Lesson 13 COMP 2121 ~~Descn~~ Summer 2008

Measuring algorithms.

An engineering approach: ie Real Life.

↳ Efficiency.

↳ currency.

↳ trade this.

eg Java is 3x slower than C.

We trade efficiency for:

- less time writing
- less time testing
- portability

More impt than efficiency:

- correctness
- stable/robust
- readable
- testable
- modular / expandable
- secure
- simplicity
- usability
- maintainable

- efficiency

Dfn: efficient:

- ~~fast~~
- can't be done much faster

Relative

-

Some problems can't be solved at all.
Some problems can't be solved "efficiently"
Some problems can't be solved efficiently
eg 400 000 years to solve it.

---

dfn algorithms

a finite series of well-defined steps / instructions for accomplishing some task which, given an initial state, will terminate in a defined end-state.

Big-Oh notation: Specifies a "category" of efficiency.

Algorithms that $O(n \log n)$ are efficient.
"of the order"

Algorithms that $O(n^7)$ or worse are useless.

Big - Oh only gives an upper bound on an algorithm's efficiency.
ie The Worst Case.

In computer science we usually only care about the worst case.

eg: "The data will be sorted in max 10 minutes"

eg: "Sometimes the data can be maybe possibly be sorted if you're lucky in 3 seconds"

Useful

Useless

a guarantee

What Big - Oh measures: the worst case.

We measure:

1. Time complexity } often a trade-off
2. Space Complexity }

★ Which algorithm is faster ~~[illegible]~~ for an operation depends on the data size, n.

$O(n \log n)$

$O(n^7)$

Example algorithm formula:

$T(n) = 4n^2 - 2n + 2$

\# of items to be algorithmed

let $n = 1$	: $T(n) = 4$		$= 4n^2$
$n = 10$	:	$= 382$	$\approx 4n^2$
$n = 100$	:	$= 39802$	$\approx 4n^2$
$n = 1000$	:	$= 3998002$	$\approx 4n^2$

The bigger that n becomes, the more dominant is the highest exponent term. IGNORE the smaller exponent terms.

In fact we will also ignore the coefficient -s- of the highest-exponent term:

$$T(n) = 4n^2 - 2n + 2 \approx O(n^2)$$

$$P(n) = 3n^2 + 7n + 40000 \approx O(n^2)$$

T(n) and P(n) are the same complexity (aka "same efficiency class") $\rightarrow n^2$

Q/ Which is actually faster?

A/ It depends, on n.

Java's collection classes look at n in ~~deciding~~ deciding which algorithm to use.

For very small n, T(n) < P(n)

As n increases, eventually, T(n) > P(n).

We divide algorithms into "efficiency classes."

eg $O(n^2)$ versus $O(n)$

All $O(n^2)$ algorithms are considered the same but it actually depends on n.

eg Group humans into "financial classes"

- billionaires
- multi millionaire
- millionaire
- upper class
- upper middle class
- middle class
- lower middle class
- lower class
- poverty
- sustenance

Why we measure algorithms:

f(n)	n=10	n=1000	n= 100 000	n= 10 000 000
lg n	3.3 nsec	10 nsec	17 nsec	23 nsec
n	100 nsec	1 μsec	0.1 msec	10 msec
n lg n	33 nsec	10 μsec	1.7 msec	.23 sec
n^2	0.1 μsec	1 msec	10 sec	27.8 min.
n^3	1 μsec	1 sec	11.6 days	31 688 yr
2^n	1 μsec	$3.4 \cdot 10^{284}$ years	$3.2 \cdot 10^{30095}$ yr	$3.1 \cdot 10^{3001022}$ yr

dfn Logarithm: exponent.

Binary logarithm: base 2: lg or ld

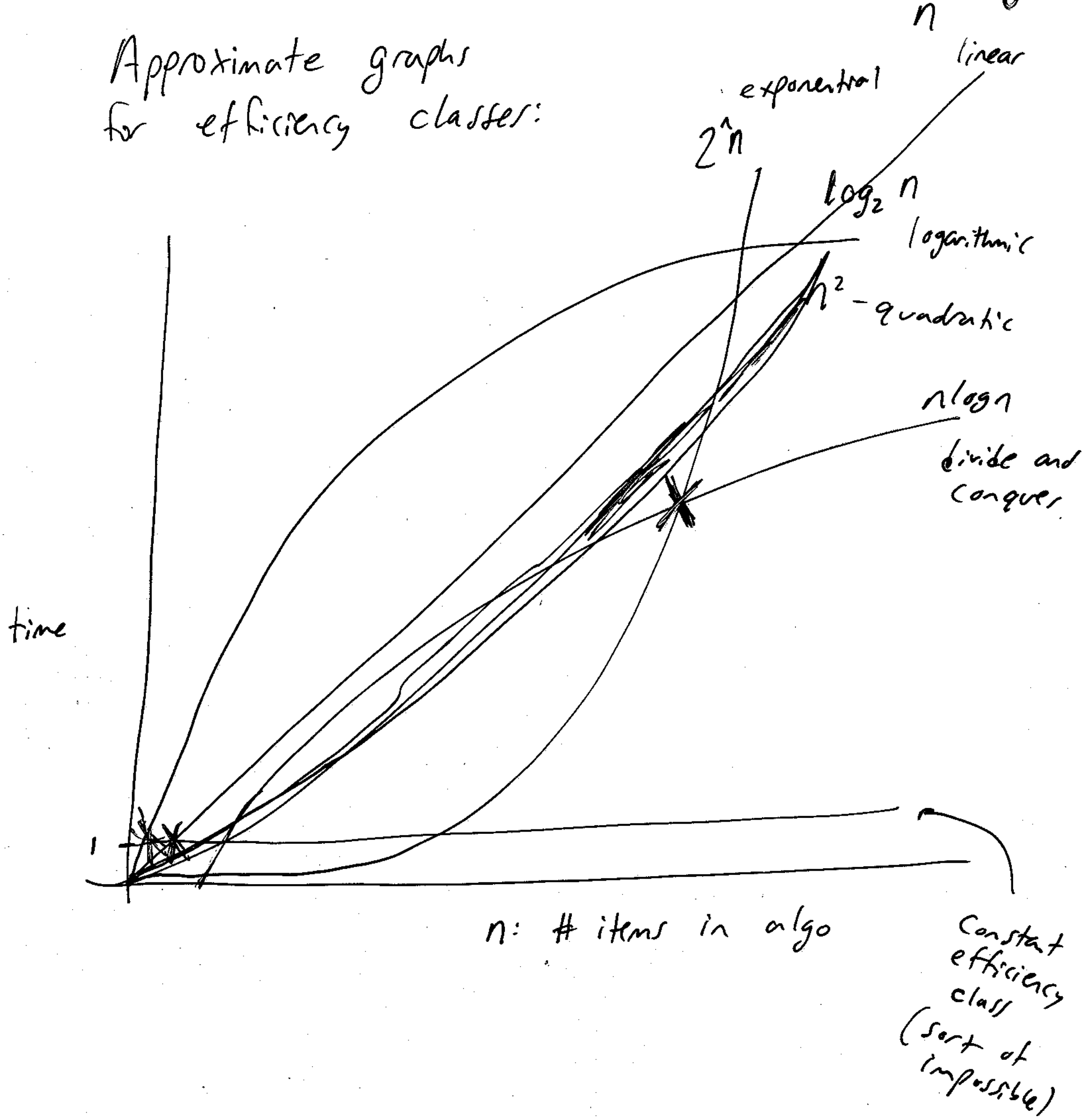
eg divide-and-conquer algorithms use it.

$\lg 32 = 5$

$\lg 1024 = 10$

$\lg 1000 = \approx 10$

Approximate graphs
for efficiency classes:



Basically:

a loop is $O(n)$

```
for {
  ⋮
}
```

a nested loop is $O(n^2)$

```
for {
  for {
    ⋮
  }
}
```

two back-to-back, unnested loops is $O(2n) = O(n)$

```
for {

}
for {

}
```

Pascal's formula + Binomial Theorem

$$\binom{n+1}{r} = \binom{n}{r-1} + \binom{n}{r}$$

Pascal's Triangle:

Row Ø:	1
Row 1:	1 + 1
Row 2:	1 2 1
Row 3:	1 3 3 1
Row 4:	1 4 6 4 1
Row 5:	1 5 10 10 5 1
Row 6:	1 6 15 20 15 6 1
Row 7:	1 7 21 35 35 21 7 1

etc

Pascal's Δ in $\binom{n}{r}$ format:

n \ r	0	1	2	3	4	5	6	7
0	1							
1	1	1						
2	1	2	1					
3	1	3	3	1				
4	1	4	6	4	1			
5	1	5	10	10	5	1		
6	1	6	15	20	15	6	1	
7	1	7	21	35	35	21	7	1

Pascal's formula:

$$\binom{4+1}{3} = \binom{4}{3-1} + \binom{4}{3}$$

$$10 = 6 + 4$$

$$\binom{n+1}{r} = \binom{n}{r-1} + \binom{n}{r}$$

Q What is $\binom{7}{3}$?

A $\binom{7}{3} = \binom{7}{4} = 35$

Binomial Theorem

dfn: binomial: sum of two terms

coefficients

$(a+b)^0 = 1$ 1

$(a+b)^1 = 1a + 1b$ 1 1

$(a+b)^2 = 1a^2 + 2ab + 1b^2$ 1 2 1

$(a+b)^3 = 1a^3 + 3a^2b + 3ab^2 + 1b^3$ 1 3 3 1

$(a+b)^7 = 1a^7 + 7a^6b + 21a^5b^2 + 35a^4b^3 +$
$35a^3b^4 + 21a^2b^5 + 7ab^6 + 1b^7$

$(a-4y)^4 =$

$\boxed{a^4 - 16a^3y + 96a^2y^2 - 256ay^3 + 256y^4}$ //

Let $b = -4y$

The $(a+b)^4 = 1a^4 + 4a^3\underset{\substack{\downarrow\\-4y}}{b} + 6a^2\underset{\substack{\downarrow\\-4y}}{b}{}^2 + 4a\underset{\substack{\downarrow\\-4y}}{b}{}^3 + 1\underset{\substack{\downarrow\\-4y}}{b}{}^4$

$= a^4 + 4a^3(-4y) + 6a^2(-4y)^2 + 4(a)(-4y)^3 + (-4y)$

Quiz Monday: permutations, combinations, today's lesson.
Read 5.8 of text.