F_018

Constructing and Using a Web Service to Find Theorems

何 成† 鈴木秀男‡ 小林英恒†
He Cheng Suzuki Hideo Kobayashi Hidetsune

1. Introduction

Automated reasoning is an important field in artificial intelligence. Until now, many kinds of special systems have been developed to use computers to do the work. Among them, Isabelle[1] is a well-known interactive prover for theorem proving.

Theorems finding is very important in theorem proving because all theorem provers including Isabelle need proved theorems to prove new ones. There are often thousands of them pre-loaded by one prover for powerful proof. However, at the same time, it is hard for users to find or memorize them for application while useing the provers. Therefore, it is necessary to build a special system to help theorem finding. Two main problems have to be solved in realizing the system. One is how to get the theorem data, and the other one is how to design a special search interface for use.

This paper proposes a system providing a web service to find theorems based on having solved the two problems. The design are presented in [2]. The key idea of the system is using web browsers to search in a theorem database, which is constructed by extracting theorems from the Isabelle prover and using standard techniques of database and web server.

The structure of the paper is arranged as follows: section 2 gives an overview of the system. Section 3 describes its implementation in detail. Section 4 introduces the ways of finding theorems, and an evaluation of the system is given in Sect.5. Section 6 summarizes the paper.

2. Overview

Figure 1 shows the system's architecture. It mainly consists of the two components: an extraction interface and a search interface. As to the software used, besides Isabelle, the system utilizes other software such as PostgreSQL database system, PHP language package and Apache web server. Note that all of them are free for use.

The work flow of the system is: at first, theorems are extracted and saved as records by the extraction Interface. Then these records are inserted into a database constructed by the PostgreSQL system. Finally, using the search Interface, the Apache server, and a network,

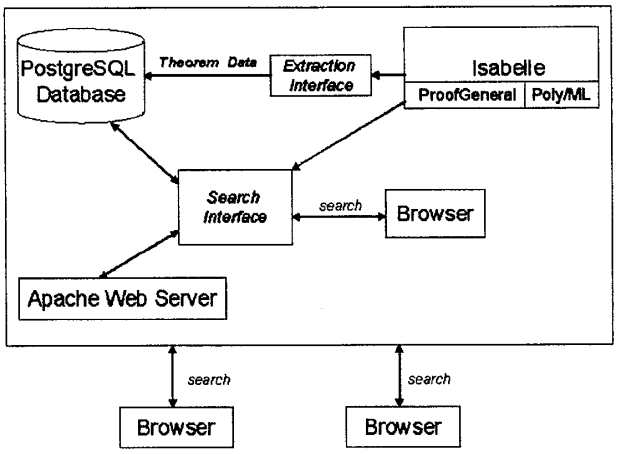


Fig. 1: Overview of the architecture

on-line users can search the theorems anywhere by web browser.

The service system mainly has the following features:

- easy theorem extraction
- flexible theorem search
- search by web browser

3. Implementation Details

The implementation of the system could be divided into three parts: data extraction, data processing and search interface construction.

3.1 Data Extraction

Data extraction is the key part of building up the whole system. Obviously, inputting them one by one is not possible because there are too many theorems for different theories or logics. Fortunately, Isabelle has been integrated with a plentiful database of theorems. However, the database is invisible and not open since it is kept as special heap files in the prover soft. The only way to access the database is using Isabelle's implementation language, Meta Language (ML[3]).

The extraction interface is made up of ML functions to do the extracting work. Finally, the extracting operation is done by a function named **thms_of**, which is shown in Fig.2. The function takes a logic name as an argument, and returns the logic's theorems during

† Nihon University
‡ Tokyo Institute, Polytechnic University

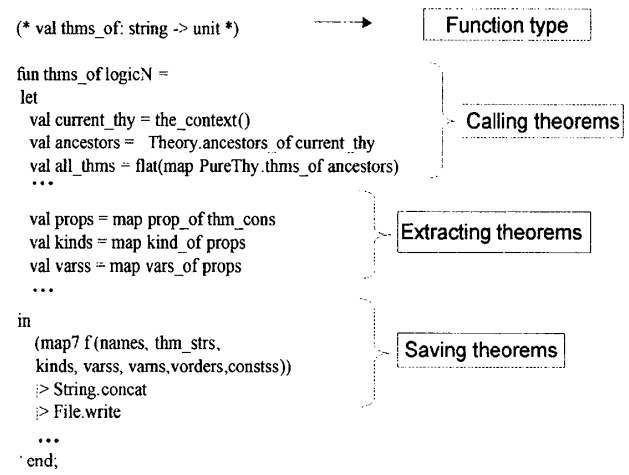


Fig. 2: Main ML function for extraction

the process. By this function, one logic's all theorems can be obtained and stored into records in just a few seconds.

The records for theorems (or lemmas*) are arranged in a format as shown in Fig.3. It can be found from the format that the record contains seven kinds of information about theorem: theorem name (name), theorem content (content), kind, variables, number of variables (number), variable order (order) and constants. Note that variables in these records have been enclosed by the question mark "?" for declaring their positions; this is achieved by a modification of Isabelle's ML sources.

In a word, this part takes out original data i.e. theorems, and the data offers a good basis for theorem matching later.

3.2 Data Processing

This part does the work of managing the data like inserting, index creating. It is easier than the two other parts because the PostgreSQL system has enough functions to do this job. The theorem records can be quickly inserted into the database by some simple SQL commands as follows.

At first, create a table named **thm_table** for theorems using the following scripts:

```
drop table thm_table;
create table thm_table (
  thmN text,
  thmCon text,
  kind text,
  vars text,
  var_number int,
  var_order text,
  consts text,
  length text
);
```

*In Isabelle, there is no essential difference between theorem and lemma.

And then, using "copy" command to copy the records from a file named 'thms' into the database:

```
\copy thm_table (thmN, thmCon, kind, vars,
var_number, var_order, consts) FROM 'thms'
```

Lastly, two indexes for theorem name and variable number are created for the table in the database in order to accelerate searching.

```
DROP INDEX thmN_index;
CREATE INDEX thmN_index ON thm_table (thmN);

DROP INDEX varnum_index;
CREATE INDEX varnum_index ON
       thm_table (var_number);
```

3.3 Search Interface Construction

The task of this part is providing users a friendly user interface to search. In fact, all the search processes can be preformed using commands of an interactive interface provided by the database system, but it is not convenient as it needs too much input. The search interface is still necessary because it can deal with search queries, and communicating with the database, and displaying results. The interface is written in the PHP language[4] and its codes can be separated into the following parts:

- script functions: functions for query processing;
- input area: four search boxes for entering queries;
- SQL query generation: produce SQL queries from the queries;
- communication: communicate with the database;
- printing: display results if they exist

Figure 4 shows the look of the search interface. The interface leaves four items for searching. They are name, number, order, and content. How to use them is explained in the next section.

4. Finding Theorems

This section mainly discusses the methods used in finding theorems according theorem properities or contents. Theorems can be searched by five kinds of ways: search by name, number, order, content, or by their combination†:

- search by name: the name also contains theory name information that shows where a theorem comes from. Therefore, the users can type any keywords relating to the name (some type information like "Real" or "Int" are often included in the name) into the name search box for search.

†Also noted as compound search or advanced search

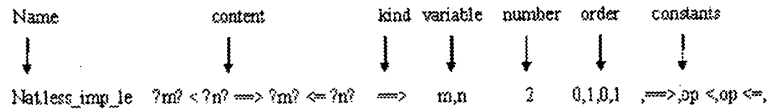


Fig. 3: Theorem record example

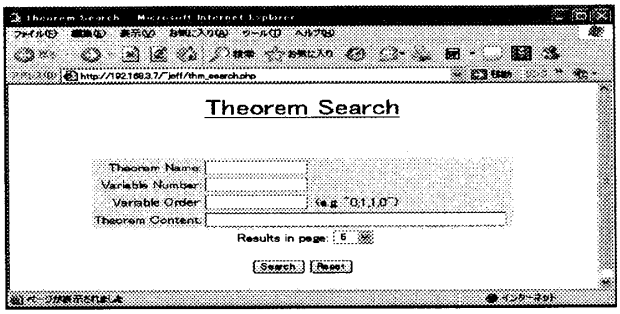


Fig. 4: The search interface

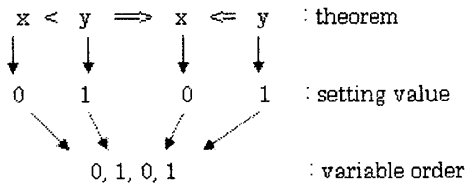


Fig. 5: Example of computing the variable order of one theorem

- search by number: if users don't know any keywords about name but the number, they can try this way. Since many theorems may have the same number of variables especially the lower numbers such as 2 or 3, results might become too many. Therefore, the search is often combined with other items to search.
- search by order: the order is defined as the order of variables in a theorem. Its calculating way presented in Fig.5 shows it can be got easily. The order can be taken as a kind of pattern for theorem matching. Experiments show that when the orders often return good results after the matching.
- search by content: the content is the body of one theorem. This way is more complicated than the ways above because it usually considers several things such as the number, the order, or constants. Moreover, keywords can also be typed into the content search box for getting more suitable results.
- compound search: this is a method to be mostly used because it combines all the previous ways in a time. If knowing any theorem information of name, the number, the order, users can enter them all for one search.

5. Evaluation

Isabelle's theorem finding function has a chief limitation in theory. That means the users need to load one theory in advance before they can search the theorems in the theory because Isabelle can not load multiple theories concurrently. In contrast, all the search methods provided by the system have no this limitation.

To evaluate the system, the system is tested on theorem extraction and theorem finding.

The following is a display of successfully extracting all theorems from Higher Order Logic (HOL[5]) in Isabelle's interactive interface. It tells that in total 6309 theorem are taken out and saved in a file named "/tmp/HOL_thms". The whole process costs about 2 seconds.

```
> thms_of "HOL";

+++ 6309 theorems of HOL to "/tmp/HOL_thms"
> val it = () : unit
```

For doing a theorem finding experiment, totally 18910 theorems were obtained from Isabelle's all logic heap files like this way. The theorem distribution[6] is listed in Tab.1. Ten kinds of logics exist in the prover. It can be found from the table that the Higher Order Logic of Computable Function (HOLCF) logic has the most theorems and the Pure logic only owes 23.

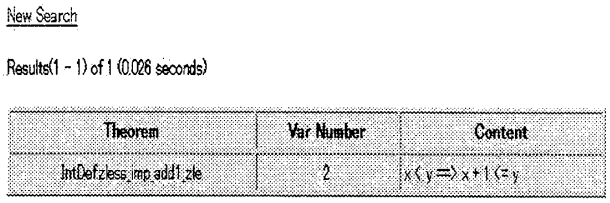
Using the methods introduced in Sect.4, the experiment was performed to see if the system could find theorems as expected. The executing efficiency is also an important item we concerned. The results are given in Tab.2. Note that **thm-ex** is a theorem-like text: "$?x? <?y? ==>?x? + 1 <=?y?$", and the compound search is shown in Fig.7.

Through the experiment and the results, we can observe that:

- theorems can be extracted from Isabelle very quickly by the functions of the extraction interface. The interface saves much manual work of typing theorems. It solved the first problem of obtaining data.
- theorems can be found by various ways by using the simple search interface. As shown in Fig.6,

Tab. 1: Theorem distribution

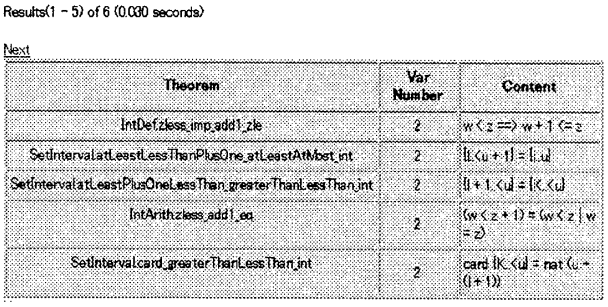
Logic	Pure	HOL	FOL	CCL	CTT	FOLP	HOLCF	LCF	Sequents	ZF	Total
Number	23	6309	241	586	154	88	7200	320	82	3907	18910


New Search

Results(1 - 1) of 1 (0.026 seconds)

Theorem	Var Number	Content
IntDef.zless_imp_add1_zle	2	x < y ==> x + 1 <= y

Fig. 6: Search for theorem "$x < y ==> x + 1 <= y$" by **thm-ex** as content


Results(1 - 5) of 6 (0.030 seconds)

Next

Theorem	Var Number	Content
IntDef.zless_imp_add1_zle	2	w < z ==> w + 1 <= z
SetInterval.atLeastLessThanPlusOne_atLeastAtMost_int	2	{l..<u + 1} = {l..u}
SetInterval.atLeastPlusOneLessThan_greaterThanLessThan_int	2	{l + 1..<u} = {l<..<u}
IntArith.zless_add1_eq	2	(w < z + 1) = (w < z | w = z)
SetInterval.card_greaterThanLessThan_int	2	card {l<..<u} = nat (u - (l + 1))

Fig. 7: Search for theorem "$x < y ==> x + 1 <= y$" by name "int less", number 2 and constant 1 as content

searching by **thm-ex** containing variable names finally gets only one result exactly matching the query. And Fig.7 shows the same theorem can also be returned by the simpler and shorter queries though more results are sent back together. Compared to entering **thm-ex**, the information like name "Int less" and number 1 is easy to be thought and entered.

- the search speed is satisfactory as all the searches can be finished in 1/5 seconds. Search by the number 3 is relatively slower than others because it got the results in the biggest number as 2742, and search by theorem content or compound method is fast enough for use.

6. Conclusions

This paper describes how to construct a web search system and use the web service to search theorems efficiently. The system can be considered as a general system for theorem finding because the service could be utilized on internet by web browsers and it bases on Isabelle's large database of proved theorems.

For future work, we plan to connect theorem proving with the system directly and develop distributed parallel theorem proving based on the system.

Tab. 2: Experiment results. The search times are obtained from a 2.4GHz Intel PC with 640MB of RAM.

Method	Query	Theorems	Time(s)
name	"HOL"	279	0.021
number	3	2742	0.162
order	0,1,0,1	1354	0.002
content	**thm-ex**	1	0.026
compound	"int less"; 2; 1	6	0.030

Reference

[1] L. C. Paulson. Isabelle – A Generic Theorem Prover. Springer Verlag.

[2] He Cheng, H. Suzuki, K. Hidetsune, A Web Search System For Theorems, Proc. of the Thirteenth Workshop on Automated Reasoning,pp.15-16 (2006)

[3] L. C. Paulson. ML for the Working Programmer. CAMBRIDGE.

[4] 三木秀治. 初めての人のための PHP Web データベースプログラミング. MYCOM.

[5] Tobias Nipkow, L. C. Paulson, M. Wenzel, A Proof Assistant for Higher-Order Logic, http://www4.in.tum.de/ nipkow/LNCS2283/ tutorial.pdf

[6] The Isablle library, http://isabelle.in.tum.de/dist/library