Machine Translation and its Components

Francis Bond

Ling 473

2016-09-01

Overview

- Machine Translation and Post Editing
- The Empirical Revolution in NLP
- NLP Pipeline
  - Parsing
  - Generation
  - Transfer^{later}
- EBMT and SMT

Machine Translation

Machine Translation

- There is a great demand for Machine Translation
- But there are many problems
  - Linguistic
  - Technical
  - Interface
- Kinds of Machine Translation
  - Rule-based (Knowledge-based):

    Transfer : $n(n-1)$ engines for n languages
    Interlingua : $2n$ engines for n languages
  - Data-driven: Example-based, Statistical
- Successful and Unsuccessful Applications

Machine translation: problems and issues

- An overall look at the current state of machine translation based on a panel presentation by John Hutchins (13 December 2007)

- Abbreviations:

RBMT Rule-based Machine Translation
EBMT Example-based Machine Translation
SMT Statistical Machine Translation
SL Source Language
TL Target Language
PE Post-Editing
HT Human Translation
MT Machine Translation

Kinds of Machine Translation

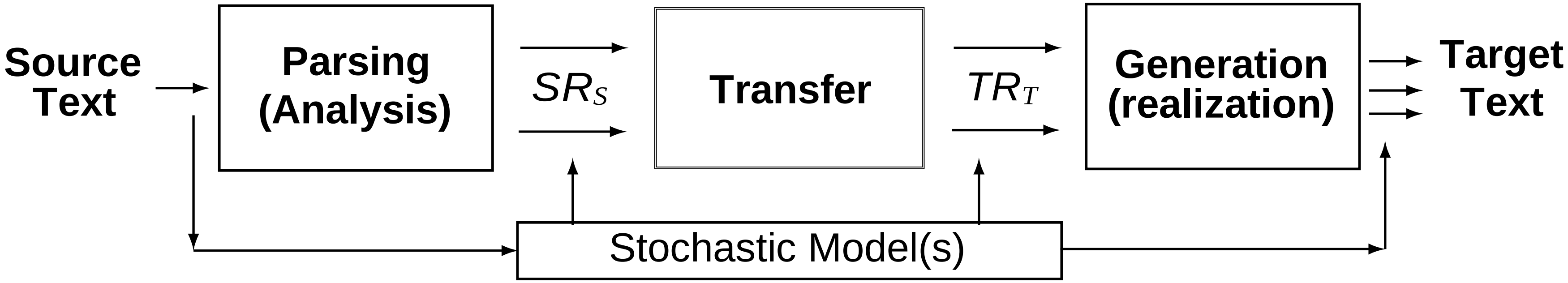
- Knowledge-driven
  - Rule-based (Knowledge-based): Transfer, Interlingual

  Attempt to understand the text in some way and then translate it, normally with a hand built model.

- Data-driven
  - Example-based MT, Statistical MT

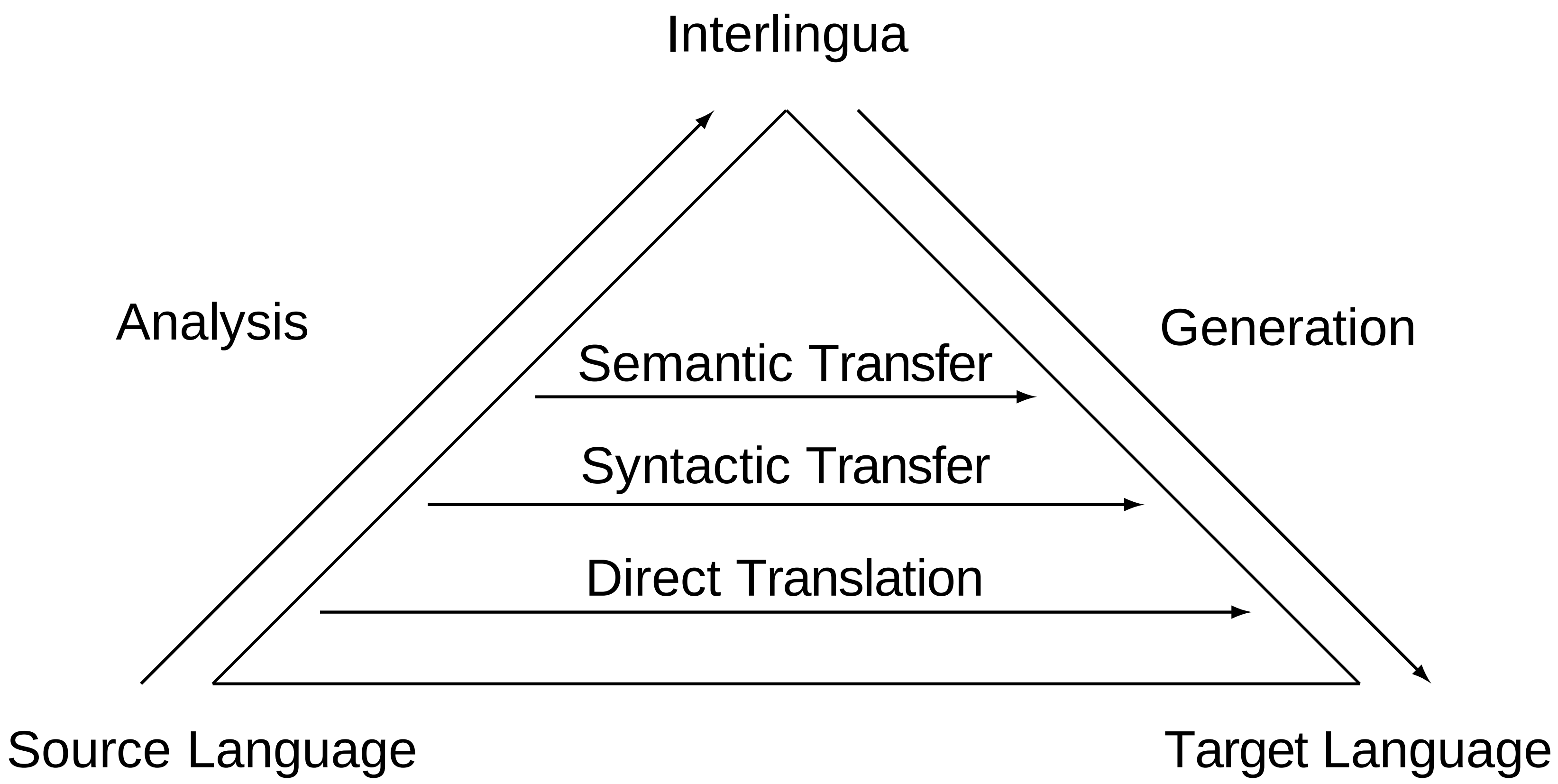
  Attempt to build a model based on existing translations, don't attempt understanding, just approximation.

Components of Transfer based MT



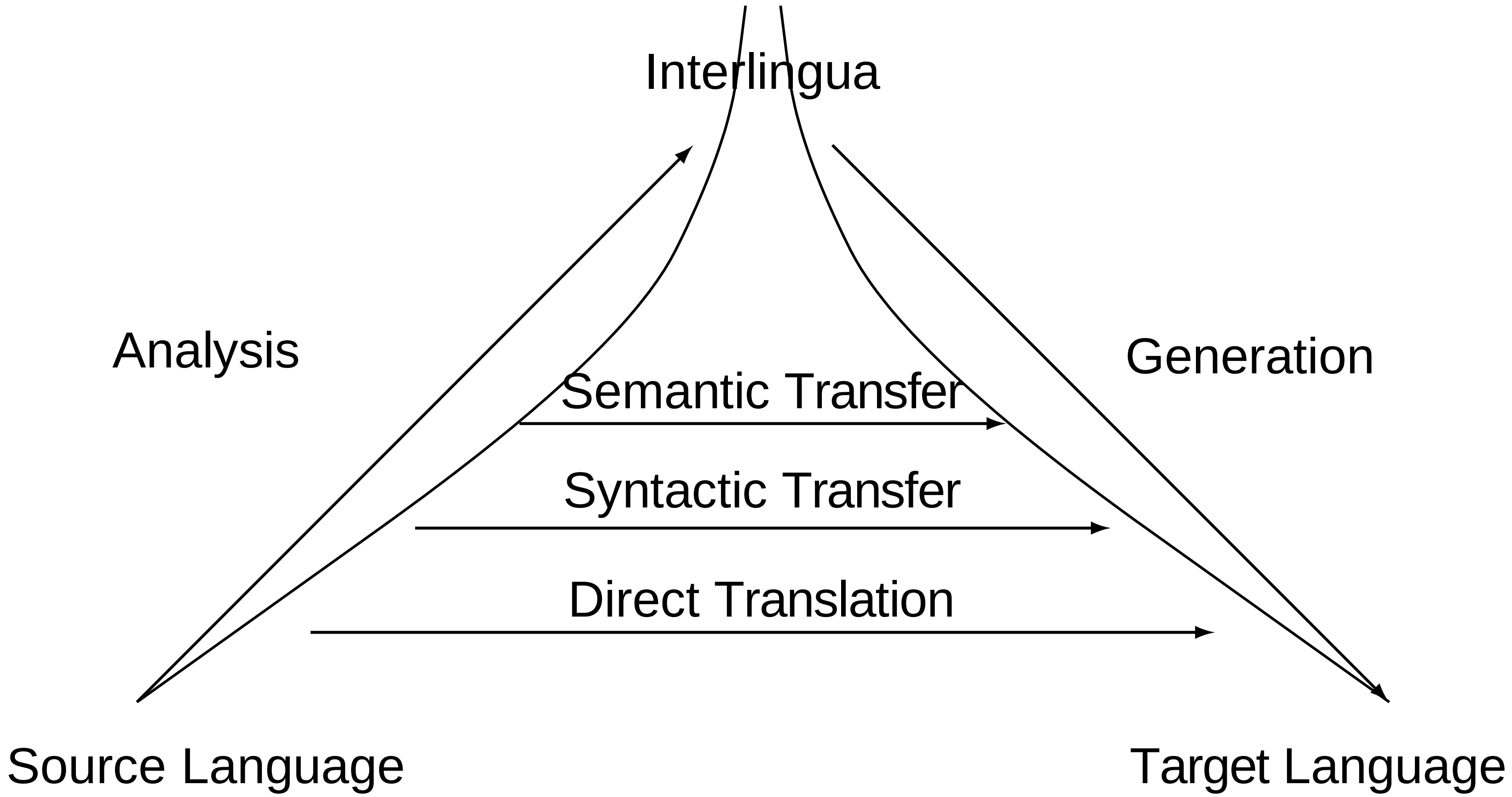
- Parse source text to source representation (SR)
- Transfer this to some target representation (TR)
- Generate target text from the TR
- If the source = target, then we are paraphrasing

How Deep Should We Go?



The Vauquois Triangle

Rule Based Machine Translation



The Ikehara Discontinuity

Also known as the Copestake Inverted Funnel.

Inherent (linguistic) problems:

Bilingual lexical differences

- Bilingual Lexical Ambiguity
  (more than one equivalent, whether ambiguous in SL or not):
  - *Schraube*: *screw/bolt/propeller*
  - *corner* : *coin/angle*; *Ecke/Winkel*
  - *light*: *léger, clair, facile, allumer, lumière, lampe, feu*
  - *look* : *regarder, chercher, sembler*
  - *bank*: 銀行 *ginkou* “financial institution”, 土手 *dote* “embankment”

- Lexical Gap: the absence of a word in a particular language
  Lexicalized in one language but not the other
  - *river*: *fleuve/rivière*
  - *Taube*: *dove/pigeon*
  - 牛油 *nūuyōu*: *butter, ghee*
  - *shallow waters*: *ape puţin adânci* “not so deep waters” (Romanian)
  - *wear* : *kiru* “wear clothes on upper body”, *haku* “wear clothes on lower body”
  - **ungood*

Solved (?) by

- one-to-many contextual rules (RBMT)
- examples (EBMT)
- frequencies and ‘language models’ (SMT)

Inherent (linguistic) problems:

Structural ambiguity

(1) *Peter mentioned the book I sent to Mary*

Peter mentioned the book which I sent to Mary

Peter mentioned to Mary the book which I sent [to Peter/David]

(2) *We will meet the man you told us about yesterday*

. . . *the man you told us about yesterday*

(3) *We will meet the man you told us about tomorrow*

we will meet tomorrow the man . . .

(4) *pregnant women and children* [unambiguous for HT]

des femmes et des enfants enceintes [produced by MT system]

(5) a. *Smog and pollution control are important factors*

b. *Smog and pollution control is under consideration*

c. *The authorities encouraged smog and pollution control*

Often, problems such as (1), (2), and (3) are problematic for RBMT, but they may be ‘solved’ by SMT ‘language model’ and by EBMT databases. But problem (4) requires ‘knowledge’ (i.e. rule-based KBMT)

Inherent (linguistic) problems:

Bilingual structural differences

(6) Young people like this music

Cette musique plaˆıt aux jeunes gens
This music played by young people

(7) The boy likes to play tennis

Der Junge spielt gern Tennis
The boy plays happily tennis

(8) He happened to arrive in time

Er ist zufällig zur rechten Zeit angekommen
He is accidentally on right time came

Difficult to specify transfer rules (RBMT) to cover all circumstances and contexts; but example-based (EBMT) and statistics-based (SMT) approaches no better. You need some kind of parsing.

Multi-word Expressions

- A multi word expression is one in which the meaning of the whole is not fully predictable from the meanings of the individual words.
  - *Kick the bucket*
  - *Look up*
  - *New York Giants*

- In MT a MWE is when a translation is not word to word, but rather a set of words translates to a set of words.

(9) *1993* 年 の 頭
1993 nen-no atama
1993 year 's head

"the beginning of 1993"

"early 1993"

Non-linguistic problems of ‘reality’

(10) *The soldiers$_i$ shot at the women$_j$ and some of them$_j$ fell*

(11) *The soldiers$_i$ shot at the women$_j$ and some of them$_i$ missed*

➢ must know what ‘them’ refers to e.g. if translating into French (*ils* or *elles*)

⊗ No easy solutions with linguistic rule-based approaches

⊗ No easy solutions with corpus-based approaches

➢ Need to add reference resolution (currently a hot topic!)

- Perhaps only solution using Artificial Intelligence approaches Knowledge-based **machine translation**, e.g. Carnegie-Mellon University)

- However, perhaps this problem is exaggerated: no need to understand what AIDS and HIV are in order to translate:

(12) *The AIDS epidemic is sweeping rapidly through Southern Africa. It is estimated that more than half the population is now HIV positive.*

Problems of stylistic difference

(13) *The possibility of rectification of the fault by the insertion of a valve was discussed by the engineers*

(14) *The engineers discussed whether it was possible to rectify the fault by inserting a valve*

- **English**: *Advances in technology created new opportunities*
- **Japanese**: *Because technology has advanced, opportunities have been created*
- **Japanese**: *Technology has advanced. There are new opportunities.*

All methods of MT tend to retain SL structural features; however, theoretically SMT *language model* approach should be more TL-oriented.

Hybrid systems

- clearly, none of the current MT ‘models’ are capable of solving all **problems**
- hence search for hybrid architectures
- in theory, it would seem that (on average):
  - RBMT better for SL analysis
  - EBMT better for transfer
  - SMT best for TL generation

- Problem is that different approaches not easily compatible:
  - there are however research prototypes combining:
  - EBMT with statistical methods
  - EBMT using rules similar to those in RBMT systems
  - perhaps a version of EBMT will be the answer
- Currently ‘hybrid’ systems are mainly parallel systems with a selection mechanism
  - i.e., translate with several systems and chose the **best** translation
- However, more RBMT systems use statistical models and more SMT systems use parsers, so there is gradual convergence

Translation demand

- Dissemination: production of ‘publishable quality’ texts
  - but, since raw output inadequate:
    - post-editing
    - control of input (pre-editing, controlled language)
    - domain restriction (reducing ambiguities)
- Assimilation: for extracting essential information
  - use of raw output, with or without light editing
- Interchange: for cross-language communication (correspondence, email, etc.)
  - if important: with post-editing; otherwise: without editing
- Information access: to databases and document collections

Post-editing

Post-editing:

Types of errors for correction

- Misspelling in original not recognised, therefore not translated;
- missing punctuation
- e.g. *The Commission vice president* translated as *Le président du vice de la Commission* (because no hyphen between *vice* and *president*)

Post-Editing: Complex syntax

- prepositions:
  - . . . *el desarrollo de programs de educación nutricional . . .*
    - MT: *. . . the development of programs of nutritional education*
    - PE: *. . . in nutritional education . . .*
- verb phrases:
  - . . . *el procedimento para registrar los hogares . . .*
    - MT: *the procedure in order to register the households*
    - PE: *. . . the procedure for registering households*

Post-editing: types of errors (contd.)

- inversions:
  - *. . . la inversión de la Argentina en las investigaciones de malaria*
    - MT: *. . . the investment of Argentina in the research of malaria*
    - PE: *Argentina's investment in malaria research*
- reflexive verbs with inversions:
  - *Se estudiarán todos los pacientes diagnostocados como . . .*
    - MT: *There will be studied all the patients diagnosed as . . .*
    - PE: *Studies will be done on all patients diagnosed as . . .*
  - *En 1972 se formuló el Plan Decenal de Salud para las Américas.*
    - MT: *In 1972 there was formulated the Ten-Year Health Plan for the Americas*
    - PE: *The year 1972 saw the formulation of the Ten-Year Health Plan for the Americas.*

Translators and post-editors

- post-editing by translators:
  - not foreseen initially
  - skills acquired over time and practice in real working conditions
  - requires perseverance
    (initially post-editing takes longer than complete translation)
- High quality human translation is also normally edited

Post-editing pros and cons

- advantages:
  - translators can maintain quality control
  - consistency of terminology
  - repetitive matter produced by MT, linguistic quality by HT
- disadvantages:
  - correction of ‘trivial’ mistakes; too often correcting same type of error
  - style too much SL oriented
  - translators as ‘slaves’ to machine
- need for special post-editing tools (not always provided)
- specially trained post-editors [still rare]

Controlled language

- Controlled authoring of the source text in standard manner, suitable for unambiguous translation

- Typical rules:
    - use only approved terminology, e.g. *windscreen* rather than *windshield*
    - use only approved sense: *follow* only as ‘come after, not ‘obey’
    - avoid ambiguous words: *replace*, either (a) remove and put back, or (b) remove and put something else in place; not *appear* but: come into view, be possible, show, think
    - only one ‘topic’ per sentence, e.g. one instruction, command
    - use short sentences, e.g. maximum 20 words
    - . . .

- Typical rules (cont):
  - . . .
  - do not omit articles
  - do not use pronouns instead of nouns if possible
  - do not use phrasal verbs, such as *pour out*
  - do not omit implied nouns
  - do not drop subjects
  - avoid co-ordination of phrases and clauses

Fully Automatic High Quality Machine Translation

- METEO
  - Canadian English ↔ French system
  - Translates meteorology text (weather reports)
  - Short, repetitive sentences
  - 30 million sentences a year
  - MT with human revision (< 9% of sentences revised)
- ALT-FLASH
  - Japanese → English system
  - Translates Stock market flash reports
  - Short, repetitive sentences, speed very important
  - 10 thousand sentences a year
  - MT with human revision (< 2% of sentences revised)

MT for assimilation

- publication quality not necessary
- fast/immediate
- readable (intelligible), for information use
    - intelligence services (e.g. NAIC)
    - occasional translation (home use)

- as draft for translation
- aid for writing in foreign language
  - as used by EC administrators
- emails, Web pages
- any system type can be used (including those originally for mainframes and PCs
  - online MT has all types of rule-based systems - and now also SMT

Empirical Natural Language Processing

The Empirical revolution in NLP

- As systems get bigger, behavior is harder to predict

- Looking at system output one sentence at a time is slow

- Can we automate testing?

1. Create a gold standard or reference (the right answer)
2. Compare your result to the reference
3. Measure the error
4. Attempt to minimize it globally (over a large test set)

The Empirical approach

1. Develop an algorithm and gather examples/rules from training data

2. Optimize any parameters on development data

   ➢ Normally about 10% of the training data

3. Test on held-out, unseen test data

This gives a fair estimate of how good the algorithm is
— if the test criteria are appropriate.

Word Error Rate in Speech Recognition

- The first successful wide spread testing:
  - Compare your output to a reference
  - Calculate the number of substitutions, deletions and insertions to make them match (Minimum edit distance)
  - Normalize by dividing by the length of the reference

$$WER = \frac{S+D+I}{N}$$

Reference:	I	want	to	recognize		speech	today
System:	I	want	wreck	a	nice	peach	today
Eval:			S	S	I	S	

- $WER = \frac{3+0+1}{6} = 0.667$

Some properties of WER

- Correlates well with the task
- Reducing WER is always a good thing
- A WER of 0 implies perfect results
  (assuming the reference is correct)
- $WER < 0.05$ generally considered the minimum to be useful
- Competitions were held to see who could get the lowest WER
  - Speech Recognition had 10 years of rapid improvement
  - It has slowed down now

How good are the systems?

Task	Vocab	WER (%)	WER (%) adapted
Digits	11	0.4	0.2
Dialogue (travel)	21,000	10.9	—
Dictation (WSJ)	5,000	3.9	3.0
Dictation (WSJ)	20,000	10.0	8.6
Dialogue (noisy, army)	3,000	42.2	31.0
Phone Conversations	4,000	41.9	31.0

Speaker adapted systems have a lower WER.

Speaker Adaptation as you use the system

- Personal Recognition launched by Google in 2010
  - associate the user’s speech input with the user
    build a small speech corpus
  - use these words to build a speech model specifically for the user
  - accuracy improvements begin fairly quickly and build over time
  - you must opt in to use the system
- Improve the data, not the algorithm
- Build specialized models

http://googlemobile.blogspot.sg/2010/12/voice-search-gets-personal.html

Empirical vs Rational NLP

- The 1990s went through an empirical revolution
- Funding agencies sponsored competitions
  - TREC: Text REtrieval Conference
  - MUC: Message Understanding Conference
  - DARPA Machine Translation Competitions
- Data to test with became more available
- Reviewers demanded evaluation in papers
- A lot of research on evaluation methods

Why do we test in general?

Testing is important for the following reasons

1. Confirm Coverage of the System

2. Discover Problems

3. Stop Backsliding
   - ➢ Regression testing — test that changes don’t make things worse

4. Algorithm Comparison
   - ➢ Discover the best way to do something

5. System comparison
   - ➢ Discover the best system for a task

How do we test?

- Functional Tests (Unit tests)
  - Test system on test suites
- Regression Tests
  - Test different versions of the system
- Performance Tests
  - Test on normal input data
- Stress Tests (Fuzz tests)
  - Test on abnormal input data

MT Evaluation

- Evaluating MT output is non-trivial
  - There may be multiple correct answers.
    - *I like to swim*
    - *I like swimming*
    - *Swimming turns me on*
- Hand evaluation requires a bilingual evaluator - expensive
- Automatic evaluation can be done by comparing results (in a held out test set) to a set of reference translations
  - The most common metric is BLEU
  - Other scores are: Word Error Rate; METEOR

MT Evaluation: Fluency and Adequacy

- Fluency: How do you judge the fluency of this translation?
  - 5 = Flawless English
  - 4 = Good English
  - 3 = Non-native English
  - 2 = Disfluent English
  - 1 = Incomprehensible

- Adequacy: How much of the meaning expressed in the reference translation is also expressed in the hypothesis translation?
  - 5 = All
  - 4 = Most
  - 3 = Much
  - 2 = Little
  - 1 = None

MT Evaluation: The BLEU score

- BLEU score compares n-grams (normally up to 4) with those in the reference translation(s) (with a brevity penalty)

$$BLEU \approx \sum_{i=1}^{n} \frac{\text{n-grams in sentence and reference}}{|\text{n-grams}|}$$

- 0.3–0.5 typical; 0.6+ approaches human
- Only really meaningful summed over a test set
  - individual sentences are too short

An Example of Variation

1. *Early and frequent releases are a critical part of the Linux development model*

REF 早期 かつ 頻繁 な 公開 は 、 リナックス の 開発 モデル の 重要 な 部分 で ある 。

(a) 早く 、 そして 頻繁 に 公表 する こと は リナックス の 発展 モデル の 重要 な 一部 で ある 。

(b) 早く 、 そして 頻繁 な リリース は 、 Linux 開発 モデル にとって は 重要 な 部分 で ある 。

More overlap with the reference is better so (a) is better.

BLEU pros and cons

- Good
  - Easy to calculate (if you have reference translations)
  - Correlates with human judgement to some extent
  - Used in standard competitions
- Bad
  - Doesn't deal well with variation
    - Exact string match
    - Near misses score zero: *cat* $/=$ *cats*!
  - Biased toward n-gram models
    - SMT systems optimize for BLEU

Misleading Bleu Scores

- 信号は赤でした。
  - The light was red.
  - The signal was red. (0.35)
- 大丈夫です。
  - I'm all right.
  - I am all right. (0.27)
- 空港から電話しています。
  - I'm calling from the airport.
  - I am telephoning from the airports. (0.22)

How to improve the reliability?

- Use more reference sentences

- Use more translations per sentence
  - Can be automatically created by paraphrasing

- Improve the metric: METEOR
  - add stemmed words (partial score): *cat* ≈ *cats*!
  - add WordNet matches (partial score): *cat* ≈ *feline*!

- Unfortunately this adds noise
  - Errors in stemming
  - Uneven cover in WordNet

- Still better than BLEU (so far) — but harder to calculate

Problems with testing

- You get better at what you test
- If the metric is not the actual goal things go wrong
  - BLEU score originally correlated with human judgement
  - As systems optimized for BLEU
  - . . . they lost the correlation
  - You can improve the metric, not the goal
- The solution is better metrics, but that is hard for MT
- We need to test for similar meaning: a very hard problem

Conclusion

- A surprising amount of variation is possible in MT
- This makes evaluation difficult
  - If we know the correct answer, the problem is solved
- But evaluation is very important in NLP
  - Use automatic evaluation
  - Recognize the risks

NLP Pipeline

Natural Language Processing Steps

- Morphological Analysis — dealing with words
  - Segmentation — splitting into words
  - Lemmatization — finding the base form (or lemma)
  - Part of Speech Tagging – assigning categories
- Parsing — assigning structure
- Generation — producing text

Morphological analysis

Morphological analysis

- ➢ This deals with the analysis of words as isolated units
- ➢ Separate into words (segmentation)
- ➢ Determine the base form (lemmatize)
- ➢ Determine the POS (one or many)
- ➢ Determine the sense (one or many)

⊗ Unknown words are a big problem

- × ライ\ナス\は\この\よう\な\こと\を\信じ\て\いる
- × Rai\nasu\ha\kono\you\na\koto\wo\shinji\te\iru
- ➢ Linus believed something like this

Linus vs Linus

Lainus
ライナス

Leenus
リーナス

Biases in Morphological Analysis

- *Linus* /linnuss/ in ChaSen (Japanese Morphological Analyser)

  リーナス\は\、\この\よう\な\こと\を\信じ\て\いる

  × ライ\ナス\は\この\よう\な\こと\を\信じ\て\いる

- Which is more common?

Word	Goo	Google	Comment
ライナス	18,200	259,000	The character in Peanuts
リーナス	7,300	101,000	ChaSen developers are Computer users

⇒ We need to add ライナス to ChaSen’s dictionary

Word Segmentation — Why

Assistant: *Can you stick around? I need supervision for this.*

Manager: *Really? I'd rather have super hearing than super vision. (pause) Oh, you meant "supervision," didn't you?*

Assistant: *Do you think someone else could help me with this?*

`http://www.overheardintheoffice.com/archives/010378.html`

supervision, supervising, superintendence, oversight – management by overseeing the performance or operation of a person or group (WordNet).

super power — supernatural power exhibited by super heroes and super villains; super vision, super strength, spider sense . . .

Word Separation

- Non separated text
  Japanese, Chinese, Thai, Old English

- Speech recognition output
  recognize speech vs *wreck a nice peach* (Robert Dale)

Word Separation Example (1)

- How many ways can we separate 森永前日銀総裁?

  rin ei zen hi gin sou sai
  mori ei zen hi gin sou sai
  rin ei mae hi gin sou sai
  mori ei mae hi gin sou sai
  rin ei zen nichi gin sou sai
  mori ei zen nichi gin sou sai
  rin ei mae nichi gin sou sai
  mori ei mae nichi gin sou sai
  morinaga zen hi sou sai
  morinaga mae nichi gin sou sai
  . . .

- Far too many

Word Separation Example (2)

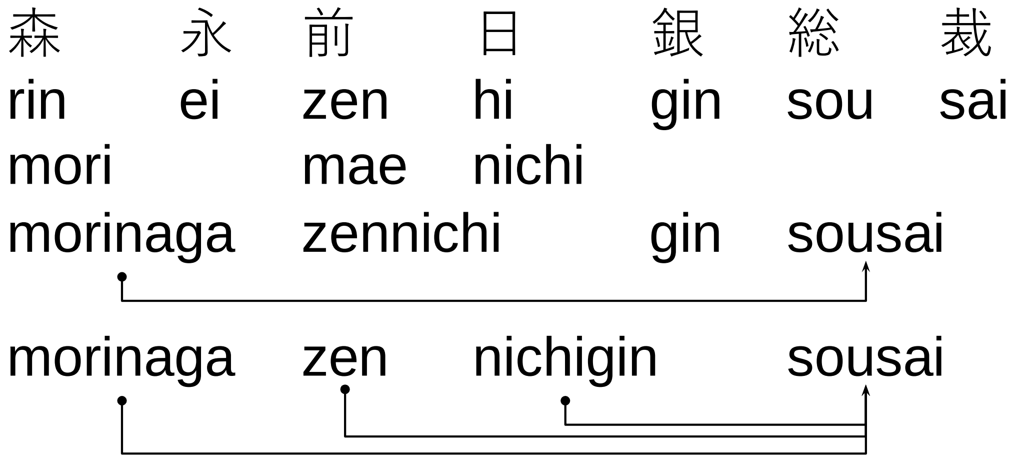
- The ambiguity is local
- Pack the results in a lattice:

<table>
<tr><td>森</td><td>永</td><td>前</td><td>日</td><td>銀</td><td>総</td><td>裁</td></tr>
<tr><td>rin</td><td>ei</td><td>zen</td><td>hi</td><td>gin</td><td>sou</td><td>sai</td></tr>
<tr><td>mori</td><td></td><td>mae</td><td>nichi</td><td></td><td></td><td></td></tr>
<tr><td colspan="2">morinaga</td><td colspan="2">zennichi</td><td></td><td colspan="2">sousai</td></tr>
<tr><td></td><td></td><td></td><td colspan="2">nichigin</td><td></td><td></td></tr>
</table>

- Any path that covers the whole input is a possibility

Word Separation Example (3)

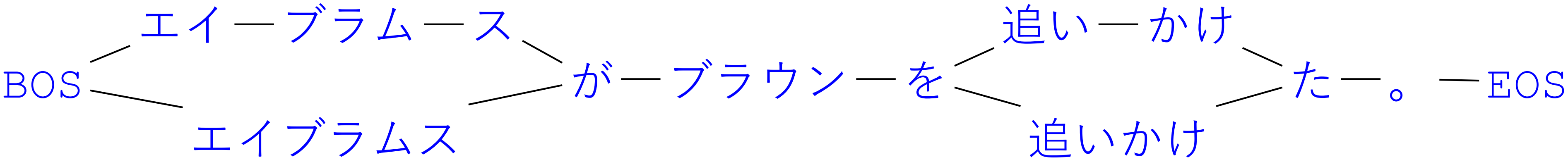
- How do we choose the best path?
  - Shortest number of words
    - Pick the path with the least number of words
  - Most common words
    - Prefer frequently occurring words
  - Word-word/pos-pos/class-class coherence weight
    - How far away should we look? (normally ±2)



森	永	前	日	銀	総	裁
rin	ei	zen	hi	gin	sou	sai
mori		mae	nichi			
morinaga		zennichi		gin	sousai	
morinaga		zen	nichigin		sousai	

森永	前	日銀	総裁
Morinaga	former	Bank of Japan	President

A more general approach — keep ambiguity


BOS

エイ

ブラム

ス

エイブラムス

が

ブラウン

を

追い

かけ

追いかけ

た

。

EOS

- ➢ Try parsing the n-best paths
- ➢ Try parsing all paths within probability β of the best path
- ➢ Try parsing all paths within probability β of the best path, increasing β until we find a parse (adaptive super-tagging)
- ➢ Parsing can look further away, so can disambiguate better
  - ➢ Keeping only the best paths reduces ambiguity and makes parsing faster
  - ➢ Speed increases of up to 20 times faster!

Lemmatization

- Once you have words
  - You may want to look them up in a dictionary
  - To look up a word in a dictionary
    - need either its canonical form
    - or a very big dictionary
- Lemmatization is the process of finding the canonical form

(15) *itta*

iku/iu/iru+ta

行く/た or 言う/た or いる/た

(16) 行った *itta/okonatta*

iku/okonau+ta

行く/た or 行う/た

(17) *dogs* → ***dog***+pl (sometimes just the stem)

(18) *mice* → ***mouse***+pl

(19) *went* → ***go***+ed

(20) *saw* → ***see***+ed “perceive”

(21) *saw* → ***saw*** “cut”

Morphological Processes

- Two possible mappings
  - Derivation (lexeme → lexeme) non-productive — don’t do
  - Inflection (lexeme → word) may be irregular
- Various Processes
  - Prefix: *un+happy*
  - Suffix: *cat+s*
  - Circumfix: *ke+raja+an* (*raja* “king” → *kerajaan* “kingdom”)
  - Infix: *g+er+igi* (*gigi* “teeth” → *gerigi* “toothed blade”)
  - Reduplicaton: *yama+yama* (*yama* “mountain” → *yamayama* “mountains”)
- English has relatively impoverished morphology

Kinds of Languages

Languages differ in how much they use morphology, and how much they use closed class words.

- Chinese has little morphology.
- English has some.
- Japanese has more.
- Turkish has a lot

Different approaches are needed for different languages.

Approaches to Lemmatization

- You must store all irregular forms
- You need rules for the rest (inflectional morphology)
- Rare words tend to be regular
  - For languages without much morphology, you can expand everything offline
- Most rules depend on the part-of-speech
  - So lemmatization is done with (or after) part-of-speech tagging

Inflectional Morphology: Verbs in English

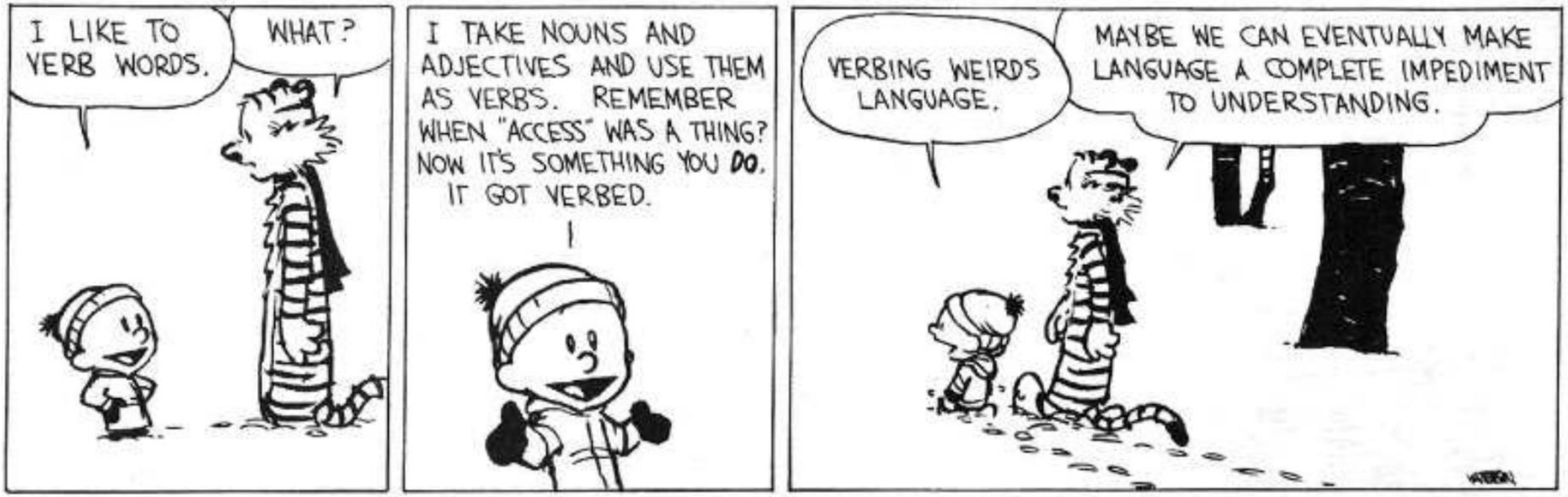
Form	want	take	put	find	Tag
present	want	take	put	find	VBP
3rd sg	wants	takes	puts	finds	VBZ
past	wanted	took	put	found	VBD
-en	wanted	taken	put	found	VBN
-ing	wanting	taking	putting	finding	VBG

- *want* is regular, most rare verbs are like this
- Can do with rules on the fly
  - Note ambiguity: *His wants are few*
  - Need to combine with POS tagging.

Derivational Morphology

- Change one word to another one
  - *happy* (a) → *unhappy* (a)
  - *venom* (n) → *antivenom* (n)
- Can change the POS
  - *canal* (n) → *canalize* (v)
  - *decide* (v) → *decision* (n)
- Sometimes without changing the word!
  - *address* (v) → *address* (n)
  - *turn* (v) → *turn* (n)

Verbing Weirds Language


I LIKE TO VERB WORDS.
WHAT?
I TAKE NOUNS AND ADJECTIVES AND USE THEM AS VERBS. REMEMBER WHEN "ACCESS" WAS A THING? NOW IT'S SOMETHING YOU DO. IT GOT VERBED.
VERBING WEIRDS LANGUAGE.
MAYBE WE CAN EVENTUALLY MAKE LANGUAGE A COMPLETE IMPEDIMENT TO UNDERSTANDING.
WATTERSON

Calvin and Hobbes

➢ If you allow zero derivations, then you get infinite loops
n → v → n . . .

➢ Trade-off between cover and efficiency

Tokenization

- Splitting words into tokens — the units needed for further parsing
  - Separating punctuation
  - Adding BOS/EOS (Beginning/Eng of sentence) markers
  - Splitting into stem+morph: *went* → *go*+ed
  - Normalization
    - *data base*
    - *data-base*
    - *database*
- This process is very task dependent
- Basically segmentation for already segmented languages

Parts of Speech I

- Grammatical Categories (word classes) that describe word usage

- Four main open-class categories

  Noun (N) heads a noun phrase, refers to things
  Verb (V) heads a verb phrase, refers to actions
  Adjective (A or J) modifies Nouns, refers to states or properties
  Adverb (R or A) modifies Verbs, refers to manner or degree

- Some closed sub-categories

  Pronoun (Pr) that refers to people or things in context (*I, you, her*)
  Auxilliary (Aux) adds information about a main verb (*was, will, have*)

- Different languages make different distinctions

Parts of Speech II

- Closed class categories vary more

  Preposition (P) *in, of* : links noun to verb (postposition)
  Conjunction (C) *and, because*: links like things
  Determiner (D) *the, this, a*: delimits noun's reference
  Interjection (Int) *Wow, um*:
  Number (CD) *three, 125*: counts things
  Classifier (CL) 匹 "animal-CL": classifies things
  . . .

- Multi-word Expressions used to extend classes
  - *in order to, with regards to* (complex prepositions)

POS tagging

- NLP systems use detailed POS tags
  - 30–300 is typical
- If you don't include syntactic information then 30-50
  - Penn Treebank tags (most common set) `http://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html`
  - 45 tags (including punctuation)

Penn Treebank Examples

Tag	Description	Tag	Description
NN	Noun, singular or mass	VB	Verb, base form
NNS	Noun, plural	VBD	Verb, past tense
NNP	Proper noun, singular	VBG	Verb, gerund or present participle
NNPS	Proper noun, plural	VBN	Verb, past participle
PRP	Personal pronoun	VBP	Verb, non-3rd person singular present
IN	Preposition	VBZ	Verb, 3rd person singular present
TO	*to*	.	Sentence Final punct (*.,?,!*)

- The tags include inflectional information
  - If you know the tag, you can find the lemma
- Some tags are very specialized: I/PRP wanted/VBD to/TO go/VB ./.

Why is it so hard?

➢ Large dictionaries cause unexpected problems

(22)	*I*	*saw*	*a*	*bench*
	N	V	D	N
	N	N	N	N

(23)	*machine*	*translation*	*evaluation*	*system*
	N	N	N	N

➢ Language is often extended: new words and new uses of old words

What can we do?

- Exploit knowledge about distribution
  - Markup (tag) a text (corpus) with part of speech
- With them, it suddenly looks easier
  - Just choose the most frequent tag for known words (*I* pronoun, *saw* verb, *a* article, . . .)
  - Make all unknown words proper nouns
  - This gives a baseline of 90% (for English)
- The upper bound is 97-99% (human agreement)
  - The last few percent are very hard

POS tagging methods

- Rule-based (now rare)
  - But still best results for English
- Statistical (simplest using just 2 previous words and tags)
  - Fast and cheap to build
- Other machine learning based
- All methods require expensive resources
  - Dictionaries of known words
  - Corpora tagged with parts of speech

Statistical Systems

- Learn rules automatically from tagged text
  - Many learning methods
  - Current popular learner is MIRA, before then CRF, before then SVM, . . .
  - Algorithms and CPU speeds are improving
- 96%+ accuracy using these features
  - Previous n words, (succeeding n words)
  - Previous n tags
  - Combinations of words and tags
  - Word Shape

Out of Vocabulary (OOV) words

- Unknown words are a big problem
  - Completely unknown words (not in lexicon)
  - Unknown uses of known words (derivation or lexicon gaps)
- Big, accurate lexicons are most useful!
- Otherwise guess from word shape (and context)
  - lowercase → common noun
  - uppercase → Proper noun
  - ends in *-ly* → adverb
  - ends in *-ing* and has vowel → verb
- You can learn these features

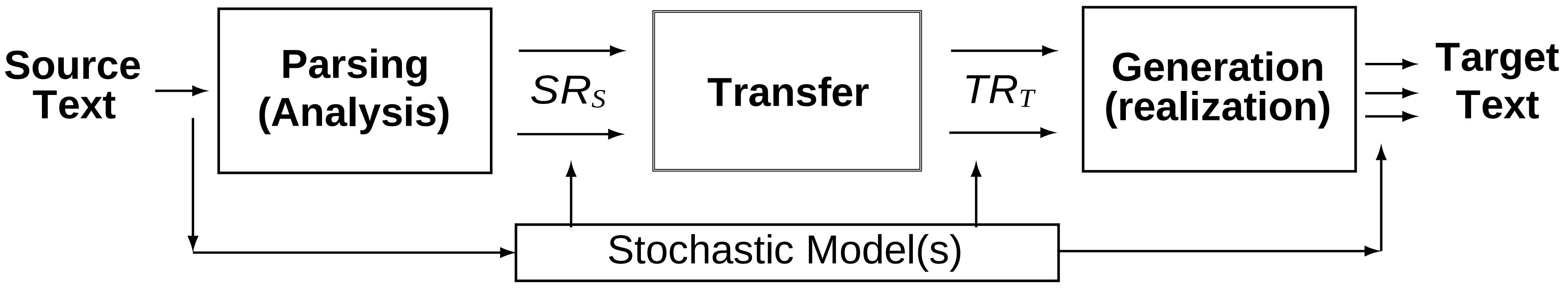
Representing ambiguities

- Two opposite needs:
  - Disambiguate early (leaving fewer choices)
    → Improve speed and efficiency
  - Disambiguate late (leaving more choices)
    → Can resolve ambiguities with more information
- Several Strategies:
  - Prune: Discard very low-ranking alternatives, but keep some
  - Under specify (keep ambiguity efficiently)
  - Pack information in a lattice (keep ambiguity efficiently)

Summary

- Morphological analysis is the analysis of units within the word
  - Segmentation: splitting text into words
  - Lemmatization: finding the base form
  - Tokenization: splitting text into tokens (for further processing)
  - Part of Speech Tagging: assigning POS to tokens
- State of the art is to tag a training corpus and then learn a classifier

Components of Transfer based MT



- Parse source text to source representation (SR)
- Transfer this to some target representation (TR) (next weeks)
- Generate target text from the TR
- If the source language = target language, then we are paraphrasing

Parsing

Parsing

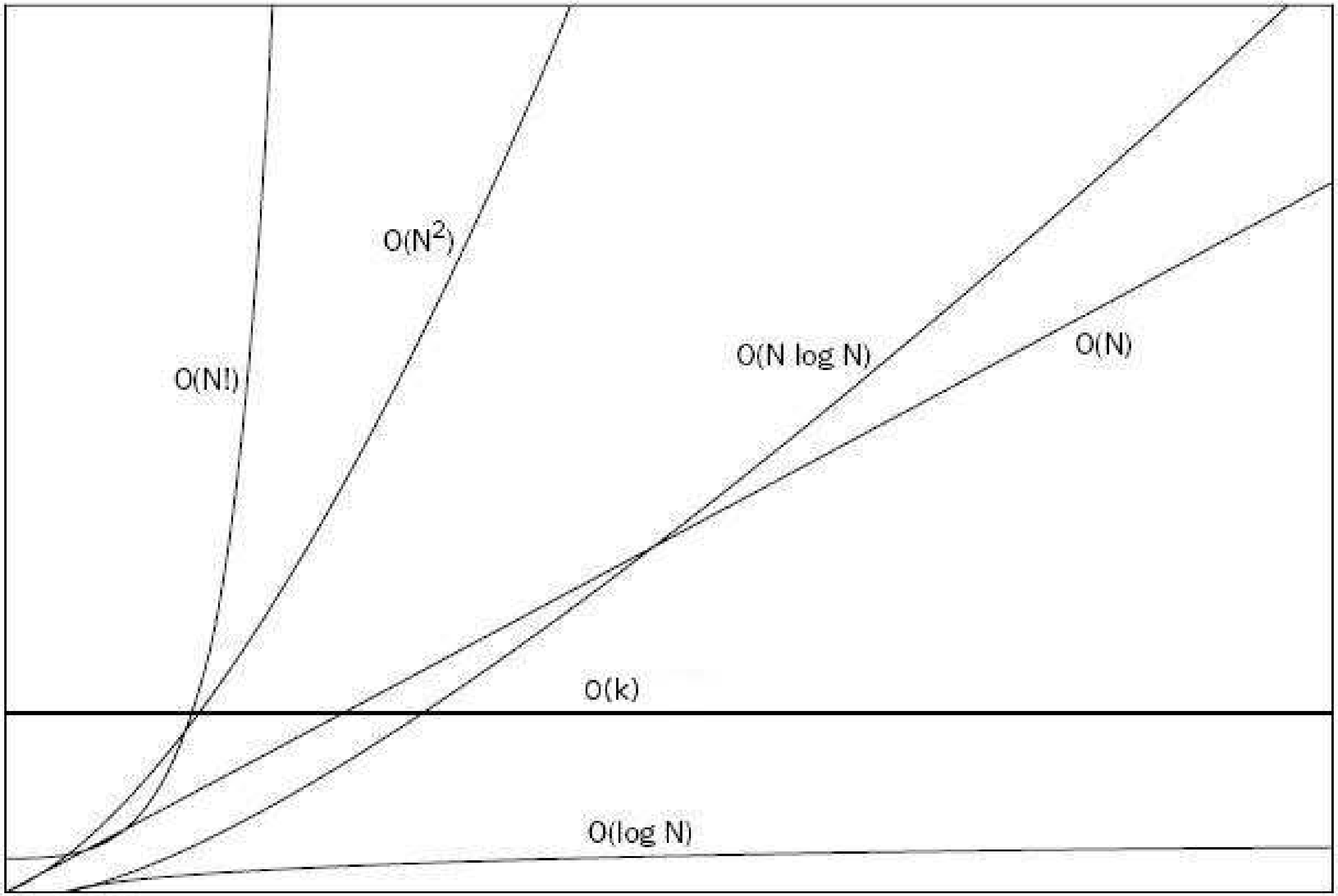
- Trade off between complexity/power and speed
  - Simple formalisms are faster
    - less useful output
    - may not parse some phenomena
  - Parsing to full meaning representations is slower
    - more useful output
    - harder to write rules

Algorithmic Complexity and Big-O Notation

- We measure the efficiency of an algorithm using big-O notation
- $f(n) = O(g(x))$ iff $f(n) < M|g(x)|$ forall $x > x_0$
  - describes the behaviour of a function for big numbers.
- We want the function $g(x)$ to grow as slowly as possible

Function	Name	Example
$O(0)$	Constant	hash lookup
$O(n)$	Linear	parsing n sentences
$O(n^2)$	Quadratic	dependency parsing n word sentence
$O(n^3)$	Cubic	LR parsing n word sentence
$O(n^c)$	Polynomial of degree c	HPSG parsing n word sentence ($c = 5$)
$O(c^n)$	Exponential	ambiguity in n word sentence
$O(n!)$	Factorial	

Comparing Growth of Functions



Comparison of different orders of complexity.

Why it is important

- What happens when n goes to $10n$

Function	Name	Speed
$O(0)$	Constant	no change
$O(n)$	Linear	10 times slower
$O(n^2)$	Quadratic	100 times slower
$O(n^3)$	Cubic	1000 times slower
$O(n^c)$	Polynomial of degree c	10^n times slower
$O(c^n)$	Exponential	c^{10} times slower
		(c = 10 → 10,000,0000,000)
$O(n!)$	Factorial	even slower

- 100 word sentences are not that rare

Structural Ambiguity

- You get structural ambiguity with any long sentence
  - *While hunting in Africa, I shot an elephant in my pajamas. How an elephant got into my pajamas I'll never know.*
    Groucho Marx movie, Animal Crackers (1930)
- Lexical ambiguity makes this worse
  - *People saw her duck.*

The problem: assign a structure to a sentence

We get words

People saw her duck.

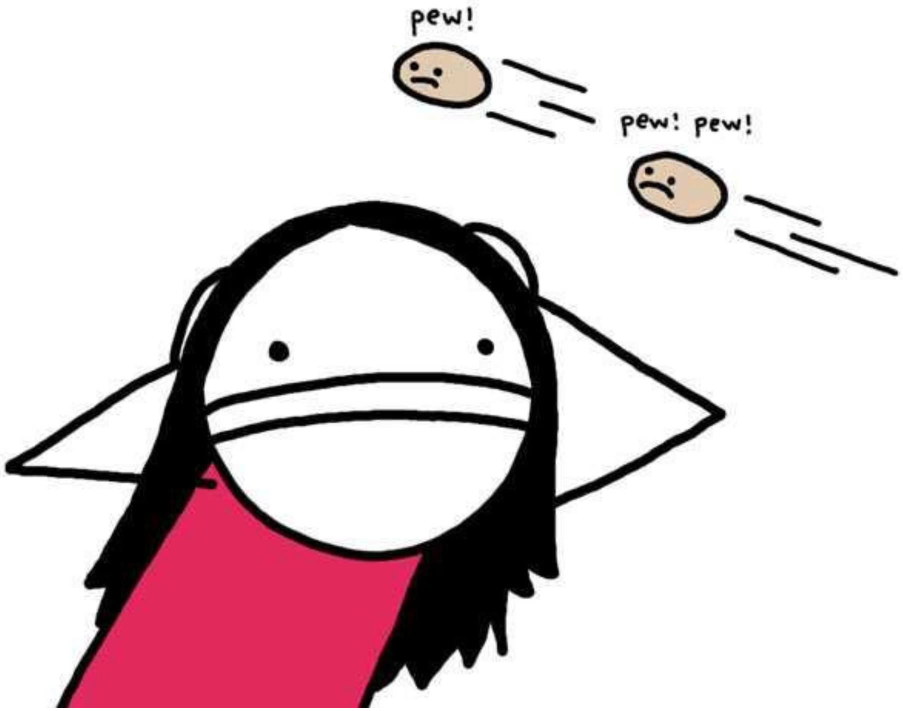
We want meaning
Or at least syntax

http://www.animaltalk.us/for/Animals/fw-cute-picture-of-your-daughter-with-duck/

People saw her duck

People saw the duck belonging to her.

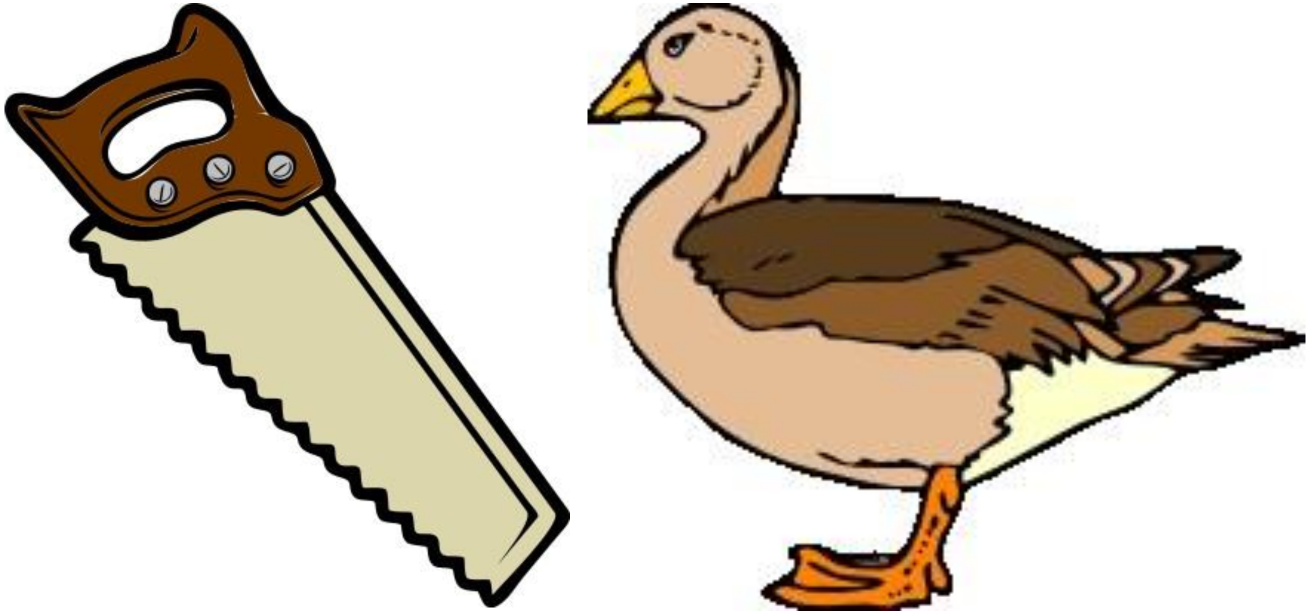
People saw her duck



People saw that she ducks.

http://www.nataliedee.com/012109/ducking-incoming-balls.jpg

People saw her duck

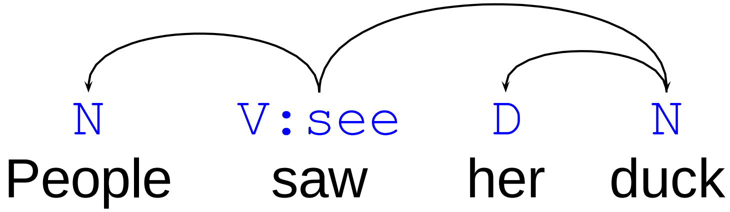


People cut her duck with a saw.

OpenClipArtLibrary

Dependency Parsing

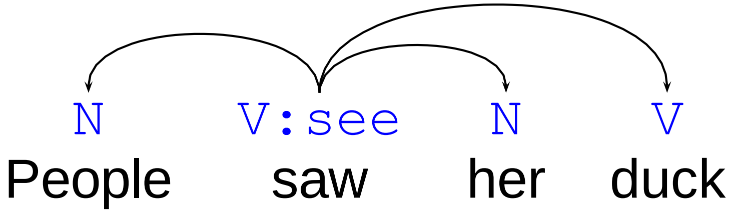
(1)



N V:see D N

People saw her duck

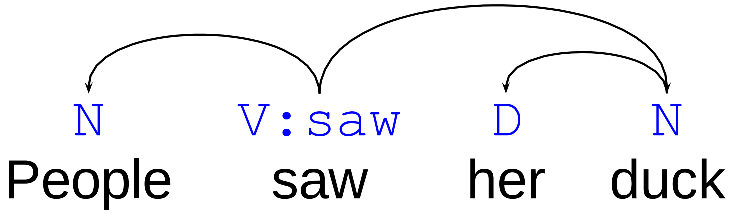
(2)



N V:see N V

People saw her duck

(3)



N V:saw D N

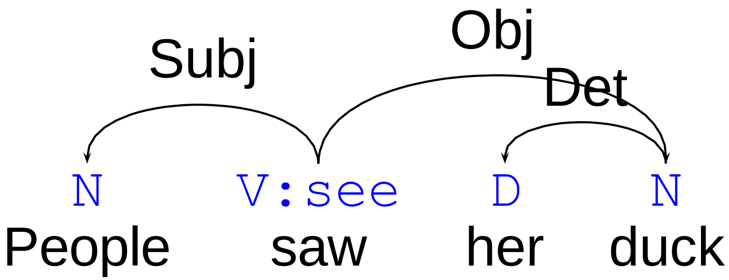
People saw her duck

Directly link words to words (Head (main word) to Dependents).

Dependency Parsing

- Complexity: $O(n^2)$ (efficient)
- Cannot handle
  - Control structures: *I want to go* (*I go*)
  - Coordination: *A and B are both OK*
- Can combine with labeled arcs (Semantic Role Labeling)

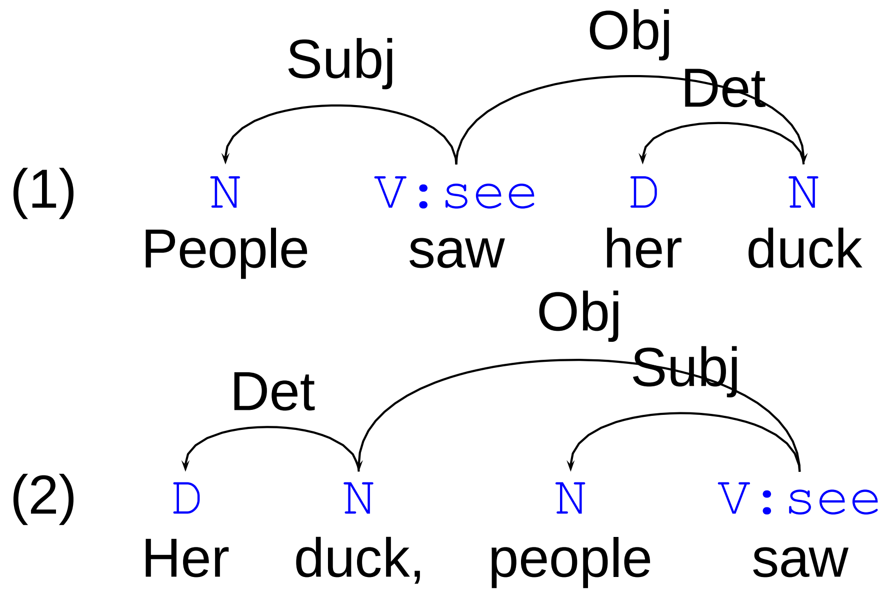
(1)



Dependency Parsing in action

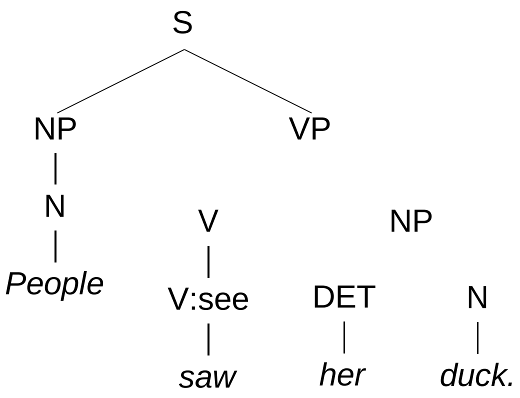
- Most popular for text mining
  - Efficient time means it can be run on large text samples
  - Dependency relations are enough for relational extraction

- Popular for free word order languages

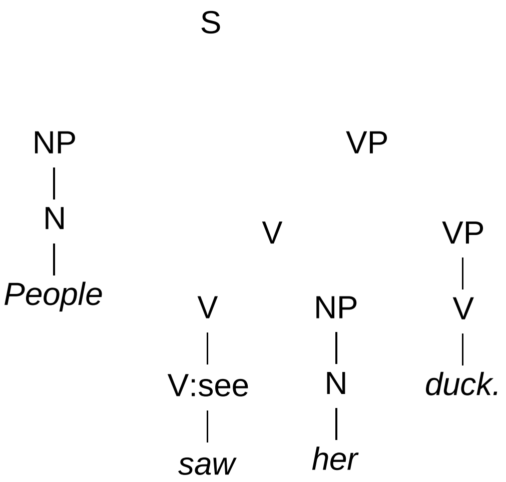


Syntax (Phrase Structure Grammars)

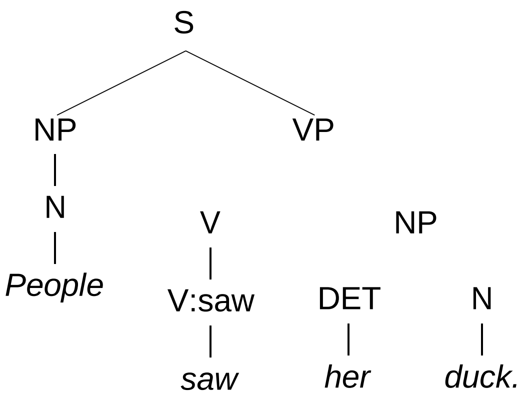
(24)



(25)



(26)



Phrase Structure Grammars (Context Free Grammars)

- Add intermediate nodes: Noun Phrase (NP), Verb Phrase (VP), . . .
- Complexity: $O(n^3)$ (polynomial)
- Cannot handle
  - Control structures: *I want to go*
  - Crossed structures: found in e.g., Swiss German
- Still needs more work to give full semantics
  - Add in semantics (LFG, HPSG: → $O(n^6)$)

An example grammar

```
S -> NP VP
PP -> P NP
NP -> Det N | Det N PP | 'I' | 'people'
VP -> V NP | VP PP
Det -> 'an' | 'my' | 'her'
N -> 'elephant' | 'pajamas' | 'duck'
V -> 'shot'
P -> 'in'
```

- Root node (start symbol): S
- Leaf nodes: 'I', 'an', 'my', . . .
- Pre-terminals: part of speech labels (POS)

`nltk.app.rdparser(),nltk.app.srparser()`

How to build the trees (with demo)

- ➢ Start at the top and work down: (top-down)
  - ➢ Recursive Descent Parsing: limited, inefficient ($O(c^n)$)
- ➢ Start at the top and work down: (bottom up)
  - ➢ Shift Reduce Parser: needs back tracking ($O(n^c)$)
- ➢ Remember intermediate structures:
  - ➢ Chart Parsing: currently the best! ($O(n^3)$)
- ➢ Many tweaks are possible to get more efficiency
  - ➢ hyper-active parsing, packing, . . .

Recursive Descent Parsing

- Start at the root node (start symbol)
- Break things down into parts using the rules
- Keep going until things match at the leaves
- If you don’t find the solution then backtrack
- Cons
  - NP –> NP PP goes into an infinite loop
  - Many unused structures created

Shift Reduce Parser

- Try to find sequences of words that match the right hand side of rule
- Replace them with the left-hand side
- Until the whole sentence is reduced to an S
  - **Shift** a word onto the stack
  - **Reduce** two things using a rule
  - Stop when all words are shifted and the last rule gives S
- Cons
  - Can fail to find a parse
- Pros
  - Only builds things that may be needed

Chart Parser

- Store partial solutions in a chart — no backtracking
  - dynamic programming approach by Martin Kay
- Pros
  - Only builds the structures you need
- Cons
  - Only works with binary rules
- This is the most common form of parser for phrase structure grammars

Stochastic Parsing

- We don’t want all trees, just the best one
  - Probabilistic Context Free Grammars: PCFG
- Create all trees and rank them
- Create only the most probable trees
  - prune low probabilities
  - may lose good trees
- Currently produce the top 1,000 or so and then rerank
- 90%+ accuracies now available (Charniak, Collins, etc)

An example PCFG grammar

```
S -> NP VP
PP -> P NP
NP -> Det N (0.6) | Det N PP (0.2) | 'I' (0.1)| 'people' (0.1)
VP -> V NP (0.4) | VP PP (0.6)
Det -> 'an' (0.8) | 'my' (0.15) | 'her' (0.05)
N -> 'elephant' (0.001) | 'pajamas' (0.001) |  ...
V -> 'shot'
P -> 'in'
```

➢ The probability of a sentence is the product of the probabilities of the rules

➢ With aggressive pruning you can get close to linear complexity

Interactive Parsing

- Ask someone about ambiguities
  typically the author
- Accurate
- Provides data for learning
- Slow and labor intensive
  - Mainly used for building treebanks

Incremental Parsing

- Build parse as words arrive
- Even look ahead!
- Important for speech recognition
- Psychologically realistic

Chunking (Shallow Parsing)

- Before parsing, identify phrases
  - Especially useful for proper names and unknown words

- Identify idioms
  Complex prepositions
  Complex determiners

- Don’t build structure for *New York* , *in order to*, . . .
  but: *Queensland and Melbourne Universities*

- Often more robust than a full parse

Tense, Aspect and Modality

- Not considered by syntactic parsers
  - but can be got from morphology
  - take input as *go ed* not *went*
- Normally treated as a feature on clauses
- Often determined after the initial parse
- Hard to do with no context

Discourse Structure

- Often neglected by parsers
- Needed for
  - Anaphor resolution
    e.g., Zero pronouns
  - Recovery of ellipsis
  - Look ahead and prediction
  - Text understanding

Constraint based approaches

- Use one mechanism for many levels: LFG, HPSG
- Often treat syntax/semantics/pragmatics
- Inherently slow, but can be sped up
  - Parallel processing
    Tsuji (Tokyo)
  - Branch and bound searches
    Beale (CRL, New Mexico)
  - Packed Representations
    Oepen (CSLI, Stanford)
  - Super-tagging Curran (Edinburgh), Miyamoto (Tokyo), . . .

HPSG semantics for *People saw her duck*

(1)

□ h_3:udef_q(x_5{PERS *3*, NUM *pl*, IND +}, h_4, h_6),

□ h_7:_people_n_of(x_5, i_8),

h_9:_see_v_1(e_2{SF *prop*, TENSE *past*, }, x_5, x_{10}{PERS *3*, NUM *sg*}),

h_{11}:def_explicit_q(x_{10}, h_{13}, h_{12}),

$(h_1,$

h_{14}:poss(e_{16}{SF *prop*, TENSE *untensed*, }, x_{10}, x_{15}),

□ h_{17}:pronoun_q(x_{15}{PERS *3*, NUM *sg*, GEND *f* , PRONTYPE *std_pron*}, h_{18}, h_{19}),

h_{20}:pron(x_{15}),

□

h_{14}:_duck_n_1(x_{10})

$\{h_{18} =_q h_{20}, h_{13} =_q h_{14}, h_4 =_q h_7\})$

People saw the duck belonging to her.

(2)

$$\left(h_1\,, \left\langle \begin{array}{l} h_3{:}\text{udef\_q}(x_5\{\textsc{pers}\ 3, \textsc{num}\ pl, \textsc{ind}\ +\}, h_4, h_6), \\ h_7{:}\text{\_people\_n\_of}(x_5, i_8), \\ h_9{:}\text{\_see\_v\_1}(e_2\{\textsc{sf}\ prop, \textsc{tense}\ past, \}, x_5, h_{10}), \\ h_{11}{:}\text{pron}(x_{12}\{\textsc{pers}\ 3, \textsc{num}\ sg, \textsc{gend}\ f\ , \textsc{prontype}\ std\ pron\}), \\ h_{13}{:}\text{pronoun\_q}(x_{12}, h_{14}, h_{15}), \\ h_{16}{:}\text{\_duck\_v\_1}(e_{17}\{\textsc{sf}\ prop, \textsc{tense}\ untensed, \}, x_{12}, p_{18}) \end{array} \right\rangle, \\ \{h_{14} =_q h_{11}, h_{10} =_q h_{16}, h_4 =_q h_7\}\right)$$

People saw that she ducks.

(3)

$(h_1,$

h_3:udef_q(x_5{PERS *3*, NUM *pl*, IND +}, h_4, h_6),

h_7:_people_n_of(x_5, i_8),

h_9:_saw_v_1(e_2{SF *prop*, TENSE *pres*, }, x_5, x_{10}{PERS *3*, NUM *sg*}),

h_{11}:def_explicit_q(x_{10}, h_{13}, h_{12}),

h_{14}:poss(e_{16}{SF *prop*, TENSE *untensed*, }, x_{10}, x_{15}),

h_{17}:pronoun_q(x_{15}{PERS *3*, NUM *sg*, GEND *f* , PRONTYPE *std_pron*}, h_{18}, h_{19}),

h_{20}:pron(x_{15}),

h_{14}:_duck_n_1(x_{10})

$\{h_{18} =_q h_{20}, h_{13} =_q h_{14}, h_4 =_q h_7\})$

People cut her duck with a saw.

Parsing Complexity in the real world

- Number of parses $\approx n^{3-5}$
  - Highly constrained grammar (HPSG) $\approx n^3$
  - CFG learned from corpus (PCFG) $\approx n^5$
- Long sentences dominate processing
  - 10 word sentence will have 1,000–100,000 parses
  - 25 word sentence (average for newspaper) 15,625–976,5625
  - 40 word sentence (long for newspaper) 64,000–102,400,000
- Many systems ignore $n > 40$

Current Mainstream Approach to Parsing

1. Build a rough grammar
2. Parse a corpus
3. Hand correct the parse trees in the corpus (expensive)
4. Learn a new grammar from the corpus OR fix the grammar
5. Learn a ranking model from the corpus

Generation

Generation (Big Picture)

- You need to decide What-to-say (discourse level)
  - May produce different amounts of detail for beginner vs expert
  - Can personalize information (e.g. health reports)
- Then you decide How-to-say (sentence level)
  - You also need to look across sentences for, e.g., pronouns
  - The same referent can have multiple realizations
    - *Nanyang Technological University*
    - *NTU*
    - *my university*
    - *the university in Jurong*
    - . . .

Generation: Applications

- Natural-language front ends used to present
  - information in databases etc.
  - weather forecasts, train systems,
  - (personalized) museum/restaurant/shopping guides, . . .
- In dialog systems
- In summarization systems
- In authoring aids to help people create routine documents: customer support, job ads, etc
- Machine Translation

Generation: Process

- *Goal*
- Text Planner → *Text Plan* (What-to-say)
- Sentence Planner → *Sentence Plan*
- Linguistic Realizer → *Surface Text* (How-to-say)

Generation: Realization

- Take an abstract representation and produce a string

- The opposite of parsing

- Underspecified input produces multiple strings: (paraphrasing)

  - *It follows from this that the company is not responsible for the accident.*

→ *It follows that the company isn't responsible for the accident from this.*

→ *It follows that the company is not responsible for the accident from this.*

→ *That the company isn't responsible for the accident follows from this.*

Generation in Machine Translation

- What-to-Say decided, but not complete and may contain errors
- How-to-Say it is the problem
- Discourse information not explicit
- Sentence planning is difficult
- As a result, non-cohesive text is common

Handling underspecified input

- Underspecified input has two sources
  - Incompletely analysed input
  - Transfer mismatches

    葡萄 を 食べた → *I ate grape* ??? *I ate grapes*

- Need Encyclopedic knowledge to choose
  - Use domain, genre and register knowledge
  - Encode frequency knowledge

- Need to fail gracefully

Graceful defaults

- AI systems always have incomplete knowledge
  - as indeed do humans
- Should always provide reasonable output
  - transliterate unknown words
  - deduce missing elements
  - use as much information as possible
- For post-edited MT it is also important to mark iffy output
  - supplemented pronouns, unknown words
- The importance of defaults has given rise to statistical models

Summary

- Parsing
  - Words to representation
- Generation
  - Representation to words
- Two main syntactic representations:
  - Dependencies (word-to-word)
    efficient, cannot represent all structures
  - Phrase Structure Trees (with phrasal nodes)
    cannot represent more structures

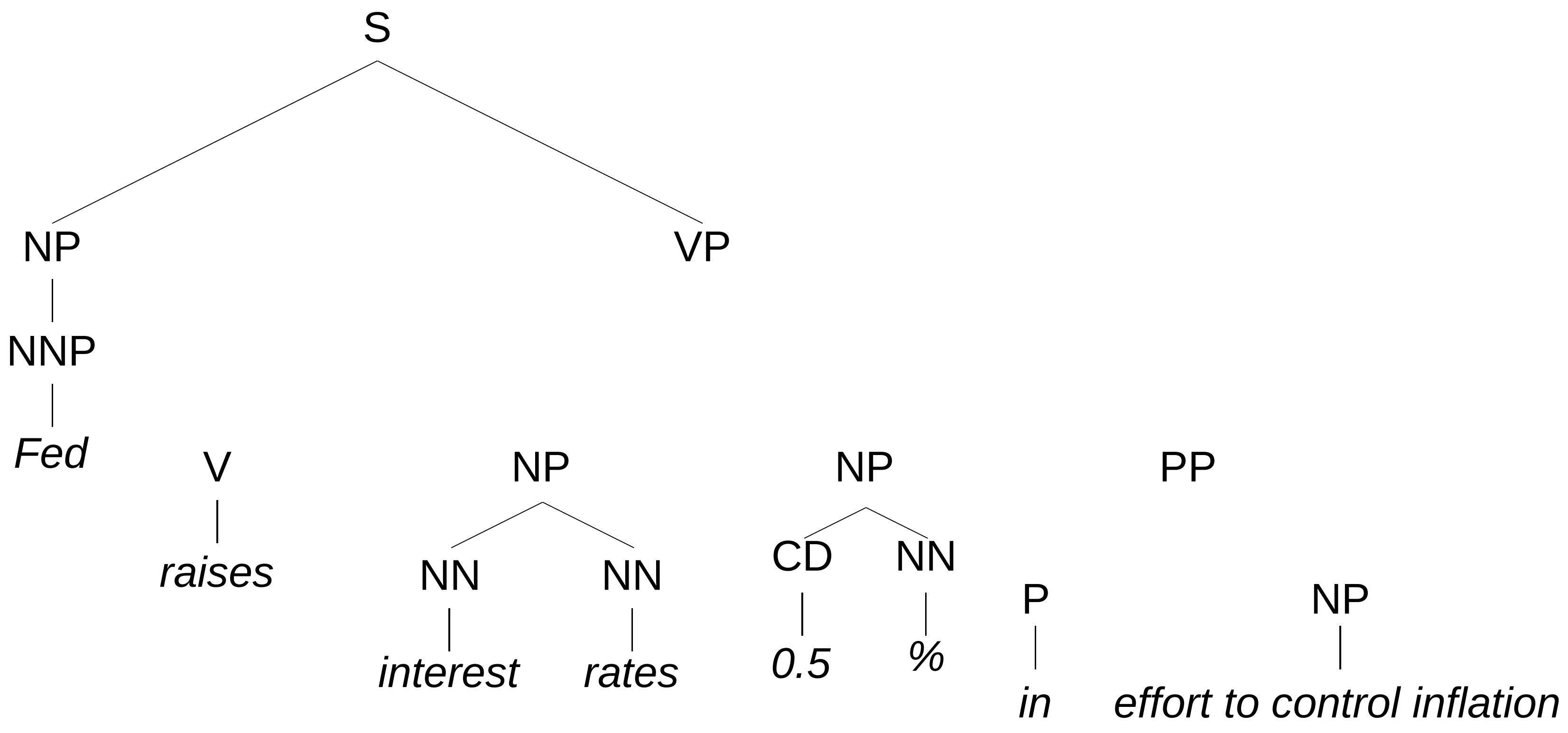
Efficiency is important

- Need to avoid exponential processing
- Least complex is best
  - constant $<$ linear $<$ polynomial $<$ exponential
- May sacrifice some accuracy for speed
  - prune low probability alternatives

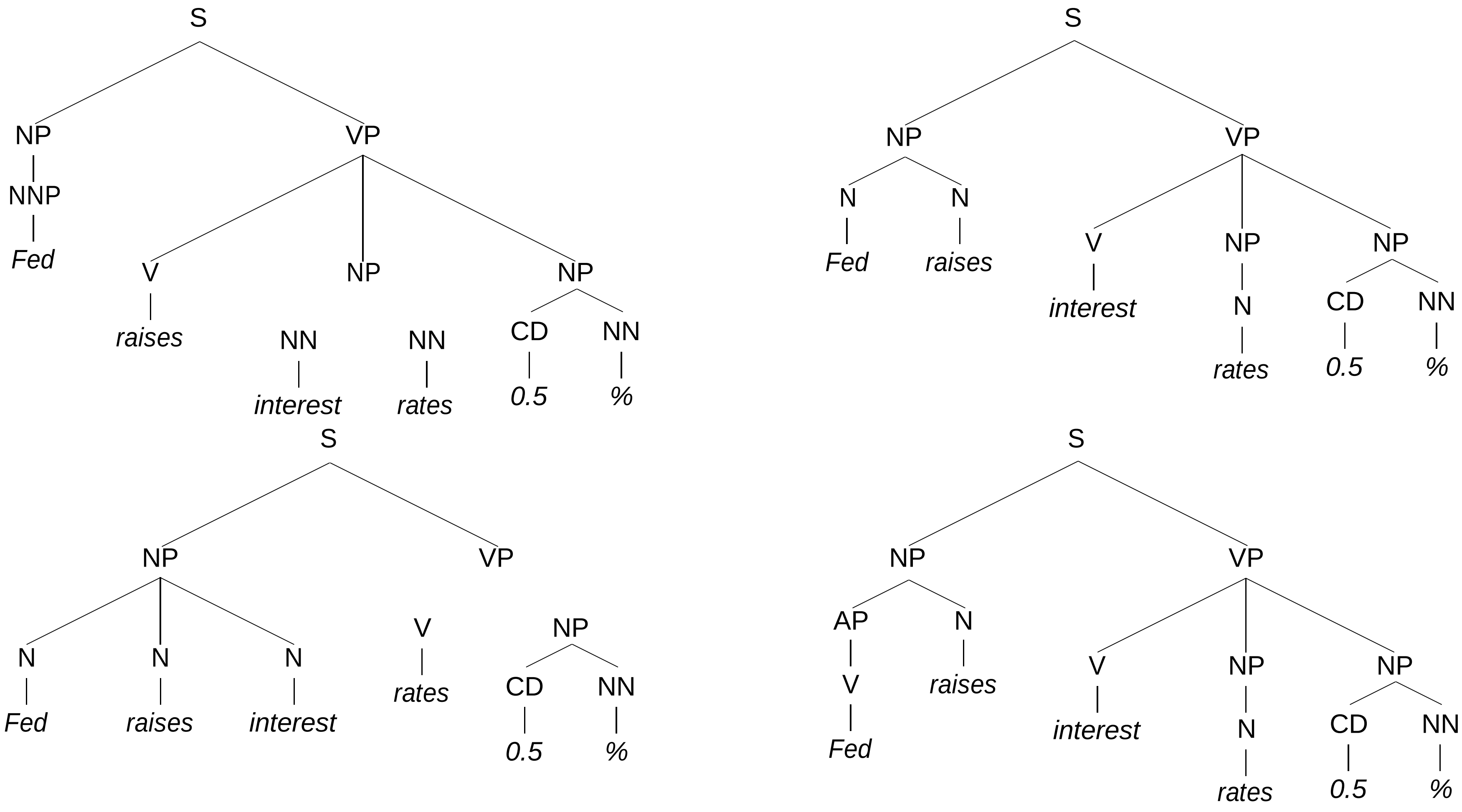
Inherent Ambiguity in Syntax

(24) Fed raises interest rates 0.5% in effort to control inflation

NY Times headline 17 May 2000



Some of the ambiguities? (33 parses!)



Approached to Machine Translation

Approaches to Machine Translation

Rule-based MT Find the meaning, translate it, generate it

Example-based MT See how this has been translated before, translate it the same way

Statistical MT Imagine that the text has been encoded in some way, try to decode it

EBMT Basic Philosophy

“Man does not translate a simple sentence by doing deep linguistic analysis, rather, man does translation, first, by properly decomposing an input sentence into certain fragmental phrases, and finally by properly composing these fragmental translations into one long sentence. The translation of each fragmental phrase will be done by the analogy translation principle with proper examples as its reference.”

Makoto Nagao (1984)

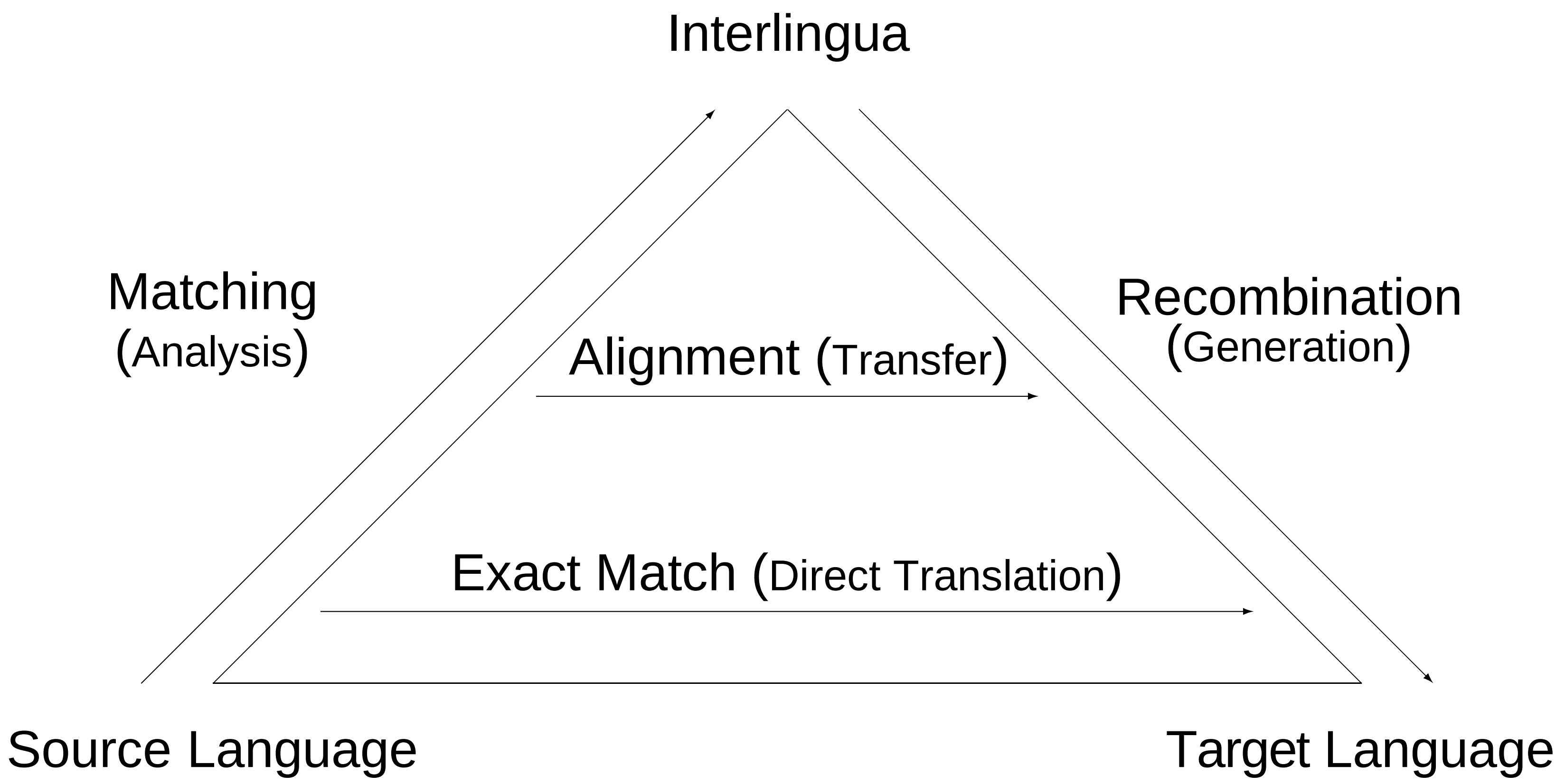
EBMT Method

- When translating, reuse existing knowledge:

  0. Compile and align a database of examples
  1. Match input to a database of translation examples
  2. Identify corresponding translation fragments
  3. Recombine fragments into target text

- Example:
  - Input: He buys a book on international politics
  - Data:
    - He buys a notebook – Kare wa noto o kau
    - I read a book on international politics – Watashi wa kokusai seiji nitsuite kakareta hon o yomu
  - Output: Kare wa kokusai seiji nitsuite kakareta hon o kau

EBMT ‘Pyramid’



H. Somers, 2003, “An Overview of EBMT”

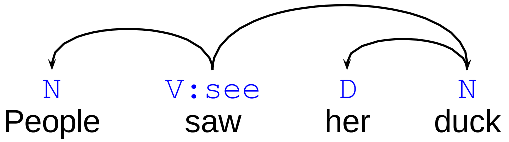
Example-based Translation: Advantages/Disadvantages

- Advantages
  - Correspondences can be found from raw data
  - Examples give well structured output if the match is big enough
- Disadvantages
  - Lack of well aligned bitexts
  - Generated text tends to be incohesive

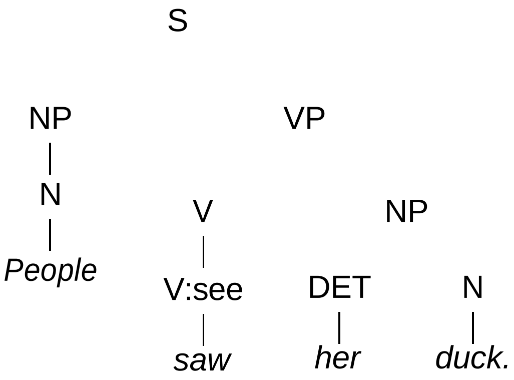
Open Questions

- Representation of examples:
  - Which representation should be used for examples?
    - *n*-grams

      `people saw her`; `saw her duck`; `her duck .`
    - Dependencies



    - Phrase-structure trees



  - How much information do we need encoded in examples? (lemmas, morphological, syntactic, semantic, ...)

- How big should the translation unit be:
  - Large → better translation / less likely to be used
  - Small → better cover / leads to literal translations
  - Mixed: combine different sizes (best of both worlds)
- Score translation candidates:
  - heuristics
    - size of units
    - semantic closeness to input
  - probabilistic models (becomes SMT)

Early EBMT: ATR System (1991)

Experiments and Prospects of Example-based Machine Translation, Eiichiro Sumita and Hitoshi Iida. In 29th Annual Meeting of the Association for Computational Linguistics, 1991.

- Japanese-English translation: N_1 *no* N_2 “N_2 of N_1” problem
- When EBMT is better suited than Rule-based MT

Translating “N_1 no N_2”

- の *no* “ADN” is an adnominal particle
  - Variants: での *deno*, までの *madeno*, . . .
- “N_1 no N_2” → “N_2 of/for/in/φ/’s N_1”

youka no gogo	The afternoon of the 8th
kaigi no mokuteki	The objective of the conference
kaigi no sankaryou	The application fee for the conference
isshukan no kyuka	A week’s holiday
kyouto deno kaigi	The conference in Kyoto
mittsu no hoteru	Three hotels

- Many different combinations

Difficult linguistic phenomena

- It is difficult to hand-craft linguistic rules for “N_1 no N_2”
  - Requires deep semantic analysis for each word
  - So remember examples, and then match the closest

    input *mikka no asa* “3rd of morning”

    match *youka no gogo* → “the afternoon of the 8th”

    - matching was done using an ontology
    - *3rd* ≈ *8th*
    - *morning* ≈ *afternoon*

    Translation “the morning of the 3rd”

Example of a larger match

Head switching between the English verb *like* into French *plaire*.

- Sam likes the new laser printer.
- La nouvelle imprimante à laser plaît à Sam.

Remember the whole pattern:

- X likes Y
- Y plaˆıt X.

When EBMT works better than Rule-based MT

- The translation rule is difficult to formulate
- General rule cannot accurately describe phenomena due to special cases (e.g. idioms)
- Translation cannot be made compositionally using only target words
- The sentence to be translated has a close match in the database.

EBMT Example System: Eureka

- A Hybrid Translation Method (Shirai, Bond and Takahashi, 1997)
  - Dynamically filters example sentences
  - Uses components of existing rule based systems

- General Approach

  1. Select a set of candidate sentences similar to the input sentence
  2. Select the most typical translation from the candidates’ translations
  3. Use this translation and its source as templates
     to translate the input sentence

- The closest source language match may not be the best translation!

(1) Select Similar Sentences: source

S_I *nikkei heikin 10 gatsu mono wa zokuraku .*
Nikkei average 10 month thing TOP continue-decline .
The Nikkei Average October contracts continued declining

(25) ***nikkei heikin*** *9* ***gatsu mono wa zokuraku*** *.*
Nikkei average 9 month thing TOP continue-decline .
The Nikkei Average September contracts were lower.

(26) ***nikkei*** *tentoo* ***heikin wa zokuraku*** *.*
Nikkei over-the-counter average TOP continue-decline .
The Nikkei over-the-counter average continued declining

(27) *8* ***gatsu mono wa zokuraku*** *.*
8 month thing TOP continue-decline .
August contracts continued declining.

(2) Identify the Most Similar Pair

S_I nikkei heikin 10 gatsu mono wa zokuraku .

S_1 nikkei heikin 9 gatsu mono wa zokuraku .
(7/8 shared segments = 0.875)

S_2 8 gatsu mono wa zokuraku .
(5/6 shared segments = 0.833)

S_3 nikkei-tentoo-heikin wa zokuraku .
(3/4 shared segments = 0.75)

Actual metric in Eureka considers:

- the number of matching characters
- the number of non-matching characters
- the number of continuous matching characters

(3) Recombination

1. Find differences

2. Choose constitutent

3. Replace differing constituent

4. Smooth the output

   - Number agreement
   - Filter adjuncts

S_I nikkei heikin 10 gatsu mono wa zokuraku .

S_t nikkei heikin 9 gatsu mono wa zokuraku .

T_t The Nikkei Average September contracts were lower.

T_I The Nikkei Average October contracts were lower.

Eureka is adaptive: also consider the translation

- When we select the translation fragment
  - Cluster candidates with similar translations
  - Choose the template from the largest cluster

T_1 The Nikkei$_2$ Average$_1$ September$_1$ contracts$_2$ were lower$_1$

T_2 August$_1$ contracts$_2$ continued$_2$ declining$_2$

T_3 The Nikkei$_2$ over-the-counter$_1$ average$_1$ continued$_2$ declining$_2$

- The source and target template are: The sentence pair in the best cluster with the highest similarity to the input sentence: T_2

- Weed out atypical (strange) translations

Recombination$'$

1. Find differences

2. Choose constitutent

3. Replace differing constituent

4. Smooth the output
   - Number agreement
   - Filter adjuncts

S_I nikkei heikin 10 gatsu mono wa zokuraku .

S_t 8 gatsu mono wa zokuraku .

T_t August contracts continued declining.

T_I The Nikkei Average October contracts continued declining.

General EBMT Issues:

- Sentence or sub-sentence?
  - Sentence:
    - Better quality translation
    - Boundaries are easy to determine
    - Harder to find a match
  - Sub-sentence:
    - Studies suggest this is how humans translate
  - Boundary friction
    - The handsome boy ate his breakfast ↔ Der schone Junge as seinen Fruhstuck
    - I saw the handsome boy ↔ Ich sah den schonen Jungen

General EBMT Issues: Suitability of Examples

- Some EBMT systems do not use raw corpus directly, but use manually-constructed examples or carefully-filtered set of real-world examples

- Real-world examples may contain:
  - Examples that mutually reinforce each other (overgeneration)
  - Examples that conflict
  - Examples that mislead the distance metric
    - Watashi wa kompyuta o kyoyosuru ↔ I share the use of a computer
    - Watashi wa kuruma o tsukau ↔ I use a car
    - Watashi wa dentaku o shiyosuru ↔ * I share the use of a calculator

General EBMT Issues: Matching

- String matching / IR-style matching
  - “This is shown as A in the diagram”
    ↔ “This is shown as B in the diagram”
  - “The large paper tray holds 400 sheets of paper”
    ↔ “The small paper tray holds 300 sheets of paper”
- Matching by meaning:
  - use thesaurus and distance based on semantic similarity
- Matching by structure:
  - Tree edit distance, etc.

General EBMT Issues: Alignment and Recombination

- Once a set of examples close to the input are found, we need to carry out:
  - Alignment:
    Identify which portion of the associated translation corresponds to input
  - Recombination:
    Stitch together these portions to create smooth output

State of the Art

- EBMT does best with well aligned data in a narrow domain
  - There are not so many domains with such data
- EBMT not used in commercial systems
- EBMT eclipsed by SMT in competitions
- Still a healthy research community
- EBMT and SMT converging
  - EBMT adds probablisitic models
  - SMT adds larger phrases

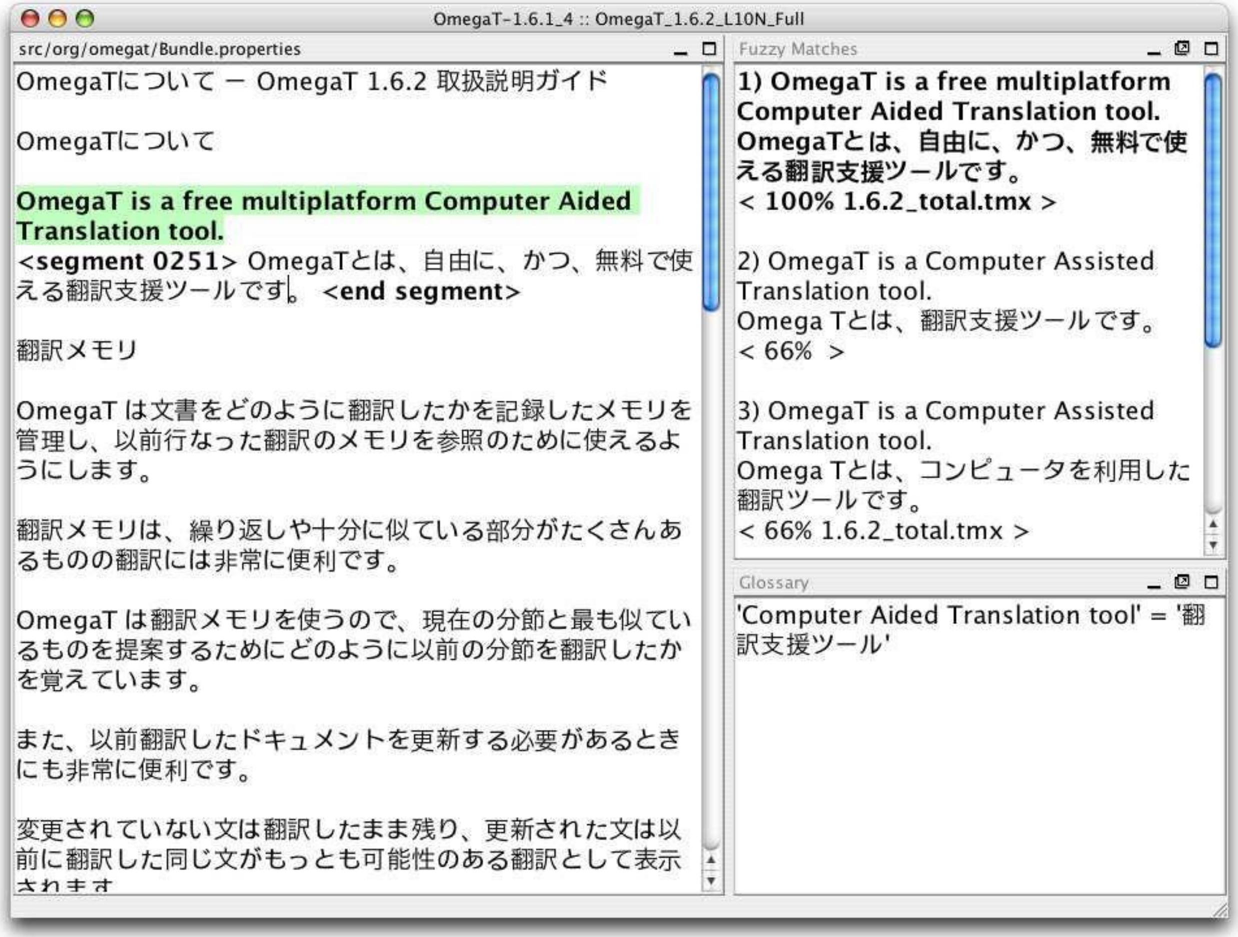
Translation Memories

- **Translation Memories** are aids for human translators
  - Store and index entire existing translations
  - Before translating new text
    - Check to see if you have translated it before
    - If so, reuse the original translation
- Checks tend to be very strict ⇒ translation is reliable
  - Identical except for white-space differences
- Now extended to **fuzzy** matching and replacing
  - Equivalent to EBMT
  - More flexible, greater cover, less reliable

Translation Memories

- TM are popular with translators
- Well integrated with word processors
- The translator is in control
- Translation companies can pool memories, giving them an advantage
- Simple solutions sell well
- Good tools also integrate
  - highlighted differences from closest match
  - dictionary look up
  - collocational look up (words in context)

TranslationMemory: OmegaT


OmegaT-1.6.1_4 :: OmegaT_1.6.2_L10N_Full
src/org/omegat/Bundle.properties
Fuzzy Matches
Glossary

OmegaTについて ー OmegaT 1.6.2 取扱説明ガイド

OmegaTについて

OmegaT is a free multiplatform Computer Aided Translation tool.
<segment 0251> OmegaTとは、自由に、かつ、無料で使える翻訳支援ツールです。 **<end segment>**

翻訳メモリ

OmegaT は文書をどのように翻訳したかを記録したメモリを管理し、以前行なった翻訳のメモリを参照のために使えるようにします。

翻訳メモリは、繰り返しや十分に似ている部分がたくさんあるものの翻訳には非常に便利です。

OmegaT は翻訳メモリを使うので、現在の分節と最も似ているものを提案するためにどのように以前の分節を翻訳したかを覚えています。

また、以前翻訳したドキュメントを更新する必要があるときにも非常に便利です。

変更されていない文は翻訳したまま残り、更新された文は以前に翻訳した同じ文がもっとも可能性のある翻訳として表示されます

1) OmegaT is a free multiplatform Computer Aided Translation tool.
OmegaTとは、自由に、かつ、無料で使える翻訳支援ツールです。
< 100% 1.6.2_total.tmx >

2) OmegaT is a Computer Assisted Translation tool.
Omega Tとは、翻訳支援ツールです。
< 66% >

3) OmegaT is a Computer Assisted Translation tool.
Omega Tとは、コンピュータを利用した翻訳ツールです。
< 66% 1.6.2_total.tmx >

'Computer Aided Translation tool' = '翻訳支援ツール'

EBMT/TM Summary

- Example-based Machine Translation (**EBMT**)

  0. Compile and align a database of examples
  1. Match input to a database of translation examples
  2. Identify corresponding translation fragments
  3. Recombine fragments into target text

- Translation Memories (**TM**)

  - Reuse complete existing translations
  - Keep the translator in control
  - TM: Popular with human translators

- Recent TMs also allow some variables, mainly numbers and nouns

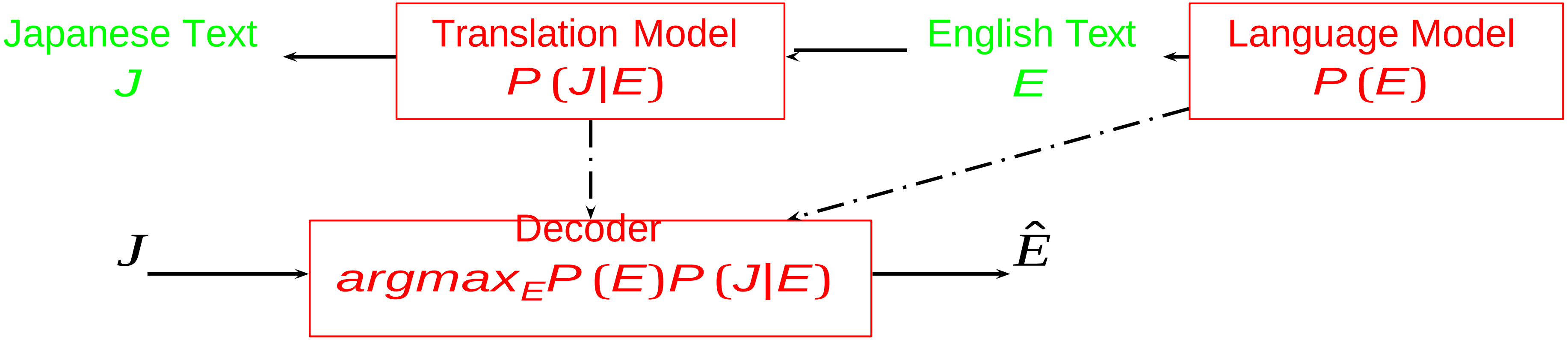
Statistical Machine Translation (SMT)

- Find the translation with the highest probability of being the best.
  - Probability based on existing translations (bitext)
- Balance two things:
  - Adequacy (how faithful the translation to the source)
  - Fluency (how natural is the translation)
- These are modeled by:
  - Translation Model: $P\,(T|S)$
    how likely is it that this translation matches the source
  - Language Model: $P\,(T)$
    how likely is it that this translation is good English
- Overall: $\hat{T} = argmax_T P\,(S|T) = argmax_T P\,(T|S)(T)$

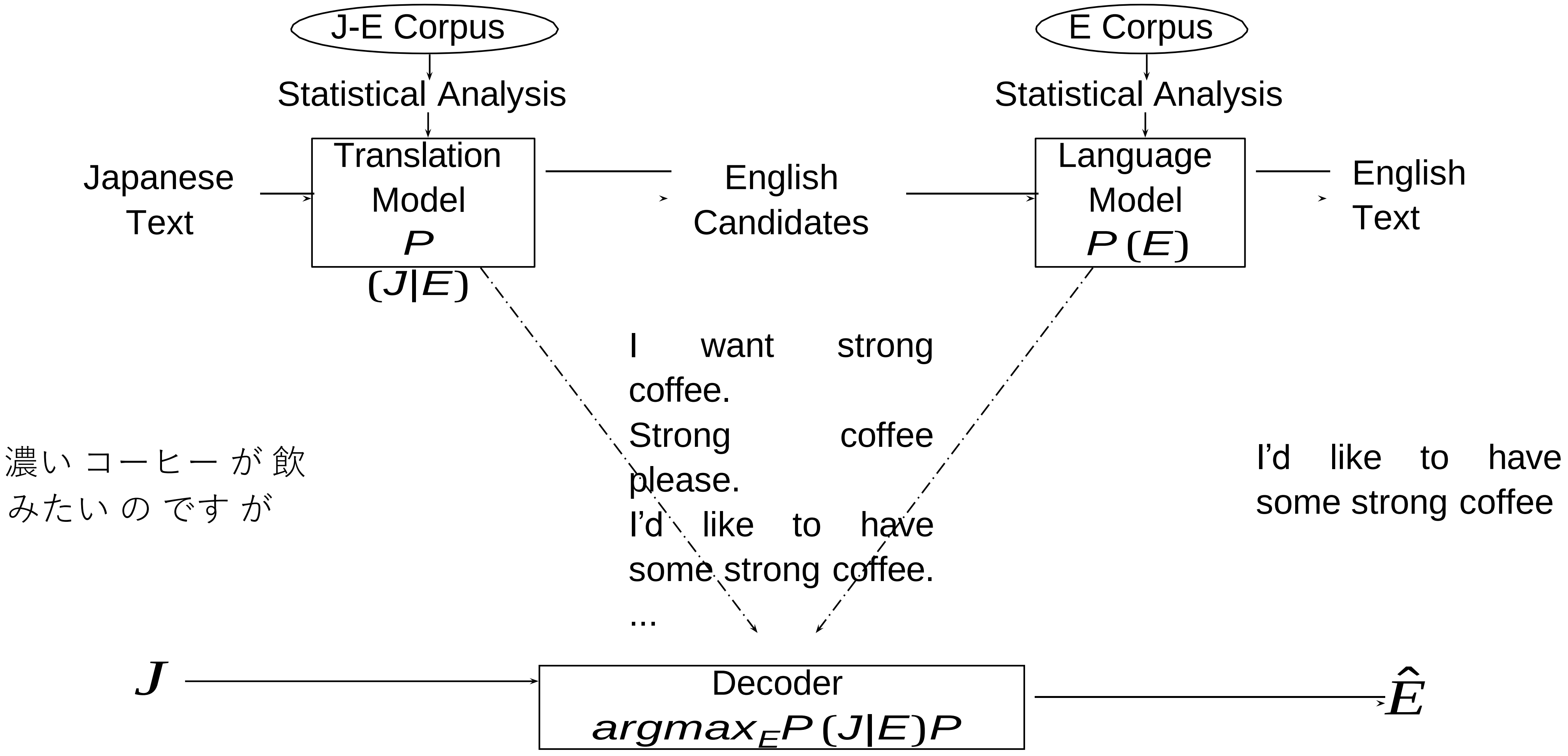
SMT: Basic Method

The basic method (Brown *et al* 1990): Choose the English Translation with the highest probability of translating the Japanese text.

$$\hat{E} = argmax_E P(E|J)$$



SMT Framework



J-E Corpus

Statistical Analysis

Translation Model $P(J|E)$

Japanese Text

English Candidates

E Corpus

Statistical Analysis

Language Model $P(E)$

English Text

濃い コーヒー が 飲
みたい の です が

I want strong coffee.
Strong coffee please.
I'd like to have some strong coffee.
...

I'd like to have some strong coffee

J

Decoder
$argmax_E P(J|E)P$

$\hat{E}$

Word Alignment

	show$_1$	me$_2$	the$_3$	one$_4$	in$_5$	the$_6$	window$_7$
ウィンドー$_1$	□	□	□	□	□	■	■
の$_2$	□	□	□	□	□	□	□
品物$_3$	□	□	□	■	□	□	□
を$_4$	□	□	□	□	□	□	□
見せ$_5$	■	■	□	□	□	□	□
てください$_6$	■	■	□	□	□	□	□

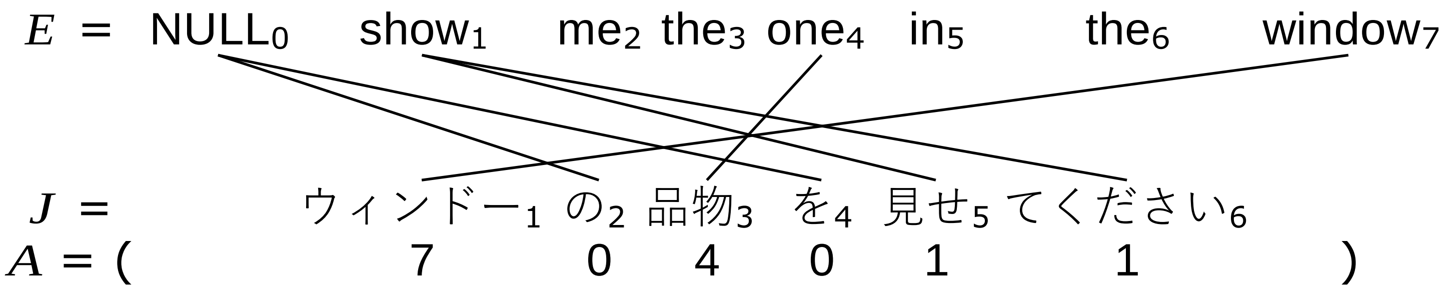
- Not all words align
  を$_4$ ACC, the$_3$, in$_5$

- Some alignments are unexpected
  てください$_1$ *te-kudasi* “please” → me$_2$

- Tend to be diagonal

Word Alignment

- Another way of looking at it



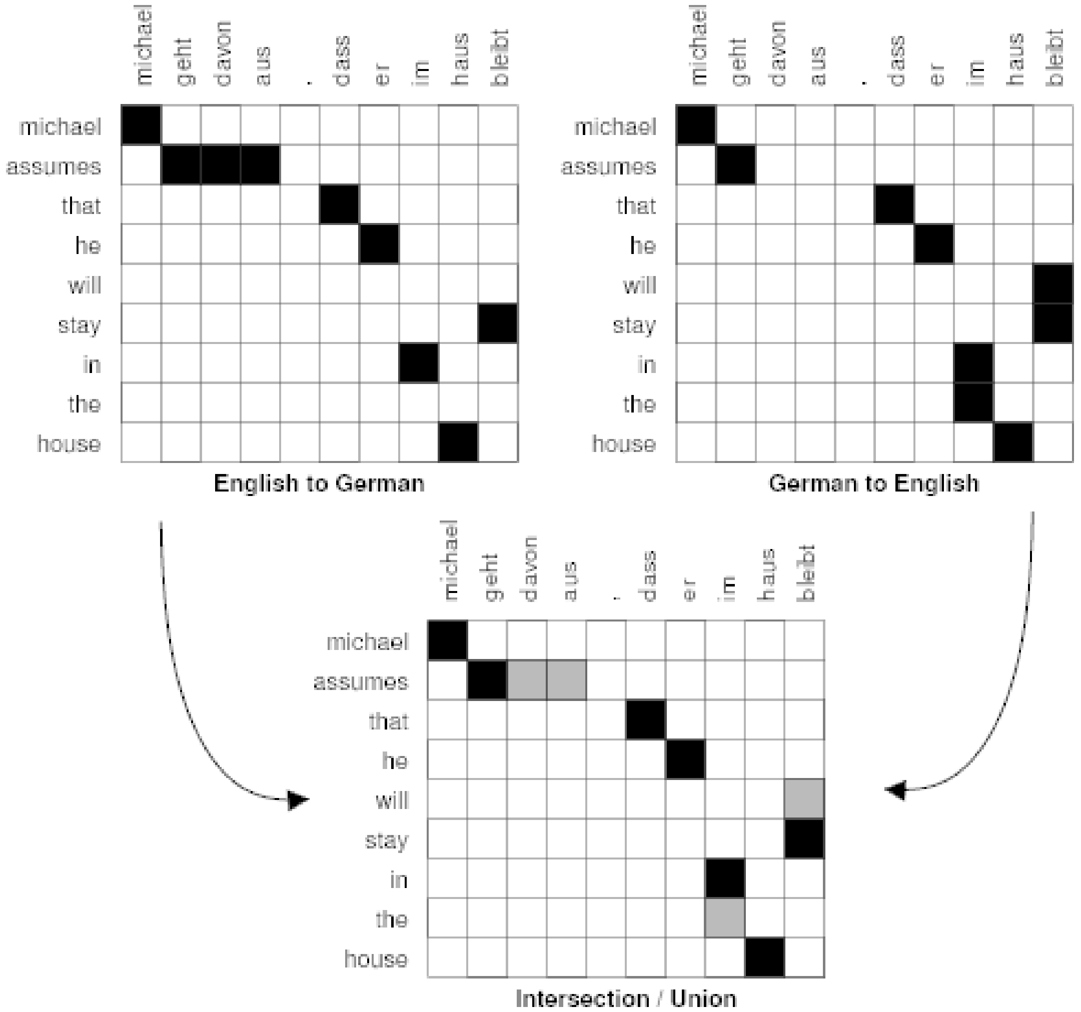
E = NULL$_0$ show$_1$ me$_2$ the$_3$ one$_4$ in$_5$ the$_6$ window$_7$

J = ウィンドー$_1$ の$_2$ 品物$_3$ を$_4$ 見せ$_5$ てください$_6$

A = (7 0 4 0 1 1)

- Note that the alignment is one way
  - NULL only on source side (fertility)
  - one to many mapping

Combining Alignments



English to German

	michael	geht	davon	aus	,	dass	er	im	haus	bleibt
michael	■									
assumes		■	■	■						
that						■				
he							■			
will										
stay										■
in								■		
the										
house									■	

German to English

	michael	geht	davon	aus	,	dass	er	im	haus	bleibt
michael	■									
assumes		■								
that						■				
he							■			
will										■
stay										■
in								■		
the								■		
house									■	

Intersection / Union

	michael	geht	davon	aus	,	dass	er	im	haus	bleibt
michael	■									
assumes		■	▨	▨						
that						■				
he							■			
will										▨
stay										■
in								■		
the								▨		
house									■	

In SMT, these chunks are called phrases although they may not be linguistic units (e.g. *in the* ↔ *im*).

Improving Alignments

- The most common alignment approaches use word coocurrence over large bitexts
  - How often do words s_i and t_j appear in the same sentence pair compared to appearing in different sentence pairs.
- Other possible features include
  - same semantic class (from an ontology or cluster)
  - orthographic similarity
  - shared translations in a third language
- These are especially useful for matching rare words

Translation Model (IBM Model 4)

$P(J, A|E)$

Fertility Model

$$\prod n(\varphi_i|E_i)$$

NULL Generation Model

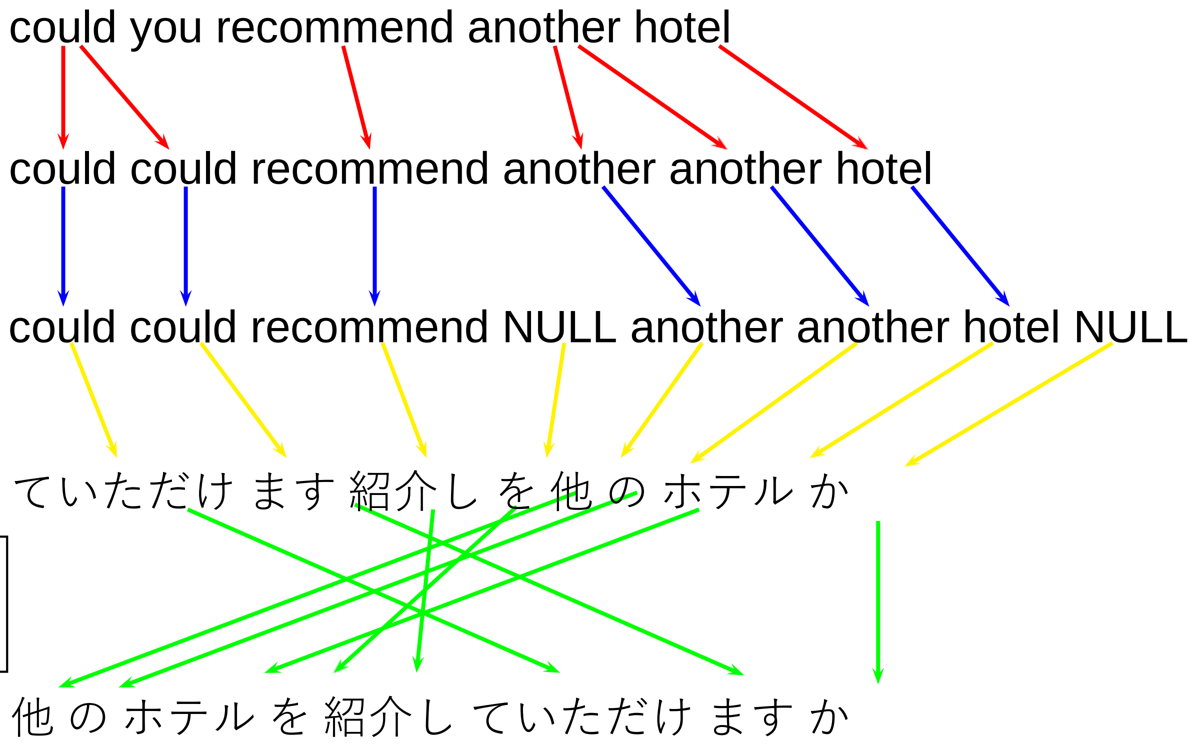
$$\binom{m-\varphi_0}{\varphi_0} p_0^{m-2\varphi_0} p_1^{\varphi_0}$$

Lexicon Model

$$\prod t(J_j|E_{A_j})$$

Distortion Model

$$\prod d_1(j-k|A(E_i)B(J_j)) \quad d_{1>}(j-j'|B(J_j))$$



could you recommend another hotel

could could recommend another another hotel

could could recommend NULL another another hotel NULL

ていただけ ます 紹介し を 他 の ホテル か

他 の ホテル を 紹介し ていただけ ます か

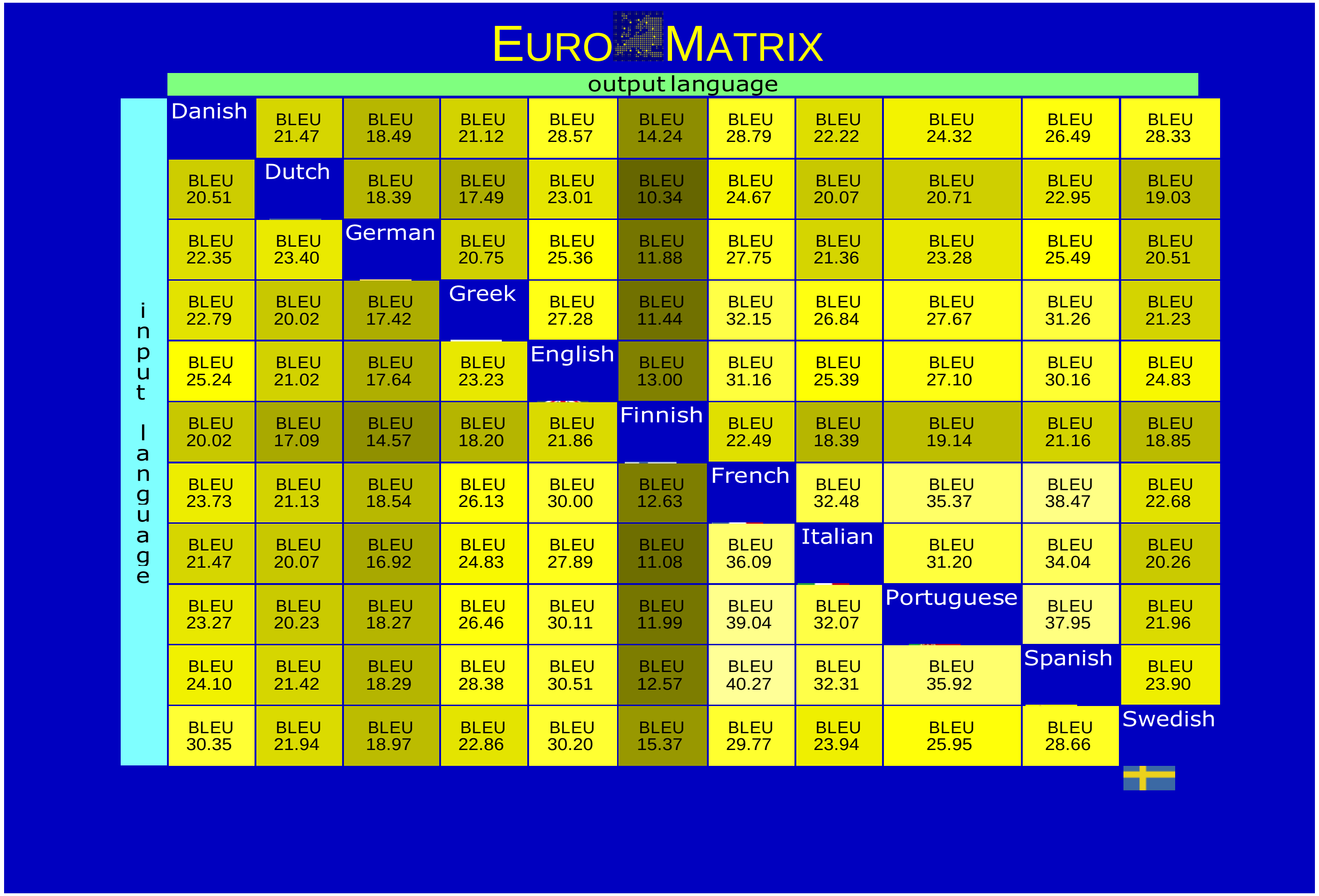
Millions of candidates are produced and ranked.

Euro Matrix

- Translation between EU languages (EU funded project)
- Europarl: proceedings of the European Parliament.
- 18-40 million words, .6–1.3 million sentences
- Freely available text in all European Languages

Europarl: A Parallel Corpus for Statistical Machine Translation, Philipp Koehn, MT Summit 2005

Euro Matrix Results



EuroMatrix

output language

input language

Danish	BLEU 21.47	BLEU 18.49	BLEU 21.12	BLEU 28.57	BLEU 14.24	BLEU 28.79	BLEU 22.22	BLEU 24.32	BLEU 26.49	BLEU 28.33
BLEU 20.51	Dutch	BLEU 18.39	BLEU 17.49	BLEU 23.01	BLEU 10.34	BLEU 24.67	BLEU 20.07	BLEU 20.71	BLEU 22.95	BLEU 19.03
BLEU 22.35	BLEU 23.40	German	BLEU 20.75	BLEU 25.36	BLEU 11.88	BLEU 27.75	BLEU 21.36	BLEU 23.28	BLEU 25.49	BLEU 20.51
BLEU 22.79	BLEU 20.02	BLEU 17.42	Greek	BLEU 27.28	BLEU 11.44	BLEU 32.15	BLEU 26.84	BLEU 27.67	BLEU 31.26	BLEU 21.23
BLEU 25.24	BLEU 21.02	BLEU 17.64	BLEU 23.23	English	BLEU 13.00	BLEU 31.16	BLEU 25.39	BLEU 27.10	BLEU 30.16	BLEU 24.83
BLEU 20.02	BLEU 17.09	BLEU 14.57	BLEU 18.20	BLEU 21.86	Finnish	BLEU 22.49	BLEU 18.39	BLEU 19.14	BLEU 21.16	BLEU 18.85
BLEU 23.73	BLEU 21.13	BLEU 18.54	BLEU 26.13	BLEU 30.00	BLEU 12.63	French	BLEU 32.48	BLEU 35.37	BLEU 38.47	BLEU 22.68
BLEU 21.47	BLEU 20.07	BLEU 16.92	BLEU 24.83	BLEU 27.89	BLEU 11.08	BLEU 36.09	Italian	BLEU 31.20	BLEU 34.04	BLEU 20.26
BLEU 23.27	BLEU 20.23	BLEU 18.27	BLEU 26.46	BLEU 30.11	BLEU 11.99	BLEU 39.04	BLEU 32.07	Portuguese	BLEU 37.95	BLEU 21.96
BLEU 24.10	BLEU 21.42	BLEU 18.29	BLEU 28.38	BLEU 30.51	BLEU 12.57	BLEU 40.27	BLEU 32.31	BLEU 35.92	Spanish	BLEU 23.90
BLEU 30.35	BLEU 21.94	BLEU 18.97	BLEU 22.86	BLEU 30.20	BLEU 15.37	BLEU 29.77	BLEU 23.94	BLEU 25.95	BLEU 28.66	Swedish

Euro Matrix Discussion

- Linguistic similarity affects the score:
  - Highest: Spanish → French (BLEU = 40.27)
  - Lowest: Italian → Finnish (BLEU = 11.08)
- Creating all $n(n - 1)$ language pairs took a week
  - It is easy to add new languages if you have a multi-lingual corpus
- Translation done using the open source SMT System:
  Moses `<statmt.org>`

SMT State of the Art

- More data improves BLEU: (Och, 2005)
  - Doubling the translation model data gives a 2.5% boost.
  - Doubling the language model data gives a 0.5% boost.
  - For linear improvement in translation quality the data must increase exponentially
    - BLEU +10% needs $2^4 = 16$ times as much bilingual data
    - BLEU +20% needs $2^8 = 256$ times as much bilingual data
    - BLEU +30% needs $2^{12} = 4096$ times as much bilingual data

- Richer models improve quality ⇒ syntax based models

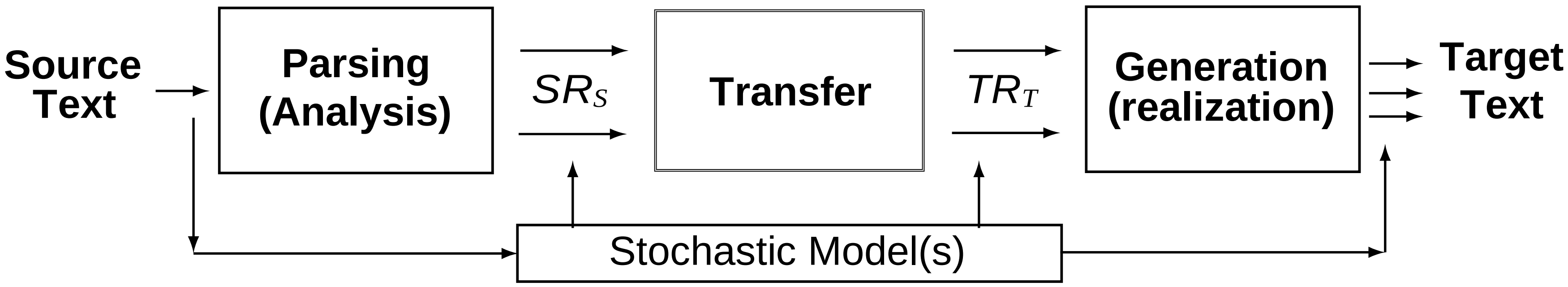
- Pruning bad models improves quality ⇒ multilingual models

Transfer

Transfer in Machine Translation

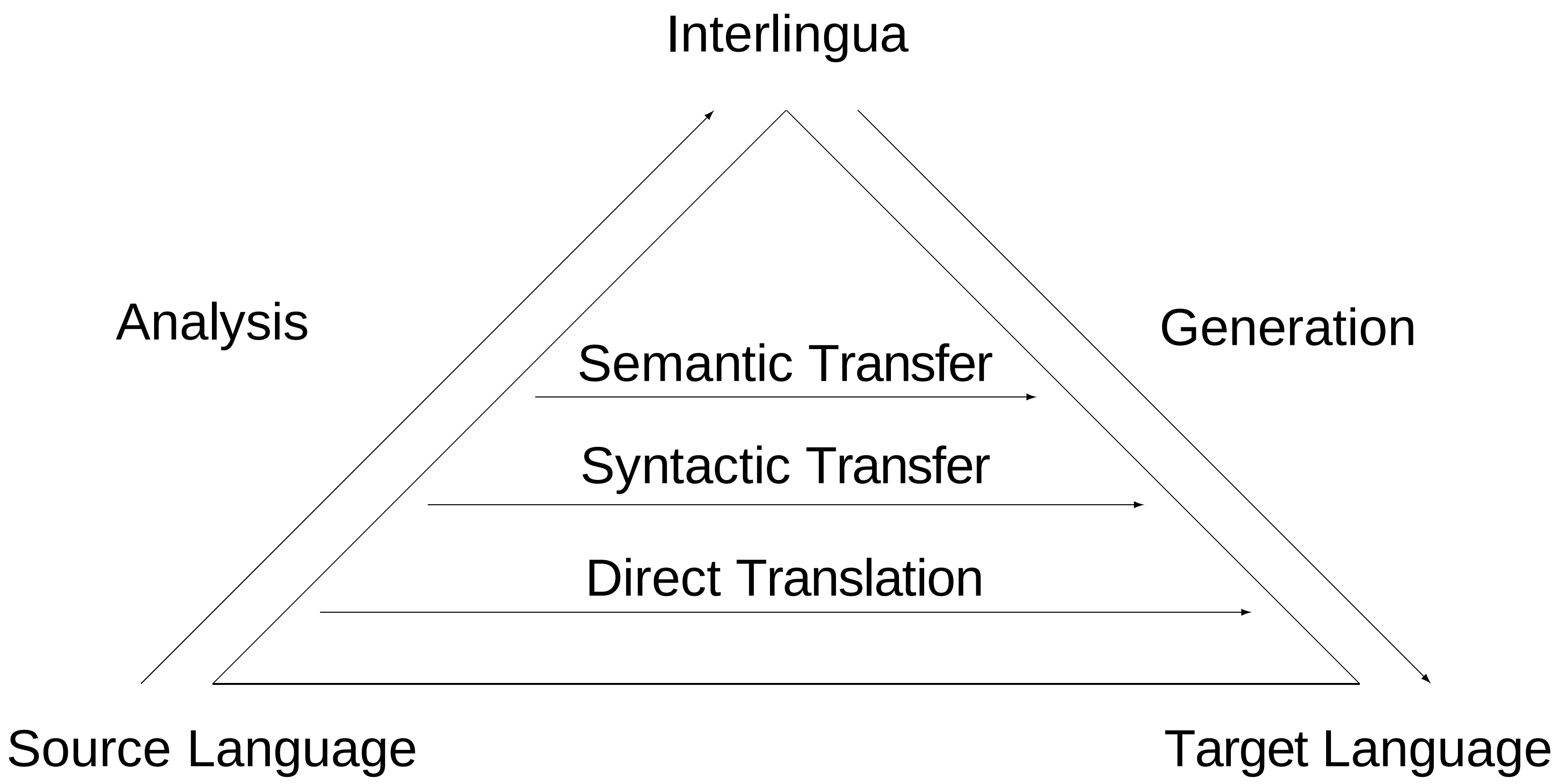
- Approaches to Transfer
- Particular Problems (and solutions)
- Ways to improve

The Overall Architecture



- ➢ Parse source text to source representation (SR)
- ➢ Transfer this to some target representation (TR) (This week)
- ➢ Generate target text from the TR

How Deep Should We Go?



The Vauquois Triangle

Direct Transfer

Input *Mary didn’t slap the green witch*

Morphology Mary do-PAST NOT slap the green witch

Lexical Transfer Maria dar PAST no una bofetafa a la verde bruja

Morphology/Reordering Maria no dió una bofetafa a la bruja verde

- Just morphological analysis, no syntactic analysis
  - Works quite well for very similar languages
    - Galician/Catalan
    - Japanese/Korean
    - Malay/Indonesian
- Works very badly for languages with different word order

Lexical Selection is a problem

- People write very detailed rules to select the correct translation

- Japanese-English example: 鼻 *hana* “nose”
  - 鼻 proper noun → Hana
  - 鼻 possessed by 象 *zou* “elephant” → *trunk*
  - 鼻 possessed by 馬 *uma* “horse” → *muzzle*
  - 鼻 possessed by 豚 *buta* “pig” → *snout*
  - 鼻 → nose

- Ontologies/thesauruses make the rules more flexible
  - mammoth ⊂ elephant
  - wild boar, hog, pig ⊂ swine

- Otherwise you have a lot of rules or miss cases

Japanese-English example: 群れ ***mure*** “group”

- 群れ *group* of
  - fish → *school* (semantic class)
  - insect → *swarm*
  - lion → *pride* (word)
  - wolf, wild dog → *pack*
  - star, computer → *cluster*
  - sheep → *flock*
  - bird → *flock*
  - animal → *herd*
  - people → *crowd*
- Many more are possible (*bevy, mob, pod, . . .*)
- This is filling in a lexical gap:
  . . . Japanese just doesn’t make these distinctions

Syntactic Transfer

- Word for word won’t work with very different word orders
- The condition for a transfer rule may be far away
  - *pack of wolves*
  - *pack of large, hungry, gray wolves*
- We should look at the sentence structure

Syntactic Transfer: Spanish-English

- In Spanish, Italian, French, Malay, . . . adjectives follow nouns
  - the green witch → la bruja verde
- Try to make general rules for this
  NP → AdjP N ↔ NP → N AdjP
- The general strategy is to apply transfer rules top down from the root

Syntactic Transfer: English-Japanese

- *He likes music.*

-

彼が	音楽を	聞くのが	大好きだ
kare-ga	ongaku-wo	kiku-no-ga	daisuki-da
he-SUBJ	music-OBJ	listen-NOM-SUBJ	likes

- Word order is very different!

Semantic Transfer

- Aim for simpler semantic transfer
  - Push work to the monolingual grammars
  - Moving toward an interlingua
  - Transfer can ignore language specific syntax
- Modularize the components
  - Define a clean **Sem**antic-**I**nterface
  - Allow independent work on components
- Reduce, Reuse, Recycle

Example: Source

- ビール を 三つ もって きて ください

biiru-wo	mittsu	motte	kite	kudasai
beer-ACC	three-CL	hold	come	give:honorific

Please bring three beers.

- $(h_1, \{h_1$: motsu_v$(e_1 :$ COMMAND$, u_2, x_1)$,
  h_1: kuru_v(e_2, u_3),
  h_4: biiru_n(x_1),
  h_6: udef_q(x_1, h_7, h_8),
  h_9: card$(u_1, x_1,$ “3”$)$,
  h_{15}: kudasaru_v$(e_3, u_4, u_5, h_2)\ \}$,
  $\{h_7 = h_4, h_2 = h_1\})$

- *motte kuru* “hold come” grouped together (bring)

Example: Transfer

Transfer:

- biiru_n(x_i) → beer_n(x_i)
- h_j: motsu_v(e_1, u_2, x_1), “hold” h_j: kuru_v(e_2, u_3) “come” → h_j: bring_v(e_1, u_2, x_1)
- h_i: kudasaru_v(e_j, h_k) → h_i: please_a(e_j, h_k) (verb → adverb)

Example: Target

- (h_0, {h_0: please_a(e_3, h_1),
  h_1: imp_m(h_3),
  h_2: pronoun_q(x_0, h_7, h_8), h_4: pron($x_0\{2nd\}$),
  h_5: bring_v(e_2, x_0, x_1),
  h_4: beer_n(x_1), h_6: udef_q(x_1, h_{10}, h_8), h_{11}: card(u_1, x_1, “3”) },
  $\{h_3 = h_5, h_7 = h_4, h_{10} = h_{11}, \}$)
- Two word orders possible
  - Please bring three beers.
  - Bring three beers please.

Semantic Transfer Pros and Cons

- Source and Target grammars do much of the work
  - Pro: modular, transfer easier
  - Cons: brittle (if parsing fails, everything fails)
- Language specific details hidden by the semantic interface
- General Problems Remain
  - Sense Disambiguation (lexical choice)
    is 鳩 *hato* a *dove* or a *pigeon*
  - Language Differences
    - number, countability, articles
- Over-generate and choose with a statistical model

The Importance of Multiword Expressions

- Context beyond a single word is very important

- In a typical system most rules (entries in the transfer dictionary) are multiword (60% in **ALT-J/E**)
  - 機械 翻訳 *kikai honyaku* “machine translation” → machine translation
  - 雨 が 降 る *ame-ga furu* “rain falls” → rains

- If you consider conditions as part of the translation, then this goes up more
  - 鼻 *hana* “nose” possessed by 象 *zou* “elephant” → trunk
  - 鼻 *hana* “nose” possessed by 豚 *buta* “pig” → snout
  - 鼻 *hana* “nose” → nose

Issues with Transfer

- ➢ Choosing between multiple options is difficult
  - ⇒ Create larger rules with more context
  - ⇒ Try to weight with statistical models
- ➢ The number of rules is far greater than the number of words
  Context multiplies rules
  - ⇒ Generalize rules with ontologies
  - ⇒ Learn from bilingual corpora
  - ⇒ Restrict according to domain
  - ⇒ Share rules (open source)

Some well known problems

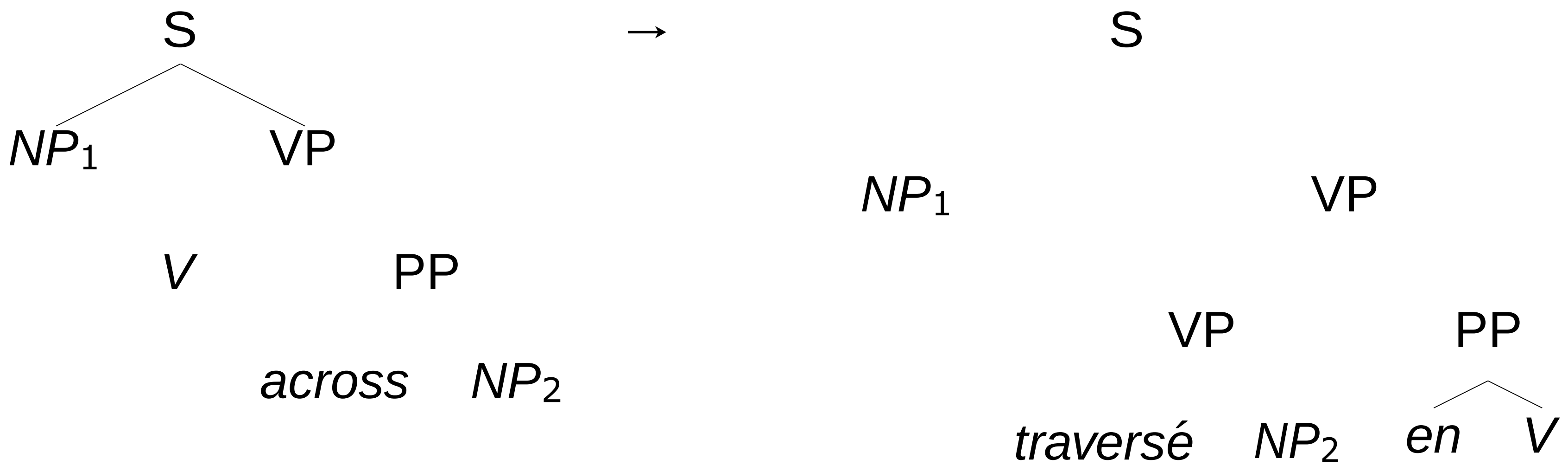
- Head-switching: head is dependent in the other language
- Relation-changing: e.g. verb → adjective
- Lexical Gaps: translation missing in the source or target language
- Possessive Pronoun Drop: possessive pronouns required in some languages, but not others
- Number mismatch: number required in one language but not the other
- Argument mismatch: Verb structure is different
- Idiom mismatch: Idiomatic in one language but not the other

Head Switching

- Head switching is just a more complicated rule:

(28) *I swam across the river*

(29) *J'ai traversé le fleuve en nageant*
I crossed the river by swimming



Relation Changing

- Translation equivalents may be different POS:

(30)

濡れて	いる	紙
nurete	iru	kami
wetting	is	paper

wet paper

- Verb → Adjective
- Allow translation rules to do this
  - Normally anchor lexically to reduce complexity
    - ⊗ VP → AP
    - · *nureru v* → *wet a*

Lexical Gaps

- More specific to less specific
  - Just lose some information
    - *herd, pack, mob, crowd, group* → *mure*
- Less specific to more specific
  - Add context to the transfer rules to disambiguate
  - Add multiword expressions to the dictionary

Possessive Pronoun Drop

REF	Kanji:	鼻が	かゆい	
	Jap:	*hana-ga*	*kayui*	
	Gloss:	nose-SUBJ	itch	
	Eng:	‘My nose itches’		
GEN	Kanji:	鼻は	感覚器官	だ
	Jap:	*hana-wa*	*kankakukikan*	*da*
	Gloss:	nose-TOP	sensory organ	is
	Eng:	‘Noses are sensory organs’ ‘The nose is a sensory organ’ ‘A nose is a sensory organ’		

- Possessive pronouns are obligatory for some nouns (possessed-nouns): Nouns that denote kin, body parts, work, personal possessions, attributes and people defined by their relation to another person

Generating possessive pronouns:

A If a referential phrase is headed by a possessed-noun and is not the direct object of a verb with meaning POSSESSION or ACQUISITION then:

- ➢ Generate a possessive pronoun whose referent is the subject of the sentence.

I scratched my nose; *She scratched her nose*

B Generate possessive pronouns for all noun phrases

- ➢ Rank with a language model

➢ There is no perfect solution

- ➢ **A** requires very complex processing
- ➢ **B** makes every noun phrase very ambiguous

Number mismatch

- Some examples (Nouns are unmarked for number in Japanese)
  - マンモスは全滅した。→ *Mammoths are extinct.*
  - 花を集まった。→ *I gathered flowers.*
  - この３人は、友達だ。
→ *These three people are friends.*
  - ３人は大勢だ。→ *Three people are a crowd.*

A Write rules that use context: (accurate)

- Verb/Adjective: *be extinct*, *gather*
- Modifiers: *three*, *many*
- Defaults: *noodles*

B Over generate and rank with a language model (easy)

Argument Mismatch

- Verb (or adjective) structure is different
  - *watashi-ni kodomo-ga iru* “to me children are”

    → *I have children*

    to→SUBJECT; SUBJECT→OBJECT
  - *Kim married Sandy*

    → *Kim-ga Sandy-to kekkon-shita* “Kim married with Sandy”

    OBJECT→*-to* “with”

Idiom mismatch

- Idiomatic in one language but not the other (or not in the same way)
  - *I lost my head* "I got angry"
    → *atama-ni kita* "it came to my head"
  - *I racked my brains* "I thought hard"
    → *chie-wo shibotta* "I squeezed knowledge"

    I lost my head → *I got angry*

- Some idioms are so common that we don't notice them
  - *I catch the bus* "I get on the bus"
  - *I follow you* "I understand you"

User Dictionaries

- The simplest way to improve translation quality
- Build a special dictionary: the user dictionary
- User dictionary entries are preferred to words in the system dictionaries
  - You can force the translation you want
- Typical MT use for large projects is to
  1. Translate once
  2. Find common errors
  3. Fix them by adding entries to the user dictionary
  4. Re-translate

How to Predict Machine Translation Quality

- The following phenomena are hard to translate:
  - Long sentences
  - Coordination
  - Unknown words (either new words or spelling errors)
    - new genre
    - poorly edited text
  - Different language families
- We can identify these and give a translatability score
  - This is useful to identify text for post-editing

Summary

- MT is very hard
- We have handle on many of the sub-problems
- But it is probably AI-complete

Acknowledgments and Readings

- These slides were made by Francis Bond

- Much of the Machine Translation section was taken from a panel discussion by John Hutchins (13 December 2007)
  `http://www.hutchinsweb.me.uk/`

- Much more on Morphological Analysis (Chapter 3) and Part of Speech Tagging (Chapter 5) in Jurafsky and Martin (2009)

- Nice discussion of big-O notation
  `http://science.slc.edu/~jmarshall/courses/2002/spring/cs50/BigO/index.html`

- Parsing (including demo code) in Bird, Klein and Loper (2009) *Natural*

Language Processing with Python, O'Reilly
`http://nltk.googlecode.com/svn/trunk/doc/book/ch08.html`

- More on Parsing in Jurafsky and Martin (2009), Chapters 11 and 12.
- Good site for SMT: `statmt.org`
- **Machine Translation**: Jurafsky and Martin (2009), Chapter 25.1–2
- **Statistical Machine Translation**: Jurafsky and Martin (2009), Chapter 25.3–8
- **Word Sense Disambiguation**: Jurafsky and Martin (2009), Chapter 20.1–8

- Some slides based on Rada Mihalcea and Ted Pedersen’s tutorial at AAAI-2005 “Advances in Word Sense Disambiguation”

- Nice demo of similarities at:
  `marimba.d.umn.edu/cgi-bin/similarity/similarity.cgi`