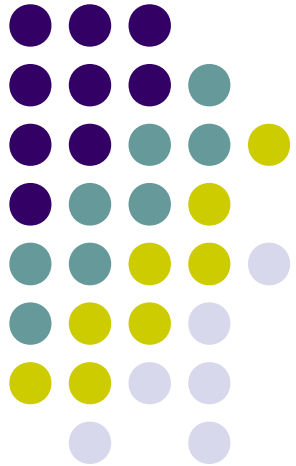
COMP20003 Algorithms and Data Structures Why sorting?

Nir Lipovetzky

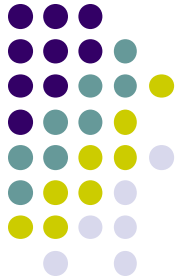
Department of Computing and Information Systems

University of Melbourne

Semester 2



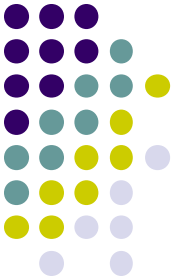
Why is sorting useful to study?



- Sorting has many applications and is used widely
  - In the business world.
  - In science.
  - In many disciplines.
- Sorting is used by many *other* algorithms.
- Sorting is very well-studied.
- Sorting demonstrates many fundamental concepts in computer science.

- Skiena: Chapter 4

Why is sorting useful to study?



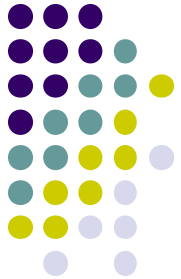
- Different algorithms for sorting have different properties, which affect performance.

n	$n^2/4$	$n \lg n$
10	*25*	33
100	2,500	*664*
1,000	250,000	*9,965*
10,000	25,000,000	*132,877*
100,000	2,500,000,000	*1,660,960*

Table from Skiena, The Algorithm Design Manual

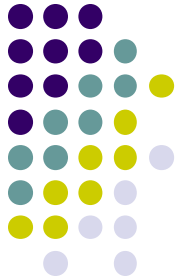
- When data are big, efficiency really matters.

Selection Sort



```
void selection(item A[ ], int n)
{
   int i,j,min;
   for(i=0;i<n-1;i++)      /* why n-1? */
   {
       min = i;
       for(j=i+1;j<n;j++)
       {
           if(cmp(A[j],A[min)])<0) min = j;
       }
        SWAP(A[i],A[min]);
   }
}
```

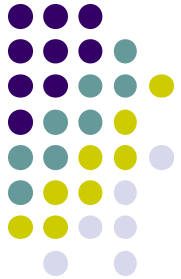
Selection Sort



- Worst case:
- Best case:
- Average case:

- Usefulness?
  - When records are large, and costly to move, Selection Sort takes O(n) data movements.

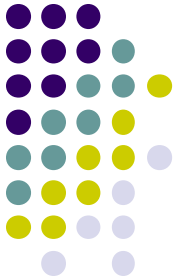
Selection Sort



- Is selection sort stable?

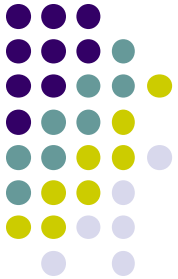
- Whenever there is a swap in an algorithm, there is the potential for being unstable.

Insertion Sort: The idea



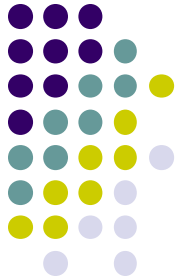
```
void insertion(item A[ ], int n)
{
   int i,j,val;
   for( i=1; i < n; i++ )
   {
       val = A[i]; j=i;
       while( A[j-1] > val )
       {
           A[j] = A[j-1]; j--;
       }
       A[j] = val;
   }
} /* this code doesn't usually work – why not? */
```

Insertion Sort



- Need to either:
  - Check at each point that **`j>0`**; or
  - Make one pass in the beginning, put the smallest element in position 0; or
  - Use **`A[0]`** for a sentinel, *e.g.* **`MAXINT`**, and put the elements in **`A[1]`** through **`A[n];`**
- Sentinel is fastest.

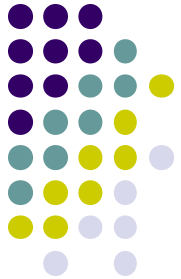
Insertion Sort



- Worst case:
- Average case:
- Best case:
- Stability?

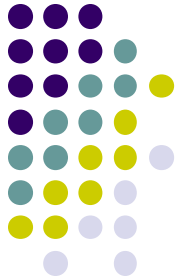
- Usefulness of insertion sort:

Divide and Conquer



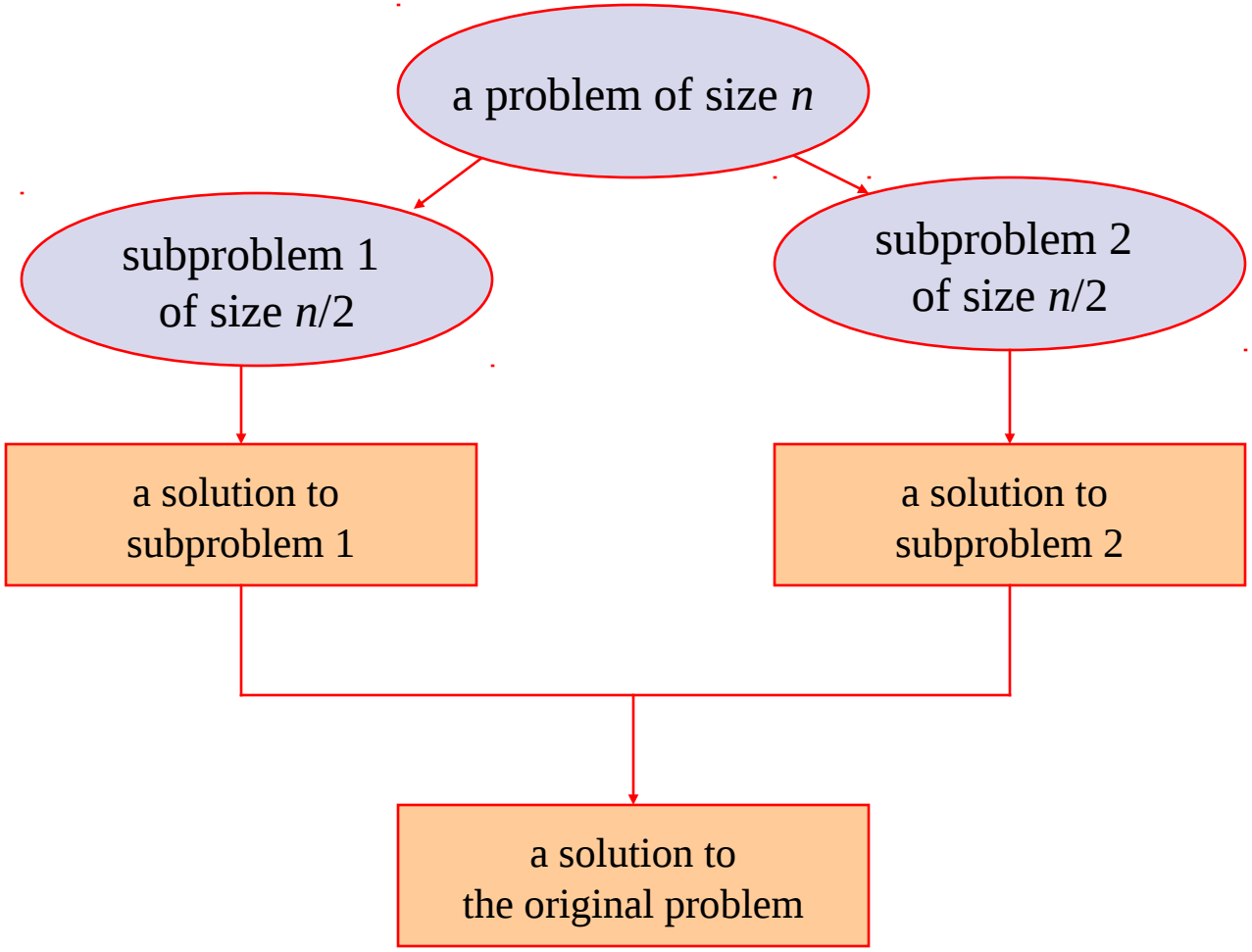
- Divide-and-conquer is a common strategy in efficient algorithms.
- Divide and Conquer Strategy:
  - Divide instance of problem into smaller instances.
  - Solve smaller instances – usually recursively.
  - *e.g.* Binary Search

Divide and Conquer

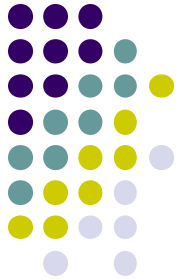


- In sorting, the usual strategy is:
  - Divide instance of problem into smaller instances.
  - Solve smaller instances – usually recursively.
  - Combine smaller solutions.

Divide and Conquer, or Split-solve-*join*



Split-solve-join



- Hard split, easy join: Quicksort
- Easy split, hard join: Mergesort