Institut for Matematik og Datalogi
Syddansk Universitet
JBJ
February 22, 2024

DM553 – Spring 2024– Weekly Note 5

Material covered in Week 8:
Sections 3.2 and 3.3 in Sipser. The relevant videos are Videos 8, 9 and 10.

Key points:

- Many different variants of the Turing machines have been defined, and none of them have the more computational power than a normal deterministic TM.
- A nondeterministic TM is not more powerful than a standard TM either. However, nondeterministic TM **may** be exponentially faster (but we don't know whether this is true). This open question is the well known P = NP question.
- A TM is said to **enumerate** a language L if it, when started on an empty tape, prints all strings in L to an attached printer (and no strings that are not in L). Such a Turing machine with a printing tape is called an **enumerator**. A language L is called **Turing-enumerable** if L = L(E) is some enumerator E. We proved that L is Turing-acceptable (is accepted by some Turing machine) if and only if L is Turing-enumerable.
- The **Universal Turing machine** U. I showed that we can code all Turing machines such that their tape alphabet is a subset of the so-called universal alphabet $A^* = \{a_1, a_2, \ldots\}$ and the state set is a subset of the universal state set $Q^* = \{q_1, q_2, \ldots\}$. This gives rise to an encoding $< M >$ of a Turing machine where we only use symbols from A^*, Q^* and few extra special symbols which we use to separate letters and transitions. Using a binary encoding of integers we can further reduce the needed symbols to be the set $\{q, a, 0, 1, (,), R, L, S\}$ and komma (plus possibly also the symbols $<, >$). Similarly, every string w has a code $< w >$ where w is expressed in the universal alphabet A^*, e.g. $< w >= (a_4)(a_6)(a_1)$ means that w consists of the fourth, the sixth and the first symbol from A^* in that order. Using the codings $< M >$ and $< w >$ the universal Turing machine can simulate M on w and U will accept/reject/loop on input $< M >< w >$ if and only if M accepts/rejects/loops on w.
- A 2-PDA is a PDA with two stacks instead of one. This increases the computational power immensely. In fact: 2-PDAs are equivalent (have the same computational power) to Turing machines (see the exercises below).

Topics to be covered in Week 9

- Section 4.1 on decidability (video 11)
- Section 4.2 on undecidability (video 12)
- Sipser 5.1 pages 215-220 (in both books). The rest of Section 5.1 as well as Section 5.2 will not be covered and are not part of the pensum for the course. (Video 13)
- Sipser 5.3 (Video 13)

Exercises in Week 9

- 3.15 (a)-(d) (3.16 in 3rd edition). Hint: you must simulate Turing machines in parallel if you consider two at the same time (why?).
- 3.16 (3.15 in 3rd edition)
- 3.18 (3.11 in 3rd edition). Hint: to show that a normal TM M can simulate a TM M_2 with a 2-way infinite tape by labelling the cells of the tape of ..., -2,-1,1,2,3,... and then using the standad one-way infinite tape as follows: there is a marker # in the the leftmost position and the position i of M's tape contains a pair (a_i, a_{-i}) where a_i is the content of cell i and a_i is the content of cell $-i$ on M_2's tape. Now M can work on the cells with a negative index by considering the second coordinate of such a pair.
- 3.22 (3.9 in 3rd edition). Hint show how to use two stacks to simulate a Turing machine. Let the first stack contain what is to the left of the tape head and the second the other part.
- Describe in words a 2-PDA for recognizing the language $\{a^n b^n c^n d^n | n \geq 0\}$.
- Show that every 2-PDA can be simulated by a 3-tape Turing machine.
- January 2008 problem 4 (Note that a Turing machine calculates a function f if it, starting in configuration $q_0 w$ terminates in configuration $q_{accept} f(w)$ for every legal input w).