Computation Theory

p3q1
JKMM

2

Solution "There is no algorithm to decide whether a ~~particular~~ general computation will terminate". That must be made precise. First, to specify a "general computation" means adopting a specific coding for both the algorithm and the initial data to it. "decide" means that the decision algorithm must always terminate, returning a single bit of information to indicate whether given computation would HALT.

Question 1 Solution ctd)

In the notes I proved the result for register machines, using the coding adopted in the explicit universal machine, and returning the bit as 0 or 1 in register A_1 at termination

5

To record the state of play in a 2-reg machine computation requires a triplet (pc, a_1, a_2).

3

Suppose given a program code p for a 2-register machine computation, also a stack code d representing initial register contents

Question 1 Solution ctd)

Then certainly one can extract the number of instructions in the program, also the initial register contents, say x_1, x_2.

Suppose we could compute functions $M_i(p, x_1, x_2)$ giving bounds on the contents of registers A_i during HALTing computations.

Now given a general 2-register computation coded by p, d proceed as follows. First extract the number n of instructions in the program, and initial arguments x_1, x_2. Next calculate functions $M_i(p, x_1, x_2)$

Question 1 Solution ctd)

to obtain bounds K_1, K_2 on the contents of registers A_1, A_2 for halting computations.

If a computation enters the same configuration twice it is looping! So no terminating computation can take more than $n K_1 . K_2$ steps ...

Since 2-register computations are general, we have solved the HALTing problem! The conclusion is that I cannot compute both M_i

(12)

I've made a bit of a meal of it. It's quite easy, but they won't have seen anything quite like it. We shall see.