

Lezione n.7

Distributed Hash Tables

Laura Ricci

Materiale didattico:
Peer-to-Peer Systems
and Applications
Capitolo 7

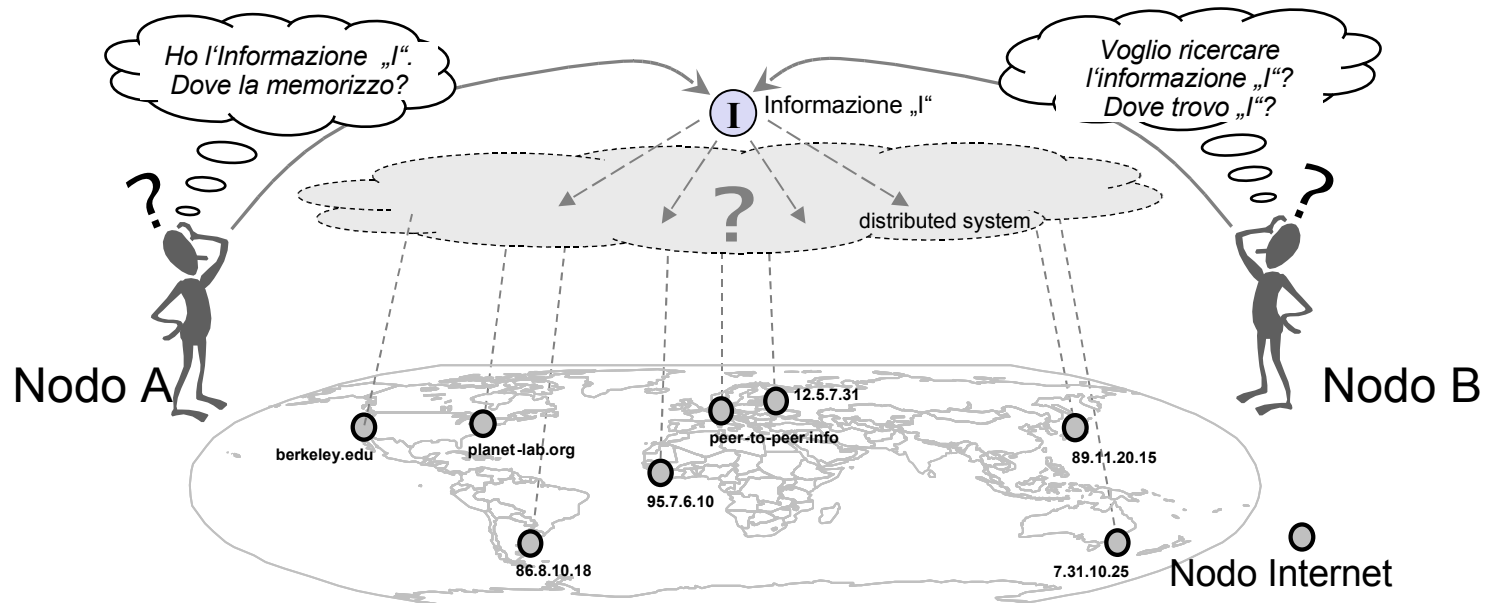
Riassunto della Presentazione

1. Distributed Hash Tables (DHT): Introduzione
 1. Motivazioni
 2. Caratteristiche
 3. Confronti
2. DHT: Aspetti Fondamentali
 1. Gestione distribuita dei dati
 2. Indirizzamento nelle Distributed Hash Tables
 3. Routing
 4. Memorizzazione dei dati
3. DHT: I meccanismi
 1. Inserzione di nuovi nodi
 2. Fallimento/Uscita volontaria di nodi dal sistema
4. DHT: Le interfacce
5. Conclusioni

Riassunto della Presentazione

1. Distributed Hash Tables (DHT): Introduzione
 1. Motivazioni
 2. Caratteristiche
 3. Confronti
2. DHT: Aspetti Fondamentali
 1. Gestione distribuita dei dati
 2. Indirizzamento nelle Distributed Hash Tables
 3. Routing
 4. Memorizzazione dei dati
3. DHT: I meccanismi
 1. Inserzione di nuovi nodi
 2. Fallimento/Uscita volontaria di nodi dal sistema
4. DHT: Le interfacce
5. Conclusioni

Sistemi P2P: Look Up Problem



- A vuole memorizzare una informazione I all'interno del sistema distribuito, B vuole reperire I senza conoscere a priori l'effettiva locazione di I
- Come organizzare il sistema distribuito? In particolar modo, quali sono i meccanismi utilizzati per decidere **dove memorizzare** l'informazione e **come reperirla**?
- Qualsiasi soluzione deve tenere particolarmente in considerazione

Scalabilità del Sistema. Occorre controllare l'overhead di comunicazione e la memoria utilizzata da ogni nodo, in funzione del numero dei nodi del sistema

Robustezza ed adattabilità in caso di faults e frequenti cambiamenti

P2P: Analisi e Confronti

- *Approccio Centralizzato*

Ricerca: $O(1)$ - "basta chiedere al server"

Quantità di Memoria Richiesta sul Server: $O(N)$ (N = numero di informazioni disponibili nel sistema)

Banda richiesta (connessione server/rete): $O(N)$

Possibilità di sottomettere al sistema queries complesse

- *Approccio Completamente Distribuito*

Ricerca: caso pessimo $O(N^2)$ - "ogni nodo chiede a tutti i vicini". Possibili ottimizzazioni (TTL, identificatori per evitare cammini ciclici)

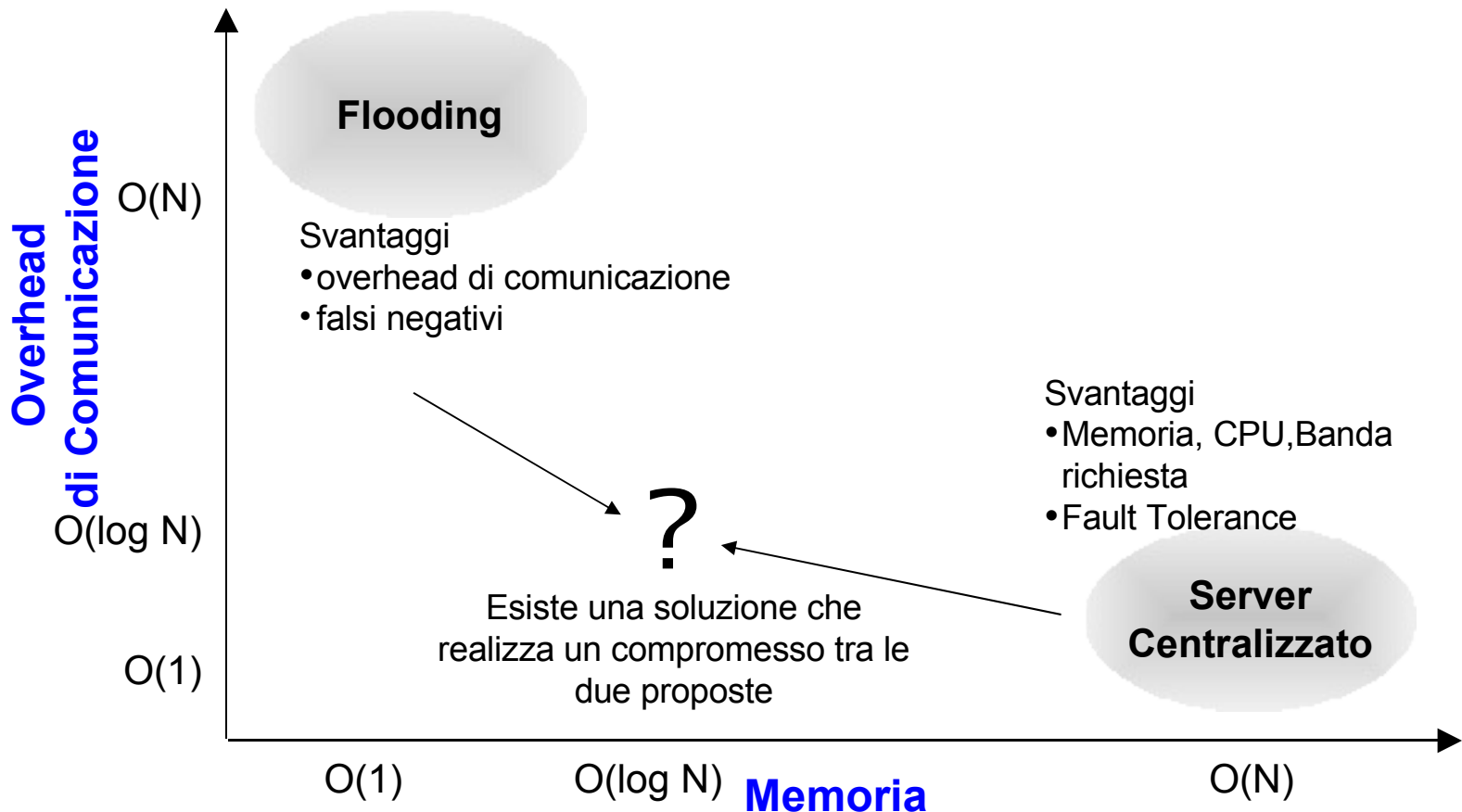
Quantità di memoria richiesta : $O(1)$

Informazione condivisa: non dipende dal numero di nodi del sistema

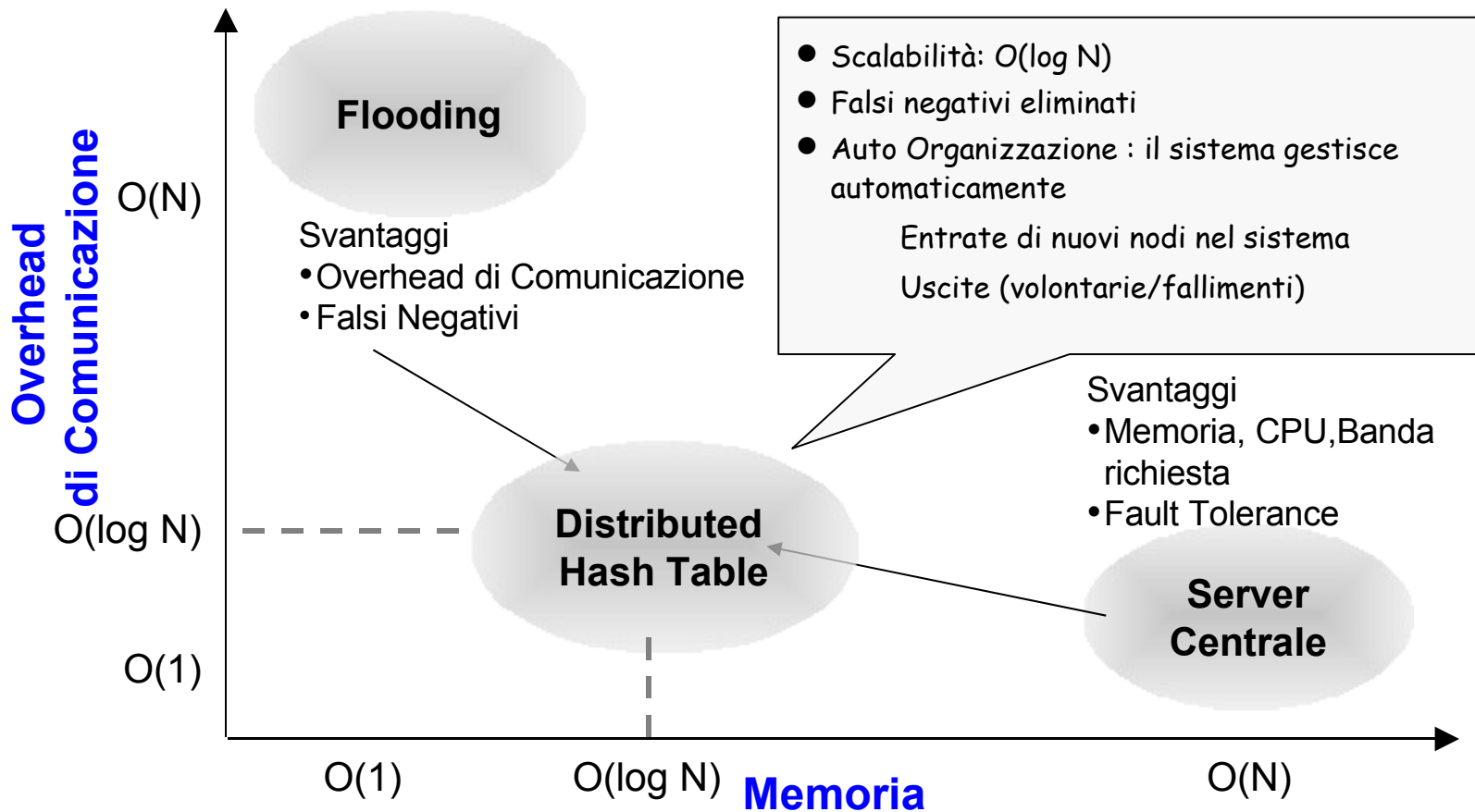
Non si utilizzano strutture dati per ottimizzare il routing della query (flooding)

Distributed Hash Tables: Motivazioni

Analisi dei sistemi Esistenti



Distributed Hash Tables: Motivazioni

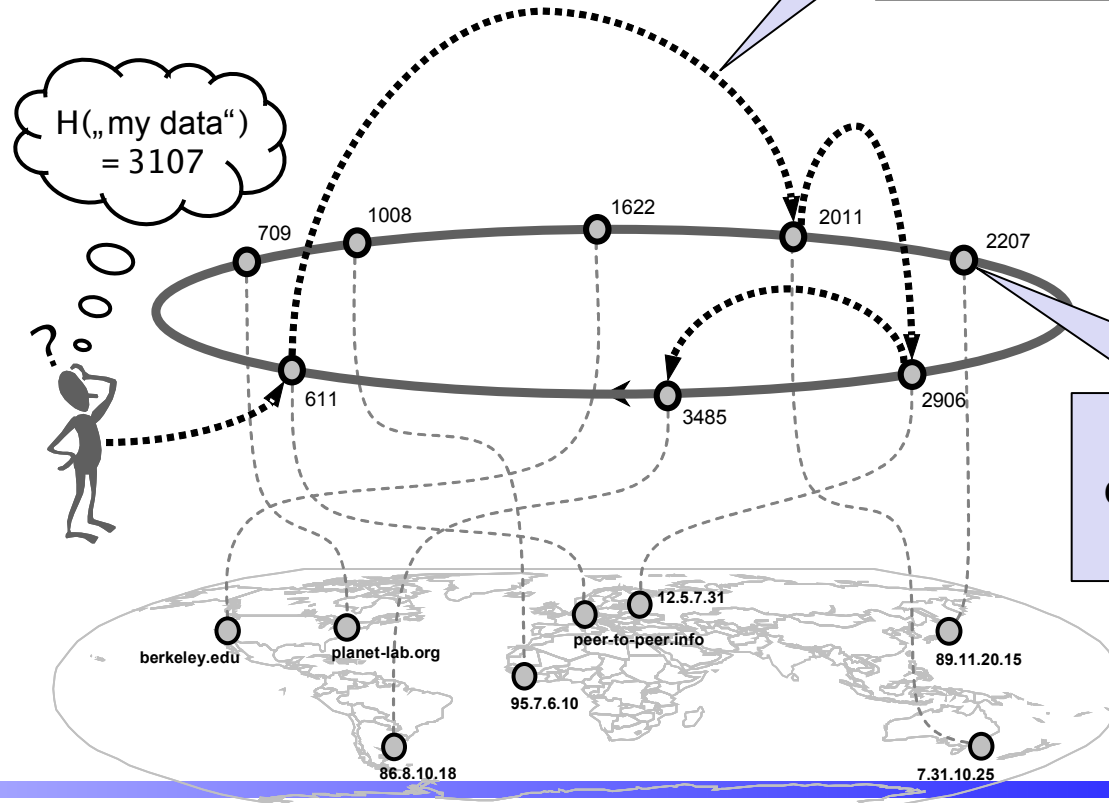


Distributed Hash Tables: Concetti Generali

- Obiettivi

$O(\log(N))$ hops per la ricerca di un'informazione
 $O(\log(N))$ entrate nella tabella di routing

Il routing richiede $O(\log(N))$ passi per raggiungere il nodo che memorizza l'informazione



$O(\log(N))$ dimensione della tabella di routing di ogni nodo

Distributed Hash Tables: Obiettivi

- DHT: Obiettivi
 - Scalabilità
 - Flessibilità
 - Affidabilità
- Adattabilità a fallimenti, inserimento ed eliminazione di nodi
 - Assegnamento di informazioni ai nuovi nodi
 - Re-assegnamento e re-distribuzione delle informazioni in caso di fallimento o disconnessione volontaria dei nodi dalla rete
- Bilanciamento delle informazioni tra i nodi
 - Fondamentale per l'efficienza della ricerca

Confronti tra i diversi Approcci

Approccio	Memoria per Nodo	Overhead di Comunicazione	Queries Complesse	Falsi Negativi	Robustezza
Server Centrale	$O(N)$	$O(1)$	✓	✓	✗
P2P puro (flooding)	$O(1)$	$O(N^2)$	✓	✗	✓
DHT	$O(\log N)$	$O(\log N)$	✗	✓	✓

Riassunto della Presentazione

1. Distributed Hash Table: Introduzione
 1. Motivazioni
 2. Caratteristiche
 3. Confronti
2. Distributed Hash Tables: aspetti fondamentali
 1. Gestione distribuita dei dati
 2. Indirizzamento
 3. Routing
 4. Memorizzazione dei dati
3. DHT: I meccanismi
 1. Inserzione di nuovi nodi
 2. Fallimento/Uscita Volontaria di Nodi dal Sstema
4. DHT: Le Interfacce
5. Conclusioni

DHT: Gestione Distribuita dei Dati

Operazioni fondamentali

3. Mapping dei nodi e dei dati nello stesso spazio di indirizzamento

Ai peers ed ai dati sono associati degli identificatori unici (ID), che li individuano univocamente all'interno del sistema

Esiste uno spazio comune degli indirizzi per i dati e per i peer.

I nodi sono responsabili della gestione di una porzione dello spazio degli indirizzi corrispondente ad un sottoinsieme dei dati

La corrispondenza tra i dati ed i nodi può variare per l'inserimento/cancellazione di nodi

4. Memorizzazione/ Ricerca dei dati

Ricerca di un dato = routing verso il nodo responsabile

Ogni nodo mantiene una tabella di routing, che fornisce al nodo una visibilità parziale del sistema

Il routing è guidato dalla conoscenza dell'ID del dato ricercato

Falsi negativi eliminati

Distributed Hash Tables: Indirizzamento

- Tecniche di indicizzazione distribuita

Spazio degli indirizzi in cui vengono mappati sia i dati che i nodi

I nodi intermedi mantengono delle tabelle di routing

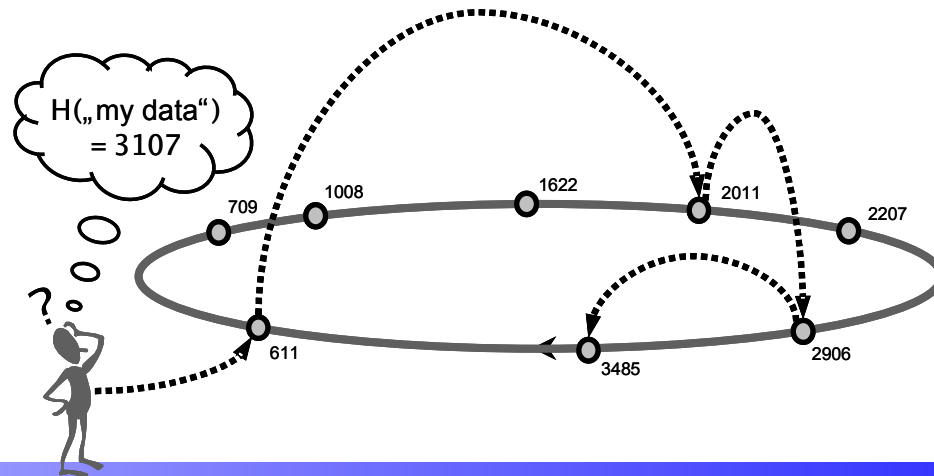
Instradamento efficiente verso il nodo "destinazione"

Content routing (guidato dagli ID dei dati),

- Problemi

Gestione dinamica delle tabelle di Routing (inserzioni, eliminazioni)

Queries complesse non supportate (e.g, ricerche contenenti wildcard)

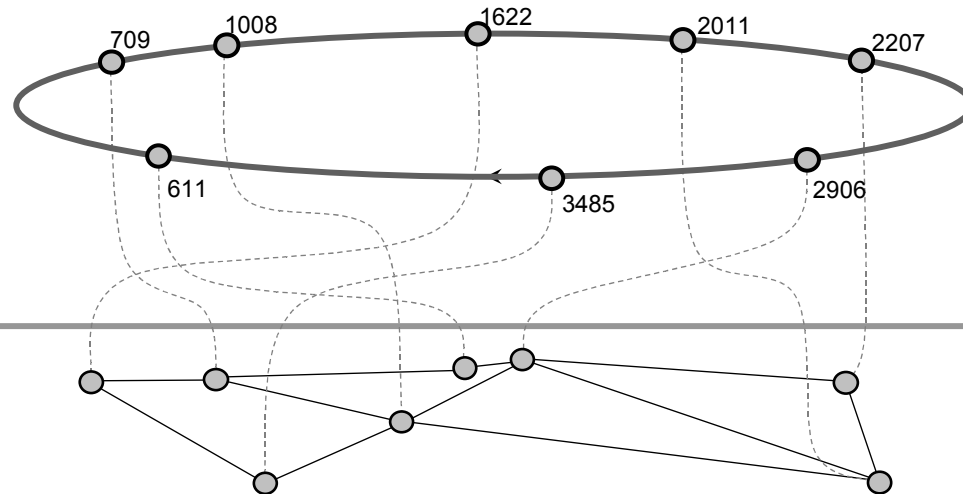


Distributed Hash Tables: Indirizzamento

Passo 1:

- Definizione dello **spazio degli indirizzi**. Esempio: Spazio degli indirizzi strutturato secondo un **anello logico**. S
spazio lineare degli indirizzi $0, \dots, 2^m - 1 \gg$ del numero di oggetti da memorizzare (es $m=160$), su cui è definito un ordinamento totale (operazioni in modulo)
- Associazione nodi-indirizzi, mediante **una funzione hash**
- La topologia reale e logica (overlay network) non sono in genere correlate

**Distributed Hash Table:
Visione logica**



**Mapping sulla topologia
reale**

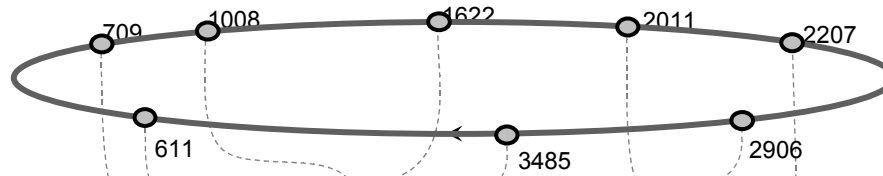
Distributed Hash Tables: Indirizzamento

Passo 2:

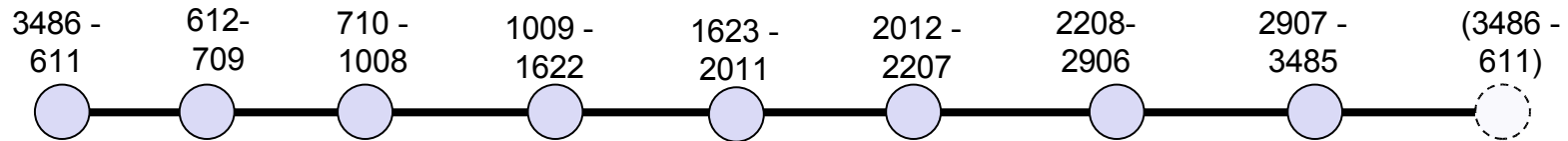
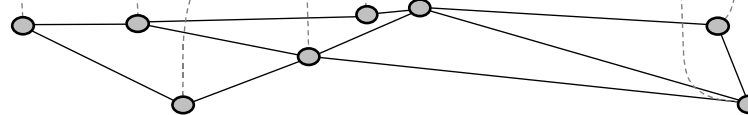
- Ogni nodo è **responsabile di una parte dei dati** memorizzati nella DHT
- I dati vengono mappati nello stesso spazio degli indirizzi dei nodi, mediante la **funzione hash**
E.g., $\text{Hash}(\text{String}): H(„ LucidiLezione12-03-06 “) \rightarrow 2313$
Esempi: hashing del nome del file o del suo intero contenuto
- Ad ogni nodo viene assegnata una **porzione contigua** dello spazio degli indirizzi.
- Ogni nodo memorizza informazioni relative ai dati mappati sulla propria porzione di indirizzi
- Spesso si introduce una certa ridondanza (overlapping)

Distributed Hash Table: Indirizzamento

Distributed Hash Table:
Visione logica

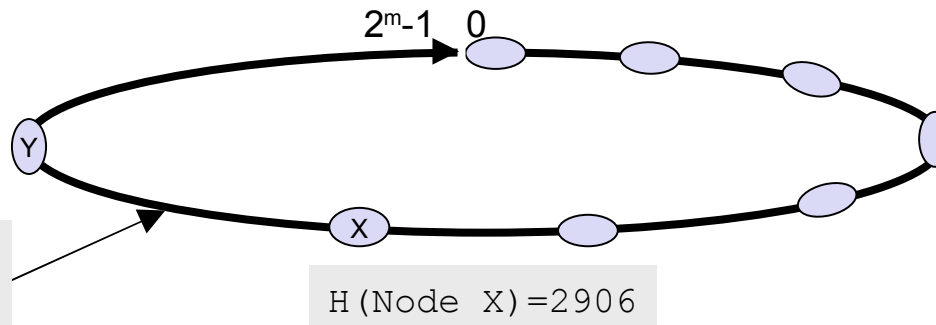


Mapping sulla topologia
reale



$H(\text{Node Y}) = 3485$

Dato "D":
 $H("D") = 3107$



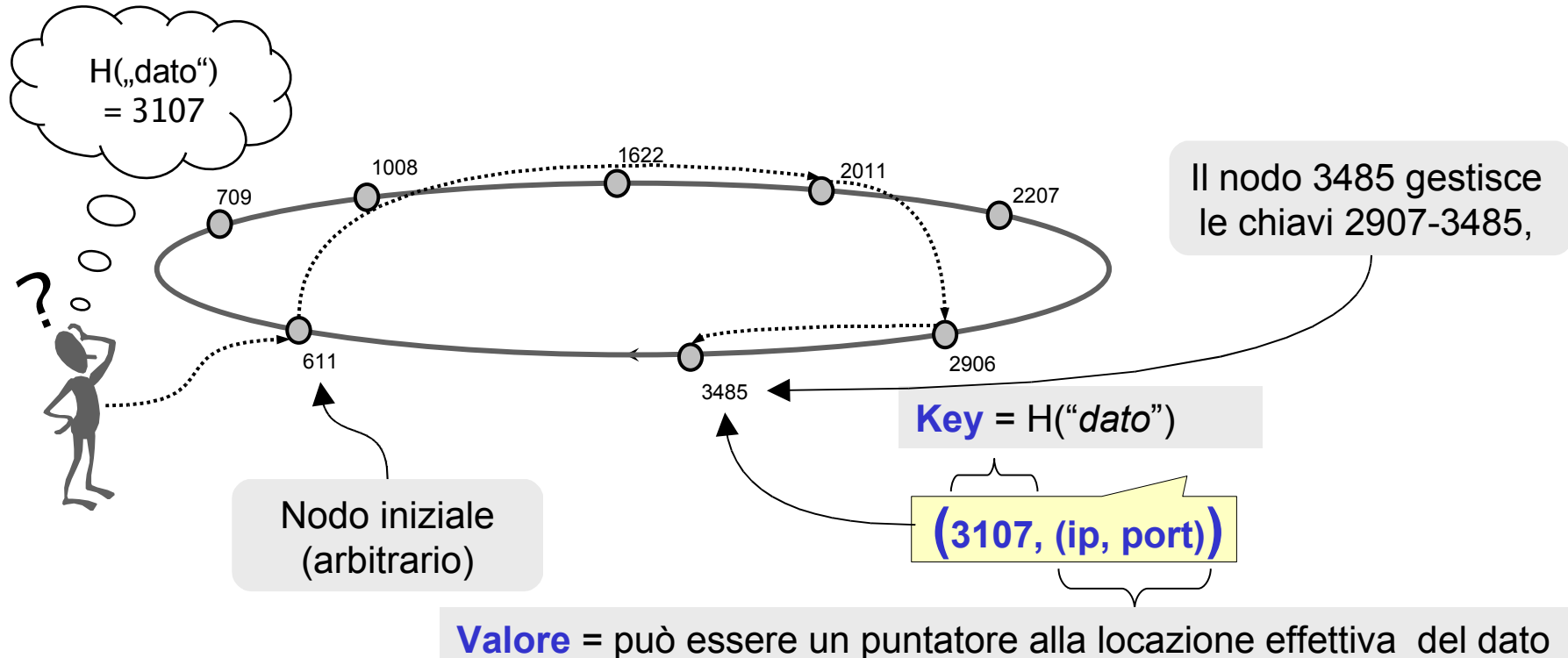
$H(\text{Node X}) = 2906$

Distributed Hash Tables: Bilanciamento del Carico

- Problema: distribuzione uniforme degli indirizzi tra i peers che definiscono la DHT
- Cause di possibili sbilanciamenti del carico:
 - Un nodo deve gestire una grossa porzione dello spazio degli indirizzi
 - Gli spazi degli indirizzi sono distribuiti in modo uniforme tra i nodi, ma gli indirizzi gestiti da un nodo corrispondono a molti dati
 - Un nodo deve gestire diverse queries, perché i dati corrispondenti agli indirizzi gestiti sono molto richiesti
- Sbilanciamento del carico comporta minor robustezza del sistema, minor scalabilità. Complessità $O(\log N)$ non garantita
- Definiti algoritmi di bilanciamento del carico

Distributed Hash Tables: Routing

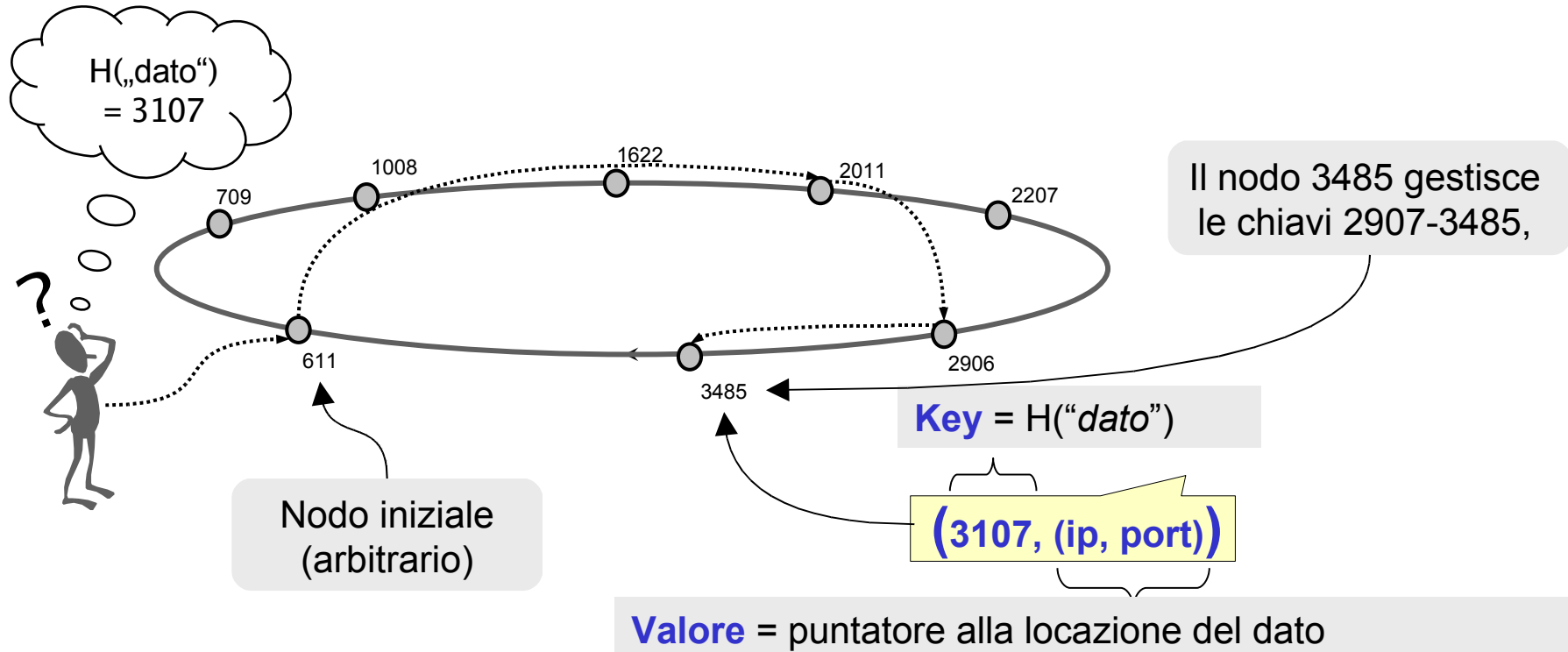
- Problema: dato D , cercare il nodo che gestisce $key=H(D)$
- La ricerca inizia in un nodo arbitrario della DHT
- La ricerca è guidata da $H(D)$



Distributed Hash Tables: Routing

- Ogni nodo ha in genere una visione limitata della DHT
- **Next hop:** dipende dall'algoritmo di routing.

Esempio: basato sulla "vicinanza" tra l'ID del dato e l'ID del nodo (routing content based). Sistemi non strutturati: routing basato solo su connessioni tra nodi vicini



DHT: Memorizzazione Diretta

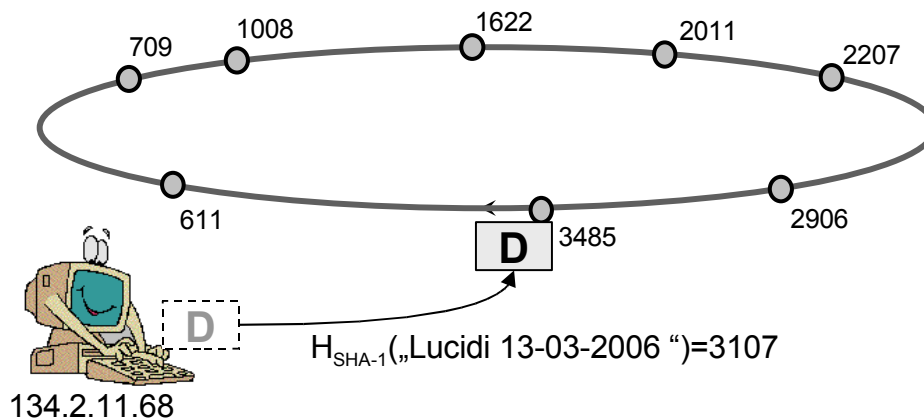
- Memorizzazione Diretta

La DHT memorizza coppie del tipo (key, valore)

Value = valore del dato ricercato

Il dato viene copiato, al momento del suo inserimento nella DHT, nel nodo che ne è responsabile (secondo il mapping dati-nodi). **NOTA BENE:** tale nodo non è in generale il nodo che ha inserito il dato nella DHT.

Esempio: $key = H(\text{"Lucidi 13-03-2006"}) = 3107$. Il dato viene memorizzato sul nodo responsabile dell'indirizzo 3107.



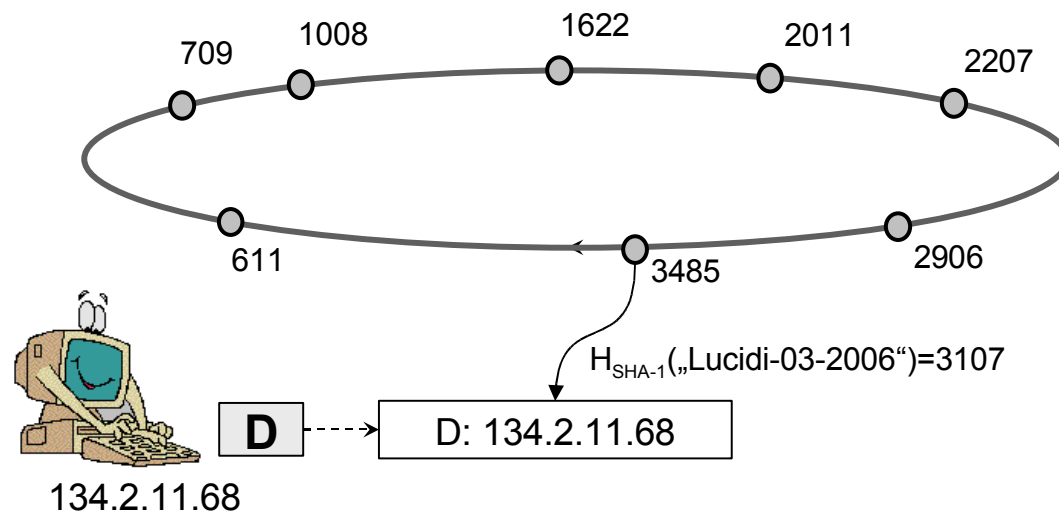
DHT: Memorizzazione Indiretta

- Memorizzazione Indiretta

Valore = riferimento al dato ricercato (es: indirizzo fisico del nodo che memorizza il contenuto)

Il nodo che memorizza il dato può essere quello che lo ha inserito nel sistema

Più flessibile, richiede un passo in più per l'accesso al dato

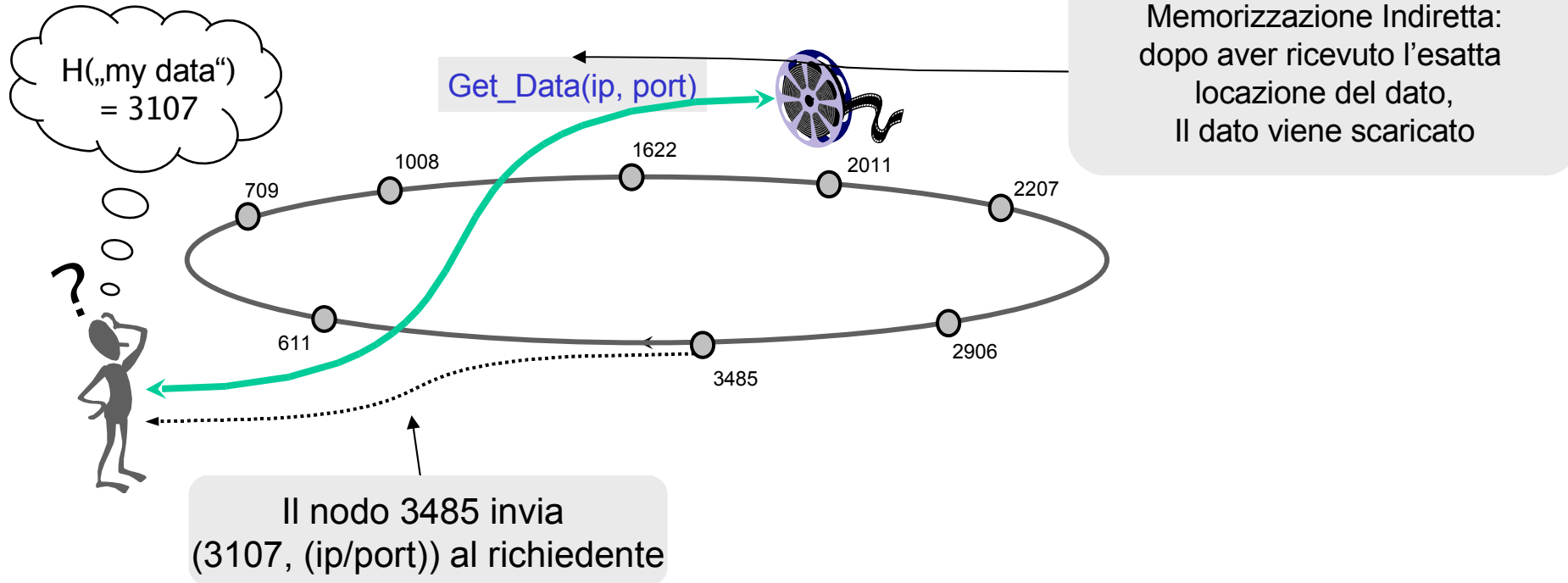


Distributed Hash Tables: Memorizzazione Indiretta

- Trasferimento dei dati

Si spediscono indirizzo IP e porta al richiedente

Il richiedente effettua il download dei dati

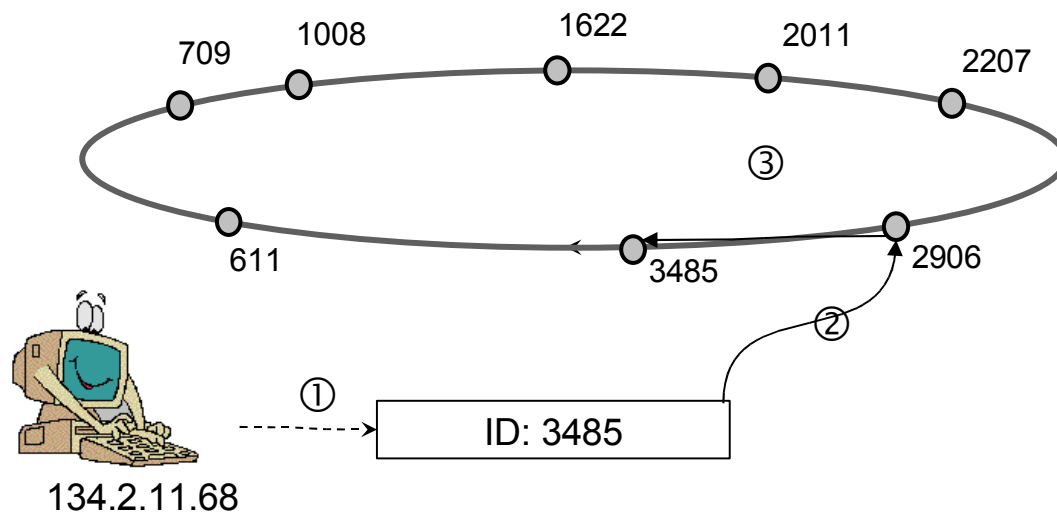


Riassunto della Presentazione

1. Distributed Hash Tables: Introduzione
 1. Motivazioni
 2. Caratteristiche
 3. Confronti
2. Distributed Hash Tables: Aspetti fondamentali
 1. Gestione Distribuita dei Dati
 2. Distributed Hash Tables: indirizzamento
 3. Routing
 4. Memorizzazione dei dati
3. DHT: I meccanismi
 1. Inserzione di nuovi nodi
 2. Fallimento/Uscita Volontaria di Nodi dal Sistema
4. DHT: Le interfacce
5. Conclusioni

Inserzione di Nuovi Nodi

- Inserzione di un nuovo nodo:
 1. Calcolo dell' identificatore del nodo, ID
 2. Il nuovo nodo contatta un nodo arbitrario della DHT (bootstrap)
 3. Assegnamento di una porzione dello spazio degli indirizzi ad ID
 4. Copia delle coppie K/V assegnate (in genere si utilizza ridondanza)
 5. Inserzione nella DHT (collegamento con nodi vicini)



Fallimenti/Ritiro Volontario dei Nodi

- Ritiro Volontario di un nodo

Partizionamento della propria porzione degli indirizzi sui nodi vicini

Copia delle coppie chiave/valore sui nodi corrispondenti

Eliminazione del nodo dalle tabelle di routing

- Fallimento di un Nodo

Se un nodo si disconnette in modo inatteso, tutti i dati memorizzati vengono persi a meno che non siano memorizzati su altri nodi

Memorizzazione di **informazioni ridondanti (replicazione)**

Perdita delle informazioni; refreshing periodico delle informazioni

Utilizzo di percorsi di routing alternativi/ridondanti

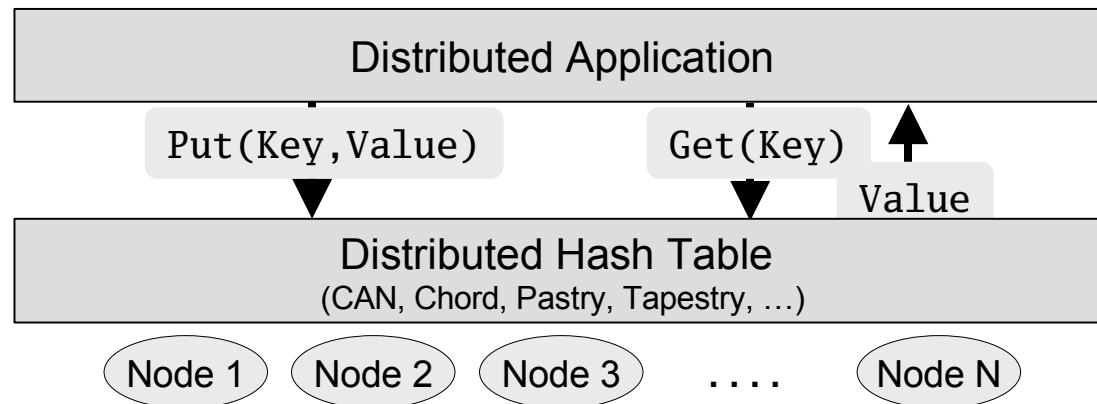
Probing periodico dei nodi vicini per verificarne la operatività. In caso di fault, aggiornamento delle routing tables

Riassunto della Presentazione

1. Distributed Hash Tables: Introduzione
 1. Motivazioni
 2. Caratteristiche
 3. Confronti
2. Distributed Hash Tables: aspetti fondamentali
 1. Gestione Distribuita dei Dati
 2. Distributed Hash Tables: indirizzamento
 3. Routing
 4. Memorizzazione dei dati
3. DHT: I meccanismi
 1. Inserzione di nuovi nodi
 2. Fallimento/Uscita volontaria di nodi dal sistema
4. DHT: Applicazioni
5. Conclusioni

DHT: Applicazioni

- Interfaccia per l'accesso alla DHT
 - Inserimento di Informazione Condivisa
`Put(key,value)`
 - Richiesta di Informazione (content search)
`Get(key)`
 - Risposte
`Value`
- L'interfaccia è comune a molti sistemi basati su DHT



DHT: Applicazioni

- Le DHT offrono un servizio generico distribuito per la memorizzazione e l'indicizzazione di informazioni
- Il valore memorizzato in corrispondenza di una chiave può essere
 - Un file
 - Un indirizzo IP
 - O qualsiasi altro dato.....
- Esempi di applicazioni che possono utilizzare le DHT
 - Realizzazione di DNS
 - Chiave: hostname, valore: lista di indirizzi IP corrispondenti
 - P2P storage systems: es. Freenet
 -

Riassunto della Presentazione

1. Distributed Hash Tables: Introduzione
 1. Motivazioni
 2. Caratteristiche
 3. Confronti
2. Distributed Hash Tables: aspetti fondamentali
 1. Gestione Distribuita dei Dati
 2. Distributed Hash Tables: indirizzamento
 3. Routing
 4. Memorizzazione dei dati
3. DHT: I meccanismi
 1. Inserzione di nuovi nodi
 2. Fallimento/Uscita volontaria di nodi dal sistema
4. DHT: Le interfacce
5. Conclusioni

Conclusioni

- Proprietà delle DHT

Il routing è basato **sul contenuto** della query

Le chiavi sono equamente distribuite tra i nodi della DHT

Si evitano i colli di bottiglia

Supportano l'inserzione incrementale di chiavi nel sistema

Tolleranti ai guasti

Sistemi auto-organizzanti

Realizzazione semplice ed efficiente

Supportano un ampio spettro di applicazioni

I valori associati alle chiavi dipendono dalla applicazione

DHT: Sistemi Esistenti

- Distributed Hash Tables: esempi
 - Chord
UC Berkeley, MIT
 - Pastry
Microsoft Research, Rice University
 - Tapestry
UC Berkeley
 - CAN
UC Berkeley, ICSI
 - P-Grid
EPFL Lausanne
- ... ed altri: Kademia, Symphony, Viceroy, ...