Homework 9: Due Tuesday June 2, 1pm

1. (10 pts) Construct a Turing machine so that if the head of the machine is positioned starting at the left of a positive integer n written in binary, the machine halts outputting $n-1$ in binary.

2. (no collab, 10 pts) A *finite tape* Turing machines has a tape with a fixed finite number of cells l. If the machine tries to move left at the left edge of the tape or right at the right edge of the tape, the head stays in the same place.

   Give an algorithm which given a finite tape Turing machine program and an input written on its tape, decides whether the Turing machine will eventually halt or not.

   [Hint: A *configuration* of a Turing machine consists of what state the Turing machine program is in, and what is written on its tape. The first thing you'll want to show is that there are only finitely many configurations that a given finite tape Turing machine can have.]

3. (40 pts.) Consider Turing machines on the alphabet $\{1, \diamondsuit\}$. Let T_n be the set of Turing machines on this alphabet that have *at most* n states. Clearly T_n is finite.

   For each TM M on $\{1, \diamondsuit\}$, let P_M be defined by

   $P_M =$the number of 1's appearing in the output, when the input is M is θ (the empty word) and M stops on that input; 0 otherwise.

   So $P_M \in \mathbb{N}$. Let

   $$B(n) = \max\{P_M : M \in T_n\}.$$

   So $B : \mathbb{N} \to \mathbb{N}$. (By convention $B(0) = 0$.) So for each $n \geq 1, B(n)$ is the maximum number of 1's that a TM with at most n states can print in the output, assuming it stops, when started at the empty input. We call B the *busy beaver function*.

   Prove that there is no TM on the alphabet $\{1, \diamondsuit\}$ which computes B, i.e., when presented with input $**\underbrace{1\,1\ldots1}_{n}**$ produces as output $**\underbrace{1\,1\ldots1}_{B(n)}**$

   You can take for granted that we can construct a Turing machine M_0 on $\{1, \diamondsuit\}$ that on input

   $$***\underbrace{1\,1\ldots1}_{m}\diamondsuit\underbrace{1\,1\ldots1}_{n}***$$

produces output

$$\cdots * * * \underbrace{1\,1\ldots 1}_{mn} * * * \ldots$$

(i.e., performs multiplication in unary notation). Also you can take for granted that for each k there is a Turing machine M_k on $\{1, \diamondsuit\}$ that on input θ produces output

$$\cdots * * * \underbrace{1\,1\ldots 1}_{k} \diamondsuit \underbrace{1\,1\ldots 1}_{k} * * \ldots$$

and which has at most ak states for some fixed $a \in \mathbb{N}$.

[Hint: Assume, towards a contradiction, that there is such a TM M_B that computes B. Then it is easy to construct a TM M'_B that computes $f(n) = B(n)+1$. (You don't have to write up the details of this.) Consider then the TM M' which is the "concatenation" of M_n followed by M_0 followed by M'_B, so that on input

$$\theta$$

M' gives as output

$$\cdots * * * \underbrace{1\,1\ldots 1}_{B(n^2)+1} * * * \ldots$$

Again you don't have to write up the details of this, but you should note that the number of states of M' is at most: (the sum of the number of states of M_n, M_0, M'_B)$+b$, for some constant b. Derive now a contradiction from this.]