Turing Machines in the Pi-Calculus

Iain Phillips

iccp@doc.ic.ac.uk

Department of Computing

Imperial College

London SW7 2BZ

November 1, 1996

In this brief note we show how to model Turing machines [Tur36] in the pi-calculus [MPW92]. It has long been known that TMs can be modelled in CCS [Mil89] and other process algebras [BBK87], and so of course this is no surprise. However it provides quite a nice example of the pi-calculus in action, since we can move the channels around to model the movement of the head along the tape.

First let us create the two-way infinite blank tape $T(rd, wr)$. $C(rd, wr, outL, outR, \sigma)$ represents the cell being scanned containing symbol $\sigma \in \Sigma \cup \{B\}$ (where B is the blank symbol). $BL(inR, outL, \sigma)$ represents a cell containing σ to the left of the head. Similarly for $BR(inL, outR, \sigma)$. Finally $LT(inR)$ represents the unscanned portion of the tape to the left of the head. Similarly for $RT(inL)$.

Following a convention of Milner, we enclose names in parentheses if they can be further instantiated, and in angle brackets if they cannot.

$$T(rd, wr) = (\nu\, outL, outR)(LT\langle outL\rangle | C\langle rd, wr, outL, outR, B\rangle | RT\langle outR\rangle)$$

$$\begin{aligned} C(rd, wr, outL, outR, \sigma) \quad = \quad & \overline{rd}\langle\sigma\rangle.C\langle rd, wr, outL, outR, \sigma\rangle + \\ & wr(\sigma').C\langle rd, wr, outL, outR, \sigma'\rangle + \\ & wr\langle L\rangle.(\nu\, inL)outL\langle rd, wr, inL\rangle.BR\langle inL, outR, \sigma\rangle + \\ & wr\langle R\rangle.(\nu\, inR)outR\langle rd, wr, inR\rangle.BL\langle inR, outL, \sigma\rangle \end{aligned}$$

$$BL(inR, outL, \sigma) = inR(rd, wr, outR).C\langle rd, wr, outL, outR, \sigma\rangle$$

$$BR(inL, outR, \sigma) = inL(rd, wr, outL).C\langle rd, wr, outL, outR, \sigma\rangle$$
$$LT(inR) = inR(rd, wr, outR).(\nu\, outL)(LT\langle outL\rangle | C\langle rd, wr, outL, outR, B\rangle)$$
$$RT(inL) = inL(rd, wr, outL).(\nu\, outR)(C\langle rd, wr, outL, outR, B\rangle | RT\langle outR\rangle)$$

This completes the definition of the tape $T(rd, wr)$.

Now we define the input process $I_w(rd, wr, st)$ which places the word $w \in \Sigma^*$ on the tape and then starts the program proper. The definition is by induction on w:

$$I_\epsilon(rd, wr, st) = \overline{st}\langle rd, wr\rangle.\mathbf{0}$$
$$I_{w\sigma}(rd, wr, st) = \overline{wr}\langle\sigma\rangle.\overline{wr}\langle L\rangle.I_w\langle rd, wr, st\rangle$$

The tape with input word w is therefore

$$IT(st) = (\nu\, rd, wr)(I_w\langle rd, wr, st\rangle | T\langle rd, wr\rangle)$$

Now we define the process $P(st, h)$ corresponding to a deterministic TM program M. M has states $q \in Q$ and initial state q_0. Instructions are of the form $(q, \sigma, \sigma', d, q')$ where $d = L, R, -$.

$$P(st, h) = st(rd, wr).P\langle rd, wr, h, q_0\rangle$$
$$P(rd, wr, h, q) = rd(\sigma).P\langle rd, wr, h, q, \sigma\rangle$$
$$P(rd, wr, h, q, \sigma) = \begin{cases} \overline{wr}\langle\sigma'\rangle.\overline{wr}\langle d\rangle.P\langle rd, wr, h, q'\rangle & \text{if } (q, \sigma, \sigma', d, q'),\ d \text{ is } L \text{ or } R \\ \overline{wr}\langle\sigma'\rangle.P\langle rd, wr, h, q'\rangle & \text{if } (q, \sigma, \sigma', -, q') \\ \overline{h}\langle rd, wr\rangle.\mathbf{0} & \text{otherwise (M halts)} \end{cases}$$

Now the whole system is

$$PIT(h) = (\nu\, st)(P\langle st, h\rangle | IT\langle st\rangle)$$

It can be seen that if M halts then $PIT(h)$ can eventually perform h after some number of τ actions. If M does not halt then $PIT(h)$ performs τs for ever.

If we wish to output the result by reading the tape starting from the final position of the head and moving to the right until we encounter a blank, we can use the following output process:

$$O(h, out) = h(rd, wr).O\langle rd, wr, out\rangle$$
$$O(rd, wr, out) = rd(\sigma).O\langle rd, wr, out, \sigma\rangle$$
$$O(rd, wr, out, \sigma) = \begin{cases} \overline{out}.\mathbf{0} & \text{if } \sigma = B \\ \overline{out}\langle\sigma\rangle.\overline{wr}\langle R\rangle.O\langle rd, wr, out\rangle & \text{otherwise} \end{cases}$$

Now the complete system is

$$PITO(out) = (\nu\, h)(PIT\langle h\rangle | O\langle h, out\rangle)$$

References

[BBK87] J.C.M. Baeten, J. Bergstra, and J.-W. Klop. On the consistency of Koomen's fair abstraction rule. *Theoretical Computer Science*, 51:129–176, 1987.

[Dav65] M. Davis, editor. *The Undecidable*. Raven, New York, 1965.

[Mil89] R. Milner. *Communication and Concurrency*. Prentice-Hall, 1989.

[MPW92] R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes. *Information and Computation*, 100:1–77, 1992.

[Tur36] A.M. Turing. On computable numbers, with an application to the entsheidungsproblem. *Proc. Lond. Math. Soc.*, 42:230–65, 1936. Reprinted in [Dav65].