General Software Libraries – Text and Speech Processing

- **FSM Library** (Finite-State-Machine Library), general-purpose software tools for building, combining, optimizing, and searching weighted finite-state acceptors and transducers.

  http://www.research.att.com/sw/tools/fsm/

- **GRM Library** (Grammar Library), general-purpose software tools for constructing, modifying, and compiling grammars.

  http://www.research.att.com/sw/tools/grm/

- **DCD Library** (Decoder Library), software collection for speech recognition decoding and related functions.

  http://www.research.att.com/sw/tools/dcd/

- **AMTOOLS** (Acoustic Modeling Tools), library for building acoustic models, probabilistic models of speech signals, for automatic speech recognition and related applications.

Infrastructure for Processing Uncertain Data

Mehryar Mohri

AT&T Labs - Research

mohri@research.att.com

General Ideas

- More data.
- Better learning techniques.
- Other information sources.

Problem: Data is costly or uncertain

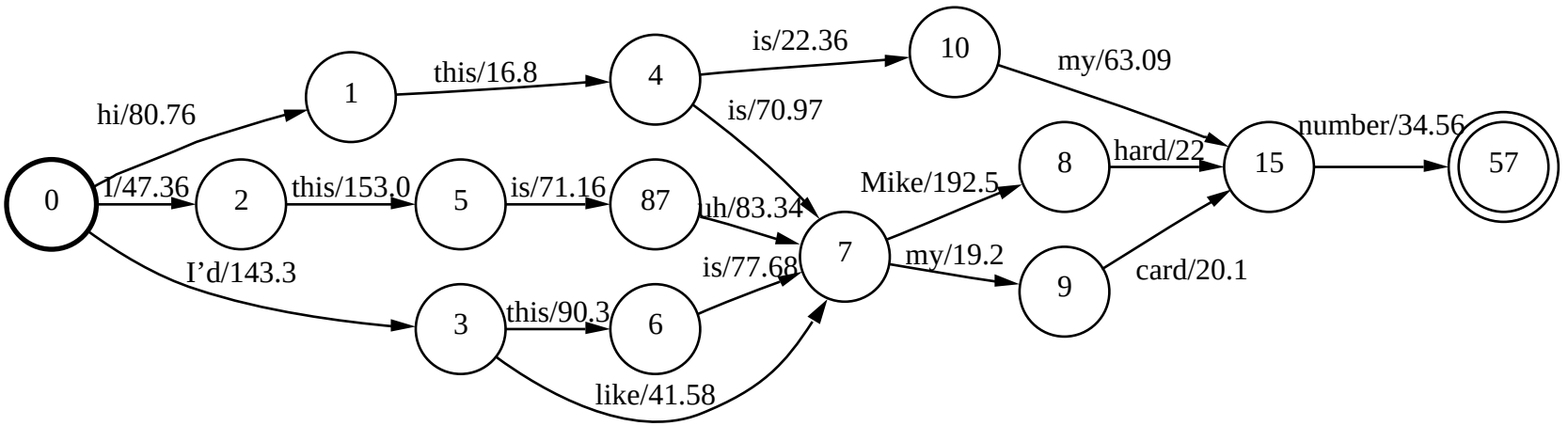
- **Bootstrapping**
  - Large quantities of untranscribed or poorly labeled speech data available (e.g., hours of broadcast programs).
  - Use recognition outputs (lattices) to improve accuracy.
- **Adaptation**
  - Data available for some tasks.
  - Adapt language models and acoustic models to new task (use lattices).
- **Other similar problems**
  - Spoken-Dialog Classification.
  - Speech mining.
  - Information extraction.

$\rightarrow$ **Algorithms for building models from lattices.**

This talk

- **Algorithm**: counting/collecting statistics from lattices.
- **Software Library**: GRM Library
  - brief overview.
  - utilities for building language models based on lattices.
- **Applications**:
  - adaptation.
  - voice signatures (detection of emotion using lattice kernels).

Output of a Speech Recognizer – Phone/Word Lattice



Count of Sequence x in Weighted Automaton A

$|u|_x$: number of occurrences of x in string u.

- **Expected count** of the sequence x in weighted automaton A:

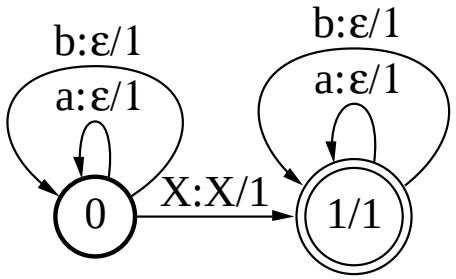
$$c(x) = \sum_{u \in \Sigma^*} |u|_x \, [\![A]\!](u)$$

- m**-th moment of the count** of the sequence x in A, $m \geq 1$:

$$c_m(x) = \sum_{u \in \Sigma^*} |u|_x^m \, [\![A]\!](u)$$

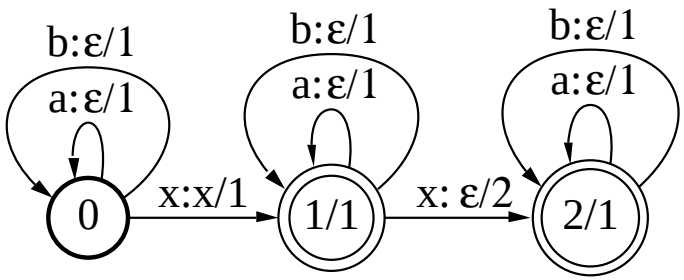
Expected Counts – Algorithm

- **Problem**
  - Regular expression X, weighted automaton A.
  - Expected counts of $x \in L(X)$ in A?
- **Simple algorithm**
  - Transducer T_X.

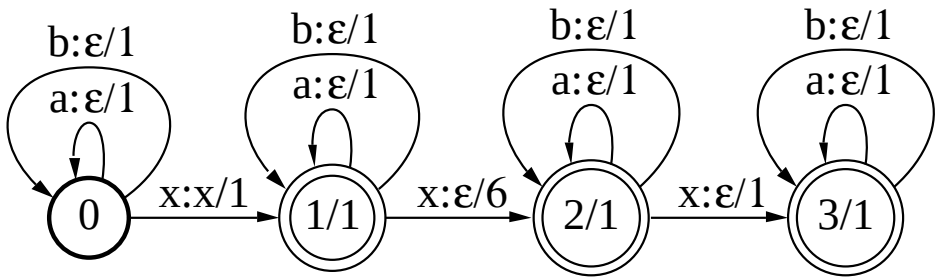


  - Composition with weighted transducer T_X and projection: $B = \Pi_2(A \circ T_X)$, complexity: $O(|A||X|)$.
  - Shortest-distance algorithm: $c(x) = [\![B]\!](x)$.

Second and Third Moments



Weighted transducer T_2 for computing the second moment.



Weighted transducer T_3 for computing the third moment.

GRM Library: Content

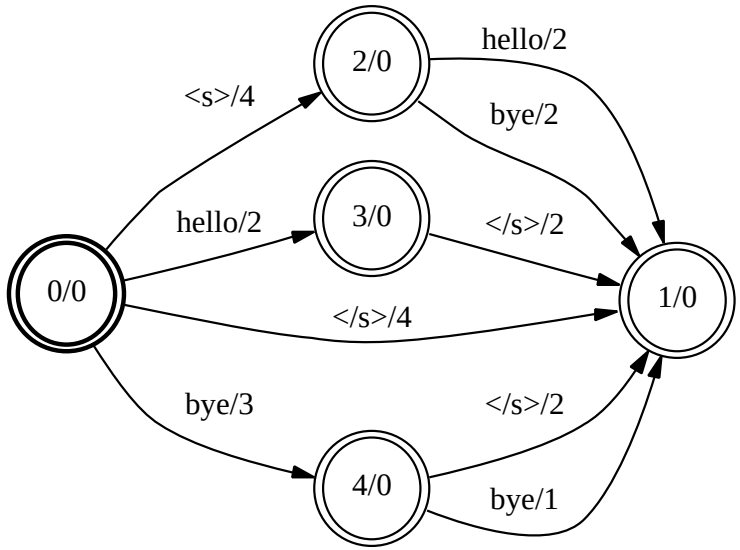
1. **Grammar compilation into weighted FSMs**
   (a) Weighted context-dependent rules into weighted transducers
   (b) Weighted context-free grammars into weighted automata
   (c) Regular approximations of weighted context-free grammars
   (d) Replace class (mutate, substitute)
2. **Text and Grammar Processing Utilities**
   (a) Failure class
   (b) Local grammars
   (c) Suffix Automata
   (d) Local determinization
   (e) Counting and merging counts
3. **Statistical language modeling**
   (a) Making, shrinking and converting language models

Counting from weighted automata

- **Programs:**

```
grmcount -n2 -s1 -f2 foo.far > foo.2g.counts.fsm
grmmerge foo.counts.fsm bar.counts.fsm > foobar.counts.fsm
```

- **Graphical Representation:**

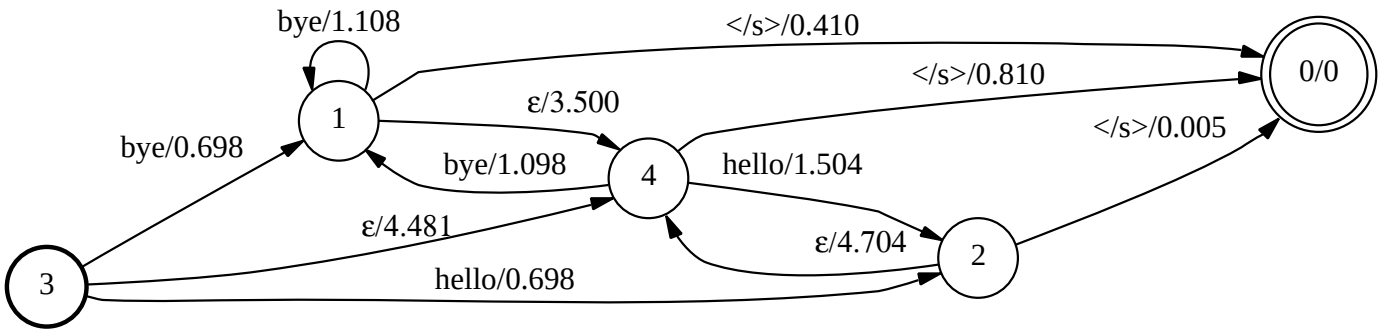


Creating a backoff model from counts

- **Program:**

```
grmmake foo.2g.counts.fsm > foo.2g.lm.fsm
```

- **Graphical Representation:**

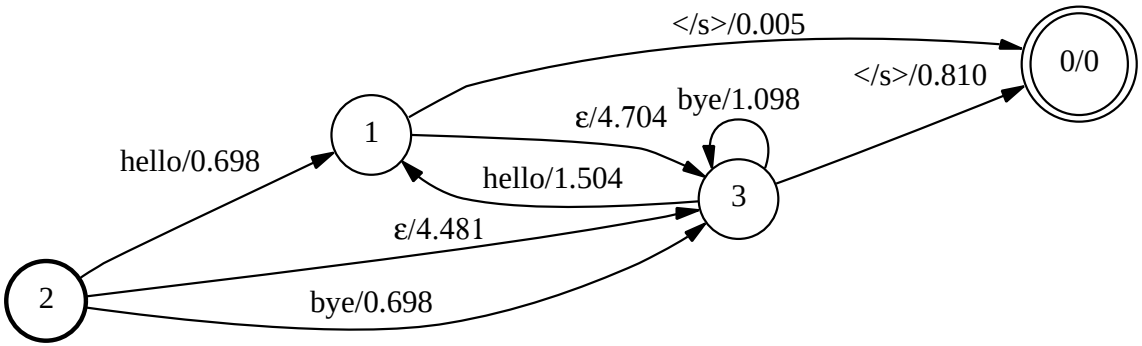


Shrinking a backoff model

- **Program:**

```
grmshrink -s4 foo.2g.lm.fsm > foo.2g.s4.lm.fsm
```

- **Graphical Representation:**



Application – Language Modeling

(GRM Library, joint work with Cyril Allauzen and Brian Roark)

- **Language model adaptation**: built on weighted automata output by recognizer.
- **Utility**: `grmcount` (GRM Library).
- **Experiments**
  - compute expected counts of all n-gram sequences, $n \leq 3$.
  - 41,000 weighted automata outputs of speech recognizer, 18.8M transitions.
  - 1h52m on a single processor of 1GHz Intel Pentium Linux cluster.
  - $\frac{1}{60}$*th* of total duration of the speech utterances.

Application – Voice Signatures

(joint work with Itzhak Shafran and Michael Riley)

Classification of emotion, AT&T HMIHY 0300 spoken-dialog system.

- **Categories**: originally seven categories (positive-neutral, very-angry, somewhat-angry, very-frustrated, somewhat-frustrated, other-negative).

  Binary case: negative vs. non-negative.

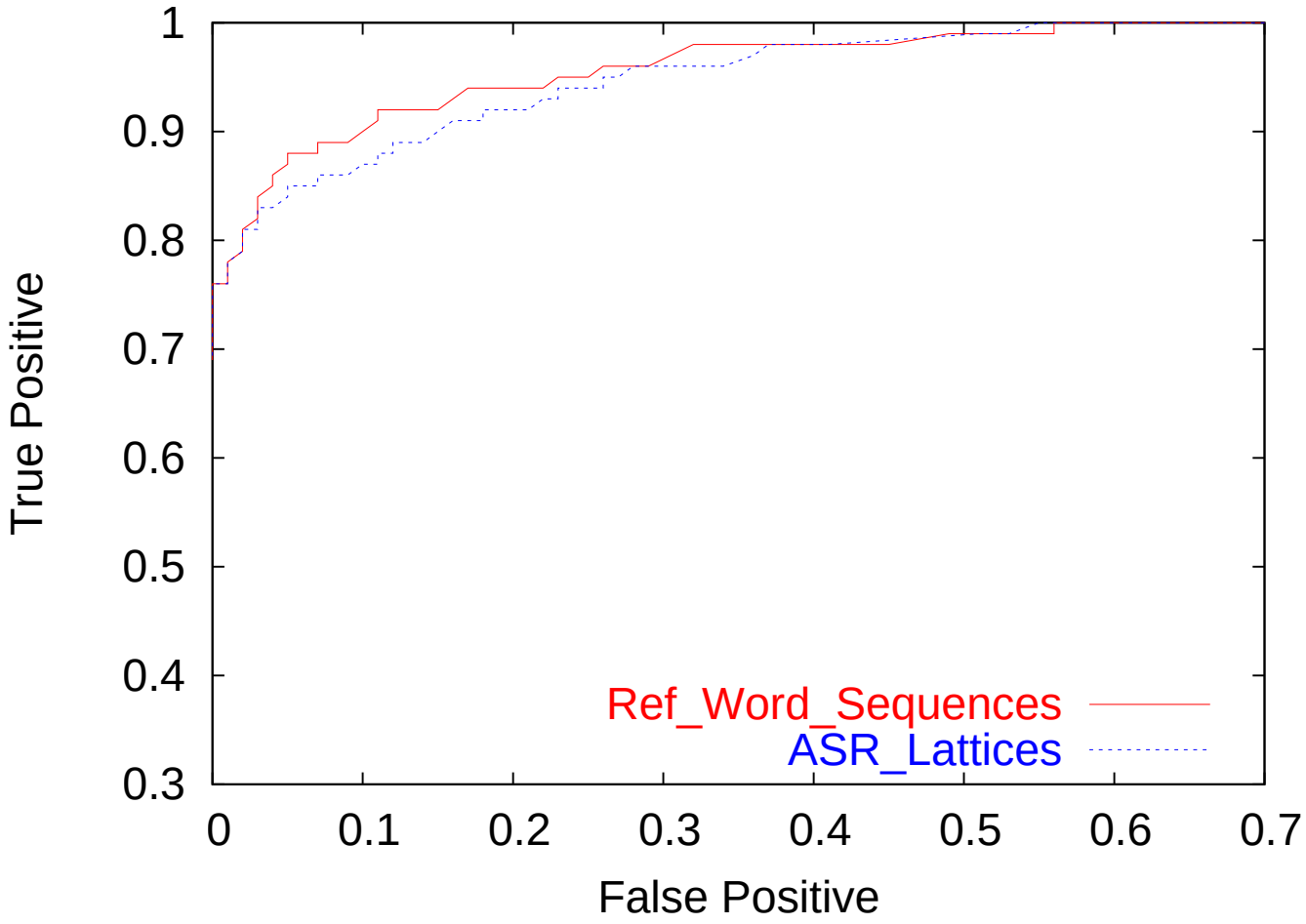
- **Test Data**: about 650 word lattices.

- **Rational Kernels**: n-gram kernels with $n = 4$.

Detection of Emotion – Experimental Results

HMM-based classifiers on Cepstra	76.8
SVM w/ rational kernels on ref. word transcripts	81.7
SVM w/ rational kernels on ASR word lattices	80.6

Comparison of two classifiers on the task of binary emotion classification.



Binary classification (negative emotion vs. other): percentage of true positive versus false positive.

Conclusion

- **Algorithms**: Collecting statistics, classification, clustering, etc.
- **Software Libraries**: Implementations publicly available – DCD Library, FSM library, GRM Library.
- **Coverage**: speech recognition, speech synthesis, spoken-dialog applications, classification, clustering, etc.
- **Generality**: algorithms used in other domains, e.g., computational biology, information extraction, network optimizations.