CS 124 Lectures 1-2

José Meseguer

1. Concurrent Systems and their Computational Logics

A concurrent system will be defined by some specification $\mathcal{S}$. We will then define its concurrent computations as follows:

1. We will define the computational logic $\mathcal{L}(\mathcal{S})$ of $\mathcal{S}$ so that:

   (i) a proof in $\mathcal{L}(\mathcal{S})$ corresponds to
   (ii) a specification of a concurrent computation in $\mathcal{S}$

2. We will define a concurrent computation in $\mathcal{S}$ as an equivalence class of proofs by suitable equivalence relation $\equiv$ between proofs.

To see how this can be done, we will begin with a very simple class of computational systems, namely, automata. Automata are not concurrent: they are totally sequential. This is because sequential computation is a degenerate special case of concurrent computation. However, beginning with the sequential case will help us understand

later on the essential difference between sequential computation and truly concurrent systems.

1.1 Automata

Definition. An automaton $\mathcal{A}$ is a triple $\mathcal{A} = (Q, L, \rightarrow_{\mathcal{A}})$ where:

1. Q is a set, called its set of states
2. L is a set of symbols, called it alphabet
3. $\rightarrow_{\mathcal{A}} \subseteq Q \times L \times Q$ is a ternary relation called its transition relation

Note: In what follows we will always assume that Q and L (and therefore $\rightarrow_{\mathcal{A}}$) are finite sets, so that $\mathcal{A}$ is a finite automaton.

Remark. The following two observations are important:

1. An automaton $\mathcal{A} = (Q, L, \rightarrow_{\mathcal{A}})$ is also called a labeled transition system. In that terminology, L is called its set of labels or, sometimes its set of actions.
2. An automaton $\mathcal{A} = (Q, L, \rightarrow_{\mathcal{A}})$ is exactly the same thing as a labeled directed graph. In that terminology, Q is called its set of nodes, L its set of labels and $\rightarrow_{\mathcal{A}}$ its set of labeled edges.

Therefore, we have the following equivalence between these concepts:

Automaton $\Leftrightarrow$ Labeled Transition System $\Leftrightarrow$ Labeled Directed Graph

they are all the same thing, with somewhat different terminology.

Notation Given $(q, l, q') \in \to_A$, so that $q, q' \in Q$, $l \in L$,

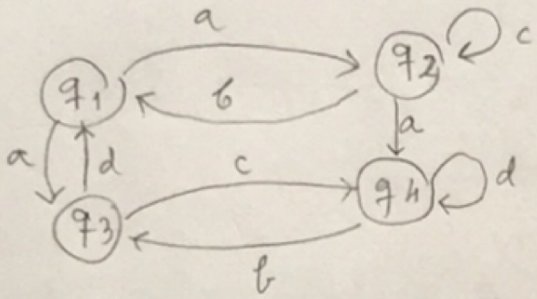
we use the notation $q \xrightarrow{l} q'$, or $l : q \to q'$, and

call $q \xrightarrow{l} q'$ a:

1. transition from state q to state q' with letter l in the automaton A
2. a transition from state q to state q' with label l in the labeled transition system A
3. A directed edge from node q to node q' labeled by l in the labeled directed graph A

Again, the are the same thing with somewhat different terminology.

Example Since an automaton is a directed graph, it is very easy to definite in graphical form. For example:



is an automaton with set of states $Q = \{q_1, q_2, q_3, q_4\}$, alphabet $L = \{a, b, c, d\}$ and transition relation

$$\xrightarrow{}_{\mathcal{A}} = \{(q_1,a,q_2), (q_1,a,q_3), (q_2,b,q_1), (q_2,c,q_2), (q_2,a,q_4), (q_4,d,q_4), (q_4,b,q_3), (q_3,d,q_1), (q_3,c,q_4)\}$$

1.2 The Logic of an Automaton

Given an automaton $\mathcal{A}$, its computational logic $\mathcal{L}(\mathcal{A})$ is defined by a set of inference rules that prove the set of all its possible computation specifications, where a computation specification has the form:

$$q \xrightarrow{\alpha} q'$$

where α is called the proof term of the computation specification. What $q \xrightarrow{\alpha} q'$ specifies is a possible