CS1113
Selection Sort, Insertion Sort, and Proof by Induction

Lecturer:

Professor Barry O’Sullivan

Office: 2.65, Western Gateway Building

email: *b.osullivan@cs.ucc.ie*

http://osullivan.ucc.ie/teaching/cs1113/

Selection Sort, Insertion Sort, and Proof by Induction

Selection Sort

Insertion sort

Principle of mathematical induction

Example proofs of statements from algorithm analysis

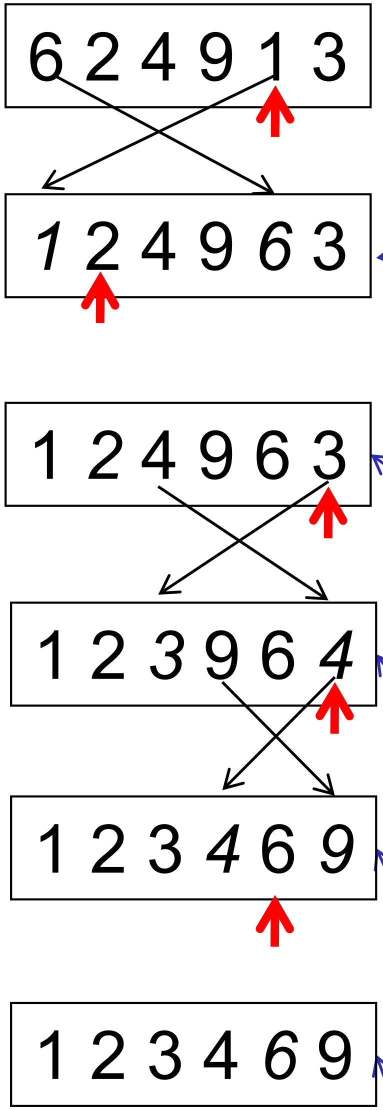
Selection Sort

Pass through the list (*n-1*) times.

On the first pass, we find the smallest item and swap it with the item in 1st place.

On the second pass, we find the 2nd smallest item, and swap it with the item in 2nd place.

...



1st pass complete
Smallest element
now in first place

2nd pass complete
2nd smallest element
now in 2nd place

3rd pass complete
3rd smallest element
now in 3rd place

4th pass complete
4th smallest element
now in 4th place

5th pass complete
5th smallest element
now in 5th place

Algorithm: Selection Sort

```
Input: a sequence x1,x2,...,xn (where n > 1)
Output: a sequence y1,y2,...,yn, which is an
        ordering of the xi

for each i from 1 to n-1
    min := i
    for each j from i+1 to n
        if xj < xi
            min := j
    temp := xi
    xi := xj
    xj := temp
return x sequence
```

x_i is the new position we are trying to establish

consider all values after x_i

if x_j is smallest so far

remember j

swap xi with smallest remaining

Selection sort is O(n^2)

<u>Proof</u>

We go round the outer loop n-1 times.

Each time, we check the remaining elements, so
n-1, *n-2*, ..., 1 checks

So we have

$$(n-1)+(n-2)+\ldots+1=\sum_{i=1}^{n-1} i=(n(n-1)/2)=n^2/2-n/2$$

comparisons, which is O(n^2)

Insertion sort

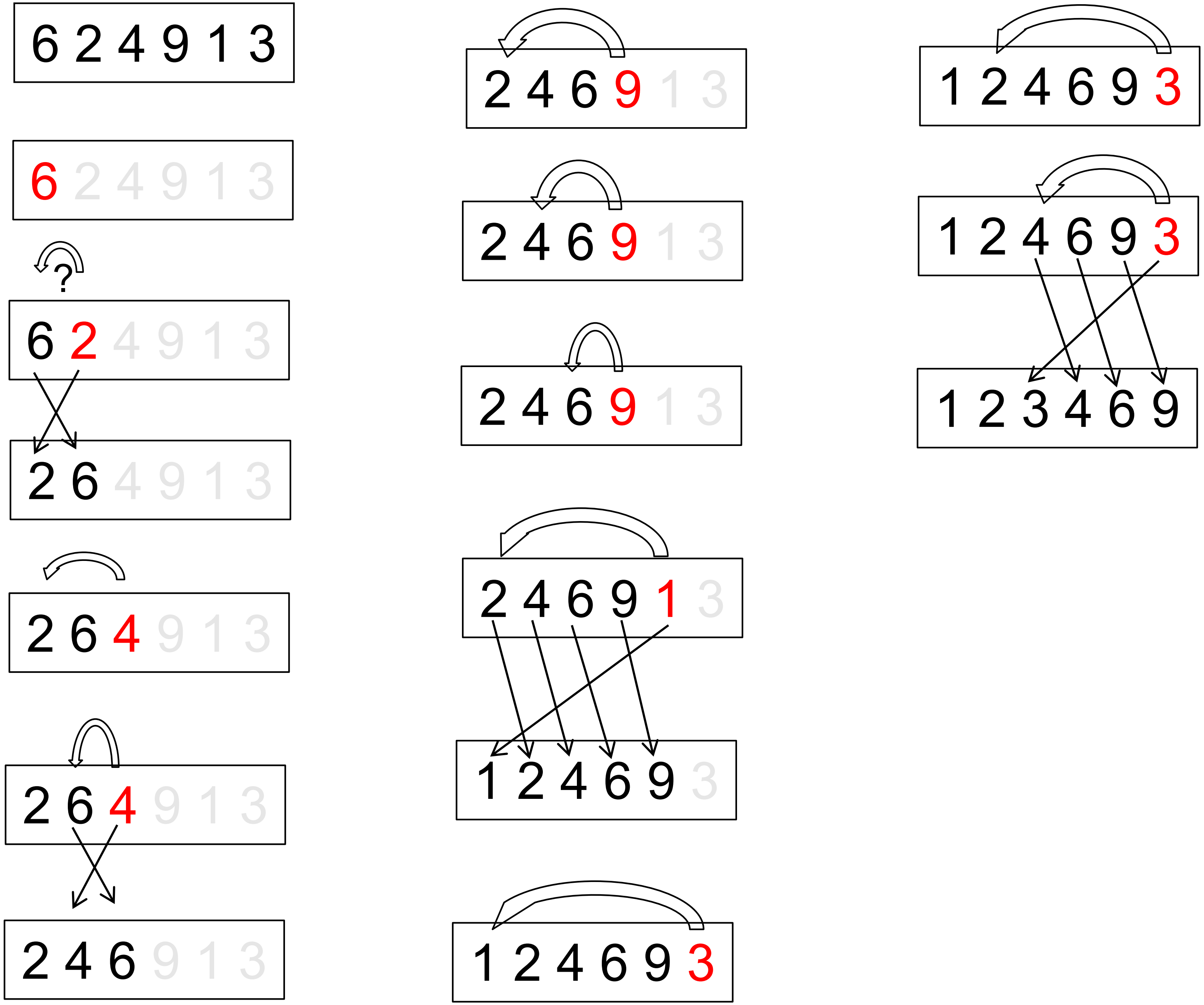
We now consider a different algorithm for sorting a sequence.

First we consider the sub-sequence of just the first element. It is obviously in the right position in that sub-sequence.

Now we add the current 2nd element to that subsequence. Either it goes after the 1st (and so we do nothing), or it goes before, so we move the old 1st element to 2nd place.

Then we add the 3rd element. Either it goes in 3rd place (so we do nothing), or it goes into 2nd place, so we shove the element that was there into 3rd, or it goes 1st, and we shove the other two elements back one place.

We repeat until we have added the last element.


6 2 4 9 1 3
6 2 4 9 1 3
?
6 2 4 9 1 3
2 6 4 9 1 3
2 6 4 9 1 3
2 6 4 9 1 3
2 4 6 9 1 3
2 4 6 9 1 3
2 4 6 9 1 3
2 4 6 9 1 3
2 4 6 9 1 3
1 2 4 6 9 3
1 2 4 6 9 3
1 2 4 6 9 3
1 2 4 6 9 3
1 2 3 4 6 9

Algorithm: Insertion Sort

Input: a sequence $x_1, x_2, \ldots, x_n$ (where $n > 1$)
Output: a sequence $y_1, y_2, \ldots, y_n$, which is an ordering of the x_i

```
for each i from 2 to n
    j:=1
    while x_j < x_i
        j:=j+1
    temp := x_i
    for each k from i down to j+1
        x_k := x_{k-1}
    x_j := temp
```

x_i is the new element we are trying to insert

keep going until we find the right place

we will insert x_i in front of x_j but store x_i for now

working forwards, copy each element into the next position

which leaves x_j's old position free, and we now insert the value we stored into that position

Insertion sort is O(n^2)

<u>Proof</u>

In the worst case, all the elements are already in the correct order, and each time we extend the subsequence by one more element, we have to compare it against all other elements in the subsequence (and then against itself to stop the while loop).

The length of the subsequence ranges from 2 to *n*, and so we need to compare against 1, 2, 3, ..., *n-1* elements. So we have

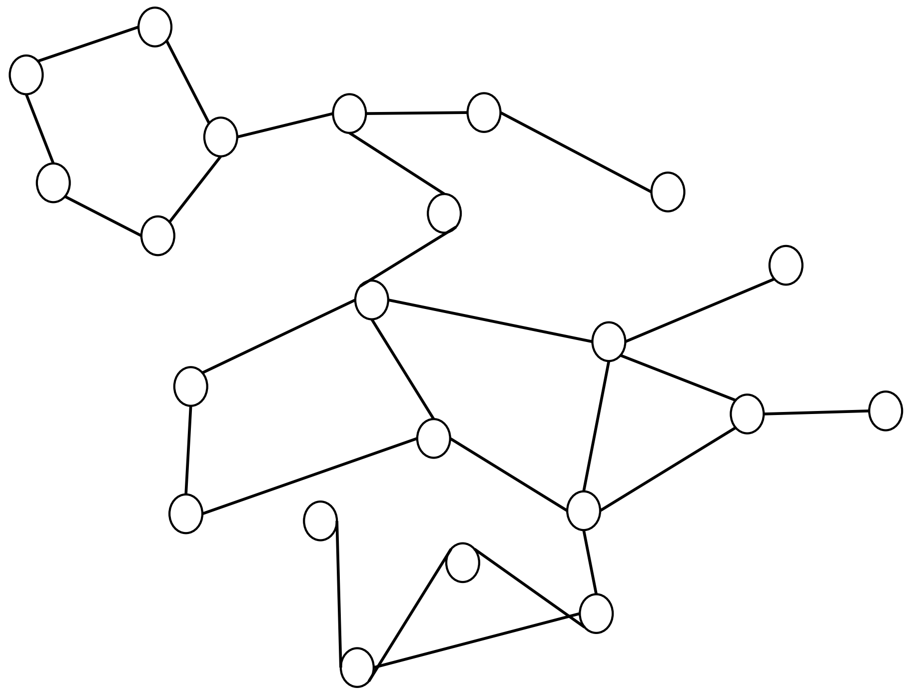
$$1+2+3+\ldots+n-1=\sum_{i=1}^{n-1} i=(n(n-1)/2)=n^2/2-n/2$$

comparisons, which is O(n^2)

Example

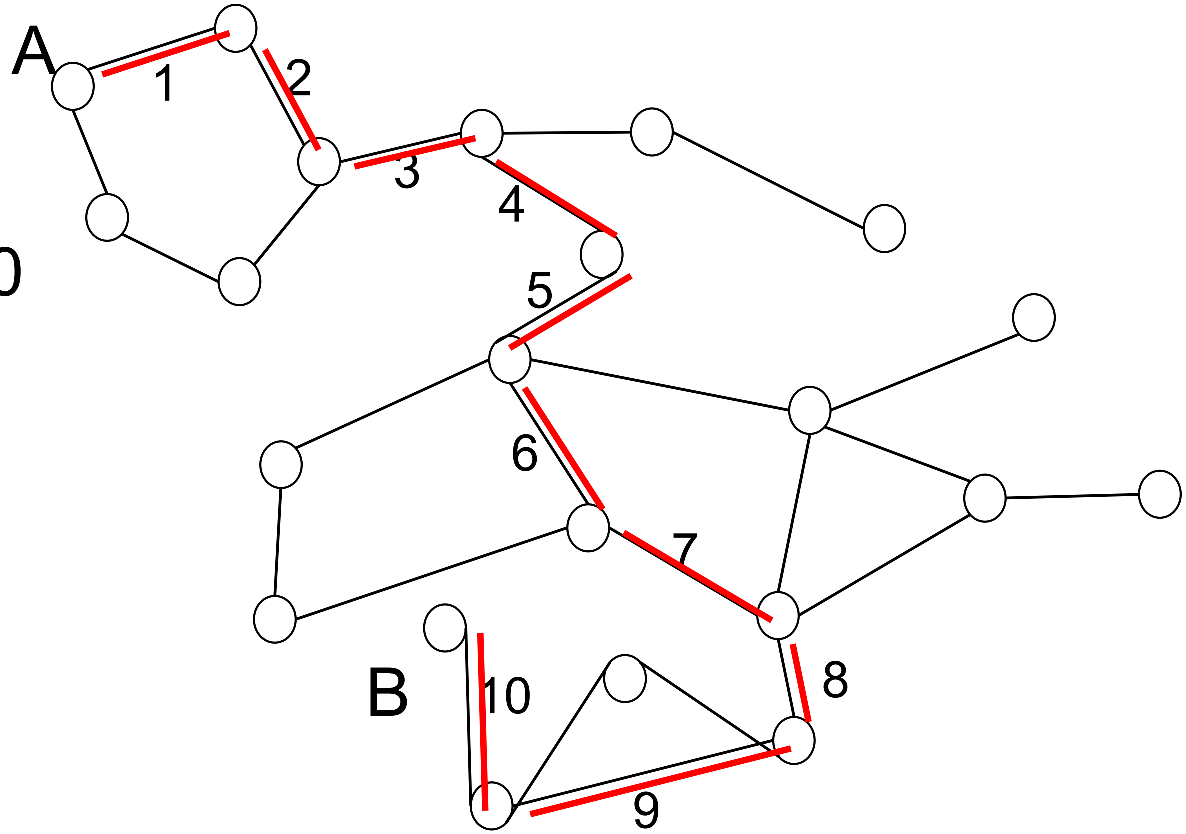
A network is a collection of devices, with links between pairs of devices, and where there is a finite chain of links between any pair.

I can send a packet from one device along a link to its neighbour. Any device that receives a packet can send it on in turn to one of its neighbours.



Can I send a packet from any node to any other node?

Choose A and B.
There is a chain that connects them with 10 links.
I can send a packet from A along link 1.
It can then be sent along link 2. Then 3, then 4, 5,6,7,8,9 and 10, when it reaches B.



But that argument was for a specific chain of known length.
Can we prove it for all possible pairs (or all possible chains)?

Proof

Every pair of devices has a finite chain of links connecting them.

If the chain between a pair is of length 1, then there is a single link between them, so I can send a packet along it.

Now suppose I can prove that I can send a packet along any chain of length *k*. (**)
Consider a chain of length *k*+1 connecting nodes A and B. The chain must consist of a chain of length *k* from A to some device C, and then a single link from C to B. By (**), I can send a packet from A to C. It can then be sent on from C to B over the single link. So I can send a packet along any chain of length *k*+1.

So I have proved that if I can send a packet along a chain of length *k*, then I can also send it along a chain of length *k*+1. But I have also proved I can send it along a chain of length 1. Therefore also of length 2, length 3, and so on for all finite lengths, and so I have proved I can send a packet between any pair of devices in the network.

Mathematical Induction

The proof on the previous slide used the principle of mathematical induction.

base case

We prove a claim for the simplest case, typically the value 1.

We then show that if we can prove it for some value *k*, then we can also prove it for *k*+1.

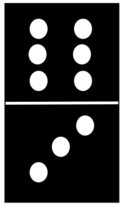
inductive step

We can then prove it for all values of *k*.

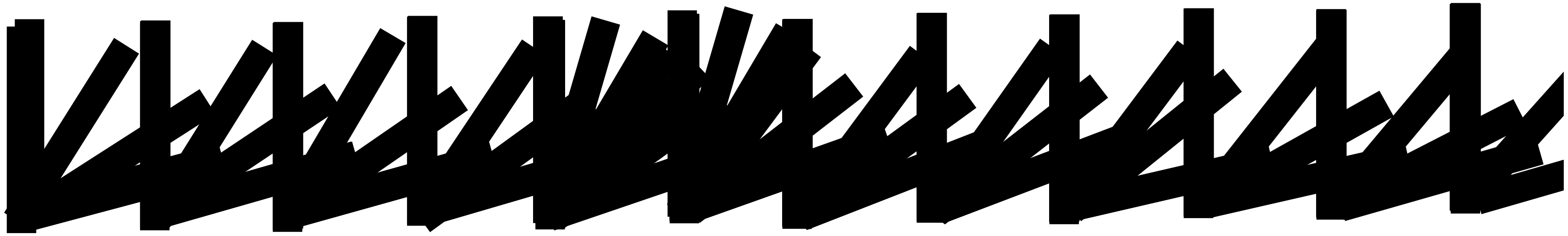
In logic terms, for some predicate P,

we have	$P(1)$
and we also have	$\forall x\ (P(x) \rightarrow P(x+1))$
so we can deduce	P(2), P(3), P(4), etc.

Induction as toppling dominoes



prove the dominoes are set up so that if any one domino falls, the one after it will also fall



prove that the first one falls

and so all dominoes must fall

Example Proof by Induction

Prove n^2+n is even, for all integers $n>0$.

<u>Proof</u>

Note that if a number x is even, then there is another number y such that $x=2y$, and if two numbers w and z are even, then $w+z$ is even.

When $n=1$, $n^2+n = 1^2+1 = 1+1 = 2$, which is even.

Now suppose result true for n=k, for some k>0. That is, k^2+k is even.

When $n=k+1$, $n^2+n=(k+1)^2+k+1 = k^2+2k+1+k+1$

$$=k^2+k+k+1+k+1$$

$$=k^2+k \quad + \quad 2(k+1)$$

But we have assumed k^2+k is even, and $2(k+1)$ is even, so their sum is also even. Therefore, if the result is true for $n=k$, it is also true for $n=k+1$. But we showed the result was true for $n=1$. Therefore, by induction, result is true for all $n>0$.

Example: $\sum_{i=1}^{n} i = n*(n+1)/2$

Proof

When n=1, $\sum_{i=1}^{1} i = 1 = \frac{1}{2}*1*2$ and so result is true for n=1.

Now suppose true for n=k. i.e. $\sum_{i=1}^{k} i = \frac{1}{2}*k*(k+1)$ [✶]

Consider n=k+1: $\sum_{i=1}^{k+1} i = \left(\sum_{i=1}^{k} i\right) + (k+1) = \frac{1}{2}*k*(k+1) + (k+1)$ (using ✶)

$$= \left(\frac{1}{2}*k+1\right)*(k+1)$$

$$= \left(\frac{1}{2}(k+2)\right)*(k+1)$$

$$= (k+1)*((k+1)+1)/2$$

So if result is true for n=k, it is true for n=k+1. But result is true for n=1, and so by induction, is true for all n>1.

$$\sum_{i=0}^{n} 2^i = 2^{n+1} - 1$$

This result is needed in the analysis of the runtime of various algorithms.

<u>Proof</u>

Will be done on the blackboard

Next lecture ...

other proof methods

recursion