Lecture 1

Yutong Yan

1 The Central Paradigm of Computer Science

- The central ***paradigm*** in computer science is that an algorithm ***A*** is good if:
  - ***A*** runs in ***polynomial time*** in the input size n.
  - That is, ***A*** runs in time T(n) = $O(n^k)$ for some constant number k.
    - · T(n) = 100n + 55
    - · T(n) = $\frac{1}{2}n^2 + 999 \log n$
    - · T(n) = $6n^7 + 900000n^2 - \sqrt{n}$
  - An algorithm is ***bad*** if it runs in exponential time.
    - · T(n) = $2^n + 100n^5$
    - · T(n) = $1.000000001^n - n^3 - n$
  - An algorithm is ***good*** if it runs in ***polynomial time*** in the input size n.

e.g.

		Input Size n		
		10	100	1000
Runtime of Algorithm	n	10	100	1000
	n^2	100	10000	1000000
	2^n	10^3	10^{30}	10^{300}

2 Good versus Bad (Algorithms)

- For example, consider the problem of sorting n numbers.
  - A Good Algorithm: **MergeSort** runs in time $O(n \cdot logn)$
  - A Bad Algorithm: **BruteForce Search** runs in time $O(n \cdot n!) \gg 2^n$

3 An Equivalent Characterization

- This central ***paradigm*** has an equivalent formulation
  - ***A*** runs in ***polynomial time*** in the input size n.
  - The input sizes that ***A*** can solve, in a fixed amount T of time, ***scales multiplicatively*** with increasing computational power.

Input Sizes solved in Time T

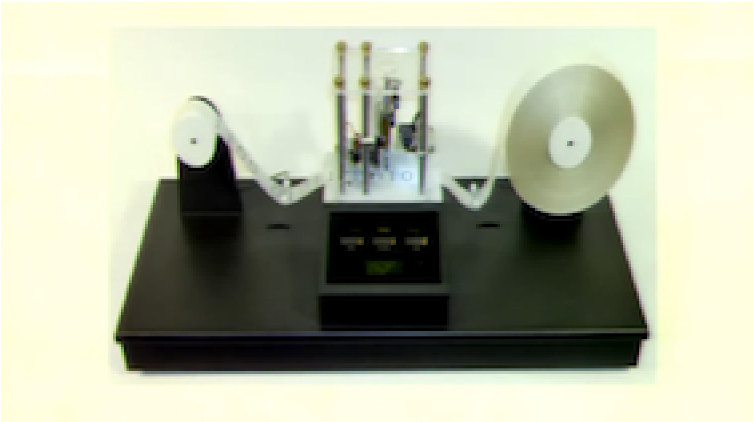
Runtime of Algorithm		Power = 1	Power = 2
	n	T	$2T$
	n^2	$\sqrt{T}$	$\sqrt{2} \cdot \sqrt{T}$
	2^n	$\log T$	$1 + \log T$

  - Moore's Law: Computational power *doubles* roughly every two years.
    → Functional time algorithms will never be able to solve large problems.

- The practical implications are perhaps simpler to understand with this latter formulation.
- Thus, improvements in hardware will *never* overcome ***bad algorithm design***.
- Indeed, the current dramatic breakthroughs in computer science are based upon batter (faster and higher performance) algorithmic techniques.

4 Robustness

- This measure of quality or "goodness" is ***robust***



- All reasonable models of algorithms are polynomial time equivalent.
  - Otherwise one model could perform, say, an exponential number of operations in the time another model took to perform just one.
- The standard formal model is the **Turing Machine**.

5 Cryptography