

# Study Formulæ: Chapter 10

Philip Gross

December 21, 1998

## 10 Recursively Enumerable and Recursive Languages

### 10.1 Recursively Enumerable and Recursive

**Def 10.1:** Let  $L \subseteq \Sigma^*$  be a language. A Turing machine  $T$  with input alphabet  $\Sigma$  is said to *accept*  $L$  if  $L(T) = L$ . The TM  $T$  *recognizes*, or *decides*,  $L$  if  $T$  computes the characteristic function  $\chi_L : \Sigma^* \rightarrow \{0,1\}$ . In other words,  $T$  recognizes  $L$  if  $T$  halts for every string  $x$  in  $\Sigma^*$ , producing output 1 if  $x \in L$  and output 0 otherwise.

A language  $L$  is *recursively enumerable* if there is a TM that accepts  $L$  and *recursive* if there is a TM that recognizes  $L$  (i.e. will always return 1 if  $x \in L$ , 0 otherwise).

Every language that is recursive is recursively enumerable: just crash instead of returning 0.

**Theorem 10.1:** If  $L$  is accepted by a nondeterministic Turing machine  $T$ , and every possible sequence of moves of  $T$  results in a halt or a crash, then  $L$  is recursive.

**Theorem 10.2:** If  $L_1$  and  $L_2$  are recursively enumerable languages over  $\Sigma$ , then  $L_1 \cup L_2$  and  $L_1 \cap L_2$  are also recursively enumerable.

**Theorem 10.3:** If  $L$  is recursive, so is  $L'$ .

**Theorem 10.4:** If  $L$  is a recursively enumerable language whose complement is recursively enumerable, then  $L$  is recursive.

### 10.2 Enumeration

**Def 10.2:** Let  $T$  be a  $k$ -tape Turing machine, where  $k \geq 2$ , and let  $L \subseteq \Sigma^*$ . We say that  $T$  enumerates  $L$  if it operates so that the following conditions are satisfied:

1. The tape head on the first tape never moves to the left.
2. At each point in the operation of  $T$ , tape 1 has the contents

$$x_1 \# x_2 \# \cdots \# x_n \# y$$

where  $n \geq 0$ , each  $x_i \in L$ , the  $x_i$ 's are distinct, and  $y$  is a prefix of an element of  $L$ .

3. For every  $x \in L$ ,  $x$  eventually appears as one of the strings  $x_i$  on tape 1.

**Theorem 10.5:** A language  $L \subseteq \Sigma^*$  is recursively enumerable (i.e., can be accepted by some TM) iff  $L$  can be enumerated by some TM.

**proof:** Enumerate  $\rightarrow$  Accept: instead of writing a '#' after each string, check to see if we match the input string.

Accept  $\rightarrow$  Enumerate: set up two other tapes. Generate all strings in  $\Sigma^*$  and feed them to the accepting TM. If accepted, add to enumeration list.

**Theorem 10.6:**  $L$  is recursive iff there is a TM that enumerates  $L$  in canonical order.

### 10.3 Some Languages are not Recursively Enumerable

**Def 10.3:** A set  $S$  is *countably infinite* if there is a bijection from  $\mathcal{N}$  to  $S$ , and *countable* if it is either countably infinite or finite. A set is *uncountably infinite*, or simply *uncountable* if it is not countable.  $f : \mathcal{N} \rightarrow S$  is a bijection implies:

1. For every natural number  $n$ ,  $f(n) \in S$
2. For any two different numbers  $m$  and  $n$ ,  $f(m) \neq f(n)$ .
3. Every element of  $S$  is  $f(n)$  for some natural number  $n$ .

**Lemma 10.1:** Every infinite set has a countably infinite subset.

**proof:** Brain twister: assign  $f(0)$  to some element of  $S$ . For each  $f(n+1)$  assign it to some other element of  $S$  – hey, there's infinitely many where that came from.

**Theorem 10.7:** Suppose that  $S_0, S_1, \dots$  are countable sets. Then the set

$$S = \bigcup_{n=0}^{\infty} S_n$$

is countable.

**proof:** Arrange the  $S$ 's in columns. Number them in a diagonal pattern starting from upper left.

**Theorem 10.8:** If  $S$  is any countably infinite set, then the set  $2^S$  of all subsets of  $S$  is uncountably infinite. In particular for any nonempty alphabet  $\Sigma$ , the set of languages over  $\Sigma$  is uncountable.

**Corollary 10.1:** The set of languages over  $\{0,1\}$  that are not recursively enumerable is uncountable. In particular, there exists at least one such language.

## 10.4 A Language that is not Recursively Enumerable

**Def 10.4:** Let  $NSA$  (not self-accepting) be the following subset of  $\{0,1\}^*$ :

$$NSA = NSA_1 \cup NSA_2$$

where

$$NSA_1 = \{w \in \{0,1\}^* \mid w = e(T) \text{ for some TM } T, \text{ and } T \text{ does not accept } w\}$$

$$NSA_2 = \{w \in \{0,1\}^* \mid w \text{ is not } e(T) \text{ for any TM } T\}$$

Let  $SA$  (self-accepting) be the complement of  $NSA$  in  $0,1^*$ , so that

$$SA = \{w \in \{0,1\}^* \mid w = e(T) \text{ for some TM } T, \text{ and } T \text{ accepts } w\}$$

**Theorem 10.9:** The language  $NSA$  is not recursively enumerable.

**Theorem 10.10:** The language  $SA$  is recursively enumerable but not recursive. In other words, although  $SA$  can be accepted by a Turing machine, any TM accepting it will loop forever on at least one input string not in the language.