Lezione n.9

PASTRY

Laura Ricci

27/3/2009

PASTRY

- Pastry: proposto nel 2001 da Rowstron (Rice University) e Druschel (Microsoft)

- Bamboo, una reingegnerizzazione del protocollo Pastry può essere scaricata dal sito http://bamboo-dht.org/

- Obiettivo principale: definire un middleware per la costruzione di applicazioni P2P di tipo diverso
  - File sharing
  - Memoria distribuita
  - Group Communication

- Caratteristiche:
  - decentralizzazione completa
  - routing efficiente

PASTRY

- Auto-organizzazione della overlay network
- Per una rete di N nodi:
  - Numero di routing hops O(log N)
  - Memoria per nodo : O(log N)
- Proximity Routing: la scelta dei nodi da inserire nella tabella di routing tiene conto della prossimità fisica dei nodi (prossimità misurata sulla base di diverse metriche, ad esempio, numero di IP hops,.....)

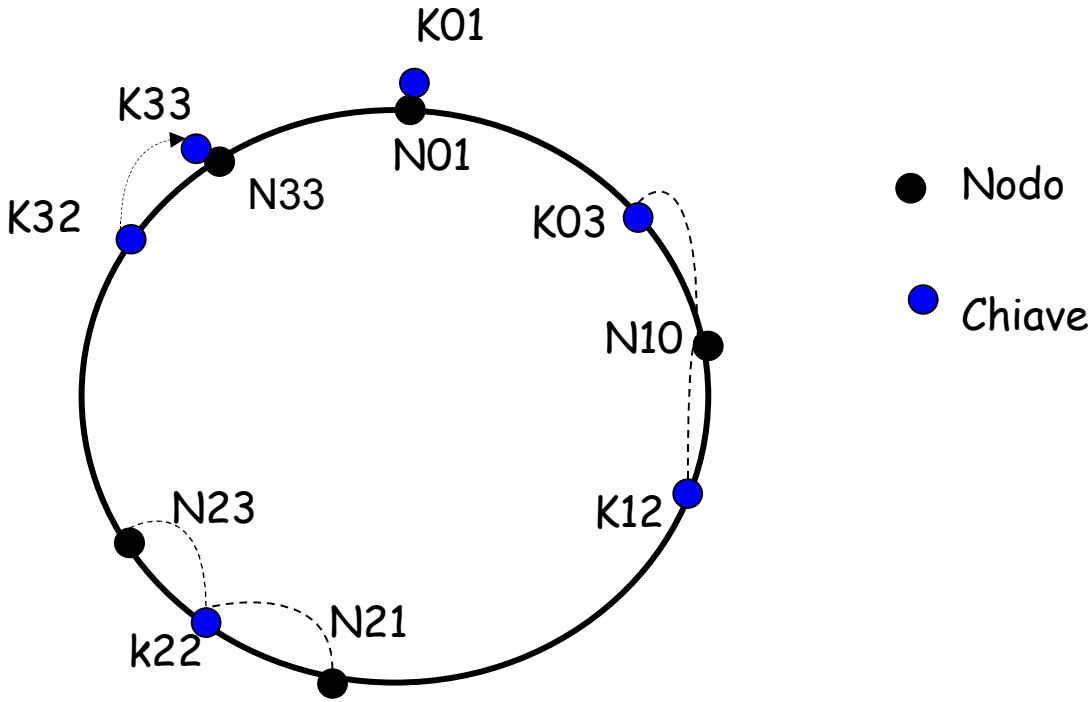
PASTRY

- Pastry associa ad ogni nodo (nodeID) e ad ogni chiave (key) un identificatore, mediante una funzione hash (es: hash dell'IP o della chiave pubblica)
- Identificatori di l bits (l in genere uguale a 128),
- Le sequenze di bits vengono interpretate come valori in base 2^b (in genere b=4)

  esempio: 1000 1111 1010=8FA, per b=4, 2^b=16
- Gli identificatori contengono quindi l/b cifre in base b
- Ogni chiave viene associata al nodo il cui nodeID ha valore numericamente più vicino al valore della chiave, tra i nodi attivi sulla rete Pastry
- Nel caso in cui vi siano più nodi a uguale distanza numerica dalla chiave, la chiave viene replicata su ognuno di essi

PASTRY: ASSOCIAZIONE CHIAVI NODI

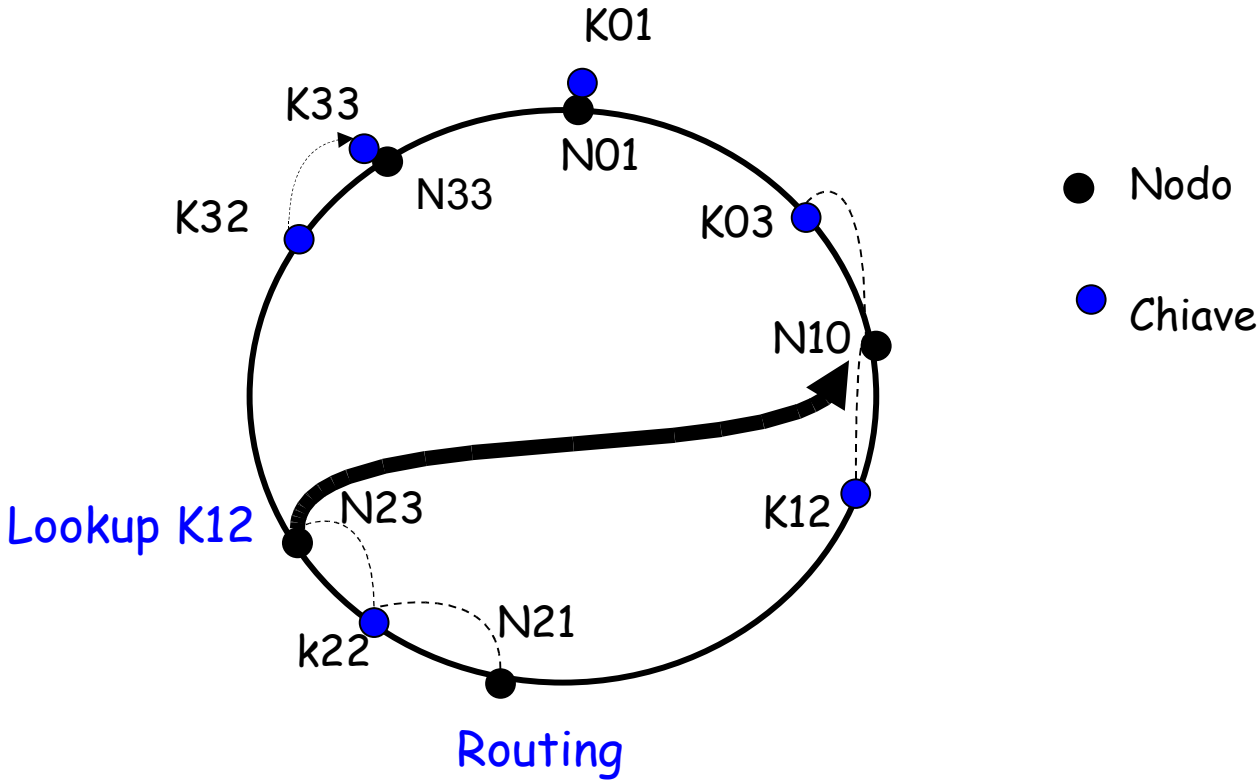
- Ogni chiave viene associata al nodo con identificatore numericamente più vicino alla chiave
- Esempio: spazio degli identificatori con l=4, b=2, 2^b=4. Ogni stringa di 4 bits viene interpretata come un valore in base 2^b=4



Associazione delle chiavi ai nodi

PASTRY

- Il routing di una query avviene individuando il nodo presente sull'anello con l'identificatore più vicino numericamente alla chiave
- Se N23 che ricerca la chiave k12 deve individuare il nodo numericamente più vicino alla chiave, cioè N10.



PASTRY: IL ROUTING

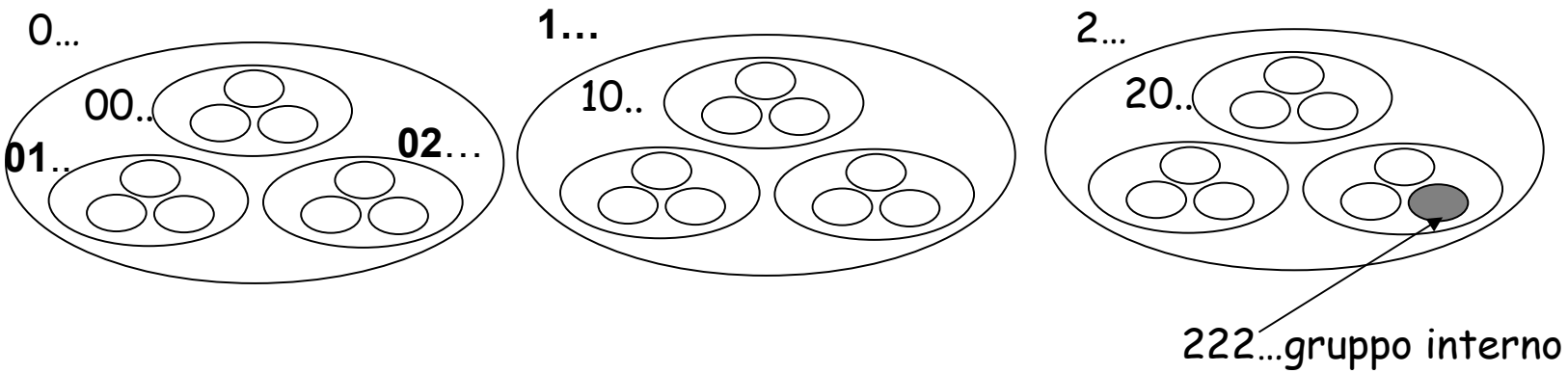
- Obiettivo: mantenere una tabella di dimensione O(log N)
- Memorizzare nella tabella informazioni che permettano di individuare un percorso di routing che porti via via verso nodi numericamente più vicini alla chiave
- Mantenere il numero di routing hops all'ordine O(log N)

PSEUDO PASTRY: CONCETTI BASE

- Esempio: alle chiavi ed ai nodi sono associati identificatori di l cifre in base 2^b Per semplicità nell'esempio seguente consideriamo la base 3, anche se non potenza di 2 es: 02112100101022

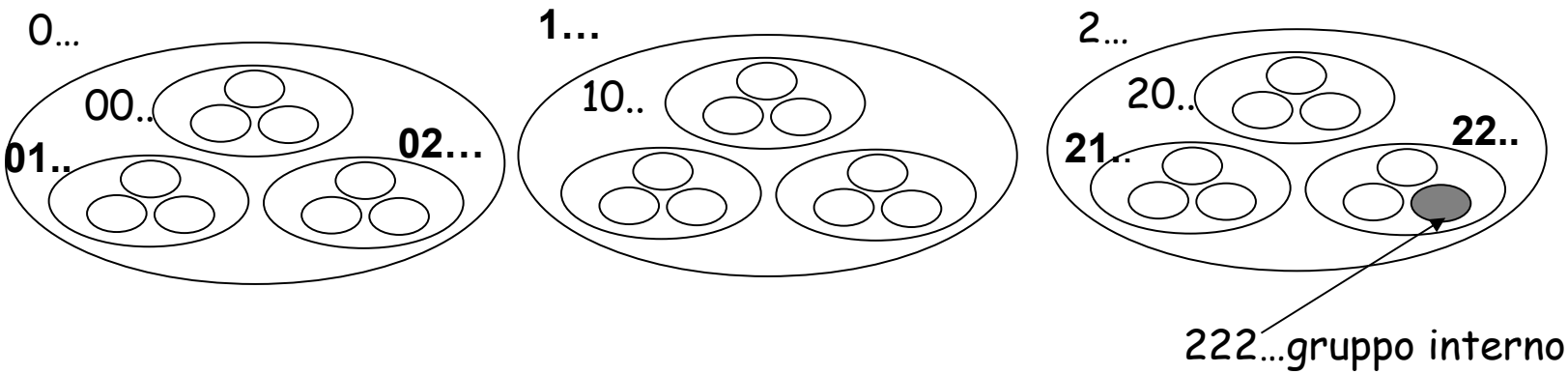
- Ogni chiave K viene memorizzata nel nodo con il valore dell'id più vicino al va lore di K

- I nodi sono logicamente raggruppati in base al loro indirizzo



PSEUDO PASTRY: CONCETTI BASE

- Ogni nodo appartenente ad uno dei gruppi più interni conosce l'indirizzo IP di ogni altro nodo dello stesso gruppo
- Ogni nodo conosce l'indirizzo IP di un nodo "rappresentante" di ogni altro gruppo
- Nodo 222….: conosce un nodo rappresentante per 0…,1…,20…,21…,220…,221…
- Più possibilità nella scelta dei nodi rappresentanti
- In questo modo il nodo 222 mantiene informazioni su 6 "rappresentanti", invece di 27

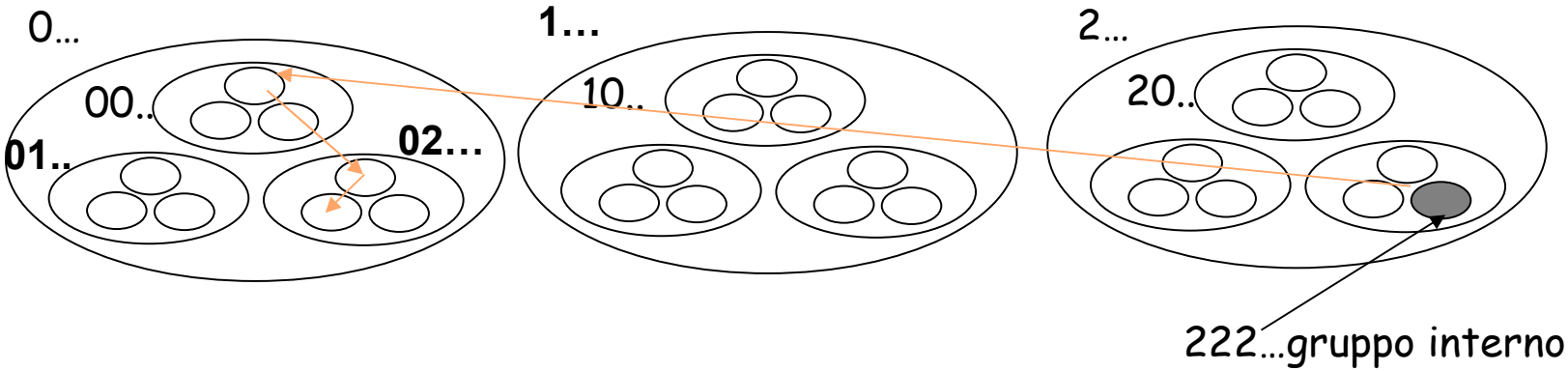


PSEUDO PASTRY:CONCETTI BASE

- Supponiamo che un nodo del gruppo 222… voglia ricercare la chiave k=02112100210. Il nodo adotta una strategia di tipo divide and conquer

- K viene inoltrata ad un nodo "rappresentante" del gruppo 0…, poi al nodo "rappresentante" di 02,…. quindi a quello di 021. La chiave viene inoltrata ad un nodo sempre più vicino, numericamente, al suo valore

- Se non si definiscono sottogruppi, 021…. inoltra la chiave al nodo numericamente più vicino ad essa tra tutti quelli del suo sottogruppo



PASTRY: TABELLE DI ROUTING

- Ai nodi vengono assegnati identificatori di 128 bits. La distribuzione degli identificatori deve essere uniforme

- Ogni identificatore in base 2^b, ad esempio $2^4 = 16$
  - Esempio 65A1FC04
  - In questo caso occorre definire 16 gruppi ed ogni gruppo definisce 16 sottogruppi e così via.

- Ogni nodo gestisce:
  - La tabella di routing che contiene un riferimento ad un solo nodo rappresentante per ogni gruppo
  - Il leaf set contiene riferimenti agli altri nodi del gruppo interno
  - Il neighborhood set : contiene riferimenti a nodi vicini in termini di distanza fisica (ad esempio: numero di IP hops,....)

PASTRY: TABELLE DI ROUTING

Tabella di routing per il nodo n=65A1FC04

0x	1x	2x	3x	4x	5x		7x	8x	9x	Ax	Bx	Cx	Dx	Ex	Fx
60x	61x	62x	63x	64x		66x	67x	68x	69x	6Ax	6Bx	6Cx	6Dx	6Ex	6Fx
650x	**651x**	**652x**	**653x**	**654x**	**655x**	**656x**	**657x**	**658x**	**659x**		**65Bx**	**65Cx**	**65Dx**	**65Ex**	**65Fx**
						.....									

- **l/b** ($\log_{2^b} N$) righe
- 2^b-1 entrate per ogni riga
- x = stringa di cifre esadecimali

Sul nodo n, gli elementi della riga i, i∈0..della tabella contengono riferimenti ai nodi il cui NodeID

- condivide un prefisso di lunghezza i con il NodeID di n
- differisce nella i-esima cifra rispetto al NodeID di n

PASTRY: TABELLE DI ROUTING

Tabella di routing per il nodo n=65A1FC04

0x	1x	2x	3x	4x	5x		7x	8x	9x	Ax	Bx	Cx	Dx	Ex	Fx
60x	61x	62x	63x	64x		66x	67x	68x	69x	6Ax	6Bx	6Cx	6Dx	6Ex	6Fx
650x	**651x**	**652x**	**653x**	**654x**	**655x**	**656x**	**657x**	**658x**	**659x**		**65Bx**	**65Cx**	**65Dx**	**65Ex**	**65Fx**
						.....									

- **l/b** ($\log_{2^b} N$) righe
- 2^b-1 entrate per ogni riga
- x = stringa di cifre esadecimali

Sul nodo n, l'elemento della riga i corrispondente alla cifra di indice i dell'identificatore del nodo

- corrisponde allo stesso gruppo del nodo
- non contiene riferimenti a rappresentanti di altri gruppi perchè la ricerca deve proseguire all'interno dello stesso gruppo del nodo

PASTRY: TABELLE DI ROUTING

Tabella di routing per il nodo n=65A1FC04

0x	1x	2x	3x	4x	5x		7x	8x	9x	Ax	Bx	Cx	Dx	Ex	Fx
60x	61x	62x	63x	64x		66x	67x	68x	69x	6Ax	6Bx	6Cx	6Dx	6Ex	6Fx
650x	**651x**	**652x**	**653x**	**654x**	**655x**	**656x**	**657x**	**658x**	**659x**		**65Bx**	**65Cx**	**65Dx**	**65Ex**	**65Fx**
						.....									

- **l/b** ($\log_{2^b} N$) righe
- 2^b-1 entrate per ogni riga
- x = stringa di cifre esadecimali

- Alcune entrate della tabella possono risultare vuote

- Una entrata risulta vuota se non esiste sull'overlay un nodo con prefisso corrispondente a quell'entrata

PASTRY:TABELLA DI ROUTING

Node10233102 (2), (b = 2, l = 8)

0	1	2	3
02212102		22301203	31203203
	11301233	12230203	13021022
10031203	10132102		10323302
10200230	10211302	1022302	
10230322	10231000	10232121	
10233001		10233232	
		10233120	

9

PASTRY: TABELLE DI ROUTING

- Alcune entrate della tabella di Routing possono essere vuote

- Approccio analogo a Chord. Ogni tabella di routing contiene una conoscenza:
  - approssimata rispetto ai nodi distanti nello spazio degli identificatori
  - più accurata dei nodi vicini

- Proximity Routing
  - la scelta dei nodi da inserire nella routing table di un nodo è guidata da un criterio di prossimità
    - ping delay
    - numero di IP hops
  - ogni elemento della tabella di routing di n si riferisce ad un nodo vicino ad n (con riferimento alla nozione di prossimità utilizzata), tra tutti i nodi con il prefisso appropriato per quella entrata

PASTRY: LEAF SET

- Leaf Set L di un nodo n identificato da un identificatore ID: memorizza riferimenti (indirizzi IP) agli |L| nodi che posseggono gli identificatori più vicini ad ID
  - La metrica ultilizzata si riferisce allo spazio logico degli identificatori
- Svolge un ruolo simile alla lista dei successori di Chord
- In generale $|L|=2^b$
- Di solito è partizionata in due insiemi che contengono riferimenti ad |L/2| identificatori maggiori di ID ed |L/2| identificatori minori id ID.
- In generale non contiene tutti gli identificatori di un gruppo interno, per cui la tabella di routing deve comunque contenere l/b righe
- Prima di utilizzare la tabella di routing ogni nodo controlla se la chiave è compresa nell'intervallo definito dal leaf set

PASTRY: LEAF SET

Tabella di routing per il nodo n=65A1FC04

0x	1x	2x	3x	4x	5x		7x	8x	9x	Ax	Bx	Cx	Dx	Ex	Fx
60x	61x	62x	63x	64x		66x	67x	68x	69x	6Ax	6Bx	6Cx	6Dx	6Ex	6Fx
650x	**651x**	**652x**	**653x**	**654x**	**655x**	**656x**	**657x**	**658x**	**659x**		**65Bx**	**65Cx**	**65Dx**	**65Ex**	**65Fx**
						.....									

- **l/b** ($\log_{2^b} N$) righe
- 2^b-1 entrate per ogni riga
- x = stringa di cifre esadecimali

Leaf Set (16 elementi)= 65A1FB80, 65A1FB86, 65A1FB91,.....65A1FC07, 65A1FC13,....

PASTRY:LO STATO COMPLETO DI UN NODO

NodeId 10233102

Leaf set	SMALLER	LARGER	
10233033	10233021	10233120	10233122
10233001	10233000	10233230	10233232

Routing table			
-0-2212102	1	-2-2301203	-3-1203203
0	1-1-301233	1-2-230203	1-3-021022
10-0-31203	10-1-32102	2	10-3-23302
102-0-0230	102-1-1302	102-2-2302	3
1023-0-322	1023-1-000	1023-2-121	3
10233-0-01	1	10233-2-32	
0		102331-2-0	
		2	

Neighborhood set			
13021022	10200230	11301233	31301233
02212102	22301203	31203203	33213321

PASTRY:NEIGHBOROOD SET

- Neighborhood set M = Contiene identificatori ed indirizzi IP dei nodi più fisicamente più vicini
- M non viene utilizzato per il routing
- $|M| = 2^b$ (in generale)

NodeID 10233102

Neighborhood set			
13021022	10200230	11301233	31301233
02212102	22301203	31203203	33213321

Gli identificatori(logici) dei nodi nel neighbourood set non hanno alcuna relazione con l'identificatore del nodo

PASTRY: L'ALGORITMO DI ROUTING

Routing effettuato sul nodo n di identificatore ID

if (KEY appartiene all'intervallo definito dal leaf set)
inviare key al nodo del leaf set il cui NodeID è numericamente più vicino a KEY

else

{

if (esiste nella tabella di routing un identificatore ID' che condivide con la chiave un prefisso più lungo del prefisso comune tra ID e KEY)

invia KEY al nodo identificato da ID'

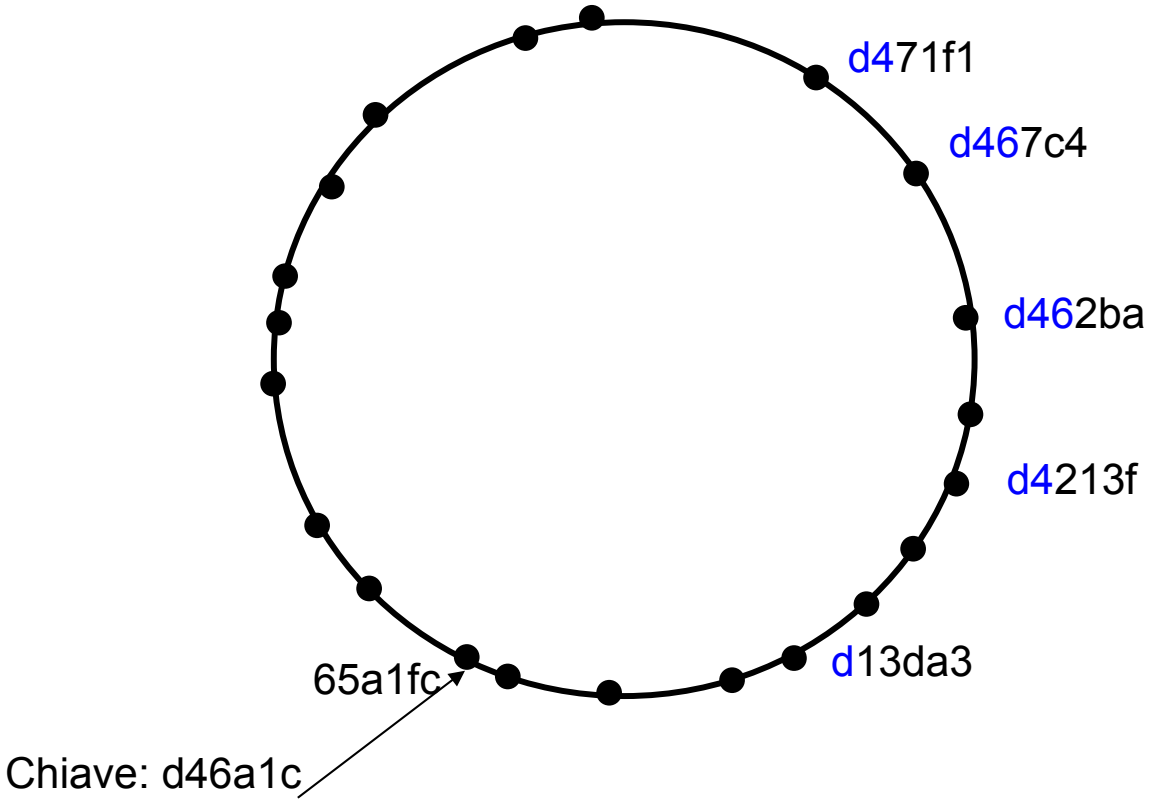
else

invia KEY ad un nodo che

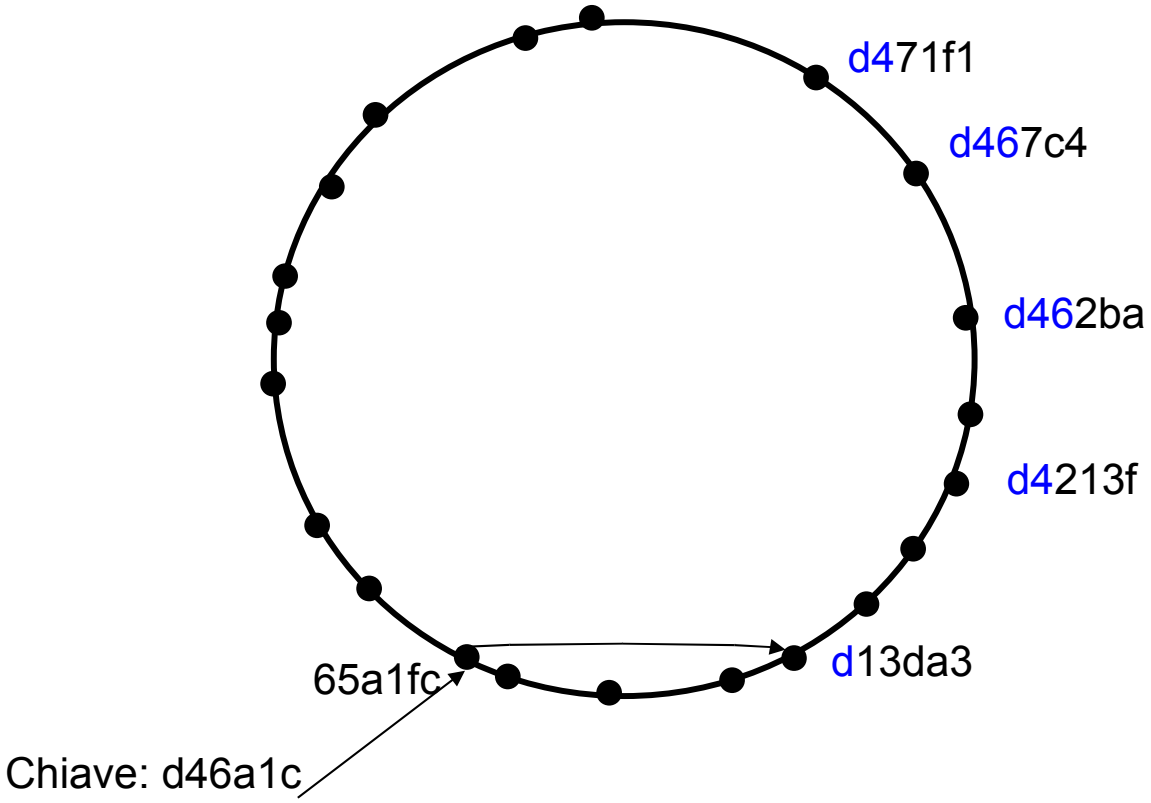
condivida con la chiave un prefisso di lunghezza pari al prefisso comune tra ID e KEY e sia numericamente più vicino alla chiave

}

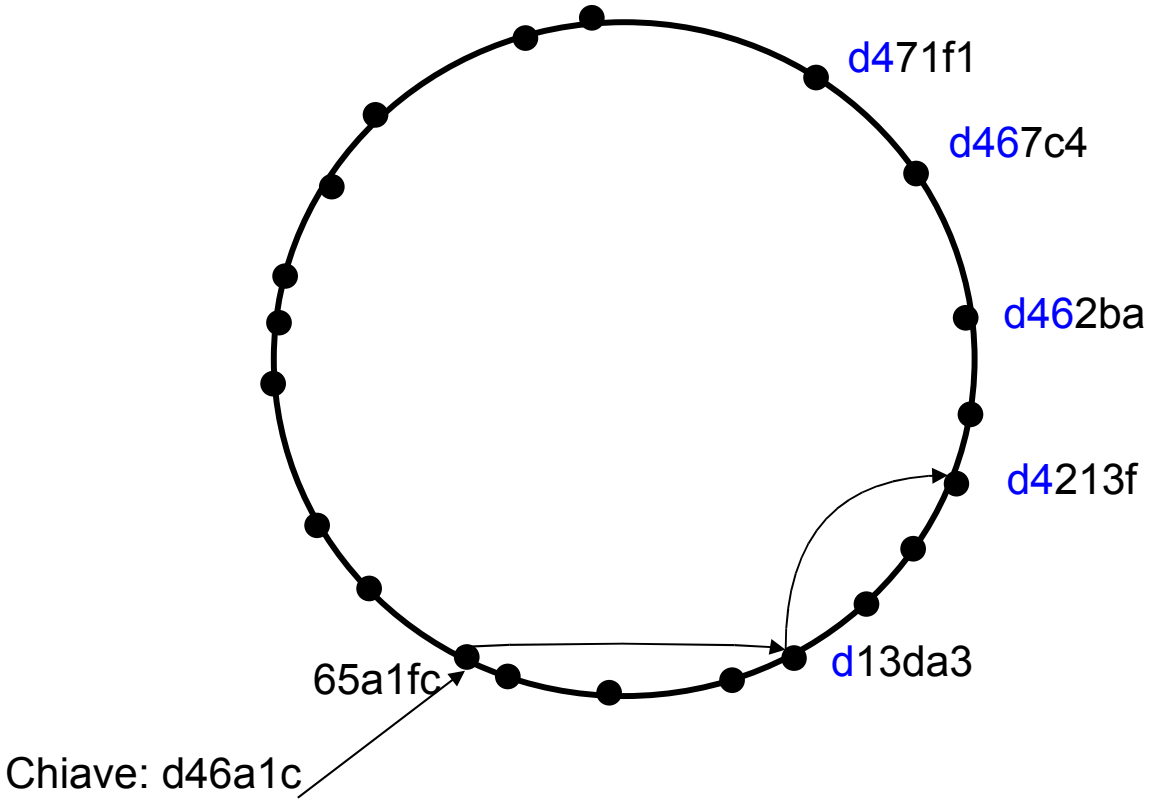
PASTRY: L'ALGORITMO DI ROUTING



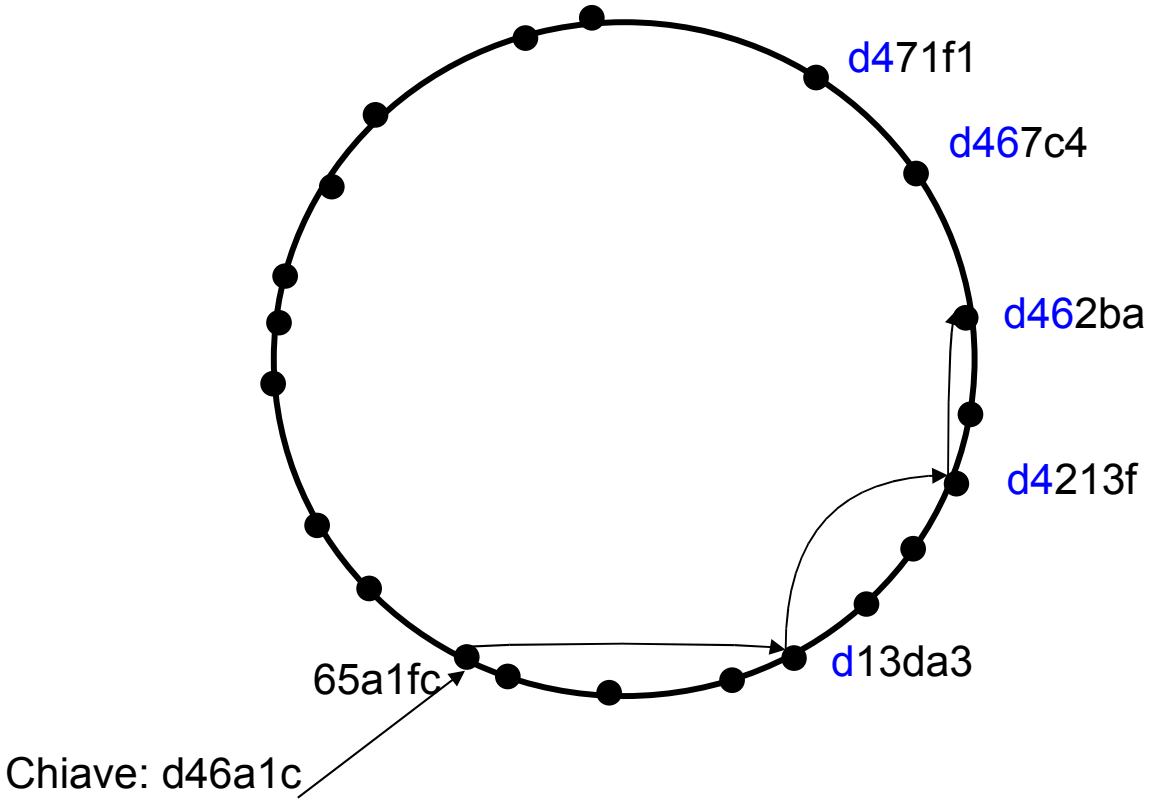
PASTRY: L?ALGORITMO DI ROUTING



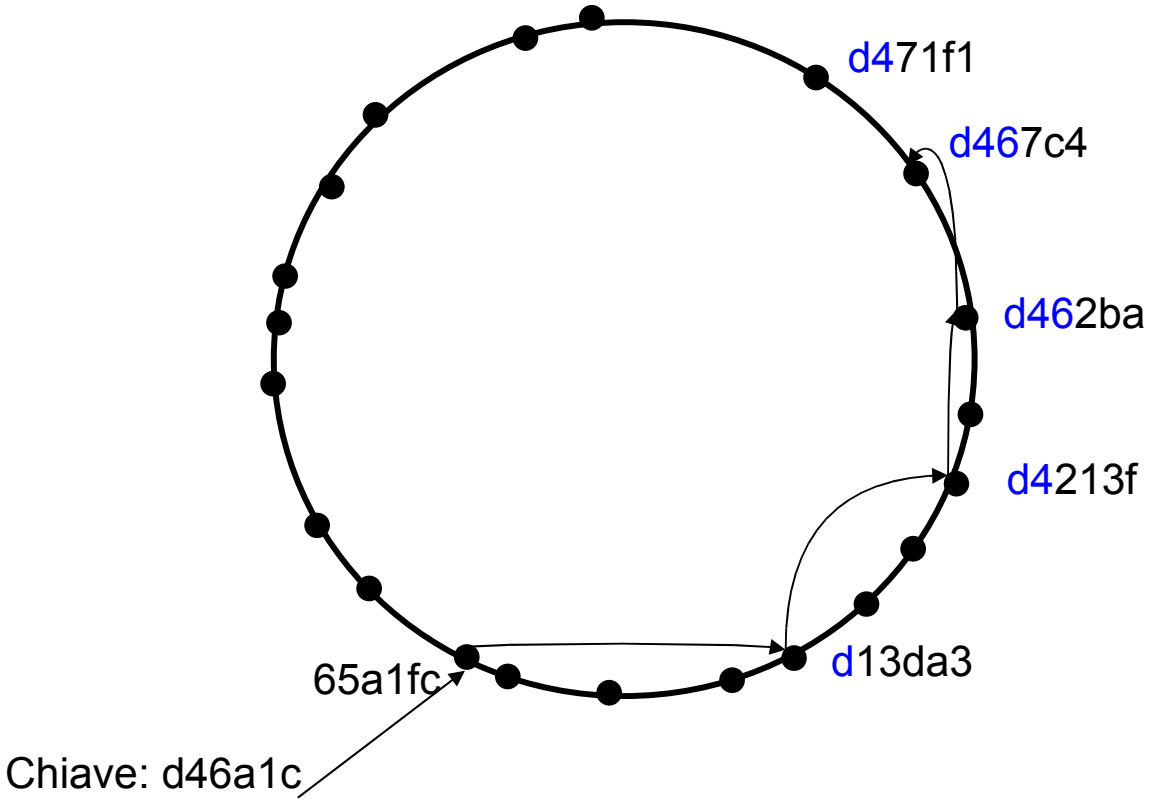
PASTRY:L?ALGORITMO DI ROUTING



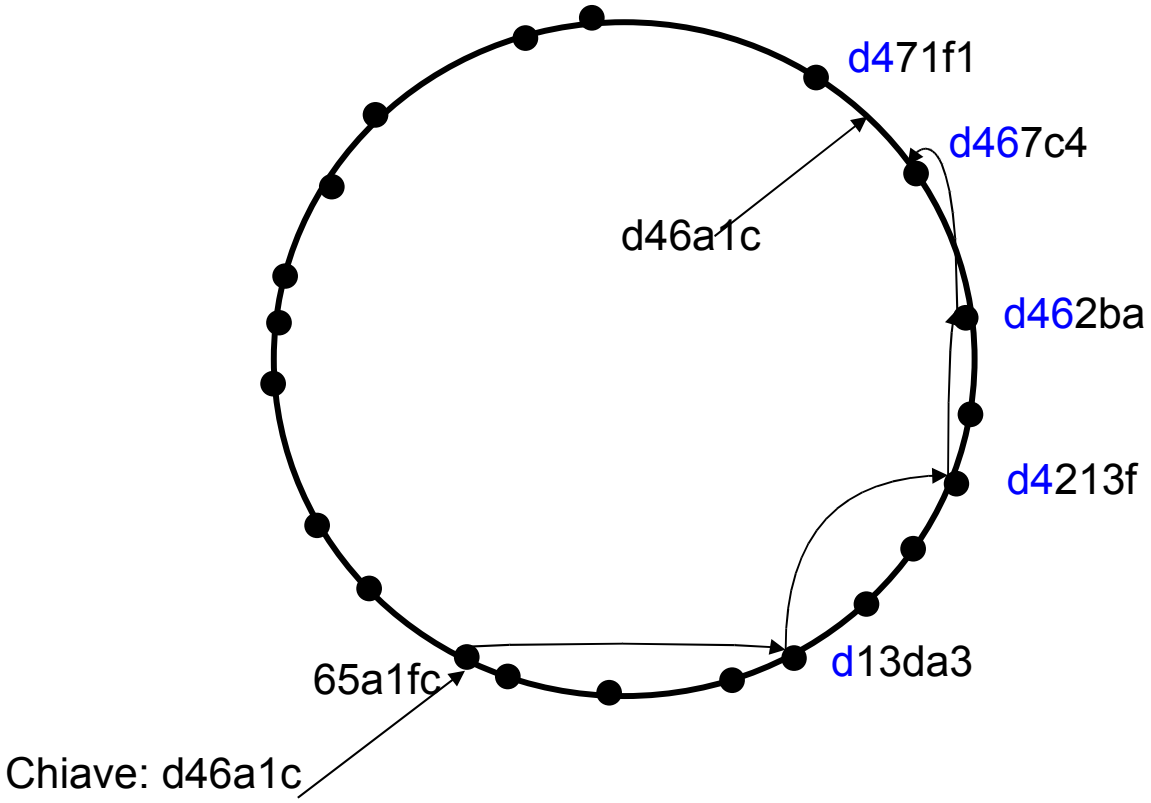
PASTRY: L'ALGORITMO DI ROUTING



PASTRY:L?ALGORITMO DI ROUTING



PASTRY:L'ALGORITMO DI ROUTING



PASTRY: L'ALGORITMO DI ROUTING

Tabella di routing per il nodo n=65a1fc04

0x	1x	2x	3x	4x	5x		7x	8x	9x	ax	bx	cx	dx	ex	fx
60x	61x	62x	63x	64x		66x	67x	68x	69x	6ax	6bx	6cx	6dx	6ex	6fx
650x	**651x**	**652x**	**653x**	**654x**	**655x**	**656x**	**657x**	**658x**	**659x**		65bx	**65cx**	**65dx**	**65ex**	**65fx**
						.....									

n riceve una richiesta per la chiave k=65b23c05

- 65 = prefisso comune tra il nodo e la chiave
- n invia la chiave al nodo evidenziato della tabella, perché questo nodo condivide un prefisso più lungo con la chiave (65b)

PASTRY:L'ALGORITMO DI ROUTING

Tabella di routing per il nodo n=65a1fc04

0x	1x	2x	3x	4x	5x		7x	8x	9x	ax	bx	cx	dx	ex	fx
60x	61x	62x	63x	64x		66x	67x	68x	69x	6ax	6bx	6cx	6dx	6ex	6fx
	651x	**652x**	**653x**	**654x**			**657x**	**658x**	**659x**		**65bx**	**65cx**	**65dx**		**65fx**
						.....									

Ipotesi: la parte non evidenziata della tabella è vuota

n riceve una richiesta per la chiave 65523c05

- 65 = prefisso comune tra nodo e chiave
- non esiste un riferimento ad un nodo che condivide un prefisso più lungo con la chiave,
- la chiave viene inviata al nodo evidenziato in tabella, perché il suo valore numerico è più vicino alla chiave

PASTRY: INSERIMENTO DI NUOVI NODI

Join: un nuovo nodo X deve inizializzare il proprio leaf set e la propria routing table ed informare gli altri nodi della propria presenza

- X sceglie un identificatore ID, mediante consistent hashing ed un peer PB per il bootstrap. PB deve essere vicino ad X, secondo la metrica scelta per la prossimità fisica
- invia a PB un messaggio join(key=ID)
- il messaggio viene inoltrato verso il nodo Y numericamente più vicino ad ID
- tutti i nodi sul cammino percorso dal messaggio join(key=ID) inviano le informazioni contenute nelle proprie routing tables e nei propri leaf set a X
- X costruisce le proprie tabelle a partire dalle informazioni ricevute

PASTRY: INSERIMENTO DI NUOVI NODI

- la tabella di routing di X viene costruita a partire dalle tabelle di routing di tutti i nodi sul cammino percorso dal messaggio join(key=ID)

- Il leaf set di X può essere costruito a partire da quello di Y, cioè il nodo numericamente più vicino ad X, individuato mediante il rooting

- Nel caso generale in cui l'i-esima cifra della chiave ID differisce dalla i-esima cifra del nodo incontrato all'i-esimo hop :

  La riga i-esima della tabella di routing di X può essere costruita a partire dalla riga i-esima della tabella di routing dell'i-esimo nodo incontrato sul cammino del messaggio join(key=ID)

PASTRY:INSERIMENTO DI NUOVI NODI

- X= nuovo nodo, Z= nodo numericamente più vicino ad X, A = nodo di bootstrap, vicino fisicamente a X
- X invia un messaggio di join(X) ad A
- A inoltra il messaggio ricevuto a B→C→....fino a Z, che è il nodo numericamente più vicino ad X
- A,B,...Z inviano una parte delle proprie tabelle di routing a X
- Il neighbourood set di X è uguale a quello di A
- Il Leaf Set di X è uguale a quello di Z
- La tabelle di routing di X viene riempita come segue
  - La riga 0 è uguale alla riga 0 di quella di A ($X_0 = A_0$)
  - La riga 1 è uguale alla riga 1di quella di B ($X_1 = B_1$)
  - etc....
- Infine X manda il proprio identificatore ad ogni nodo conosciuto (tabella di routing, leaf set, neighborood set,...)

PASTRY:INSERIMENTO DI NUOVI NODI

- Se ad un certo hop, più cifre del nodo incontrato n e della chiave k coincidono, più righe di n possono essere copiate nella tabella di routing del nodo che effettua la join( )

- Poichè il nodo di boostrap viene scelto vicino ad X, il neighborood set di X viene costruito a partire da quello di PB

- X trasmette anche una copia del proprio stato ad ognuno dei nodi nella sua routing table e nel suo leaf set

- Questi nodi possono a loro volta aggiornare il proprio stato

PASTRY: INSERIMENTO DI NUOVI NODI

Tabella di routing per il nodo n=65a1fc03, il nodo riceve il messaggio join(key=75b2ff03)

0x	1x	2x	3x	4x	5x		7x	8x	9x	ax	bx	cx	dx	ex	fx
60x	61x	62x	63x	64x		66x	67x	68x	69x	6ax	6bx	6cx	6dx	6ex	6fx
	651x	**652x**	**653x**	**6544x**			**657x**	**658x**	**659x**		**65bx**	**65cx**	**65dx**		**65fx**
						.....									

Ipotesi: la parte non evidenziata della tabella è vuota

- la prima riga della tabella di routing contiene riferimenti a nodi rappresentanti di tutti i 16 gruppi (o,..f)
- questa riga (nella posizione vuota viene inserito un riferimento al nodo n) viene inviata al nodo 75b2ff03 che la utilizza per inizializzare la prima riga della propria tabella di routing, poi il msg viene inviato al nodo 7x

PASTRY: INSERIMENTO DI NUOVI NODI

Tabella di routing per il nodo n=76a1fc03, il nodo riceve il messaggio join(key=75b2ff03)

0x	1x	2x	3x	4x	5x	6x		8x	9x	ax	bx	cx	dx	ex	fx
70x	71x	72x	73x	74x	75x		77x	78x	79x	7ax	7bx	7cx	7dx	7ex	7fx
	761x	**762x**	**763x**	**764x**			**767x**	**768x**	**769x**		**76bx**	**76cx**	**76dx**		**76fx**

Ipotesi: la parte non evidenziata della tabella è vuota

- la seconda riga della tabella di routing di 76a1fc03 contiene riferimenti a nodi il cui identificatore presenta la prima cifra uguale a quella del nodo di join( ) e differisce per la seconda cifra
- la seconda riga della tabella di routing di 76a1fc03 può essere utilizzata per inizializzare la seconda riga della tabella di routing di 75b2ff03

PASTRY: COMPLESSITA'

Analisi della complessità

- Rete Pastry di N nodi. Identificatori di l-bits, valori in base $B=2^{b,}$
- Complessità ricerca chiavi: $O(\log_B (N))$
- Complessità tabelle di routing: $O(\log_B(N))x(B-1))$
- Valore di B scelto come tradeoff tra numero medio di passi per la ricerca e dimensione delle tabelle di routing (in generale b=4)

PASTRY: LOCALITA'

- Pastry non ottimizza solamente il numero di passi di routing, ma anche il costo di ogni hop
- La riga i-esima della tabella di routing contiene riferimenti ai nodi che condividono un prefisso lungo i con il nodo.
- Più scelte possibili per ogni elemento per la tabella
- Definizione di una nozione di prossimità fisica
- Ogni elemento della tabella di routing di n contiene un riferimento ad un nodo vicino ad n, secondo la nozione di prossimità introdotta, scelto tra tutti i nodi caratterizzati dal prefisso opportuno

PASTRY: LOCALITA'

- Metrica per misurare la prossimità fisica: ping delay, numero di hops IP, internet coordinates

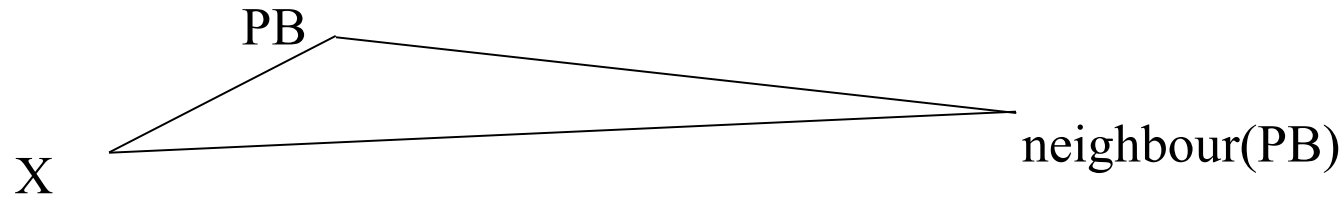
- Si suppone che l'applicazione che utilizza Pastry utilizzi una funzione che permetta di calcolare la distanza tra due nodi, secondo la metrica definita
  - ping
  - traceroute....

- In uno spazio metrico è euclideo vale la disuguaglianza triangolare:
  dati tre nodi A, B, C, d(A,B)+d(B,C) > d(A,C), dove d è la funzione che misura la distanza secondo la metrica definita

- Se vale la disuguaglianza (non sempre vale in una rete) PASTRY sfrutta la località fisica tra i nodi, altrimenti il routing rimane comunque corretto, ma non necessariamente sfrutta la località fisica.

PASTRY:LOCALITA'NELLE TABELLE DI ROUTING

- Come si garantisce che le entrate nella tabella di routing siano scelte secondo la nozione di prossimità fisica?
- Supponiamo che questa proprietà valga prima dell'inserimento di un nodo, e vediamo cosa accade quando si inserisce il nodo X
- X contatta un bootstrap nodo PB 'vicino' e copia la prima riga della sua tabella di routing
- Poiché
  - X è vicino a PB
  - le entrate della tabella di routing di PB contengono nodi vicini a PB
  - Se vale la disuguaglianza triangolare allora la riga 0 della tabella di routing di X conterrà nodi vicini ad X



d(X, neighbour(PB)) < d(X,PB)+d(PB,neighbour(PB))

PASTRY: LOCALITA' NELLE TABELLE DI ROUTING

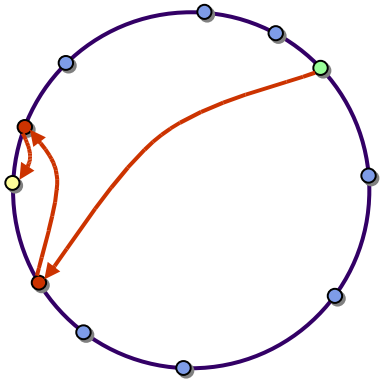
Per le righe successive:

- La riga i-esima è ottenuta dall'i-esimo nodo N_i sul cammino del join( )
- Questa riga contiene riferimenti a nodi vicini ad N_i, che non è detto siano anche vicini ad X (il nodo che si inserisce nell'overlay)

Euristica::

- I primi hops di routing sono i più importanti, perchè mi consentono di scegliere tra un insieme ampio di nodi
- Ai passi successivi
  - la scelta decresce esponenzialmente, perchè ad ogni hop escludo un gruppo di nodi ed ho meno nodi da scegliere
  - la scelta del nodo più vicino diventa via via meno importante perchè le distanze con i vicini saranno comunque grandi rispetto a quelle individuate ai passi precedenti

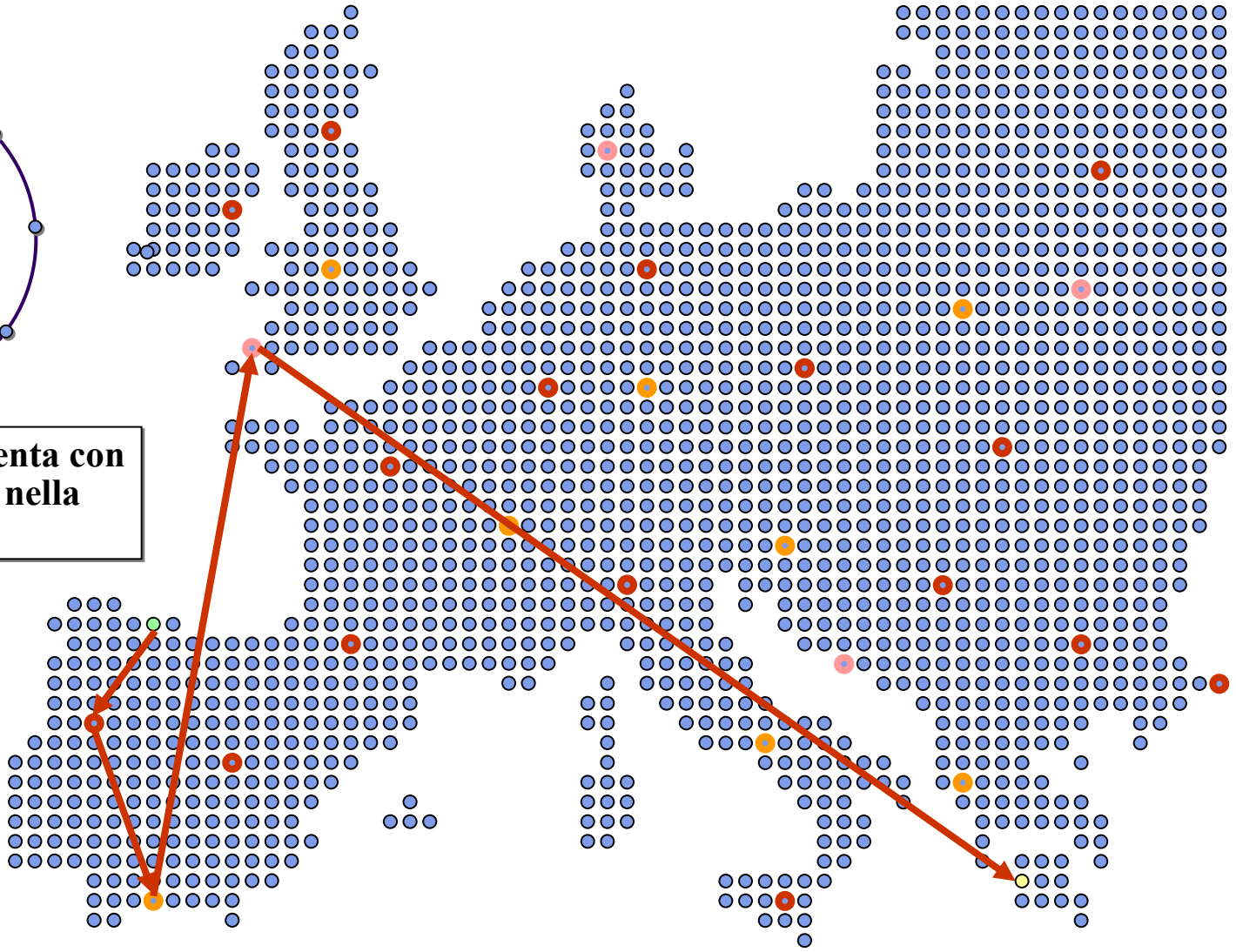
PASTRY:LOCALITA'



La distanza aumenta con il numero di riga nella routing table

I salti numerici diminuiscono

I salti topologici aumentano



PASTRY: LOCALITA' NELLE TABELLE DI ROUTING

- Pastry:
  - Ottimizza la località nel primo passo di routing
  - definisce altre procedure per migliorare prograssivamente la qualità dell'approssimazione

- Ogni nodo periodicamente
  - richiede lo stato dei nodi contenuti nella propria routing table
  - scambia il proprio neighborood set con i nodi contenuti nella sua routing table
  - confronta la distanza (ping time) dai nuovi nodi con quella dai nodi contenuti nella propria tabella di routing ed eventualmente aggiorna gli elementi della tabella di routing con i nuovi nodi acquisiti
  - i nuovi nodi inseriti nella tabella di routing devono ovviamente essere caratterizzati dai prefissi opportuni

PASTRY: LOCALITA' NEL ROUTING

- Il routing Pastry non garantisce di individuare il cammino minimo tra due nodi
- Gode comunque delle seguenti proprietà:
  - Se un messaggio viene spedito da un nodo A ad un nodo B che si trova a distanza d da A, il messaggio non viene poi successivamente inoltrato ad un nodo che dista meno di d da A
  - La distanza percorsa da ogni messaggio in ogni hop aumenta esponenzialmente. Questo è conseguenza dei seguenti fatti:
    - le entrate in righe successive della tabella di routing vengono scelte da insiemi di nodi la cui cardinalità diminuisce esponenzialmente
    - la distribuzione è uniforme