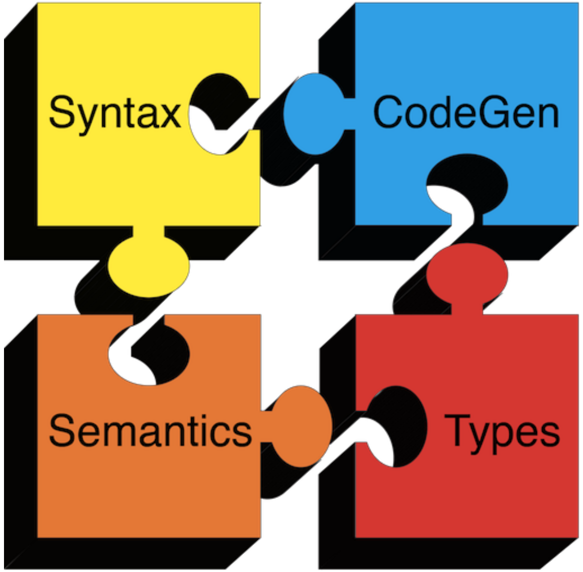

Compilers

Lexical Specification

Alex Aiken

Lexical Specification

- At least one: $A^+ \equiv AA^*$
- Union: $A \mid B \equiv A + B$
- Option: $A? \equiv A + \varepsilon$
- Range: ‘a’ +’ b’ +…+’ z’ ≡ [a-z]
- Excluded range:

  complement of [a-z] ≡ [^a-z]

Lexical Specification

- Last lecture: a specification for the predicate

$$s \in L(R)$$

- Not enough!

Lexical Specification

1. Write a rexp for the lexemes of each token class
    - Number = digit$^+$
    - Keyword = ‘if’ + ‘else’ + …
    - Identifier = letter (letter + digit)*
    - OpenPar = ‘(‘
    - …

Lexical Specification

2. Construct R, matching all lexemes for all tokens

R = Keyword + Identifier + Number + …

$= R_1 + R_2 + \ldots$

Lexical Specification

3. Let input be $x_1 \ldots x_n$

   For $1 \leq i \leq n$ check

   $$x_1 \ldots x_i \in L(R)$$

4. If success, then we know that

   $x_1 \ldots x_i \in L(R_j)$ for some j

5. Remove $x_1 \ldots x_i$ from input and go to (3)

Lexical Specification

- How much input is used?

Lexical Specification

- Which token is used?

Lexical Specification

- What if no rule matches?

Lexical Specification

- Regular expressions are a concise notation for string patterns

- Use in lexical analysis requires small extensions
  - To resolve ambiguities
  - To handle errors

- Good algorithms known
  - Require only single pass over the input
  - Few operations per character (table lookup)