Handout: More About Turing Machines

Laura Kovács May 23, 2013

---

1. Decision problems, instances. A *decision problem* (I, f) has the form "Given $x \in I$, is $f(x) = \textsc{Yes}$ or $f(x) = \textsc{No}$?", for some set I of possible inputs, and some function f that maps all inputs $x \in I$ to either Yes or No. Here are two examples of decision problems:

GraphReachability Given a directed graph (V, E) and two nodes $u, w \in V$, is there a path from u to w?

HilbertsTenth Given a multivariate polynomial equation e, does e have an integer solution?

An *instance* of the decision problem "Given $x \in I$, is $f(x) = \textsc{Yes}$ or $f(x) = \textsc{No}$?" is a particular input $x \in I$. Here are two sample instances, one for each of the above problems:

GraphReachability

$$V = \{v_0, v_1, v_2, v_3\}$$
$$E = \{(v_0, v_1), (v_1, v_2), (v_3, v_2)\}$$
$$u = v_1$$
$$w = v_3$$

This is a No-instance, because this particular graph has no path from v_1 to v_3.

HilbertsTenth

$$e: \quad 3x^2y + 27x^{13}z^2 - xyz + 5z = 0$$

This is a Yes-instance, because $x = y = z = 0$ is an integer solution of e.

2. Decidability. Let Question be the decision problem "Given $x \in I$, is $f(x) = \textsc{Yes}$ or $f(x) = \textsc{No}$?". The set

$$L(\textsc{Question}) = \{x \mid x \in I \ \wedge \ f(x) = \textsc{Yes}\}$$

of Yes-instances is a language. The decision problem Question is *recursive*, or *decidable*, if the language $L(\textsc{Question})$ is recursive. Decidable problems can be solved by deterministic Turing deciders: for a recursive problem Question, there is a deterministic Turing decider M such that if you wish to know whether $f(x) = \textsc{Yes}$ or $f(x) = \textsc{No}$ for some $x \in I$, you can run M with the initial tape content x and wait until it halts. If M halts in q_a, then the answer is Yes; if it halts in q_r, then the answer is No. Since M is a Turing decider, it is guaranteed to halt.

The decision problem Question is *r.e.*, or *semi-decidable*, if the language $L(\textsc{Question})$ is r.e.

3. Examples. The problem GraphReachability is decidable. The problem HilbertsTenth is semi-decidable and is not decidable. The proof of HilbertsTenth not being decidable was a major mathematical breakthrough in the 20th century.

4. Church-Turing thesis. Since a DTM is a very simple computational apparatus (you surely can build a simulator of DTMs in your favorite programming language), it follows that every recursive problem can be solved by computational means. The Church-Turing thesis states the converse, namely, that all problems which can be solved by computational means are recursive. This is a claim that cannot be proved, because any proof would require an unassailable, general definition of what it means to solve a problem "by computational means." But the claim has been proved to be true for all specific definitions of "computational means" that people have suggested. For example, all problems that can be solved by Java programs, even when run on idealized machines with unbounded memory, are recursive. So in a very fundamental sense, Java programs are no more "powerful" than DTMs: both Java programs and DTMs can solve the same problems (namely, the recursive ones). And the same can be said for every programming language.