Formal Languages
- Alphabet (Σ)
- String (Σ*)
- Length (|x|)
- Empty String (ε)
- Empty Language (∅)

Language/String Operations
- "Regular" Operations:
  • Union (∪)
  • Concatenation (•)
  • (Kleene) Star (*)
- Other:
  • Intersection
  • Complement
  • Reversal
  • ...

English, mathematical
• DFAs
- States
- Start states
- Accept states
- Transitions (δ function)
- M accepts w ∈ Σ*
- M recognizes L ⊆ Σ*
• NFAs
- Transitions (δ relation)
- Missing out-edges
- ε-moves
- Multiple out-edges
- N accepts w ∈ Σ*
- N recognizes L ⊆ Σ*
• Regular Expressions
- ∅, a ∈ Σ, ∪, •, *, ( )
• GNFA

The class of regular languages is closed under:
- Regular ops: union, concatenation, star
- Also: intersection, complementation, (δ reversal, prefix, no-prefix, ...) NOT closed under ⊆, ⊇
- Also: Cross-product construction (union, ...)

Key idea: once M is in some state q, it doesn’t remember how it got there. E.g. "hybrids": if xy ∈ L(M) and x, y both go to q, then xy ∈ L(M) too. E.g. "loops": if xyz ∈ L(M) and x, y both go to q, then xy*z ∈ L(M) for all i ≥ 0.

Cor: Pumping Lemma
- Important examples:
  - L1 = { a^n b^n | n > 0 } L2 = { w | #a(w) = #b(w) } L3 = { w w | w ∈ Σ* } L4 = { w^i w^j | i, j ≥ 0 } L5 = { balanced parens }
- Also: closure under ∩, complementation sometimes useful:
  - L1 = L2 ∩ a*b*
- PS: don’t say "irregular"
Applications

• "globbing"
  - lpr *.txt
• pattern-match searching:
  - grep "Ruzzo.*terrific" *.txt

• Compilers:
  - Id ::= letter ( letter|digit )*
  - Int ::= digit digit*
  - Float ::= d d* . d* ( e | E d d* )