CS O(1) today...


Facing recursion's big-O:
recurrence relationships

Spam @ LACMA!

This week: the art and science of designing fast algorithms

Next CS 60 assignment...

Sorting out big-O!

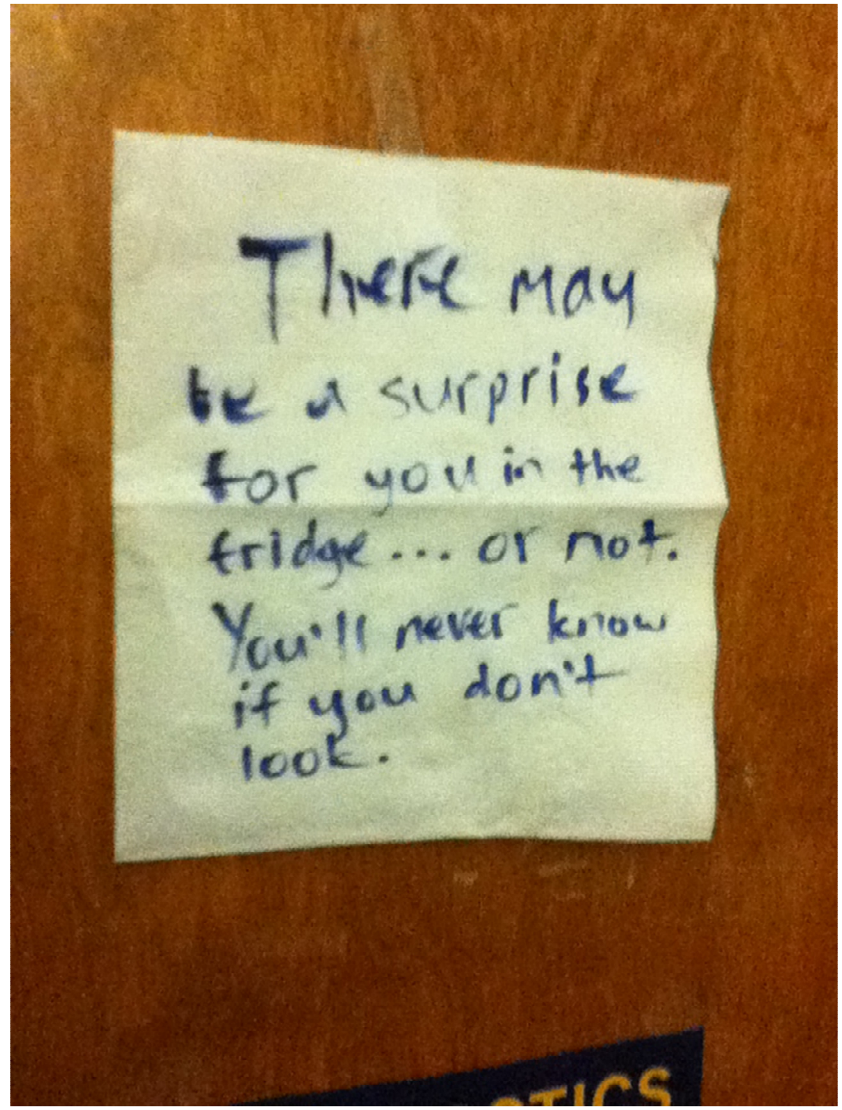
Due ***Tuesday*, 4/15/14** ...

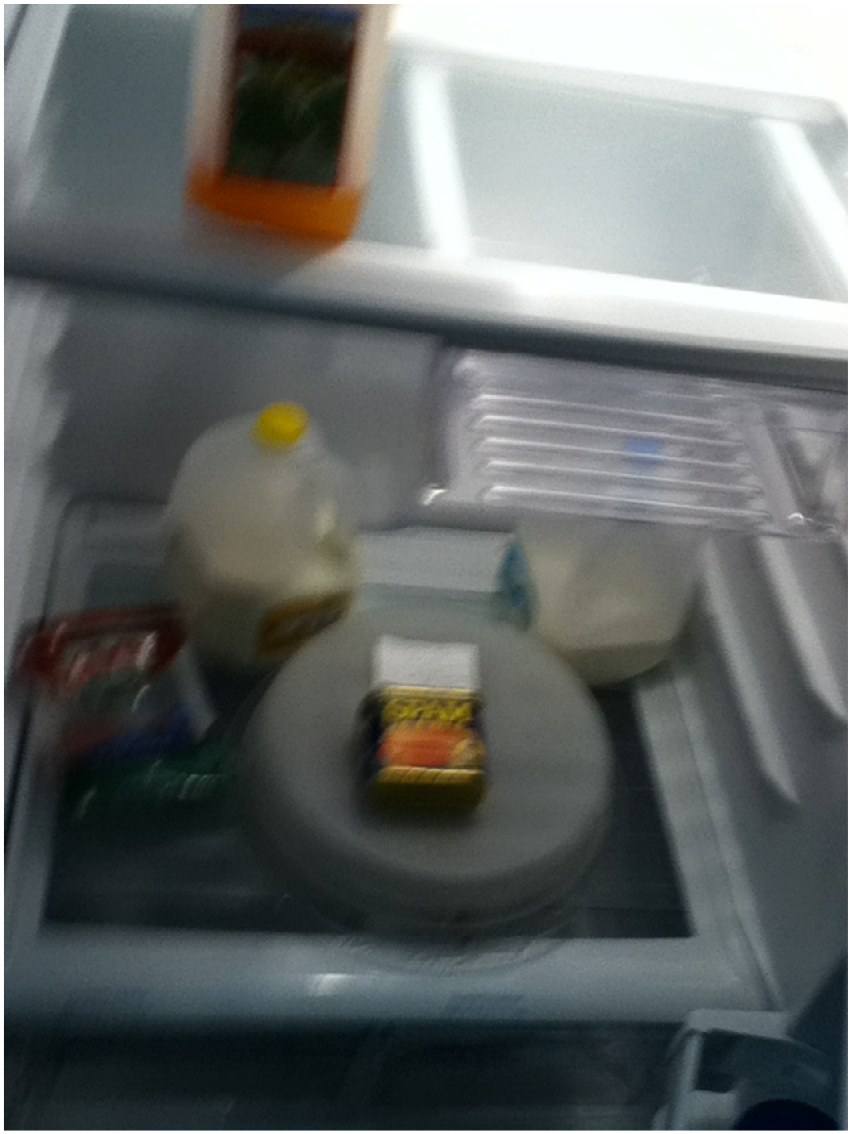
*Do **less!***

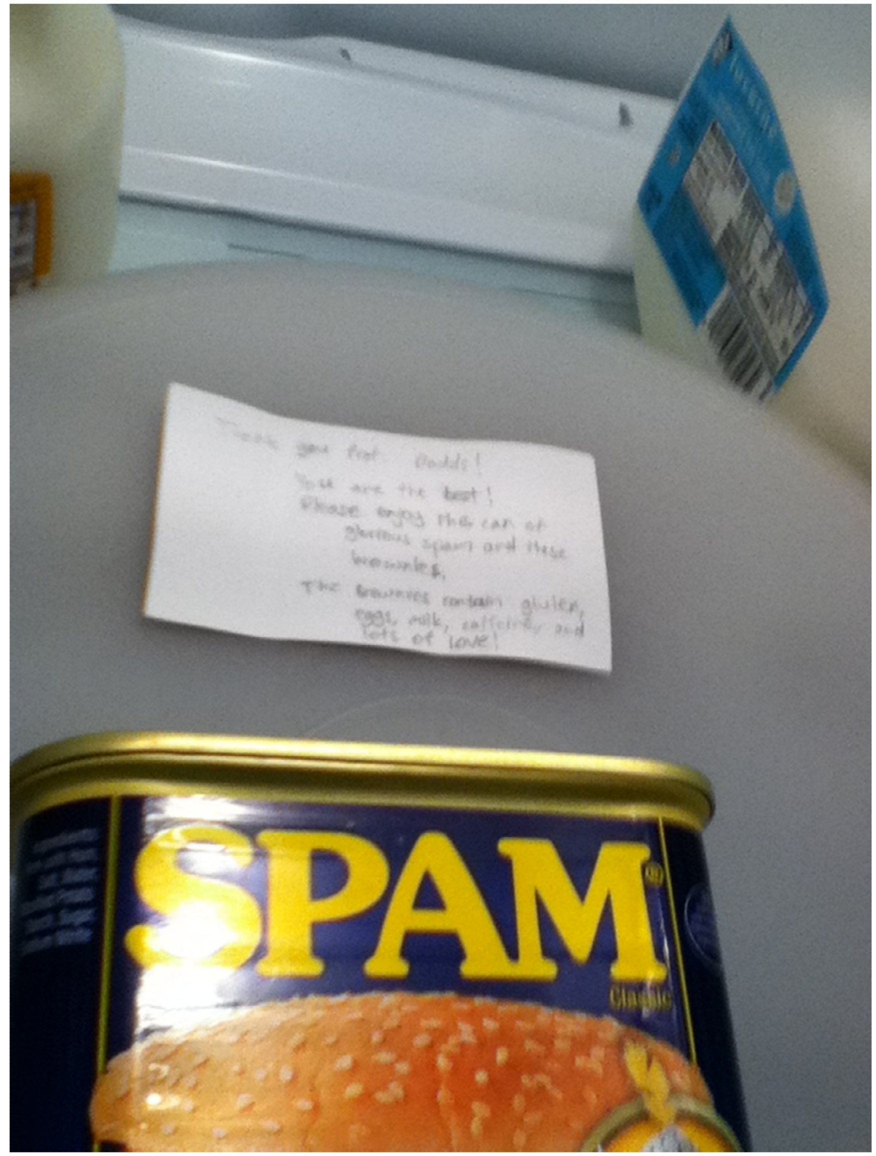
with big-O!

finally some
advice I can use!!

Caught up? grading or other concerns? Let me know...


There may
be a surprise
for you in the
fridge... or not.
You'll never know
if you don't
look.




You are the best!
Please enjoy this can of glorious spam and these brownies.
The brownies contain gluten, eggs, milk, caffeine, and lots of love!
SPAM
Classic

we want the ***fewest*** coins that will produce this total

out of these denominations (we can use as many times as we want)

This week's HW

change(18, [1,5,9,10,12])

use :

lose :

+1

in Java: *dynamic programming!*

Watch out for the INITIAL value for each entry in these tables!

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	2	3	4	1	2	3	4	1	1	2	1	2	2	2
0	1	1	1	1	5	1	1	1	9	10	1	12	1	5	5

Fill in this table from 0 up to the actual amount of interest…

Each top cell has **Min[amt]**, the minimum # of coins that can create **amt**

18

16	17	18
3	2	2
1	5	9

other values possible

Try filling out this table by hand – *this is the DP algorithm!*

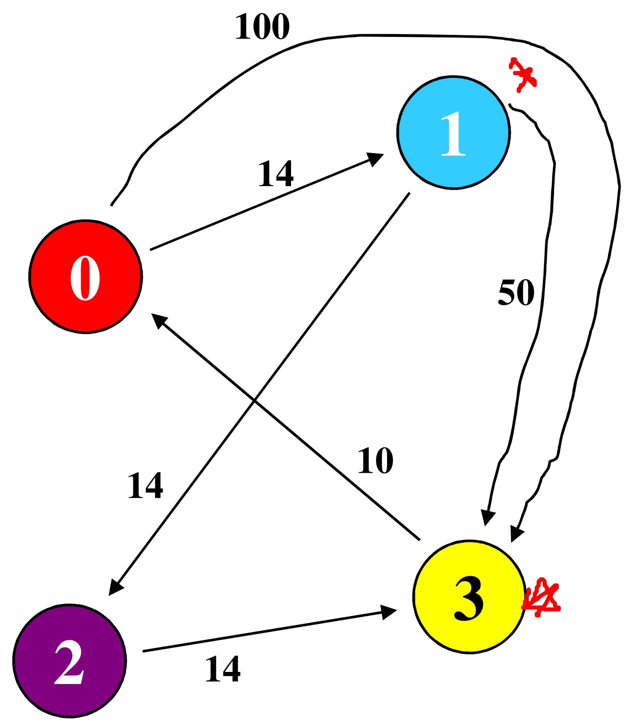
Step 1 check each *src* to each *dst* THROUGH 0

1 entry will change – which?

dst "to"

src "from"	0	1	2	3
0	0	14	inf	100
1	inf	0	14	50
2	inf	inf	0	14
3	10	inf	inf	0

intermediate nodes: 0



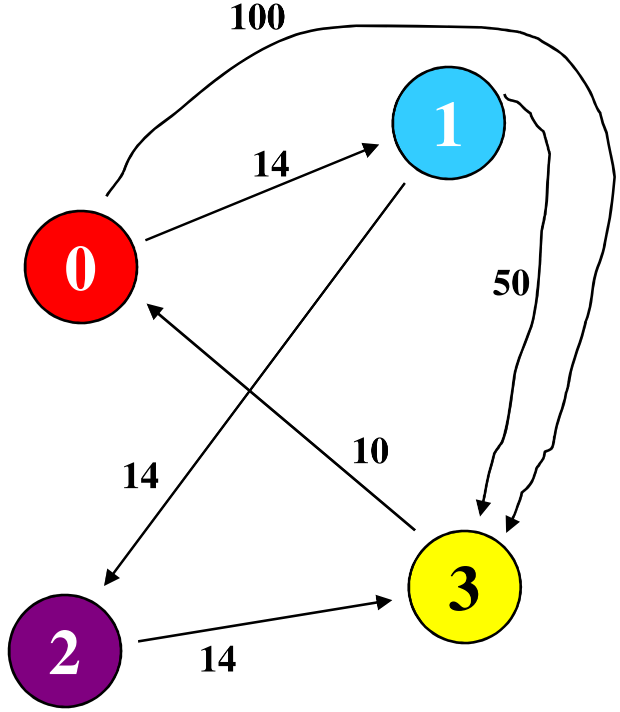
Step 1 check each ***src*** to each ***dst*** THROUGH 0

dst
"to"

src
"from"

src \ dst	0	1	2	3
0	0	14	inf	100
1	inf	0	14	50
2	inf	inf	0	14
3	10	24	inf	0

intermediate nodes: 0



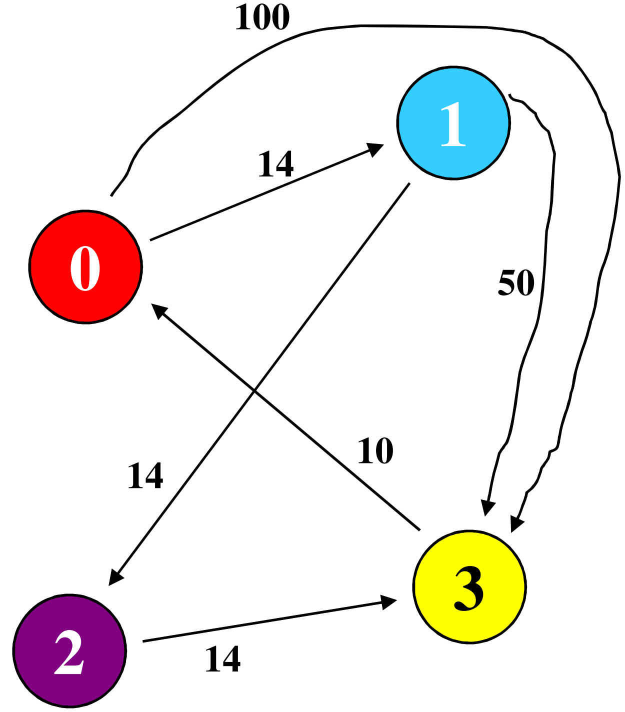
Step 2 check each *src* to each *dst* THROUGH 1

dst
"to"

src
"from"

src \ dst	0	1	2	3
0	0	14	28	64
1	inf	0	14	50
2	inf	inf	0	14
3	10	24	38	0

intermediate nodes: 0 1



Step 3 check each *src* to each *dst* THROUGH 2

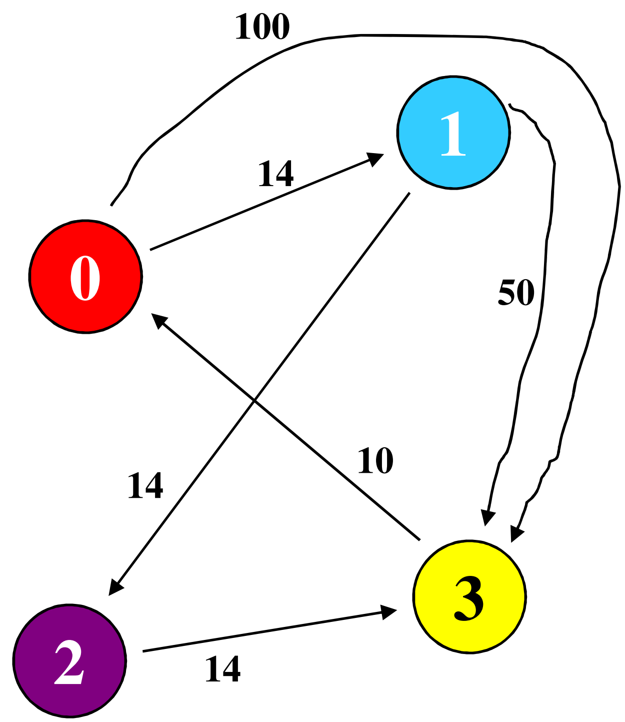
2 entries will change – which?

dst
"to"

src
"from"

	0	1	2	3
0	0	14	28	42
1	inf	0	14	28
2	inf	inf	0	14
3	10	24	38	0

intermediate nodes: 0 1 2



Done!

We now have all of the shortest-distances between all (src, dst) pairs!

What's the big-O runtime?

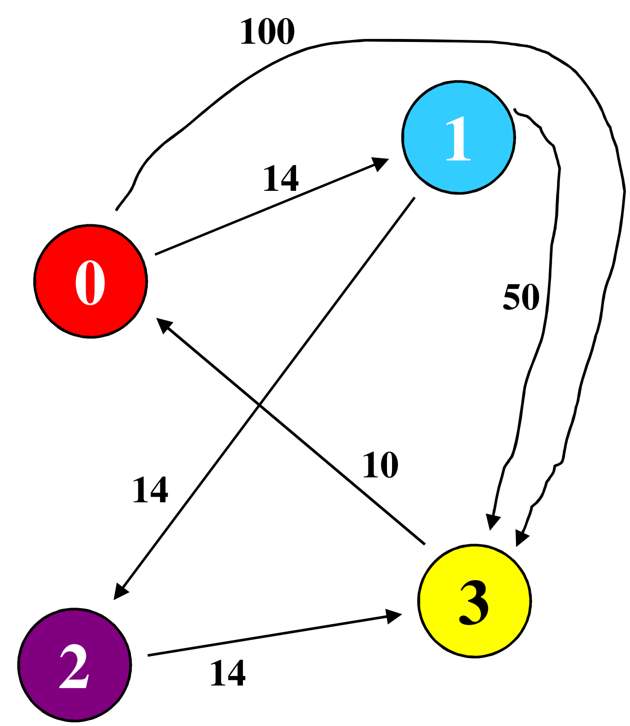
And what about the ***paths***?

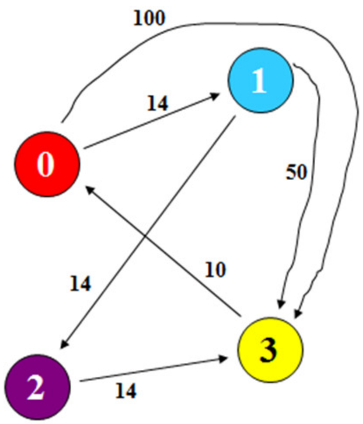
dst
"to"

src
"from"

src \ dst	0	1	2	3
0	0	14	28	42
1	38	0	14	28
2	24	38	0	14
3	10	24	38	0

intermediate nodes: 0 1 2 3





We now have all of the shortest-distances between all (src, dst) pairs!

dst "to"

src "from"

src \ dst	0	1	2	3
0	0	14	28	42
1	38	0	14	28
2	24	38	0	14
3	10	24	38	0

intermediate nodes: 0 1 2 3

What's the big-O runtime?

And what about the ***paths***?

0	0	0	0
1	1	1	1
2	2	2	2
3	3	3	3

How could we keep track of the paths as we go...?!

FW is popular *in practice*…

Kevin Burke to me — show details 10/27/10 — Rep

I think the key takeaway from the toll problem is, *if Floyd Warshall doesn't solve your problem, clearly you didn't run Floyd Warshall enough times.*

I'd like to submit the revised ACM problem solving approach:

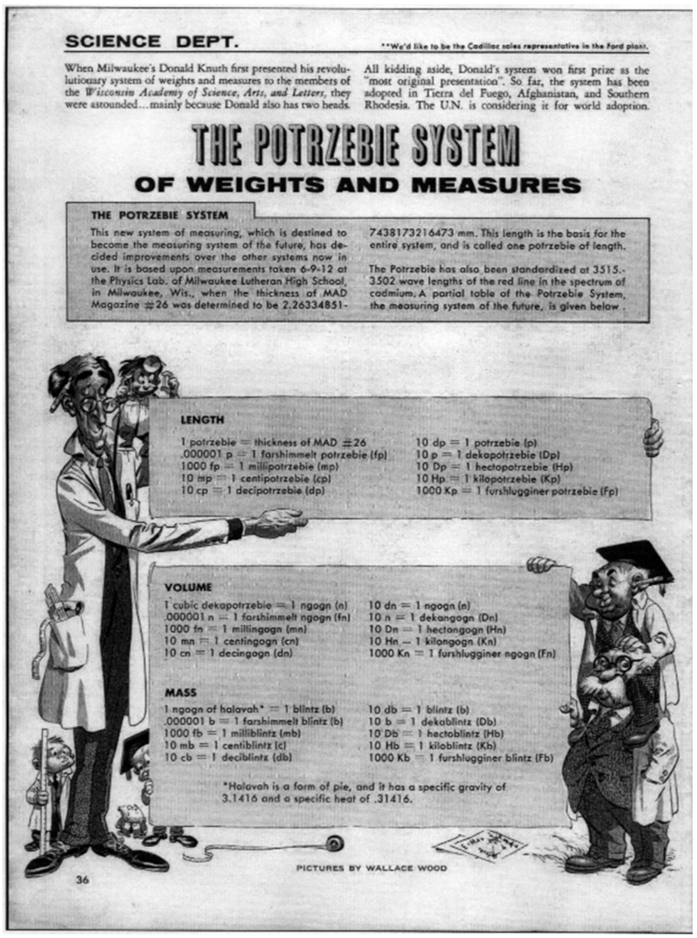
Old
Try Floyd Warshall
If that doesn't work consider other algorithms

New
Try Floyd Warshall
Try running Floyd Warshall n times
If that still doesn't solve your problem then consider other algorithms

Thanks, Kevin!

Publishing success?

Looking to publish that first article? Consider Knuth's example…



SCIENCE DEPT. **We'd like to be the Cadillac sales representative in the Ford plant.

When Milwaukee's Donald Knuth first presented his revolutionary system of weights and measures to the members of the *Wisconsin Academy of Science, Arts, and Letters*, they were astounded…mainly because Donald also has two heads. All kidding aside, Donald's system won first prize as the "most original presentation". So far, the system has been adopted in Tierra del Fuego, Afghanistan, and Southern Rhodesia. The U.N. is considering it for world adoption.

THE POTRZEBIE SYSTEM
OF WEIGHTS AND MEASURES

THE POTRZEBIE SYSTEM

This new system of measuring, which is destined to become the measuring system of the future, has decided improvements over the other systems now in use. It is based upon measurements taken 6-9-12 at the Physics Lab. of Milwaukee Lutheran High School, in Milwaukee, Wis., when the thickness of MAD Magazine #26 was determined to be 2.263348517438173216473 mm. This length is the basis for the entire system, and is called one potrzebie of length.

The Potrzebie has also been standardized at 3515.3502 wave lengths of the red line in the spectrum of cadmium. A partial table of the Potrzebie System, the measuring system of the future, is given below.

LENGTH

1 potrzebie = thickness of MAD #26
.000001 p = 1 farshimmelt potrzebie (fp)
1000 fp = 1 millipotrzebie (mp)
10 mp = 1 centipotrzebie (cp)
10 cp = 1 decipotrzebie (dp)
10 dp = 1 potrzebie (p)
10 p = 1 dekapotrzebie (Dp)
10 Dp = 1 hectopotrzebie (Hp)
10 Hp = 1 kilopotrzebie (Kp)
1000 Kp = 1 furshlugginer potrzebie (Fp)

VOLUME

1 cubic dekapotrzebie = 1 ngogn (n)
.000001 n = 1 farshimmelt ngogn (fn)
1000 fn = 1 millingogn (mn)
10 mn = 1 centingogn (cn)
10 cn = 1 decingogn (dn)
10 dn = 1 ngogn (n)
10 n = 1 dekangogn (Dn)
10 Dn = 1 hectongogn (Hn)
10 Hn — 1 kilongogn (Kn)
1000 Kn = 1 furshlugginer ngogn (Fn)

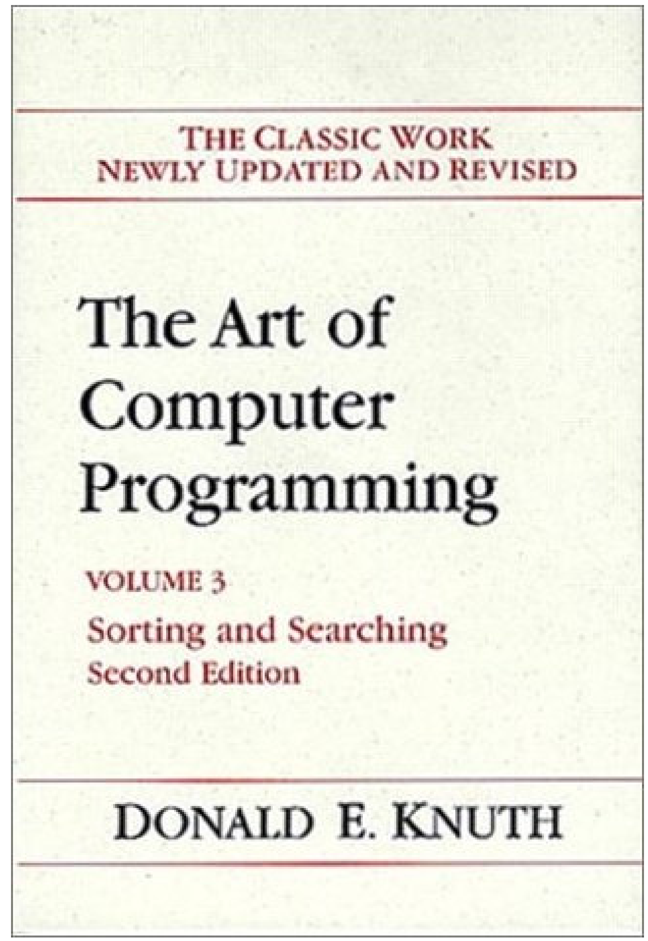
MASS

1 ngogn of halavah* = 1 blintz (b)
.000001 b = 1 farshimmelt blintz (b)
1000 fb = 1 milliblintz (mb)
10 mb = 1 centiblintz (c)
10 cb = 1 deciblintz (db)
10 db = 1 blintz (b)
10 b = 1 dekablintz (Db)
10 Db = 1 hectoblintz (Hb)
10 Hb = 1 kiloblintz (Kb)
1000 Kb = 1 furshlugginer blintz (Fb)

*Halavah is a form of pie, and it has a specific gravity of 3.1416 and a specific heat of .31416.

PICTURES BY WALLACE WOOD

36



**We'd like to be the Cadillac sales representative in the Ford plant.

When Milwaukee's Donald Knuth first presented his revolutionary system of weights and measures to the members of the *Wisconsin Academy of Science, Arts, and Letters*, they were astounded ... mainly because Donald also has two heads. All kidding aside, Donald's system won first prize as the "most original presentation". So far, the system has been adopted in Tierra del Fuego, Afghanistan, and Southern Rhodesia. The U.N. is considering it for world adoption.

THE POTRZEBIE SYSTEM OF WEIGHTS AND MEASURES

THE POTRZEBIE SYSTEM

This new system of measuring, which is destined to become the measuring system of the future, has decided improvements over the other systems now in use. It is based upon measurements taken 6-9-12 at the Physics Lab. of Milwaukee Lutheran High School, in Milwaukee, Wis., when the thickness of MAD Magazine #26 was determined to be 2.263348517438173216473 mm. This length is the basis for the entire system, and is called one potrzebie of length.

The Potrzebie has also been standardized at 3515.3502 wave lengths of the red line in the spectrum of cadmium. A partial table of the Potrzebie System, the measuring system of the future, is given below.

LENGTH

1 potrzebie = thickness of MAD #26
.000001 p = 1 farshimmelt potrzebie (fp)
1000 fp = 1 millipotrzebie (mp)
10 mp = 1 centipotrzebie (cp)
10 cp = 1 decipotrzebie (dp)
10 dp = 1 potrzebie (p)
10 p = 1 dekapotrzebie (Dp)
10 Dp = 1 hectopotrzebie (Hp)
10 Hp = 1 kilopotrzebie (Kp)
1000 Kp = 1 furshlugginer potrzebie (Fp)

VOLUME

1 cubic dekapotrzebie = 1 ngogn (n)
.000001 n = 1 farshimmelt ngogn (fn)
1000 fn = 1 millingogn (mn)
10 mn = 1 centingogn (cn)
10 cn = 1 decingogn (dn)
10 dn = 1 ngogn (n)
10 n = 1 dekangogn (Dn)
10 Dn = 1 hectangogn (Hn)
10 Hn — 1 kilongogn (Kn)
1000 Kn = 1 furshlugginer ngogn (Fn)

MASS

1 ngogn of halavah* = 1 blintz (b)
.000001 b = 1 farshimmelt blintz (fb)
1000 fb = 1 milliblintz (mb)
10 mb = 1 centiblintz (c)
10 cb = 1 deciblintz (db)
10 db = 1 blintz (b)
10 b = 1 dekablintz (Db)
10 Db = 1 hectablintz (Hb)
10 Hb = 1 kiloblintz (Kb)
1000 Kb = 1 furshlugginer blintz (Fb)

*Halavah is a form of pie, and it has a specific gravity of 3.1416 and a specific heat of .31416.

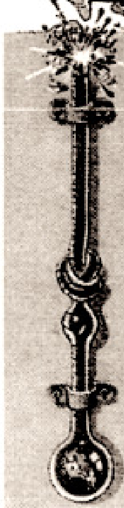
PICTURES BY WALLACE WOOD

TIME

1 average rotation of the earth = 1 clarke (cl)
.00001 cl = 1 wolverton (wl)
1000 wl = 1 kovac (kv)
100 kov = 1 martin (mn)
100 mn = 1 wood (wd)
10 wd = 1 clarke (cl)
10 cl = 1 mingo (mi)
10 mi = 1 cowznofski (cow)

DATE

October 1, 1952 is the day MAD was first published according to the old calendar. On new calendar, this is clarke 1 of cowznofski 1. Cowznofskis before this date are referred to as "Before MAD (B.M.)" and cowznofskis following this date are referred to as "Cowznofsko Madi (C.M.)" The calendar for each cowznofski contains 10 mingoes named as follows: 1. Tales (Tal.) 2. Calculated (Cal.) 3. To (To) 4. Drive (Dri.) 5. You (You) 6. Humor (Hum.) 7. In (In) 8. A (A) 9. Jugular (Jug.) 10. Vein (Vei.)



TEMPERATURE

100° Smurdley (S) — optimum temperature for eating halavah (37°C)

COUNTING

48 things = 1 MAD
49 things = 1 baker's MAD

ANGULAR MEASURE

100 quircits (''') = 1 zorch ('')
100 zorch ('') = 1 zygo (§)
100 zygo (§) = 1 circle or circumference

MISCELLANEOUS MEASUREMENTS

1 light Cow. = 1.14×10^{18} fp.
1 vreeble = 574.8 hah
1 cosmo per sq. potrzebie = 3.1416 bumbles.
1 faraday = 133300 blobs/blintz equivalent weight.
1 electron-ach = 5.5794×10^{-18} hoo
1 blintz molecular wt. of a gas = 77.4 Cn
Density of H_2O = .31823 blintzes per ngogn (distilled water).
1 atmosphere pressure = $1.4533 \frac{h^2}{p}$ = 333.79 p of Hg.
1 b = 3234.4 b-al
gravity = 3234.4 p per kov. per kov.
speed of light = 114440 Fp/Kov.

FORCE (absolute)

a force which, when acting upon 1 blintz of mass for 1 kovac, causes it to attain a velocity of 1 potrzebie per kovac = 1 blintzal (b-al).
1000000 b-al = 1 furshlugginer blintzal (Fb-al)

ENERGY AND WORK

1 blintzal-potrzebie = 1 hoo (h)
1000000 h = 1 hah (hh)

POWER

1 hah per kovac = 1 whatmeworry (WMW)
1000 WMW = 1 kilowhatmeworry (KWMW)
100 KWMW = 1 aeolipower (A.P.)

RADIOACTIVITY

The quantity of radon which is in equilibrium with one blintz of radium = 1 hysa (how's your mom, ed)

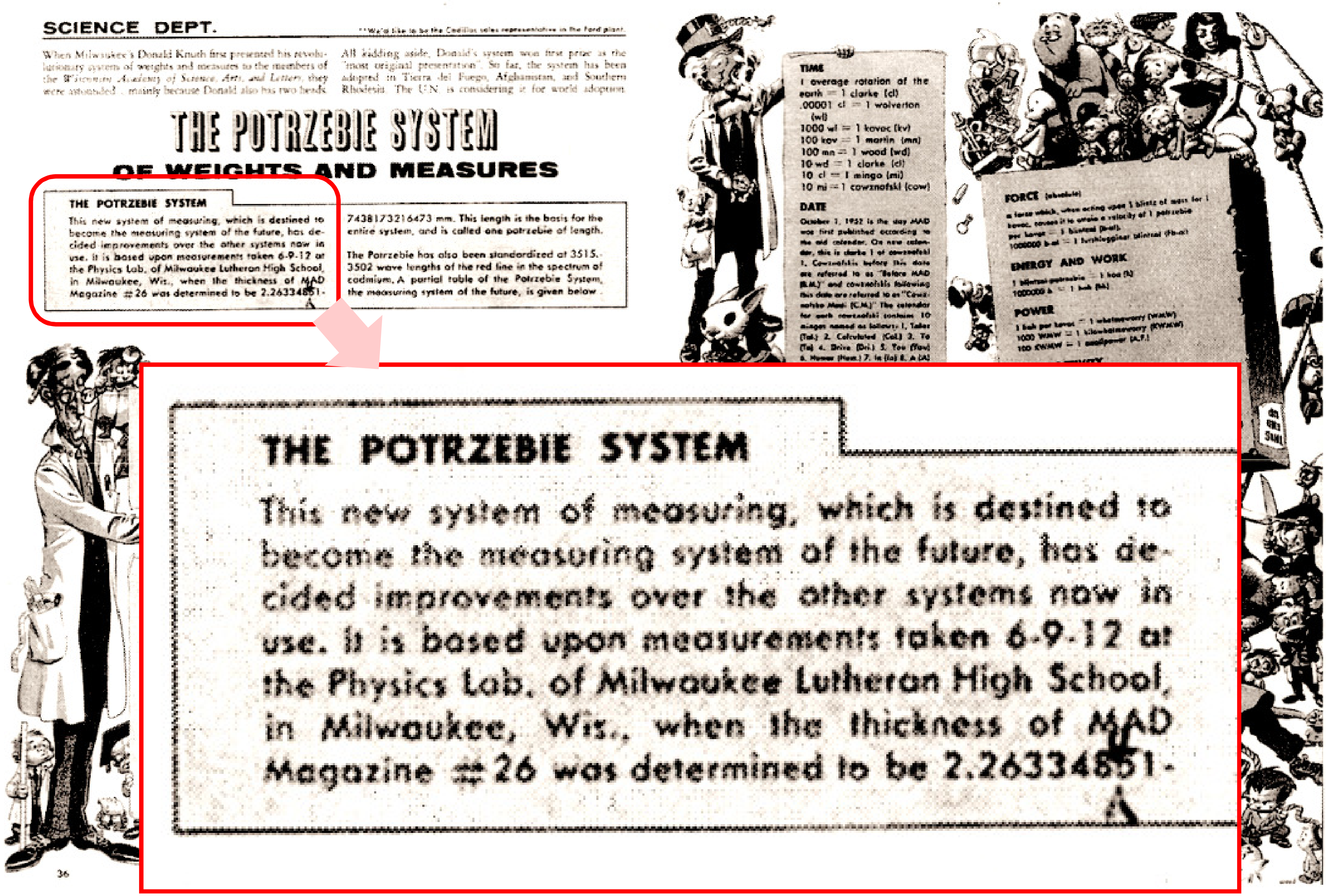
HEAT ENERGY

The amount of heat necessary to raise one blintz of halavah 1° S = 1 vreeble (v.) 1000 v = 1 large vreeble (V)

BASIC CONVERSION FACTORS (ABBREVIATED TABLE)

1 inch = 11.222 potrzebie
1 mile = .71106 Fur-potrzebie
1 blintz = 36.42538631 grams
1 kiloblintz = 80.3048 pounds
1 Fur-blintz = 36.425 metric tons
1 ngogn = 11.59455 cc = .0125 qts. = 2.3523 teaspoons
1 potrzebie = 2.2633 millimeters .089106 inches
1 Furshlugginer potrzebie = 2.2633 Km. 1.4064 miles
1 centimeter = 4.4182 potrzebie
1 kilometer = .44182 Fur-potrzebie
1 wood = 144 minutes
1 kovac = .864 seconds
1 year = 3.6524 Cowznofskis
1 watt = 3.499651 Whatmeworries
1 Horse Power = 2.57 Kilowhatmeworry
1 Aeolipower = 38.797 Horse Power
1 vreeble = 24.330 calories

Don Knuth 11 March 2011



SCIENCE DEPT.

"We'd like to be the Cadillac sales representative in the Ford plant."

When Milwaukee's Donald Knuth first presented his revolutionary system of weights and measures to the members of the *Wisconsin Academy of Science, Arts, and Letters*, they were astounded . . mainly because Donald also has two heads. All kidding aside, Donald's system won first prize as the "most original presentation". So far, the system has been adopted in Tierra del Fuego, Afghanistan, and Southern Rhodesia. The U.N. is considering it for world adoption.

THE POTRZEBIE SYSTEM

OF WEIGHTS AND MEASURES

THE POTRZEBIE SYSTEM

This new system of measuring, which is destined to become the measuring system of the future, has decided improvements over the other systems now in use. It is based upon measurements taken 6-9-12 at the Physics Lab. of Milwaukee Lutheran High School, in Milwaukee, Wis., when the thickness of MAD Magazine #26 was determined to be 2.263348517438173216473 mm. This length is the basis for the entire system, and is called one potrzebie of length.

The Potrzebie has also been standardized at 3515.3502 wave lengths of the red line in the spectrum of cadmium. A partial table of the Potrzebie System, the measuring system of the future, is given below.

TIME

1 average rotation of the earth = 1 clarke (cl)
.00001 cl = 1 wolverton (wl)
1000 wl = 1 kovac (kv)
100 kov = 1 martin (mn)
100 mn = 1 wood (wd)
10 wd = 1 clarke (cl)
10 cl = 1 mingo (mi)
10 mi = 1 cowznofski (cow)

DATE

October 1, 1952 is the day MAD was first published according to the old calendar. On new calendar, this is clarke 1 of cowznofski 1. Cowznofskis before this date are referred to as "Before MAD (B.M.)" and cowznofskis following this date are referred to as "Cowznofsko Madi (C.M.)" The calendar for each cowznofski contains 10 mingos named as follows: 1. Tales (Tal.) 2. Calculated (Cal.) 3. To (To) 4. Drive (Dri.) 5. You (You) 6. Humor (Hum.) 7. In (In) 8. A (A)

FORCE (absolute)

a force which, when acting upon 1 blintz of mass for 1 kovac, causes it to attain a velocity of 1 potrzebie per kovac = 1 [illegible]
1000000 [illegible] = 1 [illegible]

ENERGY AND WORK

1 [illegible] = 1 hoo (h)
1000000 h = 1 hah (hh)

POWER

1 hah per kovac = 1 [illegible]
1000 [illegible] = 1 [illegible]
100 [illegible] = 1 [illegible]

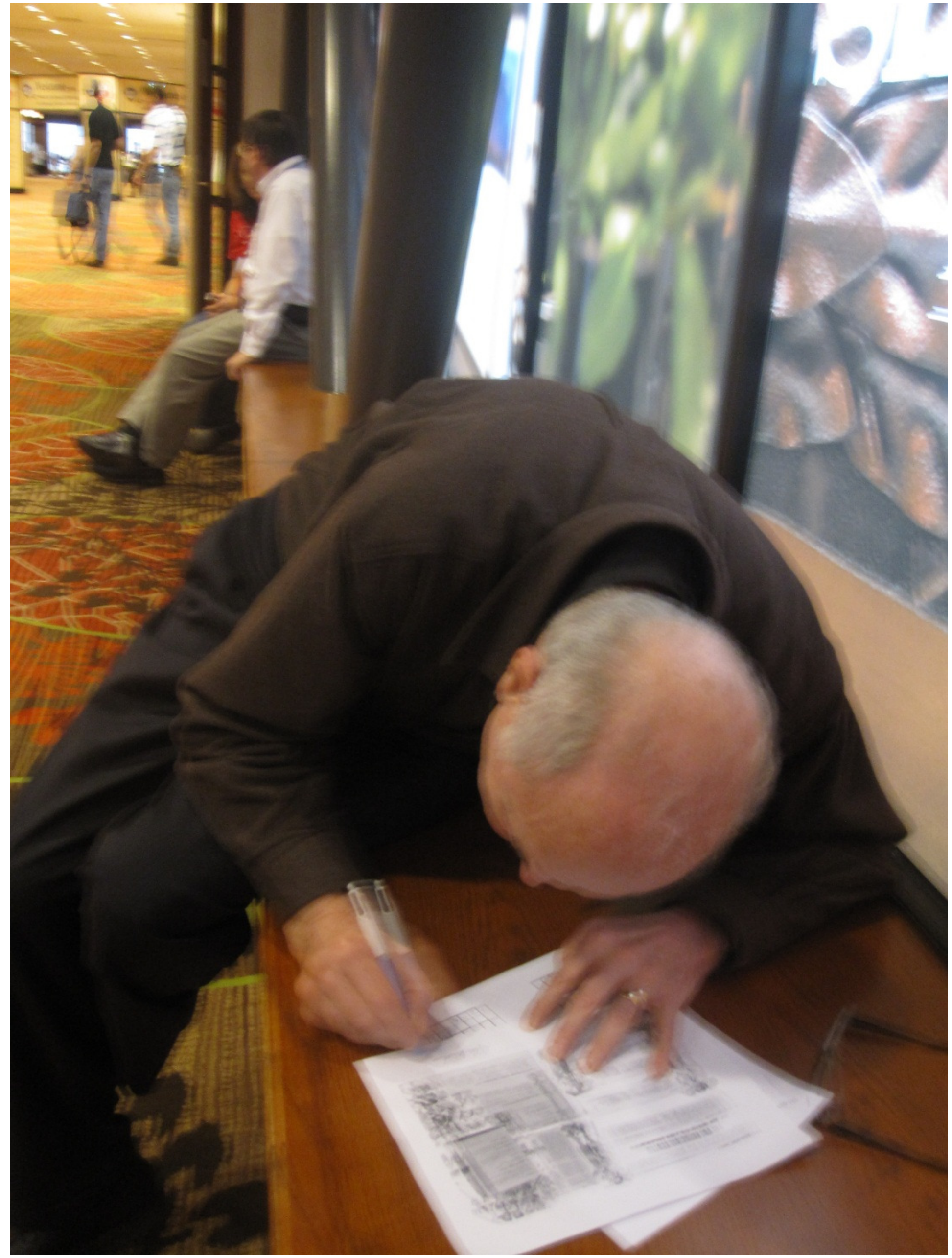
36

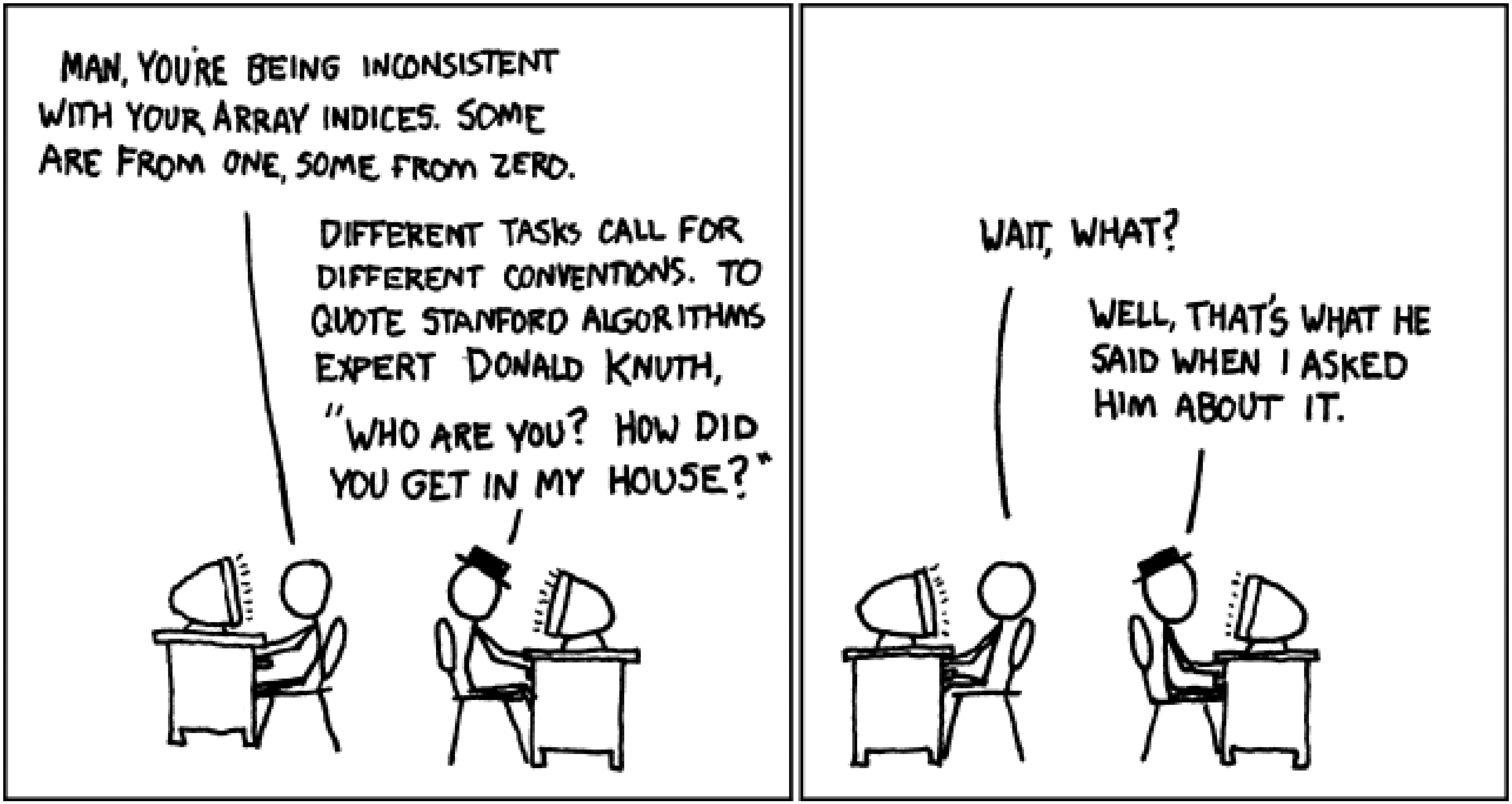
THE POTRZEBIE SYSTEM

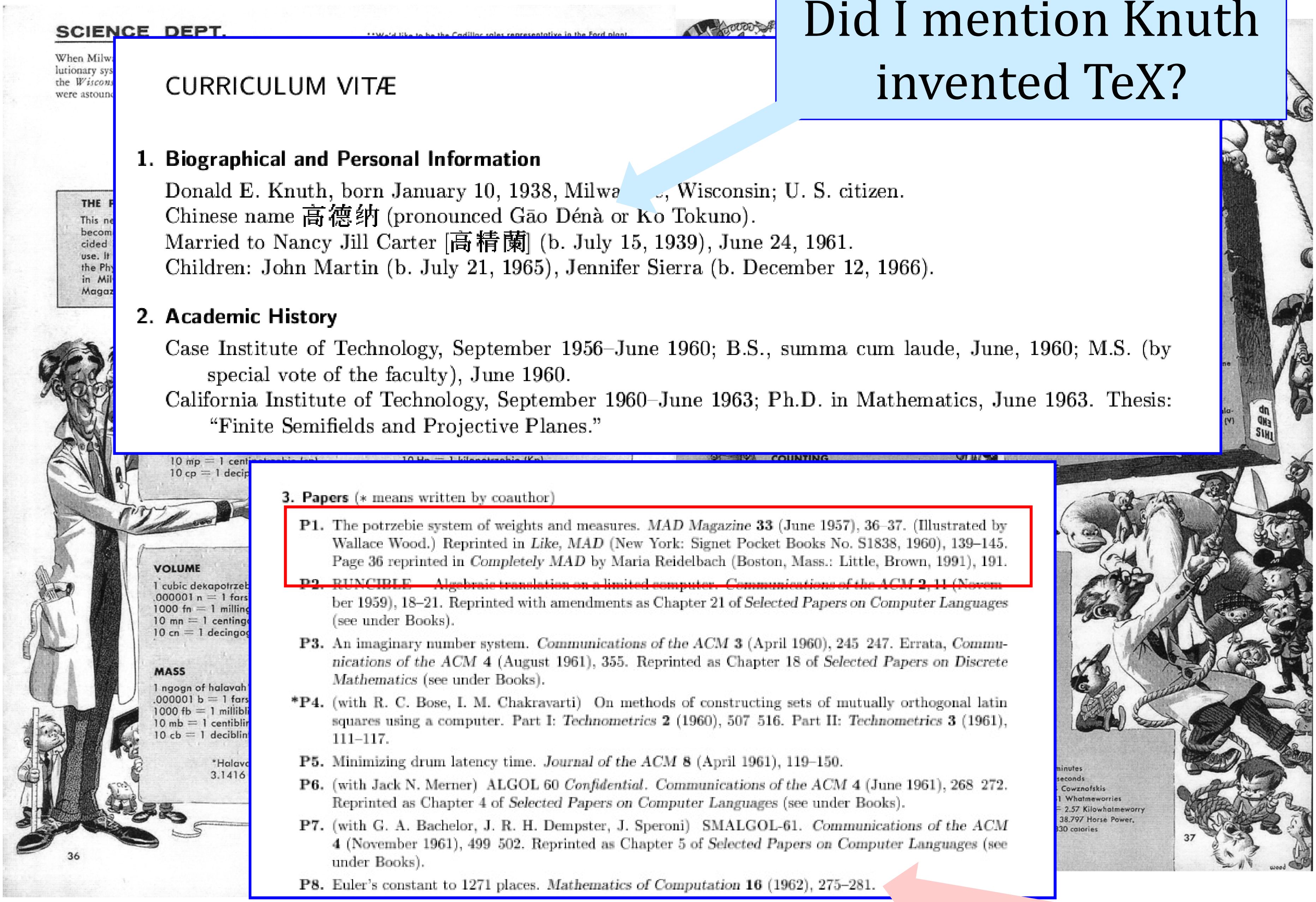
This new system of measuring, which is destined to become the measuring system of the future, has decided improvements over the other systems now in use. It is based upon measurements taken 6-9-12 at the Physics Lab. of Milwaukee Lutheran High School, in Milwaukee, Wis., when the thickness of MAD Magazine #26 was determined to be 2.26334851-

Don Knuth 11 March 2011

… not *necessarily*
stalking Don Knuth




MAN, YOU'RE BEING INCONSISTENT WITH YOUR ARRAY INDICES. SOME ARE FROM ONE, SOME FROM ZERO.
DIFFERENT TASKS CALL FOR DIFFERENT CONVENTIONS. TO QUOTE STANFORD ALGORITHMS EXPERT DONALD KNUTH, "WHO ARE YOU? HOW DID YOU GET IN MY HOUSE?"
WAIT, WHAT?
WELL, THAT'S WHAT HE SAID WHEN I ASKED HIM ABOUT IT.



Did I mention Knuth invented TeX?

CURRICULUM VITÆ

1. Biographical and Personal Information

Donald E. Knuth, born January 10, 1938, Milwaukee, Wisconsin; U. S. citizen.
Chinese name 高德纳 (pronounced Gāo Dénà or Ko Tokuno).
Married to Nancy Jill Carter [高精蘭] (b. July 15, 1939), June 24, 1961.
Children: John Martin (b. July 21, 1965), Jennifer Sierra (b. December 12, 1966).

2. Academic History

Case Institute of Technology, September 1956–June 1960; B.S., summa cum laude, June, 1960; M.S. (by special vote of the faculty), June 1960.

California Institute of Technology, September 1960–June 1963; Ph.D. in Mathematics, June 1963. Thesis: "Finite Semifields and Projective Planes."

3. Papers (∗ means written by coauthor)

P1. The potrzebie system of weights and measures. *MAD Magazine* **33** (June 1957), 36–37. (Illustrated by Wallace Wood.) Reprinted in *Like, MAD* (New York: Signet Pocket Books No. S1838, 1960), 139–145. Page 36 reprinted in *Completely MAD* by Maria Reidelbach (Boston, Mass.: Little, Brown, 1991), 191.

P2. RUNCIBLE — Algebraic translation on a limited computer. *Communications of the ACM* **2**, 11 (November 1959), 18–21. Reprinted with amendments as Chapter 21 of *Selected Papers on Computer Languages* (see under Books).

P3. An imaginary number system. *Communications of the ACM* **3** (April 1960), 245–247. Errata, *Communications of the ACM* **4** (August 1961), 355. Reprinted as Chapter 18 of *Selected Papers on Discrete Mathematics* (see under Books).

∗P4. (with R. C. Bose, I. M. Chakravarti) On methods of constructing sets of mutually orthogonal latin squares using a computer. Part I: *Technometrics* **2** (1960), 507–516. Part II: *Technometrics* **3** (1961), 111–117.

P5. Minimizing drum latency time. *Journal of the ACM* **8** (April 1961), 119–150.

P6. (with Jack N. Merner) ALGOL 60 *Confidential*. *Communications of the ACM* **4** (June 1961), 268–272. Reprinted as Chapter 4 of *Selected Papers on Computer Languages* (see under Books).

P7. (with G. A. Bachelor, J. R. H. Dempster, J. Speroni) SMALGOL-61. *Communications of the ACM* **4** (November 1961), 499–502. Reprinted as Chapter 5 of *Selected Papers on Computer Languages* (see under Books).

P8. Euler's constant to 1271 places. *Mathematics of Computation* **16** (1962), 275–281.

Time matters…

But *counting all steps* is too precise.

New machines make code run faster...

*… **but not more efficiently*** !

To capture the efficiency of an ***algorithm***…

*… or the difficulty of a **problem**!*

Big-O

Big **O** definition

I thought it stood for **O**ppressing constants!

$f \in O(g)$ means

$$(\exists c)\ (\exists N)\ (\forall n > N)\ \ f(n) < c*g(n)$$

there exists a coefficient c

and a threshold N

such that, for all input sizes bigger than that threshold, N

g is bigger than f

up to a constant factor

To claim that $42n^2 + 100n \in O(n^2)$

We need to find c and N such that for all n > N, $42n^2 + 100n < cn^2$

c =

N =

Big **O** in practice

f $\in$ O(g) means

g is bigger than f

up to a constant factor

Big O in practice

Ignore everything but the dominant term

even then, ignore the constant

$$420n^2 + 3n^3 - 201$$

$$17 + 300\sqrt{n} + 2\log(n)$$

$$\frac{117}{n} + \frac{2^n}{3^n} + 1700$$

Big-O is formally an upper bound, but in general we want the **BEST** upper bound…

big-O hierarchy

Name(s): ________________________

function

N^3

$N!$

$\sqrt{N}$

$\log(N)$

N^N

N

2^N

42

N^2

$N \cdot \log(N)$

running times…

Sort by asymptotic size (O)

largest? next? smallest?

Match to algorithms

A. finding **x** in an N-element balanced binary search tree?

B. finding the **rest** of a length-N list

C. finding **x** in an length-N list

D. closest pair in N unsorted #s

E. tautology checking with N vars

F. is N prime?

G. NxN matrix multiplication

H. shortest tour of N cities

I. sorting a list of N elements

J. finding the median of a list

CS's challenge…

Problem-solving strategy:

Push a problem as far down the hierarchy as possible, <u>then</u> worry about constants... .

Big-**O** Order

n^n
$n!$
2^n
Intractable Problems (*Exponential*)

n^3
n^2
$n \log(n)$
n
Tractable Problems (*Polynomial*)

$\sqrt{n}$
$\log(n)$
1
No Problems!

Running Times

(or why people worry about algorithm complexity)

problem size →

complexity	n = 10	100	1,000	10,000	100,000	1,000,000
log n	3.3219	6.6438	9.9658	13.287	16.609	19.931
$\log^2 n$	10.361	44.140	99.317	176.54	275.85	397.24
sqrt n	3.162	10	31.622	100	316.22	1000
n	**10**	**100**	**1000**	**10000**	**100000**	**1000000**
n log n	33.219	664.38	9965.8	132877	$1.66*10^6$	$1.99*10^7$
$n^{1.5}$	31.6	10^3	$31.6*10^4$	10^6	$31.6*10^7$	10^9
n^2	100	10^4	10^6	10^8	10^{10}	10^{12}
n^3	1000	10^6	10^9	10^{12}	10^{15}	10^{18}
2^n	1024	10^{30}	10^{301}	10^{3010}	10^{30103}	10^{301030}

Polynomial

Exp.

O

factorial? don't even ask!

imagine the time units ~ nanoseconds: 10**(-9)

So, I have an algorithm…

What's the big-O runtime?

Mudd Math Fun Facts

Series, big-O style!

big-O

How many terms are here? What's the big-O sum?

$$1 + 2 + 4 + 8 + \ldots + N/4 + N/2 + N$$

What's the big-O sum of these values?

$$1 + 2 + 4 + 8 + \ldots + 2^{N-2} + 2^{N-1} + 2^{N}$$

What's the big-O sum of these values?

$$1 + 2 + 3 + 4 + \ldots + (N-2) + (N-1) + N$$

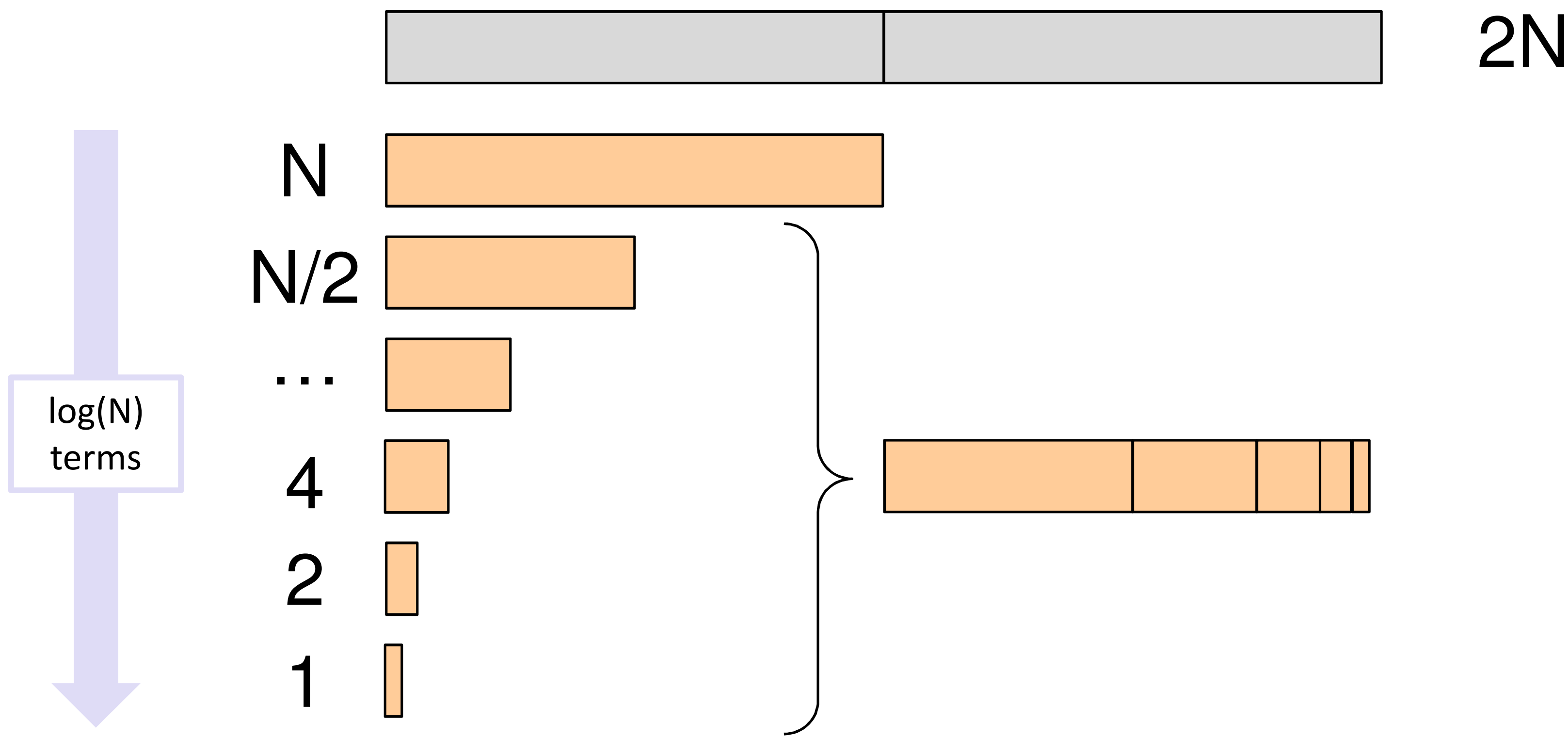
Proofs!

(by CS profs)

Mudd Math Fun Facts

How many terms are here? What's the big-O sum?

$$1 + 2 + 4 + 8 + \ldots + N/4 + N/2 + N$$



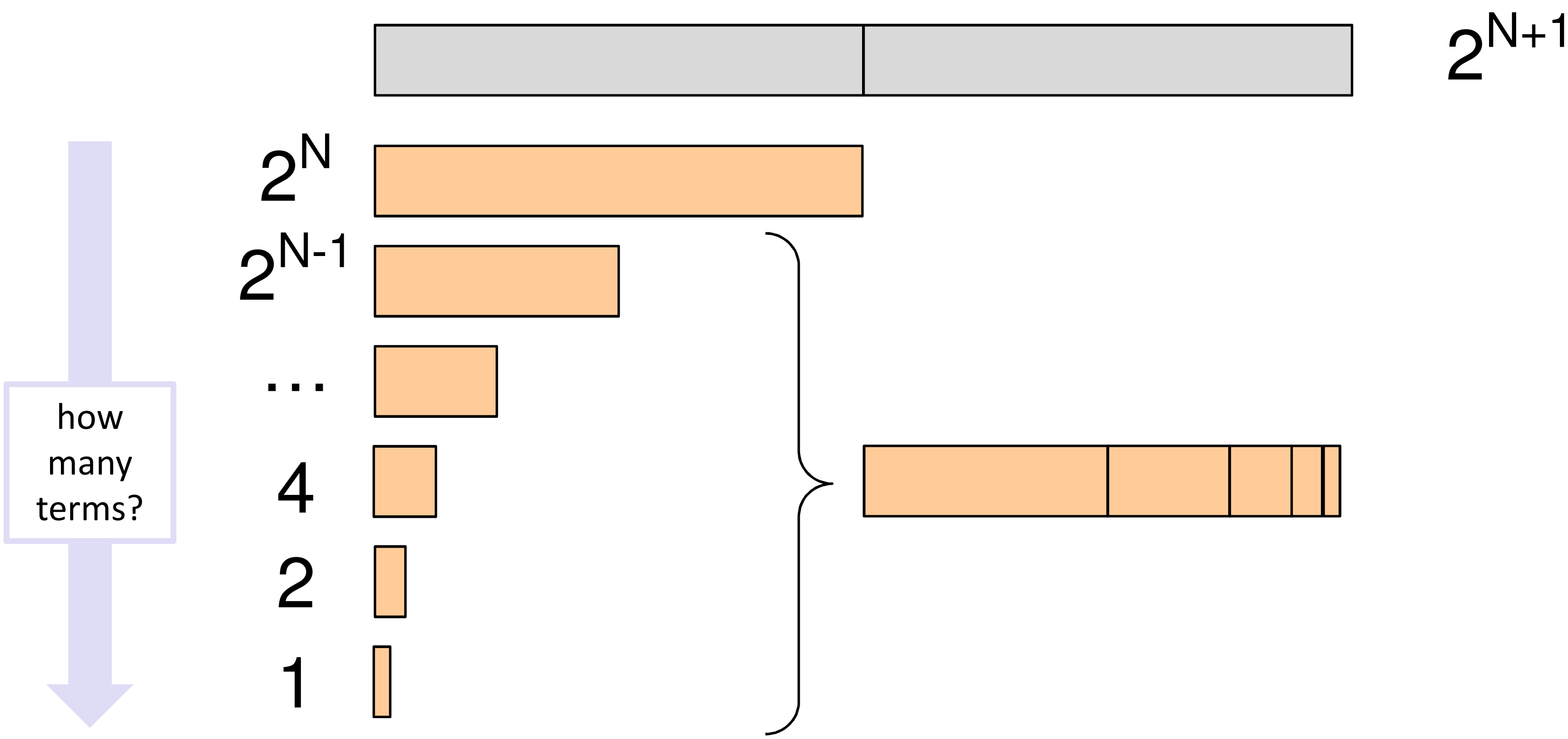
Proofs!

(by CS profs)

Mudd Math Fun Facts

How many terms are here? What's the big-O sum?

$$1 + 2 + 4 + 8 + \ldots + 2^{N-2} + 2^{N-1} + 2^{N}$$



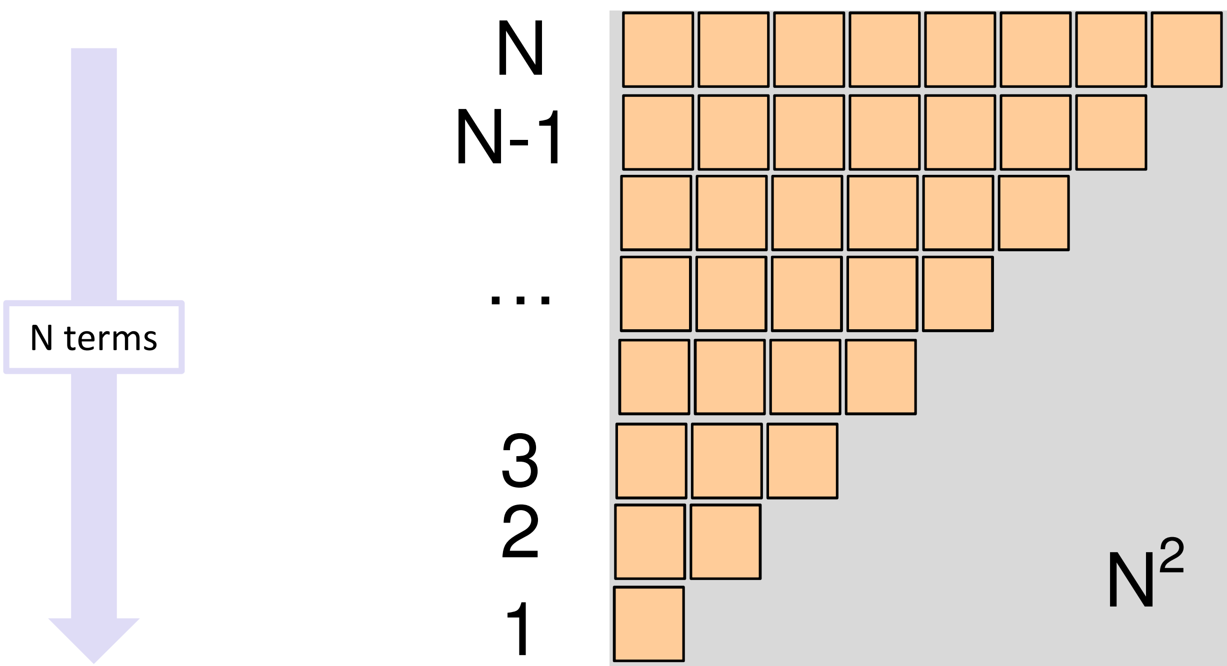
Proofs!

(by CS profs)

Mudd Math Fun Facts

How many terms are here? What's the big-O sum?

$$1 + 2 + 3 + 4 + \ldots + (N-2) + (N-1) + N$$



Loop Counting

What's the big-O running time of each of these loops?

```
for ( int i=0 ; i<N ; ++i )
    1 step of work here…
```

```
for ( int i=N ; i>0 ; i/=2 )
    1 step of work here…
```

Track the Outer Loop index

OL	IL
0	1
1	1
2	1
$\ldots i \ldots$	$\ldots 1 \ldots$
$N-2$	1
$N-1$	$+\ 1$

Sum up the work Inside the Loop – ***using big-O…***

OL	IL

i see you're up to something…

Loop Counting

What's the big-O running time of each of these loops?

```
for ( int i=0 ; i<N ; ++i )
  for ( int j=0 ; j<N ; ++j )
    1 time step of work here…
```

```
for ( int i=1 ; i<=N ; ++i )
  for ( int j=1 ; j<=i ; ++j )
    1 time step of work here…
```

Track the Outer Loop index

OL	IL
0	*N*
1	*N*
2	*N*
…i…	*…N…*
N-2	*N*
N-1	*N*
	+

Sum up the work Inside the Loop – ***using big-O…***

OL	IL

Loop Counting

```
for ( int i=1 ; i<=N ; ++i )
  for ( int j=1 ; j<=i ; ++j )
    1 time step of work here…
```

In general, create a table of work done…

Track the **O**uter **L**oop index

OL	IL	Compute the **I**nner **L**oop's work
i=1	1	j ~ 1 to 1
i=2	2	j ~ 1 to 2
…	i	j ~ 1 to i
i=N	N	j ~ 1 to N

Complicated nested loops may require additional thought for each of these rows…!

$+$

$O(N^2)$ ← Ask Prof. Su for the sum of the series…

What about *recursive* code?

Recurrence Relations

```
def sum( L ):
    if L == []: return 0
    else: return L[0] + sum( L[1:] )
```

counting additions

$T(0) = 0$ (additions)

$T(N) =$

recurrence relation

T(N) is the running time of **sum**, with input size N

Recurrence Relations

```
def sum( L ):
    if L == []: return 0
    else: return L[0] + sum( L[1:] )
```

counting additions

$$T(0) = 0$$

$$T(N) = 1 + T(N-1)$$

recurrence relation

T(N) is the running time of **`sum`** with input size N

To solve: *"Unwind" the relation until all of the terms are known.*

Recurrence *Unwinding*

```
def g( L ):
    if L == []: return 1
    h = len(L)/2
    return sum(L), g(L[:h]), g(L[h:])
```

g, this seems weird...

still counting additions !

$T(0) = 0$

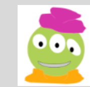
$T(N) =$

T(N) is now the running time of **g**, with input size N

Recurrence *Unwinding*

```
def g( L ):
    if L == []: return 1
    h = len(L)/2
    return sum(L), g(L[:h]), g(L[h:])
```

g, this seems weird…



still counting additions !

$$T(0) = 0$$

$$T(N) = N + 2T(N/2)$$

T(N) is now the running time of **g**, with input size N

Recurrence *Unwinding*

$T(0) = 0$

$T(N) = N + 2T(N/2)$

$N + 2(\qquad\qquad\qquad)$

$T(A) =$

T() =

$T(N/2) =$

Recurrence *Unwinding*

$T(0) = 0$

$T(N) = N + 2T(N/2)$

$N + 2(N/2 + 2*T(N/4))$

$T(A) = A + 2T(A/2)$

T() =  + 2T(/2)

$T(N/2) = (N/2 + 2T(N/4))$

Recurrence *Unwinding*

$T(0) = 0$

$T(N) = N + 2T(N/2)$

$N + 2(\ N/2 + 2*T(N/4)\ )$

Recurrence *Unwinding*

$$T(0) = 0$$

$$T(N) = N + 2T(N/2)$$

$$N + 2(\ N/2 + 2^*T(N/4)\ )$$

$$N + \underbrace{2(\ N/2}_{N} + \underbrace{2^*(\ N/4}_{N} + \underbrace{2^*T(N/8)}_{N}\ )\ )$$

$$N + N + N + N + \ldots$$

log(N) times!

O(N log(N))

T(N) is the running time of **g**, with input size N

Try it!

Finding big-O running times of looping and recursive code…

These are similar in spirit to problem #1 on the homework…

1

```
for (int i=1 ; i<=N ; i*=2)
  for (int j=0 ; j<i ; j++ )
    // O(1) work here…
```

Values of i (**OL**)	j work (**IL**)
i=1	
i=2	
i=4	
i	
i=N	

2

$T(1) = 0$

$T(N) = 2N + T(N/2)$

3

count each move as 1 operation

```
hanoi(1,A,B) => move disk from A to B
hanoi(N,A,B) => C is the "other peg",
                hanoi(N-1,A,C),
                hanoi(1,A,B),
                hanoi(N-1,C,B)
```

T(1) =

T(N) =

Write and solve the hanoi recurrence relation, T(N)

4

```
for (int i=1 ; i<=N ; i*=2)
  for (int j=1 ; j<i ; j*=2)
    // O(1) work here…
```

Values of i (**OL**)	j work (**IL**)

Try it!

Finding big-O running times of looping and recursive code…

These are similar in spirit to problem #1 on the homework…

1

```
for (int i=1 ; i<=N ; i*=2)
  for (int j=0 ; j<i ; j++ )
    // O(1) work here…
```

OL Values of i	IL j runs
i=1	1 time
i=2	2 times
i=4	4 times
i	i times
i=N	+ N times
	= 2N - 1 work

sum this column

O(N)

3

count each move as 1 operation

```
hanoi(1,A,B) => move disk from A to B
hanoi(N,A,B) => C is the "other peg",
                hanoi(N-1,A,C),
                hanoi(1,A,B),
                hanoi(N-1,C,B)
```

$T(1) = 1$ (one move for one disk)

$T(N) = 1 + 2T(N-1)$ (1 base case + 2 big recursions)

$$= 1 + 2(\,1 + 2T(N-2)\,)$$
$$= 1 + 2(\,1 + 2(\,1 + 2T(N-3)\,)\,)$$
$$= 1 + 2(\,1 + 2(\,1 + 2(\,\dots\ 2(\,1 + 2T(1)\,)\dots)\,)\,)$$
$$= 1 + 2 + 4 + 8 + \dots + 2^{N-1}$$
$$= 2^{N-1} - 1$$

$O(2^N)$

2

$T(1) = 0$

$T(N) = 2N + T(N/2)$

$$= 2N + \overbrace{2(N/2) + T(N/4)}$$
$$= 2N + 2(N/2) + \overbrace{2(N/4) + T(N/8)}$$
$$= 2N + 2(N/2) + 2(N/4) + \dots + 2(2) + T(1)$$
$$= 2N + N + N/2 + \dots + 4 + 0$$
$$= 4N - 4$$

O(N)

4

```
for (int i=1 ; i<=N ; i*=2)
  for (int j=1 ; j<i ; j*=2)
    // O(1) work here…
```

OL Values of i	IL j runs
i = 1	0 times
i = 2	1 times
i = 4	2 times
…	…
i = N	+ log(N) times
	= log(N)(log(N)+1) / 2

sum up this column

$O(\log^2(N))$

Strategies for fast-algorithm design…

Brute Force -- always consider it first!

How valuable is computer time?

Divide and Conquer

Divide the problem recursively and then reassemble the pieces

Use data structures

Remembering previous function calls

Look-up tables of partial results

Donald Knuth?

www-cs-faculty.stanford.edu/~knuth

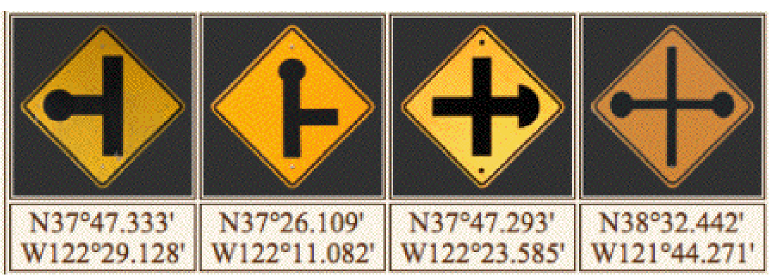
Slightly eccentric perfectionist + Fan of road signs =

Diamond Signs

During our summer vacation last year, my wife and I amused ourselves by taking leisurely drives in Ohio and photographing every diamond-shaped highway sign that we saw along the roadsides. (Well, not *every* sign; only the distinct ones.) For provenance, I also stood at the base of each sign and measured its GPS coordinates.

This turned out to be even more fun than a scavenger hunt, so we filled in some gaps when we returned to California. And we intend to keep adding to this collection as we drive further, although we realize that we may have to venture to New England in order to see `FROST HEAVES'.

Here are the images of our collection so far.



my favorites...

Donald Knuth?

www-cs-faculty.stanford.edu/~knuth

Slightly eccentric perfectionist + Fan of road signs =

Diamond Signs

During our summer vacation last year, my wife and I amused ourselves by taking leisurely drives in Ohio and photographing every diamond-shaped highway sign that we saw along the roadsides. (Well, not *every* sign; only the distinct ones.) For provenance, I also stood at the base of each sign and measured its GPS coordinates.

This turned out to be even more fun than a scavenger hunt, so we filled in some gaps when we returned to California. And we intend to keep adding to this collection as we drive further, although we realize that we may have to venture to New England in order to see `FROST HEAVES'.

Here are the images of our collection so far.

Donald Knuth?

www-cs-faculty.stanford.edu/~knuth

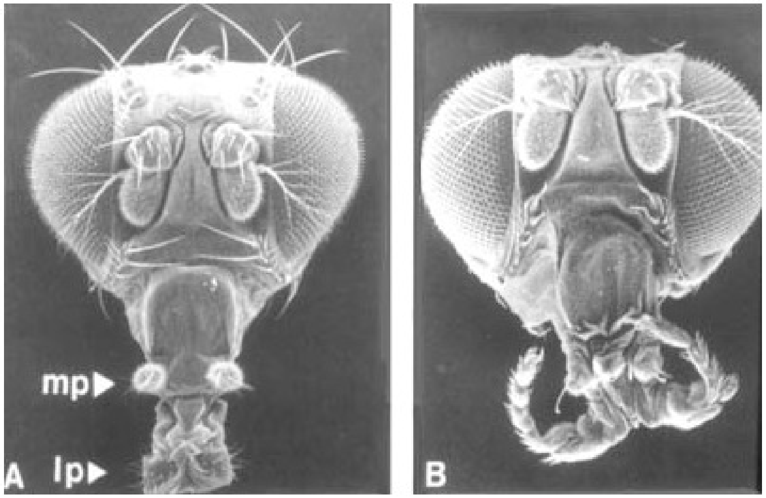
Slightly eccentric perfectionist + Fan of road signs =

Diamond Signs

During our summer vacation last year, my wife and I amused ourselves by taking leisurely drives in Ohio and photographing every diamond-shaped highway sign that we saw along the roadsides. (Well, not *every* sign; only the distinct ones.) For provenance, I also stood at the base of each sign and measured its GPS coordinates.

This turned out to be even more fun than a scavenger hunt, so we filled in some gaps when we returned to California. And we intend to keep adding to this collection as we drive further, although we realize that we may have to venture to New England in order to see `FROST HEAVES'.

Here are the images of our collection so far.



This past journal illustration by T.C. Kaufman and colleagues from Indiana University shows how when the 'pb' (proboscipedia) gene in the fruit fly – the fruit fly's homologue or equivalent gene to the Hoxa2 gene being studied by Nazarali, Doucette and Wolf – is mutated, it causes a transformation of the fruit fly's mouth parts (slide A) into legs (slide B).

Some fruit flies are less equal than others…

Sorting out big-O ! ⟶

Sorting is the *drosophila* of computer science…

Sorting Algorithms

the *fruit flies* of complexity theory...

PermutationSort

PrologSort

check all permutations until sorted

List of length n

3	1	2

<u>Check</u>:

3 1 2

3 2 1

1 2 3

1 3 2

2 1 3

2 3 1

BogoSort !

shuffle numbers

check if sorted

if not, repeat….

List of length n

3	1	2

running time ??

Sorting Algorithms

the *fruit flies* of complexity theory...

PermutationSort

PrologSort

check all permutations until sorted

List of length n

3	1	2

Check:

3 1 2
3 2 1
1 2 3
1 3 2
2 1 3
2 3 1

BogoSort !

shuffle numbers
check if sorted
if not, repeat….

List of length n

3	1	2

running time ??

Slow:

1 1 1
1 1 2
1 1 3
1 2 1
1 2 2
1 2 3
1 3 1
1 3 2
1 3 3
2 1 1
2 1 2
2 1 3
2 2 1
2 2 2
2 2 3
2 3 1
2 3 2
2 3 3
3 1 1
3 1 2
3 1 3
3 2 1
3 2 2
3 2 3
3 3 1
3 3 2
3 3 3

Polynomial sorting

```
def cs5sort(L):
  if len(L) < 2: return L

  if L[0] == min(L):
    return [L[0]] + cs5sort(L[1:])

  return cs5sort(L[1:] + L[0])
```

cs5 sort

My favorite sorting algorithm!

big-O ?

MinSort + InsertionSort

keep finding the min…

4	7	5	2	1	3	0	6

***worst*-case?**

***best*-case?**

keep inserting elements…

4	7	5	2	1	3	0	6

better than $O(N^2)$?

Strategies for algorithm design

Brute Force -- always consider it first!

How valuable is computer time?

Divide and Conquer

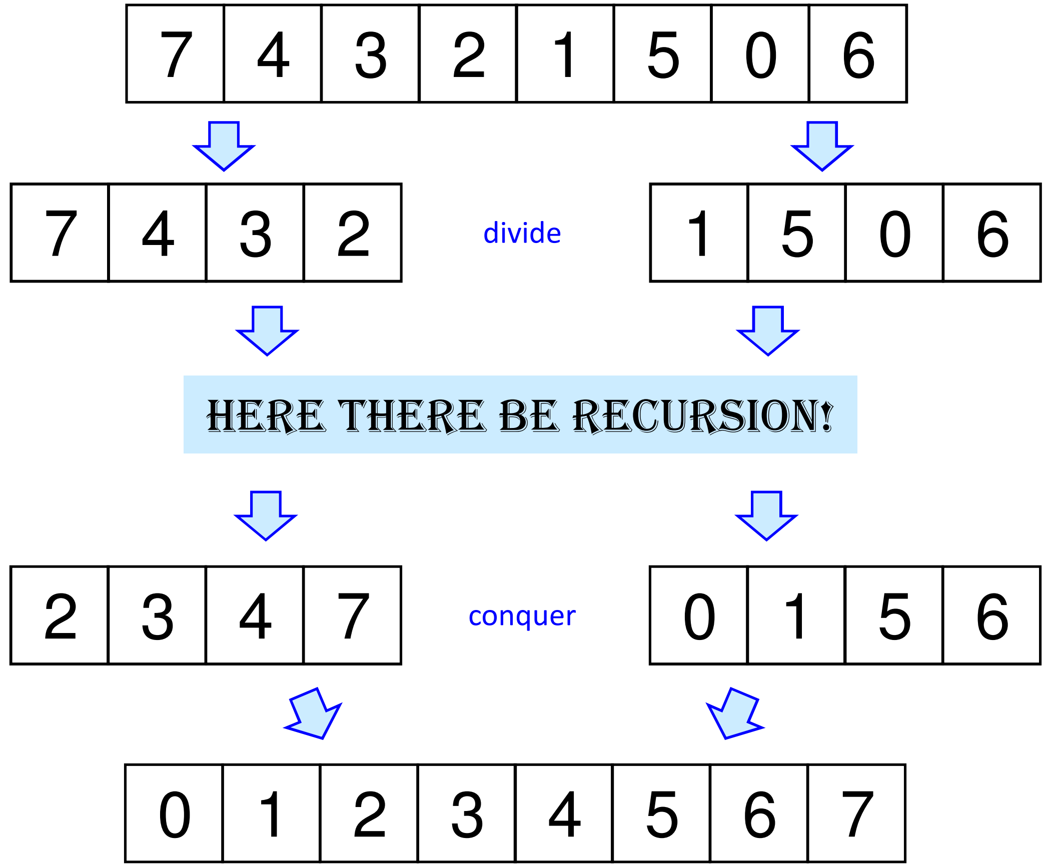
Divide the problem recursively and then reassemble the pieces

Use data structures

Remembering previous function calls

Look-up tables of partial results

MergeSort



Recurrence relation?

even better than O(Nlog(N))?

Can we sort faster than Nlog(N) ?

<u>Order</u>

n^n
$n!$
2^n } Intractable Problems (Exponential)

n^3
n^2
$n \log(n)$ } Tractable Problems (Polynomial)

?

n

$\sqrt{n}$
$\log(n)$
1 } No Problems!

Problem-solving strategy:

Push a problem as far down the hierarchy as possible, <u>then</u> worry about constants... .

Can we sort faster than Nlog(N) ?

<u>Order</u>

n^n

$n!$

2^n

Intractable Problem

n^3

n^2

$n \log(n)$

?

n

$\sqrt{n}$

$\log(n)$

1

No Problems!

No! *and we can prove it...*

and we will on Thursday...