lecture 16 – Oct 09

please complete onboarding exam!

space complexity of insertion sort:

$S(N) = \Theta(N)$

sorting "in-place", i.e, it uses no additional memory at all on top of the given array.

3.2 Shellsort

insertion sort is very efficient on fairly pre-sorted input

⇝ idea
- perform insertion sort on a sub-sequence of elements that are far apart (separated by gap g).
- decrease g and repeat

⇝ when g reaches 1, the round equals insertion sort but then the array is well pre-sorted.

"Shell sort" Donald Shell 1959

How to decrease g? "gap sequence"

- important: g must eventually become 1.
- Shell's suggestion:

$\frac{N}{2}, \frac{N}{4}, \frac{N}{8}, \ldots, 1$

~>

depend on gap sequence

```
for (g = N/2; g>0; ~~g/2~~ g=g/2)
{
    for (i=g; i<N; i++)
    {
        tmp = a[i];
        for (j=i; j>=g && a[j-g]>tmp;
                                j=j-g)
        {   a[j] = a[j-g];
        }
        a[j] = tmp;
    }
}
```

Example 33. $N = 9$ ~> $g = 4, 2, 1$

										# comp.
g=4	57	7	32	11	27	21	29	5	17	1
	27	7	32	11	57	21	29	5	17	1
	27	7	32	11	57	21	29	5	17	1
	27	7	29	11	57	21	32	5	17	1
	27	7	29	5	57	21	32	11	17	2
g=2	17	7	29	5	27	21	32	11	57	1
	17	7	29	5						1
	17	5	29	7	27					2
	17	5	27	7	29					

⋮

$g=1$ array somewhat presorted
run regular insertion sort

Shellsort with Shell gap sequence needs 31 comp.
Insertion 29 comp.

Shell's gap sequence makes shellsort <u>usually</u> better than insertion sort, but not always.

Shell's gap sequence has the disadvantage that odd positions get compared to even positions only when $g=1$.

Hibbard's gap sequence:

$$2^k-1,\ 2^{k-1}-1,\ 2^{k-2}-1,\ \dots,\ 7,\ 3,\ 1$$

for the largest k such that $2^k-1 < N$.

In Example 33, Hibbard's gap sequence would be

$$7,3,1$$

and would use only 20 comparisons.

Other sequence proposed in the literature ..

$S(N) = \Theta(N)$; in-place

running time analysis: tricky!, depends on gap sequence.

Theorem 9. For Shellsort with Shell's gap sequence,

$$T_{worst}(N) = \Theta(N^2).$$

Theorem 10. For Shellsort with Hibbard's gap sequence,

$$T_{worst}(N) = \Theta(N \cdot \sqrt{N}).$$

Sketch of proof of Theorem 9.

▷ $T(N) = O(N^2)$

a pass with gap g involves

$\underline{\;g\;}$ insertion sorts of $\sim$ $\underline{\;N/g\;}$ elements each

$\leadsto O\left(g \cdot \frac{N^2}{g^2}\right)$ for pass with gap g

$$\leadsto T(N) = O\left(\sum_{k=1}^{\log N} \frac{N^2}{2^k}\right)$$

$$= O\Big(N^2 \cdot \underbrace{\sum_{k=1}^{\log N} \frac{1}{2^k}}_{\leq 1}\Big)$$

$$= O(N^2)$$

▷ $T(N) = \Omega(N^2)$

The upper bound $O(N^2)$ is also a lower bound by providing input sequences that require $\Omega(N^2)$ comparisons

suppose list elems. $1, \ldots, N$

$1, \frac{N}{2}+1, 2, \frac{N}{2}+2, 3, \frac{N}{2}+3, \ldots, \frac{N}{2}, N$

(where $N = 2^k$ for some k)

idea: this array does not change before $g = 1$

then placing $2, 3, \ldots, \frac{N}{2}$ in their correct positions costs roughly

$$2 + 3 + 4 + \ldots + \frac{N}{2} = \sum_{i=2}^{N/2} i = \Theta(N^2).$$