

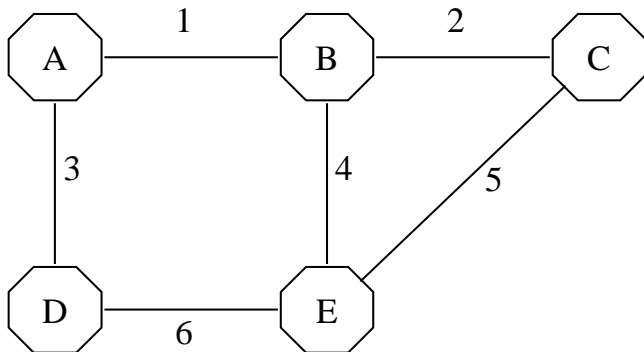
Linkkien tilaan perustuva reititys

Linkkien tilaan perustuva reititys

- Tavoitteena on välttää EV-protokollan luomat silmukat ja skaalautua laajempiin verkkoihin ja monenlaisiin topologioihin.
- Linkintilaprotokolla ylläpitää verkon topologiakarttaa.
 - Jokaisessa solmussa on sama topologiakartta
 - Kun topologia muuttuu, kartat päivitetään nopeasti.
 - Karttoja käytetään reittilaskennassa
- OSPF (Open Shortest Path First) on IETF:n määrittelemä linkintilaprotokolla Internetiä varten.
 - OSPF on suositeltu RIP:n seuraaja.

Kartta esitetään täydellisenä linkkien luettelona

- Esimerkkiverkko



Mistä	Mihin	Linkki	Etäisyys
A	B	1	1
A	D	3	1
B	A	1	1
B	C	2	1
B	E	4	1
C	B	2	1
C	E	5	1
D	A	3	1
D	E	6	1
E	B	4	1
E	C	5	1
E	D	6	1

- Kustakin rivistä vastaa tietty solmu
- Linkin suunnat esitetään erillisinä
- Jokaisessa solmussa sama kartta
⇒ Silmukoita ei voi syntyä

Reititystaulu muodostetaan linkkikannan tiedoista

Mistä	Mihin	Linkki	Etäisyys
A	B	1	1
A	D	3	1
B	A	1	1
B	C	2	1
B	E	4	1
C	B	2	1
C	E	5	1
D	A	3	1
D	E	6	1
E	B	4	1
E	C	5	1
E	D	6	1



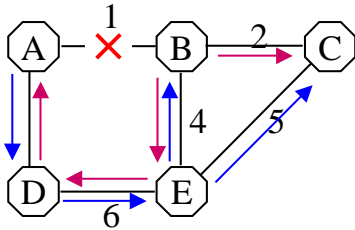
Mistä	Mihin	Linkki	Etäisyys
A	A	local	0
A	B	1	1
A	C	1	2
A	D	3	1
A	E	3	2

A:n reititystaulu

Linkkikanta

Levitysprotokolla (flooding protocol) levittää tiedot topologiamuutoksista

- Päivityssanomat levitetään koko verkkoon



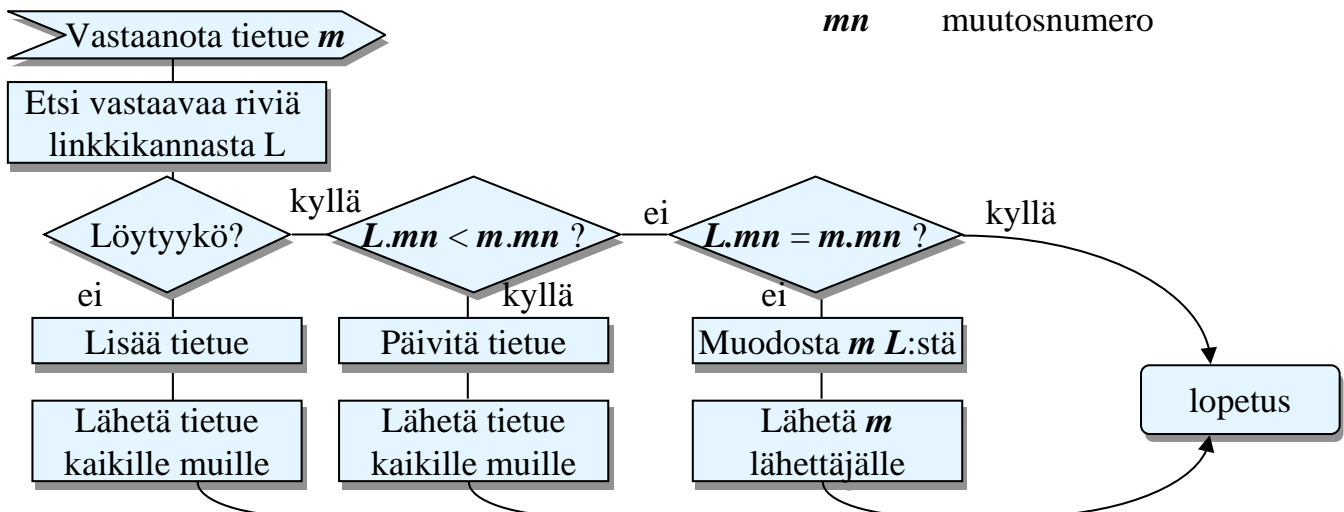
Mistä	Mihin	Linkki	Etäisyys	Mnumero
A	B	1	inf	2

Mistä	Mihin	Linkki	Etäisyys	Mnumero
B	A	1	inf	2

Levitysprotokolla (flooding protocol) levittää tiedot topologiamuutoksista

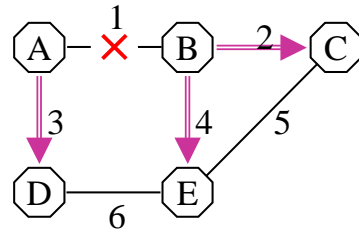
Levitysalgoritmi:

m vastaanotettu tietue
 L vastaava rivi linkkikannassa
 mn muutosnumero



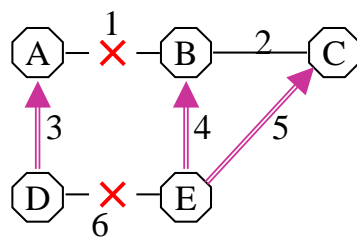
Linkkikanta A-B vian levityksen jälkeen

- Muutosnumeroiden laskenta alkaa käynnistyksessä 1:stä.
- Modulo aritmetiikka määrittelee mikä on “vähän suurempi kuin”
 \Rightarrow muutosnumerolaskuri voi pyörähtää ympäri.
 $\Rightarrow 4294967295 + 1 = 0$



Mistä	Mihin	Linkki	Etäisyys	Mnumero
A	B	1	inf	2
A	D	3	1	1
B	A	1	inf	2
B	C	2	1	1
B	E	4	1	1
C	B	2	1	1
C	E	5	1	1
D	A	3	1	1
D	E	6	1	1
E	B	4	1	1
E	C	5	1	1
E	D	6	1	1

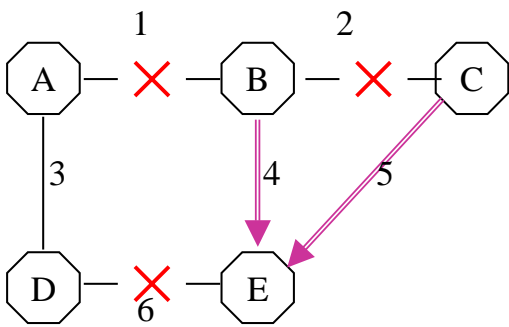
Kun verkko osittuu, puoliskojen kannat eroavat toisistaan



Mistä	Mihin	Linkki	Etäisyys	Mnumero
A	B	1	inf	2
A	D	3	1	1
B	A	1	inf	2
B	C	2	1	1
B	E	4	1	1
C	B	2	1	1
C	E	5	1	1
D	A	3	1	1
D	E	6	inf	2
E	B	4	1	1
E	C	5	1	1
E	D	6	1	1

Mistä	Mihin	Linkki	Etäisyys	Mnumero
A	B	1	inf	2
A	D	3	1	1
B	A	1	inf	2
B	C	2	1	1
B	E	4	1	1
C	B	2	1	1
C	E	5	1	1
D	A	3	1	1
D	E	6	1	1
E	B	4	1	1
E	C	5	1	1
E	D	6	inf	2

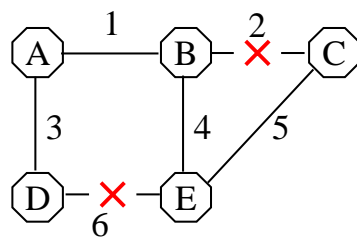
Linkki 2 vikaantuu \Rightarrow kannat eroavat lisää



B:n, C:n ja E:n kannat:

Mistä	Mihin	Linkki	Etäisyys	Mnumero
A	B	1	inf	2
A	D	3	1	1
B	A	1	inf	2
B	C	2	inf	2
B	E	4	1	1
C	B	2	inf	2
C	E	5	1	1
D	A	3	1	1
D	E	6	1	1
E	B	4	1	1
E	C	5	1	1
E	D	6	inf	2

Linkki 1 elpyy



Mistä	Mihin	Linkki	Etäisyys	Mnumero
A	B	1	1	3
A	D	3	1	1
B	A	1	1	3
B	C	2	1	1
B	E	4	1	1
C	B	2	1	1
C	E	5	1	1
D	A	3	1	1
D	E	6	inf	2
E	B	4	1	1
E	C	5	1	1
E	D	6	1	1

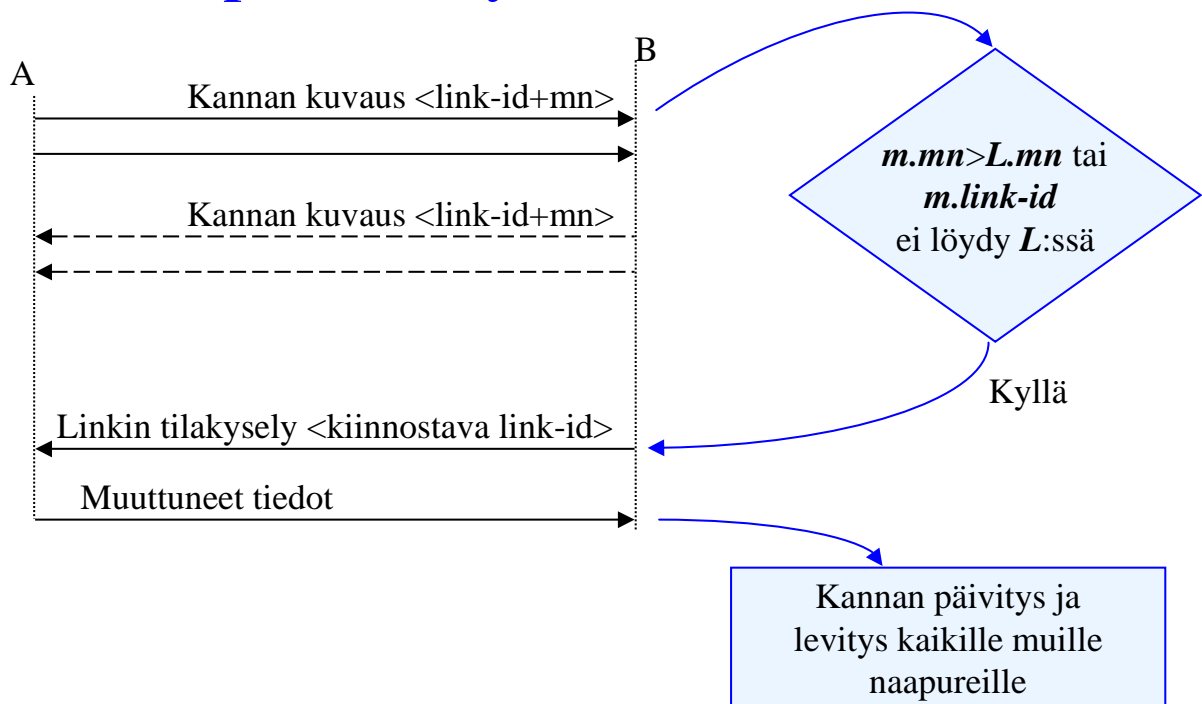


Mistä	Mihin	Linkki	Etäisyys	Mnumero
A	B	1	1	3
A	D	3	1	1
B	A	1	1	3
B	C	2	inf	2
B	E	4	1	1
C	B	2	inf	2
C	E	5	1	1
D	A	3	1	1
D	E	6	1	1
E	B	4	1	1
E	C	5	1	1
E	D	6	inf	2



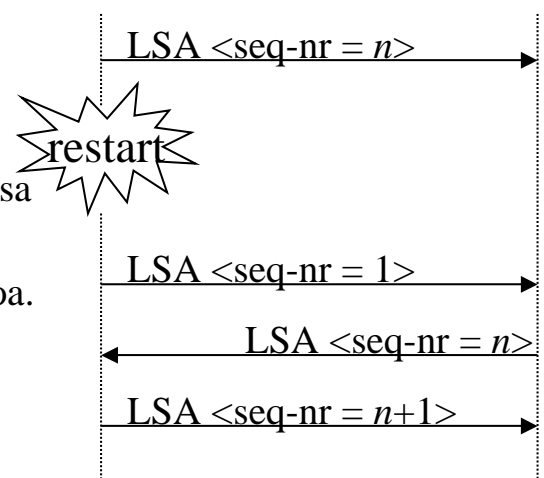
Mistä	Mihin	Linkki	Etäisyys	Mnumero
A	B	1	1	3
A	D	3	1	1
B	A	1	1	3
B	C	2	inf	2
B	E	4	1	1
C	B	2	inf	2
C	E	5	1	1
D	A	3	1	1
D	E	6	inf	2
E	B	4	1	1
E	C	5	1	1
E	D	6	inf	2

Osittuneen verkon jälleenyhdistyessä tarvitaan “naapurusten yhteenkasvatusta”



Mitä jos reititin C uudelleenkäynnistyy?

- Verkossa voi olla C:n ennen uudelleenkäynnistystä luomia ja levittämiä LSA:ta.
 - Resetin jälkeen reitin numeroi omat LSA:nsa numerolla “InitialSequenceNumber”
 - Naapuri vastaa, että sillä on uudempaa tietoa.
 - Mikäli C haluaa pitää oman LSA:nsa voimassa, se kasvattaa muutosnumeron naapurilta samaansa + 1 ja levittää uudestaan.
 - Mikäli naapurin tieto ei ole kuranttia resetin jälkeen, C poistaa sen asettamalla iän MaxAge arvoon ja levittämällä tietoa uudestaan.



Linkkikantojen yhtenäisyys on varmistettava

- *Levityssanomat* kuitataan linkki kerrallaan
- Kannan *kuvaussanomien* perillemeno varmistetaan
- Kannan kukin tietue suojataan vanhenemisajastimella, jos päivitys ei saavu ajoissa, tietue poistetaan
- Kannan tietueet on suojattu tarkistussummalla
- Sanomat sisältävät autentikointitietoa

- *Mutta: päivityksen ollessa käynnissä toiset solmut ovat paremmin ajan tasalla kuin toiset \Rightarrow reititysvirheitä*

Dijkstran lyhin-polku-ensin algoritmi

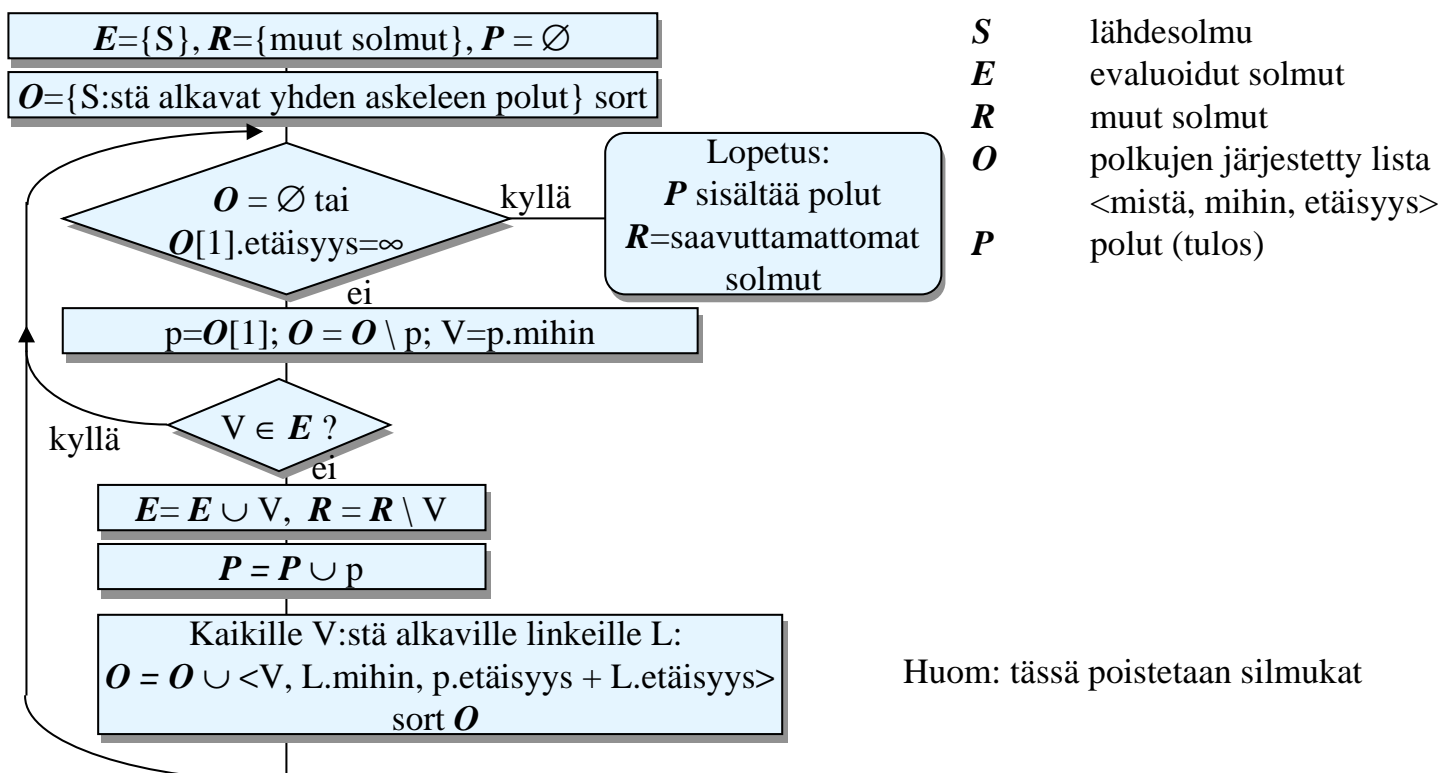
Dijkstran lyhin-polku-ensin algoritmiin

- OSPF perustuu Dijkstran lyhin-polku-ensin (SPF, shortest path first) algoritmiin.
- Algoritmin tavoitteena on tuottaa reititystaulu.
- Dijkstran algoritmi laskee lyhimmän polun lähdesolmun S ja kaikkien muiden solmujen välillä.
- Dijkstran algoritmi konvergoi nopeammin kuin Bellman-Ford.
 - $O(M \log M) < O(N \cdot M)$
 - M on linkkien lukumäärä, N on solmujen lukumäärä
- Solmut jaetaan evaluoituihin E , joista alkavat polut tunnetaan ja muihin R .
- Lisäksi tarvitaan polkujen järjestetty lista O .

S-38.121 / S-04 / RKa, NB

OSPF-16

Dijkstran lyhin-polku-ensin algoritmi



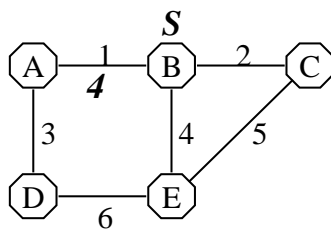
S-38.121 / S-04 / RKa, NB

OSPF-17

Dijkstran lyhin-polku-ensin algoritmi

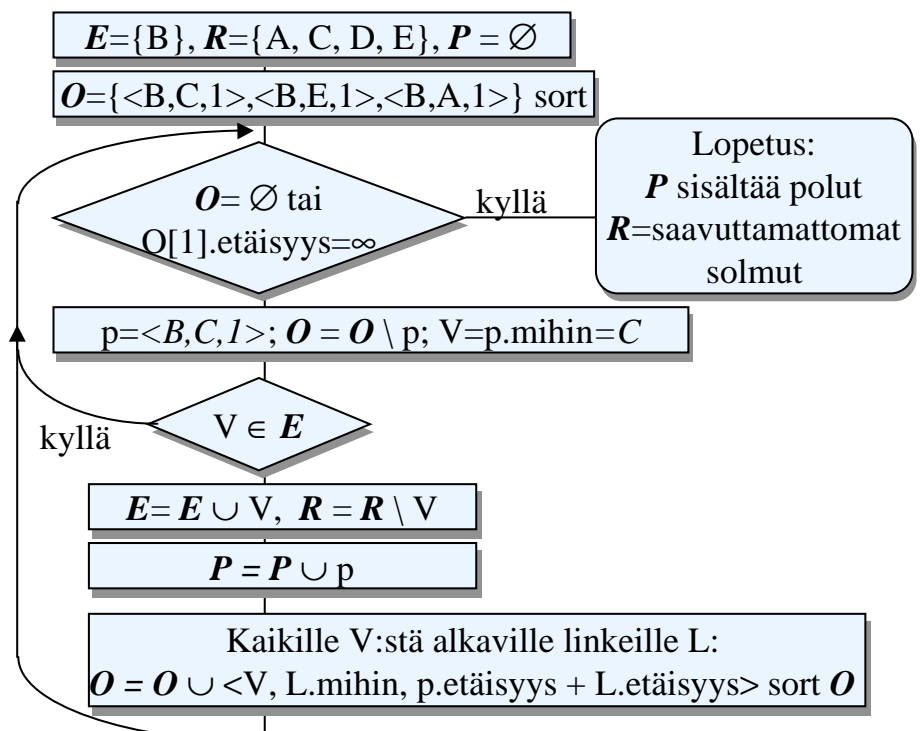
1. $E=\{S\}$, $R=\{N-S\}$, $O=\{\text{kaikki } S\text{:stä lähtevät yhden hypyn polut}\}$
2. Jos O on tyhjä tai jos O :n ensimmäisen polun pituus on ääretön:
 - Merkitse loput R :n solmut saavuttamattomiksi
 - Lopeta
3. P on listan O :n lyhin polku. Poista P O :sta. V on P :n viimeinen solmu.
4. Jos V on E :ssä:
 - Mene kohtaan 2
5. Luo joukko uusia polkuja lisäämällä P :hen kaikki V :stä lähtevät linkit. Polun pituus on vanha pituus + linkin kustannus. Lisää nämä polut O :hon pituusjärjestykseen.
6. Mene kohtaan 2

Dijkstran lyhin-polku-ensin algoritmi – esimerkki

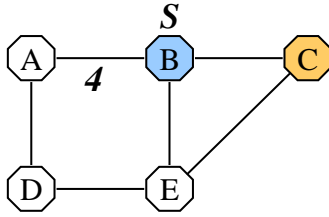


L

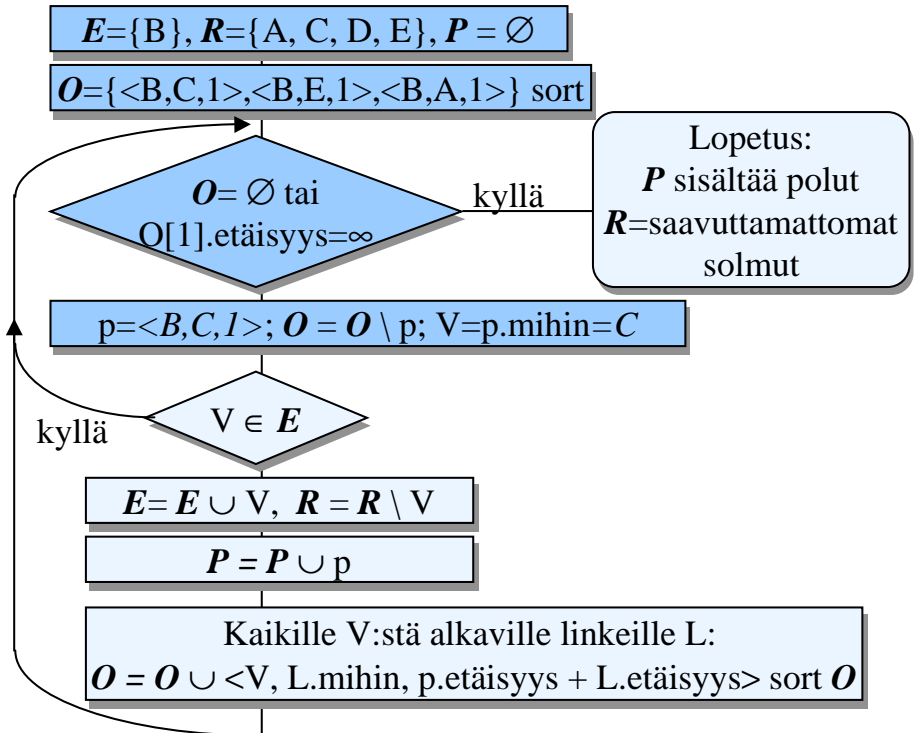
Mistä	Mihin	Linkki	Etäisyys
A	B	1	4
A	D	3	1
B	A	1	4
B	C	2	1
B	E	4	1
C	B	2	1
C	E	5	1
D	A	3	1
D	E	6	1
E	B	4	1
E	C	5	1
E	D	6	1



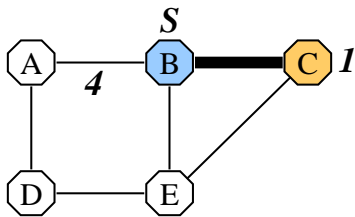
Dijkstran lyhin-polku-ensin algoritmi – esimerkki



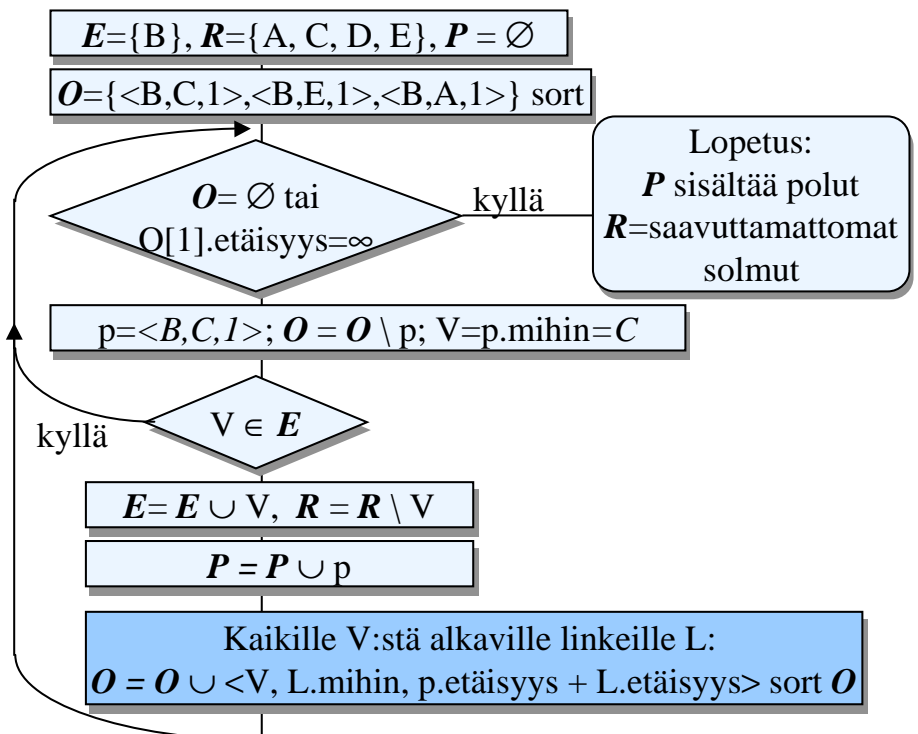
E
B
R
A, C, D, E
O
<B,C,1>
<B,E,1>
<B,A,4>



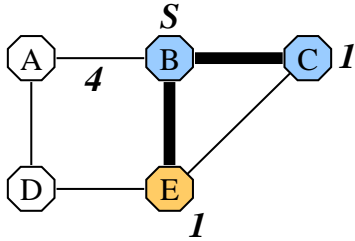
Dijkstran lyhin-polku-ensin algoritmi – esimerkki



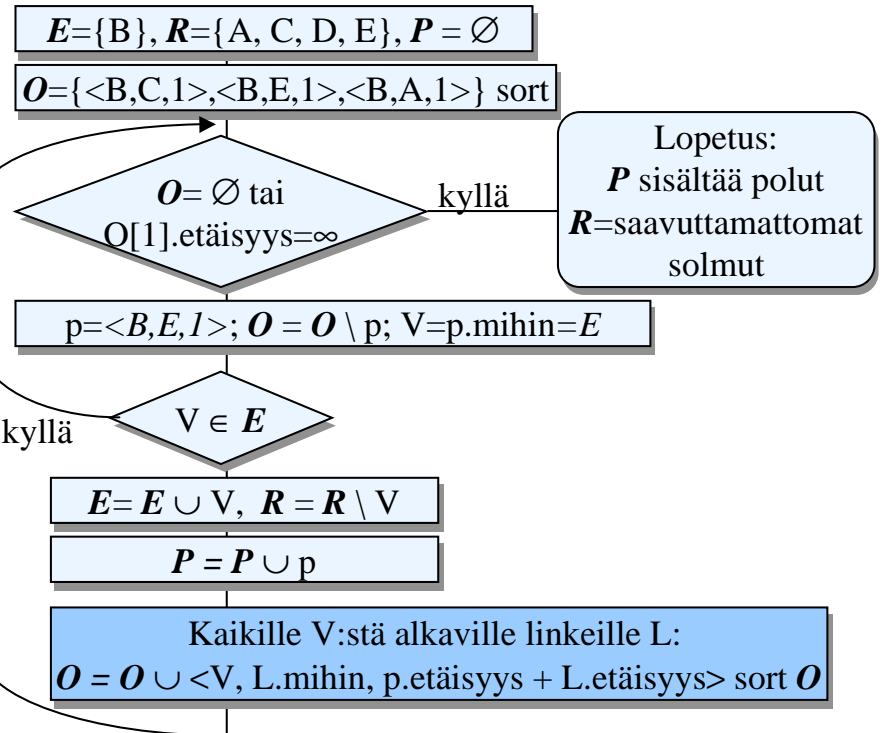
E
B, C
R
A, D, E
O
<B,E,1>
<C,E,2>
<B,A,4>



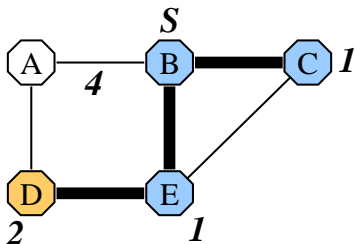
Dijkstran lyhin-polku-ensin algoritmi – esimerkki



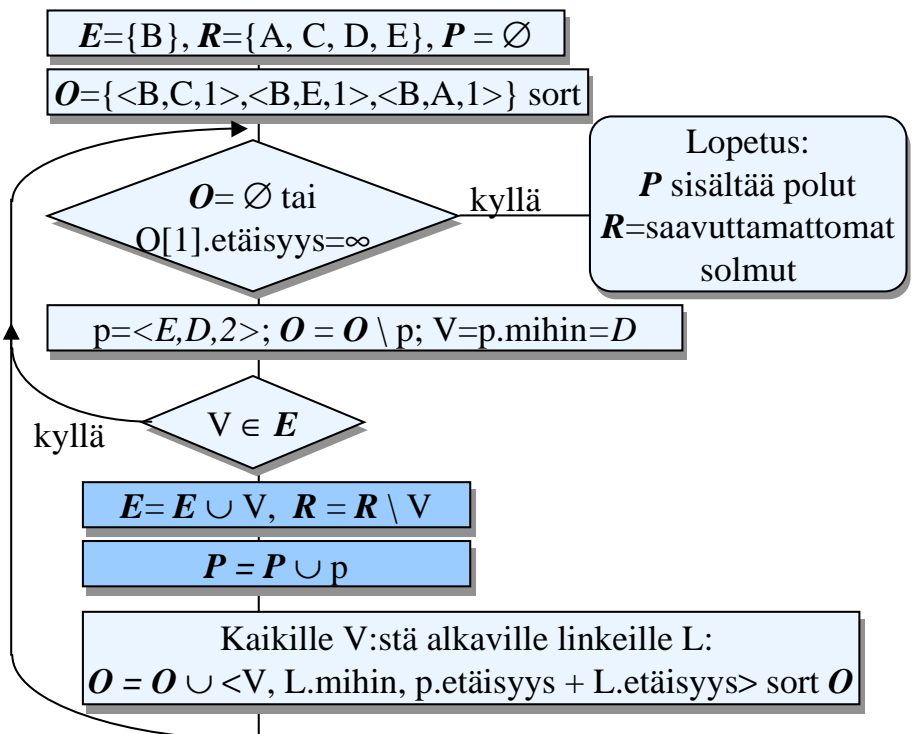
E	B, C, E
R	A, D
O	<C,E,2>
	<E,D,2>
	<B,A,4>



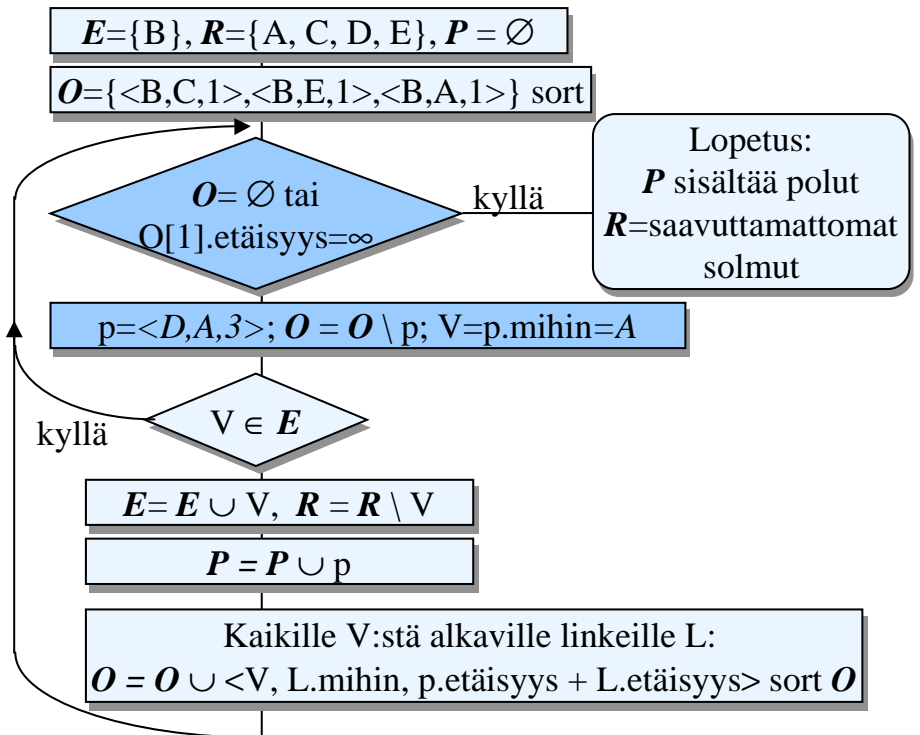
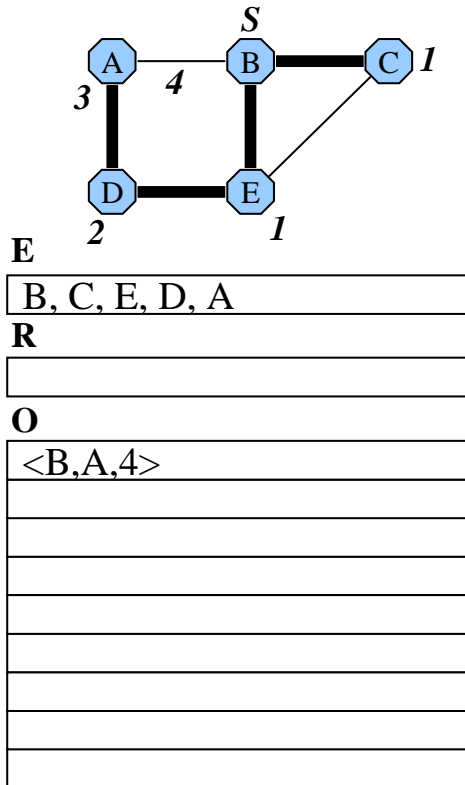
Dijkstran lyhin-polku-ensin algoritmi – esimerkki



E	B, C, E, D
R	A
O	<B,A,4>



Dijkstran lyhin-polku-ensin algoritmi – esimerkki



S-38.121 / S-04 / RKa, NB

OSPF-36

Linkintilaprotokollien etuja

- Linkkikantojen tila konvergoituu nopeasti, ilman silmukoita
 - $O(M \log M)$ $M =$ linkkien määrä
- Mittoina voidaan käyttää tarkkoja lukuja.
 - EV-protokollassa äärettömyyteen laskenta (inf=16) rajoittaa
- Yksi protokolla tukee monta etäisyysmittaa: ➡
 - Kapasiteetti, viive, hinta, luotettavuus.
- Ylläpitää useita polkuja yhteen kohteeseen. ➡
- Ulkoisten polkujen erillinen esitys. ➡

S-38.121 / S-04 / RKa, NB

OSPF-39

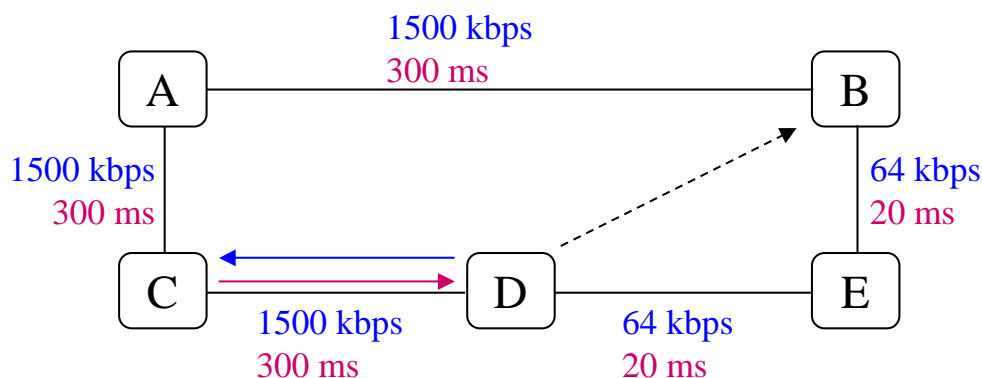
Useiden etäisyysmittojen käyttö (1)

Useiden etäisyysmittojen käyttö edellyttää:

- Mittojen tallennusta jokaiselle linkille (*L.et1*, *L.et2*, ...)
- Linkkiprotokolla kuljettaa kaikki mitat
- Erillisten reititystaulujen laskemista mitoille (*P(et1)*, *P(et2)*, ...)
- Käyttäjäsanomiiin merkataan vaadittu mitta.

Useiden etäisyysmittojen käyttö (2)

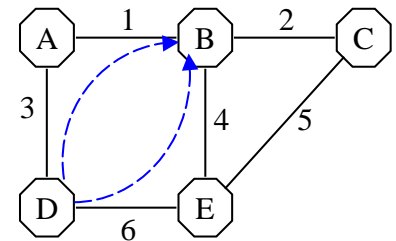
Reittisilmukka on mahdollinen, jos solmut käyttävät eri mittaa samalle käyttäjän paketille



⇒ Käyttäjäsanomiiin täytyy merkata vaadittu mitta

Liikenteen jakaminen vaihtoehtoisille samanmittaisille poluille tehostaa verkon käyttöä

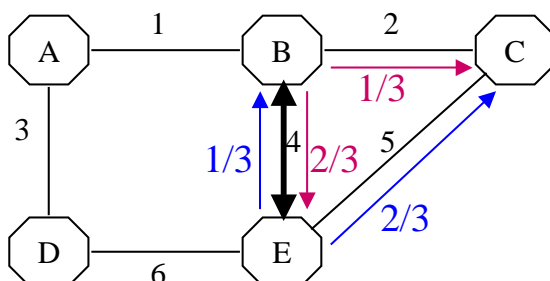
- + Solmujen kokonaisjonopituus laskee
- + Keskimääräinen viive laskee
- + Päästä päähän viiveen vaihtelut pienenee
- + Vikatilanteissa liikenteen heilahtelut vähenee



- Vaarana on pakettien järjestyksen muuttuminen, koska polkuviiheet (solmujen jonojen pituudet) vaihtelevat
- Hankaluus: menossa oleva liikenne ei pysy nykypolulla (no route pindown) \Rightarrow epästabiilisuus mahdollista (2000)
- ? Milloin polut ovat riittävän samanmittaisia?

Milloin polut ovat riittävän samanmittaisia?

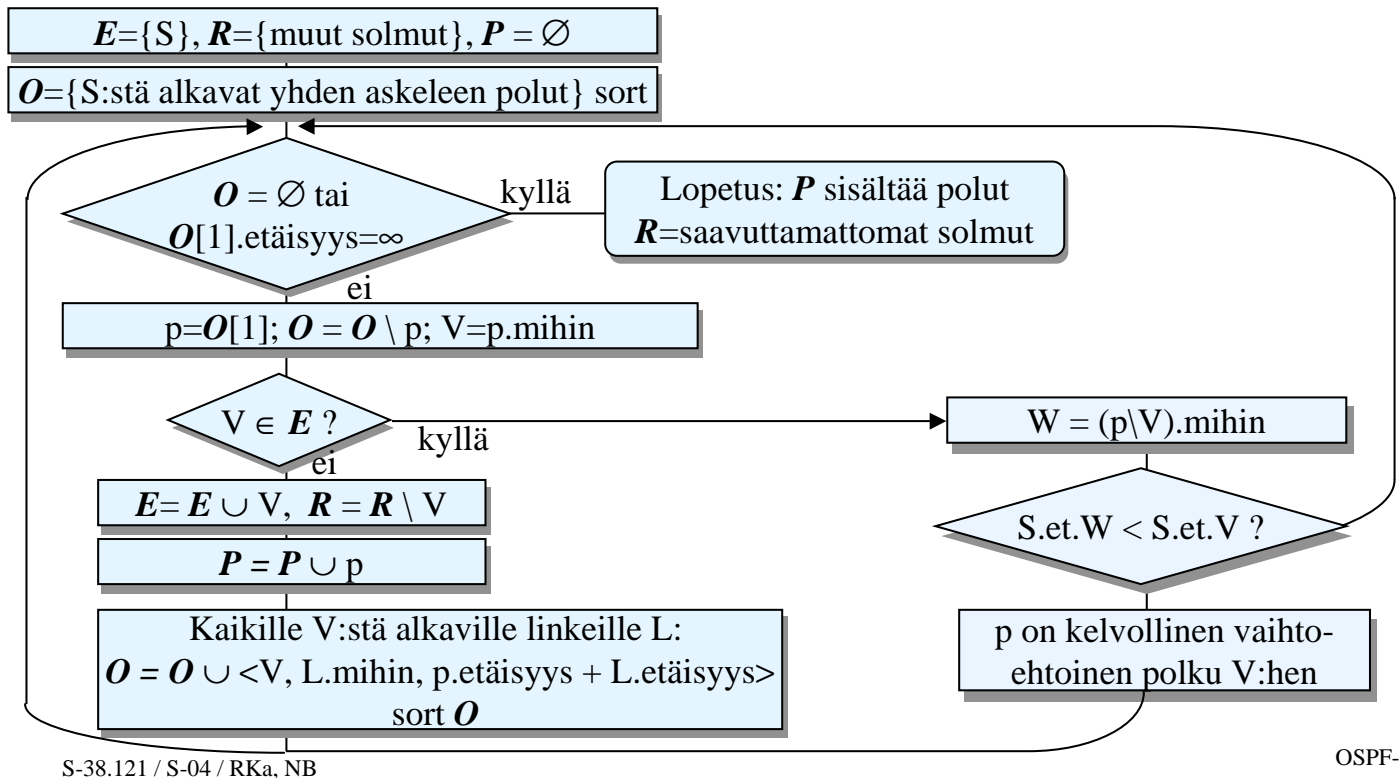
- Mitä tapahtuu jos C:hen menevä liikenne jaetaan kahden vaihtoehtoisen polun välille?



\Rightarrow X:ään menevä paketti voidaan lähettää solmun Y kautta jos Y on lähempänä kohdetta kuin paikallinen solmu

- Sääntö $A \rightarrow Y \dots \rightarrow X$, jos $\text{etäisyys}(Y \rightarrow X) < \text{etäisyys}(A \rightarrow X)$ rajaa vaihtoehtoiset reitit (monotonisiin)

Dijkstran lyhin-polku-ensin algoritmi joka löytää vaihtoehtoiset polut



S-38.121 / S-04 / RKa, NB

OSPF-44

Dijkstran lyhin-polku-ensin algoritmi joka löytää vaihtoehtoiset polut

1. $E=\{S\}$, $R=\{N-S\}$, $O=\{\text{kaikki } S:\text{stä lähtevät yhden hypyn polut}\}$
2. Jos O on tyhjä tai jos O :n ensimmäisen polun pituus on ääretön:
 - Merkitse loput R :n solmut saavuttamattomiksi.
 - Lopeta
3. P on listan O :n lyhin polku. Poista P O :sta. V on P :n viimeinen solmu.
4. Jos V on E :ssä:
 - Mene kohtaan 6
5. Luo joukko uusia polkuja lisäämällä P :hen kaikki V :stä lähtevät linkit. Polun pituus on vanha pituus + linkin kustannus. Lisää nämä polut O :hon pituusjärjestykseen. Mene kohtaan 2
6. Jos polun P etäisyys S :stä V :hen on sama kuin aiemmin laskettu etäisyys S :stä V :hen
 - Lisää vaihtoehtoinen polku V :hen
7. Mene kohtaan 2

S-38.121 / S-04 / RKa, NB

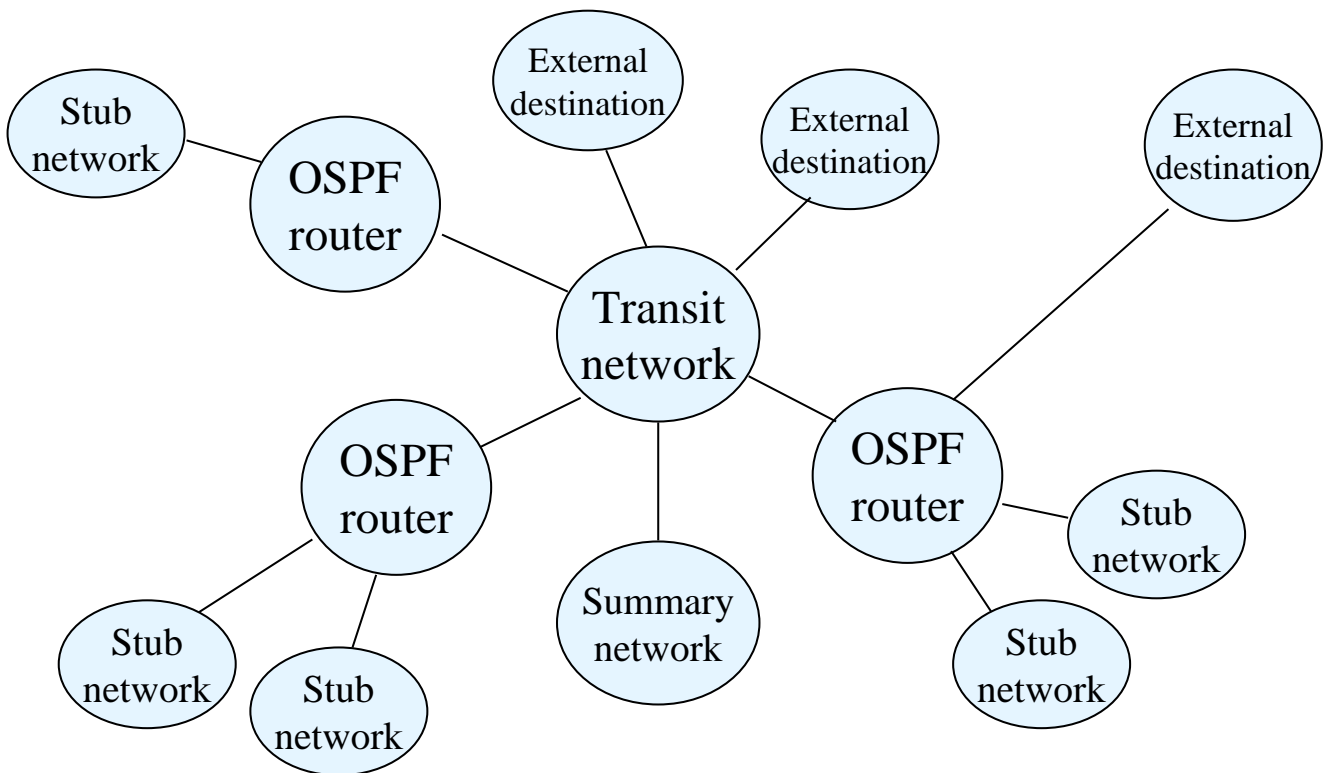
OSPF-45

Linkintilaprotokolla kuvaa useita ulkoisia reittejä tarkoilla (halutuilla) mitoilla

- EV-protokollan mahdollisuudet kuvata reittejä ovat rajalliset äärettömyyteen laskennan ja Bellman-Fordin monimutkaisuuden takia
 - Inf=16 \Rightarrow suurin etäisyys rajallinen
 - Bellman-Fordin monimutkaisuus on $O(N^2)$
- Linkintilaprotokolla on vapaa yo rajoitteista.
 - SPF reittilaskennan monimutkaisuus on $O(N \log N)$
N=ulkoisten reittien määrä
- Esim. noin 30 000 ulkoista reittiä $\Rightarrow 9 \cdot 10^8$ vs. 450 000

OSPF protokolla

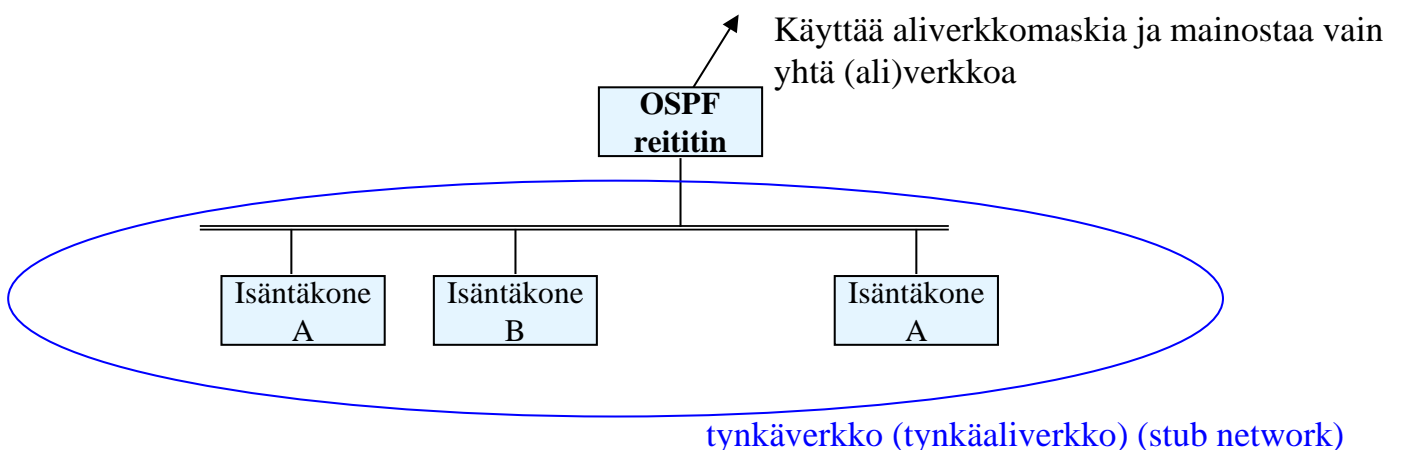
OSPF sees the network as a graph



S-38.121 / S-04 / RKa, NB

OSPF-48

OSPF erottelee reitittimet ja isäntäkoneet



Eli tästä syntyy kaksi linkintilatietuetta:

- + reititintietue
- + tynkäverkkotietue

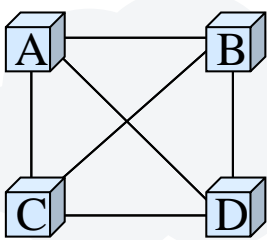
S-38.121 / S-04 / RKa, NB

OSPF-49

OSPF tukee yleislähetysverkkoja (1)

Yleislähetysverkossa

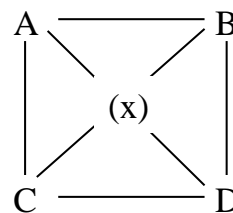
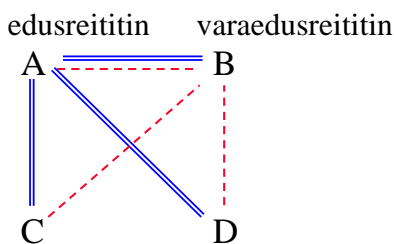
- Kaikki voivat lähettää kaikille
- Yksi voi lähettää kaikille tai joukolle
- Jos siinä on N reititintä, niillä on $N \cdot (N-1)/2$ naapuruussuhdetta
- Jokainen reititin mainostaa N-1 reittiä toisiin reitittimiin + yhtä tynkäverkkoa $\Rightarrow N^2$ kpl



$N \cdot (N-1)/2$
naapuruussuhdetta

Esim. Ethernet, Token ring, FDDI

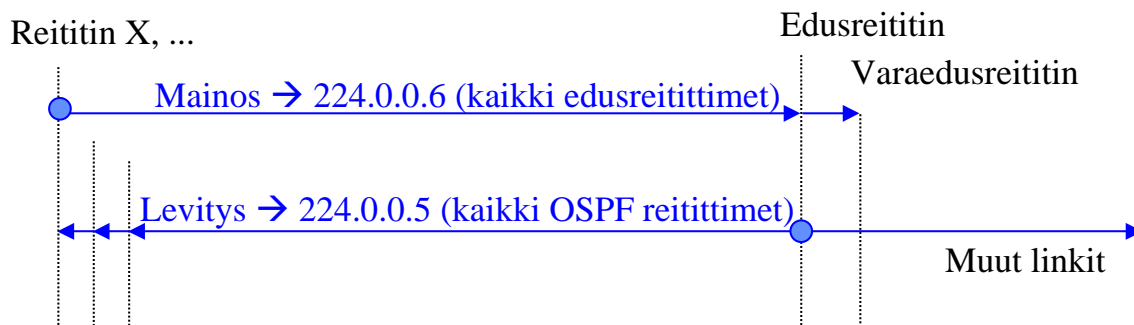
OSPF tukee yleislähetysverkkoja (2)



- Naapuruus rajattu vain **edusreitittimeen** (*designated router*)
 - \Rightarrow Edusreititin täytyy valita Hello-protokollan avulla
 - \Rightarrow Linkkikantojen synkronointi yksinkertaistuu
- **Varaedisreititin** (*backup designated router*) valitaan samalla kertaa

- Yleislähetysverkko kuvataan ”**virtuaalireitittimen**” avulla
- Linkit virtuaalireitittimestä reitittimiin ovat **verkkolinkkejä** (*network link*)
 - Edusreititin mainostaa
 - Etäisyys = 0
- Linkit reitittimistä virtuaalireitittimeen
 - Reitittimet mainostavat

OSPF levitysprotokolla yleislähetysverkossa

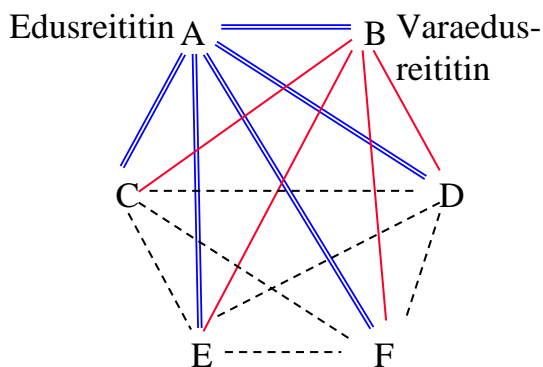


⇒ ei tarvitse käsitellä kuittauksia kaikilta muilta verkon reitittimiltä

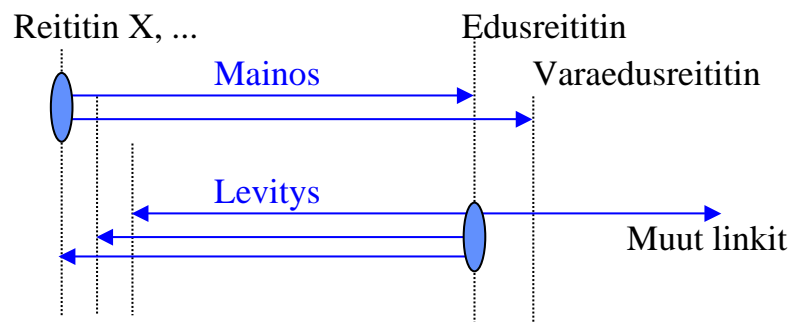
Varaedisreititin on mahdollisimman hiljaa

OSPF levitysprotokolla ei-yleislähetysverkossa

- Ei-yleislähetysverkoissa (esim. X.25, ATM, frame relay), OSPF toimii samoin kuin yleislähetysverkoissa paitsi, että yleislähetykset korvataan yksipistelähetyksillä

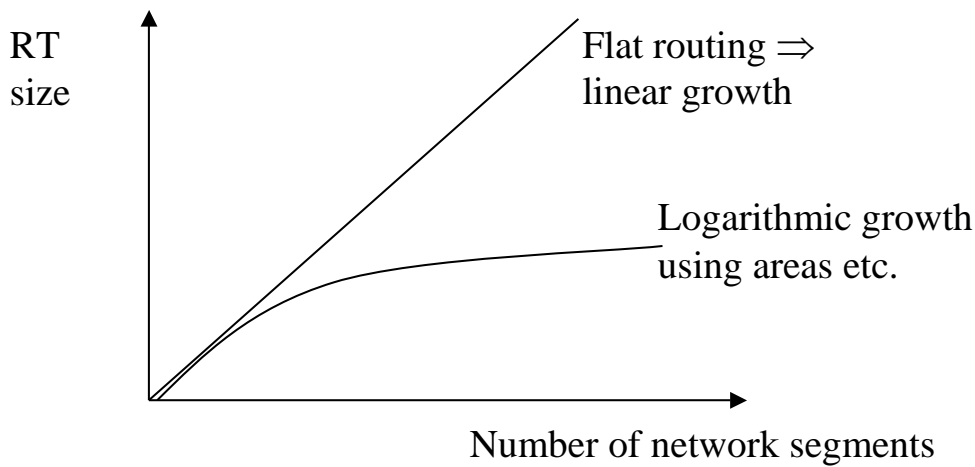


- Kiinteä yhteys edusreitittimeen
- Kiinteä yhteys varareitittimeen
- - - Valinnainen yhteys muiden reitittimien välillä (muu liikenne)



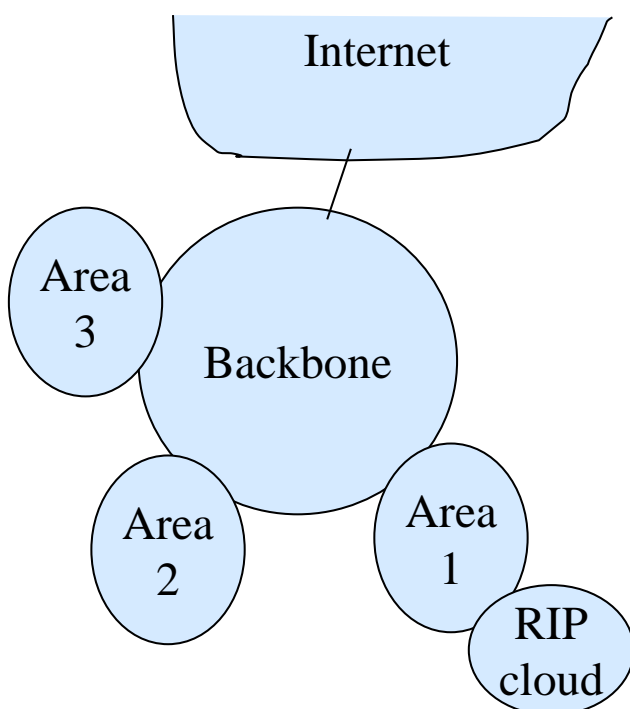
Huom: kiinteitä yhteyksiä kannattaa minimoida, koska ne ovat kalliita

The purpose of hierarchical routing in OSPF is to reduce routing table growth



The cost is: sometimes suboptimal routes.

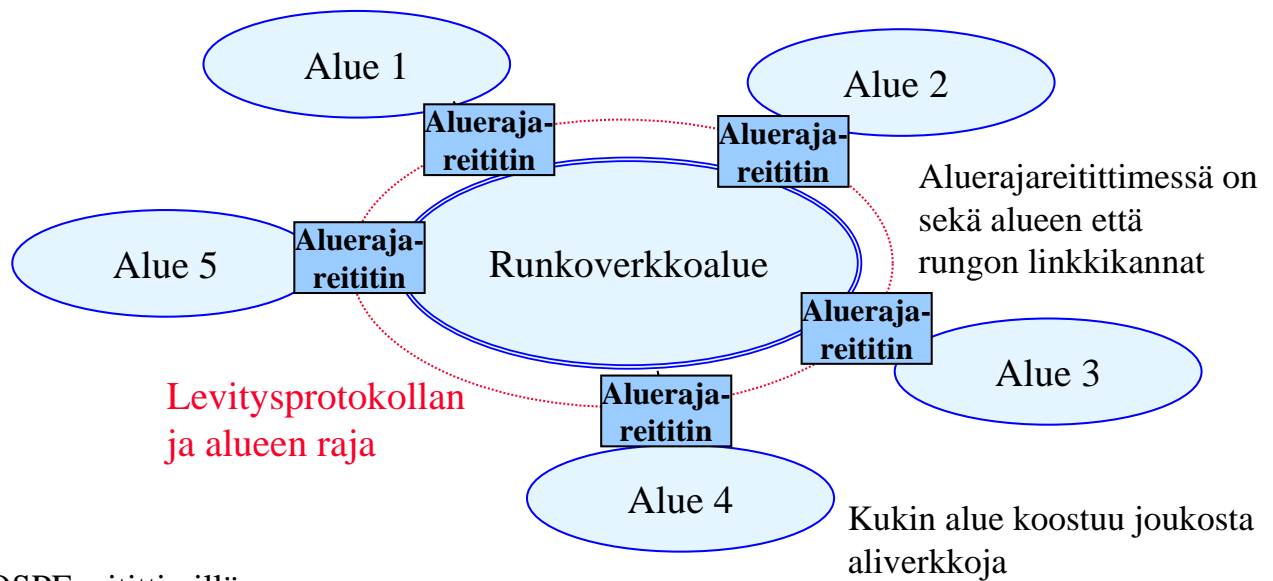
OSPF supports 4 level routing hierarchy



Level	Description
1	Intra-area routing
2	Inter-area routing
3	External Type 1 metrics
4	External Type 2 metrics

- Type 1 metrics are of the same order as OSPF metrics, e.g. hop count (for RIP and OSPF)
- Type 2 metrics are always more significant than OSPF internal metrics (e.g. BGP-4)

Jakamalla laaja verkko alueisiin OSPF helpottaa levitystä ja pienentää linkkikantoja

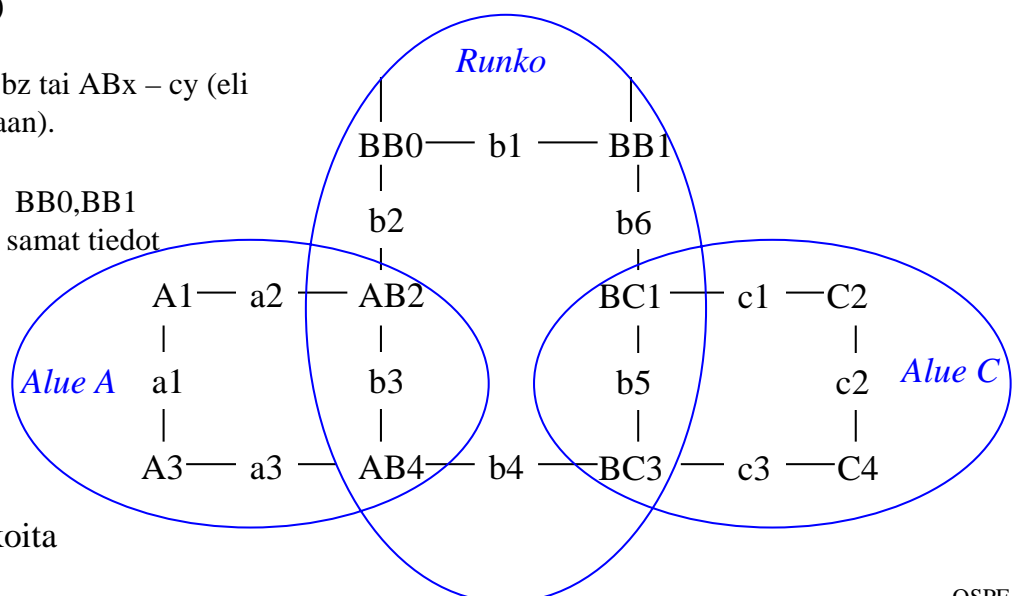


Alueen OSPF reitittimillä on samansisältöinen linkkien tilatietokanta

Muiden alueiden (ali)verkot kuvataan yhteenvetotietueissa, joiden mitta lasketaan RIP:n tapaan

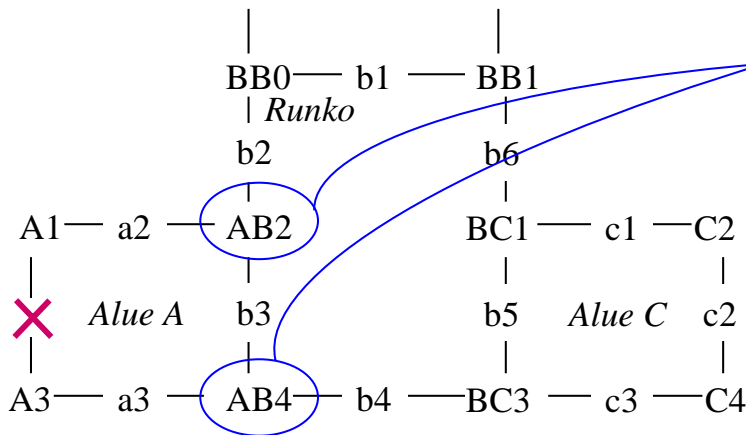
Alueen A linkkikanta:

- a1, a2, a3
- Rungon ja alueen C aliverkkotietueet (*summary records*)
 - ← AB2, AB4
 - Etäisyys ABx – bz tai ABx – cy (eli mittoja summataan).
- *Ulkoiset tietueet*
 - ← AB2, AB4 ← BB0, BB1
 - Kaikilla alueilla samat tiedot



Hierarkia ⇒ Ei silmukoita

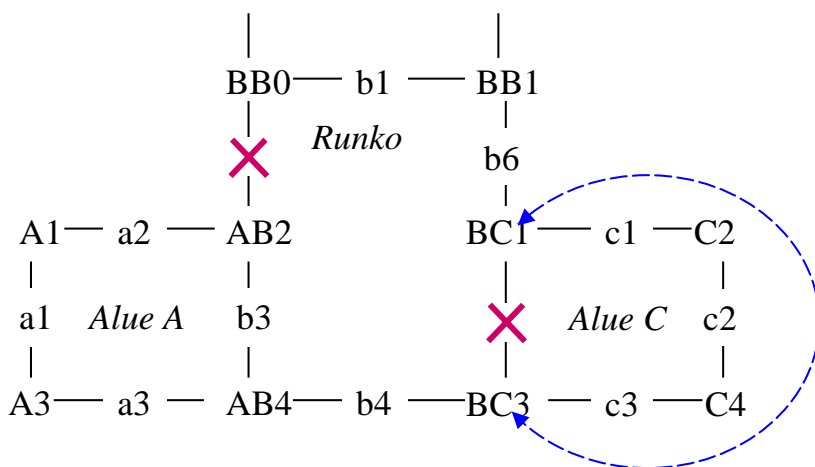
OSPF elpyy helposti alueiden vikatilanteista



AB2 ja AB4 mainostavat vain niitä verkkoja, jotka ne todella voivat saavuttaa: AB2: a2 ja AB4: a3.

Runko ei tunne Alueen A tarkkaa rakennetta, vaikkakin se tuntee A:n kaikki saavutettavat aliverkot.

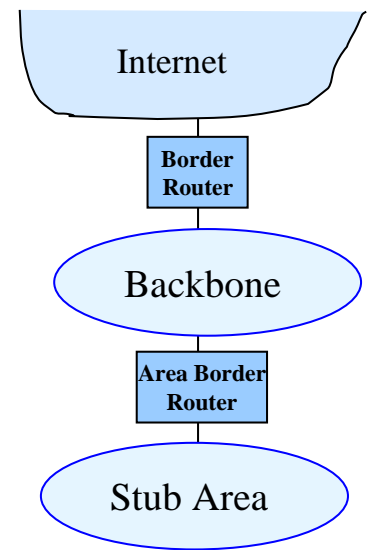
Virtuaalilinkki voi pelastaa rungon jakaantuessa lohkoihin vikojen takia



Virtuaalilinkki alueen C kautta: etäisyys=c1+c2+c3

Tynkäalueella (*stub area*) kaikki ulkoiset reitit summataan oletusreittiin

- Jos OSPF alueella on vain yksi reunareititin, kaikki liikenne ulkoiseen Internetiin ja ulkoisesta Internetistä kulkee tämän reunareitittimen kautta. Ei maksa vaivaa mainostaa kaikkia Internetin reittejä tällaisella alueella.
- Reunareitittimiä voi olla myös useita, mutta niistä ei voi valita sopivinta kohteen perusteella.
- NSSA - “*Not So Stubby Area*” on alue, jolla kaikki ulkoiset reitit on summattu oletusreittiin, paitsi jotkut.



OSPF link state records

Link State Advertisement (LSA) types in OSPF

- LS Type = 1 **Router LSA**
 - Describes a set of links starting from a router
- LS Type = 2 **Network LSA**
 - describes a network segment (BC or NBMA) along with the IDs of currently attached routers
- LS Type = 3 **Summary LSA for IP Network**
- LS Type = 4 **Summary LSA for Border Router**
- LS Type = 5 **External LSA**
 - describes external routes
- LS Type = 6 **Group Membership LSA**
 - used in MOSPF for multicast routing
- LS Type = 7 **Not So Stubby Area LSA**
 - to import limited external info
- LS Type = 8 (proposed) **external attributes LSA**
 - in lieu of Internal BGP

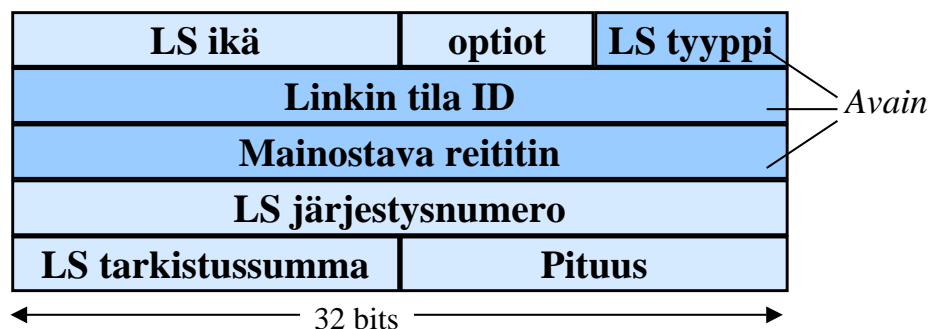
} Hierarchical
Routing

BC = Broadcast, e.g. Ethernet
NBMA = Non-Broadcast
Multiple Access, e.g. ATM

S-38.121 / S-04 / RKa, NB

OSPF-62

Yhteinen linkintilamainoksen otsikko



LS ikä:

- Sekunteina mainoksesta

Optiot:

- E - ulkoiset linkit
- T - palvelun tyyppiin perustuva reititys

LS tarkistussumma:

- Suojaa otsikon ja sisällön

Pituus:

- Otsikon + sisällön pituus

Linkin tila ID:

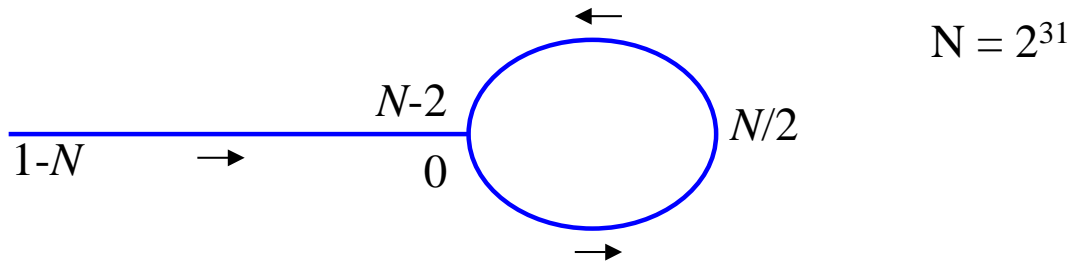
- Riippuu LS tyyplistä

S-38.121 / S-04 / RKa, NB

OSPF-63

LSA Sequence Numbers

- "Lollipop sequence space"



- If one of the number is < 0
 - The higher number is newer
- If both numbers are ≥ 0
 - If $(b-a) < (N-1)/2$ then b is newer

Router LSA (type 1)

Describes links starting from a router.

for each link	RouterType	0	Number of links
	Link ID		
	Link data		
	Type (E,B)	# TOS	TOS 0 metric
	TOS=x	0	TOS x metric
	TOS=y	0	TOS y metric
	...		
	TOS=z	0	TOS z metric

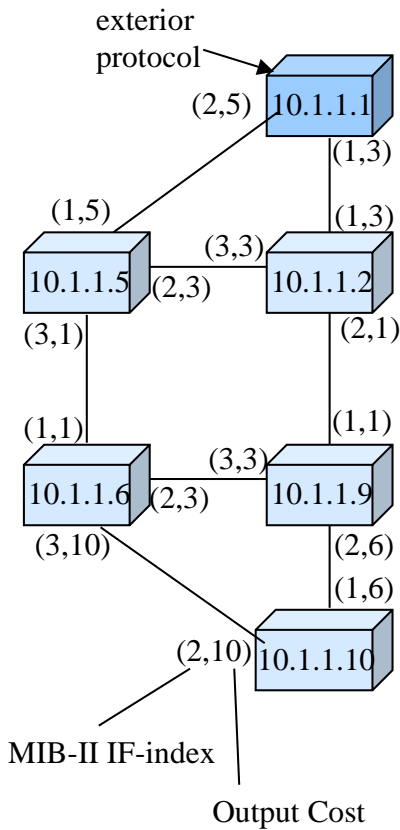
Router type

- E-bit (External)
 - This router is an area-border router
- B-bit (Border)
 - This router is a border router

Type

1. Link is a *point-to-point link* to another router
 - Link ID = neighboring router's OSPF ID
 - Link data = router's interface address
2. Link connects to a *transit network*
 - Link ID = IP address of designated router's interface
 - Link data = router's interface address
3. Link connects to a *stub network*
 - Link ID = Network/subnet number
 - Link data = network/subnet mask

Router LSA example



Router 10.1.1.1's router-LSA:

LS Age = 0 seconds	Options	LS type=1
Link State ID = 10.1.1.1		
Advertising Router = 10.1.1.1		
LS Sequence Number = 0x80000006		
Checksum= 0x9b47		Length = 60 bytes
RouterType=0	0	Nrof links = 3
Link ID = 10.1.1.2 (neighb)		
Link Data = IF-index 1 (unnum)		
Type=1	#TOS=0	Metric=3
Link ID = 10.1.1.5 (neighb)		
Link Data = IF-index 2 (unnum)		
Type=1	#TOS=0	Metric=5
Link ID = 10.1.1.1		
Link Data = 255.255.255.255		
Type=3	#TOS=0	Metric=0

E-bit
1=Router LSA

0=ordinary
1=pt-to-pt
1=pt-to-pt
3=stub network

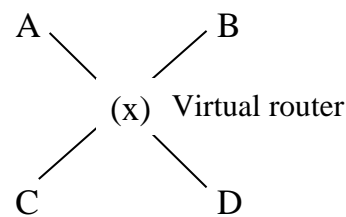
Length = 24 + 3 * 12 = 60 bytes

Router with 100 interfaces:

- Length = 24 + 100 * 12 = 1224 bytes

Network LSA (type 2)

Network mask
Attached router
Attached router
...
Attached router



- Advertised by designated routers for transit networks
- Link state ID (in header) = interface ID of designated router
- Attached router = OSPF identifier of the attached router

Summary Link LSA (type 3,4)

Network mask		
0	0	TOS 0 metric
TOS=x	0	TOS x metric
TOS=y	0	TOS y metric
...		
TOS=z	0	TOS z metric

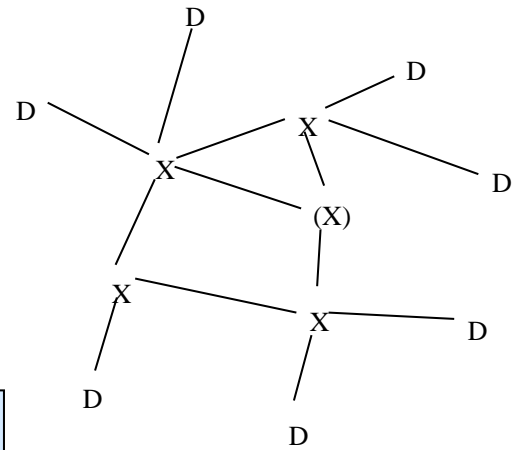
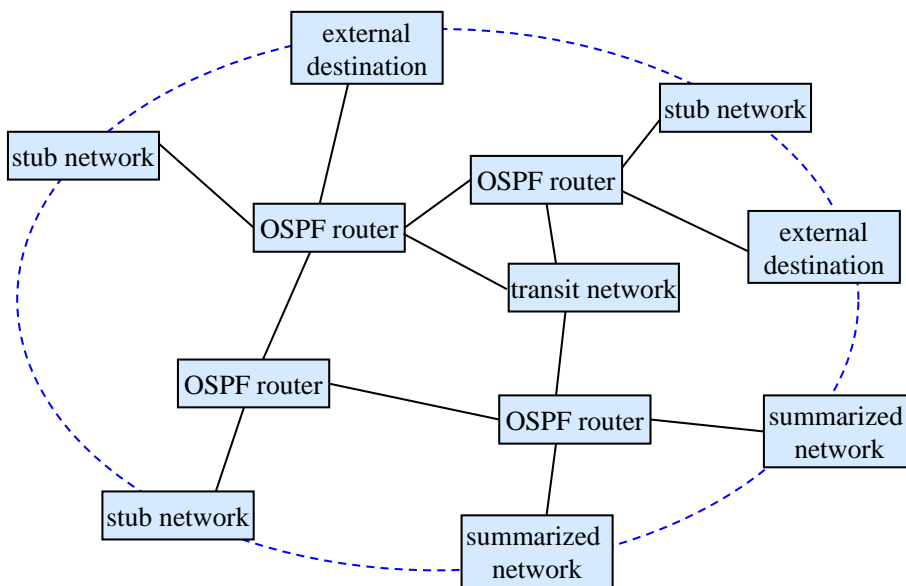
- For *IP networks* (type 3)
 - Network mask of network/subnet
 - Link state ID (in header) = IP network/subnet number
- For *border routers* (type 4)
 - Network mask = 0xFFFFFFFF
 - Link state ID (in header) = IP address of border router
- One separate advertisement for each destination

External Link LSA (type 5)

Network mask		
E, TOS=0	0	TOS 0 metric
External route tag (0)		
E, TOS=x	0	TOS x metric
External route tag (x)		
...		
E, TOS=z	0	TOS z metric
External route tag (z)		

- Link state ID (in header) = IP network/subnet of destination
 - Network mask = network/subnet mask
 - E-bit indicates that distance is not comparable to internal metrics
 - Larger than any internal metric
 - Route tag is only used by border routers (not used by OSPF)
- Advertised by border routers
 - Information from external gateway protocols (BGP-4)
 - One destination per record

Routes are computed after a change in the topology



- Separate routes for each TOS and for TOS 0
 - Possibly unreachable destinations for some TOS if not all routers support TOS \Rightarrow routed with TOS 0 (no loops because same decision in all nodes)

OSPF protokolla

OSPF packets – the protocol itself

- OSPF works directly on top of IP.
 - OSPF protocol number is 89.
- For most packets TTL = 1, except for hierarchical routing
- Destination IP address =
 - Neighbors IP address or
 - AllOSPF Routers (224.0.0.5) or
 - AllDesignatedRouters (224.0.0.6)
- OSPF has 3 sub-protocols:
 - Hello protocol (huomio protokolla)
 - Exchange protocol (tiedonvaihto protokolla)
 - Flooding protocol (levitys protokolla)

OSPF sanomien yhteinen otsikko

Versio	Tyyppi	Paketin pituus
Reitittimen ID		
Alueen tunnus		
Tarkistussumma	Autentikointityyppi	
Autentikointi		
Autentikointi		

Tyyppi erottelee OSPF sanomat toisistaan

- Type 1: Hello
- Type 2: Database Description
- Type 3: Link State Request
- Type 4: Link State Update
- Type 5: Link State Ack

- OSPF nykyversio on 2.
- Area ID
 - Yleensä (ali)verkkonumero
 - 0 = Runkoverkko (Backbone)
- Autentikointityyppi
 - 0 = Ei autentikointia
 - 1 = Salasana
 - Rajallinen merkitys
 - 2 = Kryptografinen autentikointi
 - MD5

0	Avain ID	Pituus
Kryptografinen järjestysnumero		

Huomioprotokolla varmistaa, että linkit toimivat ja valitsee edus- ja varaedusreitittimen



- Naapuri - lista naapureista, joilta on tullut viestejä viimeisen vanhenemisvälin aikana
- Huomioväli kertoo, kuinka usein paketteja lähetetään.
- Prioriteetti kertoo kelpoisuudesta edusreitittimeksi.
- Huomioviestien pitää kulkea linkillä molempiin suuntiin, jotta linkki kelpaisi reitiksi

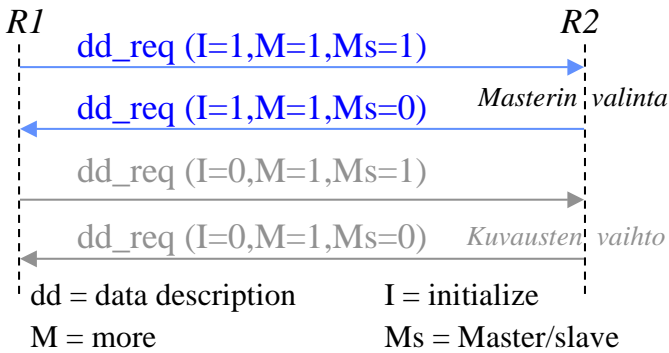
Otsikkotyyppi = 1		
Verkkomaski		
Huomioväli	Optiot	Prioriteetti
Vanhenemisväli		
Edusreititin		
Varaedisreititin		
Naapuri		
- - -		
Naapuri		

- Optiot
 - E = ulkoisia linkkejä
 - T = TOS reititys onnistuu.
- Edusreititin ja varaedisreititin = 0 jos ei tiedossa

Huomioprotokollan avulla valitaan edus- ja varaedisreititin

1. Vaalikelpoisuus saavutetaan yhden vanhenemisvälin jälkeen mikäli kaksisuuntainen saavutettavuus on OK.
2. Varaedisreitittimeksi valitaan korkeimman prioriteetin omaava niistä, jotka ilmoittautuivat. Tasatilanteessa valitaan suurimman ID:n omaava.
3. Jos kukaan ei ilmoittautunut varaedisreitittimeksi, valitaan korkeimman prioriteetin omaava naapuri. Tasatilanteessa suurimman ID:n omaava.
4. Edusreitittimeksi valitaan ilmoittautuneista yo. säännöllä.
5. Jos yksikään ei ilmoittaudu edusreitittimeksi, varaedisreititin ylennetään. Kohdat 2 ja 3 suoritetaan uudestaan varaedisreitittimen valitsemiseksi.
6. Muutoksia minimoidaan siten, että korjattu entinen edusreititin ei kiirehdi ilmoittautumaan uudelleen. Kohtia 2....5 suoritetaan jatkuvasti.

Vaihtoprotokolla alkusynkronoi linkkikannan edusreitittimen kannan kanssa (1)

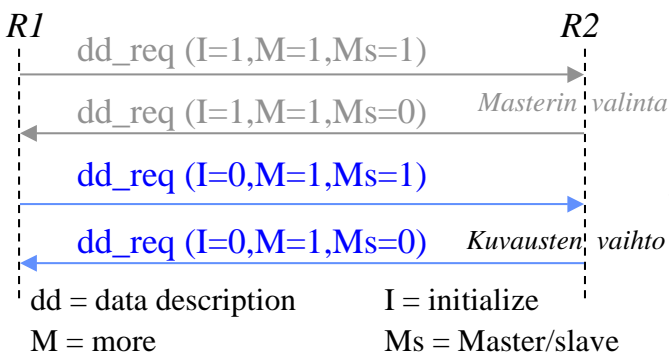


Otsikkotyyppi = 2 (dd)			
0	0	Optiot	0 IMM s
dd järjestysnumero			

- Vaihtoprotokolla käyttää tietokannan kuvauspaketteja (database description packets)
- Ensin valitaan master ja slave

- Jos molemmat ilmoittautuvat mastereiksi, korkein osoite voittaa
- Uudelleenlähetyks jos paketti hukataan
- Sama järjestysnumero vastauksessa

Vaihtoprotokolla alkusynkronoi linkkikannan edusreitittimen kannan kanssa (2)

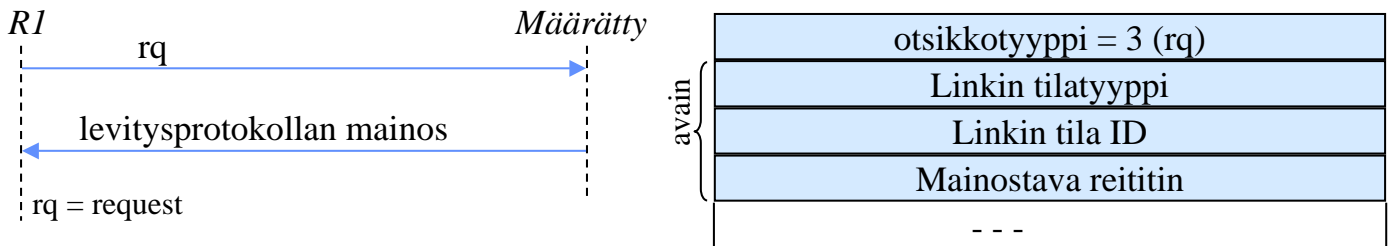


Otsikkotyyppi = 2 (dd)			
0	0	Optiot	0 IMM s
dd järjestysnumero			
Linkin tilatyyppi			
Linkin tila ID			
Mainostava reititin			
Linkin tilan järjestysnumero			
Linkin tilan tark.summa		Linkin tilan ikä	

- Master lähettää oman kantansa kuvauksen järjestysnumeroiduissa paketeissa
- Slave kuittaa paketit lähettämällä vastaavat omat kuvauksensa.

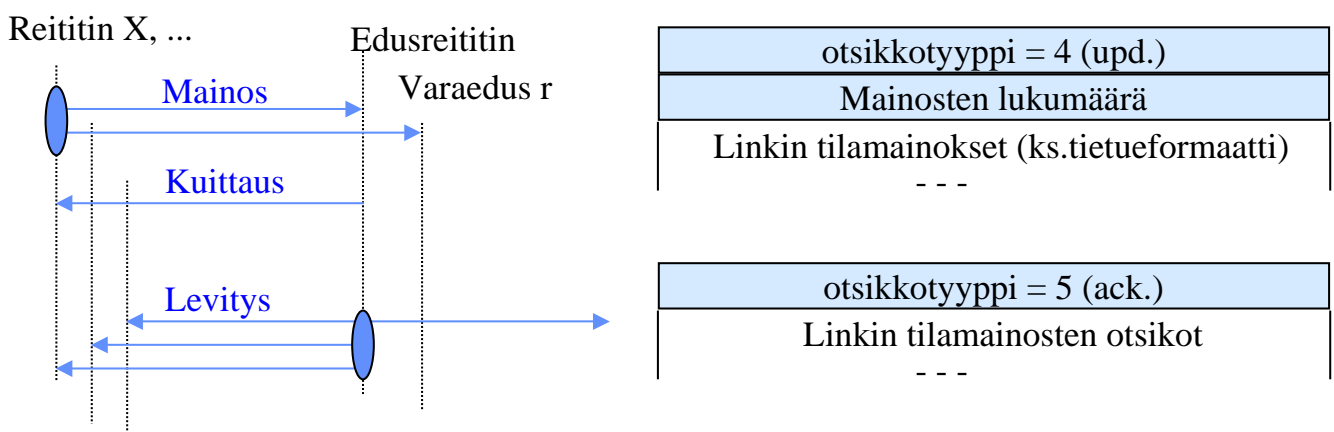
- Vaihto jatkuu kunnes molemmat ovat lähettäneet omat kuvauksensa. (M=0)
- Erot kirjataan kyseltävien tietueiden listaan.

Tietueiden sisältö kysellään pyyntöpaketeilla, jotka kuitataan levitysprotokollan paketein



- Vastausta odotetaan uudelleenlähetyksen väli. Jos ei vastausta, pyyntö toistetaan.
- Jos kyseltävät tietueet eivät mahdu yhteen pakettiin, tietueet jaetaan useaan kyselyyn.
- Jonkin mennessä pieleen, palataan roolien uudelleen neuvotteluun.
- Tietuesisältöjen kysely voi alkaa heti, kun yksikin eroava tietue on havaittu, jolloin dd-pakettien ja rq-pakettien vaihto tapahtuu rinnan.

Levitysprotokolla ylläpitää alueen linkkikantojen yhtenäisyyttä



- Alkuperäisen mainoksen lähettää aina linkistä vastaava reititin.
- Mainosta levitetään levityssääntöjen mukaan koko alueelle (age = age + 1).
- Uuden tietueen kuittaus voidaan BC verkossa korvata levityssanomalla
- Yksi ack voi kuitata monta mainosta.
- Viiveen avulla saadaan monta kuittausta samaan pakettiin.

Summary of OSPF subprotocols

Message \ Protocol	Hello (1)	DD (2)	LS rq (3)	LS upd (4)	LS ack (5)
Hello protocol	X				
Database exchange		X	X	X	X
Flooding protocol				X	X

Server Cache Synchronization Protocol (SCSP) is OSPF without Dijkstra's algorithm and with more generic data objects.

Linkkitietueilla on ikä, vanhat poistetaan kannasta (1)

- Vanhat tiedot on poistettava kannasta
- Kaikilla solmuilla on oltava samat tiedot
 - ⇒ Poistot täytyy synkronoida
- OSPF:n linkkitietuet vanhenevat
 - Ikä = 0 kun tietue luodaan
 - Ikä = mainosten kulkemien linkkien lkm + sekunnit vastaanotosta
- Maksimi-ikä on 1 tunti
 - Ei käytetä reittien laskennassa
 - Poistettava
- Jokaista tietuetta pitää mainostaa vähintään 30 min välein.
 - Uusi mainos nolaa iän ja inkrementoi tietueen järjestysnumeron.

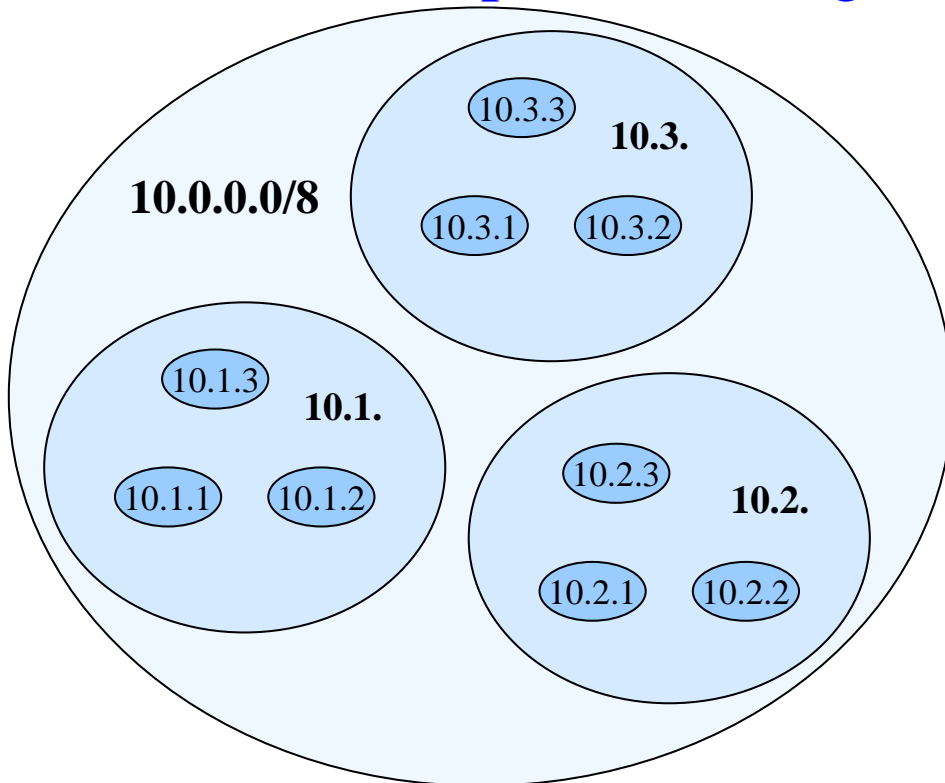
Linkkitietueilla on ikä, vanhat poistetaan kannasta (2)

- Kun ikä tulee täyteen (MaxAge = 1 h) tietue poistetaan
 - Reititin on lähetettävä mainos naapurille kun se poistaa vanhentuneen tietueen
- Levitysalgoritmi tutkii vastaanotetun tietueen iän
 1. MaxAge mainos hyväksytään ja levitetään - tämä poistaa vanhan tiedon.
 2. Jos mainoksen ikäero kantaan on pieni, mainosta ei levitetä, jotta saman tiedon kopiot eivät kuormittaisi verkkoa. Tämä tapahtuu normaalissa levityksessä kun tietue saapuu eri solmujen kautta.
 3. Jos mainoksen ikäero kantaan on suuri (>MaxAgeDiff), uusin mainos hyväksytään ja levitetään. Tässä tapauksessa reititin on todennäköisesti käynnistetty uudelleen.
 4. Jos MaxAge tietuetta ei löydy, mainos ei aiheuta toimenpiteitä, (koska reititin on jo ehtinyt poistaa vanhan tiedon.)

OSPF timeouts – LS Age field

Constant	Value	Action of OSPF router
MinLSArrival	1 second	Max rate at which a router will accept updates of any LSA via flooding
MinLSInterval	5 seconds	Max rate at which a router can update an LSA
CheckAge	5 min	Rate to verify an LSA Checksum in DB
MaxAgeDiff	15 min	When Ages differ more than 15 min, they are considered separate. Smaller LS age - newer!
LSRefreshTime	30 min	A Router must refresh any self-originated LSA whose age has reached 30 min.
MaxAge	1 hour	LSA is removed from DB.

Example of routing hierarchy



Example:

- 16 segments in each lowest level network
- flat routing:
RTsize= $16 * 9 = 144$
- areas 10.1.1:
16 local routes +
10.1.2/24
10.1.3/24
10.2/16
10.3/16
== 20 RT entries!

Why is it difficult to route packets around network congestion?

- BBN ARPANET link state metric varied with the length of the output queue of the link \Rightarrow lead to route trashing.
- The problem is there is no route pin-down for existing traffic.
- By limiting the range of the metric changes, an equilibrium could be reached. Nevertheless routing instability is the problem.

When QoS or Class of Service a'la DiffServ is introduced this problem again becomes important.

OSPF development history

