'the last premise is inconsistent with the previous ones'. Taking a closer look at the resulting models, the reason for that difference can be found in the different handling of the relation (0, 1, 0), which encodes "while" in temporal deduction problems. In order to correctly add a new item, which is the object of a premise, to an already existing model where the subject of the new premise already is a part of, the relation needs to be negated, meaning to insert the new item in the relation (0, -1, 0) instead of (0, 1, 0). In the Temporal Model this is however not done in case of a while-relation, since the order between items in a while-relation does not matter, so the new item is always inserted with the non-negated relation (0, 1, 0). On the other hand, the Spatial Model will always negate the relation, meaning depending on whether the item to insert is the object of the new premise, it will be inserted at the 'wrong' place.
The second major difference of the two implementations leading to opposing results for problem no. 5, is the verification-function which is much more strict in the Spatial Model. For two items to correctly support a given relation, one of the items needs to have a bigger coordinate than the other one, for instance in the relation "A is to the right of B", encoded as (1, 0, 0), the x-coordinate of A has to be bigger than the x-coordinate of B. Moreover, all other coordi-

nates that are 0 in the encoded relation, have to be equivalent in the two items, meaning they always have to be placed at the same axis in order to be successfully verified. In contrast, this second restriction doesn't need to hold for the success-full verification in models of the Temporal implementation, therefore the two items never have to be placed on the same axis. This difference makes the verification-method of the Temporal Model a bit more 'natural', since f.i. a 'in-front-of' relation between two items should also hold when one item is placed at the diagonal-front of the other item.
To sum up, the deduction problems are handled very similar in the two implementations and lead mostly to the same results, the few found differences are based on technicalities like the stricter verification-function of the Spatial Model and the different handling of the '(0, 1, 0)' relation in both Models.

5.3 Indeterminate problems

For the spatial indeterminate problems, the Spatial Model returned 'Premise was previously possibly true' for the first, and 'Premise was previously possibly false' for the problems 2-9. For the first problem that means, the Spatial Model found that the given last premise of the problem was True in the first constructed, preferred model, it was however possible to find another model in the model-variation phase which was consistent with all premises but didn't satisfy the last premise. Analogous, the first constructed and preferred model didn't satisfy the last premise in the indeterminate problems 2-9, the Spatial Model could however find another correct model in all of these problems that would satisfy the last premise. These results are achieved through the model variation phase of the Spatial Model, which will try to negate a conclusion-premise that was True in the initial constructed model (by calling the function 'make-false'), or the other way around when the initial model didn't satisfy the conclusion-premise (by calling the function 'make-true'). On the other hand, the Temporal Model always returned 'The premise was hitherto possibly false' for all spatial indeterminate problems, which basically only states the fact that it could find both models that did support the conclusion premise and models that did not support the conclusion premise. From that we can conclude that both implementations achieved the same results by finding both models that support the conclusion and models that didn't, even though the output-print differs a bit.
For the most part, the temporal indeterminate problems as well leaded to the expected results. The problems no. 2-6 leaded to the conclusion 'hitherto possibly false' or respectively 'previously possibly false' and for problem no. 7 both models answered that the last premise follows validly from the previous ones. Conclusively, the two models worked the same way in problems 2-7.
The interesting problem that leaded to a different result was problem no. 1. The Temporal Model answered that 'the premise was hitherto possibly false', meaning some of the models did satisfy the conclusion while others didn't. The Spatial Model's answer was however, that the last premise follows validly from the previous ones, meaning that the implementation couldn't find a model that

would satisfy the premises while falsifying the conclusion-premise. This result supports our assumption that the Spatial model doesn't exhaustively test all kind of model variations and conclusively will sometimes give a wrong answer. A similar case occurs in the last problem no. 8, where the Temporal Model returns "the premise was hitherto possibly false" while the Spatial Model returns 'The premises is inconsistent with the previous ones.' Here again the Spatial Model wasn't able to find a model that supports the conclusion-premise trough the model variation-phase, even though it existed, since the Temporal Model could find both supporting and neglecting models regarding the conclusion-premise.

5.4 Inconsistent premises problems

The problems containing a last premise that is inconsistent to all previous ones, leaded to the expected same results by both implementations. Both for temporal and spatial problems and in both Models, the answer was always that the last premise is inconsistent with the previous ones.
However, regarding the results of 5.3, it is conceivable that the Spatial Model might sometimes give that answer as a result of its non-exhaustive search that couldn't find a fitting model even though one existed and not because it's the right answer.

In conclusion, most of the results of all four problem-types indeed matched the expected results when solving Spatial deduction problems with the Temporal Model and vice versa. The found differences are mostly based on some technicalities, like the different verification-function, the handling of the relation '(0, 1, 0)' and the non-exhaustive search of the Spatial Model.

6 Implementation of the WCS-approach in Python

The implementation of the WCS-model by A. Dietz was programmed in Prolog. In spite of the fact that Prolog is a programming language especially appropriate for logic programs, it has its limitations in the presented approach. Since the additional time-index leads to many rules that have to be written down in order to successfully calculate a given deduction problem in the presented order, the amount of rules grows exponentially with each premise, as every premise in a problem needs its unique time-index. We therefore propose to implement the Weak-Completion-Semantics approach in Python in order to circumvent the limitations. This has the additional advantage that it could easily be applied to temporal deduction problems in addition to spatial deduction problems and is better comparable to the other two Mental Model Theory - implementations.

To implement the WCS-approach in Python, a logic programming framework is needed. However, another approach would be to implement the rules more implicitly in form of functions. These functions could check whether the rule of the function is applicable, and then return either None or the resulting clause of the rule. Depending on the type of rule, all rules with the same logic structure could be put together in a list of rules for this type, which would also prevent redundancies. The clauses of the current logic program could be stored in two lists to distinguish the clauses that hold in the logic program from the ones that do not hold.
The program needs to iterate over all the rules in an update-function that also needs to decide if any rule could be applied at all. This could be realized by simply creating a boolean variable that is set to True, if any rule could be applied in the current iteration, likewise to the functionality of the parser in the function 'parse' in the Spatial and Temporal Model by P. Johnson-Laird. If no more rule is applicable with the current processed premise, the next premise is processed and the program checks whether there are now any rules applicable with the new premise. If all premises were processed, no more rule is applicable and there were no failures or mistakes, the model was completely constructed and all relations between all the objects in the model are contained in either the list of positive or negative clauses.
For the verification-process in the model inspection phase, the program simply needs to search for a clause with the two given objects from the premise that matches the given relation. If such a clause can be found and is in the list of positive clauses, the conclusion-premise holds in the model, if not, the premise does not hold in the model.
For the model variation phase which is not present in the WCS-approach in Prolog, the program could, likewise to the Spatial Model, swap two objects and move an object to another place in order to modify the model. For swapping two object from a given premise, it is sufficient to replace all x by y and all y by x to successfully swap the objects x and y. If an object should be moved to another place, all the clauses where this object occurs need to be deleted from the list of clauses. A new position needs to be found to insert the object. After the object is inserted at a new and fitting position, the general program loop can run again until all possible clauses were added while applying the rules (no more changes in the two lists) and the model can be verified again.
The input of the program can be realized as a set of premises in the form of the relation clauses, or otherwise a parser is needed like in the Spatial and Temporal Model of P. Johnson-Laird. Moreover, the time-index which was needed in the WCS-approach in order to preserve a certain order in which the premises are processed, would not be necessary in the Python implementation as the order is implicitly given through the input list of premises and since the main update-loop will only process the next premise when no more rule is applicable with the current model.

7 Future Work

For ideas to work on in the future, we first suggest to implement the WCS-approach by A. Dietz in Python as explained in chapter 6. After transferring the WCS-approach to Python, it would be interesting to draw a direct comparison between all three models, since they are all implemented in the same programming-language by then. Especially the model-variation phase would be interesting to compare with the corresponding steps of the Spatial Model, since this was not tested in the original WCS-approach by A. Dietz.
Furthermore, all other domains (like syllogisms and conditional reasoning) of the Mental Model Theory put together in one framework implemented in Python would be of great interest. This would lead to an implementation of the whole Mental Model Theory with directly comparable domains, enabling to test whether these kind of deduction problems are all solvable in a very similar way. This could cast light on whether the Mental Model Theory is actually a profound theory for all sorts of deduction problems compared to the opposing theory of Inferential rules.
Another interesting aspect to focus research on is to combine the spatial and temporal deduction problems into four-dimensional time-space-deduction problems. These are probably much harder to solve and there is not much research on this topic yet. Additionally, to calculate combined four-dimensional deduction problems and make them more realistic, the static time events of the Temporal Model by P. Johnson-Laird could be extended to time-spans to enable overlapping time-events. The same idea holds for the Spatial Model, it would be interesting to have spatial objects of different sizes.
To sum up, even though the Mental Model Theory is already quite old and many papers have been published and given further insights on it, there are still many interesting aspects and unresolved issues to focus future research on.

References

[1] Computational Models. http://mentalmodels.princeton.edu/models/. Accessed: 2018-7-26.

[2] P.N. Johnson-Laird. Mental models: towards a cognitive science of language, inference and consciousness. 1983.

[3] M. Knauff M. Ragni. A theory and a computational model of spatial reasoning with preferred mental models. 2013.

[4] Ruth M. J. Byrne P.N. Johnson-Laird. Spatial reasoning. 1989.

[5] Emmanuelle-Anna Dietz Saldanha. From logic programming to human reasoning: How to be artificially human. 2017.