Finite-State and the Noisy Channel

Word Segmentation

theprophetsaidtothecity

- What does this say?
  - And what other words are substrings?
- Could segment with parsing (how?), but slow.

- Given L = a "lexicon" FSA that matches all English words.
- How to apply to this problem?
- What if *Lexicon* is weighted?
- From unigrams to bigrams?
- Smooth L to include unseen words?

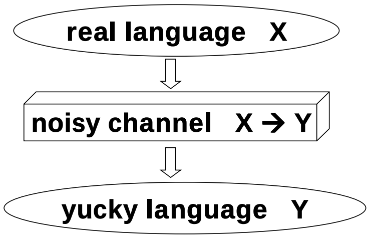
Spelling correction

- Spelling correction also needs a lexicon L
- But there is distortion ...
  - Let T be a transducer that models common typos and other spelling errors

ance → ence	(deliverance, ...)
e → ε	(deliverance, ...)
ε → e // Cons _ Cons	(athlete, ...)
rr → r	(embarrass, occurrence, ...)
ge → dge	(privilege, ...)
etc.	

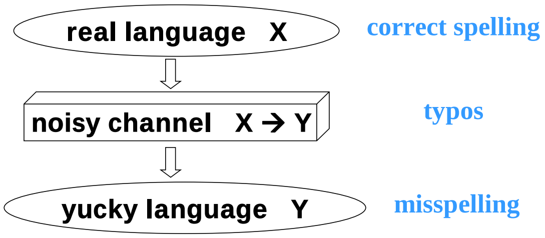
  - Now what can you do with L .o. T ?
- Should T and L have probabilities?
- Want T to include "all possible" errors ...

Noisy Channel Model



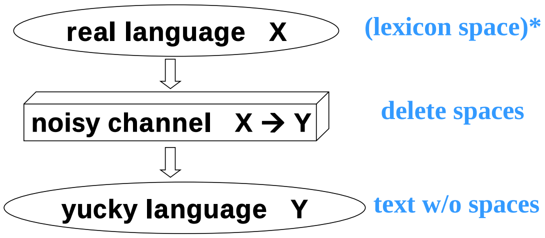
want to recover X from Y

Noisy Channel Model

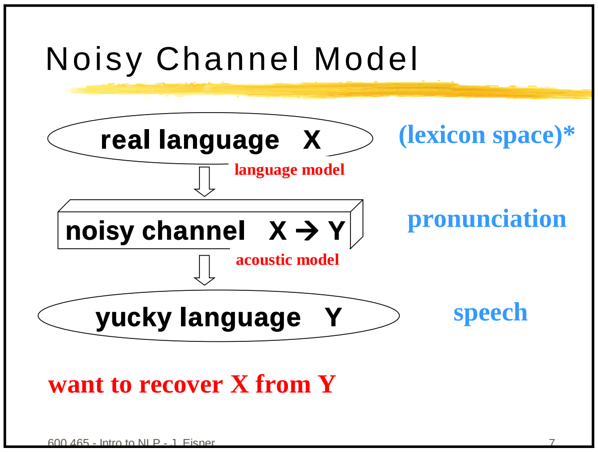


want to recover X from Y

Noisy Channel Model



want to recover X from Y



Noisy Channel Model

real language X

language model

noisy channel X → Y

acoustic model

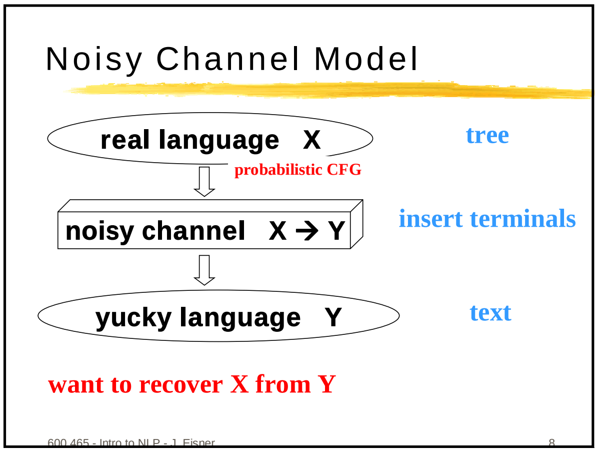
yucky language Y

(lexicon space)*

pronunciation

speech

want to recover X from Y



Noisy Channel Model

real language X

probabilistic CFG

noisy channel X → Y

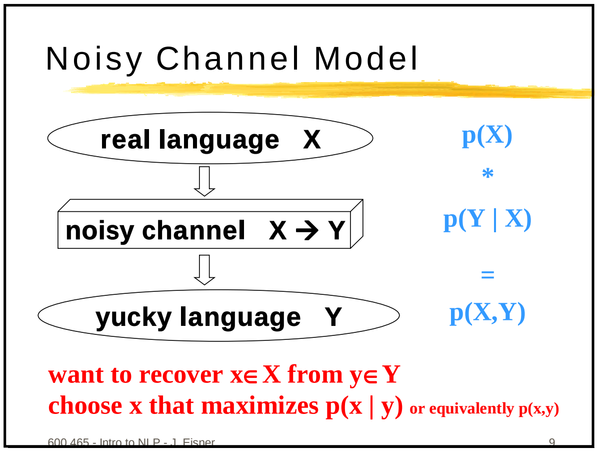
yucky language Y

tree

insert terminals

text

want to recover X from Y



Noisy Channel Model

real language X

noisy channel X → Y

yucky language Y

p(X)

*

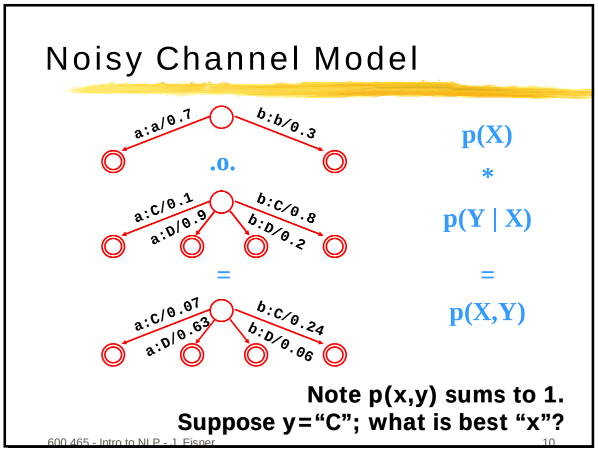
p(Y | X)

=

p(X,Y)

want to recover x∈X from y∈Y

choose x that maximizes p(x | y) or equivalently p(x,y)



Noisy Channel Model

p(X)

*

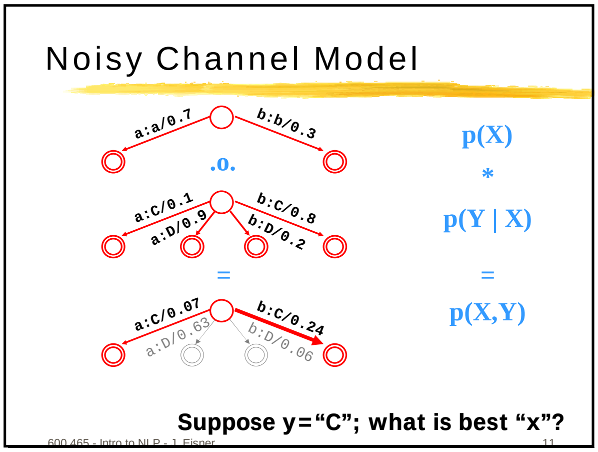
p(Y | X)

=

p(X,Y)

Note p(x,y) sums to 1.

Suppose y="C"; what is best "x"?



Noisy Channel Model

p(X)

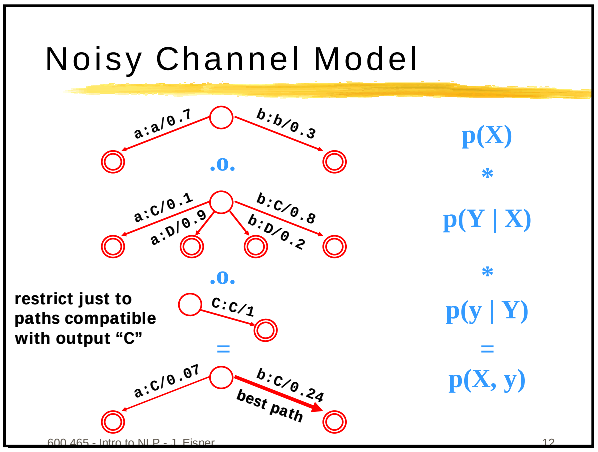
*

p(Y | X)

=

p(X,Y)

Suppose y="C"; what is best "x"?



Noisy Channel Model

restrict just to paths compatible with output "C"

p(X)

*

p(Y | X)

*

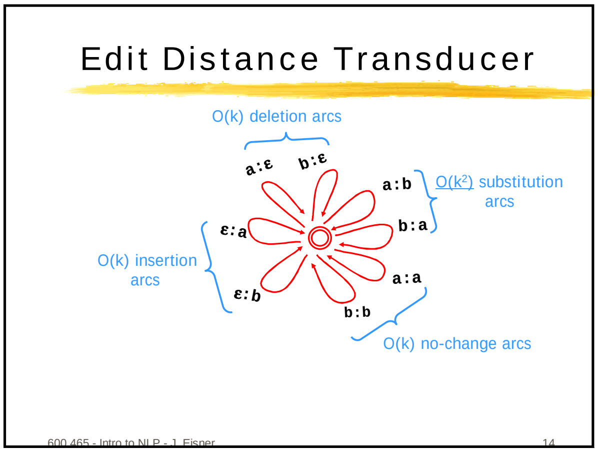
p(y | Y)

=

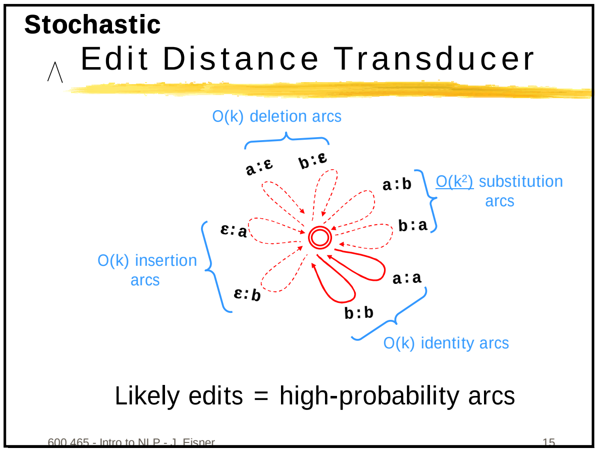
p(X, y)

Morpheme Segmentation

- Let *Lexicon* be a machine that matches all Turkish words
  - Same problem as word segmentation
  - Just at a lower level: morpheme segmentation
  - Turkish word: uygarlas,tiramadiklarimizdanmis,sinizcasina = uygar+las,+tir+ama+dik+lar+imiz+dan+mis,+siniz+casina (behaving) as if you are among those whom we could not cause to become civilized
  - Some constraints on morpheme sequence: bigram probs
  - Generative model – concatenate then fix up joints
    - stop + -ing = stopping,   fly + s = flies
    - Use a cascade of transducers to handle all the fixups
  - But this is just morphology!
  - Can use probabilities here too (but people often don't)

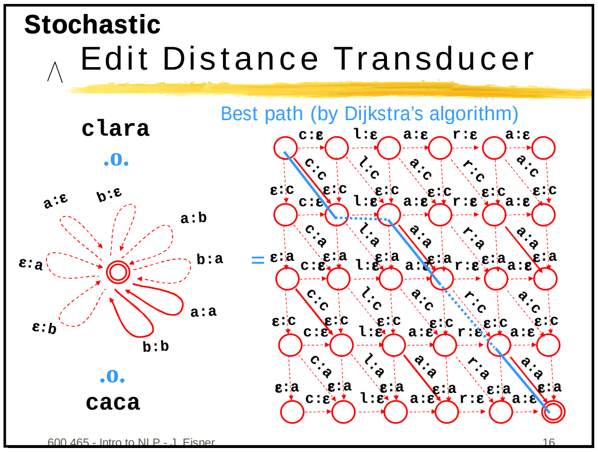


Edit Distance Transducer



Stochastic
∧ Edit Distance Transducer

Likely edits = high-probability arcs



Stochastic
∧ Edit Distance Transducer