Evaluation of Media Interpretation Algorithms

Master Thesis

by

Björn Haß

Start date:	1st March 2011
End date:	4th October 2001
Supervising Professor:	Prof. Dr. rer. nat. habil. Ralf Möller
	Prof. Dr. Dieter Gollmann
Supervisor:	Dipl.-Ing. Anahita Nafissi
	Dipl.-Ing. Maurice Rosenfeld

Abstract

The CASAM Project is a innovative new approach to combine the manual annotation of video data with automated machine learning algorithms. Central part of the approach is the Reasoning for Multimedia Interpretation (RMI) engine, that is responsible for deducing new knowledge in the system. This part of the software has high demands in terms of quantity and quality of the results. In this thesis a Test Framework is developed to improve the parameters of the engine. The RMI engine is tested with several parameters and proposals for optimal settings are given.

Statement

Hereby I do state that this work has been prepared by myself and with the help which is referred within this thesis.

Hamburg, October 3th 2011

Foreword

First of all I want to thank Prof. Dr. rer. nat. habil. Möller for giving me the opportunity to work on this topic and also for the support. I also want to thank my supervisors Dipl.-Ing. Anahita Nafissi and Dipl.-Ing. Maurice Rosenfeld for all those inspiring discussions.

Hamburg,29th September 2011

Contents

Chapter 1

Motivation

1.1 Introduction

The development of digital technology and the Internet lead to an enormous increase in multimedia data. Annotation of this data is needed to handle this data efficiently. The common approach used today is the manual annotation of this data. This is a time and manpower consuming task that also leads to ambiguities in the annotation. New approaches try to automate this task by using machine algorithms to classify this data and automate the process of annotation. The CASAM approach tries to combine this two approaches, to minimize or eliminate the weaknesses of the other approaches.

The core element to enrich the knowledge of the system in the CASAM project is the RMI engine. It uses the formalism of abduction to deduce new knowledge. In this thesis a test framework is presented that evaluates the engine and optimizes the parameters of the engine in terms of optimal performance and output quality.

Chapter two gives a brief background knowledge to Description Logic. Chapter three explains the CASAM Project and its modules and focuses on relevant RMI module for this thesis. Chapter four gives an introduction to the used Test Framework. Chapter five discusses the results of the performed tests. Chapter six gives an overview over this thesis and an outlook for possible further research.

Chapter 2

Description Logic

2.1 Introduction

In this chapter a brief introduction to the Description Logic (DL) used in the CASAM Project is given. A deeper introduction can be found in [Baader and W. Nutt, 2007]. DL use elements of first-order logic, but stay decidable. DL consists of concepts, individuals and roles. A knowledge base (kb) used in DL consist of so called TBoxes and ABoxes it is defined as $\sum_S = (\mathcal{T},\mathcal{A})$ with the Signature S. TBoxes stand for terminological component, they describe a conceptualization, a set of concepts and properties for these concepts. ABox stands for assertion components. The statements in the ABox are TBox compliant statements about that vocabulary.

2.2 $\mathcal{ALH}_f\ ^-(\mathcal{D})$

In the CASAM project, reactivity and real time performance is a significant feature of the reasoning process. The DL used in the CASAM Project is $\mathcal{ALH}_f\ ^-(\mathcal{D})$. This DL has been choosen as a good trade off between expressivity and real-time requirements. An extensive discussion of the justification can be found in [Oliver Gries et al., b]

The Syntax of $\mathcal{ALH}_f\ ^-(\mathcal{D})$ is the minimal attributive language ($\mathcal{AL}$) extended with restricted attributive concept language with role hierarchies, functional roles and concrete domain objects. Concept descriptions in $\mathcal{ALH}_f\ ^-(\mathcal{D})$ are formed according to the following syntax rules (see [Oliver Gries et al., b]):

C,D $\longrightarrow$	A |	(atomic concept)
	$\top$ |	(universal concept)
	$\bot$ |	(bottom concept)
	$\neg$A |	(atomic negation)
	C $\sqcap$ D |	(intersection)
	$\forall$ R.C |	(value restriction)
	$\exists$ R.$\top$	(limited existential quantification)

The semantics of $\mathcal{ALH}_f\ ^-(\mathcal{D})$ is defined with interpretations $\mathcal{I} = (\Delta^\mathcal{I}, \cdot)^\mathcal{I}$, where $\Delta^\mathcal{I}$ is a non-empty set with all objects in $\mathcal{I}$ called domain. $\cdot^\mathcal{I}$ is an interpretation function which maps constants to objects of the domain ($a^\mathcal{I} \in \Delta^\mathcal{I}$, atomic concepts to subsets of the domain ($A^\mathcal{I} \subseteq \Delta^\mathcal{I}$) and roles to subsets of the Cartesian product of the domain ($R^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$). The interpretation of arbitrary $\mathcal{ALH}_f$ concept descriptions then is defined by extending $\cdot^\mathcal{I}$ to all $\mathcal{ALH}_f$ concept constructors as follows:

$$
\begin{array}{ll}
\top^\mathcal{I} & = \Delta^\mathcal{I} \\
\bot^\mathcal{I} & = \varnothing \\
(\neg A)^\mathcal{I} & = \Delta^\mathcal{I} \backslash A^\mathcal{I} \\
(C \sqcap D)^\mathcal{I} & = C^\mathcal{I} \sqcap D^\mathcal{I} \\
(\forall R.C)^\mathcal{I} & = \{u \in \Delta^\mathcal{I} | (\forall v)[(u,v) \in R^\mathcal{I} \rightarrow v \in C^\mathcal{I}]\} \\
(\exists R.\top)^\mathcal{I} & = \{u \in \Delta^\mathcal{I} | (\exists v)[(u,v) \in R^\mathcal{I}]\}
\end{array}
$$

Chapter 3

CASAM

3.1 Overview

The Computer Aided Semantic Annotation of Multimedia (CASAM) is a new approach for the annotation of multimedia data. Existing solutions are either based on human annotation or machine-derived. Both have pros and cons. Manual annotation by tagging each content by humans is the most usual approach today. This approach results in high quality annotations. The disadvantages are the expensivness of this approach because of the enormous time consumption this is not feasible for larger content. Also the possiblity of different taggings of the same content is high because of the different interpretation of humans. Some of this disadvantages can be reduced/eliminated by social tagging approaches. Another approach is the automatic tagging by using algorithms in computer vision, audio perception and OCR to aquire as much informations as possible from the multimedia data. The increased computational power in the last years makes it feasible to automatically tag large amount of multimedia content by computers in a reasonable amount of time. This approach is still not practically because of the not sufficient quality of the result. This is the approach of CASAM, it targets towards the minimizing user effort and bridging the gap between both solutions by using machine knowledge combined with human input see Figure 3.2. CASAM uses different modules to interfere as much information as possible and ask the user for specific knowledge to improve the result and lower the error rate. In the description of the CASAM project in [Giorgos Kinoktimon,] the interaction loop of casam is described in six steps and are visualized in Figure 3.1:

1. KDMA analyzes the multimedia content (video, image, natural language) and extracts the low level information. This information consists of basic image characteristics like key objects, presence and number of people, context identification etc, as well as speech recognition and simple concepts derived from text processing.
2. At the same time, the user enters a small number of keywords that describe the high-level concepts present in the multimedia content. Note that the keywords need not belong to a predefined set.
3. RMI augments the KDMA-derived and user-provided information, instantiating appropriately an ontology. Moreover, RMI infers new concept instances and reassesses the context and previous input from KDMA. Results are then fed back to the KDMA for

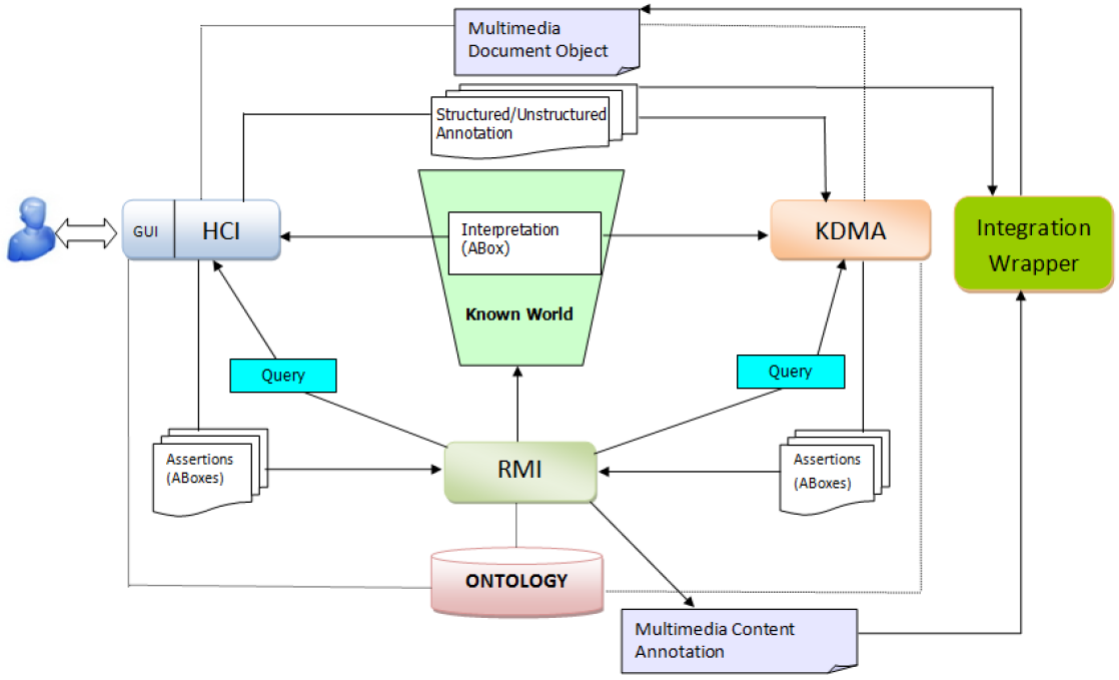


Figure 3.1: *Information flow in the CASAM system*

multimedia analysis driven by the renewed information. This RMI-KDMA internal information exchange loop continues until nothing more can be inferred by RMI or recognized by KDMA.

4. RMI reasons about what information is needed in order to add missing instances to the ontology or resolve any ambiguities that have arisen in the previous step. If the annotation target has been achieved the loop exits, otherwise the information requirements are fed to HCI.

5. HCI transforms the information requirements to input requests towards the user. HCI optimizes the user interaction using an effort-cost model, possibly by augmenting requirements to a single input request, using user-modelling to adapt to user information input patterns, etc. Furthermore, the user interface provides the user with the opportunity to alter the knowledge acquisition path devised by the system.

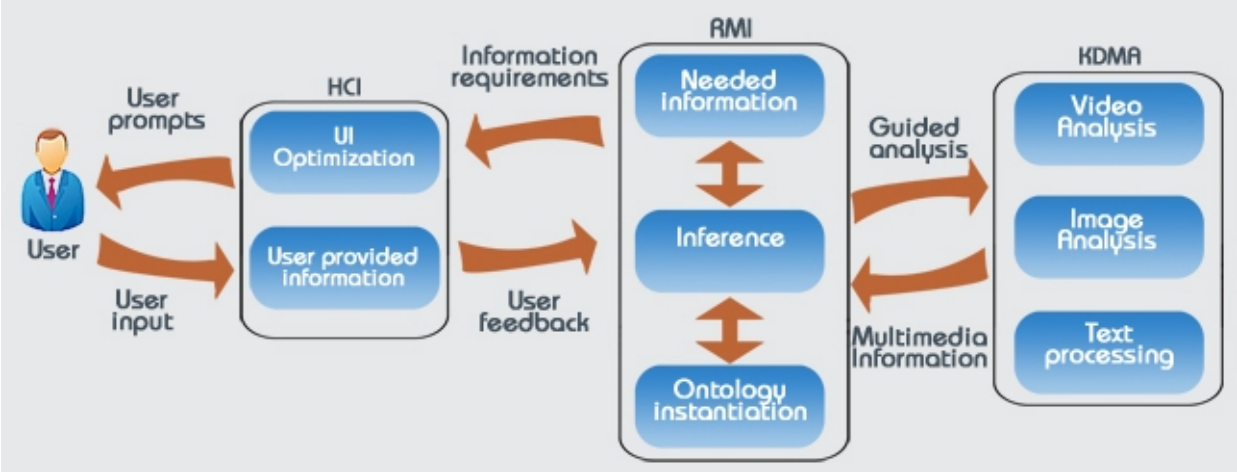
6. HCI input is passed to the RMI and the loop continues from step 3.

The modules work on-the-fly communicate with each other to share newly found knowledge. This enables the system to present results to the user as soon as possible in real-time.

The CASAM project is collaborative realized by a consortium of the following institutes and organisations: Intrasoft International, National Centre of Scientific Research "Demokritos", University of Birmingham, Technische Universität Hamburg-Harburg, Athens Technology Center SA, Deutsche Welle, Agência de Notícias de Portugal and European Journalism Centre.It is funded by the European union.

3.2 Modules

The Software is divided into several modules. These are shortly described in the following to give an overview of the system. An exact description is given in [] [D4.2] and [Giorgos Kinoktimon,]. The relevant module for this thesis is the RMI module, that is responsible for the high level reasoning of the data.



3.2.1 KDMA

The Knowledge-Driven Multimedia Analysis (KDMA) Module is responsible for the low level analysis of the multimedia content. It is subdivided into three submodules:

- Knowledge-driven Audio Analysis (KDAA)
- Knowledge-driven Video Analysis (KDVA)
- Knowledge-driven Text Analysis (KDTA)
- Knowledge-driven Fusion Analysis (KDFA)

The Knowledge-Driven Video Analysis detects shots boundaries, applies a global classification and uses object detection in the video data. The Knowledge-Driven Audio Analysis describes the audio content with respect to quality and quantity, analyses the audio content by means of audio keywords and audio scene characterizations and creates a non speech audio transcription of the audiovisual material provided by LUSA and DW. The Video is analyzed by a Video-OCR (VOCR) engine for text detection and tracking synthetic text. The Knowledge-Driven Text Analysis examines the coverage of the initial ontology and uses statistical and unsupervised learning approaches for analyzing textual content. The Knowledge-driven Fusion Analysis fuses information extracted from KDAA, KDVA and KDTA modules and provides further complementary information that could not be obtained by any of the other modules alone.

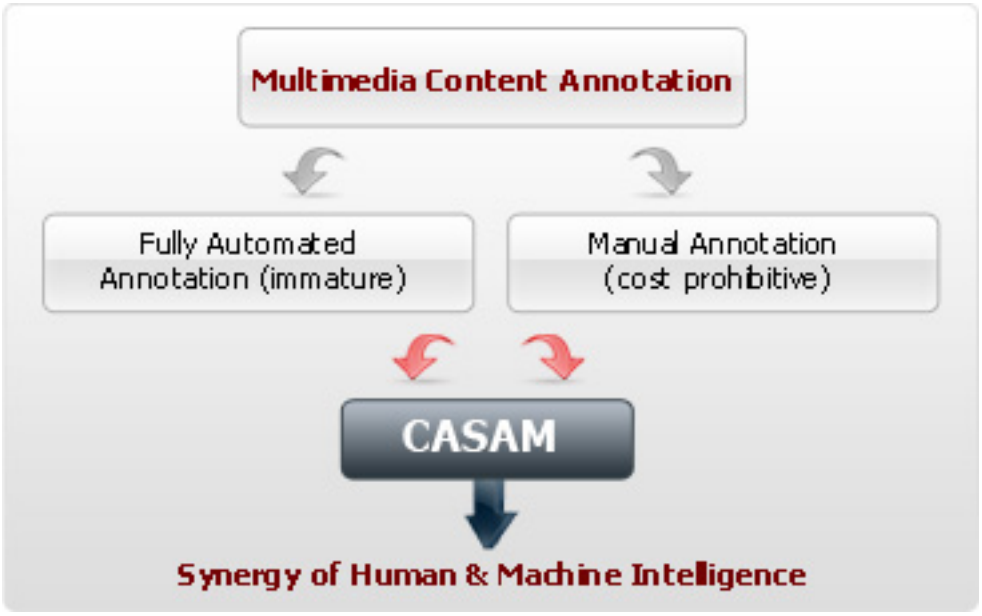


Figure 3.2: *CASAM combines the manual and automatic annotation process*

3.2.2 HCI

The Human-Computer Interface (HCI) described in [Giorgos Kinoktimon,] is responsible for the interaction between the CASAM system and the end-user. The research in this project focuses on an co-operative annotation of multimedia assets. This means CASAM will provide annotations by automatic analysis of the multimedia data. The end-user is also able to insert their own annotations independently of the systems. But the aim of CASAM is to combine this to a cooperation between end-user and machine and translate their internal representation in description logic into human readable form. The HCI is responsible for the visual interface of this interaction, for a Screen shot of the Adobe Flash based front-end running in a Browser see Figure 3.3. The software is split into this part that runs into the browser and provides a highly responsive user interface and a back-end that is responsible for the heavy processing task and runs on more capable server machine.

3.2.3 RMI

The Reasoning for Multimedia Interpretation (RMI) module is responsible for the high-level interpretation of the low-level analysis that is created by KDMA and HCI. This information is send back to the components to display them for the user and to refine its analysis processes.

3.2.3.1 Reasoning-based Media Interpretation Engine

The RMI engine is the central element in the RMI module. An overview of the engine is shown in figure 3.4.

The engine is explained in [Oliver Gries et al., a] as follows:

Figure 3.3: *The front-end of the CASAM system*

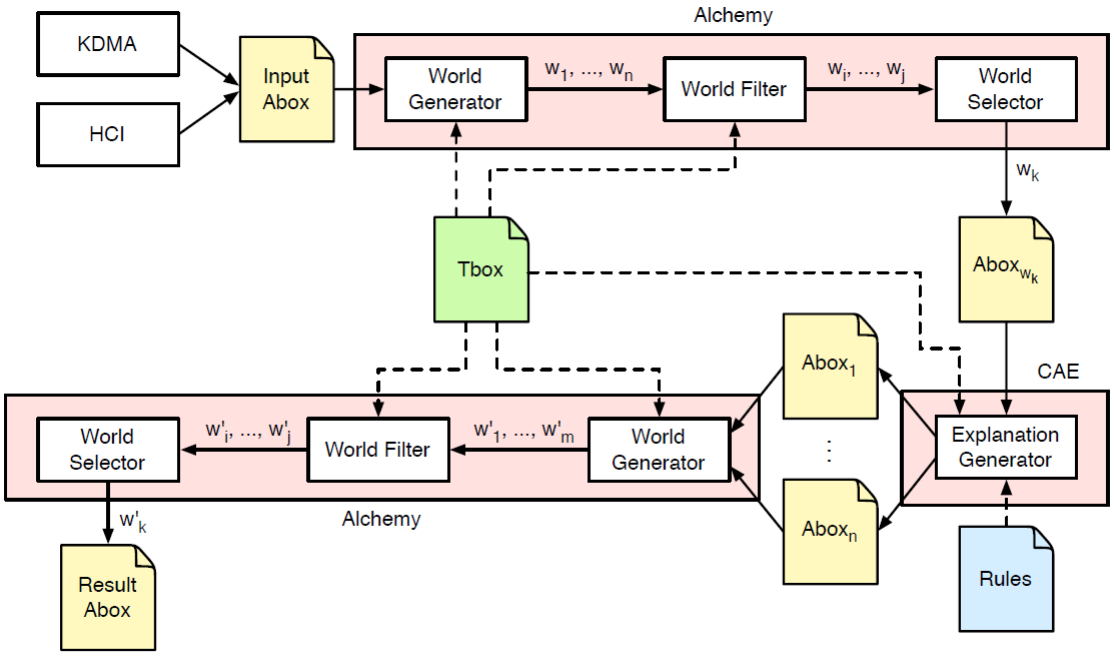


Figure 3.4: *Overview of the Reasoning-based Media Interpretation Engine*

- World Generator: Based on the Tbox and the preprocessed input Abox this component produces all Markov logic worlds. Which are indicated in Figure [] by $w_1, ...w_n$. A world is a vector of ground atoms. Assume there are m ground atoms. Consequently, the number of worlds is 2^m. The generated worlds are the input ot the next component called World Filter.

- World Filter: This component removes the impossible worlds. In this step, the subsumption axioms and domain and range restriction in the Tbox are considered for world elimination. The remaining worlds w_i,...,w_j, also known as possible worlds, are the input to the next component called World Selector.

- World Selector: This component selects the most-probable world among the set of possible worlds. The most-probable world has the highest probability based on the Markov logic score. A deeper introduction to Markov logic can be found in [Matthew Richardson and Pedro Domingos,]. The Alchemy system is used for this task and described later in this chapter. The The most-probable world w_k is transformed into an Abox (called $Abox_{w_k}$).

After this three steps the $Abox_{w_k}$ is used as input for the Explanation Generator. This part generates further evidence by using domain knowledge to generate new assertions. At this process multiple explanations are possible. Therefore, multiple output Aboxes can occur. The three step World Generator/Filter/Selector described above is used again to compute the most probable world. If there are several worlds with the same maximal probability, one of them is chosen non-deterministically.

3.2.3.2 CASAM Abduction Engine

One part of the RMI engine is called the CASAM Abduction Engine (CAE). It is used to generate so-called explanations for the assertions found in the input Aboxes. This is done to derive additional knowledge for Abox assertions. The abduction interference service of RacerPro is used to compute explanations.

The CAE algorithm is described as follows, a detailed explanation can be found in [Sofia Espinosa Peraldi et al.,] and [Oliver Gries et al., a]:

Function CAE(Ω, Ξ, Σ, $\mathcal{R}$, S, $\mathfrak{A}$):
Input: a strategy function Ω, a termination function Ξ, a knowledge base Σ, a set of rules $\mathcal{R}$, a scoring function S, and an agenda $\mathfrak{A}$
Output: a set of interpretation Aboxes $\mathfrak{I}'$
$\mathfrak{I}' := \{assign\_level(l,\mathfrak{A})\}$;
repeat
  $\mathfrak{I} := \mathfrak{I}'$;
  $(\mathcal{A}, \alpha) := \Omega(\mathfrak{I})$;
  $l = l + 1$;
  $\mathfrak{I}' := (\mathfrak{A} \setminus \{\mathcal{A}\}) \cup assign\_level(l, explanation\_step(\Sigma, \mathcal{R}, S, \mathcal{A}, \alpha))$;
until $\Xi(\mathfrak{I})$ *or no* $\mathcal{A}$ *and* α *can be selected such that* $\mathfrak{I}' \neq \mathfrak{I}$;
return $\mathfrak{I}'$

where $assign\_level(l,\mathfrak{A})$ is defined as follows:

$$assign\_level(l,\mathfrak{A}) = map(\lambda(\mathcal{A}) \bullet assign\_level(l,\mathcal{A}),\mathfrak{A}) \tag{3.1}$$

3.2.3.3 Example of the algorithm

Figure 3.5 is giving an example of the CAE algorithm

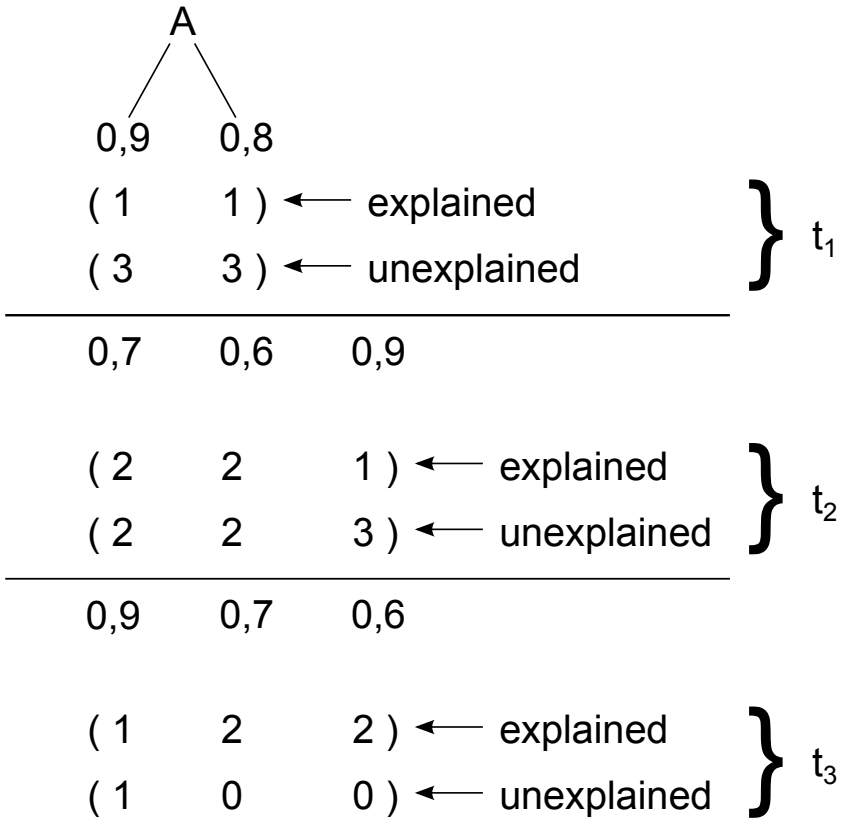


Figure 3.5: *example of the RMI algorithm*

The algorithm works in several rounds (t_{1-3}). The Assertions an the Agenda are sorted by their scoring values. One bin contains the explained fiats and another bin the unexplained. In each round on unexplained fiat from the bin with the highest score is explained. To calculate the new score the average is used.

3.2.3.4 Alchemy

The Alchemy package provides a series of algorithms for statistical relational learning and probabilistic logic inference, based on the Markov logic representation. It is developed at the Dept. of Computer Science and Engineering at the University of Washington. A deeper description of the system can be found in [Kok, 2007] and in the user manual [Stanley Kok et al., 2010].

The RMI engine uses the Inference algorithm of the Alchemy software.

Alchemy supports two basic types of inference: probabilistic and MAP/MPE. The probabilistic

inference algorithms are Lifted Belief Propagation (-bp) described in [P. Singla and P. Domingos, 2008], MC-SAT (-ms) [H. Poon and P. Domingos,], Gibbs sampling (-p) and simulated tempering (-simtp) [E. Marinari and G. Parisi, 1992]. The option -maxSteps is used to specify the maximum number of steps in the algorithm.

3.2.3.5 Parameters of the RMI engine

The main configuration file of the RMI engine is the racer.init file in the main directory. It defines the parameters of the RMI engine. The Parameter **agenda-max-no-of-interpretations** specifies the length of the Agenda (agendaLength). The Parameter **number-of-explanations-considered** defines the maximum number of rules applied. The Parameter **interpretation-loop-max-iterations** defines the rounds that are executed to explain fiats. The Parameter **start-next-bunch-with-single-interpretation** is set to t to avoid resetting the score to one after each processed batch and is set to nil during the testing process in this thesis.

Chapter 4

RMI Test Framework

4.1 Introduction

The described RMI engine in the last Chapter

The framework is written in the Python language. The advantage of this language is the availability of the interpreter for the most common systems available today. This allows the script to be run on several different systems from desktop systems to high-performance servers with multiple cores. For the numerical processing the numpy library [26, 2011] is used. The output is plotted with the matplotlib [27, 2011].

4.2 Overview of the System

The dataflow in the CASAM system is emulated by the AssertionSender. Figure 4.1 gives an overview of the data flow of the different applications running on an server with the IP 134.28.70.136.

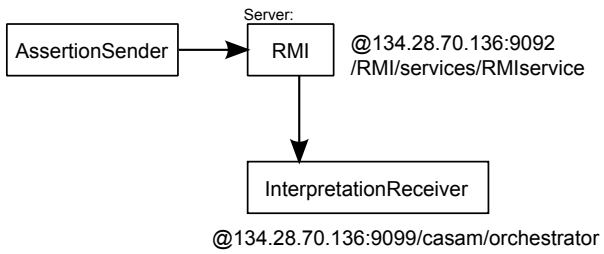


Figure 4.1: *Dataflow between the AssertionSender, RMI and the InterpretationReceiver.*

4.3 Subsystems

The Framework is split into three subsystems, that are described in the following. Figure 4.2 shows the Class digram of the systems the subsystems are marked in green, orange and blue.

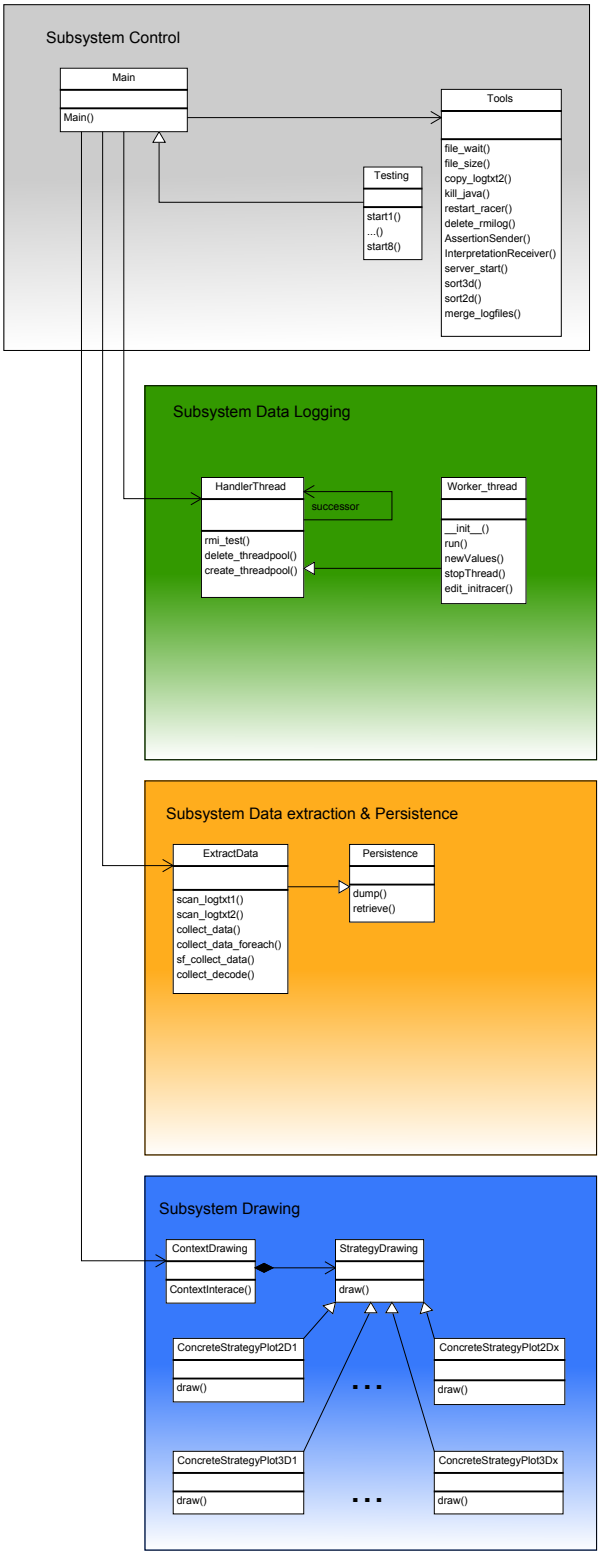


Figure 4.2: *UML Diagram of the RMI Test Framework*

4.3.1 Data logging Subsystem

The logging subsystem is responsible for starting the applications and for handling log files. The system starts the RMI Server, the AssertionSender Java application and the InterpretationReceiver Java application. The AssertionSender emulates a real data source, so the data is send in real time. This can be very time consuming in practice. Thus the Test Framework can generate the test data in parallel by using a thread pool and maximize the data flow. Figure 4.3 is showing the activity diagram of each thread.

4.3.2 Data extraction Subsystem

The data extraction subsystem parses the log files for the relevant data by using regular expressions. To speed up the time consuming parsing process, a state machine is used to only parse relevant data and exclude impossible matches. It is also possible to redirect the extracted data to a binary data format for faster processing in the next step.

4.3.3 Drawing subsystem

The drawing subsystem transforms the chosen data values into a graphical representation. Several representations that best fit the data can be selected by command line parameters.

4.4 Testing Workflow

The test framework is completely controlled by command line options, an overview of the available commands is given in the Appendix. The test procedure can be split into three tasks, they basically correspond to each subsystem.

An example of this procedure is given in the following:

Parse log files and output to 7.rmi output file:

```
python testscript.py --scan log7.txt 7.rmi
```

Plot number 6 with 6.rmi input binary file. Show the result and retrieve x=numberOfInterpretations, y=rounds, z=ignoredBunches take only the variable numberOfExplanationsConsidered with the value 20 and the variable agendaLength with the value 30. Label the axis with x=numberOfInterpretations, y=rounds, z=ignoredBunches:

```
python testscript.py --plot 6 6.rmi --show --what "numberOfInterpretations;roun
```

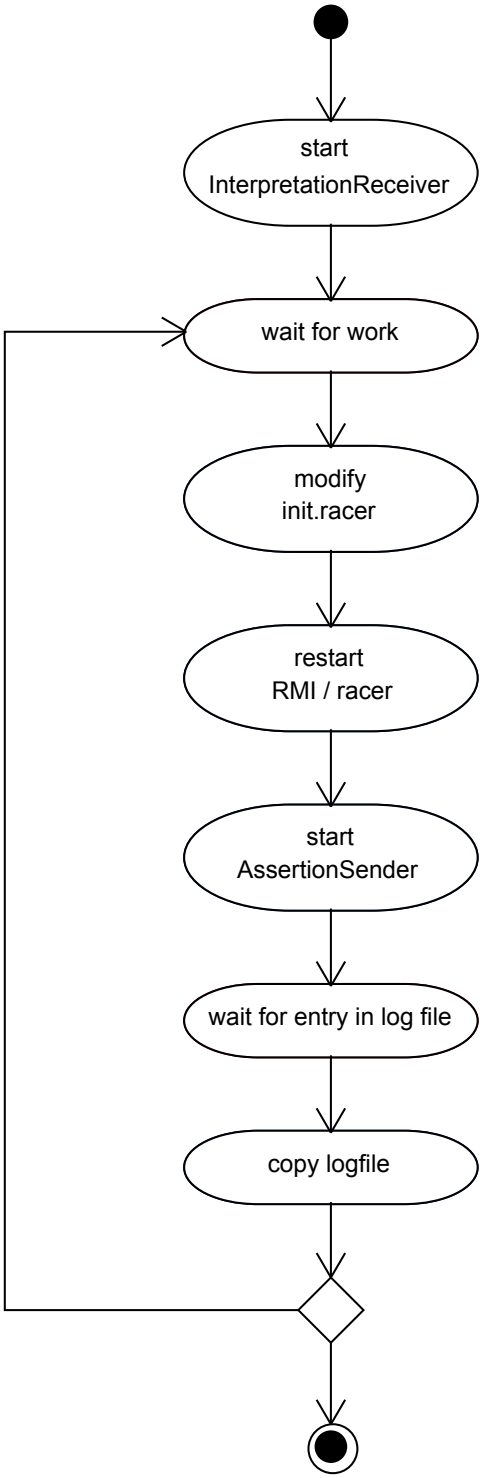


Figure 4.3: *Activity diagram of the data logging thread*

Chapter 5

Discussion of the test results

5.1 Introduction

The Testing Framework has been used to create data for all relevant parameters. These are discussed in this chapter and a proposal for optimal parameters is given.

5.2 Hypothesis one: Increasing the maxSteps sampling size increases the processing time

Figures 5.1 - 5.4 are showing the test results. The scatter plot shows the maxSteps in the abscissa and the real time in the ordinate. Real time means the accumulated time needed to process all bunches. The red line shows a linear best line fit for all measured points. In the first plot in figure 5.1 the MC-SAT algorithm is used. In the second figure 5.2 Gibbs sampling is used and in the third graph in figure 5.3 belief propagation is used. The last graph in figure 5.4 uses lifted belief propagation It can be seen that with all the algorithms the processing time is indeed increasing. Ordered by the slope, the Gibbs sampling shows the best performance, followed by MC-SAT algorithm, the worst performance is shown by belief propagation.

5.3 Hypothesis two: Increasing the maxSteps samples size increases the final score

Figures 5.5 to 5.8 are showing the test results. The graph shows the points of higher concentration with a different color. The maxSteps is plotted in the abscissa and the final score in the ordinate. The red line shows a linear best line fit for all measured points. It can be seen that a higher sampling size not necessarily increases the final score. The increase of the final score not necessarily means that the score is more accurate.

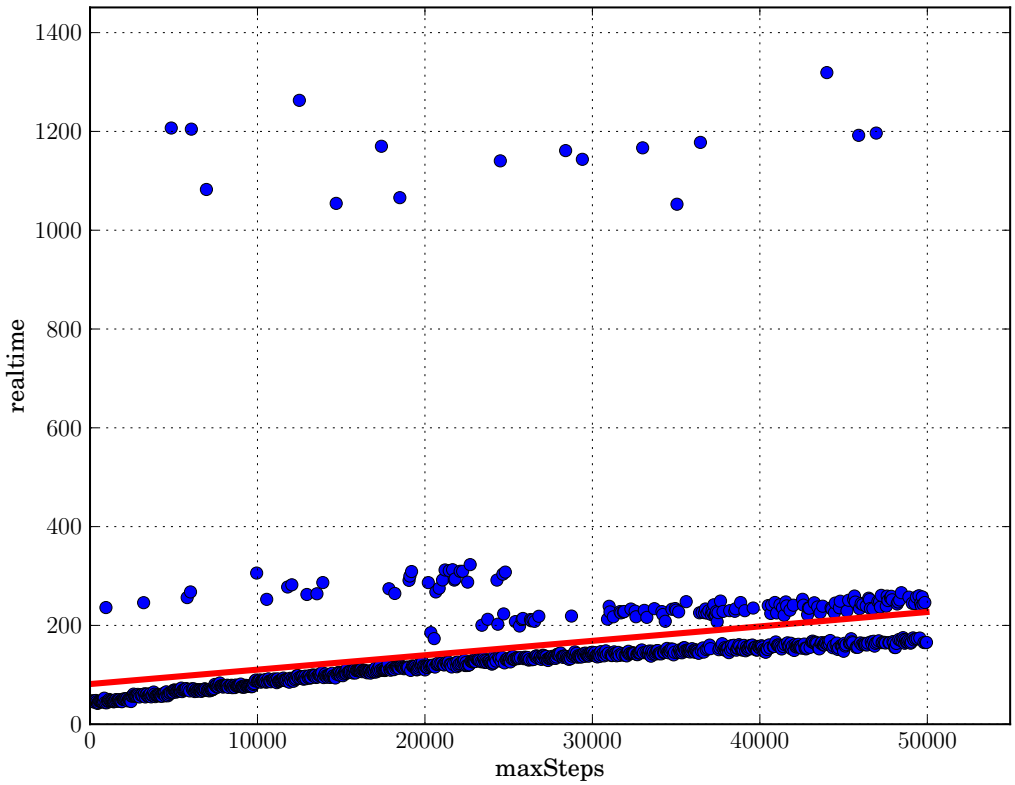


Figure 5.1: *Data set 6 with the MC-SAT algorithm and maxSteps at abscissa and real time at the ordinate.*

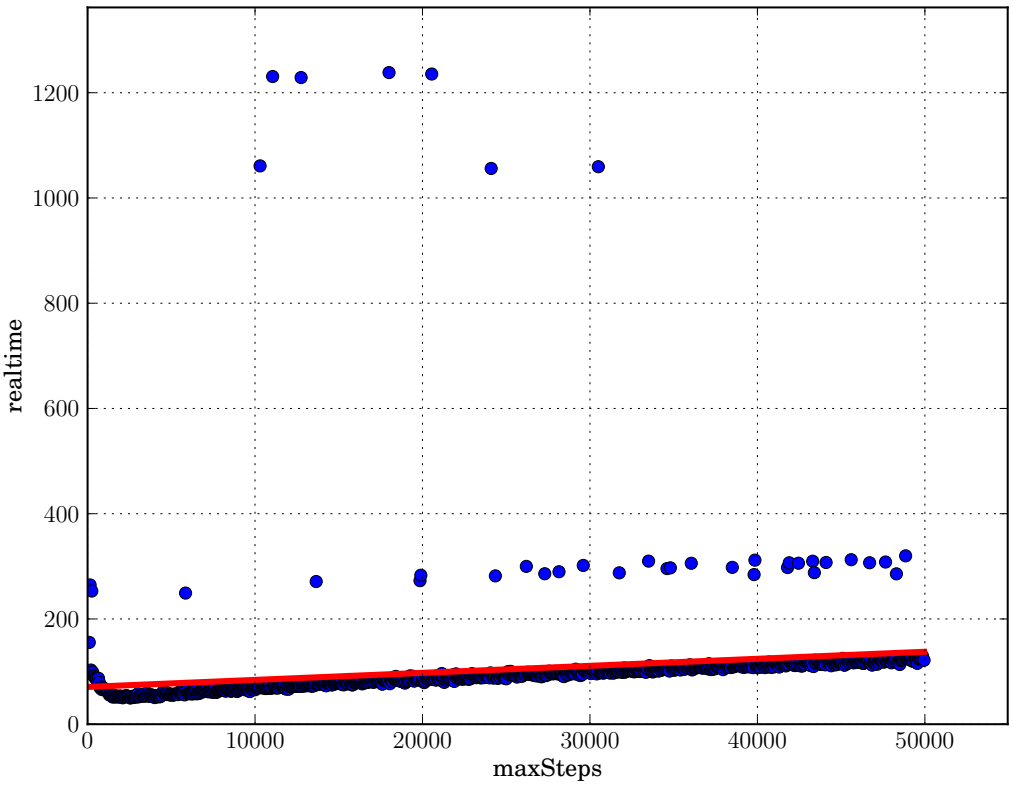


Figure 5.2: *Data set 6 with the Gibbs sampling algorithm and maxSteps at abscissa and real time at the ordinate.*

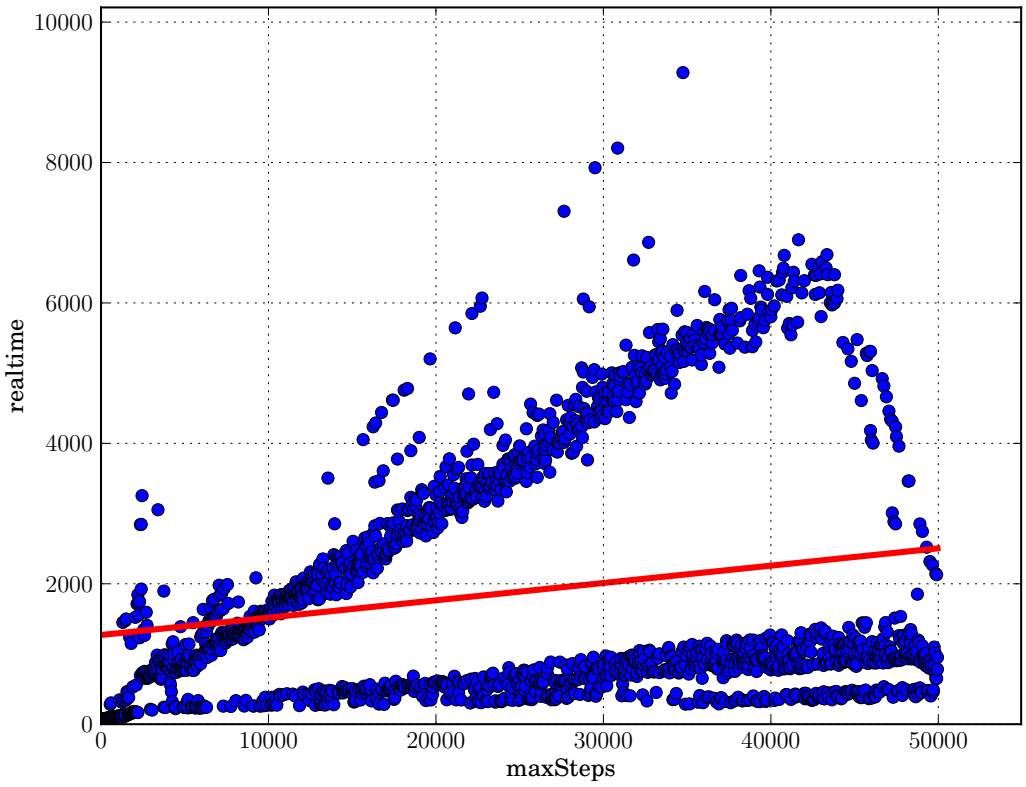


Figure 5.3: *Data set 6 with the belief propagation algorithm and maxSteps at abscissa and realtime at the ordinate.*

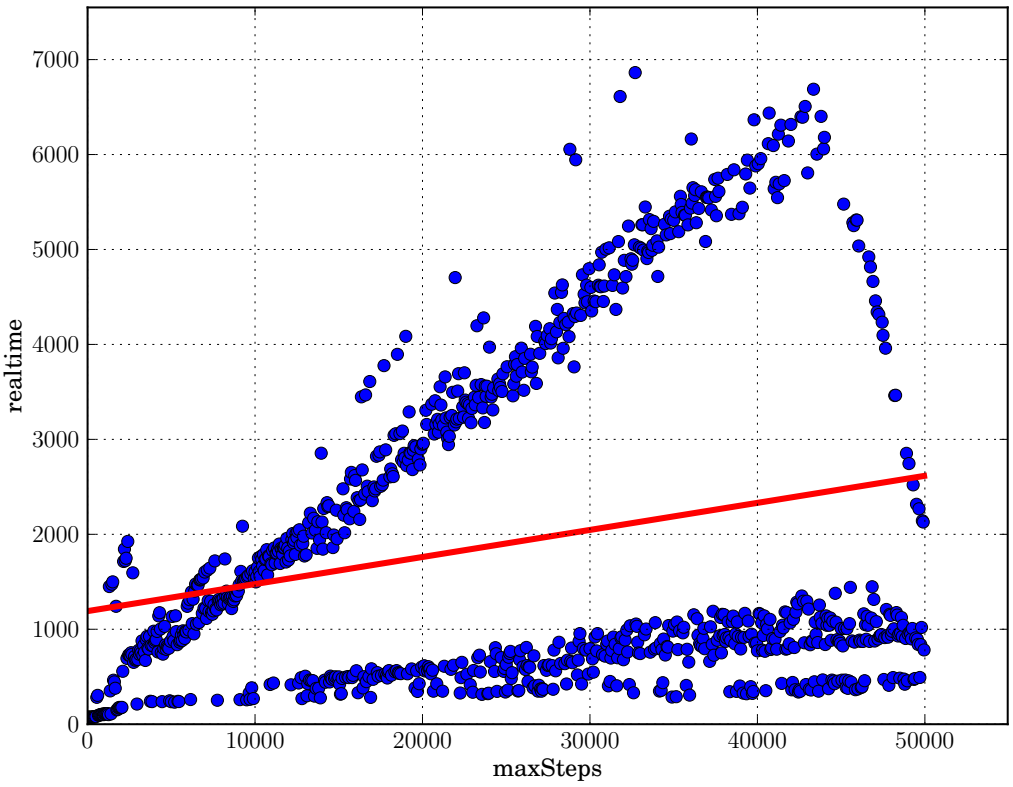


Figure 5.4: *Data set 6 with the lifted belief propagation algorithm and maxSteps at abscissa and realtime at the ordinate.*

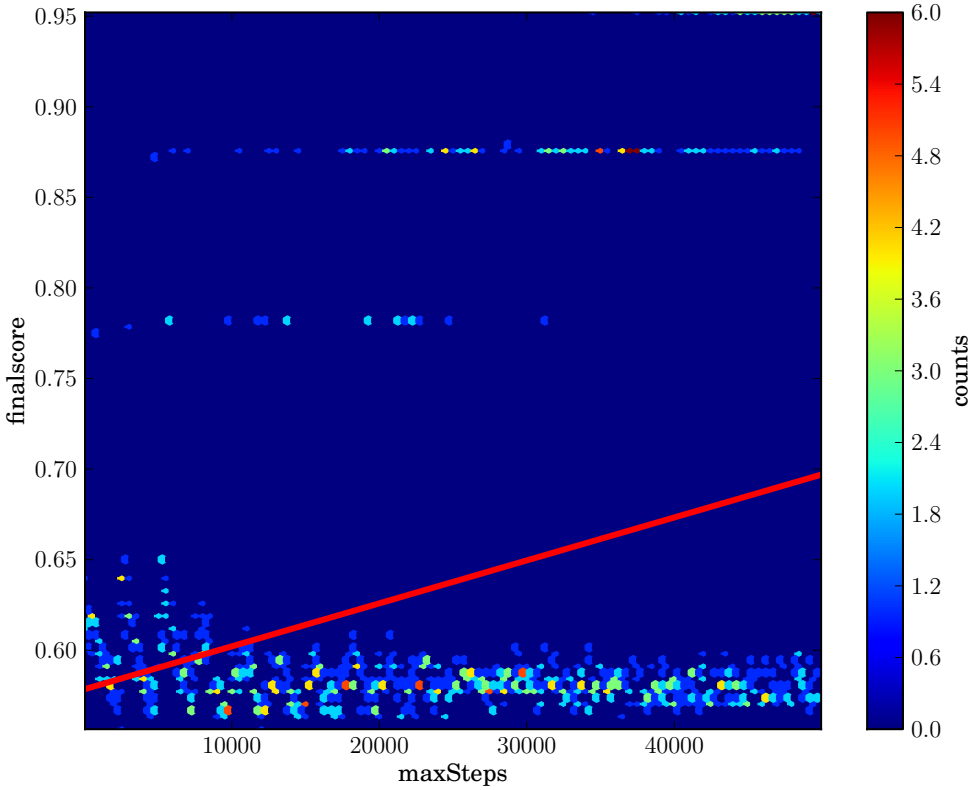


Figure 5.5: *Data set 6 with the MC-SAT algorithm and maxSteps at abscissa and the final score result at the ordinate.*

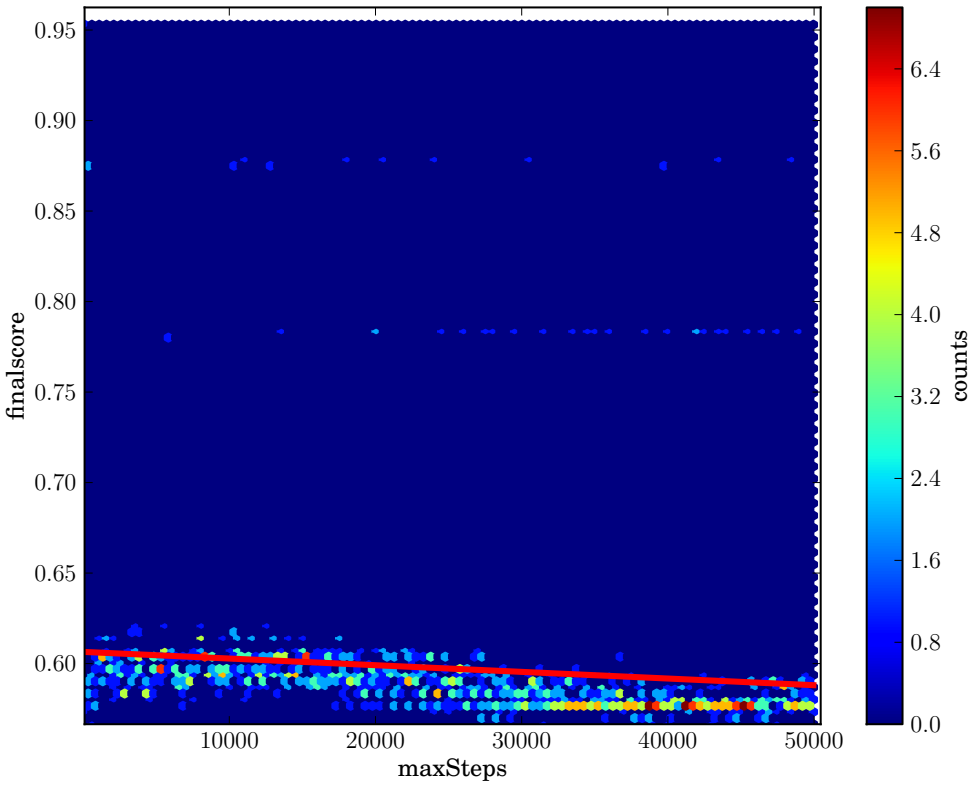


Figure 5.6: *Data set 6 with the Gibbs sampling algorithm and maxSteps at abscissa and the final score result at the ordinate.*

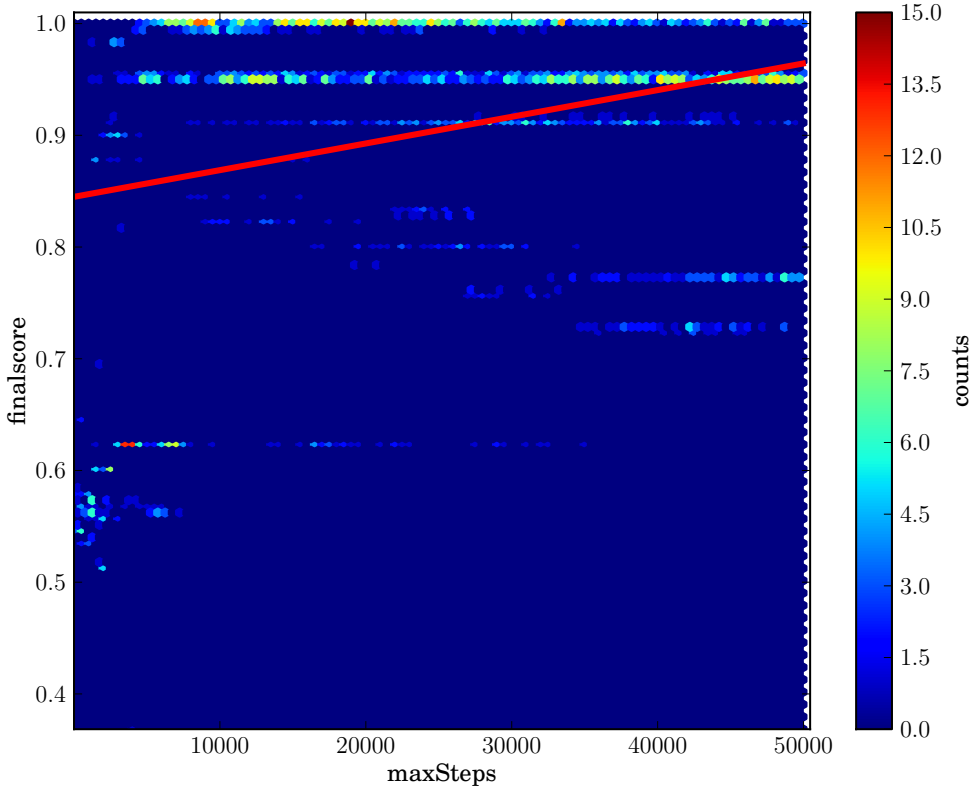


Figure 5.7: *Data set 6 with the lifted belief propagation algorithm and maxSteps at abscissa and the final score result at the ordinate.*

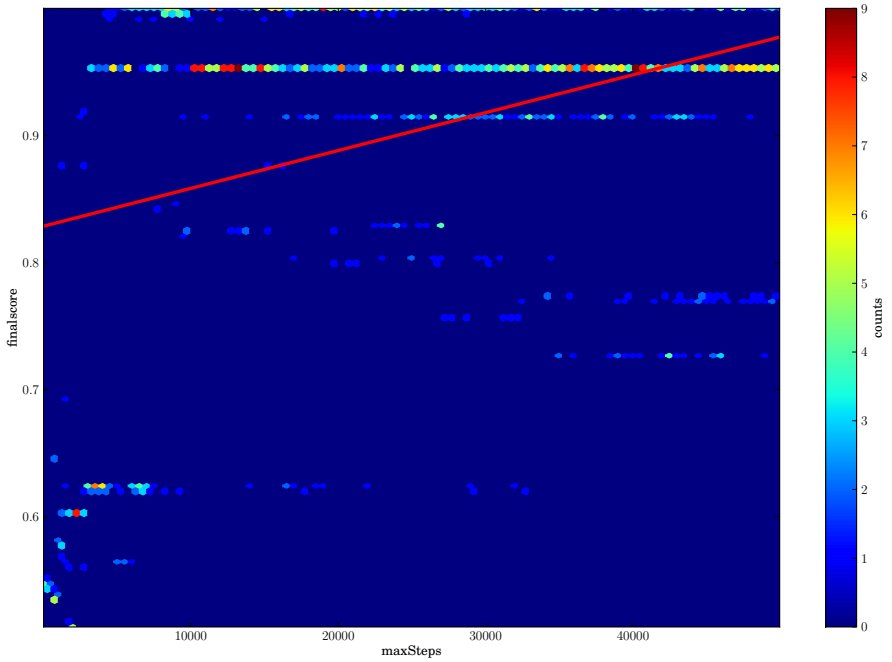


Figure 5.8: *Data set 6 with the lifted belief propagation algorithm and maxSteps at abscissa and the final score result at the ordinate.*

5.4 Hypothesis three: Too many rounds / longer agendaLength increases the ignoredBunches

Figure 5.9 shows that the agendaLength does not show an influence on the ignoredBunches. The rounds do have an influence on the ignored bunches. A value higher than about 16 rounds shows a significant increase in ignoredBunches. So a value up to 16 rounds should be chosen here.

5.5 Hypothesis four: Every new bunch decreases the score

For this hypothesis several measurements are examined. In the first figure 5.9, six runs are chosen. The colored lines mark the processing step when a new bunch arrives. The graph shows the decrease of the score for every bunch that is arriving.

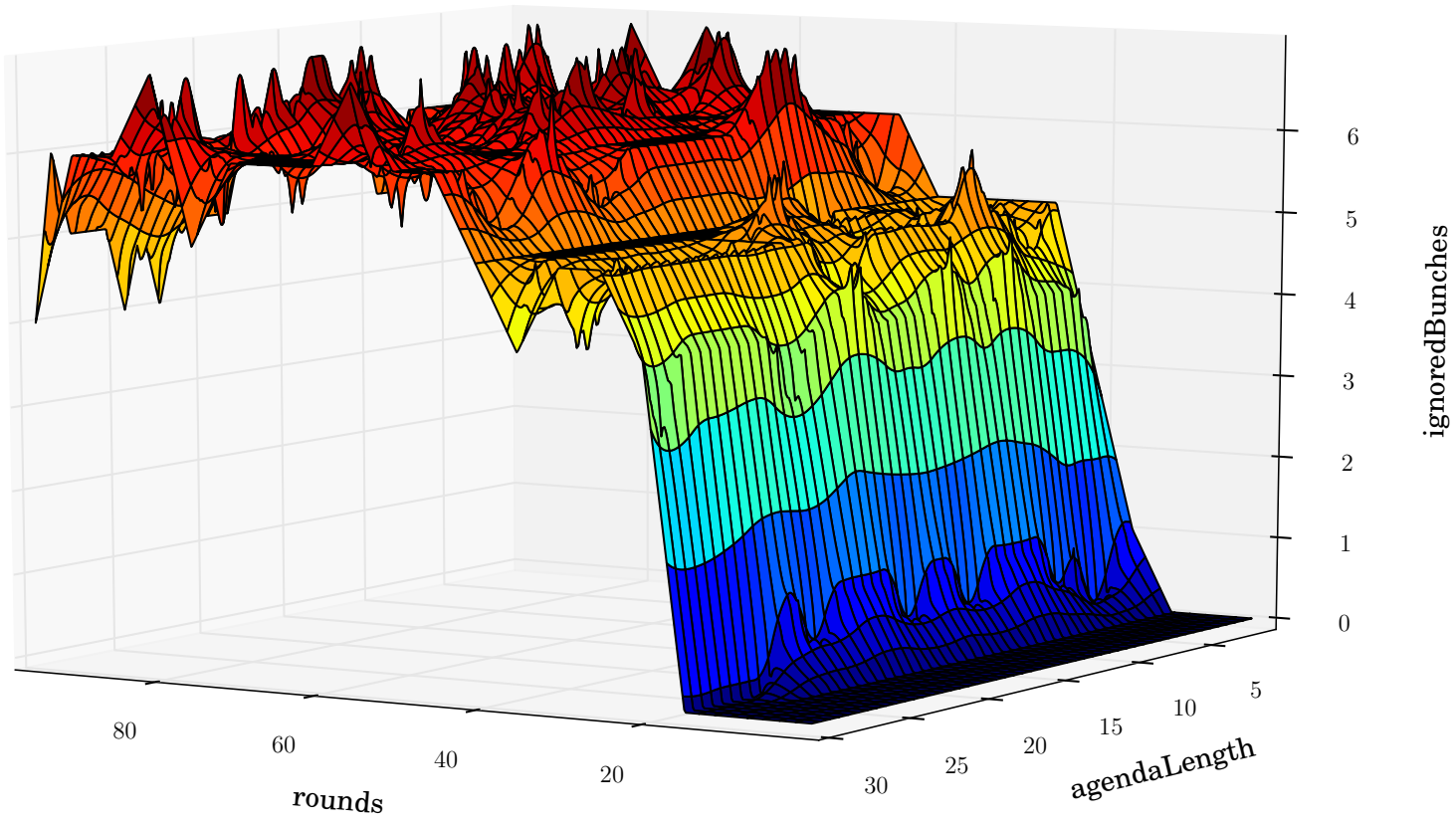


Figure 5.9: *Data set 6 rounds and agendaLength / ignoredBunches*

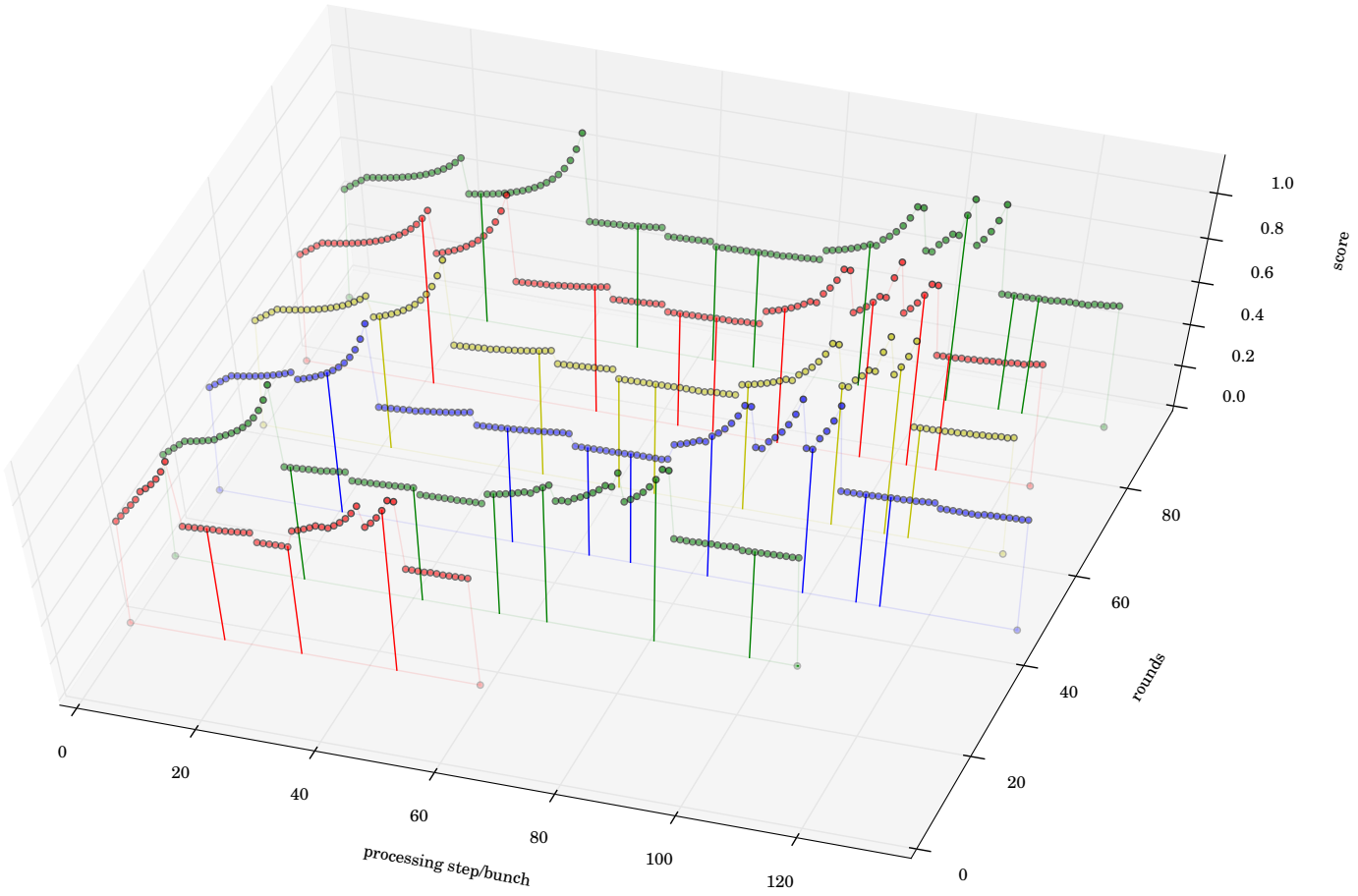


Figure 5.10: *Data set 6 with rounds / processing steps / score. The colored lines mark new bunches*

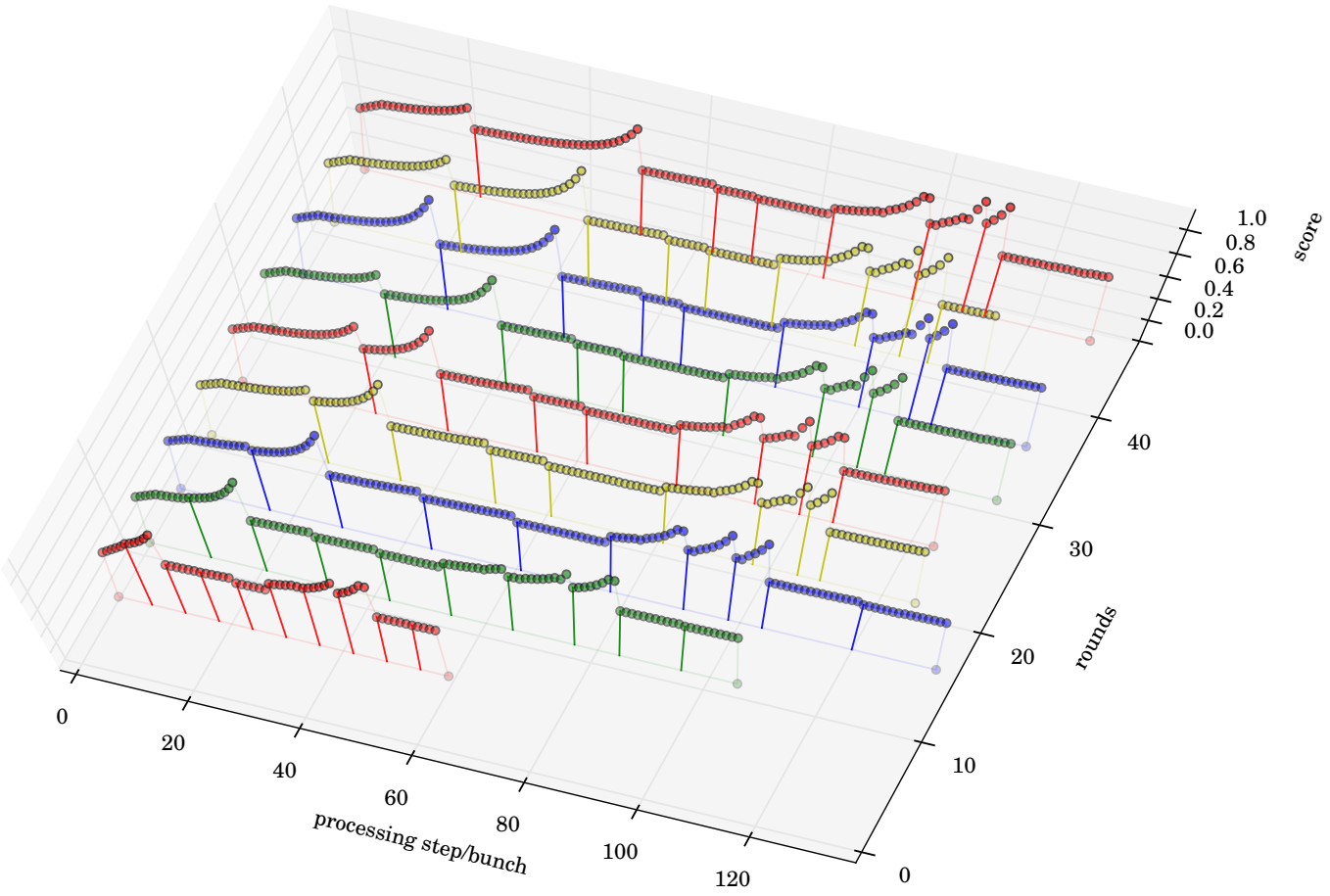


Figure 5.11: *The same data set as in 5.10 before but zoomed into rounds 0-40*

5.6 Hypothesis five: Increasing the numberOfExplanations leads to more ignoredBunches

Hypothesis three already stated that the agendaLength not significantly influences the ignoredBunches. Does the numberOfExplanationsConsidered influence the ignoredBunches? The Heatmap in figure ?? shows that bunches get ignored starting from 10 Explanations Considered. So numberOfExplanations does have an influence on ignoredBunches.

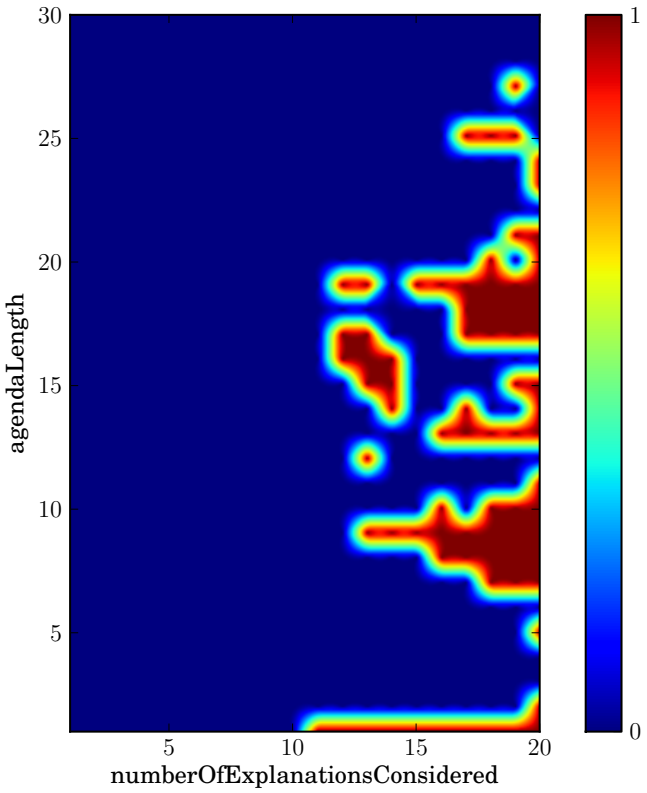


Figure 5.12: *Heatmap agendaLength/numberOfExplanationsConsidered rounds=16*

5.7 Hypothesis six: Increasing the rounds leads to a higher final score

The graph in hypothesis three already showed that the rounds increase the ignoredBunches. So does it also influence the final score? The heatmap in figure ?? shows that bunches get a higher final score when the rounds increase. It seems to converge to its peak at about six rounds. Taking into consideration the result from the graph in hypothesis three, a value between 6 and 16 should be optimal. The agendaLength does not seem to have an influence on the final score.

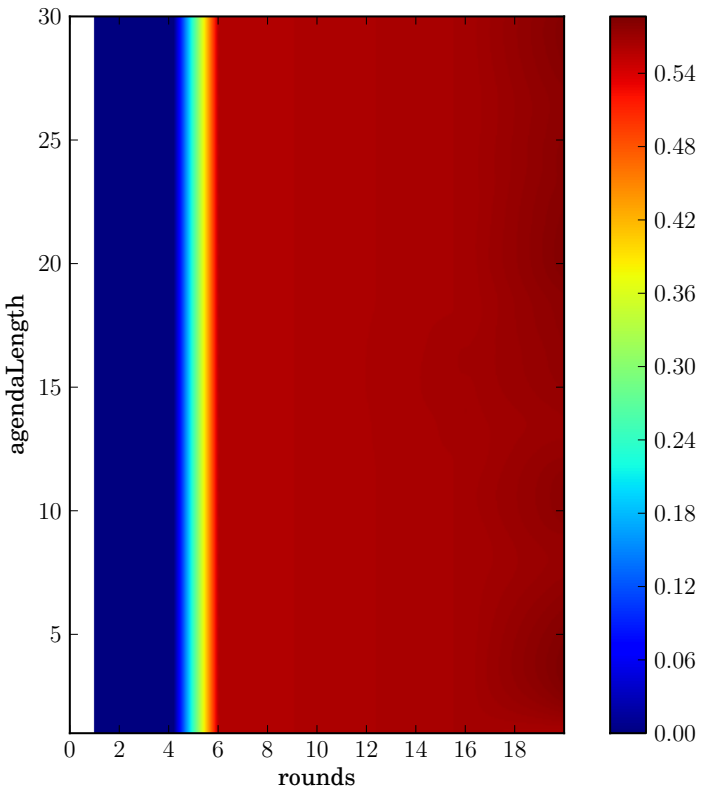


Figure 5.13: *Heatmap rounds/agendaLength/finalScore numberOfExplanationsConsidered=20*

Chapter 6

Conclusion and Outlook

This thesis evaluated the CASAM RMI engine. A description of the CASAM project has been given in chapter 3. In chapter 4 a modular test framework has been developed that can be easily adopted to new parameters. Several tests have been run on the RMI engine. The tests performed showed that the RMI engine can be still optimized by setting parameters of the engine. The documentation of the tests and proposals are given in chapter 5.

Further research is needed to test the parameters with different data. This might also influence the parameters. All the tests in this thesis have been done with the dataset 6.

It is possible to extend the Test Framework. Currently the Test Framework uses threads to work in parallel. It should be evaluated if this impacts the test results and if a switch to a SMP approach would be appropriate.

Appendix A

Test Framework Command Options

```
Usage: testscript.py [options]

Options:
  -h, --help            show this help message and exit

  log options:
    creates log files for the given parameters

    -l ARG_LOG, --log=ARG_LOG
                        create test logs --log [logfilename]
    -t ARG_THREADS, --threads=ARG_THREADS
                        size of threading pool used to create test data
                        --threads [numberthreads]
    -a ARG_AGENDALENGTH, --agendaLength=ARG_AGENDALENGTH
                        --agendaLength [min] [max] [step]
    -r ARG_ROUNDS, --rounds=ARG_ROUNDS
                        --rounds [min] [max] [step]
    -n ARG_NUMBEROFEXPLANATIONSCONSIDERED,
                        --numberOfExplanationsConsidered [min] [max] [step]
    -m ARG_MAXSTEPS, --maxSteps=ARG_MAXSTEPS
                        --maxSteps [min] [max] [step]
    -M ARG_SAMPLINGMETHODS, --samplingMethods=ARG_SAMPLINGMETHODS
                        --samplingMethods seperated by ; example: "-ms;-bp"
                        ...
    -i ARG_ASID, --id=ARG_ASID
                        --id [id]

  Plot options:
    creates plots from log files or binary dump file

    -p ARG_PLOT, --plot=ARG_PLOT
                        plot output --plot [1-3] [databasefilename ("" if no
                        database)] [outputfilename]
    -w ARG_WHAT, --what=ARG_WHAT
```

```
                        name of values to be plotted seperated by ; example:
                        --what "rounds;agendaLength"
  -d ARG_HOLD, --hold=ARG_HOLD
                        restrict variables to a certain value seperated by ;
                        example: --hold "rounds;agendaLength" "30;16"
  -o, --show            Show plot default: False
  -b ARG_LABELS, --labels=ARG_LABELS
                        labels seperated by ; example: --labels
                        "rounds;agendaLength" ...
  -e ARG_TITLE, --title=ARG_TITLE
                        Plot title --title [title]

Scan options:
  parse log files and dump to binary database file

  -s ARG_SCAN, --scan=ARG_SCAN
                        scan test logs and dump to binary file --scan
                        [logfile] [binaryfile]
```

Bibliography

[26, 2011] (August 27, 2011). Numpy reference.

[27, 2011] (June 03, 2011). matplotlib: user's guide.

[Baader and W. Nutt, 2007] Baader, F. and W. Nutt (2007). *The description logic handbook: Theory, implementation, and applications.* Cambridge University Press, Cambridge and , New York, 2nd edition.

[E. Marinari and G. Parisi, 1992] E. Marinari and G. Parisi (1992). Simulated tempering: A new monte carlo scheme.

[Giorgos Kinoktimon,] Giorgos Kinoktimon. Casam: a prototype system for computer-aided semantic annotation of multimedia.

[H. Poon and P. Domingos,] H. Poon and P. Domingos, editors. *Sound and efficient inference with probabilistic and deterministic dependencies: In Proceedings of the Twenty-First National Conference on Artificial Intelligence.* AAAI Press.

[Kok, 2007] Kok, S. (2007). The alchemy system for statistical relational ai.

[Matthew Richardson and Pedro Domingos,] Matthew Richardson and Pedro Domingos. Markov logic networks.

[Oliver Gries et al., a] Oliver Gries, Ralf Möller, Anahita Nafissi, Maurice Rosenfeld, Kamil Sokolski, and Michael Wessel. Dealing efficently with ontology-enhanced linked data for multimedia.

[Oliver Gries et al., b] Oliver Gries, Ralf Möller, Anahita Nafissi, Maurice Rosenfeld, Kamil Sokolski, and Michael Wessel. Media interpretation and companion feedback for multimedia annotation.

[P. Singla and P. Domingos, 2008] P. Singla and P. Domingos, editors (2008). *Lifted first-order belief propagation: In Proceedings of the Twenty-Third National Conference on Artificial Intelligence.* AAAI Press.

[Sofia Espinosa Peraldi et al.,] Sofia Espinosa Peraldi, Atila Kaya, and Ralf Möller. Formalizing multimedia interpretation based on abduction over description logic aboxes.

[Stanley Kok et al., 2010] Stanley Kok, Parag Singla, Matthew Richardson, Pedro Domingos, Marc Sumner, Hoifung Poon, Daniel Lowd, Jue Wang, and Aniruddh Nath (2010). The alchemy system for statistical relational ai: User manual.