

# Data Structures G5029

## Lecture 10

Kingsley Sage  
Room 5C16, Pevensey III  
[khs20@sussex.ac.uk](mailto:khs20@sussex.ac.uk)

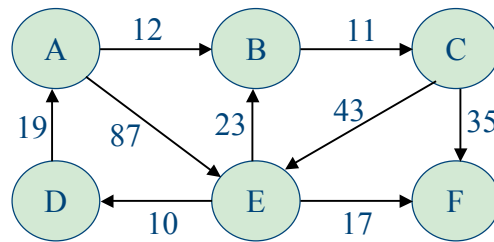
© University of Sussex 2006

## Lecture 10

- Shortest paths in weighted graphs
  - Dijkstra's algorithm

## What are we trying to achieve?

- Consider the weighted directed graph shown below.
- The shortest path from A to E is not the direct path A→E (length 87), but the indirect route A →B →C →E (length 66).
- Our aim is to find the shortest paths in directed graphs.
- We concentrate on **Dijkstra's algorithm** that solves the single source all paths problem for positively weighted directed graphs.



## Paths and lengths

- Recall that we are dealing with finite positively weighted directed graphs. These consist of a finite set of nodes and finite set of arcs.
- An arc can be represented by a weighted arrow:

$$a \xrightarrow{w} b$$

- which has a tail node  $a$ , a head node  $b$  and a weight  $w$ . When such an arc exists,  $b$  is said to be adjacent to  $a$ .
- We refer to the weight  $w$  as the length of the arc.

## Paths and lengths

- A path in the graph is a non-empty sequence of arcs such that the head of all but the last is the tail of the next. We can picture this as:

$$a_0 \xrightarrow{w_1} a_1 \xrightarrow{w_2} a_2 \dots a_{n-1} \xrightarrow{w_n} a_n$$

- We say that the path is from  $a_0$  to  $a_n$ , and that it visits  $a_0 \dots a_n$ . The length of a path  $p$ , written  $len(p)$ , is the sum of the weights (lengths) of its member  $w_1 + \dots + w_n$ .
- We say that node  $b$  is reachable from  $a$  if either  $a=b$ , or there exists a path from  $a$  to  $b$ .

## Set theory notation

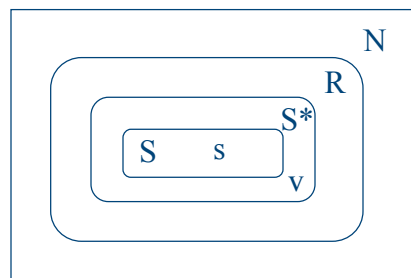
- $X \in Y$ :  $X$  is a member of the set  $Y$
- $X \subset Y$ :  $X$  is a proper sub set of the set  $Y$
- $X \subseteq Y$ :  $X$  is a proper sub set or the same set as  $Y$
- $X \cup Y$ : union (cup) of sets  $X$  and  $Y$

## Dijkstra's algorithm

- We wish to find the shortest path from some node  $s$  to every other node in the graph which is reachable from  $s$ .
- Let  $N$  be the set of all nodes in the graph. Our aim is to determine:
  - the subset  $R \subseteq N$  consisting of all nodes  $v \in N$  that are reachable from  $s$
  - the function  $d(v)$ , defined on  $R$ , such that if  $v \neq s$ ,  $d(v)$  is the minimum of the lengths of paths from  $s$  to  $v$ , otherwise conventionally we set  $d(s)=0$
  - Initially we only know that  $s \in R$  and  $d(s)=0$

## Dijkstra's algorithm

- The algorithm is iterative. At each stage we maintain two subset of nodes,  $S$  and  $S^*$  such that  $s \in S \subseteq S^* \subseteq R$



- $S^*$  consists of those nodes that belong to  $S$  or are adjacent to some node in  $S$
- $S$  and  $S^*$  grow as the algorithm proceeds
- Algorithm terminates when  $S=S^*=R$

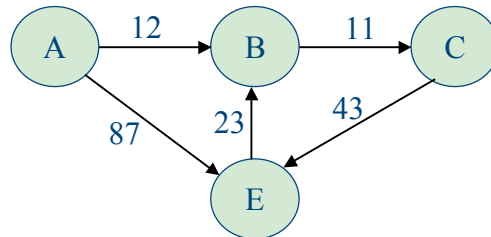
## Dijkstra's algorithm

- We assume that a function  $D(u)$  is defined for all  $u \in S^*$  that satisfies:
  - If  $u \in S$ , then  $D(u) = d(u)$
  - If  $u \in S^* - S$ , then there is a  $S$ -path from  $s$  to  $u$  and  $D(u)$  is the minimum length of all such paths
- An  $S$ -path is defined to be one that only visits nodes in  $S$  before finally jumping to  $u$ . Note that  $D$  will change as  $S$  grows.
- Initialisation of the algorithm:
  - Set  $S = \{s\}$  with  $D(s) = 0$
  - Set  $S^*$  to be  $S$  together with all nodes  $u$  adjacent to  $s$
  - Define  $D(u)$  for such nodes to be the length of the shortest arc from  $s$  to  $u$

## Iterative loop

- While  $(S^* - S)$  is not empty:
  - pick any  $v \in S^* - S$  such that  $D(v) \leq D(u)$  for all  $u \in S^* - S$
  - redefine function  $D$  on  $(S \cup \{v\})^*$  as follows:
    - $\Rightarrow$  if  $w \in S \cup \{v\}$ , leave  $D(w)$  unchanged
    - $\Rightarrow$  otherwise, if  $w$  is not adjacent to  $v$ , leave  $D(w)$  unchanged
    - $\Rightarrow$  otherwise if  $D(w)$  was not previously defined, set  $D(w)$  equal to the sum  $D(v)$  and the length of the shortest arc from  $v$  to  $w$
    - $\Rightarrow$  otherwise, set  $D(w)$  to be the smaller of the old value of  $D(w)$  and the sum of  $D(v)$  and the length of the shortest arc from  $v$  to  $w$
  - extend  $S$  to  $S \cup \{v\}$

## OK, time for an example ...

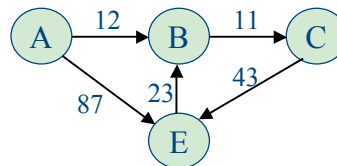


$S = \{A\}$ ,  $d(A) = 0$

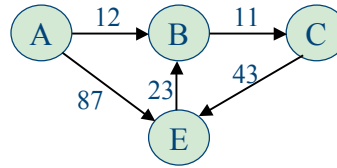
$S^* = \{A, B, E\}$  ( $S$  + all nodes adjacent to  $S$ )

$D$  initialised as  $d(B) = 12$ ,  $d(E) = 87$ ,  $d(A) = 0$  and  $d(C)$  undefined

## Example ...

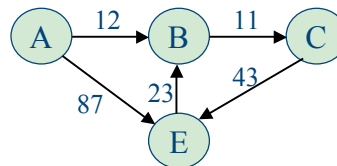


- While  $(S^* - S)$  is not empty:
  - pick any  $v \in S^* - S$  such that  $D(v) \leq D(u)$  for all  $u \in S^* - S$
  - redefine function  $D$  on  $(S \cup \{v\})^*$  as follows:
    - ⇒ if  $w \in S \cup \{v\}$ , leave  $D(w)$  unchanged
    - ⇒ otherwise, if  $w$  is not adjacent to  $v$ , leave  $D(w)$  unchanged
    - ⇒ otherwise if  $D(w)$  was not previously defined, set  $D(w)$  equal to the sum  $D(v)$  and the length of the shortest arc from  $v$  to  $w$
    - ⇒ otherwise, set  $D(w)$  to be the smaller of the old value of  $D(w)$  and the sum of  $D(v)$  and the length of the shortest arc from  $v$  to  $w$
  - extend  $S$  to  $S \cup \{v\}$



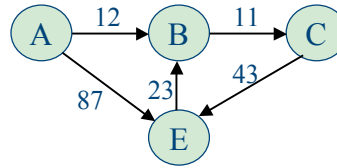
## Example ...

- $S=\{A\}$ ,  $S^*=\{A,B,E\}$ ,  $S^*-S=\{B,E\}$
- pick  $v \in \{B,E\}$  such that  $D(v) \leq D(u)$  for all  $u \in S^*-S$
- $d(B)=12$  and  $d(E)=87$ , so pick  $v=B$
- Redefine  $D$  on  $(S \cup \{v\})^*$  i.e.  $(\{A,B\})^* = \{A,B,C,E\}$ 
  - we don't update  $d(A)$  and  $d(B)$  as these are  $\subset (S \cup \{v\})=\{A,B\}$
  - Only  $C$  is adjacent to  $B$ , so we update  $d(C)$  and leave  $d(E)$  unchanged
  - $d(C) = 12 + 11 = 23$
  - So now  $D$  becomes  $d(A)=0$ ,  $d(B)=12$ ,  $d(C)=23$  and  $d(E)=87$
- $S$  becomes  $S \cup \{v\} = \{A,B\}$



## Example ...

- $S=\{A,B\}$ ,  $S^*=\{A,B,C,E\}$ ,  $S^*-S=\{E,C\}$
- pick  $v \in \{E,C\}$  such that  $D(v) \leq D(u)$  for all  $u \in S^*-S$
- $d(E)=87$  and  $d(C)=23$ , so pick  $v=C$
- Redefine  $D$  on  $(S \cup \{v\})^*$  i.e.  $(\{A,B,C\})^* = \{A,B,C,E\}$ 
  - we don't update  $d(A)$ ,  $d(B)$  and  $d(C)$  as these are  $\subset (S \cup \{v\})=\{A,B,C\}$
  - update  $d(E)$
  - $d(E)$  becomes smaller of old  $d(E)$  and sum  $d(C)$  and shortest arc from  $C$  to  $E$ , so  $d(E)$  becomes  $23+43=66$
  - So now  $D$  becomes  $d(A)=0$ ,  $d(B)=12$ ,  $d(C)=23$  and  $d(E)=66$
- $S$  becomes  $S \cup \{v\} = \{A,B,C\}$



## Example ...

- $S=\{A,B,C\}$ ,  $S^*=\{A,B,C,E\}$ ,  $S^*-S=\{E\}$
- pick  $v \in \{E\}$  so pick  $v=E$
- Redefine  $D$  on  $(S \cup \{v\})^*$  i.e.  $(\{A,B,C,E\})^* = \{A,B,C,E\}$ 
  - we don't update  $d(A), d(B), d(C)$  and  $d(E)$  as these are  $\subset (S \cup \{v\}) = \{A,B,C,E\}$
  - So  $D$  remains  $d(A)=0$ ,  $d(B)=12$ ,  $d(C)=23$  and  $d(E)=66$
- $S$  becomes  $S \cup \{v\} = \{A,B,C,E\}$
- And the algorithm terminates ...

## Justification

- See on-line course notes

## Next time ...

- Revision and past paper questions ...