From Singleton to Linear Logic
Frank Pfenning
OPLSS 2019

June 24, 2019

Date Performed:	June 18th 2019
Students:	J.W.N. Paulus
	R. Gurdeep Singh
	H. C. A. Tavante

3 Lecture 3: From Singleton to linear logic

3.1 A counter

We shall implement a counter with the following:

$$bin \;=\; \oplus\{b_0 : bin, b_1 : bin, e : \alpha\}$$

$$bin \;\vdash\; succ\; bin$$

$$\begin{aligned} succ = \textsc{case}\mathrm{L}(b_0 &\Rightarrow R.b_1; \leftrightarrow && (1)\\ |b_1 &\Rightarrow R.b_0; succ && (2)\\ |e &\Rightarrow R.b_1; R.e; \leftrightarrow) && (3) \end{aligned}$$

$$ctr = \&\{inc : ctr, reset : ctr, val : bin\} \qquad (4)$$

$$bin \;\vdash\; counter : ctr$$

$$\begin{aligned} counter = \textsc{case}\mathrm{R}(inc &\Rightarrow succ \overset{bin}{\mid} counter\\ |reset &\Rightarrow delete \overset{bin}{\mid} counter\\ |val &\Rightarrow \leftrightarrow) \end{aligned}$$

For delete:

$$bin \vdash delete : bin$$

$$\begin{aligned} delete = caseL(&b_0 \Rightarrow delete \\ &|b_1 \Rightarrow delete \\ &|e \Rightarrow R.e; \leftrightarrow) \end{aligned}$$

3.2 Turing machine

To show that the system at hand is Turing complete, we emulate a Turing machine (TM) (Wikipeda).
A Turing machine is a theoretical device with and infinite tape as memory. A head read and writes to this tape and can travel LEFT or RIGHT. The movement and actions of the head are encoded in a so called transition function δ.
The transition function δ is has following form:

$$\delta(\underbrace{q}_{\text{current state}}, \underbrace{a}_{\text{current tape data}}) = (\underbrace{b}_{\text{new tape contents}}, \underbrace{\begin{matrix}\text{LEFT}\\ \text{RIGHT}\end{matrix}}_{\text{movement}}, \underbrace{p}_{\text{next state}})$$

We will represent each state of our TM as a process that has tape data left and right of it. That is, the state is encoded as sitting in between to cells of the tape. For clarity, we give each state a name with a triangle ($\blacktriangleleft$ or $\blacktriangleright$) that indicates at which location on the tape the head in "looking".
The states of our Turing machine will be consuming data on the tape. Therefore, the tape must be emitting data, or in type terms, the tape should have an external choice type ($\oplus$). The tape left and right of the head should emit data in opposite directions. The tape data should be coming towards the processes. In our pictures we indicate the direction that tape content is being emitted in with an arrow. Let $\{a, b, \dots\}$ be the alphabet of the tape. The type of the tape left of the head is given by *tape*, the content on the right has type *etap*:

$$tape = \oplus\{a : tape, b : tape, \dots\} \qquad epat = \oplus\{a : tape, b : tape, \dots\}$$

For a state q of the Turing machine we wish to implement, we have two states in our logical system. One where the state "looks" left ($\blacktriangleleft\ q$) and one where the state "looks" right ($q\ \blacktriangleright$).

$$tape \vdash \blacktriangleleft\ q : epat$$
$$tape \vdash q\ \blacktriangleright : epat$$

To implement the TMs transition function, we need to define the above two processes for each state q in the TM. These processes should take into account the value of $\delta(q, \cdot)$.

a. For left moving rules ($\delta(q, a) = (b, \text{LEFT}, p)$) we have that:

- $\boxed{\ldots \mid \overrightarrow{a} \mid \blacktriangleleft\, q \mid \ldots}$ must become $\boxed{\ldots \mid \blacktriangleleft\, p \mid \overleftarrow{b} \mid \ldots}$

$$tape \vdash \blacktriangleleft\, q : epat$$
$$\blacktriangleleft\, q = \text{caseL}(a \Rightarrow \blacktriangleleft\, p | (Lb; \leftrightarrow) \quad , b \Rightarrow \ldots)$$

- $\boxed{\ldots \mid q \blacktriangleright \mid \overleftarrow{a} \mid \ldots}$ must become $\boxed{\ldots \mid \blacktriangleleft\, p \mid \overleftarrow{b} \mid \ldots}$

$$tape \vdash q \blacktriangleright : epat$$
$$q \blacktriangleright = \text{caseR}(a \Rightarrow \blacktriangleleft\, p | (Lb; \leftrightarrow) \quad , b \Rightarrow \ldots)$$

b. For right moving rules ($\delta(q, a) = (b, \text{RIGHT}, p)$) we have that:

- $\boxed{\ldots \mid \overrightarrow{a} \mid \blacktriangleleft\, q \mid \ldots}$ must become $\boxed{\ldots \mid \overrightarrow{b} \mid q \blacktriangleright \mid \ldots}$

$$tape \vdash \blacktriangleleft\, q : epat$$
$$\blacktriangleleft\, q = \text{caseL}(a \Rightarrow (Rb; \leftrightarrow) | q \blacktriangleright \quad , b \Rightarrow \ldots)$$

- $\boxed{\ldots \mid \overrightarrow{a} \mid q \blacktriangleright \mid \ldots}$ must become $\boxed{\ldots \mid \overrightarrow{b} \mid q \blacktriangleright \mid \ldots}$

$$tape \vdash \blacktriangleleft\, q : epat$$
$$\blacktriangleleft\, q = \text{caseR}(a \Rightarrow (Rb; \leftrightarrow) | q \blacktriangleright \quad , b \Rightarrow \ldots)$$

With this we have a translation scheme to translate any TM to our model. If s is the start state of the TM, we can use $s \blacktriangleright$ to emulate the TM. (One must still proof that the executions will yield the same result...)

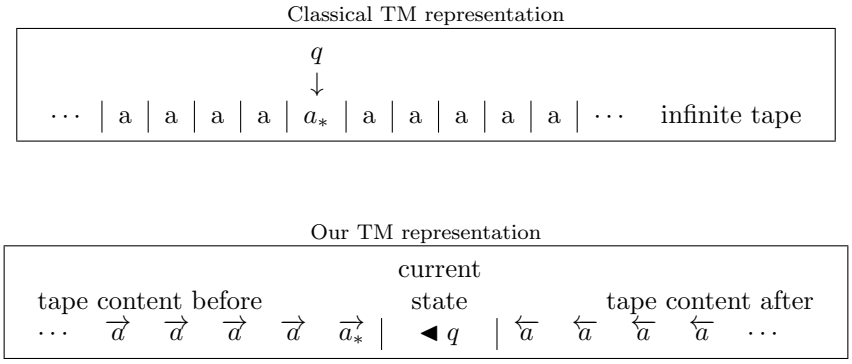

Figure 1: *Top*: A classical TM that is in state q and has a_* on the tape. *Bottom*:Visualization of a process "◀ q" on a tape. The Turing machines current cell under the head contains a_*