

VL2: Building a Search Engine for Democracy

K. Sreepathi Pai

kspb46@yahoo.com

Don Bosco Institute of Technology

Premier Automobiles Road,

Kurla (West),

Mumbai 400 057.

Abstract

The ability to find a name in the electoral rolls is a very important requirement in an election. Unfortunately, no automated search engine currently exists for the world's biggest democracy: India.

VL2, described in this paper, is an automated, web-based search engine developed for searching through the electoral rolls. It includes features like phonetic matching (in Indian languages) and multilingual interfaces for maximum usability and accessibility. Many measures are also taken to ensure VL2 usability on machines that may not support multilingual software.

VL2 is fully-functional, and has been tested with real data. These tests have shown VL2 to be remarkably fast, accurate and relevant. Currently, VL2 fulfills the need of voters to find information about their voting booth, its address and so on. It is envisioned that VL2 may also be used in the future for a completely automated search-assisted voter registration system.

Keywords: e-governance, electoral rolls, search engine, phonetic matching, SouIndics

1 Introduction

In the last Lok Sabha elections, a large number of voters found their names missing, and were left without any recourse to remedy their situation. The primary cause of this was the inability to locate their names in the voterlist.

Clearly, computers can be used to help solve this problem. In this paper is described *VL2*, a fully-functional, working search engine for the Electoral Rolls, purpose-built to help solve this problem.

VL2 is standards-based, works with the standard electoral data as in the format with the Election Commission (with automatic conversions wherever needed), and provides a Web-based interface to the voter list data.

2 Acknowledgments

The idea of a search-engine like interface to the voterlist data was originally conceived by Prof. Jitendra Shah, of VJTI and Indictrans. He was also responsible for proving that the concept works by producing the initial version of the software. The author gratefully acknowledges his guidance and exceptional support for the development of the current version of the software that is described here. The support from IndicTrans, especially from Swapnil Hajare, who helped with development of VL2, is deeply appreciated.

3 Challenges

The electoral rolls present several challenges to anybody working on a search engine for it. The most important among these are summarized below:

1. **Multilingual Data:** Data in the Electoral database is in the local language. For example, Marathi in Maharashtra, Kannada in Karnataka, and so on. For a search engine to be truly national, it must be able to handle data in all of these languages.
2. **Spellings:** The primary search keys are names and addresses. This is the biggest stumbling block to accurate results. Consider what happens when, say a Christian name or a South Indian name is written in Marathi. In fact, even local names have considerable variation in spellings in the database. In a quick test across a single constituency (about 3 lakh voters), we have found that a name is usually spelt on average in 3 different ways!
3. **Scale of data:** The scale of the data, evaluated per city, is on average with most large databases. For example, a city like Pune contains about 50–60 lakh people on its electoral rolls.
4. **Speed and Relevancy:** Given the scale of the data, and the imperfections in spelling; the search engine must be fast and produce relevant results. This will enhance user experience, as well as enable deployment of the software in time-sensitive environments like that of a helpline/call-center.
5. **Legacy data formats:** The database is usually in the FoxPro .dbf format. Hence a conversion process that converts this data to the more modern SQL-based databases is required.

In addition to the above challenges, the software must, of course, be easy to use. In an ideal scenario, the software should be usable across the Web by any person, and should be just like any other search engine!

4 Current Approach

Most state governments today do use IT for enhancing the electoral process. In fact, most governments have put up the electoral rolls on the Internet. Unfortunately, the approach used for locating a name is not very different from the existing manual process.

Usually, the user is taken to a website which first requires geographic details such as city/district, assembly constituency, etc. Once such information has been

obtained (it is assumed that the user knows such information – which is *not* true all the time), a list of voters in that area is provided on a web page. The user is then expected to look through the list to find his/her name.

Clearly, this does not work if the name is not found. Either the user must repeat the process with different details, or give up. With most web users pampered by Google, it is not difficult to see how effective this approach is for the most important case – when the name is *not* found.

5 Our Approach

Most approaches today are *data-driven*, in that the user is secondary to the format of the data. The approach used by VL2 is *user-driven*, the user specifies what to search for. In this sense, VL2 is a true search engine for the electoral data.

The only information the user must provide is his name and surname. VL2 requires no other data. Of course, it can also use information such as address (building name, etc.), and assembly constituency to deliver more specific results, in the case of very common names.

[\[Help\]](#) [\[Marathi\]](#)

Important Note: Name and address *must* be entered in the Devanagari (Marathi) script with Unicode keyboards (such as INSCRIPT). If you prefer, the ITRANS coding scheme will allow you to enter Devanagari using the English alphabet on a normal keyboard. The *To Marathi* button can then convert ITRANS to Devanagari. For details on the ITRANS-to-Devanagari scheme, please [click here](#).

Search Assembly:

Name:
Name and surname

Address:
Area, locality, building name, etc.

Or

Photo ID No.: (e.g. MT/06/035/1234567)

Options

Default results are in Unicode (Devanagari)

Show results also in non-Unicode

Show results also in English

Maximum results: (max. 100)

Powered by: [PostgreSQL](#) | [PHP](#) | [tsearch2](#)
 Phonetic matching powered by: [Soundix](#)

Figure 1 The main search application.

The interface is carefully built to resemble most contemporary search engines, so that adoption by users is not hampered by a steep learning curve. Once a search has been carried out, the results are shown ordered by relevancy. Once the name has been found, a user clicking on it is led to a page giving voter details, such as booth number, booth address, etc.

Most of the challenges mentioned in Section 3 have been tackled, as outlined in the following sections.

5.1 Multilingual Issues

Tackling multilinguality is not very difficult in theory. As per best practices today, the software is based entirely on Unicode. This means that all input, output, and storage is Unicode-based. The conversion process automatically converts the ISCII-based electoral data to Unicode.

On Unicode-based client operating systems like GNU/Linux, Windows XP, Windows 2000, etc., the search engine can accept input in Unicode as well display output in Unicode (which is also font-independent).

Non-Unicode operating systems do pose input/output problems, which have been solved using the following features:

1. **Input in ITRANS:** The ITRANS (1) transliteration standard provides a common way to input Indian languages using the English alphabet only. So for systems that do not support Unicode-based keyboards like INSCRIPT, the users can use ordinary keyboards for typing in the local languages. ITRANS is then converted to the Unicode.

The ITRANS input feature is also convenient for users who are not conversant with local language keyboards, but who otherwise have Unicode-supporting systems.

2. **Output in Legacy ASCII-based fonts:** The software can automatically convert all Unicode output to a legacy ASCII font encoding like Shusha, ISFOC, etc. This enables users on legacy operating systems (like Windows 9x) to view the data in the local script limited only by the particular font chosen.
3. **Output in Roman script:** Since fonts for native scripts may not be available always, or it may not be feasible to install them, the software also provides output in the Roman script. This causes all local language Unicode output to be Romanized using a slightly modified variant of the ITRANS transliteration

करुणसवामी
karunasavaamii
करुणसवामी

Figure 2 By line, first line shows Unicode-font output, second shows Romanized output, the third line shows ASCII-font (Shusha) output.

standard. This feature can also be used by users who may not be comfortable with the script of the local language.

5.2 Phonetic Name/Address matching

To solve the problem of variations in spellings across the languages, it is evident we need to use approximate matching techniques. Since the data being searched is names, it is clear that a *phonetic* matching technique be used. Such a technique would allow matches based on how a name *sounds*, rather than how it is written.

Phonetic matching algorithms for the Indian languages, unfortunately, do not exist. This led to the development of *SouIndics*, a phonetic matching algorithm for Indian languages. Like existing phonetic matching techniques for the English language (SoundEx and Metaphone), *SouIndics* returns a code for a given name. If the codes for two names match, the two names are identical phonetically.

With proper integration of *SouIndics* into the database, searches using *SouIndics* include almost all variants of the names. The results of this wide search are then reordered using a stricter variant of *SouIndics*, called *SouIndicsV*, before display.

Thus, exact spellings of names are not required, a spelling that approximates the sound well is good enough.

Closest matches for चेल्लम् using SouIndics

चेल्ला	चेल्लमा	चल्लम	चील्लाम्मा	चल्लाम्मा
चालाम	चल्ला	चेल्लामा	चल्लामा	चेल्लामा
चल्लामा	चील्लाम्मा	छल्लामा	चील्लामा	चल्लाम्मा
चेल्लाम	चल्लामा	छेल्लामा	चल्लाम्मा	चेल्लाम्मा
छेल्लाम	चेल्लाम्मा	चेल्लामु	चल्लाम्मा	चेल्लामा
चेल्लाम्मा				

Figure 3 A demonstration of *SouIndics* capabilities.

5.3 Platform and Performance

For maximum flexibility and performance–cost ratio, the software has been completely built using Free/Open Source components. The components are:

1. *Operating System*: GNU/Linux
2. *Database Server*: PostgreSQL 7.4 with tsearch2
3. *Programming Language*: PHP/4.3
4. *Web server*: Apache 2.0

The search engine is fast. Tests on searches with any combination of name/address across six constituencies of Pune (~25 lakh records), return relevant results in *less than 1 second*. In all tests so far, including one conducted by C-DAC, the software has never failed to return a result, unless of course the name does not exist in the database.

6 Future Directions

Although the software is fully functional, many features are planned to enhance the user experience:

- **Multilingual Input**: Since most languages in India are phonetically similar, it is possible to allow a user to enter his name in his native language (say Gujarati), when searching across a Marathi database.
- **Integration into registration process**: As the core of the software is independent of the current web interface, it becomes possible to integrate the search engine into any other software that may require such capability. Top among these is an search-assisted registration system, that would considerably improve the current manual process.

Indeed, even in its current form, the current software is suitable for use in the registration process, to prevent multiple registrations and other such errors.

7 Conclusion

As with all software created to solve a problem for the first time, there is very little awareness on the possibilities of VL2. Although it was created to solve the “find your name in the voterlist” problem efficiently, it can be applied to other processes that require search capabilities as well. Paramount among these is the voter registration process, which requires a check of whether the name already exists in the rolls or not, currently a tedious manual process.

Any tool that makes the electoral rolls more accessible makes *democracy* more accessible. Clearly, VL2 is such a tool. With VL2, problems of missing names may soon become a thing of the past.

8 References

1. Chopde, Avinash. *ITRANS - Indian Language Transliteration Package*.
<http://www.aczone.org/itrans/>
2. *The Indictrans Website*.
<http://www.indictrans.org>
3. *The Voterlist page at Indictrans.org*.
<http://www.indictrans.org/voterlist/English/voter.php>
4. Unicode Consortium, *The Unicode Standard, Version 4.0*, Boston, MA, Addison-Wesley, 2003. <http://www.unicode.org>