An Efficient Graph Search Decoder for Phrase-Based Statistical Machine Translation

Brian Delaney, Wade Shen, and Timothy Anderson

28 November 2006

Introduction

- **Efficient search remains an important goal for practical implementations of statistical machine translation**

- **Our goals were to create a decoder that:**
  - **Can be used in “real-time” speech translation**
  - **Can handle large vocabulary tasks at or near real-time**
  - **Enables easy integration with other speech components (ASR, TTS, etc.)**

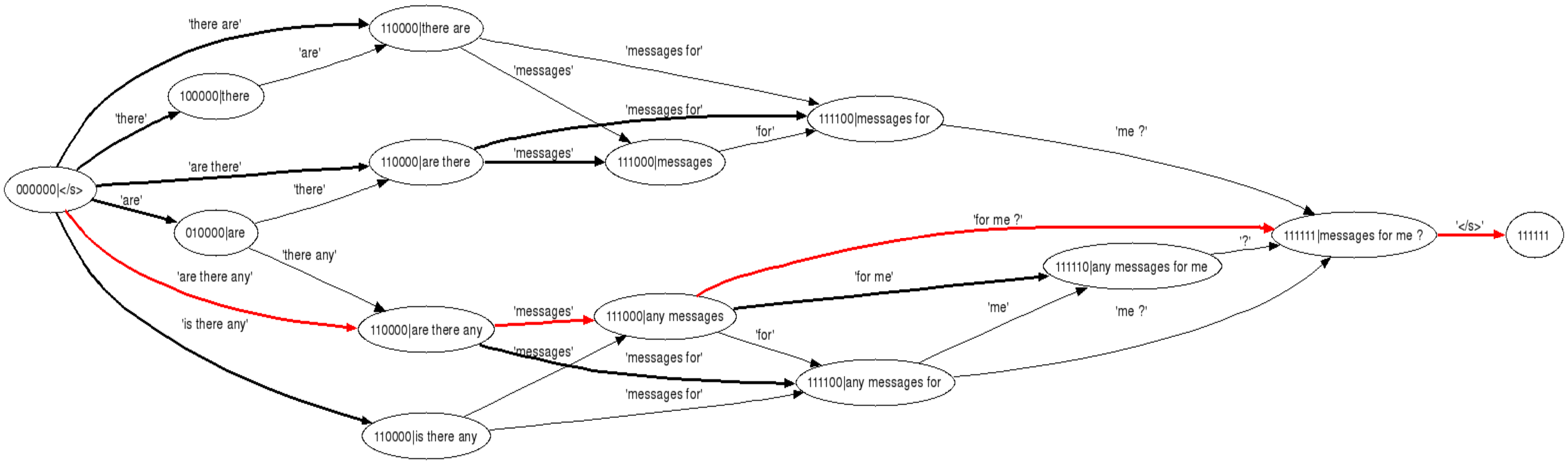
- **Overview**
  - **Our implementation of a graph search decoder**
  - **Analysis of performance on the IWSLT-06 task**

Decoder Highlights

- The basics
  - Uses phrase-based models with log-linear parameter combination
  - A-star graph search with beam and histogram pruning

- New features
  - Decoding with up to 5-gram language model
  - Output phrase lattice for optimization and rescoring
  - On-demand disk-based models for decoding of large vocabulary speech input in real-time
  - Reordering constraints for improved speed
  - Galaxy Communicator API to interface with other speech components (i.e. ASR, TTS, Language ID, etc.)

Decoding Algorithm



Ci sono messaggi per me ? → Are there any messages for me ?

- **Start state: No source words covered**
- **Select source/target phrase pairs from phrase table**
- **Expand nodes according to source coverage and LM context, using LM back-off structure**
- **Keep best path back pointer**
- **Back-trace along best path for 1-best result**

Pruning and A-star Heuristic

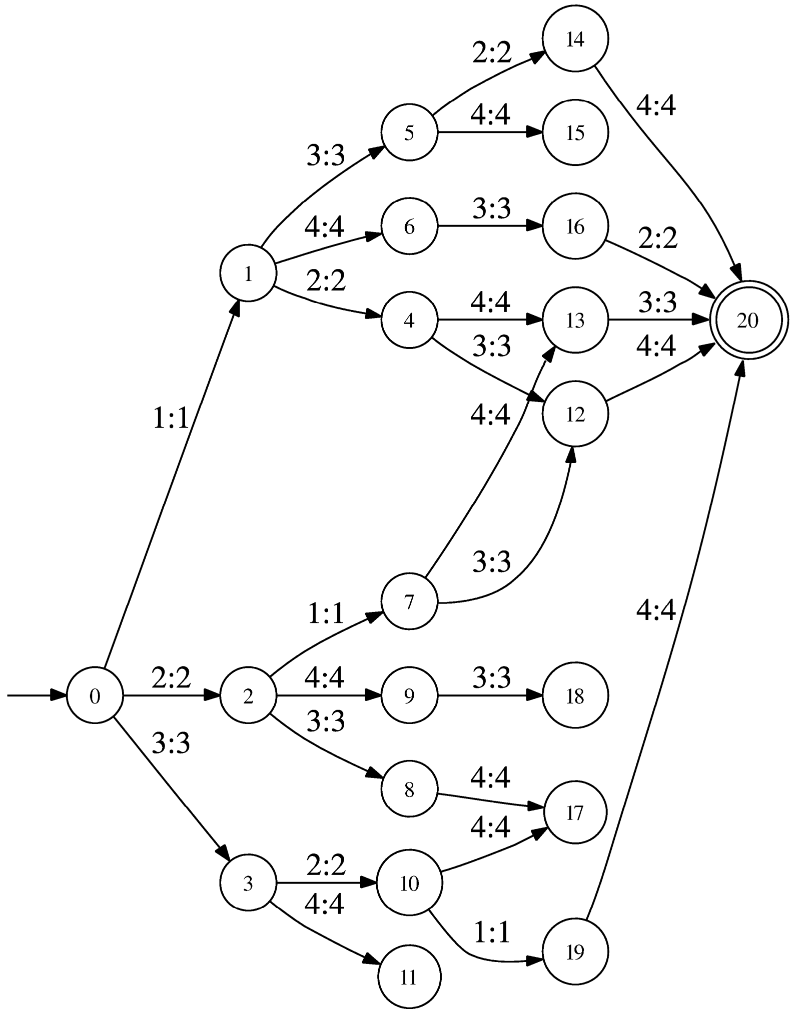
- **Standard beam and histogram pruning using best path score into each node**
- **All nodes that cover the same *number* of words are pruned together**
- **Because of distortion, “easy” words tend to get translated first**
  - **Need an estimate of future cost (A-star heuristic)**
- **Heuristic is based on words not yet translated**
  - **Same as with Pharaoh**
  - **Tried several enhancements to the Pharaoh:**
    - ***Best case distortion for next phrase***
    - ***Best/average language model expansion using current node context***
  - **Neither gave consistent improvement in accuracy or speed**

On-The-Fly Beam Pruning

- **Profiling revealed that computing language model scores at phrase boundaries is costly**
  - This is done when considering a new hypothesis
  - Most of these hypotheses get pruned out immediately
- **Solution**
  - Keep track of best path cost during search loop
  - Skip translations whose partial scores (i.e. without language model) fall outside the beam
- **Results in almost 2x speedup with a very little change in BLEU**
- **Sorting list of translations options upfront by the *best* future cost helps to find best translation faster**
  - results in faster search

Phrase Reordering (1)

- To allow word movement, source words may be translated in any order
- Without any constraints, the search grows exponentially with sentence length
- Limiting word movement by some maximum helps reduce complexity
- Incomplete paths can occur, resulting in wasted search effort



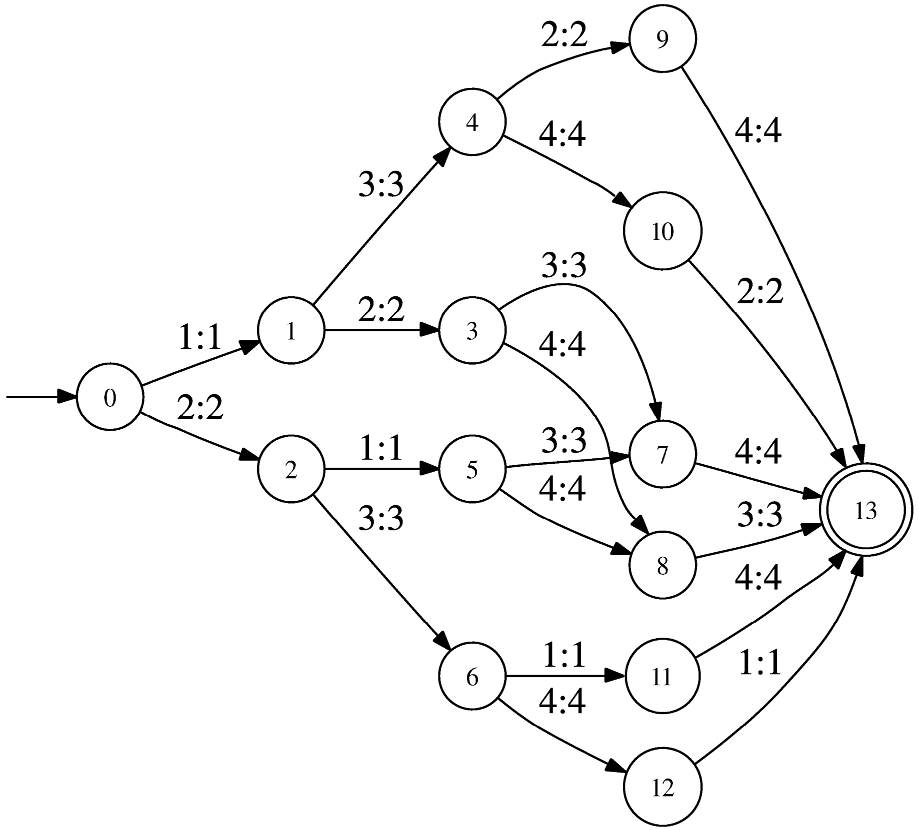
Reordering graph for 4 input words with dlimit=2

Phrase Reordering (2)

- **Additional reordering constraints (Zens 03)**
  - **IBM:** ***only choose words or phrases that fill the first k unfilled words***
  - **ITG:** ***do not allow "inside out" reordering patterns***

- **ITG + distortion limit can produce graph with incomplete paths**

- **IBM constraints do not have this problem**



Reordering graph for 4 input words with IBM constraints (k=2)

Phrase Reordering (3)

- We implemented an additional reordering constraint that allows for fast decoding with reasonable accuracy
  - Choose a new phrase that covers some portion of the first available gap
  - any new gaps must be less than the allowed distortion limit
- Not strictly a phrase swap and more constrained than IBM
- Results in fast decoding with good accuracy
  - Ideal for real-time speech translation

Results (1)

Configuration	Language Pair					
	CE		JE		IE	
	BLEU	Chars/sec	BLEU	Words/sec	BLEU	Words/sec
Pharaoh	20.41	0.85	23.07	1.39	35.63	55.48
free-3g	20.19	2.99	22.79	5.39	35.90	113.36
free-4g	20.73	1.45	21.76	2.26	35.64	63.06
free-5g	20.39	1.23	21.99	1.65	36.92	42.93
IBM-3g	20.31	3.70	22.55	6.14	36.60	201.05
IBM-4g	20.15	0.92	21.64	2.04	36.77	124.09
IBM-5g	20.29	0.66	23.04	2.05	36.56	81.15
ITG-3g	20.18	4.36	21.99	7.01	35.70	162.99
ITG-4g	18.89	1.04	22.56	3.50	36.81	60.99
ITG-5g	20.31	1.11	22.39	2.38	36.78	48.45
NEW-3g	19.10	8.52	**23.23**	12.72	36.56	305.29
NEW-4g	20.38	1.70	22.03	5.29	**36.96**	216.92
NEW-5g	**20.90**	1.54	22.81	4.36	36.66	142.47

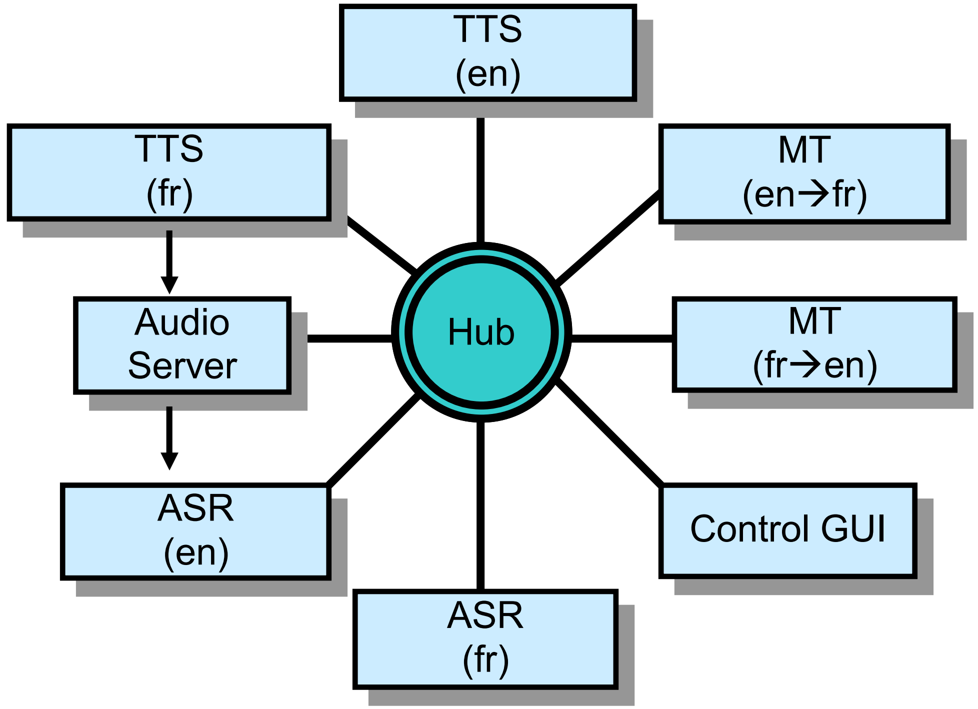
Results (2)

- Scores are similar to Pharaoh with some speed advantage
  - 2-4 times faster in base configuration
- Increased n-gram order didn't always improve score
  - Largest decrease in speed between 3-gram and 4-gram
- Proposed reordering constraints result in good scores with fastest decoding times
- It is difficult to pick a winner out of the IBM or ITG constraints with respect to speed or accuracy

Real-Time Speech Translation System

- **Use Galaxy Communicator Architecture as a common API to a variety of speech components**
  - **TTS: *AT&T, Delta Electronics, Festival, Cepstral***
  - **ASR: *MIT-LL, SONIC, Nuance***
  - **MT: *MIT-LL***

- **Runs large vocab English ↔ Spanish task (Europarl) on a single laptop**



Conclusion and Future Work

- Lessons learned
  - Fast decoding requires effective handling of reordering, either through better modeling and/or constraints
  - Prune the search graph early and often for maximum speed
  - “Real” systems require fast access to very large models;

    Berkeley DB makes this simple

- Future Work
  - Better reordering models (lexicalized or factored)
  - Additional language model support

    Class n-gram, large LMs (e.g. google n-gram), etc.