# Reti di Calcolatori I

**Prof. Roberto Canonico**
**Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione**

**Corso di Laurea in Ingegneria Informatica**

**A.A. 2018-2019**

## Algoritmo di Dijkstra



Università degli Studi di Napoli
Federico II

# Nota di copyright per le slide COMICS

## Nota di Copyright



Autori:
Simon Pietro Romano, Antonio Pescapè, Stefano Avallone,
Marcello Esposito, Roberto Canonico, Giorgio Ventre

# Algoritmo di Dijkstra

- Ogni nodo ha a disposizione il grafo della rete:
  - i nodi sono i router
  - gli archi sono le linee di collegamento tra router:
    - agli archi è associato un costo
- Ogni nodo usa l'algoritmo di Dijkstra per costruire lo *Shortest Path Tree* del grafo, ovvero l'albero dei cammini di costo minimo
- Ad ogni nodo si assegna un'etichetta che rappresenta il costo massimo per raggiungere quel nodo
- L'algoritmo modifica le etichette cercando di minimizzarne il valore e di renderle permanenti

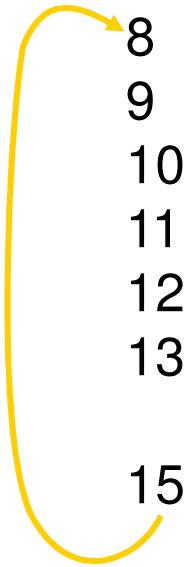
# Algoritmo di Dijkstra: formalizzazione

- La Topologia della rete è nota a tutti i nodi:
  - la diffusione è realizzata via "link state broadcast"
  - tutti i nodi hanno la stessa informazione
- Si calcola il percorso minimo da un nodo a tutti gli altri:
  - l'algoritmo fornisce la tavola di routing per quel nodo
- Iterativo: un nodo, dopo k iterazioni, conosce i cammini meno costosi verso k destinazioni

## Notazione:

- c(i,j): costo collegamento da i a j: $c(i,j) \geq 0$
  - infinito se non c'è collegamento
  - per semplicità, $c(i,j) = c(j,i)$
- D(v): costo corrente del percorso, dalla sorgente al nodo v
- p(v): predecessore (collegato a v) lungo il cammino dalla sorgente a v
- N: insieme di nodi per cui la distanza è stata trovata

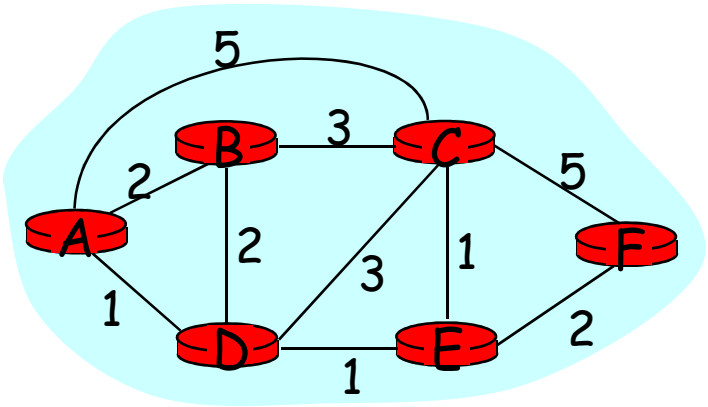
# Algoritmo di Dijkstra (eseguito da A)



1 ***Inizializzazione:***
2 N = {A}
3 per tutti i nodi v
4 if (v e' adiacente a A)
5 then D(v) = c(A,v)
6 else D(v) = ∞
7
8 ***Loop***
9 sia w non in N tale che D(w) è minimo
10 aggiungi w a N
11 aggiorna D(v) per ogni v adiacente a w e non in N:
12 D(v) = min( D(v), D(w) + c(w,v) )
13 {il nuovo costo fino a v è o il vecchio costo, oppure il costo del cammino piu breve fino a w più il costo da w a v }
15 ***fino a quando tutti i nodi sono in N***

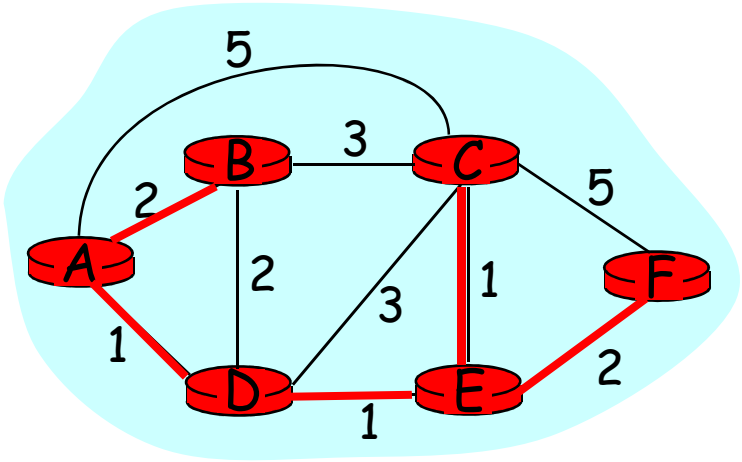
# Algoritmo di Dijkstra: interpretazione

- L'algoritmo consiste in un passo di inizializzazione, più un ciclo di durata pari al numero di nodi della rete. Al termine avremo i percorsi più brevi dal nodo sorgente a tutti gli altri nodi
- Esempio. Calcoliamo sulla rete data i percorsi di costo minimo da A a tutte le possibili destinazioni. Ciascuna riga della tabella della slide seguente fornisce i valori delle variabili dell'algoritmo alla fine di ciascuna iterazione



# Algoritmo di Dijkstra: esempio

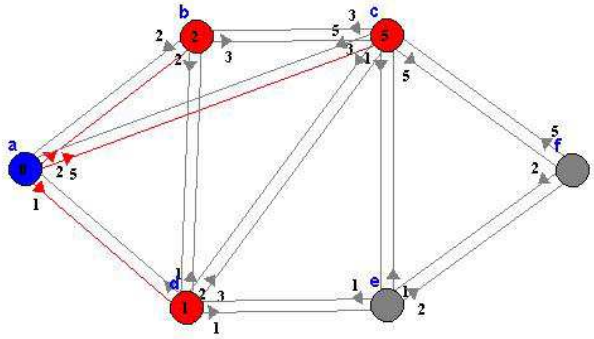
| Step | start N | D(B),p(B) | D(C),p(C) | D(D),p(D) | D(E),p(E) | D(F),p(F) |
|---|---|---|---|---|---|---|
| 0 | A | 2,A | 5,A | 1,A | infinity | infinity |
| 1 | AD | 2,A | 4,D | | 2,D | infinity |
| 2 | ADE | 2,A | 3,E | | | 4,E |
| 3 | ADEB | | 3,E | | | 4,E |
| 4 | ADEBC | | | | | 4,E |
| 5 | ADEBCF | | | | | |



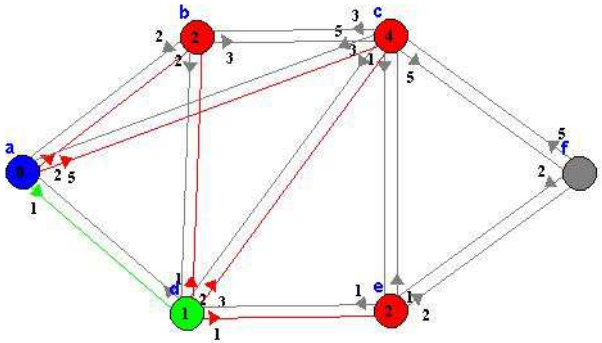

Notazione:

- c(i,j): costo collegamento da i a j (infinito se non c'e' collegamento e per semplicità c(i,j) = c(j,i))
- D(v): costo corrente del percorso, dalla sorgente al nodo v
- p(v): predecessore (collegato a v) lungo il cammino dalla sorgente a v
- N: insieme di nodi per cui la distanza è stata trovata

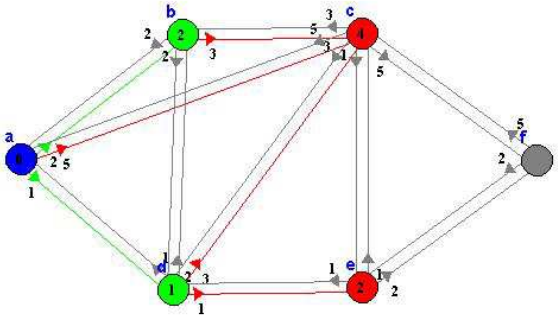
# Dijkstra: esempio



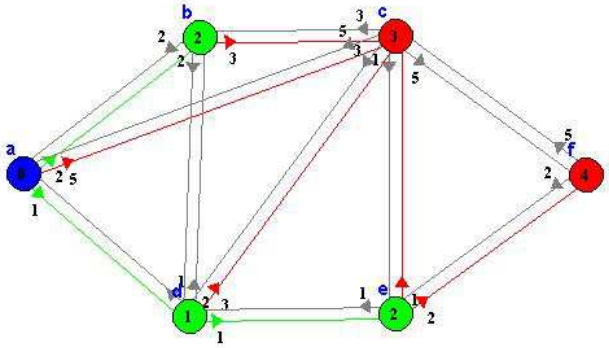
Passo 1



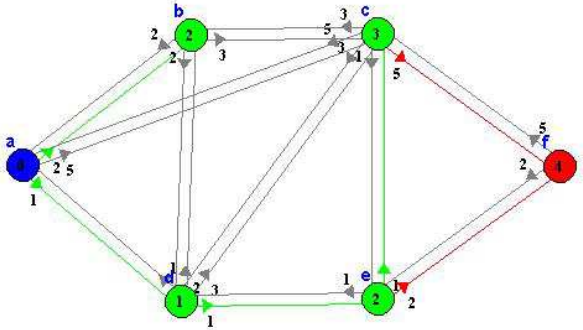
Passo 2



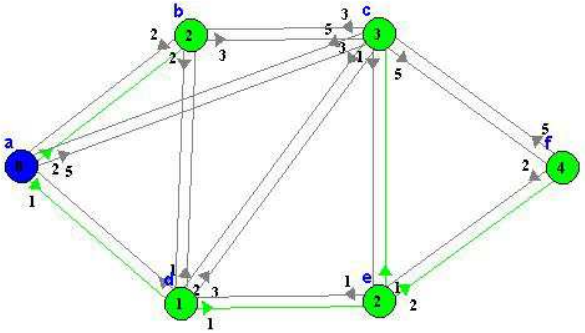
Passo 3



Passo 4

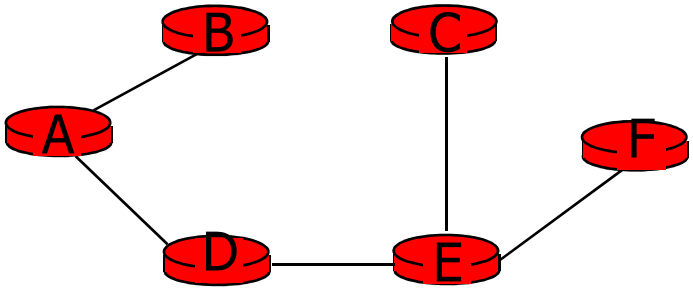


Passo 5



Passo 6

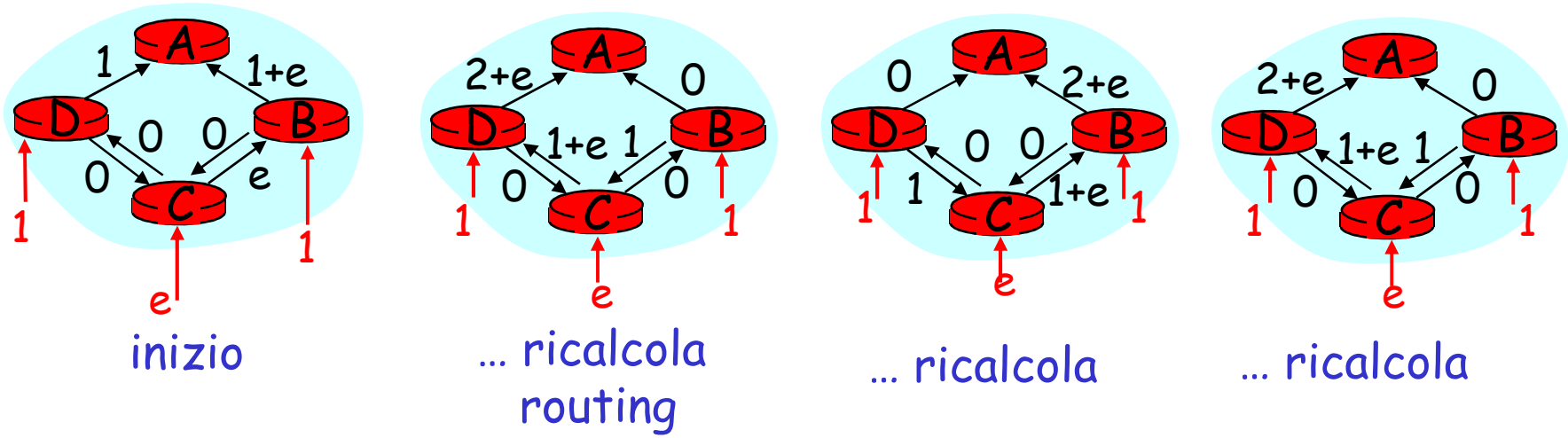
# Esempio: tabella di instradamento in A




| destination | link |
|---|---|
| B | (A,B) |
| C | (A,D) |
| D | (A,D) |
| E | (A,D) |
| F | (A,D) |

# Algoritmo di Dijkstra: discussione

Se il costo di un link è proporzionale al traffico su quel link, allora sono possibili oscillazioni




Soluzione: evitare la sincronizzazione nell'invio dei messaggi dei router