More Turing Machines

CS154
Chris Pollett
Mar 15, 2006.

Outline

- Multitape Turing Machines
- Nondeterministic Turing Machines
- Enumerators

Introduction

- There have been many different proposals for what it means to compute something.
- We will look at a few based on Turing Machines and show they don't add to the class of computable algorithms.
- As a quick example, consider TMs where the transition function is δ: $Q \times \Gamma \dashrightarrow Q \times \Gamma \times \{L,R, S\}$ where S means the head stays put.
- We could simulate such a with our original machine by replacing each stationary transition with a move right followed by a move left transition.

Multitape Turing Machines

- A Multitape Turing Machine (MTM) is like a usual Turing machine except that it has some finite number of tapes and tape heads rather than just one.
- The transition function now looks like:

  δ: Q x Γ^k --> Q x Γ^k x $(L, R, S)^k$
- So $\delta(q, a_1, .., a_k) = (q', b_1, .., b_k, L, R,..S)$ says in state q reading from tape 1 an a_1, from tape 2 an a_2,…; we transition to state q', write to tape 1 a b_1, write to tape 2 a b_2, … then move left on tape 1, right on tape 2, … finally stay put on tape k.
- Initially, the input is assumed to be written on the left hand side of the first tape.
- The machine halts if an accept or reject state is entered. If it halts in an accept state then it accepts the input

Simulation of Multitape machines

Theorem Any language recognizable by an MTM is recognizable by a usual TM.

Proof: Let M be a k-tape MTM. We will describe how to simulate it with a single tape machine S. For each tape symbol 'a' of M, the machine S will have two symbols 'a' and '$\underline{a}$'. We will also have a special new tape symbol '#' not in M's tape alphabet called the configuration separator. As M's tape alphabet is finite so is S's. Here is a higher level description of how M operates on input $w=w_1\ldots w_n$:

1. First S does a single pass over the input string to convert it to the form $\#\underline{w}_1 w_2\ldots w_n\#\underline{\ }\#\underline{\ }\#\ldots\#$. This is supposed to represent the initial configuration of each of M's tapes. The underscore denotes where each tape head is currently. S now rewinds to the start of the tape
2. To simulate a move of M, the machine S scans left to right, looking for underscored characters. When it finds one it tries to simulate the action of the machine on that tape. It knows which tape it is dealing with by remember how many '#' it has seen. This will always be bounded by k+1.
3. If at any point S tries to move a virtual head over a #, S first writes a space on the #, shifts the contents of everything to the right over by 1, finds the space it wrote and continues the simulation.

Nondeterministic Turing Machines

- Nondeterministic Turing Machines (NTMs) are defined as one would guess.
- One modifies the transition function so that now it is a map:

  $\delta: Q \times \Gamma \text{ --> } P(Q \times \Gamma \times (L, R, S))$.

Simulation of Nondeterministic Turing Machines

Theorem Any language recognizable by an NTM is recognizable by a usual TM.

Proof: Let N be a NTM that recognizes some language. We will make a Turing Machine D to simulate N. The idea is to have D try all possible branches of N's computation. If D ever finds an accept state of N on any of the branches then D accepts. So that we don't get stuck on infinite branches we will do our simulation in an iterated deepening, breadth first manner. D will have three tapes: the input tape, a simulation tape, and an address tape. D operates as follows:

1. Initially the input tape has the input and the other two tapes are blank.
2. D copies the input tape to the simulation tape.
3. D then simulate N according to the nondeterministic choices on the address tape. Let m be the finite maximum number of choices in any given state reading a given symbol for a next state.
4. If the address tape is blank D, checks to see if N immediately accepts. Otherwise, D writes a 1 on the address tape and simulates N one step using the first possible nondeterministic choice. If this doesn't accept. Then D writes a 2 over the 1, erases the simulation tape, and simulate N according to the second nondeterministic choice. Once we have tried all single step computations, we then cycle over computations of length 2, etc.

Enumerators

- The Turing Recognizable languages are sometimes called the recursive enumerable languages.
- Enumerate means to list out.
- We can imagine a machine with several work tapes and a dedicated output tape.
- The machine computes forever but might periodically enter an output state, when it does the string on the output tape is said to be in the language recognized by the enumerator.