NTNU
Norwegian University of Science and Technology

Knowledge Discovery in Climate Science using Jess rule Engine

Knut Harald Ryager

Master of Science in Informatics	
Submission date:	June 2016
Supervisor:	Pinar Öztürk, IDI
Co-supervisor:	Erwin Marsi, IDI

Norwegian University of Science and Technology
Department of Computer and Information Science

Knowledge Discovery in Climate Science using Jess rule engine

Knut Harald Ryager

June 2016

Norwegian University of Science and Technology

Department of Computer and Information Science

Supervisor: Pinar Öztürk, IDI

Co-supervisor: Erwin Marsi, IDI

Abstract

Climate change is a difficult research problem, requiring insight from vast fields such as chemistry, biology, climatology and oceanography. As the ever increasing publishing of information in these fields continue, text mining and knowledge discovery systems are being developed to alleviate the workload of extracting relevant information from literature. This master thesis focuses on knowledge discovery from extracted information, and ranking of proposed hypotheses. A rule-based inference system was implemented i Java using the Jess rule engine. The application consist of a set of general and domain specific logic rules, which operate on domain knowledge inputted to the system, and proposes hypotheses based on this input. The system can also abductively attempt to explain contradictions between expected and inputted knowledge. The system was tested on the domain hypotheses known as the iron hypothesis and the DOC hypothesis. The system was able to infer these hypotheses.

Trondheim, 2016-06-01

Knut Harald Ryager

Acknowledgment

Thank to supervisors Pinar Öztürk and Erwin Marsi, who has offered their guidance for this thesis. The thesis is written in contribution to the EU Framework 7 OCEAN-CERTAIN project, where the work of Pinar, Erwin, Elias Aamot and domain experts Murat V. Ardelan and Frede Thingstad has been important to provide use cases for the system developed in the thesis. Also thanks to Rajendra Prasath who also work under OCEAN CERTAIN, and performed some experiments with me, although these experiments didn't fit into the thesis.

Contents

Chapter 1

Introduction

1.1 Motivation

With an ever growing body of scientific literature, it is harder to follow along with all that is published. The result is that scientists become more specialized, and form new subfields. When scientists don't follow up on the research closely related to their specialized field, the field becomes isolated, and there is a growing concern that this isolation can lead to knowledge not being utilized by a wider scientific community. Literature-based discovery systems (LBKD or LBD) works to alleviate this, using text mining and AI methods to connect knowledge from across literature works.

One of the pioneers in pointing out this issue is the librarian Swanson (1986), who produced the first results in LBD, which aims to automatically connect knowledge from isolated fields to form a discovery. The issue is not new however. A good historical example of this happening is when *Darwin* wanted to explain how variability within species arose for his theory of evolution. *Mendel* on the other hand, had already discovered how variability was inherited, but was not aware of Darwin's evolution theory to see this wider implications of genetic variably. It turns out Darwin actually owned a book describing Mendel's experiments, yet he had not gotten as far as reading about them. (Example from Gordon et al., 2001). Much of the research in LBD has been done in the domain of biomedicine, as this is one of the biggest domains where specializations and publishing are rampant. In MEDLINE, The U.S National Library of Medicine, 2,000-4,000 references to journal articles are added each day. The issue has been growing also in the domain of climate science, with heavy specializations to understand different nature systems, yet a need to understand the complex interactions between the systems.

The OCEAN-CERTAIN[1] project, funded by the EU and coordinated by NTNU, involves 11 partners from Norway, Germany, Belgium, Turkey, Sweden, Australia, Chile and the UK. It focuses on marine science, and is the first project to begin literature based-discovery in climate science. The goal is to make a discovery support system, being able to link knowledge from different sources

1.2 Research goals

The focus of this thesis is on the reasoning component of a literature-based discovery system intended for OCEAN-CERTAIN. The system developed is not a final system, but an exploration of inference systems that may guide design decisions. The research goals addressed by the thesis are:

1. Make a model of the knowledge and inferences to
   - a) generate hypotheses of new knowledge.
   - b) Detect contradictions in the system's knowledge
   - c) Explain the contradictions through a reasoning process
2. Develop an inference system to implement the reasoning model.
3. Suggest how to visualize the reasoning of the system in ways that can be analyzed by both AI people and domain experts.

1.3 Limitations

As the text extraction system of the OCEAN-CERTAIN project is not finished, only the previously manually annotated corpus, along with any other manually entered data, is available to use by the system. This limits the ability to perform large-scale statistical experiments on big data sets.

[1]http://oceancertain.eu/what-is-ocean-certain/

1.4 Approach

The approach taken was to first construct a prototype version of the program, and discuss the resulting hypotheses generated by the prototype with a domain expert. This is to identify the most important features and limitations of the system. By going through use cases, the system was iteratively improved by adding more desired features.

1.5 Structure of the Report

Figure 1.1 shows the workflow of the thesis. First, a domain knowledge model was made, fulfilling research goal 1. Then the model was implemented in the Jess rule engine, fulfilling goal 2. Then the system was made able to print out the results from running the inference, both as a Neo4j relational database, and a HTML/JavaScript page, fulfilling research goal 3. Then, some use cases were tested with the system.

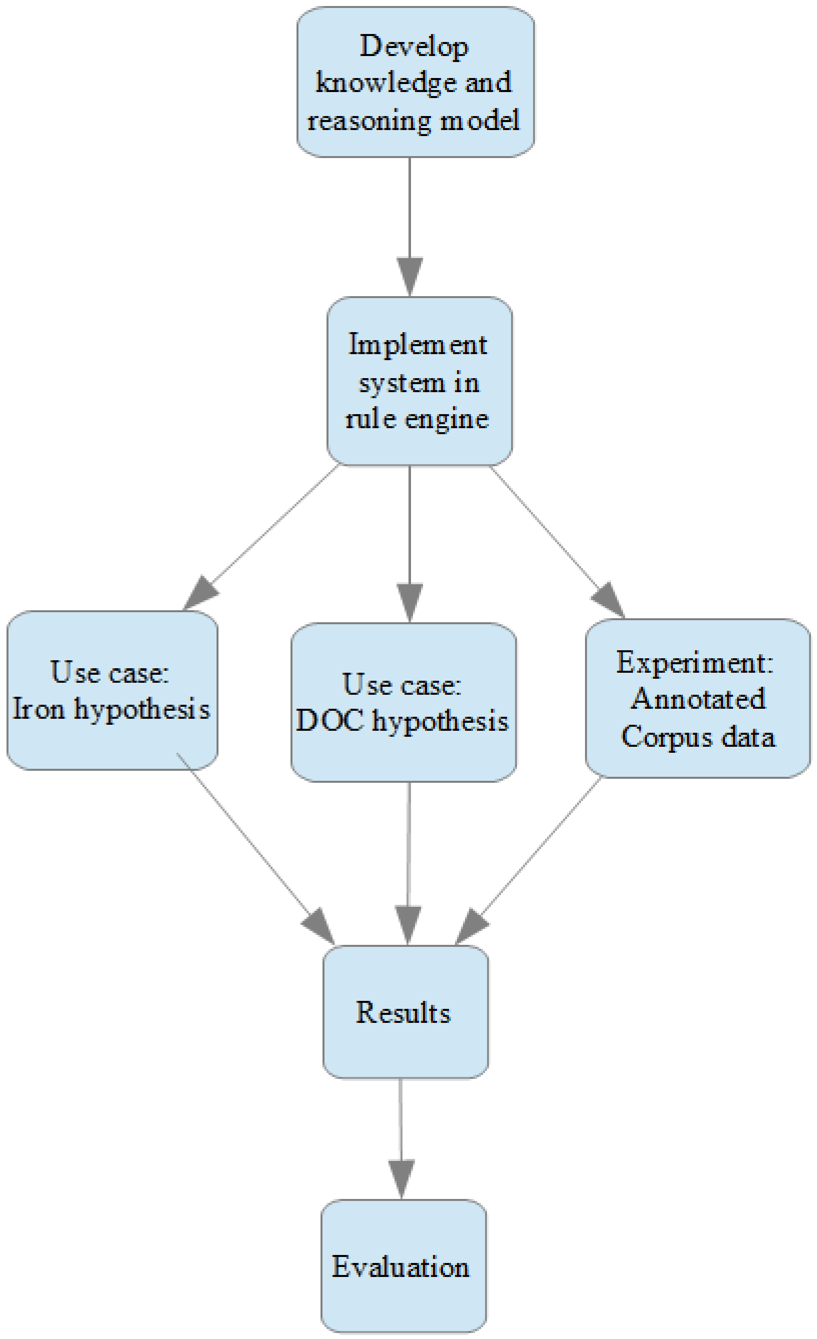

Develop knowledge and reasoning model
Implement system in rule engine
Use case: Iron hypothesis
Use case: DOC hypothesis
Experiment: Annotated Corpus data
Results
Evaluation

Figure 1.1: Flow diagram of the work presented in the thesis.

Chapter 2

Background

This chapter reviews the structure of literature-based discovery systems, the OCEAN-CERTAIN project, and what work has been already done with regards to LBD-systems in OCEAN-CERTAIN. Then, various AI methods for inferring knowledge is looked at, and compared to the more broader inference methods used in science. From this, an AI method is chosen for this thesis' LBD system, to address research goal 1 and 2. (See chapter 4).

2.1 Literature-based discovery (LBD)

LBD aims to discover publicly available knowledge in literature, beyond what is already possible with keyword searches. A knowledge support system can use text mining to extract structured data of objects and relationships between them. The system can then suggest new hypothesises based on it's findings, possibly combined with existing domain knowledge.

Such a hypothesis was first published by Swanson (1986), who used a search algorithm based on statistical co-occurrence, which he manually ran through some dozens papers related to Raynaud's syndrome and/or fish oil. In this literature, it is established that fish oil (A) is able to reduce vascular reactivity (B), while a reduction in vascular reactivity is able to ameliorate Raynaud's syndrome (C). This knowledge, $A \rightarrow B$ and $B \rightarrow C$, can be linked as $A \rightarrow C$. Swanson showed how this connection was plausible from his algorithm, and such linking in LBD is now referred to as Swanson's ABC model. DiGiacomo et al. (1989) later confirmed Swanson's findings through clinical trials.

LBD can be done in either an *open* or *closed* form. In open discovery, a single initial concept is explored, leading to some new concept. In closed discovery, there are two initial concepts, and the goal is to find a connection between the two concepts. Figure 2.1 illustrates this with a graph. This thesis focuses mainly on open discovery.

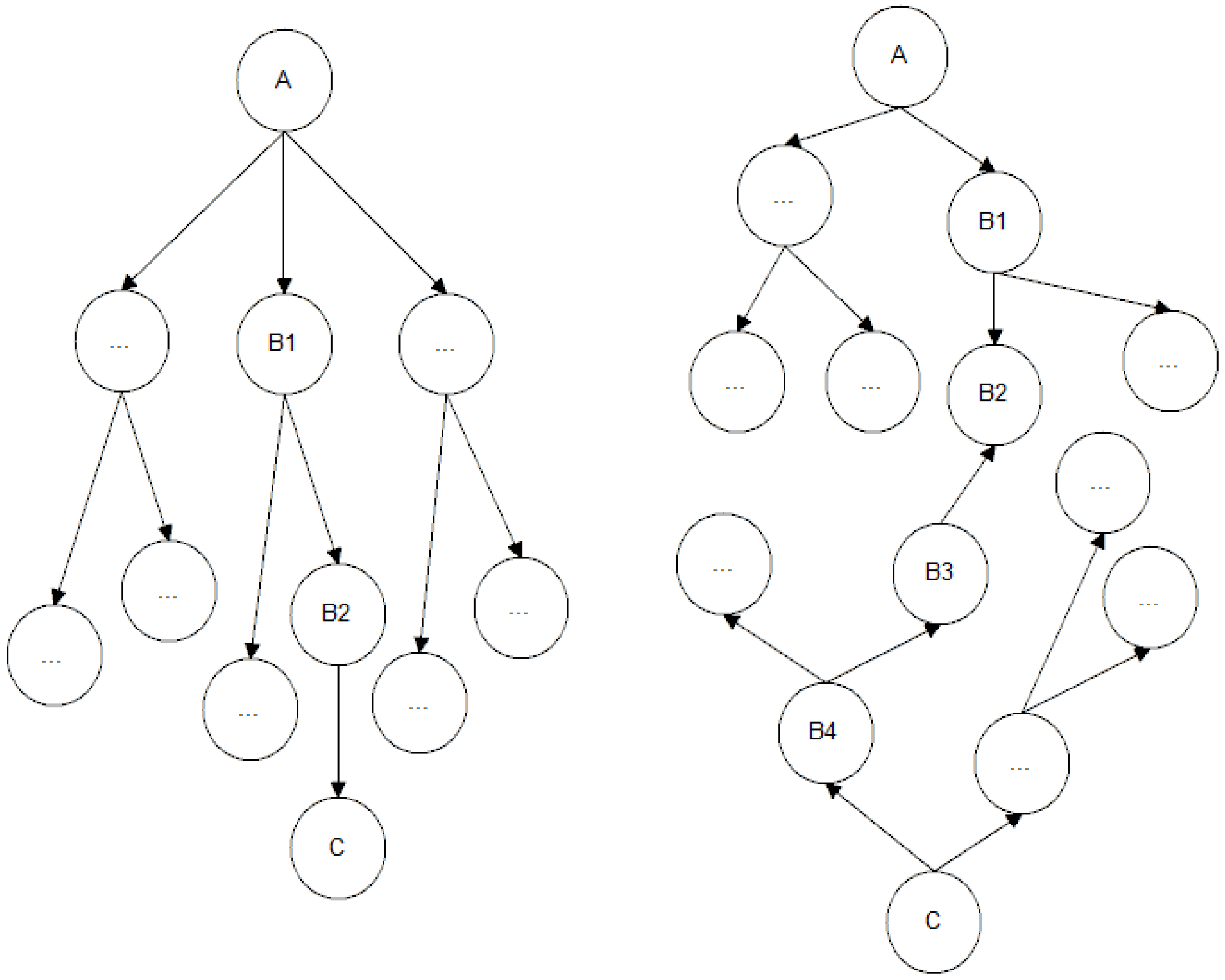


Figure 2.1: Open discovery (to the left) and closed discovery (to the right). Each node is a concept, raised from node pointing to it.

A typical LBD system has a pipeline consisting of several steps:

Document retrieval: First, documents (scientific articles) are crawled from various scientific publishers. These publishers have varying document formats and interaction APIs, so the they are converted to a standard format and indexed. Relevant documents, in oceanography and related fields, can then be filtered uniformly by some metrics.

Information Extraction: Text documents are converted into structured information. Sentences are parsed, words are standardized by their semantic meanings, and stored in data structures as used by programming languages and data bases.

Information extraction is typically divided into three subtasks: In (Named) Entity recognition (ER), each entity is classified as a category, e.g. “Tom works at Google” could be annotated: [Tom]person works at [Google]organization. "Named" means that the entity is linked to some canonical proper name, e.g. if the name "Obama" appears in the text, he may be linked to the canonical “Barack Hussein Obama II”. Typically the hardest part of ER is to link referring expressions, such as “him”, which could also refer to "Barack Hussein Obama II", or some other entity, based on it’s context. Relation Extraction (RE) aims to discover relations existing between entities in the text. It is common to limit the extracted relations to a predefined set of binary relations for the task domain, and extract these as (subject,predicate,object) triples. The above example could yield (Tom,WorksFor,Google), with WorksFor being the canonical binary relation for “works at”. Event Extraction (EE) involves the extraction of structures more complex than relation triples. It could be n-ary relations, relations having other relations as arguments, or the elements in a triple could have modifiers describing some condition, like time or space, where it holds true. ”Tom works at Google in the weekends” could have a modifier AtWeekends either surrounding WorksFor, or the whole triple. The development of IE have gone from focusing mostly on ER and RE, to EE receiving more attention in IE-focused domains like biomedicine, where ER and RE tools have reached some reasonable performance.

Inference engine: Reasoning about structured information can be standardized, which allows a computer to make various inferences, both deductive and abductive, that a human is able to make from the information. These include causal transitivity, generalizations and specializations. This thesis focuses mainly on this step of the pipeline.

User interaction: Because the system does not have the understanding of a domain expert, it is useful for the system to be able to take input to adjust it’s reasoning, such as focusing on what the user deems more important when investigating reasoning paths. Designing this interaction is a major job, and as this thesis does not cover the final implementation of the system, it will only serve in guiding to design this system.

2.2 OCEAN-CERTAIN

OCEAN-CERTAIN is a cross-disciplinary research project aiming to discover the impact of various stressors on the oceanic *food web* and the *biological pump*, starting in 2013 and expected to run for 4 years. The development of knowledge discovery support for the oceanographic climate science domain is a part of this project.

Food webs are the interconnection of food chains of species in an ecological community, describing what specifies feed on each other in a graph. Specifies can be divided into two trophic levels, autotrophs and heterotrophs. Autotrophs consume inorganic material like minerals and gasses (such as CO2) to produce organic material, which can be consumed by heterotrophs. A food chain goes from fully autotroph at the bottom, to more heterotroph further up. Thus autotrophs lays the foundation for all biomass in the food web, and are as such referred to as primary producers. The most common primary producer in the ocean are phytoplankton, which comes in several types. Some are calcifying, meaning that they form a calcium carbonate exoskeleton. Whether calcifying or non-calcifying phytoplankton dominate an environment, is dependent on various factors, one being the available mineral nutrients. Phytoplankton lives near the surface of the ocean, as they require CO2 and sunlight. On death, their biomass is mostly released back into the atmosphere. The shell of calcifying phytoplankton however may drag some biomass to the bottom of the sea, removing it from the environment. Specifies feeding on phytoplankton may also move the biomass deeper into the sea, and cause it to eventually reach the bottom. This effect is known as the *biological pump*, and is a significant mitigator of increasing CO2 in the atmosphere from man-made activities.

There are however various stressors on the food-web, such as ocean pollution, acidification, and overfishing, that while directly may only affect one species, may indirectly affect the entire food web. Thus they may effect the effectiveness of the biological pump, which is dependant on the food web composition. While stressors' effects are often well studied in isolation, the overall effect of all stressors is hard to predict.

Marsi and Öztürk (2015) observes that a common occurrence with variables in the Earth science domain is that they represent complex entities expressed as noun phrases with up to multiple modifiers, e.g. oxygen depletion in the upper 500 m of the ocean, rather than simple atomic entities often found in biomedicine. Extracting and reasoning about such complex variables poses a greater challenge. To more easily reason about the variables, they perform generalizations, e.g. by pruning away modifiers, such as oxygen depletion in the upper 500 m of the ocean to oxygen depletion. This pruning is done by performing tree transformation operations on the sentence's syntax tree. They build a browsable variable hierarchy, allowing the user to inspect all mentions of each variable type, both the original mention in the text and it's generalizations. They used text from journals published by Nature Publishing Group, selecting the top 10k abstracts based on matching with search terms given to them by domain experts. After tokenization, sentence splitting, POS-tagging lemmatization and parsing, they were left with 9,586 article abstracts, 59,787 sentences and approximately 4 million tokens. On the lemmatized syntax trees, they were able to use the Tregex engine (Levy and Andrew, 2006), to match the trees with regular expressions for changing variables. These patterns were generated from a small hand-written patterns, yielding 320 patterns. The total number of matched variables in the corpus is 21,817. Among these, 66% were unique, supporting the need to generalize them into something more common. The tree transformations for generalizing the Tregex expression was implemented in Tsurgeon. With different variables being generated based on what transformations were applied, the resulting number of variable instances after the generalizations was 150,716. The number of variable types, after removing those that appeared only once, was 17,613.

2.3 LBD focus in the OCEAN-CERTAIN project

Based on discussion with a domain expert, Marsi et al. (2014) decided that the focus of the LBD research of Ocean Certain will be to extract causal relations between events of variables that are changing - either increasing, decreasing, or an unspecified change. The extracted rules will be on the form "If variable X increases, then variable Y decreases". The system will also look for correlations, without an identified causal direction, and contradictory findings. The system will then be able to identify chains of change events. A discovery of particular interest in the relevant domains are feedback loops, where a causal chain forms a circle.

They propose an annotation scheme to capture change events and causal relations. Each change event must have a trigger, and the change must apply to a variable. The trigger is tagged either INCREASE, DECREASE or CHANGE, for undirected changes. The variable of an event is tagged as THEME. Some examples are

a. [DECREASE reduced] [THEME calcite production] b. [CHANGE significant changes in] [THEME surface ocean pH]

A cause event has a trigger for the cause relation, tagged CAUSE. They must have a AGENT representing the cause, and a THEME representing the effect. Some examples are

a. [AGENT rise in atmospheric CO2 levels] [CAUSE causes] [THEME significant changes in surface ocean pH] b. [AGENT Fe(III) addition in the presence of GA (FeGA)] [CAUSE gave] [THEME higher Fe(II) concentration]

The extracted if-then rules can be written with an arrow syntax, with $\uparrow$ signifying an increase, $\downarrow$ a decrease, and $\updownarrow$ an unspecified change, $\implies$ for causal relations, curly arrow $\rightsquigarrow$ for correlations, $\iff$ for feedbacks, optionally with + or - for positive or negative feedbacks. The logical predicates $\wedge$ (and), $\vee$ (or), $\neg$ (not) are also supported, and delimited my square brackets. An example is show below, where sentence a is extracted to rule b.

In the light-limited regime, the carbon:nutrient ratio turns out to decrease with increasing mixed-layer depth and temperature.

[$\uparrow$ mixed-layer depth $\wedge$ $\uparrow$ temperature] $\rightsquigarrow$ $\downarrow$ the carbon:nutrient ratio

2.4 Discussion of LBD prototype with domain expert

One challenge of developing a knowledge discovery system, as with other software systems, is to align the expectations from the users of what the system will be capable of, with the technical possibilities and limitations. One issue is that it is often hard for the user to fully express what they want from the system until they see some working prototype they can comment on. Thus after making a basic working system, a discussion session was done with a domain expert (Murat Ardelan), where the prototype was shown off.

First, the input to the rule engine was looked at. This consists mostly of manually annotated causal relations and syntactical generalizations. As it turns out, some of the input relations were contradictory, as they came from conflicting studies. The domain expert commented that such contradictions would be very important for the user to find.

In regards to the syntactical generalizations, the domain expert commented that when a *process* is involved in a complex variable, such as fixation, recycling, enrichment, uptake, removal, generalizing these to a the variable involved in the process could yield bad reasoning. He suggested that these processes should be recognized as processes within an ontology, which would allow for some more accurate reasoning in regards to them. For example, "uptake" is often correlated with "removal", which could mean that a variable is decreasing. Another issue with syntactical generalizations are such as "growth rate of phytoplankton" being generalized to "phytoplankton", as a change in growth rate of a variable does not necessarily imply the same change in that variable, based on consumption of that variable. This is among those generalizations that have been shown useful however, as one of the rules from an article, "iron increasing causes growth rate of phytoplankton increasing" are among those that helps to yield the iron hypothesis. So whether ultimately whether this generalization will prove too problematic or mostly useful, will have to be judged on a larger data set.

In regards to the inferred rules, the domain expert commented that some variables were too general or specific to inherit relationships, e.g. the variable "plant growth in the sea" which shown up in a rule, is too general. Phytoplankton is an example of ocean plants that are very important however, and it may be useful to generalize a relationship from one species of phytoplankton into phytoplankton. Which generalizations that are more plausible or useful ,is heavily on the domain side of knowledge, and may be part of the ontology used by the application.

2.5 Inference methods

Here is presented some of the basic reasoning forms most used, both in science and daily life, before we look at reasoning forms in AI.

2.5.1 Deduction

Deductive reasoning begins with a set of premises, and infer new conclusions based on some logical rules. The conclusions holds true as long as the premises are true, and deductions are thus *valid*. One of the simplest and well-known deductive rules is *modus ponens*, which uses the rule "a implies b", written $a \rightarrow b$. With the premise that a is true, written simply a, the deduction goes

$$\begin{array}{l} \quad a \\ \quad \underline{a \rightarrow b} \\ \therefore b \end{array}$$

With the premises above the horizontal line, and the conclusions below it.

2.5.2 Induction

Unlike deductions, *inductions* may infer conclusions without a definite logical rule implying the conclusion based on the premises. They are said to be top-down, unlike deductions which are bottom-up. Most commonly, inductions are generalizations, justified by statistical plausibility. Induction must not be confused with mathematical induction, where a generalization is proved to hold for every possible scenario, which actually constitutes a deduction.

2.5.3 Abduction

Abduction is the process of finding the most likely explanation for an observation. It is thus said to be reasoning "backwards" rather than "forwards". Falkenhainer (1990) uses the following notion of abduction:

$$\begin{array}{l} \quad \mathscr{D} \text{ is a collection of data} \\ \quad \mathscr{H} \text{ explains } \mathscr{D} \\ \quad \underline{\text{No other hypothesis explains } \mathscr{D} \text{ as well as } \mathscr{H}} \\ \therefore \text{Therefore, } \mathscr{H} \text{ is correct} \end{array}$$

Like with induction, the conclusion is not guaranteed. First, the reasoning used to construct $\mathscr{H}$ may include non-valid reasoning, such that $\mathscr{H}$ may not be valid. Second, there is no guarantee that all possible hypotheses are actually considered. Third, the most likely hypothesis may differ based on the measure of how probable hypotheses are.

2.6 AI inference methods

2.6.1 State space search

In AI, problems are often considered as a *state space*; a set of states and operations that can be applied to states to transform them to different states in the state space. One of the states is an initial states, and at least one is a goal state. Applying the operations to the initial state and the following generated states in an attempt to reach a goal state, constitutes a state space search.

A state space constitute a graph, with states being the nodes and operations the edges. Thus, any graph searching algorithm, such as breadth-first or depth-first, can be used for a state space search. However, AI methods usually bias the search order in such a way that fewer states will have to be considered than in an exhaustive search. Rules that disregard certain states from the search are called *pruning* rules. Pruning is often done by ranking each state by how promising they are, such that the most promising branches can be searched first. Such ranking rules are called *heuristics*.

When a state space has more than one goal state, one may not just want to find any goal, but the goal with the shortest path to the initial state. An algorithm that can do this is the *A** search algorithm. It uses heuristics which ranks states based on an estimated path length from the initial state, through that state, and to a goal state. If the estimate is guaranteed to never overestimate the path length, the heuristic is said to be *admissible*. With an admissible heuristic, A* is guaranteed to find the shortest path to a goal state. The shortest path problem in graph can be mapped to typical shortest paths problems found in real life, such as that of finding the shortest path between two points on a geographical map. An admissible heuristic in this scenario is to take the direct flight path to the goal destination, as there are no possible shorter paths.

In other problems, finding a goal state may be the only real concern. In an *equation solving* problem, the unsolved equation is the initial state, algebraic operations are the operations to be applied on states, and an equation with the unknown variable isolated on one side, is the goal state. The ranking of states in this problem is more concerned about the remaining path length of a state to the goal, rather than the path length from the initial state.

In a *chess game*, the goal states are those positions where a player has won, with a state search attempting to reach such a state for one of the players from the current board state. Here the opponent decide where the search will go for every other move, and the search prunes away the branches from those moves where the opponent is considered not to make good plays.

Often, there are no efficient admissible heuristics known to solve a problem. Often however inadmissible heuristics may perform well in a range of problems. Here we look a some common AI heuristic types.

Means-End Analysis

Means-End Analysis takes into account both an initial state and a goal state while reasoning. It works by considering the differences in the initial and goal state, and attempts actions that will reduce the number of differences. In an equation solving system, with the logical operations *add*, *subtract*, *multiply*, *divide* and *factor out*, a difference between the unsolved and solved equation can be that the unknown variable x have several occurrences, as opposed to one. If two terms $ax + bx$ in the equation are being considered, Means-End Analysis thus suggests the factoring operation to derive $x(a + b)$.

Means-End analysis thus provide similarity measures as search heuristics. It can be employed in forward chaining, for a more depth-first search. The heuristics may however not be admissible, such that the required number of moves to the goal state increases rather than decreases by applying a move suggested by the heuristic. In the above example, if the full equation was $ax + bx = ax + c$, the factoring, while getting closer to the solution by the metric of x-occurrences, would only pull further away from the goal state, as opposed to subtracting ax, which lead to the goal state in just one additional operation.

Closed LBD, where both search terms are searched for a connection to the search graph springing from the other term, is an example of means-end analysis.

Hill climbing

Like Means-End Analysis, Hill climbing attempts to reach a goal state by changing a state into a more similar one to the goal state. For a continuous function $f(x)$, where the max value of f is the goal state, hill climbing on the x-axis is equivalent to travelling in the positive direction of $f'(x)$, or 'climbing' the graph of $f(x)$. For a many-dimensional search space, where several parameters must fulfil a goal state, hill climbing combines all the parameters into a single similarity measure, equivalent to traveling in the direction of the largest positive derivative.

Hill climbing is prone to local optimums. One way to avoid local optimums is to run several instances of hill climbing from different initial conditions and choose the best result, or during the search, randomly move in either the direction of the derivative or some other direction. In *simulated annealing*, a "temperature" variable T dictates the chance of moving in a random direction, with a high T giving a high chance for random movement. During the simulated annealing session, T begins at a high value, and then decreases towards 0 at a rate of dt. At $T = 0$, either the global or some local maximum has been reached. For the chance p of reaching the global maximum, $p \to 1$ as $dt \to 0$, typically much faster than for an exhaustive search, but how much faster depends on the structure of the state space.

2.6.2 Knowledge based reasoning

Some problems are more difficult to model as a simple state space search, as the number of states would be too high. A knowledge system that contain a potentially large number of knowledge statements, from which new knowledge statements can be derived, would require one state for each combination of possible knowledge statements. This problem can be more easily modeled by a graph, where each node represents a knowledge statement, and new nodes of knowledge are reached by applying *logical rules* to knowledge already reached. This is called *knowledge based* reasoning.

A logical rule may take several knowledge statements as antecedents, thus the problem can't be narrowed down to just finding a short path through the graph to a desired goal, as the goal node may be dependent on having discovered some number of other nodes. The process of finding new knowledge statements by such repeated application of rules is called *forward chaining*. Forward chaining is said to be *data driven*, as the initial data determines what inferences are made and what conclusions are reached. The challenge in forward chaining is to apply the most useful inferences to the data, which will ultimately lead to some goal statement. Deduction and induction is often used in forward chaining.

The opposite of forward chaining is *backward chaining*, which begins with a goal statement, and attempts to infer some satisfying initial statements that would imply the goal statement. It can thus be seen as reasoning in the opposite direction of forward chaining, and is fit for abductive reasoning. Backward chaining is said to be *goal driven*, as the focus is on explaining a specific goal. The advantage of backward chaining, is that searching the graph does not require the consideration of several nodes at once to expand and explore additional nodes. The disadvantage however, is that expanding a new node may require other new nodes to be expanded, causing more "loose ends" that have to be explained with initial statements.

An *inference system* for knowledge based reasoning requires formal definitions of how knowledge can be used along with logical rules. Figure 2.2 shows an example *informal* inference graph for a mathematical proof, where several knowledge statements are used at once to infer new knowledge. Here, it is rather unclear what the set of applicable logical rules may be - the inference rules seems to include a lot of background knowledge not shown in the figure, such as properties of expressions containing prime numbers, which may be better modeled as pattern rules evaluating input expressions against each piece of knowledge, rather than rules for applying particular knowledge to particular expressions. And in general, the *"consider"-rule* may have a selection of operators such as $+$, $-$, $\times$ or $/$, and raise an infinite number of possible expressions using these operators in combination with known or newly proposed constants and variables, making the problem intractable with a breadth-first expansion strategy.

An inference system that were to discover this proof, must either restrict the number of possible inferences, such as by making them more general, or bias them in a sensible way between breadth and depth exploration. For example, examining the expressions raised for consideration in the figure, reveals that they have very important properties related to prime numbers. A selector for expressions to consider, may limit the selection to only expressions with such interesting properties. Then if no proof is found after inferring a graph when considering these interesting expressions, new expressions with less interesting properties may be considered.

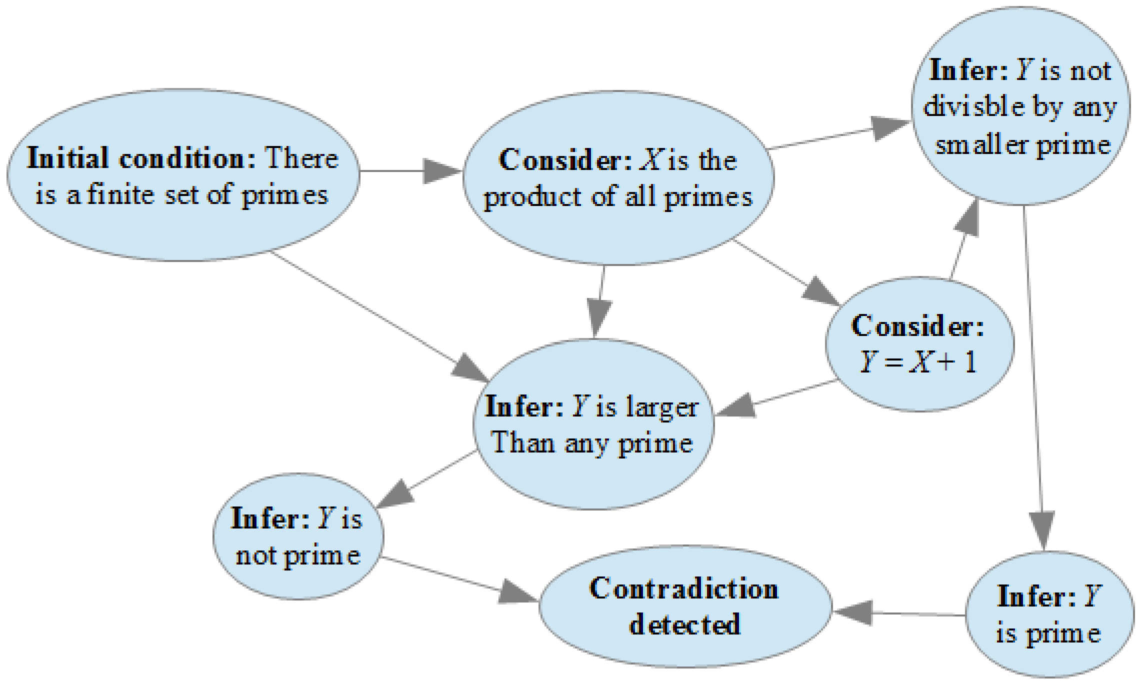

Initial condition: There is a finite set of primes
Consider: X is the product of all primes
Infer: Y is not divisble by any smaller prime
Consider: Y = X + 1
Infer: Y is larger Than any prime
Infer: Y is not prime
Contradiction detected
Infer: Y is prime

Figure 2.2: Proof by contradiction of infinite primes. (Variables can only have positive integer values.)

2.7 Semantic networks

Semantic networks represents knowledge in a graph of concepts (nodes) and semantic relations (edges) between the concepts. Reasoning can be performed along the edges by looking at the concepts associated with the concept in question.

Figure 2.3 shows a semantic network for a car. Such a network could be utilized by a diagnostics system to determine why a car is not working properly, by searching through the graph for potential causes to the problem. Commonly, statistical reasoning is applied to semantic networks, such that if the problem description for the car was "engine makes noise", the edge leading through the *engine* node may be rated the most relevant, simply due to containing a problem description word. From engine, the path to fuel could be followed associatively, and end up as the suggested issue due to the it having the "requires" relation for engine, which signals a possibility for inadequacy. It is however also possible to employ more knowledge intense reasoning for semantic networks.

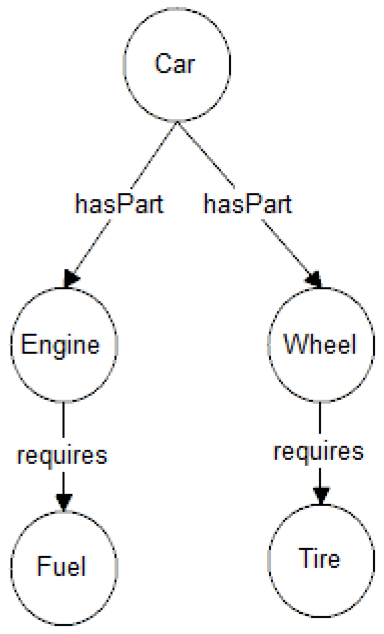


Figure 2.3: A simple semantic network of a car.

2.8 Production rule systems

Production rule systems are among the most widely used knowledge representations, much due to their use in constructing expert systems. (Reichgelt (1991)). They are commonly used in cognitive psychology to model human brain processing, such as by the cognitive architecture *Soar* (Laird et al. (1987)).

A rule is on the modus-ponens form 'IF (conditions) THEN (actions)'. The conditions (called the left-hand side or LHS), can be a pattern of data structures and variables that must be matched, while the actions (the right-hand side, RHS) can be any procedure. The memory containing the matchable structures is referred to as the *working memory* (WM), or *knowledge base* (KB). (In this thesis, it is referred to as KB.) The result of performing actions, called *firing* of rules, is often that elements are added to or removed from the KB, allowing new rules to fire.

An advantage of production rule systems is their modularity, in that knowledge can be loaded into and removed from temporary memory, while a permanent memory (the rule base) remains. This allows a large system to be partitioned into more efficient sub-systems. One variant is the *blackboard system*, where each subsystem communicates with each others by writing data to the blackboard. (A part of memory that remains when running subsystems are switched).

One weakness of production rule systems is dealing with uncertain knowledge. Experts often deal with rules that are only true in most cases, such as "birds can fly". One way of dealing with this is to have these rules as *default reasoning*, while an exception rule may trigger at a in the case of the rule not holding up, and preventing it from firing. Another way is to assign each piece of KB elements and inference a confidence rating, so that a confidence in new inferred elements can be estimated. This is done in the architecture outside of the rule engine, which runs like before. It is common to separate confidence into belief and disbelief, such that a confidence p in an assertion does not imply a confidence $(1 - p)$ in the negated assertion. (Which follows from Bayes' theorem under standard probability theory).

Another weakness of production rule systems is their computational efficiency, as the matching process is slower the more rules are in the system, and the more working elements are in KB. This can be mitigated by more computational efficient matching capabilities however, such as auto-associative memory.

2.8.1 Rete algorithm

The Rete Math Algorithm (Forgy (1982)) is an efficient approach for comparing a large collection of patterns to a large collection of objects, compared to a naive approach of checking every combination of object and pattern on each rule firing. Each LHS has a list of working memory elements it matches. Instead of iterating over the working memory elements on each rule firing, the LHS lists are updated when a working memory element changes. Thus to detect a pattern match, it is only necessary to look through each pattern's list of working elements. The productions are indexed in a tree structure with each condition of an LHS being a node, grouping rules that tests for the same thing together. Working element changes produces a token, which is propagated through the tree. From each affected pattern by the token, the agenda, a list of all pairs of productions and the matching status of their elements, is updated. Afterwards, all applicable productions in the agenda forms the conflict set, from which a production can be selected for the next firing.

2.8.2 Jess rule engine

Jess (Friedman-Hill (2003)) is a expert system consisting of a functional scripting language *Jess*, interpreted in Java, and a pattern matched-based production rule system. It is owned by Sandia National Laboratories. It uses an enchanted version of the Rete algorithm to solve the many-to-many matching problem. When the Jess engine is run, it will run until no more rules can fire, thus the amount of working elements in it's KB must be kept to a manageable amount to prevent the engine from running out of memory.

Jess variables and facts

Jess allow variables of regular Java format, and Jess template variables used for pattern matching. Each template has a name, and some number of fields with a name and value. Syntax:

```
(deftemplate TEMPLATE1
  (slot SLOT1)
  (slot SLOT2))
```

An instance of a template (with some values in it's slots) can be asserted as a Jess Fact. A Jess fact is uniquely identified with an integer ID, starting at 0 and increasing by 1 for each asserted fact. Only Jess facts are used during pattern matching. The syntax is

```
(assert (TEMPLATE1 (SLOT1 3)(SLOT2 "abc"))
```

which asserts a TEMPLATE1 with SLOT1 of value 3, and SLOT2 of value "abc". By using other template instance as the slot values, a graph structure can be built for use with pattern matching.

A variable in Jess must have a name starting with the '?' character. During pattern matching, every variable identifier is used for matching. The syntax for binding variables is

```
?template1 <- (TEMPLATE1 (SLOT1 ?value1)(SLOT2 ?value2))
```

Here the template called TEMPLATE1 is bound to ?template1, while it's slot values is bound to ?value1 and ?value2. Further pattern matching statements can specify properties of these variables.

Here, a second type of template, called TEMPLATE2, also with slots SLOT1 and SLOT2, is specified as also required to contain ?template1 and ?value1 for a pattern match:

```
(TEMPLATE 2 (SLOT1 ?template1)(SLOT2 ?value1))
```

Rules and pattern matching

In the Jess rule engine, the LHS of a rule is a pattern matching, leading to Jess statements in the RHS clause being executed for each combination of variables that matches the pattern. The syntax is

```
(defrule RULE1
  (conditions)
=>
(execute statements))
```

The statements to be executed will be function calls. Function calls can be indistinguishable in syntax to pattern matching, E.g. the Jess display facts function is written (facts), the same as a template named facts with no slots referred. However, everything in the LHS will be interpreted as pattern matching, while everything in the RHS will be interpreted as function calls. In the RHS, a variable binding is done with the function call (bind variable), rather than the arrow syntax used in the LHS.

2.9 Neo4j

Neo4j is a relational database that can be accessed through it's APIs, available in Java, Python, or a web page. It allows for creation of nodes with relationships between them, serving as edges in a graph. This graph can be searched with the declarative query language Cypher, returning a graph result that can be displayed graphically.

2.9.1 Cypher

Much like SQL, Cypher uses the MATCH keyword to express a pattern to be matched, and WHERE to pose restrictions on variables in the pattern. The RETURN keyword declares what variables (typically nodes) to be included in the query result. All relationships between those nodes will also be included. The query

```
MATCH (n) WHERE a.name = "a" RETURN n
```

Will return all nodes and relationships in the database with a property name with the value a.

Relationships between two nodes *a* and *b* can be expressed with the (a)-[:tag]->(b) notation:

```
MATCH (a)-[:KNOWS]->(b) RETURN a,b
```

This query return any pair of nodes *a*, *b* where *a* has a KNOWS relation to *b*.

Chapter 3

Literature review of related work

This chapter looks at relates work in knowledge discovery, starting with the more knowledge-based and moving to the more literature-based.

3.1 Early knowledge intensive discovery systems

Shrager and Langley (1990) describe computational approaches to scientific discovery. With backgrounds in both computer science and cognitive psychology. They divide scientific behaviour in a domain into *knowledge structures* and *processes* operating on them. Knowledge structures include

- Observations - recordings from the environment
- Taxonomies - defines domain concepts and subsumption relationships between them
- Laws - statements that summarize relations between variables, objects or events that can be observed
- Theories - hypotheses about the structures or processes in the environment, making references to unobservable objects
- Background knowledge - beliefs or knowledge about the environment not belonging to the domain

- Models - describes both observable and unobservable environmental conditions in a particular setting
- Explanations - connects a theory or hypothesis to a law by a chain of inferences
- Predictions - expected observations how the environment will develop from specific conditions, based on laws
- Anomalies - observations that cannot be predicted by laws

Scientific processes include the generation of observations, taxonomies, inductive/deductive laws, theories, models and predictions. *Experiments*, where physical settings are manipulated to correspond to a model, allows *Evaluations* - the comparison of observations with predictions. The *explanation* process connects a theory to a law, where an anomaly arise in the case that an explanation cannot be constructed.

They explain the advances in machine discovery, constituting mainly of representations of observations, laws, models and theories. Various systems by various authors for knowledge discovery are presented.

Langley et al. (1992) treats scientific knowledge as a search problem in a problem space consisting of symbols and operators operating them to achieve a goal state. A problem solver may use heuristics such as hill-climbing and means-end analysis to prune the search space. An *expert* in a field may have production rules that generate a useful operation based on recognizable problem states. They place problems on a continuum of *structured* vs. *ill-structured*. Problems must have a definite mechanically applicable criterion for testing any proposed solution, a problem space that can represent the initial state, goal state, and all other states that could be considered while attempting to solve the problem, and state changes that can be represented in some problem space. Knowledge the problem can acquire about the problem must also be representable in a problem space. The problem fall on the the continuum of well structuredness depending on whether the conditions are met with a practicable amount of computation.

They construct the BACON.1 program, using the production system language PRISM, to perform heuristic searches and discover empirical physical laws. One such is Kepler's Law, regarding the relation between a planet's distance D from the sun and it's period P, and it states

$D^3/P^2 = c$, where c is a constant. BACON.1 is allowed to make observations about some planets A, B, C, such as "For planet A, what is P?" It also has 3 heuristics: 1. "If the observed values of a term are all the same, infer it as constant.", 2. "If two terms increase together, consider the ratio of the terms.", 3. "If one term increase while another decrease, consider their product."

Starting by observing P, D for all planets, BACON.1 observes $P_A = 1.0, D_A = 1.0$, $P_B = 8.0, D_B = 4.0$ and $P_C = 27.0, D_C = 9.0$. From these observations, D increases with P, the second heuristic applies, and BACON.1 considers the ratio $term1 = D/P$. Calculating it finds $term1_A = 1.0$, $term1_B = 0.5$, $term1_C = 0.333$. BACON.1 spots that D increases as $term1$ increases, so it considers their product, $term2 = D(D/P) = D^2/P$, giving $term2_A = 1.0$, $term2_B = 2.0$, $term2_C = 3.0$. The second heuristics applies again, with $term1$ and $term2$ increasing together, and BACON.1 defines their product: $term3 = (D/P)(D^2/P) = D^3/P^2$. With $term3_A = 1.0$, $term3_B = 1.0$, $term3_C = 1.0$, BACON.1 is finally able to apply the first heuristic, that $term3$ is constant, which so happens to be Kepler's law.

BACON.1 was able to generate this protocol using it's production rule system, consisting of 16 production rules, 5 of them involving gathering of data, another 5 detecting regularities in data, 3 proposing higher-level terms, and 3 performing garbage collection on memory. Other laws found by BACON.1 includes Boyle's law, Galileo's law of uniform acceleration, and Ohm's Law. They also describe a later version of BACON, BACON.3, which is able to discover more complex laws, such as Coulomb's Law and the Ideal-Gas Law.

Falkenhainer (1990) argues that when constructing plausible explanations for phenomenas, separating between deduction, abduction and analogy is superfluous. He refers to finding candidate hypotheses while making assumptions about missing knowledge, as the interpretation-construction task.

He proposed the *similarity conjecture*, which states that all interpretation-construction tasks fall on a continuum of explanatory similarity between the current scenario and some previous scenario. The previous scenario may be an actually recorded experience, a prototypical (generalized) experience, or imagined from existing knowledge. He divides the spectrum into 4 scenarios. In the *deduction scenario*, an observed phenomena's explanation can be directly deduced from known facts. In the *assumption scenario*, the phenomena cannot be deduced from known facts, but by making a consistent set of assumptions about some unknown facts. In the *gen-*

eralization scenario, a inferred candidate explanation is known to be false, due to a required condition is false, yet by replacing the condition with a next most general relation, it is true. In the *analogy scenario*, no candidate explanation is found directly. A serial of assumptions, taken from an assumed analogical situation, can however produce a candidate explanation.

He created the system *PHINEAS* to illustrate this on various physical systems. PHINEAS has 3 sources of knowledge, it's initial domain theory, prior experiences, and observations, which are the current target for the interpretation-construction task. PHINEAS runs in 4 stages: The *access* stage, where memory is searched for similarity to an behavioural abstraction of the current observation. In the *mapping and transfer stage*, an initial hypothesis is generated. It begins by retrieving the model used to explain the recalled earlier experience, and mapping them over to the current observation. This gives rise to candidate inferences for the explanation. These may however be incomplete. For example, the inferences may reference objects that existed in the recalled experience, but not in the current observation. The domain theory is consulted to retrieve more details about each candidate inference. This may repeat in cycle until an candidate explanation is formed. In the *verification stage*, simulations are performed to compare the observations with simulation results. In the *revision stage*, inadequate hypotheses are analysed based on the differences in observation and simulation results. This stage is however not implemented.

3 examples are used to illustrate the similarity continuum, using liquid flow as PHINEAS' domain theory. An observation of liquid flow is directly explained as a liquid flow instance, *osmosis* is explained as a close generalization of liquid flow, and *heat flow* is explained by conjecturing the existence of temperature-affecting fluid.

From an initial knowledge consisting of 9 processes - liquid flow, liquid drain, heat flow, boiling, heat-replenish, dissolve, osmosis, linear motion and spring-applied force, when presented with an observations of harmonic motion, including an LC circuit, a cantilever pendulum and a torsion oscillator, PHINEAS was able to explain them using the spring-mass knowledge.

3.2 Literature based discovery systems

Swanson and Smallheiser (1997) created the open discovery support system Arrowsmith. The user's goal is to perform a Swanson linking $A \rightarrow C$, through an intermediary B. The discovery process is split into two parts: Procedure I and procedure II. For procedure I, the user provides the A term. Arrowsmith collects all titles from MEDLINE containing A, and extracts all unique words from those titles. This constitute all potential B candidates. The list is filtered with an a priori stop list of 5000 words, containing both generic non-domain words (such as 'able','about'), and also domain words deemed too general to meaningfully link terms (such as 'clinical', 'drugs'). Then, all B candidates are searched in MEDLINE, to find the total amount of titles they occur in. Only those B terms that occurs more frequently in titles along with A than in titles without A are kept. After that, the user is presented with the B-list, and can further remove candidates that is deemed too broad. The remaining words each forms the basis for a new MEDLINE search, with a new restriction: Each set of records from these searches are narrowed down to particular record categories, (such as 'deficiencies', 'toxins'), chosen in advance for their likelihood of containing particularly interesting target words. After executing the search, a list of $A - B$ linkages is formed, such that each A term now links with other A terms through B, forming an $A \rightarrow B \rightarrow C$ Swanson link *discovery candidate*. Each candidate is ranked according to the amount of B-words participating in the $A \rightarrow B \rightarrow C$ linkage.

Arrowsmith was able to rediscover Swanson's earlier linkages between Raynaud's Syndrome/-Fish Oil and migraine/magnesium, but failed to link the earlier discovery of somatomedin C and arginie, due to a Swanson link never occurring in the titles of the MEDLINE records.

Wilkowski et al. (2011) proposed the term "discovery browsing", an open discovery form where a user interactively assist in the knowledge discovery process, by marking interesting findings for further investigation. It uses semantic predications from the SemRep (Cutting et al. (1992)) relation extraction tool to form a graph with predication arguments as nodes, and predicates as edges. The user is presented with paths in the graph, forming chains of relationships. Their goal was for the exploration of this path to form extension of Swanson linking with several intermediate B terms: $A \rightarrow (B_1 \rightarrow B_2 \rightarrow ... \rightarrow B_n) \rightarrow C$. They rank expansion suggestions by the node's *degree centrality*, a measure of how much other nodes are connected to it.

They tested the system with Depressive disorder, with serotonin as the starting node for browsing. Although no new discoveries were made, the top ranked concepts related to depression were not all studied together, and the authors suggest the system may highlight areas that benefit from further study.

Camerona et al. (2013) used a graph-based approach to recover and decompose Swanson's *Raynaud Syndrome-Fish Oil* hypothesis. They extended Anyanwu's notion of a semantic association to include reachability, so that any vertex in a directed graph is semantically associated with the other vertexes it can reach. They criticized Wilkowski et. al.'s degree centrality approach, due to it's potential of eliminating outliers.

Leveraging the semantics of assertions extracted from literature in combination with background knowledge, they recovered the 3 informative associations commonly considered to make up Swanson's hypothesis, and further decomposed it into 16 additional associations, formulated as chains of semantic predications. They do however point out some issues with LBD. First, most of the literature they leveraged in their experiment was not part of article titles or abstracts, meaning the full article texts may be necessary for throughout LBD. Second, they were not able to extract all the semantic information from the articles, relying so far on manual annotation. Third, the heavy performance cost of traversing large graphs is a hindrance, causing many query execution platforms to time-out.

3.3 Evaluation of hypotheses

Lee et al. (2011) extended Swanson's ABC model to include context information, using *context vectors*, which consists of term frequencies of various terms in an article. They used Cosine similarity and Spearman Correlation to assess the similarity of the vectors. Thus two abstracts, one containing $A \rightarrow B$, and the other $B \rightarrow C$ were ranked based on the similarity score of their context vectors. They looked only at ABC patterns of the form disease-gene-drug.

The experiment was done on PubMed literature related to Alzheimer's disease. ER was done using a Conditional Random Field-based sentence detector, and terms were categorized into four categories: Gene, Drug, Disease and Symptom. This was done using PharmGKB (Pharmacogenetics Knowledge Base) and CTD (Comparative Toxicogenomics Database) entity dictionaries. Then the entities were mapped to the UMLS (Universal Medical Language System) entity dictionary, to obtain the context vectors.

To evaluate their system, they measured the precision of the results when matched with known answer sets in the PharmGKB and CTD database. They grouped the results in top 100, top 500 and top 1000, and as a baseline, compared the groups to the precision of the frequency based ABC model. A hybrid score, utilizing both the frequency information, and the context similarity, was able to outperform the baseline for all cases.

Anyanwu and Sheth (2002) Argues that different simple and complex notions of relationships between entities will be important to reason about in query languages for the Semantic Web. They illustrate this with a framework for finding semantic associations between entities, and illustrates it's use with the semantic web knowledge modeling standard RDF. They define a notion of semantic association, and a semantic association query (SAQ) operator, named the ρ operator, which return ranked sets of relations connecting the entity pair being queried.

They formally define a property sequence, which connects two RDF properties through relationships. Property sequences can be ρ-isomorphic, such that paths along subproperties of the corresponding properties in another path, are ρ-isomorphic. Other similarities between corresponding properties in two property sequences, such as being a subproperty of the same property, are also considered isomorphic. They define a semantic association as follow. Property sequences can also be joined, meaning that both sequences intersects at some property node.

Finally, a semantic association between two properties x, is defined as either x and y being the end of a property sequence, or x and y being the origin node of two property sequences that are either joined or ρ-isomorphic.

The ρ operator attempts to find ρ-isomorphic relationships. First prunes the search space using schema knowledge to check if the query entities belong to a common class, parent class, or classes related by a subclass relationship, as it they cannot be ρ-isomorphic. The user can enter a context for the query, consisting of scope declarations such as RDF schemas, limiting the search space. Optionally, relevance rankings can be used to assign RDF properties an ordered relevance. Higher ranking can also be assigned for longer or shorter sequences.

Anyanwu et al. (205) presents SemRank, which ranks semantic associations using semantic and information-theoretic techniques.

The SemRank value of a query result combines the metrics predictability, refractions, and semantic matches.

Predictability is the amount of information gained by a user from seeing a search result. This depends on the likelihood of whether the user could have predicted the result beforehand. By measuring the information conveyed by a result, the

Refractions are deviations by a path from that specified in an RDF schema due to multiple classifications of nodes.

A semantic match (S-Match) of a user provided property is a match between a itself or it's super/subproperty. The degree of the match depends on the distance from the search property to it's match in the property hierarchy, resembling the similarity score between concepts in an ontology.

A user provided context adjusts what metrics are considered the most relevant for ranking, example being a conventional or investigative context. In a conventional context, results with a high predictability score are favored, while in an investigative context, results with a low predictability score are favored.

To compute the SemRank values, they introduce the SSARK (*S*emantic *S*earching of *A* different *K*ind) system, producing the top-K results. To save computational time, the resulting ordering is only approximately correct.

Wren et al. (2004) discovered and ranked relationships between objects of interest in titles and abstracts taken from MEDLINE, based on co-occurrence of terms. If an object A co-occurred with another object B, and B co-occurred with another object C, then A and C are said to be implicitly related

As each co-occurrence may not constitute a meaningful relationship, they manually surveyed each co-mentioned object within a MEDLINE record sample, and assigned them a fuzzy probability of being meaningful. The strength of the relatedness between objects was based on this probability and the number of co-occurrences. They used entity recognition of primary names/synonyms for medical entities, such as genes, diseases and phenotypes from databases such as OMIM, HGNC and Genome Ontology.

An expected observation count, based on how often two terms would be connected based on pure chance, was found based on a network of randomly connected nodes. Thus rare terms in the literature, such as specific transporter genes, would be less expected to serve as the B object connecting A and C. The ranking of results was based on the observed to expected ratio

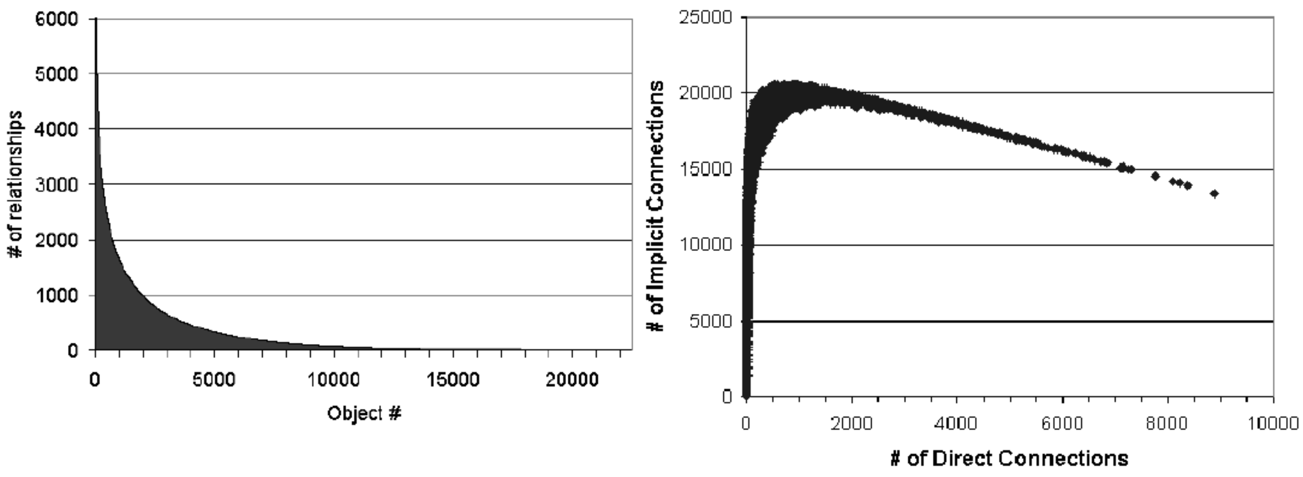


Figure 3.1: Direct versus implicit relationships (diagram by Wren et al.)

(Obs/Exp), such that nodes observed more than expected were higher ranked.

A total of 12 037 763 records were used, yielding a network of 3 482 204 unique relationships. When comparing the number of direct relationships to the number of implicit relationships, they found that the amount of implicit relationships quickly adds up to similar numbers for those objects with fewer direct relationships, to those with more. (As seen in figure 3.1.) Thus, ranking of the relationships constitutes a much bigger task than discovering potentially useful relationships.

Based on their resulting ranking of terms related to cardiac hypertrophy, Wren et al. were able to hypothesise that the drug Chlorpromazine (CPZ) may provide an anti-hypertrophic effect in the heart. They confirmed this to be the case in experiments on rodents.

Lally et al. (2014) built on the IBM Watson question answering system to be used with medical questions with a higher accuracy than the base Watson system. The system, WatsonPaths, takes as input a natural language scenario, which can be divided into subquestions. The subquestions and inferences made from them make up an assertion graph. For the scenario's they used a set of medical test preparation questions, which includes a section of information, followed by a diagnostics question.

The system has several stages. First, *scenario analysis* identifies important factors in the input scenario, such as patient age, gender, preexisting conditions and test results described for the scenario. Each factor becomes a node in the graph. Second, *Node prioritization*, the priority of exploring paths from each node is determined, in the case that exploring all of them is not feasible. One measure used is the confidence in the node's factor, and how fruitful expansion of the node is expected to be. Third, *Relation generation* constructs edges in the assertion graph. By taking some factors x, y, the base Watson system is asked question such as "How does x relate to y"? Using a medical ontology to classify the factors, a recognized disease z may pose the question "What symptoms does z cause?". The answers to the questions are added as nodes to the graph, with edges to all factors used to pose them. Next, *belief computation* calculates the confidence in each assertion in the graph. These steps may be repeated, adding new subquestions based on previous subquestion. After each iteration, *hypothesis identification* is performed, marking some nodes as potential final answers to the scenario.

To acquire good edge confidence scores, WatsonPaths was trained with machine learning, using both full scenarios and atomic subquestions as training data. Their future plans involves cooperation between WatsonPaths and users, so that both can learn through it's usage.

3.4 Summary

Knowledge discovery started out with traditional AI methods, using knowledge intensive techniques such as expert systems. Assembling the knowledge that the systems require however, is expensive. Like much of AI, the research moved in the direction of using more statistical methods, from Swanson's manual attempts, to analysis of big data bases of scientific records retrieved through the modern Internet.

While statistical methods are very successful at linking potential related knowledge together by uncovering implicit relationships, just uncovering these implicit relationships is often too shallow to be helpful. With some huge efforts being put into various knowledge bases and systems, such as semantic web and open ontologies, more knowledge intensive methods are beginning to make a comeback. The use of IE techniques are now more common, as extracted knowledge can be linked with the semantic web knowledge bases. The importance of ranking hypotheses, to sort out the good ones from more spurious candidates, has been grown with the systems' abilities to suggest more hypothesis candidates.

One big problem in LBD system, is to evaluate their performance. Although newer systems are believed to perform better than the older systems, there is currently no standardized evaluation procedure to verify this. One challenge is that more knowledge-intensive systems may rely on very domain-specific tools and resources, making them hard to compare to domains without the same resources and tools.

A new trend is LBD on a more cross-disciplinary level, such as OCEAN-CERTAIN, which involves several large fields (biology, chemistry) related to the biological pump. With such large fields playing an important in a system, the potential for isolated knowledge is high.

The knowledge discovery system developed in this thesis follows this development path, being part of the OCEAN-CERTAIN project, and using knowledge intensive inferences which are to be used along with NLP-extracted domain knowledge. The contradiction detection goal of the system closely follows Shrager's et al.'s model of scientific behaviour, by comparing observations from experiments with expected results based on background knowledge, similarly to the early PHINEAS system by Falkenhainer. A challenge in the climate/oceanographic domain however is that unlike in physics, a lot of the knowledge is typically handled informally, while knowledge discovery systems rely on formal knowledge. The next chapter seeks to address this issue.

Chapter 4

Conceptual modeling

The conceptual architecture consists of two main components: A knowledge base consisting of entities and relations, and an inference engine along with inference rules, for use with literature-based knowledge discovery. The important consideration for such a representation is that of *ontological commitment* - that it is *expressive* enough to hold the necessary knowledge and reason about it, while being *computationally tractable*, which often falls at odds with expressibility. For the OCEAN-CERTAIN project, the focus is on reasoning with causal relations between change events, and the inference engine chosen is a production rule system, which commits to using IF-THEN rules, where concepts and their relations utilized by the rules are encoded in a semantic network (the domain ontology). The use of a production rule system excludes the use of other techniques such as cased-based reasoning.

The system makes a distinction between *working memory* (WM) and the *Knowledge Base* (KB). Only the KB is accessible to the *inference engine*, which consists of a set of general and more domain specific inference rules operating on the knowledge in KB. Initially, the KB contains background knowledge from the domain, such as a domain ontology (. Figure 4.1 shows an overview of the system. as new observations enter the system, they are first entered into the WM, from which the system decides which knowledge is transferred to the inference engine's knowledge base. New statements inferred from the inference engine are again evaluated in working memory before being able to enter the knowledge base. Section 4.4.2 explains how the knowledge is evaluated.

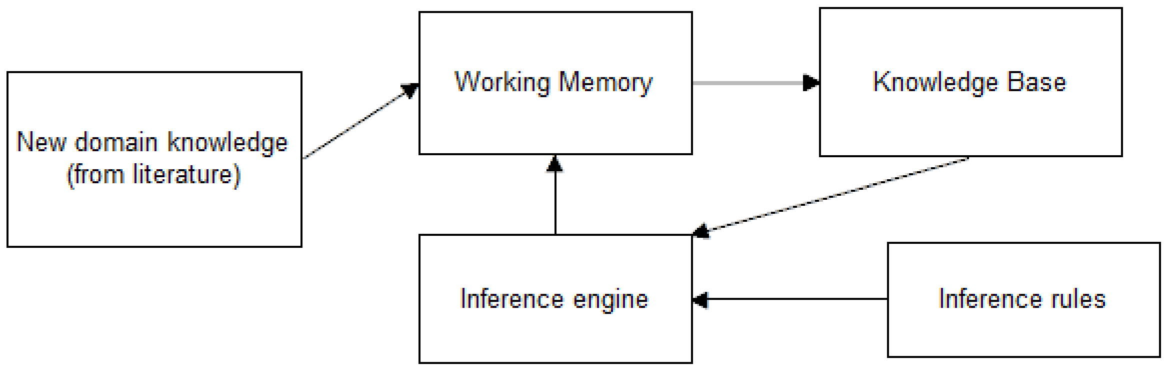


Figure 4.1: Overview of the system.

4.1 Domain knowledge modeling

The main knowledge representation is split into two types, the domain independent knowledge and domain dependent. The domain independent rules are committed to handle inferences with causal relations between change events, while the domain dependent can be seen as extendable to handle more use cases.

Variable

A variable corresponds to a concept as expressed in the domain literature, be it a single term like "iron" or "phytoplankton", or a longer expression such as "pco_2 last ice age". The system has knowledge about variables based on relations involving the variable, such as "CO_2 is-a chemical substance", where metal is another variable, however the system has an open-world assumption of any variables, such that possibilities such as also "CO_2 is-a species" are not excluded even if they are not in the system's knowledge base. For ontological purposes, it is useful that variables has canonical forms, so that the system may recognize that two variables, such as "iron" and "Fe" refer to the same concept. (See section 4.4.1 regarding ER.)

Event

An event has a *variable* that the event involves, and a *type*. Event types, range from *increasing*, *decreasing* or *changing* for an unspecified change, to more domain specific such as *limitation*. Events can be written in predicate form, as $event(variable, type)$, or in arrow notation for change events. (↑X, ↓Y, ↕Z). Too indicate the observation of an event, the event must be enclosed by the predicate $Observation()$, e.g. $Observation(\uparrow X)$, or when the system has inferred that an event should take place, $ExpectedObservation(\uparrow X)$.

Causal relation

A causal relation has two events, a *cause* event and *effect* event. If E_1 is the cause event of a causal relation, and E_2 is the effect event, the relation can be interpreted as "event E_1 causes event E_2". The predicate form is $causal(E_1, E_2)$, and the arrow notation is $E_1 \implies E_2$.

Correlation

A *correlation relation* states a correlation between two events E_1 and E_2, without a known direction. It can be written $correlation(E_1, E_2)$, or $E_1 \rightsquigarrow E_2$. While correlations may not themselves be conclusive regarding any causal direction, they may serve as weaker evidence for some other reasoning path.

Subsumption

A subsumption has two variables, the *general* variable and the *specific* variable. If X is the specific variable of a subsumption, and Y is the general variable, the relation can be interpreted as "X is subsumed under Y", "X is-a Y", or "X is an instance of Y". The predicate form is $is-a(X, Y)$.

Subsumptions are important during reasoning, for example if a person P_1 asks P_2 'Do you have a car?' and P_2 answers 'I own a truck', then P_1 can use a subsumption tree (See figure 4.2) to generalize 'truck' and 'own' to reason that indeed, P_2 has a car.

It is important to note that in the domain, a subsumption may not hold strictly. For a subsumption "X is-a Y" one may say that "Y has property Z", when a majority of Y has property Z, without necessarily X having property Z. Strict subsumptions are often found in mathematics, such as "every natural number is a real number", which hold true for the infinite amount of natural numbers. Less strict subsumptions however, called defeasible logic rules, such as "birds can fly", may hold true only for a large fraction of birds, but have some exceptions, (such as

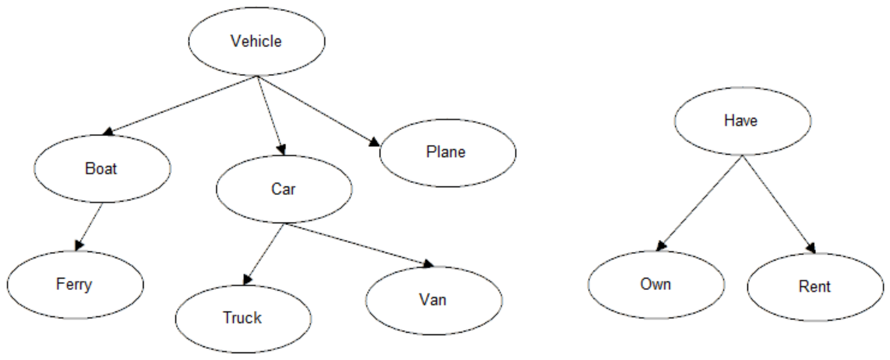


Figure 4.2: Example subsumption hierarchies.

penguins). Thus, if P_1 tells P_2 'A bird zoomed past in front of my face!' P_2 may plausibly assume that a bird was flying past P_1. If P_1 adds however, 'On my boat expedition to Antarctica.' another plausible inference arises, as the bird may have instead been a penguin being thrown around by playful killer whales, as the 'birds can fly' rule is not nearly as plausible in Antarctica.

As such contexts plays a huge rule in determining what inferences are plausible, subsumptions in the model also comes with a $bound$ property, either "upper", "lower" or "none", for use during inferences. An "upper" bound means that relations cannot be generalized from the specific variable to the general variable. A "lower" bound means that relations cannot be inherited from the general variable to the specific variable. A "none" bound has no such restrictions. Some such bounds that may be appropriate for figure 4.2 is an upper bound for vehicle in regards to plane and boat, as it may not be as plausible if P_1 says 'I have a vehicle' for P_2 to infer that the vehicle may fly through the skies, as it is to assume that it may ride on the road.

While there is an important distinction between subsumptions that represents an is-a relation that may have been crafted for an ontology, such as *penguin* fitting into a species hierarchy of biology for birds, and a syntactical generalization such as 'pco$_2$ last ice age' to 'pco$_2$', for the purpose of this modeling they are both treated as a subsumption relation, but given different confidences. (See section 4.4.2)

Domain specific relations

Domain specific relations are relations such as *eats, produces* and *outcompetes*. In predicate form, they are written with the subject as the first parameter, and the object as the last parameter, such that $eats(X,Y)$ means "X eats Y." This knowledge comes from the domain ontology created for use with the system, and contributes to the background knowledge necessary to make inferences.

4.2 Inference engine

Inference rules produce new domain knowledge from existing domain knowledge. The reasoning employed is not sound, but is reminiscent of plausible reasoning performed both in science as well as in daily life. Among the most common such reasoning is the transfer of properties from one entity to another, such as "milk contains bacteria, and bacteria may cause food poisoning, therefore maybe milk may cause food poisoning".

Each inference has some input statements and output statements. The system uses six domain independent and three domain dependent inferences, each taking some input statements which are relations between variables, and produce some number of output statements. For rules involving causal relations, the unspecified change direction ($\updownarrow$) is given to mean that all change events in the rule must have the same direction.

4.2.1 Domain independent inferences

IR1 *Causal transitivity* connects two causal relations between events. It corresponds to

$$\begin{array}{l} \updownarrow X \Longrightarrow \updownarrow Y \\ \underline{\updownarrow Y \Longrightarrow \updownarrow Z} \\ \therefore \updownarrow X \Longrightarrow \updownarrow Z \end{array}$$

IR2 *Generalization of cause* poses that if the variable in the *cause* event of a causal relation is a *specific* variable in a generalization relation, then the *general* variable of that generalization is also the cause event variable in a corresponding causal relation. It corresponds to

$$\begin{array}{l} \updownarrow X \Longrightarrow \updownarrow Y \\ \underline{X \text{ is-a } Z} \\ \therefore \updownarrow Z \Longrightarrow \updownarrow Y \end{array}$$

IR3 *Generalization of effect* poses that if the variable in the *effect* event of a causal relation is a *specific* variable in a generalization relation, then the *general* variable of that generalization is also the effect event variable in a corresponding causal relation. It corresponds to

$$\begin{array}{l} \updownarrow X \Longrightarrow \updownarrow Y \\ Y \text{ is-a } Z \\ \hline \therefore \updownarrow X \Longrightarrow \updownarrow Z \end{array}$$

IR4 *Specialization of cause* poses that if the variable in the *cause* event of a causal relation is a *general* variable in a generalization relation, then the *specific* variable of that generalization is also the cause event variable in a corresponding causal relation. It corresponds to

$$\begin{array}{l} \updownarrow X \Longrightarrow \updownarrow Y \\ Z \text{ is-a } X \\ \hline \therefore \updownarrow Z \Longrightarrow \updownarrow Y \end{array}$$

IR5 *Specialization of effect* poses that if the variable in the *effect* event of a causal relation is a *general* variable in a generalization relation, then the *specific* variable of that generalization is also the effect event variable in a corresponding causal relation. It corresponds to

$$\begin{array}{l} \updownarrow X \Longrightarrow \updownarrow Y \\ Z \text{ is-a } Y \\ \hline \therefore \updownarrow X \Longrightarrow \updownarrow Z \end{array}$$

IR6 *Modus ponens* infers expected observation from earlier observations through causal relations.

$$\begin{array}{l} \updownarrow X \Longrightarrow \updownarrow Y \\ Observation(\updownarrow X) \\ \hline \therefore ExpectedObservation(\updownarrow Y) \end{array}$$

4.2.2 Domain dependent inferences

The domain dependent inferences are mostly used during the abduction phase to plausibly explain contradictions, where the predicate $MustExplain$ is used to restrict the activation of the inferences. (See section 4.4.3 for when they are applicable).

DR1 *Eats-rule* tries to explain change events, in the case that some entity is *eating* the entity in the change event. Using the predicate $oppositeChange(type)$ to express

$$oppositeChange(increasing) = decreasing$$

$$oppositeChange(decreasing) = increasing$$

Given *event* $E_1(X, a)$ and *event* $E_2(Y, oppositeChange(a))$:

$$\frac{MustExplain(E_1) \quad Y\ eats\ X}{E_2 \implies E_1}$$

DR2 *Produces-rule* tries to explain change events, in the case that some entity is *producing* the entity in the change event. Given *event* $E_1(X, a)$ and *event* $E_2(Y, a)$:

$$\frac{MustExplain(E_1) \quad Y\ produces\ X}{E_2 \implies E_1}$$

DR3 *Outcompete-rule* works in the case of variables limited by some other variable. It states that the outcompeted variable is expected to decrease, while the limitation has no effect on the other variable.

$$\frac{Observation(event(X, limitation)) \quad Observation(event(Y, limitation)) \quad Outcompetes(X, Y)}{ExpectedObservation(\downarrow Y)}$$

4.2.3 plausibility of inferences

For the domain independent inferences, only causal transitivity and modus ponens are logically valid. The remaining four rules are examples of plausible inferences that are often used by domain experts. The plausibility of the inferences are dependent on the properties of the particular variables involved. Constructing hypothesises using these inferences is therefore a balance between breadth of hypothesises constructed, and plausibility. One thing to note however is that since the events in causal relations all are in the category of *change*, whether it is "increasing", "decreasing" or "changing", the *generalization of effect* will typically hold true. *specialization of cause* would be valid for generalizations that held strictly true for all members, but remains merely plausible for non-strict generalizations. *Generalization of cause* and *specialization of effect* are usually the least plausible inferences, and more akin to begging a question.

For the domain dependent inferences, none of them are valid, and due to the more complex interactions that can take place in the domain, they are only considered only slightly plausible.

4.3 Domain ontology

For this thesis, a small ontology was created ad hoc, shown in figure 4.3. The system's background domain knowledge is comprised of statements using he entities and relations in this ontology.

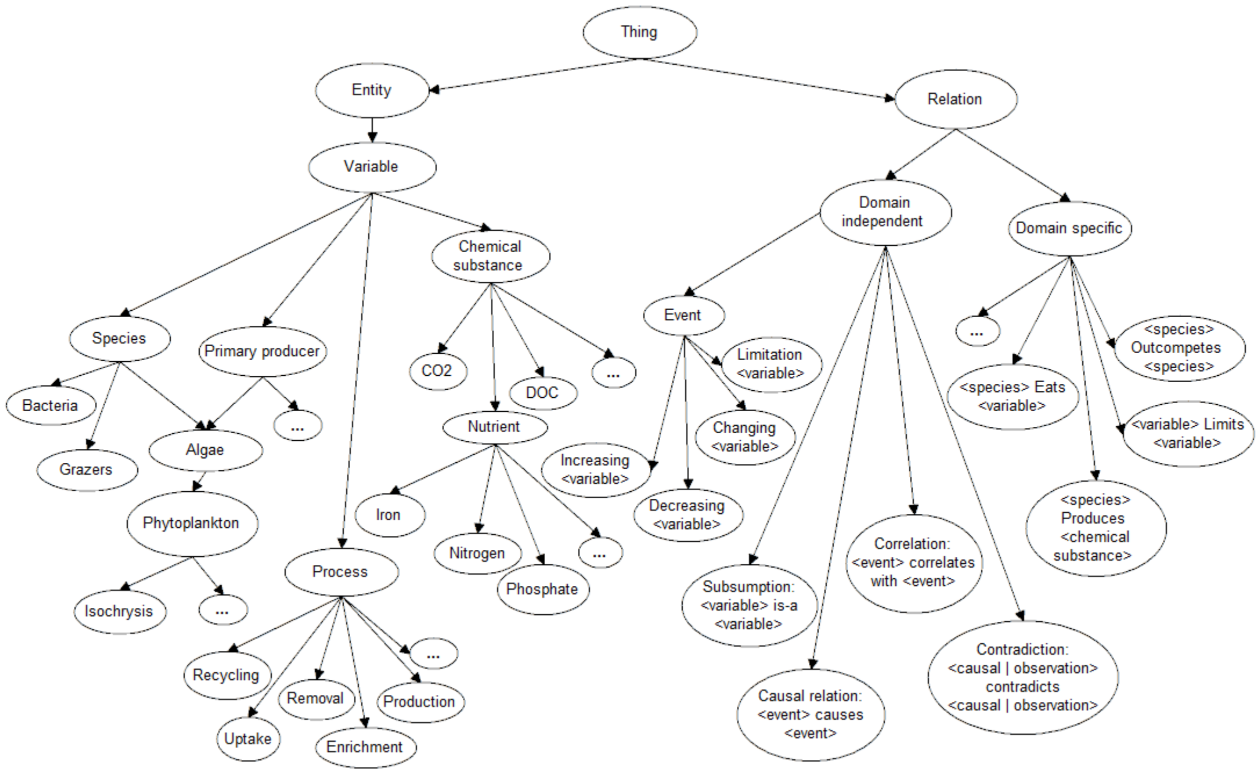


Figure 4.3: Domain knowledge model.

4.4 Work flow of the LBD process

When knowledge statements from the literature are entered into the system[1], they are first entered in WM (see figure 4.4), where syntactical generalization and ER is performed. The generalizations created by syntactical generalization is marked with a lower bound, to prevent specialization rules from utilizing the non-generalized statements. The same is done for manually entered generalizations that represents syntactical generalizations.

Then the system starts a new round of discovery process on the knowledge in KB, explained in Algorithm 1. An inference rule of which all antecedents match with some literature knowledge and other knowledge in the KB can be fired and produce new statements. The inferred statements can in turn trigger new inferences. A discovery round continues as long as there are inference rules that can fire using knowledge in the KB.

Algorithm 1 A round of discovery process

```
$LitKnow \Leftarrow$ new causal relation from literature
**if** $cause$ or $effect$ part of $LitKnow$ can be generalized **then**
    $LitKnow \Leftarrow LitKnow \cup \text{GENERALIZE}(cause, effect)$
**end if**
$WM \Leftarrow WM \cup LitKnow$
$KB \Leftarrow KB \cup LitKnow$
$hypCandidates \Leftarrow \emptyset$
**while** an inference rule $IR_i$ can fire **do**
    $K \Leftarrow \emptyset$
    **for all** antecedents $S_j$ in $IR_i$ **do**
        $K_i \Leftarrow$ statement in $KB$ matching $S_j$
        $K \Leftarrow K \cup K_i$
    **end for**
    $newKnowledge \Leftarrow \text{EXECUTE}(IR_i, K)$
    **if** confidence of $newKnowledge \geq threshold$ **then**
        $KB \Leftarrow KB \cup newKnowledge$
        $hypCandidates \Leftarrow hypCandidates \cup newKnowledge$
    **end if**
    $inferencePath \Leftarrow$ link $K$ with $newKnowledge$
    update $WM$ with $inferencePath$
**end while**
```

[1]Note that while the goal of the OCEAN CERTAIN project is to extract causal relations from literature, this has not yet been achieved, and knowledge statements are either added to the system from manually annotated files, or ad-hoc.

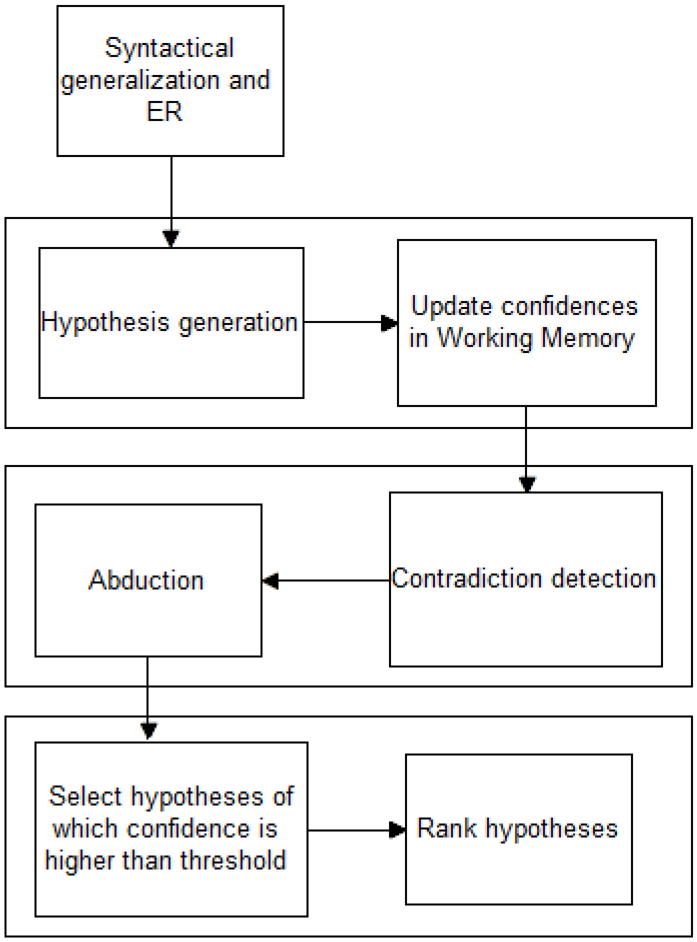

Syntactical generalization and ER
Hypothesis generation
Update confidences in Working Memory
Abduction
Contradiction detection
Select hypotheses of which confidence is higher than threshold
Rank hypotheses

Figure 4.4: Diagram of the work flow.

4.4.1 Syntactic generalization and ER

Variables in natural sciences tend to be complex rather than atomic entities and expressed as noun phrases containing multiple modifiers Marsi and Öztürk (2015), e.g. *oxygen depletion in the upper 500 m of the ocean* or *timing and magnitude of surface temperature evolution in the Southern Hemisphere in deglacial proxy records.* Generalizing these variables by stripping away these modifier words and using canonical names for entities, such as equating PCO_2 with CO_2 allows the system to connect such expressions from different literature sources. A simple entity recognition scheme, consisting of looking for entity names and it's list of synonyms in a variable expression, was used for the manually annotated corpus.

**Consider the causal relation "$\uparrow$net high latitude productivity $\implies$ $\downarrow$PCO$_2$ of the last ice age" extracted from a literature paper. Upon entering the system, entity recognition is performed on each of the variables in the relation. "PCO_2 of the last ice age" does not have it's own entry, but is recognized as CO_2, yielding the relation "PCO_2 of the last ice age is-a CO_2". "Net high latitude productivity" is recognized as "productivity", yielding "net high latitude productivity is-a productivity". is-a relations can by the system be assigned a *bound* property, taking the value UPPER, LOWER or NONE, which prevents generalization and specialization inferences from being being made regarding is-a relations with UPPER and LOWER bound respectively. These syntactically relations are assigned the bound property LOWER, and an for the relation is inserted in the WM inference graph with a confidence corresponding to syntactical generalization. If the confidence meets the confidence threshold, it is asserted in KB. When IR matching occurs, the causal relation from literature may match with the generalization of cause rule with the CO_2 generalization, forming "$\uparrow$net high latitude productivity $\implies$ $\downarrow CO_2$".

If the confidence for this new hypothesis was above the threshold, it is asserted in KB, and matching with generalization of effect and the primary productivity relation may produce $\uparrow$productivity $\implies$ $\downarrow CO_2$.

4.4.2 Hypotheses generation

Whenever one of the inference rules is executed and produces a statement, the LBD system will try to judge how much *confidence* it has in the inferred statement, based on it's confidence in the input statement to the inference, and the confidence in the used *inference rule*. When an inference $IR_i(X,Y) \implies I$ is performed, confidence of the newly inferred statement I is calculated as

$$c(I) = c(X) \times c(Y) \times c(IR_i) \tag{4.1}$$

If the knowledge inferred in this way have a high enough confidence, and does not exist in any of the inputted knowledge to the system, it will gain the status of *Candidate Hypothesis*, to be further evaluated by the system. If it is judged by the system to be interesting then it qualifies as *D*iscovery Hypothesis, and if appreciated by the researcher(s) then it is considered a *d*iscovery.

The inference rules *causal transitivity, generalization of effect* and *specialization of cause* are assigned a high confidence, $c(IR_1) = c(IR_3) = c(IR_4) = 0.95$ while *generalization of cause* and *specialization of effect* are assigned a lower confidence, $c(IR_2) = c(IR_4) = 0.5$. The outcompete domain rule is assigned a medium confidence, $c(DR_3) = 0.75$, while the abductive domain rules are assigned a low confidence, $c(DR_1) = c(DR_2) = 0.25$.

Update of Confidences in Working Memory

To reduce the computations of finding hypotheses, a confidence threshold is set such that any inferred statement below that confidence threshold will not be a candidate for discovery hypothesis. However, this leads to a problem: When there is more than one inference path to an inferred statement, the statement may first be disregarded due to low confidence (computed using Eq.4.1) but it may later get a confidence above the threshold through another inference path. When an inferred statement has sufficient confidence, it may be used as an antecedent to infer further knowledge.Therefore, disregarded hypotheses are still kept "in mind", in the working memory. When a discovery round has finished, the confidence value of each statement in the WM is updated to the maximum confidence value it can be obtained through the best inference path that leads to it. At the same time, each statement is attached to this best inference path that maximizes its confidence.

Algorithm 2 shows how confidence of statements are propagated in WM.

1. For example, the following is an instantiation of IR4, *specialization of cause* rule:

$$\begin{array}{l} \text{X: algae IS-A primary producer} \\ \underline{\text{Y: } \uparrow \text{primary producer} \implies \uparrow \text{productivity}} \\ \therefore \text{I: } \uparrow \text{algae} \implies \uparrow \text{productivity} \end{array}$$

Here X and Y is background knowledge, giving $c(X) = c(Y) = 0.98$ and $c(IR_4) = 0.95$. This yields $c(I) = 0.98 \times 0.98 \times 0.95 = 0.932$.

When other statements have been used to infer a new one, the confidence of the inferred statement is recursively defined based on the statements it was inferred from (following from Equation 4.1). Assume that an statement is inferred a second time using a second inference path and obtained a better confidence than before. Assume also that this statement was further used as antecedent to infer another statement. When the confidence value of the first statement increases, it propagates to the second one and consequently the latter also receives a better

Algorithm 2 Update confidence of statements in WM

```
procedure CALCULATECONFIDENCE($WM$)
    for all statement $S_i$ in $WM$ do
        $c_i \Leftarrow$ CALCULATECONFIDENCE($S_i$)
    end for
end procedure
function CALCULATECONFIDENCE($S$)
    if ISINPUTKNOWLEDGE($S$) then
        $c \Leftarrow c(S)$
    else
        $c \Leftarrow 0$
        for all rule $IR_i$ in inferences that led to $S$ do
            $c_i \Leftarrow c_i(IR_i)$
            for all statement $S_j$ in ANTECEDENTS($IR_i$) do
                $c_i \Leftarrow c_i \times$ CALCULATECONFIDENCE($S_j$)
            end for
            $c \Leftarrow$ MAX($c, c_i$)
        end for
    end if
    return $c$
end function
```
2

confidence score than it previously had. Each statement prefers the inference path that gives it the maximum confidence and re-routes its inference path accordingly.

For example, consider the inference path in Figure 4.5 where each newly inferred statement has a 0.1 lower confidence than the preceding statement on its inference path. For simplicity, each inference (IR) have only one antecedent in the example. Assume that the confidence threshold is 0.5. At first, H_2 is inferred from H_1 where they have confidence values 0.6 and 0.5 respectively. When H_3 is inferred from H_2 with 0.4 confidence it stays under confidence threshold. Black arrows show the statements above the threshold. Assume that later, H_4 yields H_2 through another inference rule, where H_2 ultimately obtains 0.7 confidence. The new confidence score propagates to H_3 which follows H_2 on the inference path and H_3, with 0.6 now, comes over the confidence threshold. H_2 reroutes its inference to H_4 instead of having H_4. Let further running of the system yield H_5 from H_3. Then a background statement, S_{14}, is used to infer H_3, which gives higher confidence (0.6) to H_3. Then H_3 reroutes its inference path to S_{14} instead of H_2.

One potential issue is the case of H_4 depending on H_2. In that case, H_4 is not appropriate to

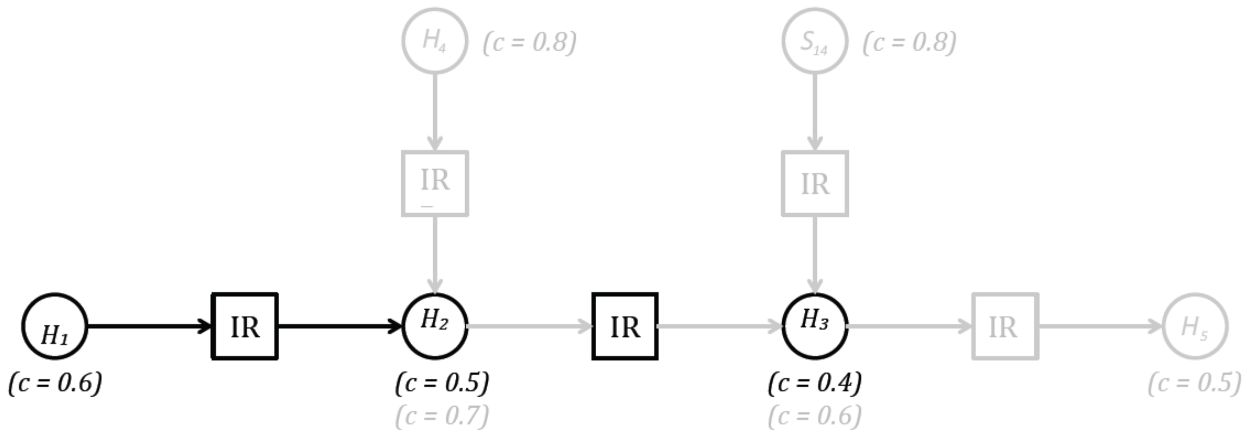


Figure 4.5: Multiple paths to a statement may give it different confidence scores. Arrows leading to hypotheses represents hypothesis candidates.

explain H_2. This does not occur in the system however, as a hypothesis can not have a stronger confidence than those it is dependent on.

4.4.3 Contradiction detection and abduction

When it is impossible for two statements to hold true at the same time, they contradict each others. The system operate with contradictions between causal relations, or observations. On predicate form, they are written $Contradiction(\uparrow X \implies \downarrow Y, \uparrow X \implies \uparrow Y)$, $Contradiction(Observation(\uparrow X, \downarrow X))$. These contradictions may represent inconsistent findings in literature, or simply statements made in different contexts. In the same context, they may indicate *feedback loops.* It is left to the user of the system to judge what contradictions between causal relations mean, while the system itself will try to explain contradictions between observed events.

After new knowledge is generated, the system enters the contradiction detection/abduction phase, where it compares each causal relation each of the other causal relations, and each event with each of the other events, and generate contradiction statements for those found to be in contradiction. When two events are in contradiction, the system will also assert $MustExplain$ predicates into the knowledge base for events in the contradiction, which doesn't currently have an explanation in the system. These are the user *inputted* event observations - *expected* observations are already explained by their inference path.

With the $MustExplain$ predicates in the knowledge base, abductive inferences that depend on the $MustExplain$ predicates such as the *eats* and *produces* rule may be able to fire. The explanation phase will continue as long as possible, and it's up to the user to decide whether the explanation is sufficient to explain the observation, and what statement in the contradiction's inference path may be potentially wrong.

4.4.4 Ranking of hypotheses

It is easy to generate a lot of hypothesis candidates, a bigger challenge is to rank the candidates generated by the application, so that the most interesting ones are shown first Wren et al. (2004). We define $interestingness(H)$ in terms of both the confidence, $c(H)$, and a *significance* rating, $s(H)$:

$$Interestingness(H) = c(H) \times s(H) \tag{4.2}$$

The *significance* measure is used to rate how interesting a hypothesis is, in the case that it is indeed true. Significance combines several metrics, such as how many inferences was necessary to generate a hypothesis and how isolated the knowledge was required to perform the inferences. The idea is that with a high significance score, a human would be less likely to have considered the hypothesis. If a the user has an interest in particular variables, that is clue to rate candidates containing those variables higher. The significance ranking of a hypothesis is thus split into $s_{inferred}$, $s_{sources}$ and s_{query}. These are explained separately in the following section. Then, the method of combining these scores to the final significance score is shown in a Algorithm 3.

Along inference paths, significance ratings develop in the opposite direction of confidence ratings, because a hypothesis with little confidence, would be interesting if proven true. Thus confidence decreases with longer inference chains while significance increases. Like confidences, the inference significance $s_{inferred}$ rating is given to a hypothesis both based on an inference's antecedents and the inference type. For inference $IR(X,Y) \Rightarrow H$: $s_{inferred}(H) = s(X) \times s(Y) \times s(IR)$. Significances used for the rules were chosen so that the system goal findings (causal transitivity and contradictions) are rated the highest, abductively generated explanations also have a high significance, while the generalization and specialization inferences are considered more trivial. The values used are $s(IR_1) = 5$, $s(IR_2) = s(IR_5) = 1.5$, $s(IR_3) = s(IR_4) = 1.1$. $s(DR_1) = s(DR_2) = 3$, s(DR_3) = 1.25. Detected contradictions also have a significance rating of 5.

Literature sources can be rated significant based on how many of them supports the same hypothesis, and whether sources connected through inferences were isolated (e.g. from different domains, or have a long path to each other in a citation network). It is however a complicated task to determine how independent different sources truly are from one another. In the current system, the source significance $s_{sources}$ was simply set to an exponentially growing value based on the number of sources appearing during the in a hypothesis' inference path: $s_{sources} = SourceCountSignificance^{sourceCount}$, where $SourceCountSignificance$ was set to 2. Note that currently the system rates the sources involved in each inference path separately.

By using a query of expressions from the user, such as ["iron","PCO2"], all hypotheses containing the query variables "iron" or "PCO2" can be rated as more significant. The resulting s_{query} depends on how many times these variables appear in a hypothesis or it's inference path. By querying for either one or two variables, s_{query} allows for both open and closed discovery browsing based on variables.

To avoid the accumulation of significance score from sources and query significance, the final significance s calculation for each hypothesis is split into in several steps.

Algorithm 3 Calculate significances of statements in WM

```
procedure CalculateSignificance($WM$)
    for all Candidate hypothesis $H_i$ in $WM$ do
        $s_i \Leftarrow$ selfSignificance($H_i$)
    end for
    for all Statement $S_i$ do
        $s_i \Leftarrow s_i \times s_{i_{sources}}$
        $s_i \Leftarrow s_i \times s_{i_{query}}$
        $s_i \Leftarrow log(s_i)$
    end for
end procedure
function selfSignificance($S$)
    if $\neg$ isInputKnowledge($S$) then
        $s \Leftarrow 1$
    else
        $s \Leftarrow s_{inferred}(IR)$
        for all statement $S_j$ in antecedents($IR$) do
            $s \Leftarrow s \times$ selfSignificance($S_j$)
        end for
    end if
    return $s$
end function
```

Chapter 5

Implementation

The system was implemented in the Jess production rule system, using Java. The Neo4j database was also utilized.

5.1 Input knowledge

The input knowledge are Jess Facts assertions of variable, event, causal and generalization (subsumption) templates, while other predicates such as contradiction, observation and mustExplain also have Jess templates. For the purpose of easily asserting all the necessary Jess facts, the application have corresponding Java classes for each template, such that variables will only have to be declared once, and the Java object can be referenced when it's Jess fact is used in further declarations. When referring to events of a variable, one can call methods that will automatically assert these events, and return their Java object. To assert the causal event “iron increasing causes phytoplankton increasing”, one writes:

```
Variable iron = new Variable("iron");
Variable phytoplankton = new Variable("phytoplankton");
Causal iron_phytoplankton = new Causal(iron.increasingEvent(),
   phytoplankton.increasingEvent());
```

Phytoplankton is a primary producer:

```
Variable primaryProducer = new Variable("primary producer");
```

```
phytoplankton_producer = new Generalization(phytoplankton,primaryProducer)
    ;
```

5.2 Sources

Each input relation should have a SourceType. These types are literature, background knowledge or syntactical generalization. These source types have different assigned confidences, just like the inference types. We declare the source for Ryther and Kramer's paper (from section 6.1)with a URL location:

```
Source rytherKramerPaper = new Source(SourceType.LITERATURE,
  "Ryther and Kramer","Ryther_Kramer_1961.pdf");
```

And add it to our iron-phytoplankton relation (the second page):

```
iron_phytoplankton.setSource(rytherKramerPaper.getSourceAtPage(2)
     ,SourceType.LITERATURE);
```

We declare the phytoplankton primary producer relation as background knowledge:

```
phytoplankton_producer.setSource(SourceType.BACKGROUND_KNOWLEDGE);
```

Input classified as background knowledge have a high confidence rating, regardless of whether it has a source. The default source type, when no source has been given, is "unknown", and has a very low confidence rating.

5.3 Implementation of inference rules

The application contains a Jess rule corresponding to each of the domain independent and dependent rules. They operate on Jess templates for variables, events, causal relationships and generalizations.

5.3.1 Domain independent rules

```
(defrule causal−transitivity
  ?event1 <− (event (type ?t1)(variable ?v1))
  ?event2 <− (event (type ?t2)(variable ?v2))
  ?event3 <− (event (type ?t3)(variable ?v3))
  ?causal1 <− (causal (cause ?event1)(effect ?event2) )
  ?causal2 <− (causal (cause ?event2)(effect ?event3) )
  =>
  (?∗external_class∗ examineCausalConclusion
  ?event1 ?event3 ?causal1 ?causal2 (SourceType.DEDUCTION_ABC)))
```

The function examineCausalConclusion (called by an external Java class) will analyze and record the conclusion, before asserting as a Jess fact (if applicable):

```
(assert (causal (cause ?event1)(effect ?event3) ))
```

The assert statement returns the unique ID of the asserted fact, or false if the fact is already asserted. For each template, an external function assertTemplateIfNew was made, which return the ID of the asserted fact, regardless if it was already asserted or not, which is used instead.

The RHS of the remaining rules operate similarly with a examineCausalConclusion call, but for simplicity are presented only with a variable binding call to assert the new event instance and bind the fact ID to a variable, and a Jess assertion call for asserting the conclusion fact. Although this new "event3" could be asserted beforehand for every combination of variable with event type of either increasing/decreasing, as it in some cases will have been already, this is not done, to avoid unnecessary matching efforts.

For abductive reasoning, a causal relation or observation may be marked as *MustExplain* when a contradiction is detected. It can be written $MustExplain(Observation(\uparrow X))$ or $MustExplain(\uparrow X \implies \downarrow Y)$.

```
(defrule generalize−cause
  ?generalization <− (generalization (specific ?specific )(general ?
      general))
  ?event1 <− (event (type ?t)(variable ?specific))
  ?event2 <− (event (type ?t2)(variable ?v2))
  ?causal <- (causal (cause ?event1)(effect ?event2) )
  =>
   (bind ?event3 (?*external_class* assertEventIfNew ?t ?general))
(assert (causal (cause ?event3)(effect ?event2))) )



 (defrule generalize-effect
  ?generalization <- (generalization (specific ?specific )(general ?
     general))
  ?event1 <- (event (type ?t)(variable ?v1))
  ?event2 <- (event (type ?t2)(variable ?specific))
  ?causal <- (causal (cause ?event1)(effect ?event2) )
  =>
  (bind ?event3 (?*external_class* assertEventIfNew ?t2 ?general))
(assert (causal (cause ?event1)(effect ?event3))) )

 (defrule specialize-cause
  ?generalization <- (generalization (specific ?specific )(general ?
     general))
  ?event1 <- (event (type ?t)(variable ?general))
  ?event2 <- (event (type ?t2)(variable ?v2))
  ?causal <- (causal (cause ?event1)(effect ?event2) )
  =>
  (bind ?event3 (?*external_class* assertEventIfNew ?t ?specific))
(assert (causal (cause ?event3)(effect ?event2))) )

 (defrule specialize-effect
  ?generalization <- (generalization (specific ?specific )(general ?
     general))
  ?event1 <- (event (type ?t)(variable ?v1))
  ?event2 <- (event (type ?t2)(variable ?general))
  ?causal <- (causal (cause ?event1)(effect ?event2) )
  =>
  (bind ?event3 (?*external_class* assertEventIfNew ?t2 ?specific))
(assert (causal (cause ?event1)(effect ?event3))) )


 (defrule modus-ponens
  ?event1 <- (event (type ?t1)(variable ?v1))
  ?event2 <- (event (type ?t2)(variable ?v2))
  ?causal <- (causal (cause ?event1)(effect ?event2))
  ?observation <- (observation (event ?event1))
  =>
  (?*external_class* examineObservationConclusion ?event2
      ?causal ?observation (SourceType.MODUS_PONENS))
)
```

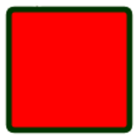
5.3.2 Domain dependent rules

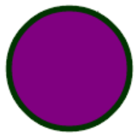
```
(defrule outcompete-rule
  ?observation1 <- (observation(event ?event1))
  ?observation2 <- (observation(event ?event2))
  ?event1 <- (event (type p-limitation)(variable ?v1))
  ?event2 <- (event (type p-limitation)(variable ?v2))
  ?outcompete <- (outcompete (winner ?v1)(loser ?v2))
  =>
  (bind ?event3 (?*external_class* assertEventIfNew decreasing ?v2))
  (?*external_class* examineObservationConclusion
      ?event3 ?observation1 ?observation2
          ?outcompete (SourceType.DOMAIN_RULE))
)
```

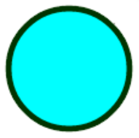
```
(defrule eats-rule
  (isExplanationMode (isExplanationMode TRUE))
  ?eat <- (eats (eater ?eater)(eaten ?eaten))
  ?mustExplainEvent <- (event (variable ?eaten)(type ?type))
  ?mustExplain <- (mustExplain (mustExplain ?mustExplainEvent
      (reason ?reason))
  =>
  (bind ?event1 (?*external_class* assertEventIfNew (?*external_class*
      getOppositeEventType ?type) ?eater))
  (bind ?causal (?*external_class* examineCausalConclusion ?event1
      ?mustExplainEvent ?eat ?reason (SourceType.ABDUCTION)))
  (?*external_class* assertMustExplainIfNew ?event1 ?causal)
)

(defrule produces-rule
  (isExplanationMode (isExplanationMode TRUE))
  ?produce <- (produces (producer ?producer)(produced ?produced))
  ?mustExplainEvent <- (event (variable ?produce)(type ?type))
  ?mustExplain <- (mustExplain (mustExplain ?mustExplainEvent)
      (reason ?reason))
  =>
  (bind ?event1 (?*external_class* assertEventIfNew ?type ?producer))
  (bind ?causal (?*external_class* examineCausalConclusion ?event1
      ?mustExplainEvent ?produce ?reason (SourceType.ABDUCTION)))
  (?*external_class* assertMustExplainIfNew ?event1 ?causal)
)
```

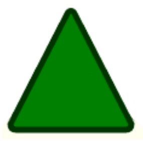
5.4 HTML browser view of system results

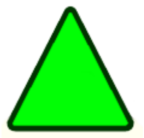
When the system has run, it will output it's hypotheses, along with the path of input knowledge and previous hypotheses required to reach it, referred to as the *inference path*. Such an inference


(a) Found in literature, a statement interpreted from a paper


(b) Background knowledge, drawn from some domain ontology


(c) Generalized by syntactical manipulation of input phrases


(d) Inferred by the rule engine using a domain independent rule.


(e) Inferred by the rule engine using a domain dependent rule.

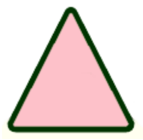

(f) Inferred by the rule engine, using abduction.

Figure 5.1: Node types in the inference tree

path forms a tree structure. Expressing all of the inference paths at the same time to the users may get cumbersome, the system has a HTML results view where each hypothesis is shown only with their most confident inference path. Each outputted hypothesis is numbered from 1 and upwards, beginning with the inputs to the rule engine. (Extracted rules, background knowledge, and syntactical generalizations.) A HTML/JavaScript document is produced, where each output hypothesis is presented with links to it's inference sources, such a literature text or another generated hypothesis. A graph tree is also produced for each hypothesis, with nodes shaped and color based on their sources. Figure 5.1 shows the different type of nodes in the inference tree. Input nodes are either square, representing a rule mined from the literature, or a circle. Circles are either purple, representing background knowledge that the system has, and which it asserts a high confidence. Blue circles represent syntactically generalized statements, and thus they are more prone to error. Triangles represents inferences, where orange triangles may be less likely to be plausible than green triangles. Figure 5.2 shows an inference tree produced by the system.

Figure 5.3 shows the full view of the HTML page, with both the inference path for a selected hypothesis, a top list of best ranked hypotheses, which is followed by a list of "findings", where each hypothesis has all the inferences that lead to them listed, along with potentially related

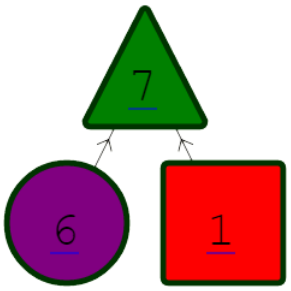
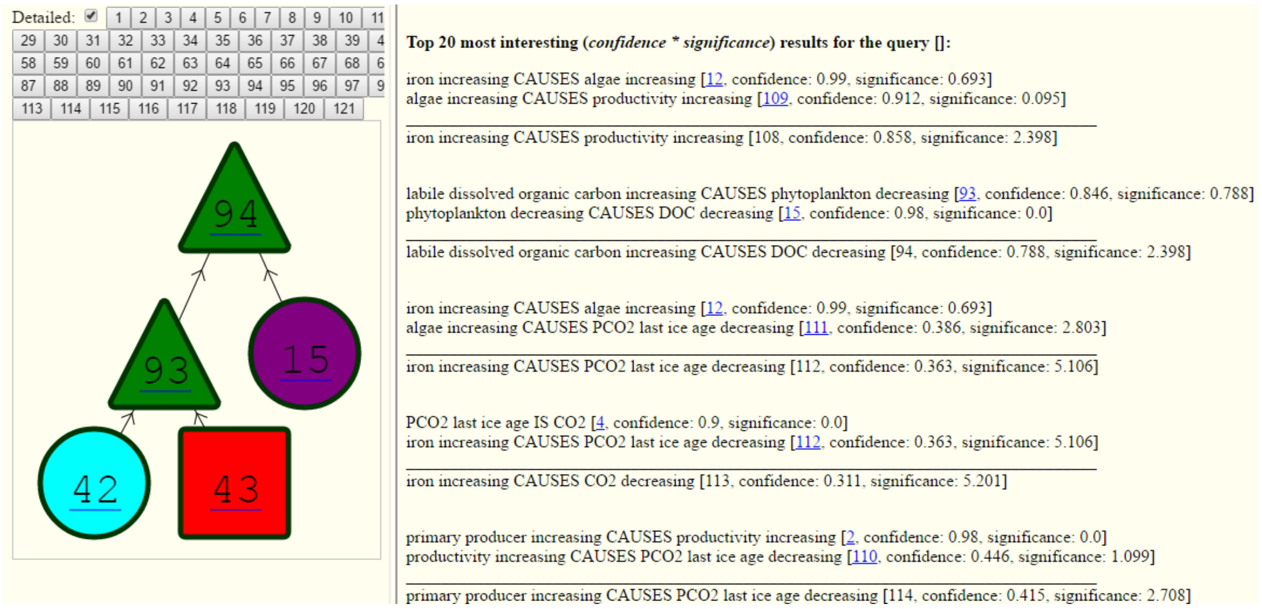


Figure 5.2: An inference tree created by the rule engine. Each node's corresponding output hypothesis statement is shown when hovering over it.

hypotheses. (Figure 5.4)



Top 20 most interesting (*confidence * significance*) results for the query []:

iron increasing CAUSES algae increasing [12, confidence: 0.99, significance: 0.693]
algae increasing CAUSES productivity increasing [109, confidence: 0.912, significance: 0.095]

---

iron increasing CAUSES productivity increasing [108, confidence: 0.858, significance: 2.398]

labile dissolved organic carbon increasing CAUSES phytoplankton decreasing [93, confidence: 0.846, significance: 0.788]
phytoplankton decreasing CAUSES DOC decreasing [15, confidence: 0.98, significance: 0.0]

---

labile dissolved organic carbon increasing CAUSES DOC decreasing [94, confidence: 0.788, significance: 2.398]

iron increasing CAUSES algae increasing [12, confidence: 0.99, significance: 0.693]
algae increasing CAUSES PCO2 last ice age decreasing [111, confidence: 0.386, significance: 2.803]

---

iron increasing CAUSES PCO2 last ice age decreasing [112, confidence: 0.363, significance: 5.106]

PCO2 last ice age IS CO2 [4, confidence: 0.9, significance: 0.0]
iron increasing CAUSES PCO2 last ice age decreasing [112, confidence: 0.363, significance: 5.106]

---

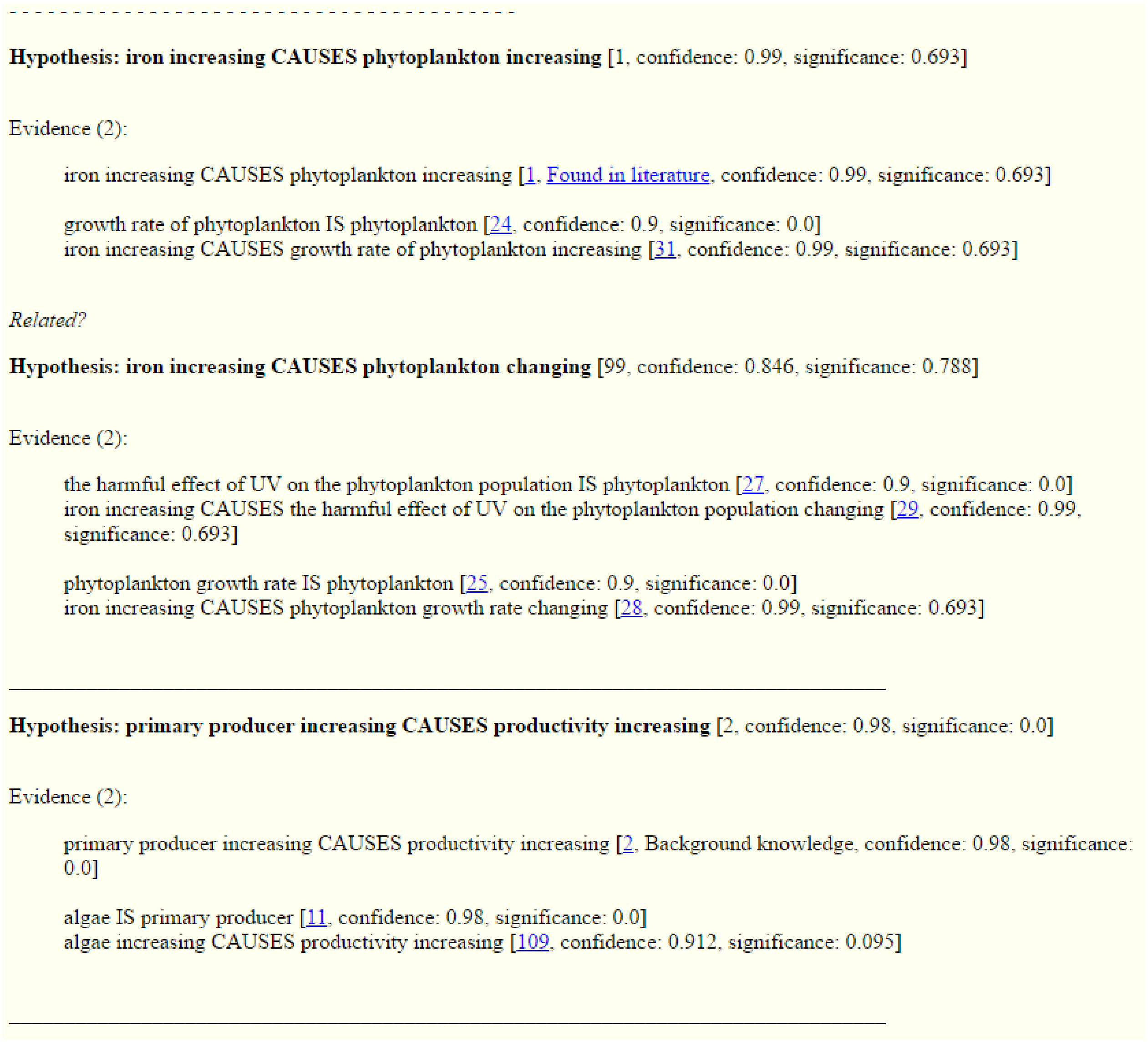
iron increasing CAUSES CO2 decreasing [113, confidence: 0.311, significance: 5.201]

primary producer increasing CAUSES productivity increasing [2, confidence: 0.98, significance: 0.0]
productivity increasing CAUSES PCO2 last ice age decreasing [110, confidence: 0.446, significance: 1.099]

---

primary producer increasing CAUSES PCO2 last ice age decreasing [114, confidence: 0.415, significance: 2.708]

Figure 5.3: The user a list of all hypotheses on the right, and click a number on the left to see the most confident inference path for each hypothesis.



- -

Hypothesis: iron increasing CAUSES phytoplankton increasing [1, confidence: 0.99, significance: 0.693]

Evidence (2):

iron increasing CAUSES phytoplankton increasing [1, Found in literature, confidence: 0.99, significance: 0.693]

growth rate of phytoplankton IS phytoplankton [24, confidence: 0.9, significance: 0.0]
iron increasing CAUSES growth rate of phytoplankton increasing [31, confidence: 0.99, significance: 0.693]

Related?

Hypothesis: iron increasing CAUSES phytoplankton changing [99, confidence: 0.846, significance: 0.788]

Evidence (2):

the harmful effect of UV on the phytoplankton population IS phytoplankton [27, confidence: 0.9, significance: 0.0]
iron increasing CAUSES the harmful effect of UV on the phytoplankton population changing [29, confidence: 0.99, significance: 0.693]

phytoplankton growth rate IS phytoplankton [25, confidence: 0.9, significance: 0.0]
iron increasing CAUSES phytoplankton growth rate changing [28, confidence: 0.99, significance: 0.693]

Hypothesis: primary producer increasing CAUSES productivity increasing [2, confidence: 0.98, significance: 0.0]

Evidence (2):

primary producer increasing CAUSES productivity increasing [2, Background knowledge, confidence: 0.98, significance: 0.0]

algae IS primary producer [11, confidence: 0.98, significance: 0.0]
algae increasing CAUSES productivity increasing [109, confidence: 0.912, significance: 0.095]

Figure 5.4: Each hypothesis is listed, along with the evidence that led to it. When the source is a literature source, the hyperlink leads to the literature source. Numbered links leads to the listing for the hypothesis of that ID number.

5.5 Neo4j graph view of system results

The system can write the results to a Neo4j database, allowing the user to view graphs retrieved with Cypher queries, shown below. Figure 5.5 shows a result view. The nodes are color coded, with input nodes being red for literature sources, purple for background knowledge, blue for inferred knowledge, and green for inference nodes. Yellow nodes are event or generalization nodes, while the remaining nodes (causal relations, variables) are gray.

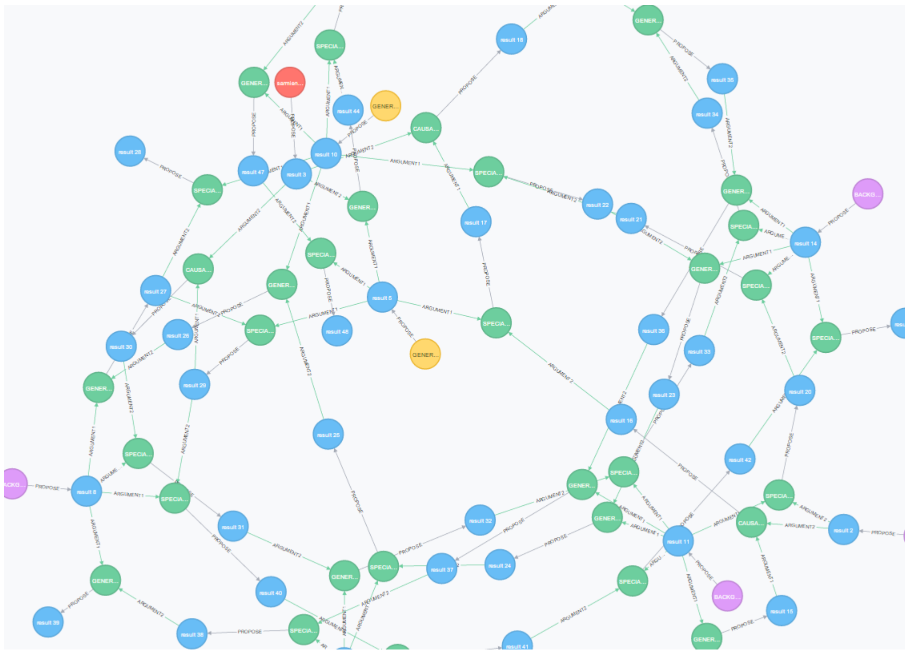


Figure 5.5: Neo4j view of the system's results.

```
MATCH n
RETURN n
```

Shows every node in the database. As seen in figure 5.6, the result can be very clustered and hard to comprehend.

```
MATCH b −[:EFFECT]−> a
MATCH c −[:CAUSE]−> a
MATCH b −[:THEME]−> d
MATCH c −[:THEME]−> e
RETURN a,b,c,d,e
```

Shows all causal relations.

```
MATCH a −[:THEME]−> b WHERE b.name = "iron"
MATCH c −[:THEME]−> d WHERE d.name = "CO2"
MATCH a −[:CAUSE]−> e
MATCH c −[:EFFECT]−> e
RETURN a,b,c,d,e
```

Shows a specific causal relation (in this example, the iron hypothesis).

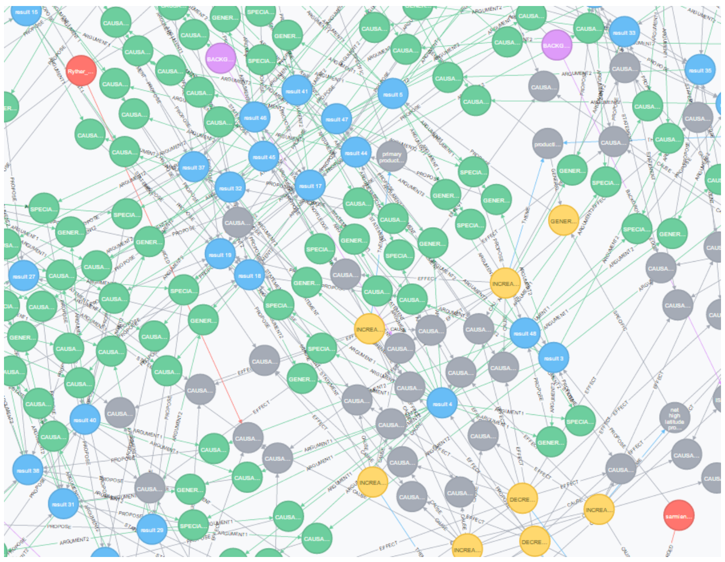


Figure 5.6: Result of "MATCH n return n".

```
MATCH (a:RESULT)
MATCH (b:INFERENCE)
MATCH (c:INPUT)
RETURN a,b,c
```

Shows only inference path nodes. (Not events/relations/variables).

Chapter 6

Use cases

As part of the development of the LBD system, domain experts were asked to give some example hypotheses that has been important in the domain, and that may have been found by an LBD system system if it had been available at the time of their conception. Currently, two hypotheses related to the food web and biological pump has been worked out to be expressed by the knowledge representation used by the system.

These use cases has helped demonstrate what sort of reasoning the inference system may utilize to construct hypotheses, and has served as the testing ground for the system.

6.1 Iron hypothesis

Proposed by Martin and Gordon (1988), and independently by Gribbin (1988), the iron hypothesis links together already long known knowledge in that phytoplankton, an important actor in removing CO_2 from the atmosphere through photosynthesis, is often limited by iron deficiency. This leads to the possibility that iron fertilization of the ocean may help mitigate global warming. While the hypothesis didn't appear until 1988, the link between iron and phytoplankton has been studied all the way back to 1933.

The hypothesis serves as a good test candidate for LBD, with the knowledge pieces being linkable through Swanson linking. It also fits well with the OCEAN-CERTAIN project, as it can be expressed though the causal relation between events of changing variables. It is also a good example of what open discovery could uncover, as there is a huge interest in reducing CO_2, but a lack of understanding of the complex mechanisms that contribute to changing CO_2 levels.

Marsi (2016) shows how the hypothesis can be extracted from two literature sources published before 1988, using the above discussed methods of syntactical generalization of variables, extraction of causal relations between change events, and by transferring these causal properties along variables connected with is-a relations, found in background knowledge. Following is his method, adapted to this thesis' inference system. Background knowledge is abbreviated B, literature sources are abbreviated L, inferred statements are abbreviated I, and statements that were syntactically generalized abbreviated as G.

The following statement, from Ryther and Kramer (1961) states the importance of iron for phytoplankton:

(1) *Gran (1933) was among the first to demonstrate that the addition of iron to seawater may stimulate the growth of phytoplankton.*

Using the method of extraction of change events, causal relations and syntactical generalization (see section 2.3), "addition" can serve as the trigger for an increase event pertaining to "iron". "Stimulate" signals a causal relation between two events, and "stimulate" servers as a trigger for an increase in the second variable, "phytoplankton". This could yield the relation

$$\mathbf{L1}: iron \uparrow \implies phytoplankton \uparrow$$

Now, assuming a background knowledge containing a subsumption relationship:

$$\mathbf{B1}: phytoplankton\ is-a\ primary\ producer$$

It can be inferred, using *generalization of effect* (IR3 from section 4.2):

$$\mathbf{I1}: iron \uparrow \implies primary\ producer \uparrow$$

Now assuming a background knowledge:

$$\mathbf{B2}: primaryproducer \uparrow \implies productivity \uparrow$$

Which with *causal transitivity* (IR1) yields:

$$\mathbf{I2}: iron\uparrow \implies productivity\uparrow$$

The second transitive link between iron and CO_2, the relation between productivity and CO_2 is also well known in the scientific literature. Marsi uses an example from Sarmiento and Toggweiler (1984):

(2) *A major contribution to the low PCO_2 of the last ice age may have been an increase in the net high latitude productivity, possibly coupled with a decrease in the thermohaline overturning.*

"PCO_2 of the last ice" is recognizable as a variable, in a decrease event with "low" as trigger word. The second variable "the net high latitude productivity" is recognized as increasing. "A major contribution to" suggests that the first event is triggered by the second. This can be represented as

$$\mathbf{L2}: the\ net\ high\ latitude\ productivity\uparrow \implies$$

$$PCO2\ of\ the\ last\ ice\ age\downarrow$$

Syntactical generalizations, stripping away the adjectives ("net" and "high latitude"), and prepositional modifiers, ("of the last ice age"), and ER of PCO2 as CO_2 yields

$$\mathbf{G1}:\ the\ net\ high\ latitude\ productivity\ is-a\ productivity$$

$$\mathbf{G2}:\ PCO2\ of\ the\ last\ ice\ age\ is-a\ CO_2$$

now using generalization of effect and generalization of cause, (IR2 and IR3) yields

$$\mathbf{I4}: productivity\uparrow \implies PCO2\downarrow$$

Thus allowing the linking of iron and PCO_2 through I2 and G1 by causal transitivity (IR1):

$$\mathbf{I5}: iron\uparrow \implies PCO2\downarrow$$

Figure 6.1 shows each inference step. Note that the syntactical generalization is done in two steps.

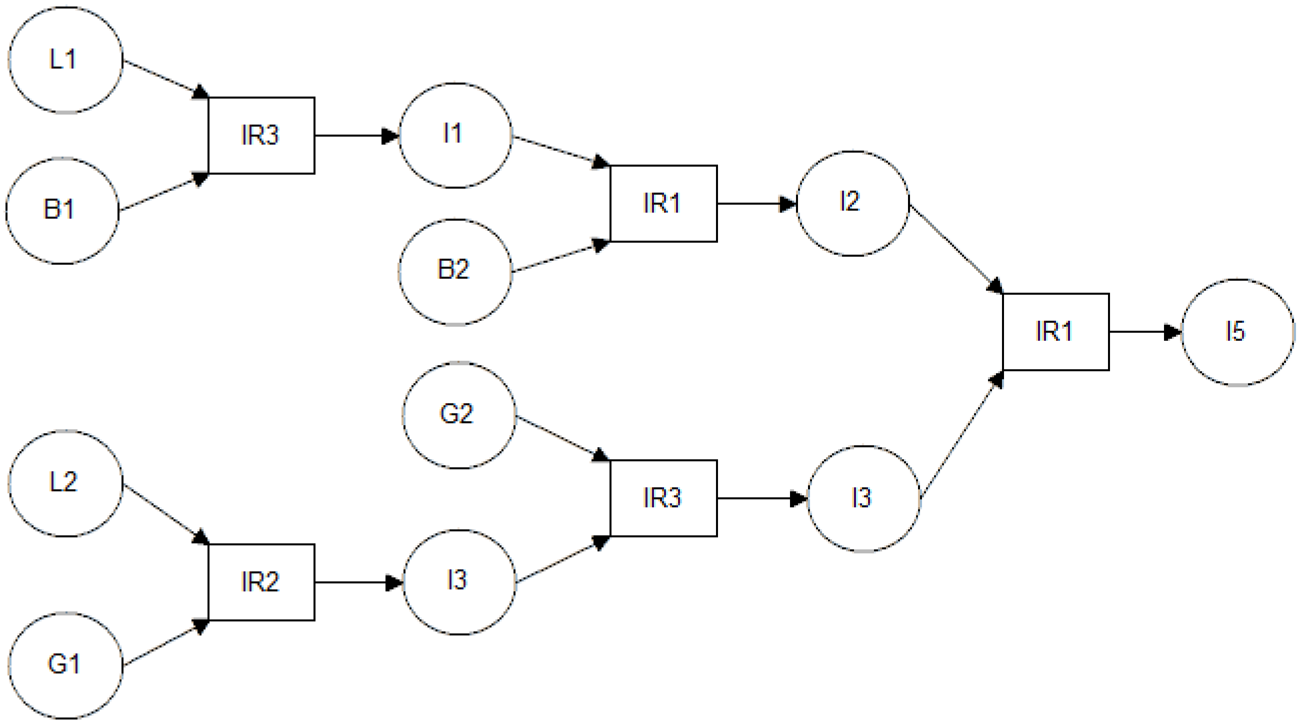


Figure 6.1: Inference steps for the iron hypothesis.

6.1.1 Additional literature statements for inferring the iron hypothesis

An attempt was made to locate more causal relations of change events to use for inferring the iron hypothesis in papers before 1988. While there was no direct such causal relations, a similar expression, involving *limitation*, is often used to express a similar relation. When a biological species or chemical entity x is limited by a factor y, that factor is what's most required to create more of x. Such a limitation is usually discovered by observing that adding the limiting y to x causes x to increase. In other words, statements of the form "y LIMITS x" can plausibly be converted to the form $\uparrow y \implies \uparrow x$. Four such statements were found in Ryther and Kramer

(1961) and converted to a causal form.

(3) *Since then many workers have proposed that iron may, at times, limit plant growth in the sea.*

This could yield

$$\mathbf{L3}: iron\uparrow \implies plant\uparrow$$

The subsumption relation

$$\mathbf{B3}: \; phytoplankton\; is-a\; plant$$

would allow specialization of cause to yield literature statement L1, but with less confidence compared to directly extracting L1.

(4) *Recent experiments by Menzel and Ryther (1961) have shown rather conclusively that iron is the most critical nutrient limiting primary production in the tropical and semi-tropical Western Atlantic.*

This statement on the other hand, cuts to the chase and link iron directly with primary production:

$$\mathbf{L4}: iron\uparrow \implies \; primary\; production\uparrow$$

(5) *After one or 2 transfers through this medium, all the algae with one exception were obviously iron-limited and had ceased growing.*

This could yield

$$\mathbf{L5}: iron\uparrow \implies \; algae\uparrow$$

Similarly to before, this can lead to L1 through a subsumption

$$\mathbf{B4}: phytoplankton\; is-a\; algae$$

(6) *Maximum populations of Isochrysis, Skeletonenia and Pyramimonas were attained at the highest level of iron used, at which point growth may still have been iron limited.*

This statement mentions several species of algae conjunctively with "and". This could yield

$$\mathbf{L6}: iron\uparrow \implies [Isochrysis\uparrow \wedge Skeletonenia\uparrow \wedge Pyramimonas\uparrow]$$

Which can be split into the corresponding causal relations for each variable in the conjunction. These are all algae species, and generalization of effect would, along with B4, yield L1.

6.2 Dissolved organic carbon (DOC) hypothesis

Phosphorus (P) is another important mineral nutrient. Thingstad and Rassoulzadegan (1995) looked at the situation in the Mediterranean Sea, where evidence suggests that P is a limiting factor of both phytoplankton and bacteria. As known from background knowledge, in a competition for phosphate, bacteria outcompetes phytoplankton. Figure 6.2 shows a simplified domain model.

As phytoplankton is an important producer of DOC, and bacteria is a consumer of DOC, one may expect that DOC in the sea would be reduced. Observations on the other hand, indicate that DOC is accumulating. Frede et al. suggested that this anomaly may be explained by heterotrophic microzooplankton, known for short as "grazers", who feed on bacteria. Stated in the notion of change events, the relation between grazers and DOC (the DOC hypothesis) is thus

$$Grazers\uparrow \implies DOC\uparrow$$

While the full explanation of the anomaly may be more complicated due to the additional interactions in the full food web, the DOC hypothesis serves as an entry point for further investigation into the issue. We now look at how the DOC hypothesis may be inferred from domain

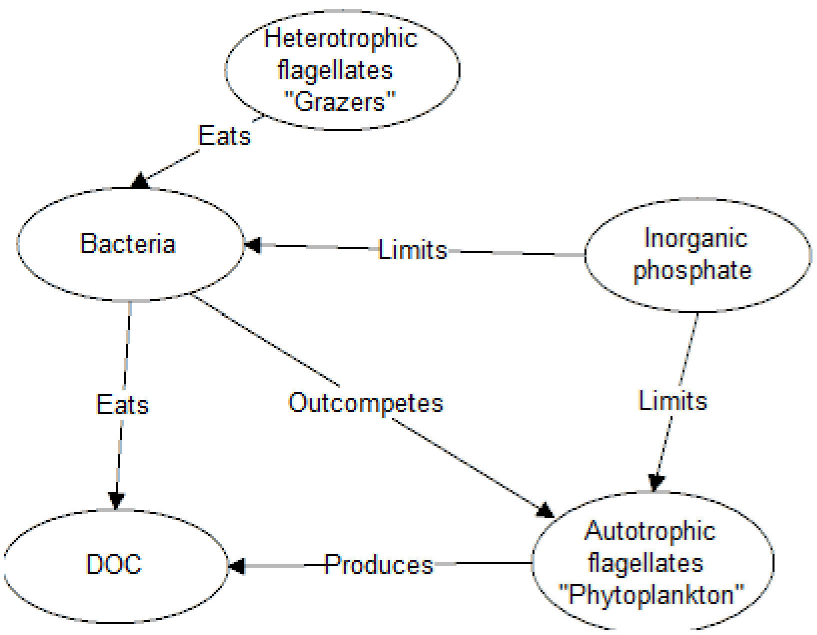

Heterotrophic flagellates "Grazers"
Eats
Bacteria
Limits
Inorganic phosphate
Eats
Outcompetes
Limits
DOC
Produces
Autotrophic flagellates "Phytoplankton"

Figure 6.2: DOC hypothesis ontology. Flagellates are small organisms with whip-like organelles which allows for swimming. Autotrophs produce their own energy through photosynthesis, while heterotroph rely on feeding on others.

background and literature knowledge.

This use case is explained from the inference engine's perspective, and each knowledge statement is named as simply their knowledge ID when entering the system, such as S1, S2, S3, ... Internal statements to the system, are labeled S1', S2', S3', ... To begin, we have in the knowledge base the following variables

Variables: $P, phytoplankton, bacteria, DOC, grazers$

And background knowledge:

S1 $\downarrow phytoplankton \implies \downarrow DOC$

S2 $Eats(bacteria, DOC)$

S3 $Eats(grazers, bacteria)$

S4 $Produces(phytoplankton, DOC)$

S5 $Outcompetes(bacteria, phytoplankton)$

And literature knowledge:

S6 $Observation(\uparrow DOC)$

S7 $Observation(event(bacteria, p\_limit))$

S8 $Observation(event(phytoplankton, p\_limit))$

Considering the limitation of P for phytoplankton and bacteria ($S7$,$S8$) and that bacteria outcompetes phytoplankton ($S5$), we infer a new expected observation using the *outcompete rule* (DR3):

$$\begin{array}{l} S5\ Outcompetes(bacteria, phytoplankton) \\ S7\ Observation(event(bacteria, p\_limit)) \\ \underline{S8\ Observation(event(phytoplankton, p\_limit))} \\ \therefore S9\ ExpectedObservation(\downarrow phytoplankton) \end{array}$$

The newly inferred $S9$, together with $S1$ and modus ponens (IR6) leads to:

$$\begin{array}{l} S1\ \downarrow phytoplankton \implies \downarrow DOC \\ \underline{S9\ ExpectedObservation(\downarrow phytoplankton)} \\ \therefore S10\ ExpectedObservation(\downarrow DOC) \end{array}$$

Contradiction is detected between the expected an actual observation of DOC. With the contradiction occurring, the system decides that $S6$ must be explained, as it contradicts what the system has inferred from it's knowledge. ($S1'$). The expected observation $S10$ must however not be explained, as it is already explained by the inference path the system used to infer it.

$$\begin{array}{l} S6\ Observation(\uparrow DOC) \\ \underline{S10\ ExpectedObservation(\downarrow DOC)} \\ \therefore S11\ Contradiction(S6, S10) \\ S1'\ MustExplain(\uparrow DOC) \end{array}$$

Now, during the abduction phase of the system, the eats-rule (DR1 from section 4.2.2) can be applied:

$$\begin{array}{ll} & S1' \; MustExplain(\uparrow DOC) \\ & \underline{S2 \; Eats(bacteria, DOC)} \\ \therefore & S12 \; \downarrow bacteria \implies \uparrow DOC \\ & S2' \; MustExplain(\downarrow bacteria) \end{array}$$

DR1 can be applied again for grazers and bacteria:

$$\begin{array}{ll} & S2' \; MustExplain(\downarrow bacteria) \\ & S12 \; \downarrow bacteria \implies \uparrow DOC \\ & \underline{S3 \; Eats(grazers, bacteria)} \\ \therefore & S13 \; \uparrow grazers \implies \downarrow bacteria \\ & S3' \; MustExplain(\downarrow bacteria) \end{array}$$

Finally, causal transitivity (IR1) connects grazers with DOC. This rule does not require $S3'$.

$$\begin{array}{ll} & S13 \; \uparrow grazers \implies \downarrow bacteria \\ & \underline{S12 \; \downarrow bacteria \implies \uparrow DOC} \\ \therefore & S14 \; \uparrow grazers \implies \uparrow DOC \end{array}$$

6.3 Annotated corpus

In addition to the above domain hypotheses, an experiment was done with some literature sources not intended for any particular discovery. The goal of this experiment was not to see any particular hypothesis proven, but to see what unknown connection the system could potentially make to either aid in relation to the iron/DOC hypotheses, or some discovery that was otherwise interesting.

Marsi et al. (2014) developed an annotation scheme for the OCEAN-CERTAIN project. Using this scheme, they performed manual annotation on a selection of 12 nature abstracts from the domain literature, chosen by domain experts. The *brat* rapid annotation tool was used, and the annotations stored in .brat files. A .brat file contains an entry for each entity and relation of the annotated file, marking it's entity/relation type, location span in the text, and giving it a unique

ID. Figure 6.3 show an example of an annotated text, where two increase event together cause a decrease event.

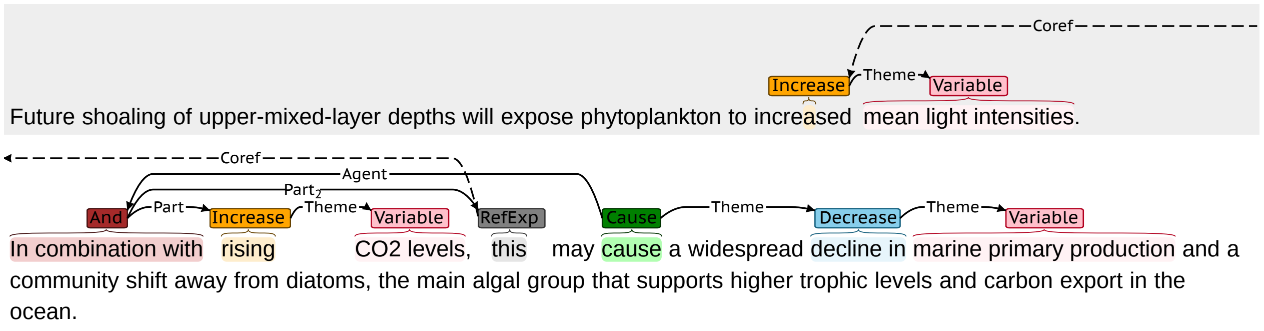

Coref
Increase
Theme
Variable
Future shoaling of upper-mixed-layer depths will expose phytoplankton to increased mean light intensities.
Coref
Agent
Part2
Part
Theme
And
Increase
Variable
RefExp
Cause
Theme
Decrease
Theme
Variable
In combination with rising CO2 levels, this may cause a widespread decline in marine primary production and a community shift away from diatoms, the main algal group that supports higher trophic levels and carbon export in the ocean.

Figure 6.3: Example of causal relation extracted from a pair of sentences.

A simple parser was written in Java to read all the all the .brat files, and enter the entities and relations into the inference system. A simple entity recognition procedure, based on a list of synonyms for known entities, such as (iron,Fe(II),Fe(III)) was used to identify entities, and syntactical generalization simply stripped away all other words from a variable containing a known entity. For causal or correlation relationships containing AND, such as $\uparrow X \implies [\uparrow Y \wedge \uparrow Z]$, the conjunction was split into two expressions, e.g. $\uparrow X \implies \uparrow Y$ and $\uparrow X \implies \uparrow Z$. Other more complicated (containing OR/NOT/FEEDBACK) were ignored. The resulting input to the system was 21 causal relations and 2 correlation relations. Table 6.1 shows the resulting input to the inference system. Note that the statement "organic carbon $\uparrow \implies$ organic carbon $\downarrow$" is filtered out, as the system doesn't intend to explain contradicting statements that were not discovered by the system.

Relation
iron $\uparrow \implies$ phytoplankton growth rate $\updownarrow$
iron $\uparrow \implies$ the harmful effect of UV on the phytoplankton population $\updownarrow$
iron $\uparrow \implies$ Fe(II) concentration $\uparrow$
iron $\uparrow \implies$ growth rate of phytoplankton $\uparrow$
atmospheric CO_2 levels $\uparrow \implies$ surface ocean pH $\updownarrow$
calcification $\downarrow \implies$ the ratio of calcite precipitation to organic matter production $\downarrow$
atmospheric CO_2 concentrations $\uparrow \implies$ production of C carbonate in the surface ocean $\downarrow$
labile dissolved organic C $\uparrow \implies$ phytoplankton biomass and activity $\downarrow$
labile dissolved organic C $\uparrow \implies$ rate at which total organic carbon accumulated $\downarrow$
organic carbon $\uparrow \implies$ organic carbon $\downarrow$
CO_2 levels $\uparrow \implies$ marine primary production $\downarrow$
CO_2 levels in oceanic surface waters $\uparrow \implies$ marine primary production $\uparrow$
some essential metals $\downarrow \implies$ concentrations in surface seawater $\downarrow$
CO_2 partial pressures $\uparrow \implies$ net primary production in coccolithophore species (...) $\uparrow$
CO2 partial pressures $\uparrow \implies$ calcification $\uparrow$
atmospheric CO2 $\uparrow \implies$ the spread of suboxic regions in the ocean $\uparrow$
N recycling $\downarrow \implies$ export of accumulated organic matter $\uparrow$
N recycling in nutrient-poor oligotrophic environments $\uparrow \implies$ C export/unit limiting nutrient $\uparrow$
stoichiometry of exported organic matter $\updownarrow \implies$ modelled POC export $\updownarrow$
atmospheric CO_2 $\uparrow \implies$ carbon $\downarrow$
CO_2 $\uparrow \implies$ cellular particulate inorganic carbon $\uparrow$
Fe(II) concentration $\uparrow \leadsto$ growth rate of phytoplankton $\uparrow$
coccospheres $\uparrow \leadsto$ coccoliths $\uparrow$

Table 6.1: Statements extracted from the literature constituting the .brat annotation files. Before application of syntactical generalization and ER. Some expressions were shortened to fit page width.

Chapter 7

Experiments, results and discussion

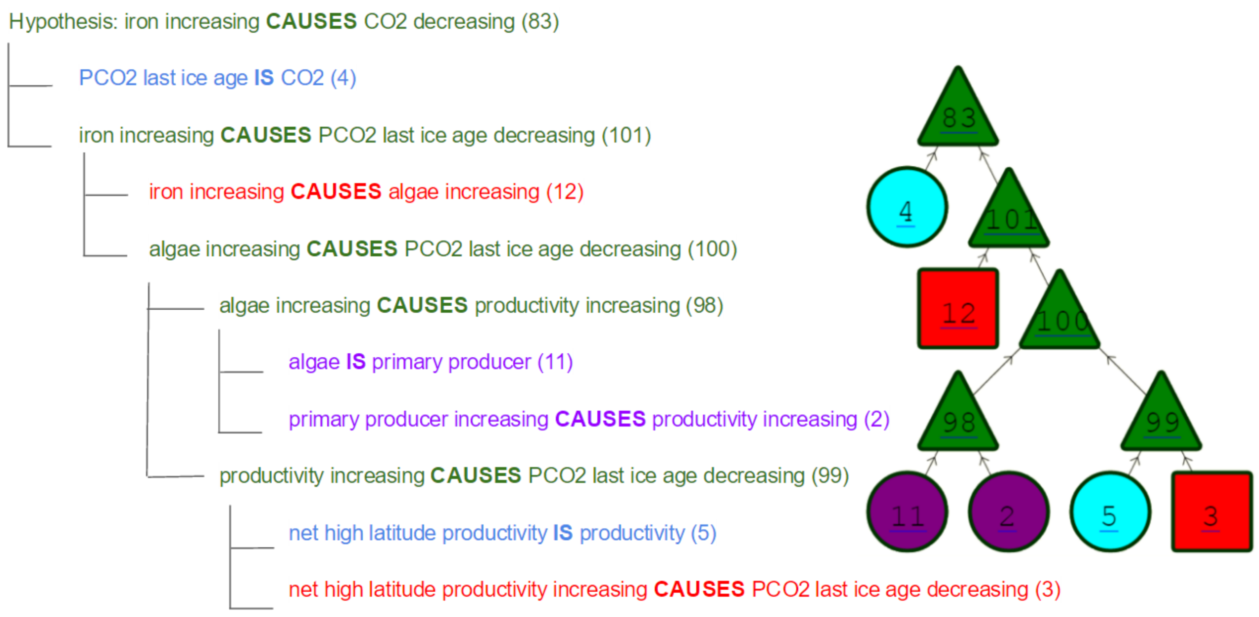
Experiments were done for each use case to see if the system was able to infer the targeted hypothesis, or in the case of the annotated corpus, infer something unplanned of interest. First, One separate experiment was conducted for each use case, using only the knowledge related to that use case as the input to the system. Then, an experiment was executed where inputs from all use cases is given to the system, to collect a top list of best ranked hypotheses. For the experiments, each statement is given a predefined confidence value based on it's type and source. These values were chosen to represent a somewhat plausible average confidence one may have that each type of knowledge is correct compared to each others, e.g. background/literature knowledge has more confidence than syntactically generalized/ER statements, as some error may have occured during generalization. The values used were $c(backgroundKnowledge) = c(literature) = 0.98$, $c(syntacticallyGeneralized) = 0.9$. In a real usage scenario of the system however, there may be different confidence values associated with different knowledge sources, while the confidence of syntactically generalized subsumption relations may be related to error rates in the text parser and generalization scheme.

7.1 Iron hypothesis

Figure 7.1 shows the most confident inference path inferred for the iron hypothesis. Each statement is assigned an ID number by the system, shown on it's node in the graph. The path begins with leaf nodes, which are the input knowledge to the system, progressing through inferences (triangles) to the root hypothesis, with the text using brackets and indents to show the antecedent leading to each inferred statement.

The system chose the literature statement relating iron with algae (L5) as the most confident inference path, however several paths tied in score. The ordering of inferences also differed a bit from the one used one thought out beforehand, applying one of the syntactical generalization inferences last. Otherwise, the result is much as planned.

Figure 7.2 shows a Neo4j view of the inference paths, showing that indeed several paths exists. The path from 7.1 is marked with red, and the iron hypothesis node (ID 83) is encircled.



Hypothesis: iron increasing **CAUSES** CO2 decreasing (83)

- PCO2 last ice age **IS** CO2 (4)
- iron increasing **CAUSES** PCO2 last ice age decreasing (101)
  - iron increasing **CAUSES** algae increasing (12)
  - algae increasing **CAUSES** PCO2 last ice age decreasing (100)
    - algae increasing **CAUSES** productivity increasing (98)
      - algae **IS** primary producer (11)
      - primary producer increasing **CAUSES** productivity increasing (2)
    - productivity increasing **CAUSES** PCO2 last ice age decreasing (99)
      - net high latitude productivity **IS** productivity (5)
      - net high latitude productivity increasing **CAUSES** PCO2 last ice age decreasing (3)

Figure 7.1: Inference path of the iron hypothesis. Each text statement corresponds to a node in the graph, and the brackets corresponds to arrows. (Read from bottom to top.)

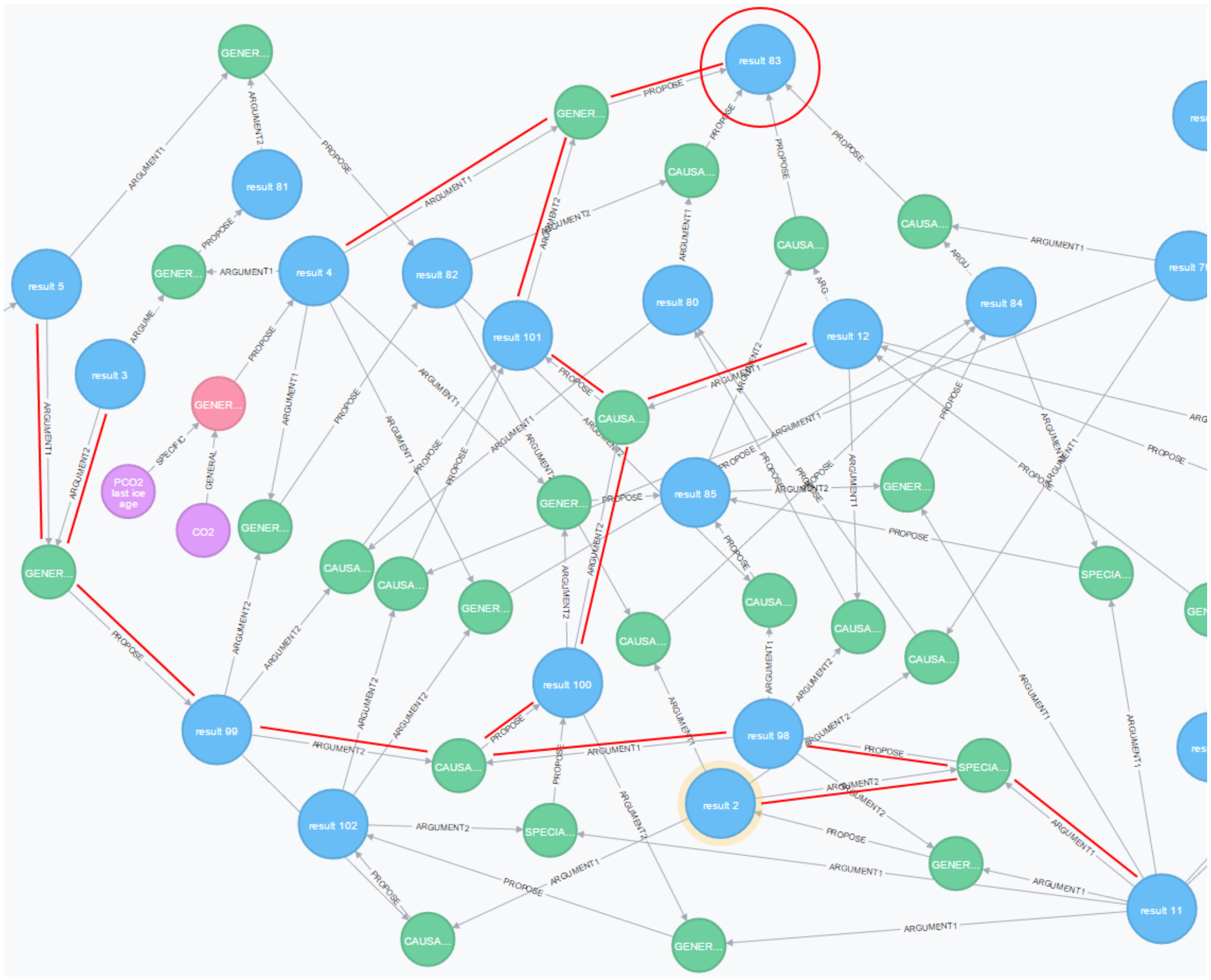


Figure 7.2: Neo4j view of all inference paths found for the iron hypothesis.

7.2 DOC hypothesis

Figure 7.3 shows the most confident inference path inferred for the DOC hypothesis, equivalent to figure 7.1 for the iron hypothesis. Here the result is exactly what as intended, and the IDs for the graph nodes correspond to those used in section 6.2. Note that events here are marked with "observation" versus "expected observation", due to the importance of this when the system chooses which side of the contradiction to attempt explain.

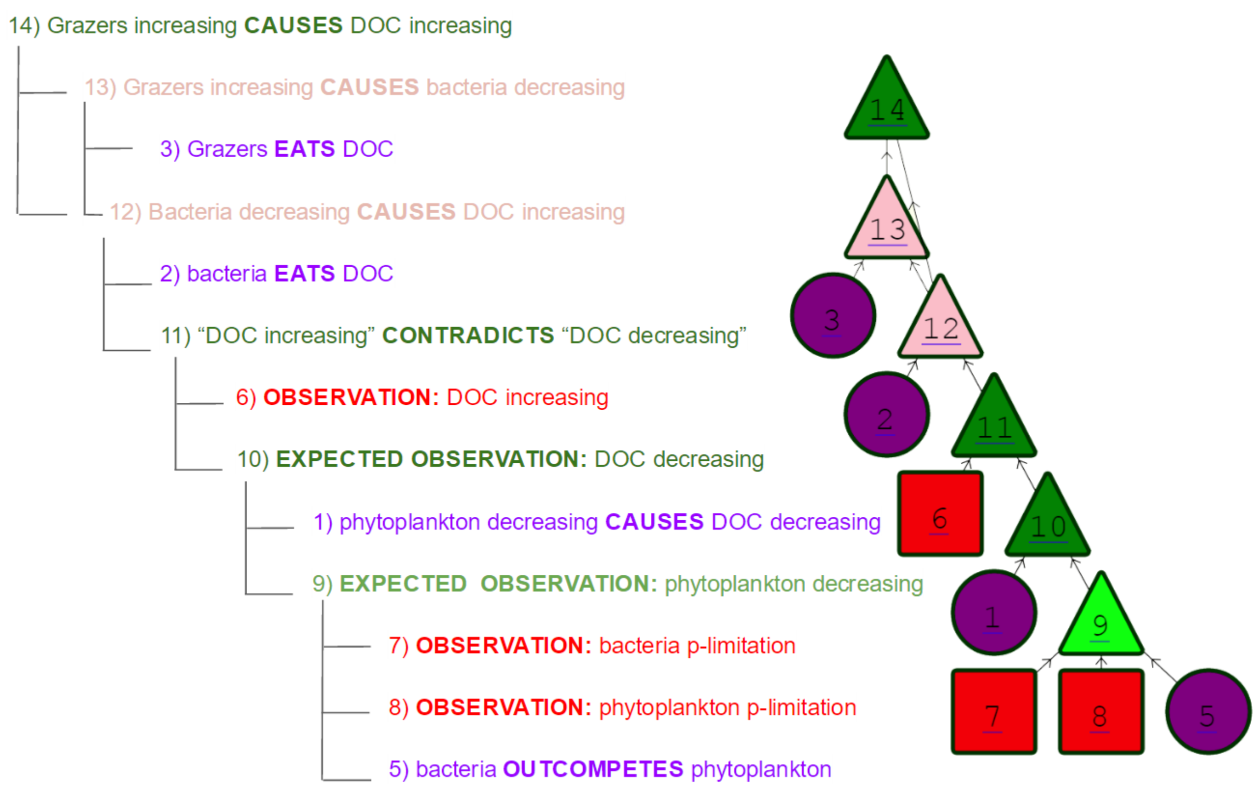

14) Grazers increasing CAUSES DOC increasing
13) Grazers increasing CAUSES bacteria decreasing
3) Grazers EATS DOC
12) Bacteria decreasing CAUSES DOC increasing
2) bacteria EATS DOC
11) "DOC increasing" CONTRADICTS "DOC decreasing"
6) OBSERVATION: DOC increasing
10) EXPECTED OBSERVATION: DOC decreasing
1) phytoplankton decreasing CAUSES DOC decreasing
9) EXPECTED OBSERVATION: phytoplankton decreasing
7) OBSERVATION: bacteria p-limitation
8) OBSERVATION: phytoplankton p-limitation
5) bacteria OUTCOMPETES phytoplankton
14
13
3
12
2
11
6
10
1
9
7
8
5

Figure 7.3: Inference path of the DOC hypothesis. The path shown in figure 7.1 is marked with red lines.

7.3 Annotated corpus

As the annotated corpus was very small, but still an attempt was made in order to see if the system would be able to infer anything from that knowledge. One noteworthy thing spotted by the system was the contradiction between two opposing views in the domain, whether more CO_2 causes an increase or decrease in marine primary production. Figure 7.4 shows the system's display of this contradiction, along with evidence for each side. Following the links in the evidence, we see that the first evidence, in support of $\uparrow CO_2 \implies \uparrow$ marine primary production, leads back to an abstract stating:

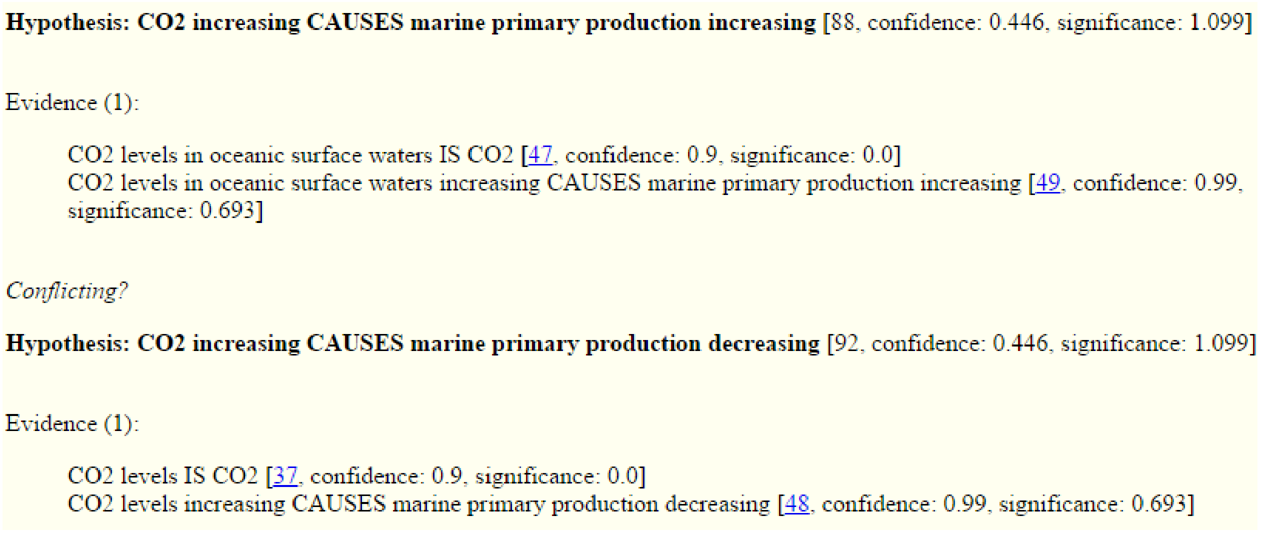
"Rising CO2 levels in oceanic surface waters in combination with ample light supply are therefore often considered stimulatory to marine primary production."

Following the links from the evidence to the contrary, $\uparrow CO_2 \implies \downarrow$ marine primary production, leads to the same abstract, which later states:

"Here we show that the combination of an increase in both CO2 and light exposure negatively

impacts photosynthesis and growth of marine primary producers."

The contradiction was thus already known at the time of publication. A scenario where the finding could be more useful, is one where the publisher's of the contradictory finding were not aware of previous findings. Even if the contradiction is known however, the collection of evidence for each side performed by the system can aid in a literature review of the issue.



Hypothesis: CO2 increasing CAUSES marine primary production increasing [88, confidence: 0.446, significance: 1.099]

Evidence (1):

CO2 levels in oceanic surface waters IS CO2 [47, confidence: 0.9, significance: 0.0]
CO2 levels in oceanic surface waters increasing CAUSES marine primary production increasing [49, confidence: 0.99, significance: 0.693]

Conflicting?

Hypothesis: CO2 increasing CAUSES marine primary production decreasing [92, confidence: 0.446, significance: 1.099]

Evidence (1):

CO2 levels IS CO2 [37, confidence: 0.9, significance: 0.0]
CO2 levels increasing CAUSES marine primary production decreasing [48, confidence: 0.99, significance: 0.693]

Figure 7.4: Contradiction found in the annotated corpus regarding CO_2 and primary production.

7.4 Ranking top list

Table 7.1 shows the top ranked hypotheses across all use cases, while table 7.2 shows the ranking for only the use case findings. Notably, only the iron hypothesis makes it to the top 10 list.

Although the use case hypotheses were ranked with a high significance, the system lacked confidence in them, due to the type of inferences being utilized in their inference paths. The intermediary step to the iron hypothesis "$\uparrow$iron $\implies$ $\uparrow$productivity" got the best ranking, due to it's high confidence, while still achieving a notable significance due to the causal transitivity involved. Worse fared the DOC hypothesis, which had the highest significance ranking, but among the lowest confidences, due to the system's poor confidence in abductive reasoning (see confidences in section 4.2.2). The iron hypothesis was notably beaten by the more specific "$\uparrow$iron $\implies$ $\downarrow PCO_2$ last ice age", due it having more confidence by involving less syntactical generalizations. It may seem a bit counterintuitive that the more specific hypothesis gained

Hypothesis	confidence	significance	score
$\uparrow$iron $\Longrightarrow$ $\uparrow$productivity	0.858	2.398	2.057
$\uparrow$labile dissolved organic carbon $\Longrightarrow$ $\downarrow$DOC	0.788	2.398	1.890
$\uparrow$iron $\Longrightarrow$ $\downarrow PCO_2$ last ice age	0.363	5.106	1.853
$\uparrow$iron $\Longrightarrow$ $\downarrow CO_2$	0.311	5.201	1.618
Obs: $\uparrow$DOC *CONTRADICTS* Expected obs: $\downarrow$DOC	0.631	2.526	1.594
$\uparrow$primary producer $\Longrightarrow$ $\downarrow PCO_2$ last ice age	0.415	2.708	1.124
$\uparrow$algae $\Longrightarrow$ $\downarrow PCO_2$ last ice age	0.386	2.803	1.082
$\uparrow$primary producer $\Longrightarrow$ $\downarrow CO_2$	0.355	2.803	0.995
$\uparrow$algae $\Longrightarrow$ $\downarrow CO_2$	0.33	2.899	0.957
$\uparrow$iron $\Longrightarrow$ $\uparrow$primary producer	0.922	0.788	0.727

Table 7.1: Top 10 most interesting ranked hypotheses.

Hypothesis	conf.	sign.	score
$\uparrow$iron $\Longrightarrow$ $\downarrow CO_2$	0.311	5.201	1.618
$\uparrow CO_2$ $\Longrightarrow$ $\uparrow$marine p.p. *CONTRADICTS* $\uparrow CO_2$ $\Longrightarrow$ $\downarrow$marine p.p.	0.189	3.114	0.589
$\uparrow$grazers $\Longrightarrow$ $\downarrow$DOC	0.006	6.738	0.040

Table 7.2: Rankings for the use cases.

more confidence, which is a result of less inferences having been made to reach it. One may intuitively perceive a more general hypothesis to be acceptable if it holds in many cases, with a few exceptions being fine, while the more specific hypothesis would need to hold more strongly in it's limited coverage. This interpretation is currently not captured by the system. The bottom five hypotheses are all related to the iron hypothesis, either by substituting "CO_2" with "PCO_2 last ice age", or being intermediary steps, and as such doesn't hold much interest.

More interestingly second in the ranking list, *labile dissolved organic carbon* was linked with *DOC* with a high confidence. This is in fact a contradiction, though it is not detected by the system, as labile dissolved organic carbon was not resolved to DOC. (DOC range from a spectrum of *labile* to *refractory* - labile is easily edible and fits correctly with DOC in the DOC hypothesis. Refractory DOC however not so much).

This illustrates another potential entry point to infer the DOC hypothesis from a contradiction, and provide some use in further work on the DOC hypothesis in the system. Following the trail back to the literature by clicking the URLs to the antecedent statements backwards in the inference path, leads to one of the nature abstracts, stating

When bacteria were limited by organic carbon, however, addition of labile dissolved organic carbon reduced phytoplankton biomass and activity and also the rate at which total organic carbon accumulated, explained as the result of stimulated bacterial competition for mineral nutrients."

The annotation contains the variables "labile dissolved organic carbon" and "phytoplankton biomass and activity". This is generalized by the system to

$$labile\ dissolved\ organic\ carbon \uparrow \implies phytoplankton \downarrow$$

In combination with the following background knowledge used for the DOC use case:

$$phytoplankton \downarrow \implies DOC \downarrow$$

Lead to the result by causal transitivity (IR1):

$$labile\ dissolved\ organic\ carbon \uparrow \implies DOC \downarrow$$

Once again, the contradiction is presented in the literature source, with the paper's finding being described at "counterintuitive", even if an LBD system were to spot the contradiction at the time of the paper's release, this would not have constituted a novel discovery. This is why LBD has better luck being novel by connecting more isolated knowledge.

7.5 Summary of findings

The system was able to infer both of the target hypotheses. From the annotated corpus, along with knowledge statements used for the DOC hypothesis use case, the system found an alternate contradiction that can be used to infer the DOC hypothesis, however the system did not detect this contradiction due to lacking ER. Visualizing inference paths of hypotheses where many paths exists can produce overwhelming graphs, but limiting the view to a single path at a time can make the output more comprehensible.

Chapter 8

Summary and Recommendations for Further Work

8.1 Summary and Conclusions

This thesis aimed to build a prototype inference system for literature based discovery, focusing on domain modeling, hypothesis construction/evaluation, and contradiction detection. A domain ontology, along with domain independent and domain independent inference rules were developed to fulfil research goal 1. Implementation of the model fulfilled research goal 2, with the HTML/JavaScript view and Neo4j view fulfilling research goal 3. The iron hypothesis, DOC hypothesis, and annotated corpus use cases allowed the fulfillment of research goal 4.

8.2 Recommendations for Further Work

8.2.1 More use cases

While the iron hypothesis was successfully inferred using only domain independent rules, the DOC use case showed that a hypothesis may require a lot more domain dependent rules to be inferred. More use cases could reveal the extent of varying requirements to infer hypotheses.

8.2.2 Better entity recognition

The entity recognition used for the thesis was a simple list mostly added to cover a few cases interesting for the use cases, however when use cases where combined, this lead to a lacking resolution of "DOC" and "dissolved organic carbon", which meant that the system would not detect a contradiction involving these variables. A more exhaustive entity recognition would be less likely to result in such failures.

8.2.3 Confidence rating

Currently, the system selects the most confident inference path to represent a hypothesis' confidence. However, the inference paths may interfere, or be independent. Instead of just choosing the best ranked one, they could be compared, so that independent inference paths strengthen the confidence in a hypothesis. For example, the multiple links from the literature between iron and primary production should increase the system's confidence in the iron hypothesis. (Although the links found so far are from the same article, this would compensate for the possibility of errors in each statement.) An Interesting experiments could be to have different sources of background knowledge, with a different confidence rating, and a large sample of syntactical generalizations to determine an accurate confidence value.

The iron hypothesis landed 4th in the ranking, being beaten by the more specific PCO_2 last ice age, and the intermediary connection of iron with productivity, which both won due to their higher confidence rating. It may be a good idea to give an increased confidence rating to a hypothesis where all variables are generalized, compared to one where only one variable is generalized, in line with the intuitive view that the confidence of a specific hypothesis does indeed yield more confidence to that specific hypothesis than more general hypothesis does.

8.2.4 Hypothesis evaluation

The system currently has several ways of scoring a hypothesis as more interesting, such as looking at the amount of literature sources in it's inference path, or whether inferences such as causal transitivity were used. However, these features are much more interesting under specific conditions, such as different literature sources being Swanson linked. A more in depth comparison of sources, judging their level of isolation from each others, could yield a better ranking scheme.

8.2.5 User collaboration

The system could be extended with interactive discovery browsing, such that the user can choose to keep or discard proposed hypotheses, altering their confidences and thus what can be inferred from them above the confidence threshold. With a bigger data set, the confidence threshold may start of high and decrease during the system run, for a more breadth-first search to handle the data without being overloaded. Ranking parameters could also be tuned as part of the user's interaction, such as specifying how important some variables are in comparison to others, and lowering the confidence threshold for hypotheses that are eligible for a high rank.

8.2.6 Performance

As the amount of knowledge in the system increases, the amount of knowledge base facts required for pattern matching will increase, thus harming performance. It will thus be important to keep the amount of facts in the KB within sustainable levels. Here various ideas are proposed to achieve this.

Modularization of the Knowledge Base

Organization of knowledge into modules, where each module can be loaded into or removed from the KB. This would limit discovery potential, and may require new strategies for determining what knowledge modules to load into KB.

Hypothesis caching (theorems)

When useful hypotheses are reached by the system, they should be stored so that they do not have to be recomputed in the future, and so they can be easily used to form new conclusions. Such useful conclusions in rule engines are referred to as theorems. When a new hypothesis becomes a theorem however, it stop being a "new" conclusion, thus it achieves the status of inputted background knowledge, with a high confidence and low significance. In a full working knowledge discovery system, users' input could be of use as for marking what conclusions should be considered theorems (e.g. if the system produces a hypothesis that happens to already be well-known background knowledge). For example, given an ontology containing the classes and predicates umbrella, head-cover, rain, falling-object and protects-against-falling-objects, and relationships “umbrella IS-A head-cover”, “rain IS-A falling-object”, “protect-against-falling-objects(head-cover)”, one can conclude that umbrellas protect against rain. This is indeed an important theorem about umbrellas, yet it is seen as utmost trivial. Thus if the system were to make this hypothesis, the user may want to inform it that it is to be considered basic background knowledge that may get loaded into main memory when doing reasoning involving umbrellas.

Reinforcement learning

Although theorems represent an important conclusion one wants to avoid recomputing, part of the path used to reach the theorem may still be of use in reaching other conclusions. During inferences, a lot of conclusions will be reached on way to a “final” hypothesis that may be of interest to the user. For example, while every sub-conclusion such “protect-against-falling-objects(umbrella)” may not be worthy of being cached, this could be useful for concluding that the umbrella will protect against other falling objects, such as snow. One could remember the importance of this hypothesis without caching it and loading it automatically to main memory during umbrella inferences, by having a large database, where each hypothesis formed at one point by the system is given a back-traced bonus when they lead to a useful hypothesis, or a penalty for a bad hypothesis. Thus paths of useful inferences can be followed later by looking up the bonuses given to the nearest inferrable hypotheses.

Appendix A

Acronyms

LBD	Literature-based discovery
LHS	Left-hand side
RHS	Right-hand side
WM	Working memory
KB	Knowledge base
IE	Information extraction
ER	Entity recognition
RE	Relation extraction
EE	Event extraction

Bibliography

Anyanwu, K., Maduko, A., and Sheth, A. (205). Semrank: Ranking complex relationship search results on the semantic web. *International World Wide Web Conference Committee*.

Anyanwu, K. and Sheth, A. (2002). The p operator: Discovering and ranking associations on the semantic web. *SIGMOD Record*.

Camerona, D., Bodenreiderc, O., Yalamanchilib, H., Danhb, T., Vallabhanenib, S., Thirunarayana, K., Shetha, A. P., and Rindfleschc, T. C. (2013). A graph-based recovery and decomposition of swanson's hypothesis using semantic predications. *Journal of Biomedical Informatics*.

Cutting, D., Kupiec, J., Pedersen, J., and Sibun, P. (1992). A practical partof-speech tagger. In *In Proceedings of the Third Conference on Applied Natural Language Processing*, pages 133–140, Stroudsburg, PA, USA. Association for Computational Linguistics.

DiGiacomo, R., Kremer, J., and Shah, D. (1989). Fish-oil dietary supplementation in patients with raynaud's phenomenon: a double-blind, controlled, prospective study. *The American Journal of Medicine*, pages 158–164.

Falkenhainer, B. (1990). A unified approach to explanation and theory formation. In *Computational Models of Scientific Discovery and Theory Formation*, pages 157–196, San Mateo, California.

Forgy, C. L. (1982). Rete: A fadst algorithm for the many pattern/many object pattern match problem*. *Artificial Intelligence*.

Friedman-Hill, E. (2003). *Jess in Action*. Manning Publications Co., Greenwich, 1st edition.

Gribbin, J. (1988). Any old iron? *Nature*, 331:570.

Laird, J. E., Newell, A., and Rosenbloom, P. S. (1987). Soar: An architecture for general intelligence. *Artificial Intelligence*.

Lally, A., Bachi, S., Barborak, M. A., Buchanan, D. W., Chu-Carroll, J., Ferrucci*, D. A., Glass, M. R., Kalyanpur, A., Mueller, E. T., Murdock, J., Patwardhan, S., Prager, J. M., and Welty, C. A. (2014). Watsonpaths: Scenario-based question answering and inference over unstructured information. *IBM Research Report*.

Langley, P., Simon, H. A., Bradshaw, G. L., and Zytkow, J. M. (1992). *Scientific Discovery: Computational Explorations of the Creative Processes*. The MIT press, Cambridge, Massachutas, 2nd edition.

Lee, S., Choi, J., Park, K., Song, M., and Lee, D. (2011). Inferring hidden relationships from biological literature with multi-level context terms. In *In Proceedings of the ACM Fifth International Workshop on Data and Text Mining in Biomedical Informatics*, pages 27–34, New York, NY, USA.

Marsi, E. (2016). Deriving the iron hypothesis: A case study of literature-based knowledge discovery in marine science. *Unpublished manuscript*.

Marsi, E. and Öztürk, P. (2015). Extraction and generalisation of variables from scientific publications. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 505–511, Lisbon, Portugal. Association for Computational Linguistics.

Marsi, E., Öztürk, P., Aamot, E., Sizov, G., and Ardelan, M. V. (2014). Towards text mining in climate science: Extraction of quantitative variables and their relations. In *Proceedings of the Fourth Workshop on Building and Evaluating Resources for Health and Biomedical Text Processing*, Reykjavik, Iceland.

Martin, J. H. and Gordon, R. M. (1988). Northeast pacific iron distributions in relation to phytoplankton productivity. *Deep Sea Research Part A. Oceanographic Research Papers*, 35(2):177–196.

Reichgelt, H. (1991). *Knowledge Representation: An AI Perspective.* Ablex Publishing Corporation, Norwood, New Jersey, 1st edition.

Ryther, J. and Kramer, D. (1961). Relative iron requirement of some coastal and offshore plankton algae. *Ecological Society of AmericaStable*, page 444.

Sarmiento, J. and Toggweiler, J. (1984). A new model for the role of the oceans in determining atmospheric pco2. *Nature*, page 621.

Shrager, J. and Langley, P. (1990). *Computational Models of Scientific Discovery and Theory Formation.* Morgan Kaufmann Publishers Inc., San Mateo, California, 1st edition.

Swanson, D. R. (1986). Fish oil, raynaud's syndrome, and undiscovered public knowledge. *Perspectives in biology and medicine*, pages 7–18.

Swanson, D. R. and Smallheiser, N. R. (1997). An interactive system for finding complementary literatures: a stimulus to scientific discovery. *Artificial Intelligence*, pages 183–203.

Thingstad, F. and Rassoulzadegan, F. (1995). Nutrient limitations, microbial food webs, and 'biological c-pumps': suggested interactions in a p-limited mediterranean. In *Marine ecology progress series*, pages 299–306, Department of Microbiology, University of Bergen, Jahnebakken 5, N-5020 Bergen, Norway.

Wilkowski, B., Fiszman, M., Miller, C. M., Hristovski, D., Arabandi, S., Rosemblat, G., and Rindflesch, T. C. (2011). Graph-based methods for discovery browsing with semantic predications. *AMIA Annu Symp Proc.*

Wren, J., Bekeredjian, R., Stewart, J., Shohet, R., and Garner, H. (2004). Knowledge discovery by automated identification and ranking of implicit relationships. *Bioinformatics (Oxford,England)*, pages 389–398.