1 The Role of Algorithms in Computing

1.1 Algorithms

1.1-1

> Give a real-world example that requires sorting or a real-world example that requires computing a convex hull.

- Sorting: browse the price of the restaurants with ascending prices on NTU street.
- Convex hull: computing the diameter of set of points.

1.1-2

> Other than speed, what other measures of efficiency might one use in a real-world setting?

Memory efficiency and coding efficiency.

1.1-3

> Select a data structure that you have seen previously, and discuss its strengths and limitations.

Linked-list:

- Strengths: insertion and deletion.
- Limitations: random access.

1.1-4

> How are the shortest-path and traveling-salesman problems given above similar? How are they different?

- Similar: finding path with shortest distance.
- Different: traveling-salesman has more constraints.

1.1-5

> Come up with a real-world problem in which only the best solution will do. Then come up with one in which a solution that is "approximately" the best is good enough.

- Best: find the GCD of two positive integer numbers.
- Approximately: find the solution of differential equations.

1.2 Algorithms as a technology

1.2-1

> Give an example of an application that requires algorithmic content at the application level, and discuss the function of the algorithms involved.

Drive navigation.

1.2-2

> Suppose we are comparing implementations of insertion sort and merge sort on the same machine. For inputs of size n , insertion sort runs in $8n^2$ steps, while merge sort runs in $64n \lg n$ steps. For which values of n does insertion sort beat merge sort?

$$
\begin{aligned}
8n^2 &< 64n \lg n \\
2^n &< n^8 \\
2 \le n &\le 43.
\end{aligned}
$$

1.2-3

> What is the smallest value of n such that an algorithm whose running time is $100n^2$ runs faster than an algorithm whose running time is 2^n on the same machine?

$$
\begin{aligned}
100n^2 &< 2^n \\
n &\ge 15.
\end{aligned}
$$

Problem 1-1

> For each function $f(n)$ and time t in the following table, determine the largest size n of a problem that can be solved in time t, assuming that the algorithm to solve the problem takes $f(n)$ microseconds.

	1 second	1 minute	1 hour	1 day	1 month	1 year	1 century
$\lg n$	2^{10^6}	$2^{6\times10^7}$	$2^{3.6\times10^9}$	$2^{8.64\times10^{10}}$	$2^{2.59\times10^{12}}$	$2^{3.15\times10^{13}}$	$2^{3.15\times10^{15}}$
$\sqrt{n}$	10^{12}	3.6×10^{15}	1.3×10^{19}	7.46×10^{21}	6.72×10^{24}	9.95×10^{26}	9.95×10^{30}
n	10^6	6×10^7	3.6×10^9	8.64×10^{10}	2.59×10^{12}	3.15×10^{13}	3.15×10^{15}
$n \lg n$	6.24×10^4	2.8×10^6	1.33×10^8	2.76×10^9	7.19×10^{10}	7.98×10^{11}	6.86×10^{13}
n^2	1000	7745	60000	293938	1609968	5615692	56156922
n^3	100	391	1532	4420	13736	31593	146645
2^n	19	25	31	36	41	44	51
$n!$	9	11	12	13	15	16	17