MULTI-ATTRIBUTE, RANGE QUERIES IN P2P SYSTEMS

Contatti

Matteo Mordacchini

matteo.mordacchini@isti.cnr.it

Gestione di risorse in sistemi P2P

- I sistemi P2P si sono dimostrati nel tempo un paradigma efficiente per la condivisione di risorse
- In particolare, hanno trovato applicazione per la condivisione di file, risorse computazionali, ecc.

Gestione di risorse in sistemi P2P

- Generalmente, queste risorse vengono caratterezzate ed indicizzate sulla base di un unico attributo
  - Es. File → Nome del file
- Le interrogazioni vengono fatte ricercando un valore esatto tra quelli indicizzati

Gestione di risorse in sistemi P2P

Problemi

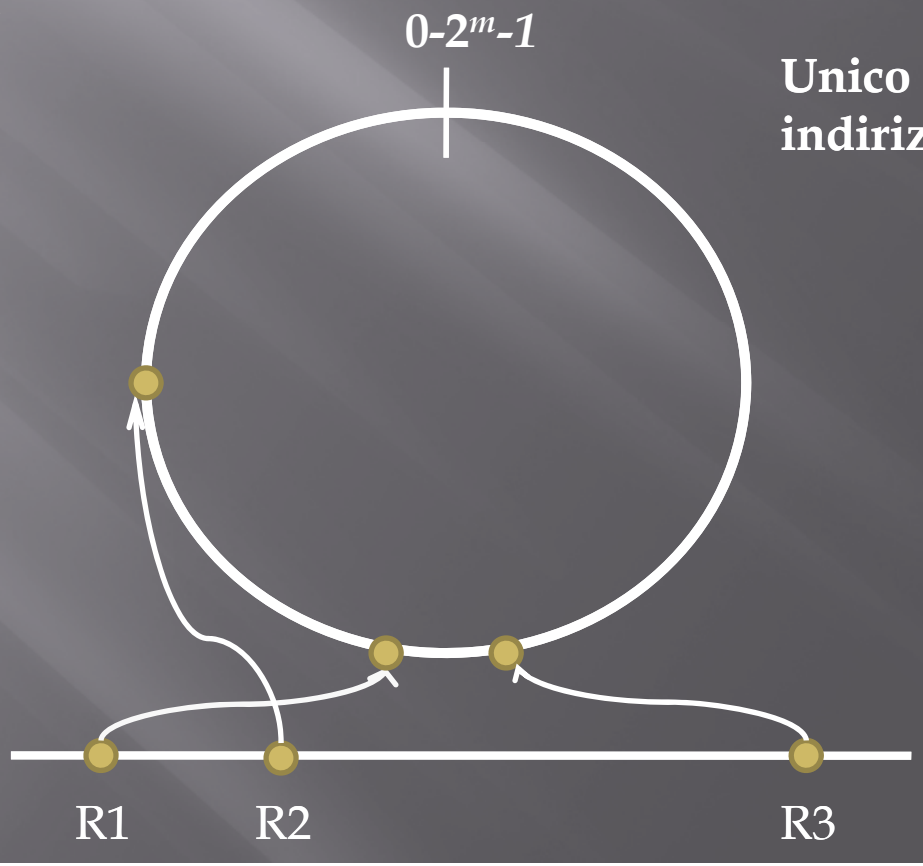
1. Molte risorse sono caratterizzate da insiemi di attributi
   - File:
     - Nome
     - Autore
     - Dimensione …
   - Risorse Computazionale:
     - Vel. CPU
     - Dimensione RAM
     - Dimensione disco …
   - …

Gestione di risorse in sistemi P2P

2. Su ogni singolo attributo, potrebbe essere necessario richiedere insiemi di risorse i cui valori siano compresi in un determianto intervallo
   - Esempi:
     - Vel. CPU ≥ velMin
     - Dimensione File ≤ dimMax
     - Data1 ≤ CreazioneFile ≤ Data2

Gestione di risorse in sistemi P2P

- Sistemi P2P molto efficienti, come le DHT, non trattano direttamente entrambi questi problemi!!
- Es. Chord



Gestione di risorse in sistemi P2P

- Entrambi questi aspetti rappresantano caratteristiche e richieste molti comuni per numerosi tipi di risorse
- È importante avere sistemi in grado di affrontarle contemporaneamente

Gestione di risorse in sistemi P2P

- Le soluzioni che vedremo tentano di combinare i vantaggi di reti come le DHT con la possibilità di trattare più attributi con ricerche su intervalli

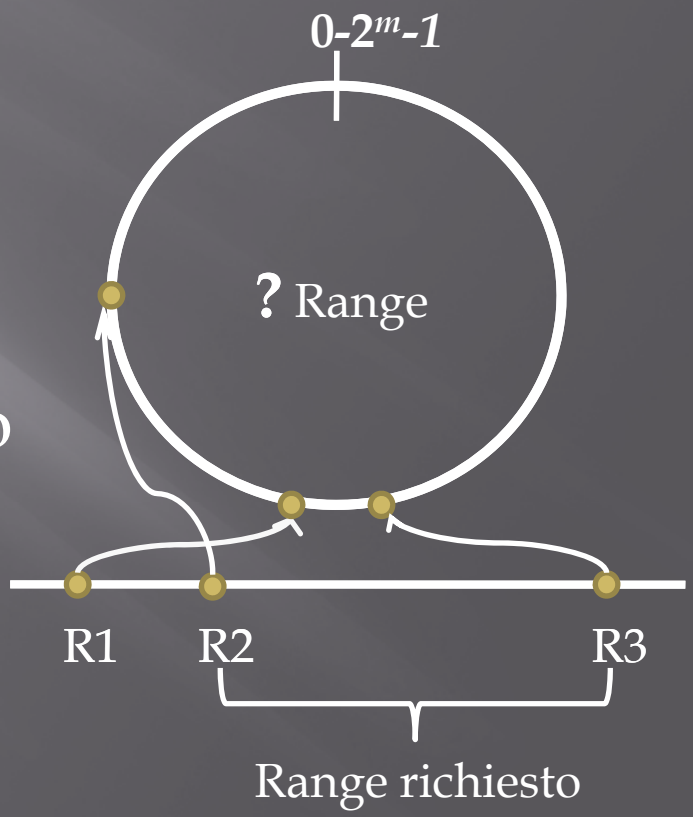
- Per farlo, affrontano i problemi uno alla volta
  - Ricerche su intervalli
  - Gestioni di più atributi

P2P Range Queries

- Nei sistemi P2P ogni nodo gestisce una parte degli indici delle risorse dell'intero sistema
- Gli indici sono assegnati in base a determinati criteri (ad es. funzioni hash) che mappano i valori delle risorse nello spazio di indirizzamento della rete
- La mappatura non tiene conto della disposizione dei valori nel loro dominio
- Fare ricerche su intervalli può diventare particolarmente dispendioso

P2P Range Queries

- Non conoscendo la distribuzione degli oggetti nello spazio degli indici, potrebbe essere necessario un **flooding** per poterli recuperare tutti



Locality-preserving Hash Functions

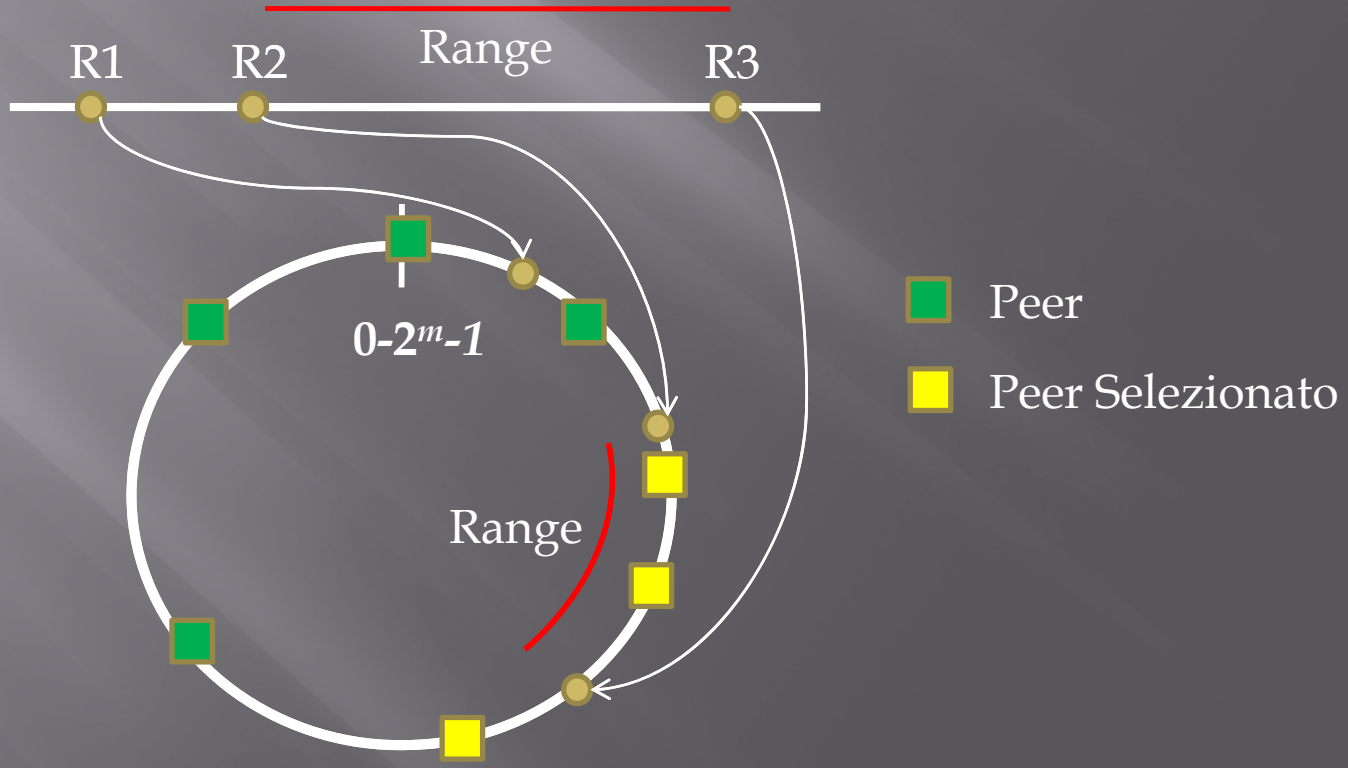
- È necessario disporre di funzioni di mappatura che rispettino l'ordine dei valori nel dominio
- Data una funzione H e dati $x,y \in D$ con $x \leq y$, si vuole che

$$H(x) \leq H(y)$$

- È possibile creare funzioni di questo tipo che distribuiscano i valori in modo uniforme nel codominio

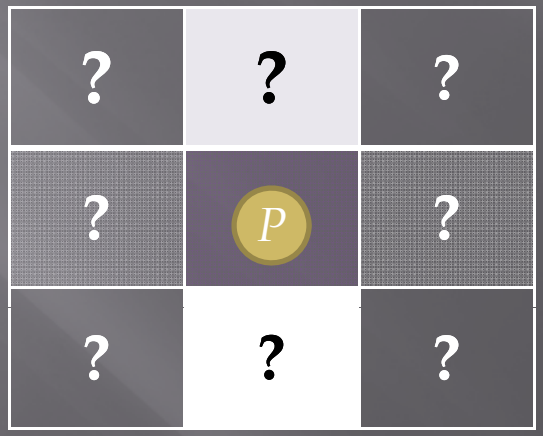
Locality-preserving Hash Functions

- È così possibile determinare quali sono i nodi coinvolti in una range query, riducendo il numero di messaggi richiesti



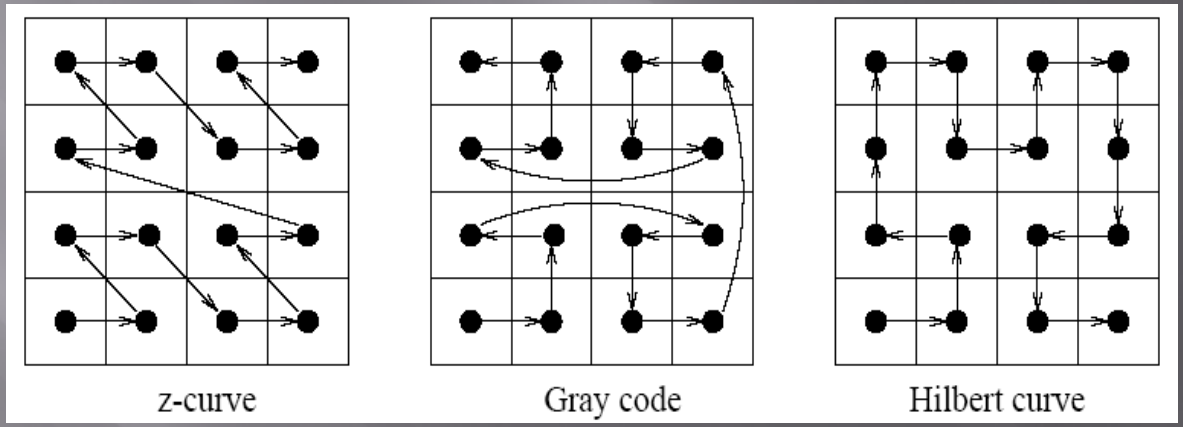
Space-filling Curves

- È possibile determinare un ordinamento di questo tipo anche per domini a più di una dimensione?
- Occorre stabilire un criterio per determinare l'ordinamento dei punti nello spazio
- Es. In quale ordine vanno presi i vicini di *P* ?



Space-filling Curves

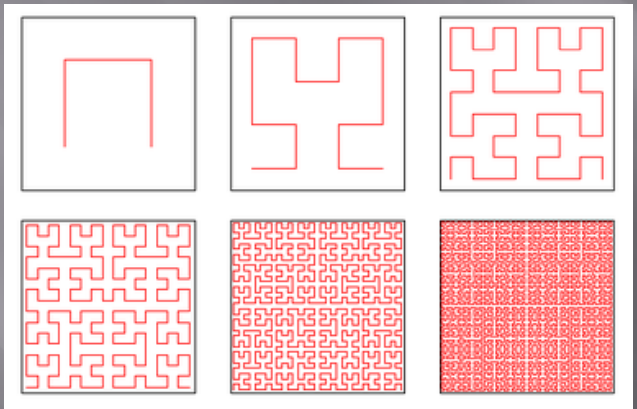
- Le **space-filling curves** (o **curve di Peano**) sono funzioni parametriche suriettive che determinano curve in grado di "riempire" un dato spazio *n*-dimensionale

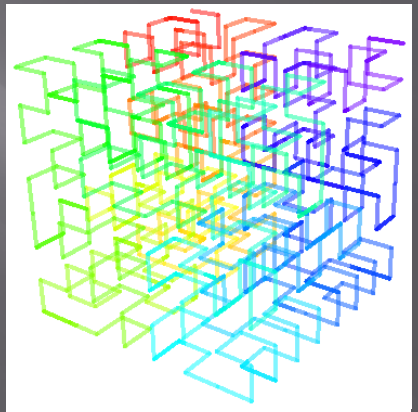


- Il loro andamento permette di stabilire un ordine di percorrenza dello spazio

Curva di Hilbert

- I parametri delle SFC permetto di stabile coperture più o meno fitte del piano
- Tra le **space-filling curves**, la più usata è la cosiddetta **curva di Hilbert**





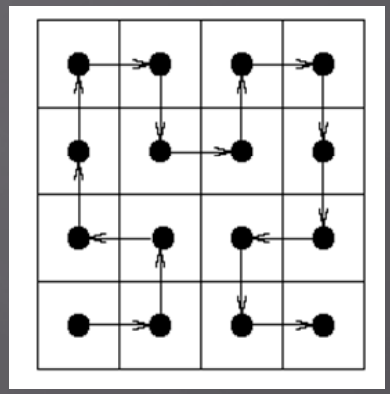
- I suoi parametri consentono di stabilire coperture più fitte anche solo per alcune zone dello spazio

Estensione di CAN 1/8

- Le SFC possono essere usate per mappare spazi di attributi *n*-dimensionali in reti DHT, assegnando porzioni dello spazio ai peer della rete
- Una prima applicazione può essere fatta estendendo il protocollo di CAN per trattare range queries su dati multi-attributo

Estensione di CAN 2/8

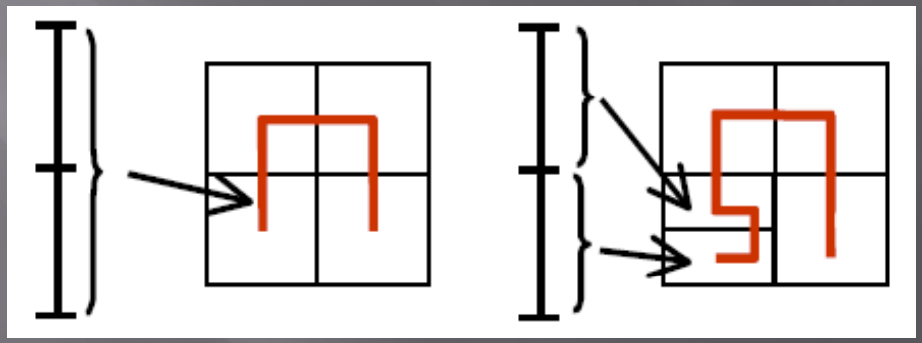
- Attraverso una generalizzazione in *n*-dimensioni della curva di Hilbert, si divide lo spazio in iper-rettangoli



- Ogni iper-rettangolo viene assegnato ad un peer, detto *Interval Keeper* (IK).
- Ogni IK è responsabile delle risorse i cui valori congiunti degli attributi ricadono nell'intervallo assegnatogli

Estensione di CAN 3/8

- La rete può gestire così siano risorse multi attributo che range-queries
- Nel caso in cui i dati gestiti da un IK siano troppo numerosi o siano soggetti a update troppo frequenti, la sua zona può essere adattata dinamicamente, divindendola con un altro IK.



Estensione di CAN 4/8

Strategie di Routing

- Data una range query con lower bound l e upper bound u, la query viene inizialmente diretta al IK che gestisce la zona dove cade il punto centrale $(l+u)/2$

- Questo provvede a trasmetterla agli altri IK coinvolti usando 3 diverse possibili strategie

Estensione di CAN 5/8

Forza Bruta

- Si calcola il più piccolo ipercubo che contenga tutte le zone richieste nella query
- Si calcola quali siano gli IK coinvolti e si passa la richiesta a tutti questi ultimi.
- Pros: semplice da realizzare e con alto grado di parallelismo
- Cons: alto numero di messaggi richiesti

Estensione di CAN 6/8

Flooding controllato

- Il nodo iniziale passa la richiesta solo ai vicini le cui zone abbiano un intersezione non nulla con la query
- Pros: vengono contattati solo IK che effettivamente sono correlati con la query
- Cons: possibile ripetizione dei messaggi

Estensione di CAN 7/8

Flooding controllato diretto

- Il nodo iniziale fa partire due "ondate" di propagazione della query verso i vicini le cui zone abbiano un intersezione non nulla con la query
  - Una verso quelli che intersecano la query maggiormente di quella del nodo corrente
  - Una verso quelli che la intersecano per un'area minore
- Pros: minor duplicazione dei messaggi

Estensione di CAN 8/8

- Pros: riesce ad estendere una rete DHT già esistente cercando di preservarne le caratteristiche positive
- Cons: che succede se le query riguardano solo un sottoinsieme ridotto degli attributi?

Estensione di CAN

Riferimenti

A. Andrzejak, Z. Xu, *Scalable Efficent Range Queries for Grid Information Services*, 2nd IEEE International Conference on Peer-to-Peer Computing, Linköping, Sweden, 5-7 Sept. 2002

MAAN – Multi-Attribute Addressable Network

- Il problema riscontrato per il network presentato in precedenza è dovuto al fatto che tutti gli attributi sono mappati nello stesso spazio
- Una possibile soluzione potrebbe consistere nel trattare ogni attributo separatamente

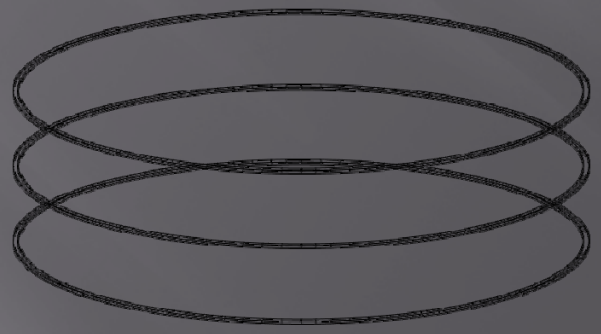
MAAN 1/7

- MAAN (Multi-Attribute Addressable Network) presenta un'estensione di Chord
- Ogni attributo è mappato in un proprio layer Chord
- I valori di ogni layer sono mappati usando funzioni hash uniformi locality-preserving

Attributo 1

Attributo 2

Attributo3



MAAN 2/7

- Per usare queste funzioni di hash, i limiti del dominio devono essere noti a priori
- Ogni risorsa è registrata in tutti i layer corrispondenti agli attributi che la caratterizzano
- Ogni risorsa si registra come una coppia

  <*val. attributo; info risorsa*>
- La seconda parte contiene tutte le informazioni della risorsa, compresi i valori degli altri attributi

MAAN 3/7

- In ogni layer, i nodi si occupano di mantenere gli indici delle risorse che cadono nello spazio di loro competenza
- Ogni nodo può prendere parte a più layer contemporaneamente
- In questo caso, mantiene Finger Table separate per ogni layer

MAAN 4/7

Strategie di Routing: Iterative Query Resolution

- Ogni query è divisa in *M* sotto-query, una per ognuno degli attributi che la compongono.
- Per ogni sotto-query, viene interrogato il layer relativo
- Il primo nodo che gestisce il lower bound del range richiesto viene trovato con il classico meccanismo di routing di Chord
- Si interrogano tutti i nodi successivi fino a quando non si raggiunge il nodo che gestisce l'upper bound del range

MAAN 5/7

Strategie di Routing: Iterative Query Resolution

- Da ogni layer vengono inviati i risultati al nodo da cui è partita la query
- Questi interseca i risultati per trovare l'insieme di risultati che rispetta i vincoli su tutti gli attributi richiesti
- La complessità è

$$O(\sum_{i=1}^{M} (\log N + N * s_i))$$

MAAN 6/7

Strategie di Routing: Single Attribute Dominated Query Resolution

- Il nodo da cui parte la query seleziona, tra gli attributi richiesti, quello che coinvolge il range più piccolo (maggiore selettività)
- La query viene inviata solamente al layer di questo attributo
- Grazie alla replicazione delle informazioni, i risultati vengono filtrati localmente, restituendo sole le risorse che rispettano tutti i vincoli
- La complessità è $O(\log N + N * s_{min})$,

MAAN 7/7

- Pros: La seconda strategia di routing è molto efficiente e consente di interrogare un numero estremamente ridotto di nodi
- Cons:
  - Gli attributi devono essere replicati
  - Gestire più layer può molto costoso con nodi dinamici

MAAN

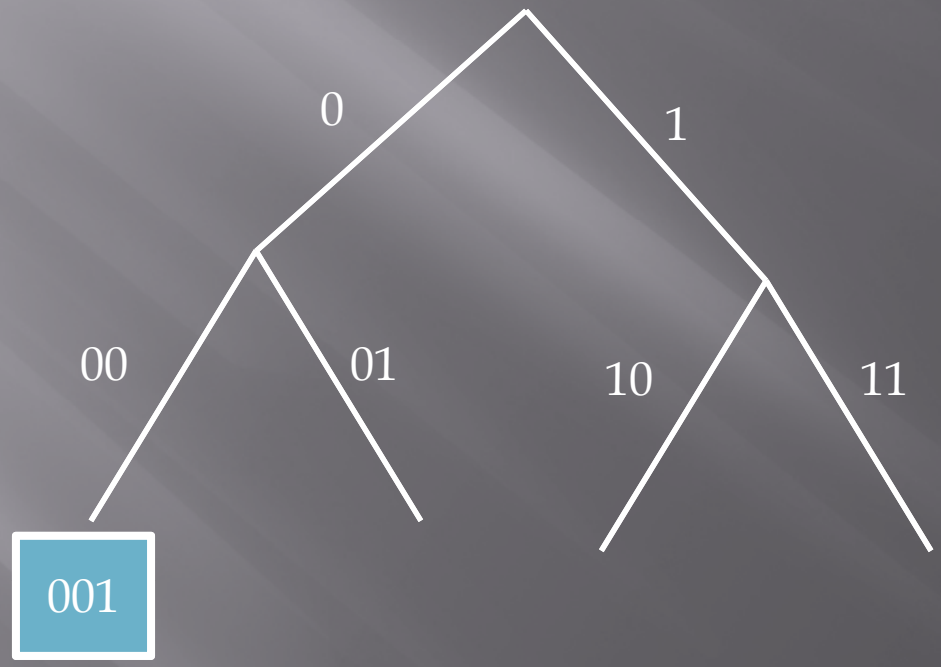
Riferimenti

M. Cai, M. Frank, J. Chen, P. Szekely, *MAAN: A Multi-Attribute Addressable Network for Grid Information Services*, 4th International Workshop on Grid Computing, Phoenix, Arizona, 2003

Prefix Trees

- Un modo alternativo per registrare dati suddividendoli in range è costitutito dai cosiddetti **prefix-tree**.
- In queste strutture, i dati sono memorizzati come foglie all'interno di un albero.
- Ogni ramo dell'albero corrisponde ad una suddivisione del dominio dei valori dei dati
- Ogni passo della suddivisione fa crescere il prefisso dell'etichetta finale dei dati
- Ad esempio, in un **binary prefix tree**, il dominio dei valori è suddiviso ricorsivamente a metà

Prefix Trees



P-Grid 1/8

- P-Grid è una rete P2P basata sul principio dei binary prefix tree
- Lo spazio dei dati (anche multi-dimensionale) è suddiviso in griglie sempre più fini
- I dati sono organizzati all'interno di un binary prefix tree virtuale

P-Grid 2/8

Data assignment

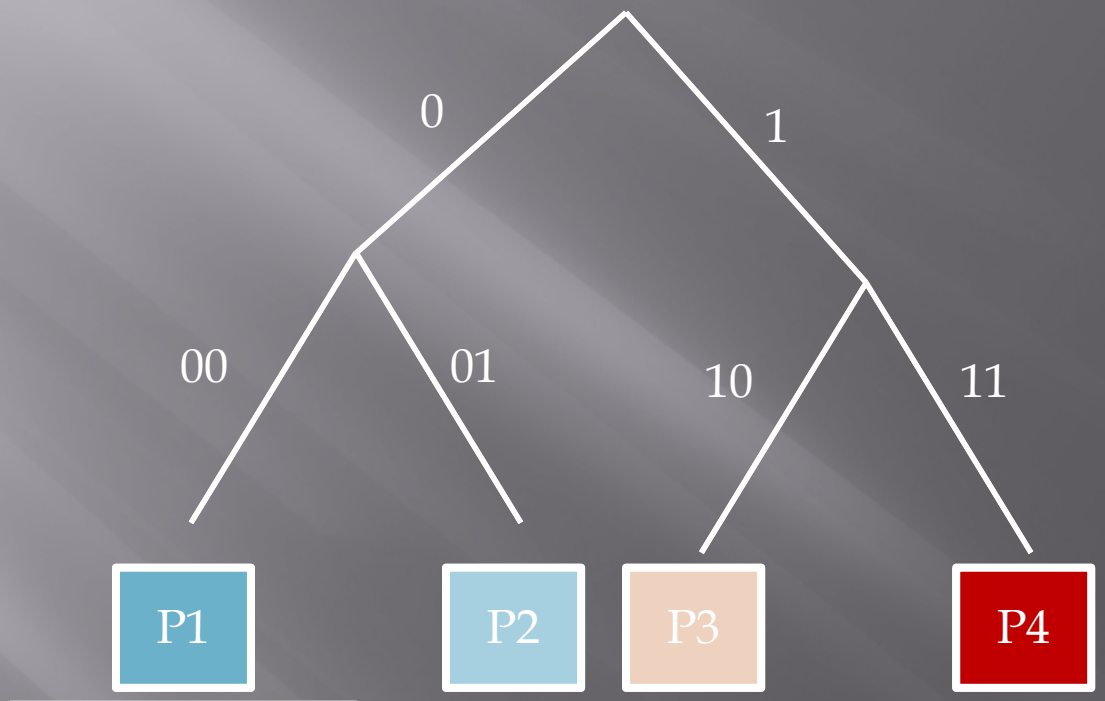
- Ogni peer p è associato ad una chiave binaria presa da uno spazio $K = \{0,1\}^*$
- La chiave è detta $path(p)$ e rappresenta un intero percorso dalla radice alle foglie del prefix tree.
- p gestisce tutti i dati del dominio che hanno $path(p)$ come prefisso.
- Più peer possono gestire la stessa zona di dati (replicazione)

P-Grid 3/8

Network Structure

- Per poter avere una rete connessa e in grado di fare un routing efficiente, è necessario stabilire connessioni tra peer che rispettino la struttura di suddivisione dei dati
- Per ogni bit contenuto nel prefisso *path*(*p*), *p* stabilisce una connessione con un peer che ha un bit completamentare in quella posizione

P-Grid 4/8



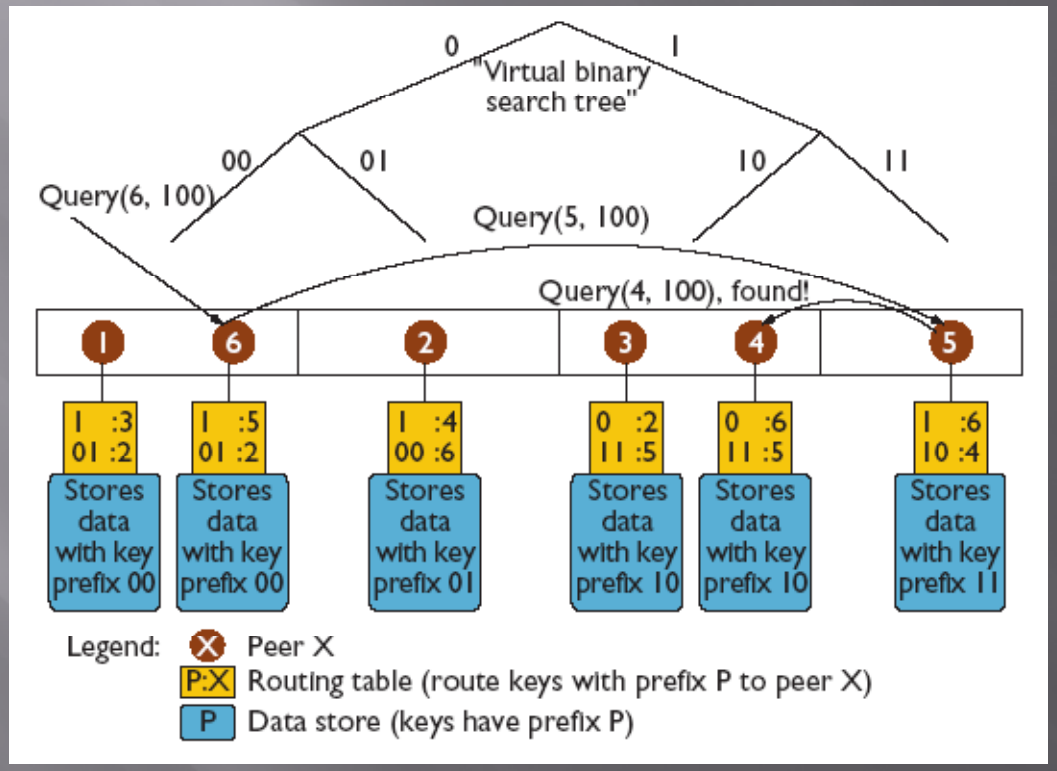
Routing Table
0: P3
01: P4

P-Grid 5/8

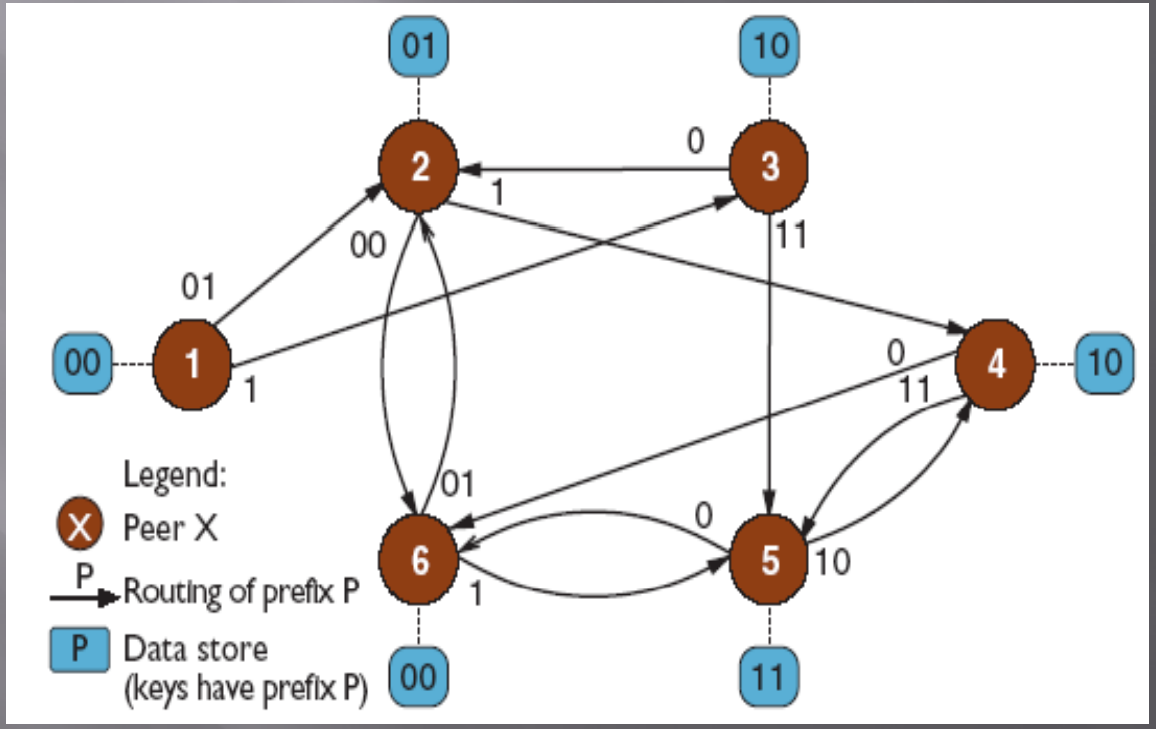
Query routing

- Quando un peer *p* riceve una query, scorre il prefisso della query stessa
- Se un bit non corrisponde al prefisso gestito da *p,* la query viene inviata al vicino di *p* che gestisce il bit richiesto
- Si procede ricorsivamente finché non vengono contattati tutti i nodi che gestiscono le risorse richieste

P-Grid 6/8


0
1
"Virtual binary search tree"
00
01
10
11
Query(6, 100)
Query(5, 100)
Query(4, 100), found!
1
6
2
3
4
5
1 :3
01 :2
1 :5
01 :2
1 :4
00 :6
0 :2
11 :5
0 :6
11 :5
1 :6
10 :4
Stores data with key prefix 00
Stores data with key prefix 00
Stores data with key prefix 01
Stores data with key prefix 10
Stores data with key prefix 10
Stores data with key prefix 11
Legend:
X
Peer X
P:X
Routing table (route keys with prefix P to peer X)
P
Data store (keys have prefix P)

P-Grid 7/8


01
10
2
3
0
1
11
00
01
00
1
1
0
11
4
10
Legend:
X
Peer X
P
Routing of prefix P
P
Data store
(keys have prefix P)
01
0
6
5
10
1
00
11

P-Grid 8/8

- Pros: meccanismo di routing efficiente
- Cons: tutti gli attributi mappati in un unico spazio (v. estensione di CAN)

P-Grid

Riferimenti

- K. Aberer, M. Punceva, M. Hauswirth, R. Schmidt, *Improving Data Access in P2P Systems*, IEEE Internet Computing, January-February 2002
- K. Aberer, P. Cudré-Mauroux, A. Datta, Z. Despotovic, M. Hauswirth, M. Punceva, R. Schmidt, *P-Grid: A Self-organizing Structured P2P System*, ACM SIGMOD Record 32(3), September 2003

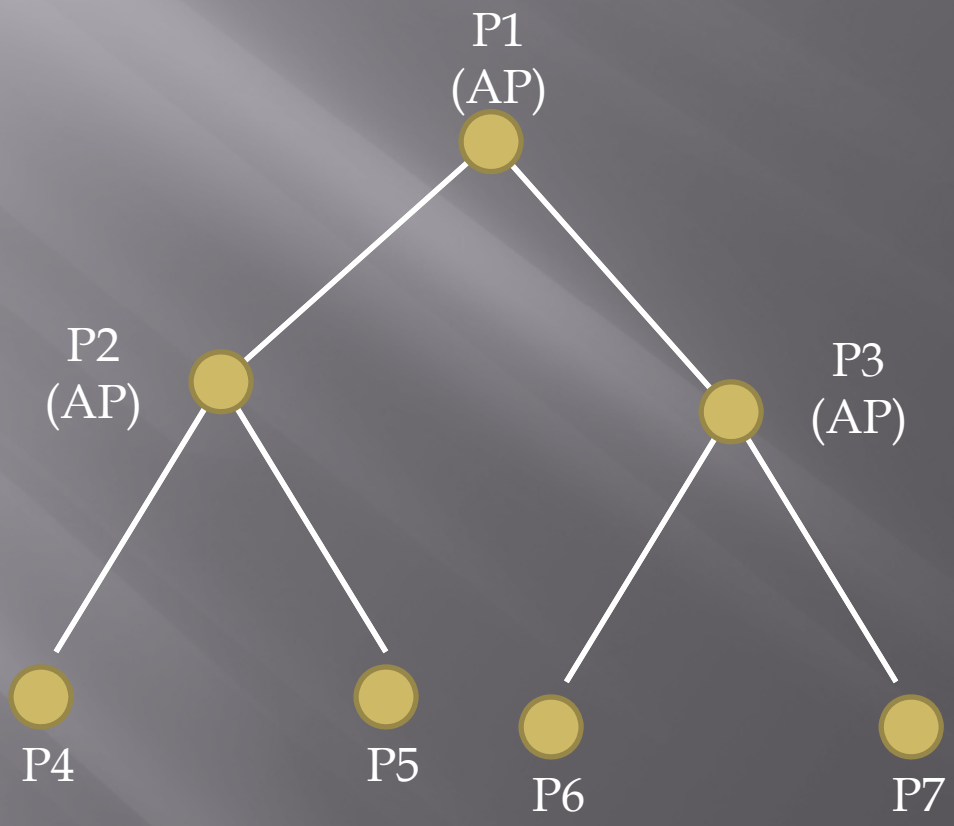
XenoSearch 1/8

- XenoSearch è un sistema che combina un organizzazione DHT con dei prefix tree
- Il sistema estende la DHT di Pastry
- Per consentire ricerche multi-attributo, viene usata una rete Pastry per ogni attributo

XenoSearch 2/8

- I peer (detti XenoServer) si registrano in ogni layer corrispondente agli attributi delle risorse che gestiscono
- Per consentire range queries, i dati di ogni attributo sono organizzati in dei prefix tree
- Ogni peer si fa carico della responsabilità di un nodo del prefix tree
- I peer interni al prefix tree sono detti **Aggregator Point** (AP)

XenoSearch 3/8



XenoSearch 4/8

- Gli AP si registrano nel rispettivo layer Pastry anche con la chiave posseduta nel prefix tree
- È possibile effettuare nello stesso overlay sia ricerche esatte, sia per range, contattando gli AP

XenoSearch 5/8

Query resolution

- Ogni query multi-attributo viene decomposta in *m* sottoquery, una per attributo
- Per ogni sottoquery si interroga il layer relativo
- Se la sottoquery riguarda un range, verranno contattati degli AP, che forniranno la lista di XenoServer che rientrano nel range

XenoSearch 6/8

Query resolution

- Ogni layer fornisce al nodo da cui è partita la query una lista di XenoServer
- Il nodo iniziale interseca queste liste per trovare gli XenoServer che soddisfano tutte le condizioni
- Contatta quindi tutti questi XenoServer per recuperare le risorse
- È necessario perché le informazioni registrate nellE DHT non sono sempre del tutto aggiornate

XenoSearch 7/8

Query resolution

- Ogni layer fornisce al nodo da cui è partita la query una lista di XenoServer
- Il nodo iniziale interseca queste liste per trovare gli XenoServer che soddisfano tutte le condizioni
- Contatta quindi tutti questi XenoServer per recuperare le risorse
- È necessario perché le informazioni registrate nelle DHT non sono sempre del tutto aggiornate

XenoSearch 8/8

- Il nodo iniziale interseca queste liste per trovare gli XenoServer che soddisfano tutte le condizioni
- Contatta quindi tutti questi XenoServer per recuperare le risorse
- È necessario perché le informazioni registrate nelle DHT non sono sempre del tutto aggiornate

XenoSearch

Riferimenti

D. Spence, T. Harris, *XenoSearch: Distributed resource discovery in the XenoServer open platform*, Twelfth IEEE Int. Symposium on High Performance Distributed Computing, HPDC-12, 2003

Riassunto

Sistema	Protocollo Base	Multi-attributo	Range Resolution
Estensione CAN	CAN	Mappaggio in CAN con SFC	Flooding tra i nodi nell'ipercubo che contiene la query
MAAN	Chord	Un overlay Chord per attributo	Sequenziale
P-Grid	P-Grid	Mappaggio in un unico spazio di indirizzamento	Ricerche in un Binary Prefix Tree
XenoSearch	Pastry	Un overlay Pastry per attributo	Ricerche in Prefix Tree