Natural Language Processing

Fall 2022

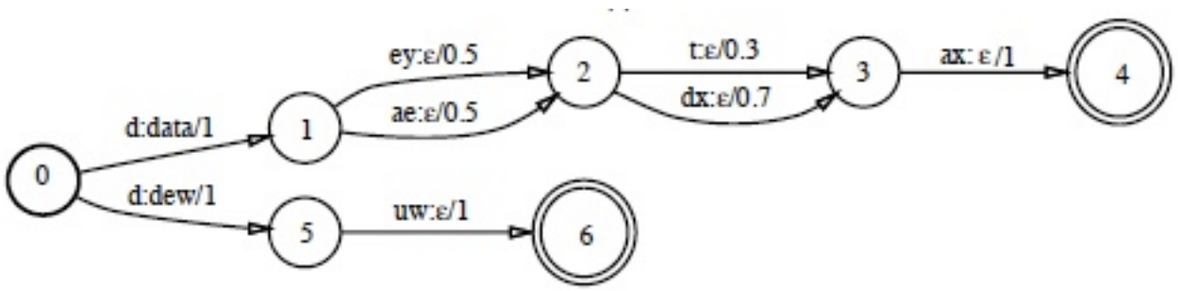
Unit 1: Sequence Models

Lectures 7-8: Stochastic String Transformations

(a.k.a. "channel-models")

required

optional



Professor Liang Huang

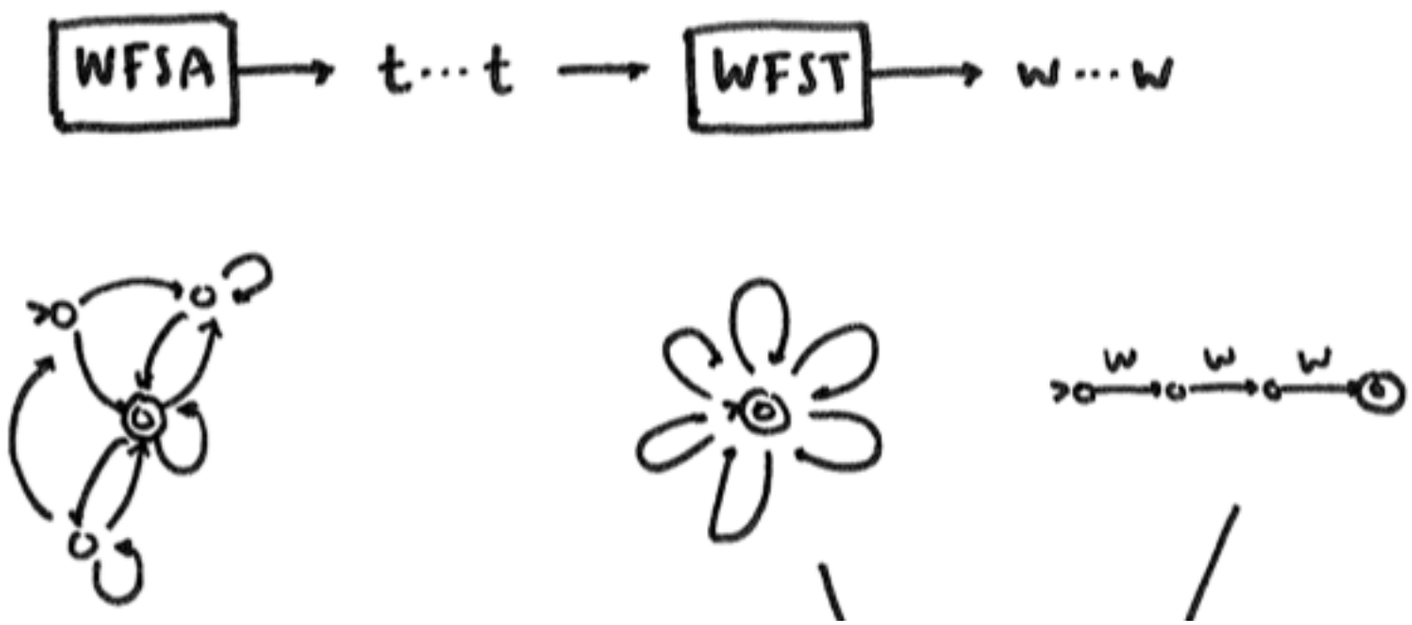
liang.huang.sh@gmail.com

String Transformations

- General Framework for many NLP problems
- Examples

  - Part-of-Speech Tagging
  - Spelling Correction (Edit Distance)
  - Word Segmentation
  - Transliteration, Sound/Spelling Conversion, Morphology
  - Chunking (Shallow Parsing)
  - Beyond Finite-State Models (i.e., tree transformations)
    - Summarization, Translation, Parsing, Information Retrieval, ...
- Algorithms: Viterbi (both max and sum)

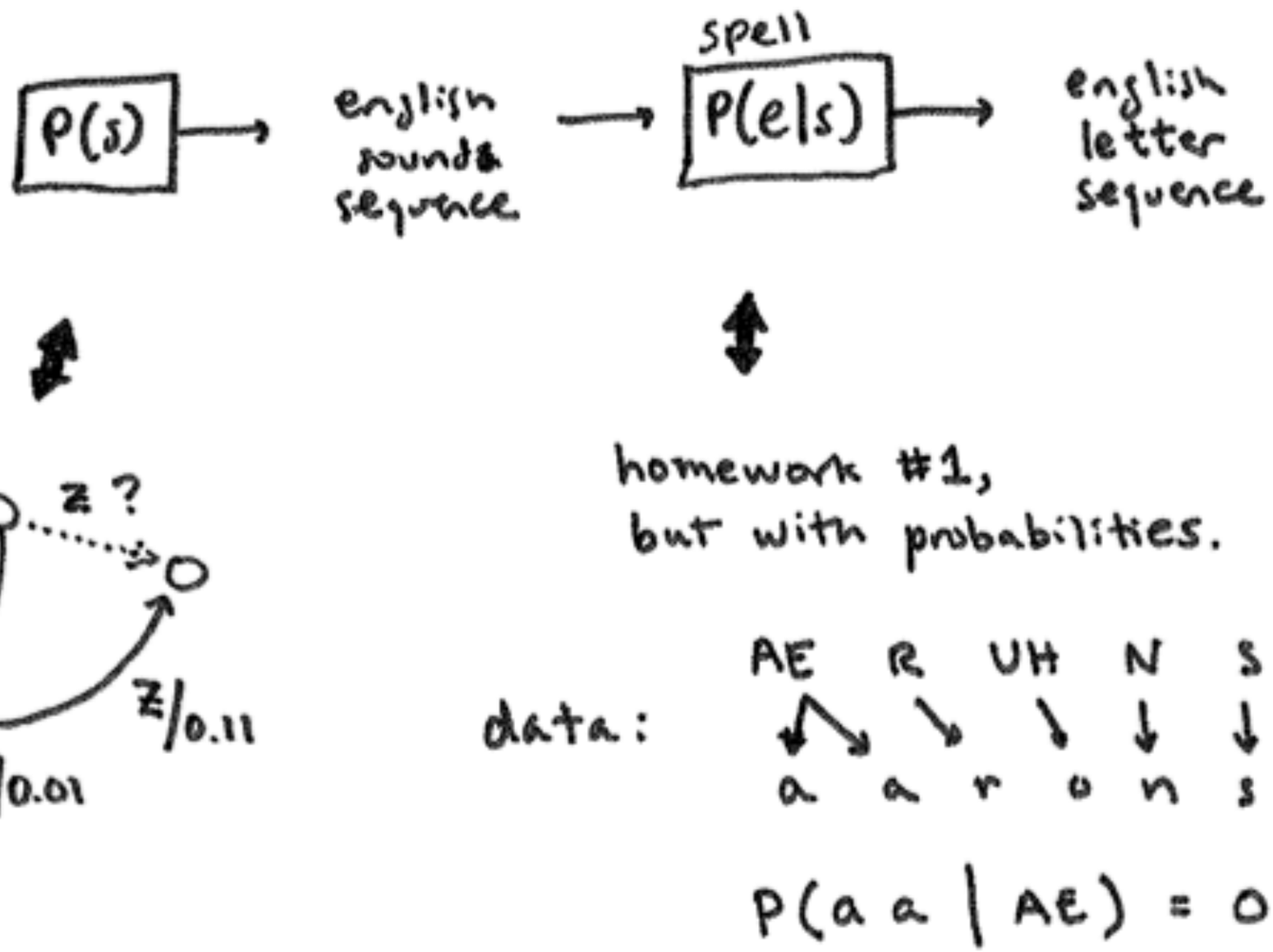
Review of Noisy-Channel Model



Application	Input	Output	p(i)	p(o|i)
Machine Translation	L_1 word sequences	L_2 word sequences	p(L_1) in a language model	translation model
Optical Character Recognition (OCR)	actual text	text with mistakes	prob of language text	model of OCR errors
Part Of Speech (POS) tagging	POS tag sequences	English words	prob of POS sequences	p(w|t)
Speech recognition	word sequences	speech signal	prob of word sequences	acoustic model

(hw2) From Spelling to Sound

- word-based or char-based



Pronunciation Dictionary

- (hw3: eword-epron.data)
  - ...
  - AARON EH R AH N
  - AARONSON AA R AH N S AH N
  - ...
  - PEOPLE P IY P AH L
  - VIDEO V IH D IY OW
- you can train p(s..s|w) from this, but what about unseen words?
- also need alignment to train the channel model p(s|e) & p(e|s)

http://www.speech.cs.cmu.edu/cgi-bin/cmudict
from CMU Pronunciation Dictionary
39 phonemes (15 vowels + 24 consonants)

```
echo 'W H A L E B O N E S' |
     carmel -sriIEQk 5 epron.wfsa epron-espell.wfst
```

CMU Dict: 39 Ame. Eng. Phonemes

WRONG! missing the SCHWA ə (merged with the STRUT ʌ "AH")!

CMU/IPA	Example	Translation
AA /ɑ/	odd	AA D
AE /æ/	at	AE T
AH /ʌ/	hut	HH AH T
AO /ɔ:/	ought	AO T
AW /aʊ/	cow	K AW
AY /aɪ/	hide	HH AY D
B /b/	be	B IY
CH /tʃ/	cheese	CH IY Z
D /d/	dee	D IY
DH /ð/	thee	DH IY
EH /ɛ/	Ed	EH D
ER /ɚ/	hurt	HH ER T
EY /eɪ/	ate	EY T
F /f/	fee	F IY
G /g/	green	G R IY N
HH /h/	he	HH IY
IH /ɪ/	it	IH T
IY /i:/	eat	IY T
JH /dʒ/	gee	JH IY

CMU/IPA	Example	Translation
K /k/	key	K IY
L /l/	lee	L IY
M /m/	me	M IY
N /n/	knee	N IY
NG /ŋ/	ping	P IH NG
OW /oʊ/	oat	OW T
OY /ɔɪ/	toy	T OY
P /p/	pee	P IY
R /ɹ/	read	R IY D
S /s/	sea	S IY
SH /ʃ/	she	SH IY
T /t/	tea	T IY
TH /θ/	theta	TH EY T AH
UH /ʊ/	hood	HH UH D
UW /u/	too	T UW
V /v/	vee	V IY
W /w/	we	W IY
Y /j/	yield	Y IY L D
Z /z/	zee	Z IY
ZH /ʒ/	usual	Y UW ZH UW AH L

CMU Pronunciation Dictionary

WRONG! missing the SCHWA ə (merged with the STRUT ʌ "AH")!
DOES NOT ANNOTATE STRESSES

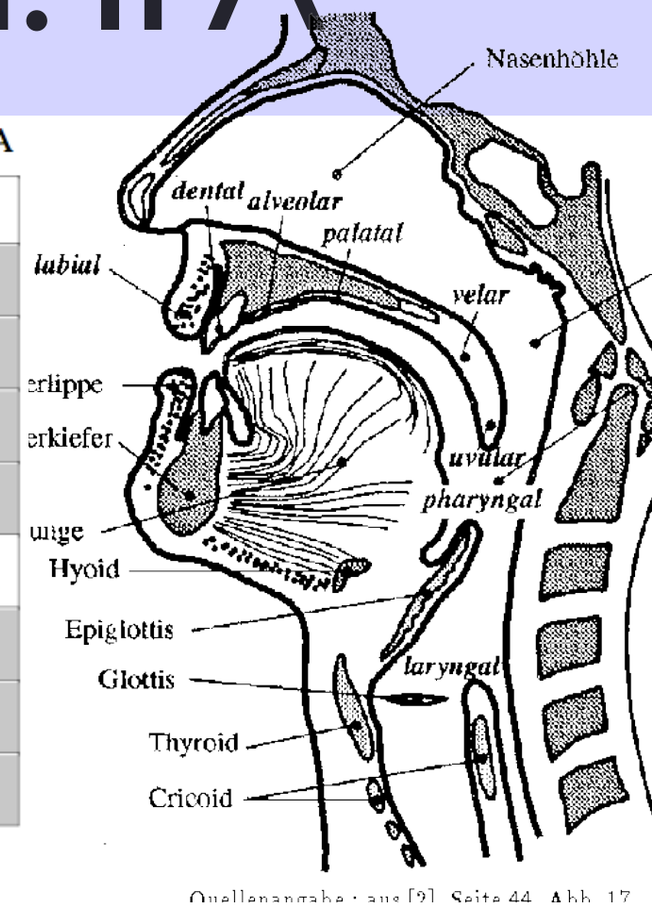
```
A            AH
A            EY
AAA          T R IH P AH L EY
AABERG       AA B ER G
AACHEN       AA K AH N
...
ABOUT        AH B AW T
...
ABRAMOVITZ AH B R AA M AH V IH T S
ABRAMOWICZ AH B R AA M AH V IH CH
ABRAMOWITZ AH B R AA M AH W IH T S
...
FATHER       F AA DH ER
...
ZYDECO       Z AY D EH K OW
ZYDECO       Z IH D AH K OW
ZYDECO       Z AY D AH K OW
...
ZZZZ         Z IY Z
```

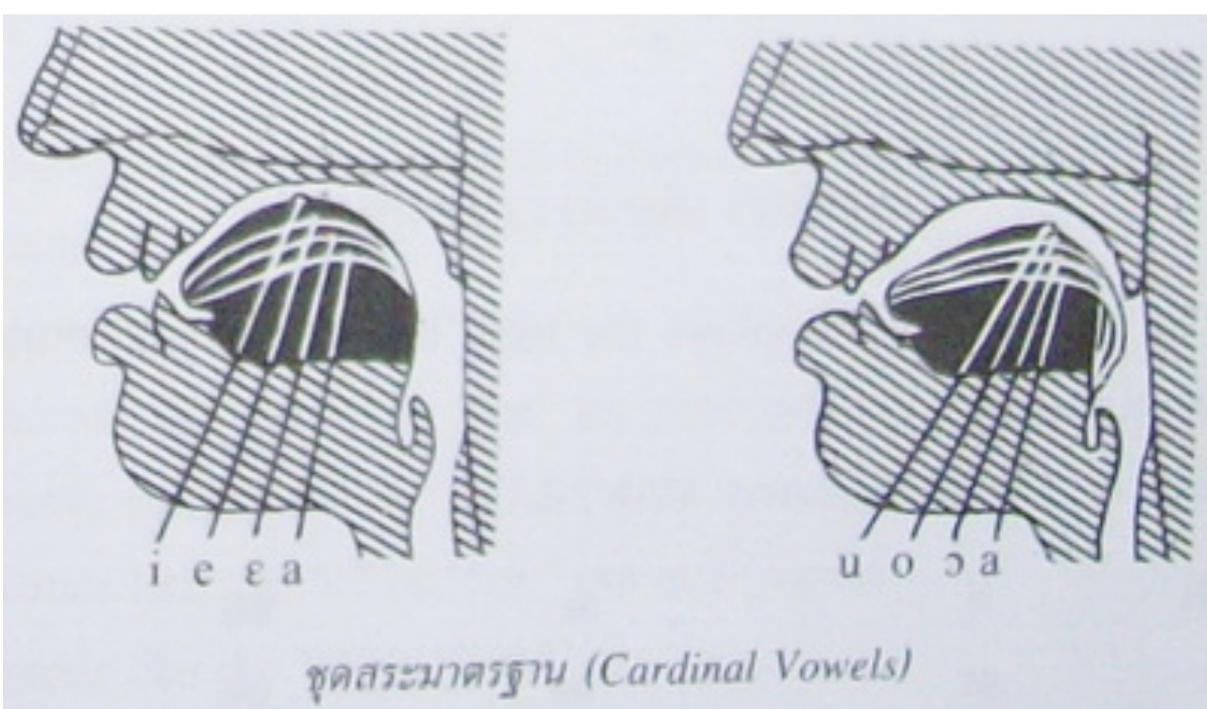
Linguistics Background: IPA

CONSONANTS (PULMONIC) © 2005 IPA

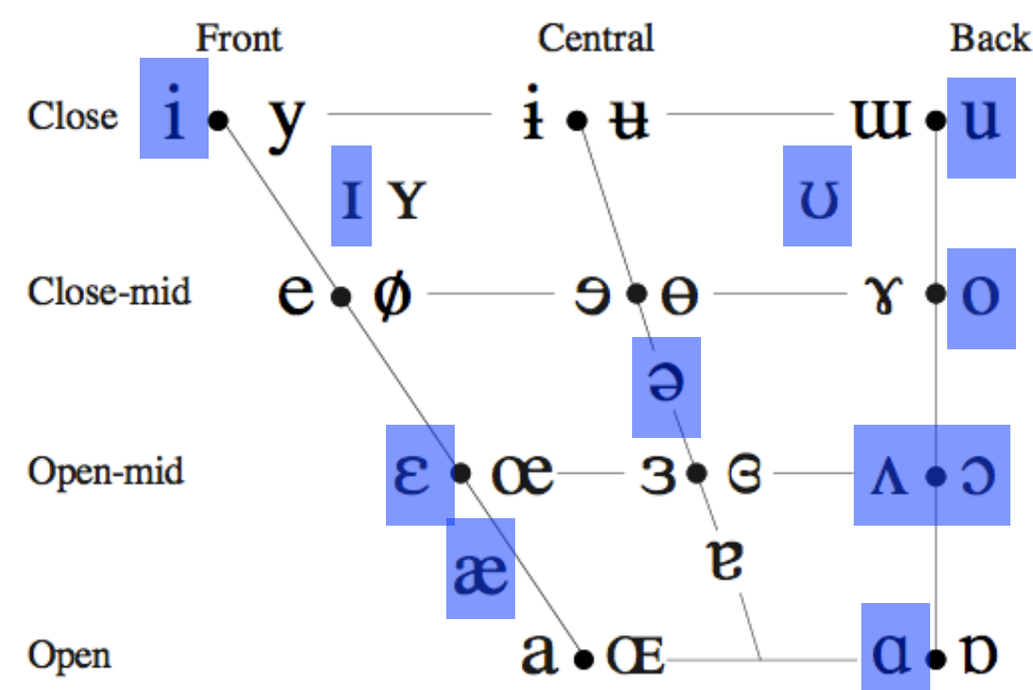
	Bilabial	Labiodental	Dental	Alveolar	Postalveolar	Retroflex	Palatal	Velar	Uvular	Pharyngeal	Glottal
Plosive	p b			t d		ʈ ɖ	c ɟ	k g	q ɢ		ʔ
Nasal	m	ɱ		n		ɳ	ɲ	ŋ	ɴ		
Trill	ʙ			r					ʀ		
Tap or Flap		ⱱ		ɾ		ɽ					
Fricative	ɸ β	f v	θ ð	s z	ʃ ʒ	ʂ ʐ	ç ʝ	x ɣ	χ ʁ	ħ ʕ	h ɦ
Lateral fricative				ɬ ɮ							
Approximant		ʋ		ɹ		ɻ	j	ɰ			
Lateral approximant				l		ɭ	ʎ	ʟ			

Where symbols appear in pairs, the one to the right represents a voiced consonant. Shaded areas denote articulations judged impossible.





VOWELS



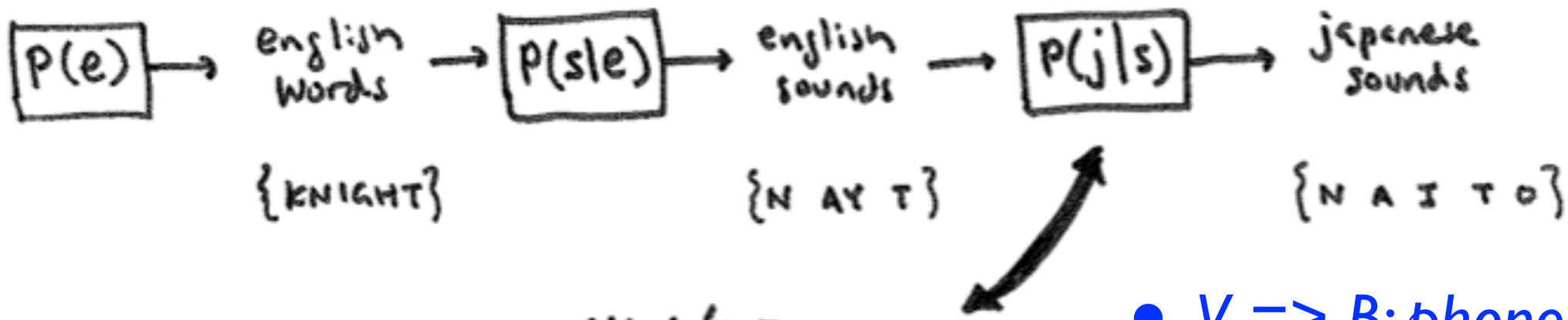
Where symbols appear in pairs, the one to the right represents a rounded vowel.

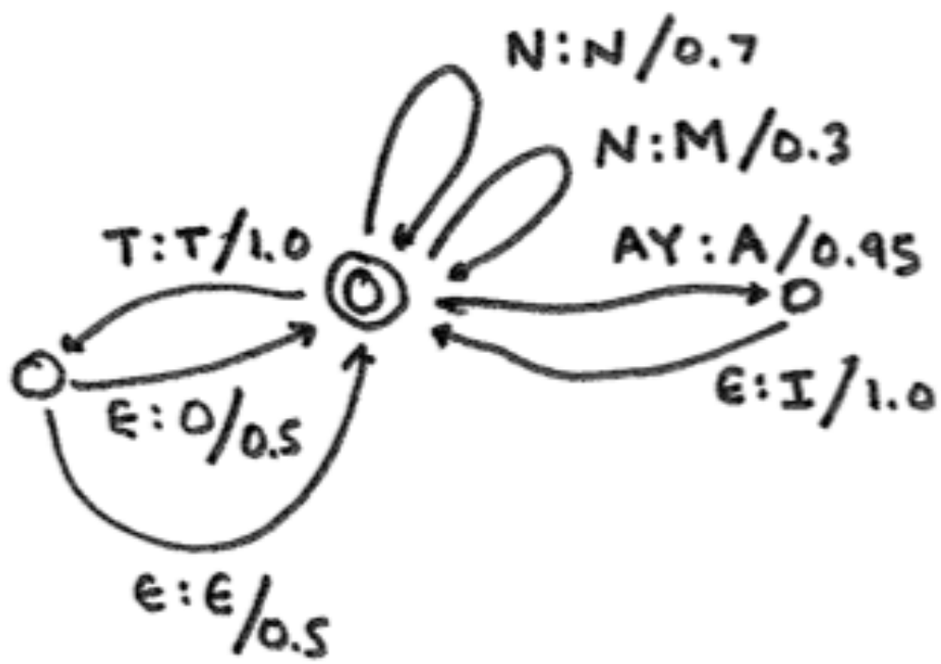
(hw2) From Sound *s* to Spelling *e*

- input: HH EH L OW B EH R
- output: H E L L O B E A R *or* H E L O B A R E ?
- p(e) => e => p(s|e) => s
- p(w) => w => p(e|w) => e => p(s|e) => s
- p(w) => w => p(s|w) => s
- e <= p(e|s) <= s <= p(s)
- w <= p(w|e) <= e <= p(e|s) <= s <= p(s)
- w <= p(w|s) <= s <= p(s)
- what else?

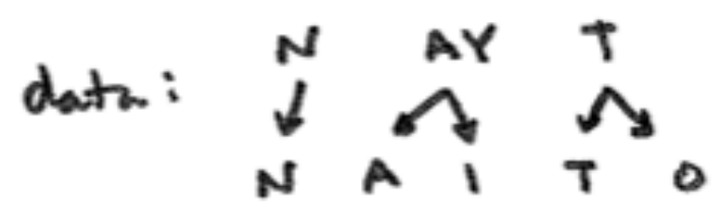
```
echo 'HH EH L OW' | carmel -sliOEQk 50 epron-espell.wfst
espell-eword.wfst eword.wfsa
```

Example: Transliteration





- *V => B: phoneme inventory mismatch*
- *T=>T O: phonotactic constraint*



- KEVIN KNIGHT => KH EH VH IH N N AY T
  K E *B* I N N A I *T O*
  ケ ビ ン ナ イ ト

Japanese 101 (writing systems)

- Japanese writing system has four components
  - Kanji (Chinese chars): nouns, verb/adj stems, CJKV names
    - 日本 “Japan” 东京 “Tokyo” 电车 “train” 食べる “eat [inf.]”
  - Syllabaries
    - Hiragana: function words (e.g. particles), suffices
      - で de (“at”) か ka (question) 食べました“ate”
    - Katakana: transliterated foreign words/names
      - コーヒー koohii (“coffee”)
  - Romaji (Latin alphabet): auxiliary purposes

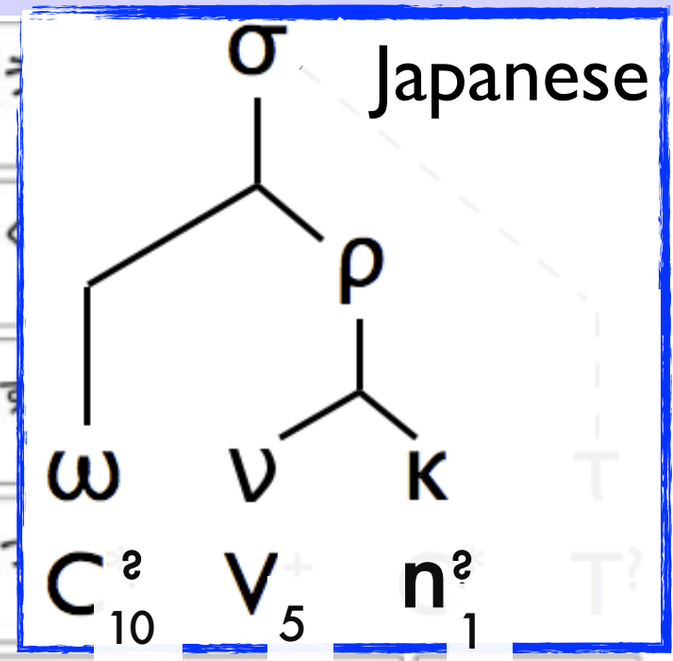
Annual Kanji of the Year

Annual Kanji of the Year

1995	震	发生阪神大地震；奥姆真理教沙林毒气袭击事件震撼日本社会。
1996	食	O-157大腸桿菌引發集體食物中毒，影響學校午餐供應。
1997	倒	數家大企業倒閉；大銀行瀕臨危機；日本國家足球隊打倒強敵，取得世界盃足球賽的出賽資格。
1998	毒	和歌山毒咖哩事件及其引發的多起模仿下毒犯罪；有毒物質二噁英的污染問題。
1999	末	世紀末來臨；東海村JCO核轉化工廠臨界事故。
2000	金	田村亮子（現名谷亮子）憑女子柔道、高橋尚子憑女子馬拉松獲得悉尼奧運金牌；南北韓領袖金正日、金大中對話；發行2000日圓紙幣；雙胞胎人瑞姐妹中的金婆婆逝世。
2001	戦	美國911事件引發的反恐戰爭；全球經濟不景氣。
2002	帰	日本與北韓領袖首次會談交涉後，帶回數十年前遭北韓特務綁架的5名日本人歸國；日本經濟回復至泡沫經濟前的狀態。
2003	虎	日本政府派遣自衛隊赴伊拉克，如履虎尾；職棒阪神虎隊奪得久違18年的聯盟冠軍。
2004	災	9個颱風侵襲日本，尤以颱風23號「蠍虎」災情嚴重；新潟縣中越地震；大雨和酷熱天氣；福井縣美濱發電所發生熱水及蒸氣洩漏事故，造成5死6傷；三菱汽車設計缺陷隱匿事件。
2005	愛	紀宮清子內親王結婚；敬宮愛子內親王成為女性天皇的可能性提高；日文名與「愛」同音（あい）的女性運動選手（福原愛、宮里藍等）屢創佳績；21世紀首次萬國博覽會「2005年日本國際博覽會：愛・地球博」舉行成功；南亞大海嘯災害及美國卡崔娜風災讓世人感受到互愛的重要。
2006	命	日本皇室新生命悠仁親王出生；校園霸凌問題導致學生自殺事件頻傳；多宗酒後駕車奪命意外；北海道佐呂間町的龍捲風造成重大傷亡。
2007	偽	數起知名食品偽造標示事件、年金記錄缺漏問題、防衛省前事務次官守屋武昌被揭發任內受賄、中國北京市石景山遊樂園侵犯著作權、關西電視台節目《發掘！真有其事大百科2》涉嫌假造實驗數據以宣傳「納豆可以減肥」的不實消息。
2008	変	政治的改變（日本首相的變動、美國總統歐巴馬的競選口號：「Change」）、經濟的改變（全球股市暴跌、日圓匯率急升、原油價格暴起暴落）、食品安全意識的改變等等。
2009	新	取代長期執政的自民黨的民主黨新政權、美國新任總統歐巴馬就職、新型流感蔓延、鈴木一朗再創新紀錄。
2010	暑	夏天创纪录的酷暑导致多人中暑、蔬菜价格猛涨、智利矿难33名矿工熬过酷暑后生还、隼鸟号探测器穿过温度高达1万度的大气层后返回并带回小行星物质。[2]
2011	絆	東日本大地震等大量自然災害，日本國家女足贏得世界盃冠軍，是再次感到「羈絆」的重要性的一年。
2012	金	倫敦奧運選手力爭金牌，取得史上最多的38枚獎牌；日本取得很多「金字塔」式的成就如：京都大學教授山中伸彌榮獲諾貝爾醫學獎、東京晴空塔開幕；時隔932年在日本再次能觀測金環日食（日環食），消費稅稅率上升和生活保護費用騙取問題使金錢問題受到關注。
2013	輪	2020年東京奧運（五輪）申辦成功；東北樂天金鷹首次制霸日本大賽；富士山登錄成為世界遺產；日本國家隊順利晉級2014年世界盃決賽圈…都是民眾團結在一起，凝聚成環狀共同努力的結果。颱風韋帕導致伊豆大島泥石流、颱風海燕導致菲律賓萊特島水災，日本和世界各國都積極互相支援。
2014	税	日本時隔17年調漲消費稅，稅率由5%調漲到8%，給人們生活環境帶來很大變化；原預定2015年10月調漲為10%，但日本首相安倍晉三考量經濟復甦力道，決定延期18個月調漲消費稅，並為此宣布解散眾議院進行大選問信於民。
2015	安	• 安倍晉三政權下，通過新安保法案，贊否兩論，贊成派和反對派都在國會和全國各地舉行集會； • 迎來戰後70週年，民眾希望今後國家都能平安； • 伊斯蘭國殺害兩名日本人質，巴黎發生連環恐怖襲擊……世界上人們安全受到威脅的事態不斷發生； • 全球規模的異常氣候導致世界各地災難頻發，引致人們不安； • 旭化成的樓盤打樁數據篡改問題，使人們對居住環境的不信任和不安增加； • 大眾汽車廢氣排放作弊事件、化血研疫苗造假問題，與生活息息相關的動搖安全性的事件不斷發生； • 搞笑藝人總之很開朗安村憑「放心吧，我有穿哦。」（安心して下さい、穿いてますよ。）的段子博得廣泛人氣。[3]
2016	金	• 里约奥运上日本选手力争金牌； • 达成众多“金字塔”式成就，包括铃木一朗击出其美国职棒生涯的第3000次安打，摔跤选手伊调馨连续获得4届奥运金牌； • 东京都财政和税金使用方面出现问题，如前东京都知事舛添要一涉嫌挪用政治资金，丰洲市场搬迁工程因资金弊案延期，2020年东京奥运主场馆计划因预估资金过高推倒重来等等； • 以富山市議会为首，多位地方议会议员因被揭发不当使用政務活動費而辞职，“政治与金钱”问题愈发显著； • 日本银行推行的负利率等金融货币政策引发关注； • 金髮的不动产富豪（お金持ち）唐纳德·特朗普在美国大选中胜出； • YouTube短片《Pen-Pineapple-Apple-Pen》引发世界潮流，表演者Piko太郎全身着金色服装。[4][5]

Why Japanese uses Syllabaries

- all syllables are:
  [consonant] + vowel + [nasal *n*]
- 10 C x 5 V = 50 syllables
  - plus some variations
- also possible for Mandarin
- other languages have many more syllables: use *alphabets*
  - alphabet = 10+5; syllabary = 10x5
- read the Writing Systems tutorial from course page!



あ ア a	い イ i			
か カ ka	き キ ki			
さ サ sa	し シ shi			
た タ ta	ち チ chi			
な ナ na	に ニ ni	ぬ ヌ nu	ね ネ ne	の ノ no
は ハ ha	ひ ヒ hi	ふ フ hu / fu	へ ヘ he	ほ ホ ho
ま マ ma	み ミ mi	む ム mu	め メ me	も モ mo
や ヤ ya		ゆ ユ yu		よ ヨ yo
ら ラ ra	り リ ri	る ル ru	れ レ re	ろ ロ ro
わ ワ wa	http://brng.jp/ 90459562			を ヲ wo
んン n				

Origins of Katakana

- they come from Chinese characters (kanji's)
  - but often got the wrong (semantic not phonetic) radicals...

ア	阿	イ	伊	ウ	宇	エ	江	オ	於
カ	加	キ	機	ク	久	ケ	介	コ	己
サ	散	シ	之	ス	須	セ	世	ソ	曽
タ	多	チ	千	ツ	川	テ	天	ト	止
ナ	奈	ニ	仁	ヌ	奴	ネ	祢	ノ	乃
ハ	八	ヒ	比	フ	不	ヘ	部	ホ	保
マ	末	ミ	三	ム	牟	メ	女	モ	毛
ヤ	也			ユ	由			ヨ	與
ラ	良	リ	利	ル	流	レ	礼	ロ	呂
ワ	和	ヰ	井			ヱ	恵	ヲ	乎
ン	尓								

Kana Development Chart

Hiragana 平仮名					Katakana 片仮名				
あ 安	い 以	う 宇	え 衣	お 於	ア 阿	イ 伊	ウ 宇	エ 江	オ 於
か 加	き 機	く 久	け 計	こ 己	カ 加	キ 機	ク 久	ケ 介	コ 己
さ 左	し 之	す 寸	せ 世	そ 曽	サ 散	シ 之	ス 須	セ 世	ソ 曽
た 太	ち 知	つ 川	て 天	と 止	タ 多	チ 千	ツ 川	テ 天	ト 止
な 奈	に 仁	ぬ 奴	ね 祢	の 乃	ナ 奈	ニ 仁	ヌ 奴	ネ 祢	ノ 乃
は 波	ひ 比	ふ 不	へ 部	ほ 保	ハ 八	ヒ 比	フ 不	ヘ 部	ホ 保
ま 末	み 美	む 武	め 女	も 毛	マ 末	ミ 三	ム 牟	メ 女	モ 毛
や 也		ゆ 由		よ 与	ヤ 也	丿 以	ユ 由	工 衣	ヨ 与
ら 良	り 利	る 留	れ 礼	ろ 呂	ラ 良	リ 利	ル 流	レ 礼	ロ 呂
わ 和	ゐ 為		ゑ 恵	を 遠	ワ 和	十 井	于 宇	ヱ 恵	ヲ 乎
		ん 无					ン 尓		

Japanese Phonemes (too few sounds!)

CONSONANTS (PULMONIC) © 2005 IPA

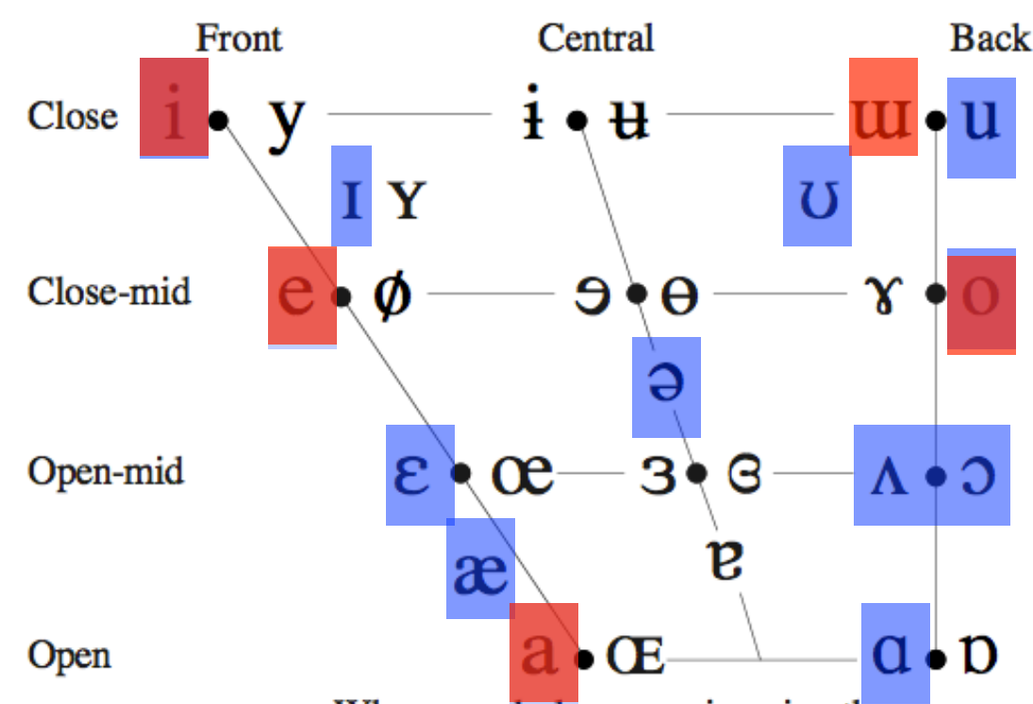
	Bilabial	Labiodental	Dental	Alveolar	Postalveolar	Retroflex	Palatal	Velar	Uvular	Pharyngeal	Glottal
Plosive	p b			t d		ʈ ɖ	c ɟ	k g	q ɢ		ʔ
Nasal	m	ɱ		n		ɳ	ɲ	ŋ	ɴ		
Trill	ʙ			r					ʀ		
Tap or Flap		ⱱ		ɾ		ɽ					
Fricative	ɸ β	f v	θ ð	s z	ʃ ʒ	ʂ ʐ	ç ʝ	x ɣ	χ ʁ	ħ ʕ	h ɦ
Lateral fricative				ɬ ɮ							
Approximant		ʋ		ɹ		ɻ	j	ɰ			
Lateral approximant				l		ɭ	ʎ	ʟ			

Where symbols appear in pairs, the one to the right represents a voiced consonant. Shaded areas denote articulations judged impossible.

Eng

Jap

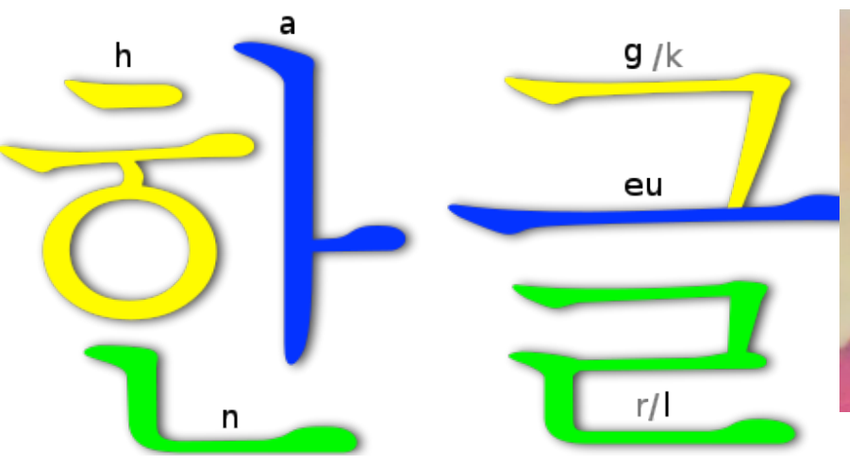
VOWELS

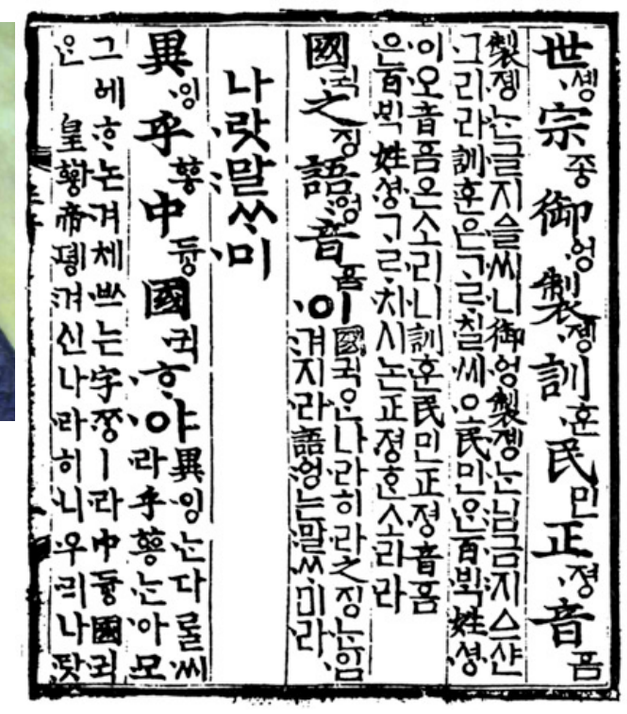


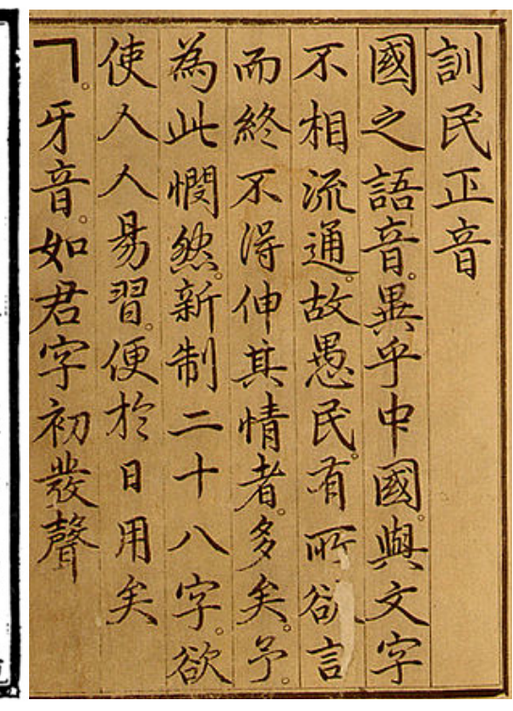
Where symbols appear in pairs, the one to the right represents a rounded vowel.

Aside: Is Korean a Syllabary? No!

- A: Hangul is not a syllabary, but a "featural alphabet"
  - a special alphabet where shapes encode phonological features
- the inventor of Hangul (c. 1440s) was the first real linguist







- 14 consonants: ㄱ*g*, ㄴ*n*, ㄷ*d*, ㄹ*l/r*, ㅁ*m*, ㅂ*b*, ㅅ*s*, ㅇ*null/ng*, ㅈ*j*, ㅊ*ch*, ㅋ*k*, ㅌ*t*, ㅍ*p*, ㅎ*h*
- 5 double consonants: ㄲ*kk*, ㄸ*tt*, ㅃ*pp*, ㅆ*ss*, ㅉ*jj*
- 11 consonant clusters: ㄳ*gs*, ㄵ*nj*, ㄶ*nh*, ㄺ*lg*, ㄻ*lm*, ㄼ*lb*, ㄽ*ls*, ㄾ*lt*, ㄿ*lp*, ㅀ*lh*, ㅄ*bs*
- 6 vowel letters: ㅇㅏ *a*, ㅇㅓ *eo*, ㅗ *o*, ㅜ *u*, ㅡ *eu*, ㅇㅣ *i*
- 4 iotized vowels (with a *y*): ㅇㅑ *ya*, ㅇㅕ *yeo*, ㅛ *yo*, ㅠ *yu*
- 5 (iotized) diphthongs: ㅐ *ae*, ㅒ *yae*, ㅔ *e*, ㅖ *ye*, ㅢ *ui*
- 6 vowels and diphthongs with a *w*: ㅘ *wa*, ㅙ *wae*, ㅚ *oe*, ㅝ *wo*, ㅞ *we*, ㅟ *wi*

Q: 강남 스타일 = ?

Katakana Transliteration Examples

- コンピューター
- ko n py u - ta -
- kompyuutaa (uu=û)
- computer

- アイスクリーム
- a i su ku ri - mu
- aisukuriimu
- ice cream

- アンドリュー・ビタビ
- andoryuubitabi
- Andrew Viterbi

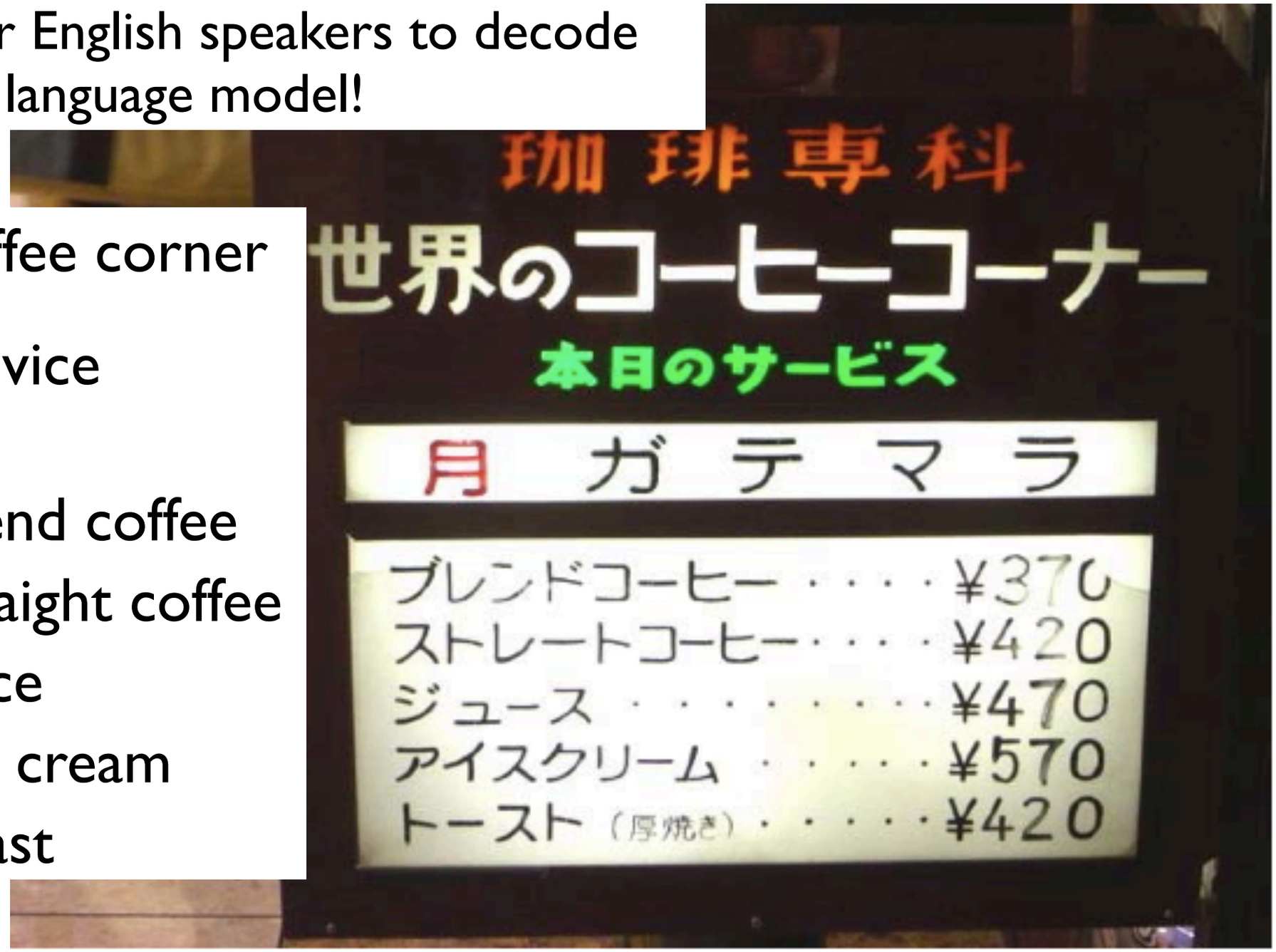
- ヨーグルト
- yo - gu ru to
- yogurt

Katakana on Streets of Tokyo

Japanese just transliterates almost everything
(even though its syllable inventory is really small...)
but... it is quite easy for English speakers to decode
.... if you have a good language model!

from Knight & Sproat 09

koohiikoonaa	coffee corner
saabisu	service
bulendokoohii	blend coffee
sutoreetokoohii	straight coffee
juusu	juice
aisukuriimu	ice cream
toosuto	toast



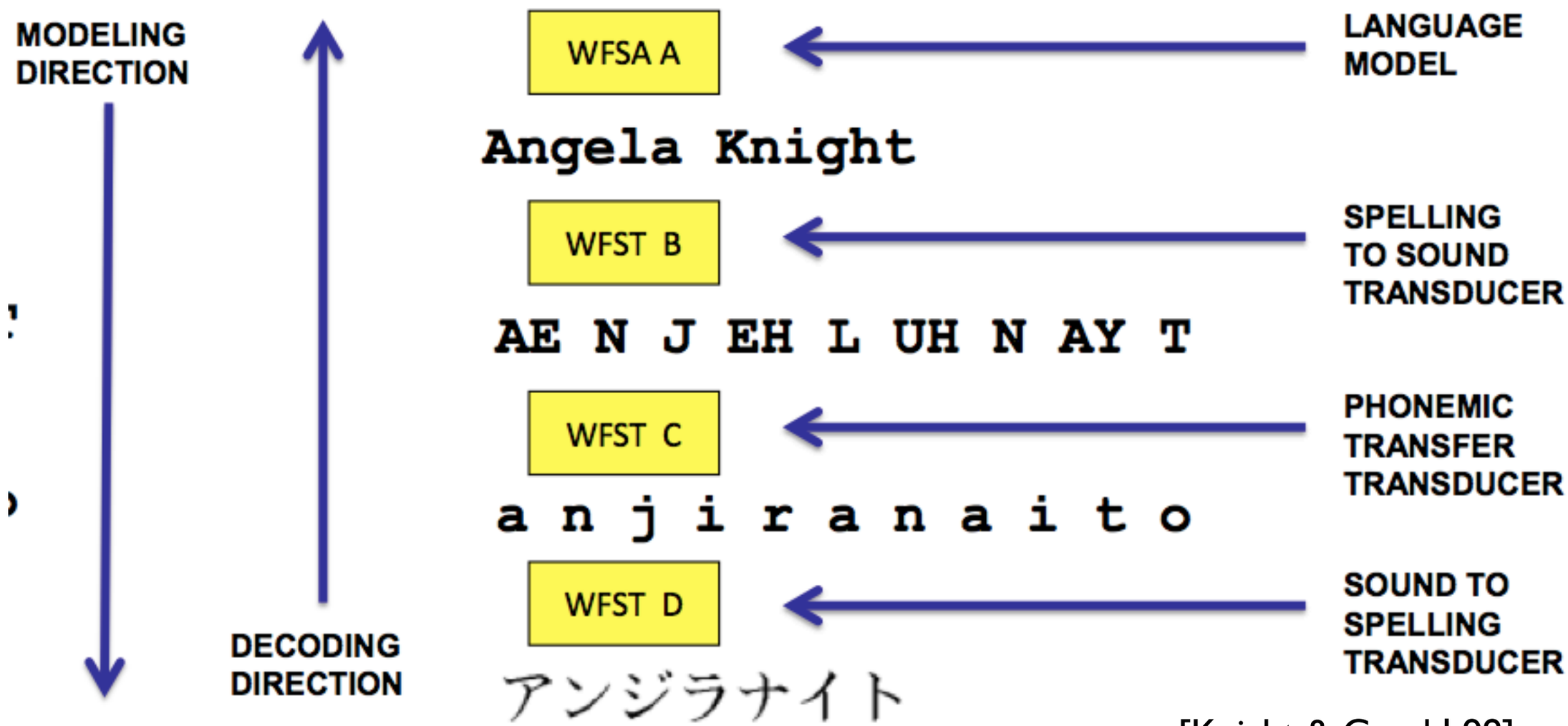
Katakana on Streets of Tokyo

More Japanese Transliterations

• rapputoppu	ラプトプ	• laptop
• bideoteepu	ビデオテープ	• video tape
• shoppingusentaa	ショッピングセンター	• shopping center
• shiitoberuto	シートベルト	• seat belt
• chairudoshiito	チャイルトシート	• child seat
• andoryuubitabi	アンドリュー・ビタビ	• Andrew Viterbi
• bitabiarugorizumu	ビタビアルゴリズム	• Viterbi Algorithm

(hw2) Katakana => English

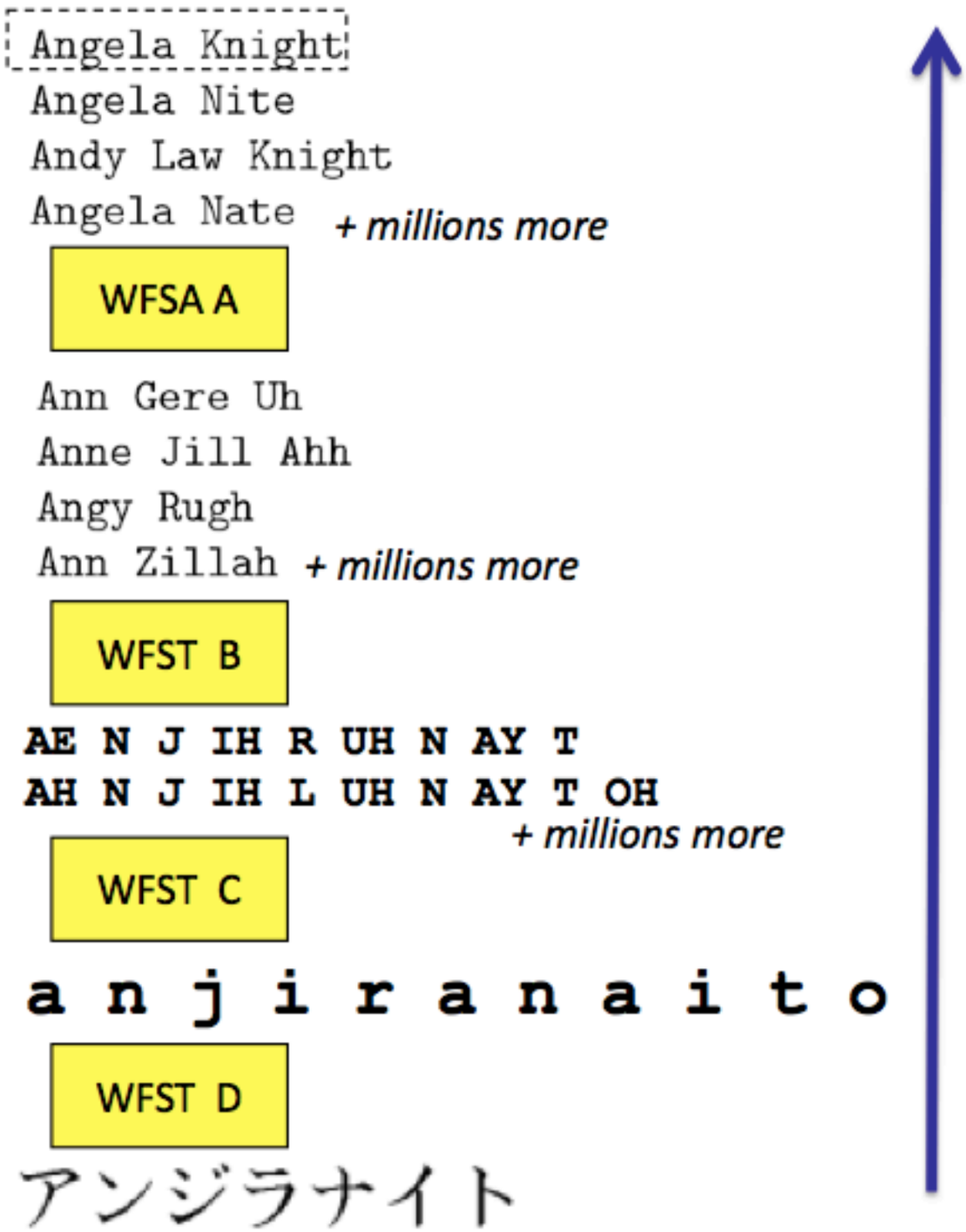
- your job in HW2: decode Japanese Katakana words (transcribed in Romaji) back to English words
  - koohiikoonaa => coffee corner



[Knight & Graehl 98]

(hw2) Katakana => English

- Decoding (HW3)
  - really decipherment!
- what about duplicate strings?
  - from different paths in WFST!
  - n-best cruching, or...
  - weighted determinisation
    - see extra reading on course website for Mohri+Riley paper

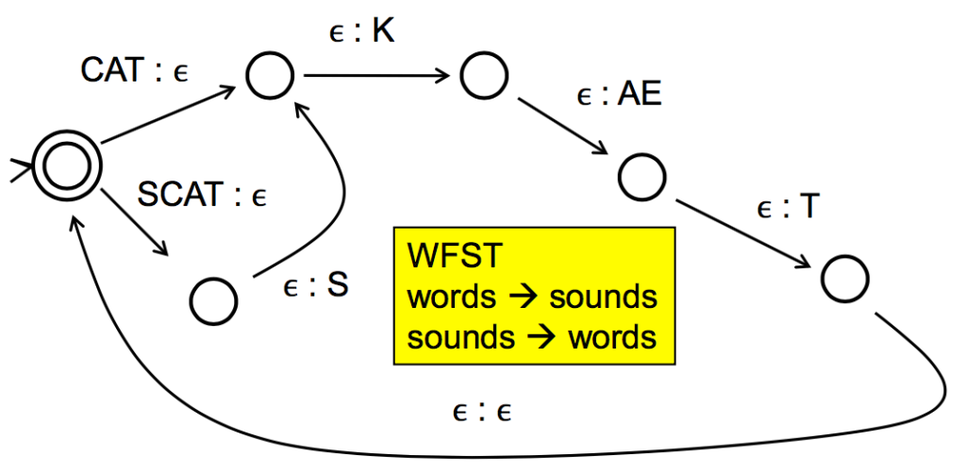


[Knight & Graehl 98]

How to Learn p(e|w) and p(j|e)?

- Such a system captures an infinite relation of <sound-sequence, writing-sequence> pairs.

HW2
eword-epron.data



Learning Sequence Transformation Probabilities

HW2
epron-jpron.data
(MLE)

Ideal training data:

etc

P(n | M) = 0.5
P(m u | M) = 0.5

need much more data, of course

HW3
Viterbi decoding

HW4
epron-jpron.data
(EM)

Actual training data:

etc

Automatically align string pairs using the unsupervised Expectation-Maximization (EM) algorithm.

String Transformations

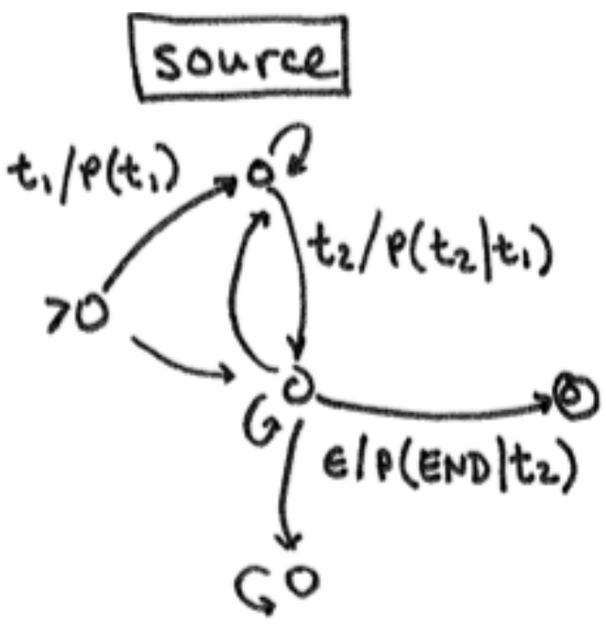
- General Framework for many NLP problems
- Examples

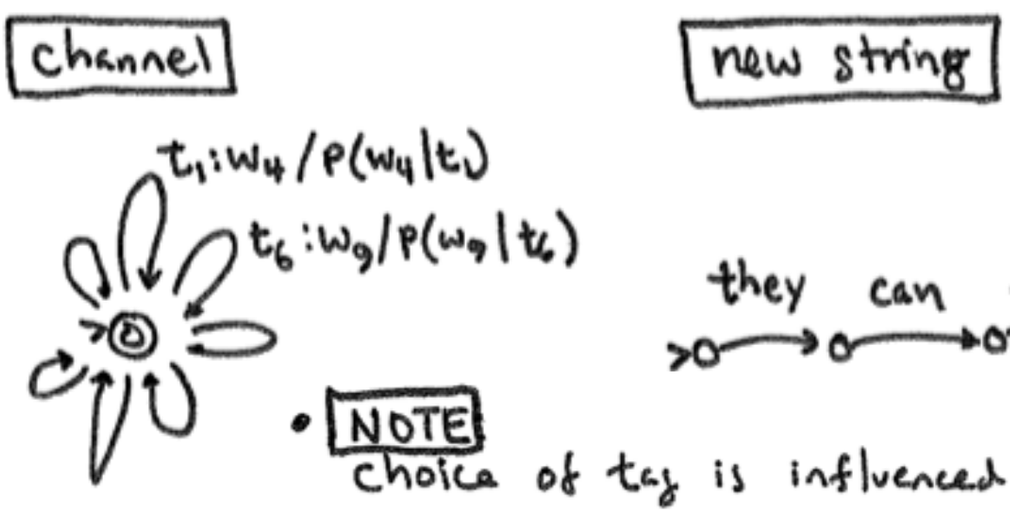
  - Part-of-Speech Tagging
  - Spelling Correction (Edit Distance)
  - Word Segmentation
  - Transliteration, Sound/Spelling Conversion, Morphology
  - Chunking (Shallow Parsing)
  - Beyond Finite-State Models (i.e., tree transformations)
    - Summarization, Translation, Parsing, Information Retrieval, ...
- Algorithms: Viterbi (both max and sum)

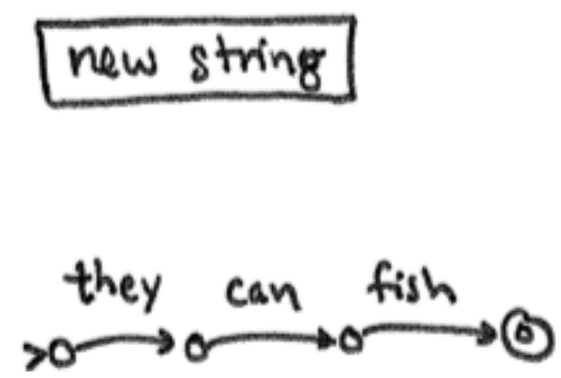
Example 2: Part-of-Speech Tagging

$$P(t \cdots t \mid w \cdots w)$$
$$\sim P(t \cdots t) \cdot P(w \cdots w \mid t \cdots t)$$
$$\sim \underbrace{P(t_1) \cdot P(t_2 \mid t_1) \cdots P(t_n \mid t_{n-1})}_{\text{local grammar preference}} \cdot \underbrace{P(w_1 \mid t_1) \cdots P(w_n \mid t_n)}_{\text{lexical preference}}$$

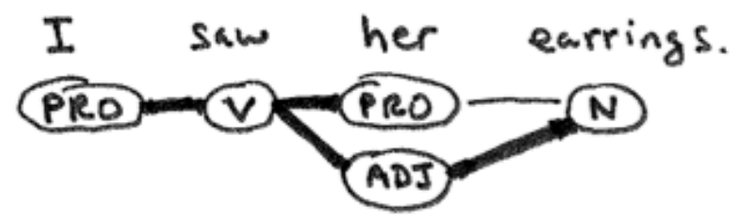
- use tag bigram as a language model
- channel model is context-indep.







- NOTE
  choice of tag is influenced by <u>both</u> left & right context

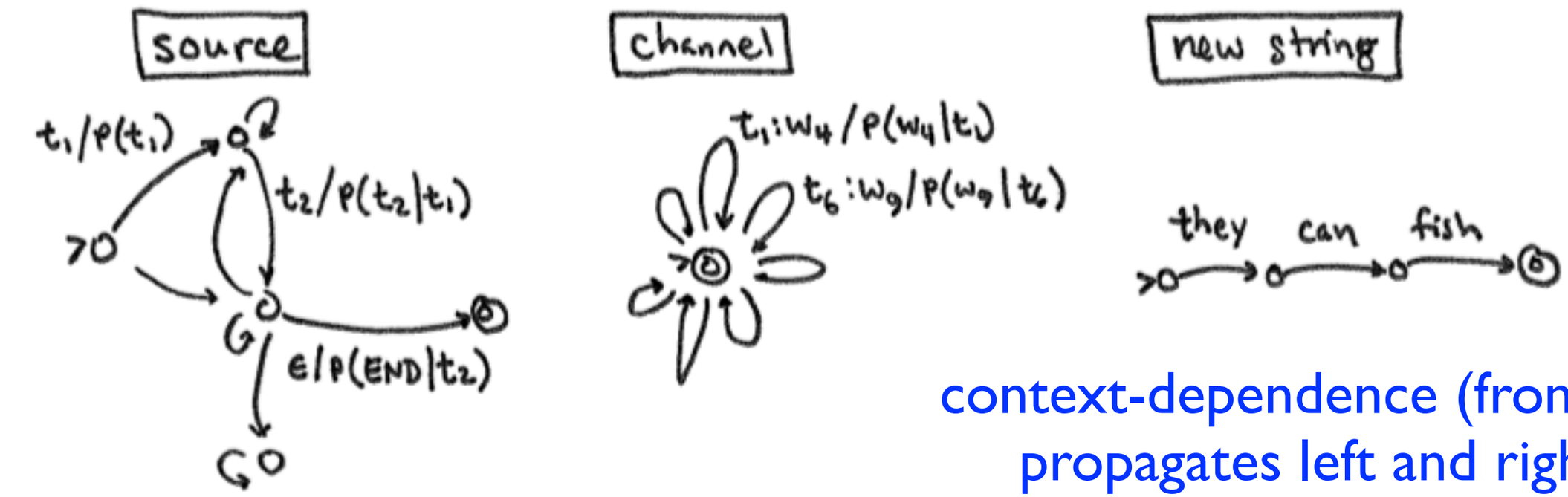


- NOTE
  influences can theoretically be long ranging

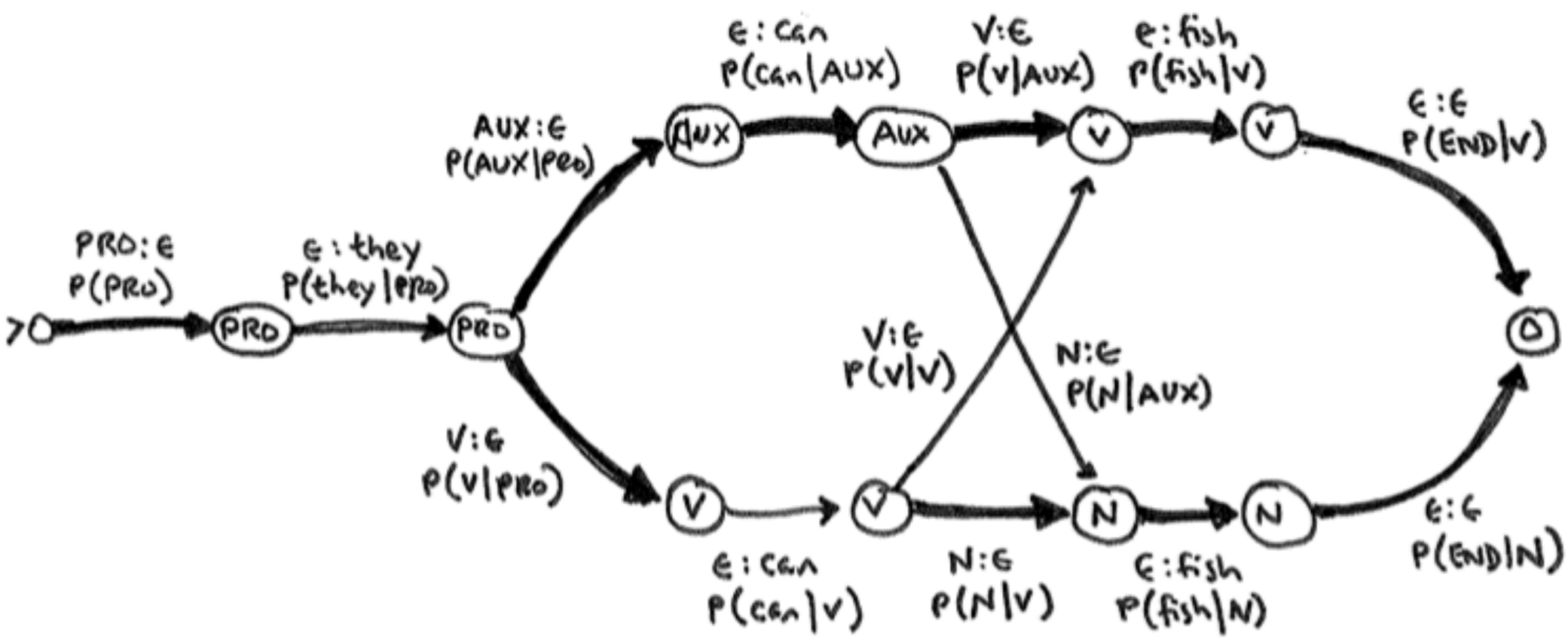
Work out the compositions

- if you want to implement Viterbi...
- case 1: language model is a tag unigram model
  - $p(t...t) = p(t_1)p(t_2) \ldots p(t_n)$
  - how many states do you get?
- case 1: language model is a tag bigram model
  - $p(t...t) = p(t_1)p(t_2 \mid t_1) \ldots p(t_n \mid t_{n-1})$
  - how many states do you get?
- case 3: language model is a tag trigram model...

The case of bigram model

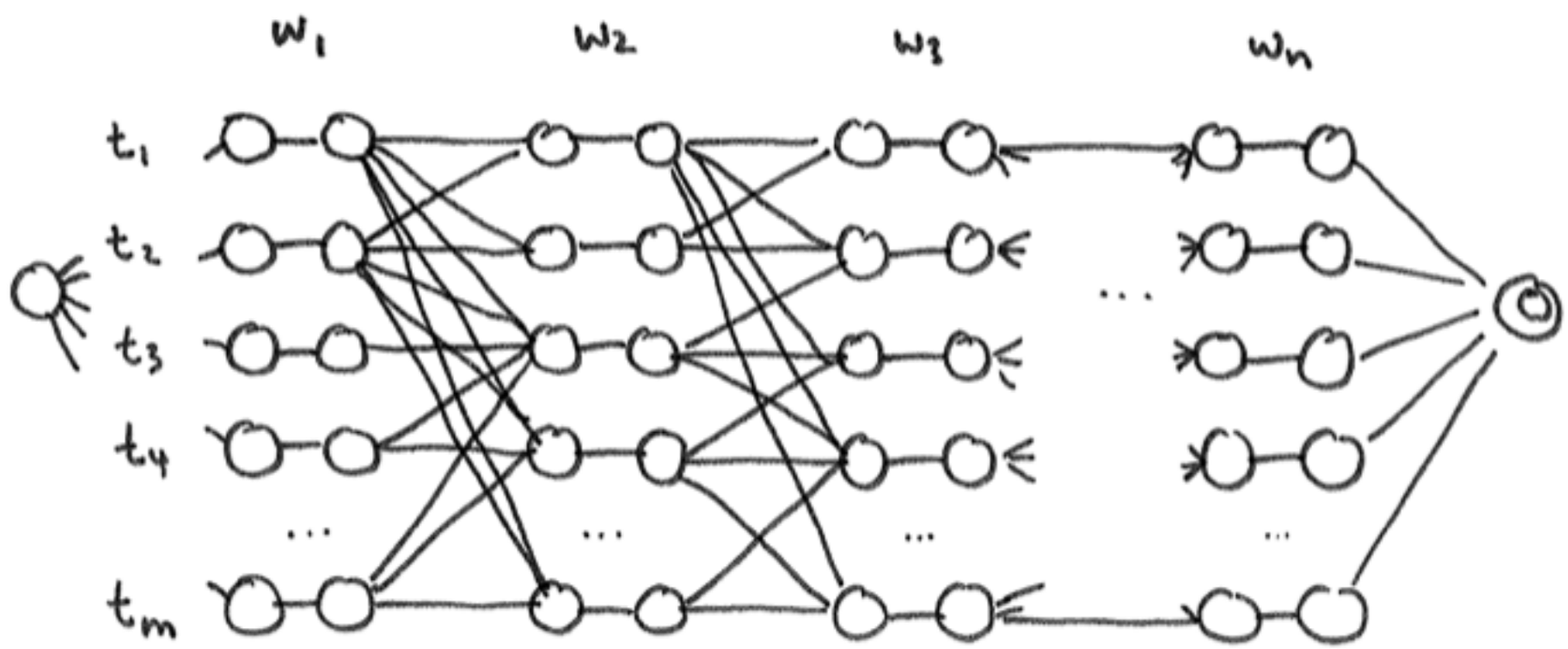


context-dependence (from LM)
propagates left and right!

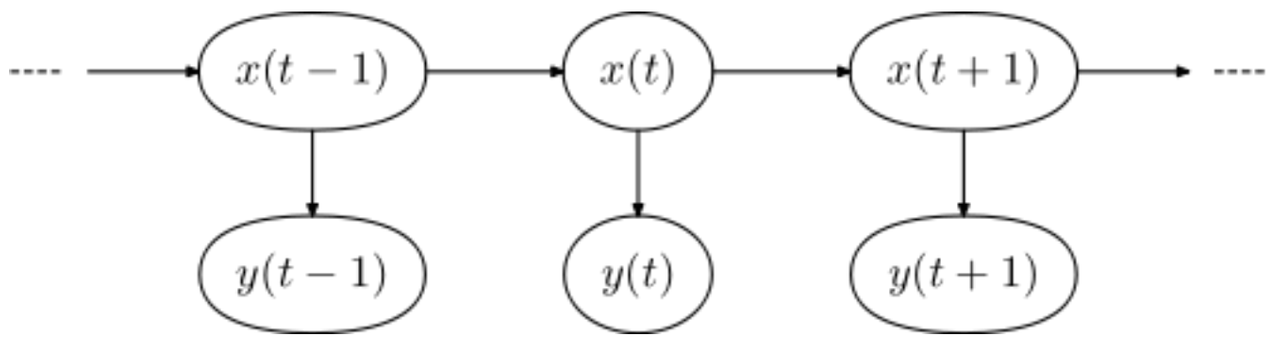


In general...

- bigram LM with context-independent CM
  - O(n m) states after composition
- g-gram LM with context-independent CM
  - O(n m^{g-1}) states after composition
  - the g-gram LM itself has O(m^{g-1}) states



HMM Representation



- HMM representation is not explicit about the search
  - "hidden states" have choices over "variables"
  - in FST composition, paths/states are explicitly drawn

Viterbi for argmax

Viterbi search for $\underset{t \cdots t}{\operatorname{argmax}}\ P(t \cdots t) \cdot P(w \cdots w \mid t \cdots t)$:

```
for j = 1 to m
   Q[1,j] = P(t_j) · P(w_1 | t_j)

for i = 2 to n
   for j = 1 to m
      Q[i,j] = 0
      best-pred[i,j] = 0
      best-score = -∞
      for k = 1 to m
         r = P(t_j | t_k) · P(w_i | t_j) · Q[i-1, k]
         if r > best-score
            best-score = r
            best-pred[i,j] = k
            Q[i,j] = r

final-best = 0
final-score = -∞
for j = 1 to m
   if Q[n,j] > final-score
      final-score = Q[n,j]
      final-best = j

print t_final-best
current = final-best
for i = n-1 down to 1
   current = best-prev[i+1, current]
   print t_current
```

$Q[i,j]$ = cost of shortest path ending with word i getting assigned tag j.

sets back pointers

how about unigram?

prints best tags in reverse order

Python implementation

Complete this Python code implementing the Viterbi algorithm for part-of-speech tagging. It should print a list of word/tag pairs, e.g. `[('a', 'D'), ('can', 'N'), ('can', 'A'), ('can', 'V'), ('a', 'D'), ('can', 'N')]`.

Q: what about top-down recursive + memoization?

```
from collections import defaultdict

best = defaultdict(lambda : defaultdict(float))
best[0]["<s>"] = 1
back = defaultdict(dict)

words = "<s> a can can can a can </s>".split()

tags = {"a": ["D"], "can": ["N", "A", "V"], "</s>": ["</s>"]}  # possible tags for each word
ptag = {"D": {"N": 1}, "V": {"</s>": 0.5, "D":0.5}, ... }      # ptag[x][y] = p(y | x)
pword = {"D": {"a": 0.5}, "N": {"can": 0.1}, ...        }      # pword[x][w] = p(w | x)

for i, word in enumerate(words[1:], 1):                        # i=1,2...; word=a,can,...
    for tag in tags[word]:
        for prev in best[i-1] :
            if tag in ptag[prev] :
                score = best[i-1][prev]  * ptag[prev][tag]  * pword[tag][word]
                if score > best[i][tag]:
                    best[i][tag] = score
                    back[i][tag] = prev

def backtrack(i, tag):
    if i == 0:
        return []
    return backtrack(i-1, back[i][tag])  + [(words[i], tag)]

print backtrack(len(words)-1, "</s>")[:-1]
```

Viterbi Tagging Example

prob of \ given	START	PRO	V	N	AUX
END		.1	.1	.1	.1
PRO	.6				
V	.05	.6		.2	.9
N	.3		.9	.7	
AUX	.05				

prob of \ given	PRO	V	N	AUX
they	.07			
can		10^{-5}	10^{-4}	.21
fish		10^{-4}	10^{-4}	

Q1. why is this table *not* normalized?

Q2. is "fish" equally likely to be a V or N?

Q3: how to train p(w|t)?

	they	can	fish
PRO	Q = P(PRO|START)·P(they|PRO) = .6 · .07 = .042	Q = 0 P(PRO|PRO) = 0 P(can|PRO) = 0 !	Q = 0
V	Q = 0 P(they|V) = 0	Q = max⟨.042 · .6 · 10^{-5}⟩ = .000000252 bp = PRO	Q = max⟨.000000252 · 0 · 10^{-4}, .002646 · .9 · 10^{-4}⟩ = .0000023814 bp = AUX
N	Q = 0	Q = 0 P(N|PRO) = 0	Q = max⟨.000000252 · .9 · 10^{-4}, .002646 · 0 · 10^{-4}⟩ = .0000000002268 bp = V
AUX	Q = 0	Q = max⟨.042 · .3 · .21⟩ = .002646 bp = PRO	Q = 0

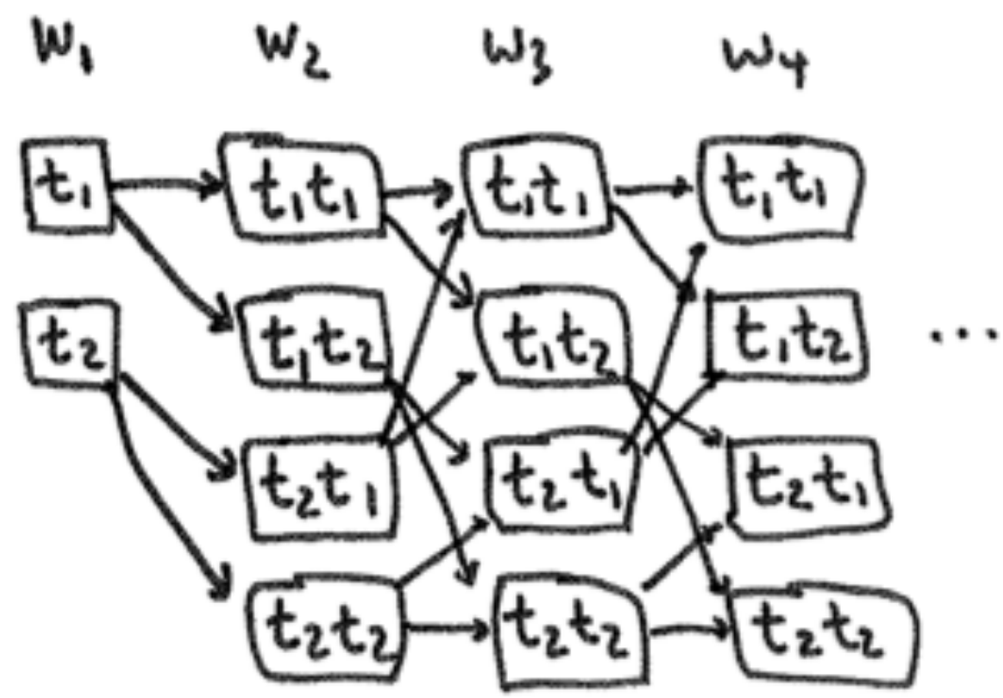
bp = V

$$Q[1,j] = P(t_j \mid START) \cdot P(w_1 \mid t_j)$$

$$Q[i,j] = \max_k Q[i-1,k] \cdot P(t_j \mid t_k) \cdot P(w_i \mid t_j)$$

Trigram HMM

```
for j = 1 to m
   Q1[1,j] = ...

for j = 1 to m
   for j2 = 1 to m
      Q[2,j,j2] = ...

for i = 3 to n
   for j = 1 to m
      for j2 = 1 to m
         Q[i,j,j2] = 0
         best-pred[i,j,j2] = 0
         best-score = -∞
         for k = 1 to m
            r = P(t_j2 | t_k t_j) · P(w_i | t_j2) · Q[i-1, k, j]
            if r > best-score ...
```



time complexity: $O(nT^3)$

in general: $O(nT^g)$ for g-gram

A Side Note on Normalization

NOTE

final-best gives $P(t \cdots t) \cdot P(w \cdots w \mid t \cdots t)$

but this is not the same as $P(t \cdots t \mid w \cdots w)$

e.g. suppose there is only one $t \cdots t$ (all words unambiguous)

then $P(t \cdots t \mid w \cdots w) = 1$

need to divide

$$P(t \cdots t \mid w \cdots w) = \frac{P(t \cdots t) \cdot P(w \cdots w \mid t \cdots t)}{P(w \cdots w)} = \frac{P(t \cdots t) \cdot P(w \cdots w \mid t \cdots t)}{\sum_{t \cdots t} P(t \cdots t) \cdot P(w \cdots w \mid t \cdots t)}$$

how to compute the normalization factor?

Forward (sum instead of max)

Forward search: $\sum_t P(t) \cdot P(w|t) = P(w)$

$\alpha[1,j] = P(t_j|START) \cdot P(w_1|t_j)$

$\alpha[i,j] = \text{sum}_k\ \alpha[i-1,k] \cdot P(t_j|t_k) \cdot P(w_i|t_j)$

no back pointer

$P(w) = \text{sum}_k\ \alpha[n,k]$

"Forward" procedure for $P(w \cdots w)$

```
for j = 1 to m
  α[1,j] = P(t_j) · P(w_1|t_j)

for i = 2 to n
  for j = 1 to m
    α[i,j] = 0
    for k = 1 to m
      α[i,j] += P(t_j|t_k) · P(w_i|t_j) · α[i-1,k]

P(w···w) = 0
for j = 1 to m
  P(w···w) += α[n,j]
```

$\alpha[i,j]$ = costs of all paths ending w/ word w_i getting tag t_j (costs summed)

Forward vs. Argmax

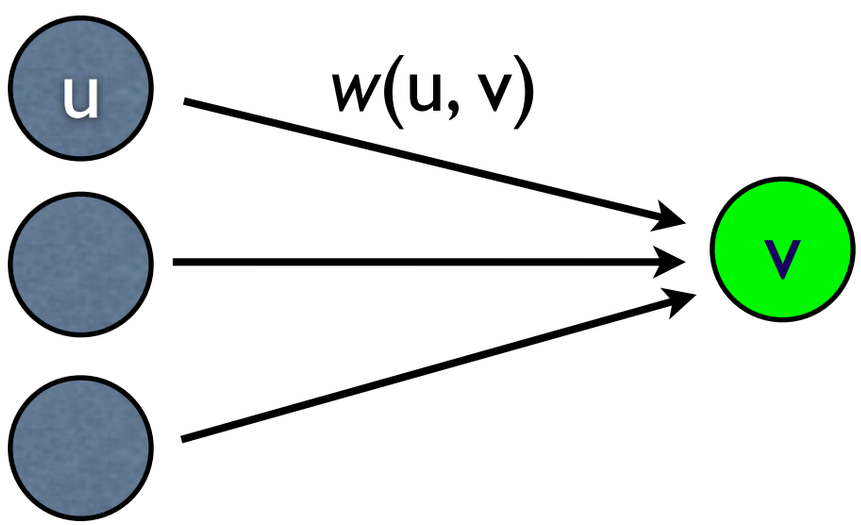
- same complexity, different semirings (+, x) vs (max, x)
- for g-gram LM with context-indep. CM
  - time complexity $O(n\ m^g)$ space complexity $O(n\ m^{g-1})$

```
for j = 1 to m
   Q1[1,j] = ...

for j = 1 to m
   for j2 = 1 to m
      Q[2,j,j2] = ...

for i = 3 to n
   for j = 1 to m
      for j2 = 1 to m
         Q[i,j,j2] = 0
         best-pred[i,j,j2] = 0
         best-score = -∞
         for k = 1 to m
            r = P(t_j2 | k t_j) · P(w_i | t_j2) · Q[i-1, k, j]
            if r > best-score ...
```

$\boxed{O(nm^3) \text{ complexity}}$

Viterbi for DAGs with Semiring

1. topological sort $(A, \oplus, \otimes, \bar{0}, \bar{1})$
2. visit each vertex v in sorted order and do updates
   - for each incoming edge (u, v) in E
   - use d(u) to update d(v): $d(v) \oplus = d(u) \otimes w(u, v)$
   - key observation: d(u) is fixed to optimal at this time



see tutorial on DP from course page

- time complexity: O(V + E)

Example: Word Segmentation

- you noticed that Japanese (e.g., Katakana) is written *without* spaces between words
  - in order to guess the English you also do segmentation
  - e.g. アイスクリーム => アイス クリーム => ice cream
  - how about "gaaruhurendo" and "shingururuumu" ?
- this is an even more important issue in Chinese
  - 南京市长江大桥
- also in other East Asian Languages
- also in English: sounds => words (speech recognition)

What if English were written as Chinese...

- thisisacoursetaughtinthefallsemesterofthisyearatusc
- actually, Latin used to be written exactly like this!
  - “scripta continua” => “interpuncts” (center dots) =>
- this might be a final project topic (on the easier side)

Chinese Word Segmentation

民主

min-zhu

people-dominate

"democracy"

this was 5 years ago.

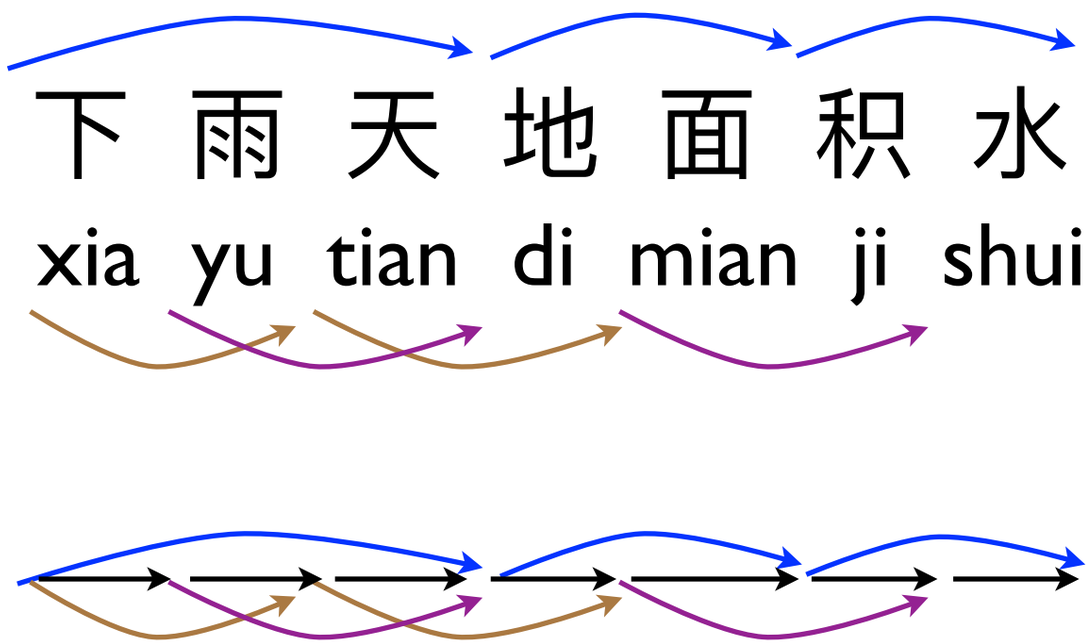
now Google is
good at segmentation!

江泽民 主席

jiang-ze-min zhu-xi

... - ... - *people* *dominate-podium*

"President Jiang Zemin"



下 雨 天 地 面 积 水

graph search

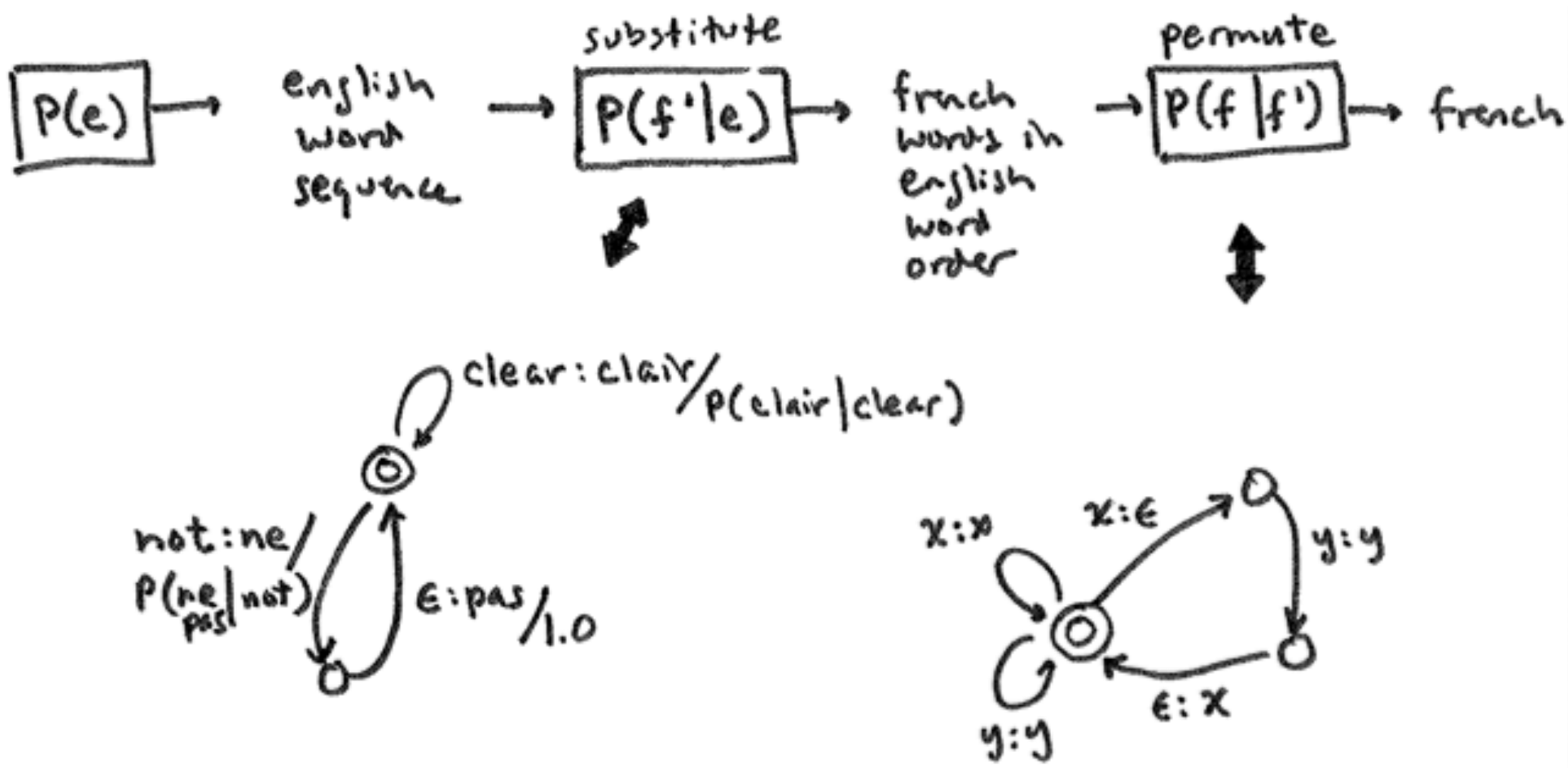
tagging problem

Word Segmentation Cascades

- a good idea for final project (Chinese/Japanese)

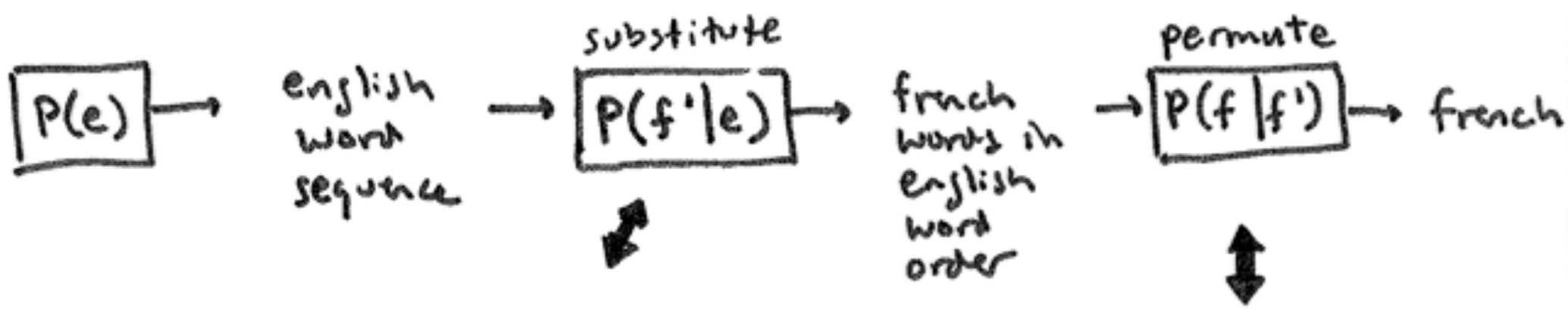
Machine Translation

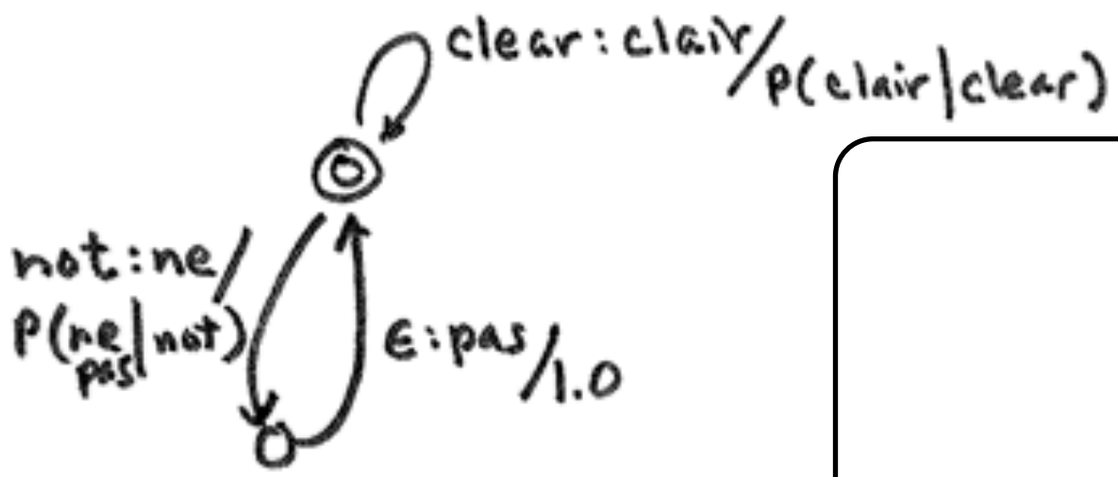
- simplest model: word-substitution and permutation
- does it really work??



Machine Translation Permutation

- how would you model permutation in FSTs?



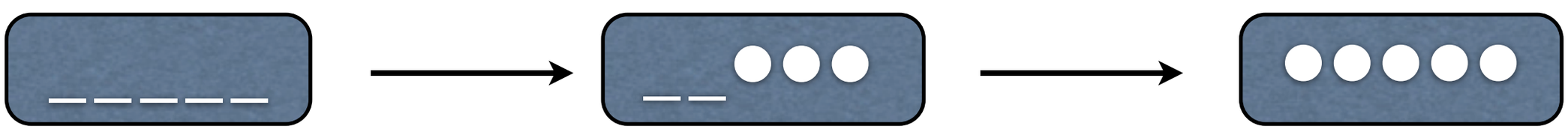


Phrase-based Decoding

与 沙龙 举行 了 会谈

yu Shalong *juxing le huitan*

held a talk with Sharon



with	Sharon held	talks
with Sharon	held a talk	

yu Shalong juxing le huitan

Phrase-based Decoding

与　沙龙　　举行　了　会谈

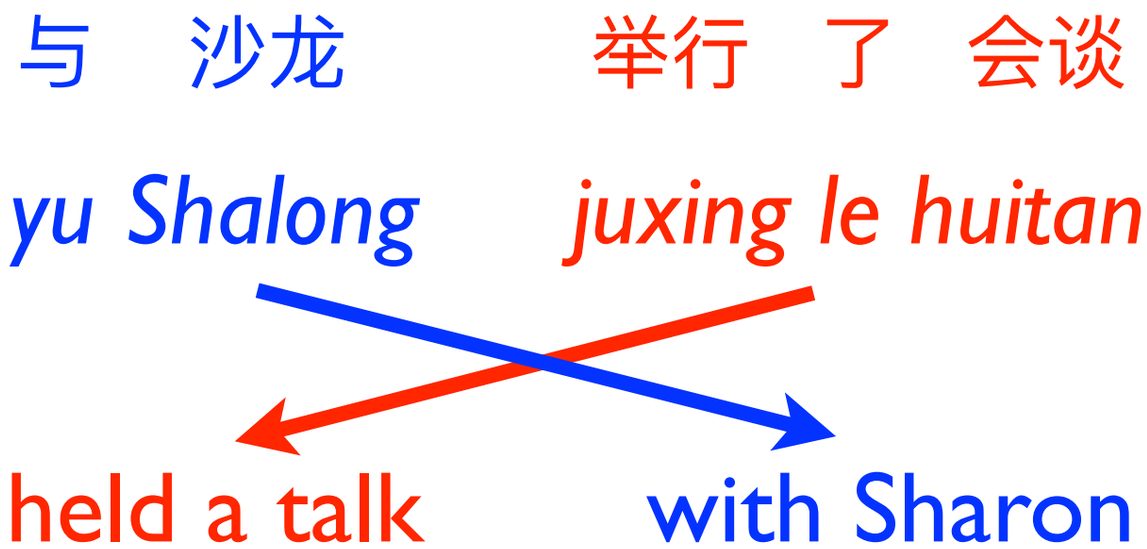
yu Shalong　　*juxing le huitan*

held a talk　　with Sharon

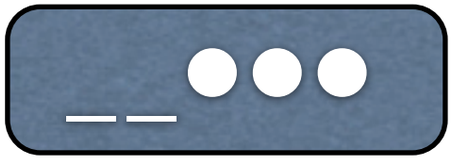
with	Sharon held	talks
with Sharon	held a talk	

yu　*Shalong*　*juxing*　*le*　*huitan*

Phrase-based Decoding

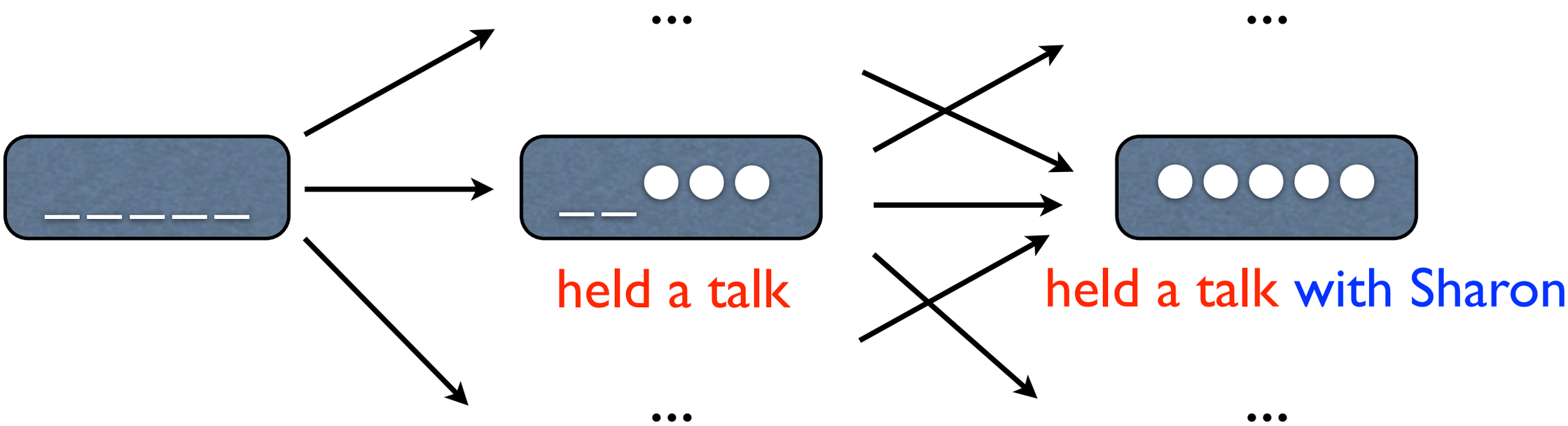


source-side: coverage vector



held a talk

target-side: grow hypotheses strictly left-to-right



space: $O(2^n)$, time: $O(2^n\, n^2)$ -- cf. traveling salesman problem

Phrase-based Cascades

- english LM => (english) => phrase substitutions (n^2) => (foreign phrases in english word order) => permutations (2^n)=> (foreign)
- a good idea for final project (on the harder end)
- wait, where does the phrase table come from?
  - => word-aligned english-foreign sentence pairs

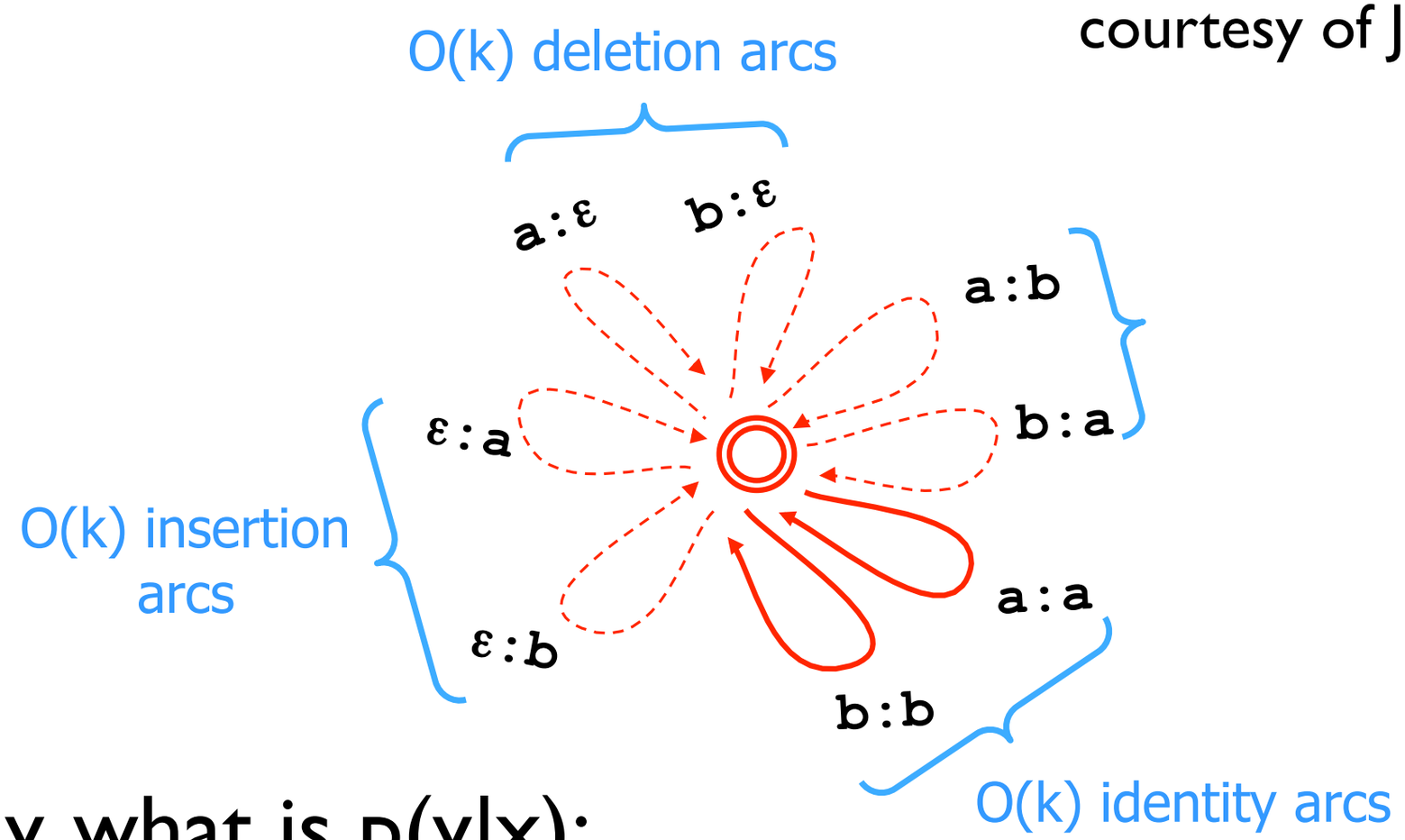
Traveling Salesman Problem & MT

- a classical NP-hard problem
  - goal: visit each city once and only once
- exponential-time dynamic programming
  - state: cities visited so far (bit-vector)
  - search in this $O(2^n)$ transformed graph
- MT: each city is a source-language word
  - restrictions in reordering can reduce complexity => distortion limit
  - => syntax-based MT

(Held and Karp, 1962; Knight, 1999)

Example: Edit Distance

courtesy of Jason Eisner



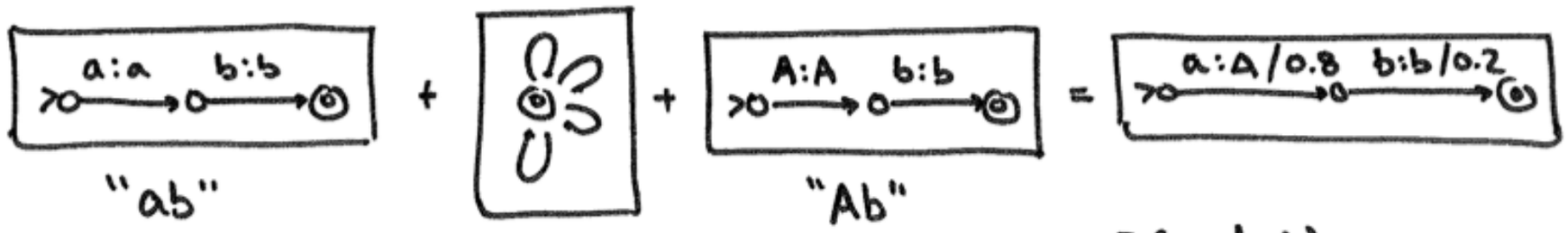
- a) given x, y, what is p(y|x);
- b) what is the most likely seq. of operations?
- c) given x, what is the most likely output y?
- d) given y, what is the most likely input x (with LM) ?

Edit Distance can model...

- part-of-speech tagging
- transliteration
- sound-spelling conversion
- word-segmentation

Given x and y...

- given x, y: a) what is p(y | x)? (sum of all paths)
  b) what is the most likely conversion path?

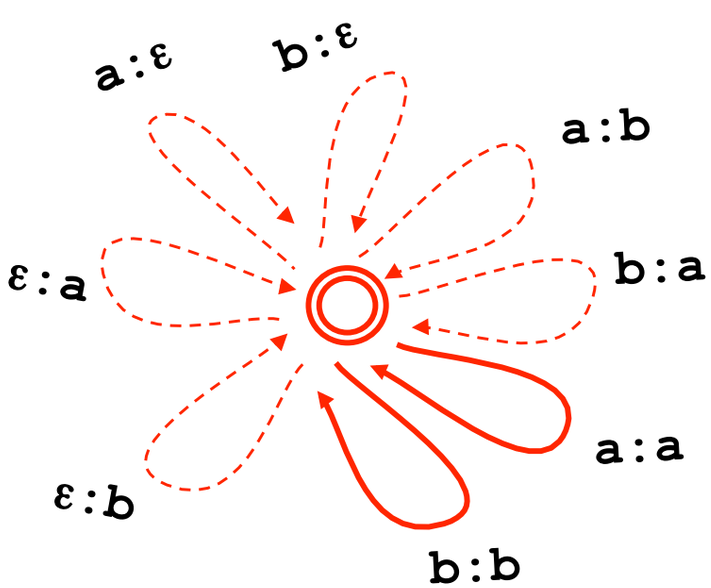


$P(Ab|ab) = 0.16$

Best path (by Dijkstra's algorithm)

clara

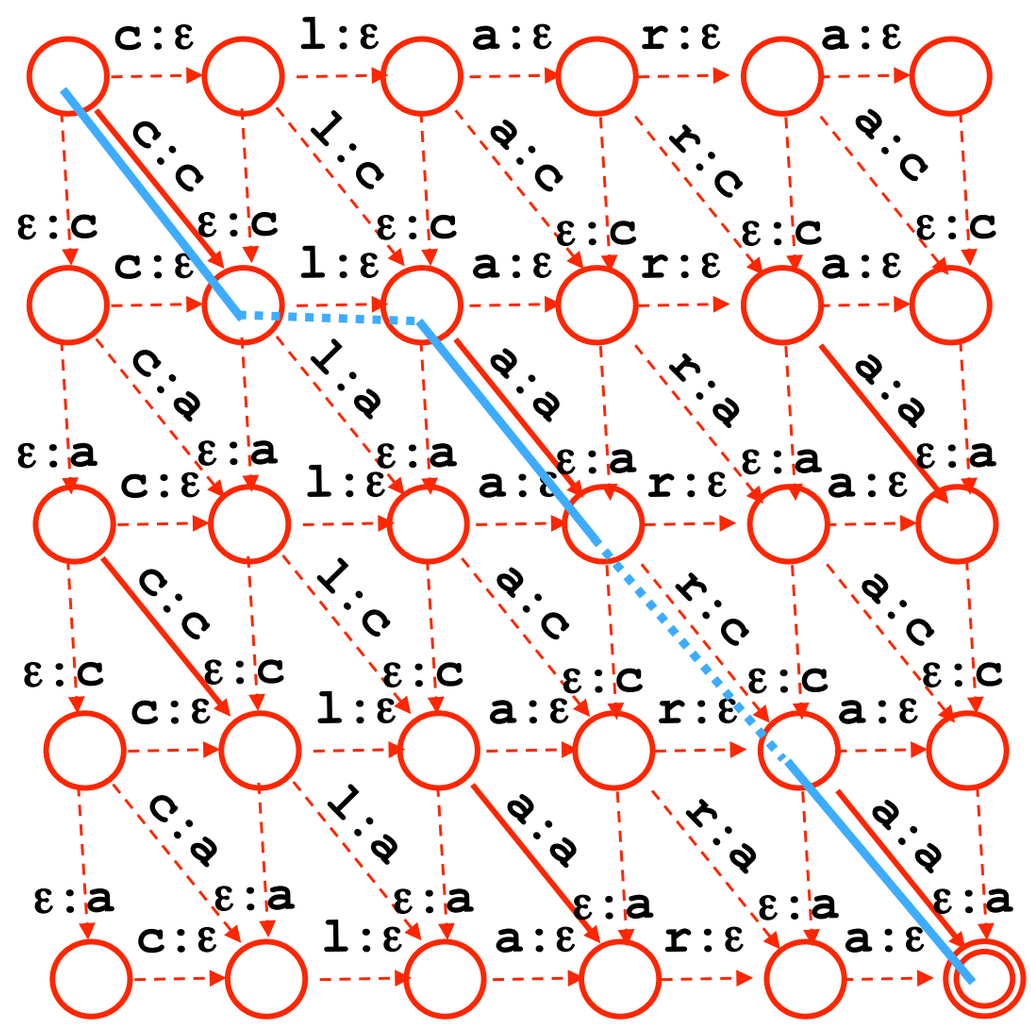
.o.



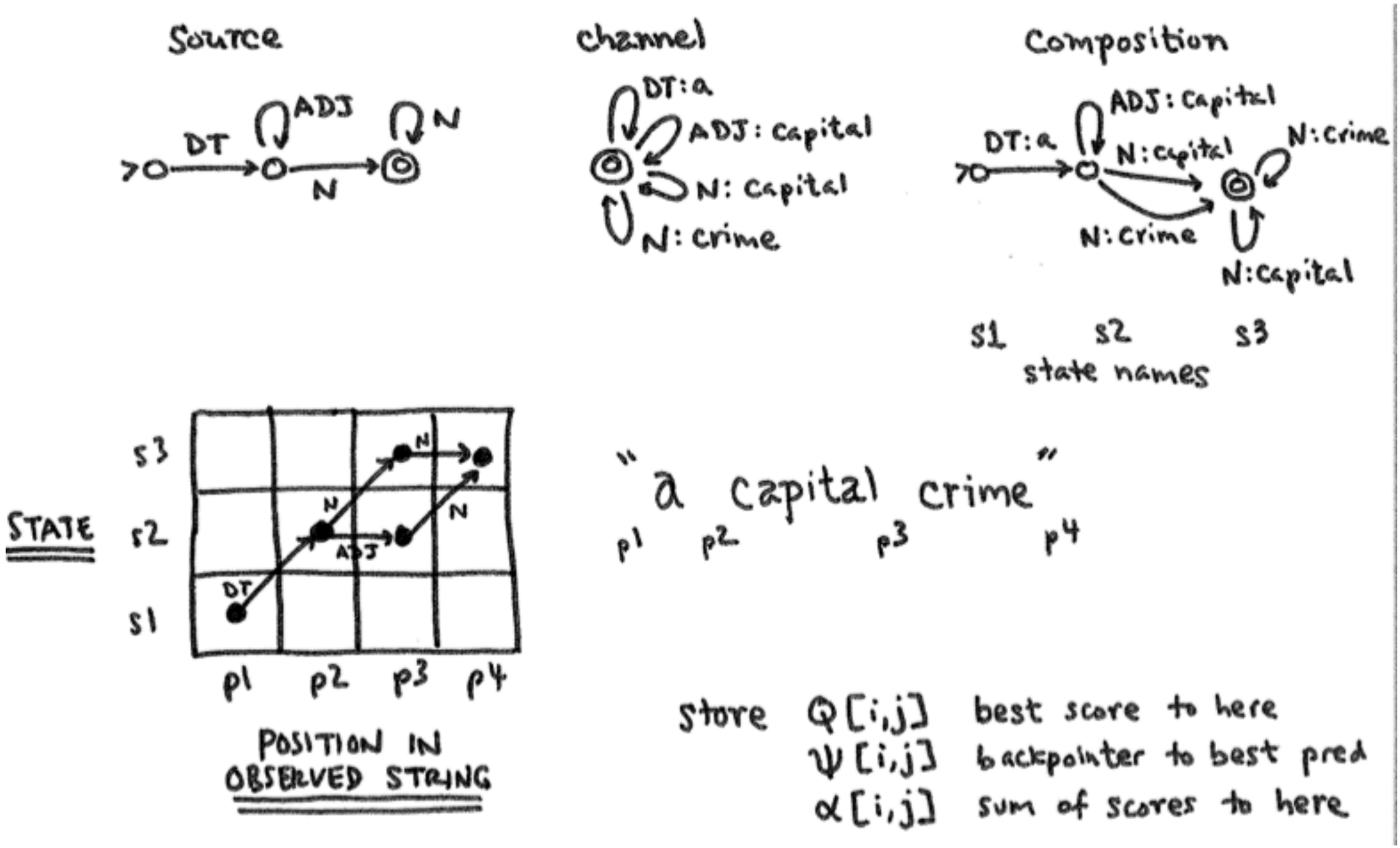
.o.

caca

=

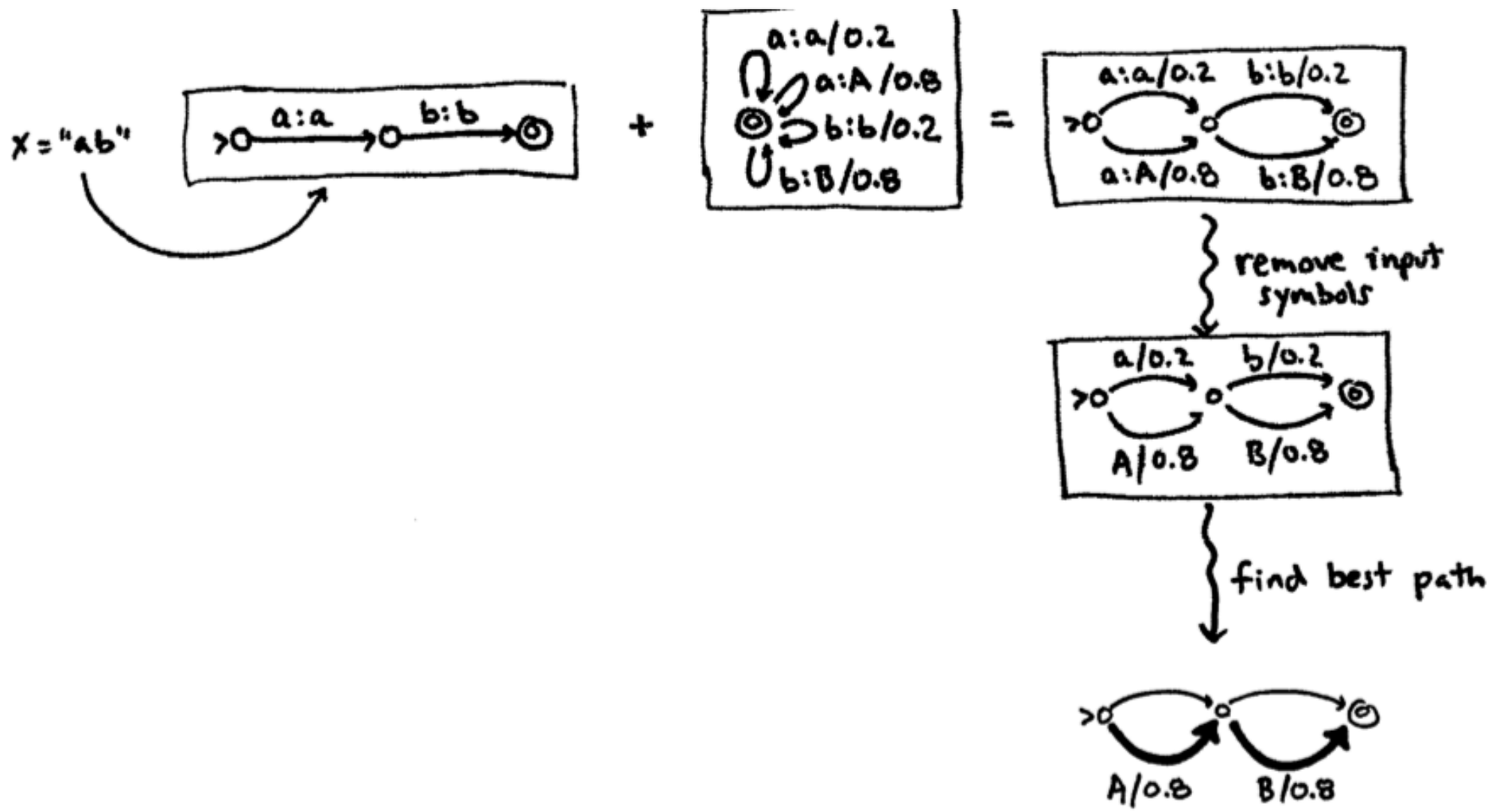


Example: General Tagging



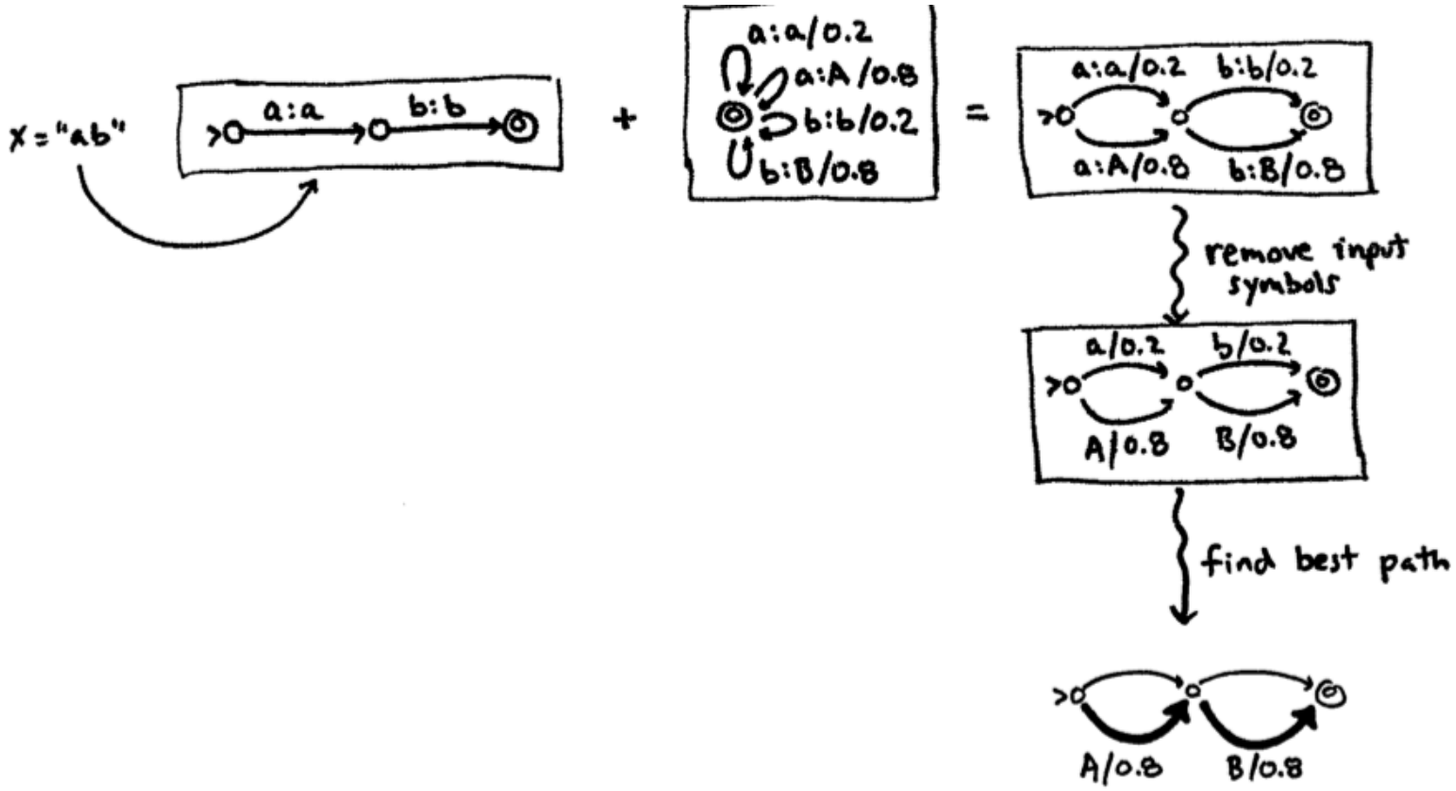
Most Likely "Corrupted Output"

- c) given correct English x, what's the corrupted y with the highest score?



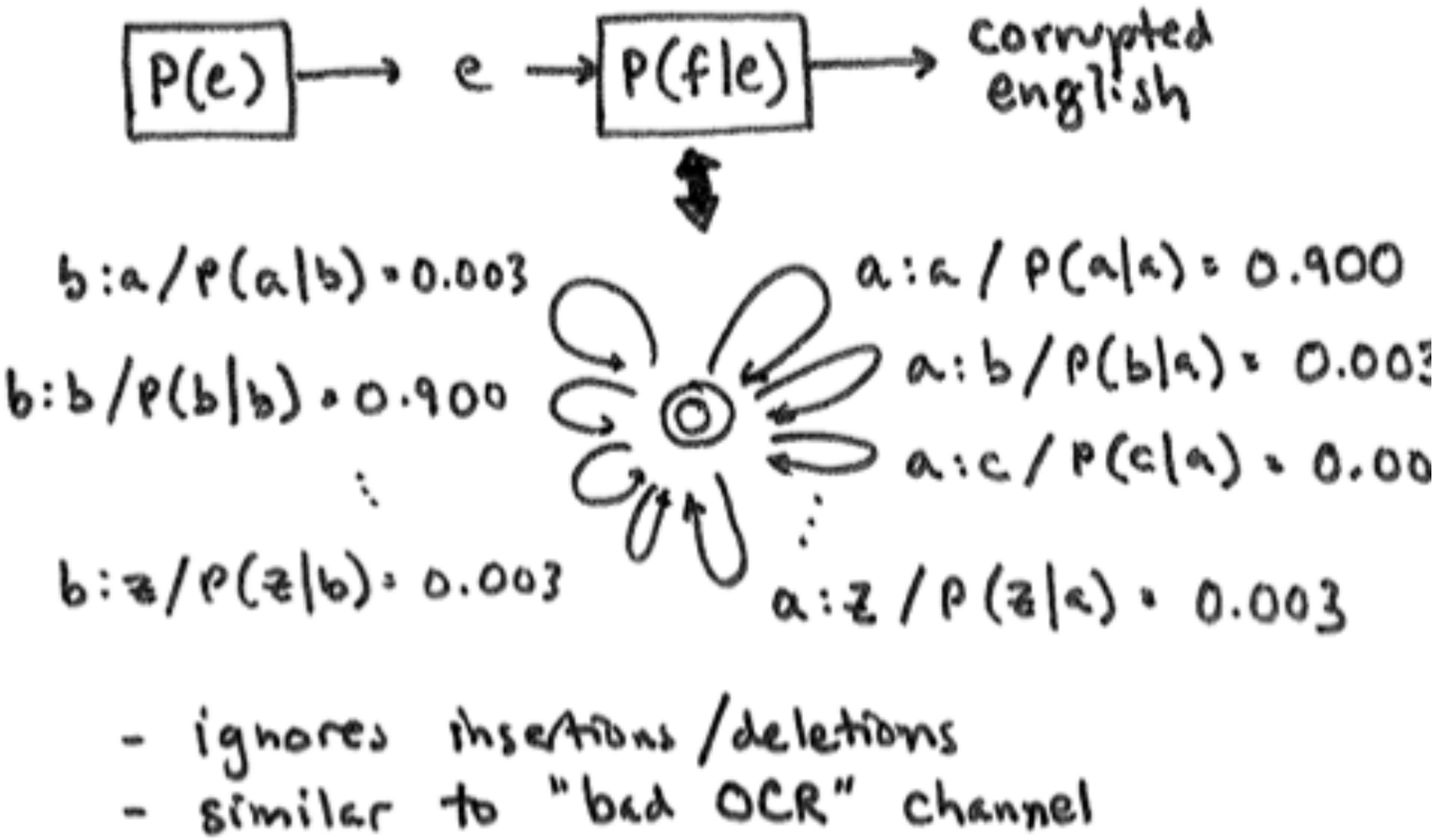
so, $\underset{y}{\operatorname{argmax}}\ P(y \mid ab) = AB$

DP for "most likely corrupted"



so, $\underset{y}{\operatorname{argmax}}\ P(y \mid ab) = AB$

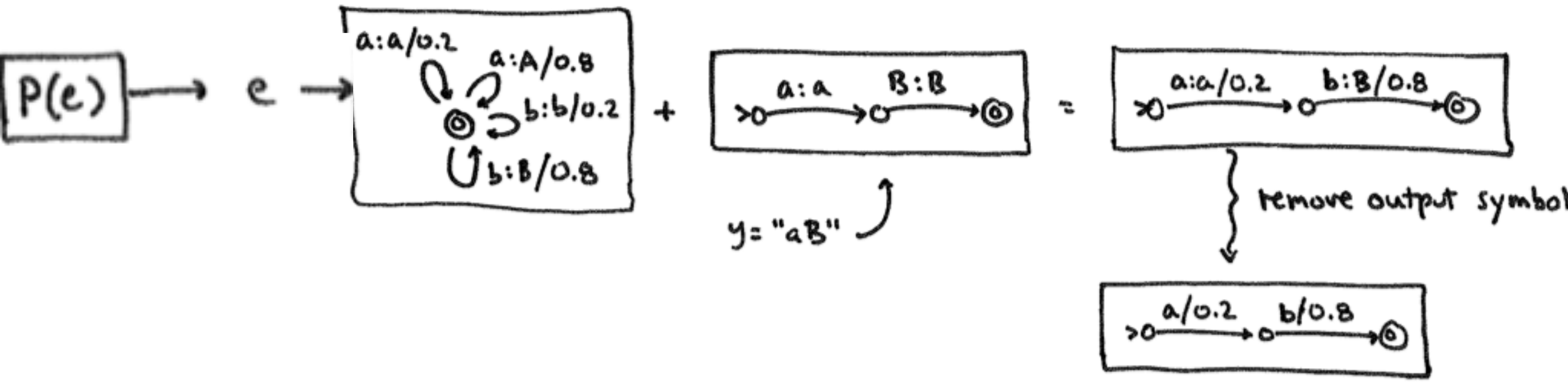
d) Most Likely "Original Input"



- using an LM p(e) as source model for *spelling correction*
  - case 1: letter-based language model $p_L(e)$
  - case 2: word-based language model $p_w(e)$
- How would dynamic programming work for cases 1/2?

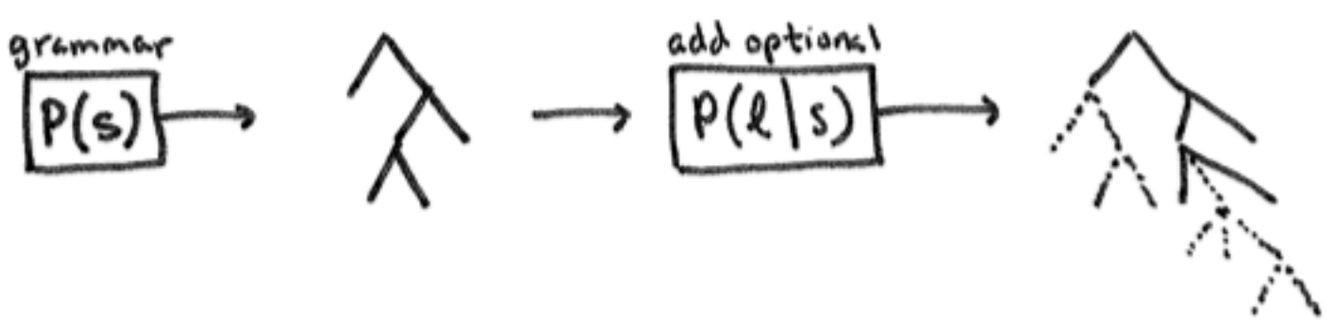
Dynamic Programming for d)

- given y, what is the most likely x with max p(x) p(y|x)



Beyond Finite-State Models

- sentence summarization



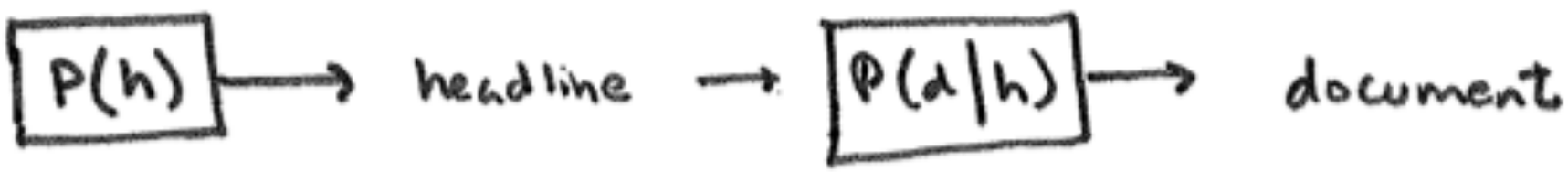
$$\underset{s}{\operatorname{argmax}} \; P(s) \cdot P(\ell \mid s)$$

original short sentence (encourage grammaticality)

particular "lengthening" (encourage addition of optional material only;
e.g., P(he didn't go | he did go) = LOW
P(the big dog | the dog) = MED
P(off of the couch | off the couch = HIGH)

Beyond Finite-State Models

- headline generation



$$\underset{h}{\operatorname{argmax}} \; \underline{P(h)} \cdot \underline{P(d|h)}$$

looks like a proper headline

if this were a headline, d would be a reasonable document to go with it (i.e., d fleshes out h).

Beyond Finite-State Models

- information retrieval

used to rank documents, not construct new ones!

query may contain words not in document.