Part IV

Epilogue: Conclusion and Future Work
CHAPTER 11
SUMMARY AND CONCLUSION

11.1 Summary

This work can be summarized as follows:

- In part I, we introduced basic elements of automata theory and computer architecture that served as basis for understanding various aspects of the thesis. In particular, the formal definition of a string recognizer in terms of its denotational semantics was given. The chapter also depicted the complex operational diagram of modern processors which showed some of the detail of how cache memory plays an important role in an algorithm’s performance.

- Part II was devoted to the investigation of alternative algorithms for string recognition. To this end, we started off (in Chapter 3) by presenting the core TD and HC algorithms which led to the design of the core MM algorithm. Furthermore, the denotational semantics of the core algorithms was specified in terms of a single function. The next three chapters (Chapters 4, 5 & 6) of part II were devoted to investigations of new strategies for the implementation of DFA-based string recognizers. The function defining the recognizer’s denotational semantics was then expanded to incorporate appropriate variables that specify whether or not, and —where relevant— the way in which, each of the respective strategies is to be deployed. In Chapter 7, a unified version of this denotational semantics function was suggested, taking into account all strategy variables discussed in previous chapters. The formalism resulted in the suggestion of a total of 168 algorithms. This was then used for the construction of a taxonomy tree whose nodes represent the different algorithms. The taxonomy was further mapped into a class-diagram (Chapter 8) that represented the architectural view for a toolkit of DFA-based string recognizers.

- In part III, attention was given on the implementation of some algorithms. An introductory note on the way our experiments were conducted as well as the software and hardware used for the experiments was given in Chapter 9. In Chapter 10, experimental results of some selected algorithms were discussed. In general, experiments conducted revealed that algorithms suggested throughout the thesis are of interest and could prove useful when processing particular kinds of strings. It was also shown in this part that each of the investigated algorithms could be processed at optimum, as long as the kind of string to
be processed reflects the algorithm’s best case behaviour. Moreover, although it was already known that the core HC algorithm outperforms the core TD algorithm for automata of size in the order of hundreds, it was also shown that the MM algorithm could be used as a performance booster since it has been explicitly designed to take advantage of both HC and TD capabilities.

Over the past few years, various aspects of our work have been published in scientific journals, peer reviewed conferences, and workshops:

1. [NKW06a] represents the initial work that lead us to the investigation of new implementation strategies since it was established that there is a correlation between algorithm performance and cache memory capabilities.

2. In [NWK04, NWK05] various ideas were being shaped and we suggested a framework for the dynamic implementation of FAs whose original concept is still under investigation, and the notion of dynamic implementation resulted in the investigation of the DSA strategy whose early publication in [NKW05b] referred the strategy as state reordering, and an enhanced version in [NKW06b] kept the terminology DSA. The hardcoded version of the DSA strategy was published in [NKW06c].

3. Based on previous results, the idea of SpO and AVC strategies were investigated. Resulting in a formal characterization of string recognizers using the notion of denotational semantics. An early version of the idea was presented at a workshop ([Nga05]) where the notion of taxonomy was first suggested. Further improvements on the idea yielded the publication in [NKW06d] of a taxonomy of DFA-based string processors.

4. Finally, the performance of all the TD algorithms was published in [NWK06].

In the next section, we provide a conclusion to our work.

11.2 Conclusion

We believe that the followings have been achieved in this thesis:

- **Denotational Semantics of String Recognizers:** Our work has established the basic foundation for mathematically representing DFA-based string recognizers. Although our parametric function relied on the suggested implementation strategies, many other strategies could be investigated, resulting therefore to more algorithms, and of course several challenges for solving FA-based string processing problems.

- **New DFA-recognition algorithms:** To date, DFA-based string recognition has been limited to two alternative solutions: the core TD and HC algorithms. Our work has provided for up to 166 different algorithms.
• **Knowledge of hardware**: Although the design of optimal algorithms requires sound theoretical knowledge, their implementation requires sound knowledge of the computation medium on which the implemented algorithm is to be processed. This work has empirically proven that data/instructions organization plays an important role on the overall efficiency of any given algorithm.

• **Processor Performance**: Although our work only relied on the design of cache optimized algorithms, the complex operational diagram of modern processors is made of various other components that may be regarded as time consumers. This work has thus raised the need to algorithmically investigate the effect of those components on running programs in general. Such investigation may lead to the design of algorithms that take advantage of the capabilities of those various components.

Based on the above mentioned lessons learnt from our work, a personal perspective could be summarized as follows: the design of efficient algorithms based on theoretical abstractions is good; but a design that takes into account the potentialities and weaknesses of the computational medium on which the algorithm is to be processed, is better. It is thus of importance to account for hardware capabilities when designing algorithms that have to account for efficiency. In the next chapter, we present further directions that this work could take.
CHAPTER 12
FUTURE WORK

Although this research has closed the gap in the availability of alternative implementation strategies for FA-based string processors, there is a variety of future challenges that require further investigations. Each subsection below contains a list of projects that could be undertaken by future researchers.

12.1 Projects of limited scale

The following projects are of limited scale and could conceivably be undertaken by junior graduate students (for example, in their 4th year of study).

- **Algorithm Performance Analysis**: Carry out an analysis of the performance of the various algorithms, based on artificial data, in order to capture their strengths and weaknesses. In this project the researcher should select an algorithm from the 168 available. The best case behaviour of each algorithm should be determined as well as a complete analysis of the effect of caching on the algorithm. The researcher should also determine the effect of any other hardware component (pipeline, trace-cache, branch prediction, etc.) on the algorithm.

- **Applied Algorithms Analysis**: This involves an investigation of each algorithm as applied to specific problems such as network intrusion detection, tandem repeat finding, natural and computer virus scanning, etc. In this project, an algorithm should be identified and various analysis on it should be performed, using a real-life application. There is a need to investigate whether or not the chosen algorithm will outperform the conventional one (usually the TD). If the algorithm appears to under-perform its conventional counterpart, then there would be a need to investigate possible ways of improving the algorithm.

12.2 Medium-scale projects

- **A toolkit for FA-based string recognition**: This project aims at improving and implementing the architectural design suggested in Chapter 8. The project is currently being undertaken as a postgraduate (masters level) exercise.
• **Extension of the taxonomy for DFA-based string recognizers.** The taxonomy suggested in Chapter 7 relied on the three implementation strategies that were used as parameter variables associated with the core algorithms. However, investigations could also be conducted not only on new strategies, but also on any other data-structure based recognizers such as linked-lists, trees, graphs, etc. A new taxonomy of DFA-based string processing algorithms could be proposed.

• **Improvement of the HC, TD, and MM algorithms.** This would be a project in high-performance computing. Alternatives ways should be investigated of how to obtain even faster TD, HC, and MM algorithms.

• **Platform specific investigation of the algorithms.** Various hardware platforms (such as Intel, Power-PC, Silicon Graphics, and the like) should be used to conduct experiments testing for performance of the various algorithms. The effects of cache, pipelining, branch prediction on various platforms should be investigated.

### 12.3 Advanced research projects

• **Hardware implementation of DFA-based string processing.** The hard-coded algorithm would be the starting point of this challenging project. All the HC algorithms should be translated to hardware. The advantages/disadvantages of implementing string recognizers on hardware should be explored. This might result in the implementation of specialized DFA-based applications on hardware.

• **Design and implementation of other cache optimized applications.** There may be other computational problems, unrelated to DFA-based string recognition, that are performance sensitive and that could be investigated following the same approach used throughout this thesis.

• **Investigation of other hardware performance metrics:** The operational diagram suggested in Chapter 3 requires further investigations in that we only restricted ourselves to the cache’s locality of references. However, several other aspects could be envisaged for performance enhancement. Further analysis of the operational diagram could lead to suggestions on whether and how it might be possible to algorithmically influence positively the performance of those components.

### 12.4 End note

The source code for the various experiments conducted have intentionally not been released, but are available on request by e-mailing the author.
BIBLIOGRAPHY


