

# Ling/CSE 472: Introduction to Computational Linguistics

---

4/6/15: Morphology & FST 2

# Overview

---

- Review: FSAs & FSTs
- XFST
- xfst demo
- Examples of FSTs for spelling change rules
- Reading questions

# Review: FSAs and FSTs

---

- FSAs define sets of strings (regular languages).
- FSTs define sets of ordered pairs of strings (regular relations).
- Formally interesting because not all languages/relations can be defined by FSAs/FSTs.
- Are all finite languages and relations regular?
- Linguistically interesting because:
  - FSAs have enough power for morphotactics.
  - FSTs have (almost) enough power for morphophonology.
  - Both are very efficient.

# FSTs: “Quiz”

---

- Why do FSTs have complex symbols labeling the arcs?
- What happens if you give an FST an input on only one “tape”?
- What happens if the input has symbols outside the FST’s alphabet?
- Do the upper and lower tape strings always have the same length?

# xfst regex syntax

---

- Why is it so different from what we see in J&M and elsewhere?
- Why are there so many operators?

# B&K minimal languages



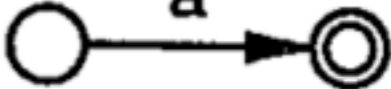
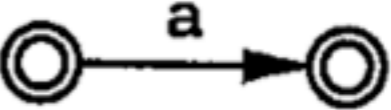
Expression	Language / Relation	Network
$\sim$ [?*]	{ }	
[]	{ "" }	
a	{ "a" }	
(a)	{ "", "a" }	

Table 2.1: Minimal Languages

# B&K: Iteration

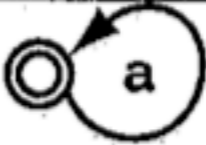
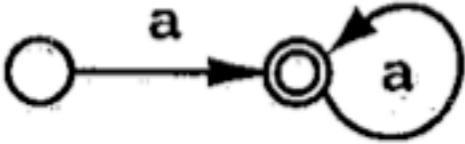
Expression	Language / Relation	Network
$a^*$	$\{\epsilon, "a", "aa", \dots\}$	
$a^+$	$\{"a", "aa", \dots\}$	

Table 2.2: Iteration

# B&K: Concatenation

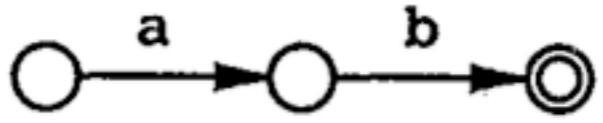
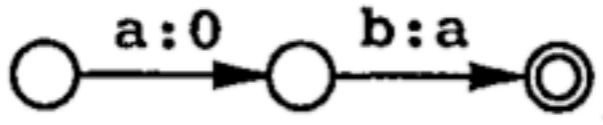
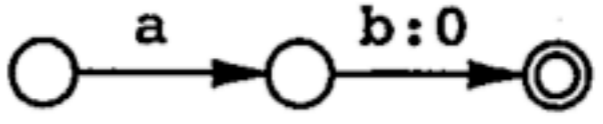
Expression	Language / Relation	Network
$a \circ b$	$\{\text{"ab"}\}$	
$a : 0 \quad b : a$	$\{\langle \text{"ab"}, \text{"a"} \rangle\}$	
$a \quad b : 0$	$\{\langle \text{"ab"}, \text{"a"} \rangle\}$	

Table 2.3: Concatenation



# B&K: Cross-product

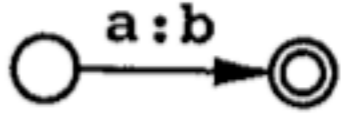
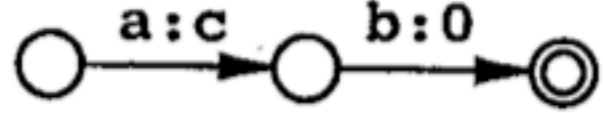
Expression	Language / Relation	Network
$a \cdot x \cdot b$	$\{ \langle "a", "b" \rangle \}$	
$[a \ b] \cdot x \cdot c$	$\{ \langle "ab", "c" \rangle \}$	

Table 2.4: Crossproduct

# B&K: Composition

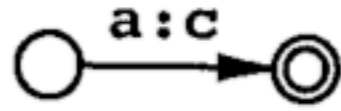
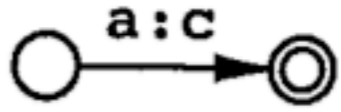
Expression	Language / Relation	Network
$a:b \ .o. \ b:c$	$\{ \langle "a", "c" \rangle \}$	
$a:b \ .o. \ b \ .o. \ b:c$	$\{ \langle "a", "c" \rangle \}$	

Table 2.5: Composition

# B&K: Closure

---

Operation	Regular Languages	Regular Relations
union	yes	yes
concatenation	yes	yes
iteration	yes	yes
reversal	yes	yes
intersection	yes	no
subtraction	yes	no
complementation	yes	no
composition	(not applicable)	yes
inversion	(not applicable)	yes

Table 2.7: Closure Properties

# B&K: Universal relations

---



Sigma: {?}

Figure 2.6: The Universal Language/Identity Relation  $?^*$

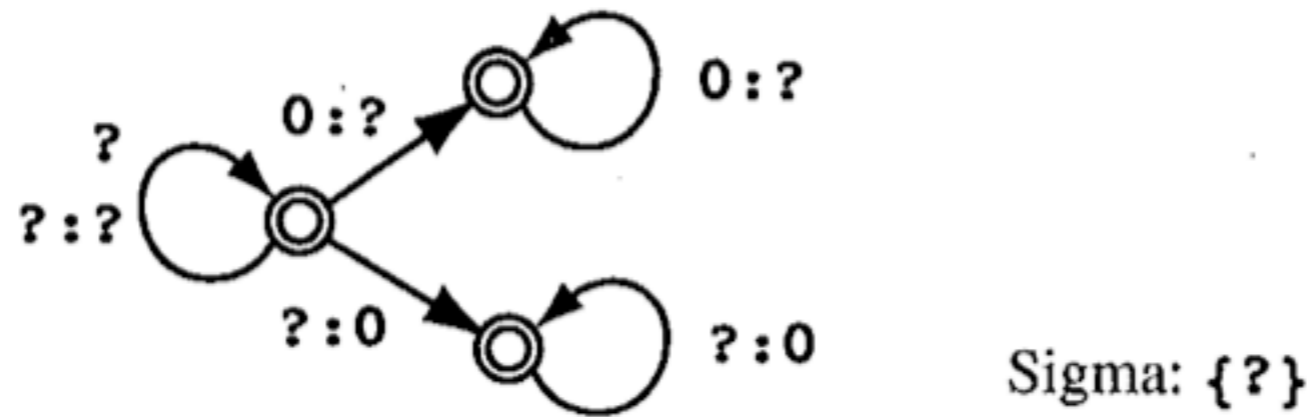


Sigma: {?}

Figure 2.7: The Universal Equal-Length Relation  $[? : ?]^*$

# B&K: Universal relations

---



- Why are these useful?

# B&K: Replacement

---

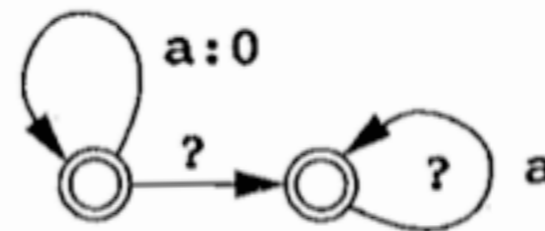
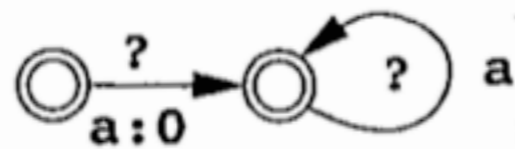


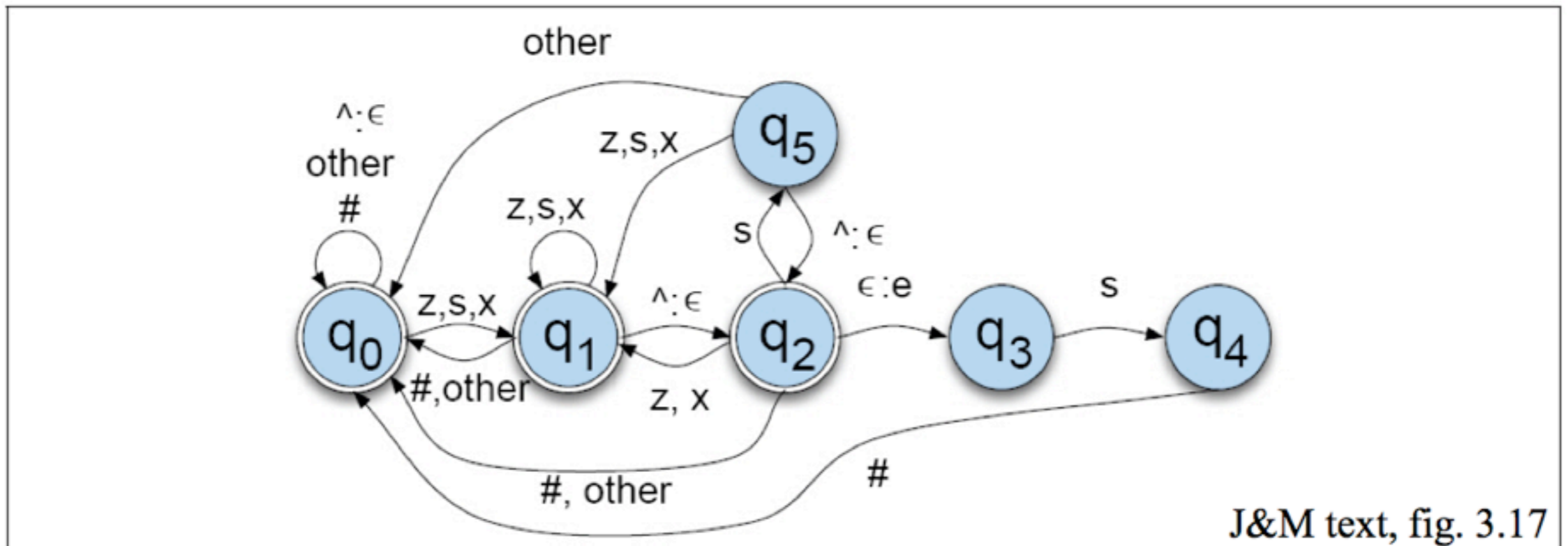
Figure 2.12:  $[a \rightarrow 0 \mid \mid \cdot \# \cdot \_]$  vs.  $[a \rightarrow 0 \mid \mid \cdot \# \cdot \_]$

# Overview

---

- Review: FSAs & FSTs
- XFST
- xfst demo
- Examples of FSTs for spelling change rules
- Reading questions

# Sample e-insertion FST



- The idea is to add  $e$  only in the proper environments while letting all other sequences pass through.



# Spelling change rule FST 1

---

```
define Rule1 [ ?* e:0 %+:0 [e|i] ?* ];
```

- Draw an FST corresponding to Rule1.
- What are the upper and lower languages of Rule1?
- What linguistic work is this rule supposed to do?
- If the upper tape has expect+ed, what goes on the lower tape?

# Spelling change rule FST 2

---

```
define Rule2 [ [?* e:0 %+:0 [e|i] ?* ] |  
               [?* e %+:0 (\[e|i]) ] |  
               [?* \e %+ ?* ] |  
               [\[%+]* ] ]
```

- What are the upper and lower languages of Rule2?
- What linguistic work is each part of this rule supposed to do?
- If the upper tape has expect+ed, what goes on the lower tape?
- If the upper tape has write+ing, what goes on the lower tape?

# Overview

---

- Review: FSAs & FSTs
- XFST
- xfst demo
- Examples of FSTs for spelling change rules
- Reading questions

# Reading questions

---

- I'm confused about how " $A^*$ ", the "union of  $A^+$  with the empty string language", means "the concatenation of  $A$  with itself zero or more times" (BK page 47). I understand the Kleene star notation but I don't think I understand how "zero or more of something" is meant to come out of the union of a relation with the empty string language. Maybe I'm confused about the terminology too?

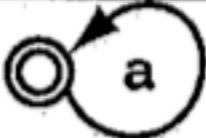
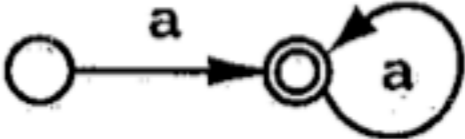
Expression	Language / Relation	Network
$a^*$	$\{\epsilon, "a", "aa", \dots\}$	
$a^+$	$\{"a", "aa", \dots\}$	

Table 2.2: Iteration

# Reading questions

---

- I don't understand how the universal relation works. I understand what it produces, at the least- a mapping a string to any other string- but the graph confuses me. How does the expression  $[?^* .x. ?^*]$  relate to the graph shown in figure 2.8?

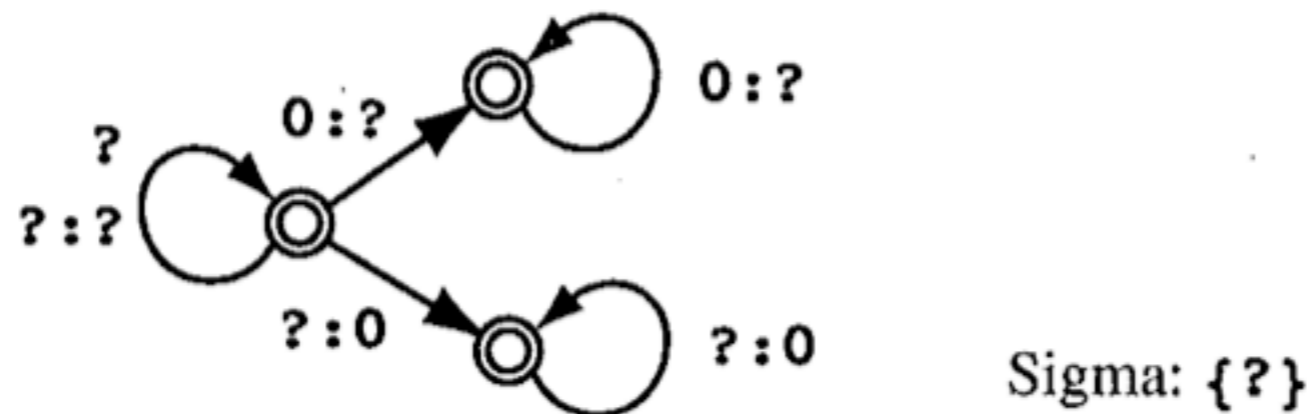


Figure 2.8: The Universal Relation  $[?* .x. ?*]$

# Reading questions

---

- I think I understand roughly all the different operations in the chapter, but I got pretty confused by the last part; it would help if we could go over an example of conditional replacement and parallel conditional replacement.
- On page 67, the authors talk about replacement ambiguity that comes from mapping upward rather than downward because of the difference between 'original strings and strings that are the result of a replacement'. Could you explain where this ambiguity comes from and why it matters if the string is the result of a previous replacement or not?

# Reading questions

---

- In the text it says, "composition is inherently an operation on relations and not on languages." Does this mean that by using these some of these operators we are taking languages and creating relations? The reading also wasn't clear to me on the projection operator. Is projection then only used to denote which language, upper or lower, the automata references? I'm confused as to its usefulness.
- Is the reason that the expression  $a \cup b$  translates to the relation  $\{ab\}$  that there is no epsilon in networks?

# Reading questions

---

- I understand the concept of concatenation, but in the reading, I am confused on why  $a : 0$   $b : a$  and  $a b : 0$  are formulated as the same language/relation:  $\{ \langle "ab" \rangle, \langle "a" \rangle \}$ .
- I can sort of understand why  $a : 0$   $b : a \rightarrow \{ \langle "ab" \rangle, \langle "a" \rangle \}$ , but why doesn't  $a b : 0 \rightarrow \{ \langle "a" \rangle, \langle "b" \rangle \}$ ?
- I understood figure 2.8 differently based on the definitions. My understanding from figure 2.8 was any character + cross product of a character to another character + any other character, where the first two are from equal length. In this example, I don't know why there should be  $0 : ?$  and  $? : 0$  to connect strings of different length to each other.
- The replace relations are not necessarily one-to-one even if the replacement language contains just one string, but possible replacements may overlap, so why would not restrict the replace relations to one-to-one only to avoid this problem?



# Reading questions

---

- The reading says that while  $[a \Rightarrow \cdot \# \cdot \sim [b ?^*] \_ ]$  expresses that 'a' cannot be preceded by a string that starts with 'b',  $[a \Rightarrow \sim [b ?^*] \_ ]$  allows 'a' to appear anywhere. I don't really understand why this is true. It seems to me like the second regular expression would instead express that 'a' cannot be preceded by a string contains 'b' in any location.

# Reading questions

---

- Why are regular relations not closed in general under intersection, complementation, and subtraction. Can we try to come up with examples that illustrate this? The book tried to give an example of this using intersection on page 54, but I did not understand it.

# Reading questions

---

- The any symbol "?" as the unknown symbol in network diagrams: I understand the use of "?" in regular expressions, but I do not fully understand why they overloaded the meaning of the symbol in network diagrams. Could you actually go over a network diagram with the "?" in it and explain what actually happens when you get to the "?".

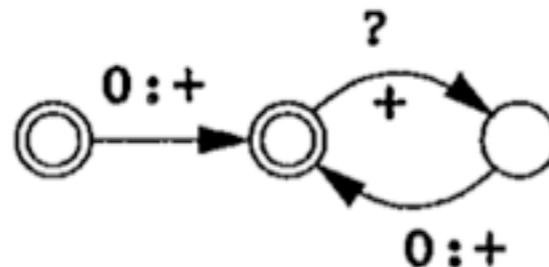


Figure 2.11: The Relation  $[ \cdot \ 0 \ \cdot ] \rightarrow \%+$

# Reading questions

---

- The Xerox compiler syntax for regular expressions from B&K is very different from the regular expression we have worked with. Will we be expected to study and use this syntax for future assignments?
- Also, are there transducer minimization algorithms in the same way there are ones for FSM minimization?
- How does a phrase-structure grammar dependent on center-embedding generate context-free language? What exactly are context free languages and why can't they be encoded by finite-state networks?

# Reading questions

---

- The reading discusses cross products applied to two regular expressions. Is the cross product of two sets useful in NLP, or did they support the operation for more formal reasons (i.e. to support regular languages as they are formally defined, rather than in regard to their use in processing language)?
- How does the Xerox regular expression compiler work? The text says that it creates networks that are epsilon-free and minimal -- where can I find the algorithms that are being used to minimize them? More generally, how do you actually prove that a FSM is minimal?

# Reading questions

---

- Also, if it's possible, could you please post the solution for "the better cola machine" in page 466? I really want to know how to do it.