A Brief Introduction to Labelled Transition Systems

Adam Banham

January 2025

1 What are (Labelled) Transition Systems?

Transition systems are graphs that allow us to model systems. Their simplicity is useful for many techniques that are concerned with verifying if a system conforms to a condition. Often we don't model systems using transition systems, but instead represent the behaviour of a system modelled in higher order modelling formalisms like Petri nets, BPMN, or UML as a transition system. Allowing us to verify the behaviour of the higher order model using a simpler transition system.

One reason to avoid modelling processes using transition systems, rather than translating from a higher order model, is that these systems have problems expressing concurrency succinctly [1]. Where concurrency requires that the transition system models all interleaving states resulting in the well-known "state explosion" problem [4]. Nonetheless, transition systems can be used to quickly introduce process modelling if the example is kept small enough and without concurrency.

I took inspiration from *Dr. Ir. Jan Martijn van der Werf* lecturers on transitions systems, and would recommend that others do the same. I have collected the recorded lecturers I have listened to in a playlist[1], whilst I was preparing this short introduction to transition systems.

2 How are they defined?

A labelled transition system, or LTS, is a directed graph with vertices and edges. However, LTS describes these components as states and transitions, respectively. As I typically focus on the behaviour of processes, we will follow a definition from Wil M.P. van der Aalst [1] for transition systems, but note many other applications exist with differing notations.

Definition 1 (Labelled Transition System)**.** *A transition system is a triplet* $LTS = (Q, \Sigma, \rightarrow)$ *where* Q *is a set of states,* Σ *is a set of possible actions, and* $\rightarrow \subseteq (Q \times \Sigma \times Q)$ *is set of directed labelled edges between states, or transitions.*

Given a LTS, the set of initial states Q^{srt} and the set of final states Q^{end} are defined implicitly. The initial states are sometimes referred to as "start" states, and the final states are sometimes referred to as "accept" states [1]. These sets can be readily derived from a LTS in the following manner:

$$Q^{srt} = \left\{ q \in Q \mid \forall_{(q',\alpha,q'') \in \rightarrow}\ q'' \neq q \right\} \qquad \text{(states without any incoming edges)} \qquad (1)$$

$$Q^{end} = \left\{ q \in Q \mid \forall_{(q',\alpha,q'') \in \rightarrow}\ q' \neq q \right\} \qquad \text{(states without any outgoing edges)} \qquad (2)$$

[1]www.youtube.com/playlist?list=PLvu231zKB7hjtFwSMUykCxQEUfHCSadPQ

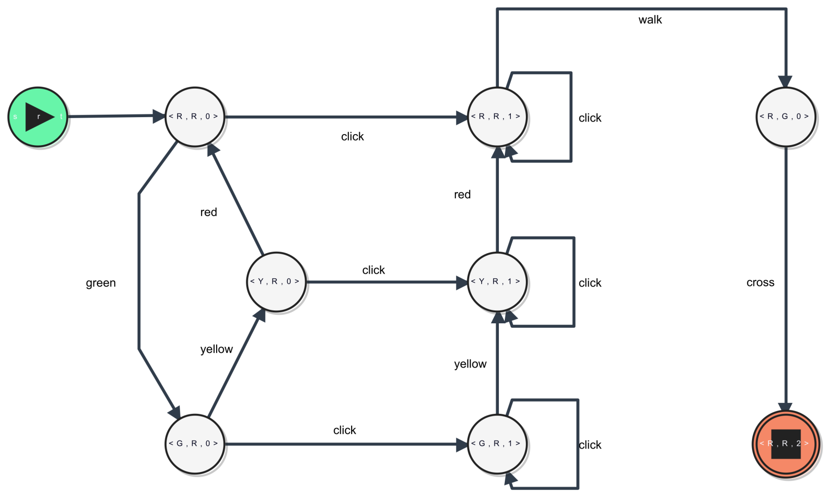

Figure 1: An example of a traffic light system to cross the road, visualised by my editor.

In principle, Q can be infinite, but for most practical applications the state space is typically finite [1]. Also a special character τ is used to denote when a transition has an unspecified label, written as (q, τ, q).

If you are interested in reading into transition systems, check out the following book chapters for some of the different formalisations:

- Chapter two in *Process Algebras for Petri Nets* by Roberto Gorrieri (2017) [2], or
- Chapter five in *Introduction to Automata Theory, Languages and Computation* by Hopcroft, Motwani and Ullman (2008) [3].

3 What is an example of such system?

4 What are their semantics?

Given a transition system LTS one can reason about its behaviour, where the system starts non-deterministically in one of the initial states [1].

$$
\begin{align}
Q =& \ \{\langle R,R,0\rangle, \langle G,R,0\rangle, \langle Y,R,0\rangle, \langle G,R,1\rangle, \langle Y,R,1\rangle, \langle R,R,1\rangle, \langle R,G,0\rangle, \langle R,R,2\rangle, \mathrm{srt}\}, \tag{3}\\
Q^{srt} =& \ \{srt\}, \tag{4}\\
Q^{end} =& \ \{\langle R,R,2\rangle\}, \tag{5}\\
\Sigma =& \ \{\mathit{green}, \mathit{yellow}, \mathit{red}, \mathit{click}, \mathit{walk}, \mathit{cross}, \tau\}, \tag{6}\\
\rightarrow =& \ \{(\langle R,R,0\rangle, \mathit{green}, \langle G,R,0\rangle), (\langle G,R,0\rangle, \mathit{yellow}, \langle Y,R,0\rangle), (\langle Y,R,0\rangle, \mathit{red}, \langle R,R,0\rangle), \notag\\
& (\langle G,R,0\rangle, \mathit{click}, \langle G,R,1\rangle), (\langle G,R,1\rangle, \mathit{yellow}, \langle Y,R,1\rangle), (\langle Y,R,1\rangle, \mathit{red}, \langle R,R,1\rangle), \notag\\
& (\langle R,R,0\rangle, \mathit{click}, \langle R,R,1\rangle), (\langle Y,R,0\rangle, \mathit{click}, \langle Y,R,1\rangle), (\langle R,R,1\rangle, \mathit{walk}, \langle R,G,0\rangle), \notag\\
& (\langle R,G,0\rangle, \mathit{cross}, \langle R,R,2\rangle), (\langle G,R,1\rangle, \mathit{click}, \langle G,R,1\rangle), (\langle Y,R,1\rangle, \mathit{click}, \langle Y,R,1\rangle), \notag\\
& (\langle R,R,1\rangle, \mathit{click}, \langle R,R,1\rangle), (\mathrm{srt}, \tau, \langle R,R,0\rangle)\}. \tag{7}
\end{align}
$$

Figure 2: The formal components of Figure 1.

5 How do I store the system on the filesystem?

6 What can you verify from the modelling the system in this way?

References

[1] Wil M. P. van der Aalst. *Process Mining - Data Science in Action, Second Edition.* Springer, 2016.

[2] Roberto Gorrieri. *Process Algebras for Petri Nets - The Alphabetization of Distributed Systems.* Monographs in Theoretical Computer Science. An EATCS Series. Springer, 2017.

[3] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to automata theory, languages, and computation, 3rd Edition.* Pearson international edition. Addison-Wesley, 2007.

[4] Antti Valmari. "The State Explosion Problem". In: *Petri Nets.* Vol. 1491. Lecture Notes in Computer Science. Springer, 1996, pp. 429–528.