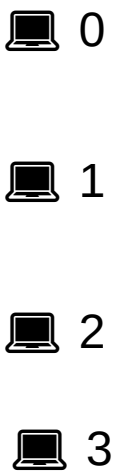
# Transfert de données dans les architectures **pair-à-pair**

# Chord


- SHA-1(fichie) mod 4 = 0 ou
- SHA-1(fichie) mod 4 = 1 ou
- SHA-1(fichie) mod 4 = 1 ou
- SHA-1(fichie) mod 4 = 3



# Chord

- Developers: I. Stoica, D. Karger, F. Kaashoek, H.Balakrishnan, R. Morris, Berkeley, and MIT
- Chord est un projet P2P subventionné par le gouvernement des États-Unis d'Amérique, utilisant notamment une topologie en anneau. Il a pour particularité de disposer d'algorithmes d'une complexité d'au plus O(log N) requêtes pour trouver une information dans un anneau de N éléments grâce à une table de hachage distribuée.

# Chord

Avantages

- Décentralisé:
- Passage à l'échelle
- Équilibrage de charge
- Disponibilité

# Chord

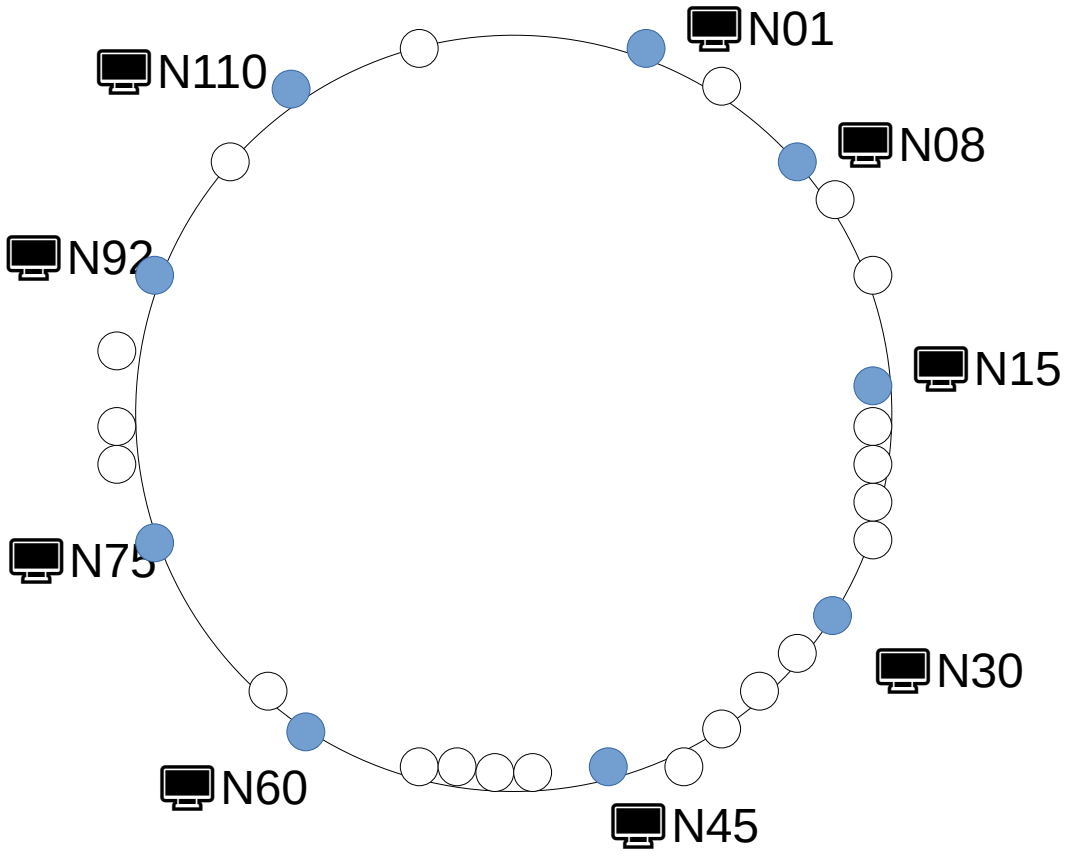
- Utilise le hachage cohérent
- SHA-1(ip_address,port) → 160 bit string ==> m bits
- Appelé nœud id  nombre entre ( 0 => $2^m$ -1 )
- Peut mapper des nœuds à l'un des $2^m$ points logiques en cercle.

# Chord

m=7

$2^7 = 128$

Nodes =0,1,2……125,126,127



# Chord

m=6

$2^6 = 64$

Nodes =0,1,2……61,62,63

Succ(n) = le premier noeud dont l'identifiant est suivi l'identifiant N.

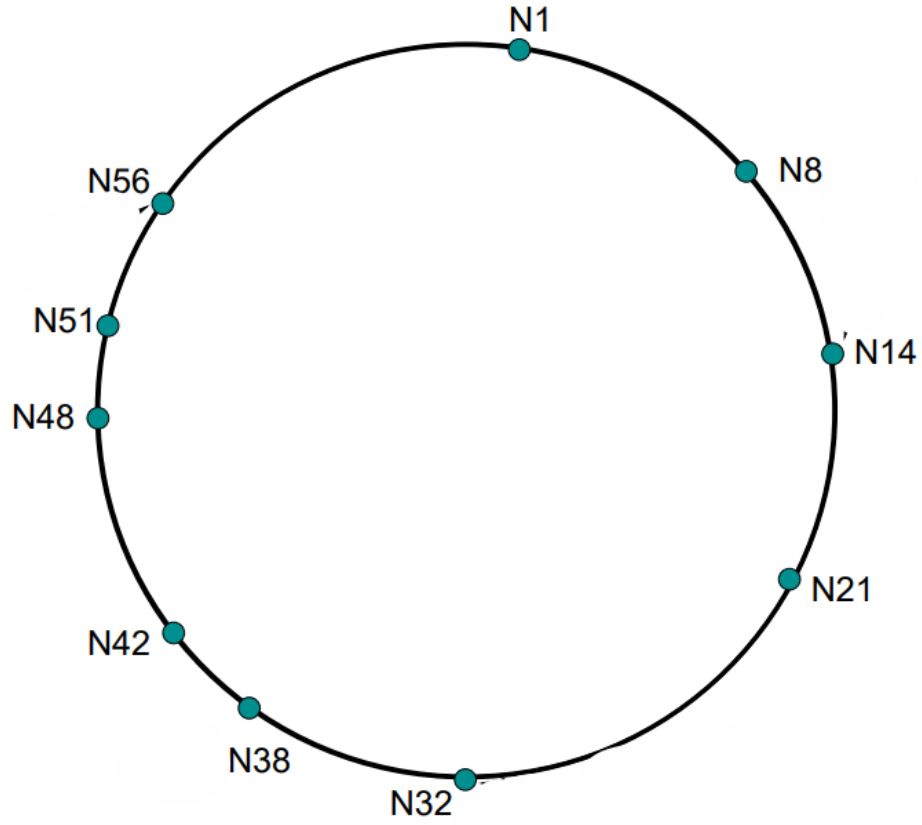
exemple

Succ(N10)=N14

Succ(N15)=N21

Succ(N40)=N42

Succ(N45)=N48



# Chord

m=6

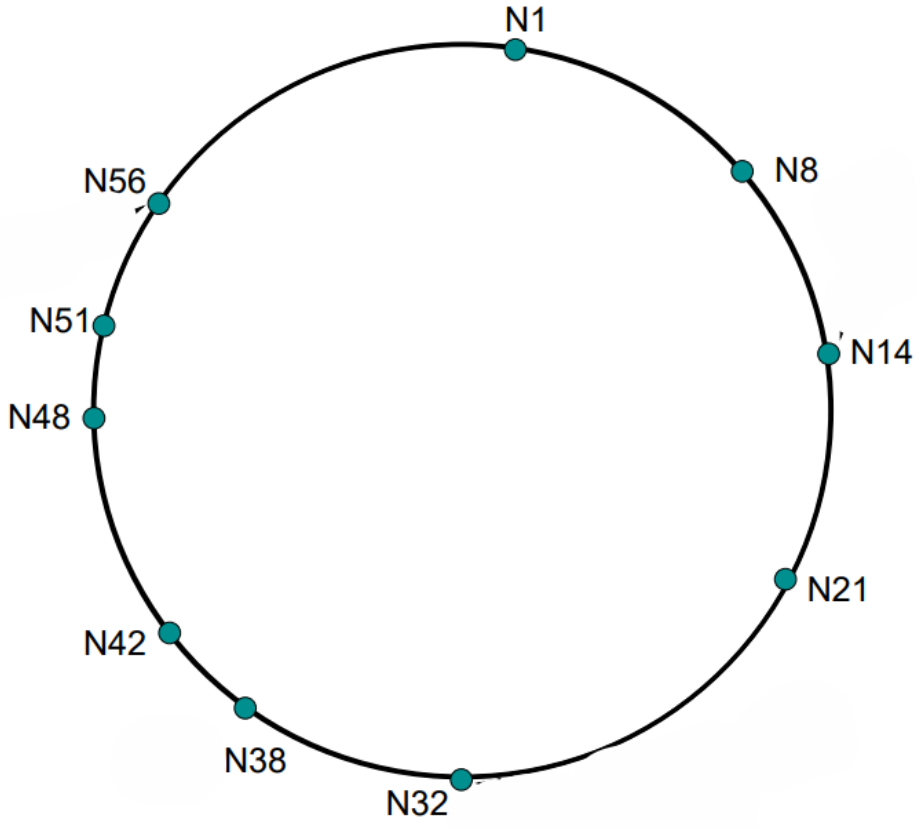
$2^6 = 64$

Nodes =0,1,2……61,62,63

Les noms de fichiers sont également mappés en utilisant la même fonction de hachage cohérente

SHA-1(fichie) → 160bit (clef k)

Le fichier est stocké sur le premier noeud dont l'ID est supérieur ou égal à sa clef K mod $2^m$



# Chord

m=6

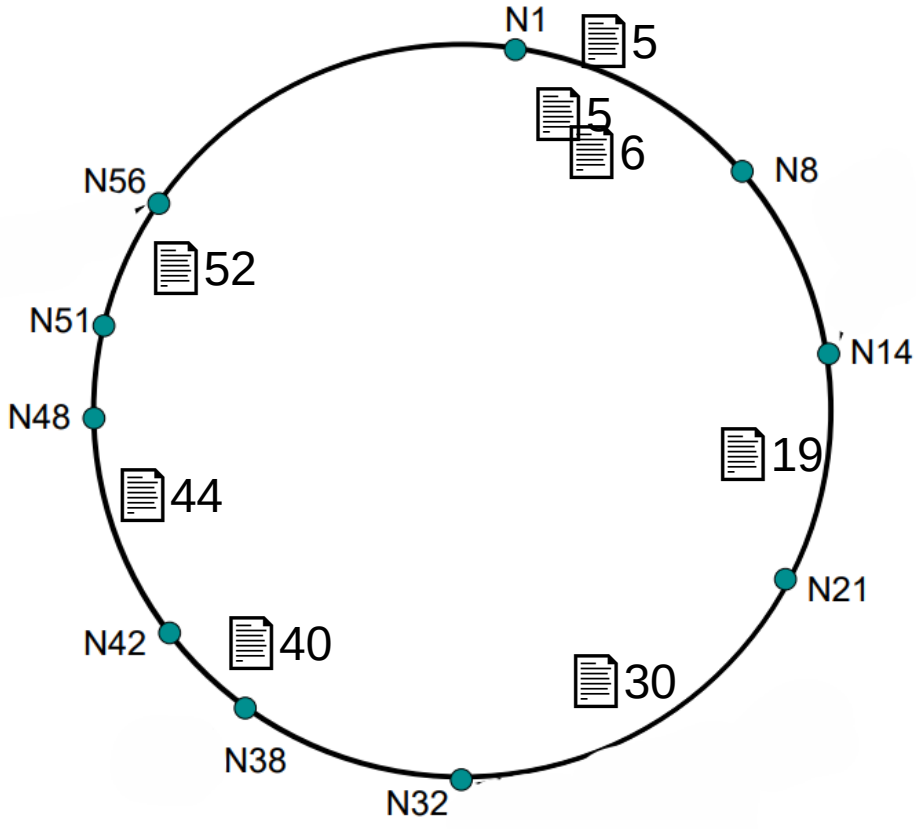
$2^6 = 64$

Nodes =0,1,2……61,62,63

Les noms de fichiers sont également mappés en utilisant la même fonction de hachage cohérente

SHA-1(fichie) → 160bit (clef k)

Le fichier est stocké sur le premier noeud dont l'ID est supérieur ou égal à sa clef k mod $2^m$



# Chord

m=6

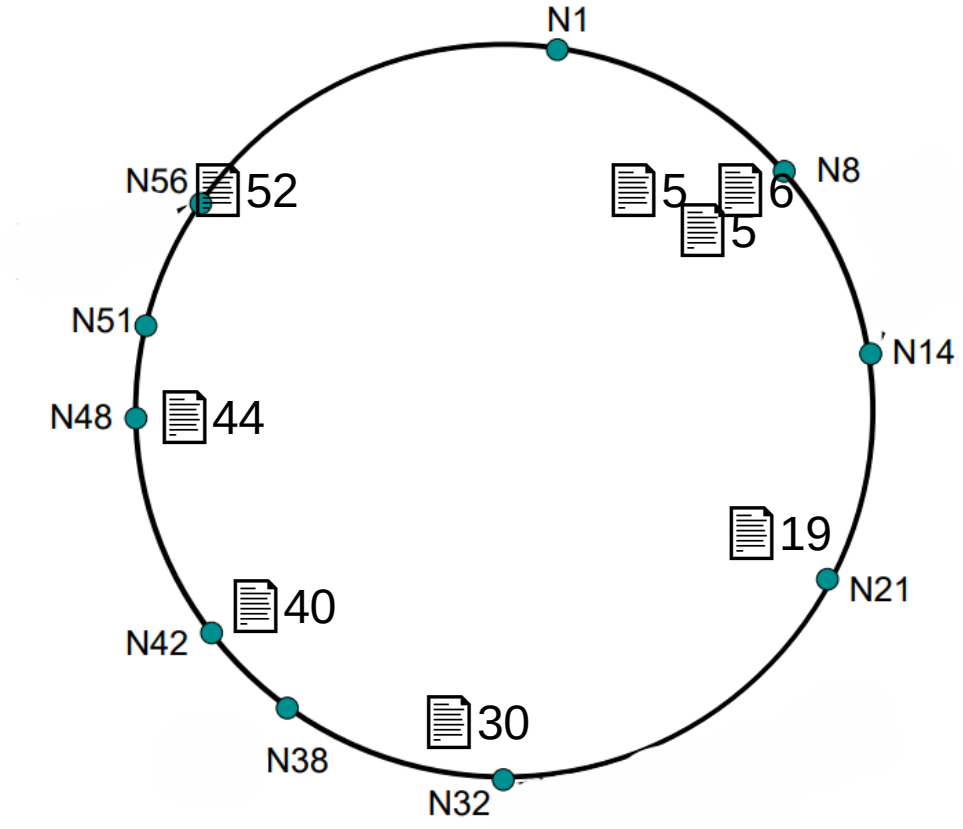
$2^6 = 64$

Nodes =0,1,2……61,62,63

Les noms de fichiers sont également mappés en utilisant la même fonction de hachage cohérente

SHA-1(fichie)→160bit (clé)

Le fichier est stocké sur le premier noeud dont l'ID est supérieur ou égal à sa clef mod $2^m$

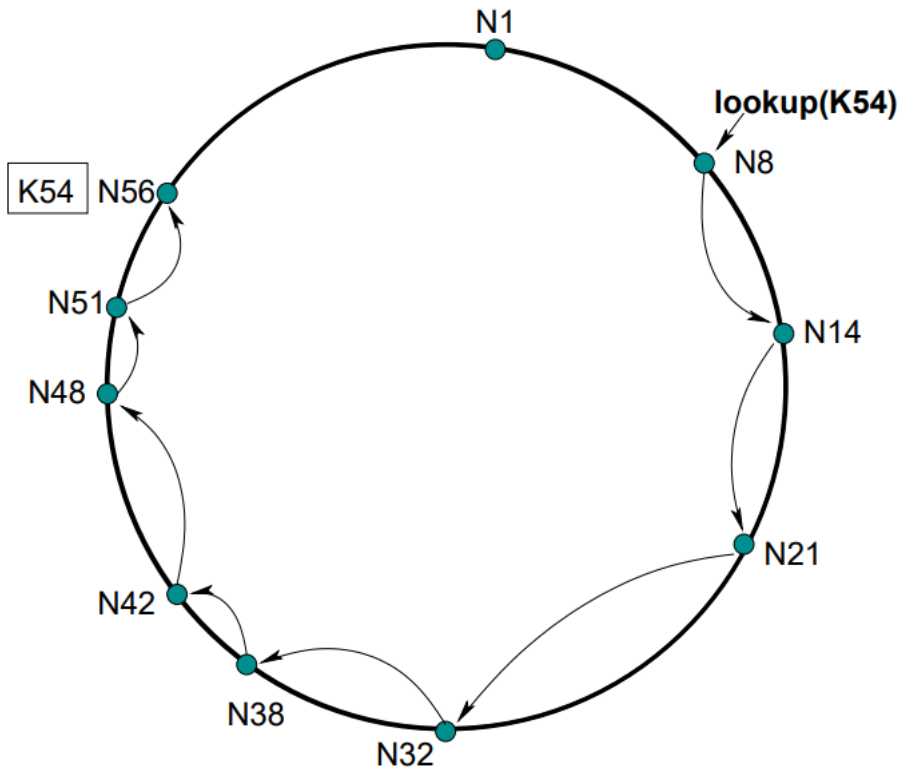


# Chord

Recherche simple

```
// ask node n to find the successor of id
n.find_successor(id)
    if (id ∈ (n, successor])
        return successor;
    else
        // forward the query around the circle
        return successor.find_successor(id);
```

(a)




(b)

# Chord

recherche avancée

Finget table pour N8

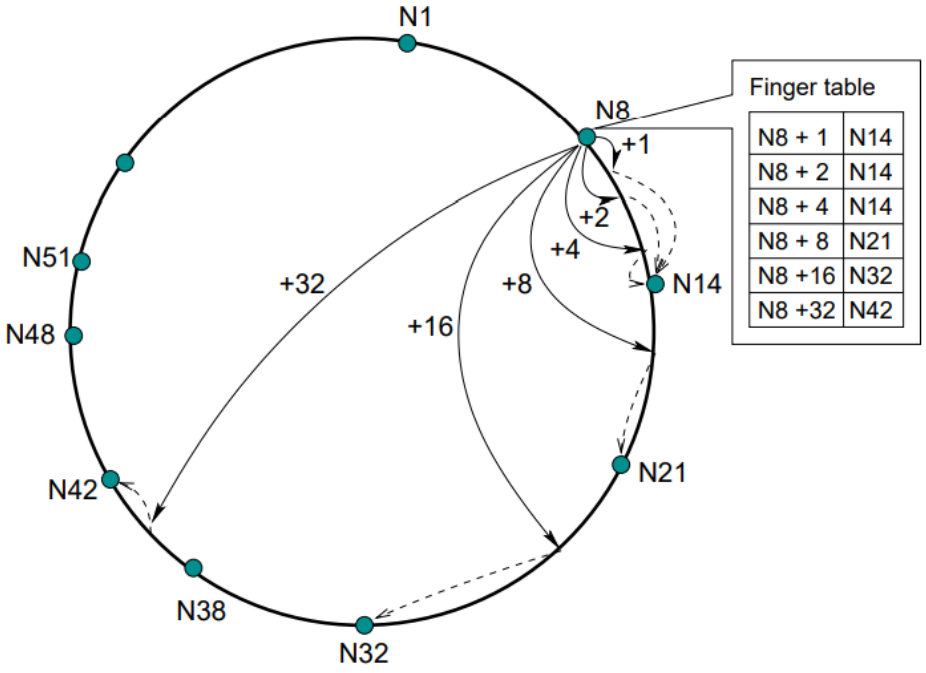
| i | f(i) | | succ(n) |
|---|---|---|---|
| 1 | $(N + 2^{i-1}) \bmod 2^m$ | (8+1)mod64=9 | N14 |
| 2 | $(N + 2^{i-1}) \bmod 2^m$ | (8+2)mod64=10 | N14 |
| 3 | $(N + 2^{i-1}) \bmod 2^m$ | (8+4)mod64=12 | N14 |
| 4 | $(N + 2^{i-1}) \bmod 2^m$ | (8+8)mod64=16 | N21 |
| 5 | $(N + 2^{i-1}) \bmod 2^m$ | (8+16)mod64=24 | N32 |
| 6 | $(N + 2^{i-1}) \bmod 2^m$ | (8+32)mod64=40 | N42 |

1- Si ID est entre N et succ(N), le succ(N) est retourné.

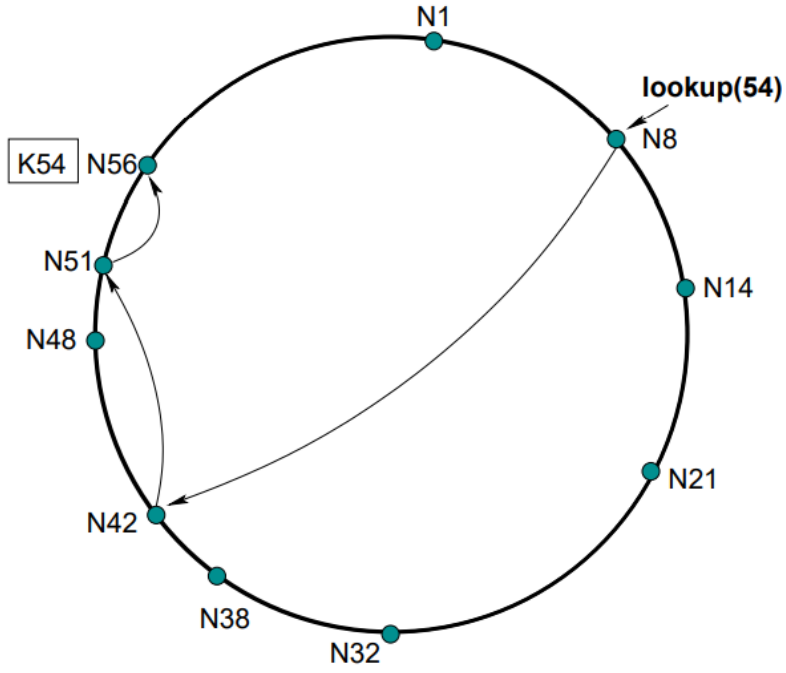
2- Sinon, la recherche est effectuée au noeud N*, qui est le noeud dans Finger table de N qui précède l'ID .

# Chord

recherche avancée




(a)




(b)