Redes de Computadores

Parte 07 – camada de aplicação – P2P

Prof. Kleber Vieira Cardoso

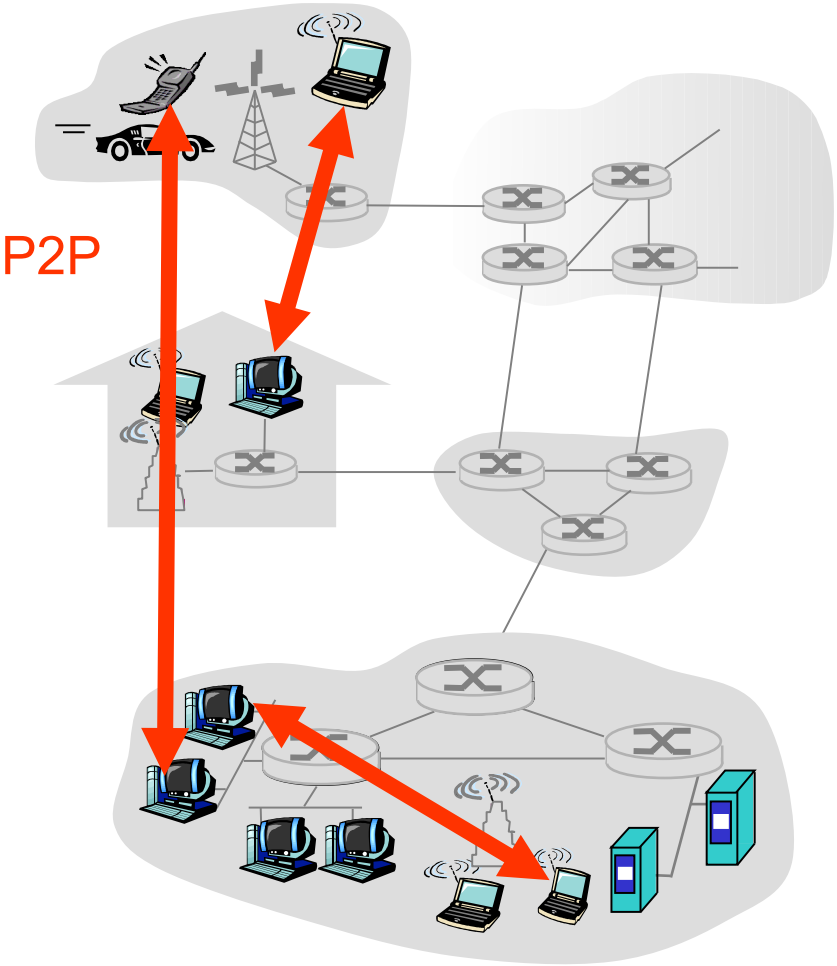
INSTITUTO DE
INFORMÁTICA
UFG

Tópicos

- Conceitos
- Aplicações
  - Distribuição de arquivos: BitTorrent
  - Banco de dados: tabela *hash* distribuída (*Distributed Hash Table* – DHT)
  - VoIP: Skype

Arquitetura P2P

- Não há servidor sempre ligado
- Sistemas finais arbitrários se comunicam diretamente
- *Peers* (equipamentos) estão conectados intermitentemente e mudam de endereços IP

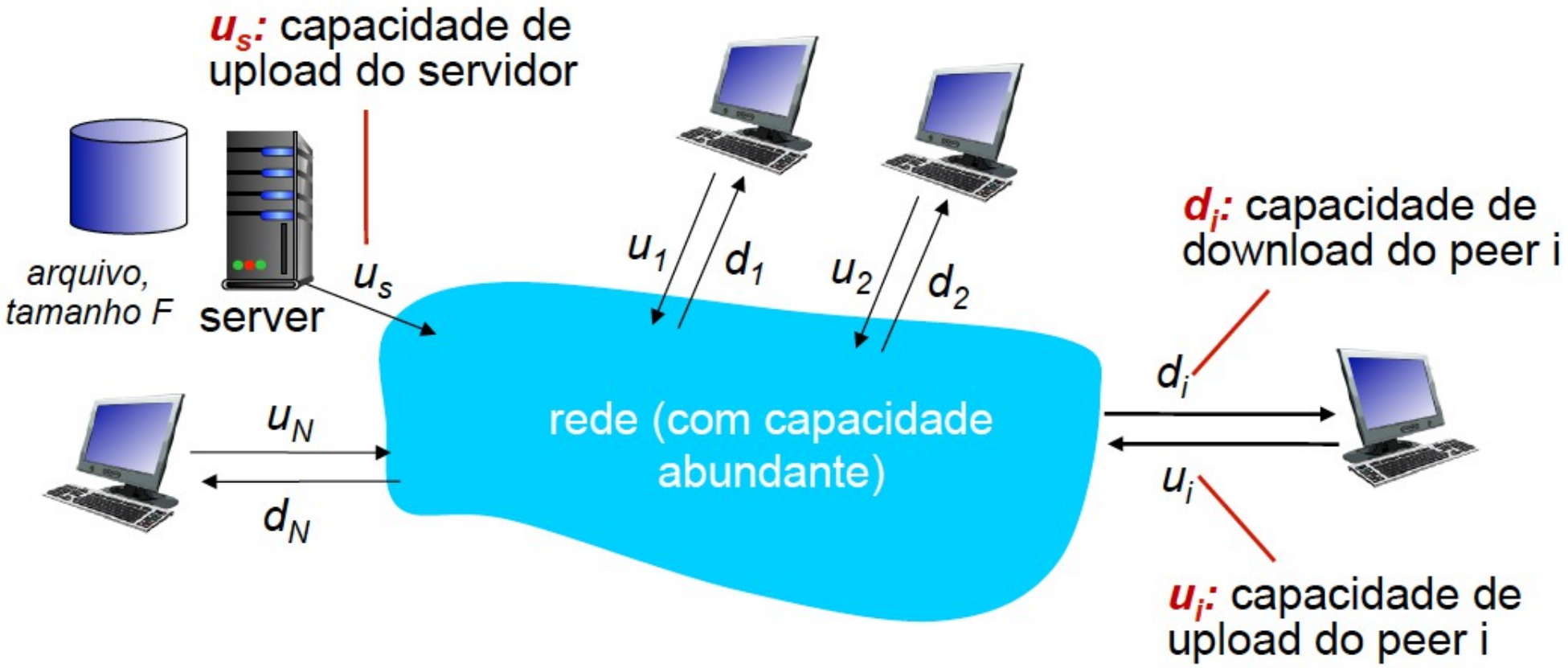


- Exemplos
  - Compartilhamento de arquivo (BitTorrent)
  - Vídeo de fluxo contínuo (Tribler, Xunlei Kankan, PPS.tv)
  - VoIP (Skype)

Distribuição de arquivo: cliente-servidor *versus* P2P

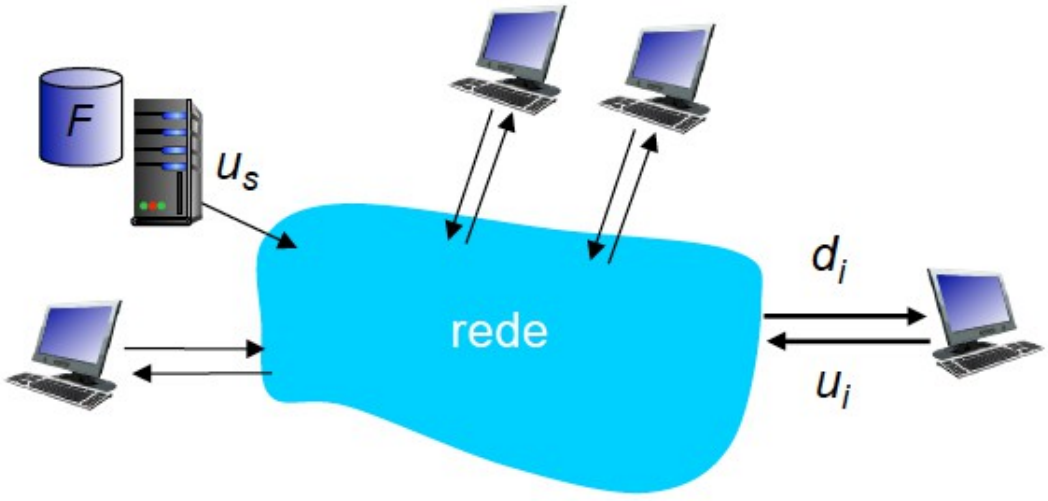
Pergunta: Quanto tempo para distribuir arquivo (de tamanho F) a partir de um servidor para *N peers* (pessoas, equipamentos)?

- Capacidade de upload/download de um *peer* é limitada



Tempo de distribuição de arquivo: cliente-servidor

- Servidor envia N cópias sequencialmente:
  - Tempo = NF/u_s
- Cliente i leva um tempo F/d_i para o *download*



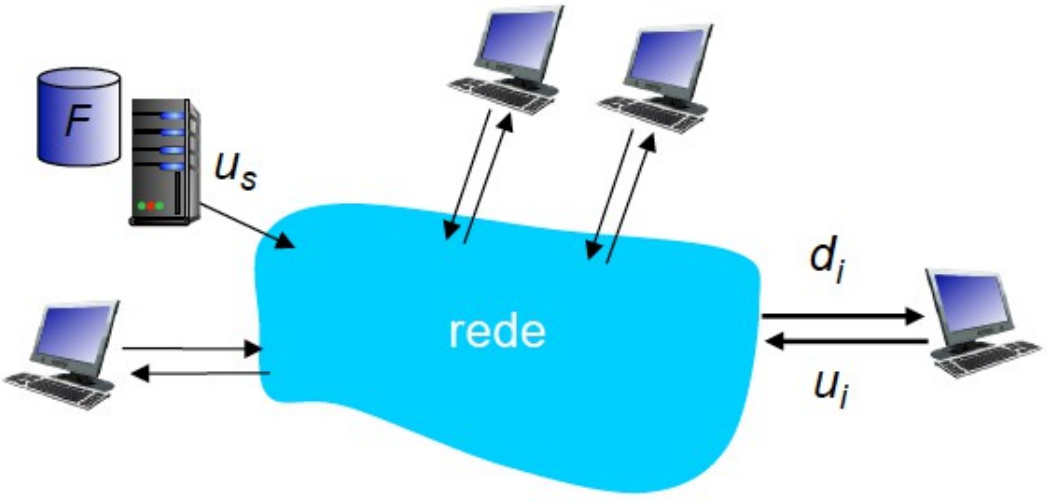
tempo para distribuir *F* a *N* clientes usando técnica cliente/servidor

$$= D_{cs} = \max \left\{ NF / u_s , F / min(d_i) \right\}$$

aumenta linearmente em função de N

Tempo de distribuição de arquivo: P2P

- Servidor deve enviar uma cópia: tempo = F/u_s
- Cliente i leva tempo F/d_i para o *download*
- Novamente, NF bits devem ser enviados (agregado)
  - Porém, nova taxa de *upload* máxima: $u_s + \Sigma u_i$



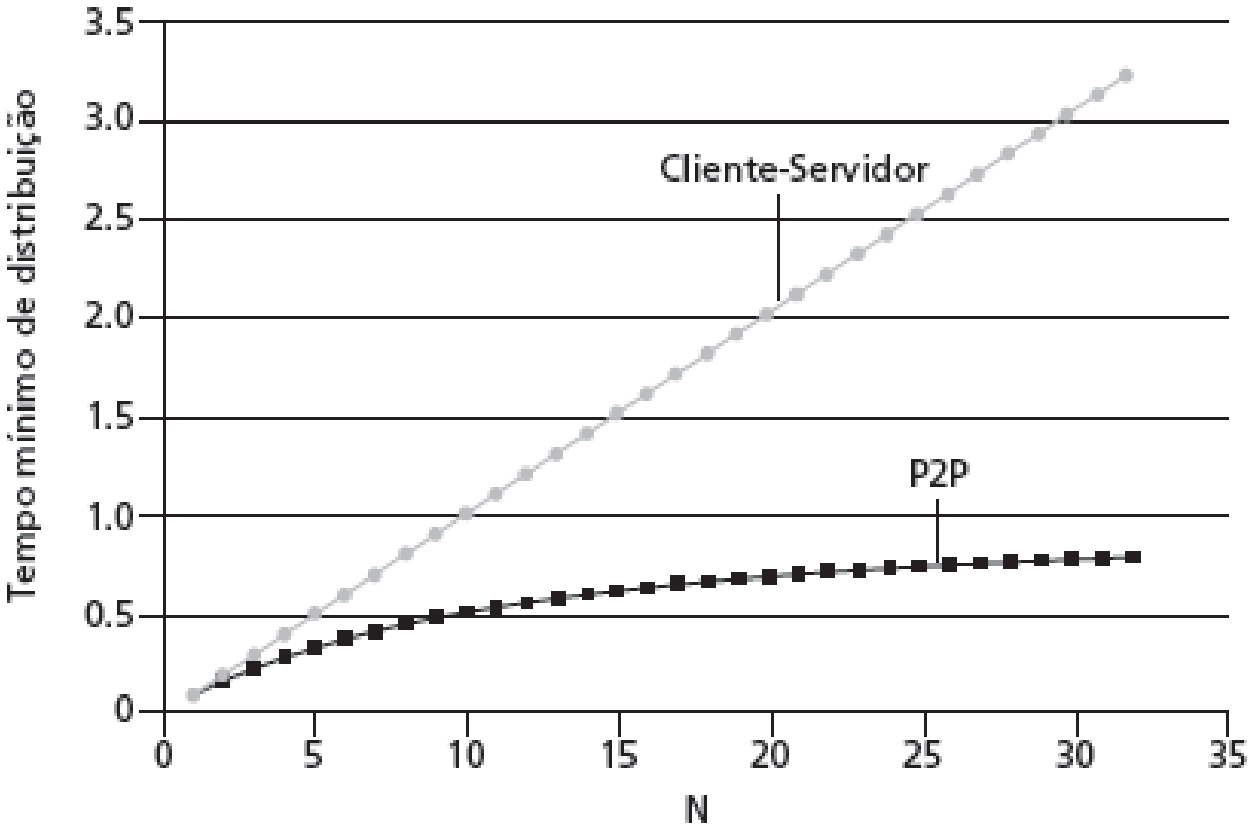
$$D_{P2P} = \max \left\{ F / u_s , F / min\ (d_i) , NF / (u_s + \sum u_i) \right\}$$

ainda aumenta linearmente em função de N

...mas isso também aumenta linearmente à medida que *peers* trazem sua capacidade de *upload*

Cliente-servidor *versus* P2P: exemplo

Taxa de *upload* cliente = u, F/u = 1 hora, $u_s = 10u$, $d_{min} \geq u_s$

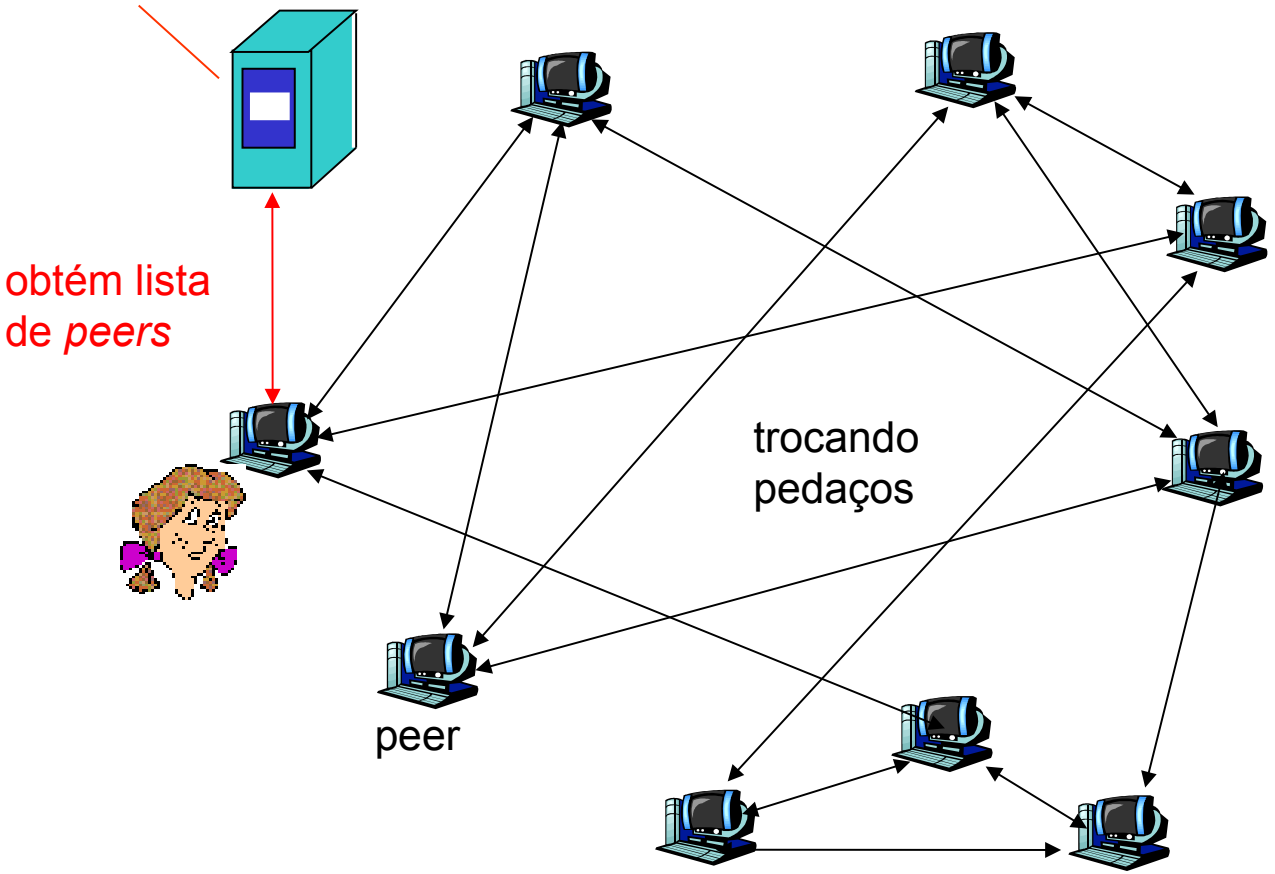


Distribuição de arquivos: BitTorrent

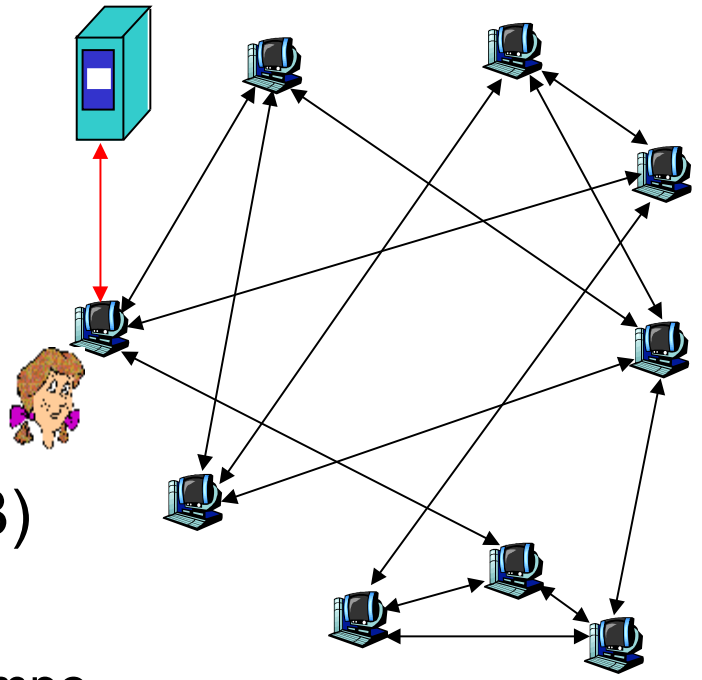
Distribuição de arquivos P2P

Tracker: registra e informa *peers* que participam do *torrent*

Torrent: grupo de *peers* trocando pedaços de um arquivo



BitTorrent: como funciona



- Arquivo dividido em *pedaços* (e.g., 256 KB)
- *Peer* que se une à *Torrent*:
  - não tem pedaços, mas os acumulará com o tempo
  - registra com o *tracker* para obter lista dos *peers*, conecta a um subconjunto de *peers* (“vizinhos”)
- Enquanto faz o *download*, *peer* faz *upload* de pedaços para outros *peers*
- *Peers* podem entrar e sair
- Quando o *peer* obtém todo o arquivo, ele pode (egoisticamente) sair ou permanecer (altruisticamente)

BitTorrent

Obtendo Pedaços

- Num determinado instante, *peers* distintos possuem diferentes subconjuntos dos pedaços do arquivo
- Periodicamente, um *peer* X pede a cada vizinho a lista de pedaços que eles possuem
- X envia pedidos para os pedaços que ainda não tem
  - Primeiro os mais raros

Enviando pedaços: toma lá, dá cá!

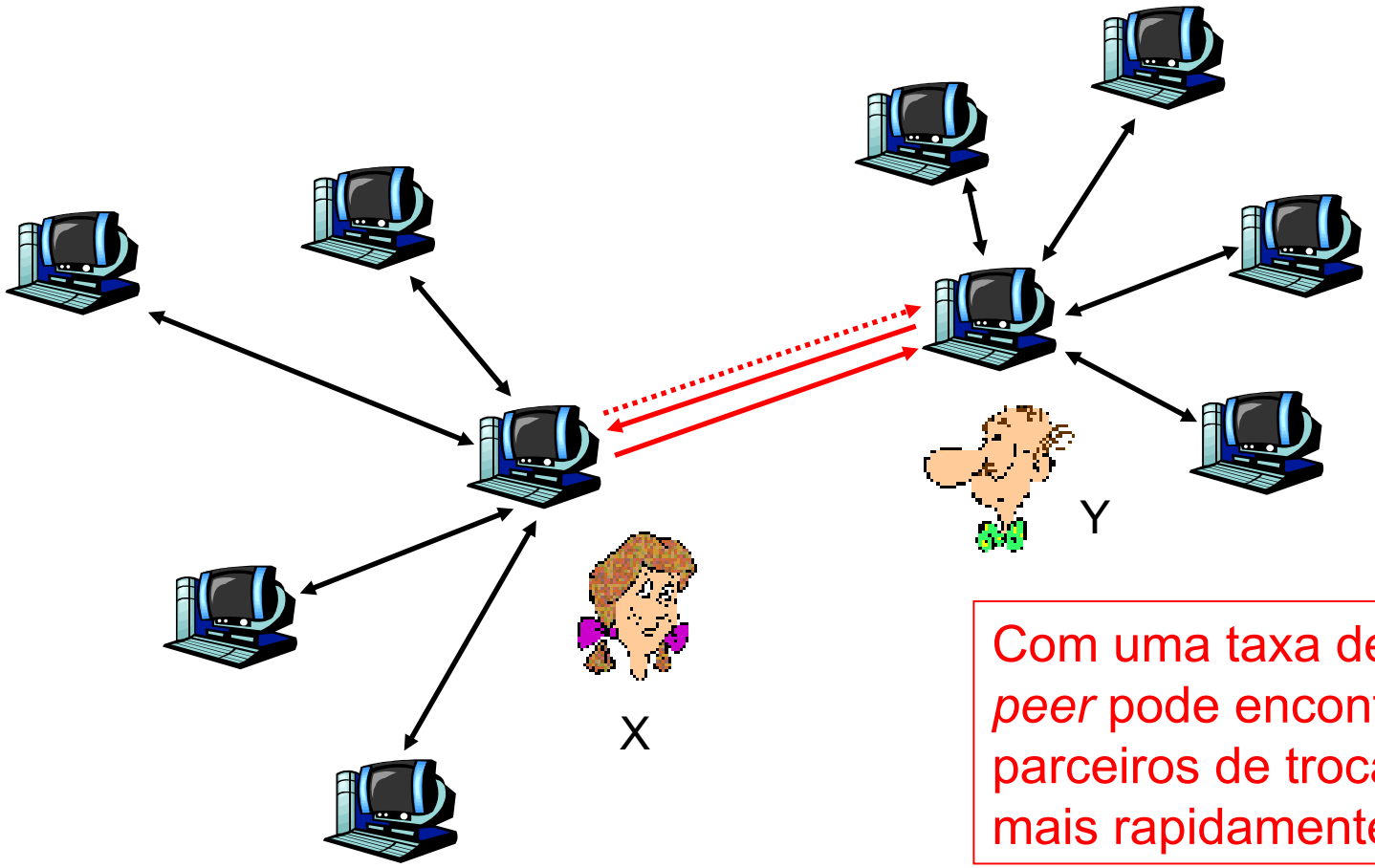
- X envia pedaços para 4 vizinhos que estejam lhe enviando pedaços *na taxa mais elevada*
  - Reavalia os 4 a cada 10 segs
- A cada 30 segs: seleciona aleatoriamente outro *peer*, começa a enviar pedaços
  - O *peer* recém escolhido pode se unir aos 4 mais
  - "*Optimistically unchoke*"

BitTorrent: toma lá, dá cá!

(1) X "*optimistically unchokes*" Y

(2) X se torna um dos quatro melhores provedores de Y;
Y age da mesma forma

(3) Y se torna um dos quatro melhores provedores de X



Com uma taxa de *upload* mais alta, *peer* pode encontrar melhores parceiros de troca e obter o arquivo mais rapidamente!

Distributed Hash Table (DHT)

- DHT = banco de dados P2P distribuído
  - Banco de dados tem duplas (chave, valor), exemplos:
    - chave: número CPF; valor: nome do indivíduo
    - chave: conteúdo; valor: endereço IP
  - *Peers* consultam BD com chave
    - BD retorna valor que combina com a chave
  - *Peers* também podem inserir duplas (chave, valor)

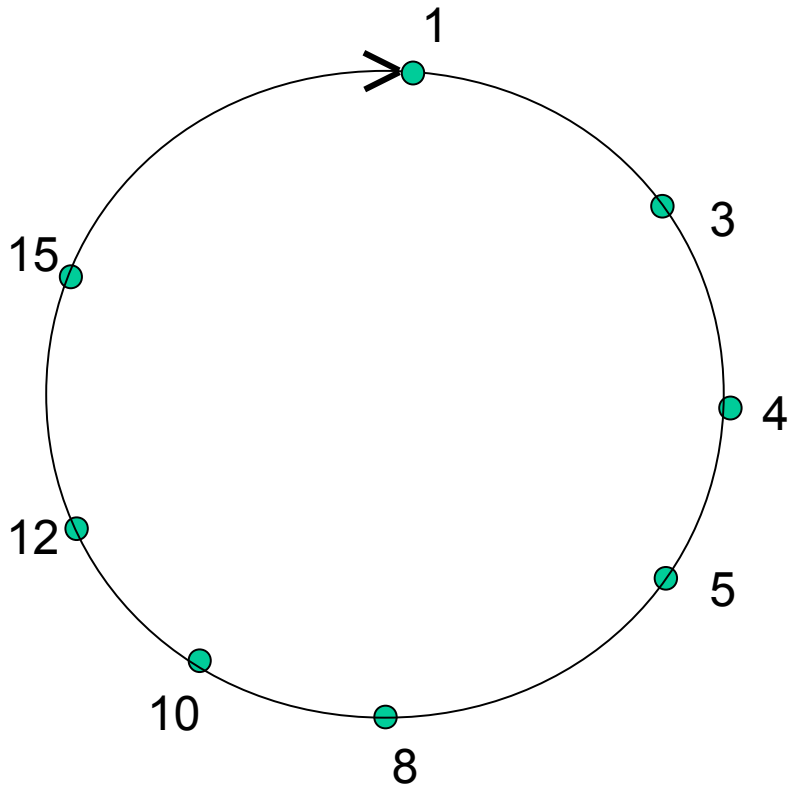
Identificadores DHT

- Atribuem identificador inteiro a cada *peer* no intervalo $[0,2^n - 1]$
  - Cada identificador pode ser representado por n bits
- Exigem que cada chave seja um inteiro no mesmo intervalo
- Para obter chaves inteiras, é feito *hash* da chave original
  - Exemplo: chave = h(“Aquarela do Brasil”)
  - Daí o nome: tabela *hash* distribuída

Como atribuir chaves aos *peers*?

- Questão central:
  - atribuir duplas (chave, valor) aos *peers*
- Regra: atribuir chave ao *peer* que tem o ID mais próximo
  - Convenção: mais próximo é o sucessor imediato da chave
- E.g., n = 4; *peers*: 1,3,4,5,8,10,12,14
  - chave = 13, então *peer* sucessor = 14
  - chave = 15, então *peer* sucessor = 1

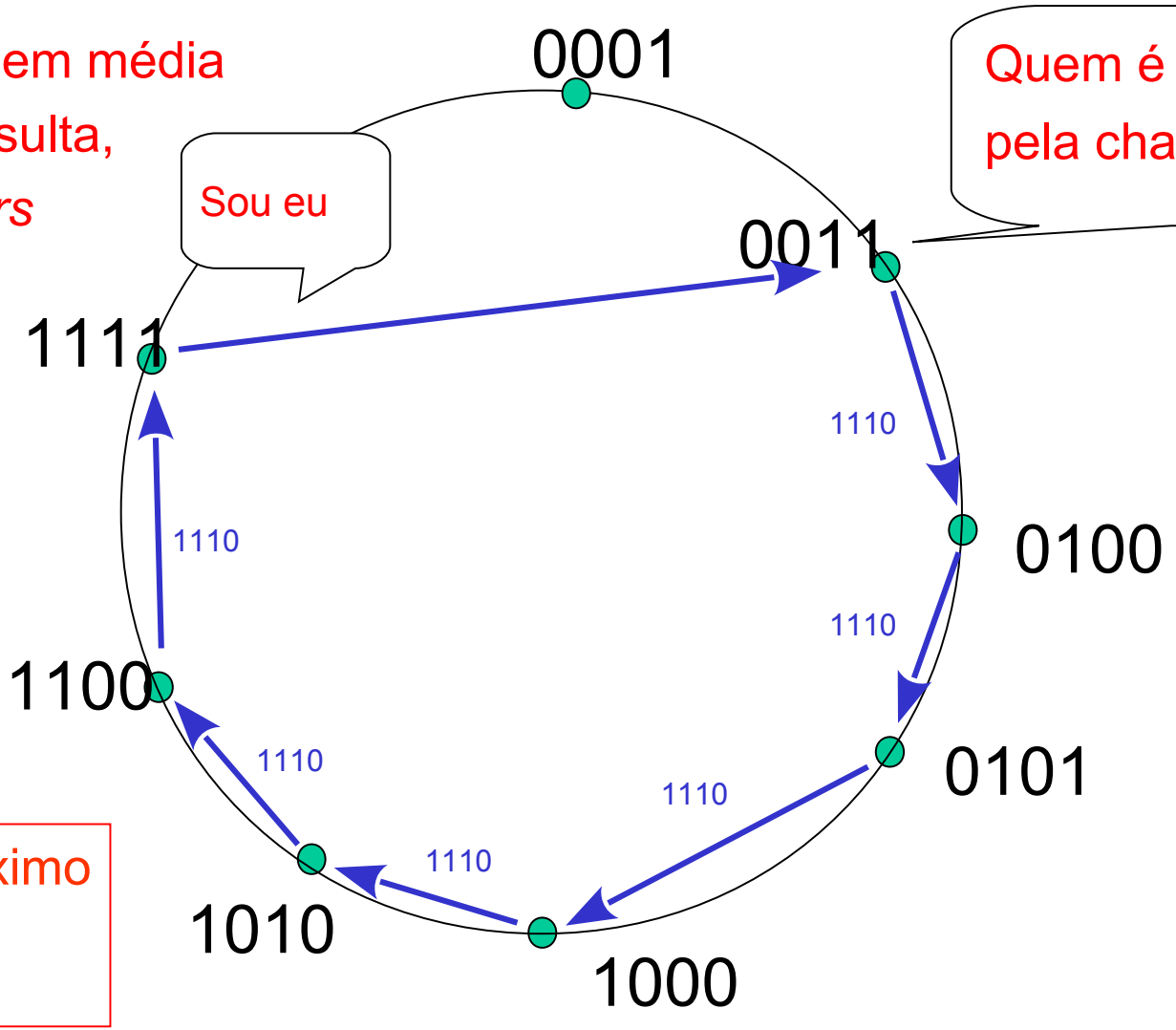
DHT circular



- Cada *peer só* conhece sucessor e predecessor imediato
  - É criada uma rede sobreposta (*overlay network*)

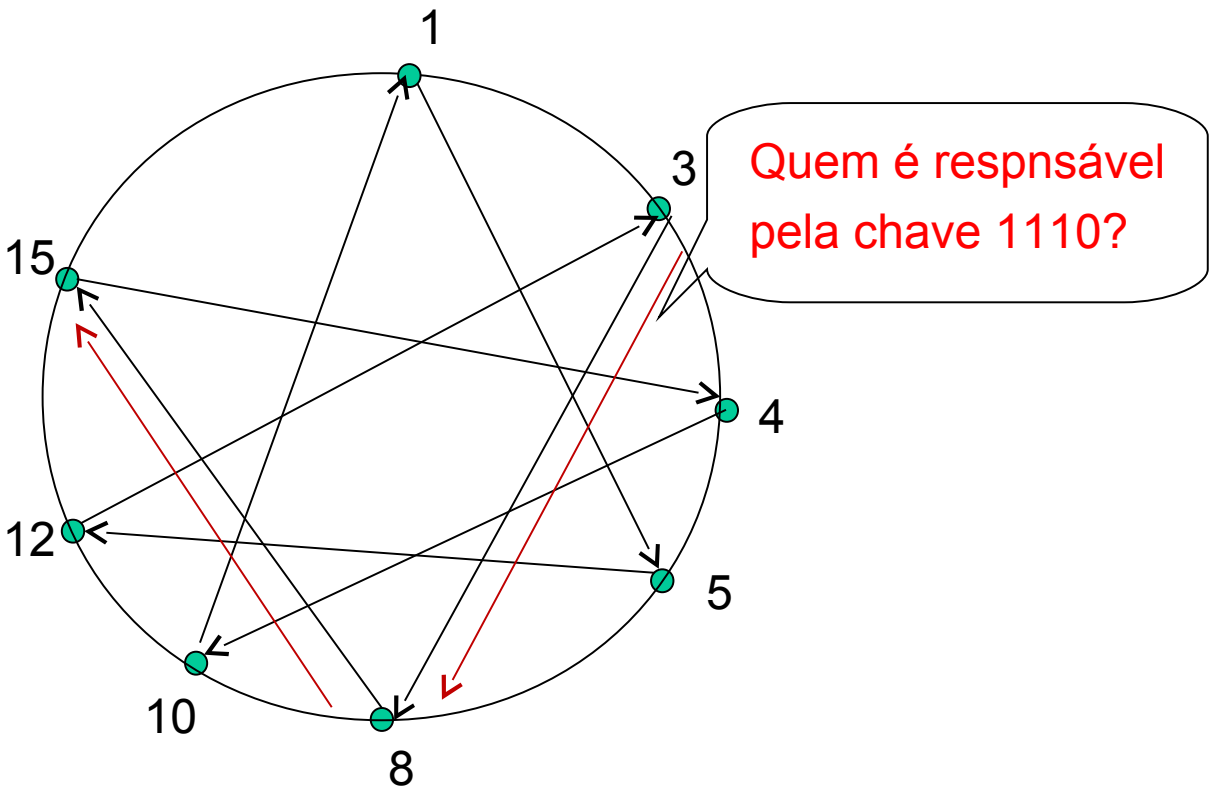
DHT circular: exemplo

O(N) mensagens em média para resolver consulta, quando há N *peers*



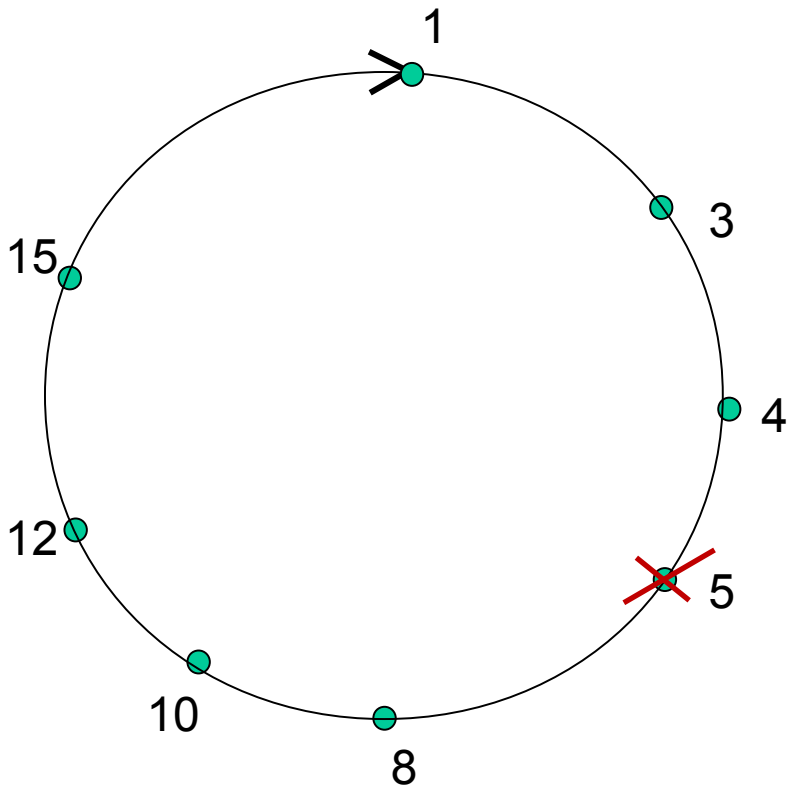
Define mais próximo como sucessor mais próximo

DHT circular com atalhos



- Cada *peer* registra endereços IP do predecessor, sucessor e atalhos
  - No exemplo, o número de mensagens foi reduzido de 6 para 2
- DHT pode ser projetado para obter número de vizinhos por *peer* e número de mensagens por consulta igual a O(log N)

Peer Churn (“Agitação” dos *Peers*)

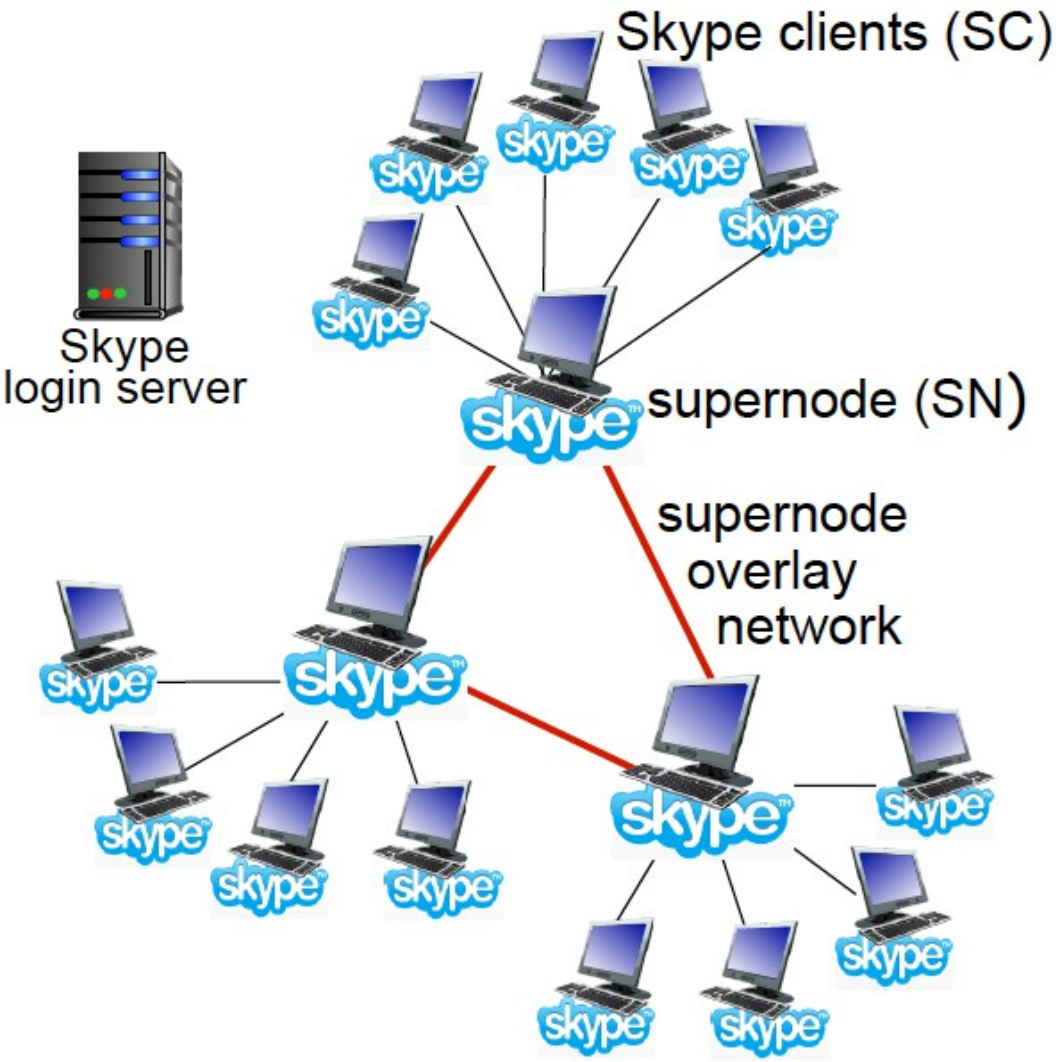


- Para tratar a *peer churn*, é preciso que cada *peer* conheça (pelo menos) o endereço IP de seus dois sucessores
- Cada *peer* periodicamente envia sondas (*ping*) aos seus dois sucessores para ver se eles ainda estão vivos

- Exemplo:
  - *Peer* 5 sai abruptamente
  - *Peer* 4 detecta; torna 8 seu sucessor imediato; pergunta a 8 quem é seu sucessor imediato; torna o sucessor imediato de 8 seu segundo sucessor
  - E se o *peer* 13 quiser se juntar (conhecendo apenas o *peer* 1)?

VoIP P2P com Skype

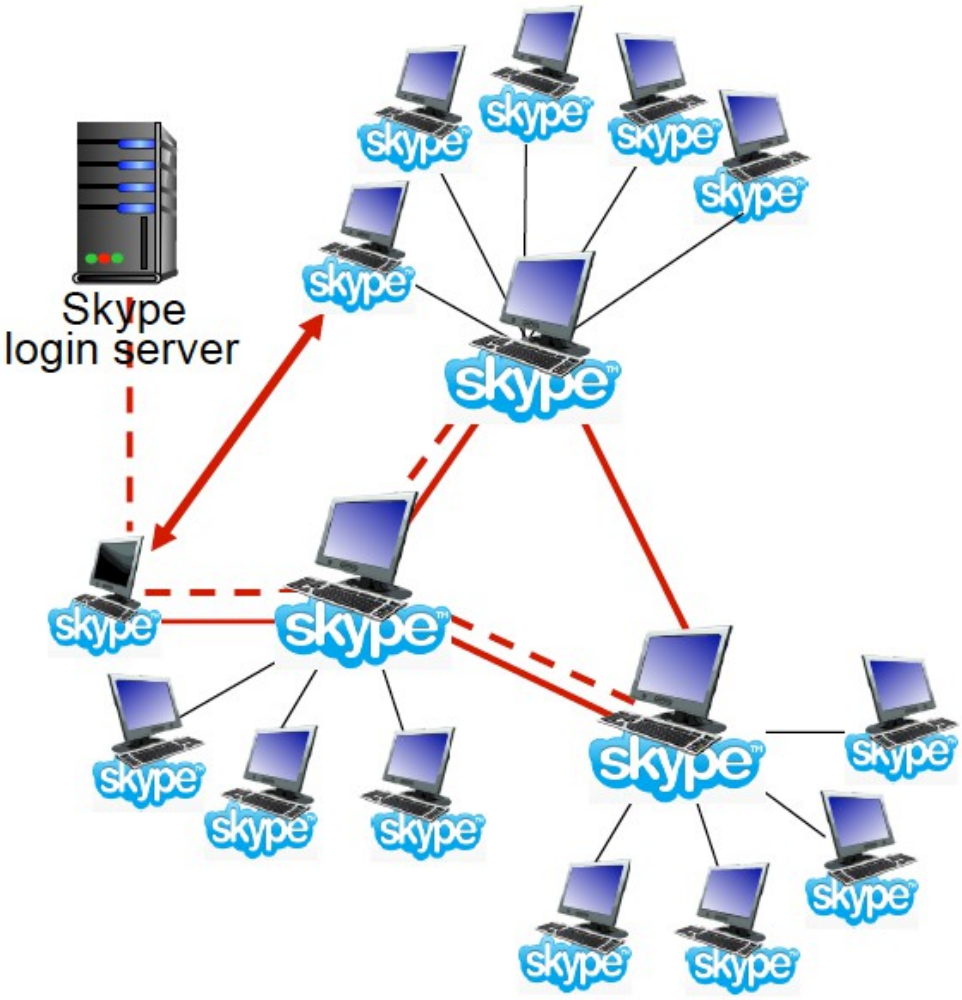
- Inerentemente P2P: comunicação “direta” entre pares de usuários
- Protocolo proprietário da camada de aplicação
  - inferido através de engenharia reversa
- Componentes P2P:
  - Clientes: *peers* Skype se conectam diretamente
  - Super nós: *peers* Skype com funções especiais
  - Rede sobreposta: entre SNs para localizar SCs
  - Servidor de *login*



VoIP P2P com Skype

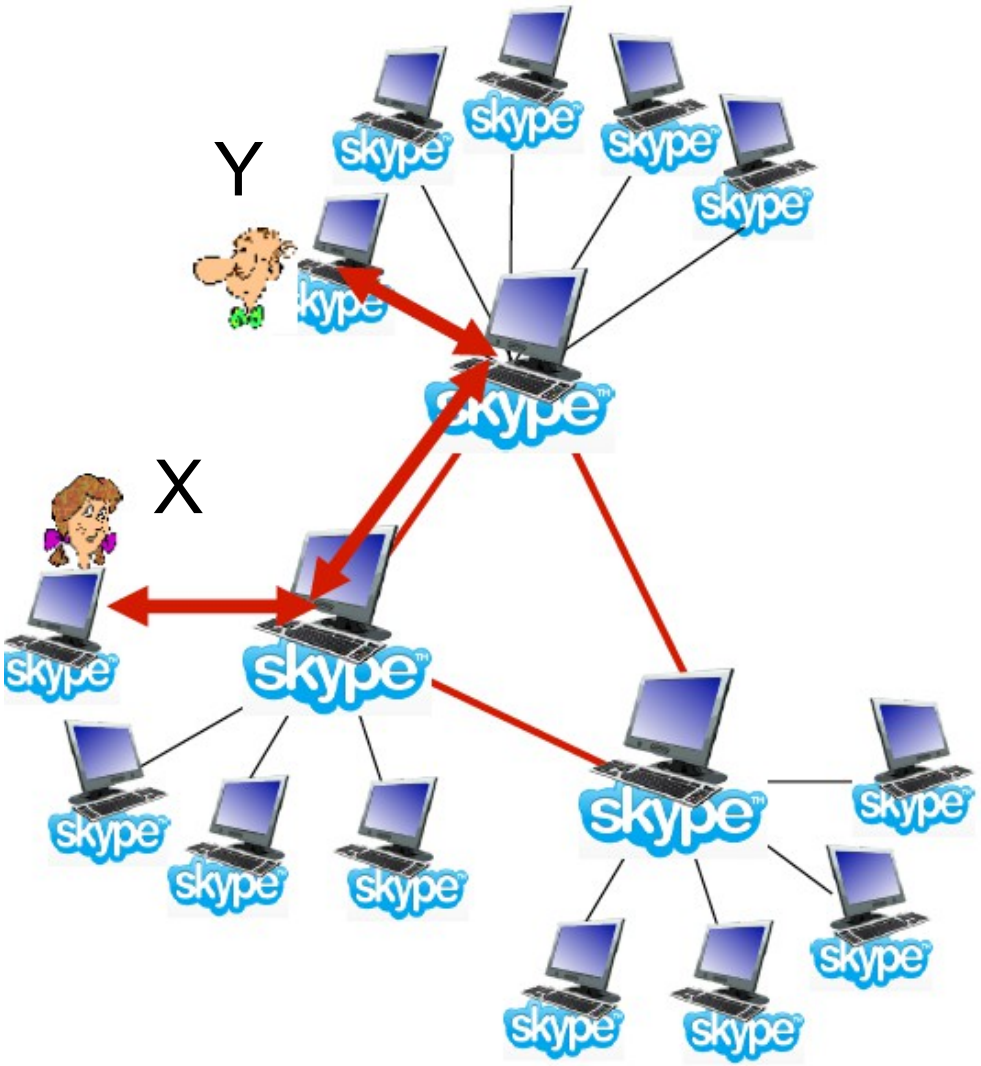
Operação de um cliente Skype:

1) Se junta à rede do Skype contactando SN
   - IP em cache
2) Se autentica no servidor de *login* do Skype
3) Obtém endereço IP do destino a partir de SN, rede sobreposta SN
   - Ou lista de contatos
4) Inicia chamada diretamente para destino



Peers como intermediários (*relays*)

- Problema: tanto X como Y estão atrás de NATs
  - O NAT impede que um *peer* externo inicie uma chamada com um *peer* interno
  - *Peer* interno pode iniciar conexão para fora
- Solução: X e Y mantêm conexão aberta com seus SNs
  - X solicita ao seu SN para conectar com Y
  - SN de X informa SN de Y
  - SN de Y informa a Y através de conexão previamente aberta (por Y)
  - Comunicação final pode ocorrer através de SNs como intermediários
    - ou poderia ser usada alguma técnica de travessia de NAT



Exercícios

1) Indique V (verdadeiro) ou F (falso).

a) Se a taxa de envio (*upload*) de um servidor for milhares de vezes maior que a taxa de envio (*upload*) de qualquer *peer*, o tempo de entrega de um arquivo para todos os *peers* com a abordagem cliente-servidor será menor que com a abordagem P2P (*peer-to-peer*), mesmo que existam milhões ou bilhões de *peers*.

b) O fato de um usuário conseguir obter um arquivo completo de uma rede que utiliza o protocolo BitTorrent, mesmo sem compartilhar ou disponibilizar o arquivo (ou partes dele) para outros usuários, demonstra que o mecanismo de incentivo implementado nesse protocolo não funciona adequadamente.

c) A "agitação" dos *peers* (*peer churn*), ou seja, a entrada e saída frequente de *peers* em sistemas P2P é um problema bem conhecido que tem como uma solução a manutenção de uma vizinhança mais ampla, ou seja, cada *peer* conhece alguns (ou vários) outros *peers*.

Exercícios

2) Considere a DHT (*Distributed Hash Table*) circular ilustrada pela figura para responder o que é pedido a seguir. Nessa figura, é ilustrado apenas o sucessor de cada *peer*, no entanto, cada *peer* conhece também o seu segundo sucessor.

a) Suponha que o *peer* 3 descobriu que o *peer* 5 deixou a rede. Como o *peer* 3 atualiza sua informação sobre o seu segundo sucessor? Qual *peer* passa a ser seu segundo sucessor?

b) Considere a situação original (com o *peer* 5 ainda na rede) e suponha que um novo *peer* 6 quer se juntar à DHT e o *peer* 6 conhece apenas o endereço IP do *peer* 15. Quais são os passos realizados para que o *peer* 6 passe a fazer parte da rede?