‘The Lactate App’

Designing a user interface to enhance interpretation of elevated lactate levels in emergency and critical care

Lourens Elzinga
January 31, 2016

Master’s thesis

Human-Machine Communication
Department of Artificial Intelligence
University of Groningen, The Netherlands

Internal supervisor:
Dr. F. Cnossen (Artificial Intelligence, University of Groningen)

External supervisor:
Dr. M. W. N. Nijsten (University Medical Center Groningen)

university of groningen / faculty of mathematics and natural sciences

Abstract

Goal The goal of this study was to create a clinical decision support tool to aid clinicians in the process of interpreting lactate levels to a sufficient explanation. This tool should also be evaluated to see how it affects the clinician's performance and how well they enjoy using such a system.

Background Lactate levels stand out as a laboratory measurement in care of emergency and critically ill patients. Hyperlactatemia has the strongest single correlation with mortality in a wide variety of patients, and an increasing number of hospitals widely use lactate levels in relevant departments. The University Medical Center Groningen (UMCG) has collected a large database of lactate measurements in various patient groups, and it will soon have the means to perform continuous lactate measurements, which makes it very attractive to design a system that uses this knowledge of lactate levels to help guide diagnoses and therapy.

Design The tool created for this study is a web-based app which used an underlying expert system with a simplified model of lactate. The app presents the user with a series of questions about the patient and eventually presents probable explanations for the lactate level.

Evaluation An evaluation study was performed involving 7 resident intensivists and 3 student-researchers working at the intensive care unit as the UMCG. In this experiment, each participant was given a set of 10 vignettes - fictional patient descriptions, containing sufficient known medical information - and was asked to find an explanation for the elevated lactate. The evaluation study had two conditions, one where the participants were unassisted and one where the app was used to find the explanation. Afterwards the participants were given a questionnaire and a brief interview.

Results The results have shown a significant increase in accuracy in finding the correct explanations when using the app. The participants also found the app easy and enjoyable to use and have shown interest in using such tools for training purposes and in their work, in particular to assist in more complicated cases, or in cases of doubt.

Conclusion Using clinical decision support for lactate has shown promising results. With further improvement of the underlying models, these support tools can prove to valuable assets for clinicians when interpreting lactate levels.

Contents

Chapter 1

Introduction

Of all laboratory measurements that are available in the care of emergency and critically ill patients, measurements of lactate levels stand out. First and foremost, hyperlactatemia – elevated lactate levels – has the strongest single correlation with the outcome (mortality) in a wide variety of patients. It also changes quite rapidly, over the course of hours or less, which makes it prognostically relevant, as it can give an indication whether the patient's condition is stable or getting better or worse. For example, a constantly high lactate level may indicate that a patient is critically ill and needs immediate care or change in treatment, whereas a high but decreasing lactate levels may indicates that a patient is recovering. In an increasing number of hospitals, lactate levels are used in relevant departments such as the emergency department, operating rooms and the intensive care unit. Over the past years, the University Medical Center Groningen (UMCG) has collected a large database of lactate measurements in various patient groups, and it will soon have means to perform continuous lactate measurements on patients. These two factors make it very attractive to design a system that uses knowledge of lactate levels to help guide diagnoses and therapy.

1.1 Research Goals

The goal of the project was to develop an application to assist physicians in interpreting lactate levels, and guide them to probable diagnoses. This application should present physicians with questions regarding the lactate level of the patient, followed by several context-dependent questions and eventually give one - or more - potential diagnoses. The main goal for this system is to enhance the physician's ability to reach a correct diagnosis; therefore, the system should be as accurate as possible, and reach potential diagnoses with a minimal numbers of questions.

1.2 Thesis Structure

This thesis will begin by providing a theoretical background for lactate and (clinical) decision support systems in chapter 2. Chapter 3 will cover the technical design of the reasoning system at the core of the decision support system created for this study, followed a guide through the system itself, with a focus on the user interface and interaction in chapter 4. Chapter 5 will cover the methods and results of the evaluation study performed at the University Medical Center in Groningen. And finally, chapter 6 will provide a discussion on the decision support system and the results of the evaluation study.

Chapter 2

Theoretical Background

2.1 Lactate

In cell metabolism, two separate stages of energy production can be identified: glycolysis and oxidative phosphorylation. The process of glycolysis processes glucose and produces two adenosine triphosphate (ATP) molecules – the energy molecules of the body – as well as two pyruvic acid molecules. This pyruvic acid is used in the next stage, where it is processed into an additional 36 ATP using oxygen in the process known as the Krebs-cycle and oxidative phosphorylation. In the event that excessive amounts of pyruvic acid are produced, it can be converted into lactate and be stored. This process prevents accumulation of pyruvic acid levels and allows glycolysis to continue. Once pyruvic acid levels are decreasing, the lactate can be converted back to pyruvic adic (Bakker et al., 2013). For most of the 20th century lactate has been considered a dead-end waste product, mostly thought to be produced as a result of glycolysis in hypoxia, as the lack of oxygen would stop the process of oxidative phosphorylation, which results in a build-up of pyruvic acid. It was also believed to be the primary cause of O_2 debt following exercise, an important factor in acidosis-induced tissue damage and a main cause of muscle fatigue (Gladden, 2004).

In 1985, Brooks proposed the hypothesis of the lactate shuttle (Brooks, 1985), which describes the movement of lactate both within and between cells. Lactate is not just used as a waste product, but is actually continuously used in a wide variety of cell types in both aerobic and anaerobic conditions (Brooks, 2009).

Recent research into the role of lactate has shown that lactate being no more than a waste product is more of an exception than a rule. Instead, lactate has been found to be a central player in cellular regional and whole body metabolism, serving an additional source of fuel which can be transferred between cells where necessary (Gladden, 2004). During exercise, the formation of lactate can be used to provide an additional boost to muscles.

However, lactate does not only play a role during exercise. Lactate has been found to play an important role in the process of wound healing, recovery from sepsis and hemorrhage and other medical conditions (Gladden, 2004). The stress response resulting from these conditions cause a surge in energy production to aid the recovery process. As glycolysis is much faster than oxidative phosphorylation, assuming sufficient glucose is available, glycolysis will be able to produce much more pyruvic acid than the Krebs-cycle and oxidative phosphorylation can keep up with. The result of this is an increase in pyruvate, and therefore in lactate production. This excess lactate can then be transported to different cells to provide additional energy. Important to note here is that lactate in this case is not produced due to a lack of oxygen, but due to

an acute high energy demand of the body resulting from a stress response or hyperventilation (Bakker et al., 2013).

2.1.1 Clinical uses

Several studies have investigated the use of lactate as a predictor of outcome (mortality) in liver failure (Bernal et al., 2002), sepsis (Berkman et al., 2009), drug overdose (Manini et al., 2010), myocardial infarction (Vermeulen et al., 2010), and other areas. These studies have shown that increased levels of lactate are correlated to mortality, which means lactate can be used as a measure of the severity of someone's condition. More severe conditions typically result in higher blood lactate levels, which can be explained by the severity of the stress response evoked, and the resulting hypermetabolic state (Levy, 2006). These characteristics of lactate make lactate levels a very useful indicator in emergency care, which has resulted in the introduction and improvement of methods to measure lactate levels in patients (Weil and Tang, 2009), and opens the possibility to use these measurements for automated decision support.

2.2 Decision support systems

Decision support systems are computer-based information systems that support users in the process of making decisions. These systems can be used in a wide variety of domains. For example, in business and organizational decision making, they can provide decision-makers with an overview of critical business information to help them identify problems and make proper decisions. Another example is choosing which medicine to put on the market out of a set of candidates, each with their own side effects. In this situation, decision support systems can factor in the preferences of the decision-maker regarding the acceptability of these side effects, e.g., a medicine with a side effect of nausea is preferred over one that can cause severe complications. Decision support systems can also be used in more mundane tasks, such as helping a user in finding a suitable car to buy, based on preferences such as price, gas mileage, top speed, and so on. However, the field that is of interest for this study is the field of medicine, in which these systems are referred to as clinical decision support systems.

2.3 Clinical decision support systems

A question one could ask is why there is a need for these types of decision support in clinical settings. There are a variety of reasons to do so. To help cope with the increasing challenges related to knowledge and information management practice; to allow more meaningful use of electronic medical records, for which there is an increasing pressure to adopt and meaningfully use; to aid in the goal of delivering increasingly personalized health care services, tailoring treatment to the patient's preferences and their individual backgrounds and genomes; or to help clinicians cope with the rapidly changing knowledge in the world of medicine, as proper diagnoses and treatment require accurate and current knowledge Musen et al. (2014). To quote Musen et al.: "Facts decay as certainty as dead tissue does.".

Clinical decision support systems have become more common in the last few years in an effort to improve the clinician's performance and patient outcome (Bright et al., 2012). An example of this at Univerity Medical Center in Groningen is the system managing glucose regulation for intensive care patients (GRIP), which is currently used for over twenty thousand patients (Vogelzang, 2008). It should be noted that it is still debated whether clinical decision support systems (CDSS) actually result in an improvement of the clinician's performance. A review from

Bright et al., 2012 shows that both commercially and locally developed CDSS are effective in the health care process across diverse settings. However, a review by Jaspers et al., 2011 concludes that only few studies have shown any benefit to patient outcomes. There is, however, significant evidence that CDSS positively impacts the performance of health care providers.

Kawamoto et al. (2005) have investigated the important features of successful clinical decision support systems. This was done by means of a literature meta-analysis. The literature used was found on Medline, CINAHL and the Cochrane Controlled Trails Register up to 2003, and involved (clinical) decision support systems, or computer assistance in general. After selection, 70 studies were used, whereby 15 factors were identified as being relevant for the success of a CDSS in practice. Out of these 15 factors, four were found to result in a statistically significant improvement in the success of the CDSS:

- The decision support system must be part of the clinician's workflow, not something completely separate from it.
- Tools should provide recommendations on what actions to perform, rather than just providing an assessment of the situation.
- It should be available at the time and location where the decision needs to be made.
- It should be computer based.

Out of 32 systems that fitted these criteria, 30 showed a significant improvement in clinical practice. Another factor important in the effectiveness of decision support systems is trust. The book The Oxford Handbook of Cognitive Engineering (Lee et al., 2013) describes three factors when interacting with automation: trust, reliance and compliance. They describe compliance as the willingness of users to follow advice given to them, and reliance as the willingness to not take any action unless given instruction to (for example an alarm). In order for a decision support system to be useful, the user must be willing to follow the advice the system gives. If not, the system serves no useful purpose. Both reliance and compliance are linked to trust. In order for someone to be able to rely on and comply with a system, they must first trust the system. Muir (1994) defines trust as the basic expectation of persistence, the system's ability to perform a specific task and the belief in a system's ability in a domain for which we have no direct knowledge or where the system may have superior abilities.

2.3.1 Types of clinical decision support

In the book Biomedical informatics, Musen et al. (2014) specify three varieties of clinical decision support. The first type uses information about the current clinical context to provide the clinician with (highly) relevant information and documentation, such as so-called infobuttons. The second type provides patient- or situation-specific feedback, such as alerts, reminders, or recommendations for action. The latter type provides an organanized overview of present information to help facilitate decision making, such as dashboards, structures reports or graphical displays.

They further identified three stages of clinical decisions: Diagnosis, diagnostic process and management. A diagnosis involves analyzing data to determine an explanation for the symptoms of a patient. The diagnostic process goes a step further and involves questions such as which test to order. Finally, the management stage involves questions such as whether to treat a patient now or wait to see if the symptoms resolve on their own, or how to react to a patient's response to a treatment, and re-evaluating previous diagnoses. Each of these benefit differently from the different types of clinical decision support. For example, info buttons and recommendations are best suited for the diagnosis and diagnostic process stages, whereas the management stage benefits most from having a good overview (e.g. dashboards).

2.3.2 Early works

In the same chapter as the previous section, Musen et al. (2014) covered three advisory systems from the 1970 to provide an overview of the origin of CDS systems. They are the Leeds Abdominal Pain System (De Dombal et al., 1972), MYCIN (Shortliffe, 1976) and HELP (Kuperman et al. 1991; Warner 1979). These systems were picked by Musen et al. (2014) because they demonstrated three paradigms for CDS implementation that are still prevalent to this day. The following section provide a summary of these three systems and their design principles.

Leeds Abdominal Pain System

In the late 1960s, the University of Leeds developed a computer-based decision aid that used Bayesian probability theory. This system, the Leeds Abdominal Pain System, used information about sensitivity, specificity and disease prevalence for various signs symptoms and test results to calculate the probability of seven explanations for acute abdominal pain, using Bayes' theorem. The conditional probabilities used in the Bayesian reasoning were derived from high-quality data obtained from thousands of patients (Adams et al., 1986). The output of the system was limited to one of the seven possible diagnoses, whichever had the highest probability. For this, the system made two assumptions: conditional independence of the findings for various diagnoses and their mutual exclusivity and exhaustiveness.

In an evaluation study, 304 cases were diagnosed by both clinicians and the program. The program's diagnoses were correct 91.8% of cases, whereas the clinician's diagnoses were correct 65%-80% of cases. One notable difference was that the program never failed to diagnose appendicitis, a diagnosis that was often made incorrectly or missed by clinicians.

The program began to achieve widespread use, both in hospitals and other areas such as the British submarine fleet. However, the system never achieved the same level of accuracy as it did in Leeds. Musen et al. proposed that a likely explanation for this could be the considerable variations in the way data is interpreted by clinicians, resulting in difference in the way data is provided to the system. Alternatively, the difference could have been caused by different probabilistic relationships between findings and diagnoses in different patient populations.

This also sheds light on the downside of the system. Its accuracy is dependent on the calculated probabilities, which requires a good data set to derive the probabilities from. In addition, since the system is purely based on statistics, it cannot provide reasoning information beyond its calculated posterior probabilities.

MYCIN

MYCIN represented a different approach to computer-aided decision support. Instead of purely statistical approaches, MYCIN is a rule-based consultation system (Shortliffe, 1976). The developers believed that systems using straightforward algorithms and statistical approaches were inadequate for clinical problems where the underlying knowledge was poorly understood by clinicians or where there was disagreement even amongst experts. The result was a system that uses a set of interacting production rules to represent knowledge about organisms that could be causing infections in patients, and how best to treat them. There production rules were simple conditional statements which linked observations to conclusions that can be inferred from it. The conclusion drawn from these rules can then also be used as observations for other rules, which gives raise to the interactive property of the system.

The MYCIN program uses two forms of inference: forward and backward chaining. With forward chaining, whenever a production rule is fired and its corresponding conclusions are drawn, the system looks for other rules that can be fired with this new information. This type

of reasoning was not unique, it was used by most if not all contemporary rule-based system. Backward chaining on the other hand, does the opposite: based on the available production rules and current knowledge of the system, it can determine for which rules it currently lacks sufficient information and what it should consequently elicit from the user. A benefit of this is that the questions the system asks depend on previous answers given by the user. This turns the interaction into a form of dialog, which is also why MYCIN is referred to as a consultation system.

A big benefit of using rules is that it provides a means for the system to give insight in its reasoning process. MYCIN's rules in particular are formulated in a way that is easily translated into English sentences. During the consultation, users are allowed to ask the system for reasoning at any point, which will present the reasoning for asking the current question, or the presented conclusion. A second benefit of the use of rules is the ability to easily modify the system's knowledge base by adding, removing or modifying individual rules, without having to touch other parts of the knowledge base. This is in contrast with the statistical approach used in the Leeds system, where any alteration would require recalculation of all prior probabilities. Work on MYCIN has helped clarify how techniques known as knowledge acquisition could help developers work with experts to build new rule bases, how general-purpose rule interpreters can be used to provide decision support and why clinical decision support system are not always capable of providing satisfactory explanations for their reasoning process.

The system is not without its downsides however. Due to the size knowledge bases can grow to, the consultation dialog can become rather lengthy. And due to the interacting nature of rules, altering the rules can have unintended side effects that can be difficult to detect and correct.

HELP

The HELP system is an integrated information system at the LDS Hospital in Salt Lake City, built during the 1970s and 1980s. Its unique characteristic (for its time) was the integration of clinical decision support with underlying information technology. Instead of requesting input form users, HELP operated on electronic patient records and provided automated alerts when anomalies were detected. The system had a large impact on the development of the field, with applications and methodologies spanning nearly the entire field of biomedical informatics (Kuperman et al. 1991; Musen et al. 2014). The system uses a specific type of rule that relates values of data within the patient database to actions that clinicians should be reminded to take, which provides an event-driven system that automatically generates alarms, warnings or reports. In the 1990s a standard was adopted for encoding decision rules known as the Arden Syntax (Hripcsak et al., 1994). In this syntax, each decision rule is called a medical logic module (MLM). These MLMs are a form of what is known as an Event-Condition-Action (ECA) rule, which consist out of three parts: an event that triggers the invocation of the rule; a condition which is a logical test, which, if evaluated to true, causes the action to be carried out; and the action in question.

Whenever new patient data was added to the patient database, the HELP system checked if the data matched any criteria for invoking MLMs. If so, the system would evaluate the MLM and execute the appropriate action if the condition was met, which was usually form or alert. This idea of alerts was extended with the concept of clinical reminders, in which time is usually the triggering event – such as the time since a previous event, or the age of a patient – combined with other conditions.

This system also had its downsides. First of all, like the Leeds system, it was not able to provide any in-depth reasoning, as all decision logic consisted of one-step production rules, which also prevent rules from being chained, as was possible with MYCIN. The system also required integration with healthcare information systems (HIT) to function, which brought a few potential

complications. Deployment in different hospitals could be problematic, especially when there are differences in what information was stored and how it was stored. And since the system was fully automated, with no means of manual input, if information was unavailable in the patient database, the associated MLMs would silently fail.

Summary

The covered clinical decision support systems each show a different approach, differing in input, decision logic and output. The Leeds system used a set of input values, which through probabilistic Bayesian reasoning provided a set of probabilities for its 7 possible explanations, the highest of which was presented as the outcome. MYCIN, using a rule-based system, interacts with the user using a dialog. The system asks the user questions, and the user provides answers, or, asks the system for its reasoning. In contrast with the Leeds, where all input it needs is given up-front, MYCIN interactively elicits the information it needs from the user. Lastly, the HELP system, does not require the user to provide any information at all, instead gathering its input from the hospital's healthcare information systems, and only providing notifications and alerts when needed. Each of these systems have demonstrated different approaches to clinical decision support which are still used in modern CDS applications today.

2.3.3 Modern CDS methods

This section will cover five different methods employed by modern CDS systems, as described by Musen et al. (2014).

Infobuttons

The main goal of info buttons is to provide the user with easy access to relevant information. It can be considered one of the simplest and possibly most common forms of decision support. Systems using info buttons can display information found in the patient database, including names of drugs taken by the patient, performed laboratory tests and other patient problems, with buttons that provide more in-depth information about the drug, test or problem in question. For example, information for drugs could contain intended uses, normal dosages, possible side-effects or alternatives.

Branching logic

Branching logic methods are, from a computational perspective, one of simplest types of reasoning system. Typically, these systems are based on a decision tree or flowchart. Many CDS system have been made which have used flowcharts designed for clinicians and encoded them into a computer program. These types of systems, however, have been largely rejected by physicians, as they were deemed too simplistic and generic for clinical use. The added advantage of using flowcharts embedded in computer programs, instead of directly using the flowchart itself was also not very clear; printed flowcharts were deemed more than adequate for clinical care.

One noteworthy exception is a system deployed in Boston in the early 1970s, which used detailed algorithmic logic to provide advice on the diagnosis and management of acid-base and electrolyte disorders (Bleich, 1972). This program, which was designed for a PDP-9 computer, requested a series of measurements and other patient information from the user, which was then presented with an evaluation containing an assessment of the measurements and recommendations for action.

Despite not being very useful for CDS systems directly, branching logic is very useful in providing a algorithmic representation of clinical practice guidelines, which can be used as a guideline to develop (the knowledge of) a more complex CDS system.

Bayesian / Probabilistic methods

Due to the computational capabilities of computers, it was recognized in the 1960s that computer could be useful for diagnostic purposes by calculating posterior probabilities of diseases based on observations of patient-specific parameters. Many Bayesian diagnosis programs were subsequently developed, which achieved good accuracy in selecting an diagnosis from a set of alternatives. An example of such a system is the Leeds Abdominal Pain System mentioned in the previous section, which adopted a naïve Bayesian model that assumed there were no conditional dependencies among findings. This assumption meant that the model would, for example, assume that the presence of fever in the patient had no influence on the likelihood of the patient experiencing chills. In reality, this is an incorrect assumption. While this assumption meant naïve Bayesian models had limitations in accurately modeling a diagnostic problem, they made up for this by their computational efficiency.

The Bayesian analysis starts by giving each possible diagnosis a prior probability. This is usually the prevalence of this diagnosis in the population. For every pair of finding and diagnosis, there is a probability of said finding being encounter in the presence or absence of the diagnosis. To calculate the probabilities of each diagnosis, the program begins by considering one of the findings, and calculating posterior probabilities for each diagnosis by applying Bayes' rule, which uses the diagnosis' prior probability, and the finding's conditional probability for the specific diagnosis. When evaluating multiple findings, this process is repeated by turning the posterior probabilities for one finding into the prior probabilities of the next. This iterative application of Bayes' rule is referred to as sequential Bayes.

The method used by sequential Bayes shows why it has the assumption of no conditional dependencies. It only considers the probability of having a particular finding given a particular diagnosis, not he probability of having a finding, given another finding. Creating a Bayesian system that included these probabilities as well was considering complex and impractical. However, recent work on the use of belief networks have demonstrated that is it is possible and realistic to develop Bayesian systems that explicitly model these conditional dependencies, while remaining computationally efficient in most cases. Currently, most CDS systems that use probabilistic means to perform reasoning use belief networks as their primary representation of the underlying problem.

Rule-based approaches / Production rules

Rule-based systems emphasize the use of symbolic associations rather than purely probabilistic relationships to drive decision support. These systems typically use knowledge bases containing symbolically encoded information obtained from experts in a particular field. In order to be useful for a computational system, this information requires a formal encoding. There are a variety of ways in which this can be done. For example – as stated in the previous section – MYCIN uses a series of If-Then rules, containing a premise and one or more conclusion. Another approach was demonstrated by HELP with the use of MLMs - a type of Event-Condition-Action (ECA) rule. In order to be able to use these rules to perform reasoning, these systems need an interpreter, known as an inference engine. There are different ways in which inference engines can evaluate rules. The previous section has mentioned three types of evaluation: forward chaining, backward chaining and event-driven evaluation.

For all of these methods, the inference engine needs both a knowledge base with rules, as well as a source of data. This can be data directly provided by the user, or a connection with an underlying patient data, or any other source. As a quick recap, forward chaining involves evaluating relevant rules whenever new information becomes available, and applying their conclusion if the premise for the rule is evaluated to be true. This process is called chaining because the conclusion of a rule may affect the premise of another. Backward chaining does the opposite; it identifies which information the system is lacking by analyzing rules and tracing them back. This is particularly useful for systems that interact with a user, instead of a database, as it is used to identify information to elicit from the user. Event-driven evaluation is more suitable for non-interactive, database oriented systems, whereby rules are triggered by events in the underlying database, such as addition of records, or specific types of modifications.

In principle, rule-based systems allow for their rules to interact with each other. Systems that use forward and backward chaining methods typically rely on this. However, while rule interaction allows for more complex inferences, it has the drawback that modifying rules in a large knowledge base can have far reaching and unintended side effects. As such, MLMs, such as the ones used in HELP, are usually designed as stand-alone rules. The MLMs nor the Arden Syntax used to write them prohibit interaction with other rules, but doing so subjects them to the same side effects as If-Then rule interactions.

One important difference between rule-based and probabilistic systems is the construction of the knowledge base. Probabilistic systems can use existing patient data to calculate the prior probabilities and conditional probabilities. Rules based systems, however, require formalized rules provided by experts. This process is known as knowledge acquisition. Converting expert knowledge into a good set of rules is not always trivial. In fact, it is a well known problem in the field of rule-based, called the knowledge acquisition problem, whereby the most complex part of building a rule-based system is not building the system itself, but rather building a good knowledge base.

Ontology-driven methods

Ontology-driven systems are a class of CDS systems that use a higher level of abstraction of knowledge. Instead of just encoding knowledge of the clinical domain, these systems also encode problem solving knowledge. The basis of these systems is, as the name implies, an ontology, which is a controlled terminology containing a formal definition of all important entities, their properties, (inter-)relationships and constraints in a machine processable format.

An example of this type of system is ATHENA decision support system (Goldstein et al., 2000). It is a automated support system for guideline based care and was developed as part of the ATHENA (Assessment and Treatment of Hypertension: EvideNce-based Automation) project. It uses knowledge bases – typically created using the protég'e ontology editor by Stanford University – which capture the knowledge of a particular guideline, such as hypertension, diabetes, etc. These knowledge bases contain information such as taxonomical relationships (e.g. *cholesterol* is a kind of *lipid* or *serum potassium* is a kind of *laboratory test*), partitive relationships (e.g. a *systolic blood pressure measurement* is part of a *blood pressure measurement*) and information such as which patients should be treated in accordance to the guideline, what the recommended treatment steps are and which medications can be given to patients who follow this guideline (Musen et al., 2014).

When encountering a patient with a relevant diagnosis, the ATHENA system takes the knowledge base of the corresponding guideline and the patient-specific data provided by the patient database and provides an output suggesting a course of action to ensure the patient's treatment is consistent with treatment the guideline prescribes. Since these guidelines can be very long

and complex, the use of this automated system is very useful by drawing the clinician's attention to the steps that should be taken to ensure the patient gets the right treatment (Musen et al., 2014).

Chapter 3

System Design

3.1 Overview

The purpose of this study was to create an application that assists clinicians in the interpretation of lactate levels. The idea was to present clinicians with questions about the patient, such as the lactate level and symptoms, and, through some reasoning system, present a probable explanation for the lactate level of the patient.

This chapter will focus on the design of this underlying reasoning system. The next chapter will cover the application that was built around this reasoning system, and focuses more on the interface and user interaction.

3.2 Requirements

The reasoning system had to accommodate a series of requirements. First of all, it should be able to present relevant questions to the user, process their responses and eventually present the user with a conclusion. This meant that the system should have a set of questions available to choose from. Furthermore, since the system was to be used in the medical field, it should be capable of dealing with uncertainty. When diagnosing patients, it is very unlikely for doctors to be 100% sure of a particular diagnosis, and rather talk in terms of a diagnosis being very probable. This is something that should be reflected in the system by using a type of fuzzy-logic reasoning, which uses probabilities or certainties, as opposed to merely using true and false. Another requirement was to make the system suitable for teaching and training purposes, which means the system should be able to give insight into its reasoning process in a meaningful and informative way. A final requirement was to make the knowledge used by the system easily modifiable by experts. This makes it much more future-proof by allowing the knowledge to be updated by new findings in the medical field. In addition, this also opens the possibility for the system to be used for purposes other than lactate.

3.3 Reasoning system

Given the requirements covered in the previous section, there were several reasoning systems that were considered. As a simple starting point, a branching logic system was considered, where the knowledge was a basic decision tree, and each split represented a question. The branches to follow would be based on the answers provided by the users, which would eventually lead to

an answer. The fuzzy logic aspect could be integrated into this by following multiple branches simultaneously, each with a different certainty or weight. However, this system had the downside of quickly growing in complexity, making it very hard for experts to create and maintain large models. The use of questions as splits on the tree could also prove to be a limiting factor.

A second option, which has not been directly covered in the background chapter, was to use a machine learning algorithm. This solved the issue of creating the complex decision tree, but introduces its own set of problems. Being very related to probabilistic methods, it would require a good training set. While there was data available on lactate levels in patients, its suitability as a training set was questionable. Even if a suitable training set was available, a machine learning algorithm would make it difficult to determine the reasoning of a given output, due to their 'black box' nature. This can be partly mitigated by using rule extraction algorithms, which allow for rules to be created from a machine learning algorithm, but this could still make it difficult to obtain rules that are meaningful for teaching and training purposes.

A middle ground between the manual and automatic construction of decision trees using the methods mentioned above was to follow the rule-based approach and create an expert system, which uses a series of if-then rules to create an implicit 'decision-web', and uses certainty factors to provide the fuzzy aspect. The benefit of this system it allows direct control over the reasoning by supplying the rules, making it possible to annotate them with additional information to aid in the teaching aspect of the system.

3.4 Structure

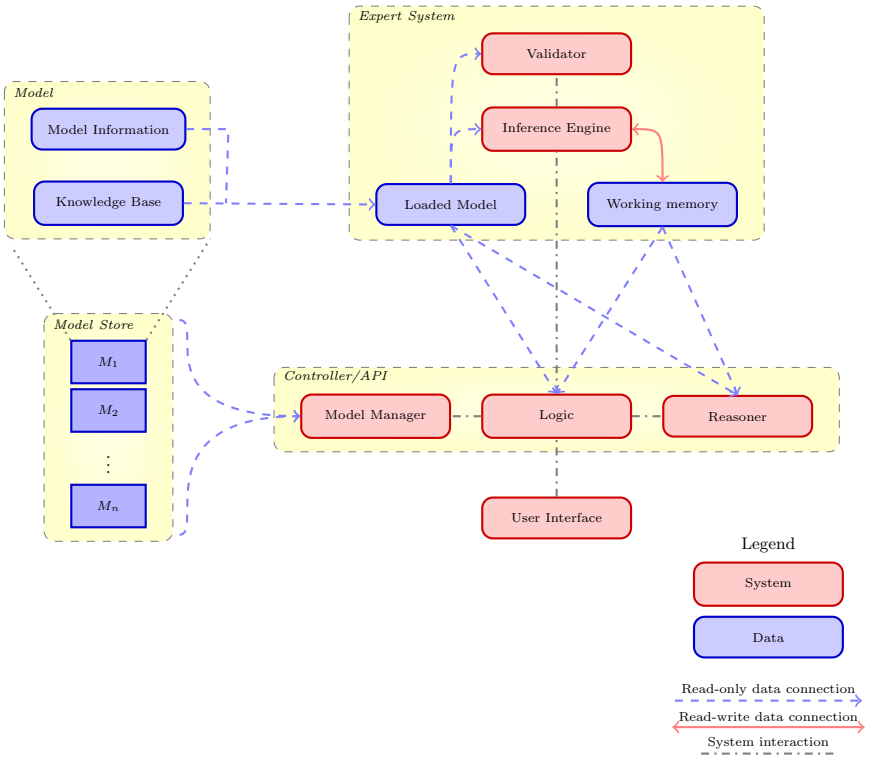

Figure 3.1: Schematic overview of the system's structure

Figure 3.1 shows a schematic overview of the structure of the system that was created for this study. In this schematic, the system is separated into three main sections: The model store containing a set of models, the expert system and the logic layer (the controller, or API) that sits between the user interface and the expert system. The next chapter will focus on the user interface and the controller while this chapter will focus on the model and expert system.

The schematic distinguishes between two types of elements: Data and System. The systems perform actual operations, while data is just information used by these systems. There are two types of interactions shown between data and systems, which are read-only or read-write connections. A read-only connection implies that the system uses the data, but does not alter it, whereas a read-write connection does. For example, the schematic shows that the inference engine, which is responsible for the reasoning of the expert system, reads information from a loaded model, but never modifies it. On the other hand, the inference engine both reads and writes to the working memory.

The following sections will cover the different parts of the expert system and models in greater detail.

3.5 Models

To make the expert system domain-agnostic, it makes use of models. Each model is fully self-contained and contains all information required for the expert system to function. This includes a variety of information, ranging from descriptive information such as a name, summary and description, to specific settings for the inference engine, and most importantly, the knowledge base, which contains all prior knowledge provided by experts.

3.6 Knowledge base

The knowledge base holds all prior knowledge of the model: the questions that can be presented to the user, the rules to be used by the inference engine and all prior beliefs. In addition to providing this information, the knowledge base is designed to help improve the expert system's performance by automatically building indexes and lookup tables. These allow for more complex queries, such as retrieving the question that answers a particular statement or finding rules with particular statements in their premise or consequent.

3.7 Beliefs

3.7.1 Statements

All beliefs in the expert system are represented as a set of hypotheses. Each of these hypotheses consists of a statement with a particular certainty, which indicates the degree in which the statement is believed to be true or false. Certainty factors are described more detail in the next section.

Most expert systems represent their knowledge (or statements) as a series of terms. For example, Peter Norvig's implementation of MYCIN uses four-term statements like "*burn patient is serious*". These four terms stand for the *object*, *context*, *attribute* and *value*. Another common approach, also used in the original MYCIN (Van Melle, 1978), is to use three-term object-attribute-value pairs. In the case of the previous example, this could be represented as "*burn is serious*" or "*patient-burn is serious*".

The distinction between the terms allows for statements to be grouped. All known information about the patient's illnesses could be grouped under the object "*patient*", for instance. This also allows the expert system to easily retrieve all beliefs on a certain object, which would be much more difficult if not impossible if all beliefs were presented as single term, and it makes it easier to convert these statements into a human-understandable format.

In both the three and four term notations, there is an attribute term, usually "*is*" or "*has*", which is mostly used to create English sentences from statements (e.g.: The *burn* of the *patient is serious*). However, the term does not add anything beyond this, and can be inferred based on the other two terms. As such, it was chosen to remove the use of the attribute term in this system, and use two terms: the object and value/property, e.g., "*burn serious*".

3.7.2 Certainty factors

The previous section stated that all beliefs are represented as series of statements and certainties. This certainty represents whether the statement is believed to be true or false, and to what degree. This could be done using probabilities, ranging from 0 to 1, where 0 means a statement is not believed, and 1 means it is believed. However, representing the degree of belief in such a manner creates a limitation. It can assert to what degree a statement is believed to be *true*, but not to what degree it is believed to be *false*. In other words, it can express the degree of belief, but not the degree of disbelief.

One could argue that these two are simply each other's opposites. A probability of 0.25 would indicate a belief of 0.25 and a disbelief of 0.75, and in strictly logical terms, this would be correct. However, humans do not tend to think about certainty in this way. To illustrate, a doctor could be 30% sure his patient has a particular disease. But if asked whether the doctor is 70% sure the patient does not have the disease, he is likely inclined to disagree.

In this respect, belief and disbelief are not each other's inverse, but are rather the opposite ends of the scale. We therefore chose to use a different system to represent this numerically: Certainty factors. These factors, introduced by the expert system MYCIN, range from -1 to 1. Where a negative certainty indicates disbelief, a positive certainty indicates belief, and zero indicates uncertainty. This is further illustrated in figure 3.2.

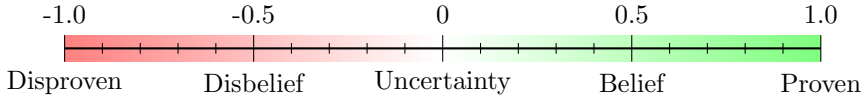

Figure 3.2: Illustration of the certainty factor scale

Certainty factors works by means of evidence. Every hypothesis H_i, has pieces of evidence E_1, E_2, ..., E_n. Each of these has a certainty factor indicating the *change* in belief that piece of evidence will cause with regards to the hypothesis. This can be written as $CF(H|E_1)$, meaning the certainty factor of a hypothesis H given evidence E_1. This change in belief can be both positive and negative. In order to determine the combined effect of multiple pieces of evidence, the parallel combination function (equation 3.1) is used, which takes two certainties and combines them into a single compound certainty.

$$CF_{combined} = \begin{cases} CF_1 + CF_2 - CF_1 CF_2 & \text{if } CF_1, CF_2 > 0 \\ CF_1 + CF_2 + CF_1 CF_2 & \text{if } CF_1, CF_2 < 0 \\ \dfrac{CF_1 + CF_2}{1 - \min(|CF_1|, |CF_2|)} & \text{otherwise} \end{cases} \tag{3.1}$$

When dealing with more than two pieces of evidence, this equation can be chained. To determine the combined certainty of three pieces of evidence, first calculate the combined certainty of E_1 and E_2, and then repeat the equation with the intermediate result and E_3. This can be used to combine any number of certainty factors, and yields the same result regardless of the order in which the evidence is presented (Heckerman, 2013).

The parallel combination equation (3.1) can be applied to any pair of certainty factors, however, a certainty factor of 1 or -1 is taken to imply the evidence *proves* or *disproves* the hypothesis. If any piece of evidence has a certainty of 1 or -1, the hypothesis remains proven or disproven (i.e. keep a certainty of 1 or -1), regardless of any other evidence provided. This can result in a contradiction, when one piece of evidence proves, and another piece disproves an hypothesis. In the combination equation, this results in a division by zero and is undefined (equation 3.2). The inference engine will detect when the certainties 1 and -1 are combined, and will treat this as an error.

$$\begin{aligned} &CF_1 = 1 \quad 0 < CF_2 < 1 \quad CF_c = 1 + CF_2 - CF_2 = 1 \\ &CF_1 = 1 \quad -1 < CF_2 \leq 0 \quad CF_c = \frac{1 + CF_2}{1 - CF_2} = 1 \\ \\ &CF_1 = -1 \quad -1 < CF_2 < 0 \quad CF_c = -1 + CF_2 + -CF_2 = -1 \\ &CF_1 = -1 \quad 0 \leq CF_2 < 1 \quad CF_c = \frac{-1 + CF_2}{1 - CF_2} = -1 \\ \\ &CF_1 = 1 \quad CF_2 = -1 \quad CF_c = \frac{1 + -1}{1 - 1} = \frac{\mathbf{0}}{\mathbf{0}} = \textbf{undefined} \end{aligned} \tag{3.2}$$

When dealing with evidence that is based on an hypothesis (i.e. an hypothesis which itself is evidence for another hypothesis), the uncertainty of the hypothesis needs to be factored in when determining the certainty of anything concluded from it. Normally, $CF(H|E_1)$ assumes that E_1 is an irrefutable piece of evidence (i.e. $CF(E_1) = 1$). If E_1 is an hypothesis, this would not necessarily be the case. This is accounted for by the serial combination function (equation 3.3), which takes the certainty hypothesis H would have had if its evidence were fully certain, and factors it with the actual certainty of the evidence. This does require that the evidence is at least believed, otherwise the compound certainty is zero. Without this constraint, disbelieved evidence would result in a reversed certainty for the hypothesis; if E_1 would increase belief in hypothesis H, disbelieving E_1 would instead cause a decrease. In logic terms, this would imply that $(A \implies B) \implies (\neg A \implies \neg B)$, which is false.

$$CF_{combined} = \begin{cases} CF_{antecedent} * CF_{consequent} & \text{if } CF_{antecedent} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.3}$$

3.8 Rules

Rules allow the expert system to perform the reasoning. They are formulated as if-then rules. Rules consist of two parts: a premise and a set of conclusions. These can also be referred to as the antecedent and the consequent of the rule. Since the expert system is based on fuzzy logic, premise is not a boolean statement, evaluating to true or false, but rather evaluates to a certainty factor. This represents how certain the expert system is of the premise, and if this certainty is above a certain threshold, which can be specified on a per-model and per-rule basis, the rule will be applied. The conclusions of the rule are formulated as a set of hypotheses. If the rule is applied, the rule will become evidence for the conclusions, with the specified certainty.

However, since the premise is based on certainty, the certainties of the conclusions are not used as-is. Instead, they are scaled based on the certainty of the premise, using the serial combination function.

3.8.1 Premises

The premise of a rule is created using a tree structure. Each node in this tree provided a certainty factor and a certainty range. The means by which this factor is obtained differs per type. Currently, six types of nodes are provided: Hypothesis, Is Uncertain, Is Unknown, Conjunction, Disjunction and Negation. These can be further subdivided in two groups: terminal and non-terminal nodes.

Terminal nodes are the leaves of the tree; they provide certainties based on the current beliefs of the expert system and cannot have any child nodes. These nodes also have additional parameters, such as the hypothesis to test. The *Hypothesis*, *Is Uncertain* and *Is Unknown* nodes belong to this group. In contrast, non-terminal nodes are always branches and derive their certainties from their child nodes. These include *Conjunction*, *Disjunction* and *Negation*

Table 3.1 contains an overview of the different node types and the source of their certainty.

Table 3.1: Premise nodes and their certainty source.

Node type	Source of certainty
Hypothesis	The current belief in a specified hypothesis.
Is Uncertain	Proven if the specified hypothesis is uncertain (i.e. has a certainty of 0), and disproven otherwise.
Is Unknown	Proven if the specified hypothesis has no evidence, and disproven otherwise.
Conjunction	The lowest certainty among its child nodes.
Disjunction	The highest certainty among its child nodes.
Negation	The negation (sign-flip) of the child node certainty. Requires exactly one child node.

To illustrate how rules are structured in practice, figure 3.3 shows a diagram of basic rule. Paraphrased in English, this rule states: "*If someObject has someProperty, then this is evidence that anotherObject has anotherProperty, with a certainty of 0.5*". In this diagram, the center green node is the 'implication' node, which connects the rule's premise to the conclusions. A more real-world example is shown in figure 3.4, which illustrates an actual rule used in the lactate model. This also illustrates the tree structure of premises, with the conjunction node being the root of the tree, and two hypothesis nodes being the leaves. Paraphrased, this rule states: "*If the lactate level's elevation is up to major and the patient has had a (recent) cardiac arrest, then this*

provides evidence that the explanation for the lactate is a recent cardiac arrest, with a certainty of 0.9.".

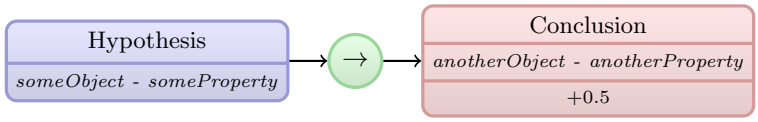

Figure 3.3: Simple rule diagram.

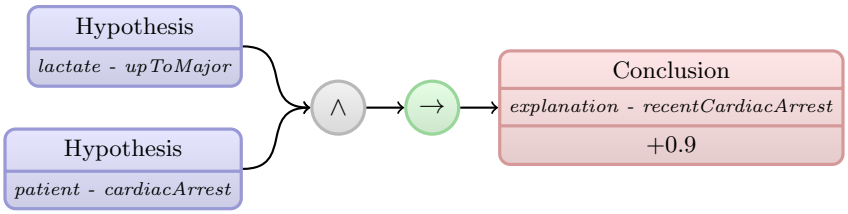

Figure 3.4: Diagram of a more complex rule.

3.8.2 Certainty ranges

The nodes in the premise tree do not only provide certainty factors, but also certainty ranges. The certainty factors provided are based on the current beliefs of the expert system and can be subject to change depending on new information acquired later. Certainty ranges take this lack of information into account, and determine the lowest and highest possible certainty based on current beliefs and information that could be acquired at a later on. These ranges play a key role in the inference engine's ability to select questions to present to the user. The algorithm used to calculate certainty ranges will be presented in section 3.12, p. 24.

3.9 Questions

In order for the expert system to acquire new information, a set of questions can be provided. Currently, the following four types of questions are supported:

- Yes/no questions.
- Multiple-choice questions.
- Integer questions. (whole numbers only)
- Numeric questions. (with decimal spaces)

In addition to the type, every question has a series of parameters, these include the question to present to the user, the elicitation order and group of the question, statements answered by the question, default answers, whether or not the question can be left unanswered (i.e. includes an 'I don't know' option), whether the question should be forcibly presented (even if the expert system is not interested in the answer), whether or not rule substitution is allowed and prerequisite questions that must be answered before the question becomes eligible for presentation. Multiple choice questions also require a set of options to select from, and numeric/integer questions require the minimum and maximum values and optionally units to be specified.

3.9.1 Elicitation order and grouping

Every question has an assigned order and can optionally be part of a group. An order is simply an integer number, and the group a string of text. These allow model makers to gain more control over the order in which questions are presented to the user. Normally, the inference engine ranks questions based how important and influential it perceives their answers to be (see section 3.12), but by specifying orders, questions with a lower order will be ranked as more important than questions with a higher order. Within the same order, questions are of course still ranked based on influence. With proper use of ordering, the expert system can be instructed to always present certain questions before others. In cases where ranking purely on influence alone results in a seemingly random order of questions, this can be used to dictate a more logical order.

Groups on the other hand do not influence the ranking process of the inference engine, but they affect the presentation of questions. Questions belonging to the same group are always displayed together, along with the group name. This not only allows a set of questions to be presented as a distinct group, but also provides this group with a title. This is further illustrated in chapter 4.

3.9.2 Membership functions

Since the expert system works using beliefs that are in the form of statements and certainties, the answers provided by questions can not directly be used as-is. In order for them to become meaningful, they must be converted into a series of statements and certainties. To facilitate this, questions can be given a set of statements with membership functions. These functions take the answer of a question and convert it to a certainty factor.

Due to the different types of questions available, each type of questions has its own types of membership functions that can be used. The simplest type of question is the boolean (i.e. Yes/No) question, and also has the simplest type of membership function. This predicate function simply provides a particular certainty depending on whether the answer was 'yes', 'no', or, if the question can be skipped, 'don't know' (see equation 3.4).

$$CF(question|answer) = \begin{cases} CF_{yes} & \text{If answer = 'yes'} \\ CF_{no} & \text{If answer = 'no'} \\ CF_{uncertain} & \text{Otherwise} \end{cases} \tag{3.4}$$

Multiple choice questions work in a way very similar to boolean questions. They use a list function, where every possible answer is given a certainty, including 'don't know'. The statement's certainty is simply the certainty factor associated with the given answer (see equation 3.5).

$$\begin{aligned} \mathbf{CFs} &= \{CF_1, CF_2, \ldots, CF_{n-1}, CF_n, CF_{uncertain}\} \\ CF(question|answer) &= \mathbf{CFs}(answer) \end{aligned} \tag{3.5}$$

Integer and numeric questions are a bit more involved. They can be given two types of functions: a threshold or a range function. These can be parameterized to set the threshold(s) and the certainties to return depending on whether the input value is above or below the threshold, or inside or outside in the case of a range. While the thresholds can act like step functions, jumping from one certainty to another, they can also be parameterized to use a slope instead by specifying an area (width) around the threshold value to interpolate across. This area can be centered on the threshold, or aligned before or after the threshold. In addition, the way this interpolation occurs can be parameterized as well. By default, linear interpolation will occur between the two certainties if the input value is inside the interpolation, but non-linear methods are also

available. Currently, three additional non-linear easing functions are available: ease in, ease out and sigmoid. All of these easing functions (including linear interpolation) are implemented using cubic bézier curves. The four control points for the curve are determined based on the type of easing function; the upper and lower boundaries, which are determined by the threshold, the width and alignment; and the curvature factor. The curvature factor is a number between 0 and 1 which determines how strong the curvature of the slope is. Table 3.2 shows how the control points are calculated for the different types of easing functions and figure 3.5 shows a plot of these functions.

Table 3.2: Control points for bézier curves used by easing functions. B_l and B_u are the lower and upper input value boundaries respectively, and c is the curvature (0.0 - 1.0)

Type	p_1	p_2	p_3	p_4
Linear	B_l	B_l	B_u	B_u
Ease In	B_l	$(1-c)B_l + cB_u$	B_u	B_u
Ease Out	B_l	B_l	$(1-c)B_u + cB_l$	B_u
Sigmoid	B_l	$(1-c)B_l + cB_u$	$(1-c)B_u + cB_l$	B_u

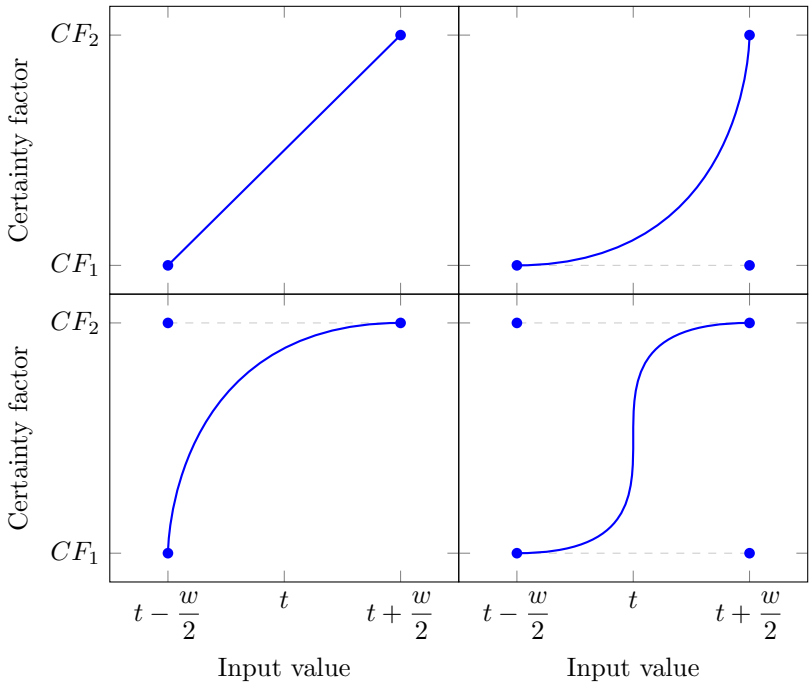

Figure 3.5: Plot of all four easing functions, centered on threshold t with a width of w. From top-left to bottom-right: *Linear*, *ease out*, *ease in* and *sigmoid*. Blue dots indicate the positions of the control points of the bézier curve. Dotted lines indicate the line on which the middle control points move depending on curvature setting. In this plot, all easing functions use use a curvature of 1.

3.9.3 Rule substitution

Normally, statements answered by questions can not be part of the consequent of a rule. Allowing this would permit rules to overrule or conflict with answers given by the user. However, there are certain circumstances where this ability is desirable. For example if the answer to one question already provides an answer for other questions. Instead of having those questions needlessly presented to the user, rule substitution can be used instead.

Rule substitution can be enabled on a per-question basis, and removes the restriction that rules can not contain statements answered by that particular question. If a rule is applied whose consequent contains statements from a substitutable question, the question will be considered answered and will no longer be eligible for elicitation. In the event that a substituting rule is applied on a question that has been answered previously, the answer to that question is discarded.

Figure 3.6 provides an example of a substitution rule used in the lactate model, this rule verifies that the user answered the question whether or not the patient has myocardial infarction, and if the answer was either "no" or "don't know", it will assert that the patient does not have cardiogenic shock, thereby preventing the associated question from being presented.

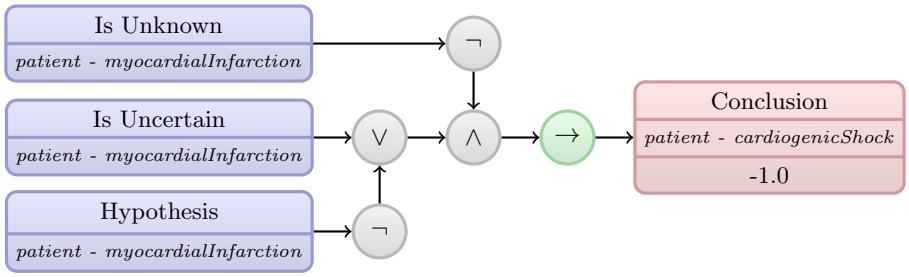

Figure 3.6: Example of a substitution rule.

3.10 Validation

To detect errors or inconsistencies in knowledge bases, the expert system includes a validator. This validator analyzes the knowledge base, searching for potential problems. Three different types of findings are distinguished: *errors*, *warnings* and *notifications*. Errors are critical problems that prevent the model from working properly. An example of these are cyclic rule structures, as these will cause an infinite loop. Warnings are problems that will not inhibit the model from working, but may have unintended consequences, such as rules using statements that do not have any sources (i.e. there are no rules or questions that provide evidence on that statement). The last group, notifications, are the least important, and inform model creators of possible oversights. Table 3.3 provides an overview of conditions checked for by the validator.

Table 3.3: Errors, warnings and notifications given by the validator.

Type	Description
Errors	
Cyclic Rule	A rule's conclusions affect its premise.
Nesting Limit	Rules are nested too deeply (150 layers), and could cause stack overflows.
Prior Belief Contradiction	A prior belief with a certainty of 1 or -1 is contradicted by another prior belief or rule.
Elicitable Prior Belief	A prior belief affects an elicitable statement.
Elicitable Consequent	A rule consequent affects an elicitable statement from a question that is not substitutable.
Warnings	
Unresolvable Premise	An hypothesis used in a premise cannot be resolved by rule, prior belief or question.
Possible Rule Contradiction	Two rules exist with conclusions that contradict each other (with certainties of -1 and 1).
Missing Consequent	There is a rule without conclusions.
Incomplete Substitution	A question-substituting rule does not substitute all statements of the question.
Notifications	
Unused Question	A question has no path to a goal statement and will never be presented. Not raised for forced questions.
Unused Forced Question	A question has no path to a goal statement but is forced to be presented.

3.11 Working memory

The working memory contains all currently held beliefs, and is tightly coupled to an inference engine. The working memory is separated from the knowledge base to allow multiple inference engines to operate on the same knowledge base in parallel without conflicting with one another.

The beliefs are stored in the form of a table, where each row contains a statement, certainty factor and source. The source can be a prior belief, a rule, or a question, and there can be many instances of the same statement in the working memory, as long as they have different sources. When requesting the certainty of particular statement, all entries are combined together, as each of them is essentially a piece of evidence.

Storing all individual sources for a particular statement separately, instead of only storing the combined total allows for individual sources to be modified or removed. This enables the inference engine to revoke rules that have previously been applied, in the event their premise is no longer believed. This also allows the end user to modify answers after they have been initially elicited.

In addition to storing current beliefs, the working memory also stores the answers that were given by the user through elicitation, allowing them to be reviewed if necessary. This also facilitates rules substitution, as this is also treated as a type of 'answer' and stored.

3.12 Inference engine

The inference engine lies at the heart of the expert system and is responsible for performing all reasoning steps, based on the information contained in the knowledge base, and current beliefs stored in the working memory. It can perform two different types of reasoning: forward and backward chaining. Both types will be explained in further detail in the following subsections.

3.12.1 Forward chaining

The process of forward chaining tries to acquire new beliefs based on current ones by applying (or revoking) rules. This process continues until no more rules can be applied or revoked. In order to optimize this process, the inference engine has an update queue, which contains all statements which have recently been changed. Whenever a belief is added or removed from the working memory, the associated statement is automatically placed on the update queue. When doing forward chaining, the inference engine pops a statement from the update queue, finds all rules that have that statement in their premise and (re-)evaluates them until the update queue is empty. If rules are applied, the beliefs added causes the associated statements to be placed on the back of the update queue. For this reason, when the update queue is empty, no more rules need to be evaluated. This is also the reason circular rules are checked for and blocked, as this would result in an infinite loop with the same set of rules being evaluated indefinitely. If necessary, the forward chaining algorithm can be instructed to do a so-called full pass. This will first evaluate every single rule in the knowledge base before using the update queue. This is used when an inference engine is initially created and run for the first time to ensure all rules are properly evaluated.

The algorithms 1 and 2 show the pseudo-code for the forward chaining and rule evaluation algorithm.

Algorithm 1 Forward chaining algorithm

```
1: function forwardChain(doFullPass)
2:     if doFullPass then
3:         for all rules ∈ KB do
4:             evaluateRule(rule)
5:     while updateQueue ≠ ∅ do
6:         statement ← pop statement from updateQueue
7:         for all rules with statement in premise do
8:             evaluateRule(rule)
```

Algorithm 2 Rule evaluation algorithm

```
1: function evaluateRule(rule)
2:     cf ← CF(rule)                                  ▷ Gets CF for rule premise.
3:     if rule has custom rule threshold then
4:         threshold ← custom rule threshold
5:     else
6:         threshold ← model rule threshold
7:     if certainty ≥ threshold then                  ▷ CF is sufficient to apply the rule.
8:         for all conclusions of rule do
9:             if conclusion ≠ elicitable then
10:                add conclusion to beliefs with rule as source
11:            else
12:                question ← question with conclusion's statement
13:                if question allows substitution then
14:                    remove current answer of question
15:                    mark question as 'substituted'
16:                    add conclusion to beliefs with rule as source
17:    else if beliefs exist from rule then
18:        remove beliefs from rule
19:        if rule has substituted questions then
20:            for all substituted questions do
21:                mark question as unanswered
```

3.12.2 Backward chaining

The goal of backward chaining is to identify new information the expert system would like to acquire by means of elicitation. Or more simply put, a list of questions that should be presented to the user. In order to do this, the inference engine requires a goal object, i.e., an object whose properties are of interest. To determine which questions to present, the inference engine scans through all rules that are linked to statements with the goal object, and evaluates their certainty range to determine if the rule can be applied or whether the certainty of the premise can fluctuate. If this is the case, the scan evaluates rules and questions that affect the current rule. When questions are encountered, they are added to a list of candidates. This list also includes information about the influence of a rule, i.e., the total variation in the certainty of goal rules the question can cause; dependencies, questions whose eligibility may be affected by the answer to the current question; and the number of paths between the question and goal statement and the rules involved. This recursive process continues until the entire network of rules has been scanned. Once completed, there will be a list of eligible questions, which is then sorted by question order, number of dependencies and influence and returned. Algorithm 3 shows the pseudo-code for this algorithm.

Algorithm 3 Backward chaining algorithm

```
1: function BACKWARDCHAIN(goalObject)
2:     candidates ← ∅
3:     for all questions ∈ KB do
4:         if question is not answered then
5:             if question is forced then
6:                 add question to candidates
7:             else if question has statement with goalObject then
8:                 add question to candidates
9:     for all rules with goalObject in consequent do
10:        dependencies ← ∅
11:        BACKWARDCHAIN(candidates, dependencies, rule, 2.0)
12:    return SORTCANDIDATES(candidates)
13:
14: function BACKWARDCHAIN(candidates, deps, rule, influence)
15:    range ← CF_range(rule)
16:    if rule has custom rule threshold then
17:        threshold ← custom rule threshold
18:    else
19:        threshold ← model rule threshold
20:    if range.max ≤ threshold or range.min ≈ range.max then
21:        return
22:    newInfluence ← min(influence, range.max − range.min)
23:    isDependency ← false
24:    chainTargets ← ∅
25:    for all statements in antecedent of rule do
26:        if statement is wildcard then
27:            for all properties for statement.object do
28:                stmt ← Statement(statement.object, property)
29:                if stmt is elicitable and stmt has no evidence then
30:                    isDependency ← true
31:                    add question to candidates
32:                    update dependencies and influence of question
33:                else
34:                    add statement to chainTargets
35:        else
36:            if statement is elicitable and statement has no evidence then
37:                isDependency ← true
38:                add question to candidates
39:                update dependencies and influence of question
40:            else
41:                add statement to chainTargets
42:    if isDependency then
43:        push rule on deps stack
44:    for all statement ∈ chainTargets do
45:        for all rules with statement in consequent do
46:            BACKWARDCHAIN(candidates, deps, rule, newInfluence)
47:    if isDependency then
48:        pop rule from deps stack
```

3.12.3 End criteria and results

While the backward chaining algorithm finds questions to present, at some point the expert system should have a result to show. To determine when enough beliefs have been acquired to present a result, models have a set of end criteria. These criteria are a set of conditions which, when met, indicate the elicitation process should be stopped and the user should be presented with the results. There is one fixed end criterion, which is the lack of further questions. If backward chaining provides no candidates for elicitation, the only course of action is to proceed to the results. There are two additional end criteria that can be enabled on a per-model basis. The first is to present results if a goal statement (i.e. a statement whose object is the goal object) has reached a certain minimal certainty, and the second is to show results if a specified (non-goal) statement reaches a particular certainty. This latter option allows for more fine-grained control as the model maker can devise a (set of) rules that lead to that statement when any arbitrary set of conditions are met.

The results that are presented are the properties of the model's goal object. Figure 3.4 (p. 19) showed an example of a rule used in the lactate model, whose conclusion used the statement "*explanation*" "*recentCardiacArrest*". In the lactate model, which will be covered in more detail in the next chapter, uses "*explanation*" as the goal statement. In other words, this rule - if applied - provides a result: The explanation for the lactate is recent cardiac arrest. Similarly, any other statement whose object is "*explanation*" provides a possible result with a particular certainty. When an end criteria is reached, all of these are presented to the user.

3.13 Reasoner

When presented with the results, it may be of interest to get information about the reasoning of the expert system. While the inference engine does all the reasoning steps, it does not keep track of when beliefs were added and rules were applied, or why. To provide a reasoning, a separate reasoner system is used. This reasoner uses information stored in the working memory and knowledge base to try to deduce the reasoning steps that led to the current belief in a specific (goal-)statement. It does this by initially looking at all sources for the statement of interest and then tracing these back until an end point is reached at a question of a prior belief. This results in a set of paths, each going from the statement, through a set of zero or more rules to a prior belief or statement. However, this set of paths only cover supports for the statement, regardless of whether it's positive or negative, it does not undercutters. Undercutters block a line of reasoning that could have resulted in a change in belief for the statement of interest. To detect these, the reasoner evaluates all possible paths leading up to the statement, and follows the paths that were not actually used. This traces continues until a question or prior belief is found that is responsible for stopping a rule from being applied, in which case this path is also added to the set of paths, but marked as an undercutter instead of a support. There is one exception to this. If the undercutter is a question, it is only added if the question has been presented and answered, otherwise this would cause confusion, as the reasoning would contain questions that were never presented.

When all paths are found, there are then built into an hierarchical structure, starting at the questions and prior beliefs and working down to the result-statement. This creates a structure that contains which questions and prior beliefs were of importance in reaching a particular result (either as support or undercutter), and which rules connected these together.

Chapter 4

The App

The previous chapter covered the design of the expert system. However, this expert system is not a stand-alone application. In order to make use of this expert system, two applications were created: a front-end for the expert system and a model editor. This chapter will cover the design and user interface of these two applications, and will also cover the lactate model that was created for this study.

4.1 Web front-end

The front end application provides a user interface that allows end users to utilize the expert system. It provides the means to load models, handles the elicitation process and presents the results to the end users.

4.1.1 Design considerations

The front end application was subject to several design considerations. First and foremost, the application should have a simple self-explanatory interface that is easy to use and understand.

The application should also be usable on a range of devices, including desktop computers, tablets and smart-phones. Especially the ability to use the application on a mobile device introduced some constraints. Not only would the application and user interface have to be rebuilt for each type of device, the limited processing power and memory on (in particular older) mobile devices could result in them having problems running the expert system, especially with very complex models. Having the expert system on the device of the end user introduces another problem: the models. These must be present on the device in order to be loaded by the expert system. So how could these models be provided? One option is to require the user to install them manually, though this is not a particularly user-friendly method. Another option would be to have some sort of central repository where models can be acquired, but this creates its own set of complications, such as where to host this, who are responsible for managing and maintaining it, who are allowed to add models to the repository, etc.

In order to deal with all these potential issues, the easiest solution was to make the application web-based. Instead of requiring the end-users to install it on their own devices, they can access it using a web browser. To use it, the application simply needs to be run on a server, or really any computer accessible from the internet. This eliminates the need to port the application to all kinds of different (mobile) platforms, as it only needs to run on server operating systems, and the models only have to be installed on the server running the application. This also solves the

mobile devices potentially having difficulty running the expert system, as this is all handled on the server side. For the end-users, all that is required is a (relatively modern) web browser and an internet connection.

4.1.2 Server design

The server application is built around a custom-made HTTP/1.1 compliant web server. It employs an asynchronous, event-driven approach to handling requests, similar to popular server software like Nginx or Node.js. This approach allows it to handle a large number of concurrent connections[1] whilst retaining a low memory footprint. The server supports both standard HTTP and secure HTTPS connections, though enabling the latter requires a server certificate and private key.

The web content is written as a set of static HTML pages that make extensive use of AngularJS. AngularJS is a javascript library that provides a framework for client-side Model-View-Controller (MVC) and Model-View-ViewModel (MVVM) architectures. It does this by extending the HTML language to allow for the definition of custom tags and attributes, which are then compiled into 'standard' HTML that the browser can understand. This also provides means for two-way data binding between HTML and javascript. HTML elements can be bound to a model in the form of javascript variables, whereby the HTML is automatically updated whenever the variable changes, and vice versa, e.g, in the case of form fields. Combined with custom HTML tags for iteration, control flow and more, this allows for creating web applications where the HTML merely provides a template for pages. The actual page shown to the user is the result of combining the template with the actual data to present, which is acquired separately by the browser, typically through static or dynamic JSON data sources.

The use of AngularJS takes the task of populating the pages with content away from the server and moves it to the client (i.e. the browser). The server is only required to deliver static content, i.e., the html pages, javascript, style sheets, images, etc; It is not required to construct a custom web page for every request. This reduces the load on the server, especially when handling many users simultaneously.

JSON API

In order for the web app to interact with the expert system, the server provides an API to provide this access. The API is available under the /api/ path, and provides a series of endpoints that allow the browser to query information or perform actions related to the expert system. Table 4.1 shows a list of all end points and their purpose. All endpoints use JSON for requests and responses.

Some endpoints require a session, in order to identify which inference engine to perform the operation on. This is implemented using a session cookie consisting of 32 random characters, which can be requested using the *new-session* endpoint. While using cookies with web APIs might not be the cleanest approach, it is the easiest way to retain the session ID across page reloads without requiring more modern features, such as the browser's local storage, which may not always be available or enabled in certain browsers.

[1] Also known as the C10K problem.

Table 4.1: List of all API endpoints provided by the web server

Endpoint	Arguments	Description
system-info		Returns basic server information. (e.g. server version)
model-list		Returns a list of all available models.
model-info	Model ID	Returns information on a specific model.
new-session		Creates a new session. A session is mandatory for all of the following endpoints.
load-model	Model ID	Sets up a new inference engine and loads the specified model.
run-rodel	Answer list	Applies all answers (if any) and performs a backward chain. Returns a list of questions to present, or signals the app to show the results page.
reset-model		Resets the working memory. Effectively restarting elicitation.
results		Returns all current results.
reasoning	Statement	Provides reasoning data for the specified statement.
answers		Returns a list of all answers provided thus far
ignore-result		Instructs the expert system to blacklist the current result. This allows for elicitation to continue after a result has been found.
report		Generates and returns a download link for a working memory dump.

4.1.3 Structure

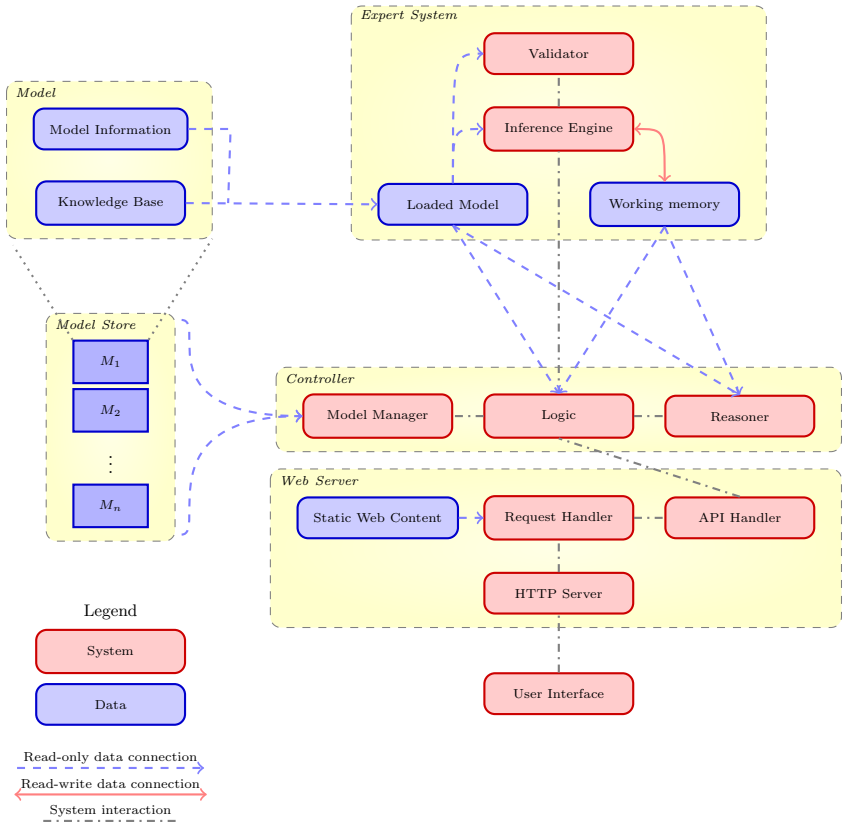

Figure 4.1: Schematic overview of the system's structure, including the HTTP server components.

Figure 4.1 shows the same schematic as shown in the design chapter, but now with the server components included. The user interface interacts with the HTTP server, which can either provide static content (i.e. HTML pages, CSS/JavaScript files) or invoke the API, which interacts with the underlying expert system.

4.1.4 User interface

The user interface was designed to be clean, easy to use and mobile-friendly. To achieve this, the web pages use the Bootstrap framework to provide the styling and make pages responsive (i.e. adjust their layout based on the size of the screen).

Intro / model selection

When the user browses to the app, they will be met with a welcoming page, giving them a brief explanation of what the app is and does. As the underlying system is based on models, the user is provided with a model selection screen upon continuing, as seen in figure 4.2.

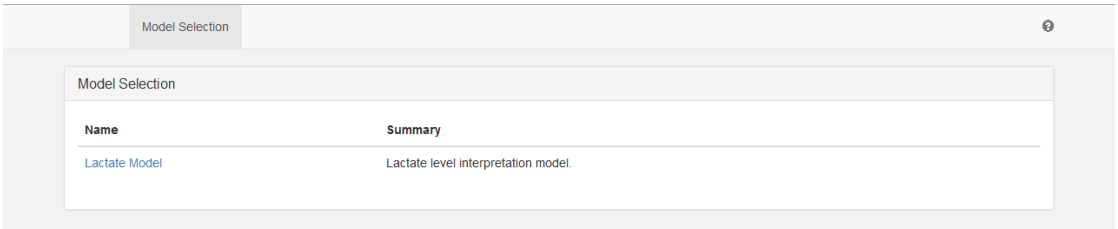

Figure 4.2: The model selection page.

This model select screen provides an overview with models available on the server, shown as a list with names and summaries. In the event the server not have any models available, a message to that effect is displayed to the user instead. If the user wants more information about a particular model, they can click (or tap) on it, which will bring them to the model confirmation screen (figure 4.3), which provides more detailed information about the model. This screen provides the user with the option to run the selected model, or to return to the model list should they wish to look at other models instead.

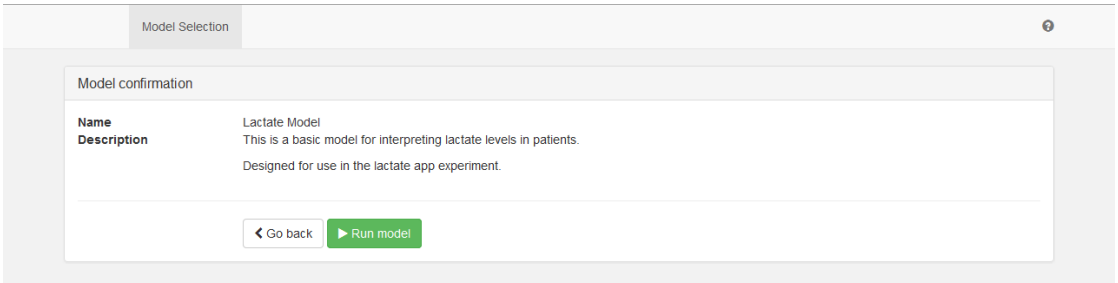

Figure 4.3: The model confirmation page.

Elicitation

Should the user decide to to run a model, the web app will establish a session using the API (See JSON API, above), set up an inference engine with the requested model, and proceed the elicitation screen (figure 4.4). On this screen, the user is presented with questions provided by the expert system. The questions to be presented are determined by having the inference engine execute a backward chain. The backward chain will provide a set of questions unless an end criteria has been reached.

Depending on the settings of the model, the user can be presented with single question; a group of questions, in which case the name of the group will be displayed as well (e.g. figure 4.4); or multiple (groups of) questions. In the latter case, the questions groups are separated from each other, and shown with their respective group name, making it very clear which questions belong to which group (e.g. figure 4.5).

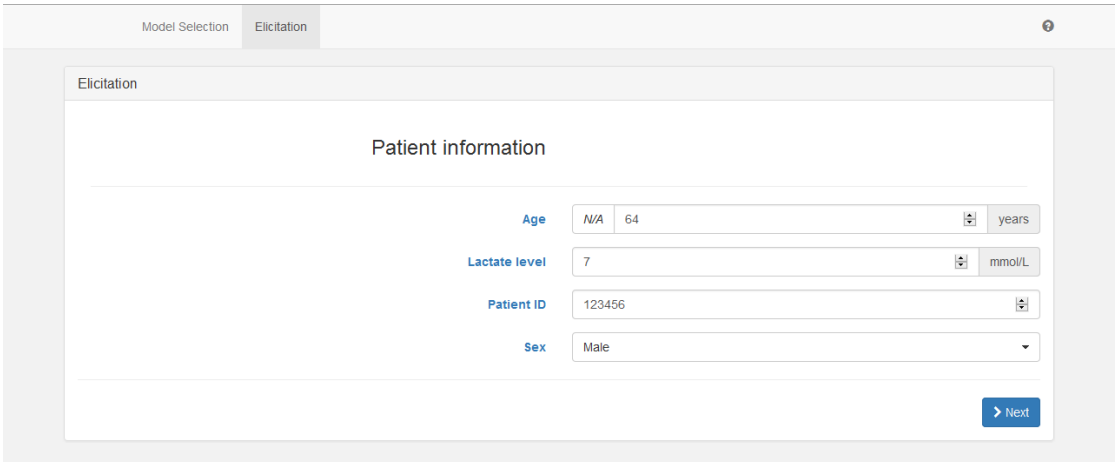

Figure 4.4: An elicitation page with numeric and multiple-choice questions.

To make the displaying of questions resolution independent, the questions and corresponding answer fields are aligned towards the center of the screen, each taking up half the available width. The questions are right-aligned, and the answer fields left-aligned. This way, the questions are always directly next to their corresponding answer field, even if questions have differing lengths. In addition, this alignment also results in all answer fields being vertically aligned, which is much easier on the eyes. There is one caveat with this layout: if the screen size is exceptionally low, the question text (and answer field) will become very cramped. To solve this, the layout is altered if the screen size becomes too low, showing the question and answer fields on separate lines, each spanning the full width of the screen instead.

In order to provide the user with additional information about the questions, the description of the questions (if provided by the model) can be brought up by hovering the mouse cursor over the question, or by clicking (tapping) it. The reason for having two separate ways to do this is to accommodate mobile devices. While desktop platforms can use mouse hovering to bring up the tool-tip, touch-based mobile devices do not commonly posses such a cursor and rely on tapping instead.

Since the expert system system supports multiple types of questions, each type of question has its own type of input field, which best suits that type of question. Numeric and integer questions use a numeric text field; yes/no question use a series of buttons with a tick, cross and question-mark; and multiple-choice questions use a dropdown menu. The answer fields are designed to be touch friendly, so they are relatively big to make them easier to tap. This is shown more clearly in figure 4.5, where all questions are yes/no questions, the figure also shows the buttons instead of having a single checkbox, as is common in desktop environments.

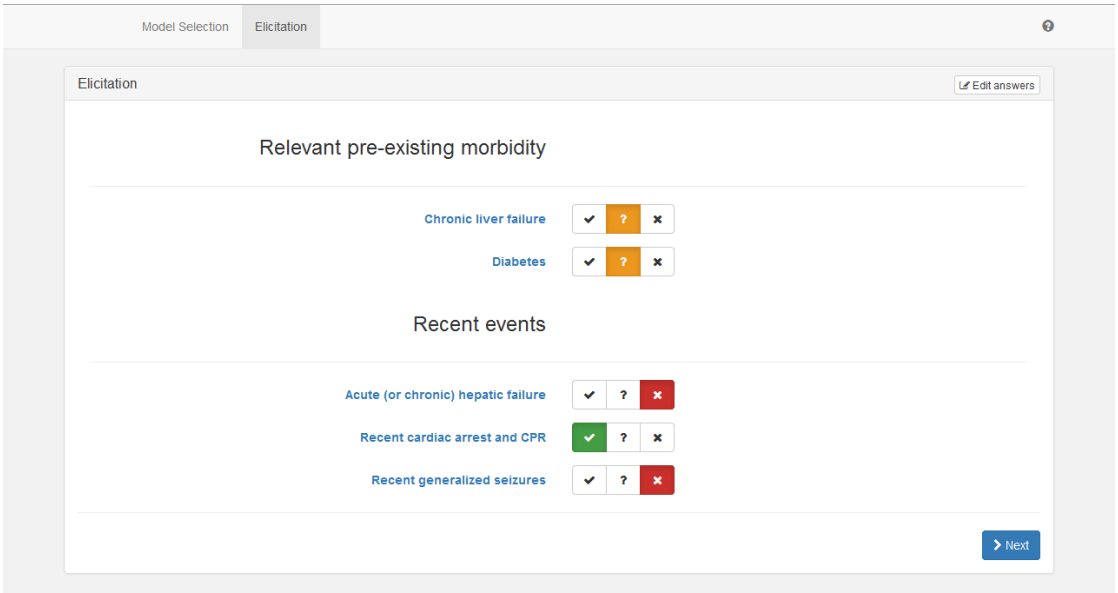

Figure 4.5: An elicitation page with yes/no questions.

The answer fields are automatically error checked. So entering a text in a numeric field, or entering a number that is out of range, results in a red error message being displayed underneath the answer field, explaining the problem. The '*Next*' button is unavailable while there are unresolved errors, to stop users from accidentally (or purposefully) submitting invalid answers.

Once the user has answered all questions, they can commit these answers by clicking on the next button, which submits the answers to the expert system. After this, the user either presented with the next set of questions, or brought to the results screen, depending on the response from the expert system.

In the event the user submits an incorrect answer and wishes to correct it, there is a an option - labeled '*Edit Answers*' - available to let them review all current answers[2], and correct them as necessary. This option becomes available after the first set of answers has been submitted, as there would be nothing to correct prior to that. When reviewing the given answers, the user has the option to commit any changes made or cancel, should they change their mind. After either committing or canceling, the elicitation process is resumed.

The elicitation process continues for as long as the expert system has not reached an end criteria. Once an end criteria has been reached, the user is presented with the results screen.

[2]This was added after the pilot of the evaluation study

Results

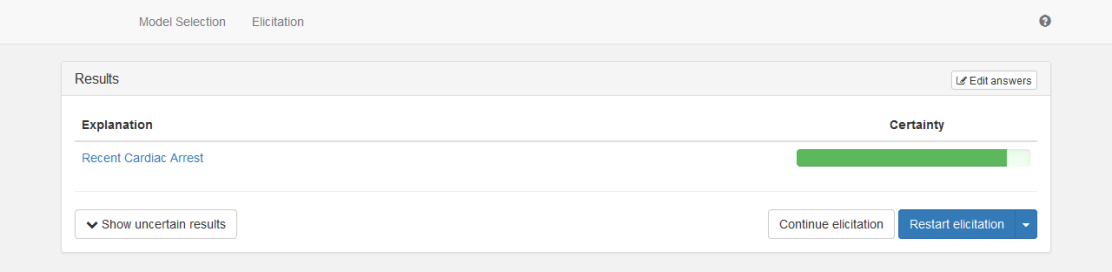

Figure 4.6: The results page.

The results page provides the user with an overview of all results and their certainty. Since there can be any number of results, they are presented as a table, showing the name of the result to the left and the certainty, as a bar, to the right. These are ordered in descending order of certainty, so that the most certain result is displayed first. Figure 4.6 shows an example of the results page with one result. In this figure, the certainties are represented using a single green bar, showing the certainty factors in the range of 0 to 1. Depending on the model's settings, the certainty can also be displayed using two bars instead, adding a red bar for negative certainty. Enabling this allows users to not only see the results the expert system considers likely, but also results it considers unlikely.

The results screen provides the user with several options. As during elicitation, the user can still choose to review their answers and correct where necessary by clicking the '*Edit answers*' button, and doing so will either return the user to the elicitation screen with new questions, or update the results. If the user is satisfied with the answer, they can leave the app by simply closing the page, or returning to the model selection screen. However, should they wish to repeat the model, e.g., for a different patient, there is an option to restart elicitation. This will discard all answers and start the elicitation process anew, and saves users the hassle of having to through the model selection process again.

In the event the user is not satisfied with the explanation given, and there is still the possibility of additional explanations (i.e. there are unanswered, relevant questions remaining), the user can instead opt to continue elicitation. Doing so marks the current result as unsatisfactory and resumes elicitation. This will continue until either the result has changed, or no further questions remain.

The results page normally only shows results that are relevant, i.e., they have a non-zero certainty, or, in the event of the certainty being presented as a single bar, a positive certainty. The user might find it of interest to view all possible results instead of only seeing the relevant subset. To allow this, the results screen provides a '*Show uncertain results*' button which will toggle the filtering of irrelevant results.

Last, but not least, the user may wonder why a particular result as been presented, and may wish to see the reasoning of the expert system. This can be done by clicking on any of the results. Doing so brings up the reasoning screen, showing the reasoning for that particular result.

Reasoning

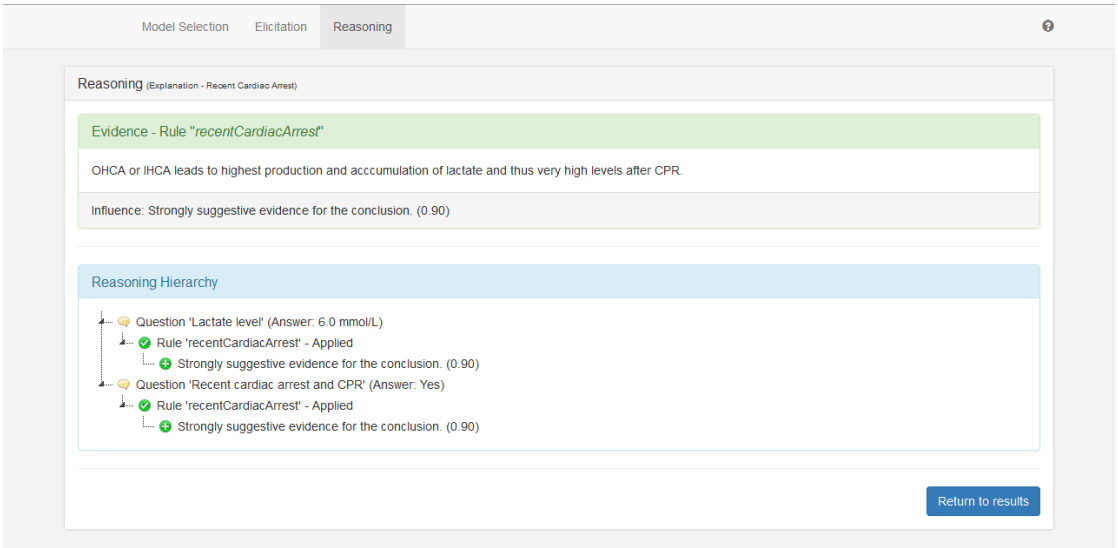

Figure 4.7: The reasoning page.

The reasoning page (shown in figure 4.7) provides insight in the reasoning process of the expert system for a particular (goal) statement. It lists the rules and prior beliefs that directly support the statement in question, including the description of said rule/prior belief (if provided) and the influence on the final certainty. The phrasing used for the influence is based on that of MYCIN, such as "*Strongly suggestive evidence for the conclusion*" or "*Weakly suggestive evidence against the conclusion*".

In addition to listing the direct evidence, the page also provides the reasoning hierarchy, which gives the user insight in different lines of reasoning that have led to the conclusion. This hierarchy starts at the relevant questions – the questions that have had an influence on the certainty of the result – and lists all rules in the rule chain leading to the explanation in the form of a tree. The same is done for prior beliefs that are part of the model. Clicking on any question, prior belief or rule in this hierarchy will display their description. Because the hierarchy is displayed as a tree, with the roots being the relevant questions, it is possible for the same (set of) rules to be listed under multiple questions, as depicted in figure 4.7.

4.2 Editor

In order for the front end to be of use, it requires (a set of) models. The editor facilitates the creation and modification of these. It provides the means to create the rules, questions and prior beliefs, set end criteria and model information, and provides to check for errors and visualize the model structure.

4.2.1 Design considerations

The editor has a different user base than the front-end. Where the front-end needed to be easily accessible for a wide audience, this does not apply as much to the editor, which is intended to

be used by experts. As such, the editor is not web-based, but is instead a desktop application. While it was made to be cross-platform, meaning it can run on Windows, Linux and Mac OS X operating systems, it is not designed - nor currently capable - of running on mobile devices.

4.2.2 User interface

To make easy for new users to get started with editor, it is designed to match the style of typical editor-style application. The interface contains 5 tabs, labeled *Model*, *End Criteria*, *Prior Beliefs*, *Rules* and *Questions*. Each of which provide the means to edit a different part of the model. Models can be opened and saved using the File menu, or by clicking on the corresponding icons in the toolbar, and to help model makers recover from possible mistakes, the editor is fitted with undo/redo capabilities.

This section will cover the process of making models and the features of the editor in greater detail.

Model

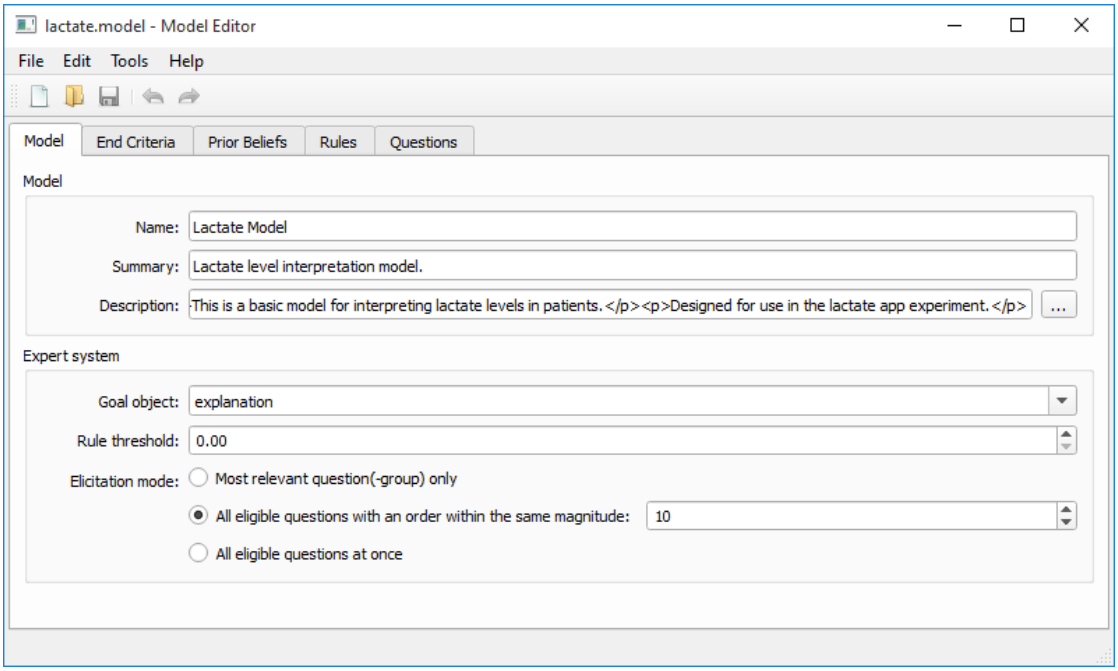

Figure 4.8: The editor's main window.

The model tab (Figure 4.8) provides the user with the meta information of the model – the name, summary, and description – each of which can be freely edited. The description of a model is allowed to contain custom formatting (e.g. bold, italics, etc) by using HTML tags. To simplify the process of applying formatting, the description can be edited with a rich text editor dialog which can be brought up by clicking the '...' button next to the text field. This editor provides a *What You See Is What You Get* (WYSIWYG) visualization of the text, and allows styling of the text in a fashion similar to simple word processing software.

In addition to the meta information, this tab also contains the settings for the expert system. The most important setting here is the goal object, which is the object all possible results are grouped under (see section 3.12.2). Without a goal object, the expert system will be unable to present any questions. To aid the model maker is specifying the goal object, the text field provides a dropdown list of all currently known objects in the knowledge base, which is automatically updated when rules, questions or prior beliefs are added, as well as providing auto-complete while typing.

If desired, the model maker can also specify the rule threshold here, which is the minimal certainty rule premises are required to have in order for the rule to be applied. By default this threshold is set to zero.

Finally, the elicitation mode can be specified here as well. The elicitation mode gives control over how questions are presented to the end user, and currently provides three modes:

- Display the most relevant (group of) question(s).
- Display the most relevant questions whose order are within the same order of magnitude.
- Display all eligible questions.

The first option – selected by default – will present the end user with the single most relevant question when performing elicitation. If this question belongs to a group, all questions belonging to this group, and the name are shown in addition. Alternatively, the model maker may opt to display all eligible questions at once, which can be achieved by selecting the third elicitation mode.

These two options are extremes, showing either the fewest, or the most questions at once. The second option provides a middle ground between these two, allowing multiple (groups of) questions to be displayed in tandem based on their order. Which orders are shown together is controlled by the magnitude setting. During elicitation, the order of the most relevant question is divided by the magnitude and rounded down. All questions whose order, divided by the magnitude and rounded down, equal to the same number are shown together. To illustrate, if the magnitude is set to 10, questions with orders 0 through 9, 10 through 19, 20 through 29, etc., are shown together. If the most relevant question has an order of 4, for example, all questions with order 0 through 9 are presented at once. If any of these questions belong to a group, the entire group is included as well.

End criteria

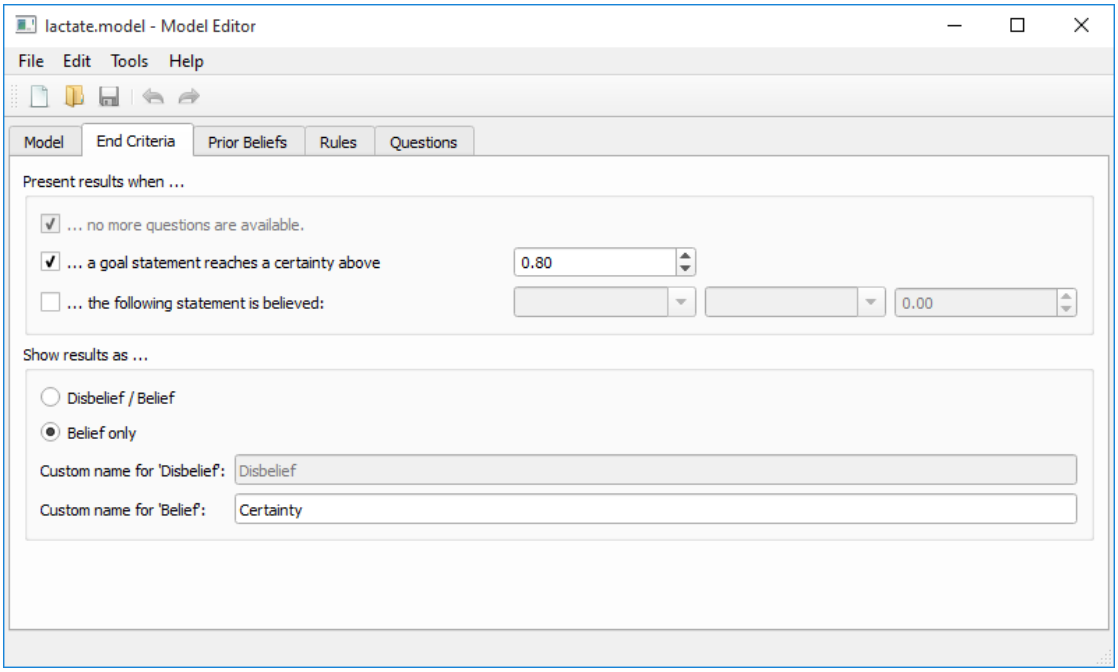

Figure 4.9: The end criteria settings.

The end criteria tab (Figure 4.9) allows the model maker to control the terminating conditions for the elicitation process. Currently, the expert system provides three different (configurable) end criteria:

- No more questions of interest remain.
- A goal statement reaches a certainty above a specific threshold.
- A specific (non-goal) statement reaches a certainty above a specific threshold.

The first end condition – no questions remain – is important to ensure the end user never encounters an empty set of questions during elicitation. It is therefore always in effect and cannot be disabled. To minimize the number of questions presented to end users, the model maker may opt to end the elicitation process early should the expert system find a sufficiently certain result, instead of exhausting the remaining questions first. This can be achieved by enabling the second end condition. Finally, should more fine-grained control be desired, custom end conditions can be modeled using a set of rules. By using the third end condition, a custom statement can be specified as end condition, which can be used by these rules to signal the end of elicitation through any kind of logic desired, provided this logic can be modeled using rules. For convenience, entering a statement for the third end condition is aided by a dropdown list of currently known objects and properties, as well as auto-complete whist typing.

In addition to specifying the end criteria, this tab also provides some control over how the results are displayed, by providing the option to display whether or not to show a disbelief bar, and to specify custom names for the terms ‘belief’ and ‘disbelief’ if desired.

Knowledge lists

The remaining three tabs provide the model maker with an overview of all knowledge in the knowledge base of the model. These tabs contain the prior beliefs, rules and questions respectively. In each of these tabs, the information is presented as a table, where each row provides a summary of a particular prior belief, rule of question. Also, in order to quickly see how many entries each table has, there is an indicator above them showing how many entries there are.

As the three types of knowledge are very distinct from one another, each table contains a different set of columns to provide an informative summary for the respective knowledge type. For example, the rule table shows the rule ID, the number of statements in the premise and conclusion, and the rule description (see figure 4.10). Questions, on the other hand, show the question ID, type (yes/no, multiple choice, etc), question text, order, tool-tip, etc. (see figure 4.11).

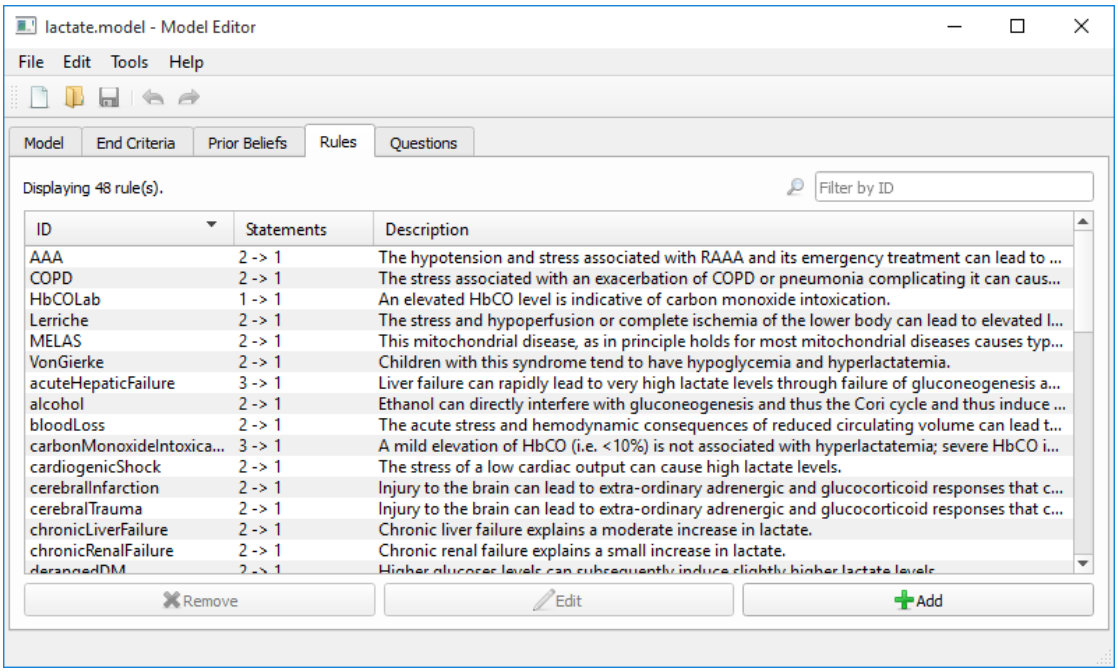

Figure 4.10: The rule list.

Knowledge bases can become rather sizable, so to make it easier to find particular entries in these lists, the tables are sortable by column. This can be done simply clicking the respective column. Clicking on them repeatedly will toggle between ascending and descending order. Additionally, in order to find a specific entry, the tables can be filtered by entering the ID (or part thereof) in the filter text field above the table, which will hide entries whose ID does not contain the text entered.

In order to edit the knowledge base, there are three buttons underneath the table which to add, remove and modify entries. As an added convenience, entries can also be edited by double clicking on them in the table.

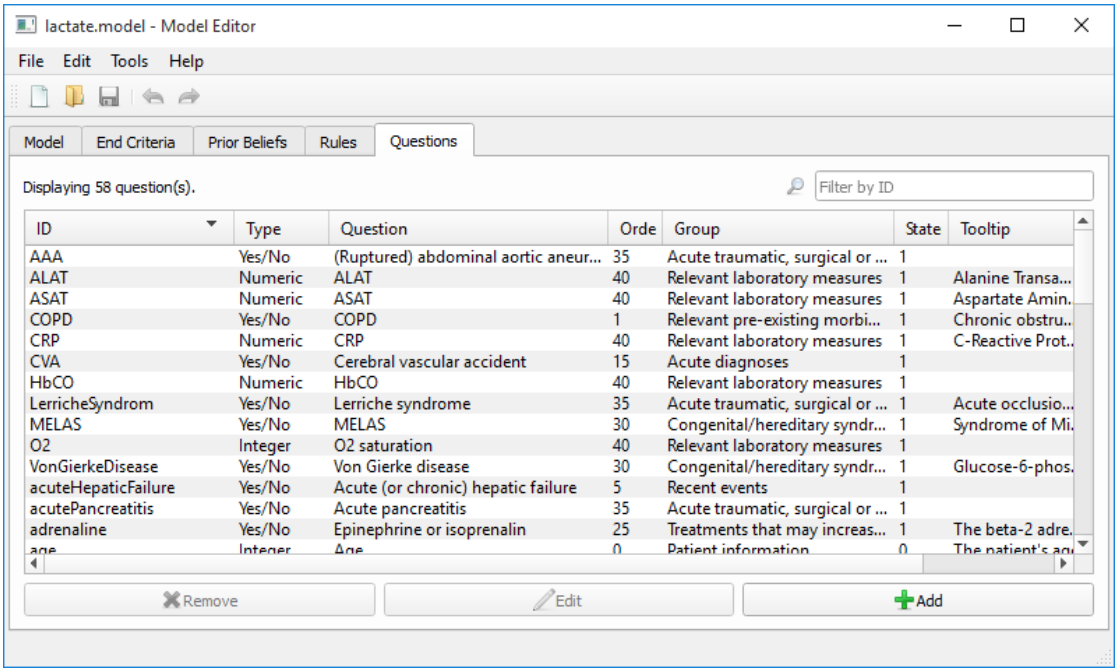

Figure 4.11: The question list.

Rules

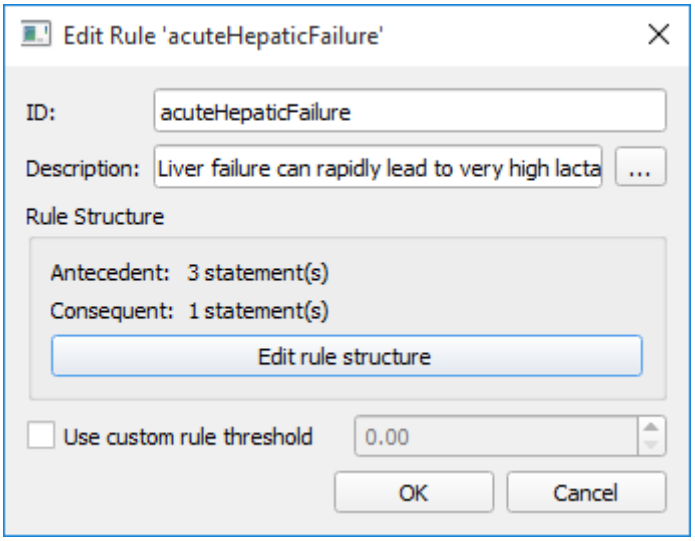

Figure 4.12: Rule editor dialog.

When creating or modifying a rule, the user is presented with the rule editor dialog, shown in figure 4.12). Due to the fact that rules can be made arbitrarily complex, this dialog only deals with the meta information of the rule, i.e., the ID, description, etc. Like the description of the

model itself, rule descriptions can also be formatted if desired. This can, for example, be used to embed links to literature in the descriptions. While the dialog does not show the rule itself, in order to still give some information about the rule, the number of statements in the premise and condition – labeled antecedent and consequent – are shown instead.

There may be cases where model makers may want specific rules to use a different certainty threshold – the minimal certainty for the premise required to apply the rule – than the default specified for the model. Enabling the custom rule threshold option allows the model maker to specify the exact threshold for that particular rule.

The most important part of rules is, of course, the rule structure. This is displayed in a separate dialog, which is shown when the user clicks on 'Edit rule structure' (figure 4.13).

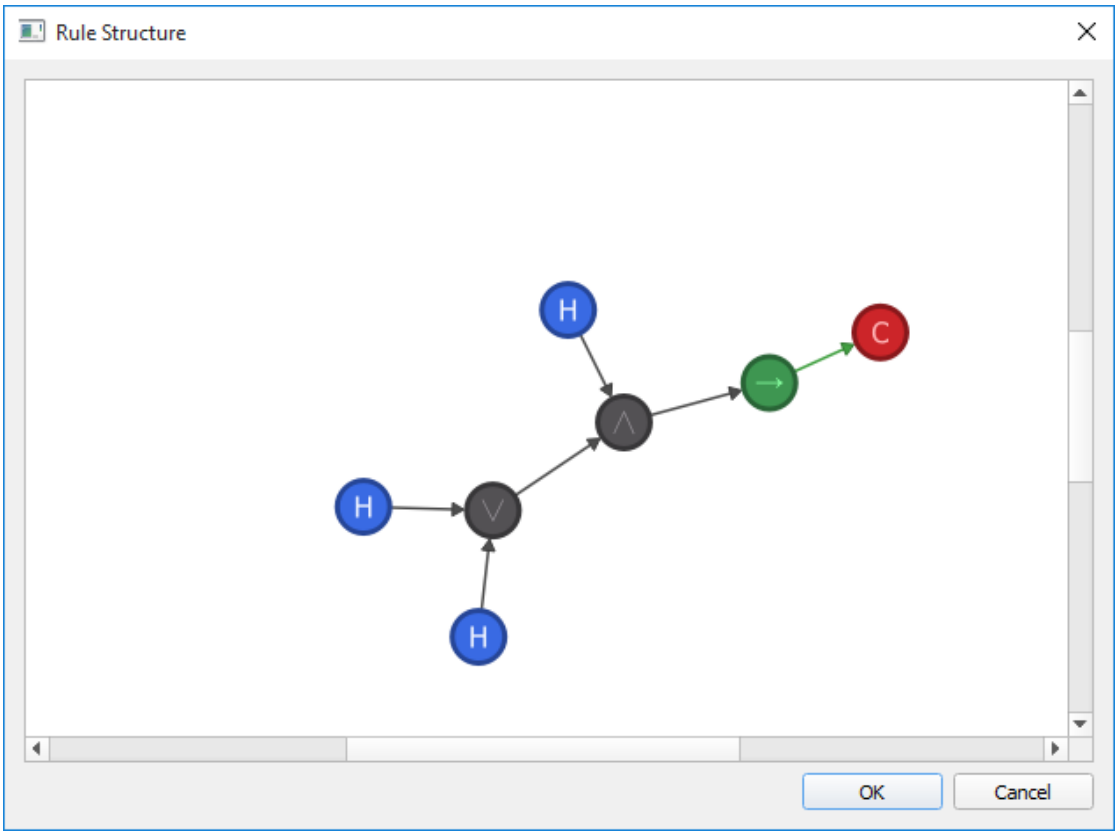

Figure 4.13: Rule structure dialog.

The rule structure editor provides a graphical representation of rules, in the same fashion as they were presented in the system design chapter: as a graph with nodes and edges. Each node represents one component of a rule, be it part of the premise or a conclusion, and the connections indicate how they are related. To easily differentiate these nodes, each type has a different color and symbol combination as seen in figure 4.14. The choice of using both colors and symbols reduces the number of colors required to represent all types, makes it more clear at a glance what the type is (e.g. H stands for **H**ypothesis), and limits problems in the event the user is color blind. Some nodes have additional information associated with them, such as the statement in the case of an hypothesis node. To keep the visualization consistent, this

information is not shown directly. Instead, hovering the mouse over nodes shows their type and additional information.

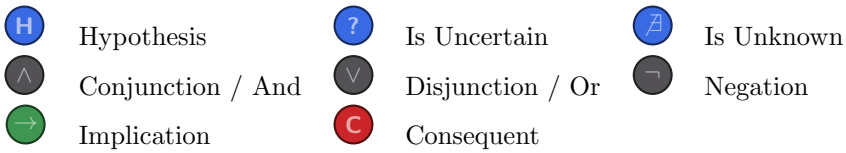

Figure 4.14: The different types of nodes in the rule structure dialog

The nodes can be moved around freely, and are automatically placed into a pseudo-optimal layout using a force directed graph algorithm. These are based on the physical phenomena of attraction and repulsion, where each node causes a repelling force on all other nodes, and the connections act as springs holding everything together.

In this visualization the premise and conclusions are tied together through the implication node, which is also the only node shown initially when creating a new rule. Modification of the rule structure is possible by right-clicking on a node. Doing so brings up a pop-up menu allowing for nodes to be added, removed or modified. This menu takes the structure's constraints into account. For example, it is not possible to add a new node to a terminal node, i.e., an hypothesis, uncertain or unknown node. Similarly, conclusions can only be attached to the implication node.

While the premise can be made as complex as desired, it must eventually converge into a single node, which is then attached to the implication. Figure 4.13 shows an example of this, where three hypothesis tests are combined through a disjunction and conjunction node. There is also an option to wrap a node into a non-terminal node (i.e. conjunction, disjunction or negation). This makes it easy to, for example, add a negation to an hypothesis node that was already added, without having to first remove it, add a negation node, and then add the hypothesis node again. The opposite is also possible. For convenience, nodes can be converted into any other node of their type (terminal/non-terminal). So it is possible to change a conjunction into a disjunction, without having to rebuild the tree. This is only possible if the change does not violate any of the constraints. For example, you cannot convert a conjunction with two child nodes into a negation, since negations can only have a single child.

Committing changes to the rule structure is done by simply closing the structure dialog, which will return them to the dialog with the meta information. In there, the changes can be committed to the knowledge base by clicking 'OK', or discarded by clicking 'Cancel'.

Questions

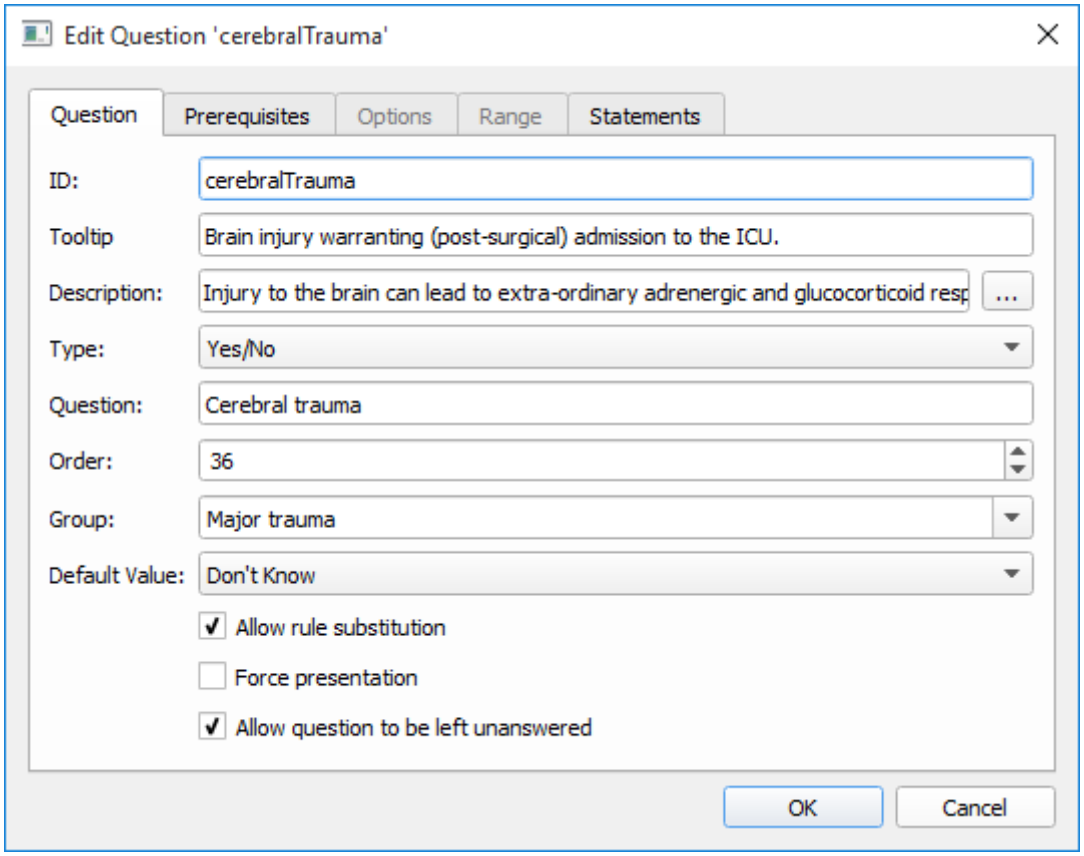

Figure 4.15: Question editor dialog.

When creating or editing questions, the user is presented with the question editor dialog (figure 4.15). This dialog provides the means to edit all aspects of the question, split up into several tabs. The main tab, shown in the figure, contains the most important parameters. This includes the question ID, type, text, tooltip, description, order, group and default value. These purpose of these options have been previously and should be self-explanatory.

In addition to these options, there are three extra options shown as checkboxes at the bottom. The first option is to allow rule substitution. As covered in the design chapter, this allows rules to essentially override the question. This is primarily useful for questions whose 'answers' can be inferred based on other questions. An an example, if the patient does not have diabetes, then one can infer that the patient also does not have diabetes-related conditions. The requirement to explicitly allow substitution is there to prevent rules from this inadvertently. The second option is to force the presentation of the question. Enabling this marks the question as relevent regardless of the results of backward chaining. This can be used to present questions that are not relevant for the outcome of the model, such as a patient ID. The last option allows to remain unanswered, and will present the end user with a "I don't know" button for that question.

Model makers may want to ensure a question is only presented after one or more prerequisite questions have been answered first. The prerequisites tab allows for this do be done. In there,

the model maker can provide a set of question (IDs) that are considered prerequisites. During elicitation, if the question is considered relevant but one or more prerequisites have not been presented yet, the relevance is applied to the prerequisite questions instead. This can have interesting consequences. Imagine two questions, A and B, where A is the prerequisite for B. In the event the expert system considers B a relevant question, but not A, then A will still be presented to the user due to this prerequisite relationship.

The remaining three tabs differ depending on the question type. The third tab, options, is only available if the question is a multiple-choice question, and allows the different choices to be specified. Similarly, the fourth tab, range, is only available for numeric and integer questions, and allows specification of minimum, maximum and default values, the number of decimal spaces in case of a numeric value, and the units if applicable.

From the perspective of the expert system, a question is only useful if it can acquire knowledge from it. The final tab, statements, allows for the specification of knowledge that can be obtained from the question. The way statements are added depends on the type of question. Yes/no and multiple choice questions provide a table, where the columns are all possible options and the rows are the statements. The certainty for all combinations can be entered into this table. Like the prior/rule/question lists in the main window, there are three buttons to add, modify and remove statements.

For numeric questions, this is presented differently. Each statement can use one of two membership functions: a threshold or a range function. In both cases, two certainty factors must be provided: the certainty if the input value is below the threshold, and the certainty if it is above the threshold. In case of a range, these become outside and inside of the range respectively. Since this cannot easily be presented in a table, this is done in a separate dialog shown in figure 4.16.

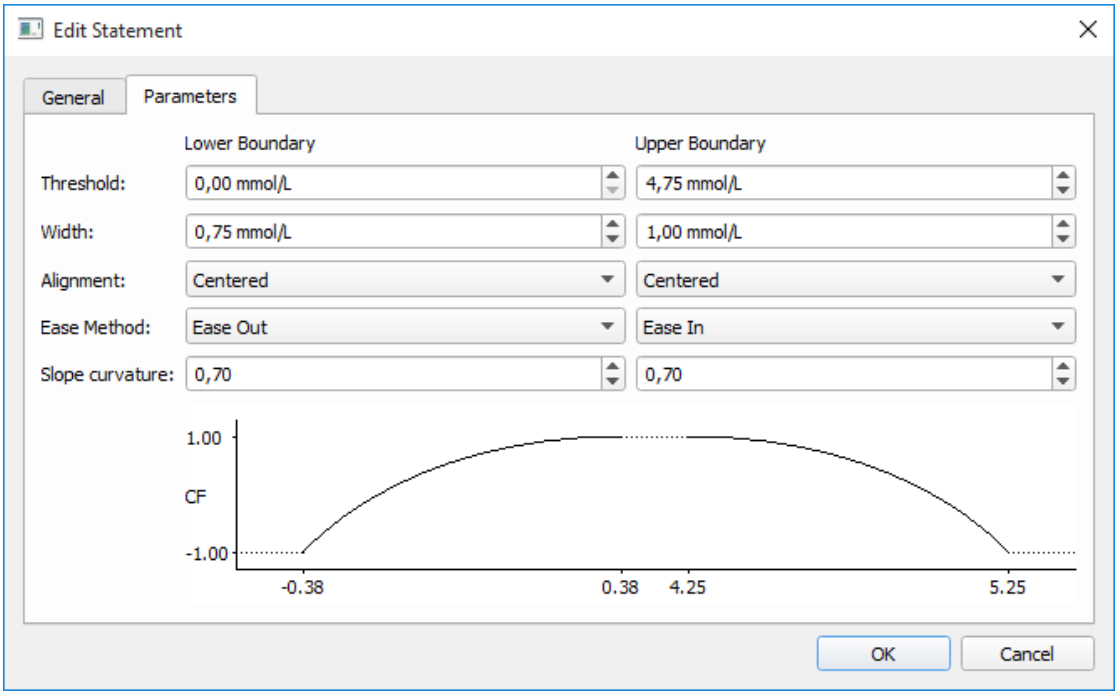

Figure 4.16: Numeric statement editor. Currently showing a statement with two thresholds (i.e. a range).

The parameterization of the boundaries provides control over how the certainties change when going from one side of the boundary to another. This can either be an abrupt change from one certainty to another (i.e. a step function), by setting the width to 0, or a more smoothened transition, by setting the width to a value higher than 0. When using a smoothened transition, the resulting certainty factor gets interpolated if the input value is within the range of $threshold \pm \frac{width}{2}$. The actual interpolation method used can be set by specifying the easing method and curvature, as covered in the system design chapter. To give model makers insight in what the parameters do, the dialog provides a schematic (i.e. not to scale) graph, showing the input values and corresponding certainty factors. These are automatically synchronized with the parameters, making it easy to see the effect of any changes.

Validation

The system design chapter has covered that the expert system contains a validation system, preventing possibly erroneous models from being loaded and causing problems. While this is good for the end user and expert system – models that run are guaranteed to be structurally sound – it can prove problematic for model makers if their model refuses to load. To solve this, the validator is available within the editor. Running the validator will not only state whether the model is valid or not, but also states exactly what problems were found, and which severity they have. These problems are reported with enough information to allow model makers to find and resolve the problem, and after a possible fix, the validator can be run again to verify the problem has been solved.

Visualization

In a knowledge base, the prior beliefs, rules and questions are all connected. As models grow bigger these interactions can become more complex, making it more difficult to keep track of how everything interacts and troubleshoot problems. In an attempt to assist with this, the editor provides the option to visualize the knowledge base. This visualization, shown in figure 4.17, looks very similar to the rule structure editor, and contains all prior beliefs, rules, questions and the statements that connect them together. To help distinguish between 'regular' statements and goal statements, they are both represented using a different color and symbol. Like the rule structure editor, these are animated using force directed graph algorithms, meaning that the nodes will automatically move towards an optimal layout. As with the rule structure dialog, nodes can be dragged around and the view can be panned and zoomed in and out freely.

Since the visualization typically contains a large numbers of nodes, it can take some time for animation to stabilize. To deal with this, there is an option available to 'fast forward' to a stable state. This uses a slightly different force directed algorithm that reaches a stable state quickly, but is not suitable for real-time animation.

This visualization provides a broad overview of the model and how everything is connected. It also provides insight in elements of the model that are not being used. For example, questions that are not actually used in rules will be floating around on their own. Similarly, statements that are provided but not used will not have any connections coming out of them. This capability of seeing the whole knowledge graph as a whole makes it much easier to spot problems than having to go through every rule or question individually.

The visualization can be pruned by removing 'floating' nodes (i.e. nodes that are not connected to a goal statement), or by removing unused statements. In addition to visualizing the full model, it is also possible to visualize subsections of the model, by focusing on a specific

statement, rule, question or prior belief. This will only show the part of the knowledge base that has a connection to the focused element.

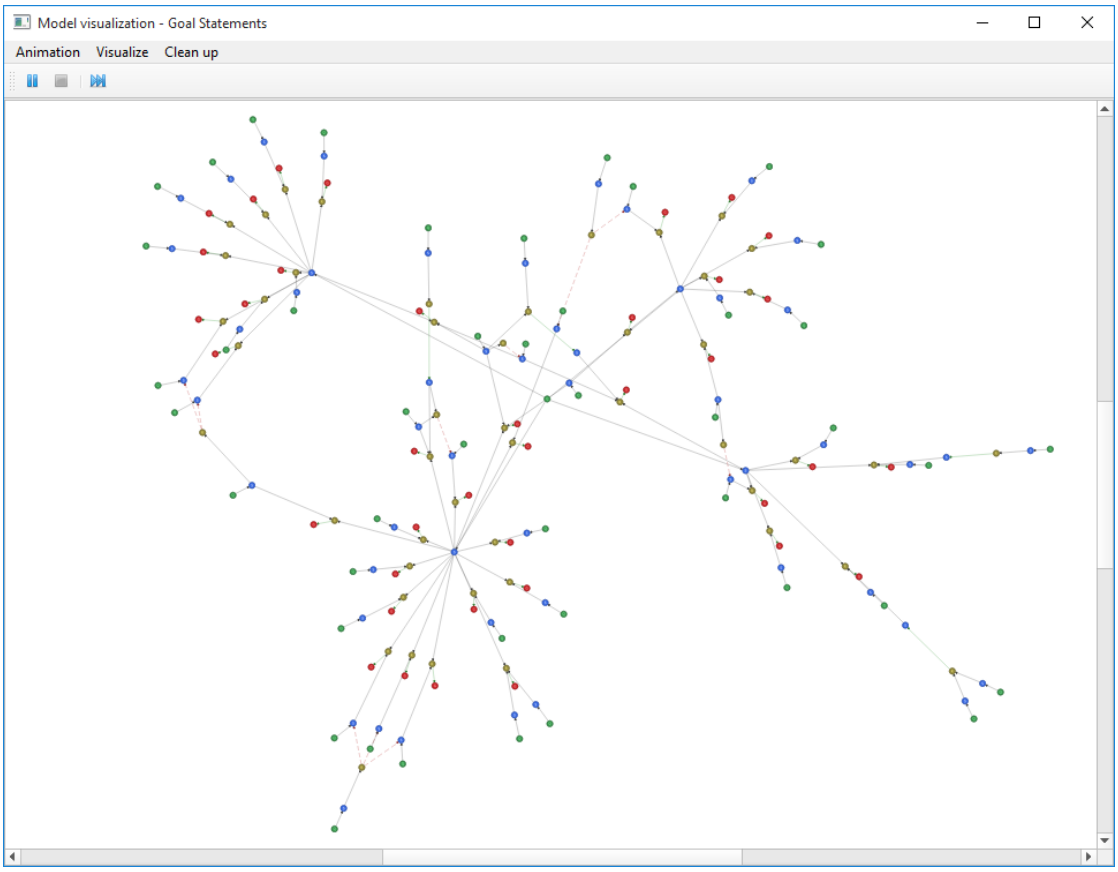

Figure 4.17: Model visualization dialog.

4.3 Lactate model

The model used in the study is a simplified model of lactate. It is based on a list of possible causes of increased lactate provided by an expert clinician at the University Medical Center in Groningen. These possible causes were grouped and annotated with the degree of lactate elevation they can cause. This degree of elevation was split into five categories:

Table 4.2: Categories for lactate levels and corresponding lactate level in provided list and model.

Category	Lactate level	Upper boundary in model
Not directly relevant	1.5 mmol/L baseline	*N/A*
Small to minor increase	+2 mmol/L	4.75 ± 0.5 mmol/L
Minor increase	+4 mmol/L	9.0 ± 0.5 mmol/L
Minor to big increase	+8 mmol/L	11.0 ± 0.5 mmol/L
Major increase	+ 10 mmol/L	30 ± 10 mmol/L

The model was given four starter questions, asking the user for basic patient information: age, sex, ID and lactate level. The lactate level provided by the user is classified into four categories, whose boundaries are based on the table above: Small minor, minor, big minor and major elevations. For each of these classifications, a certainty factor is calculated based on how well the lactate level fits within its respective boundaries. These boundaries are, as shown in table 4.2, ranges rather than fixed points. Within the boundaries, the classification will have certainty of 1, but if the lactate level lies within (or beyond) the upper boundary range, the certainty is interpolated between 1 and -1 using the easing curves shown in figure 3.5 on page 21.

The model makes the assumption that small increases in lactate can still have causes are normally attributed to higher increases. For example, a recent resuscitation is normally associated with a major increase in lactate, but it can still be an explanation for a mild increase. The opposite, however, is not the case; causes that can only explain mild increases for lactate cannot explain a major lactate increase. As such, all classifications have a lower boundary near 0 and an upper boundary as shown in table 4.2. Consequently, the lower the lactate level, the more possible causes are taken into consideration, and vice versa.

The possible causes for lactate were modeled in a fairly straightforward fashion. All explanations, with the exception of lab measurements, were added as yes/no questions, with an associated rule that states that if the patient has said particular cause, and the lactate is within the expected range, then said cause is a possible explanation. For lab values, numeric questions were used instead, which were then classified. These classifications include normal, elevated, reduced, etc., and were used in rules similar to the one mentioned previously.

In a few cases, lab measurements were used as a means to double-check other answers. For instance, when the user states the patient does not have liver failure, but the lab measurements provided suggest otherwise, liver failure is still presented as a possible explanation.

As stated previously, the list with causes was not just annotated with their effects on lactate, but also grouped different related causes together. This was used to group the (at time of writing) 50 questions in the model into the following groups, in order:

Table 4.3: The question groups and the number of questions in each group. Numbers in parenthesis indicate conditional follow-up questions.

Group	Number of questions
Patient information.	4
Relevant pre-existing morbidity.	5 (+3)
Recent events.	3
Acute signs.	3
Acute diagnoses.	8 (+4)
Acute intoxication.	3 (+1)
Treatments that may increase lactate.	2
Congenital/hereditary syndromes.	2
Acute traumatic, surgical or vascular syndromes.	4
Major trauma.	+2
Relevant laboratory measures.	6

This order was chosen as it reflects the order in which clinicians typically receive information about their patients. Some of these groups contain follow up questions which are selectively presented depending on the answers given. For example, in the *relevant pre-existing morbidity* group, the user may be asked if the patient has diabetes. If the answer to this is yes, an additional set of diabetes-related questions is presented. To facilitate this, substitution rules (see section 3.9.3 on page 22) were used. These rules, which fired if a particular pre-condition was met (e.g. the patient does not have diabetes), and subsequently provided 'answers' for the follow-up questions, thereby preventing them from being elicited.

A complete list of all rules and questions used in the lactate model can be found in appendix B on page 76.

Chapter 5

Evaluation Study

5.1 Method

To evaluate the application, an evaluation study was conducted at the University Medical Center Groningen. The purpose of this study was to evaluate the performance of the app and to assess the interest in using this type of decision support tool for finding explanations for lactate levels and possibly other users.

5.1.1 Experimental setup

The study involved ten participants whose results were anonymized, all of whom work at the Intensive Care Unit of the hospital. Seven were resident intensivists, and three were student-researchers. For the evaluation, a set of ten vignettes were created. These vignettes describe a fictional patient by providing different types of information, including the patient's sex, age and lactate levels; prior medical history; acute symptoms and lab values. In order for these vignettes to be effective in this experimental setting, they were all created to work correctly with the app. More specifically, the vignettes were created by taking an explanation (or set thereof) the app could arrive at and building a patient description around them, whist ensuring the app could find the intended explanation with the information provided. The creation of the patient description was done by an expert intensivist working at the hospital, and co-supervisor of this study. These descriptions reflected the type of information the participants would encounter when dealing with real patients. The full set of vignettes (and their intended explanations) can be found in appendix A on page 65.

The experiment consisted of two parts: finding explanations for the lactate levels in vignettes with and without the app. Prior to beginning the experiment, the participants were given instructional texts explaining the setup of the experiment, and were asked to sign an informed consent form. The instructional texts and consent form can be found in appendices E D on pages 114 and 112.

The goal of this first part was to determine how well the participants were able to find the intended explanations without assistance, and to assess the reasoning process involved. The participants were presented with the vignettes, on at a time, in random order, and asked to find an explanation for the elevated lactate. They were asked to do this whilst thinking out loud and give their answers verbally, while they were being recorded by a microphone. The actual order in which the vignettes appeared was written down so that the recordings could be analyzed later. Participants were not given any feedback as to whether their explanation was the intended

one, nor were they penalized for giving an incorrect answer. They were also informed that their performance would not be disclosed to their peers or superiors.

For the second part of the experiment, the participants were given brief instructions on how to use the app, which was running in a browser on a computer in front of them. After the instruction, they were once again presented with the ten vignettes, but in a different (pseudo-random) order. As before, they were asked to find an explanation for the lactate, whilst thinking out loud and being recorded, but now using the application instead. When presented with an answer, the participants were asked whether they agreed with the result. If they did not, they were given the opportunity to correct answers or proceed with elicitation if they so desired. These options were added after the pilot study revealed that participants occasionally gave incorrect or unexpected answers to questions, which led to the presentation of an incorrect or unsatisfying answer and required the participants to restart elicitation entirely to correct this.

Once completed, the participants were given a questionnaire with 18 Likert scale questions, which included 8 of the 10 questions from the System Usability Scale (SUS). This scale provides a "quick and dirty" tool for measuring usability (Brooke, 1996). In addition, there were questions focusing more specifically on the app, trust in the underlying system and suitability for training and clinic use. The full questionnaire can be found in appendix F on page 116. After the questionnaire, the participants were given a brief interview, in which they were asked a series of questions about the experiment. This interview included questions such as their thoughts on both parts of the experiment, whether they enjoyed using the app, their opinion on the app and the model, whether they find such an app suitable for practical use and/or training, suggestions for improvement, etc. The full set of questions can be found in appendix H on page 118.

5.1.2 Data analysis

For the unassisted part of the experiment, the responses were analyzed by replaying the recordings and writing down the answers given for each vignette. Since the actual order in which the vignettes were presented was logged, these answers could be compared to the intended answer(s) of the vignette in question.

For the assisted part, the app's web server was running in 'experiment-mode'. In this mode, it automatically saved a dump of the inference engine's working memory when the results page was shown to the user. These dumps contain all current beliefs of the inference engine, and all information elicited from the user (i.e. all answers provided). These dumps were grouped by participant and later aggregated into a dataset containing all answers provided by the participants and all explanations provided by the app. In cases where the participants restarted, corrected answers or continued elicitation after the initial results were shown, multiple dumps were created. This made it possible to track how many 'attempts' were used to solve a vignette.

To assess the accuracy of the answers, each vignette has one or more intended explanations. These explanations were split into primary and secondary explanations. The primary explanation was the intended main cause for the elevated lactate, and were what the participants were tasked to find. Secondary explanations are alternative explanations which, on their own, only provide a partial explanation. The intended explanations can be found in appendix A, where the secondary explanations are written in parentheses.

The answers given by participants were given either 1, 0.5 or 0 points. If a participant mentioned one of the primary explanations, they received the full point. If only secondary explanations were mentioned, they received 0.5 points, and 0 points otherwise.

To get the final results, all scores were aggregated into a table containing each participant and each vignette, with the corresponding score for each combination. Each participant was given a final score: the sum of all vignette scores, divided by the number of vignettes, presented

as a percentage. This table was made for each of the two parts of the experiment, so scores could be compared between the two parts.

In addition to the accuracy of participants, their speed was also compared between the two conditions. Due to the time scales involved in the experiment, the average duration per vignette for participant was approximated by taking the length of the corresponding recording, corrected for any introductory text, and divided by the number of vignettes presented in that recording. Consequently, only the mean time per vignette was compared between the two conditions, not the duration of each individual vignette.

To process the interviews, the recordings were replayed and the responses written down. However, instead of doing a full transcript, all answers were summarized into a series of bullet points. This made it easier to see they key points of the responses, and was also much less time consuming.

5.1.3 Complications

Two participants were unable to complete all vignettes, as they were required to leave early due to work-related obligations. One participant has only done 7 vignettes in each condition. For consistency, and later statistical analyses, the same 7 vignettes were used in both conditions. Another participant had done all vignettes in the unassisted condition, but was cut short in the second condition after having done only 6 vignettes. Despite these two participants not completing all vignettes, they were able to complete the questionnaire and interview that followed.

One vignette had no intended explanation for the lactate, but was rather meant for participants to state they needed more information to find an explanation. However, in both conditions, the participants often did find a possible explanation for the lactate. This caused some problems when determining the scoring for this vignette, since every explanation would have to be considered incorrect, even though the answers in question might not necessarily be wrong. As such, instead of scoring everything as incorrect, it was decided to exclude the vignette from the analysis entirely.

In addition to the aforementioned, there was one additional exclusion. In response to feedback from the pilot, some adjustments were made to the app and model. These modifications incidentally led to an error in the model, that resulted in a critical question for one of the vignettes not being presented. The first post-pilot participant was given an incorrect answer by the app on this vignette. The error was addressed after the participant completed the experiment. No other participants or vignettes were affected by this error.

5.2 Results

5.2.1 Vignettes - Scores

The scores obtained in the evaluation study are shown in table 5.1. Overall, the participants gained an average score of 61% when asked to find the explanations on their own, and an average of 90% when performing the same task with app. This difference was found to be significant through a paired t-test on the final scores of each participant across both conditions ($t(9) = 9.5771, p < 0.0001$). An additional test on the scores of every vignette across both conditions, corrected for unequal variances, confirmed this significance ($t(81) = 5.7593, p < 0.0001$).

Table 5.1: Explanation scores. For each column, the left glyph indicates the score for the explanation given by the participants when unassisted. The right glyph indicates the score for the explanation given by the app.

	Vignette & intended explanation											
	Hyperventilation	Pneumosepsis	MALA	CO intoxication (+ MALA)	Generalized seizures	Cardiogenic shock (+ renal failure)	Inotropes	Respiratory Insufficiency n.o.s Hypotension (+ Diabetic Keto-acidosis)	Acute-on-Chronic liver failure		
Participant	1	2	3	4	6	7	8	9	10	% Correct	
1	●●	●●	●●	●●	●●	●◌	○●	●●	●●	89%	100%
2	○●	●●	●●	◒●	○●	●●	○●	●●	●●	61%	100%
3	○◌	●●	●●	◒◌	●●	●●	○◌	●◌	●●	72%	100%
4	○○	◒◒	●●	○●	○●	◒●	○●	●◒	●●	44%	78%
5	○●	●●	●●	◒●	○●	◒●	○●	●●	●●	56%	100%
6	◌◌	◒●	●●	◌◌	○●	●●	○○	◌◌	●●	58%	83%
7	●●	○●	●●	◒●	●●	◒●	○○	●●	●●	67%	89%
8	○●	◒◒	○○	○●	●●	◒●	●●	○◒	●●	44%	78%
9	○●	○◒	●●	◒●	●●	◒○	○●	◒◒	●●	50%	78%
10	○●	●●	●●	●●	●●	●●	○●	◒◒	●●	72%	94%
Mean	∗			†	∗		†			**61%**	**90%**‡

●: Correct. ◒: Partially correct. ○: Incorrect. ◌: No data.
∗/†/‡: Significant difference between assisted and unassisted conditions.
($* : p < 0.05$, $† : p < 0.01$, $‡ : p < 0.001$)

Vignette 5 is excluded from this table. The answers could not be scored as the vignette had no intended explanation.

Statistical tests on each vignette have found four vignettes with scores that differed significantly between both conditions, which are indicated in table 5.1 with $*$ and †.

The first vignette had the intended explanation of hyperventilation, however many participants thought the high lactate was caused by heart failure. In the CO intoxication vignette, most participants either overlooked the fact HbCO was mentioned in the lab measurements, or were not aware what the normal range was for HbCO. As a result, they did notice CO intoxication was the explanation for the elevated lactate. One participant stated CO intoxication was the cause of the lactate, only because HbCO was mentioned at all and assumed it was therefore very likely to be the sought explanation. A similar problem occurred with the generalized seizures vignette, where some participants thought that seizures by themselves could not be responsible for the lactate. Finally, the inotropes vignette proved difficult for nearly all participants. Most participants were unaware that isoprenaline, or inotropes in general could cause an increase in lactate. and were surprised when the app mentioned they could. Despite that, even with the app, two participants still got an incorrect answer for this vignette when using the app. This occurred because participants sometimes answered earlier questions positively, leading it to present a (premature) explanation. For most participants that encountered this, there were not satisfied with this explanation and chose to continue elicitation, where they were asked whether the patient used inotropes. Two participants however did agree with this premature explanation, were therefore never asked about inotropes and thus did not get the intended explanation from the app. In literature, this phenomena is called premature closure.

5.2.2 Vignettes - Speed

The average duration per vignette for the unassisted condition was 106 (SD: 38.4) seconds, and 156 (SD: 28.7) seconds for the assisted condition, indicating that participants were slower when using the app compared to performing the task unassisted ($t(9) = -6.6012, p < 0.0001$). The results are shown in more detail in figure 5.1. The average duration per vignette was increased for every single participant when using the app.

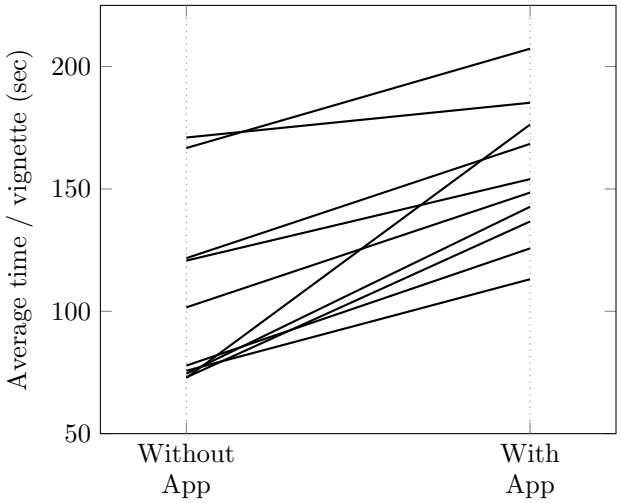

Figure 5.1: Average time the participants took to solve a vignette, both with and without the use of the app. In the left figure, each line represents a single participant. Difference in duration between conditions is significant. ($t(9) = -6.6012, p < 0.0001$)

5.2.3 Subjective measures

The results of the questionnaire are shown graphically in figures 5.2 through 5.5. The circles represent the answers given by the participants, and the size indicates the number of participants that gave said answer. As some of the questions were phrased negatively (e.g. “I found the system very cumbersome to use.”), the answers to these questions have been mirrored to make the resulting figure more consistent, and to more clearly distinguish answers as being positive or negative towards the app. To avoid confusion, some mirrored questions are presented in the figures by their antonym (e.g. “Convenient” instead “Cumbersome”.)

Overall, the results of the questionnaire were positive, with the vast majority of responses being either strongly or moderately positive towards the app.

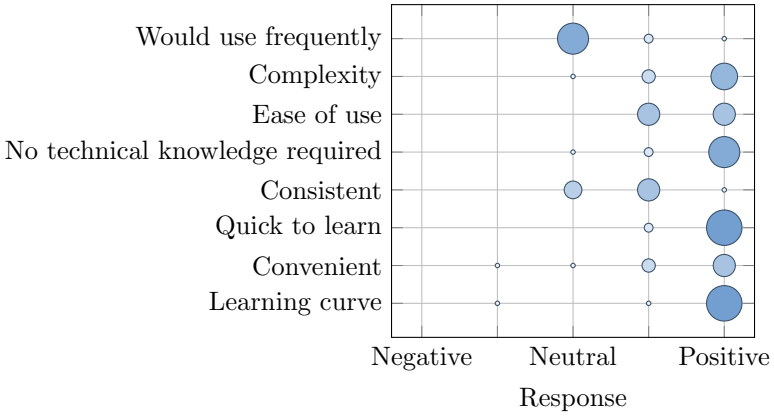

Figure 5.2

The first question (figure 5.2), ’I think that I would like to use this system frequently.’, received mostly neutral answers, and this was elaborated upon in the interview, as well as annotations added to the questionnaire by some participants. The participants stated using the app would depend on the patient in question: simple cases would not require the use of such an app, and would be faster and easier to solve on their own. They did however indicate that for more complex cases, or when there is doubt, using an app to assist would be very useful.

All of the questions in figure 5.2 are from the System Usability Scale (SUS). The answers for this scale can be combined into a total usability score on a scale of 0 to 100, whereby higher scores indicate better usability. Since only eight of the ten questions were used, the calculation of the score was adjusted to compensate for this, and the resulting SUS score was 82.2 out of 100.

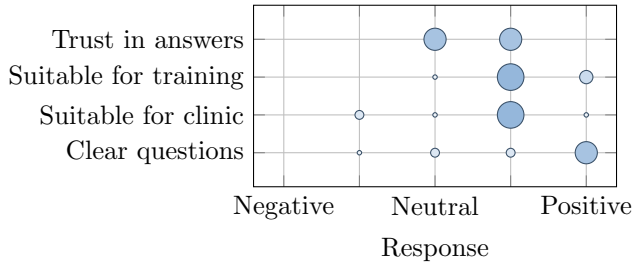

Figure 5.3

The questions in figure 5.3 covered the trust in the system, the explanations provided by the app and the app's suitability for training purposes and use in the clinic. Most participants found the questions to be very clear, however, their trust in the answers given by the application was found to only be neutral to mildly positive. Despite this reservation against trusting the results, they did find the system to be suitable for training purposes, and to a lesser extent, for use in the clinic.

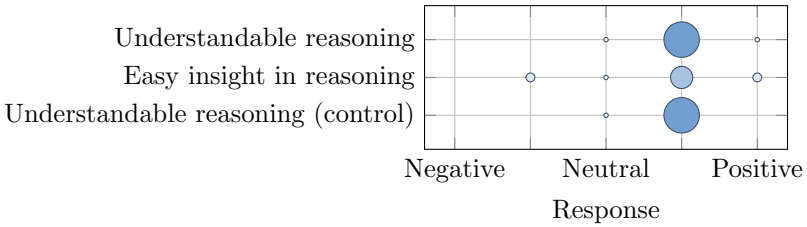

Figure 5.4

The next three questions (figure 5.4) concerned the insight in the reasoning process of the expert system. Here again, the feedback is positive, suggesting that the participants had a good understanding of the expert system's reasoning process, or could at least get sufficient insight into it. Two of these questions were formulated to act as a validation question, being phrased as each other's opposites. The responses to these two questions shows a perfect mirroring of the answers (see appendix), which indicates that the given answers were consistent.

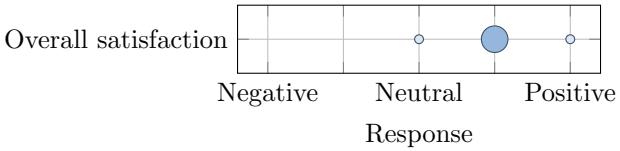

Figure 5.5

Finally, the last question (figure 5.5) asked for their overall satisfaction with the system, which also also received positive feedback. The exact numbers of responses for all questions are available in appendix G on page 117.

Interview

The remainder of this section will provide a summary the feedback of the interview.

The overall opinion of the participants regarding the app was positive. The participants reported enjoying using the app, and finding it easy to use and understand, with little to no need for instructions. They also reported finding the task – finding explanations for lactate levels – easier (and some much easier) when using the app.

There were a few suggestions regarding the user interface. One common complaint was that questions with numeric answers had their answer fields already filled in with a default value (typically 0). Consequently, this default value had to be removed before filling in the answer, which was found to be rather inconvenient. There was also some confusion about the "I don't know" option, the button for which was labeled "N/A". The meaning of that button was not always clear, suggesting that a different, more descriptive label would be more appropriate. It was also suggested to use multiple buttons instead of a dropdown list for multiple-choice questions

with only a few options, such as the patient's set, and to possibly add something akin to likert-scale questions, for questions whose answers are likely to be more nuanced than strictly yes or no.

There were also suggestions and criticism regarding the lactate model. Various participants noted that the lactate model was somewhat simplistic and has the tendency to jump to conclusions. In particular, some participants found that in many cases, answering yes to one of the questions resulted in the system presenting it as an explanation for the lactate level. Not in all cases was this the actual intended explanation, but rather a minor contributing factor that the system considered a sufficient explanation. The consequence of this is that participants occasionally had to use the "Continue elicitation" option to get a full explanation.

In addition, there were some remarks regarding the questions presented by the app. One of which was the question of whether or not the patient has multi-organ failure. It was noted that it was not always clear how to answer this question, as the threshold for what constitutes multi-organ failure was not clearly specified, and he vignettes occasionally lacked sufficient information to answer the question with certainty. It was also noted that while the app asks whether the patient has hypothermia, it does not ask whether the patient has a fever. Similarly, there were no questions regarding blood pressure or inflammatory parameters, which were deemed relevant questions that should be included.

A number of participants stated finding the experiment to have been a learning experience. For instance, the intended explanation for one of the vignettes was the use of isoprenaline. Only one of the participants correctly identified this as the explanation whilst performing the task unassisted, the rest being unaware that isoprenaline could cause an elevation in lactate. As such, the participants were pleasantly surprised when the app indicated isoprenaline as an explanation. Similarly, in another vignette, the intended explanation was a carbon-monoxide intoxication. The vignette stated a high level of carbon monoxide in the blood (HbCO) in the lab values, but this was either overlooked by many of the participants, or they were uncertain about the normal levels for HbCO. The app explicitly asked for carbon-monoxide intoxication, and the HbCO level if the question was no answered with "yes". The participants found that this way of asking questions helped alleviate tunnel vision by requiring them to carefully look at the vignette and pointing out information that may have been overlooked.

On the topic of using the system in the clinic, the responses were mildly positive. The participants found the system to be very useful for training purposes. For use in the clinic however, as stated earlier in the results of the questionnaire, most participants found the app mainly useful for complex cases, or to assist in cases of doubt. In simple cases, the app was not considered to provide much of an addition, as the explanation of the lactate level could be found on their own. While sensible, this logic has one important caveat. If a clinician unknowingly makes an incorrect diagnosis, they would not opt to use the system to verify their assumptions and subsequent diagnosis.

Finally, on the question of how the participants preferred to use the app (e.g. PC or smartphone), the responses were mixed, with no clear preference either way. And on the question of other areas which could be interesting to have an app for, hyponatremia and hypokalemia were frequently mentioned. The interpretation of these was said to be much more straight-forward than lactate, currently using a flowchart that could be easily turned into computerized format. Though whether this should be done is debatable, as pointed out in the background chapter regarding branching logic methods.

The full processed interviews can be found in appendix I on page 119.

Chapter 6

Discussion

The goal of this study was to develop a decision support system that assists clinicians in the interpretation of lactate levels. This system was subject to various design constraints. For it to be useful, it had to improve the clinician' accuracy of lactate level interpretations, and if possible, increase the speed of interpretation. Additionally, the system should be easy to use by the clinicians. The result was a rule-based expert system, using MYCIN-inspired certainty factors to deal with uncertainty and a web-based front-end as user interface.

The results of the evaluation study have shown that use of the decision support system significantly increased the accuracy of lactate level interpretations, with an average score 61% without and 90% with support of the app. While this shows a significant increase in accuracy, it is important not to misinterpret the 90% figure as an indication of the validity or real-world accuracy of the system or model. As mentioned in the methods chapter, the evaluation study used a simplified model of lactate, and therefore the vignettes that were presented to the participants were adapted to work with that model. The primary reason for choosing not to create a complete model was the required time investment. Creating such a model and validating it would require a lot of time, data and testing. For this project it was more important to determine whether a decision support system for lactate would be beneficial and of interest to clinicians before investing the time to create a (more) complete model of lactate in addition to developing the application itself.

The speed of interpretation showed a significant decrease, with an average duration of 106 seconds per vignette when unassisted, and 156 seconds per vignette when using the app. This increase in duration can be attributed to the nature of elicitation used by the system. Some questions asked by the system required the participants to go over the vignette again in order to properly answer them. While the difference is statistically significant, the actual difference is not very large. Even when using the app, the average duration is under 3 minutes, which still makes the use of this type of system very feasible. Moreover, there are ways the elicitation process could be improved to reduce the time consumption, which will be covered in the next section.

In the questionnaire and interview, the participants indicated finding the application easy and enjoyable to use. They have also considered it to be useful in complex cases, where an explanation for the lactate level is not obvious, and for training purposes. There was some remarks about the lactate model, such as it being too simplistic or "jumping to conclusions", but this was to be expected considering it actually was a simplified model.

6.1 Future work

The system in its current state is still a proof-of-concept, and there are a variety of ways it could be improved upon. This section will cover a variety of ways in which this can be done.

6.1.1 Lactate Model

A crucial point of improvement is the lactate model used by the system. As previously stated, the lactate model build and used in this study is a simplistic one, and not necessarily fit for general use. Some participants have also made remarks on the model, noting it had the tendency to jump to conclusions; lacking nuance. Since the model has not been validated, a good starting point would be to test this system against real-world cases to see how well it performs. This should expose the shortcomings of the model, and provide some guidance as to which changes need to be made. Ultimately, the model should be extended to the point where it can reliably handle real-world cases.

It is also a good idea to consider feedback from end users. The participants of the evaluation study have pointed out inconsistencies they had noticed, such the presence of a hypothermia question, but no fever question, or the lack of questions regarding blood pressure and inflammatory parameters. This feedback can lead to improvements in the clarity and usability of the model, as it was notable enough for users to point it out in the first place, and is also very useful to maintain the educational aspect of the model.

6.1.2 Enhancements

Looking back at the findings of Kawamoto et al., regarding the factors important for the success of clinical decision support systems, the current system still has some shortcomings. First of all, it is currently a standalone system, instead of being integrated in the clinician's workflow.

Another problem is that the system currently provides assessments, in the form of the possible explanations for lactate and the certainty thereof. However, tis could be improved by providing recommendations instead of, or in addition to these assessments. This could be done fairly easily by adding a translation layer on top of the output of the system. For example, when the system finds a probably explanation of the lactate level, instead of just stating this explanation (and the certainty thereof), the system could provide a recommended course of treatment for said explanation. This translation layer would then contain the recommended courses of treatment for all possible explanations the system can provide.

Finally, Kawamoto et al. recommend that clinical decision support systems should request feedback from the user, as to whether they agree with the provided assessment/recommendation or not, and if not, why. Doing this provides insight in how often the advice of the system is followed, and if there are certain outputs that are frequently disagreed with, and why this is the case. This information can then be used to improve the model, and reduce the frequency of disagreement. In addition, this also requires clinicians who disagree to formulate a reason as to why, requiring them to think critically about their reasoning. Currently, the system does not have this capability, but it would be fairly trivial to implement. The only complication would be where to store the results, which would ideally be a central database.

There are a few technical points of concern regarding the app's web server. While the server was designed to use a event-driven asynchronous architecture, specifically to enable it handle many connections simultaneously, it has not been subjected to a thorough stress-test. It has also only been used in an experimental setting, where one or two users used it simultaneously. It has not been used in a production environment, and it is current unknown how well this application

will scale and how well it will integrate into existing systems. There are also some security concerns. The JSON API only requires a valid session cookie to authenticate requests, the makes it susceptible to cross site request forgery (CSRF) attacks. Even though the attack in question could not cause much harm, the worst case being requiring the user to start a new session, it is an important consideration, and should be addressed, for example by using alternative authorization systems such as OAuth, before using this server and a production environment. As the app was made as a proof-of-concept, and was only meant for experimental purposes, these issues were left as-is.

The loss of speed when using the app was is a disappointing result of the evaluation study. This could be mitigated by integrating the system with an underlying patient database. Doing so could automatically answer questions for which the database has answers, requiring the user to only provide information that is not available. The current design of the system already makes automated entry of answers possible by using the API. In addition, this could be used to supply the system with real-time data, such as lactate measurements. This can be done once at the start of elicitation, or continuously. While the system already supports updating answers to questions, the user interface was not designed with continuous data entry in mind. The user interface, or the interaction design itself, may therefore require alterations to properly handle continuous updates in an intuitive way.

Lastly, the were a number of suggestions for improvement provided by the participants of the evaluation study. These mainly concerned the user interface, particularly during elicitation. A commonly suggested improvement is to leave numeric fields empty, instead of giving them a default value of *0*. This removes the need to first clear out the field before entering an answer. Other possible improvements include the addition of additional types of questions, such as likert-scale – a scale between "strongly agree" and "strongly disagree" – questions, displaying multiple-choice questions with a few number of options (e.g. the patient's sex) as a series of buttons instead of a dropdown list and making the "I don't know" button more clear, as the label "N/A" caused some confusion.

6.1.3 Alternative designs

An alternate direction to investigate is the use of different reasoning systems. The current system, being rule-based, and using certainty factors to handle uncertainty, has its drawbacks.

To make rule-based systems work, expert knowledge needs to be translated into a series of (interconnected) rules. The benefit of this method is that and it can provide its user with information about its reasoning and that the reasoning process is fully dictated by expert knowledge. These benefits were important for this project, as they facilitated the training/teaching aspect of the app. Because the rules were human-made and annotated with additional information, the app can provide useful feedback about its reasoning process, including explanations for its rules, instead of just providing an assessment or recommendation. As mentioned in section 2.3.3 (page 10) however, this design can also result in what is known as the knowledge acquisition problem: the difficulty of translating complex knowledge from experts into a good set of production rules. This could prove to be a complicating factor when creating more complete models of lactate. In addition, knowledge bases with a large number of rules can become complicated to work with, as altering rules - particularly those affecting other rules - could have unforeseen side-effects.

Certainty factor provide an interesting way to model uncertainty, as it makes an explicit distinction between belief, disbelief and uncertainty. This as opposed to probabilistic systems that only provide a degree of belief. The parallel and serial combination functions (see section 3.7.2, page 16) make these certainty factors very suitable for adding fuzzy-logic capabilities to rule-based systems. However, the use of certainty factors has received criticism for being ad-hoc, not

very well founded in probabilistic theory and a suboptimal way of modeling certainty (O'Leary, 1996). Bayesian networks are proposed as an alternative to certainty factors (Heckerman and Shortliffe, 1992).

The use of Bayesian networks would fundamentally change the reasoning system from a rule-based system to a probabilistic (Bayesian) system. Given the availability of a good dataset regarding lactate, this transition could prove be useful when building more complex model, as it would not suffer from the scaling problems inherent in (manually made) rule-based systems. This approach, as well as using machine learning methods, could prove very useful in creating a model of lactate usable in real-world situations. However, this comes at the cost of the training and teaching aspect of the app; use of probabilistic or machine learning methods diminishes the capability to provide the end user with useful reasoning information.

Between these two options, there is no 'best' option. Each has its own set of pros and cons. The question of whether to use a rule-based, a probabilistic (Bayesian), or yet another approach will depend on how important particular features (e.g. accuracy, ease of maintenance, suitability for teaching, etc..) are considered to be.

6.2 Conclusion

The results of this study have shown that use of decision support tools to interpret lactate levels has potential. It has shown to significantly improve accuracy of lactate level interpretations and was well received by members of the target audience. With further development and refinement, of underlying models in particular, these tools can prove to be valuable for providing decision support when interpreting lactate levels and for educational purposes.

Bibliography

Adams, I., Chan, M., Clifford, P., Cooke, W., Dallos, V., De Dombal, F., Edwards, M., Hancock, D., Hewett, D., and McIntyre, N. (1986). Computer aided diagnosis of acute abdominal pain: a multicentre study. *Bmj*, 293(6550):800–804.

Bakker, J., Nijsten, M., and Jansen, T. C. (2013). Clinical use of lactate monitoring in critically ill patients. *Ann Intensive Care*, 3(1):12.

Berkman, M., Ufberg, J., Nathanson, L. A., and Shapiro, N. I. (2009). Anion gap as a screening tool for elevated lactate in patients with an increased risk of developing sepsis in the emergency department. *The Journal of emergency medicine*, 36(4):391–394.

Bernal, W., Donaldson, N., Wyncoll, D., and Wendon, J. (2002). Blood lactate as an early predictor of outcome in paracetamol-induced acute liver failure: a cohort study. *The Lancet*, 359(9306):558–563.

Bleich, H. L. (1972). Computer-based consultation: electrolyte and acid-base disorders. *The American journal of medicine*, 53(3):285–291.

Bright, T. J., Wong, A., Dhurjati, R., Bristow, E., Bastian, L., Coeytaux, R. R., Samsa, G., Hasselblad, V., Williams, J. W., Musty, M. D., et al. (2012). Effect of clinical decision-support systems: a systematic review. *Annals of internal medicine*, 157(1):29–43.

Brooke, J. (1996). Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7.

Brooks, G. (1985). Lactate: glycolytic end product and oxidative substrate during sustained exercise in mammalsthe lactate shuttle. In *Circulation, Respiration, and Metabolism*, pages 208–218. Springer.

Brooks, G. A. (2009). Cell–cell and intracellular lactate shuttles. *The Journal of physiology*, 587(23):5591–5600.

De Dombal, F., Leaper, D., Staniland, J. R., McCann, A., and Horrocks, J. C. (1972). Computer-aided diagnosis of acute abdominal pain. *BMJ*, 2(5804):9–13.

Gladden, L. B. (2004). Lactate metabolism: a new paradigm for the third millennium. *The Journal of physiology*, 558(1):5–30.

Goldstein, M., Hoffman, B., Coleman, R., Musen, M. A., Tu, S., Advani, A., Shankar, R., and O'connor, M. (2000). Implementing clinical practice guidelines while taking account of changing evidence: Athena dss, an easily modifiable decision-support system for managing hypertension in primary care. In *Proceedings of the AMIA Symposium*, page 300. American Medical Informatics Association.

Heckerman, D. (2013). Probabilistic interpretations for mycin's certainty factors. *arXiv preprint arXiv:1304.3419*.

Heckerman, D. E. and Shortliffe, E. H. (1992). From certainty factors to belief networks. *Artificial Intelligence in Medicine*, 4(1):35–52.

Hripcsak, G., Ludemann, P., Pryor, T. A., Wigertz, O. B., and Clayton, P. D. (1994). Rationale for the arden syntax. *Computers and Biomedical Research*, 27(4):291–324.

Jaspers, M. W., Smeulers, M., Vermeulen, H., and Peute, L. W. (2011). Effects of clinical decision-support systems on practitioner performance and patient outcomes: a synthesis of high-quality systematic review findings. *Journal of the American Medical Informatics Association*, 18(3):327–334.

Kawamoto, K., Houlihan, C. A., Balas, E. A., and Lobach, D. F. (2005). Improving clinical practice using clinical decision support systems: a systematic review of trials to identify features critical to success. *Bmj*, 330(7494):765.

Kuperman, G., Maack, B., Bauer, K., and Gardner, R. (1991). Innovations and research review: the impact of the help computer system on the lds hospital paper medical record. *Topics in health record management*, 12(2):76–85.

Lee, J. D., Kirlik, A., and Dainoff, M. J. (2013). *The Oxford handbook of cognitive engineering*. Oxford University Press.

Levy, B. (2006). Lactate and shock state: the metabolic view. *Current opinion in critical care*, 12(4):315–321.

Manini, A. F., Kumar, A., Olsen, D., Vlahov, D., and Hoffman, R. S. (2010). Utility of serum lactate to predict drug-overdose fatality. *Clinical Toxicology*, 48(7):730–736.

Muir, B. M. (1994). Trust in automation: Part i. theoretical issues in the study of trust and human intervention in automated systems. *Ergonomics*, 37(11):1905–1922.

Musen, M. A., Middleton, B., and Greenes, R. A. (2014). Clinical decision-support systems. In *Biomedical informatics*, pages 643–674. Springer.

O'Leary, D. E. (1996). Verification of uncertain knowledge-based systems: an empirical verification approach. *Management Science*, 42(12):1663–1675.

Shortliffe, E. H. (1976). Mycin: Computer-based medical consultations.

Van Melle, W. (1978). Mycin: a knowledge-based consultation program for infectious disease diagnosis. *International journal of man-machine studies*, 10(3):313–322.

Vermeulen, R. P., Hoekstra, M., Nijsten, M. W., van der Horst, I. C., van Pelt, L. J., Jessurun, G. A., Jaarsma, T., Zijlstra, F., and van den Heuvel, A. F. (2010). Clinical correlates of arterial lactate levels in patients with st-segment elevation myocardial infarction at admission: a descriptive study. *Crit Care*, 14(5):R164.

Vogelzang, M. (2008). Computer assisted decision support in acutely ill patients. application in glucose management and quantification of myocardial reperfusion.

Warner, H. R. (1979). *Computer–Assisted Medical Decision-Making*. Academic Press, Inc., Orlando, FL, USA.

Weil, M. H. and Tang, W. (2009). Forty-five-year evolution of stat blood and plasma lactate measurement to guide critical care. *Clinical chemistry*, 55(11):2053–2054.

Appendix A

Vignettes

This appendix contains the ten vignettes that were used in the experiment. Each describes a fictional patient's medical information in a way similar to what the participants would have in a real situation. In addition to the patient description, the vignettes shown here include the explanation that the participants should have responded with, and matches the response(s) given by the app.

Vignette 1 (Patient 178)

Male, 65 y.o.
Lactate 5.1

History:

Recent CABG.

Presenting complaints:

Dyspnea.
Strange feeling within chest.

Physical examination:

RR 150/80 p 105 SR.
High respiratory frequency.
O2-sat 98%.

Laboratory examination:

Hb 7.2; Platelets 300; Leukocytes 12.1; CRP 34
BGA: pCO2 2.5 pO2 14.1 sat 99% pH 7.61 (oxygen at 1L/min)

Chest X-ray:

Possible infiltrate upper right lobe.

Explanation:

Hyperventilation.

Vignette 2 (Patient 216)

Female, 37 y.o.
Lactate: 5.9

History:

Heavy smoker, COPD, recurrent pneumonia since childhood.

Emergency admission:

Severe dyspnea.

Chest X-ray:

Bilateral infitrates, most pronounced in the right middle lobe.

Physical examination:

Cachectic, shivering and in severe distress, can hardly talk with non-rebreathing mask at 100%.
T 38.6
HB 110/45; pulse 132/min.
Enhanced breathing sounds and rhonchi over all lung fields.
Extremities: Cold, mottling.
Diuresis: minimal.

Laboratory measurements:

pH 7.21 pCO2 9.0 pO2 5.9
Gluc 13.4
Hb 9.4 Platelets 234 Leukocytes 16.3
LDH 400 ALAT 132 ASAT 101
CRP 230
Urine: Ketone positive.

Explanation:

Sepsis (+ Pneumonia).

Vignette 3 (Patient 287)

Male, 39 y.o.
Lactate: 9.5

History:

2007 Hypertension

2009 Type II Diabetes

Medication at home:

Hydrochlorothiazide, metformin, metoprolol.

Emergency admission:

Hyperventilation, abdominal pain, hypotension.

Acute History:

Diarrea and vomiting after barbecue party 4 days ago. Rest of family has recovered.

Physical exam:

Heavy breathing, complains of diffuse abdominal pain.
HD: RR 95/45 p 105 SR.
No abnormal breathing sounds.
O2-saturation 94% with 4L of oxygen.
Diffuse tenderness abdomen.
No evident distension. No peristalsis.
Cold extremities.

Laboratory mreasurements:

Na 145, K 3.0 Gluc 12.3 Cl 100
BGA pH 6.
Urine: Ketones positive, glucose positive, Diabetic Keto-Acidosis, Metformin use.

Explanation:

Metformin intoxication. (MALA)

Vignette 4 (Patient 326)

Female, 47 y.o.
Lactate: 9.3

History:

2001 Severe obesity.
Hypertension.

2005 Diabetes Type-II, only with oral antidiabetics.

2013 Insulin added to oral antidiabetics.

Emergency admission:

Found unconscious by a neighbour on cold winter evening in boat where she lives. No apparent signs of suicide.

Laboratory measurements:

Glucose 15.1
CRP 3.0 Hb 8.0; Leukocytes 16.2; Platelets 204
BGA pH 7.20 pCO2 4.0 pO2 12 HbCO 0.31
GGT 105 ALAT 76 ASAT 54 Total bilirubin 27

Explanation:

CO intoxication (+ possibly augmented by metformin).

Vignette 5 (Patient 489)

Male, 50 y.o.
Lactate: 5.3

History:

2012 Heart Failure NYHA 3. Cause not determined. Cardiomyopathy.

Emergency visit because of:

Progressive dyspnea on exertion.

Medication at home:

Enalapril, Furosemide, Metoprolol.

Physical exam:

Awake, well-oriented, RR 130/80 p 95 SR, cold extremities.

Laboratory measurements:

pH 7.29 pCO2 4.0 pO2 10.0

Explanation:

Unknown.

Vignette 6 (Patient 504)

Female, 50 y.o.
Lactate 12.6

History:

2008 Diabetes mellitus type II, treated successfully with weight reduction.

2012 Left Mastectomy for breast carcinoma.

2014 Heart failure.

Acute reason for emergency admission:

De novo generalized seizures.

Clinical course:

Seizures successfully treated with repeated doses of clonazepam.
Patient now intubated.

Chest X-ray:

Density lower left lobe.
Potential pathological fractures of several ribs.

Laboratory measurements:

Hb 6.9 Platelets 210 Leukocytes 15; CRP 3; Albumin 30; Ca 2.51; P 1.0
LDH 380; AP 110; ALAT 70; ASAT 85; Total Bilirubin 40.
BGA: pH 7.28 pCO2 7.1 pO2 8.7 sat 92%.

Explanation:

Recent generalized seizures.

Vignette 7 (Patient 698)

Male, 80 y.o.
Lactate: 14.2

History:

COPD, Heart failure, Chronic renal failure.

Emergency admission:

Collapse.

Medication at home:

Acetylic acid, metoprolol, simvastatine, furosemide, perindopril.

Clinical course:

Collapsed in kitchen, low blood pressure.
Daughter called 112.

Physical examination:

RR 78/50 pulse 110 irregular small complexes.
O2-saturation: 93% (with Ventimask 60%).
Bilateral wheezing and crepitations.

Laboratory measurements:

Glucose 12.7
Hb 9.0 Na 134 K 4.5 Creatinine 189, Urea 14
BGA pH 7.15 pCO2 3.3 pO2 9.9 sat 92% BE -9.2
LDH 287 ASAT 80 ALAT 65
PT 16 Fibrinogen 2.7 INR 1.5
Urine: Not available.

Chest X-ray:

Enlarged contour of heart, diffuse densities over all lung fields compatible with pulmonary edema.

ECG:

ST-elevation in inferior leads.

Explanation:

Cardiogenic shock (secondary to myocardial infarction)(+ renal failure).

Vignette 8 (Patient 745)

Female, 67 y.o.
Lactate: 3.9

History:

2005 Hypercholesterolemia.

2008 8y Sick sinus syndrome → Pacemaker.

2009 COPD.

2010 PCI LAD.

2011 Atrial fibrillation.

Emergency admission:

Bradycardia with collapse.
Received atropine in ambulance.
Now stable for 2 hours with isoprenaline infusion HR 68 RR 125/60 and Ventimask 40% sat 96%.

Medication at home:

Atorvastatine, verapamil, metoprolol, enalapril, steroid puffs, prednisolon.

Physical examination:

Responds adequately to questions.
HD: RR 110/60 80/min (supported by isoprenaline 3/50 at 6 ml/h)
Lungs: No abnormal sounds.
Abdomen: Some peristalsis.
Extremities: Cool to cold.

Laboratory measurements:

Hb 7.9 Platelets 190 Leukocytes 4.5 CRP 45
LDH 320 ALAT 40 ASAT 67 GGT 47
BGA pH 7.31 pCO2 4.7 pO2 10.3 sat 96%.
Urine: Dipstick normal.

Explanation:

Inotropes, more specifically direct 2-mimetic effect of isoprenalin.

Vignette 9 (Patient 847)

Male, 62 y.o.
Lactate: 3.2

History:

2006 Diabetes mellitus Type 1

2007 Hypertension

2008 Atrial fibrillation

Recent history obtained from family:

Progressive cough and fever since three days.

Acute reason for admission:

Obtunded, hypotensive, dehydrated, temperature 39.2°C

Physical examination:

EMV 3-5-4 RR 105/50 p 105 irregular, AF.
Pronounced breathing sounds over right dorsal chest.
Slightly distended abdomen, some peristalsis present.

Laboratory measurements:

CRP 170; Leukocytes 8.0; Platelets 130;
Glucose 34 , BGA: pH 6.95 pCO2 2.9 pO2 6.9
Urine: Ketones positive, glucose positive.

Chest X-ray:

Right-sided infiltrate upper lobe.

Explanation:

Respiratory Insufficiency n.o.s + Hypotension (+ Diabetic Keto-acidosis).

Vignette 10 (Patient 925)

Female, 42 y.o.
Lactate 7.0

History:

2011 Alcohol induced liver cirrosis.
Creatinine clearance 55 ml/min.

2013 Hematemesis for esophageal varix bleeding.
Creatinine clearance 30 ml/min.

Acute reason for admission:

Progressive malaise and collapse.

Physical examination:

EMV 3-6-5.
Icteric, obtunded, very slow reacting.
RR 95/60 HR 120 Sat 91%.

Laboratory measurements:

Hb 7.0; Platelet count 90; ASAT 250; ALAT 300
Creatinine 430; PT 21 ; Glucose 2.9.

Explanation:

Acute-on-Chronic liver failure.

Appendix B

Lactate Model

B.1 Questions

Group - Patient information

"*Age*":

Type	Integer
Tooltip	The patient's age.
Min / Max	0 / 150
Units	years
Attributes	Forced presentation, skippable
Statements	*None*

"*Lactate Level*":

Type	Numeric
Min / Max	0.0 / 50.0
Units	mmol/L
Statements:	

Object	Property	Range	CF_{in}	CF_{out}
lactate	*upToSmallMinor*	$0.00 \pm 0.75 - 4.75 \pm 1.00$	1.0	-1.0
lactate	*upToMinor*	$0.00 \pm 0.75 - 9.00 \pm 1.00$	1.0	-1.0
lactate	*upToBigMinor*	$0.00 \pm 0.75 - 11.00 \pm 1.00$	1.0	-1.0
lactate	*upToMajor*	$0.00 \pm 0.75 - 30.00 \pm 20.00$	1.0	-1.0

"*Patient ID*":

Type	Integer
Attributes	Forced presentation
Statements	*None*

"*Sex*":

Type	Multiple Choice
Description	Females have slightly higher lactate levels.
Options	Male
	Female
Attributes	Forced presentation, skippable
Statements	*None*

Group - Relevant pre-existing morbidity

"*COPD*":

Type	Yes/no
Tooltip	Chronic obstructive pulmonary disease.
Description	The stress associated with an exacerbation of COPD or pneumonia complicating it can cause elevated lactate levels.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*COPD*	1.0	-1.0

"*Chronic liver failure*":

Type	Yes/no
Description	Liver failure can lead to very high lactate levels through failure of gluconeogenesis and thus the Cori cycle.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*chronicLiverFailure*	1.0	-1.0

"*Chronic renal failure*":

Type	Yes/no
Description	Severe renal failure can impair lactate clearance and the kidney contribute to 10% to 40% of the Cori cycle depending on circumstances.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*chronicRenalFailure*	1.0	-1.0

"*Diabetes*":

Type	Yes/no
Description	Diabetes is associated with slightly higher lactate levels and the antidiabetic drug metformin can induce severe lactic acidosis.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*diabetes*	1.0	−1.0

"*Heart failure*":

Type	Yes/no
Description	A low cardiac output can cause high lactate levels.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*heartFailure*	1.0	−1.0

Group - Relevant pre-existing morbidity (II)

"*Deranged DM*":

Type	Yes/no
Description	Higher glucoses levels can subsequently induce slightly higher lactate levels.
Attributes	Skippable, substitutable
Prerequisites	Diabetes
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*derangedDM*	1.0	−1.0

"*Diabetic Keto-acidocis*":

Type	Yes/no
Description	Diabetic keto-acidosis is normally not associated with elevated lactate levels.
Attributes	Skippable, substitutable
Prerequisites	Diabetes
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*diabeticKetoAcidosis*	1.0	−1.0

"*Hyper-osmolar hyperglycemia*":

Type	Yes/no
Attributes	Skippable, substitutable
Prerequisites	Diabetes
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*hyperOsmolarHyperglycemia*	1.0	-1.0

Group - Recent Events

"*Acute (or chronic) hepatic failure*":

Type	Yes/no
Description	Liver failure can rapidly lead to very high lactate levels through failure of gluconeogenesis and thus the Cori cycle.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*hepaticFailure*	1.0	-1.0

"*Recent cardiac arrest and CPR*":

Type	Yes/no
Description	OHCA or IHCA leads to highest production and acccumulation of lactate and thus very high levels after CPR.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*cardiacArrest*	1.0	-1.0

"*Recent generalized seizures*":

Type	Yes/no
Description	Generalized seizures consume massive amounts of glucose and this is converted to lactate under relatively hypoxic conditions.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*generalizedSeizures*	1.0	-1.0

Group - Acute signs

"*Hypothermia*":

Type	Yes/no
Tooltip	$T < 35.5$°C
Description	Hypothermia may lead to stress and strongly elevated meabolic demands resulting in mild hyperlactemia.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*hypothermia*	1.0	-1.0

"*Respiratory Insufficiency n.o.s*":

Type	Yes/no
Description	The stress associated with respiratory insufficiency can cause elevated lactate levels.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*respiratoryInsufficiency*	1.0	-1.0

"*Shock*":

Type	Yes/no
Tooltip	i.e. MAP <60 or MAP >60 with catecholamine support.
Description	A low cardiac output can cause high lactate levels.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*shock*	1.0	-1.0

Group - Acute diagnoses

"*Cerebral vascular accident*":

Type	Yes/no
Description	The stress associated with CVA, and in particular subarachnoidal bleeding can cause elevated lactate and glucose levels.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*CVA*	1.0	−1.0

"*Hyperventilation*":

Type	Yes/no
Tooltip	A pCO2 below normal (<4 kPa) caused by inappropriate stimulation of respiration or a mechanical ventilator.
Description	A pCO2 below normal (<4 kPa) can lead to elevated lactate levels.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*hyperventilation*	1.0	−1.0

"*Lymphoma*":

Type	Yes/no
Description	Under massive disseminated tumor loads, the Warburg effect can cause extensive lactate production with usually also a tendency to hypoglycemia and liver impairment.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*lymphoma*	1.0	−1.0

"*Multiple organ failure*":

Type	Yes/no
Description	The various organs that have failed (in particular liver and kidneys) and the associated stress can lead to hyperlactatemia.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*multipleOrganFailure*	1.0	−1.0

"*Myocardial infarction*":

Type	Yes/no
Description	The stress of myocardial infarction and a low cardiac output can cause high lactate levels.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*myocardialInfarction*	1.0	−1.0

"*Pneumonia / Respiratory insufficiency*":

Type	Yes/no
Description	The stress associated with an exacerbation of COPD or pneumonia complicating it can cause elevated lactate levels.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*pneumonia*	1.0	−1.0

"*Renal failure*":

Type	Yes/no
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*renalFailure*	1.0	−1.0

"*Sepsis*":

Type	Yes/no
Description	The stress associated with sepsis can cause elevated lactate levels.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*sepsis*	1.0	−1.0

Group - Acute diagnoses (II)

"*Cardiogenic shock*":

Type	Yes/no
Description	The stress of a low cardiac output can cause high lactate levels.
Attributes	Skippable, substitutable
Prerequisites	Myocardial infarction
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*cardiogenicShock*	1.0	−1.0

"*Cerebral infarction*":

Type	Yes/no
Attributes	Skippable, substitutable
Prerequisites	Cerebral vascular accident
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*cerebralInfarction*	1.0	−1.0

"*Pneumosepsis*":

Type	Yes/no
Description	The stress associated with sepsis and pneumonia and respiratory insufficiency can cause elevated lactate levels.
Attributes	Skippable, substitutable
Prerequisites	Sepsis
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*pneumosepsis*	1.0	−1.0

"*Subarachnoidal hemorrhage*":

Type	Yes/no
Description	Subarachnoidal bleeding can cause elevated lactate and glucose levels.
Attributes	Skippable, substitutable
Prerequisites	Cerebral vascular accident
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*subarachnoidalHemorrhage*	1.0	−1.0

Group - Acute intoxication

"*Alcohol intoxication*":

Type	Yes/no
Description	Ethanol can directly interfere with gluconeogenesis and thus the Cori cycle and thus induce mild hyperlactatemia.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*alcoholIntoxication*	1.0	−1.0

"*Metformin use*":

Type	Yes/no
Description	Metformin directly inhibits mitochondria and therapeutic levels can lead to elevated lactate levels in disease.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
medication	*metformin*	1.0	−1.0

"*Major carbon monoxide intoxication*":

Type	Yes/no
Tooltip	HbCO >0.08
Description	A mild elevation of HbCO (i.e. <10%) is not associated with hyperlactatemia; severe HbCO intoxications can cause severe lactic acidosis (i.e. classical type B lactic acidosis resulting from mitochondrial inhibition.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*monoxideIntoxication*	1.0	−1.0

Group - Acute intoxication (II)

"*Major carbon monoxide intoxication*":

Type	Yes/no
Description	Metformin directly inhibits mitochondria and (in particular intoxications, i.e. levels >5 mg/L) can lead to (very) high lactate levels.
Attributes	Skippable, substitutable
Prerequisites	Metformin use
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*metforminIntoxication*	1.0	−1.0

Group - Treatments that may increase lactate

"*Epinephrine or isoprenalin*":

Type	Yes/no
Tooltip	The beta-2 adrenergic agonists adrenalin (=epinephrin) and isoprenalin.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
medication	*epinephrine*	1.0	−1.0

"*Inotropes or vasoconstrictors*":

Type	Yes/no
Tooltip	e.g Noradrenaline, dobutamin, dopamin, milrinone, vasopressin, etc
Description	In particular the use of vasopressin or very high doses of noradrenalin (i.e. >10 ml/hr 10mg/50ml) can lead to severe (hepatic) vasoconstriction and thus hyperlactatemia.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
medication	*inotropes*	1.0	−1.0

Group - Congenital/hereditary syndromes

"*MELAS*":

Type	Yes/no
Tooltip	Syndrome of Mitochrondrial encephalomyopathy, lactic acidosis and stroke-like episodes.
Description	This mitochondrial disease, as in principle holds for most mitochondrial diseases causes type B hyperlactatemia.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*MELAS*	1.0	−1.0

"*Von Gierke disease*":

Type	Yes/no
Tooltip	Glucose-6-phosphatase deficiency.
Description	Children with this syndrome tend to have hypoglycemia and hyperlactatemia.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*VonGierkeDisease*	1.0	−1.0

Group - Acute traumatic, surgical or vascular syndromes

"*(Ruptured) abdominal aortic aneurism*":

Type	Yes/no
Description	The hypotension and stress associated with RAAA and its emergency treatment can lead to elevated lactate levels.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*AAA*	1.0	−1.0

"*Lerriche syndrome*":

Type	Yes/no
Tooltip	Acute occlusion of the abdominal aorta.
Description	The stress and hypoperfusion or complete ischemia of the lower body can lead to elevated lactate levels.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*LerricheSyndrome*	1.0	−1.0

"*Acute pancreatitis*":

Type	Yes/no
Description	The very extensive retroperitoneal and abdominal inflammation and its associated hemodynamic consequences can lead to elevated lactate levels.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*acutePancreatitis*	1.0	−1.0

"*Major trauma*":

Type	Yes/no
Tooltip	Acute mechanical trauma warranting (post-surgical) admission to the ICU.
Description	The severe acute stress, acute systemic inflammation and blood loss and coagulopathy with its associated hemodynamic consequences can lead to elevated lactate levels.
Attributes	Skippable
Prerequisites	Lactate level
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*majorTrauma*	1.0	−1.0

Group - Major trauma

"*Blood loss (>0.5 L)*":

Type	Yes/no
Tooltip	Acute loss of >10% of the circulating volume.
Description	The acute stress and hemodynamic consequences of reduced circulating volume can lead to elevated lactate levels.
Attributes	Skippable, substitutable
Prerequisites	Major Trauma
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*majorBloodLoss*	1.0	−1.0

"*Cerebral trauma*":

Type	Yes/no
Tooltip	Brain injury warranting (post-surgical) admission to the ICU.
Description	Injury to the brain can lead to extra-ordinary adrenergic and glucocorticoid responses that can both lead to hyperglycemia and hyperlactatemia.
Attributes	Skippable, substitutable
Prerequisites	Major Trauma
Statements:	

Object	Property	CF_{yes}	CF_{no}
patient	*cerebralTrauma*	1.0	−1.0

Group - Relevant laboratory measures

"*ALAT*":

Type Numeric
Tooltip Alanine Transaminase
Attributes Skippable
Min / Max 0.0 / 1000.0
Statements:

Object	Property	Threshold	CF_{above}	CF_{below}
ALAT	*elevated*	215.00 ± 25.00	1.0	-1.0

"*ASAT*":

Type Numeric
Tooltip Aspartate Aminotransferase
Attributes Skippable
Min / Max 0.0 / 1000.0
Statements:

Object	Property	Threshold	CF_{above}	CF_{below}
ASAT	*elevated*	210.00 ± 25.00	1.0	-1.0

"*CRP*":

Type Numeric
Tooltip C-Reactive Protein
Attributes Skippable
Min / Max 0.0 / 1000.0
Statements:

Object	Property	Threshold	CF_{above}	CF_{below}
CRP	*elevated*	215.00 ± 25.00	1.0	-1.0

"*HbCO*":

Type Numeric
Attributes Skippable
Min / Max 0.00 / 20.00
Statements:

Object	Property	Threshold	CF_{above}	CF_{below}
HbCO	*elevated*	0.1 ± 0.05	1.0	-1.0

"*HbCO*":

Type	Numeric
Attributes	Skippable
Min / Max	0.00 / 20.00
Statements:	

Object	Property	Threshold	CF_{above}	CF_{below}
HbCO	*elevated*	0.1 ± 0.05	1.0	-1.0

"*pH*":

Type	Numeric
Attributes	Skippable
Min / Max	0.00 / 14.00
Statements:	

Object	Property	Threshold	CF_{above}	CF_{below}
pH	*alkalosis*	7.45 ± 0.5	1.0	0.0
pH	*acidosis*	7.35 ± 0.5	0.0	1.0
pH	*severeAcidosis*	6.8 ± 0.2	0.0	1.0

B.2 Rules

Rule 1: "*AAA*":

IF	*patient*	*AAA*	**AND**
	lactate	*upToBigMinor*	
THEN			
	explanation	*AAA*	0.9

Description: The hypotension and stress associated with RAAA and its emergency treatment can lead to elevated lactate levels.

Rule 2: "*COPD*":

IF	*patient*	*COPD*	**AND**
	lactate	*upToSmallMinor*	
THEN			
	explanation	*COPD*	0.7

Description: The stress associated with an exacerbation of COPD or pneumonia complicating it can cause elevated lactate levels.

Rule 3: "*HbCOLab*":

IF	*HbCO*	*elevated*	
THEN			
	lab	*monoxideIntoxication*	0.9

Description: An elevated HbCO level is indicative of carbon monoxide intoxication.

Rule 4: "*Lerriche*":

IF	*patient*	*LerricheSyndrome*	**AND**
	lactate	*upToBigMinor*	
THEN			
	explanation	*LerricheSyndrome*	0.9

Description: The stress and hypoperfusion or complete ischemia of the lower body can lead to elevated lactate levels.

Rule 5: "*MELAS*":

IF	*patient*	*MELAS*	**AND**
	lactate	*upToBigMinor*	
THEN			
	explanation	*MELAS*	0.9

Description: This mitochondrial disease, as in principle holds for most mitochondrial diseases causes type B hyperlactatemia.

Rule 6: "*VonGierke*":

IF	*patient*	*VonGierkeDisease*	**AND**
	lactate	*upToBigMinor*	
THEN			
	explanation	*VonGierkeDisease*	0.9

Description: Children with this syndrome tend to have hypoglycemia and hyperlactatemia.

Rule 7: "*acuteHepaticFailure*":

IF	*lactate*	*upToBigMajor*	**AND**
	patient	*hepaticFailure*	**OR**
	lab	*liverFailure*	
THEN			
	explanation	*acuteHepaticFailure*	0.9

Description: Liver failure can rapidly lead to very high lactate levels through failure of gluconeogenesis and thus the Cori cycle.

Rule 8: "*alcohol*":

IF	*patient*	*alcoholIntoxication*	**AND**
	lactate	*upToSmallMinor*	
THEN			
	explanation	*alcoholIntoxication*	0.9

Description: Ethanol can directly interfere with gluconeogenesis and thus the Cori cycle and thus induce mild hyperlactatemia.

Rule 9: "*bloodLoss*":

IF	*patient*	*majorBloodLoss*	**AND**
	lactate	*upToBigMinor*	
THEN			
	explanation	*majorBloodLoss*	0.9

Description: The acute stress and hemodynamic consequences of reduced circulating volume can lead to elevated lactate levels.

Rule 10: "*carbonMonoxideIntoxication*":

IF	*lactate*	*upToMajor*	**AND**
	patient	*monoxideIntoxication*	**OR**
	lab	*monoxideIntoxication*	
THEN			
	explanation	*carbonMonoxideIntoxication*	0.9

Description: A mild elevation of HbCO (i.e. ¡10%) is not associated with hyperlactatemia; severe HbCO intoxications can cause severe lactic acidosis (i.e. classical type B lactic acidosis resulting from mitochondrial inhibition.

Rule 11: "*cardiogenicShock*":

IF	*patient*	*cardiogenicShock*	**AND**
	lactate	*upToMajor*	
THEN			
	explanation	*cardiogenicShock*	0.9

Description: The stress of a low cardiac output can cause high lactate levels.

Rule 12: "*cerebralInfarction*":

IF	*patient*	*cerebralInfarction*	**AND**
	lactate	*upToSmallMinor*	
THEN			
	explanation	*cerebralInfarction*	0.9

Description: Injury to the brain can lead to extra-ordinary adrenergic and glucocorticoid responses that can both lead to hyperglycemia and hyperlactatemia.

Rule 13: "*cerebralTrauma*":

IF	*patient*	*cerebralTrauma*	**AND**
	lactate	*upToBigMinor*	
THEN			
	explanation	*cerebralTrauma*	0.9

Description: Injury to the brain can lead to extra-ordinary adrenergic and glucocorticoid responses that can both lead to hyperglycemia and hyperlactatemia.

Rule 14: "*chronicLiverFailure*":

IF	*patient*	*chronicLiverFailure*	**AND**
	lactate	*upToBigMinor*	
THEN			
	explanation	*chronicLiverFailure*	0.9

Description: Chronic liver failure explains a moderate increase in lactate.

Rule 15: "*chronicRenalFailure*":

IF	*patient*	*chronicRenalFailure*	**AND**
	lactate	*upToSmallMinor*	
THEN			
	explanation	*chronicRenalFailure*	0.9

Description: Chronic renal failure explains a small increase in lactate.

Rule 16: "*derangedDM*":

IF	*patient*	*derangedDM*	**AND**
	lactate	*upToMinor*	
THEN			
	explanation	*derangedDM*	0.9

Description: Higher glucoses levels can subsequently induce slightly higher lactate levels.

Rule 17: "*diabeticKetoAcidosis*":

IF	*patient*	*diabeticKetoAcidosis*	**AND**
	lactate	*upToMinor*	
THEN			
	explanation	*diabeticKeto-acidosis*	0.5

Description: Diabetic keto-acidosis is normally not associated with elevated lactate levels.

Rule 18: "*generalizedSeizures*":

IF	*patient*	*generalizedSeizures*	**AND**
	lactate	*upToMajor*	
THEN			
	explanation	*recentGeneralizedSeizures*	0.9

Description: Generalized seizures consume massive amounts of glucose and this is converted to lactate under relatively hypoxic conditions.

Rule 19: "*heartFailure*":

IF	*patient*	*heartFailure*	**AND**
	lactate	*upToSmallMinor*	
THEN			
	explanation	*heartFailure*	0.9

Description: A low cardiac output can cause high lactate levels.

Rule 20: "*hyperOsmolarHyperglycemia*":

IF	*patient*	*hyperOsmolarHyperglycemia*	**AND**
	lactate	*upToMinor*	
THEN			
	explanation	*hyper-osmolarHyperglycemia*	0.9

Rule 21: "*hyperventilation*":

IF	*patient*	*hyperventilation*	**AND**
	lactate	*upToMinor*	
THEN			
	explanation	*hyperventilation*	0.9

Description: A pCO2 below normal (¡ 4 kPa) can lead to elevated lactate levels.

Rule 22: "*hypothermia*":

IF	*patient*	*hypothermia*	**AND**
	lactate	*upToMinor*	
THEN			
	explanation	*hypothermia*	0.9

Description: Hypothermia may lead to stress and strongly elevated meabolic demands resulting in mild hyperlactemia.

Rule 23: "*inotropes*":

IF	*lactate*	*upToMinor*	**AND**
	medication	*inotropes*	**OR**
	medication	*epinephrine*	
THEN			
	explanation	*inotropes*	0.9

Description: In particular the use of vasopressin or very high doses of noradrenalin (i.e. ¿10 ml/hr 10mg/50ml) can lead to severe (hepatic) vasoconstriction and thus hyperlactatemia.

Rule 24: "*liverFailureLab*":

IF	*ASAT*	*elevated*	**OR**
	ALAT	*elevated*	
THEN			
	lab	*liverFailure*	0.9

Description: A highly increase ASAT and ALAT is indicative of liver failure.

Rule 25: "*lymphoma*":

IF	*patient*	*lymphoma*	**AND**
	lactate	*upToBigMinor*	
THEN			
	explanation	*lymphoma*	0.9

Description: Under massive disseminated tumor loads, the Warburg effect can cause extensive lactate production with usually also a tendency to hypoglycemia and liver impairment.

Rule 26: "*majorTrauma*":

IF	*patient*	*majorTrauma*	**AND**
	lactate	*upToMinor*	
THEN			
	explanation	*majorTrauma*	0.9

Description: The severe acute stress, acute systemic inflammation and blood loss and coagulopathy with its associated hemodynamic consequences can lead to elevated lactate levels.

Rule 27: "*metforminIntoxication*":

IF	*lactate*	*upToMajor*	**AND**
	patient	*metforminIntoxication*	**OR**
	lab	*metforminIntoxication*	
THEN			
	explanation	*metforminIntoxication*	0.9

Description: Metformin directly inhibits mitochondria and (in particular intoxications, i.e. levels ¿5 mg/L) can lead to (very) high lactate levels.

Rule 28: "*metforminIntoxicationLab*":

IF	*medication*	*metformin*	**AND**
	pH	*severeAcidosis*	
THEN			
	lab	*metforminIntoxication*	0.9

Description: A low pH combined with metformin use is indicative of metformin intoxication.

Rule 29: "*metforminUse*":

IF	*lactate*	*upToBigMinor*	**AND**
	medication	*metformin*	**AND**
NOT	*patient*	*metforminIntoxication*	
THEN			
	explanation	*metforminUse*	0.6

Description: Metformin directly inhibits mitochondria and therapeutic levels can lead to elevated lactate levels in disease.

Rule 30: "*metforminUseAndRenalFailure*":

IF	*patient*	*renalFailure*	**AND**
	lactate	*upToMinor*	**AND**
	medication	*metformin*	
THEN			
	explanation	*metforminUseAndRenalFailure*	0.9

Rule 31: "*multipleOrganFailure*":

IF	*patient*	*multipleOrganFailure*	**AND**
	lactate	*upToMinor*	
THEN			
	explanation	*multipleOrganFailure*	0.9

Description: The various organs that have failed (in particular liver and kidneys) and the associated stress can lead to hyperlactatemia.

Rule 32: "*myocardialInfarction*":

IF	*patient*	*myocardialInfarction*	**AND**
	lactate	*upToSmallMinor*	
THEN			
	explanation	*myocardialInfarction*	0.9

Description: The stress of myocardial infarction and a low cardiac output can cause high lactate levels.

Rule 33: "*noCVA*":

IF NOT	*patient*	*CVA*	**OR**
	patient	*CVA*	**IS UNCERTAIN AND**
	patient	*CVA*	**IS KNOWN**
THEN			
	patient	*subarachnoidalHemorrhage*	-1.0
	patient	*cerebralInfarction*	-1.0

Description: Absense of CVA precludes subarachnoidal hemorrhage and cerebral infaction.

Rule 34: "*noDiabetes*":

IF NOT	*patient*	*diabetes*	**OR**
	patient	*diabetes*	**IS UNCERTAIN AND**
	patient	*diabetes*	**IS KNOWN**
THEN			
	patient	*derangedDM*	-1.0
	patient	*diabeticKetoAcidosis*	-1.0
	patient	*hyperOsmolarHyperglycemia*	-1.0

Rule 35: "*noMajorTrauma*":

IF NOT	*patient*	*majorTrauma*	**OR**
	patient	*majorTrauma*	**IS UNCERTAIN AND**
	patient	*majorTrauma*	**IS KNOWN**
THEN			
	patient	*cerebralTrauma*	-1.0
	patient	*majorBloodLoss*	-1.0

Rule 36: "*noMetformin*":

IF NOT	*medication*	*metformin*	**OR**
	medication	*metformin*	**IS UNCERTAIN AND**
	medication	*metformin*	**IS KNOWN**
THEN			
	patient	*metforminIntoxication*	-1.0

Description: Lack of metformin use precludes metformin intoxication.

Rule 37: "*noMyocardialInfarction*":

IF NOT	*patient*	*myocardialInfarction*	**OR**
	patient	*myocardialInfarction*	**IS UNCERTAIN AND**
	patient	*myocardialInfarction*	**IS KNOWN**
THEN			
	patient	*cardiogenicShock*	-1.0

Description: Absense of myocardial infaction precludes cardiogenic shock.

Rule 38: "*noSepsis*":

IF NOT	*patient*	*sepsis*	**OR**
	patient	*sepsis*	**IS UNCERTAIN**
AND NOT	*lab*	*sepsis*	**OR**
	lab	*sepsis*	**IS UNCERTAIN**
AND	*patient*	*sepsis*	**IS KNOWN OR**
	lab	*sepsis*	**IS KNOWN**
THEN			
	patient	*pneumoSepsis*	-1.0

Description: Absense of sepsis implies the absense of pneumosepsis.

Rule 39: "*pancreatitis*":

IF	*patient*	*acutePancreatitis*	**AND**
	lactate	*upToMinor*	
THEN			
	explanation	*acutePancreatitis*	0.9

Description: The very extensive retroperitoneal and abdominal inflammation and its associated hemodynamic consequences can lead to elevated lactate levels.

Rule 40: "*pneumonia*":

IF	*patient*	*pneumonia*	**AND**
	lactate	*upToMinor*	
THEN			
	explanation	*pneumonia*	0.9

Description: The stress associated with an exacerbation of COPD or pneumonia complicating it can cause elevated lactate levels.

Rule 41: "*pneumosepsis*":

IF	*patient*	*pneumosepsis*	**AND**
	lactate	*upToMinor*	
THEN			
	explanation	*pneumosepsis*	0.9

Description: The stress associated with sepsis and pneumonia and respiratory insufficiency can cause elevated lactate levels.

Rule 42: "*recentCardiacArrest*":

IF	*patient*	*cardiacArrest*	**AND**
	lactate	*upToMajor*	
THEN			
	explanation	*recentCardiacArrest*	0.9

Description: OHCA or IHCA leads to highest production and acccumulation of lactate and thus very high levels after CPR.

Rule 43: "*renalFailure*":

IF	*patient*	*renalFailure*	**AND**
	lactate	*upToSmalMinor*	
THEN			
	explanation	*renalFailure*	0.9

Rule 44: "*respiratoryInsufficiency*":

IF	*patient*	*respiratoryInsufficiency*	**AND**
	lactate	*upToMinor*	
THEN			
	explanation	*respiratoryInsufficiency*	0.9

Description: The stress associated with respiratory insufficiency can cause elevated lactate levels.

Rule 45: "*sepsis*":

IF	*lactate*	*upToMinor*	**AND**
	patient	*sepsis*	**OR**
	lab	*sepsis*	
THEN			
	explanation	*sepsis*	0.9

Description: The stress associated with sepsis can cause elevated lactate levels.

Rule 46: "*sepsisLab*":

IF	*CRP*	*elevated*	
THEN			
	lab	*sepsis*	0.9

Description: Highly elevated CRP is indicative of sepsis.

Rule 47: "*shock*":

IF	*patient*	*shock*	**AND**
	lactate	*upToBigMinor*	
THEN			
	explanation	*shock*	0.9

Description: A low cardiac output can cause high lactate levels.

Rule 48: "*subarachnoidalHemorrhage*":

IF	*patient*	*subarachnoidalHemorrhage*	**AND**
	lactate	*upToMinor*	
THEN			
	explanation	*subarachnoidalHemorrhage*	0.9

Description: Subarachnoidal bleeding can cause elevated lactate and glucose levels.

Appendix C

Vignette Explanations

This appendix contains the data used to create scores table (Table 5.1 on page 53). For each vignette, the intended explanation and the explanations given by the participants (in both conditions) are shown, as well as the score given for said answer.

The scores are presented using the same glyphs as in table 5.1:

- : Correct answer.
- : Partly correct answer.
- : Incorrect answer.
- : No score.

Vignette: 1 (Patient 178)
Explanation: Hyperventilation

Participant	Explanation	Score
	Without App	
1	**Hyperventilation**	●
2	Myocardial ischemia	○
3	Heart failure or Pneumonia	○
4	Metabolic alkalosis caused by an infection	○
5	Unknown (+ possibly myocardial ischemia)	○
6		◌
7	**Hyperventilation** caused by heart failure	●
8	Unknown	○
9	Circulatory problems	○
10	Myocardial infarction + Hypoperfusion	○
	With App	
1	**Hyperventilation**	●
2	**Hyperventilation** & pneumonia	●
3		◌
4	Major Trauma	○
5	**Hyperventilation**, pneumonia & respiratory insufficiency	●
6		◌
7	**Hyperventilation**, pneumonia & respiratory insufficiency	●
8	**Hyperventilation**	●
9	**Hyperventilation**	●
10	**Hyperventilation** & respiratory insufficiency	●

Vignette: 2 (Patient 216)
Explanation: Sepsis (+ Pneumonia)

Participant	Explanation	Score
	Without App	
1	**Pneumosepsis**	●
2	**Pneumosepsis** + organ failure + liver failure	●
3	**Pneumosepsis**	●
4	**Pneumonia**	◒
5	**Pneumonia** + COPD (+ Sepsis)	●
6	Circulatory insufficiency + diabetes (+ **pneumonia**)	◒
7	COPD exacerbation	●
8	**Pneumonia** + dehydration	◒
9	Reduced cardiac output	○
10	**Pneumosepsis** (+ Diabetic keto-acidosis)	●
	With App	
1	Multiple organ failure, **pneumonia**, respiratory insufficiency, **sepsis**, shock	●
2	Multiple organ failure, **pneumonia**, respiratory insufficiency, **sepsis**, shock	●
3	Multiple organ failure, **pneumonia**, respiratory insufficiency, **sepsis**	●
4	**Pneumonia**, respiratory insufficiency	◒
5	Multiple organ failure, **pneumonia**, respiratory insufficiency, **sepsis**, shock	●
6	**Pneumonia**, respiratory insufficiency, **sepsis**	●
7	**Pneumonia**, respiratory insufficiency, **sepsis**	●
8	**Pneumonia**, respiratory insufficiency, shock	◒
9	**Pneumonia**, respiratory insufficiency	◒
10	**Pneumonia**, respiratory insufficiency, **sepsis**, shock	●

Vignette: 3 (Patient 287)
Explanation: Metformin intoxication. (MALA)

Participant	Explanation	Score
	Without App	
1	**MALA** (+ Diabetic keto-acidosis)	●
2	**MALA**	●
3	Gastroenteritis or **MALA** caused by dehydration	●
4	**MALA**	●
5	**MALA**	●
6	Bowel ischemia caused by pancreatitis + diabetes / **MALA**	●
7	**MALA** + Diabetic keto-acidosis	●
8	Diabetic keto-acidosis	○
9	**MALA**	●
10	**MALA**	●
	With App	
1	Shock & **metformin intoxication**	●
2	Shock & **metformin intoxication**	●
3	**Metformin intoxication**	●
4	**Metformin intoxication**	●
5	Shock & **metformin intoxication**	●
6	Shock & **metformin intoxication** (+ metformin use)	●
7	Shock & **metformin intoxication**	●
8	Shock	○
9	**Metformin intoxication**	●
10	**Metformin intoxication**	●

Vignette: 4 (Patient 326)
Explanation: CO intoxication (+ possibly augmented by metformin).

Participant	Explanation	Score
	Without App	
1	MALA (+ CO intoxication)	●
2	MALA	◒
3	MALA + (Diabetic keto-acidosis)	◒
4	Metabolic acidosis	○
5	MALA	◒
6		◌
7	MALA	◒
8	Diabetes (+ Cold diuresis + dehydration)	○
9	MALA	◒
10	Carbon monoxide intoxication	●
	With App	
1	**CO Intoxication**	●
2	**CO Intoxication** & **Metformin intoxication**	●
3		◌
4	**CO Intoxication**	●
5	**CO Intoxication**	●
6		◌
7	**CO Intoxication**	●
8	**CO Intoxication**	●
9	**CO Intoxication**	●
10	**CO Intoxication**	●

Vignette: 5 (Patient 489)
Explanation: Unknown. (Answers were not scored)

Participant	Explanation	Score
	Without App	
1	Respiratory insufficiency	
2	Heart failure + dehydration	
3	Heart failure + insufficient tissue perfusion	
4	Heart failure	
5	Respiratory or circulatory problems	
6	Cardial causes	
7	Heart failure	
8	Heart failure	
9	Circulatory	
10	Hypoperfusion	
	With App	
1	Respiratory insufficiency	
2	Pneumonia	
3	Unknown	
4	Unknown	
5	Pneumonia & respiratory insufficiency	
6	Hyperventilation	
7	Pneumonia & respiratory insufficiency	
8	Unknown	
9	Unknown	
10	Respiratory insufficiency	

Vignette: 6 (Patient 504)
Explanation: Recent generalized seizures.

Participant	Explanation	Score
	Without App	
1	**Seizures**	●
2	Reduced tissue perfusion due to bad oxygenation from rib fractures	○
3	**Generalized seizures** or heart failure resulting in insufficient perfusion	●
4	Heart failure or injuries sustained due to seizures	○
5	Hyperglycemia	○
6	MALA + Circulatory problems (+ Ischemia)	○
7	**Generalized seizures**	●
8	**Generalized seizures**	●
9	**Generalized seizures**	●
10	**Generalized seizures** + Hypoperfusion	●
	With App	
1	**Generalized seizures**	●
2	**Generalized seizures**	●
3	**Generalized seizures**	●
4	**Generalized seizures**	●
5	**Generalized seizures**	●
6	**Generalized seizures**	●
7	**Generalized seizures**	●
8	**Generalized seizures**	●
9	**Generalized seizures**	●
10	**Generalized seizures**	●

Vignette: 7 (Patient 698)
Explanation: Cardiogenic shock (secondary to myocardial infarction)(+ renal failure).

Participant	Explanation	Score
	Without App	
1	Flow problem / **cardiogenic shock**	●
2	**Myocardial infarction + renal failure**	●
3	**Cardiogenic shock** + collapse	●
4	**Myocardial infarction**	◒
5	**Myocardial infarction**	◒
6	**Cardiogenic shock**	●
7	**Heart failure (Myocardial infarction)**	◒
8	**Heart failure**	◒
9	**Heart failure**	◒
10	Hypoperfusion pulmonary edema + **Cardiogenic shock**	●
	With App	
1	Unknown	○
2	**Cardiogenic shock**	●
3	**Cardiogenic shock**	●
4	**Cardiogenic shock**	●
5	**Cardiogenic shock**	●
6	**Cardiogenic shock**	●
7	**Cardiogenic shock**	●
8	**Cardiogenic shock**	●
9	Recent cardiac arrest	○
10	**Cardiogenic shock**	●

Vignette: 8 (Patient 745)
Explanation: Inotropes, more specifically direct 2-mimetic effect of isoprenalin.

Participant	Explanation	Score
	Without App	
1	Bradycardia / flow problem	○
2	Low blood pressure due to arrhythmia combined with dehydration	○
3	Metabolic acidosis due to insufficient tissue perfusion	○
4	Unknown	○
5	Heart failure	○
6	Metabolic acidosis + circulatory insufficiency	○
7	Out of hospital cardiac arrest	○
8	Maybe **isoprenaline**	●
9	Circulatory	○
10	Hypoperfusion	○
	With App	
1	**Inotropes** & respiratory insufficiency (+ COPD)	●
2	**Inotropes** & shock	●
3		○
4	**Inotropes** (+ COPD)	●
5	**Inotropes**, respiratory insufficiency & shock (+ COPD)	●
6	Shock (+ COPD)	○
7	Recent cardiac arrest & shock (+ COPD)	○
8	**Inotropes**	●
9	**Inotropes**	●
10	**Inotropes** (+ COPD)	●

Vignette: 9 (Patient 847)
Explanation: Respiratory Insufficiency n.o.s + Hypotension (+ Diabetic Keto-acidosis).

Participant	Explanation	Score
	Without App	
1	**Pneumonia** (+ **Diabetic keto-acidosis**)	●
2	**Pneumonia** + Dehydration	●
3	**Pneumosepsis** + **hypotension** + ketones in urine	●
4	Respiratory acidosis due to infection	○
5	**Pneumosepsis** (+ **Diabetic keto-acidosis**)	●
6		◌
7	**Diabetic keto-acidosis** (+ **Hypotension** + **pneumonia**)	●
8	Unknown	○
9	Sepsis + **circulatory**	◒
10	**Diabetic keto-acidosis**	◒
	With App	
1	**Pneumonia**, respiratory insufficiency & **shock**	●
2	Deranged DM, **pneumonia**, sepsis, **shock** (+ **diabetic keto-acidosis**)	●
3		◌
4	Deranged DM (+ **diabetic keto-acidosis**)	◒
5	**Pneumonia** & sepsis	●
6		◌
7	Deranged DM, **pneumonia** & sepsis (+ **diabetic keto-acidosis**)	●
8	Deranged DM (+ **diabetic keto-acidosis**)	◒
9	Deranged DM (+ **diabetic keto-acidosis**)	◒
10	Deranged DM (+ **diabetic keto-acidosis**)	◒

Vignette: 10 (Patient 925)
Explanation: Acute-on-Chronic liver failure.

Participant	Explanation	Score
	Without App	
1	Shock + **liver failure** + (renal failure)	●
2	**Decompensated liver cirrhosis** + reduced renal function	●
3	**Liver failure** + reduced tissue perfusion	●
4	**Liver failure**	●
5	**Liver failure**	●
6	**Liver failure** (+ Circulatory problems)	●
7	**Liver failure**	●
8	**Liver failure** (+ Elevated ammonia levels)	●
9	Hemorrhagic shock + **liver failure**	●
10	**Liver failure** (+ Hypoperfusion)	●
	With App	
1	**Acute hepatic failure** & **chronic liver failure**	●
2	**Acute hepatic failure** & **chronic liver failure**	●
3	**Acute hepatic failure** & **chronic liver failure**	●
4	**Acute hepatic failure** & **chronic liver failure**	●
5	**Acute hepatic failure** & **chronic liver failure**	●
6	**Acute hepatic failure** & **chronic liver failure**	●
7	**Acute hepatic failure** & **chronic liver failure**	●
8	**Acute hepatic failure** & **chronic liver failure**	●
9	**Acute hepatic failure** & **chronic liver failure**	●
10	**Acute hepatic failure** & **chronic liver failure**	●

Appendix D

Participant Instructions

Introduction

Thank you for taking part in the lactate app experiment.

The goal of this experiment is to gain insight in the reasoning process involved in the finding an explanation for elevated lactate levels, and to test and evaluate a new web-based application, designed to assist in this process.

The experiment will consist of two parts. Both involve a series of vignettes: brief descriptions of a (fictional) patient containing all currently known information, including their lactate level.

For the first part of the experiment, you'll be asked to read these vignettes, and try to find an explanation for the lactate level without the use the app. Whilst doing this, you'll be asked to think out loud, and you will be recorded by a microphone.

For the second part, you'll be asked to go through these vignettes again, but this time with the aid of the application. This application will ask you a series of questions about the patient, and will eventually present what it thinks is a sufficient explanation for the lactate. Like the previous part, you'll be asked to think out loud whilst being recorded.

The experiment will take around 60 minutes. The two parts will be explained in more detail once you begin the experiment. In order to keep everything consistent, all written instructions, vignettes and the app are in English. However, you are free to speak either English or Dutch.

All data will be strictly used for research purposes only, and will be treated in a confidential manner. All collected data will be tied to a participant number that is not linked to your personally in any way. To participate in this experiment, you will have to sign a consent form.

If you run into any problems, feel free to ask the experimenter (Lourens) for assistance.

Part 1 - Instructions

The first part of this experiment will look at the thought process of trying to find an explanation for elevated lactate levels.

You will be provided with a series of vignettes. As said in the introduction, these contain the description of a patient, containing their lactate level and all other information that is currently known about them. When you're ready, give an audible signal to the experimenter, and flip over the vignette. Then, whilst thinking out loud, try to find an explanation for the lactate level of the patient. Once you think you've found an explanation, give the experimenter a signal again, and explain what you think the explanation is, and the reasons for the explanation. Then, put aside the vignette and proceed to the next one.

This is not an exam. The goal is not to test you, but to gain insight in the reasoning process involved in solving the vignettes. It does not matter whether the explanation you find is correct, nor will you receive feedback on your answers.

Part 2 - Instructions

You will now look at the same vignettes, and use the app to find an explanation for the lactate levels.

The app will be running on the computer in front of you. You will be presented with a series of questions. Answer these questions given the information available in the vignette, until you're presented with the a screen showing which explanation(s) the app has found. You can click on the explanations to view the reasoning, if you are interested. When finished, you can prepare for the next vignette by clicking on 'Restart elicitation'.

As before, the vignettes will be given to you. When you're ready, give the experimenter a signal and turn over the next vignette, and start answering the questions provided by the app, whilst thinking out loud. Once you've reached the results, give the experimenter a signal again, and explain whether you agree with the shown explanation(s) or not, and try to explain why this is the case. You can look at the reasoning to get better insight.

Appendix E

Consent Form

Informed consent

I agree that I will participate in the study called the lactate app experiment, conducted by Lourens Elzinga (Master Human-Machine Communication, University of Groningen), Dr. Fokie Cnossen (Institute of Artificial Intelligence and Cognitive Engineering), and Dr. Maarten W.N. Nijsten (University Medical Centre Groningen, Department of Intensive Care Medicine).

I am aware that participation in this study is voluntary; I may refuse to participate without penalty. If I decide to participate, I may withdraw from the study at any time without penalty of loss of benefits to which I am otherwise entitled. If I withdraw from the study before data collection is completed, my data will be destroyed. I also have the right to request that my data not be used.

The following has been explained to me:

- The purpose of the experiment is to gain insight in the reasoning process of interpreting lactate levels, and testing test a new application, designed to aid in this process.
- I will be asked to read patient descriptions.
- Audio recordings will be made during the experiment.
- There are no foreseeable risks or discomforts involved in this study.
- All data gathered in this experiment are stored anonymously, and no data will be stored or reported in such a way that it is traceable to individuals.
- All recorded audio will be deleted after 18 months.
- The researcher will answer any questions I might have about the experiment.

Participant

.....
Name

....
Signature Date

Appendix F

Questionnaire

		Strongly Disagree				Strongly Agree
1.	I think that I would like to use this system frequently.	◯	◯	◯	◯	◯
2.	I found the system unnecessarily complex.	◯	◯	◯	◯	◯
3.	I thought the system was easy to use.	◯	◯	◯	◯	◯
4.	I think that I would need the support of a technical person to be able to use this system.	◯	◯	◯	◯	◯
5.	I thought there was too much inconsistency in this system.	◯	◯	◯	◯	◯
6.	I would imagine that most people would learn to use this system very quickly.	◯	◯	◯	◯	◯
7.	I found the system very cumbersome to use.	◯	◯	◯	◯	◯
8.	I needed to learn a lot of things before I could get going with this system.	◯	◯	◯	◯	◯
9.	This system has all the functions and capabilities I expect such a system to have.	◯	◯	◯	◯	◯
10.	The organization of information on the system screens was clear.	◯	◯	◯	◯	◯
11.	I trust the answers given by this system.	◯	◯	◯	◯	◯
12.	I think this system is suitable for training.	◯	◯	◯	◯	◯
13.	I think this system could be used in the clinic.	◯	◯	◯	◯	◯
14.	I think the questions from the system were clear.	◯	◯	◯	◯	◯
15.	I found it hard to understand the reasoning of this system.	◯	◯	◯	◯	◯
16.	I found it easy to get insight in the reasoning of this system.	◯	◯	◯	◯	◯
17.	I found it easy to understand the reasoning of this system.	◯	◯	◯	◯	◯
18.	Overall, I am satisfied with this system.	◯	◯	◯	◯	◯

Appendix G

Questionnaire Results

		Strongly Disagree				Strongly Agree
1.	I think that I would like to use this system frequently.	0	0	7	2	1
2.	I found the system unnecessarily complex.	6	3	1	0	0
3.	I thought the system was easy to use.	0	0	0	5	5
4.	I think that I would need the support of a technical person to be able to use this system.	7	2	1	0	0
5.	I thought there was too much inconsistency in this system.	1	5	4	0	0
6.	I would imagine that most people would learn to use this system very quickly.	0	0	0	2	8
7.	I found the system very cumbersome to use.	5	3	1	1	0
8.	I needed to learn a lot of things before I could get going with this system.	8	1	0	1	0
9.	This system has all the functions and capabilities I expect such a system to have.	0	3	5	2	0
10.	The organization of information on the system screens was clear.	0	0	2	5	3
11.	I trust the answers given by this system.	0	0	5	5	0
12.	I think this system is suitable for training.	0	0	1	6	3
13.	I think this system could be used in the clinic.	0	2	1	6	1
14.	I think the questions from the system were clear.	0	1	2	2	5
15.	I found it hard to understand the reasoning of this system.	1	8	1	0	0
16.	I found it easy to get insight in the reasoning of this system.	0	2	1	5	2
17.	I found it easy to understand the reasoning of this system.[1]	0	0	1	8	0
18.	Overall, I am satisfied with this system.	0	0	2	6	2

[1] One participant left this question unanswered.

Appendix H

Interview Questions

1. How did the experiment go for you?
2. How difficult did you find the first part of the experiment?
3. How difficult did you find the second part of the experiment?
4. How did you like using the app?
5. What do you think of the app itself, excluding the model?
6. What do you think of the model?
7. What do you think of the order of questions given by the app?
8. How much does the way the app works differ from the way you would normally interpret lactate levels?
9. Do you think the way the app works is a better way to interpret lactate levels?
10. Would you like to use this app at work? How do you envision this?
11. Do you think an app like this could be useful in other (non-lactate) areas?
12. Are there things you didn't like about the app or that you think should be changed?
13. Is there anything else you would like to mention regarding the app or the experiment?

Appendix I

Interview results

This appendix contains the results of all interviews. These are not full transcripts, but instead summarize the answers in one or more key points. While the interviews were conducted in Dutch, the answers presented in this appendix are translated into English.

Participant 1 (11548)

Notes: During the evaluation, the model contained an error resulting in the omission of an important question for patient 698 which resulted in the app not finding an explanation. This error was corrected immediately after this interview.

How did the experiment go for you?
- Fine, no comments on the experiment itself.

How difficult did you find the first part of the experiment?
- Difficulty depended on the vignette, some were easy, some difficult.

How difficult did you find the second part of the experiment?
- Quite easy, the app was clear.
- Except for one vignette, where the app did not give the correct answer (see note), the app mostly came to the same conclusions.

How did you like using the app?
- Would personally use the app in cases of uncertainty.
- Good as a tool to jog your memory, by going through the questions and seeing if there is something that was overlooked or not considered and to help steer in the right direction.
- Some cases are so clear that there is no need to use the app.

What do you think of the app itself, excluding the model?
- Clear and easy to use.
- The use of N/A may need some additional explanation. Otherwise it might cause people to leave unknown values at 0 instead of using N/A.

What do you think of the model?
- Unclear why the questions presented differed between vignettes.
- Some more explanation would be nice.

- The questions are mostly logical, except the one case the app did not ask about the myocardial infarction (see note).
- In practice there is likely to be more information present about patients than was available in the vignettes, which would improve the results of the app.
- The app is rather dependent on own diagnoses (e.g shock), so in the case of a misdiagnosis, the app would not reach the right conclusion. Perhaps questions should be made more objective, so it becomes less dependent on the user.

What do you think of the order of questions given by the app?

- It always stated with patient history, which makes sense.
- Liver failure seems quite important to the app, as it was always asked first. In practice however, this does not occur very frequently.

How much does the way the app works differ from the way you would normally interpret lactate levels?

- The app does not always consider all available information, which could make it jump to conclusions.
- The app seems to jump to conclusions quite quickly, even though there could be other factors that play a role in the explanation.
- The reasoning can be quite predictable. (i.e. answering yes to a question leading to that being the explanation)
- The model needs more nuance.
- Despite the tendency to jump to conclusions, the conclusion in question in usually an agreeable one.

Do you think the way the app works is a better way to interpret lactate levels?

- Not better, nor worse.
- Possibly more structured.

Would you like to use this app at work? How do you envision this?

- Yes, especially in cases of doubt.
- Useful as a way to double-check.
- Would be good on a PC or mobile.
- Especially useful for training purposes.

Do you think an app like this could be useful in other (non-lactate) areas?

- Useful for hyponatremia, especially because there is a clear decision tree involved.
- Same for hypokalemia.
- The app might be even more useful for these than lactate. Lactate is a very important measure, but a vague one, while hyponatremia and hypokalemia are clearer.

Are there things you didn't like about the app or that you think should be changed?

- Nothing that hasn't already been said.

Is there anything else you would like to mention regarding the app or the experiment?

- No.

Participant 2 (21844)

How did the experiment go for you?
- Went pretty well.
- May have needed to read things more closely.

How difficult did you find the first part of the experiment?
- Some were easy, some were difficult.
- Couldn't find the explanation of the lactate for some of the vignettes.

How difficult did you find the second part of the experiment?
- Easier.
- Learned new explanations.
- It helped to see the cases for the second time.

How did you like using the app?
- User-friendly and clear.

What do you think of the app itself, excluding the model?
- Very clear, nice fonts.
- Looks good.
- Potentially present more questions on the same page.

What do you think of the model?
- Almost always adequate.
- Model seems quite accurate and contains most relevant things.

What do you think of the order of questions given by the app?
- Makes sense, uses the same order as cases are normally presented as.
- Sometimes misses things related to physical examination.

How much does the way the app works differ from the way you would normally interpret lactate levels?
- It works quite similarly, but the app can present things that may have been overlooked. The app is useful as a memory aide.

Do you think the way the app works is a better way to interpret lactate levels?
- You should always think for yourself and not blindly follow the app, but the app is useful to prevent you from overlooking things.

Would you like to use this app at work? How do you envision this?
- Yes.
- First try to find an explanation on your own and then use the app to see if anything was overlooked.
- In very trivial cases, the use of the app is unnecessary.
- Useful on any device with quick access (e.g. mobile or PC)

Do you think an app like this could be useful in other (non-lactate) areas?
- Yes, for example in cases of low sodium levels.

Are there things you didn't like about the app or that you think should be changed?

- The first set of questions that don't change should be presented on the same page.
- Don't use 0 as default values for numbers, but make them empty instead. Removes the need to first remove the 0 before typing.
- Having the question mark pre-selected (for yes/no questions) might be a bit confusing (e.g. have I already clicked it? / do you need to click it again?), so perhaps require it to be explicitly clicked. On the other hand, having those pre-selected makes it easier to get through questions, so it can go either way.

Is there anything else you would like to mention regarding the app or the experiment?
- No.

Participant 3 (26933)

How did the experiment go for you?
- Good.
- Very practical and easy.

How difficult did you find the first part of the experiment?
- Not very difficult.
- You don't see the patient, and there is limited information, which makes it difficult.

How difficult did you find the second part of the experiment?
- Difficult, because it asks things that are not known (e.g. does the patient have a lymphoma), you cannot provide it with all information it asks for.

How did you like using the app?
- Clear and easy to use.

What do you think of the app itself, excluding the model?
- Very clear.

What do you think of the model?
- It is lacking the use of clinical measures, like blood pressure.
- It asks for hypothermia, but not fever.
- It does not ask for inflammatory parameters.
- There are things you don't know that the model doesn't take into consideration.

What do you think of the order of questions given by the app?
- The order is good.

How much does the way the app works differ from the way you would normally interpret lactate levels?
- Not very much.
- The app usually came to the same conclusions.
- Their way of looking at the vignette was very similar to the order in which the questions were presented.

Do you think the way the app works is a better way to interpret lactate levels?
- No.

- Came to the same conclusions as the app on their own, but the app is still too limited and doesn't consider the whole picture.

Would you like to use this app at work? How do you envision this?
- It doesn't have an added value, especially for these cases.
- Perhaps useful in cases where there is doubt.
- If the model is expanded and cases get complicated, then the app may have more use.
- Useful to have on both a PC and mobile.

Do you think an app like this could be useful in other (non-lactate) areas?
- Maybe, but not sure where.

Are there things you didn't like about the app or that you think should be changed?
- Make the model more comprehensive, though that may result in it taking more time to use it.
- Expand the definition of multi organ failure.

Is there anything else you would like to mention regarding the app or the experiment?
- No.

Participant 4 (40848)

Notes: Participant was a co-assistant.
How did the experiment go for you?
- Somewhat difficult.
- Not very familiar with lab measurements.
- Good cases.
- The app does make it very easy.

How difficult did you find the first part of the experiment?
- Rather difficult.
- Needed to get into it, needed to refresh knowledge on lactate.
- Got easier as the time progressed.

How difficult did you find the second part of the experiment?
- Very, very easy. Mostly just simple yes/no questions.
- Some questions were slightly more difficult, such as shock. It is easier to determine when seeing an actual patient, than to infer it from a vignette.

How did you like using the app?
- Very easy.
- If you disagree, you can simply try again.

What do you think of the app itself, excluding the model?
- Great, clear.
- No unnecessary bells and whistles.

What do you think of the model?
- It makes sense, and presents the most probably explanation.

- In practice, this may not always be the correct explanation.

What do you think of the order of questions given by the app?
- Didn't fully understand. May be the result of the participants own experience (see note).

How much does the way the app works differ from the way you would normally interpret lactate levels?
- When looking at patient description, the first impulse is to look at the patient's acute condition and consider ways of treatment, not necessarily trying to explain why the lactate is so high.

Do you think the way the app works is a better way to interpret lactate levels?
- It is very good for a specific problem: finding out the explanation of lactate.
- Not necessarily better or worse, but an alternative way.
- Some cases are very acute, and in such cases the cause of high lactate is of lesser importance.

Would you like to use this app at work? How do you envision this?
- Yes, quite useful.
- Preferably on mobile.
- Useful for training purposes. Teaches causes of elevated lactate.

Do you think an app like this could be useful in other (non-lactate) areas?
- Possibly, but depends on what.

Are there things you didn't like about the app or that you think should be changed?
- Not that I can think of.

Is there anything else you would like to mention regarding the app or the experiment?
- No.

Participant 5 (44790)

How did the experiment go for you?
- Needed to get back into it, refresh medical knowledge.
- Experimental setup was clear.

How difficult did you find the first part of the experiment?
- Harder than the second part.
- Struggled with the first few vignettes, but improved over time.

How difficult did you find the second part of the experiment?
- It went a bit faster than the first part.
- It was easier to have faith in their own diagnosis when the app agreed with it, which was often the case. Except for the carbon monoxide vignette, where the HbCO lab value was overlooked.

How did you like using the app?
- Quite handy.
- Especially to refresh ones memory.

- If the user is more experienced, the app may not have as much added value.

What do you think of the app itself, excluding the model?
- Clear.

What do you think of the model?
- Some confusion about the timeline of questions, e.g., is an acute diagnosis something that occurs right now, is explicitly mentioned, or can you infer it, or is it an explanation for something else, etc.
- Some question-answer relations are rather obvious.

What do you think of the order of questions given by the app?
- Odd that it starts with liver failure.

How much does the way the app works differ from the way you would normally interpret lactate levels?
- The app does not consider the story of the patient.
- Sometimes you recognise things in the patient that lead to an explanation.
- The app does this differently, looking at individual pieces, instead of the whole picture.

Do you think the way the app works is a better way to interpret lactate levels?
- It's good that it gives multiple options, with certainties.
- It is good to avoid tunnel vision.

Would you like to use this app at work? How do you envision this?
- Particularly useful for training and/or memory aids.
- Useful for people less familiar with lactate.
- More experienced people have less need of such app and can rely on pattern recognition.

Do you think an app like this could be useful in other (non-lactate) areas?
- In principle, yes. Blood gas measurements for example, and possibly other areas.

Are there things you didn't like about the app or that you think should be changed?
- For sex, have two buttons instead of a dropdown list.
- Some questions are very similar and can cause confusion (e.g. respiratory insufficiency n.o.s. and 'respiratory insufficiency / pneumonia')

Is there anything else you would like to mention regarding the app or the experiment?
- Experiment was enjoyable.
- App is very useful for educational purposes.

Participant 6 (59897)

How did the experiment go for you?
- Neutral, an experiment is an experiment.

How difficult did you find the first part of the experiment?
- Not difficult.
- Would have liked to get some more information about the patients.

How difficult did you find the second part of the experiment?
- Not more difficult than without.
- Not sure about added value of the app.

How did you like using the app?
- Easy, clear.

What do you think of the app itself, excluding the model?
- Quite clear, simple yes/no/don't know questions.

What do you think of the model?
- Agreed with the model mostly, except one case where the app came to a premature conclusion.

What do you think of the order of questions given by the app?
- It doesn't really matter.
- Perhaps lab questions should be asked first, in particular blood gas values.

How much does the way the app works differ from the way you would normally interpret lactate levels?
- One has to think pragmatically. Hyperlactatemia is most often associated with bad circulation. So focus on haemodynamics and possibly relevant elements in prior history. Also, it is often present in metabolic disorders and generalized seizures. But overall, search for problems in haemodynamics, ischemia, etc. This differs from the app presents questions. The more common the cause, the earlier it should appear in the decision tree.

Do you think the way the app works is a better way to interpret lactate levels?
- Not better, nor worse.
- It could have added value if the app contained more rare causes. Everyone knows the big causes

Would you like to use this app at work? How do you envision this?
- Possibly, if the is improved.
- Preference for smartphone or tablet.

Do you think an app like this could be useful in other (non-lactate) areas?
- Differential diagnosis is always important. If a good tool can be designed for other areas, such as respiratory insufficiency, or causes of seizures, it would be very useful.

Are there things you didn't like about the app or that you think should be changed?
- It's a bit simplistic, and in its current state somewhat superfluous.

Is there anything else you would like to mention regarding the app or the experiment?
- No.

Participant 7 (70419)

How did the experiment go for you?
- Went through the cases very quickly, while that would not be the case in a real sitation.

- Went to conclusions quickly and overlooked things that were only noticed upon viewing the vignettes again.

How difficult did you find the first part of the experiment?
- Not very used to it.
- It was fairly easy.

How difficult did you find the second part of the experiment?
- More structured.
- There is a risk of blindly following the app, of trusting it too much, while you would not do that in reality.

How did you like using the app?
- Clear.
- Good that results show a certainty.
- Certainties may need more nuance.

What do you think of the app itself, excluding the model?
- Don't add zeroes to numeric fields, but leave them empty.
- Yes/no questions should instead use more of a scale (e.g. likert scale).

What do you think of the model?
- The explanations were often rather generic, but given the information on the vignettes, it is probably not possible to get more precise explanations.

What do you think of the order of questions given by the app?
- The patient history often omitted questions that could have been relevant.
- The model may give too much importance to liver failure as a cause for high lactate, while this does not occur often in practice.

How much does the way the app works differ from the way you would normally interpret lactate levels?
- The app provides more structure.
- Differential diagnosis apps are definitely good and will be more prevalent in the future, where you can just enter some information and get a list of possible causes, so you don't overlook something like a CO intoxication.

Do you think the way the app works is a better way to interpret lactate levels?
- Not better or worse.
- In theory everything is rather rigid, while in practice there is a lot more nuance.
- The app mentions causes that they would not necessarily consider themselves.

Would you like to use this app at work? How do you envision this?
- Not on the PC, but rather on a smart phone or tablet. That way it is always on hand.

Do you think an app like this could be useful in other (non-lactate) areas?
- Definitely.
- Useful for rarer areas.

Are there things you didn't like about the app or that you think should be changed?
- As said before, leave numeric fields blank instead of using 0.

- Avoid premature differentiation.

Is there anything else you would like to mention regarding the app or the experiment?
- No.

Participant 8 (71428)

Notes: Participant took part in the pilot study.

How did the experiment go for you?
- Good.

How difficult did you find the first part of the experiment?
- It was a bit difficult because there is more information about patients in practice.

How difficult did you find the second part of the experiment?
- Clear.
- Occasionally entered wrong answers.

How did you like using the app?
- Useful.
- It helps and avoids overlooking things.

What do you think of the order of questions given by the app?
- The order makes sense.
- In practice, you have a better understanding of the patient and don't necessarily think in a specific order. But overall, it follows the order normally used when describing patients.

How much does the way the app works differ from the way you would normally interpret lactate levels?
- It differs because it doesn't consider the progression of the patient's condition, however, if the patient just arrived, that information would not be available anyway.

Do you think the way the app works is a better way to interpret lactate levels?
- It keeps you grounded when thinking about it.

Would you like to use this app at work? How do you envision this?
- Depends on how much time is available, and on the patients in question.
- Useful in case of uncertainty.
- PC or mobile

Are there things you didn't like about the app or that you think should be changed?
- Leave numeric fields empty, instead of using zeroes.
- Include better explanations for questions, so it is clearer when to answer yes.

Is there anything else you would like to mention regarding the app or the experiment?
- No.

Participant 9 (89636)

Notes: Participant took part in the pilot study.

How did the experiment go for you?
- The vignettes could have contained some more information.
- Interesting cases and in a particular case an eye opener.

How difficult did you find the first part of the experiment?
- It went pretty well.
- It felt more natural than going through a series of questions.

How difficult did you find the second part of the experiment?
- More questions.
- In the first part you make a selection of things you find important and go with those, whereas the app guides you towards things that you may have initially ignored.
- It takes longer, but may reveal things that were otherwise overlooked.

How did you like using the app?
- Good overall.
- Add the ability to do corrections, instead of having to start over. [1]

What do you think of the order of questions given by the app?
- It's all about probabilities, and it makes sense to start with history and end with lab values.
- It follows the way the vignettes are read.

How much does the way the app works differ from the way you would normally interpret lactate levels?
- Not much.
- The app tries to find things to associate with lactate, and we do the same. We try to interpret them in the light of a greater whole, and the app does that too.

Do you think the way the app works is a better way to interpret lactate levels?
- Sometimes you overlook things, either for being in a rush, or not knowing something can be a cause. The app helps in these cases by pointing them out, in particular in the latter case.

Would you like to use this app at work? How do you envision this?
- Useful in cases of uncertainty.
- There is a risk of making a wrong diagnosis, but thinking the diagnosis is correct and therefore not using the app to double check.
- Maybe integrate on polieplus, if possible.
- Not very interested in using it on a phone, as it is a private device and not work related.

Are there things you didn't like about the app or that you think should be changed?
- Add the ability to correct answers, as said before.

Is there anything else you would like to mention regarding the app or the experiment?

[1] Was added after pilot.

- No.

Participant 10 (97352)

How did the experiment go for you?
- Good.

How difficult did you find the first part of the experiment?
- A bit difficult.
- Would've liked more information on the patients.

How difficult did you find the second part of the experiment?
- Still would have liked more information.
- Definitely helps in differential diagnoses.
- App sometimes jumps to conclusions.

How did you like using the app?
- Easy, clear.

What do you think of the app itself, excluding the model?
- Very clear.
- Very easy to use.

What do you think of the model?
- It felt like the app changed the questions based on patient ID.[2]
- Sometimes the explanations given are not very good. For example: The explanation given is respiratory insufficiency, but there is no information about the underlying cause or how to treat the patient.

What do you think of the order of questions given by the app?
- Hadn't considered the order of questions.
- The app can think along in the differential diagnosis.
- The order is good because it requires you to go through the patient information in a structured way instead of focussing on something specific.

How much does the way the app works differ from the way you would normally interpret lactate levels?
- Not much.
- It is very structured, and this is how it should be done.
- The difference also depends on the setting in which it is used. For example people in the emergency room may interpret lactate differently from people in the intensive care unit.

Do you think the way the app works is a better way to interpret lactate levels?
- It should be combined, not just one or the other. In one case the app gave a very good differential diagnosis, while in another case they disagreed with the app and wanted to overrule it.
- It must be synergistic. Don't blindly follow the app, but also don't blindly follow your own ideas.

[2] Explained to the participant this was caused by the lactate, not the patient id.

- Useful as memory aid.

Would you like to use this app at work? How do you envision this?

- Yes, in particular if the model is improved.
- It could have added value for the differential diagnosis.
- PC or mobile.

Do you think an app like this could be useful in other (non-lactate) areas?

- Be careful not to make an app for every measurement. This makes things too easy and stops people from actually learning it themselves. For lactate it makes sense, but if used for other measures like sodium or potassium, there would be no end to it. You'd spend half an hour clicking.

Are there things you didn't like about the app or that you think should be changed?

- The model sometimes jumps to conclusions.
- The explanations should be more meaningful and give an indication on how to treat them.

Is there anything else you would like to mention regarding the app or the experiment?

- No.