*-Languages: Given finite set Σ

- Σ*: set of all finite sequences over Σ
- A *-language is a subset of Σ*

Notation:
lower case variable takes value in Σ,
upper case in Σ*

Finite Automata (FA) – 5-tuple (Σ, S, s₀, T, A)

- Σ – finite set called the alphabet
- S – finite set of states
- s₀ ∈ S – initial state
- T ⊂ S × Σ × S – transition relation
- A ⊂ S – accepting states

Deterministic FA – (∀s)(∀x) [ | {t : (s, x, t) ∈ T} | ≤ 1 ];
otherwise non-deterministic
DEFINITIONS & BACKGROUND-II

- String $X \in \Sigma^*$ accepted by FA if there exists a sequence of states (a run) $\sigma_0\sigma_1...\sigma_n$ such that
  1. $n = |X|$,  
  2. $\sigma_0 = s_0, \sigma_n \in A$, and  
  3. $(\forall i)[(\sigma_i, X_i, \sigma_{i+1}) \in T]$  

- Language of FA = set of strings accepted by it; a $*$-language is regular if it can be constructed by 
  $\{x \in \Sigma,,.+,*,(,())\}$, e.g. $(a(b+c))*$.

Facts:

1. Language is regular $\iff$ it is accepted by a DFA (Power set construction)

2. Class of regular languages is closed under
   - Union, Intersection (product construction)
   - Projection (eraser construction)
   - Complementation (determinize with power set construction $\Rightarrow$ trivial complementation).
INTRODUCTION & BACKGROUND-III

Logic $S_1S$ (Second order theory of one successor) - Natural logic for sets of sequences.

- formalism for describing properties of sequences;
- powerful mechanism for analysis and manipulation of sequential systems (makes available full power of logic)

**Alphabet:** \{0, $S$, $\equiv$, $<$, $\in$, $\land$, $\neg$, $\exists$, $x_1$, $x_2$, ..., $X_1$, $X_2$, ...\}

**Syntax:**

- **Terms:** $0 \mid x_i \mid St_1$ where $t_1$ is a term.
  Examples of terms – $0$, $SS0$, $SSSSx_3$.

- ** Atomic formulae:** $t_1 = t_2 \mid t_1 < t_2 \mid t_1 \in X_k$ where $t_1, t_2$ are terms, and $X_k$ is a finite set of terms.
  Examples of atomic formulas – $0 < S0$, $x_3 = SSSx_5$, $Sx_7 \in X_2$.

- **$S_1S$ formulas:** $\phi \land \psi \mid (\neg \phi) \mid (\exists x_i) \phi \mid (\exists X_i) \phi$ where $\phi, \psi$ are $S_1S$ formulae or atomic formula.
  Examples of $S_1S$ formulas – $(0 < S0) \land (Sx_7 \in X_2)$, $(\exists X)(\exists x)[(x \in X) \land (Sx \in X)]$. 
**INTRODUCTION & BACKGROUND**

**S1S Semantics:** Interpret over $\omega = \{0, 1, 2, \ldots\}$.

**Example 1:** (Existence of least elements in non empty subsets of $\omega$)

$$
\psi = (\forall X)[(\exists x)(x \in X) \rightarrow (\exists y)((y \in X) \land \neg((\exists z)(z \in X \land (z < y))))]
$$

**Example 2:** (Defining subsets of $\omega$ which contain 5 whenever they contain 3.)

$$
\phi_0(X) = (SSS0 \in X) \rightarrow (SSSSSS0 \in X)
$$

**Example 3:** (Defining the subset of even integers)

$$
\phi_1(X) = (0 \in X) \land \neg(S0 \in X) \land (\forall x)(x \in X \leftrightarrow SSx \in X)
$$

**Example 4:** (Defining the relation “every even number in $X$ is in $Y$”)

$$
\phi_2(X, Y) = (\forall x)[(\forall Z)(\phi_1(Z) \land x \in Z) \rightarrow (x \in X \rightarrow x \in Y)]
$$

Given formula $\theta(X_1)$ class of subsets of $\omega$ defined by $\theta(X_1)$ is $\{\beta \subseteq \omega \mid \theta(\beta) \text{ is true}\}$; Generally, formulae $\phi(X_1, X_2, \ldots, X_n)$ define subsets of $(\{0, 1\}^n)^\omega$
**Observation:** Finite subsets of $\omega^{1-1}$ sequences over \{0,1\}. For example \{0,2,3\} is represented by 1011. We have to be careful about 0 sequences, e.g. to distinguish 00 from 000 etc. (Can do this by picking a length and extending each set to that length by putting 0’s at the end) Thus, $S_1S$ formulae define *-languages.

**Theorem:**[Büchi 1961] A *-language is definable in $S_1S$ if and only if it is regular

**Proof:**(sketch)
Reverse direction: Given an automaton, there is a straightforward coding up of it as a formula. This would be a big OR of the coding of each transition in the automaton.
Proof Continued

*Forward direction:* Given a formula, we want to produce an equivalent automaton. We use induction on length of formula.

- **Base Case:** Trivial automaton for atomic formulae. For example, $x = y$ is given by the two-state automaton with inputs $x, y$. It goes from the good state to the bad state if at any point, $x_i \neq y_i$.

- **∃:** Automaton projection (erase some labels on the edges)

- **∧:** Automaton Intersection (product automaton)

- **¬:** Automaton Complementation

**Advantages of S1S:**

Relationship between automata and S1S ⇒ formally, succinctly express behaviors as formulae in S1S. In particular,

- **Automatic** procedure for deriving automaton from formula

- **Rigorous and constructive** proofs for synthesis procedures