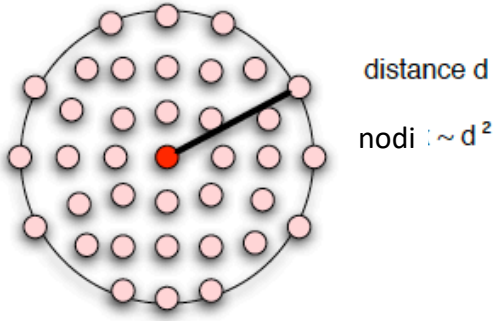
# Il Modello Small World e le Reti Sociali Reali

- **Esperimento di Milgram** (1967): concetto di ”small world” (6 gradi di separazione)

- **Modelli teorici**:
  - Watts-Strogatz: griglie con ”*cammini brevi*” con collegamenti locali e a lunga distanza

  - Kleinberg: griglie *navigabili* con collegamenti locali e a lunga distanza




**Problema principale**: modellare densità di popolazione non uniforme

# Problema della densità non uniforme

## Limite dei modelli a griglia

La distribuzione $1/d^2$ suppone densità uniforme, irrealistica per dati reali

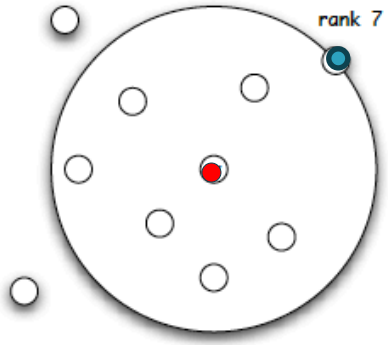
**Soluzione proposta**:

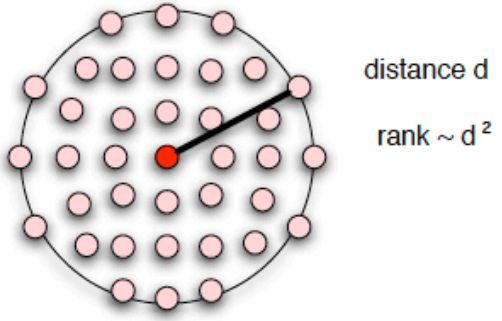
- Sostituire la distanza tra $u$ e $v$ con il *rank* di $v$ rispetto a $u$

$$rank_u(v) := |\{w \mid d(u,w) < d(u,v)\}|$$

il numero di nodi che sono più vicini a $u$ di $v$

Esempio In una griglia 2D: se $d(u, v) \sim d$ allora $rank_u(v) \sim d^2$



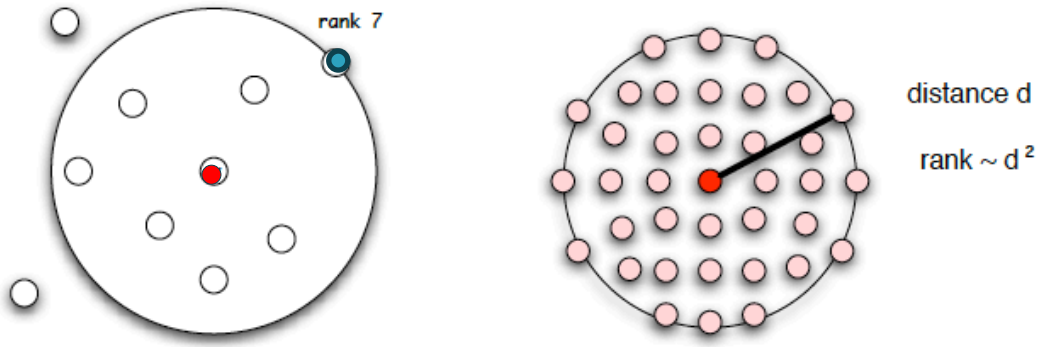



# Problema della densità non uniforme

Possiamo ridefinire la distribuzione di probabilità in base al rango

$$P[u \rightarrow v] \approx 1/\, rank_u(v)$$

*quindi solo in funzione della densità della popolazione a distanza al più d(u,v)*



# Problema della densità non uniforme

## (Liben-Nowell et al., 2005)

Il risultato di Kleinberg per la griglia si generalizza.

Liben-Nowell et al. Mostrano che per quasi ogni densità di popolazione (cioè, indipendentemente da dove si trovino le persone), se i collegamenti sono costruiti casualmente in modo che la probabilità di un'amicizia sia proporzionale a $rank^{-1,}$

allora la rete risultante ammette la ricerca decentralizzata efficiente.

# Problema della densità non uniforme

## (Liben-Nowell et al., 2005)

Liben-Nowell et al. Mostrano che per quasi ogni densità di popolazione (cioè, indipendentemente da dove si trovino le persone), se i collegamenti sono costruiti casualmente in modo che la probabilità di un'amicizia sia proporzionale a $rank^{-1,}$

allora la rete risultante ammette la ricerca decentralizzata efficiente.

**Algoritmo GeoGreedy**: dato un target t, sia u il detentore del messaggio,

u esamina il suo insieme di amici e inoltra il messaggio al vicino v che è geograficamente più vicino al target t.

**Teorema**. Sia N una griglia k-dimensionale e sia P un'arbitraria popolazione di n persone su N con link basati sul rango.

Per una sorgente s arbitraria e un target t scelto uniformemente a caso,

la lunghezza attesa del percorso da s a t trovato da GEOGREEDY è al massimo $c\log^3 n$, per una costante c indipendente da n, s e P, ma dipendente da k.

# Verifica empirica del modello
## (Liben-Nowell et al., 2005)

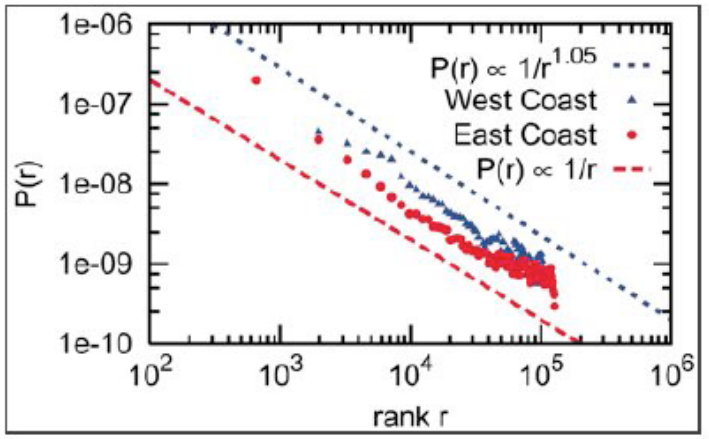
- Studio su LiveJournal:
  - 500,000 utenti e circa 4 milioni di link,
  - link *(u,v)* è presente se *u* appare nella lista degli amici di *v*
  - Usati zip code (dai profili utenti) per la localizzazione
    - d(u,v) = distanza geografica tra u e v
    - $rank_u(v) := |\{w \mid d(u,w) < d(u,v)\}|$

# Verifica empirica del modello
## (Liben-Nowell et al., 2005)

- Se $rank_u(v)=r$ allora valutata la probabilità $P(r)$ che ci sia un link $(u,v)$ in funzione di $r$

Rappresentazione di *log P(r)* in funzione di log r, con r pari al rango

- East coast in rosso, west coast in blu
- Lower bound esponente= -1.05
- Upper bound esponente= -1




(b) *Rank-based friendship: East and West coasts*

Risultato chiave: **Conferma la relazione $P(r) \propto 1/r$**

# Facebook (Backstrom et al., 2010)

- Dataset: 3.5 milioni di utenti geolocalizzati (da indirizzo fornito nel profilo)
- 30.6 milioni di connessioni analizzate

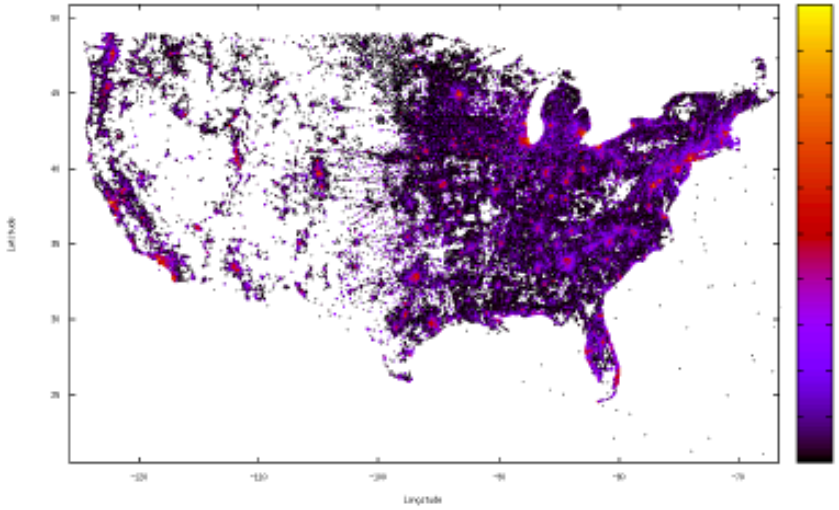


**Figure 1: US population density of geolocated Facebook users.**

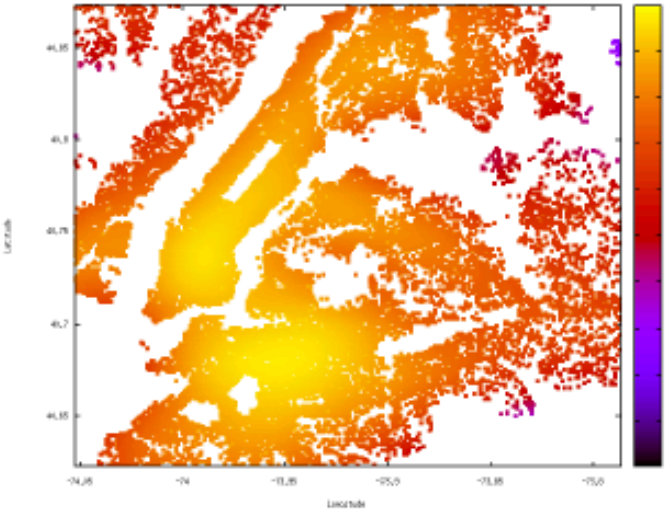


**Figure 2: NY population density of geolocated Facebook users.**

# Probabilità di amicizia vs. distanza

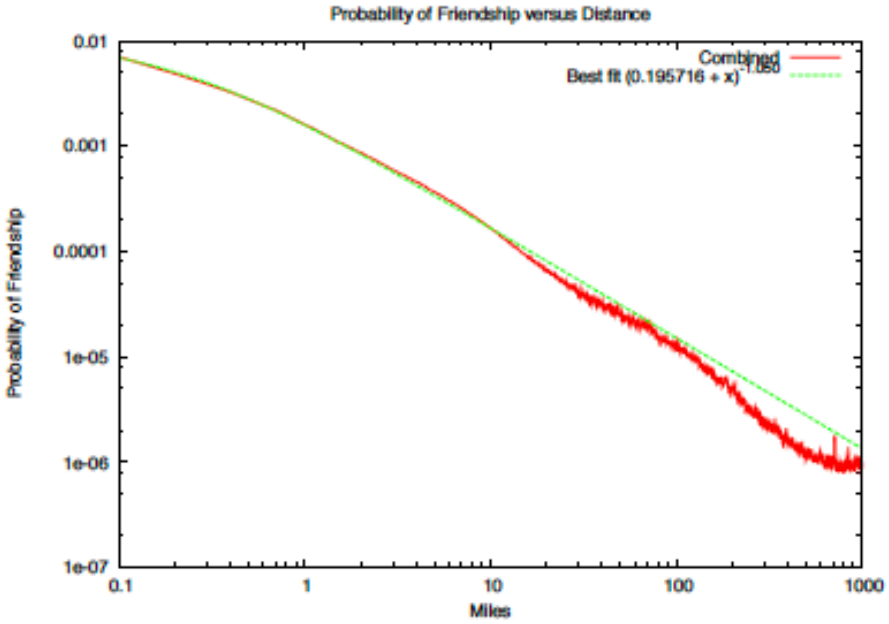



**Figure 7: Probability of friendship as a function of distance. By computing the number of pairs of individuals at varying distances, along with the number of friends at those distances, we are able to compute the probability of two people at distance $d$ knowing each other. We see here that it is a reasonably good fit to a power-law with exponent near $-1$.**

# Probabilità di amicizia vs. distanza in relazione alla densità della popolazione

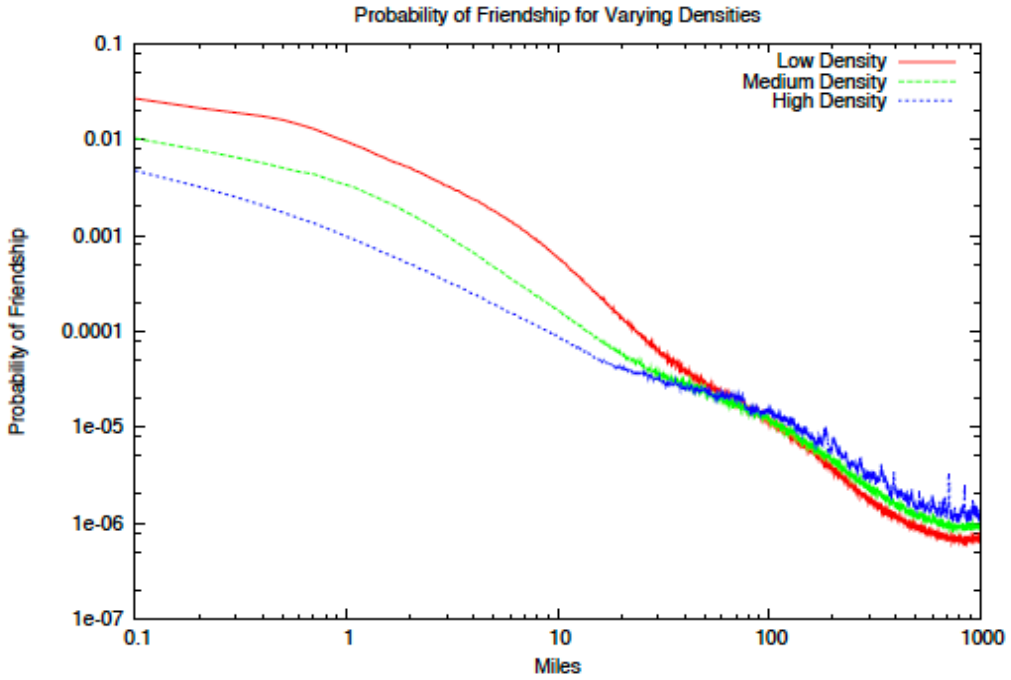



**Figure 8: Looking at the people living in low, medium and high density regions separately, we see that if you live in a high density region (a city), you are less likely to know a nearby individual, since there are so many of them. However, you are more likely to have contact with someone far away.**

# Numero di amici vs. rank

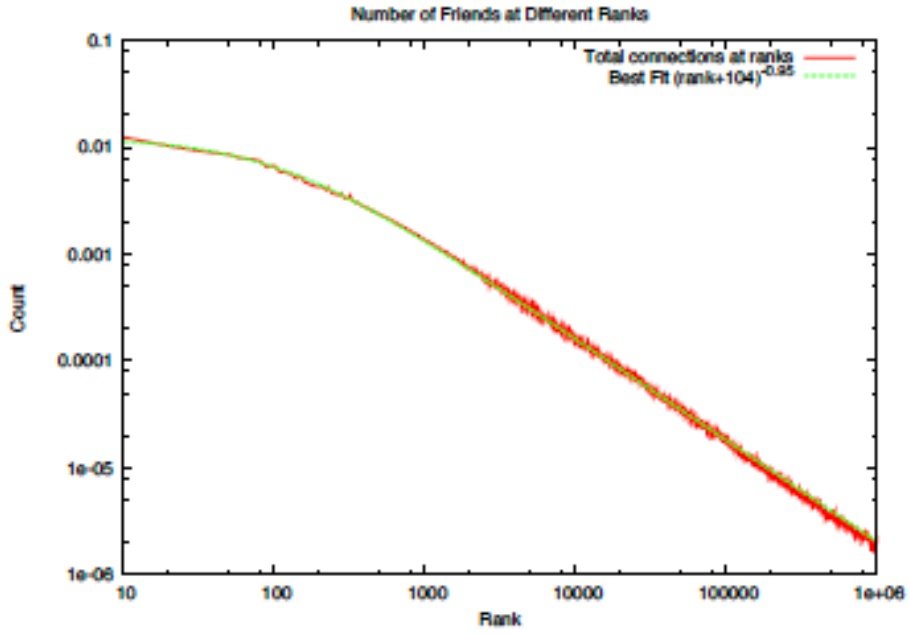



**Figure 9: The rank of a person $v$ relative to $u$ is the number of individuals $w$ such that $d(u, w) < d(u, v)$. Here we show the probability of friendship as a function of rank.**

# Facebook (Backstrom et al., 2010)

## Scoperte principali

- Relazione $P \propto rank^{-0.95}$ (vicina al $-1$ teorico)

## Applicazione pratica

Algoritmo di geolocalizzazione basato sulle amicizie

(predice la posizione di un individuo da un piccolo insieme di utenti geolocalizzati)

## Claim degli autori: supera l'accuratezza dei metodi basati su IP

# Facebook (Backstrom et al., 2010)

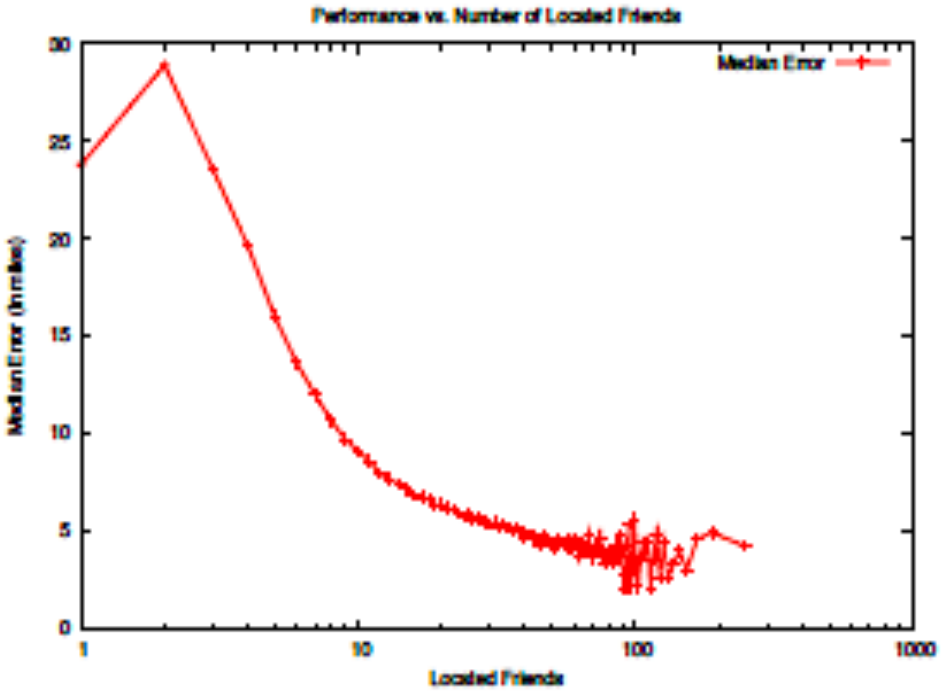



**Figure 13: Prediction performance as a function of friend count. As friend count increases, more information allows for better geolocation**

# Localizzazione

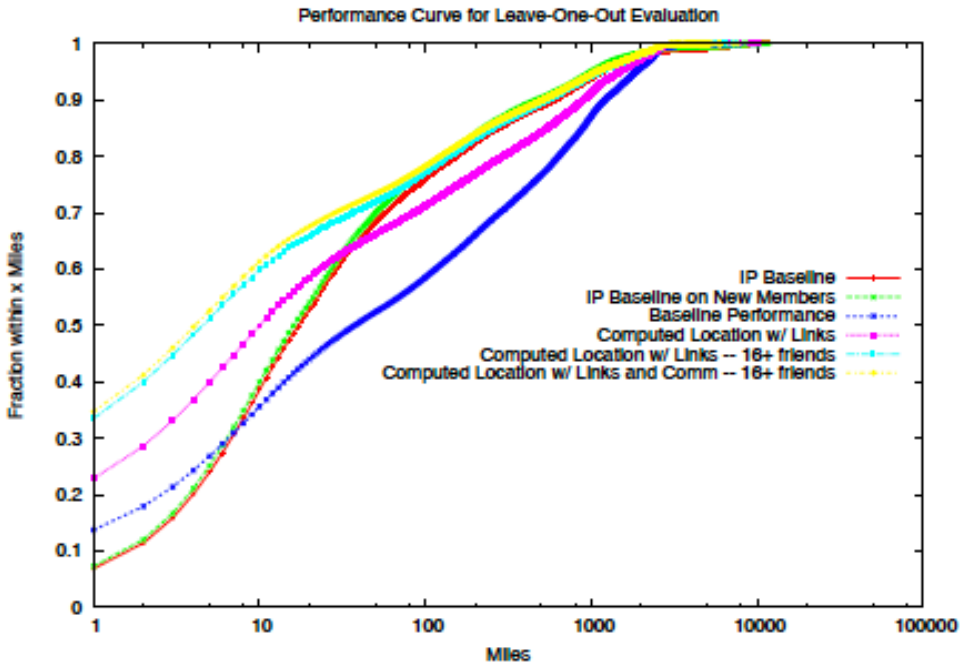



Figura 11: Previsione della posizione.

La figura confronta

- le previsioni esterne di un servizio di geolocalizzazione IP,
- lo stesso servizio limitato a utenti che hanno recentemente aggiornato il loro indirizzo,
- una scelta casuale  basata sulla posizione di un amico e
- tre previsioni utilizzando l'algoritmo (Backstrom et al.,)
  - con tutti i link,
  - per gli utenti con più di 16 amici,
  - per gli utenti con più di 16 amici limitatamente a coloro con cui hanno comunicato di recente.

# Distanza Sociale

## Definizione di distanza sociale

$s(v, w)$ = dimensione del più piccolo gruppo/focus condiviso

Esempi:

- Piccolo: stesso corso universitario
- Medio: stessa università
- Grande: stessa lingua madre

## Teorema (Kleinberg, 2001)

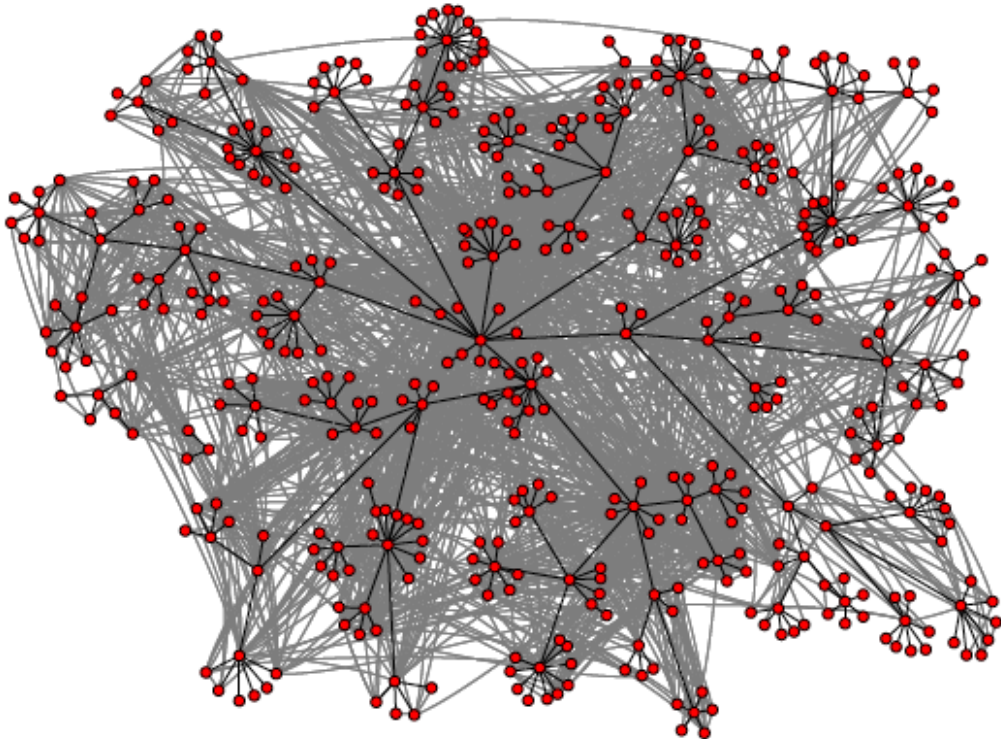
*Se la distribuzione delle amicizie è* $P \propto 1/s(v, w)$, *allora la rete supporta ricerca decentralizzata efficiente*

# Distanza Sociale

Conferma parziale [Adamic e Adar, 2005]

Studio su email di impiegati di HP.

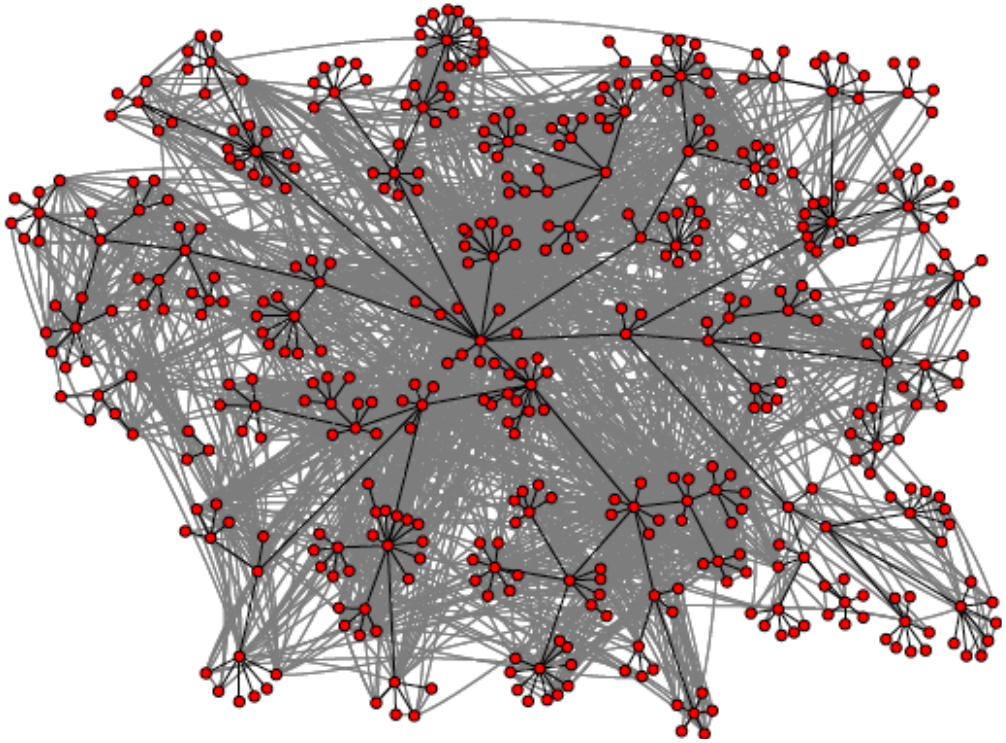
Ottenuta una rete sociale dai registri delle email definendo un contatto sociale come qualcuno con cui un individuo aveva scambiato almeno 6 email in entrambe le direzioni nell'arco di circa 3 mesi; la rete ha 430 nodi con grado medio 10.



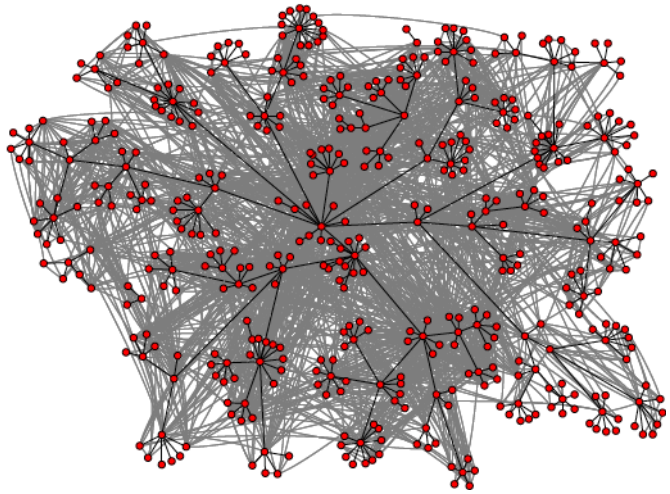
# Distanza Sociale

La soglia relativamente bassa di 6 email cattura comunque *legami deboli* tra persone di dipartimenti diversi con poca sovrapposizione nei loro contatti sociali

Gruppi/focus corrispondono ai dipartimenti nell'organizzazione gerarchica dell'azienda



# Distanza Sociale



**Ricerca decentralizzata.** Gli individui appartengono a gruppi basati su una gerarchia e hanno maggiori probabilità di interagire con individui all'interno dello stesso gruppo, allora si può adottare la strategia greedy.

La h-distanza, utilizzata per navigare nella rete, è calcolata come segue: gli individui hanno h-distanza uno dal loro responsabile e da tutti coloro con cui condividono un responsabile. Le distanze vengono poi assegnate ricorsivamente, in modo che ogni individuo abbia h-distanza 2 dai vicini dei propri primi vicini, e h-distanza 3 dai vicini dei propri secondi vicini, e così via

Numero medio di passi pari a 4
(distanza media pari a 3)

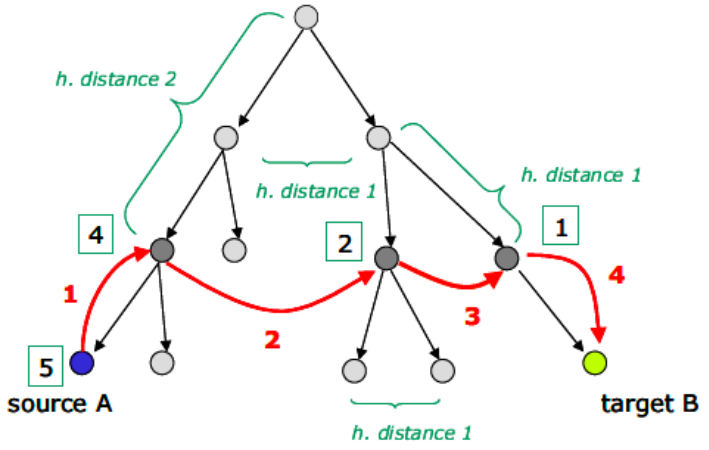



Figure 3: Example illustrating a search path using information about the target's position in the organizational hierarchy to direct a message. Numbers in the square give the h-distance from the target.

# Probabilità di scambio di email vs distanza sociale

$$P \propto s(v,w)^{-3/4}$$

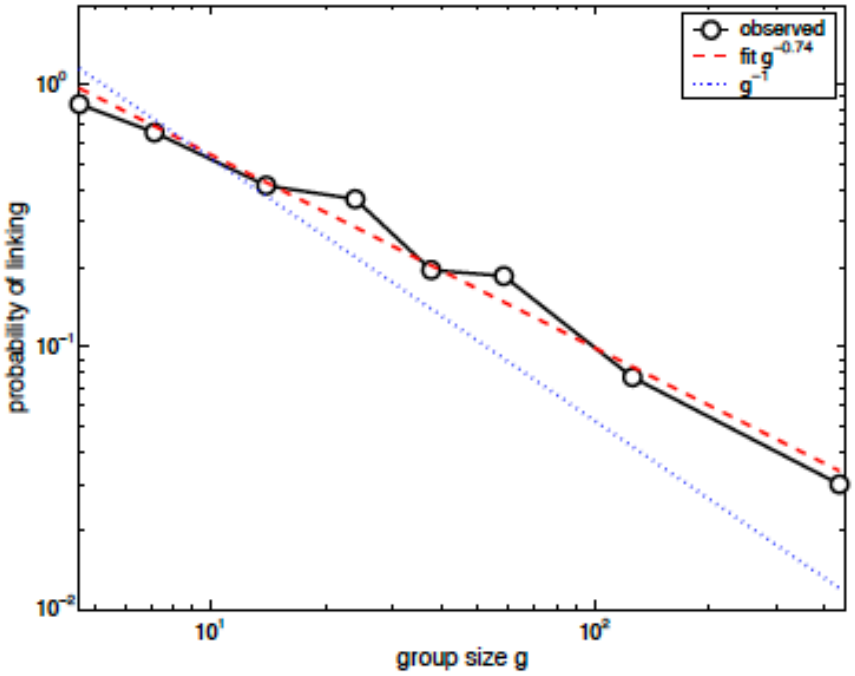



Figure 5: Probability of two individuals corresponding by email as a function of the size of the smallest organizational unit they both belong to. The optimum relationship derived in [7]is $p \sim g^{-1}$, $g$ being the group size. The observed relationship is $p \sim g^{-3/4}$.

# Riassumendo

1. Partiamo da un esperimento (quello di Milgram),

2. costruiamo modelli matematici basati su questo esperimento (combinando collegamenti locali e a lungo raggio),

3. facciamo una previsione basata sui modelli
   (il valore dell'esponente che controlla i collegamenti a lungo raggio), e poi

4. convalidiamo questa previsione su dati reali (da LiveJournal e Facebook, dopo aver generalizzato il modello per usare l'amicizia basata sul rango).

Molto simile a come si spererebbe che si svolgesse
l' interazione tra esperimenti, teorie e misurazioni.

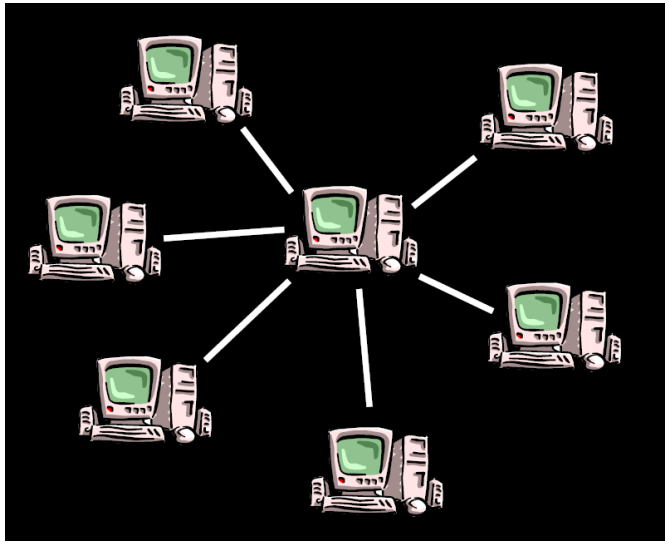
**Allineamento sorprendente tra modelli semplici e dati complessi**

# Riferimenti Bibliografici

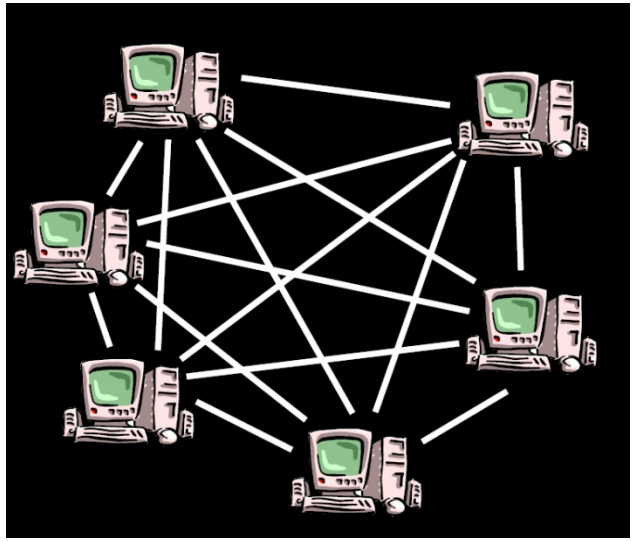

- Liben-Nowell, D., Novak, J., Kumar, R., Raghavan, P., & Tomkins, A. (2005). Geographic routing in social networks.
- Backstrom, L., Sun, E., & Marlow, C. (2010). Find me if you can: improving geographical prediction with social and spatial proximity.
- Adamic, L. A., & Adar, E. (2005). How to search a social network.
- Kleinberg, J. (2001). Small-world phenomena and the dynamics of information.

# Conseguenze Algoritmiche del fenomeno small world

- Come si trovano i file nelle reti peer-to-peer?



**Client-server**



**Peer-to-peer**

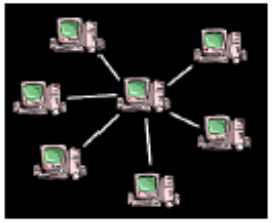
# Peer-to-Peer (P2P)

Il termine Peer-to-Peer (P2P) si riferisce ad un'architettura logica di rete in cui i nodi non sono gerarchizzati sotto forma di client o server, ma sotto forma di *nodi equivalenti* o *peer* che possono cioè fungere sia da cliente che da server verso gli altri host della rete, ogni nodo ha quindi identiche capacità e responsabilità e tutte le comunicazioni sono potenzialmente simmetriche.

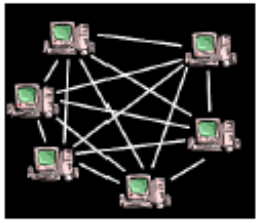
Grazie al modello P2P, i computer possono comunicare e condividere i file e altre risorse, invece di passare attraverso un server centralizzato. Ciascun computer (nodo) è responsabile del passaggio dei dati alle altre macchine.

Le connessioni non nascono spontaneamente, ma devono essere richieste da una delle parti in causa.
Attraverso un client scaricato gli utenti si connettono alla rete. Dopo la connessione, ha inizio la comunicazione tra i nodi che avviene tramite uno scambio di messaggi.
I messaggi servono a:

- segnalare la propria presenza sulla rete
- chiedere una o piu' risorse risorsa
- servire la richiesta di una o piu' risorse
- trasferire le risorse



**Client-server**



**Peer-to-peer**

Quello di nodi paritari non è un concetto recente, infatti ARPAnet consisteva di nodi paritari


L'introduzione del Web e la grande differenza, in termini di prestazioni, fra macchine "Client" e "Server" ha spostato l'attenzione verso i sistemi Client\Server;

Le reti peer-to-peer sono state sviluppate nel corso degli anni '90 e sono state utilizzate principalmente all'interno delle singole aziende per condividere informazioni tra ricercatori.


L'interesse verso questo tipo di protocolli è aumentato con la nascita dei primi sistemi per file-sharing (Napster (1999), Gnutella(2000));

- Nel 2000, 50 milioni di utenti hanno scaricato il Client di Napster;
- Napster ha avuto un picco di traffico di circa 7 TB in un giorno;

# Alcune applicazioni della rete P2P

- Condivisione di file

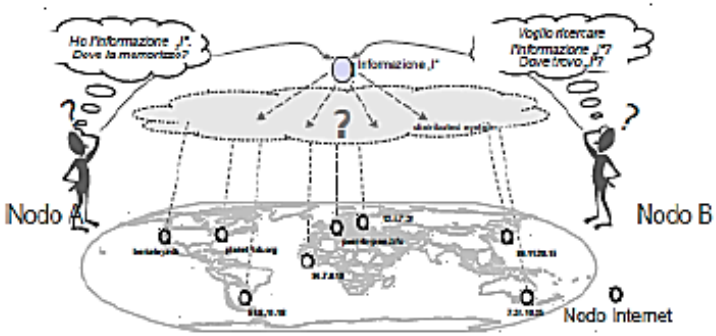
- Messaggistica diretta: la rete P2P fornisce un modo sicuro, rapido ed efficiente per comunicare. Ciò è possibile grazie all'uso della crittografia su entrambi i peer e all'accesso a strumenti di messaggistica facili.

- Telefonia IP: Skype è un buon esempio di applicazione P2P nel VoIP

- Pagamenti: Satispay offre anche il servizio di pagamenti P2P

- Blockchain: l'architettura P2P si basa sul concetto di decentralizzazione. Quando una rete peer-to-peer è abilitata sulla blockchain, aiuta a mantenere una replica completa dei record garantendo allo stesso tempo l'accuratezza dei dati. Allo stesso tempo, le reti peer-to-peer garantiscono anche la sicurezza.

## Come si trovano i file nelle reti peer-to-peer?




**Lookup.** L'operazione di lookup è utilizzata per individuare i dati che sono distribuiti tra una collezione di macchine.

Limitiamo il problema al compito comune di cercare dati che sono associati con una chiave di ricerca univoca (non, ad esempio trovare tutti gli elementi il cui contenuto ha alcune proprietà). La chiave unica ci deve far sapere su quale nodo (potenzialmente uno tra migliaia di nodi) si trovano i dati cercati.

La sfida è trovare un modo per individuare un nodo che memorizza i dati associati alla key in modo distribuito e *scalabile*.

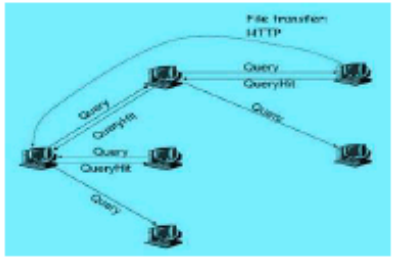
**Scalabile**: Il lavoro richiesto ad un nodo nel sistema non deve crescere (o almeno cresce lentamente) in funzione del numero di nodi nel sistema.

**Lookup.** Ci sono tre approcci di base che possono essere adottati per localizzare i dati:


**Coordinatore centrale**. Questo metodo utilizza un server che si occupa di individuare risorse che sono distribuite tra un insieme di server.

Napster è un classico esempio di questo.

Il Google File System (GFS) è un altro.



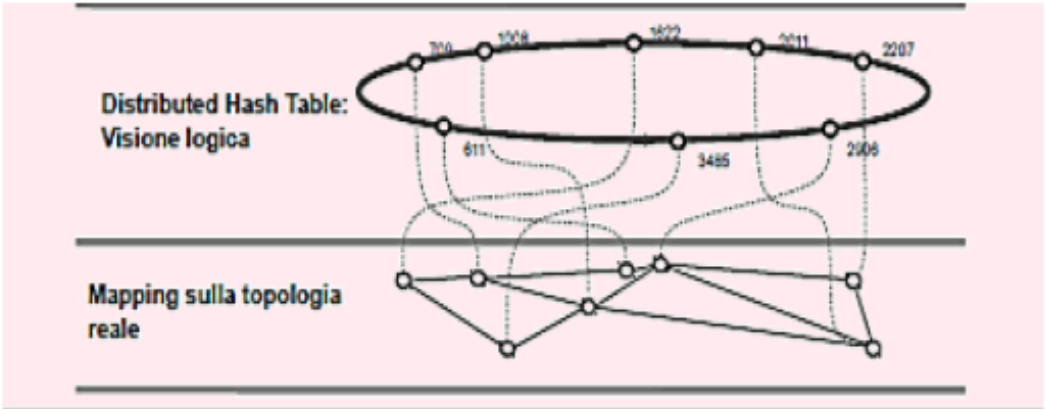

**Flooding** (Allagamento) . Questo metodo si basa sull'invio di query ad un grande insieme di macchine (potenzialmente tutte le macchine della rete), al fine di trovare il nodo che ha i dati di cui abbiamo bisogno.

Gnutella è un esempio di questo per la condivisione di file peer-to-peer.

**Distributed Hash Tables** *(DHT-* tabelle hash distribuite). Questa tecnica si basa sull'uso di tecniche di hashing della chiave per individuare il nodo che memorizza i dati associati. Ci sono molti esempi di questo, tra cui Chord, Amazon Dynamo, CAN, e Tapestry.

In una DHT, ad ogni risorsa e ad ogni nodo è associata una chiave (Key), tale chiave viene creata facendo l'hash del nome della risorsa o dell'IP del nodo.

Ogni nodo del sistema è responsabile di un insieme di risorse/chiavi



# Chord

Il protocollo Chord mappa una key (chiave- che coincide con il nome del file) ad un nodo:

- Le chiavi sono i file che stiamo cercando.
  Il computer che mantiene la chiave può poi puntare alla vera posizione del file.

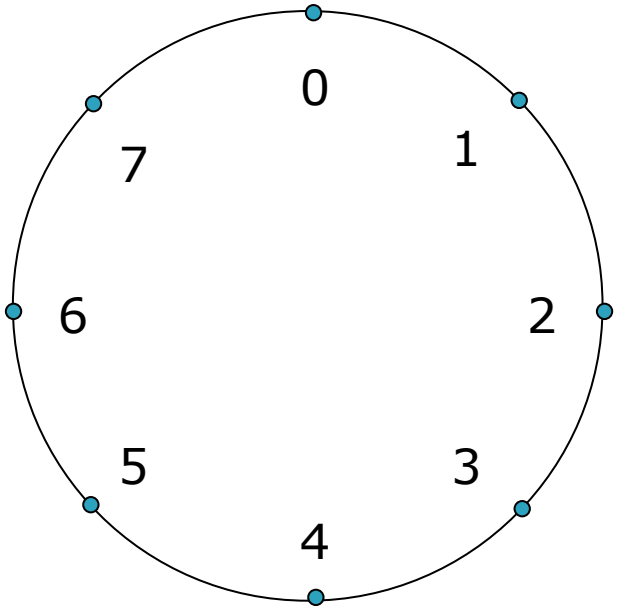
- Chiavi e nodi hanno ID di m bit loro assegnati :
  - ID del nodo è un hash-code dell'indirizzo IP (32 bit)
  - ID della chiave è un hash-code del file

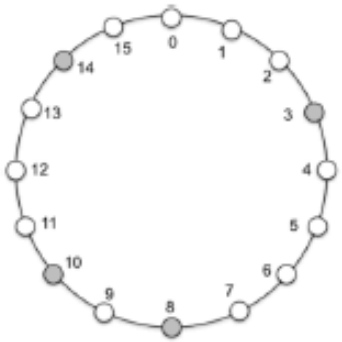
# Chord su ring

In uno spazio di ID di *m* bit, ci sono $2^m$ identificatori.
Le ID sono quindi ordinate lungo un anello modulo $2^m$.

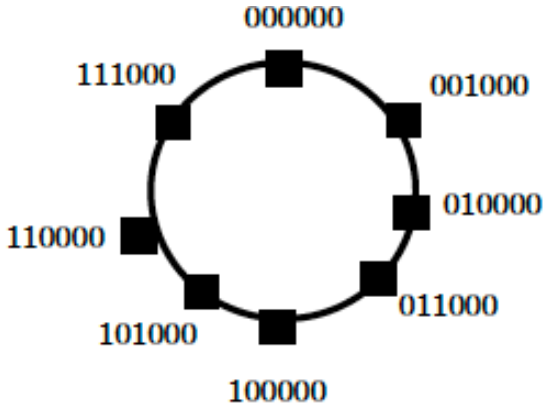
L'anello delle ID è chiamato anello Chord.

Il file k è assegnato ad un nodo con ID almeno k
Questo nodo è il nodo successore della chiave k ed è indicato come successor(k).





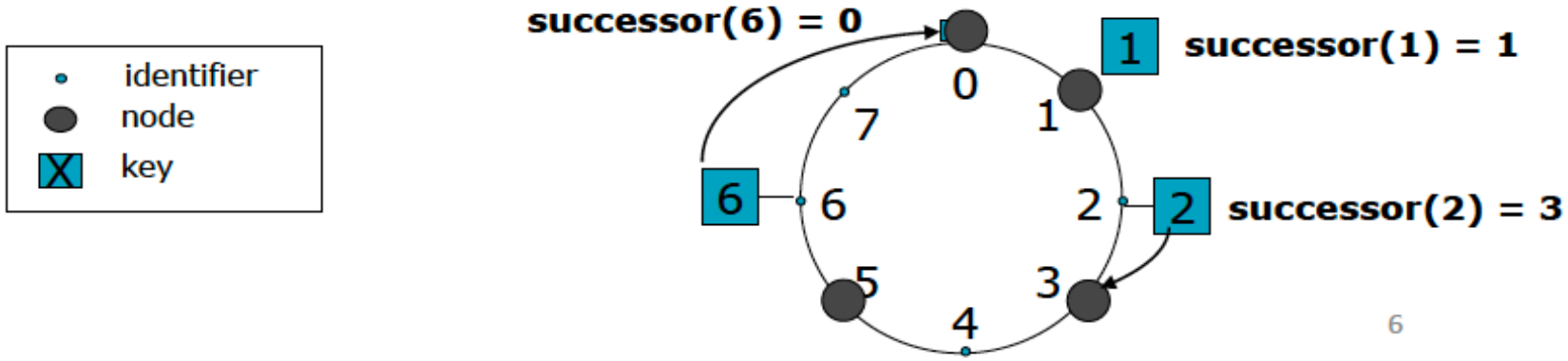

Un anello Chord con m=4. Abbiamo quindi 16 identificatori da 0 a 15 disposti lungo l'anello; solo a 4 di tali ID (i nodi pieni 3, 8, 10 e 14) corrispondono a 4 macchine effettivamente presenti nella rete e mappate su tali nodi.




Un anello Chord con m=6. Abbiamo quindi 64 identificatori da 0 a 63 disposti lungo l'anello; di cui 8 (i nodi con ID 000000, 001000,...) sono nodi effettivamente presenti nella rete.

# Chord su ring

**Assegnazione delle chiavi.** Il file k è assegnato ad un nodo con ID *almeno* k (in modulo)
Questo nodo è il nodo successore della chiave k ed è indicato come *successor(k).*



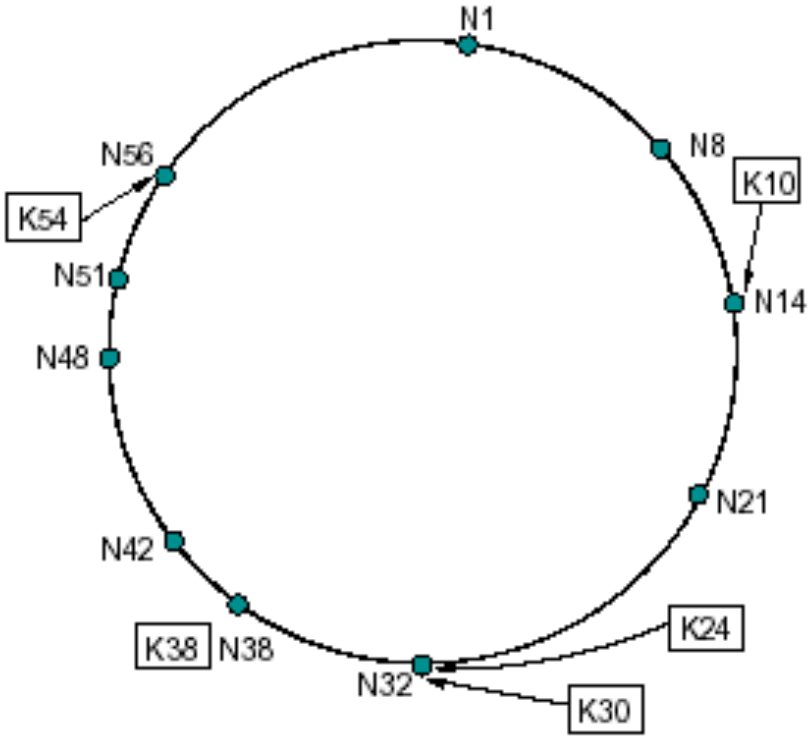

La figura mostra un anello di 8 ID, con 4 nodi ( corrispondenti alle ID 0, 1, 3, 5) e 3 chiavi (1,2 e 6). Poichè il nodo 1 è presente nel ring, allora *successor(1)=1.* La chiave 2 ha *successor(2)=3*, poiché 2 non è un nodo presente nel ring ma il nodo immediatamente maggiore lo è. Infine abbiamo *successor(6)=0* poichè nella rete non vi sono nodi corrispondenti ad ID 6 e 7, quindi il primo nodo presente con ID almeno 6 è il nodo 0.

# Chord su ring

Per m = 6, # di identificatori è 64.

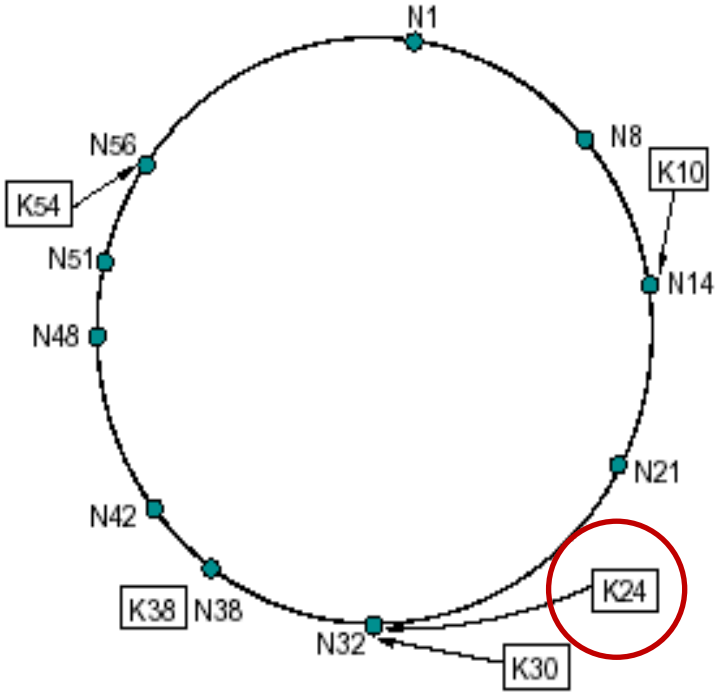
Il seguente anello Chord ha 10 nodi e mantiene 5 chiavi.

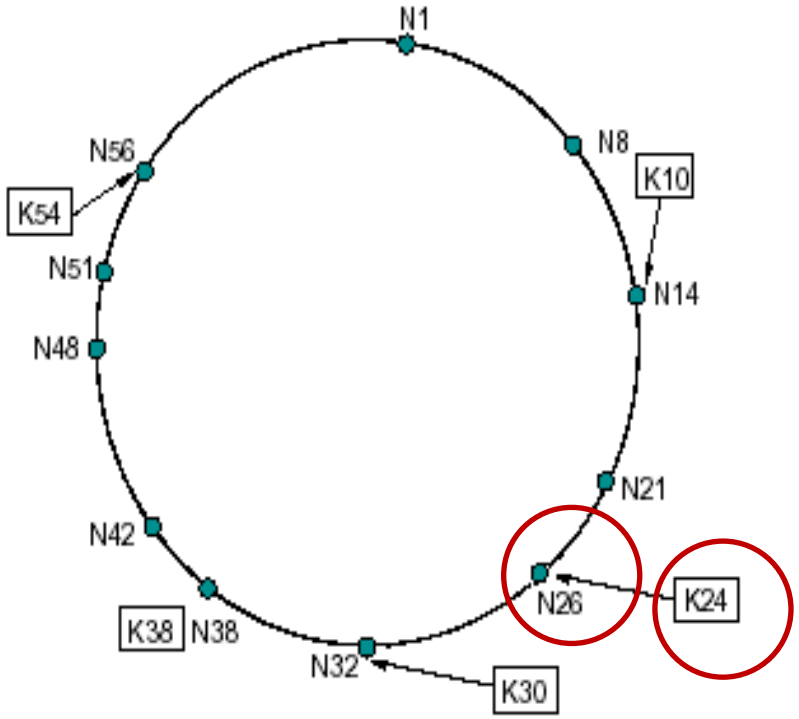
Il successore della  chiave 10 è il nodo 14.



# Chord su ring

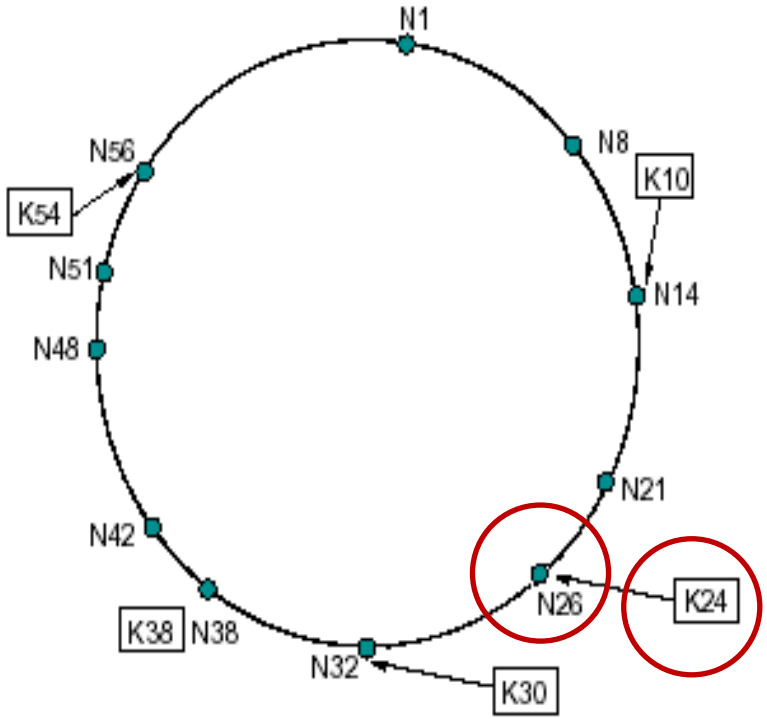
- Quando un nodo *n* si unisce alla rete, alcune chiavi precedentemente assegnate al successore di n sono riassegnate a n.
  - Il nodo 26 si unisce:

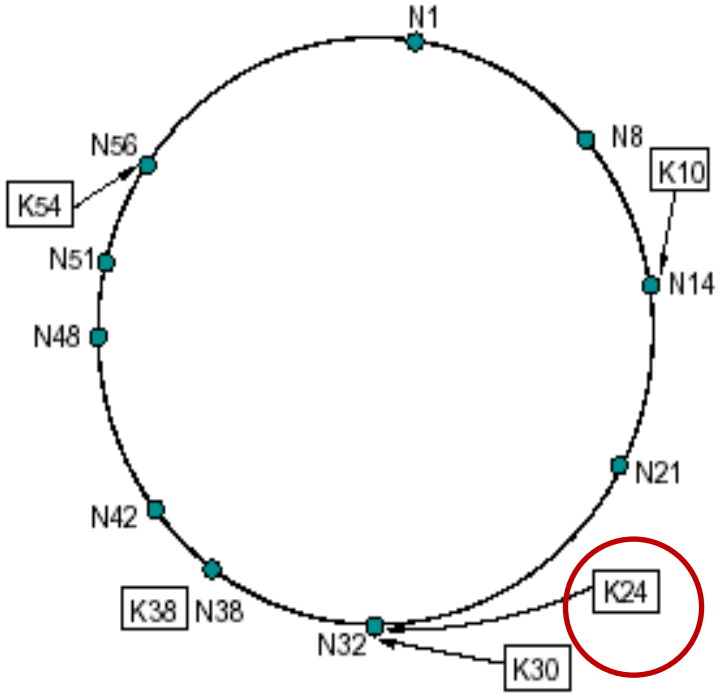


# Chord su ring

- Quando il nodo n lascia la rete, tutte  e chiavi assegnate a n vengono vengono riassegnate al  successore di n.



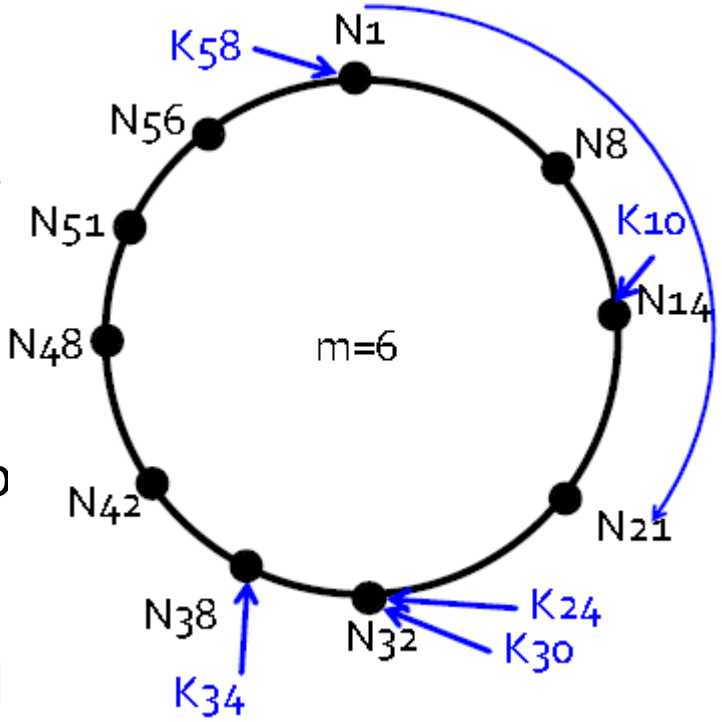



# Chord su ring

## Assunzioni base

1. Supponiamo di avere N nodi e K chiavi (file)
2. Ogni nodo conosce l'indirizzo IP dei sui vicini immediati

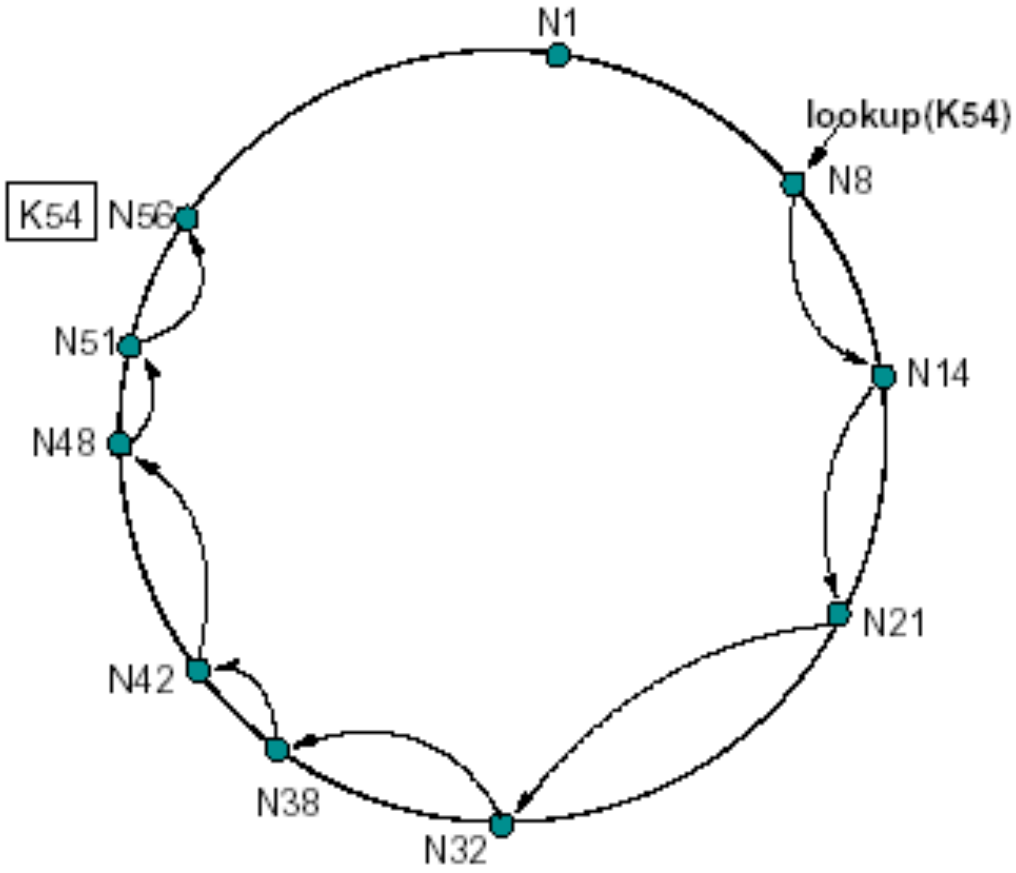
Se ciascun nodo sa come contattare il suo successore sul ring degli identificatori, tutti nodi possono essere visitati in ordine lineare.




**Ricerca sequenziale** Una query per un dato identificativo potrebbe essere passata intorno al ring attraverso puntatori al successore fino a quando non si incontra il nodo che contiene la chiave.

➔ **tempo O(N)**

# Da N8 trova la chiave con ID 54



# Ricerca veloce

- Un nodo mantiene una tabella di $m = log\ (n)$ elementi, detta **Finger Table**

- L'elemento *i*-esimo della finger table per un nodo v contiene l'indirizzo del $2^i$ *-mo* vicino di *v*
  - punta al *primo* nodo con $ID \geq v + 2^i$

  - **Nota: quando un nodo si unisce alla rete violiamo i puntatori a lungo raggio di tutti gli altri nodi (*e dobbiamo ovviare!)***

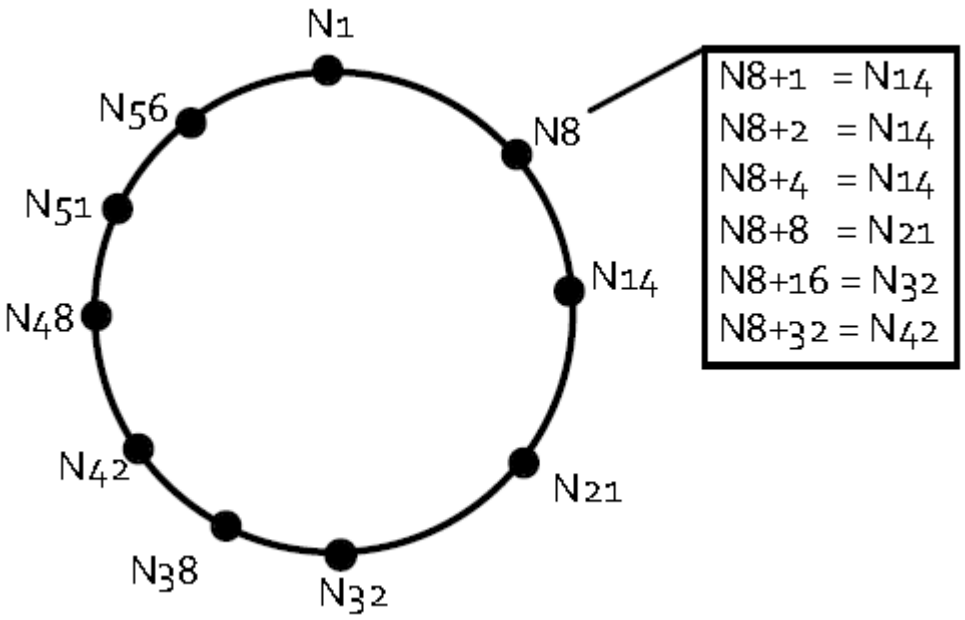
**Algoritmo di ricerca:**
Seguire il collegamento più lungo che non supera il target

➔ Ad ogni passo si **dimezza** la distanza dal target!

# La tabella di N8 su un ring di 64 nodi

**Algoritmo di ricerca:**

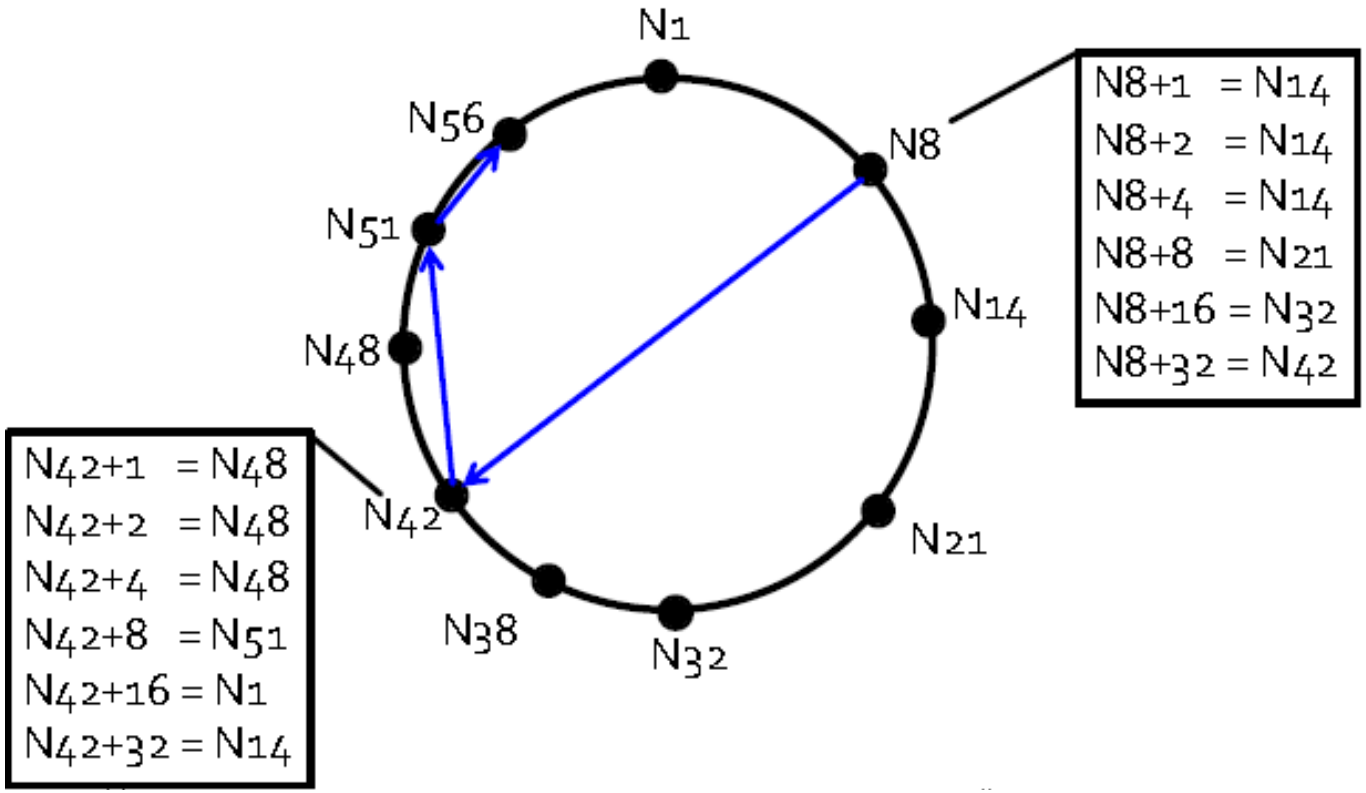
Seguire il collegamento più lungo che non supera il target



# Da N8 trova la chiave con ID 54

**Algoritmo di ricerca:**

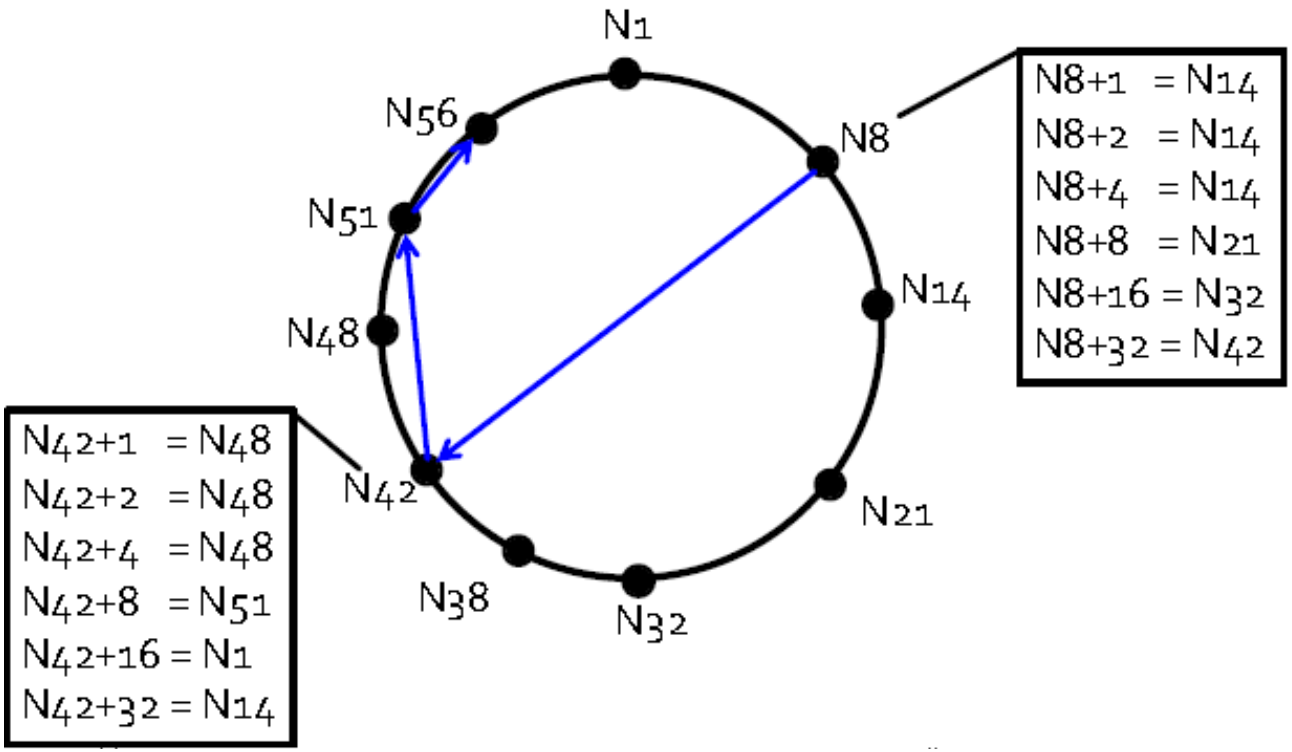
Seguire il collegamento più lungo che non supera il target



# Ricerca veloce

Ogni nodo ha nella finger table elementi ad intervalli di dimensione pari a potenze di 2 crescenti intorno al ring degli identificativi

➔ ogni nodo può inoltrare una richiesta ad almeno metà strada tra il nodo stesso e l'identificatore di destinazione.



# Ricerca veloce

**Fatto:** La ricerca di una chiave in una rete di *N* nodi visita *O (log N)* nodi.

**Dim.** Supponiamo che il nodo *n* effettua una query per la chiave *k* e che la chiave *k* risiede al nodo target *t*.

Quanti passi servono per raggiungere *t*?

# Ricerca veloce

Nella pratica possiamo assumere che gli identificatori di nodo e chiave siano casuali.

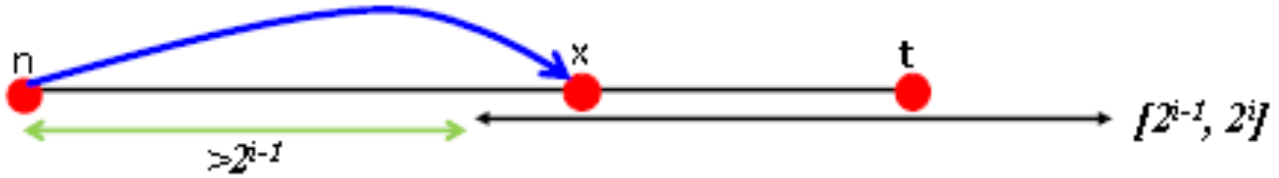
➔ Identificatori distribuiti uniformemente sull'anello

➔ La probabilità che un nodo occupi una posizione fissata sull'anello è $1 / 2^m$

➔ In un intervallo di ampiezza $2^m / N^2$ la probabilità di avere un nodo è

(ampiezza intervallo)(probabilità nodo in posizione fissata dell'intervallo)

$= (2^m / N^2)(1 / 2^m) = 1/N^2$




Ad ogni inoltro la distanza si dimezza (almeno)

➔ Dopo 2log N inoltri la distanza tra il nodo della query corrente e la chiave k sarà ridotta al massimo a $2^m / 2^{2\log N} = 2^m / N^2$.

➔ La probabilità che qualsiasi altro nodo si trovi in questo intervallo è al massimo $1 / N^2$, che è trascurabile.

➔ Il successivo di inoltro troverà il file desiderato nodo.

**Il numero di inoltri necessari sarà O (log N) con alta probabilità.**