A JOINT DECODING ALGORITHM FOR MULTIPLE-EXAMPLE-BASED ADDITION OF WORDS TO A PRONUNCIATION LEXICON

Dhananjay Bansal
Xtone Networks
Reston, VA, USA

Nishanth Nair
Indian Institute of Science
Bangalore, India

Rita Singh
HLTCoE, JHU
Baltimore, MD, USA

Bhiksha Raj
MERL
Cambridge, MA, USA

ABSTRACT

We propose an algorithm that enables *joint* Viterbi decoding of multiple independent audio recordings of a word to derive its pronunciation. Experiments show that this method results in better pronunciation estimation and word recognition accuracy than that obtained either with a single example of the word or using conventional approaches to pronunciation estimation using multiple examples.

Keywords: Speech recognition, Joint decoding, Pronunciation estimation.

1. INTRODUCTION

Speech recognition (ASR) systems are sometimes required to add words to their lexicon based only on audio recordings of the words, e.g. in name diallers or dictation systems where users may desire to add new names or keywords to the system by voice. Typically, the user records one or more instances of the word and expects the system to recognize them when they are uttered thereafter. The system must somehow learn a model for the new word, based only on these recordings.

In phoneme-based recognition systems this is typically done by learning the pronunciation of the word from the recorded examples [1]. When only one acoustic realization of a word is available, this is readily achieved by automatically recognizing the phoneme sequence in it. However, a pronunciation determined from only one recording of a word can be very unreliable. Pronunciations can be much more reliably estimated if there are multiple recorded instances of the word. To estimate its pronunciation one must now determine the phoneme sequence that best explains *all* recorded instances. This, however, poses a problem since the phoneme sequences recognized in the individual sequences may be different. A variety of approaches have been proposed to deal with this. A common technique is to derive pronunciations by voting amongst the recognition outputs from the individual recordings [2]. Another generates N-best hypotheses from each of the audio inputs and rescores the cumulative set jointly with all the recordings [3][4]. Other methods produce recognition lattices individually from each of the inputs, and identify the most likely path in the intersection of these lattices.

In this paper we note that the Bayesian principle behind most of these techniques is to determine the phoneme sequence that is maximally likely given all examples of the word. In HMM-based systems in particular, this would best be performed by *joint* decoding of all the examples. We therefore propose to derive the pronunciations of the word by joint decoding of all presented examples.

The conventional Viterbi decoding algorithm, however, is unsuited to performing such joint decoding since the multiple recordings are all *independent* observations, typically of different lengths, and have no temporal correspondence to one another, as we explain in Section 2.

We propose an algorithm, described in Sections 3-5, based on a technique presented in [5], that deals with this problem by enforcing a temporal correspondence through the introduction of a “ correspondence” variable. This recasts the otherwise-intractable problem of joint recognition as joint decoding given a correspondence between the optimal state sequences for the recordings. The correspondence itself may be derived through other criteria. Here we assume it is obtained through a DTW-based alignment algorithm.

Our experiments, described in Section 6 reveal that the pronunciations estimated using this approach are significantly superior to other methods of discovering pronunciations from audio examples of words, both in terms of the fraction of words for which pronunciations are correctly discovered, as well as in terms of the recognition accuracy obtained with discovered pronunciations.

2. JOINT RECOGNITION OF MULTIPLE INPUTS

The usual problem of automatic speech recognition is that of finding the sequence of terms[1] $W_A W_B \cdots W_K$ that were spoken in producing an acoustic signal $\mathbf{X}$. The Bayesian solution to the problem is given by:

$$\hat{W}_A \hat{W}_B \cdots = \mathrm{argmax}_{W_a W_b \cdots} P(\mathbf{X}|W_a W_b \cdots) P(W_a W_b \cdots) \tag{1}$$

where $\hat{W}_A \hat{W}_B \cdots$ is the recognized sequence of terms, $W_a W_b \cdots$ represents an arbitrary sequence of term, ands. $P(W_a W_b \cdots)$ represents the *a priori* probability of $W_a W_b \cdots$.

In practice, in HMM-based systems the probability distribution for any term sequence $P(\mathbf{X}|W_a W_b \cdots)$ is *composed* from HMMs $\mathcal{H}(W)$ representing the probability distributions of individual terms, $P(\mathbf{X}|W)$. Furthermore, the actual procdure for recognition does not explicitly enumerate the argument of Equation 1 for each term sequence as the computational expense can be prohibitive. Rather, an HMM $\mathcal{H}(G)$ is composed for a parsimoious term *graph* $\mathcal{G}$ that represents the set of all term sequences allowed by the probability distribution $P(W_a W_b \cdots)$, from the HMMs $\mathcal{H}(W)$ of the individual terms. Any sequence of states $\mathcal{S} = s_1, s_2, \cdots, s_T$ that is traversed in a walk through $\mathcal{H}(G)$ actually represents an underlying sequence of terms $\mathcal{W}(\mathcal{S}) = W_a, W_b, \cdots$ whose HMMs $\mathcal{H}(W_a), \mathcal{H}(W_b), \cdots$ are entered and exited in the process of traversing $\mathcal{S}$. Recognition is actually performed as:

$$\hat{\mathcal{S}} = \mathrm{argmax}_{\mathcal{S}} P(\mathbf{X}, \mathcal{S}|\mathcal{H}(G) \tag{2}$$

$$\hat{W}_A \hat{W}_B \cdots = \mathcal{W}(\hat{\mathcal{S}}) \tag{3}$$

[1]We refer to W_A, W_B etc as *terms* rather than *phonmes* or *words* since the discussion is applicable to recognition in general.

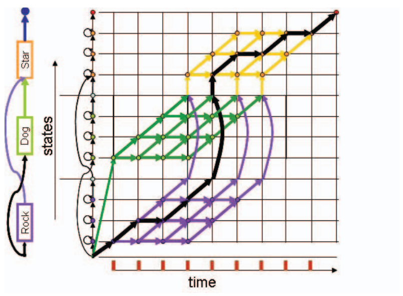

Fig. 1. The HMM on the left represents two word sequences "DOG STAR" and "ROCK STAR", and is composed from the HMMs for DOG, ROCK and STAR. The graph in the middle represents the trellis that must be searched to determine the best state sequence for the data sequence (bottom). The dark lines illustrate a state sequence that represents the word sequence "ROCK STAR".

Note that the recognition process does not directly determine the *term* sequence. Rather, it identifies the optimal *state* sequence $\hat{\mathcal{S}}$ through a trellis representing all possible state sequences that could be traversed by $\mathbf{X}$, and the term sequence is determined from this state sequence. This is illustrated by figure 1. Note too that the trellis is two-dimensional; the vertical axis represents HMM states and the horizantal axis represents time.

The problem of joint recognition of *multiple* audio inputs is defined as follows: given N independent recordings $\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_N$ that represent the same term sequence we aim to compute:

$$\hat{W}_A\hat{W}_B\cdots = \text{argmax}_{W_aW_b\cdots} P(\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_N | W_aW_b\cdots) P(W_aW_b\cdots) \quad (4)$$

For ease of presentation, in the following discussion we will assume

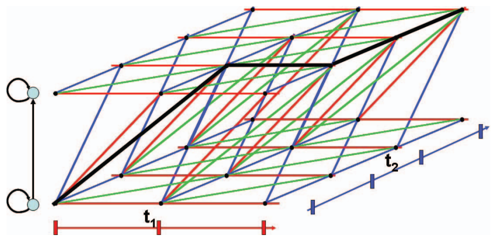

Fig. 2. A 3-dimensional trellis required for joint estimation of the optimal state sequence for two data sequences. The HMM is shown to the left, and the two data sequences with time axes t_1 and t_2 are shown in red and blue to the bottom and right. The red lines in the trellis represent transitions where there is state transition activity along the red data sequence only. The blue lines similarly represent transition activity for the blue data only. Green lines represent transitions occuring for both data sequences.

only two inputs $\mathbf{X}_1$ and $\mathbf{X}_2$; however it applies equally to larger collections of inputs. Within the HMM framework mentioned earlier, a naive translation of the problem in Equation 4 would be:

$$\hat{\mathcal{S}}_1\hat{\mathcal{S}}_2 = \text{argmax}_{\mathcal{S}_1\hat{\mathcal{S}}_2} P(\mathbf{X}_1\mathbf{X}_2, \mathcal{S}_1\mathcal{S}_2 | \mathcal{H}(G)) \quad (5)$$

This corresponds to finding the most likely *super-state sequence* $\hat{\mathbf{S}} = (\hat{\mathcal{S}}_1\hat{\mathcal{S}}_2)$ through a *three-dimensional* trellis[2], with one HMM-state axis and *two* time axes representing the temporal evolution of the two sequences $\mathbf{X}_1$ and $\mathbf{X}_2$. This is illustrated by figure 2.

There are several problems that arise in finding the optimal path through such a trellis, however.

- There is no reason to believe that $W(\hat{\mathcal{S}}_1) = W(\hat{\mathcal{S}}_2)$, i.e. that the two component state sequences $\hat{\mathcal{S}}_1$ and $\hat{\mathcal{S}}_2$ of $\hat{\mathbf{S}}$ represent the same term sequence. As a result, no unique term sequence can be derived from the outcome of the decoding process.
- Since the two data sequences are independent and accordingly have separate time axes, many of the transitions in the trellis represent state transition activity along one of the data sequences, but no activity at all (not even self transitions) along the other. Consequently, it is difficult to impose constriants that would restrict $\hat{\mathbf{S}}$ such that $\hat{\mathcal{S}}_1$ and $\hat{\mathcal{S}}_2$ represent the same term sequence.
- The computational complexity of the search increases exponentially with the number of data sequences being jointly decoded.

The problem of joint decoding of multiple data sequences representing the same term sequence is not new: it is also encountered when performing recognition with simultaneous audio-visual data [7], multi-band recognition [8] etc. However, in these problems there is strict temporal correspondence between the multiple data streams since they represent multiple observations of the same event (utterance). This is utilized to set the feature streams for the multiple observations to have the same number of vectors with one-to-one correspondence between vectors from different streams. Even when the data are not naturally thus synchronized, such as for audio-visual data, they are resampled such that they do [7]. Once such correspondence is established, the problem reduces to search over a simple 2-dimensional manifold (along a digonal) within the 3-dimensional (or more generally N+1-dimensional) trellis. It becomes relatively simple to impose constraints that ensure that all component state sequences represent the same term sequence, and any increase in complexity is merely the result of increase in the number of states in the HMM. Unfortunately, in our problem, since the data sequences are completely indepdenent no such correspondence exists and an alternate strategy must be devised.

3. THE JOINT RECOGNITION ALGORITHM

The difficulty of joint recognition of multiple independent data sequences arises primarily from the fact that no correspondence exists between the time axes of the sequences. We remedy this problem by introducing a *correspondence* variable $\mathcal{C}$ that identifies lower-dimensional manifolds within the 3-dimensional (N+1-dimensional) trellis such that the components state sequences $\mathcal{S}_i$ of any super-state sequences $\mathbf{S}$ within the manifold are *consistent*, i.e. they represent the same term sequence. In other words, for any $\mathbf{S} = (\mathcal{S}_1, \mathcal{S}_2)$ that lies within a manifold, $W(\mathcal{S}_i) = W(\mathcal{S}_j) \forall i, j$. A *correspondence-specific* decoding algorithm finds the most likely term sequence given the correspondence :

$$\hat{\mathcal{S}}_1\hat{\mathcal{S}}_2 = \text{argmax}_{\mathcal{S}_1\hat{\mathcal{S}}_2} P(\mathbf{X}_1\mathbf{X}_2, \mathcal{S}_1\mathcal{S}_2 | \mathcal{C}\mathcal{H}(G)) \quad (6)$$

$$\hat{W}_A\hat{W}_B\cdots = \mathcal{W}(\hat{\mathcal{S}}_i) \quad (7)$$

Since $\hat{\mathcal{S}}_1$ and $\hat{\mathcal{S}}_2$ are guaranteed to be consistent, either of them may be employed in Equation 7.

[2]A given pair of state sequences $(\mathcal{S}_1\mathcal{S}_2)$ can be obtained from multiple super-state sequences. However all of them are equivalent in that they also have the same likelihood and we will not distinguish between them.

We define the globally optimal super-state sequence $\hat{\mathbf{S}}$ as the most likely super-state sequence $\mathbf{S}$ that also conforms to the condition that all component state sequences $\mathcal{S}_i$ represent the same term sequence. Our eventual goal is to find $\hat{\mathbf{S}}$. In general the global optimum is not guaranteed to lie within any correspondence. However, if the HMMs for the terms are have strictly left-to-right (Bakis) topology with no skipping of states permitted, a common model for phonemes in phoneme-based recognition systems, then it is easy to show that the globally optimal solution is also the correspondence-specific solution for a correspondence $\mathcal{C}$ given by:

$$\mathcal{C} = \{\mathbf{X}_1^i \mapsto \mathbf{X}_2^j | i \in \mathcal{I}_k, j \in \mathcal{J}_k, k \in \{1 \cdots L\}\} \tag{8}$$

where $\mathbf{X}_1^i \mapsto \mathbf{X}_2^j$ indicates that the i^{th} vector of $\mathbf{X}_1$ and the j^{th} vector of $\mathbf{X}_2$ are required to be at the same HMM state, $\mathcal{I}_k$ represents a sequence of indices such that $min(i|i \in \mathcal{I}_k) = max(i|i \in \mathcal{I}_{k-1} + 1)$ and $max(i|i \in \mathcal{I}_L) = T_1$, where T_1 is the number of vectors in $\mathbf{X}_1$. $\mathcal{J}_k$ represents similar sequences of indices such that $max(\mathcal{J}_L) = T_2$. An example of such a correspondence is illustrated by figure 3. Although not every correspondence of the form in Equation 8 is guaranteed to contain the global optimum, given the correct correspondence $\hat{\mathcal{C}}$ the global optimum can be obtained simply as a correspondence-specific optimum.

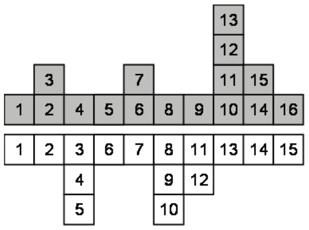

Fig. 3. Illustrating the correspondence of two observation sequences $\mathbf{X}_1$, represented by the gray blocks, and $\mathbf{X}_2$ represented by the white blocks. The alignments are $\mathbf{X}_1^1 \mapsto \mathbf{X}_2^1$; $\{\mathbf{X}_1^2, \mathbf{X}_1^3\} \mapsto \mathbf{X}_2^2$; $\mathbf{X}_1^4 \mapsto \{\mathbf{X}_2^3, \mathbf{X}_2^4, \mathbf{X}_2^5\}$ and so on.

Given a correspondence $\mathcal{C}$ such as the one in Equation 8, the joint likelihood of $\mathbf{X}_1$, $\mathbf{X}_2$ and a super-state sequence $\mathbf{S}$ is:

$$P(\mathbf{X}_1, \mathbf{X}_2, \mathbf{S}|\mathcal{C}) = P(\mathbf{S}|\mathcal{C}) \prod_{l=1}^{L} \prod_{i \in \mathcal{I}_l} P(X_1^i|s_l) \prod_{j \in \mathcal{J}_m} P(X_2^j|s_m) \tag{9}$$

where s_l is the state that all observations $X_1^i | i \in \mathcal{I}_l$ must lie in, and s_m is similarly defined for X_2 and $\mathcal{J}_m$. $P(\mathbf{S}|\mathcal{C})$ is given by

$$P(\mathbf{S}|\mathcal{C}) = \pi^2(s_1) \prod_{l=1}^{L} P(s_l|s_l)^{|\mathcal{I}_l|+|\mathcal{J}_l|-2} \prod_{l=1}^{L-1} P^2(\bar{s}_{l+1}|\bar{s}_l) \tag{10}$$

where $\pi_(s)$ is the *a priori* probability of HMM state s and $P(s_i|s_j)$ is the HMM transition probability between states s_i and s_j.

It is relatively straightforward to demonstrate that Equation 9 can be computed dynamically using the Viterbi algorithm, simply by computing the state output probability of any state s as $\prod_{i \in \mathcal{I}_l} P(X_1^i|s)$ $\prod_{j \in \mathcal{J}_l} P(X_2^j|s)$ $P(s|s)^{|\mathcal{I}_l|+|\mathcal{J}_l|-2}$, and raising the the state initial and transition probabilities to the second power (or, more generically for N data sequences, to the N^{th} power).

3.1. Determining the optimal Correspondence

Although an optimal correspondence (i.e. one that contains the optimal state sequence) cannot be known *a priori*, a reasonable hypothesis is that such an alignment will roughly follow the alignment of the data sequences $\mathbf{X}_1$ and $\mathbf{X}_2$ themselves. We therefore estimate the correspondence $\hat{\mathcal{C}}$ by aligning the two data sequences via dynamic time warping [6].

4. DETERMINING WORD PRONUNCIATIONS

We are now set to determine the pronunciation of a word from multiple acoustic examples. We align all inputs to generate a correspondence $\mathcal{C}$. We then perform a correspondence-specific joint phoneme decode of all recordings using Equation 9, where terms W are phonemes. The *a priori* probability of phoneme sequences $P(W_a W_b \cdots)$ is provided by a phoneme N-gram language model in the work reported here. However, it may also be provided by phoneme graphs derived from spellings or by any other means.

5. EXPERIMENTAL EVALUATION

We evaluated the proposed joint-decoding algorithm for pronunciation estimation using Carnegie Mellon University's Sphinx-2 semi-continuous density HMM-based speech recognition engine. The system was modified to perform correspondence-constrained decoding on multiple inputs as mentioned in section 3. The acoustic models for speech recognition system comprised 3000 tied states trained from a corpus 110 hours of read English utterances, mostly proper nouns, collected over the telephone channel (8KHz, 8 bit, ulaw compressed) with different phone lines, speakers, and acoustic environments. A held out "pronunciation learning set" comprising 1679 phrases including 1215 unique proper nouns (”subject terms") each having multiple examples, was used to learn pronunications. We note that this is a difficult corpus where the baseline recognition accuracy on our held-out test set using manually-constructed dictionary pronunciations is only 83.8%.

In the first experiment we assumed that the pronunciations of all 1215 words were unknown and estimated them using our joint-decoding algorithm. A trigram model learned from the pronunciations of 110,000 words (including a large number of proper nouns) was used as the phoneme language model (LM) for recognition. Figure 4 shows the "correctness" of the estimated pronunciations as a function of the number of recordings used to estimate them, using two different criteria. The *Auto Pron Accuracy* is the pronunciation match accuracy (*i.e.* phoneme string accuracy) of the automatically generated pronunciation w.r.t. the pronunciations in a hand-crafted reference dictionary for the words. We expect a better algorithm to provide a higher match to the reference. *Recognition accuracy* shows the recognition accuracy on a held-out test set (not used to estimate pronunciations) using the estimated pronunciations. We note from figure 4 that the *Auto Pron Accuracy and* Recognition Accuracy both increase as we increase number of examples. Also, there is a significant correlation between *Auto Pron Accuracy and* Recognition Accuracy.

We compared the pronunciations obtained with our joint-decoding algorithm with those obtained from two other commonly used algorithms for combining evidence from multiple inputs. In the first, which we call *Nbest rescoring*, N-best hypotheses from each of the audio samples were generated and the cumulative set of N-best pronunciations rescored jointly with all the recordings to find the most

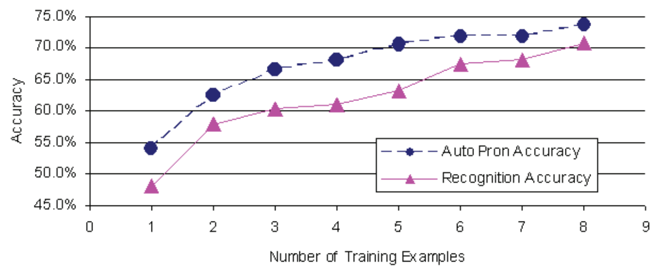

Fig. 4. Match accuracy of automatically generated pronunciation to correct orthography, and recognition accuracy over the test set using automatic pronunciation.

Method	Auto Pron Accuracy	Recognition Accuracy
Nbest Rescoring	68.1%	66.4%
Voting	61.6%	54.6%
Joint Decoding	71.1%	68.2%

Table 1. Comparison of proposed method to voting and Nbest rescoring on automatic pronunciation generation task.

likely pronunciation. For sufficiently large N, this method approximates joint decoding using all examples. In the second, *voting*, each example was decoded independently and the answer that occurred most frequently among all examples was chosen as the winner. When there were multiple winners, one of the winners was chosen at random. Six repetitions of each word were used for this experiment. Table 1 shows the comparison. We note that the joint-decoding scheme performs significantly better than both Nbest rescoring and much better than voting.

The joint decoding algorithm is generic and can be used not only for pronunciation generation, but can also be used for generic phrase recognition when repetitions of the same phrase are available. In a second experiment we evaluated the effect of using multiple examples on phrase recognition accuracy. In this experiment we used the hand-crafted dictionary to provide the pronunciations of all words, and used a large finite-state grammar that included all phrases as our language model. Figure 5 shows the results. We note that the recognition accuracy significantly improves with addition of more examples. This result has significant implications in improving recognition accuracy for tasks in which multiple examples of a phrase are available such as broadcast news, voice dialer etc. Table 2 compares the performance of the proposed joint-decoding algorithm to voting and Nbest rescoring on the phrase recognition task. Once again we note that joint decoding outperforms the other two algorithms. Interestingly, voting is noted to be superior to Nbest rescoring on this task, whereas the latter was better for pronunciation estimation.

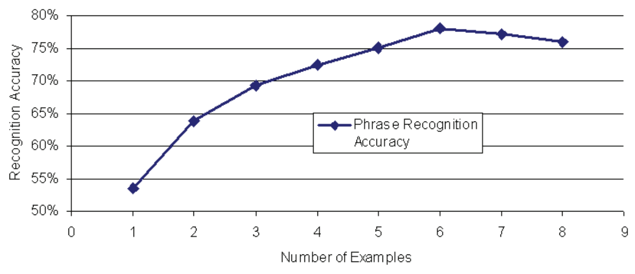

Fig. 5. Phrase accuracy Vs number of examples. Estimated over the pronunciation learning set.

Method	Phrase Recognition Accuracy
Nbest rescoring	69.3%
Voting	77.0%
Joint decoding	78.0%

Table 2. Comparison of proposed method to voting and Nbest rescoring on phrase recognition.

6. DISCUSSION

Our results show that the proposed joint decoding approach is much more effective at learning the pronunciations of words than other current approaches to recognition with multiple recorded examples. The result from Figure 5 indicates that the approach may also be effective for more generic recognition from multiple recordings.

The proposed algorithm is found to outperform both voting and Nbest rescoring. The latter in particular approximates joint decoding, yet the optimality of the proposed joint decoding algorithm reulsts in better recognition.

Much room for improvement remains. The proposed algorithm is only effective when the multiple recordings repersent *exactly* the same word sequence, and would be confounded by differences in the location of pauses or pronunciation variations in the repetitions. It is not clear how to specify correspondences in this scenario. Furthermore, our current approach to estimating the optimal correspondence is heuristic. Future research will address these issues.

7. REFERENCES

[1] Schmidt, Pl, Cole, R., Fanty, M. "Automatically generated word pronunciations from phoneme classifier output," Proc. IEEE Intl. Conf. on Acoustic Speech and Signal Processing (ICASSP), 1993.

[2] Fiscus, J.G., "A Post-Processing System to Yield Reduced Word Error Rates: Recogniser Output Voting Error Reduction (ROVER)," Proc. IEEE ASRU Workshop, Santa Barbara, 1997.

[3] Singh, R., Raj, B., Stern, R. M., "Automatic Generation of Subword Units for Speech Recognition Systems," IEEE Trans. on Speech and Audio Processing, Vol. 10(2), 89-99, 2002.

[4] Svendsen, T., "Pronunciation modeling for speech technology," Proc. Intl. Conf. on Signal Processing and Communication (SPCOM04), Dec, 2004.

[5] Nair, N., Sreenivas, T.V.S., "Joint Decoding of Multiple Speech Patterns for Robust Speech Recognition", Proc. IEEE ASRU Workshop, Kyoto, 2007.

[6] Myers, C.S., Rabiner, L.R. "A comparative study of several dynamic time-warping algorithms for connected word recognition," The Bell System Technical Journal, 60(7):1389-1409, September 1981.

[7] Saenko, K., Livescu, K., "An Asynchronous DBN for Audio-Visual Speech Recognition," in Proc. IEEE Workshop on Spoken Language Technologies, December 2006.

[8] Mirghafoni, N., "A Multi-Band Approach to Automatic Speech Recognition", ICSI Technical Report tr-99-004, 1999.