CHAPTER 6. CONCLUSION AND FUTURE WORK

6.1 Conclusion

As detailed in section 1.3, proper nouns or out of vocabulary words could be spelt in various different ways. The problem increases when names come from different ethnic origins and are written in different languages. Therefore searching for proper nouns is difficult. As described in section 1.4, the objective of this research was to build a rule based search engine to search a names database using a Sinhala input string, where the search should match records even if they are spelt differently.

The test results in the Chapter 5 clearly show that the objectives of the research have been met. For example, a person who would enter an input string 'විසාකා' would even match records with names spelt as 'විශාඛා', 'විසාඛා' or 'විශාකා'. As seen in the results analysis in section 5.3, cumulative rule sets are built to cater different levels of character replacements. A special character replacement rule set for Tamil words has also been built. A web based user interface is built to use the system, where a user can type in the search string, set the character replacement level and choose to enable rules for Tamil words. A database that with 1000+ names in Sinhala Unicode text has been built. The schema of this database supports names data to be stored in Sinhala, Tamil and English languages. Therefore, the table could consist of either original or transliterated data. As described in section 5.4.2 the application was able to meet the expectation of independent external evaluators 85% of the time.

The current rule base is documented and is available in Appendix A of this dissertation. New rules can be added to the system as they are discovered or considered necessary, since they are defined in text files as described in section 4.3.3.

The system built as a result of this research can be considered as pluggable technology and this can be expanded or wrapped into many useful multi-lingual applications in government and business organizations. In addition, a number of improvements and enhancement to this application could be done to expand the capabilities of this system. Some of them are described in the next section.

6.2 Future Work

As described in Chapter 4, this system has been designed with future expansions in mind. Therefore new features and new additions can be easily done, re-using the components that are built as part of this system. This section describes some of the possibilities.

6.2.1 User friendly Rule Authoring Interface

Currently, the rules are defined in a text file in the manner that is described in section 4.3.3. An interface that is more user-friendly can be developed for this purpose. This could be done by taking advantage of the domain specific language (DSL) development features of the JBossRules rule engine. A tabular structure or some spreadsheet can be developed to author the rules so that an end user can directly enter the rules in that interface and the system could import the rules from the interface directly. This way the intelligence of the system could be easily grown.

6.2.2 Expanding in to Other Languages

Although in this application, it was scoped to take a Sinhala input string and do processing totally in Sinhala, the same core components can be used to process words in any language. New rules for other languages have to be added to the rule base. The rules of other languages can co-exist with the Sinhala character replacement rules because the characters from the other languages do not overlap with Sinhala so the application could function in several languages. In addition, it can be noted that rules for other Indic languages can be directly derived from most of the existing Sinhala rules, by shifting the Unicode code points appropriately. This is because all the Indic languages share the same structure and ordering of characters.

The input validator has to be slightly modified to allow other language input strings to the system, and no other component has to be changed for this purpose. New rules can be written in the same structure as described in section 4.3.3.

6.2.3 Combining with Other Multilingual Applications

One can combine this application with the other multilingual research applications that are developed and are being developed. The back-ends of these systems could be integrated into a powerful multilingual processing application, which has transliteration, collation and search capability. Integration with other useful business or government applications also falls in to this category. For example, a government organization can incorporate this engine to their systems and search their databases through this application.

6.2.4 Inverted Index Lookup instead of a Database

The back-end of this application could be changed to look up an inverted index of words instead of searching the database. The index could point to relevant actual records. A great performance improvement can be achieved by using an inverted index, especially when the database, which needed to be searched, is extremely large. Indexing process can happen periodically at non-peak times. By using an inverted index, the functionality of this system can be incorporated to any kind of search application, not only databases.

6.2.5 Improved Intelligence

At present user has to select whether to enable/disable rules for Tamil words and the level of character replacement. Instead of this an additional pre-processing set of rules can be set up to automatically determine whether a name is coming from a Tamil origin and thus to enable character replacement rules for Tamil or to suggest the user to do so. In addition, the engine can be improved to incrementally increase the level of character replacement if matching records are not found.

6.2.6 Auto Correction or Search Assist

While the user is entering the input text, an auto completion list of words can be displayed in a drop down list to assist the user, so that the user could select and auto complete what is being typed. This could be achieved through an inverted index suggested in the preceding section. The index has to be searched as the user types in and words from the index with matching starting substring can be presented in the drop down list. Ajax features can be incorporated in the web interface to initiate requests to search as the user types the string.

6.2.7 Performance Improvements

Performance improvements can be done in various methods; few of them are suggested here. As mentioned in section 5.6 the time taken for a transaction is typically less than a second. Majority of this time is consumed in loading the rule sets. Therefore, one of the main improvements that can be done is to keep the rule sets cached and enable/disable them in memory rather than reading them and loading for each transaction.

As the size of the database grows, searching will take much longer. Performance in this area can be improved by using an inverted index as mentioned in section 6.2.4 before the database.

User perceived performance can be improved by performing incremental searches for different character replacement levels as suggested in section 6.2.5 and displaying these results incrementally. However, if there are no matches for lower character replacement levels this could also result a negative impact because higher character replacement level words will be searched later.