The Halting Problem
Lecture Notes COMP 330 Winter 2021

Prakash Panangaden

10^{th} March 2021

Suppose that we have a programming language in which we can manipulate strings, write `while`-loops, `if-then-else` statements, compose statements, for example with `;` and do basic arithmetic. In this language, some of the programs halt on some of their inputs and on some other inputs they do not halt but run forever.

Note that the source code of a program is itself just a (long) string. We use the notation $\#P$ to mean the source code of the program P viewed as a string. Thus the source code of a program is just data that can be used by other programs. A compiler is just such a program; it gets the source code of a program as input and generates an executable version (which is just another string) as output. If P is a program and x is an input for the program we write $P(x)$ for the effect of running P with input x. If C is a compiler then we can write $C(\#P)$ to mean, run C with with input the source code of P. There are lots of programs that take the source code of other programs as inputs and do something, perhaps answer some question about the property of the input program. A program could also take 2 inputs; if P is such a program and x and y are inputs to P, we can write $P(x, y)$ for the effect of running P with inputs x and y.

Suppose we have a program H that takes two inputs. Suppose that this H can "solve" the halting problem. If we give H input $\#P$ and x: $H(\#P, x)$ we get `true` if $P(x)$ halts and `false` if $P(x)$ loops forever. Now I can write a new program called S with the following source code:

```
Input : w
if H(w,w) then loop forever else halt.
```

It is easy to write something that loops forever (students in 202 are expert at this). So everything in the above program S can easily be written in our language. Note that S expects only one input, it makes copies of that input.

Now consider $S(\#S)$. Does it halt or not? If we unpack this we get: `if H(\#S,\#S) then loop forever else halt.`
Now if $S(\#S)$ halts, the call to H will return `true` and we go to the `then` branch and loop!! If $S(\#S)$ loops forever, then the call to H returns `false` and we go to the `else` branch and halt! In either case we have a contradiction. Thus a program like H cannot exist.