

REMEDI: Robust and Efficient Machine Translation in a Distributed Infrastructure

Bi-Annual Report: Month 6

Dr. Ivan S. Zapreev • UvA • ivan.zapreev@uva.nl
Dr. Christof Monz • UvA • c.monz@uva.nl

Table of Contents

- Introduction
- Deliverable
- Software Info
- Implementation
- Experiments
- Conclusions
- Future plans



Bios

- About:

- 35 years old
- Russian
- Married
- Mathematician (Bsc, Msc)
- PhD in Theoretical Computer Science
- 15 Years work experience (7 researcher)

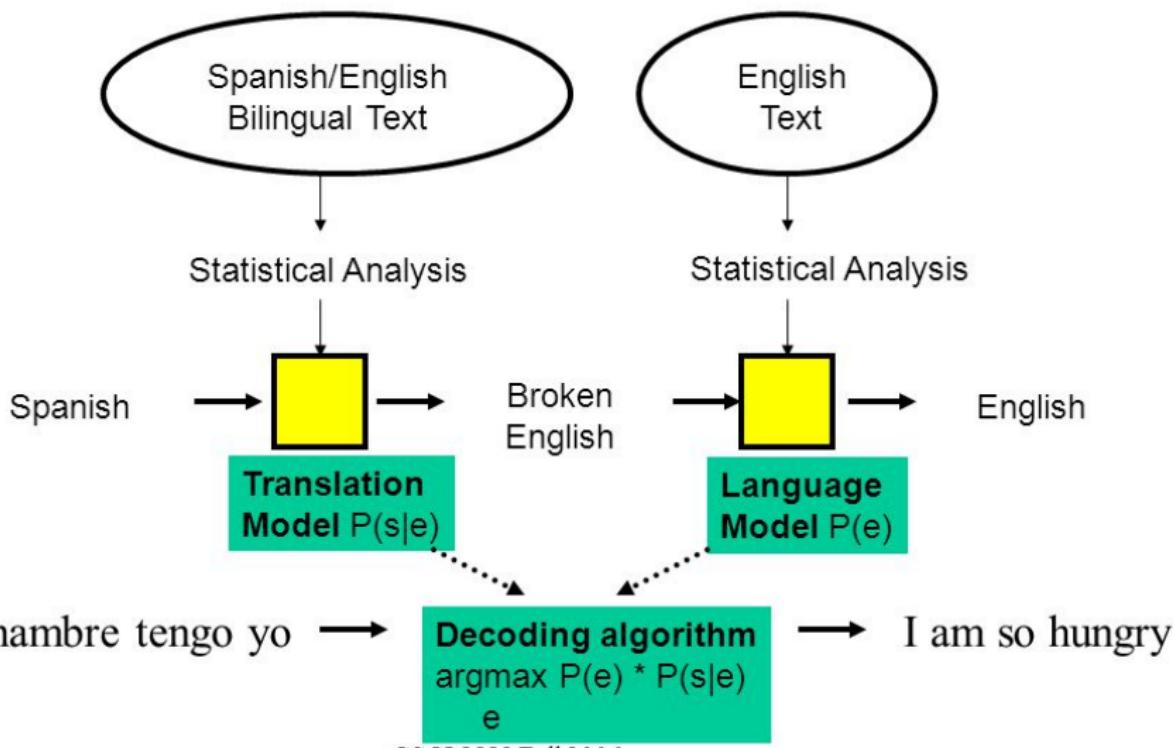


Address speed, scalability and robustness
via

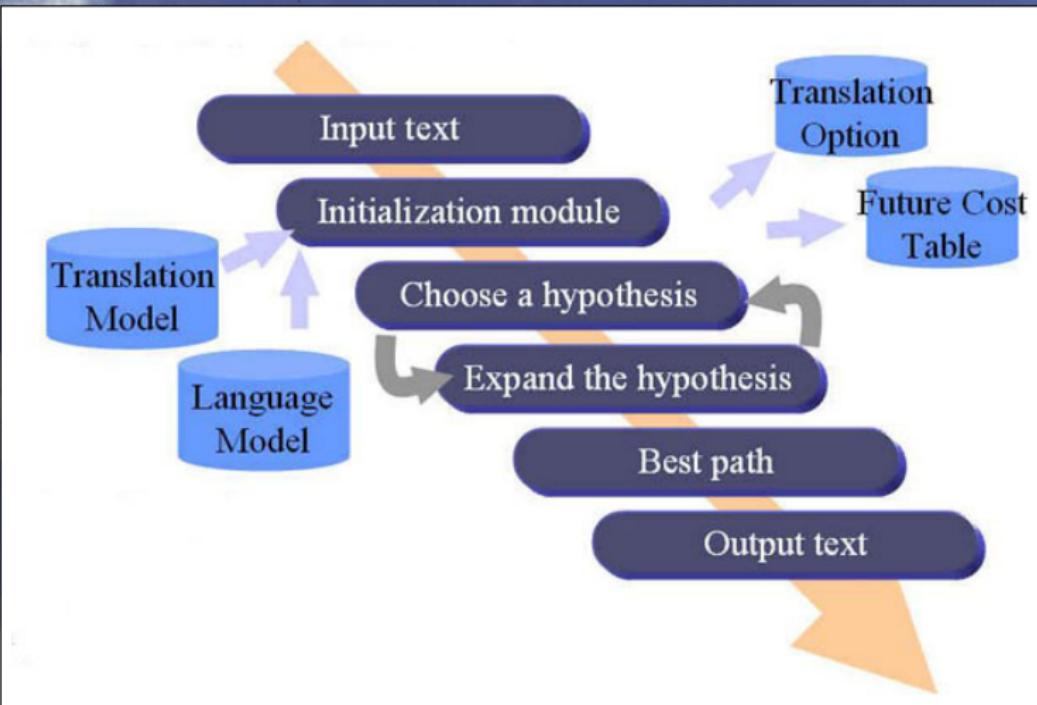
**Complete re-development of the existing
SMT infrastructure:**

1. Implementation of language models;
2. Multi-threading with large shared data;

Statistical MT Systems



SMT: Multiple computationally intensive steps.



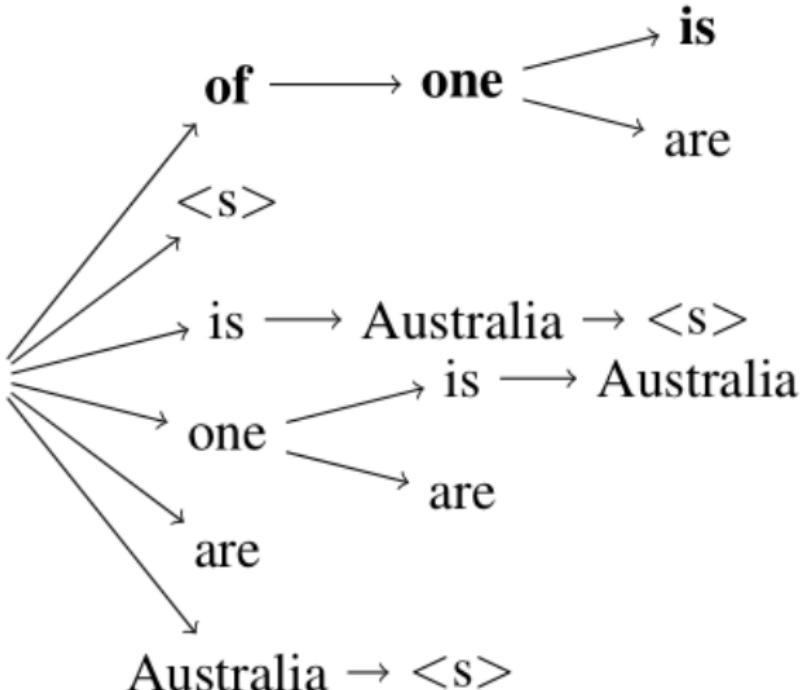
N-gram Model

- An n-gram is a sequence of n words
 - unigrams($n=1$): "is", "a", "sequence", etc.
 - bigrams($n=2$): ["is", "a"], ["a", "sequence"], etc.
 - trigrams($n=3$): ["is", "a", "sequence"], ["a", "sequence", "of"], etc.
- n-gram models estimate the conditional from n-grams counts

$$p(w_t \mid w_{t-(n-1)}, \dots, w_{t-1}) = \frac{\text{count}(w_{t-(n-1)}, \dots, w_{t-1}, w_t)}{\text{count}(w_{t-(n-1)}, \dots, w_{t-1}, \cdot)}$$

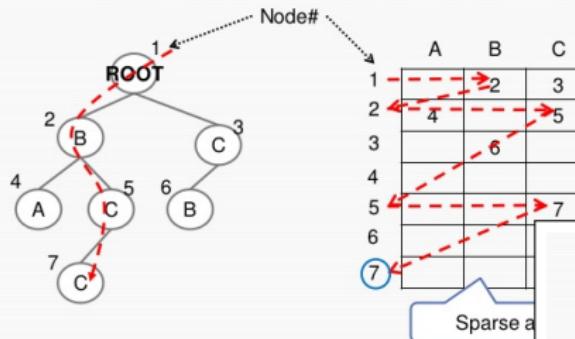
- the counts are obtained from a training corpus (a data set of word text)

Language model: Is stored in a form of a Trie



Tries: Different versions have different properties

2D Array Implementation of a Trie



Simple and fast but consumes a lot of memory

Double-Array Structures (Aoe, 1989)

What is a double-array structure?

A fast and compact representation of a trie

Abstract image

A trie is represented by two arrays (BASE and CHECK)

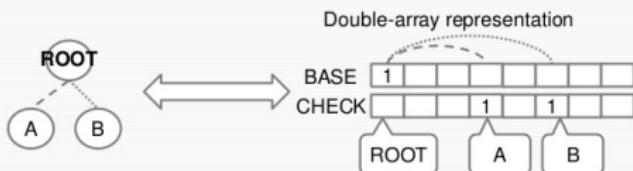


Table of Contents

- Introduction
- **Deliverable**
- Software Info
- Implementation
- Experiments
- Conclusions
- Future plans



Deliverable: Software, Report, Example models

- REMEDI/month-6/
 - software/
 - doc/
 - inc/
 - src/
 - nbproject/
 - doxygen/
 - report/
 - data/



GitHub project:

[https://github.com/ivan-zapreev/
Back-Off-Language-Model-SMT](https://github.com/ivan-zapreev/Back-Off-Language-Model-SMT)

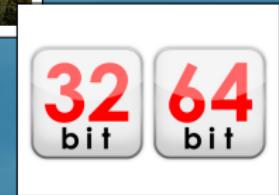
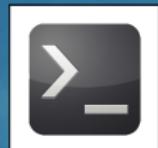
Table of Contents

- Introduction
- Deliverable
- **Software Info**
- Implementation
- Experiments
- Conclusions
- Future plans



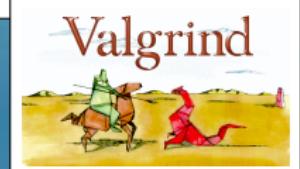
Software Info: Type, License, Platforms, Architectures

- Type:
 - Command line tool.
- License:
 - Gnu Public License v2.0
- Platforms:
 - CentOS 6.6
 - Ubuntu 15.04
 - Mac OS Yosemite 10.10
- Architectures:
 - 64-bit - full
 - 32-bit - partial



Software Info: Language, Project, Documentation, Testing

- Language:
 - C++ 11
- Project:
 - Netbeans 8.0.2
 - Makefiles
- Documentation:
 - Doxygen
- Profiling:
 - Valgrind



Software Info: Inputs and Outputs

ARPA file:

```
GNU nano 2.0.6

\data\
ngram 1=4101
ngram 2=14605
ngram 3=19222
ngram 4=19930
ngram 5=19618

\1-grams:
-2.343698    "      -0.1438925
-3.115607    '      -0.08014245
-4.031382    'll     -0.08014244
-4.031382    'm     -0.08014245
-4.031382    're     -0.08014245
-2.190489    's     -0.1138641
-4.031382    've     -0.08014245
-2.618062    (      -0.1383187
-2.630581    )      -0.1979323
-1.360232    ,      -0.4377409
```

Queries file:

```
GNU nano 2.0.6

mortgages had lured borrowers and
this will require strengthening the
, often in return for
countries , and toward targeted
and will continue the fight
, it can be counter
( ) { googletag .
believed to be falling ,
abandoning mainstream economics -- far
her colleagues in france are
region spend more on debt
are crashing from madrid to
currency convertibility with a grossly
this leads me to my
long - term climate change
chirac vehemently opposes quotas for
was kofi annan " developed
are a thing of the
and , as stanley baldwin
the three arab nme partners
```

Result:

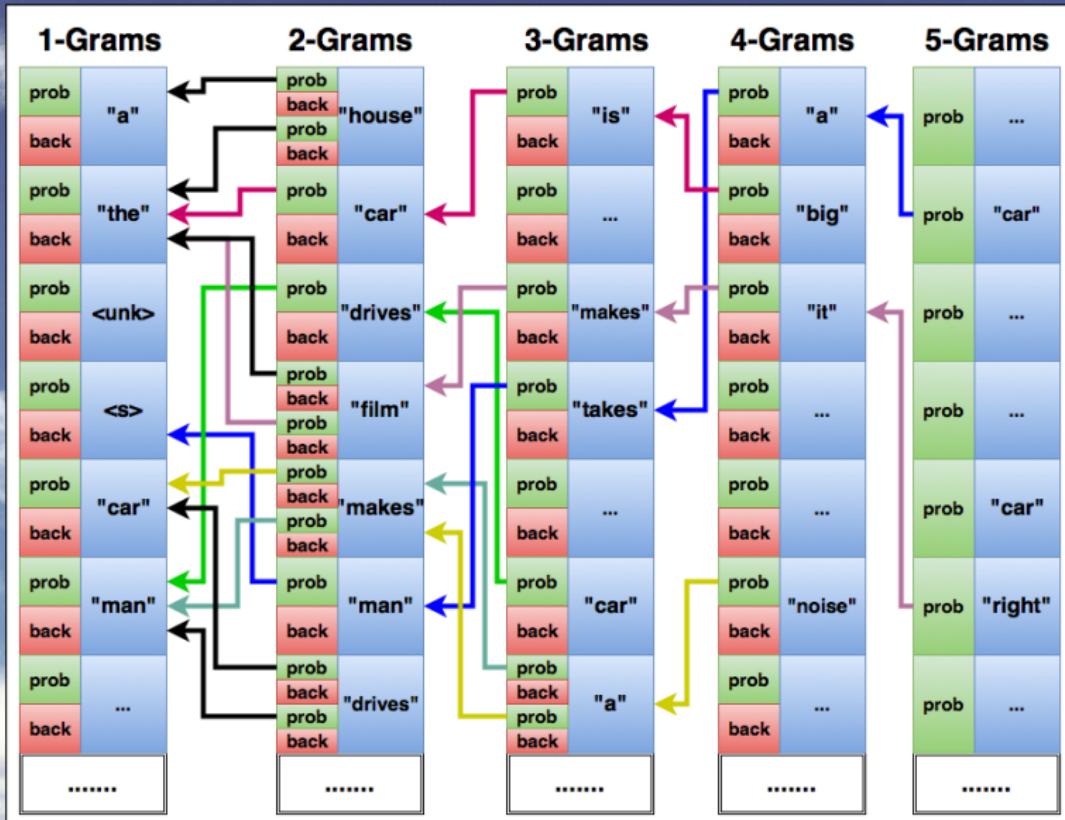
```
.....  
USAGE: Start reading and executing the test queries ...  
RESULT: log_10( Prob( the | underground publication , in ) ) = -0.62416  
INFO: Prob( the | underground publication , in ) = 0.237597  
USAGE: Total query execution time is 5.8e-05 CPU seconds.  
USAGE: Cleaning up memory ...  
INFO: Done
```

Table of Contents

- Introduction
- Deliverable
- Software Info
- **Implementation**
- Experiments
- Conclusions
- Future plans



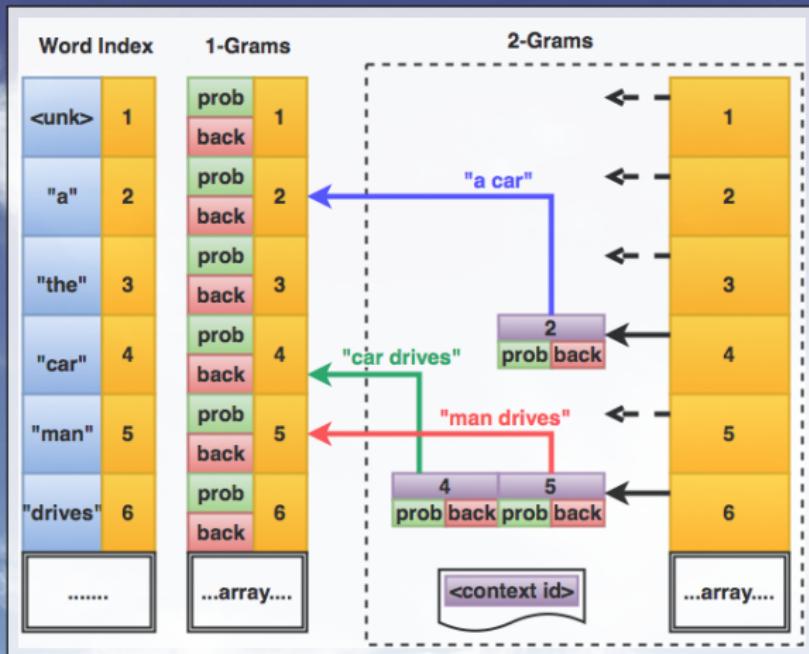
Trie types: Layered, spread n -grams



Trie types: Generic, independent n -grams

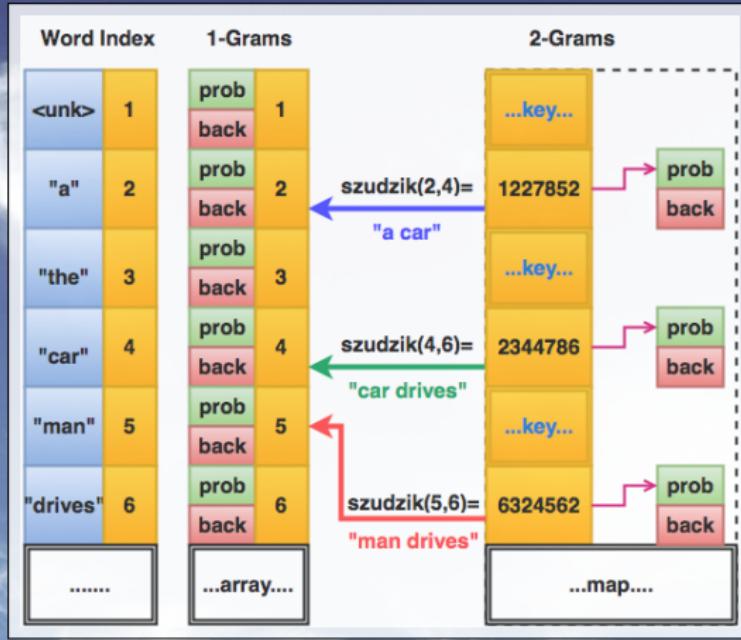
| 1-Grams | | 2-Grams | | 3-Grams | | 4-Grams | | 5-Grams | |
|---------|-------|---------|--------------|---------|------------------|---------|---------------------|---------|---------------------------|
| prob | "a" | prob | "a house" | prob | "man drives car" | prob | "the car is big" | prob | "<s> man takes a car" |
| back | | back | | back | | back | | back | |
| prob | "the" | prob | "the house" | prob | "man makes a" | prob | "car makes a noise" | prob | "the film makes it right" |
| back | | back | | back | | back | | back | |
| prob | <unk> | prob | "the film" | prob | ... | prob | ... | prob | ... |
| back | | back | | back | | back | | back | |
| prob | <s> | prob | "car drives" | prob | ... | prob | ... | prob | ... |
| back | | back | | back | | back | | back | |
| prob | "car" | prob | "man drives" | prob | ... | prob | ... | prob | ... |
| back | | back | | back | | back | | back | |
| prob | "man" | prob | ..." | prob | ..." | prob | ..." | prob | ..." |
| back | | back | | back | | back | | back | |
| | | | | | | | | | |

Layered: Word to Context Array (w2ca)



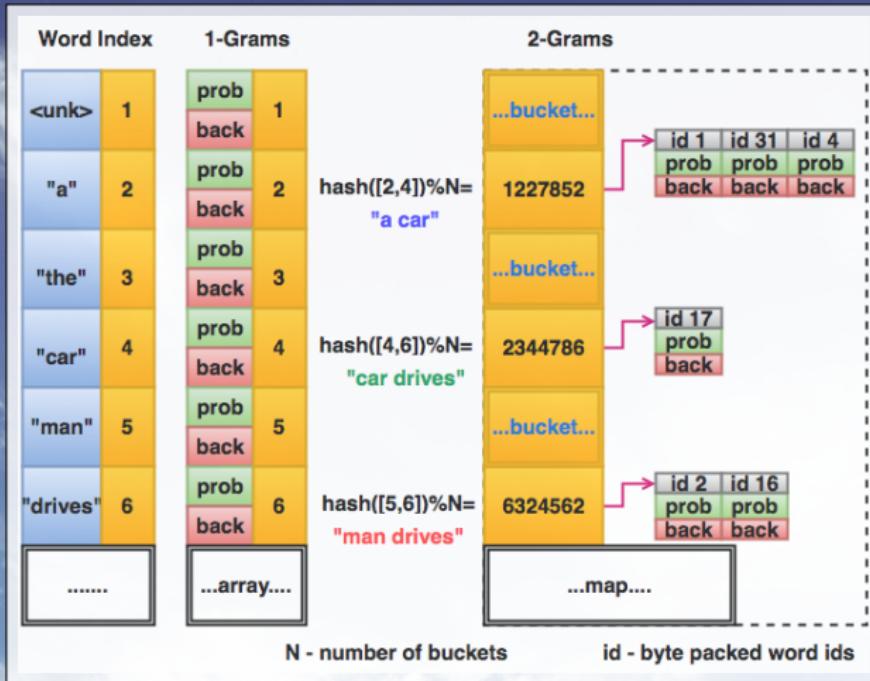
Indexed by word id; Natural Context id; Ordered arrays;
Memory efficient; Binary search;

Layered: Context to Data Map (c2dm)



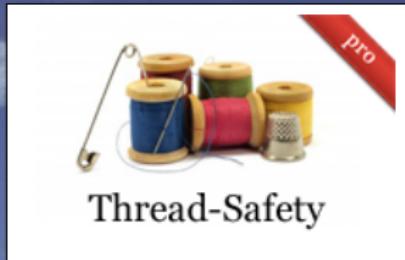
N-gram id via 2D plane enumeration; Unordered hash map;
Memory efficient; Constant average time look up;

Generic: Gram to Data Map (g2dm)

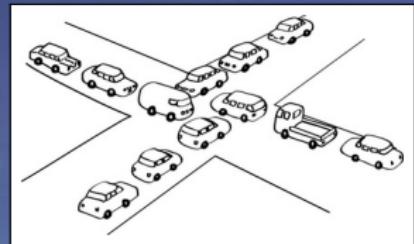


Unordered hash map; n -gram id from word ids; Memory greedy; Potentially Constant average time look up;

Multi-threading: Are we ready?



Thread-Safety

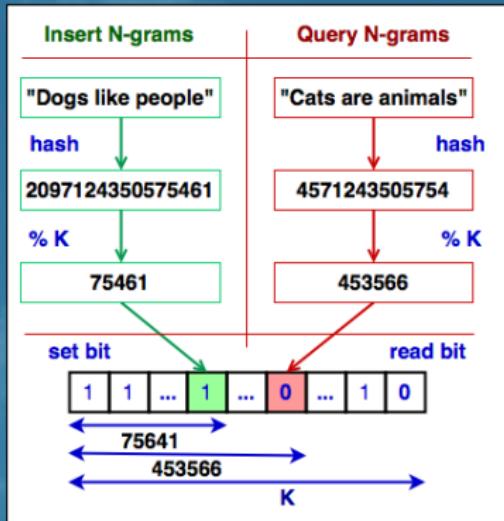


All shared resources are read only, when querying!

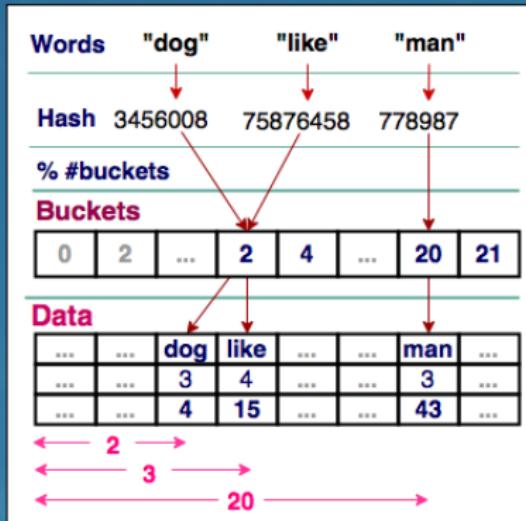


Implemented optimizations

- Hash Caching:



- Optimizing Index:



Plus:

- Lower index to more frequent words
- Etc.

Further details: w2ch, c2dh, c2wa, and etc



SEE THE REPORT!



Table of Contents

- Introduction
- Deliverable
- Software Info
- Implementation
- Experiments
- Conclusions
- Future plans



Considered tools and configurations.

- Tools:
 - Our tool:
 - Owl release, v1.0
 - Different trie types
 - Optimizations On/Off
 - KenLM:
 - GitHub snapshot
 - Modified source code
 - SRILM:
 - v1.7.0
- Target:
 - Model MRSS
 - Query CPU times
- Config:
 - Various 5-gram model sizes
 - 10^8 of 5-gram queries



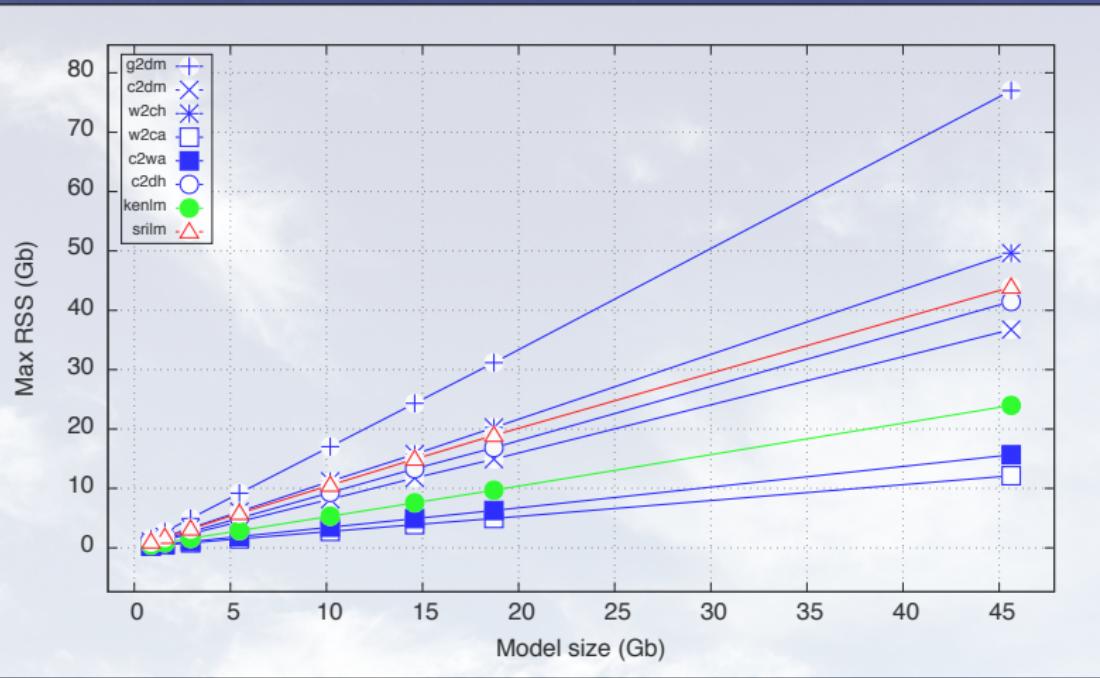
Warning: Not 100% fair!

$$\text{Prob}(w_5 | w_1 w_2 w_3 w_4)$$

vs

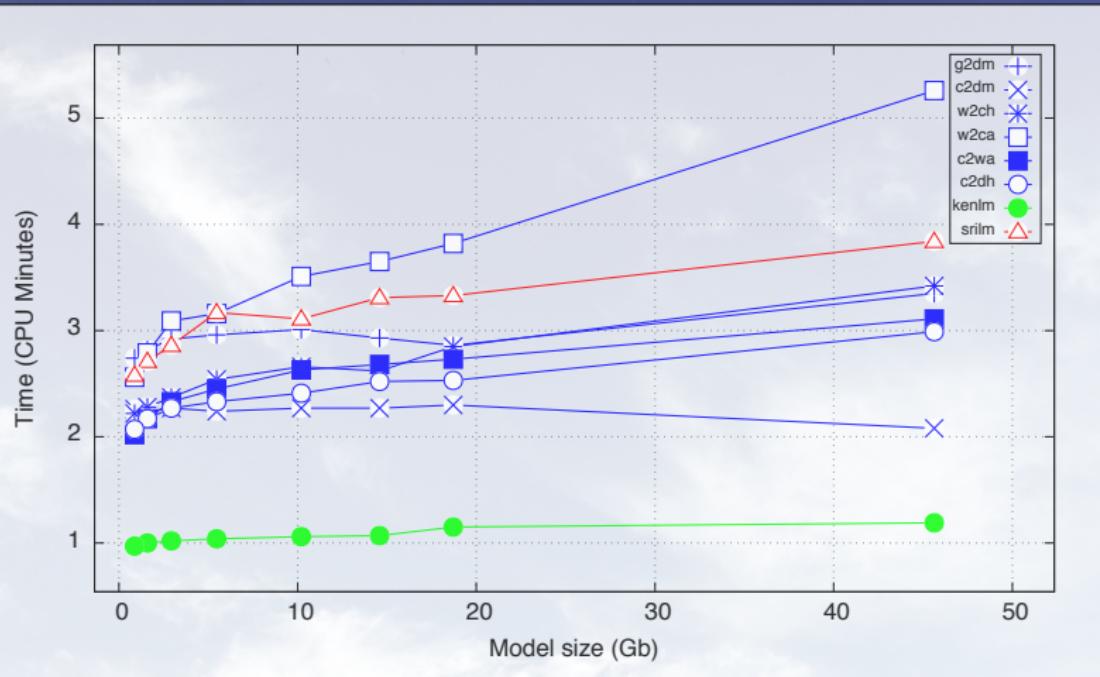
$$\sum_{i=1}^5 \text{Prob}(w_i | w_1 \dots w_{i-1})$$

MRSS: Hash Caching - Off; Optimizing Index - Off



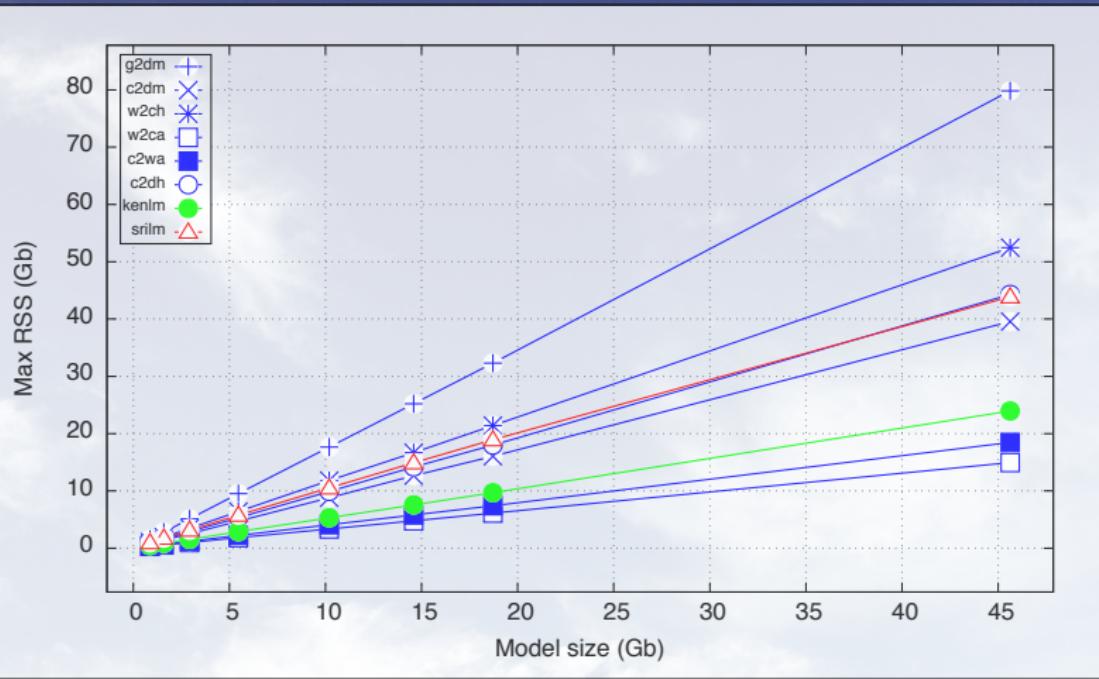
w2ca is 2 times more memory efficient than KenLM ; c2wa is almost as efficient ; g2dm is the greediest ;

CPU: Hash Caching - Off; Optimizing Index - Off



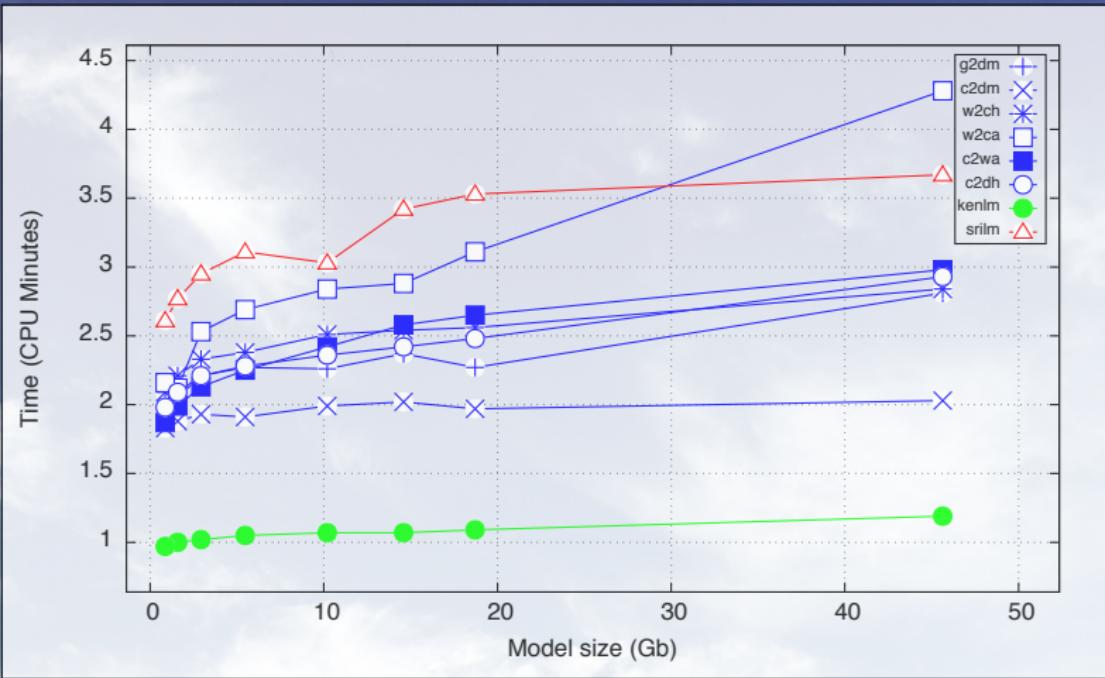
w2ca is the slowest ; c2dm is the fast and close to constant ;
c2wa exhibits a decent performance ;

MRSS: Hash Caching - On; Optimizing Index - Off



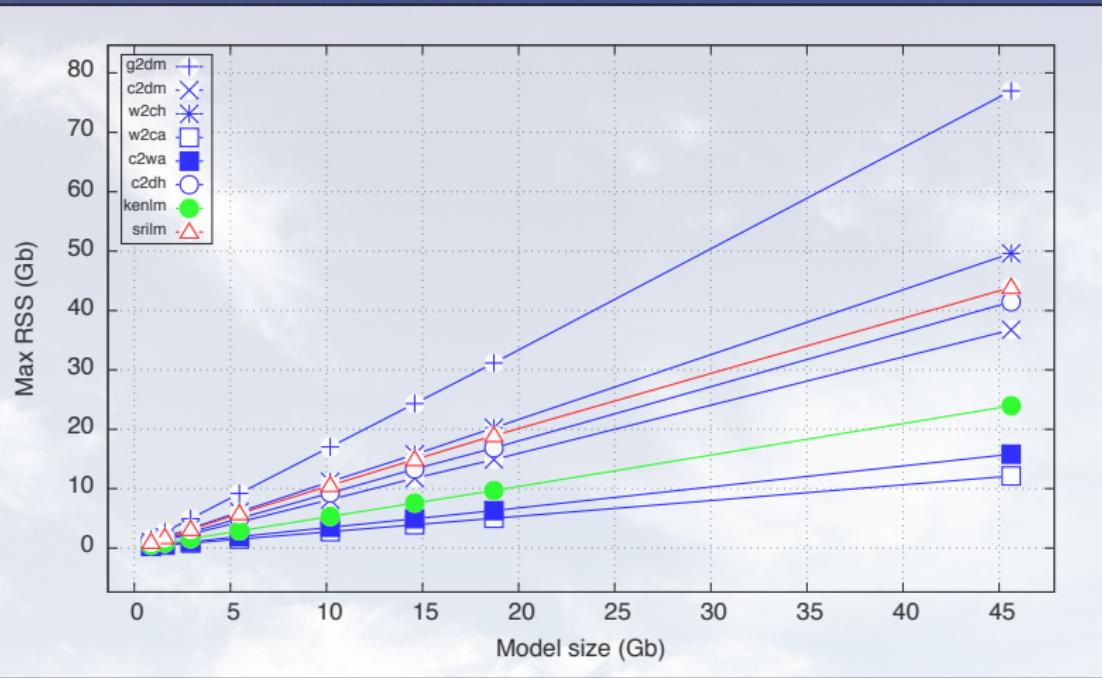
≈ 2.5% memory increase only ;

CPU: Hash Caching - On; Optimizing Index - Off



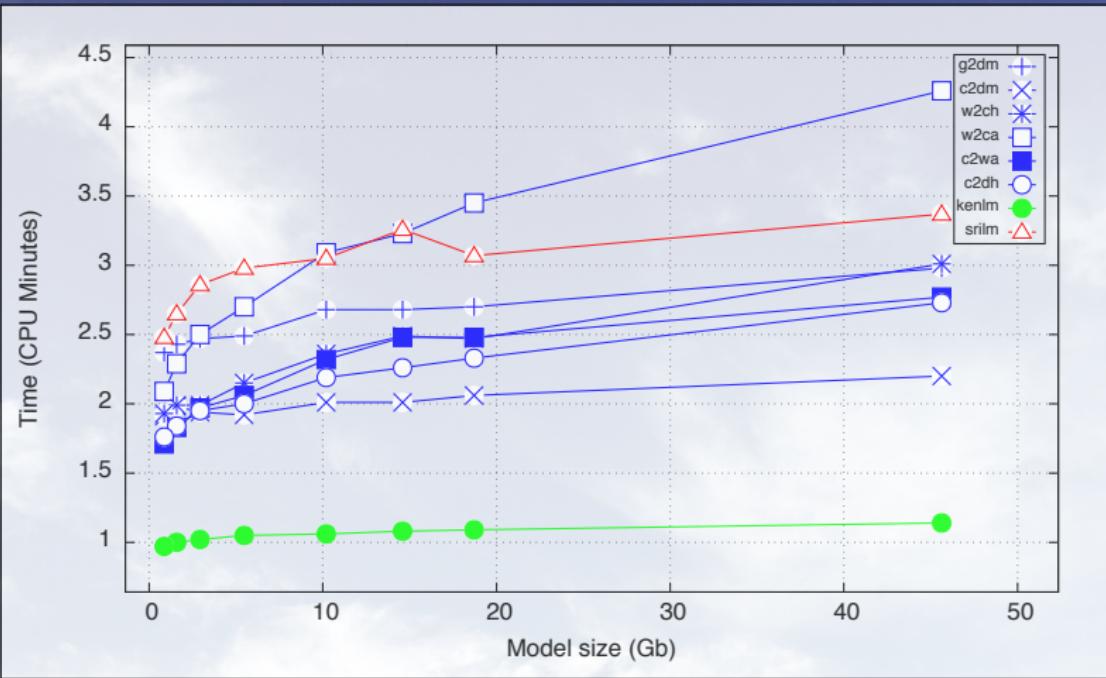
10 to 15 % performance improvement ;

MRSS: Hash Caching - Off; Optimizing Index - On



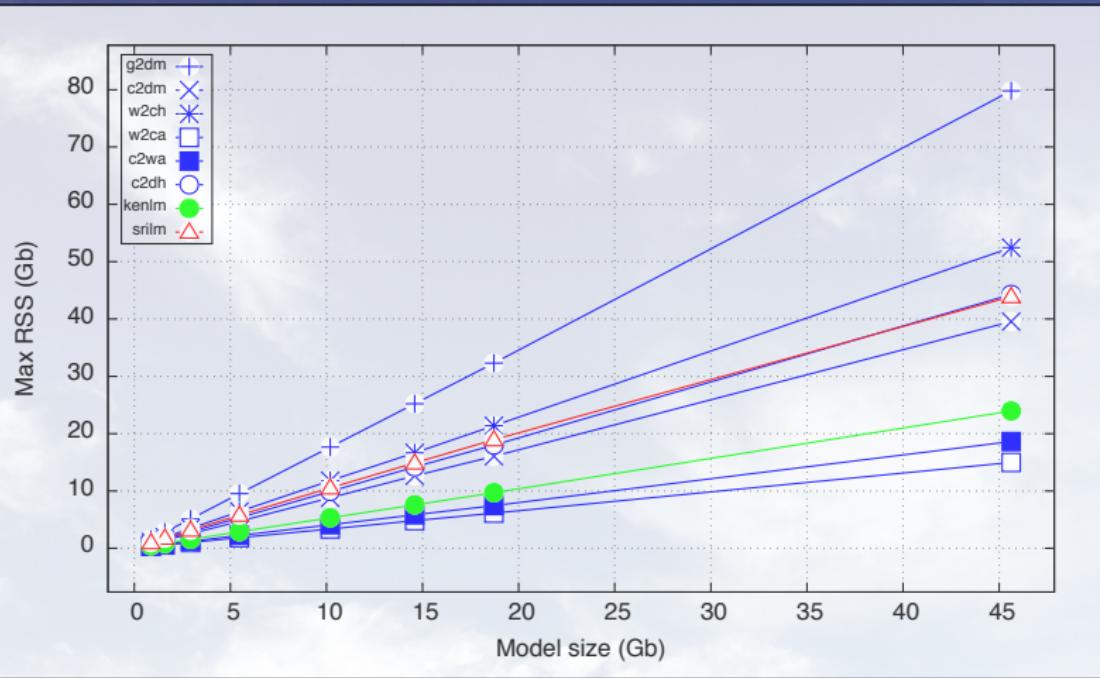
zero memory penalties ;

CPU: Hash Caching - Off; Optimizing Index - On



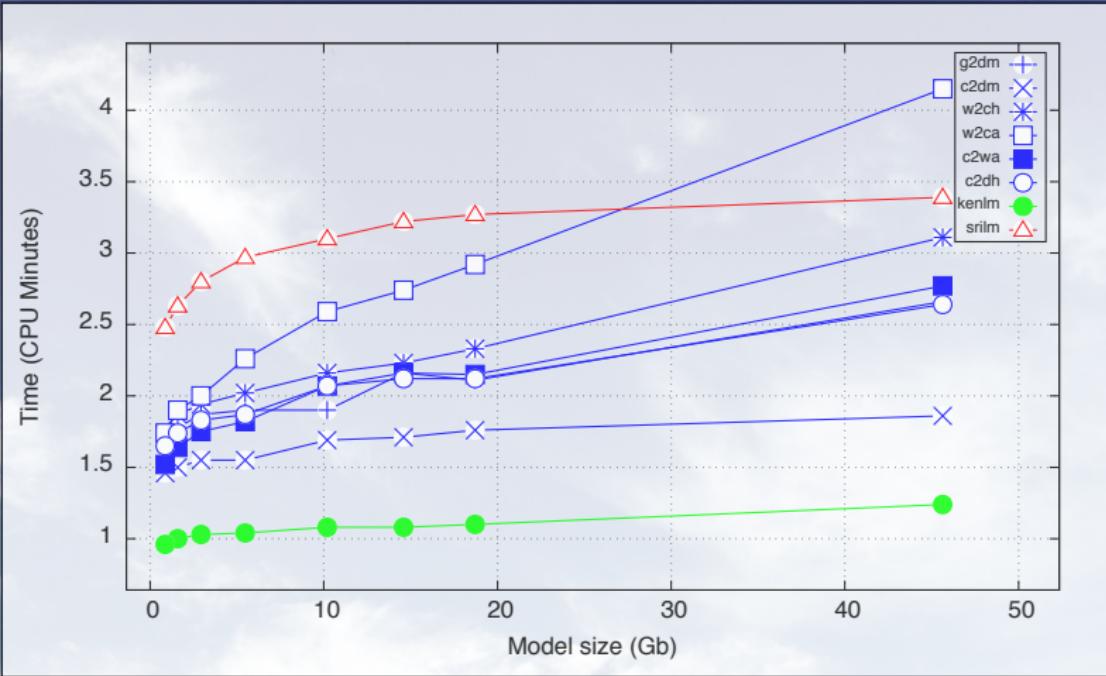
up to 10 % performance improvement ;

MRSS: Hash Caching - On; Optimizing Index - On



cumulative $\approx 2.5\%$ memory increase ;

CPU: Hash Caching - On; Optimizing Index - On



cumulative 20 to 25 % performance improvement ;

Experiments: Summary

- Bitmap hashing:
 - 10 to 15% of performance improvement
 - $\approx 2.5\%$ extra memory.
- Optimizing word index:
 - up to 10% of performance improvement
 - no extra memory.
- Using layered tries:
 - Small MRSS values
 - Low CPU times
- KenLM (Generic tries):
 - Moderate memory consumption
 - Very fast queries
 - Uses words and (n -gram) hashes as unique identifiers!

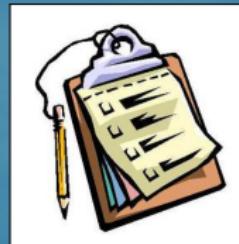


Table of Contents

- Introduction
- Deliverable
- Software Info
- Implementation
- Experiments
- **Conclusions**
- Future plans



Conclusions

- Developed 6 trie variants
 - Thread-ready
 - Easily extendable
 - Valgrind profiled
- Compared to SRILM and KenLM
 - KenLM is fast but risky
- Layered tries:
 - Up to 70% memory reduction
 - Up to 2 times slower
 - Up to 25% optimization improvement
- Guidelines:
 - *c2wa* and *w2ca* - for low memory machines
 - *c2dm* and *c2dh* - for fast querying
 - *w2ch* and *g2dm* - are to be improved

Table of Contents

- Introduction
- Deliverable
- Software Info
- Implementation
- Experiments
- Conclusions
- **Future plans**



Future plans: Remaining modifications



1. Introduce parallel querying.
 - Multi-threading within a trie instance.
2. Extend with the KenLM-type trie.
 - Be as good as the competitors!
3. Compute cumulative probabilities.
 - Fair comparison with competitors!

Future plans: Second stage



1. Model data structures

- Translation
- Reordering

2. Algorithms

- Decoder search
- Pruning strategies

Discussion



Thank you all!

Wednesday
July 1