ROMA TRE
UNIVERSITÀ DEGLI STUDI

INGEGNERIA
CIVILE, INFORMATICA E DELLE TECNOLOGIE AERONAUTICHE

Internet and Data Centers

algoritmi link state packet

G. Di Battista, M. Patrignani

copyright notice

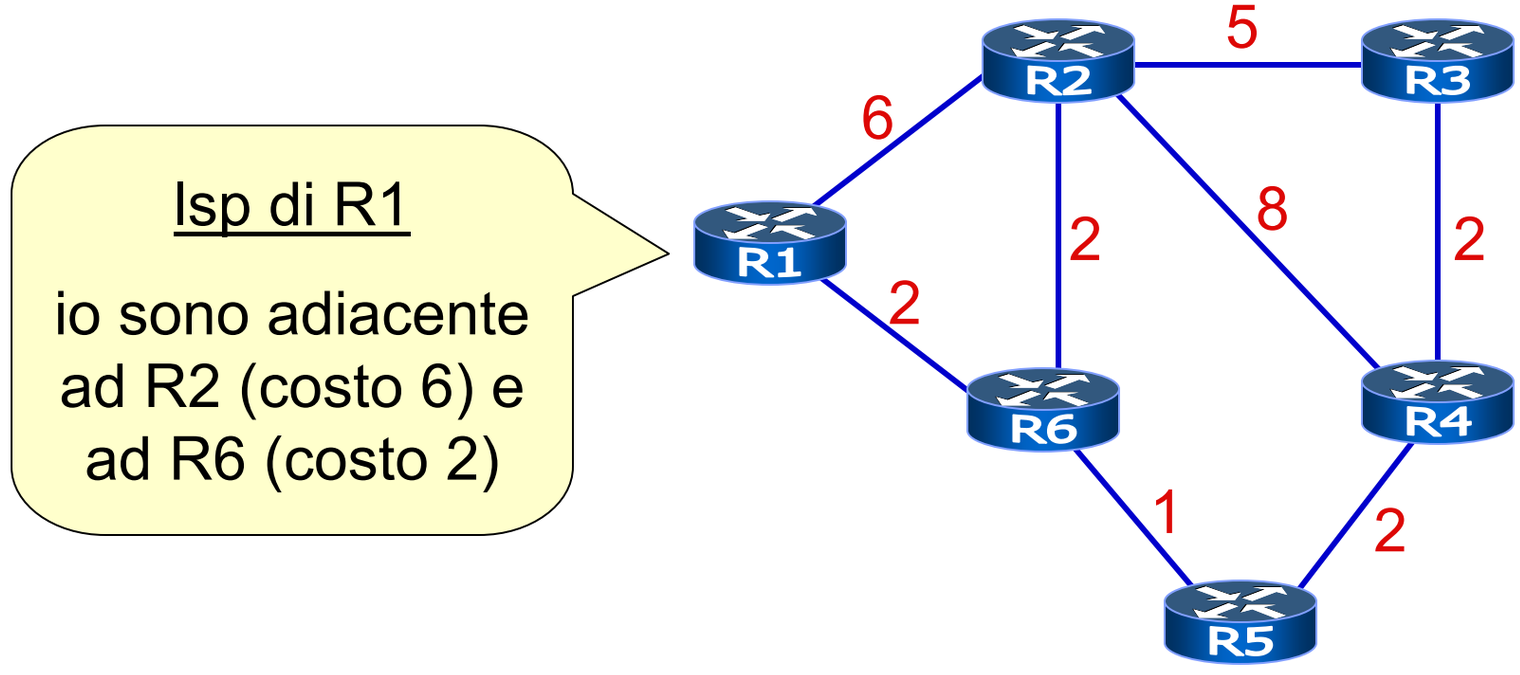
- all the pages/slides in this presentation, including but not limited to, images, photos, animations, videos, sounds, music, and text (hereby referred to as “material”) are protected by copyright
- this material, with the exception of some multimedia elements licensed by other organizations, is property of the authors and/or organizations appearing in the first slide
- this material, or its parts, can be reproduced and used for didactical purposes within universities and schools, provided that this happens for non-profit purposes
- any other use is prohibited, unless explicitly authorized by the authors on the basis of an explicit agreement
- this copyright notice must always be redistributed together with the material, or its portions

algoritmi link state packet

- ogni intermediate system (is) ha una mappa completa della rete
- ogni is calcola sulla propria mappa l'instradamento ottimale verso ogni destinazione
  - cammini di costo minimo
  - algoritmo di Dijkstra
- la tabella di instradamento di un is si ottiene considerando il primo hop dei cammini minimi
- la mappa della rete viene costruita usando pacchetti speciali, chiamati *link state packet* (lsp)
  - un lsp contiene informazioni sui nodi e sui link adiacenti ad uno specifico is
  - i lsp sono trasmessi in *selective flooding* da ogni is a tutti gli altri is della rete

esempio

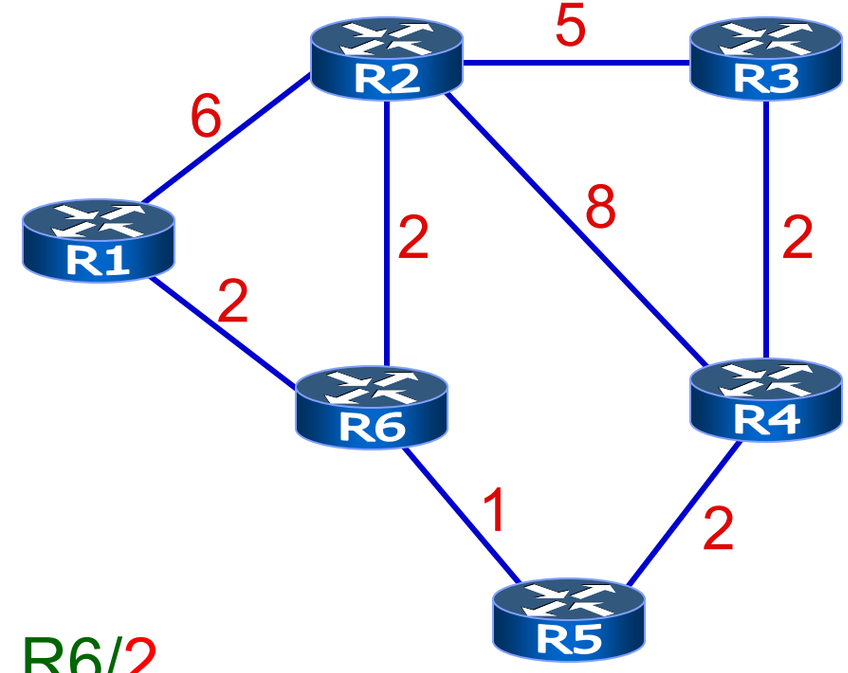
- ogni is trasmette un lsp (che arriva a tutti)



- ogni is conserva in un database il lsp più *recente* di ogni altro is della rete

esempio

- è importante che il database dei lsp sia identico su tutti gli is



- lsp database (su ogni is)
  - R1: sono adiacente a R2/6 R6/2
  - R2: sono adiacente a R1/6 R3/5 R4/8 R6/2
  - R3: sono adiacente a R2/5 R4/2
  - R4: sono adiacente a R2/8 R3/2 R5/2
  - R5: sono adiacente a R4/2 R6/1
  - R6: sono adiacente a R1/2 R2/2 R5/1

distance vector e link state packet

- differenza tra distance vector e link state packet
  - negli algoritmi distance vector la collaborazione tra is ha l'obiettivo di calcolare direttamente le tabelle di instradamento
  - negli algoritmi link state packet la collaborazione tra is ha l'obiettivo di mantenere aggiornata la mappa della rete
- differentemente dagli algoritmi distance vector, gli algoritmi link state packet
  - possono gestire reti anche con 10.000 nodi
  - convergono rapidamente

algoritmo di Dijkstra

- V = insieme di vertici numerati 1, …, n
  - il vertice 1 è la sorgente del traffico
- gli archi sono orientati
- A(i) = adiacenti di i
  - insieme dei vertici j per cui c'è un arco orientato (i,j)
- a_{ij} (≥ 0) = metrica dell'arco (i,j)
  - $a_{ij} = \infty$ se l'arco (i,j) è assente
  - la lunghezza di un cammino è somma delle metriche degli archi attraversati
- d[i] = distanza corrente dal vertice 1 al vertice i
- m(i) = distanza minima dal vertice 1 al vertice i
- S = insieme dei vertici i cui d[i] = m(i)

algoritmo di Dijkstra

- calcola il cammino di costo minimo tra il nodo 1 ed ogni altro nodo della rete
- i nodi sono inseriti in S per valori crescenti della distanza da 1

a) S = {1}

b) per ogni i tra 2 ed n poni d[i] = a_{1i}

c) finché V-S non è vuoto

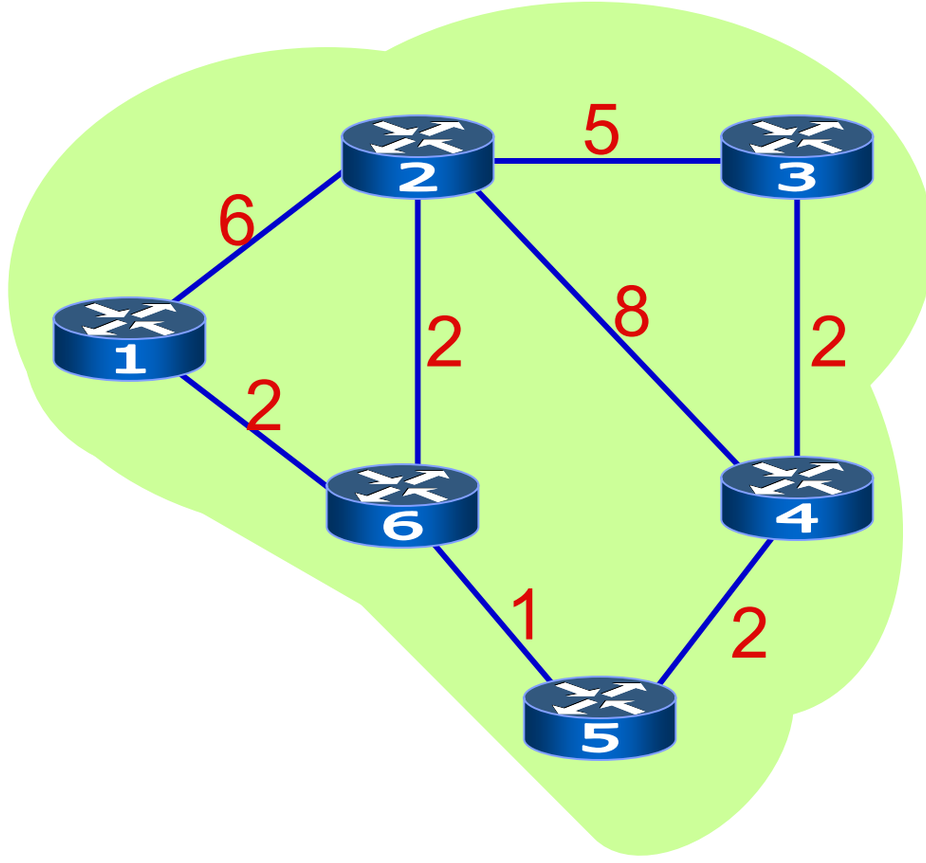
scegli un nodo i in V-S tale che d[i] sia minimo

aggiungi i ad S

per ogni j in A(i) poni d[j] = min(d[j],d[i] + a_{ij})

esempio

S = V-S =



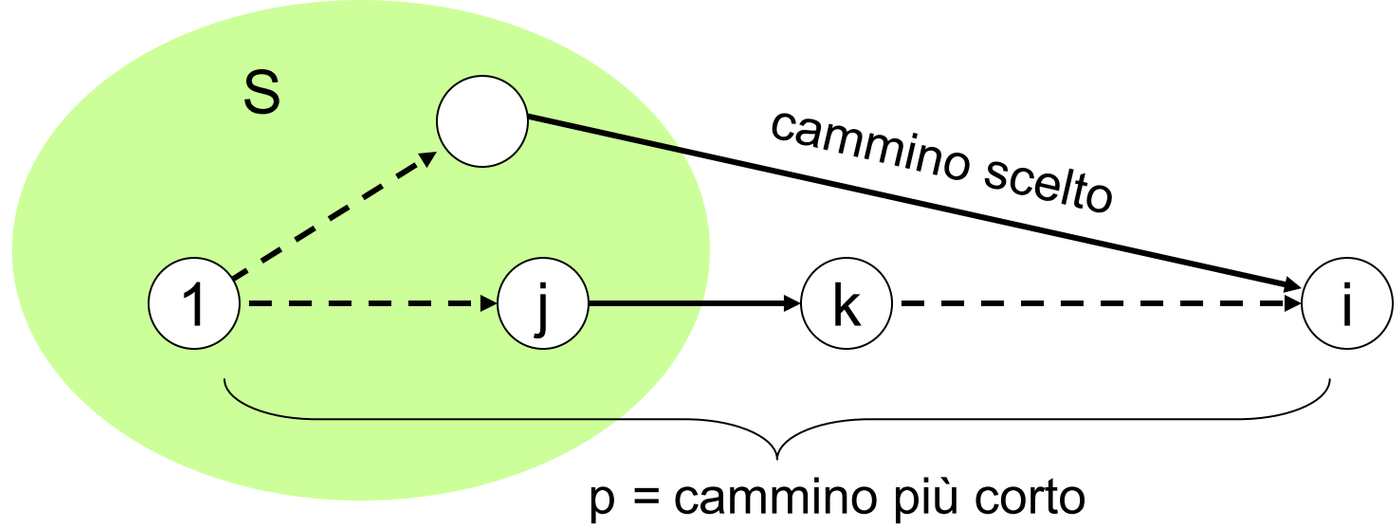
	1	2	3	4	5	6
d =	0	6	∞	∞	∞	2
d =	0	4	∞	∞	3	2
d =	0	4	∞	5	3	2
d =	0	4	9	5	3	2
d =	0	4	7	5	3	2
d =	0	4	7	5	3	2

lemma dell'invariante

- se d[v] = m(v) per ogni nodo v di S, allora quando i viene aggiunto ad S, anche per i si ha d[i] = m(i)
- dimostrazione per assurdo
  - supponiamo per assurdo che d[i] > m(i)
  - sia p il cammino minimo da 1 ad i
    - necessariamente |p| = m(i) < d[i]

dimostrazione per assurdo

- sia j l'ultimo nodo di p appartenente ad S e k il nodo seguente
  - k ≠ i altrimenti l'algoritmo avrebbe computato d[i] = |p| = m(i)



dimostrazione del lemma dell'invariante

1. d[k] = m(k), cioè d[k] è ottimo
   - infatti d[k] è calcolato da d[j]
     - per ipotesi d[j] = m(j)
     - l'arco (j,k) è utilizzato da p
2. m(k) ≤ m(i) perché k è un nodo intermedio di p

- concatenando 1., 2. e l'ipotesi assurda si ottiene
  - d[k] = m(k) ≤ m(i) < d[i]
  - cioè d[k] < d[i]
- ma questo è assurdo perché l'algoritmo avrebbe scelto k e non i

correttezza dell'algoritmo di Dijkstra

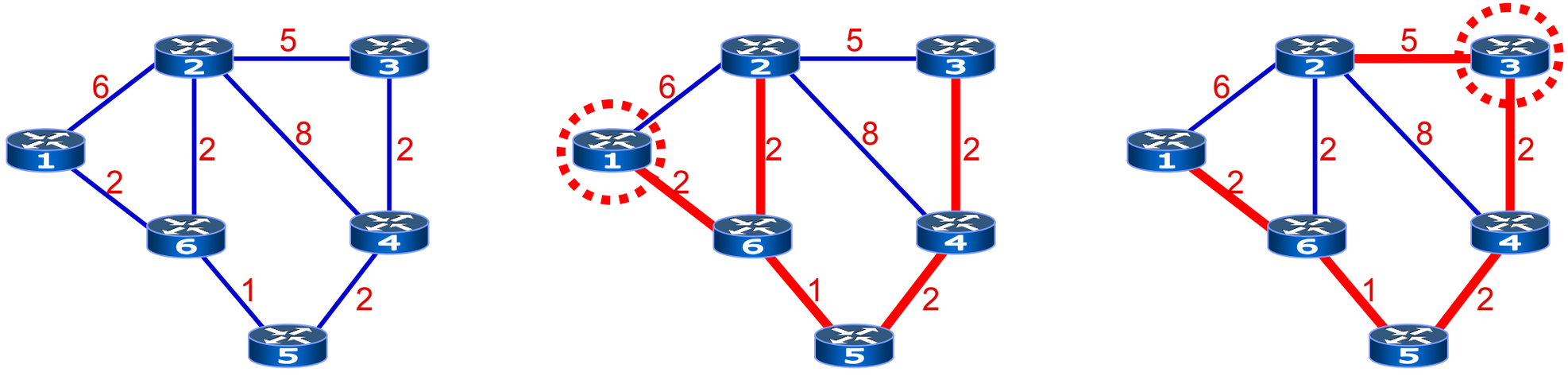
- l'algoritmo di Dijkstra trova un cammino di costo minimo tra 1 ed ogni altro nodo della rete
  - inizializzazione
    - è vero quando S = {1}
  - conservazione
    - per il lemma dell'invariante, rimane vero quando aggiungo ogni nodo ad S
  - conclusione
    - è vero quando S = V

Dijkstra – efficienza (lavoro svolto dai router)

- supponiamo che in una rete ci siano n nodi e m link
- un'implementazione semplice richiede tempo $O(n^2 + m) = O(n^2)$
- un'implementazione più sofisticata ha complessità $O(n \log n + m)$
  - complessità ammortizzata
  - richiede l'implementazione di una coda di priorità con un heap

shortest-path spanning tree

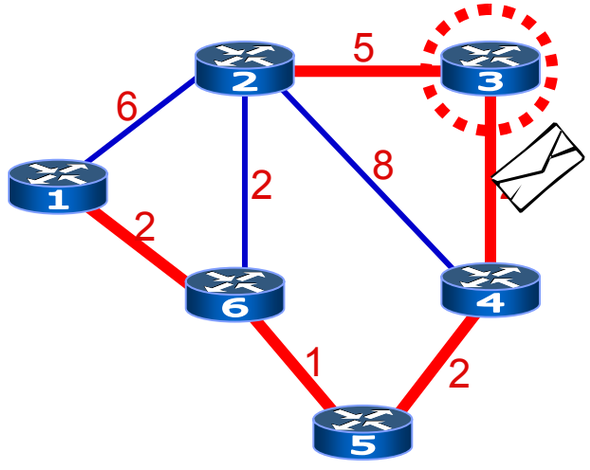
- le distanze minime calcolate corrispondono ad uno shortest-path spanning tree della rete

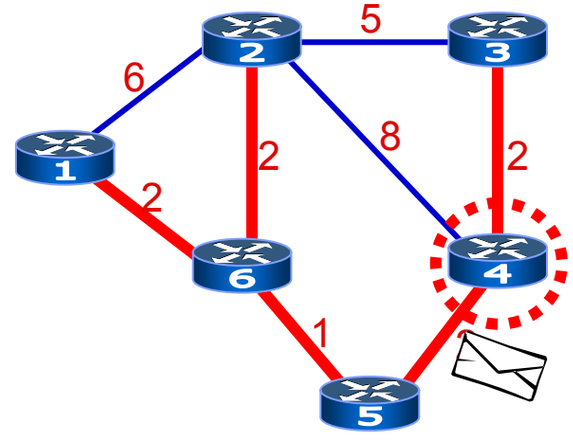


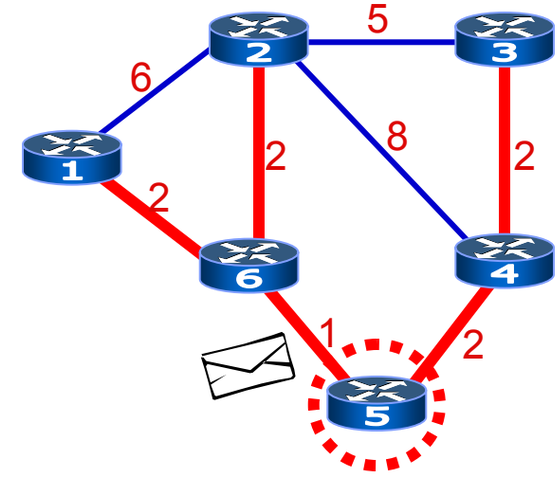
- in generale gli spanning tree sono diversi

shortest-path spanning tree

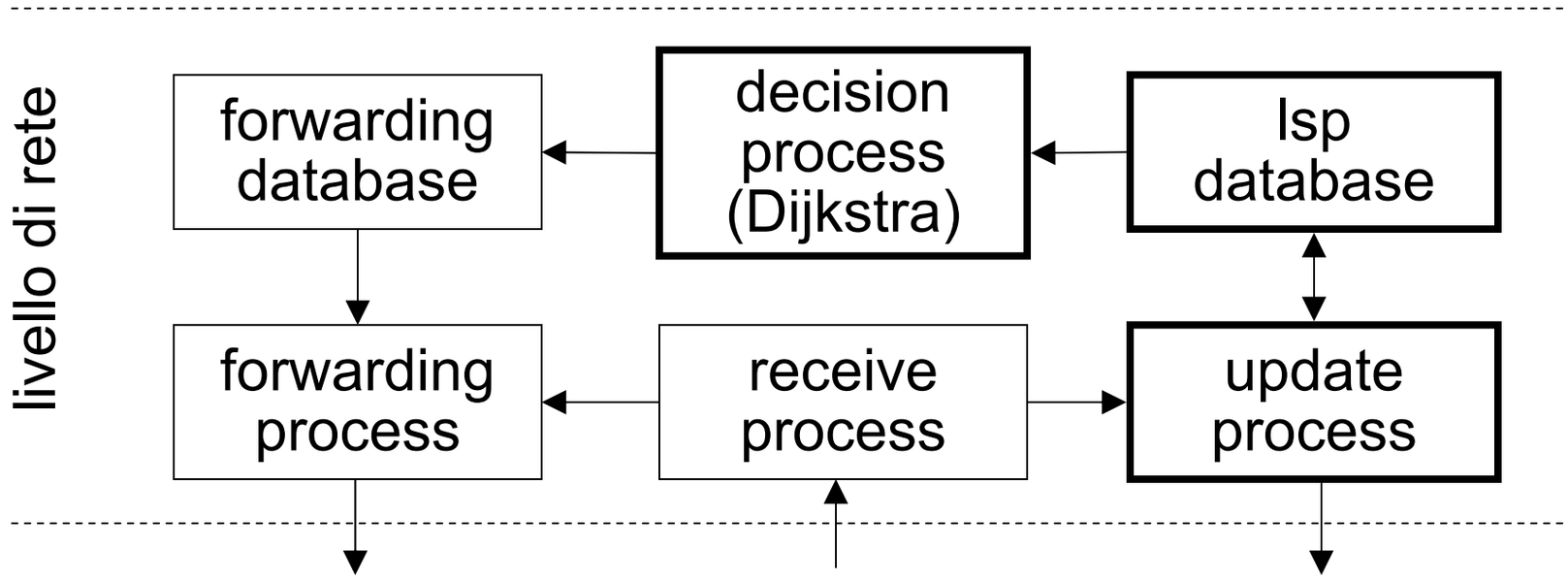
- il viaggio di un pacchetto da 3 a 6
  - ogni router instrada in base al proprio spanning-tree







architettura di un router lsp



- quando il receive process riceve un pacchetto
  - se è in transito verso altre destinazioni lo passa al forwarding process, che lo trasmette sfruttando le informazioni del forwarding database
  - se il pacchetto è di gestione viene passato ai protocolli superiori

pacchetti di gestione – neighbor greetings

- pacchetto di neighbor greetings
  - utilizzato per tenere aggiornate le adiacenze dirette
  - inviato periodicamente su tutte le interfacce
  - quando un is rileva una modifica di topologia
    - invia un lsp per far conoscere la modifica a tutti gli is della rete

lsp e selective flooding

- ogni pacchetto link-state-packet
  - contiene un numero di versione
- quando un is riceve un lsp
  - se ha la stessa versione di quello posseduto
    - non compie alcuna azione
  - se ha una versione più recente
    - aggiorna il suo database
    - ritrasmette il pacchetto in flooding su tutte le linee (eccetto quella di ricezione)
  - se ha una versione precedente
    - trasmette quello posseduto al mittente (per un rapido allineamento dei database)

router lsp e LAN

- una LAN su cui si affacciano diversi router si presta molto male ad essere modellata come un grafo
  - i router costruiscono un grafo completo, con un numero quadratico di archi
- modellazione tramite uno pseudo nodo
  - uno dei router sulla LAN (designated router) si prende l'onere di rappresentare la LAN come uno *pseudo nodo*
  - tutti gli altri router vedranno la LAN come una stella