LIBRARY
Te Tumu Herenga
THE UNIVERSITY OF AUCKLAND

RESEARCHSPACE@AUCKLAND

http://researchspace.auckland.ac.nz

ResearchSpace@Auckland

Copyright Statement

The digital copy of this thesis is protected by the Copyright Act 1994 (New Zealand).

This thesis may be consulted by you, provided you comply with the provisions of the Act and the following conditions of use:

- Any use you make of these documents or images must be for research or private study purposes only, and you may not make them available to any other person.
- Authors control the copyright of their thesis. You will recognise the author's right to be identified as the author of this thesis, and due acknowledgement will be made to the author where appropriate.
- You will obtain the author's permission before publishing any material from their thesis.

To request permissions please use the Feedback form on our webpage.
http://researchspace.auckland.ac.nz/feedback

General copyright and disclaimer

In addition to the above conditions, authors give their consent for the digital copy of their work to be used subject to the conditions specified on the Library Thesis Consent Form.

Department of Electrical & Computer Engineering
The University of Auckland
New Zealand

Embedded Speech Recognition Systems

Octavian Cheng

September 2008

Supervisors: Dr Waleed Abdulla
Prof Zoran Salcic

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS OF DOCTOR OF PHILOSOPHY IN ENGINEERING

Abstract

Despite many research efforts, automatic speech recognition (ASR) is still not widely used in embedded systems. One of the reasons for this is that embedded platforms are limited in terms of processing power and computing resources. These limitations contribute to an increase in decoding time, which can lead to poor user experiences. As a result, a compromise is often made between decoding speed and other performance criteria, such as recognition accuracy and vocabulary size.

There are two main objectives in this thesis. The first objective is to develop an embedded ASR system which is suitable for real-time applications. We focus on various kinds of approaches that can reduce the decoding time without severe performance degradation in recognition accuracy and vocabulary size. The task is a 993-word Resource Management (RM1) task, which serves as the benchmark for command-and-control type of applications. The target platform is an Altera Nios II softcore processor system running at 120MHz. The system is synthesized on a Stratix II FPGA device.

Three approaches have been successfully adopted on the target platform. First, due to lack of hardware support for floating-point arithmetic, we propose a framework for converting various data types to fixed-point formats. Experimental results on the RM1 task show that the fixed-point system is 9 times faster than the floating-point system without any degradation in recognition accuracy. Second, a hardware-software co-processing ASR system is developed. The architecture mainly consists of a Nios II processor and a hardware accelerator. The hardware accelerator is responsible for the calculation of the Gaussian mixture model (GMM) emission probabilities, which is the major computational bottleneck. The co-processing system is tested on the same RM1 task. In comparison with the pure software-based fixed-point system, the average real-time factor improves from 1.87 to 0.62 (about 3 times speed-up). The word accuracy rate is 93.33%, which is the same as that of the pure software-based system. Third, in order to further improve the timing performance, an adaptive beam pruning algorithm is introduced, which applies tighter pruning when there are a large number of active hypotheses. For the same RM1 task, the average real-time factor further reduces to 0.54. The word accuracy rate is 93.16%.

Apart from recognition accuracy, decoding speed and vocabulary size, another point of consideration when developing a practical ASR application is the adaptability of the system. An ASR system is more useful if it can cope with changes that are introduced by users, for example, new words and new grammar rules. In addition, the system can also automatically update the underlying knowledge sources, such as language model probabilities, for better recognition accuracy. Since the knowledge sources need to be adaptable, it is inflexible to statically combine them. It is because on-line modification becomes difficult once all the knowledge sources have been combined into one static search space.

The second objective of the thesis is to develop an algorithm which allows dynamic integration of knowledge sources during decoding. In this approach, each knowledge source is represented by a weighted finite state transducer (WFST). The knowledge source that is subject to adaptation is factorized from the entire search space. The adapted knowledge source is then combined with the others during decoding. In this thesis, we propose a generalized dynamic WFST composition algorithm, which avoids the creation of non-coaccessible paths, performs weight look-ahead and does not impose any constraints to the topology of the WFSTs. Experimental results on Wall Street Journal (WSJ1) 20k-word trigram task show that our proposed approach has a better word accuracy versus real-time factor characteristics than other dynamic composition approaches.

Acknowledgements

I would like to thank my thesis supervisors, Dr Waleed Abdulla and Prof Zoran Salcic, for their guidance and support. Throughout my PhD years, I learnt a lot from them. I always received many useful suggestions from them in our meetings, especially when problems occurred and research results were not as promising as expected.

I also would like to thank Prof Hervé Bourlard for offering me a research internship at IDIAP, Switzerland. The stay at IDIAP broadened my view on speech recognition research. I would like to acknowledge the members of IDIAP. In particular, I would like to thank John Dines and Mathew Magimai Doss for the collaboration.

I would like to thank New Zealand Tertiary Education Commission for the scholarship offer. With their financial support, I can focus solely on my thesis work.

Finally, I would like to thank my parents for their support and encouragement.

Contents

1 Introduction **1**

1.1 Objectives of the thesis 1

1.2 Current embedded ASR system architectures 3

1.3 Contributions of the thesis 4

1.4 Outline of the thesis 5

2 Fundamentals of speech recognition **7**

2.1 The decoding problem 7

2.2 Feature extraction 8

2.3 Acoustic modelling 10

2.3.1 Hidden Markov Model (HMM) 10

2.3.2 Evaluation of HMM 13

2.3.3 Decoding of HMM 14

2.3.4 Training of HMM 15

2.3.5 HMM/GMM system 17

2.3.6 Hybrid HMM/ANN system 18

2.4 Language modelling 19

2.5 Decoding 20

2.6 Search space representation 21

2.6.1 Re-entrant lexical tree 21

2.6.2 Weighted finite state transducer (WFST) 23

2.7 Search algorithm 25

2.7.1 Time-synchronous search 25

2.7.2 Time-asynchronous search 27

2.8 Performance metrics 28

2.9 Summary 28

3 WFST-based speech recognizer **29**

3.1 WFST theory 29

3.2 Static WFST composition 34
3.3 Overview of Juicer 35
3.4 Summary 36

4 Fixed-point speech recognition system 37
4.1 Feature extraction 37
4.1.1 Algorithm 37
4.1.2 Feature extraction with fixed-point formats 40
4.2 Emission probability calculation 42
4.2.1 Algorithm 42
4.2.2 Emission probability with fixed-point formats 43
4.3 Viterbi search 46
4.3.1 Algorithm 46
4.3.2 Viterbi search with fixed-point formats 46
4.4 Recognition accuracy 47
4.5 Summary 48

5 Pure software-based system 49
5.1 Target platform - Altera Nios II processor 49
5.2 System architecture 50
5.3 Timing profile 51
5.4 Summary 53

6 Hardware-software co-processing system 54
6.1 System architecture 54
6.2 GMM emission probability hardware accelerator 55
6.2.1 Datapath 55
6.2.2 Timing profile 59
6.2.3 Resource usage 61
6.3 Adaptive pruning 62
6.3.1 Algorithm 62
6.3.2 Timing profile 65
6.4 Performance evaluation 66
6.5 Summary 69

7 Dynamic composition of WFST 70
7.1 Motivation 70
7.2 Static WFST composition in ASR 71
7.3 Current Approaches to Dynamic WFST Composition 74

7.4 Proposed Approach to Dynamic WFST Composition 81
7.4.1 Finding the Anticipated Output Labels 81
7.4.2 The Dynamic Composition Algorithm 82
7.5 Experimental Results 84
7.6 Summary 89

8 Conclusions 90
8.1 Hardware-software co-processing ASR system 90
8.2 Dynamic composition of WFST 92
8.3 Future work 93

A Mel filter bank 95

B DCT and Liftering 97

List of Figures

2.1 Block diagram of a typical ASR system 8

2.2 The topology of an HMM example. In this example, there are 3 states. The probabilities of being in an HMM state at the start of a state sequence are called the *initial state probabilities*. State transitions are modelled by *transition probabilities*. The probabilities of the observation vectors are known as *emission probabilities*. Each HMM state has an associated emission probability distribution. 11

2.3 A re-entrant lexical tree. Language model probabilities are incorporated into the search at the leaf nodes of the tree. The asterisks in the langauge model probabilities denote the word histories. For example, if the language model is trigram, the asterisks represent word histories consisting of two words. 22

2.4 Search space represented by a WFST. There are only two words in the vocabulary. The language model is bigram. Each WFST transition $x : y/\omega$ has three attributes. x is an input symbol representing a triphone or biphone label. y is an output label representing a word. Labels can be ϵ, which are empty labels. ω is the weight representing the language model probability $P(.)$ or the back-off weight $B(.)$. Each triphone or biphone label is modelled by an HMM. 24

2.5 Pseudocode of time-synchronous Viterbi beam search 26

2.6 Pseudocode of the *Viterbi_search* function 27

3.1 An example of a WFST 29

3.2 Composition of T_1 and T_2 over the tropical semiring. 32

3.3 Determinization of T over the tropical semiring 33

3.4 Weight-pushing of T over the tropical semiring 33

3.5 Minimization of WFST 34

4.1 Flow diagram of MFCC 38

5.1 Architecture of the embedded platform for implementing the pure software-based speech recognizer 50
5.2 Real-time factor of 1200 utterances in the pure software-based fixed-point system. The utterances are from the test set of the RM1 corpus. Pruning beamwidth = 170. 52
5.3 Time delay (in seconds) of 1200 utterances in the pure software-based fixed-point system. Pruning beamwidth = 170. 53

6.1 System architecture of the hardware-software co-processing recognizer with the GMM hardware accelerator 55
6.2 Arithmetic unit of the GMM hardware accelerator. The GMM parameters ($\mu_{jm}^{(d)}$, $v_{jm}^{(d)}$, g_{jm} and C_{jm}) are "pumped" into the datapath in consecutive clock cycles. The log emission probability of each Gaussian mixture, $\log b_{jm}(\mathbf{o}_t)$, is passed to the log-add unit consecutively. The final output of the log-add unit is $\log b_j(\mathbf{o}_t)$. Number of clock cycles at each stage is also shown. 57
6.3 Datapath of the log-add unit. It performs the $x \oplus y$ operation, where $\oplus$ is the *log-add* operator. The result can be passed back to one of the input operands for recursive *log-add* operations. This unit accumulates the log probability of each Gaussian mixture, $\log b_{jm}(\mathbf{o}_t)$, and computes the log probability of all the Gaussian mixtures, $\log b_j(\mathbf{o}_t)$. 58
6.4 Double-buffering inside the GMM hardware accelerator. The arithmetic unit is reading from one buffer while another buffer is retrieving HMM parameters from off-chip memories. 59
6.5 Real-time factor of 1200 utterances in two different systems: Pure software-based system versus Hardware-software co-processing system. Pruning beamwidth = 170. 60
6.6 Time delay (in seconds) of 1200 utterances in two different systems: Pure software-based system versus Hardware-software co-processing system. Pruning beamwidth = 170. 61
6.7 Speech recognition algorithm with adaptive beam pruning 64
6.8 Real-time factor of 1200 utterances in Hardware-software co-processing system: Adaptive beam pruning versus Fixed beam pruning. 65
6.9 Time delay (in seconds) of 1200 utterances in Hardware-software co-processing system: Adaptive beam pruning versus Fixed beam pruning. 66

7.1 Static composition of $\tilde{L} \circ G$ over the tropical semiring. In $\tilde{L}$, #1 and #2 are the auxiliary word-end markers. 72

7.2 Pseudocode of the *WFST_composition* function. S_1 and S_2 are the state indices of the two constituent WFSTs which are deterministic. 73

7.3 The lexicon WFST ($\tilde{L}$) in Caseiro's approach. The $\tilde{L}$ transducer is partitioned into two regions - a prefix region (grey rectangle) and a suffix region (white rectangle). $\#_1$ and $\#_2$ are word-end markers. {} indicates an anticipated output symbol set. Two constituent WFSTs, $\tilde{L}$ and G, are composed using Caseiro's approach. The bold symbols and weights in $\tilde{L} \circ G$ indicate the changes compared with the *no lookahead* approach. 76

7.4 Pseudocode of the *WFST_dynamic_compose_caseiro* function. S_1 and S_2 are the state indices of the two constituent WFSTs which are deterministic. 77

7.5 Pseudocode of the *WFST_dynamic_prefix* function. $trans1$ is one of the ϵ-output transitions from S_1 in the prefix region. S_2 is the state index of the second constituent transducer. 79

7.6 Pseudocode of the *WFST_dynamic_non_eps* function. $trans1$ is one of the non-ϵ-output transitions from S_1. S_2 is the state index of the second constituent transducer. 80

7.7 A ($\tilde{C}_{opt} \circ \tilde{L}_{opt}$) WFST is segmented into *prefix* regions, where symbol and weight look-ahead is performed. For simplicity, only the output symbols are shown. Each ϵ-output transition has an anticipated output symbol set which is denoted by {...}. Each transition of the ($\tilde{C}_{opt} \circ \tilde{L}_{opt}$) transducer is substituted by an HMM. 82

7.8 Pseudocode of our proposed algorithm. It tries to propagate a *token* at S_1 of the ($\tilde{C}_{opt} \circ \tilde{L}_{opt}$) transducer. 83

7.9 Pseudocode of the *WFST_dynamic_proposed_prefix*() function. It checks whether the *token* can be passed to the $trans1$ transition. 85

7.10 Pseudocode of the *WFST_dynamic_proposed_non_eps*() function. 86

7.11 WER versus RTF of different approaches at various pruning beamwidths (150, 160, 180 and 200). Each data point on a curve corresponds to one particular pruning setting. The pruning beamwidth varies from the narrowest (the leftmost data point) to the widest (the rightmost data point). 88

7.12 RTF versus Average number of tokens per frame of different approaches at various pruning beamwidths (150, 160, 180 and 200). 88

A.1 Mel filter bank consisting of M triangular bandpass filters. f_m is the centre frequency of the m^{th} mel filter. $f_{Nyquist}$ is the Nyquist frequency. 95

List of Tables

3.1 Common semirings used in WFST. $\vee$ and $\wedge$ are *logical-or* and *logical-and* operators respectively. $\mathbb{R}$ denotes the real number set. $\oplus_{\log}$ is defined as $x \oplus_{\log} y = -\log(\exp(-x) + \exp(-y))$. 31

3.2 Knowledge sources in a typical ASR system. Negative log probabilities are used as weights if the semiring is log or tropical. 34

4.1 Fixed-point formats of various data types in the MFCC algorithm 40

4.2 Fixed-point formats of various data types in the GMM emission probability calculation 44

4.3 Fixed-point formats of $\mu_{jm}^{(d)}$ and $v_{jm}^{(d)}$ for each of the 39 dimensions. These formats are determined by examining all the HMM states trained using the RM1 corpus. 45

4.4 Fixed-point formats of various data types in the Viterbi search 47

4.5 Word accuracy rate (%) versus the two design variables, f and p, of the fixed-point implementation of Juicer. The word accuracy rate of the floating-point implementation is also shown. The test set consists of 1200 utterances from the RM1 corpus. 48

5.1 Timing profile of running the pure software-based speech recognizer on the Nios II platform. Both the floating-point and the fixed-point versions are considered. The speech utterance duration is 2.515s. 51

6.1 Timing profile of the pure software-based and the hardware-software co-processing speech recognizer. Data formats are fixed-point. The speech duration is 2.515s. 60

6.2 Resource usage of the GMM hardware accelerator. The device is Stratix II EP2S60F672C5ES FPGA. 61

6.3 Performance of recently developed embedded speech recognition systems and our proposed systems on the 993-word RM1 task 67

7.1 Maximum memory usage (in MB) during decoding 89

B.1 Values of $\sum_{m=0}^{M-1} |w_{m,d}|$ for $0 \leq d < D_{static}$. The other parameters include $D_{static} = 13$, $M = 26$ and $L = 22$. 98

Glossary

ALM	Adaptive logic module
ANN	Artificial neural network
API	Application programming interface
ASR	Automatic speech recognition
CD	context-dependent
CI	context-independent
CMN	Cepstral mean normalization
CMVN	Cepstral mean variance normalization
DTW	Dynamic time warping
EM	Expectation-maximization
FPGA	Field programmable gate arrays
GMM	Gaussian mixture model
HDL	Hardware description language
HMM	Hidden Markov model
IP	Intellectual property
LM	Language modelling
LPC	Linear predictive coding
MFCC	Mel frequency cepstral coefficient
MLP	Multi-layer perceptrons
PLP	Perceptual linear prediction
RTF	Real-time factor
SOPC	System-on-a-programmable-chip
VHDL	VHSIC hardware description language
VHSIC	Very-high-speed integrated circuits
WER	Word error rate
WFST	Weighted finite state transducer