# Homework 4

Joe Baker, Brett Schreiber, Brian Knotten

January 22, 2018

## 5

Definition 1:
A language $L$ is recursively enumerable $\leftrightarrow$ there is a Turing machine $M_1$ such that if $x \in L$ then $M_1$ accepts $x$ and if $x \notin L$ then $M_1$ loops forever on $x$.

Definition 2:
A language $L$ is recursively enumerable $\leftrightarrow$ there a Turing machine $M_2$, with a read/write tape that is initially empty and a write-only output tape, such that only elements of $L$ are written to the output tape, and every element of $L$ is eventually written to the output tape.

Claim:
Definition 1 is logically equivalent to definition 2.

Proof:
We will prove that assuming we have the TM $M_1$ from definition 1, that we can construct TM $M_2$ of definition 2. Next we will show the converse, assuming we have TM $M_2$ from definition 2, that we can construct TM $M_1$. Finally, we will have $M_1$ exists $\leftrightarrow$ $L$ recursively enumerable $\leftrightarrow$ $M_2$ exists.

First, assume that $M_1$ exists. Then construct $M_2$ such that $M_2$ has an input tape, an initially empty read/write tape, and a write-only output tape. Let $M_2$ operate in the following way:

1. $M_2$ writes a counter (0 if starting, 1 + previous value otherwise)
2. $M_2$ reads the counter value and appends a combination of symbols using the counter value as an encoding of the symbols
3. $M_2$ calls $M_1$ with the combination of symbols as input
4. $M_2$ writes the combination of symbols to the output tape if $M_1$ accepts
5. $M_2$ repeats steps 1...4

$M_2$ will construct every valid input from its alphabet in a countable manner. Each input will be checked with $M_1$ and written to the output tape iff the input is in $L$. Thus, $M_1$ exists $\rightarrow$ $M_2$ exists.

Now, assume that $M_2$ exists. Then construct $M_1$ as a modified copy of $M_2$ with an addition read-only input tape which contains input $x$. Also, add the following behavior to $M_1$:

1. $M_1$ reads input $x$
2. $M_1$ calls the steps of $M_2$ until $M_2$ writes to output
3. After $M_2$ writes to output, $M_1$ accepts if the most recent output matches $x$
4. Otherwise, $M_1$ goes back to step 1

If $x \in L$ $M_1$ will detect $x$ in the output tape while running $M_2$ and accept, otherwise $M_1$ will never stop reading the output tape of $M_2$ and loop forever. Thus, $M_2$ exists $\rightarrow$ $M_1$ exists.

Finally, $M_1$ exists $\leftrightarrow$ $L$ recursively enumerable $\leftrightarrow$ $M_2$ exists.

## 6

Define the proof rule as:

$$\forall x \; P(x) \text{ one can deduce the countably infite number of statements } P(0), P(1), ...$$

Let $A$ be a finite set of axioms, $S$ be a statement, and define the language

$$L = \{S \mid S \text{ statement } S \text{ is provable from the finite set of } A \text{ axioms } \}$$

Claim:
$L$ is recursively enumerable.

Proof:
We will show that all possible proofs can be mapped to a tree that can be traversed by a Turing machine $M$ such that $M$ takes as input a statement $S$, accepts if $x \in L$, and loops forever if $x \notin L$. Thus we will construct $M$ in a way that shows $L$ is recursively enumerable.

Consider a tree where each node is a proof using the proof rule. The value of the root node is the axioms of $A$ and its children are the various applications of the proof rule on the axioms. Each internal node has a proof and infinite children derived from applying the proof rule on its proof. Since the proof rule yields infinite statements, this tree has infinite breadth and depth.

Each node in the tree is mapped onto a two-dimensional Cartesian plane. The root is given the coordinates $(0, 0)$. Map the nodes of level $l$ onto the plane such that children from each parent in $l - 1$ are interleaved in increasing order. For example, if the parent level $l - 1$ had 2 parents $\{a, b\}$ and $children(a) = \{a_1, a_2\}$, $children(b) = \{b_1, b_2\}$, then the children would be mapped in level $l$ in the order: $\{a_1, b_1, a_2, b_2\}$ with coordinates $(l, 0)$ through $(l, 3)$.

Construct a Turing machine $M$ that takes statement $S$ as input and traverses the aforementioned tree to find a proof of $S$. $M$ will traverse the tree by iterating over each level starting at 0 and moving up and over to the right diagonally. For example, the first few iterations will be $(0, 0), (1, 0), (0, 1), (2, 0), (1, 1), (0, 2), ...$ . Then, $M$ will accept the proof at coordinate $(x, y)$ iff it can prove $S$, meaning $S \in L$.

Clearly if $S$ is not provable by the proofs in the tree, $M$ will iterate forever trying to find it. Now consider $S$ is provable by a proof in the tree. This proof will be in a node which has a path from the root so it will be mapped to some coordinate $(x, y)$. By our construction, $M$ will at some point iterate over that pair and accept. Thus $M$ accepts iff $S \in L$. Finally, $L$ is recursively enumerable.