

## 50x Faster: Speeding up an SQL-based legacy system with few changes

Lei Shi, Lenneke Jong, Wolfgang Mueller, Enkhjargal Algaa

Lei.Shi@h-its.org, Lenneke.jong@h-its.org, Wolfgang.Mueller@h-its.org, Enkhjargal.Algaa@h-its.org

### Abstract:

SABIO-RK <http://sabio.h-its.org/> is a database for reaction kinetic data. Its complex schema led to slow querying. The challenge described in this paper is how to improve the speed with as few as possible changes in the time tested SABIO-RK code. The dilemma of whether to rewrite a whole system or to best reuse the expertise embodied in existing code appears often in existing, grown academic bio software solutions. We describe our solution that uses the existing SQL database for consistent data storage, and uses inverted indexes for fast retrieval. We further outline how we migrated to a new interface and RESTful web services in order to improve SABIO-RK code reuse and standards compliance.

The work presented in this paper has been partly funded by the DFG project "Integrierte webbasierte Forschungsinfrastruktur zur Speicherung und Bereitstellung quantitativer zeitaufgelöster Daten im Laborkontext und fuer die Wissenschaftsfentlichkeit".

## 1 Introduction

SABIO-RK (System for the Analysis of Biochemical Pathways - Reaction Kinetics) is a database offering curated information about biochemical reactions and their kinetic properties [RGK<sup>+</sup>07]. Its web interface and web services are built upon a SQL-based search engine which makes its data accessible to the user. Our users use these data in mathematical models that deal with the analysis and prediction of the dynamic behavior of biological networks.

With a code base which has been evolving for over 10 years, SABIO-RK has started suffering from efficiency problems in its SQL-based search mechanism. Obtaining and displaying of structured objects requires too many on-line access to the underlying database. In SABIO-RK, the main object displayed is an *entry* which is (for the purposes of this discussion) a table containing multiply interconnected tables in some of its cells. While there are some gains in speed may be found, such as by materialized views, many strategies for increasing the efficiency of data retrieval would require tables with a variable numbers of columns, which is not available an SQL database such as PostgreSQL.

SABIO entries can be queried by a huge number of different attributes (properties) and many of the possible queries will yield no results. To improve the user experience it is preferable to show the expected number of results for a given query *before* running it.

This operation, however, is almost as costly as running a full query, as we need to *count* the results for the upcoming query. While daily use of SABIO-RK was possible, creating these number previews requested by our users was completely out of reach of the existing system. Thus, our challenge was to speed up the search and fulfill this additional user requirement with as few as possible changes rather than undertaking a rewrite the whole system.

In this paper we explain the SABIO-RK efficiency problems in detail in Section.2, and describe our Pump-data-into-the-index-offline solution in Section.3. We then present the new web interface and web services which are built upon the new system. Section.5 shows the results of a simple use case scenario, comparing the search efficiency between the new and old systems. Section.6 discusses some of our future work.

## 2 SQL-based Search Over Complex Relational Schema

As with all databases dealing with complex biochemical objects such as proteins, genes and pathways, SABIO-RK suffers from the inadequacy of relational data models to model these complex objects. The resulting complex data schema is often difficult to query. In addition, changing the high coupling SQL-based engine and associated interfaces and web services to adapt to those frequently evolved data models are always costly.

### 2.1 High "Cost" SQL Query

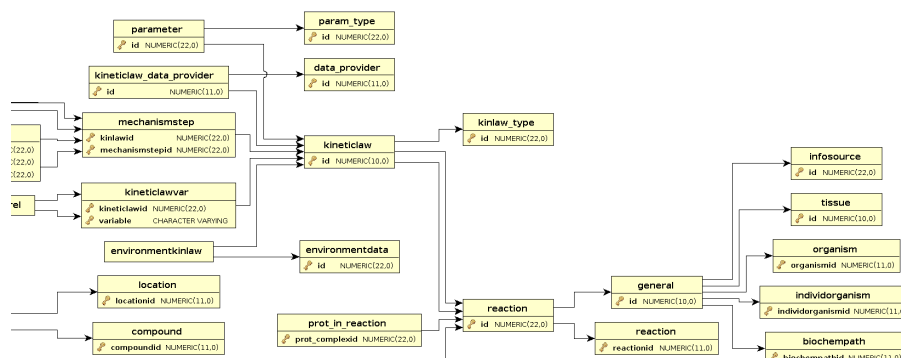


Figure 1: Partial Schema

Shown in Figure 1 is a partial schema of one of SABIO-RK databases. Consider a user who wishes to search for and sort reactions, and their correlated kinetic law entries, which have enzymes belonging to the enzyme class 2.7.1.40. This very complex query, which joins

multiple SABIO-RK tables and sorts the result respectively, are sent by the SQL-based search engine. 603 sorted <ReactionID, EntryID>pairs are retrieved, and the detailed results are returned to the user through the old SABIO-RK interface using more than 3 min. Since the query response time increases linearly with the number of returned data entries, as the amount of data stored in SABIO-RK increases, this problem will become even worse.

## **2.2 High Cost to Adapt to the Evolve of Data Models**

Within SABIO-RK, the data access interaction is based on a SQL Maps framework, IBATIS, which couples objects with stored procedures or SQL statements using a XML descriptor or annotations [BGM07]. The framework facilitates the use of a relational database with object-oriented applications, but the high coupling system structure makes the adaption of upper level components to changes in the persistence level database hard.

In order to cooperate with many different projects in system biology area, databases such as SABIO-RK face the issue of needing to either expand to include new models or to update the multiple relations among tables. This evolution allows SABIO-RK to maintain its vitality and relevance, but also leads to a higher costs for re-development.

## **3 Pump-data-into-the-index-offline Solution**

Based on the problems we stated in Section.2, and our challenge constraints stated in Section.1, we are inspired by a related work in [SW05] and propose our pump-data-into-the-index-offline solution to improve and upgrade the existing SABIO-RK system, through the decoupling of the data access.

The new system traverses the database content and builds inverted index for all data entries relative to searchable meta data offline. During system runtime, all user queries go directly to stored inverted files rather than databases. Such light-weighted solution not only improves the query efficiency, but also reduces the later potential cost for extending and adapting the system.

### **3.1 Inverted Indexing**

Though it is hard to construct relational data models which entirely describe those biochemical complex objects, the data model to describe those representative information is simple.

Take SABIO-RK for example, it is hard to construct complete relational data models for the entire biochemical pathways - reaction kinetics objects, but the most important row properties (meta data) of kinetic law data form a very simple data model represented as

table in Table. 3.1.

Kinetic Law Entry ID	Reaction	Enzyme	Organism	Tissue	...	Parameter	...
123	...	2.7.1.40	...	..	..	Km	..
...	...	...	...	...	...	...	..

Table 1: Simple Data Model

Refer to definitions of the fundamental concepts in the search engine library, Lucene, [GH04]

- An index contains a sequence of documents.
- A document is a sequence of fields.
- A field is a named sequence of terms (string values).

, the example meta-information stored in our document object is listed as following

```
<EntryID: 24734><ECNumber:2; 2.7; 2.7.1; 2.7.1.40>
<Reaction: Pyruvate + ATP + Phosphoenolpyruvate + ADP =
Phosphoenolpyruvate + ADP + Pyruvate + ATP><SABIOID:9>
<Organism: Strptococcus mutans><Pathway: Carbon fixation>...
```

and the index object is conceptually represented in Figure.2.

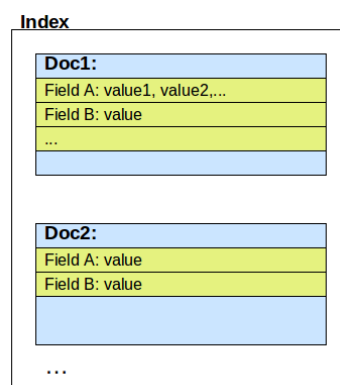


Figure 2: Conceptual Index Object

In order to achieve more efficient term-based query, the well-known inverted index structure is applied to build a list of references to documents for each named term (field), as shown in Figure 3.

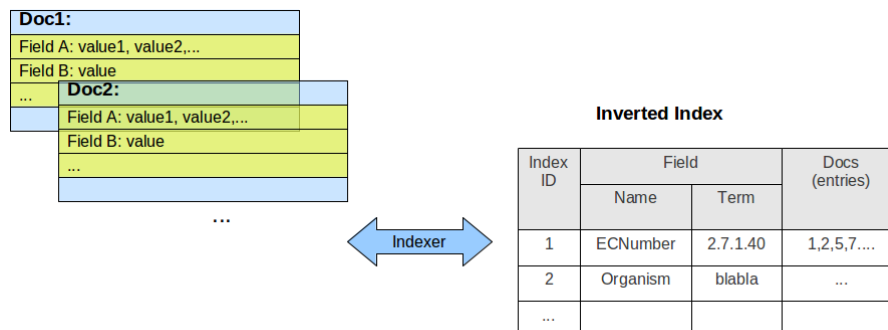


Figure 3: Inverted Index

## 3.2 Indexing Component

The indexing procedure stated in Section 3.1 is executed offline and comprises of four main components (please see Figure 4) which realize the procedure, described below:

**Data Extraction Engine** This engine is used to traverse the entire database offline and for each searchable attribute (field), such as Reactant, ECNumber, Pathway, and etc., extracts and stores data triples like <EntryID, Field, Value > from multiple tables.

**Document Loading Engine** Each triple is treated as a virtual document and added via a DocWriter.

**Indexer** Indexed documents are sent to the Indexing handler, which builds an inverted index over those documents.

**Vocabulary Builder** For each field, all possible values are sent to an N-gram analyzer to create dictionary later used for an auto-complete function during query.

### 3.2.1 New System Overview

After integrating our solution to existing SABIO-RK system, the system overview is changed and represented in Figure 5. An example query executed over the new system is displayed in Figure 6.

## 3.3 Integration With ACLs

SABIO-RK uses the Spring Security framework to customize authentication and access-control process using Access Control Lists (ACLs) [Mul10]. In the old SABIO-RK system, before result entries return to the user, the system looks up the access control list for

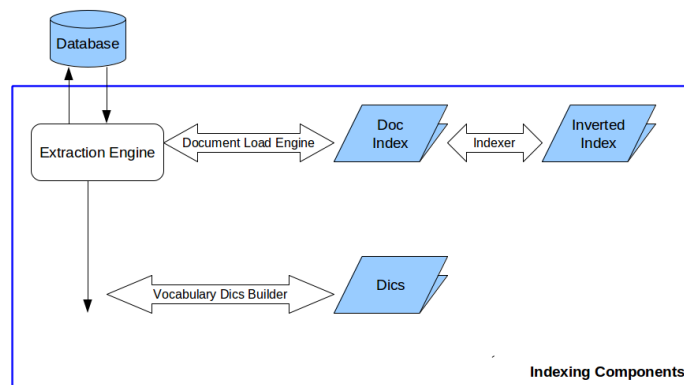


Figure 4: Indexing Component

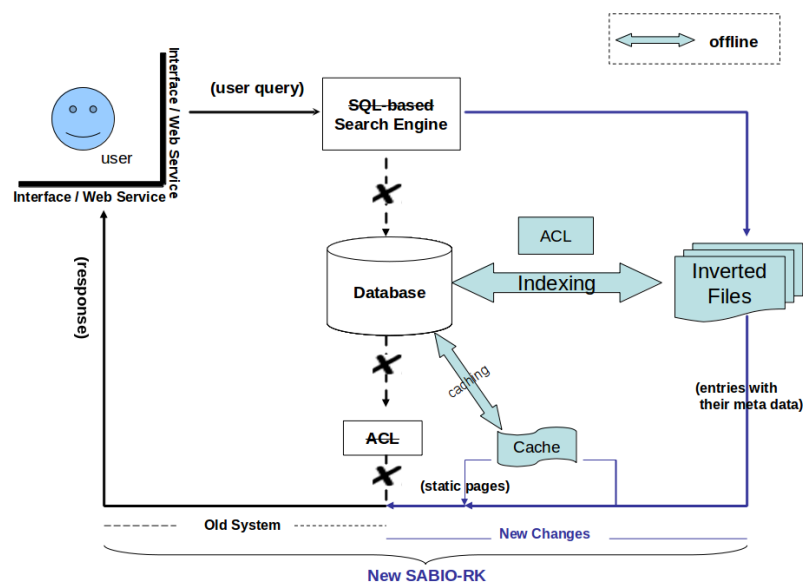


Figure 5: SABIO-RK System Change Overview

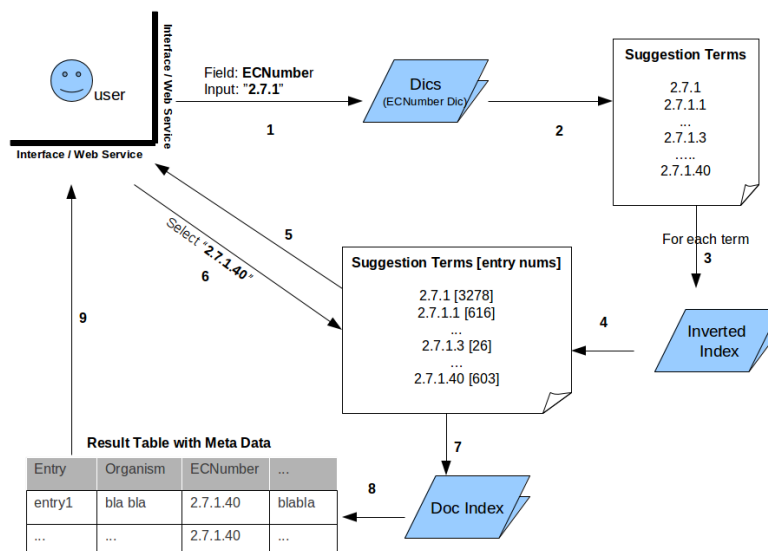


Figure 6: One Example Query Executed in New SABIO-RK System

each resulted kinetic law entry and compares with the profile of current user then judges whether such user has the permission to see all returned entries.

Given the security solution described above, determining the access rights for a given user and each returned entry causes a new bottleneck in addition to slow queries.

In order to speed up the use of ACLs, we index the access list information for each entry during the offline indexing procedure (please see Figure 5). An external 'ROLE' field and the encoded permission value are appended to the stored meta-information of index object. During query time, the system will append an AND query to user's query set automatically, according to current user's context. For example, if current user is an anonymous user (*Everybody*) and his query set is

```
{ [Reactant:O2] AND [ECNumber:2.7.1.40] }
```

now the query set changes to

```
{ [Reactant:O2] AND [ECNumber:2.7.1.40] AND [Everybody:1] }.
```

The system will only return entries which has indexed objects containing the meta-information <Everybody,1>. In contrast, for those entries which has stored meta-information <Everybody,0>, which means they are hidden from anonymous users, will not returned.

### 3.4 Caching For Optimization

Caching technology is also used in the new system as an optimization in the new SABIO-RK system. It stores those relevant static information to save the on-line generating time.

In old SABIO-RK system, the web application needed to generate the detailed information pages at run time for each returned entry and appended under corresponding access links. Though in the detail information pages there are information which might be or might not be users interest, the on-line querying and generating time are still costly. And such time is also depended on the numbers of returned entries and increased linearly.

Since complete information of a certain data entry is rather static, in the new SABIO-RK system, a cache is used (please see Figure 5) to pre-generate and preserve these pages offline. When searched entries with their meta data are returned, associated static entry detail pages are read from the cache and appended directly.

## 4 New User Interfaces and Web Services

Based on our new system and corresponding search mechanism, a new web interface and web services have been built.

### 4.1 User Interface

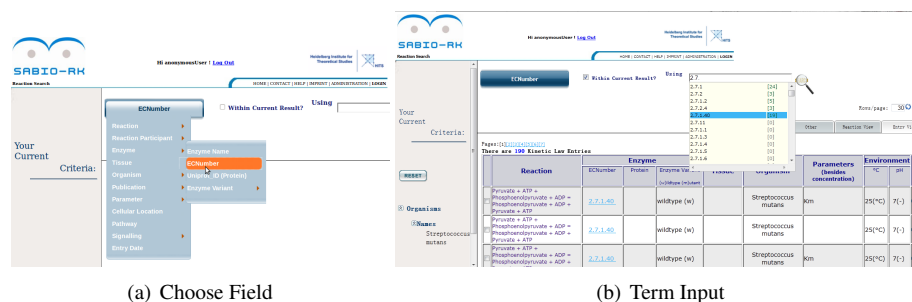


Figure 7: User Interface Auto-completion

Benefitting from the inverted indexing structure and its resulting fast response, the new interface is now able to provide summary information to users dynamically as they enter input into the search field. As shown in Figure 7(b), besides providing ordinary auto-complete for term spelling, the suggestion term list also contains the information about how many entry hits there are according to user's queries intersection. Suggestion terms which have "[0]" corresponding entries are grayed in order to prevent user spending time



on terms with zero hits [Hea09] in our database.

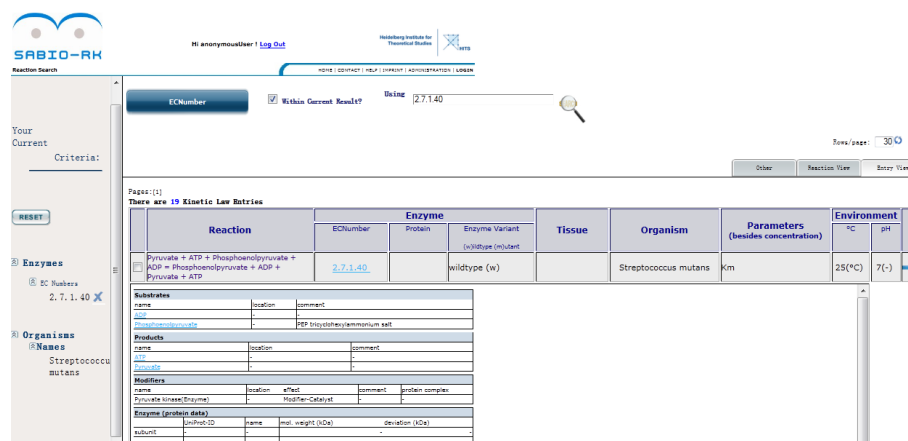


Figure 8 shows the SABIO-RK web interface. The search bar at the top contains the EC number "2.7.1.40". The sidebar on the left shows filters for "Enzymes" and "Organisms". The main results table displays the following data:

Reaction	ECNumber	Enzyme	Tissue	Organism	Parameters (besides concentration)	Environment
Pyruvate + ATP + Phosphoenolpyruvate + ADP = Phosphoenolpyruvate + ADP + Pyruvate + ATP	2.7.1.40	wildtype (w)		Streptococcus mutans	Km	25(°C) 7(-)

Below the table, there are expandable sections for "Substrates", "Products", "Modifiers", and "Enzyme (protein data)".

Figure 8: User Interface Result Table

After the search button is pressed, returned entries and some of its meta data forms the informative result table. Users are able to switch between different views, such as "entry based view" and "reaction based view" according to their customized need. And after have an overview about returned entries, user could expand those interested ones and expand to get their detail information, (please see Figure 8). And those detail pages about each entry is static pages retrieved from our cache, which is described in Section. 3.4.

## 4.2 RESTful Web Services

In addition to the new web user interface, we have developed a new suite of web services allowing programmatic access to SABIO-RK. The existing SOAP based services have several disadvantages, both for users and for the maintenance of the SABIO-RK system. From the user perspective the services were complex to use, requiring many steps to query for entries and then retrieve the data for each entry which could then be exported into the SBML format if required. From a software maintenance perspective, the lack of reuse of existing code for the user interface meant that an evolution in the functionality of the user interface and the web services remained quite separate. The aim of the new web services is to provide a faster and simpler service interface for the users, and to integrate the code more closely with the web user interface, providing consistency between the interfaces and utilizing the indexing, caching and ACL improvements which have been detailed in previous sections.

Access to the new web services is gained via simple HTTP requests, following a Representational State Transfer (REST) based approach [RR07]. The entry points we provided are simple URLs, with request parameters used to specify the queries to be performed.

Entries may be retrieved directly if the kinetic law id number is known, or entries may be searched using the same properties which may be specified using the user interface search. Searches are made using a controlled, documented vocabulary to construct a URL containing the desired attributes and corresponding entry ids are retrieved using the same search inverted indices. The output of the webservice is in the xml-based SBML format, a standard for describing biological models, making the data readily available to other applications, such as biological modelling platforms.

## 5 Improvements

A comparison test has been performed via the old and new SABIO-RK web user interfaces. The first case is built on the old system with the SQL-based mechanism and the later one is over the new system which is based on our pump-data-into-the-index-offline solution.

The task is designed according a simple search use case. Suppose the user wants to search for kinetic law data entries which has ECNumber: 2.7.1.40, and returned entries should be sorted according to their perspective reactions. Both old SABIO-RK and new SABIO-RK can functionally fulfill the information need described in such use case. The time spent on the task over both systems are compared in underlying Figure.9. The comparison result shows that the new solution makes the search efficiency about 50 times faster than the previous system.

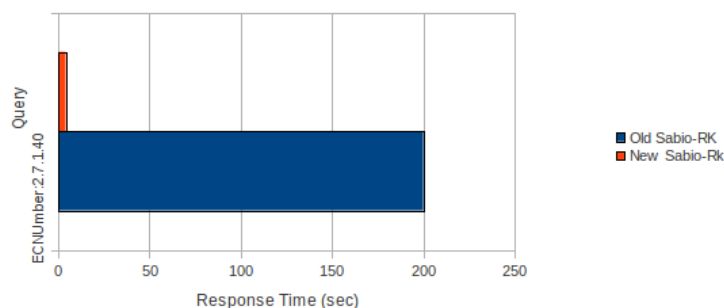


Figure 9: Query Speed Comparison

## 6 Future Work

With out light-weighted solution, the new SABIO-RK system is mostly running in the way we expected. Future work for further improving our system is to integrate a crawler

like updating mechanism which could track the modification of entries in our database and update both inverted index and cache content accordingly.

## References

- [BGM07] Clinton Begin, Brandon Goodin, and Larry Meadors. *Ibatis in Action*. Manning Publications Co., Greenwich, CT, USA, 2007.
- [GH04] Otis Gospodnetic and Erik Hatcher. *Lucene in Action (In Action series)*. Manning Publications, December 2004.
- [Hea09] Marti A. Hearst. *Search User Interfaces*. Cambridge University Press, 1 edition, September 2009.
- [Mul10] Peter Mularien. *Spring Security 3*. Packt Publishing, 2010.
- [RGK<sup>+</sup>07] Isabel Rojas, Martin Golebiewski, Renate Kania, Olga Krebs, Saqib Mir, Andreas Weidemann, and Ulrike Wittig. SABIO-RK: a database for biochemical reactions and their kinetics. *BMC Systems Biology*, 1(Suppl 1), 2007.
- [RR07] Leonard Richardson and Sam Ruby. *Restful web services*. O'Reilly, first edition, 2007.
- [SW05] Qi Su and Jennifer Widom. Indexing Relational Database Content Offline for Efficient Keyword-Based Search. In *Proceedings of the 9th International Database Engineering & Application Symposium*, pages 297–306, Washington, DC, USA, 2005. IEEE Computer Society.