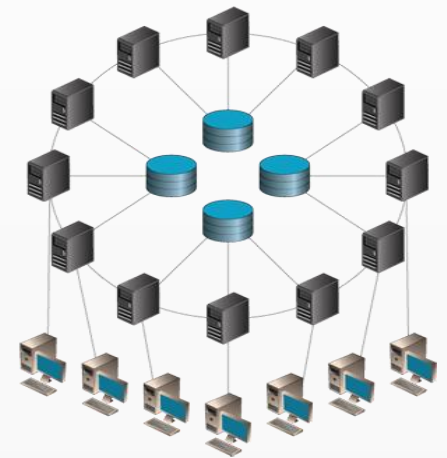


Arquiteturas P2P

# SMD0050 - SISTEMAS DISTRIBUÍDOS

1



Slides são baseados nos slides do Coulouris e Tanenbaum

# O que é P2P?

- Peer-to-peer (do inglês par-a-par ou simplesmente ponto-a-ponto)
- Arquitetura de redes de computadores onde cada um dos pontos ou nós da rede funciona tanto como cliente quanto como servidor.
  - compartilhamentos de serviços e dados sem a necessidade de um servidor central



# Diferenças entre P2P e Cliente-Servidor

- Descentralização



Cliente-Servidor



P2P

# Origem do P2P

- Surgimento por volta do ano 2000
- Primeiros sistemas P2P parcialmente descentralizados
- A “cooperação” era o principal objetivo e o “valor” da rede



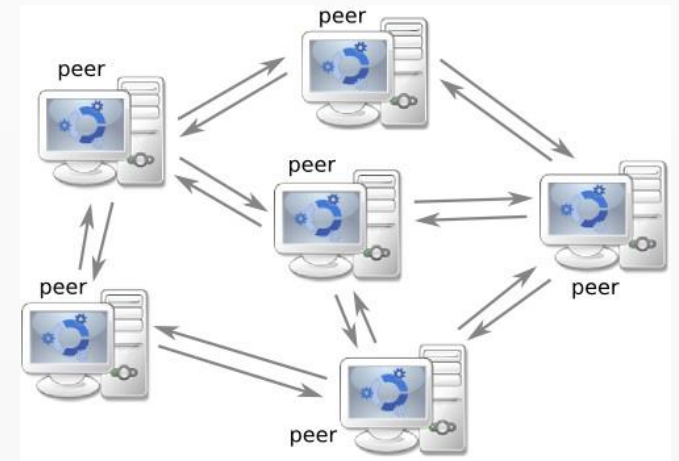
# Características – P2P Ideal

- Cada participante age como cliente e servidor ao mesmo tempo
- Cada cliente “paga” a sua participação fornecendo acesso a (alguns de) seus recursos
- Sem coordenação central
- Sem banco de dados central
- Sem local único de falha ou gargalo



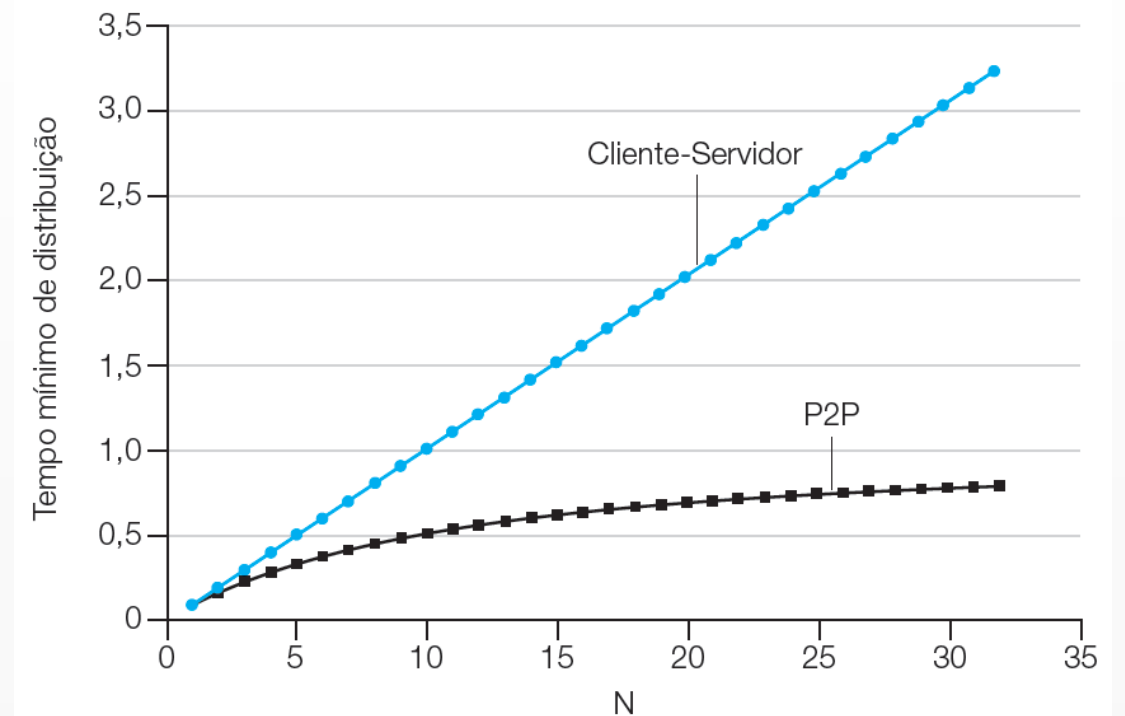
# Características – P2P Ideal

- Nenhum ponto (peer) tem visão global do sistema
  - Pontos são autônomos
  - Pontos e conexões não são confiáveis
- Comportamento global definido por interações locais
  - Todos os dados e serviços são acessíveis de qualquer ponto



# Distribuição de arquivos P2P

- Tempo de distribuição
- para arquiteturas P2P
- e cliente-servidor





# P2P- Redes de Sobreposição

- P2P – Distribuição horizontal
  - Processos são visto como “iguais”
- Como consequência, grande parte da interação entre processos é simétrica
  - cada processo agirá como um cliente e um servidor ao mesmo tempo.
- Arquiteturas peer-to-peer se desenvolvem em torno da questão de como organizar os processos em uma rede de sobreposição





# Modelos de P2P e aplicações - Classificação

Centralized Service Location (CSL ou CIA)

- – Busca centralizada
- – Exemplo Napster -

<https://www.youtube.com/watch?v=7AF18DUIH1Y>



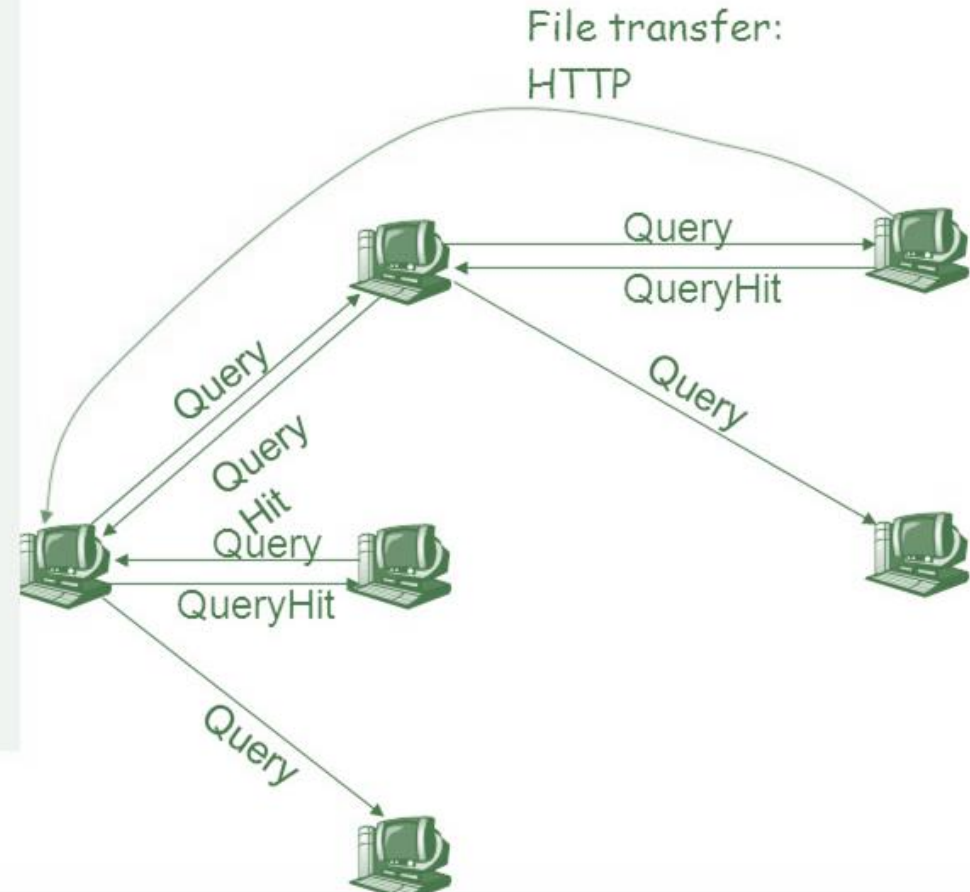
# Modelos de P2P e aplicações - Classificação

- Flooding-based Service Location (FSL ou DIFA)
  - Busca baseada em inundação
  - Gnutella
- Distributed Hash Table-based Service Location (DHT ou DIHA)
  - Busca baseada em tabela de hash distribuída
  - CAN, Pastry, Tapestry, Chord

# Modelos de P2P e aplicações – Gnutella Flooding

## Query Flooding:

- **Join:** on startup, client contacts a few other nodes (learn from bootstrap-node); these become its “neighbors” (overlay!! 😊)
- **Publish:** no need
- **Search:** ask “neighbors”, who ask their neighbors, and so on... when/if found, reply to sender.
- **Fetch:** get the file directly from peer

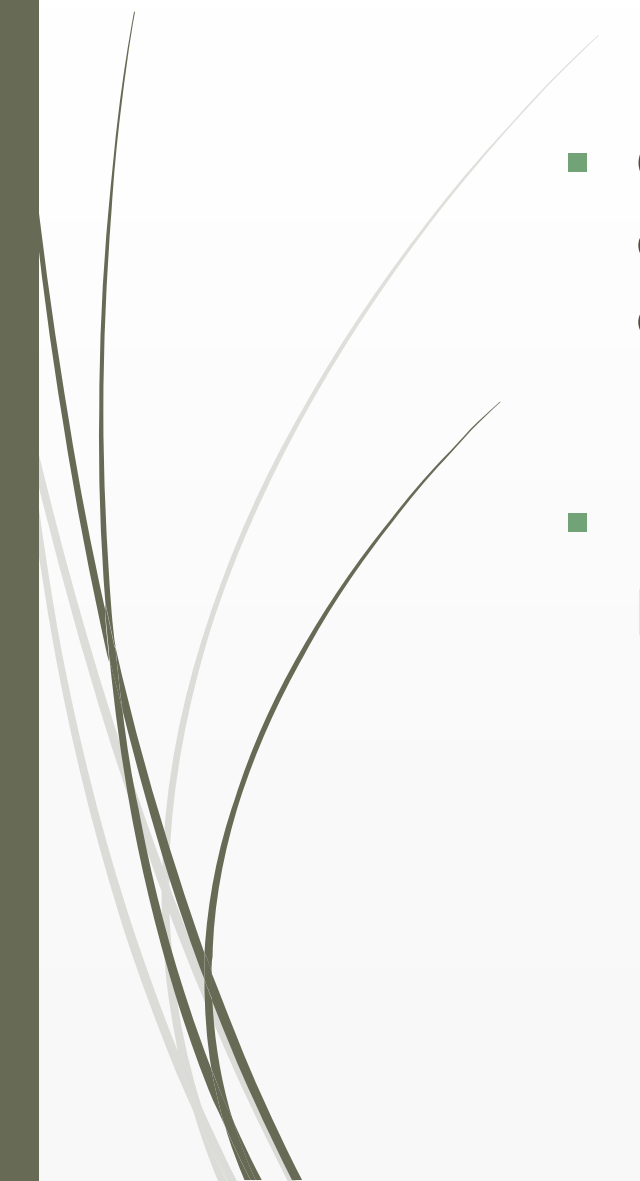


# Exercício

- Quais são as vantagens e desvantagens do modelo de inundação em relação a arquitetura do Napster?
- Enumere uma possível solução para diminuir o número de mensagens e o tempo de resposta



# Sistemas Distribuídos Colaborativos

- 
- O mais importante é que, ao consultar um item de dado, o endereço de rede do nó responsável por aquele item de dado é retornado.
  - Na verdade, consegue-se isso roteando uma requisição para um item de dado até o nó responsável.



# Arquiteturas Estruturadas

Quando um usuário final estiver procurando um arquivo....

Ele baixa porções do arquivo de outros usuários até que as porções transferidas possam ser montadas em conjunto, resultando no arquivo completo



O que isso lembra?



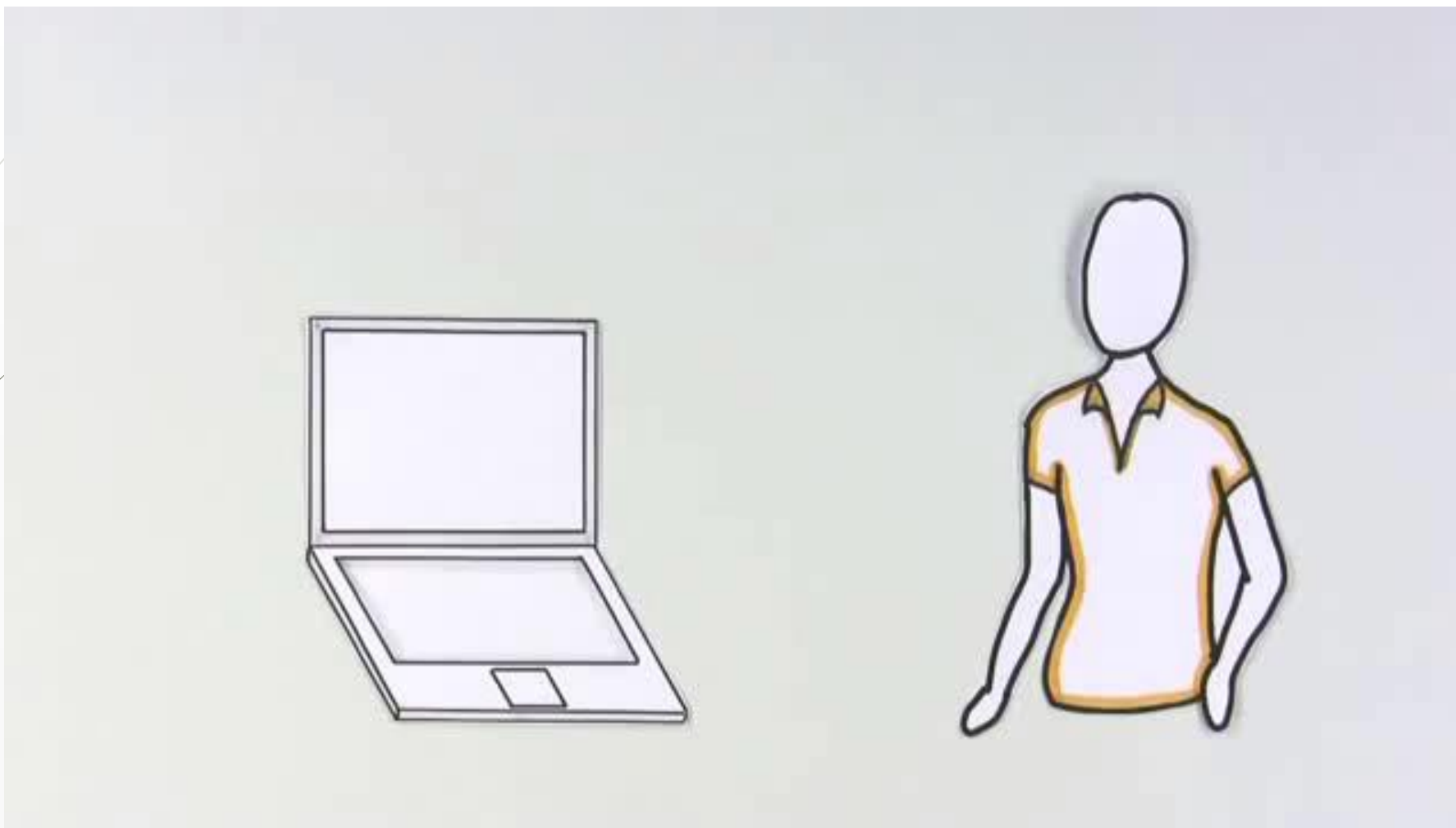
**BitTorrent®**



# A companhia BitTorrent

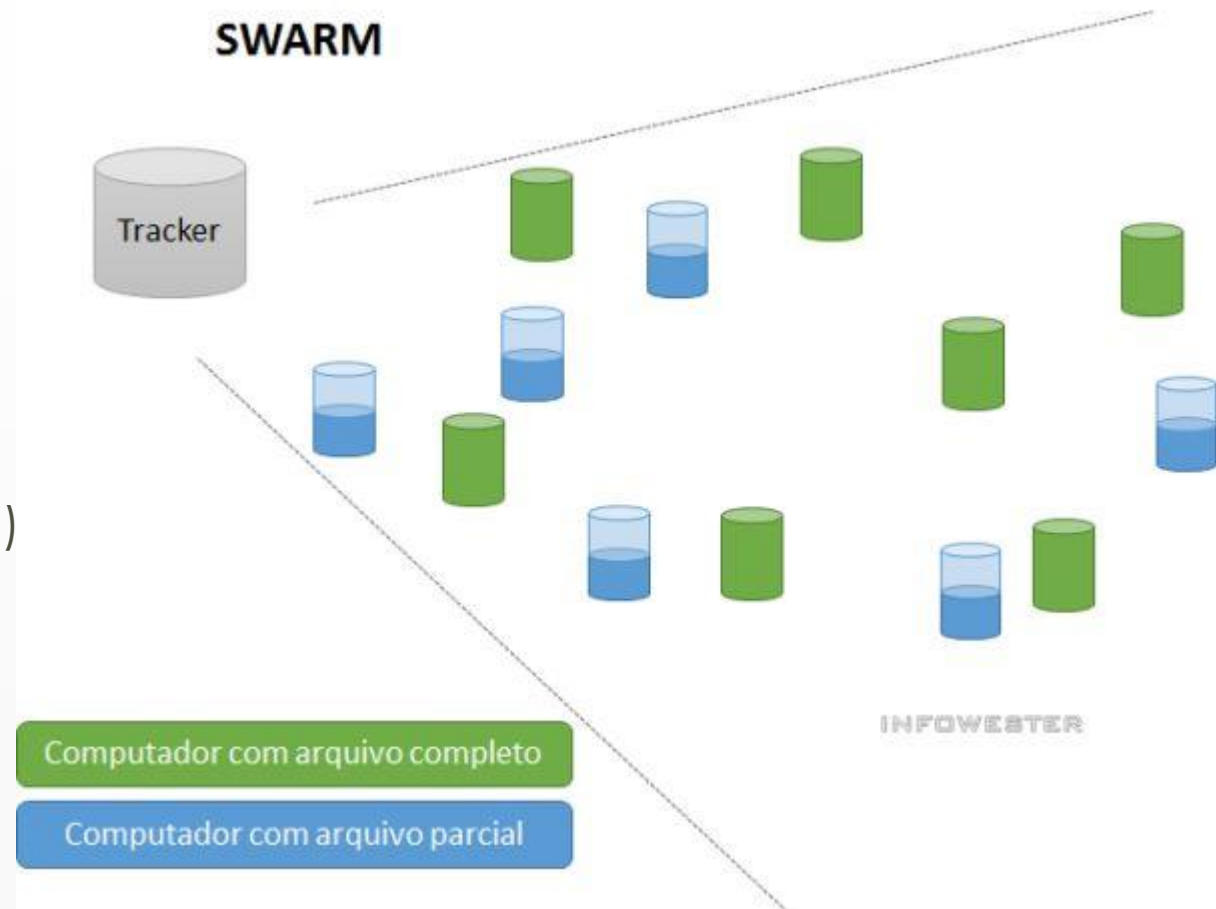
- Arquitetura Peer to Peer
- BitTorrent Live
- BitTorrent Bundle
- BitTorrent Sync
- SoShare



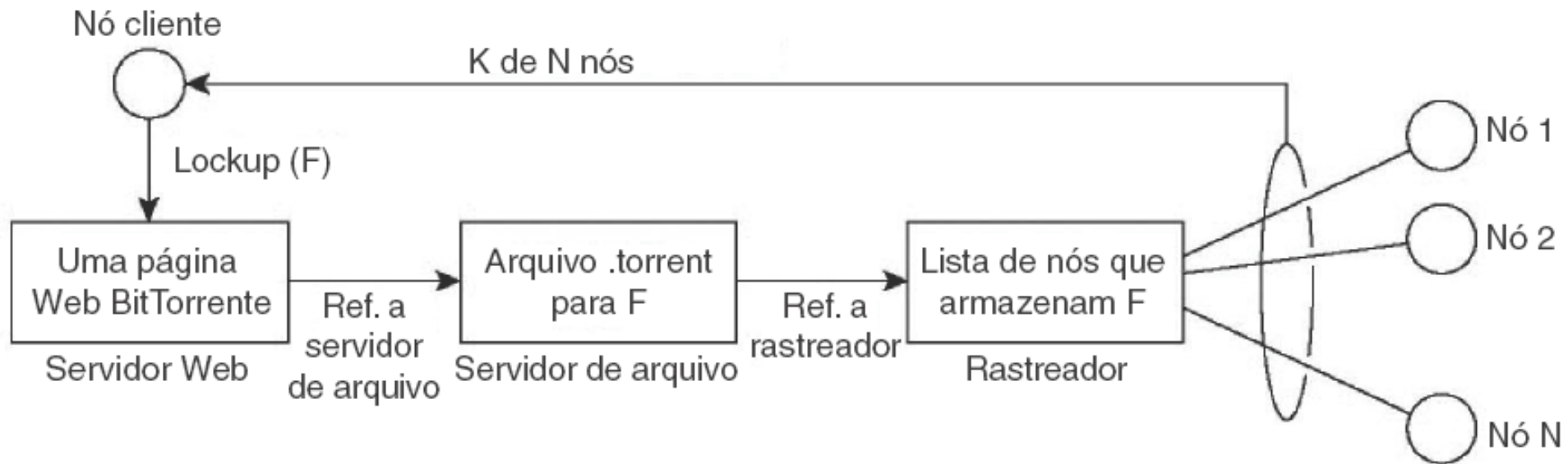


# Estrutura

- Torrent descriptor
- Pieces
- Peers, leechers, seeds, trackers e swarm
- Transmissão não-sequencial
- Rarest First (política de download)
- Tit-for-tat (política de reciprocidade)



# Outra visão



**Figura 2.14** Funcionamento principal do BitTorrent [adaptado com permissão de Pouwelse et al. (2004)].



# Vantagens e desvantagens

- Velocidade mais alta
- Evita congestionamento
- Fácil acesso e atualização por meio dos trackers
- Inconstância na velocidade
- Necessidade de seeders
- Consumo de banda



# Prêmios

- 2004 Wired Rave Award
- 2005 [MIT](#) Technology Review [TR35](#) as one of the top 35 innovators in the world under the age of 35.
- 2005 Time's 100 Most Influential People
- 2006 USENIX STUG Award
- 2010 Internet Evolution 100

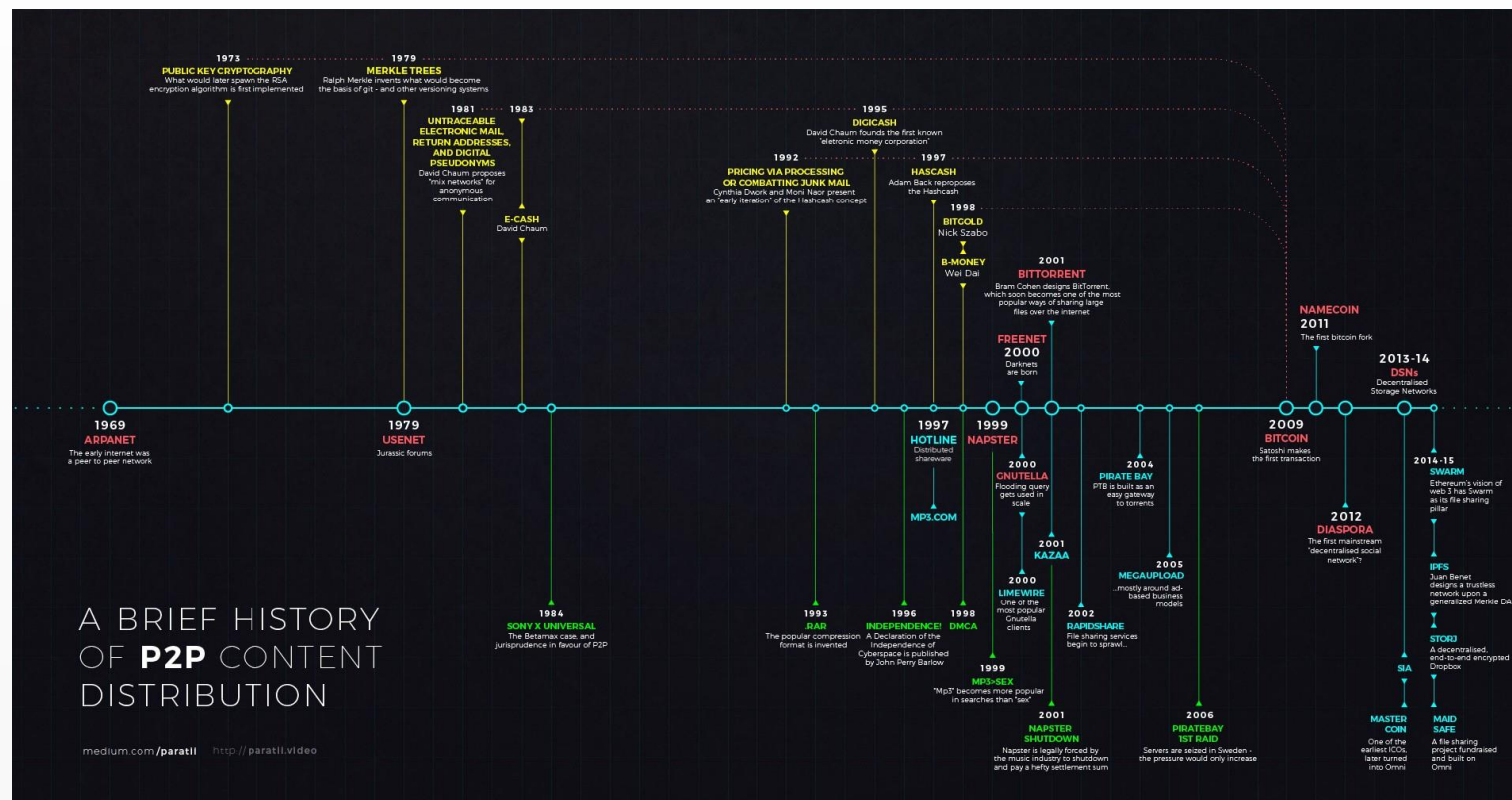


# Questões legais

- protocolo vs websites
- responsabilidade legal
- Traffic Shaping
  - Operadoras de telefonia ou ISPs que bloqueiam mensagens Bit Torrent

## Timeline – Momento Leitura (10 min)

- <https://medium.com/paratii/a-brief-history-of-p2p-content-distribution-in-10-major-steps-6d6733d25122>



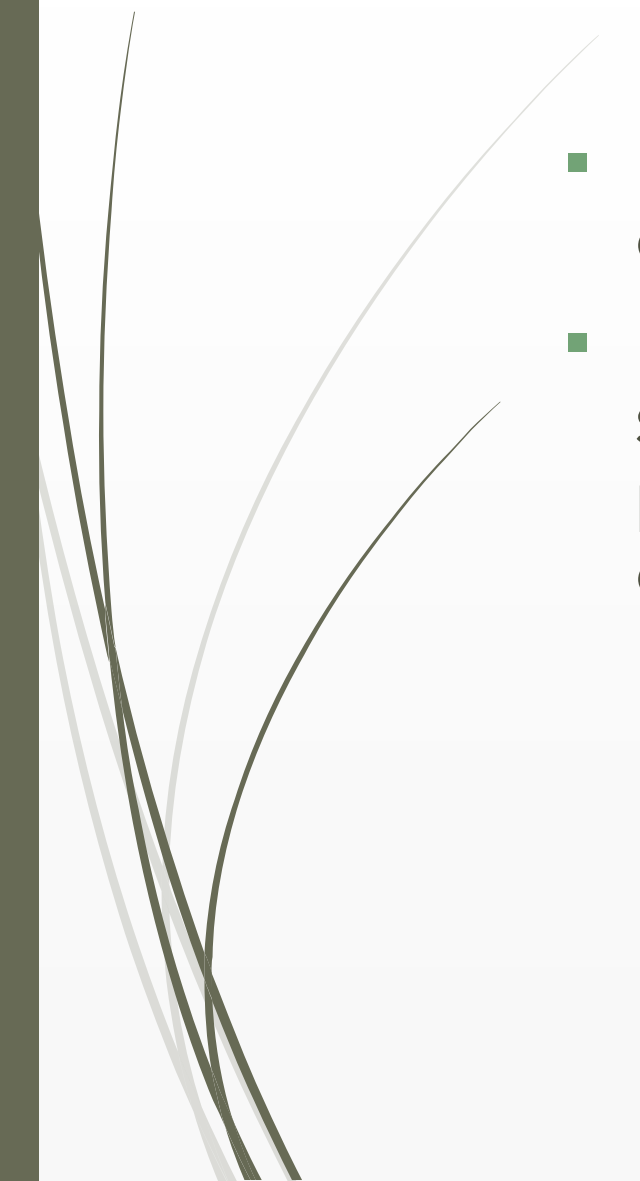


# DHT



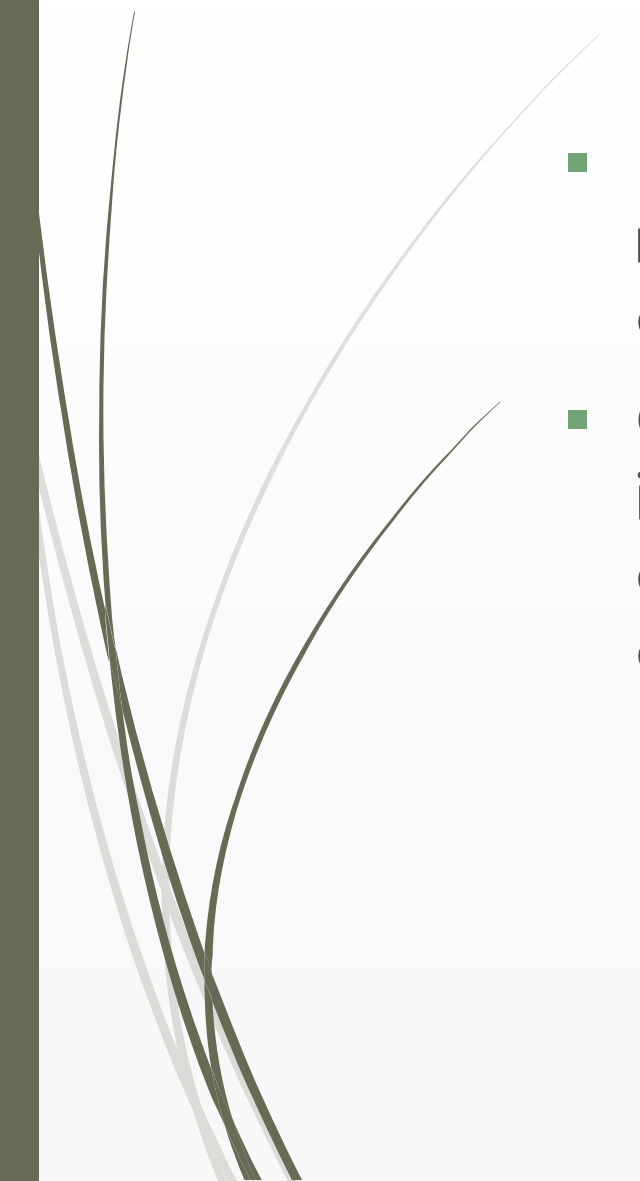


# Redes de Sobreposição Estruturada

- Existem dois tipos de redes de sobreposição: as que são estruturadas e as que não são.
  - Em uma arquitetura peer-to-peer estruturada, a rede de sobreposição é construída com a utilização de um procedimento determinístico. O procedimento mais empregado é o DHT – Distributed Hash Table
- 

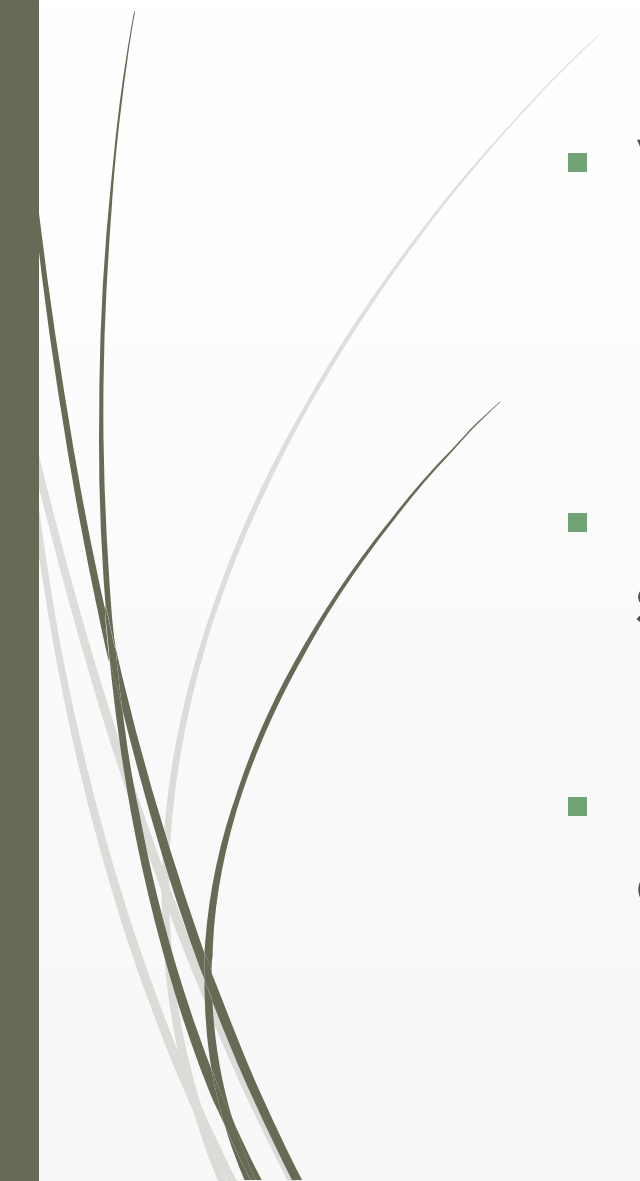


# Redes de Sobreposição Estruturada

- Em um sistema baseado em DHT, os itens de dados recebem uma chave aleatória, como um identificador de 128 ou 160 bits.
  - O ponto crucial de todo sistema baseado em DHT é implementar um esquema eficiente e determinístico que mapeie exclusivamente a chave de um item de dado para o identificador de um nó.
- 

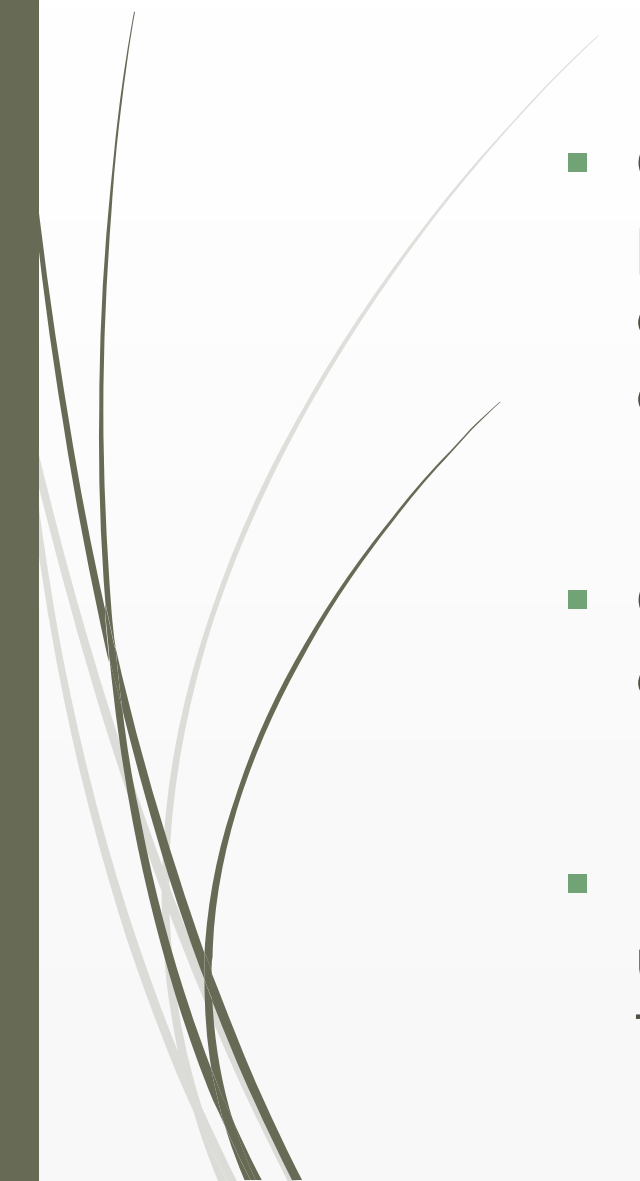


# Distributed Hash Tables (DHTs)

- Vamos considerar como montar uma versão distribuída, P2P, de um banco de dados, que guardará os pares (chave, valor) por milhões.
  - No sistema P2P, cada par só manterá um pequeno subconjunto da totalidade (chave, valor).
  - Permitiremos que qualquer par consulte o banco de dados distribuído com uma chave em particular.
- 

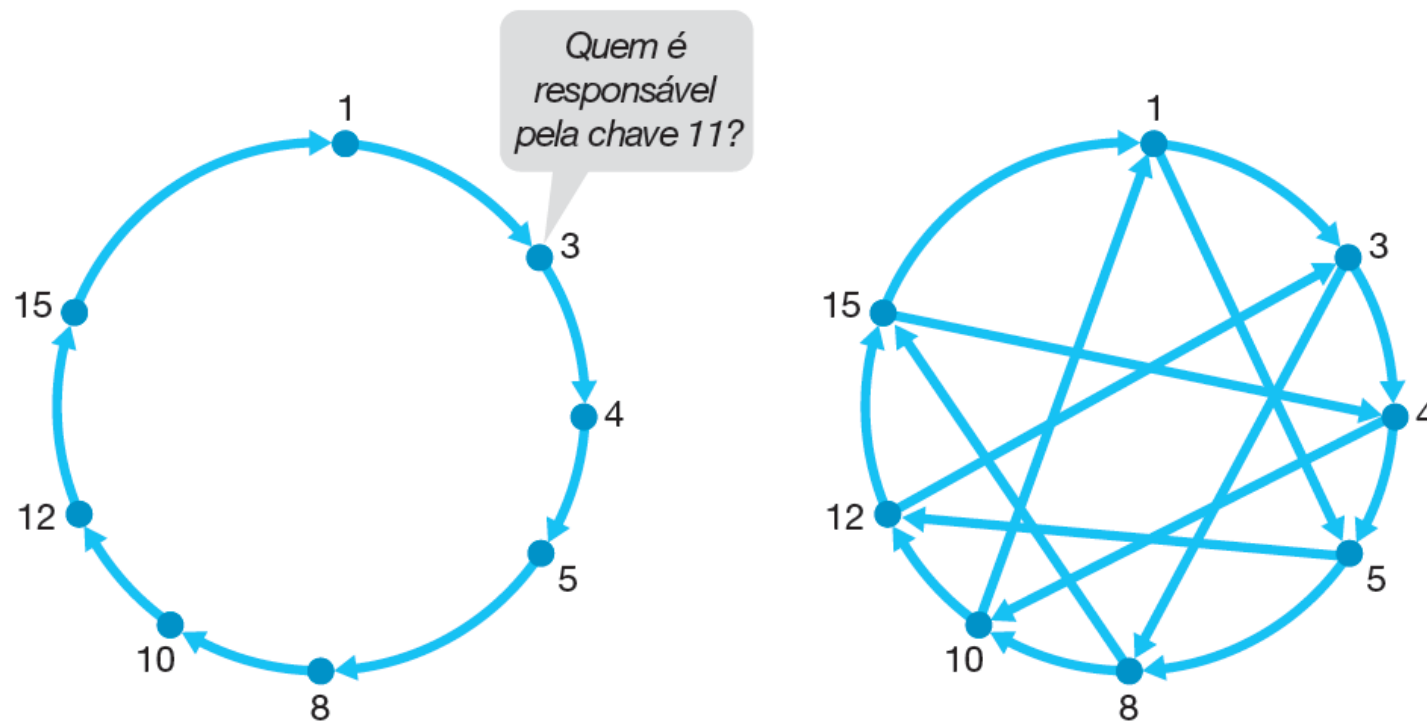


# Distributed Hash Tables (DHTs)

- O banco de dados distribuído, então, localizará os pares que possuem os pares (chave, valor) correspondentes e retornará os pares chave-valor ao consultante.
  - Qualquer par também poderá inserir novos pares chave-valor no banco de dados.
  - Esse banco de dados distribuído é considerado como uma tabela hash distribuída (DHT — Distributed Hash Table).
- 

# Exemplo de DHT - Chord

O DHT circular oferece uma solução bastante elegante para reduzir a quantidade de informação sobreposta que cada nó deve gerenciar.





# DHT - Arquiteturas Descentralizadas

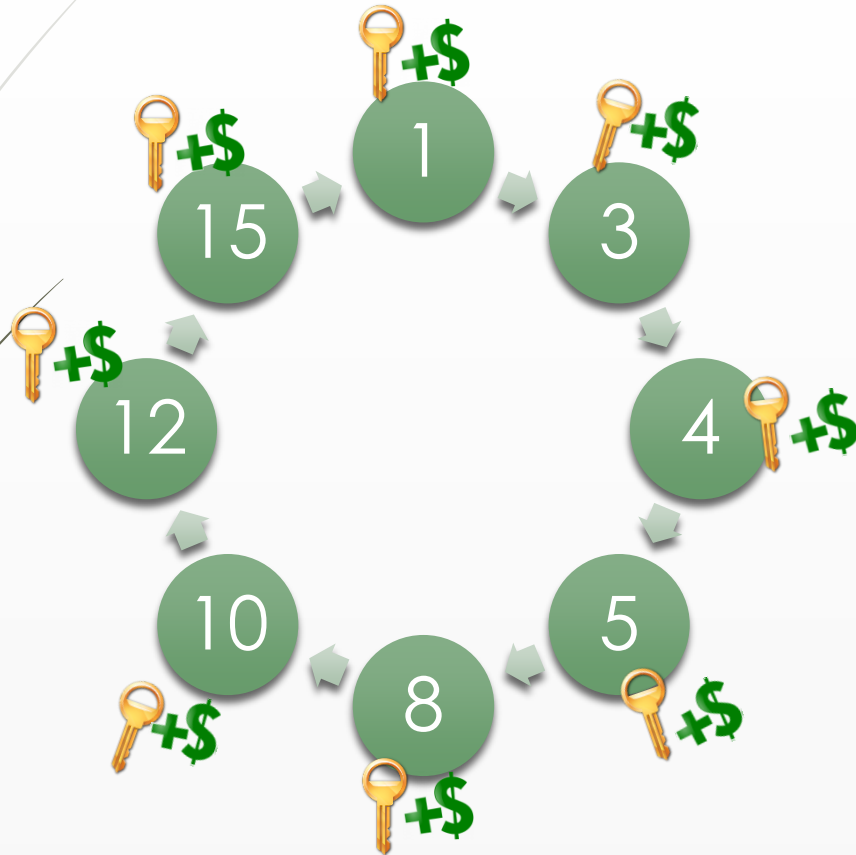
- No sistema Chord os nós estão logicamente organizados em um anel de modo tal que um item de dado com chave  $k$  seja mapeado para o nó que tenha o menor identificador  $id \geq k$ .
- Esse nó é denominado sucessor da chave  $k$  e denotado como  $\text{succ}(k)$ .

# Chave-Valor



- Cada nó mantém apenas uma pequena quantidade de informação sobre os outros nós. (barato manter índices atualizados)
- Cada nó pode pesquisar entradas no índice rapidamente
- Cada nó pode usar o índice ao mesmo tempo, mesmo que outros nós apareçam e desapareçam. (desempenho aumenta com o número de nós).

# Identificador de Pares



- Na disposição circular cada par rastreia apenas o seu sucessor imediato.
- Rastrear=identificar o IP
- Identificador = nome (numero) do par.
- No caso da figura, os identificadores são: 1,3,4,5,8,10,12 e 15.
- Cada par é responsável por conjuntos de chave-valor.
- Chave= endereço; valor= conteúdo

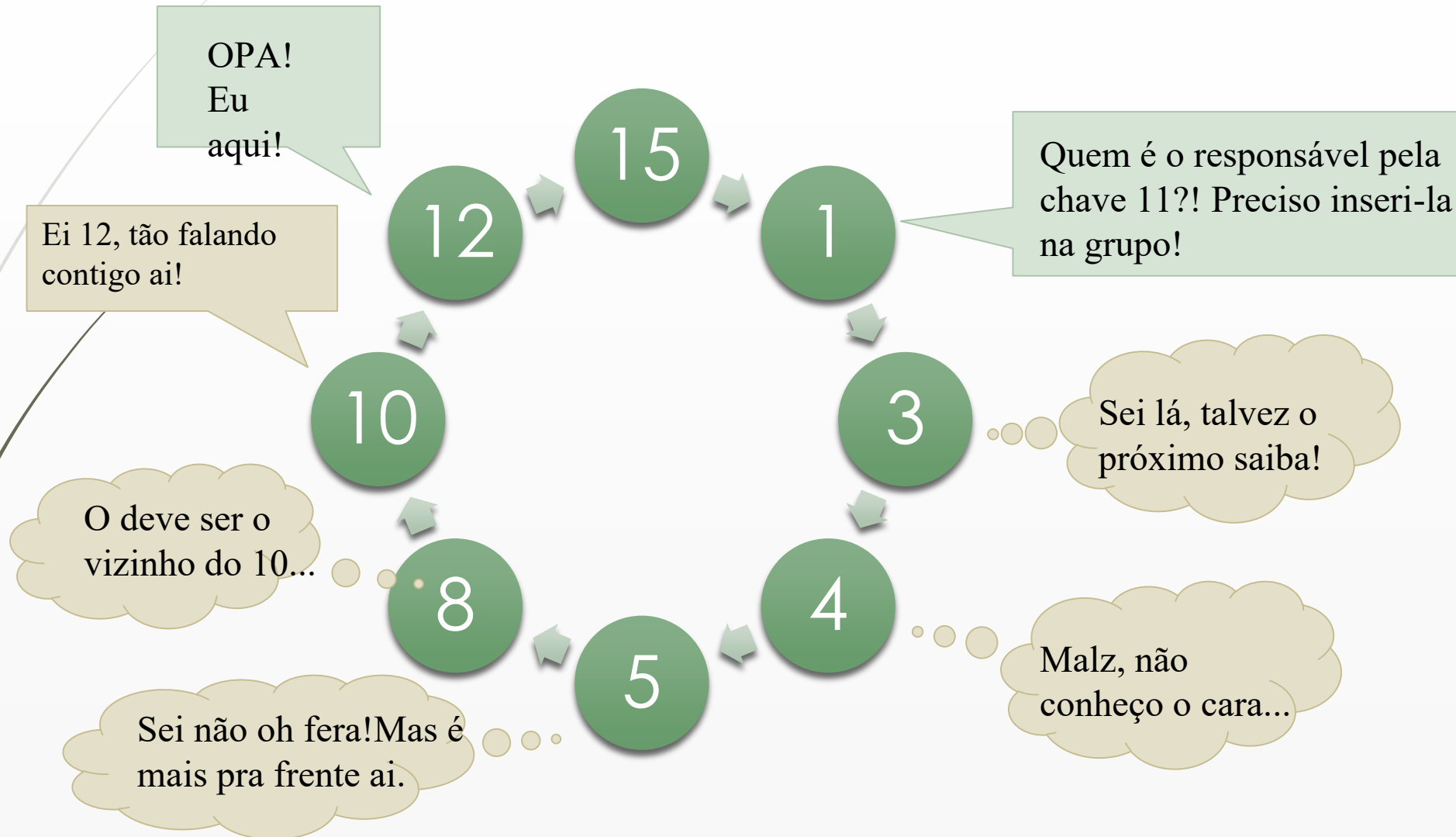


# Identificador de Pares

- Ser responsável por uma chave = ser próximo.
- Ex: par 12 é responsável pela chave 11.
- Quando um par X quer saber quem é o responsável por uma dupla chave-valor, cria uma mensagem perguntando “quem é o responsável pela chave?”. Então envia essa mensagem ao seu sucessor e este a envia até chegar ao par.



# Identificador de Pares



Como gerar esses valores?



# Hash

- A construção de uma DHT se dá de forma similar a uma tabela hash, também conhecida por tabela de dispersão ou tabela de espalhamento.
  - Uma tabela hash é uma estrutura de dados especial que usa uma função hash para associar uma identificação, conhecida por chave, e valores.
  - A função hash é empregada para transformar as chaves em índices de um vetor, no qual os valores são armazenados. O objetivo é, a partir de uma chave, fazer uma busca rápida para obter o valor desejado.



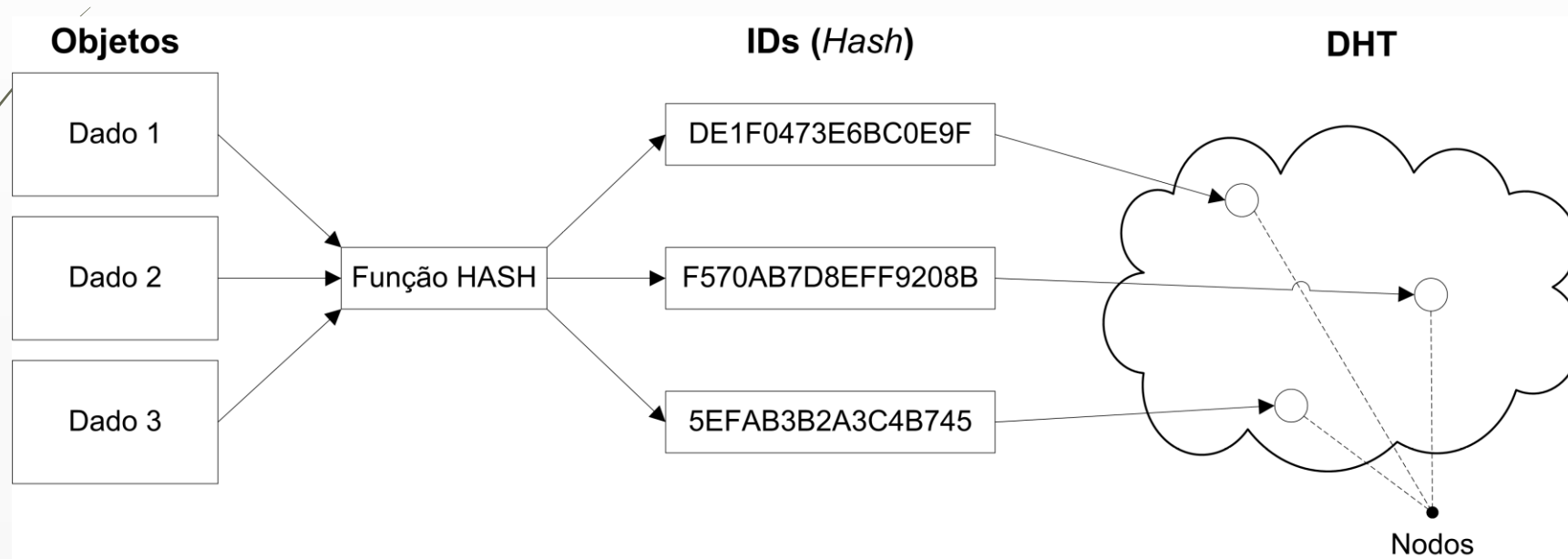
# Hash

- A função hash de uma DHT deve ser elaborada de maneira a atribuir um identificador único para diferentes objetos.
- Quando dois ou mais objetos distintos recebem o mesmo identificador ocorre uma colisão, impossibilitando a diferenciação entre estes objetos.

Para evitar as colisões, adota-se um espaço de identificadores grande o suficiente => a probabilidade de dois objetos receberem a mesma chave torne-se praticamente nula.

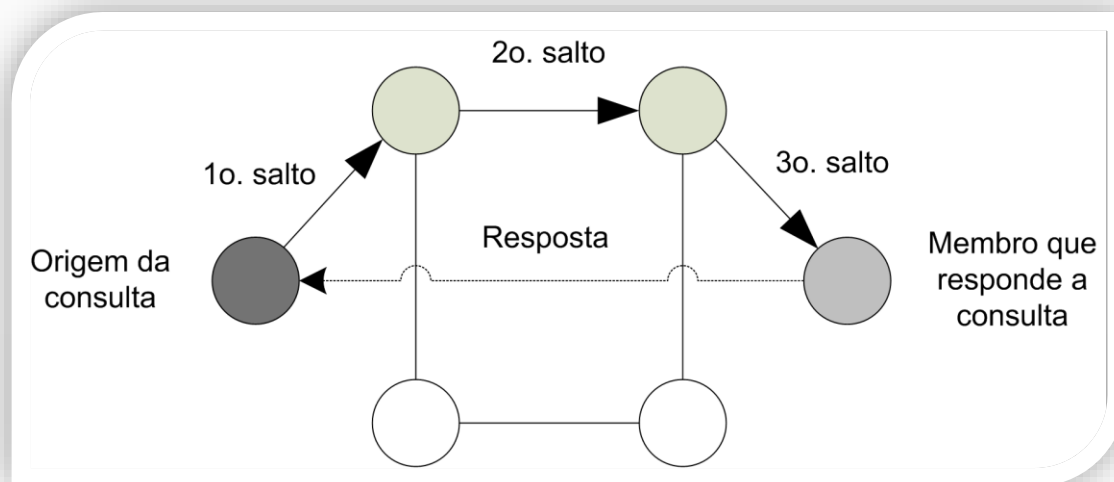
# Hash

- Geração das chaves (IDs) e armazenagem de objetos em uma DHT



# DHTs

- As DHTs que mantêm tabelas de roteamento com N entradas são conhecidas por Single Hop DHTs pois a sua função lookup consegue resolver qualquer consulta usando apenas a tabela local.
- Esta abordagem requer que os eventos de entrada, saída ou falha de qualquer nó do sistema sejam reportados para todos os participantes sem falha



# Mensagem Ping

- Rastreamento de pares próximos
- Numa rede p2p um par pode ir ou vir sem aviso. Existe então a exigência que cada par verifique se seus dois sucessores estão vivos, através do envio da mensagem ping e aguardo de resposta.








# DHT de Múltiplos Saltos

- CAN (Content Addressable Network) –
  - Nodos são mapeados pseudo-randomicamente para um espaço cartesiano virtual de d-dimensões que se juntam nas extremidades (tours)
- Chord – Circular
  - Cada nodo recebe um identificador de m-bits
  - Conhece o caminho para os vizinhos a distância  $2^0, 2^1, 2^2$ , etc.
- Pastry
  - Semelhante ao Chord
  - Conhece o caminho para apenas um vizinho
- Tapestry.
  - Baseada na técnica distribuída de Plaxton e estrutura de dados distribuídos Plaxton Mesh

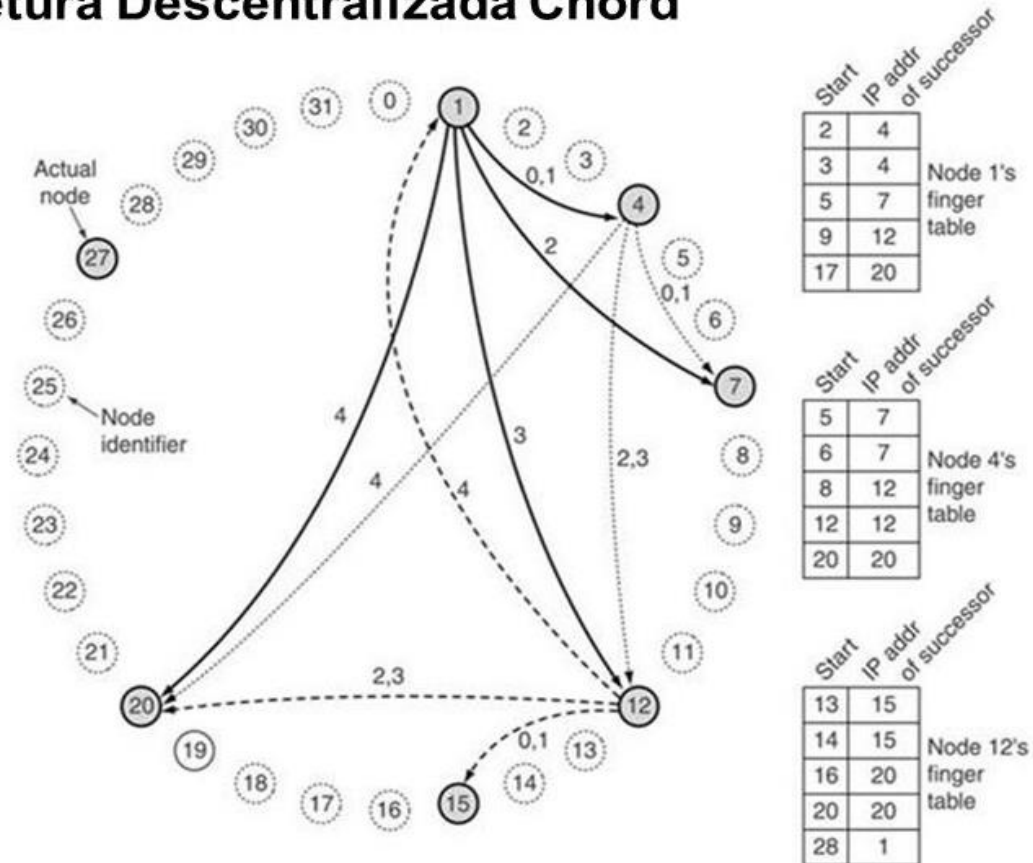


# DHT – Circular ou CHord

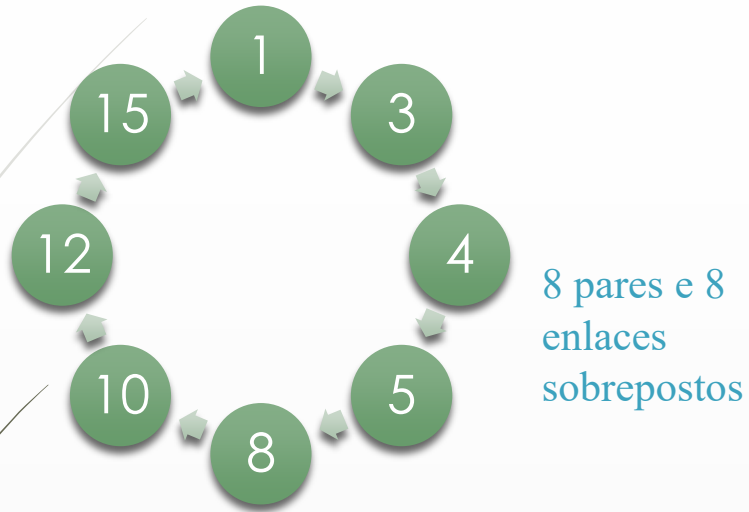
- 
1. Protocolo consiste na utilização de chaves para mapear, localizar e remover nós em uma rede P2P.
  2. Consiste em mapear os peers conectados a rede através de um código hash que identifica cada elemento.
  3. Com esse código, cada peer pode localizar e identificar seus vizinhos através de um emaranhado de peers conectados.
  4. A maneira como é feita consiste em um única operação (lookup) que mapeia o endereço IP com o hash gerado

# Chord

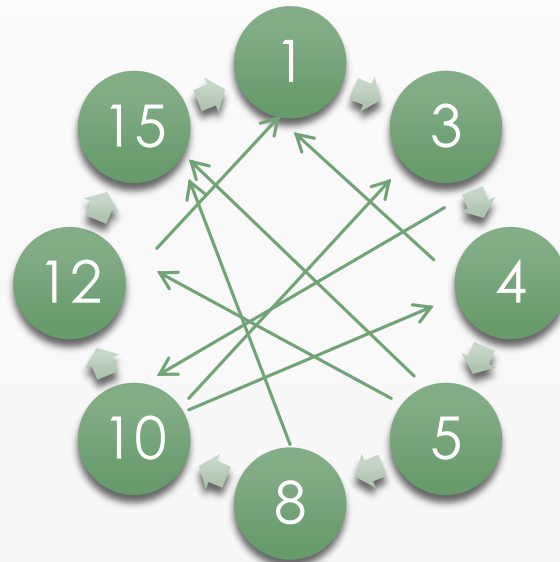
## ▪ Arquitetura Descentralizada Chord



# Atalhos



8 pares e 16  
laços  
sobrepostos



- Como cada par só envia mensagem para seu sucessor, o número de mensagens enviadas até encontrar o par responsável é enorme. Daí para **refinar** essa busca podem ser adicionados atalhos, usados para expedir o roteamento das mensagens de solicitação.
- Isso reduz consideravelmente o número de mensagens enviadas.

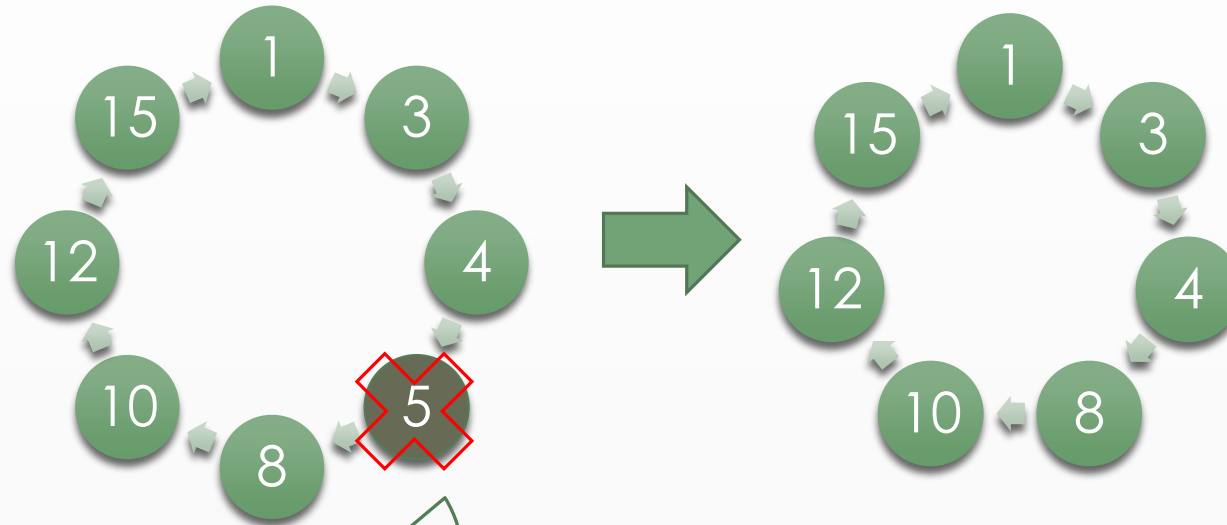


# Exercício

- Proponha **soluções** para
  - Inserir um novo nó na rede P2P
  - Remover um novo nó na rede P2P
  - Substituir um nó na rede P2P

# Substituição de sucessor

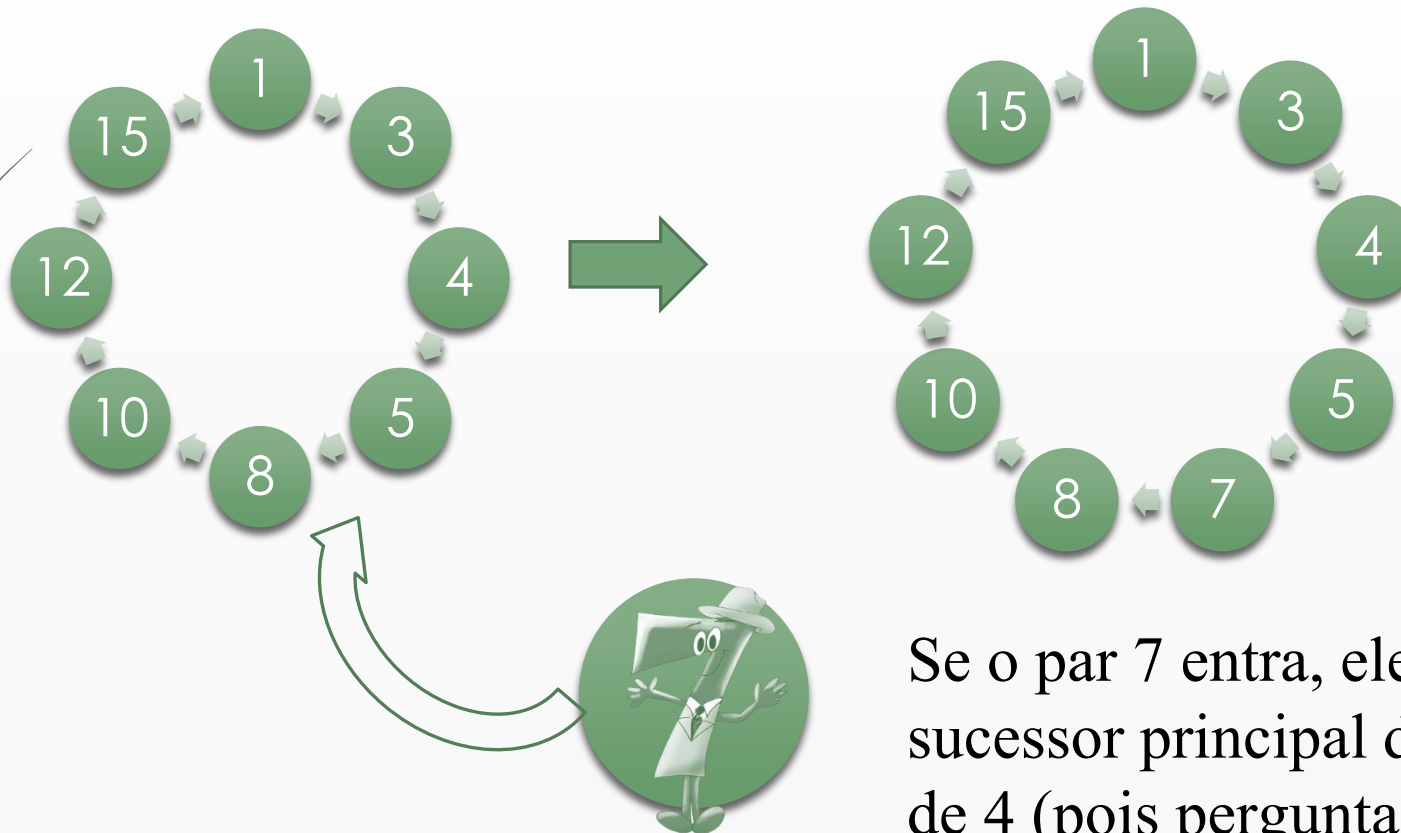
- Quando sai um par



Por exemplo, se 5 sai, 4 e 3 sabem, pois ele não responde mais as mensagens ping. Assim eles atualizam-se . O par 4 substitui seu sucessores para 8 e 10, enquanto que 3 muda seu sucessores para 4 e 8.

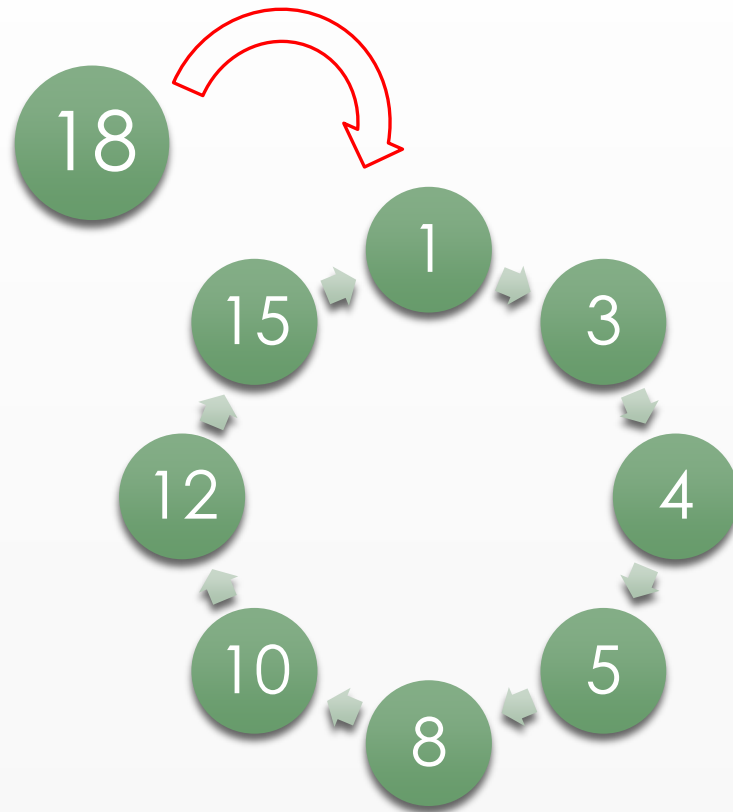
# Substituição do sucessor

- Quando entra um novo par



Se o par 7 entra, ele passa a ser o sucessor principal de 5 e o secundário de 4 (pois pergunta ao 5 o IP do sucessor dele).

# Substituição do sucessor



- Se um par de numero maior quiser entrar e não tem sucessor dele?
- Ele faz do 1 o seu sucessor para não quebrar o ciclo.





# Aplicações

- BitTorrent
  - EDonkey
  - Dynamo (Amazon)
- 



# Eleições em sistemas de grande escala

- Há situações em que é necessário trabalhar com redes maiores e é necessário eleger maior quantidade de pares, ex.: Superpares em P2P
- Requisitos a serem cumpridos por superpar:
  1. Nós normais devem ter baixa latência de acesso com superpares;
  2. Superpares devem estar uniformemente distribuídos pela rede de sobreposição;
  3. Deve haver uma porção predefinida de superpares em relação ao número total de nós na rede de sobreposição;
  4. Cada superpar não deve precisar atender mais do que um número fixo de nós normais;



# Eleições em sistemas de grande escala

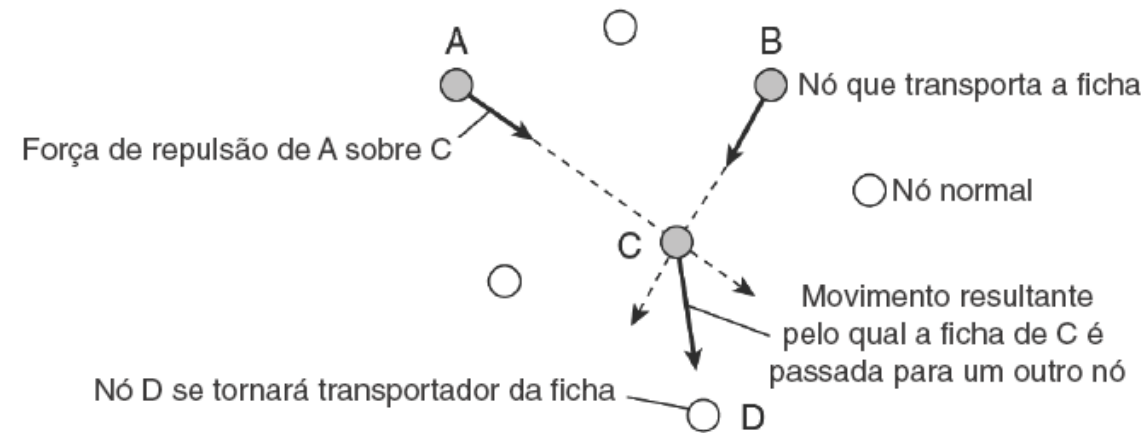
## Superpares com $k$ de $m$ bits como id

- Uma solução é dada quando se usa  $m$  bits de identificador, separar os  $k$  bits da extrema esquerda para identificar superpares;
  - Ex.:  $\log_2(N)$  Superpares,  $m=8, k=3$ .
  - $p \text{ AND } 11100000 = \text{Superpar}$ .
- Problema: não garante posicionamento geométrico para organizar os superpares *uniformemente* pela rede

# Eleições em sistemas de grande escala

## Eleição de pares por fichas repulsoras

- N fichas distribuídas aleatoriamente entre os nós;
- Nenhum nó pode ter mais de uma ficha;
- Fichas possuem uma força de repulsão;
- Um nó que mantiver a ficha por determinado tempo é eleito superpar.



**Figura 6.22** Movimentação de fichas em um espaço bidimensional que utiliza forças de repulsão.