Formal Notations of FSM

- FSM (DFSM) is a quintuple $(K, \Sigma, \delta, s, A)$ where:

  - K is a finite set of states
  - Σ is the input alphabet
  - $s \in K$ is that start state
  - $A \subset K$ is thet set of accepting states
  - δ is the transition function that maps from

$$K \times \Sigma \text{ to } \Sigma$$

- A **configuration** of DFSM is an element of $K \times \Sigma^*$. Think of it as a snapshot of M at any given point. A configuration gives us two sets of information:

  - The current state.
  - The input that is still left to read.
    Example - $(q_0, \texttt{abbabab})$, $(q_1, \texttt{bbabab})$, $(q_2, \texttt{babab})$ ...

- The **initial configuration** of DFSM is (s_M, w) where s_M is the start state of M and w is the string to be read.

- The **transition function** δ defines the operation of a DFSM M one step at a time. Relation **yeilds-in-one-step** is written $|-_M$. *Yeilds-in-one-step* relates configuration$_1$ to configuration$_2$ iff configuration$_1$ leads to configuration$_2$ in one step.

$$(q_1, cw)| -_M (q_2, w) \text{ iff } ((q1, c), q_2) \in \delta$$

- to be continued...

Designing Deterministic Finite State Machines

- We need to think of what properties of the part of w that has been read so far has an effect on M. Those are the properties that M has to record
- Strings must "cluster" which means that multiple different strings drive M to the same state. They have the same property which makes them go to that state. **The smallest DFSM of any language is the one that has exactly one state for every group of initial substrings that share that common property**
- Complement of a language - "A language that doesn't have a specific substring, we can first design the FSM such that it accepts strings with that substring and then just swap the accepting and rejecting states.

Nondeterministic FSMs

-