CS 2401 - Introduction to Complexity Theory

Lecture #1: Fall, 2015

Lecturer: Toniann Pitassi

Scribe Notes by: Alex Edmonds

1 Course Topics

- Turing machines
- Main complexity classes:
  - Time: P, NP, EXP, E, PH, NP-complete
  - Space: L, NL, PSPACE
  - Randomized complexity classes: RP, BPP
- Concrete computational models and lower bounds

2 Turing Machines

Definition A Turing machine M is described by a tuple (Γ, Q, δ) consisting of:

- Γ, called the *alphabet* of M, is a set of symbols which includes 0, 1, $\not{b}$, $\triangleright$ where $\not{b}$ and $\triangleright$ stand for "blank" and "start" resp.
- Q, a set containing the possible states of M, which includes q_{start} and q_{halt}
- $\delta : Q \times \Gamma^k \to Q \times \Gamma^{k-1} \times \{L, R, S\}^k$ called the *transition function*, where L, R and S stand for "left", "right" and "stay" resp.

The Turing machine (TM) is represented by k tapes. Each *tape* is a sequence of cells, infinite in one direction, where each cell contains a symbol from Γ. The first tape is designated the *input tape*. Initially, all tapes contain $\triangleright$ in the first cell and all tapes except the input tape contain $\not{b}$ everywhere else. The input tape begins with a finite sequence of non-blank symbols followed by $\not{b}$ everywhere else. Each tape always has one of its cells marked by a *head*. Initially, these heads mark the first cells of each tape.

On each step, the machine is in some state $q \in Q$ and the machine reads each cell marked by a head to obtain $(\sigma_1, \sigma_2, \dots \sigma_k)$. Let

$$(q', (\sigma'_2, \dots, \sigma'_k), (z_1, \dots, z_k)) = \delta(q, (\sigma_1, \dots, \sigma_k)) \tag{1}$$

Then, the following occurs:

1. On the i^{th} tape, in the cell marked by the head, σ_i is replaced with σ'_i.

2. On the i^{th} tape, the head moves to the left, right or stays in place according to $z_i \in \{L, R, S\}$.
3. The state of the machine is updated to q'. If $q' = q_{\text{halt}}$, the machine stops and ceases to update the tapes.

Definition Let $f : \{0,1\}^* \to \{0,1\}^*$. We say that the TM M *computes* f if, for $x \in \{0,1\}^*$, when M is run on x, it halts with $f(x)$ on its output tape (tape k). We say M computes f in time $T(n)$ if, for all n, for all x where $|x| = n$, the number of steps M takes on input x is at most $T(n)$.

3 Church-Turing Thesis

The Church-Turing thesis claims that $f : \{0,1\}^* \to \{0,1\}^*$ can be realized in some computational model or device iff f can be computed by some TM. It is noted that it is impossible to discuss the formal validity of such a statement since there does not exist a general characterization of computational models.

4 On Restrictions of Turing Machines

4.1 Restriction of alphabet Γ

Fact (Claim 1.5) If $f : \{0,1\}^* \to \{0,1\}^*$ is computed by $M = (Q, \Gamma, \delta)$ then f can also be computed by $M' = (Q', \Gamma', \delta')$ where $\Gamma' = \{\triangleright, 0, 1, \not{b}\}$. If M runs in time $T(n)$, then M' runs in time $T'(n) = 4T(n) \log |\Gamma|$.

Proof *(sketch)* Every tape of M is encoded by a tape of M'. Because any member of Γ may be encoded in $\log |\Gamma|$ bits, every cell in a tape of M may be represented by $\log |\Gamma|$ cells in the tape of M'. To simulate a step of M, M' does the follwing for each tape in parallel:

1. reads the $\log |\Gamma|$ sequence of cells corresponding to the one read by M;
2. uses its state register to store the symbol read;
3. uses the transition function of M to compute the symbols M writes, etc.;
4. stores this information in its state register;
5. uses $\log |\Gamma|$ steps to write the encodings of these symbols on its tapes.

To allow for such a procedure, the state register Q' of M' is extended so that it may represent k symbols in Γ and a counter from 1 to $\log |\Gamma|$. Hence, $|Q'| \leq O(|Q||\Gamma|^{k+1})$. Corresponding to a single step in M, $4 \log |\Gamma|$ steps may be required in M', whereby we obtain $T'(n) = 4T(n) \log |\Gamma|$.

4.2 Restriction to single tape

Fact (Claim 1.6) If f is computable by a k-tape TM $M = (Q, \Gamma, \delta)$ in time $T(n)$, then f is computable by a 2-tape (or 1-tape read/write) TM $M' = (Q', \Gamma', \delta')$ in time $T'(n) = 5T(n)^2$.

Proof *(sketch)* Extend $\Gamma = \{z_1, z_2, \ldots, z_n\}$ to $\Gamma' = \{z_1, z_2, \ldots, z_n, \hat{z}_1, \hat{z}_2, \ldots, \hat{z}_n\}$. The i^{th} tape of M is inscribed in the cells $i, i+k, i+2k, \ldots$ of M'. Where a cell in a tape of M is marked by a head and contains z_j, the corresponding cell in M' is inscribed with $\hat{z}_j$ instead. Corresponding to a single step in M, M' sweeps its entire tape to register those symbols marked by "ˆ" and, after registering the results of M's transition function, sweeps the entire tape once again to update the encoding accordingly. Since M never reaches further than location $T(n)$ on its tapes, M' never reaches further than $2n + kT(n) \leq (k+2)T(n)$. Thus, for a single step of M, the corresponding sequence of steps in M', described above, may be as long as $5kT(n)$, which accounts for some extra steps needed for updating head movement and book keeping. This gives $T'(n) = 5kT(n)^2$.

5 Universal Turing Machines

Fact Every TM may be represented by binary strings such that each string $\alpha \in \{0,1\}^*$ represents a TM denoted M_α (if some string is not a legal representation, map it to a trivial TM) and every TM is represented by infinitely many strings.

Theorem 1 *There is a TM $\mathcal{U}$ such that, for $x, \alpha \in \{0,1\}^*$, $\mathcal{U}(x, \alpha) = M_\alpha(x)$. Moreover, if the running time of M_α is $T(n)$, then, for $|x| \leq n$, $\mathcal{U}(x, \alpha)$ takes at most $C \cdot T(n) \log T(n)$ steps where C depends only on the alphabet size and number of tapes of M_α*

Proof *(sketch)* $\mathcal{U}$ consists of an input tape and three work tapes. The first work tape is a simulation of the work tape of M_α. Here, we invoke Claim 1.5 and Claim 1.6 so that we may assume M_α consists of single work tape and uses the same alphabet as $\mathcal{U}$, although these transformations introduce quadratic slowdown so that M_α runs in time $C \cdot T(n)^2$. Another work tapes of $\mathcal{U}$ is used to store the values of the transition function of M_α. The last work tape is used to keep record of the current state of M_α. Other techniques are required to obtain the tighter bounds of the theorem as stated above (see section 1.7 of Arora & Borak).

6 Undecidable Functions

Some functions, indeed most, are not computable. For instance,

$$UC(\alpha) = \begin{cases} 0 & \text{if } M_\alpha(\alpha) \text{ outputs } 1 \\ 1 & \text{otherwise} \end{cases}$$

is not computable. To see this, take any TM M and its encoding α so that $M = M_\alpha$. By definition, $M_\alpha(\alpha)$ and $UC(\alpha)$ do not agree.

The most famous undecidable function is the halting function:

$$HALT(x, \alpha) = \begin{cases} 1 & \text{if } M_\alpha \text{ halts on input } x \\ 0 & \text{otherwise} \end{cases}$$

We prove that it is undecidable by showing that its computability would imply the computability of UC, thereby deriving a contradiction. Indeed, suppose there exists TM M_{HALT} which computes $HALT$. Then we can compute UC with M_{UC} defined as follows: first run $M_{HALT}(\alpha, \alpha)$; if the result is 0, meaning $M_\alpha(\alpha)$ does not halt, return 1; otherwise, run $\mathcal{U}(\alpha, \alpha)$ and return its negation.

7 Time Complexity Classes

Definition A *language* L is a subset of $\{0,1\}^*$. This may be equivalently represented by a boolean function $f : \{0,1\}^* \to \{0,1\}$.

Definition A language L is in $DTIME(T(n))$ iff $\exists$ TM M, $\exists c > 0$ so that M runs in time $c\dot{T}(n)$ and decides L, i.e. $[x \in L \Rightarrow M(x)$ outputs 1$]$ and $[x \notin L \Rightarrow M(x)$ outputs 0$]$.

Definition P, which stands for *polynomial time*, is defined by $P = \bigcup_{k=1}^{\infty} DTIME(n^k)$.

Some important languages in P include:

- graph connectivity
- linear programming
- primality testing
- greatest common denominator
- circuit evaluation

Definition $L \in NP$ if there exists a polynomial $p(n) = n^c$ and a polytime TM M such that, for all x,

$$x \in L \Leftrightarrow \exists u \; \{|u| \leq |x|^c \; \& \; M(x,u) = 1\} \tag{2}$$

Here, u is referred to as the *witness*.

Important languages in NP include:

- $P \subseteq NP$
- independent set problem (IS)
- integer linear programming
- SAT, circuit-SAT, 3SAT