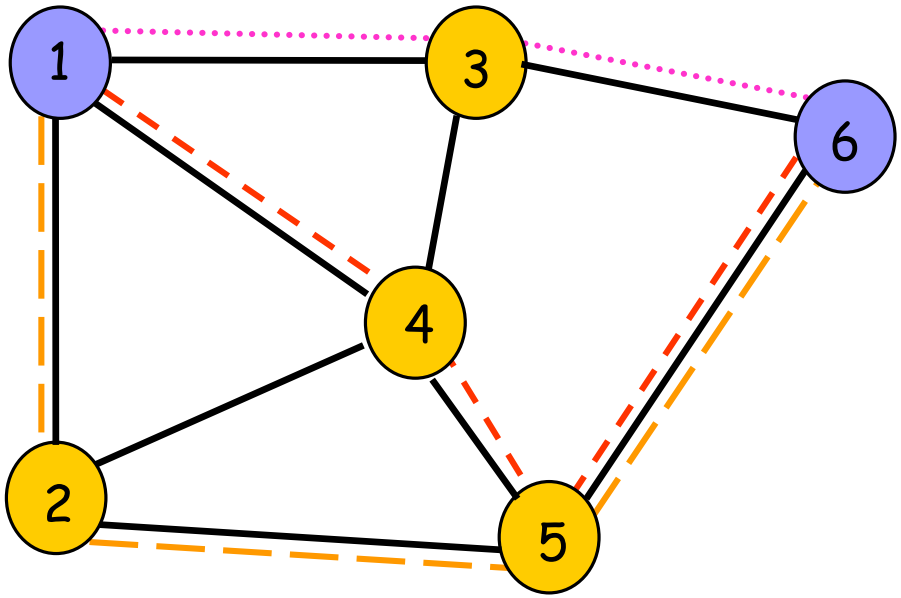
Marco Listanti

Strato di rete

"Routing in reti IP"

Funzione di instradamento: generalità

Instradamento in reti a pacchetto



- **Numerosi possibili cammini dal nodo 1 al nodo 6 (loopfree)**
  - 1-3-6, 1-4-5-6, 1-2-5-6, 1-3-4-5-6, 1-2-4-5-6, ….
- **Qual'è il cammino migliore ?**
  - Minimo ritardo
  - Minimo numero di hop
  - Minimo costo
  - Massima affidabilità

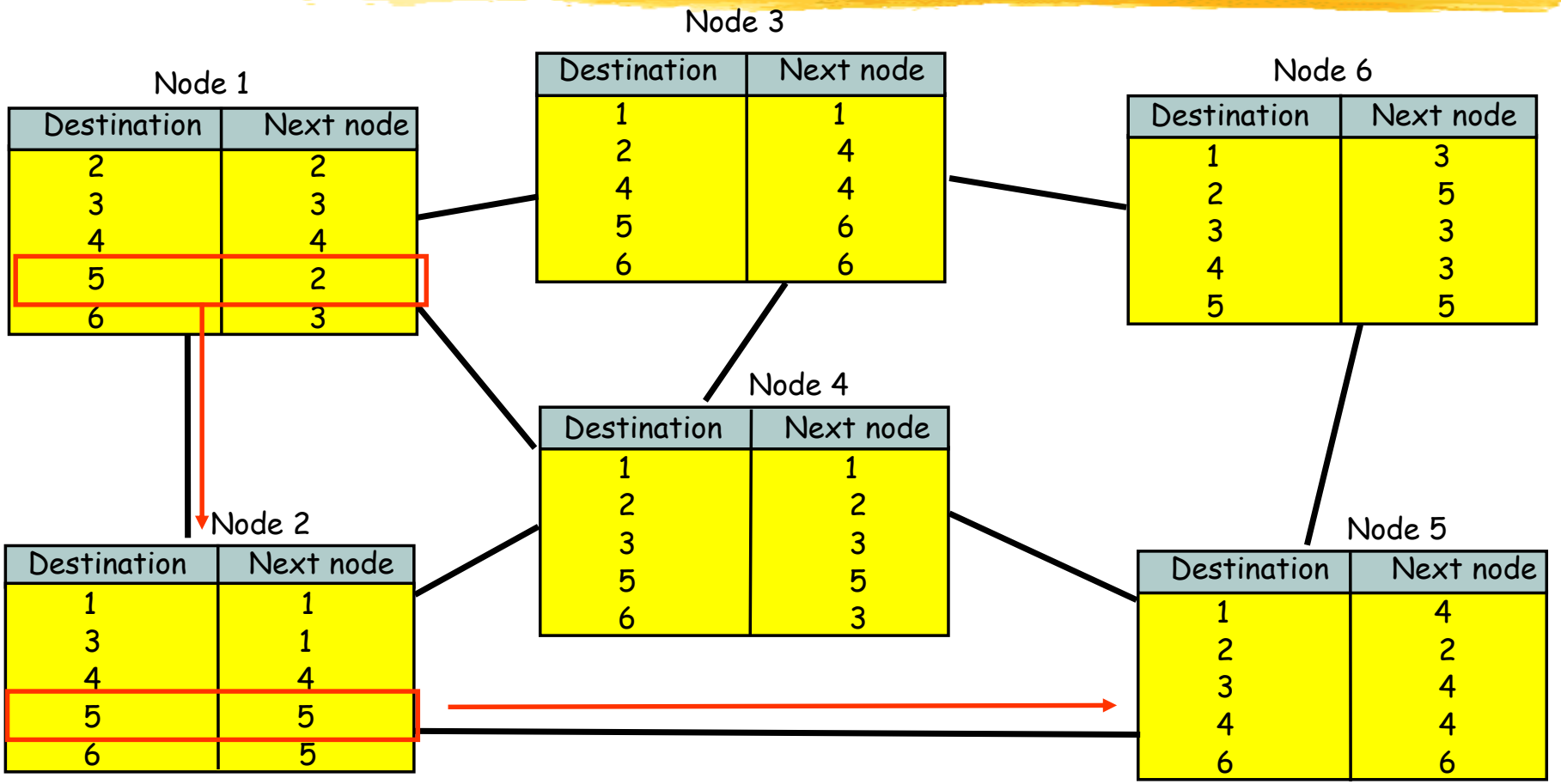
Creazione delle tabelle di routing

- **E' necessario definire la tipologia di informazioni sullo stato dei link**
  - Link up/down; stato di congestione; delay o altre metriche
- **Occorre distribuire le informazioni di stato dei link usando un protocollo di routing**
  - Quali informazioni devono essere scambiate ?
  - Con quale frequenza ?
  - Come effettuare lo scambio di informazioni: solo con i nodi con vicini, broadcast, flooding
- **Occorre calcolare i cammini migliori**
  - Algoritmo di instradamento
  - Metriche singole o multiple

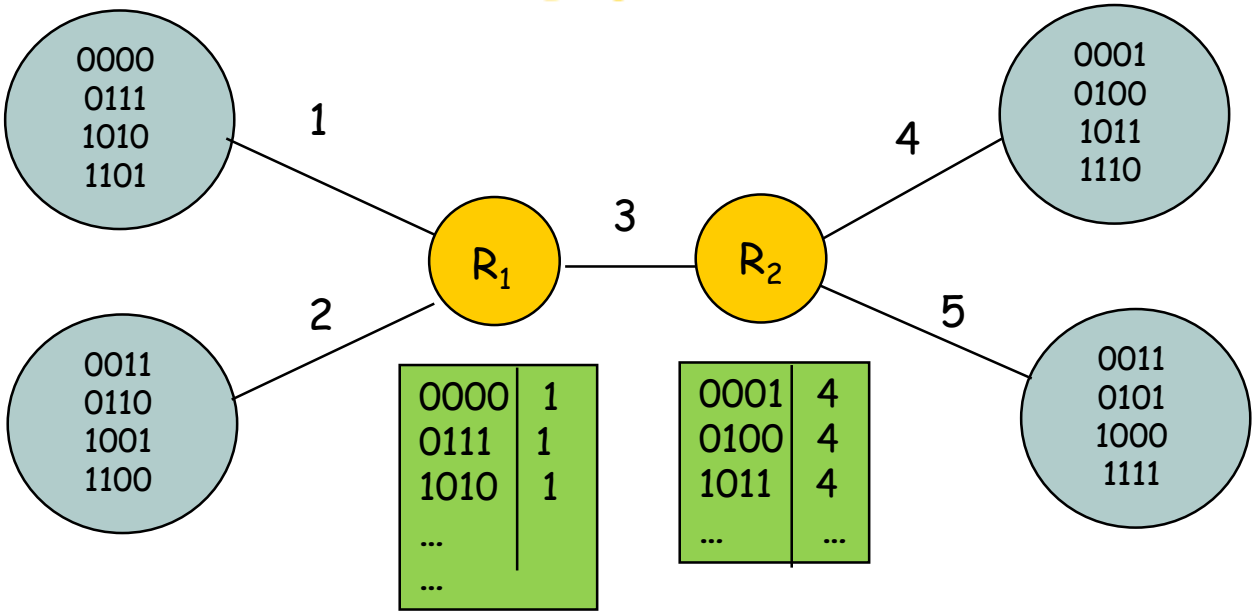
Requisiti

- **Risposta alle variazioni di stato**
  - Variazioni di topologia o banda dei link
  - Stato di congestione
  - Rapida convergenza
  - Assenza di loop
- **Ottimalità**
  - Utilizzazione ottima delle risorse di rete
  - Minimizzazione della lunghezza dei cammini
- **Robustezza**
  - Continuità di servizio in presenza di condizioni anomale (alto carico, congestione di rete, guasti, errori di implemetazione)
- **Semplicità**
  - Basso carico di elaborazione

Routing Table: principio di funzionamento

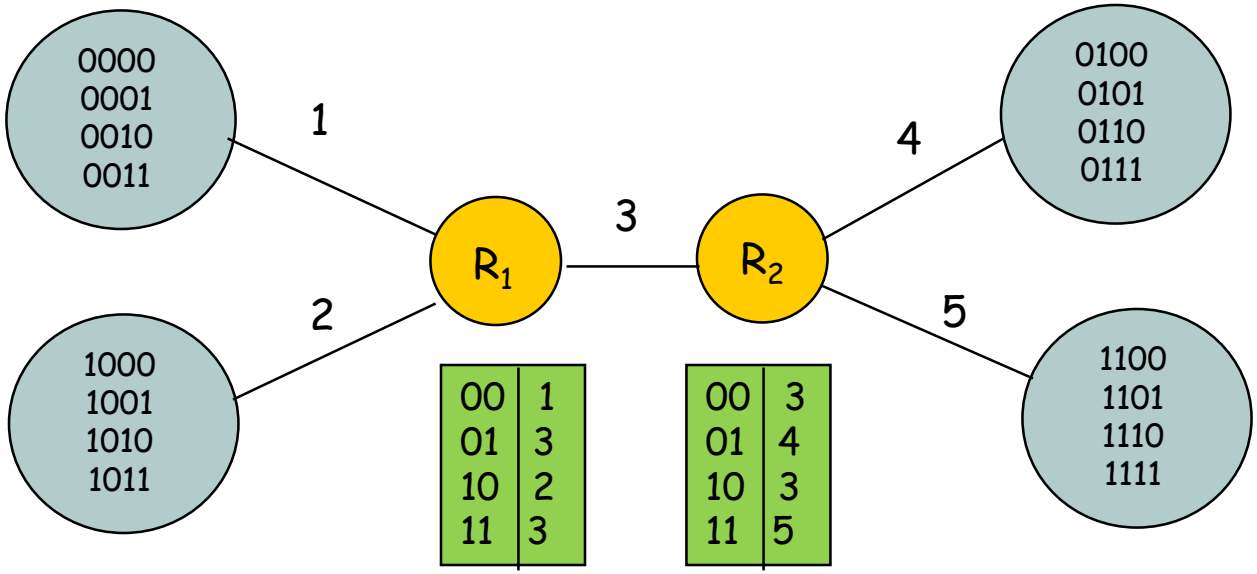


Indirizzamento e instradamento non gerarchici



- Nessuna relazione tra indirizzi e localizzazione geografica (vicinanza) delle destinazioni
- Routing table composte da 16 record ciascuna
  - Possibilità di routing table explosion

Indirizzamento e instradamento gerarchici



- I prefissi indicano la rete a cui un host è connesso
- Reti con lo stesso prefisso sono "vicine"
  - Routing table composte da 4 record ciascuna

Instradamento in reti IP

Instradamento in reti IP

- **La scelta del router verso cui inviare il pacchetto avviene utilizzando la Tabella di Instradamento (Routing Table - RT) contenuta in ogni host e in ogni router**
- **Ogni elemento di una RT contiene**
  - Indirizzo IP di destinazione (host address o network address)
  - Indirizzo del router successivo (next hop router) sul cammino verso la rete di destinazione
  - Indicazione dell'interfaccia fisica di uscita
- **Un router non conosce il cammino completo verso la destinazione**

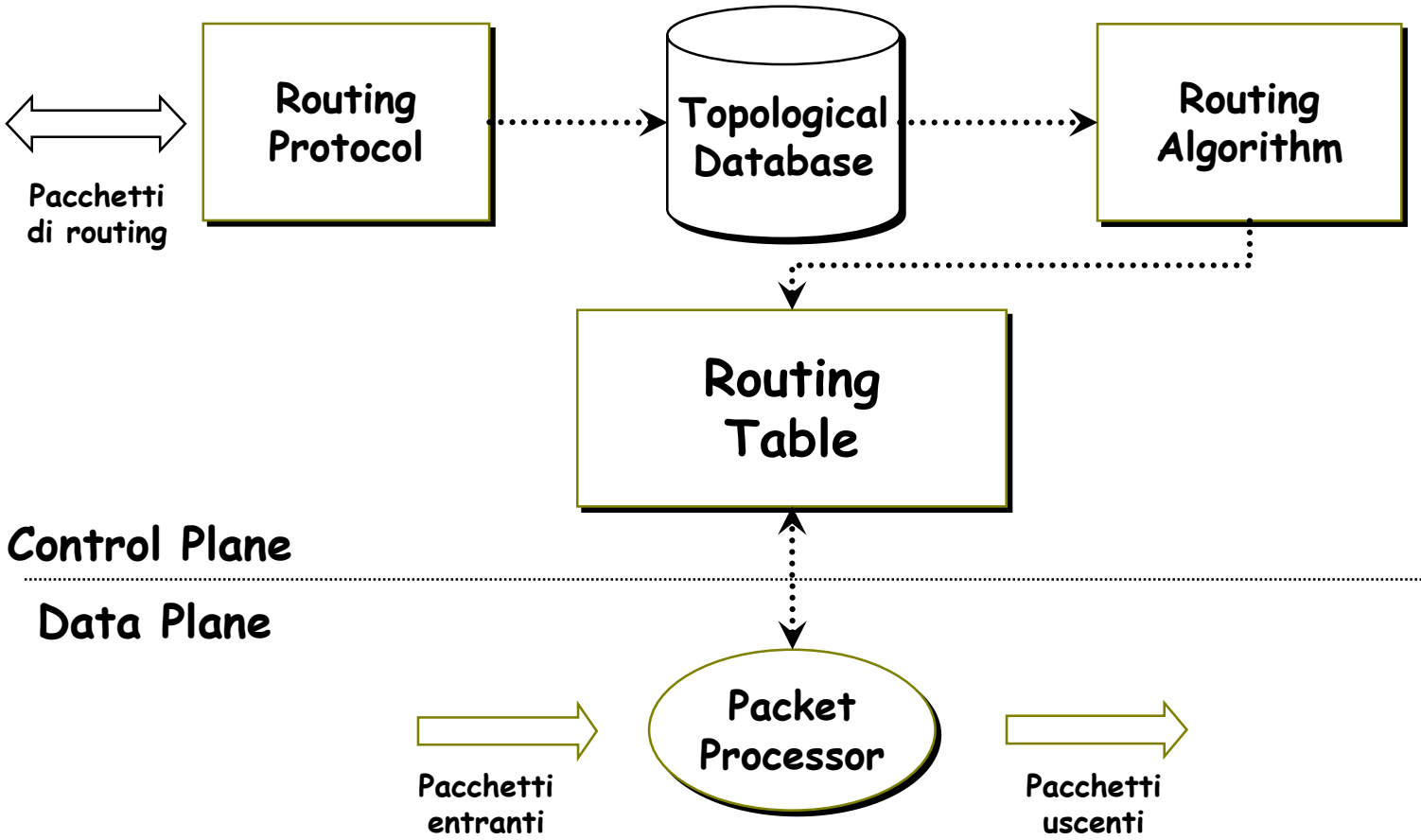
Instradamento in reti IP

- **Un router esegue i seguenti passi**
  - Estrae dal pacchetto entrante il contenuto del campo Destination Address
  - Ricerca all'interno della RT il record che contiene il "longest prefix matching" con il DA del pacchetto entrante
  - In caso di fallimento del passo precedente, ricerca l'indirizzo del "router di default"
  - Se nessuno dei passi precedenti da esito positivo, il pacchetto è classificato come "undeliverable" ed è scartato ed inviato un messaggio ICMP all'host sorgente

Instradamento in reti IP

- Un router possiede un Database Topologico in cui sono memorizzate le informazioni sulla topologia della rete
- Le informazioni sulla topologia di rete sono aggiornate dai messaggi del Protocollo di routing
- Sulla base delle informazioni contenute nel Database Topologico, l'Algoritmo di routing determina periodicamente i percorsi a costo minimo tra il router e le possibili reti di destinazione (network prefix)
- La Routing Table è costruita inserendo, per ogni destinazione, l'informazione relativa al next hop verso cui instradare il pacchetto

Instradamento in reti IP



Instradamento in reti IP

- **Le Routing Table sono dinamiche**
  - Ogni router ed ogni host aggiornano periodicamente le informazioni relative alla topologia di rete
- **L'aggiornamento dinamico è necessario perché:**
  - Internet non può essere considerata stabile, la topologia può cambiare
    - Aggiunta/eliminazione di reti/router/host
  - In caso di guasto di router/link alcuni cammini non sono utilizzabili
  - Possibile scelta del cammino in base allo stato di occupazione delle risorse di rete
- **Le RT devono essere aggiornate periodicamente (anche ad intervalli di pochi secondi)**
- **L'aggiornamento delle RT è attuato mediante protocolli di instradamnto (*Routing Protocol*)**

Sistemi autonomi

- **Un sistema autonomo (*Autonomous System* - AS) è un insieme di host e router controllato da una singola autorità amministrativa (es. ISP)**
  - Un particolare AS è detto "**Core AS**" e costituisce il backbone di Internet
  - Un router del core AS è detto "**Core Router**"
  - Gli altri AS sono detti "**Stub AS**"
- **Ogni AS ha il proprio protocollo di instradamento indipendente dai protocolli usati negli altri AS**
- **Uno Stub AS deve aver almeno un router connesso ad un core router; questi router sono detti *Exterior Gateway***
- **Un router interno ad un AS è detto *Interior Gateway***

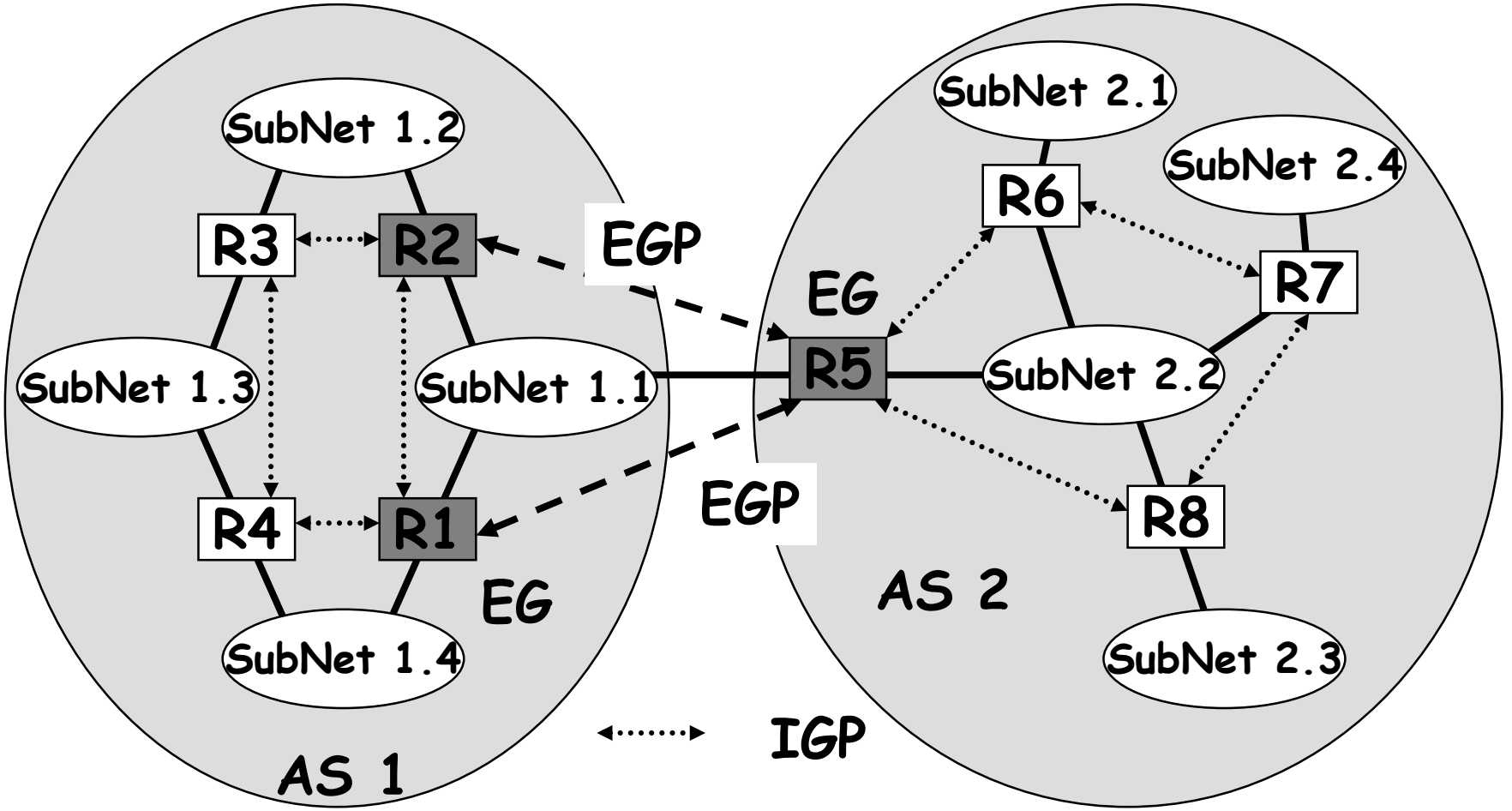
IGP e EGP

- I protocolli di instradamento all'interno di un AS sono detti *Interior Gateway Protocols* (IGP)
- Le informazioni di instradamento che coinvolgono più di un sistema autonomo sono gestite mediante gli *Exterior Gateway Protocols* (EGP)
- Le informazioni di instradamento degli EGP vengono inviate agli Exterior Gateway di ogni sistema autonomo
- L'instradamento all'interno di un sistema autonomo e la raccolta di dati da inviare ai core router avviene per mezzo degli IGP

IGP e EGP

- **Un EGP svolge tre funzioni**
  - individuazione dei router adiacenti con cui scambiare le informazioni di instradamento
  - verifica continua della funzionalità dei router interlocutori
  - scambio periodico delle informazioni di instradamento, queste riguardano la sola raggiungibilità delle reti, non la distanza

IGP e EGP

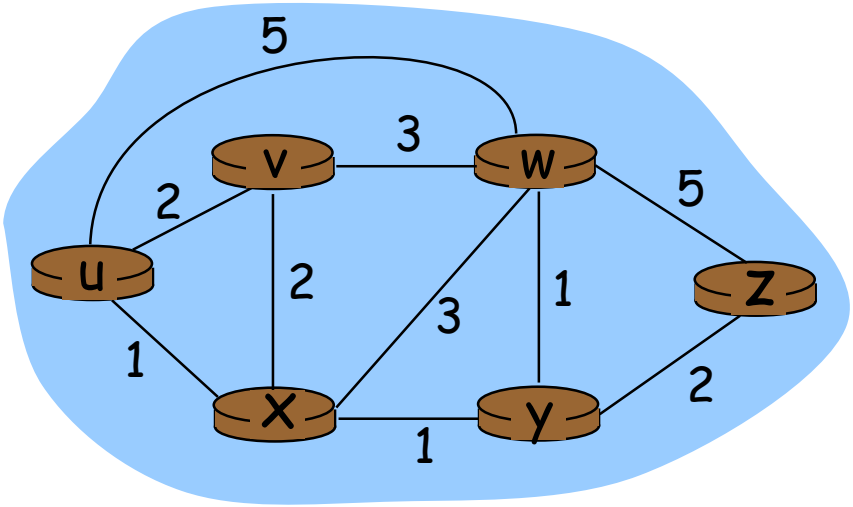

SubNet 1.2
R3
R2
EGP
EG
SubNet 1.3
SubNet 1.1
R5
SubNet 2.1
R6
SubNet 2.4
R7
SubNet 2.2
EGP
R4
R1
EG
R8
AS 2
SubNet 1.4
SubNet 2.3
IGP
AS 1

Algoritmi di instradamento
Generalità

Modello a grafo di una rete

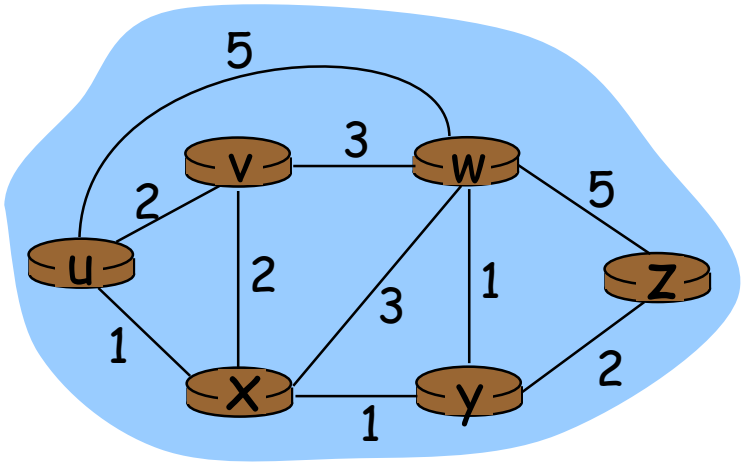
Grafo Pesato

G = (N,E,c)



- **N** = insieme di nodi (router) = { u, v, w, x, y, z }
- **E** = insieme di archi (collegamenti) = {(u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z)}
- c = insieme dei costi associati ai rami
  - c(x,x') = costo associato ramo (x,x')
  - Se il ramo (x,x') non esiste c(x,x') = ∞

Costo di un cammino



Il costo di un cammino è dato dalla somma dei costi associati ai archi componenti il cammino

Costo di un cammino $(x_1, x_2, x_3, \ldots, x_p)$ =
$= c(x_1,x_2) + c(x_2,x_3) + \ldots + c(x_{p-1},x_p)$

Il costo di un cammino è anche detto lunghezza del cammino o distanza

- Il protocollo di instradamento mette in grado ogni router di determinare il modello a grafo della rete
- L'algoritmo di instradamento determina il cammino a costo minimo tra due nodi della rete

Metriche

- **Misurano la "qualità" di un link o di un cammino**
  - Costo basso: link ad alta qualità (es. banda elevata), da includere, se possible, nei cammini
  - Costo elevato: link di bassa qualità (es. banda limitata), da escludere, se possible, nei cammini
- **Lunghezza di un cammino (Path Length) = somma dei costi dei link componenti (Distanza)**
- **Possibili metriche**
  - Numero di hop: misura approssimata delle risorse utilizzate
  - Affidabilità: grado di disponibilità del cammino; BER
  - Ritardo: somma dei ritardi lungo il path
  - Bandwidth: capacità disponibile lungo un path
  - Carico: Grado di utilizzazione dei link e dei router lungo il path

Approcci Shortest Path

- **Distance Vector Protocol**
  - Informazioni: i nodi adiacenti si scambiano la lista delle distanze verso le destinazioni
  - Viene determinato il next-hop migliore per ogni destinazione
  - **Algoritmo di Bellman-Ford**
- **Link State Protocol**
  - Informazioni: lo stato dei link (costi) è diffuso in rete (flooding)
  - I router conoscono l'intera topologia della rete
  - Ogni router calcola lo shortest path ed il next-hop verso ogni destinazione
  - **Algoritmo di Dijkstra**

Algoritmo di Bellman-Ford

Distance Vector Protocol

- **Routing Table**
  - Per ogni destinazione sono memorizzati
  - Next Hop
  - Distanza (costo del cammino minimo)

Dest	Next	Dist

- **Router adiacenti si scambiano i Distance Vector**

DV=(destinazione, distanza)

  - Periodicamente
  - Dopo un cambio di stato

- **Ogni nodo determina per ogni destinazione il next-hop migliore**

Calcolo dei cammini minimi

Obiettivo

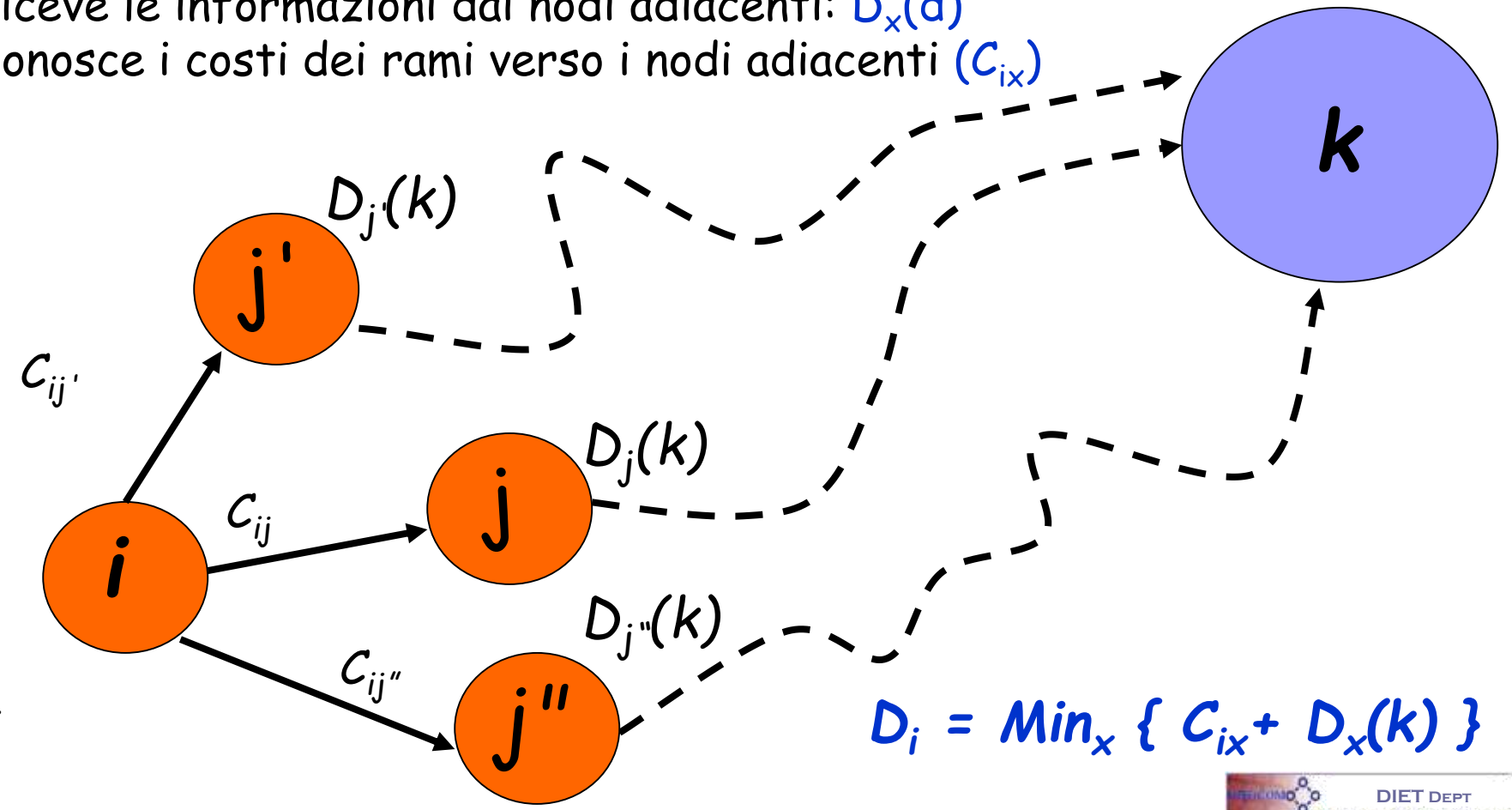
Calcolo del percorso minimo tra il nodo i ed un nodo di destinazione (es. k)

Se D_{jk} è la distanza minima dal nodo j e k e se j è il nodo adiacente a i che si trova sul percorso a costo minimo dal nodo i verso k, si ha

$$D_i = C_{ij} + D_j$$

Calcolo dei cammini minimi

Il nodo i:
- riceve le informazioni dai nodi adiacenti: $D_x(d)$
- conosce i costi dei rami verso i nodi adiacenti (C_{ix})



$$D_i = Min_x \{ C_{ix} + D_x(k) \}$$

Algoritmo con vettore distanza

- **Iterativo, asincrono**
- **Ogni iterazione locale è causata da:**
  - variazione del costo di uno dei link locali
  - ricezione da un nodo adiacente di un vettore distanza aggiornato
- **Distribuito**
  - Ogni nodo aggiorna i suoi vicini solo quando il suo DV cambia
  - Un router avvisa i nodi adiacenti solo se necessario

- **Ciascun Nodo**

*Quando **riceve** un messaggio del cambio del costo da parte del suo vicino*

*Effettua il **calcolo** del costo dei percorsi*

Se il DV cambia, invia la ***notifica*** ai suoi adiacenti

Algoritmo di Bellman-Ford

- ***Consideriamo il calcolo parallelo per tutte le destinazioni k***
- ***Inizializzazione***
  - Ogni nodo ha 1 riga per ogni destinazione *k*
  - La distanza tra il nodo k a se stesso è posta a zero: $D_k(k)=0$
  - La distanza del nodo k verso un altro nodo *j* è posta uguale ad infinito:

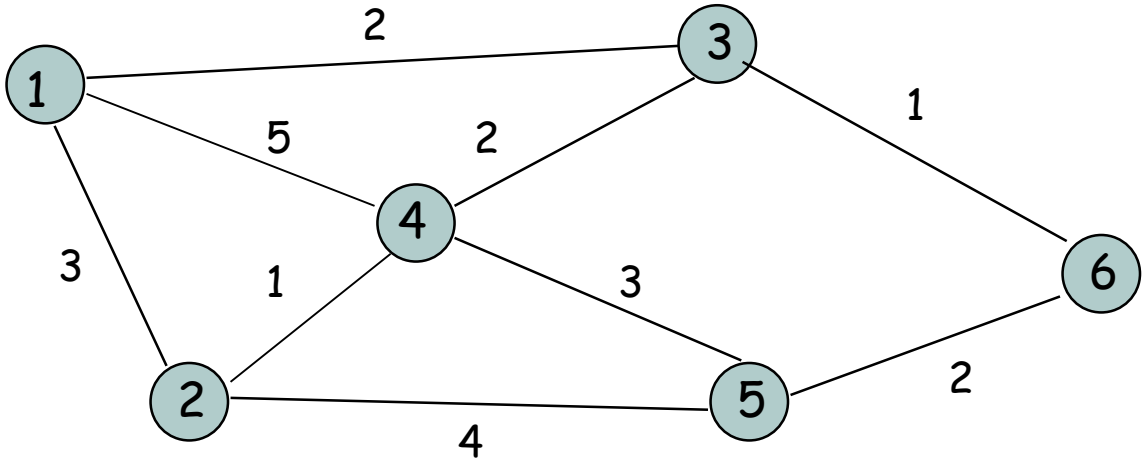
$$D_j(d)= \infty \text{ , for } j \neq k$$

- ***Passo di emissione***
  - Il nodo emette un nuovo distance vector verso i nodi adiacenti "j"
- ***Passo di ricezione***
  - Per ogni destinazione *k*, un nodo "i" calcola il next hop "j" che appartiene al percorso a minima distanza verso il nodo "k",
    - $D_i(k) = \min_j \{ C_{ij} + D_j(k) \}$
    - Si sostituisce il vecchio record $(n_j, D_i(k))$ con il nuovo record $(n_j^*, D_j^*(k))$

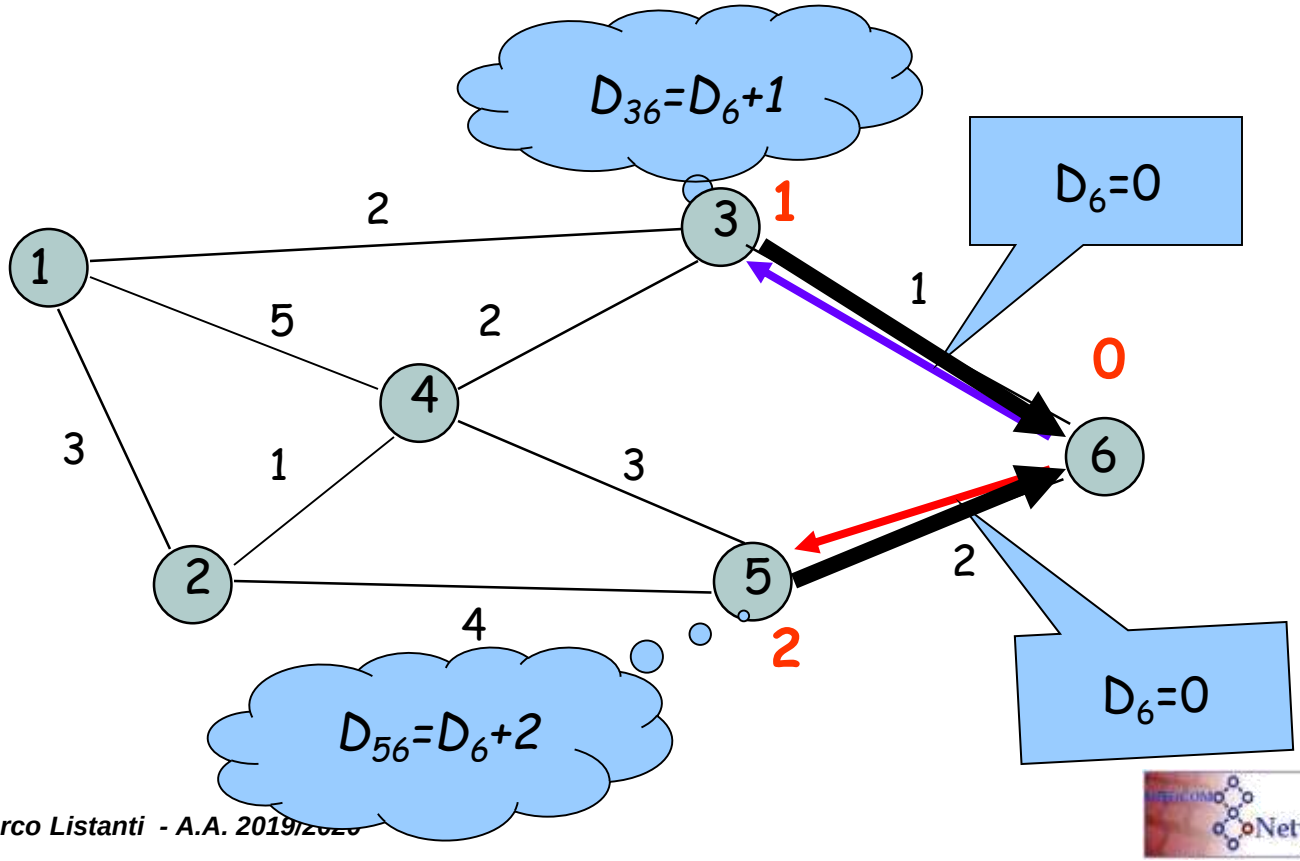
Iteration	Node 1	Node 2	Node 3	Node 4	Node 5
Initial	**(-1,∞)**	**(-1,∞)**	**(-1,∞)**	**(-1,∞)**	**(-1,∞)**
1					
2					
3					

Record della RT del nodo 1 per la destinazione 6

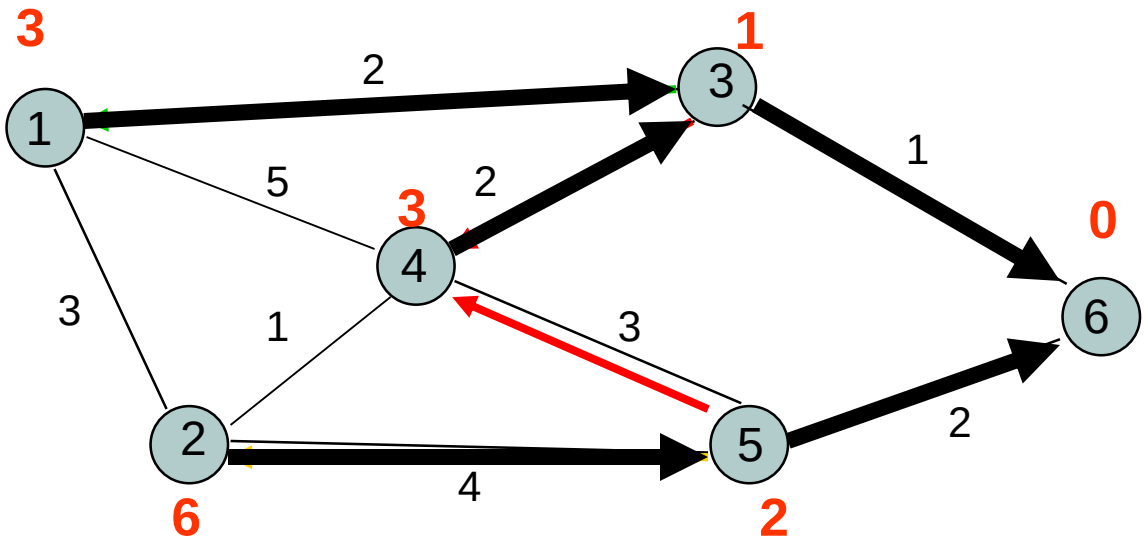
Record della RT del nodo 3 per la destinazione 6



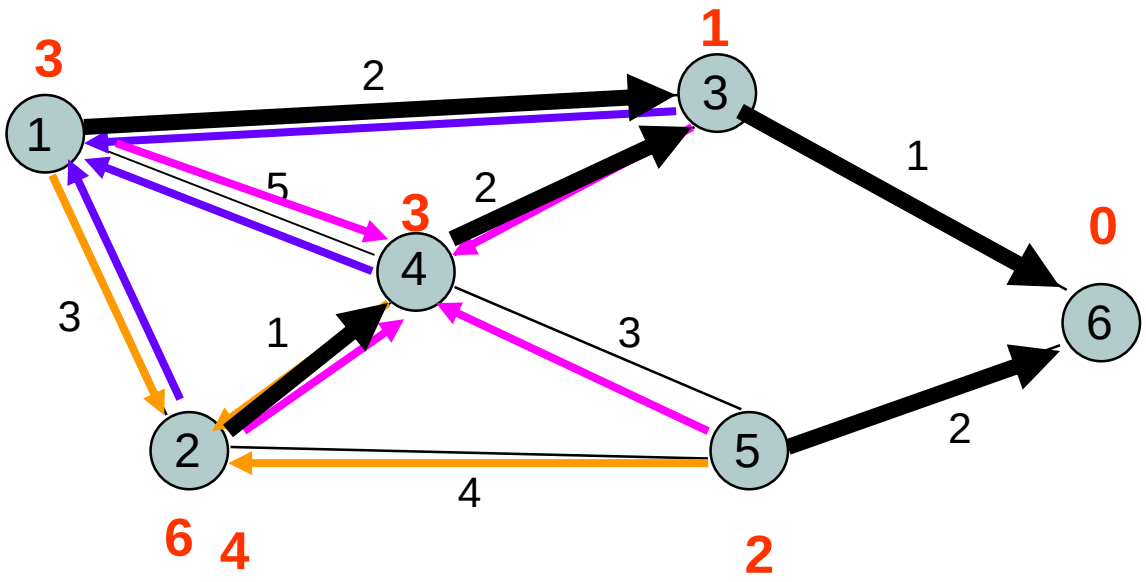
Iteration	Node 1	Node 2	Node 3	Node 4	Node 5
Initial	(-1,∞)	(-1,∞)	(-1,∞)	(-1,∞)	(-1,∞)
1	(-1,∞)	(-1,∞)	(6,1)	(-1,∞)	(6,2)
2					
3					



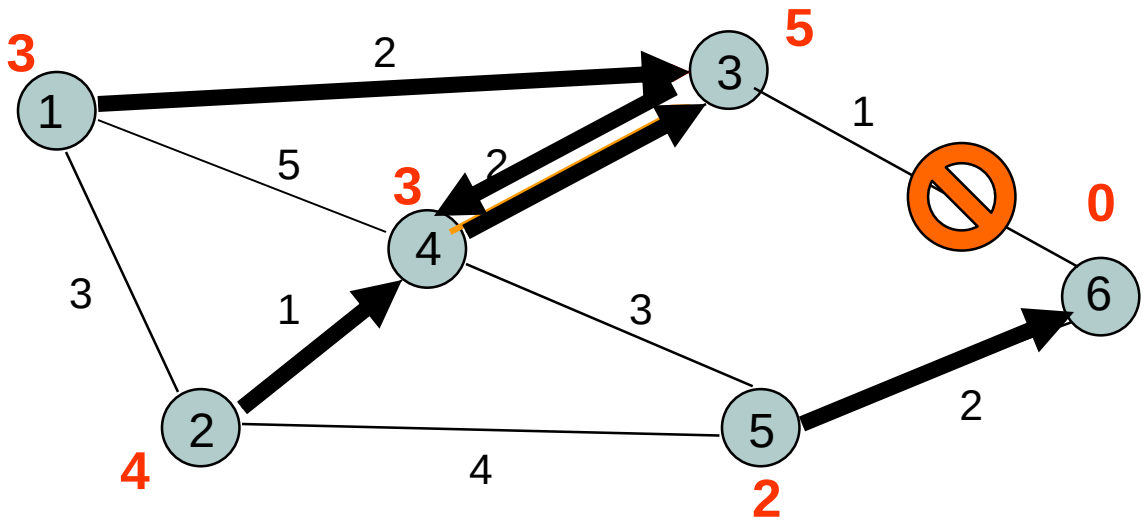
Iteration	Node 1	Node 2	Node 3	Node 4	Node 5
Initial	(-1,∞)	(-1,∞)	(-1,∞)	(-1,∞)	(-1,∞)
1	(-1,∞)	(-1,∞)	(6,1)	(-1,∞)	(6,2)
2	(3,3)	(5,6)	(6,1)	(3,3)	(6,2)
3					



Iteration	Node 1	Node 2	Node 3	Node 4	Node 5
Initial	(-1,∞)	(-1,∞)	(-1,∞)	(-1,∞)	(-1,∞)
1	(-1,∞)	(-1,∞)	(6,1)	(-1,∞)	(6,2)
2	(3,3)	(5,6)	(6,1)	(3,3)	(6,2)
3	(3,3)	(4,4)	(6,1)	(3,3)	(6,2)

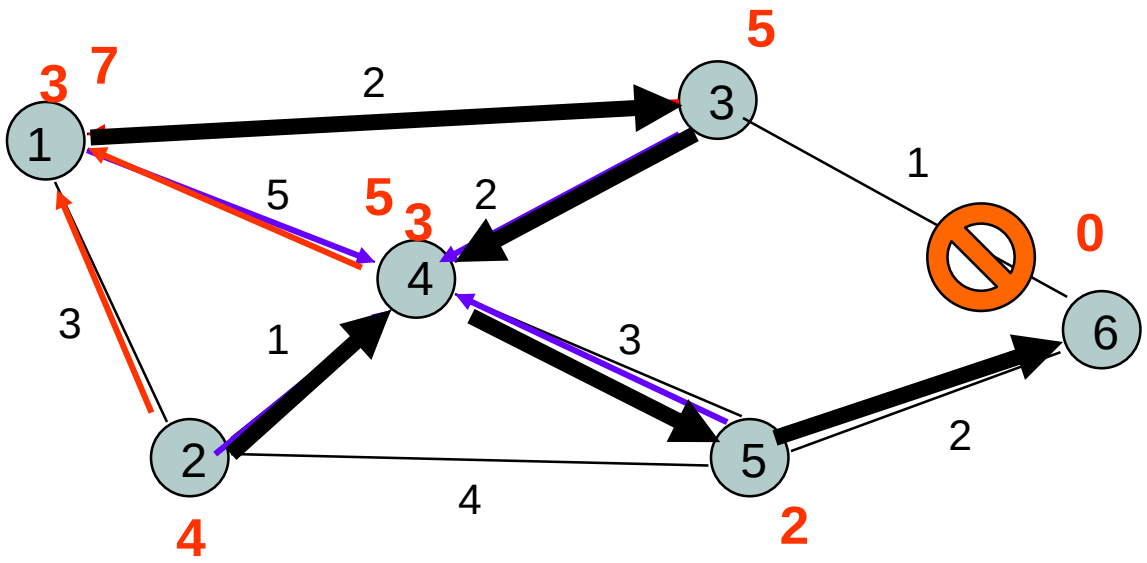


Iteration	Node 1	Node 2	Node 3	Node 4	Node 5
Initial	**(3,3)**	**(4,4)**	**(6,1)**	**(3,3)**	**(6,2)**
1	**(3,3)**	**(4,4)**	**(4,5)**	**(3,3)**	**(6,2)**
2					
3					

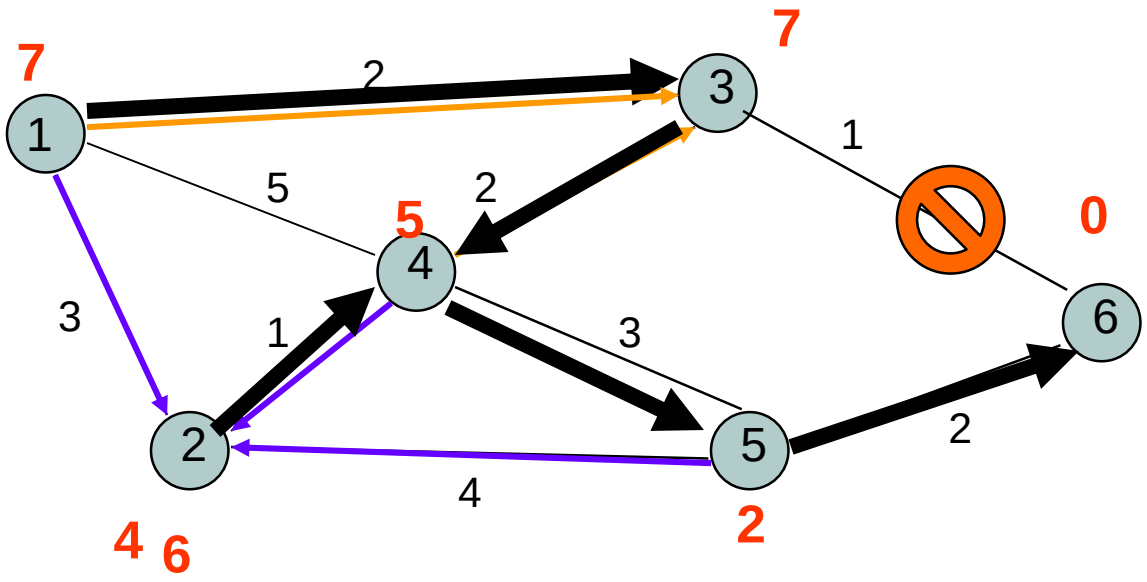


Rete disconnessa: si crea un loop tra i nodi 3 e 4

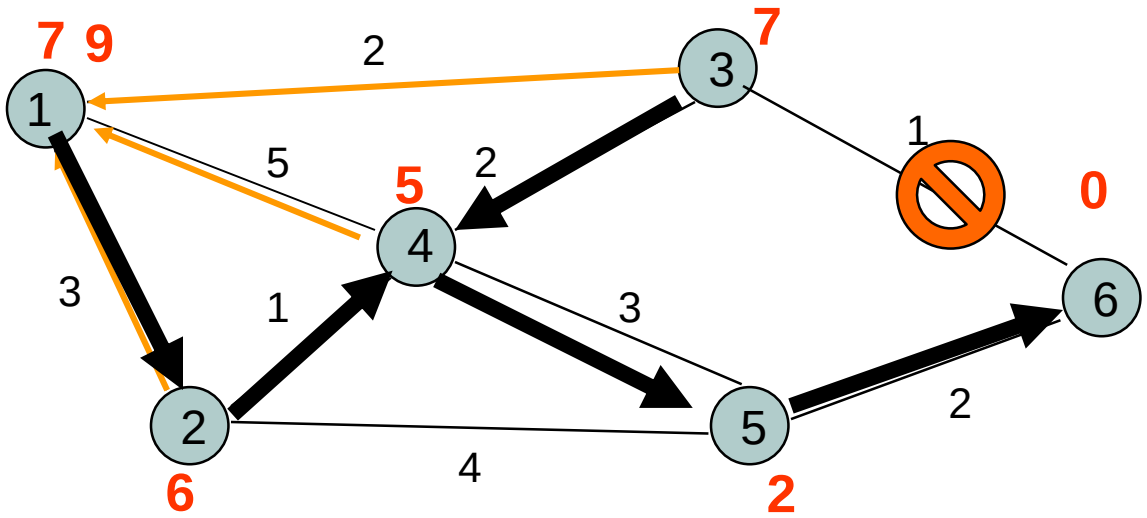
Iteration	Node 1	Node 2	Node 3	Node 4	Node 5
Initial	**(3,3)**	**(4,4)**	**(6,1)**	**(3,3)**	**(6,2)**
1	**(3,3)**	**(4,4)**	**(4,5)**	**(3,3)**	**(6,2)**
2	**(3,7)**	**(4,4)**	**(4,5)**	**(5,5)**	**(6,2)**
3					



Iteration	Node 1	Node 2	Node 3	Node 4	Node 5
Initial	**(3,3)**	**(4,4)**	**(6, 1)**	**(3,3)**	**(6,2)**
1	**(3,3)**	**(4,4)**	**(4, 5)**	**(3,3)**	**(6,2)**
2	**(3,7)**	**(4,4)**	**(4, 5)**	**(5,5)**	**(6,2)**
3	**(3,7)**	**(4,6)**	**(4,7)**	**(5,5)**	**(6,2)**

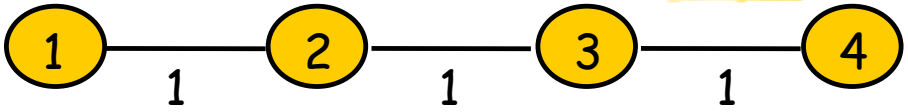


Iteration	Node 1	Node 2	Node 3	Node 4	Node 5
1	(3,3)	(4,4)	(4,5)	(3,3)	(6,2)
2	(3,7)	(4,4)	(4,5)	(2,5)	(6,2)
3	(3,7)	(4,6)	(4,7)	(5,5)	(6,2)
4	(2,9)	(4,6)	(4,7)	(5,5)	(6,2)

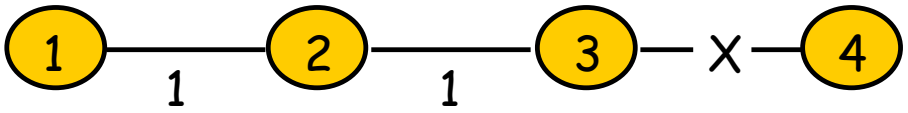


Conteggio all'infinito

(a)



(b)



I nodi credono che il esista un cammmino in realtà non disponibile

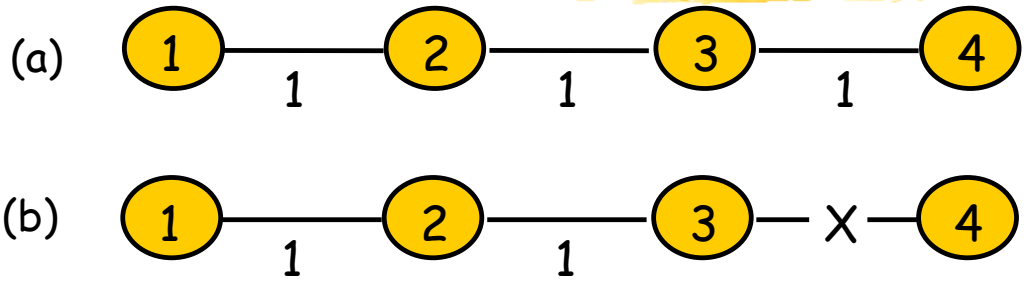
Destinazione nodo 4

Passo	Nodo 1	Nodo 2	Nodo 3
Prima del guasto	(2,3)	(3,2)	(4,1)
Dopo il guasto	(2,3)	(3,2)	(2,3)
1	(2,3)	(3,4)	(2,3)
2	(2,5)	(3,4)	(2,5)
3	(2,5)	(3,6)	(2,5)
4	(2,7)	(3,6)	(2,7)
5	(2,7)	(3,8)	(2,7)
...	...	...	...

Soluzioni al conteggio all'infinito

- **Split Horizon**
  - Un router non trasmette il proprio DV aggiornato verso il router da cui ha ricevuto l'aggiornamento
- **Poisoned Reverse**
  - Un router trasmette il proprio DV aggiornato anche verso il router da cui ha ricevuto l'aggiornamento, ma indicando per la distanza aggiornata al valore ∞
  - Si interrompe immediatamente il loop di conteggio
  - Questa soluzione non funziona in caso di loop più complessi

Split Horizon con Poisoned Reverse



Destinazione nodo 4

Update	Nodo 1	Nodo 2	Nodo 3	Eventi
Prima del guasto	(2, 3)	(3, 2)	(4, 1)	
Dopo il guasto	(2, 3)	(3, 2)	(-1, ∞)	Nodo 2 annuncia al nodo 3 il suo cammino verso il nodo 4 con distanza infinita; Il nodo 3 deduce che non esiste un cammino verso il nodo 4
1	(2, 3)	(-1, ∞)	(-1, ∞)	Nodo 1 annuncia al nodo 2 il suo cammino verso il nodo 4 con distanza infinita Il nodo 2 deduce che non esiste un cammino verso il nodo 4
2	(-1, ∞)	(-1, ∞)	(-1, ∞)	Il nodo 1 deduce che non esiste un cammino verso il nodo 4

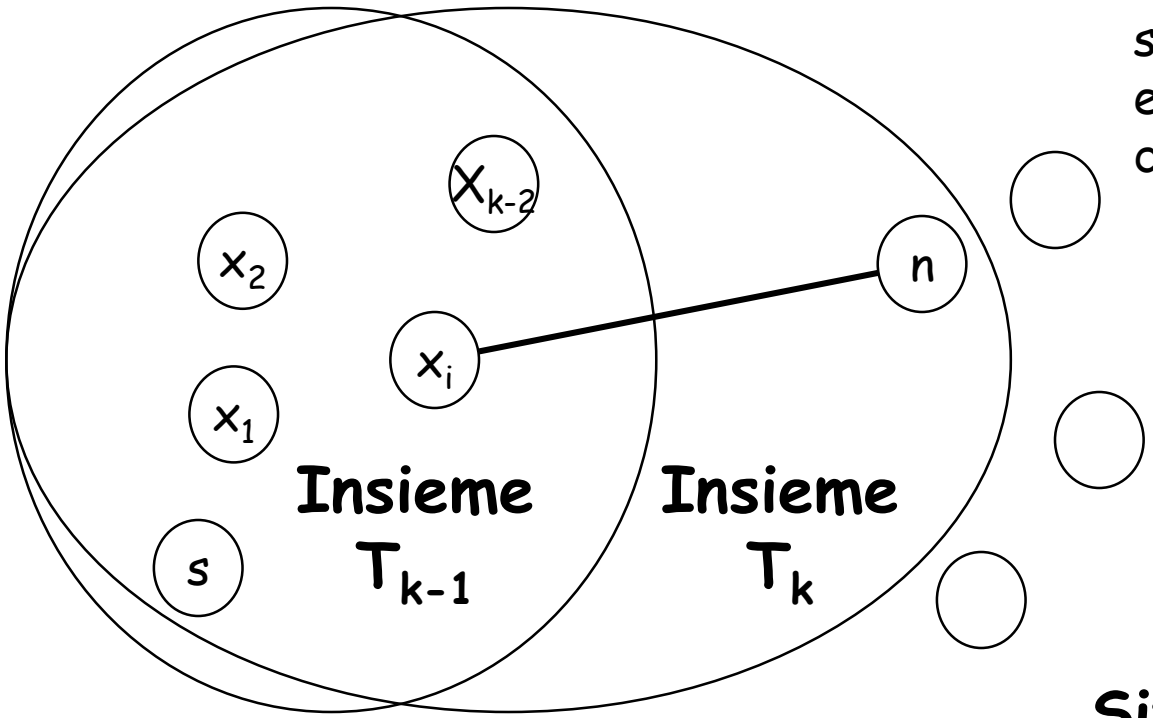
Algoritmo di Djikstra

Algoritmo di Dijkstra

- **Individua il cammino a lunghezza minima tra un nodo s e tutti gli altri nodi di un grafo G (spanning tree) procedendo in modo da aumentare progressivamente la distanza**
- **L'algoritmo procede a passi successivi**
  - al passo k-mo sono individuati i k nodi raggiungibili dal nodo sorgente tramite i cammini a costo più basso
  - tali k nodi formano l'insieme T_k
  - al passo k+1-mo si individua il nodo n che è caratterizzato dal cammino dal costo più basso dal nodo s che transita esclusivamente nei nodi dell'insieme T_k
  - viene formato l'insieme T_{k+1} aggiungendo il nodo n all'insieme T_k
  - l'algoritmo termina quando sono stati aggiunti all'insieme T_k tutti i nodi del grafo

Algoritmo di Dijkstra

Al passo k viene aggiunto all'insieme T_{k-1} il nodo n caratterizzato dal cammino a costo minimo con il nodo sorgente s che transita esclusivamente in nodi dell'insieme T_{k-1}



Situazione al passo k

Algoritmo di Dijkstra

- **Notazioni**
  - **N** : insieme dei nodi del grafo
  - **s** : nodo sorgente
  - T_k : insieme dei nodi raggiunti dall'algoritmo al passo k
  - **c(i,j)** : peso (costo) del ramo (i,j)
    - $c(i,i) = 0$
    - $c(i,j) \geq 0$ se i vertici i e j sono connessi direttamente
    - $c(i,j) = \infty$ se i vertici i e j non sono connessi direttamente
  - $L_k(n)$: costo del cammino minimo, individuato dall'algoritmo fino al passo k, tra il nodo s ed un generico nodo n

Algoritmo di Dijkstra

- **Inizializzazione (k=1)**
  - $T_1 = \{s\}$
  - $L_1(n) = c(s,n)$ per $n \neq s$
- **Aggiunta di un nodo (passo $1 \leq k \leq N$)**
  - trovare $x \notin T_{k-1}$ tale che

$$L_{k-1}(x) = \min_{j \notin T_{k-1}} \{L_{k-1}(j)\}$$

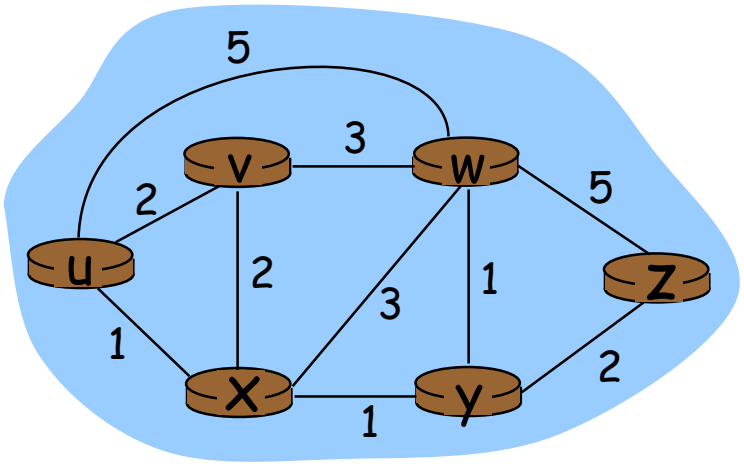
  - aggiungere all'insieme T_{k-1} il nodo x ed il ramo incidente a x
- **Aggiornamento del cammini minimi**
  - $L_k(n) = \min [L_{k-1}(n), L_{k-1}(x) + c(x,n)]$ per tutti i valori di $n \notin T_k$

Algoritmo di Dijkstra

- **Al termine**
  - l'insieme T_N è uno spanning tree del grafo di partenza contenente i cammini a costo minimo tra il nodo sorgente e tutti gli altri nodi del grafo
  - $L_N(n)$ indica il costo del cammino a costo minimo tra il nodo s ed il nodo n
- **Si noti che**
  - al passo k-mo viene aggiunto all'insieme T_{k-1} il k-mo nodo ed è individuato il cammino a costo minimo tra il tale nodo ed il nodo sorgente
  - questo cammino transita esclusivamente attraverso i nodi sinora compresi nell'insieme T_{k-1}
- **La complessità dell'algoritmo è $o(N^2)$**

Algoritmo di Dijkstra: esempio

passo	T_k	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	2,u	5,u	1,u	∞	∞
1	ux	2,u	4,x		2,x	∞
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					

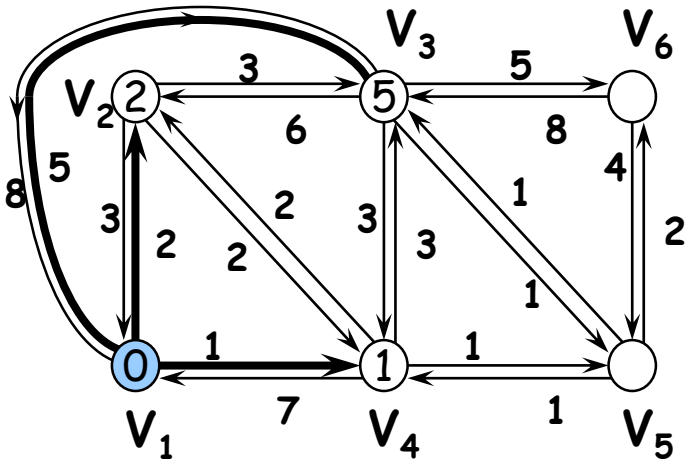


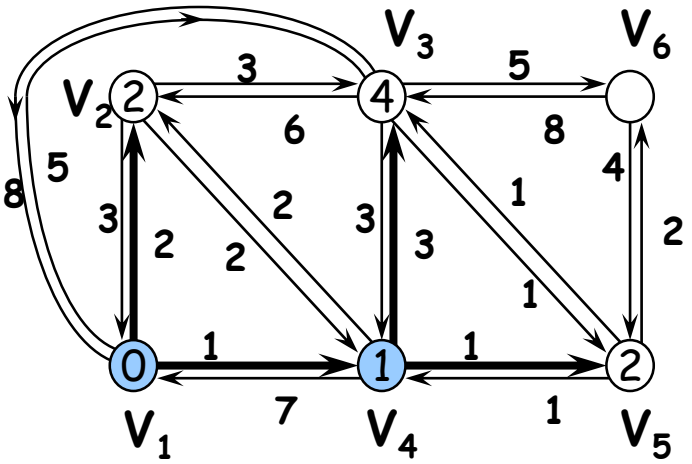
Percorsi a costo minimo tra il nodo "u" e tutti gli altri nodi

D(i) = costo minimo per percorso tra il nodo di origine ed il nodo "i"

p(i) = predecessore del nodo "i" sul percorso minimo tra in nodo origine il nodo "i"

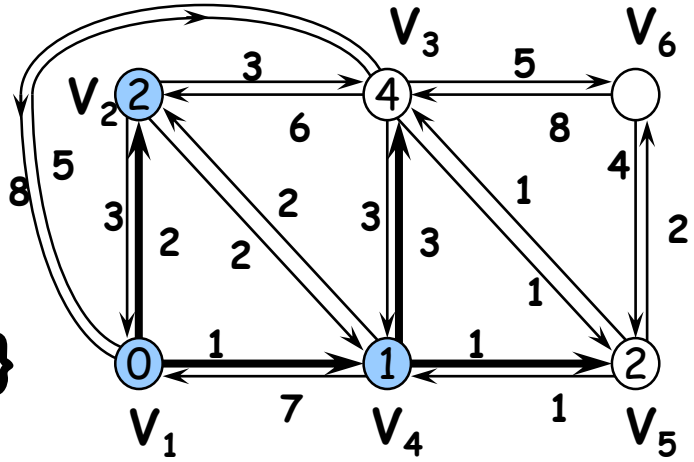
Esempio Dijkstra Algorithm 1(2)





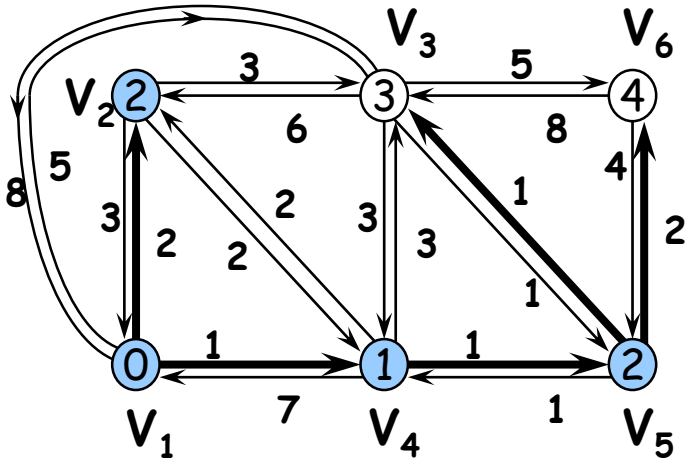
$T_1 = \{1\}$

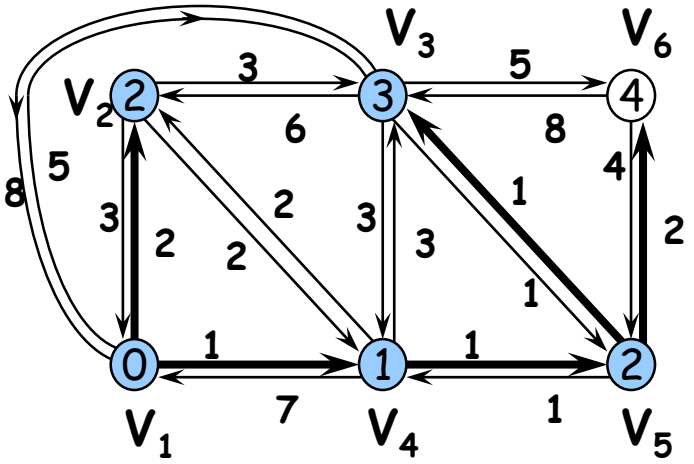
$T_2 = \{1,4\}$



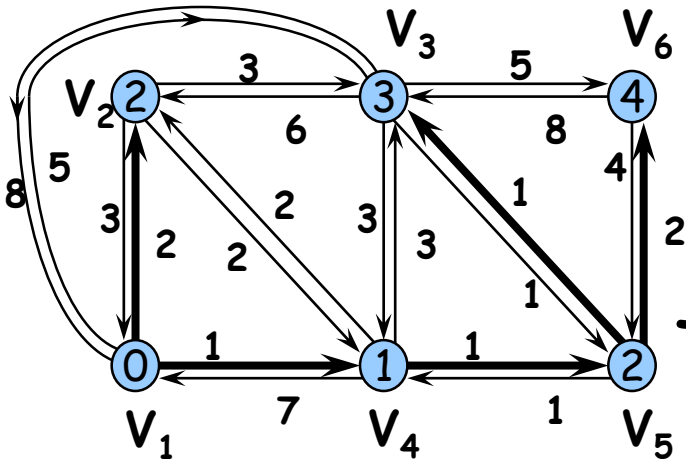
$T_3 = \{1,2,4\}$

Esempio Dijkstra Algorithm 2(2)





$T_4 = \{1,2,4,5\}$



$T_5 = \{1,2,3,4,5\}$

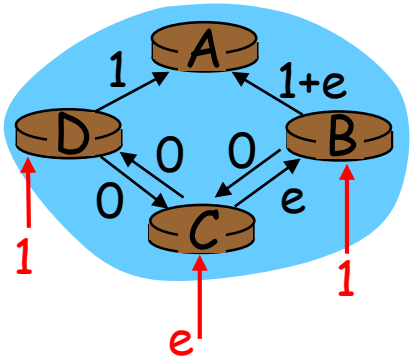
$T_6 = \{1,2,3,4,5,6\}$

Algoritmo di Dijkstra

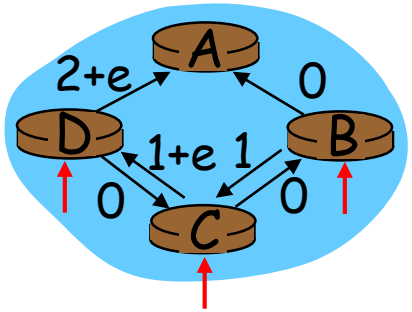
- **Complessità dell'algoritmo (n nodi)**
  - Ciascuna iterazione: controllo su tutti i nodi, w, non in N

    $n(n+1)/2 \rightarrow O(n^2)$
- **La più efficiente implementazione possibile**

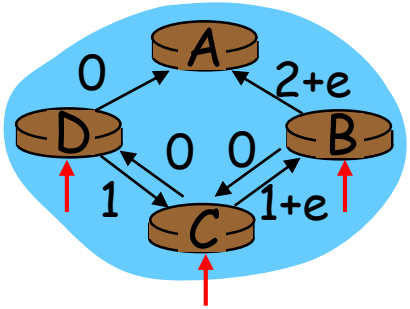
  $O(n \log n)$
- **Possibili oscillazioni**
- **Es. costo del collegamento = quantità di traffico trasportato**



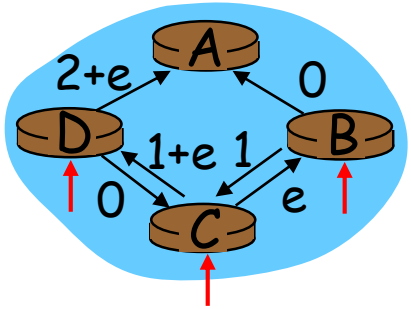
instradamento iniziale



… oscillazione



… oscillazione



… oscillazione

Instradamento gerarchico

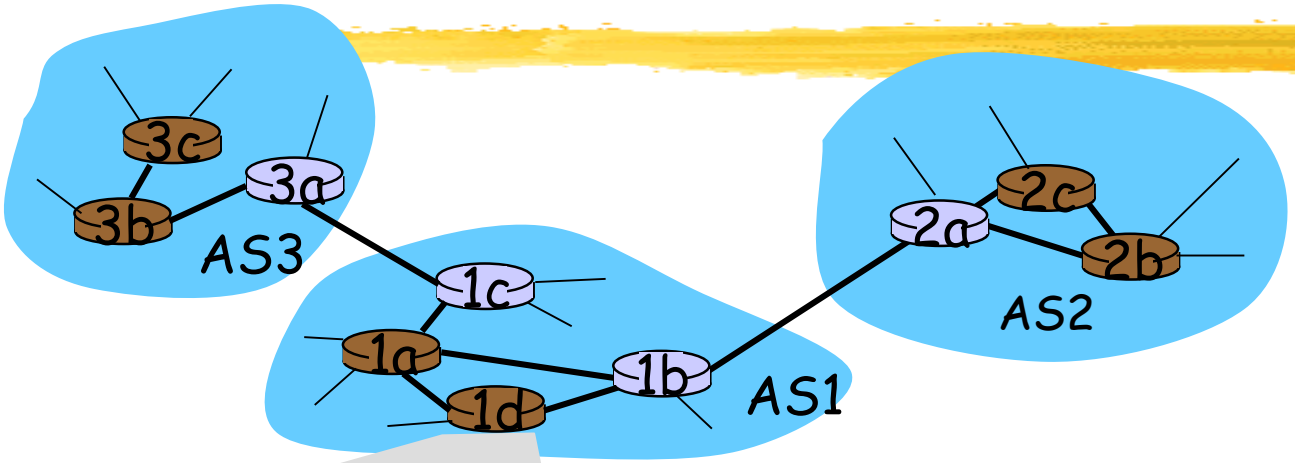
Instradamento gerarchico

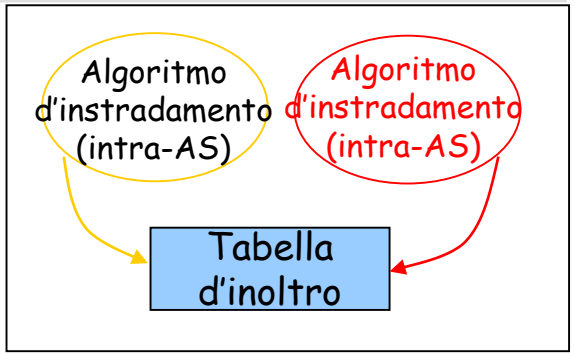
- **Ipotesi utilizzate**
  - Ciascun router era indistinguibile dagli altri
  - Visione omogenea della rete
- **Problemi**
- **Autonomia amministrativa**
  - Internet: rete di reti
  - Ogni ISP deve essere in grado di amministrare la propria rete nel modo desiderato, pur mantenendo la possibilità di connetterla alle reti esterne
- **Scalabilità**
  - 200 milioni di destinazioni
  - Archiviare le informazioni d'instradamento su ciascun host richiederebbe un'enorme quantità di memoria
  - Il traffico generato dagli aggiornamenti LS non lascerebbero banda per i pacchetti di dati

Instradamento gerarchico

- **Organizzazione di router in sistemi autonomi (AS, Autonomous System).**
- **I router di un AS eseguono lo stesso algoritmo d'instradamento**
  - Protocollo d'instradamento interno al sistema autonomo (intra-AS) (IGP)
  - I router appartenenti a differenti AS possono eseguire protocolli d'instradamento inter-AS
- **Router gateway**
  - Hanno il compito aggiuntivo d'inoltrare pacchetti a destinazioni esterne ad un AS

Sistemi autonomi interconnessi

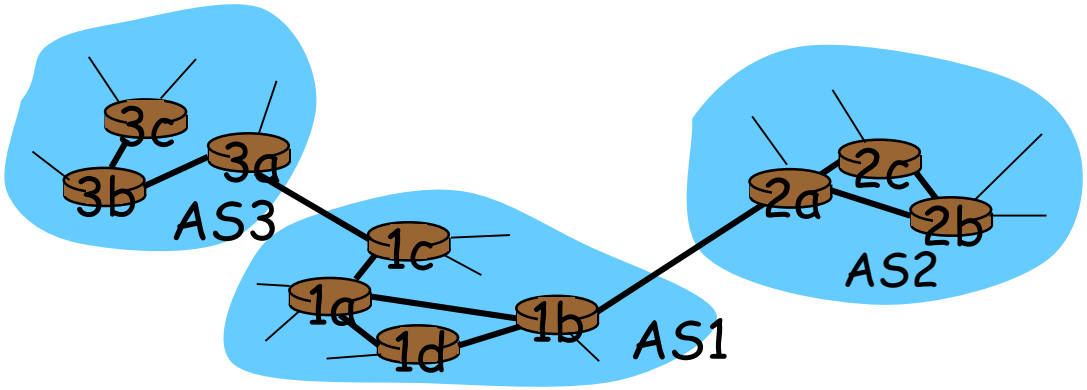




- **Ciascun router interno ad un AS sa come inoltrare pacchetti lungo il percorso ottimo verso qualsiasi destinazione interna**
  - I sistemi AS2 e AS3 hanno tre router ciascuno
  - I protocolli d'instradamento dei tre sistemi autonomi non sono necessariamente gli stessi
  - I router 1b, 1c, 2a e 3a sono **Gateway**

Instradamento tra sistemi autonomi

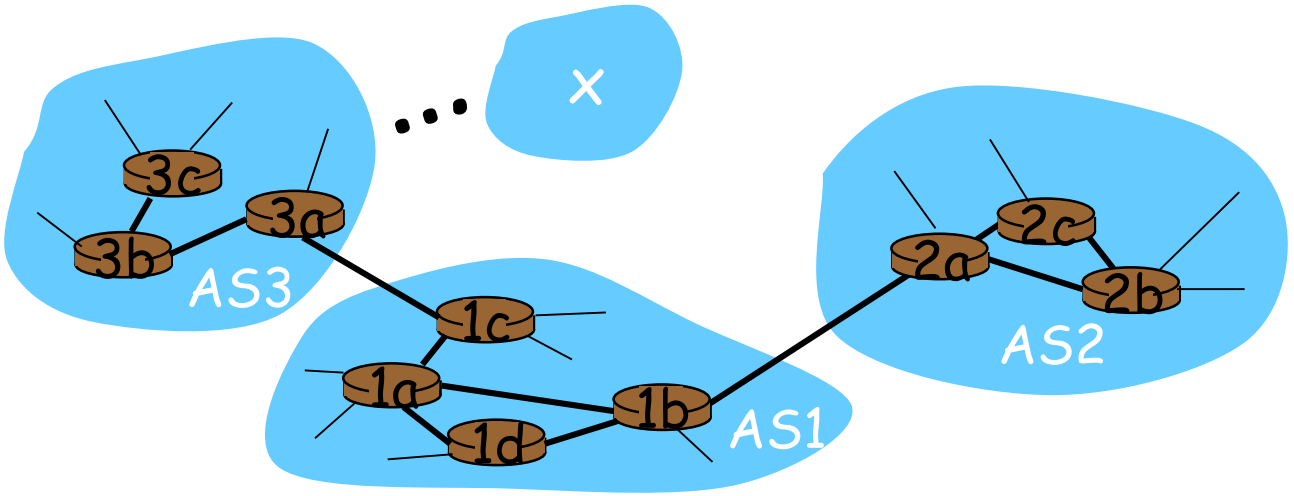
- **Un router in AS1 riceve un pacchetto con destinazione esterna a AS1**
  - Il router dovrebbe inoltrare il pacchetto verso uno dei due gateway
  - Quale ?



- **AS1 deve**
  - Sapere quali destinazioni sono raggiungibili attraverso AS2 e quali attraverso AS3
  - Informare tutti i router all'interno dell'AS in modo che ciascuno possa configurare la propria tabella d'inoltro per gestire destinazioni esterne

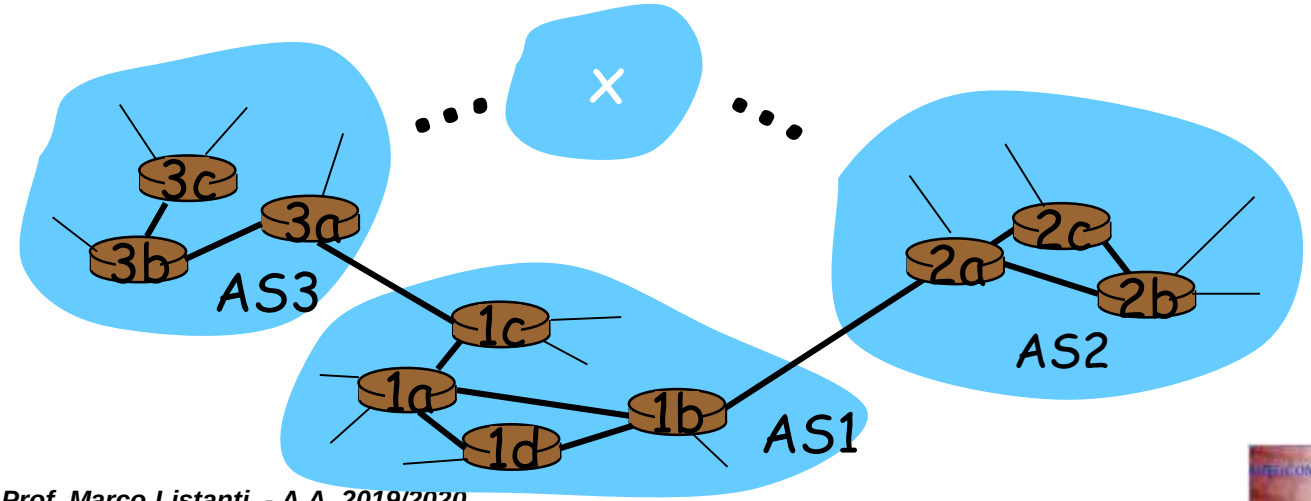
Esempio: Gateway unico

- AS1 apprende dal proprio protocollo d'instradamento inter-AS (EGP) che la sottorete x è raggiungibile da AS3 (gateway 1c), ma non da AS2
- Il protocollo inter-AS (EGP) propaga questa informazione a tutti i propri router
- Il router 1d determina, partendo dall'informazione fornita dal protocollo intra-AS (IGP), l'interfaccia I del router sul percorso a costo minimo dal router 1d al gateway 1c.
- Il router 1d può inserire la riga (x,I) nella propria tabella d'inoltro.



Esempio: Gateway multiplo

- Supponiamo che AS1 apprenda dal protocollo d'instradamento tra sistemi autonomi che la sottorete x è raggiungibile da AS2 e da AS3
- Al fine di configurare la propria tabella d'inoltro, il router 1d deve determinare a quale gateway, 1b o 1c, indirizzare i pacchetti destinati alla sottorete x.
- **Instradamento a hot potato**
  - Si sceglie il percorso più breve di uscita dalla rete
  - Si sceglie il gateway con percorso a costo minimo (protocollo IGP)



Esempio: Gateway multiplo (Hot potato)

Dal protocollo inter-AS si apprende che la sottorete x è raggiungibile attraverso più gateway

Si usa l'informazione d'instradamento proveniente dal protocollo intra-AS per determinare i costi dei percorsi a costo minimo verso i gateway

Instradamento "hot potato": si sceglie il gateway che ha il costo minimo inferiore

Della tabella di routing si determina l'interfaccia I che conduce al gateway a costo minimo. Si scrive (x,I) nella tabella di routing