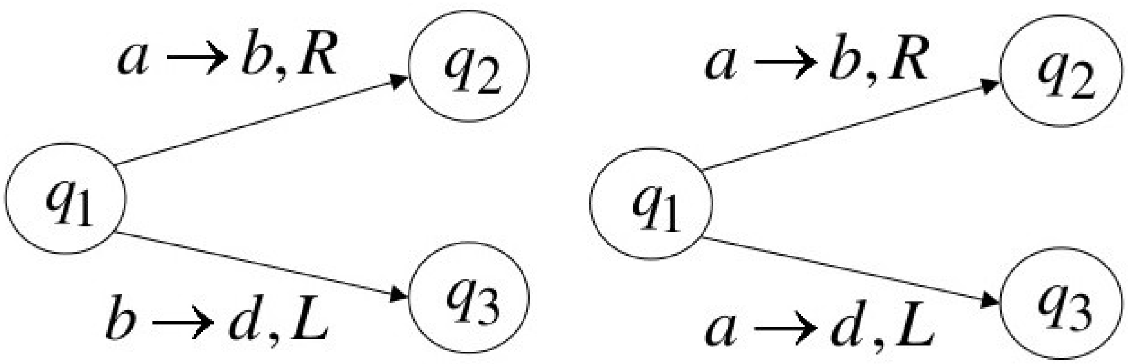
Matthew Martinez
Miguel Martinez
Joseph Reyes
Ernesto Valdez

CSCI 4341: Games, Puzzles, & Computation
Class notes for 1/23/17 & 1/25/17
TEAM C

Turing Machine: A computer with infinite memory. Can be deterministic and nondeterministic.
- Deterministic: A single path.
- Nondeterministic: All paths at the same time.

Ex.) Deterministic Turing Machine vs. Nondeterministic Turing Machine



Notice the T.M. on the left can only follow either path a or path b and the T.M. on the right takes all paths with one choice.

Complexity: Time complexity of an algorithm is the number of steps taken to decide based on the input.

Asymptotic Analysis: Highest term of a polynomial term that describes the running time based on the size of the input.

Notation:
- $\mathbb{N}$ is the set of natural numbers. ex) 0,1,2,3,4... or 1,2,3,4,...
- $\mathbb{R}^+$ is the set of real positive numbers. ex) 0.5, 2, e, 3, π, 4, 5,...
- $\exists$ means "there exists".
- $\forall$ means "for all".
- ε means "an element of".
- s.t. is the abbreviation for "such that".

Def'n: Let f and g be functions f,g: $\mathbb{N} \rightarrow \mathbb{R}^+$. We say that $f(n) \in O(g(n))$ if $\exists$ c, $n_0 >$ s.t. $0 \leq f(n) \leq c * g(n)\ \forall\ n \geq n_0$.

Matthew Martinez
Miguel Martinez
Joseph Reyes
Ernesto Valdez

Ex.1) $f(n) = 5n^3 + 2n^2 + 22n + 6$ $f(n) = O(n^3)$, $g(n) = n^3$

Let $c = 6$ and $n_o = 10$

$f(n) \leq 6n^3 \; \forall \; n \geq 10$

$5{,}428 \leq 6{,}000$

Ex.2) $f(n) = 5n^3 + 2n^2 + 22n + 6 \leq 5n^3 + 2n^3 + 22n^3 + 6n^3$

$f(n) = 35n^3$

$f(n) = O(n^3)$ where $c = 35$ and $n_o = 1$

<u>Little-o Notation</u>:

Def'n: Let f and g be functions $f, g \; \varepsilon \; \mathbb{N} \rightarrow \mathbb{R}^+$. We say that $f(n) \; \varepsilon \; o(g(n)) < \lim_{n\to\infty} f(n) / g(n) = 0$.

Ex.) $f(n) = 3n^3$

$g(n) = n^4$

$f(n) / g(n) = 3n^3 / n^4 = 1 / n \Rightarrow 0$

Def'n: Let $t:\mathbb{N} \rightarrow \mathbb{R}^+$ be a function. Define the time complexity class TIME(t(n)), to be the set of all languages decidable by a o(t(n)) time TM.

Decidable: a solution (yes/no) can always be found in a finite amount of time

Ex.1)

```
for(int i = 0; i < n; i++)
{
     add += arr[i];
}
```

time complexity = O(n)

Ex.2)

```
for(int i = 0; i < n; i++)
{
     for(int j = 0; j < n; j++)
     {
          add += arr[i];
     }
}
```

time complexity = $O(n^2)$

Matthew Martinez
Miguel Martinez
Joseph Reyes
Ernesto Valdez

Polynomial Time: All reasonable deterministic models of computation can be found in a polynomial factor and are polynomially equivalent. This allows us to develop a theory and look at the complexity of problems not specific to a single model of computation.

Complexity Classes:

- P: Decision problems that can be solved on a DTM in polynomial time.
- NP: Decision problems that can be solved on a NTM in polynomial time.

The class P:

Def'n: P is the class of languages decidable in polynomial time on a deterministic single-tape TM.

$$P = \bigcup_k \mathrm{TIME}(n^k)$$

Ex.) PATH = {<*G*, *s*, *t*> : *G* is a directed graph with a path from *s* to *t*}

// insert graphic here
PATH ε $\mathrm{TIME}(n^2)$
PATH ε TIME(nlogn) ε $\mathrm{TIME}(n^2)$ ε …

P ε O(n) <= $O(n^2)$ <= …
P ε O ε (n^2)

The class NP:

Def'n: NTIME(t(n)) = {L: L is a language decided by a O(t(n)) time nondeterministic TM.}

// insert graphic here, input dependent, worst case number of steps

$NP = \bigcup_k \mathrm{NTIME}(n^k)$

*NP is the class of languages that have polynomial time verifiers.

Polynomial Time Verifier:

Def'n: A verifier for a language *A* is an algorithm *V* where *A* = {*w*: *V* accepts <*w*, *c*> for some string *c*}. *c* is the certificate.

Hamiltonian Path:

HP = {<*G*, *s*, *t*> | *G* is a directed graph with a Hamiltonian path from *s* to *t*}

Matthew Martinez
Miguel Martinez
Joseph Reyes
Ernesto Valdez

// insert graphic here

Path = <*s*, *b*, *f*, *e*, *c*, *d*, *g*, *t*>

1. Does this visit all the nodes? // $O(n)$
2. Does this form a valid path? // $O(n^2)$
3. Does the path start at *s* and end at *t*? // $O(1)$

Polynomial and nondeterministic are equivalent.

$$P \subseteq NP$$

Verifying answer is equivalent to U_k NTIME(n^k).