SHELLSORT (Survey Talk)

Shellsort and Shellsort networks

Ancient results

Old results

Average-case analysis

Variants of Shellsort

New ideas

Shellsort

```
shellsort(itemType a[], int l, int r)
 {
   int incs[16] =
     { 1391376, 463792, 198768, 86961, 33936,
        13776, 4592, 1968, 861, 336, 112, 48,
        21, 7, 3, 1 };
   int i, j, h, v;

   for ( k = 0; k < 16; k++)
     for (h = incs[k], i = l+h; i <= r; i++)
       {
         v = a[i]; j = i;
         while (j > h && a[j-h] > v)
           { a[j] = a[j-h]; j -= h; }
         a[j] = v;
       }
 }
```

Running time depends on increment sequence

Solved problem:

* running time is

Open problems:

* "best" increment sequences for practical N
* average-case analysis for any interesting seq
* N log N variants
* variants corresponding to log N depth network

Pratt Bounds (1971)

UPPER BOUND

Use following increments

1	**2**	**4**	**8**	**16**	**32**	**64**	**128**	**256**
.	**3**	**6**	**12**	**24**	**48**	**96**	**192**	**384**
.		**9**	**18**	**36**	**72**	**144**	**288**	**576**
.			**27**	**54**	**108**	**216**	**432**	**864**
.				**81**	**162**	**324**	**648**	**1296**
.					**243**	**486**	**962**	**1924**
.						**729**	**1458**	**2916**
.							**2187**	**4374**
.								**6561**

Total running time is

Applies to networks

Too slow in practice

LOWER BOUND

If increment sequence is "almost geometric"
then total running time must be

Sedgewick Upper Bound (1982)

Use the following increments

1 8 23 77 281 1073 4193 16577 . . .

Increment sequence not "almost geometric"

Connection to "Frobenius problem"

Smaller of two bounds

first bound

second bound

Use first bound for small increments

Use second bound for large increments

Total running time is

Frobenius Problem

A country wishes to issue k different stamps

* Number of values that cannot be achieved?
* Largest value that cannot be achieved?

Examples

Two stamps, relatively prime (Curran-Sharp, 1884)

Three stamps (Selmer, 1977)

Chazelle Upper Bound

Generalize Pratt "network" construction

Example

1	**7**	**49**	**343**	**2401**	**16807**	**117649**
.	**8**	**56**	**392**	**2744**	**19208**	**134456**
.		**64**	**448**	**3136**	**21952**	**153664**
.			**512**	**3584**	**25088**	**175616**
.				**4096**	**28672**	**200704**
.					**32768**	**229376**
.						**262144**

Total running time is

Choose parameter optimally
(restrict to logarithmic number of passes)

Too slow in practice

Sedgewick-Incerpi Upper Bound

Start with a "basis" of relatively prime numbers

1 3 7 16 41

Build a sequence with every number the product of
- a basis number and
- a number earlier in the sequence

1	**1*3**	**1*3* 7**	**1*3* 7*16**	**1*3* 7*16* 41**
.	**1*7**	**1*3*16**	**1*3* 7*41**	**1*3* 7*16*101**
.		**1*7*16**	**1*3*16*41**	**1*3* 7*41*101**
.			**1*7*16*41**	**1*3*16*41*101**
.				**1*7*16*41*101**

1	**3**	**21**	**336**	**13776**
.	**7**	**48**	**861**	**33936**
.		**112**	**1968**	**86961**
.			**4592**	**198768**
.				**463792**

Asymptotically optimal (same as Chazelle)

Fast in practice

Poonen Lower Bound

Using M increments on a file of size N requires

at least

comparisons in the worst case, for some c>0.

Applies to any algorithm that

* uses a number of passes
  compare-exchanging items
  at a fixed increment
* does at least   comparisions on each pass
* does not disturb k-ordering once achieved

Complexity "gap"

thousand	**10**	**3**	**9**	**78**
million	**20**	**4**	**22**	**482**
billion	**30**	**6**	**43**	**1933**
trillion	**40**	**9**	**78**	**6233**

UPPER BOUND

passes:
total cost:

LOWER BOUND

passes:
total cost:

AVERAGE CASE

No results for any interesting sequences
Simulations show
    average case close to worst case
    for sequences designed to worst case

Average case (two or three increments)

Analysis of (h, 1) Shellsort (Knuth)

Analysis of (h, k, 1) Shellsort (Yao)

Asymptotic result for three increments?

Shakersort (Incerpi, Sedgewick, 1984)

Shellsort "network"

Do one "cocktail shaker" pass

(not full insertion sort)

for each increment

Choose increments close to

Always seems to sort (!)

Poonen’s bound applies; can’t always sort

Can serve as basis for probabilistic

sorting network with N log N comparators

Variants

try more sophisticated increment sequences

do multiple shakes for each increment

add 1-shakes at end if necessary

DISADVANTAGE

network is "depth" N

Bricksort (Sedgewick, Lemke, 1995)

Shellsort "network"

- Do one "brick" pass
  - (not full shaker pass)
  - for each increment
- Choose increments close to
- Always seems to sort (!!)

Poonen’s bound applies?

Can serve as basis for probabilistic
- sorting network of "depth" log N

Variants
- try more sophisticated increment sequences
- do multiple brick passes for each increment
- add 1-passes at end if necessary

Average-case analysis??

Analysis of Bricksort