LaTeX and Turing Machines

CS254

Chris Pollett

Aug. 30, 2006.

Outline

- LaTeX
- Single Tape Turing Machines

What is LaTeX?

- LaTeX is a markup language which can be used to specify how to typeset a document.
- It is used to prepare papers containing mathematical notation for publication.
- Most papers in theoretical computer science are written in LaTeX.

Sample Document

```
\documentclass[12pt]{article}
% create some commands of my own with people's names with accents
\newcommand{\Hastad}{H{\aa}stad}
\newcommand{\Pudlak}{Pudl\'{a}k}
% define the title
\author{C. Pollett}
\title{Simple Document}
\begin{document}
% generates the title
\maketitle
\section{This is a section title}
Here is the first paragraph to be typeset by \LaTeX{}.

Notice if I skip a line it starts a new paragraph. Backslash is used to escape special characters
like the dollar sign `\$'.  A backslash is also used to begin a \LaTeX{} command.
For instance: one could write a greek letter $\alpha$.Notice we explicit left and right quotes. Double quotes
are made using pairs of single quote ``''. The dollar
sign is used to start an inline math string. For example,
$(a_i)^{2^2}$. Simple displayed equations can be written by enclosing the equation in `\$\$'. For example,
$$\sum^n_{i=1} 2^i.$$
Notice braces are used to enclose inputs to a \LaTeX{} command. Here's an example of using user defined commands:
\Hastad, \Pudlak.
\section{My second Section}
\ldots{} as Razborv~\cite{razborov95a} said.
\begin{thebibliography}{25}
\bibitem{razborov95a}
A.A. Razborov.
\newblock Lower bounds for propositional proofs and independence results in
  bounded arithmetic.
\newblock In {\em Proceedings of 20th International Symposium on the
  Mathematical Foundations of Computer Science}, page 105. Springer-Verlag,
  1995.
\end{thebibliography}
\end{document}
```

How to get/compile LaTeX

- Links to obtaining LaTeX can be obtained off the class page.
- There are also various GUI front ends which can be used to create a LaTeX file. WinEdt (Windows),TeXShop (for Mac).
- From the command line one can compile a LaTeX document using a command like:

  `latex document.tex` (produces a dvi file)

  `pdflatex document.tex` (produces a pdf file)
- This assumes you have set up the paths to these commands.
- Once compiled you can view the file with a program like yap for dvi files or with acrobat for pdf files.
- One of the links on the class page is to JPicEdt. This allows you to draw pictures and save them as LaTeX files. You can then cut and paste the LaTeX code into the document you submit.

Turing Machines

- We would like a standard formal model for discussing algorithms.
- Having a formal models allows one to carry out proofs on runtimes.
- It turns out that in a sense all such models give a similar notions of tractable/intractable.
- So we will use the most common which is the Turing Machine.

Formal Definitions

- An **alphabet** is a finite set. $\Sigma=\{0,1\}$ or $\Sigma=\{a,b,c,.., z\}$
- A **language** is a set of strings (a finite sequence) over an alphabet. Ex: $L(1^*0^*)$.
- A **Turing Machine** is a 4-tuple $M=(K, \Sigma, \partial, s)$ where K is a finite set of states, s in K is an initial state, Σ is alphabet disjoint from K, and

  $\partial: K \times \Sigma \cup \{\#\} \text{-->} (K \cup \{h, yes, no\}) \times (\Sigma \cup \{L,R\})$
- We imagine M operating on a 1-way infinite tape made up of cells each of which can contain a symbols from the alphabet. The first tape square is required to be the symbol #.
- An input string is written to the cells to the right of the first square.
- The remaining cells to the right are blank, _.
- ∂ says what the machine does in a given state reading a given symbol: it can transition to a new state or to the h, yes, no state, it can write over the current symbol, and it can more left right.
- We require that if a # is being read the machine moves right. So it cannot move off the left hand side of the tape.
- The initial configuration of M is $(s, \underline{\#}x)$.
- A **computation** of M is a sequence of configurations of M such $(s, \underline{\#}x)$:- $(q_1, \underline{w})$:- … :- $(q_m, \underline{w})$ such that each configuration follows from the previous according to M's ∂. Read :- as *yields*.
- A computation halts if any of the states h, yes or no is reach.
- The input is said to be in M's language if it halts in the yes state.