Cloud Computing

Cap 4. Grids. P2P: Napster, Gnutella, BitTorrent.

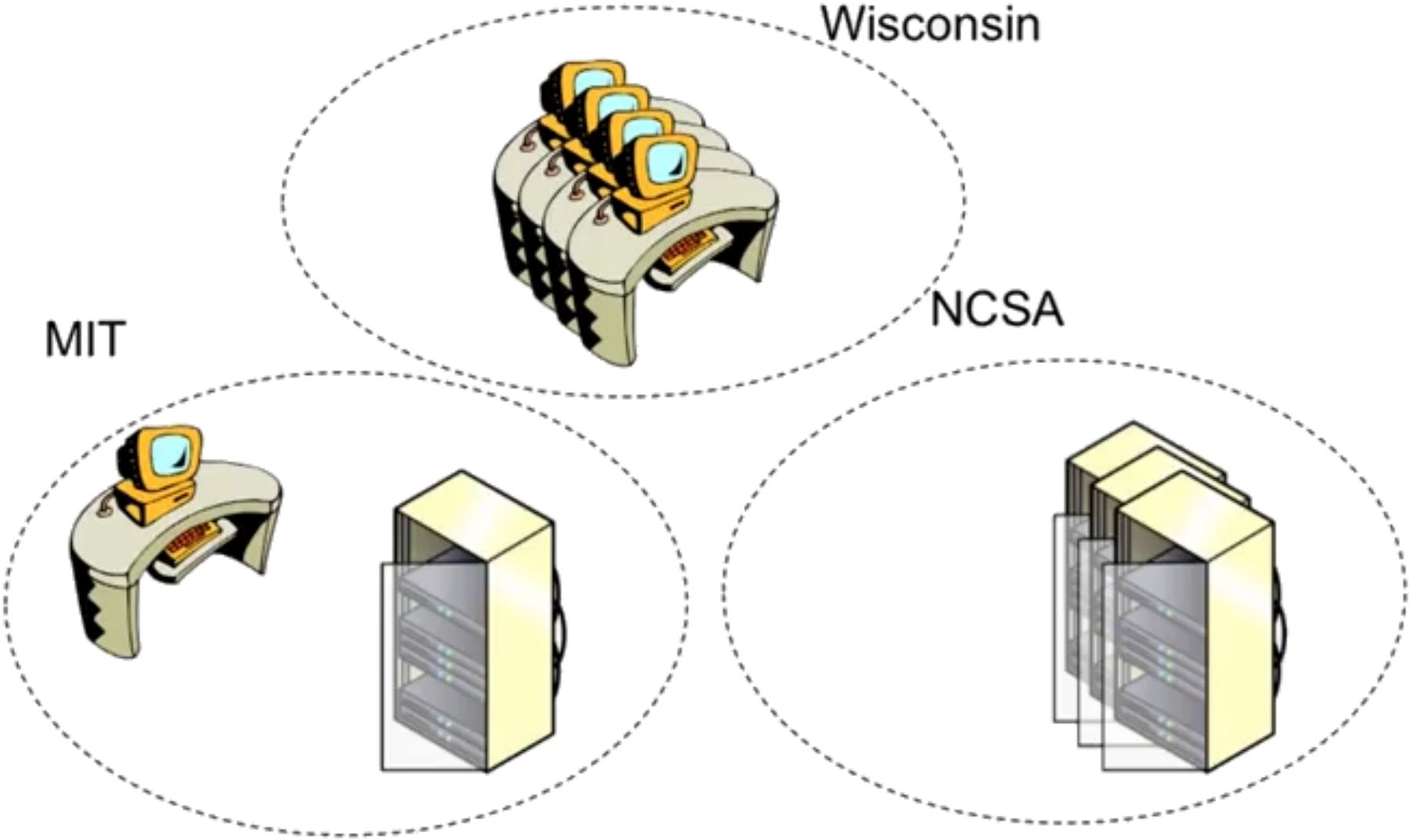
Introducere în Chord.

November 7, 2022

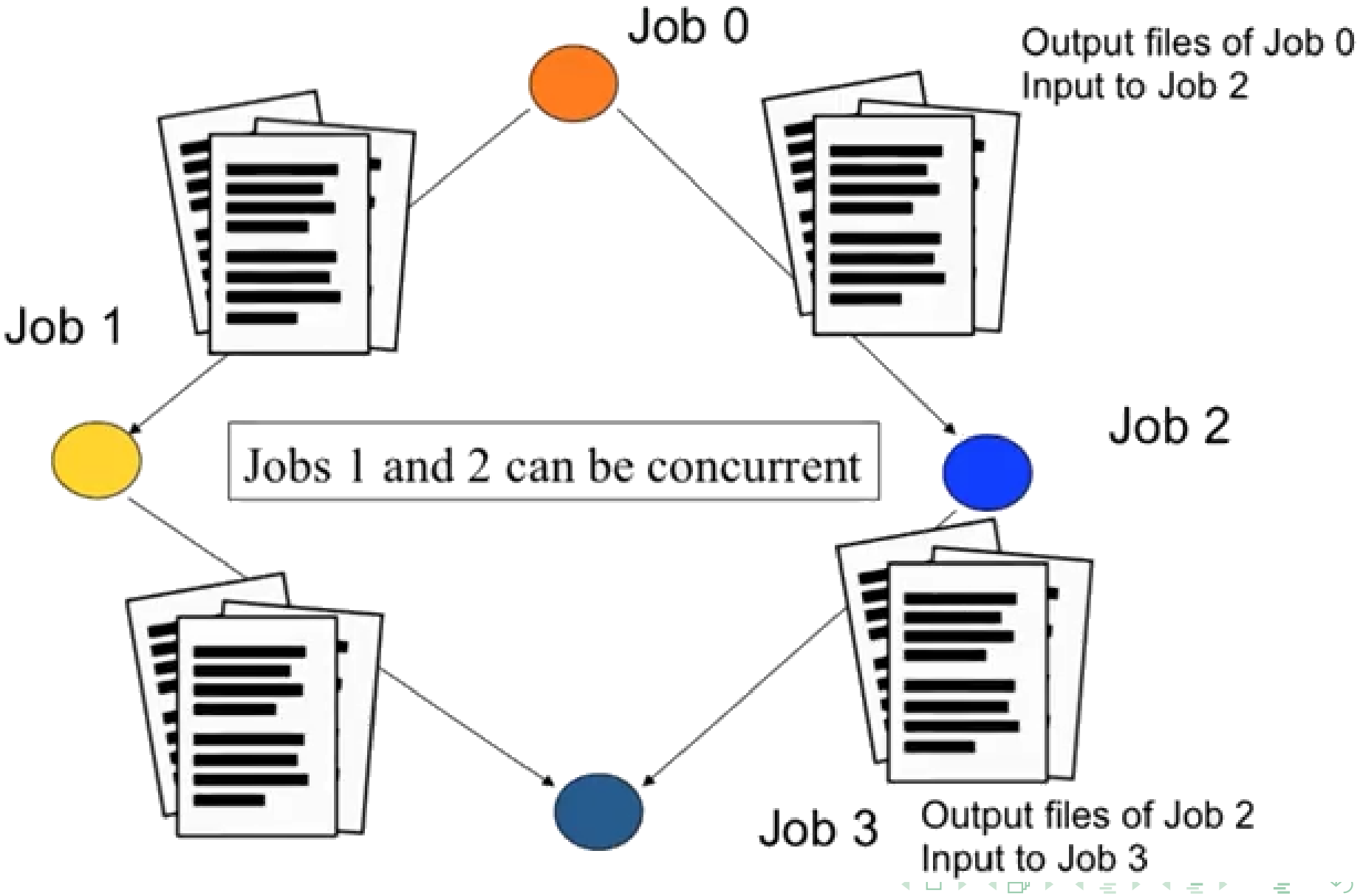
Aplicații de tip grid

- Rapid Atmospheric Modeling System (RAMS) - modelează fenomene meteorologice
- uraganul Georges, septembrie 1998
- modelare a spațiului probabil al precipitațiilor și a cantității lor
- s-a reușit limitarea la 5km față de precizia de 10km
- rulare pe 256+ procesoare
- HPC - calcul computațional intensiv
- cum se poate rula fără acces la un supercalculator?

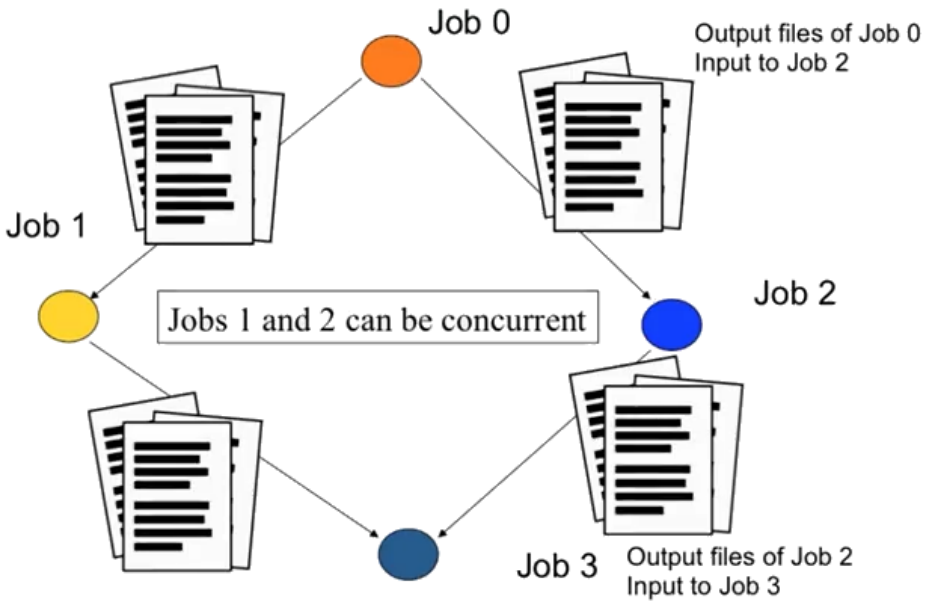
Resurse de calcul distribuit



Aplicația de calcul intensiv - DAG

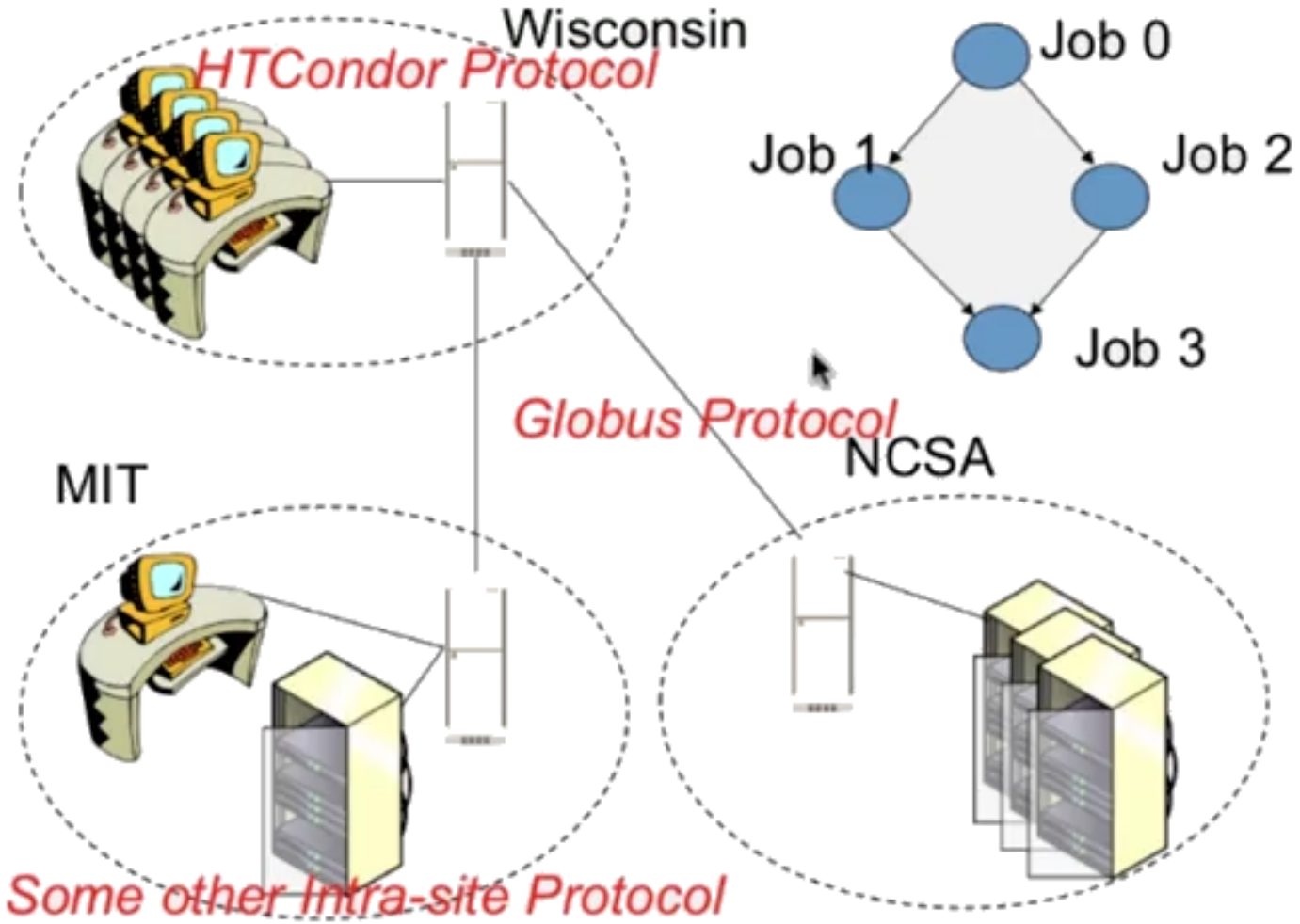


Aplicația de calcul intensiv - DAG



- fiecare job poate avea câțiva GB
- poate dura câteva ore / zile
- etape:
  - init
  - stage in (pull data)
  - execute
  - stage out (push data)
  - publish (rezultate intermediare)
  - nodul este paralelizabil în task-uri (HPC)
- scheduling?

Infrastructuri grid - planificare pe 2 nivele



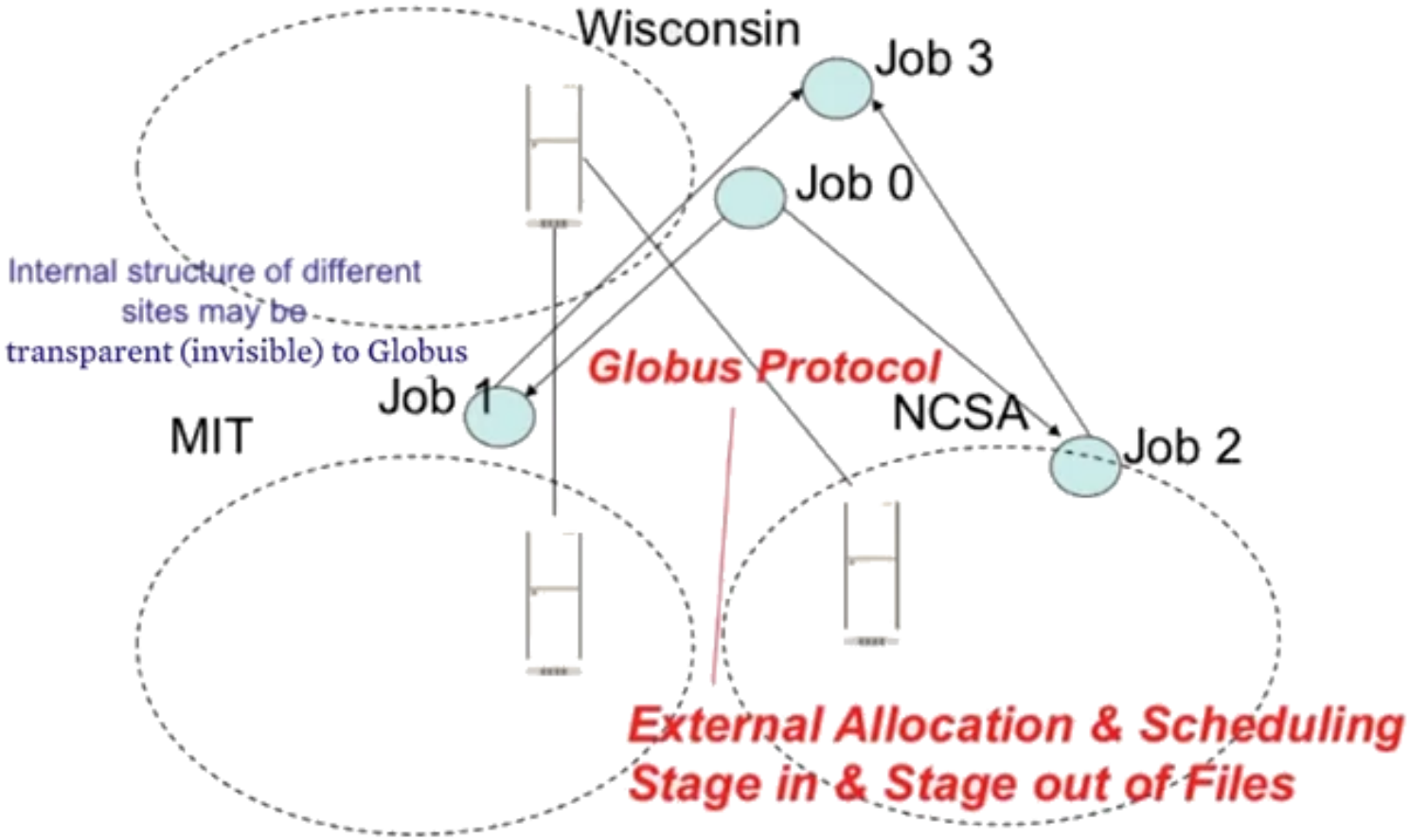
Infrastructuri grid - planificare pe 2 nivele

- protocolul intra-site este responsabil pentru:
  - alocare internă
  - planificare internă
  - restart task-uri
  - monitorizare
  - distribuția și publicarea fișierelor

Infrastructuri grid - planificare pe 2 nivele

- HTCondor (high-throughput) - U. Wisconsin
- rulează pe multe workstations
- dacă e nefolosită, stația cere task-uri server-ului central (sau Globus)
- dacă utilizatorul se log-ează, task-ul este oprit sau replanificat
- se poate rula și pe mașini dedicate

Infrastructuri grid - planificare pe 2 nivele



Globus

- universități, laboratoare, companii
- standardizează unelte software
- Open Grid Forum
- dezvoltă Globus toolkit
  - open-source
  - GridFTP: transfer de date bulk
  - GRAM (Grid Resource Allocator Manager) - nu e scheduler, comunică cu HTCondor sau PBS (Portable Batch System, intra-site)
  - RLS (Replica Location Service) - naming service
  - biblioteci precum XIO, API standard
  - GSI (Grid Security Infrastructure)

Aspecte de securitate

- grid-urile sunt federated, nu există o entitate unică ce controlează întreaga infrastructură

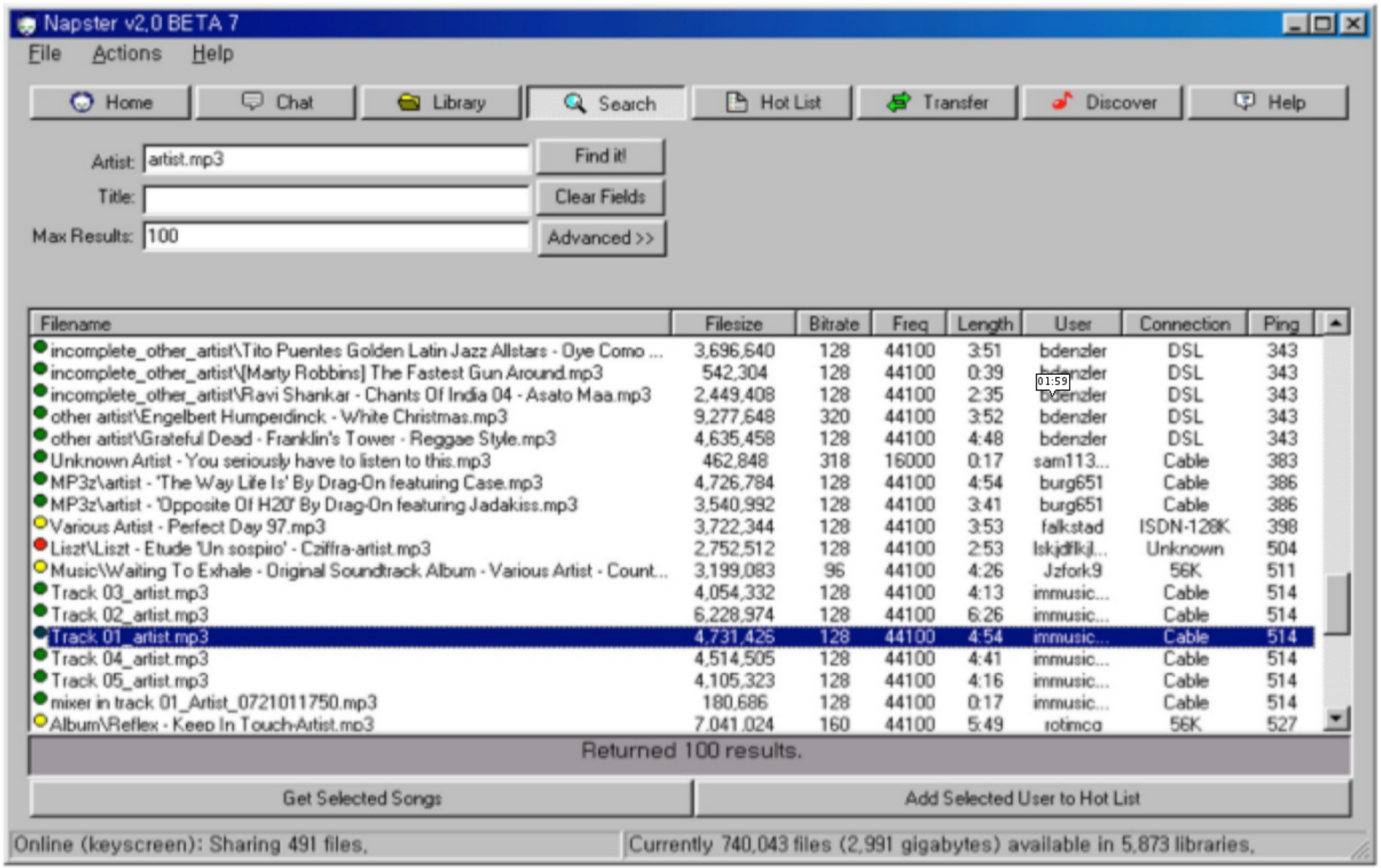
- single sign-on: job-urile colective trebuie să necesite o singură autentificare a unui user
- mapping to local security: site-uri ce folosesc Kerberos, altele folosesc UNIX
- delegare: credențiale pentru acces moștenit la resurse
- autorizare third-party (de tip grup)

- mai puțin importante în clouds, în clouds există o autoritate centrală

Sisteme Peer-to-Peer

- anii 2000, Napster, Gnutella, sute de milioane de clienți simultan
- primele sisteme distribuite ce s-au preocupat de o scalare bună odată cu creșterea numărului de noduri
- tehnologiile P2P sunt folosite în sistemele cloud
- sistemele key-value stores (Cassandra, Riak, Voldemort - Linkedin) folosesc hashing Chord de tip P2P

Sisteme Peer-to-Peer



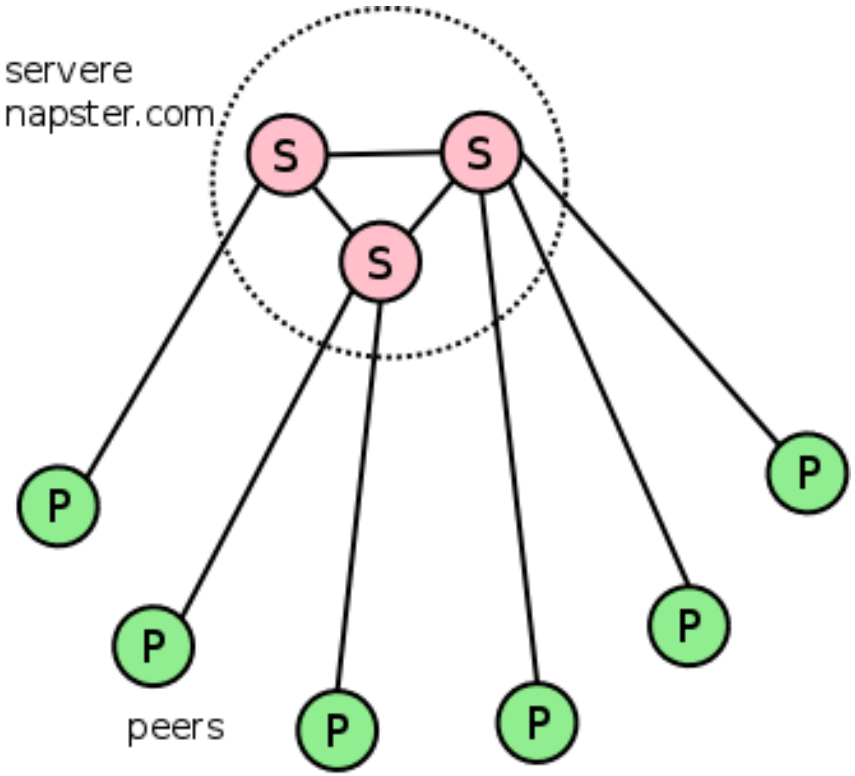
Istoric Napster

- (jun 1999) Shawn Fanning creează serviciul Napster pentru muzică on-line
- (dec 1999) RIAA dă în judecată Napster, $100K per download
- (mar 2000) 25% traficul Univ. Wisconsin dat de napster
- (2000) Napster are 60 mil. utilizatori
- (feb 2001) US Federal Appeals Court: Napster ajută utilizatorii să încalce copyright-ul
- (sep 2001) Napster ca serviciu plătit, procent compozitorilor și caselor de discuri
- (azi) open protocol, http://opennap.sourceforge.net

Conținut

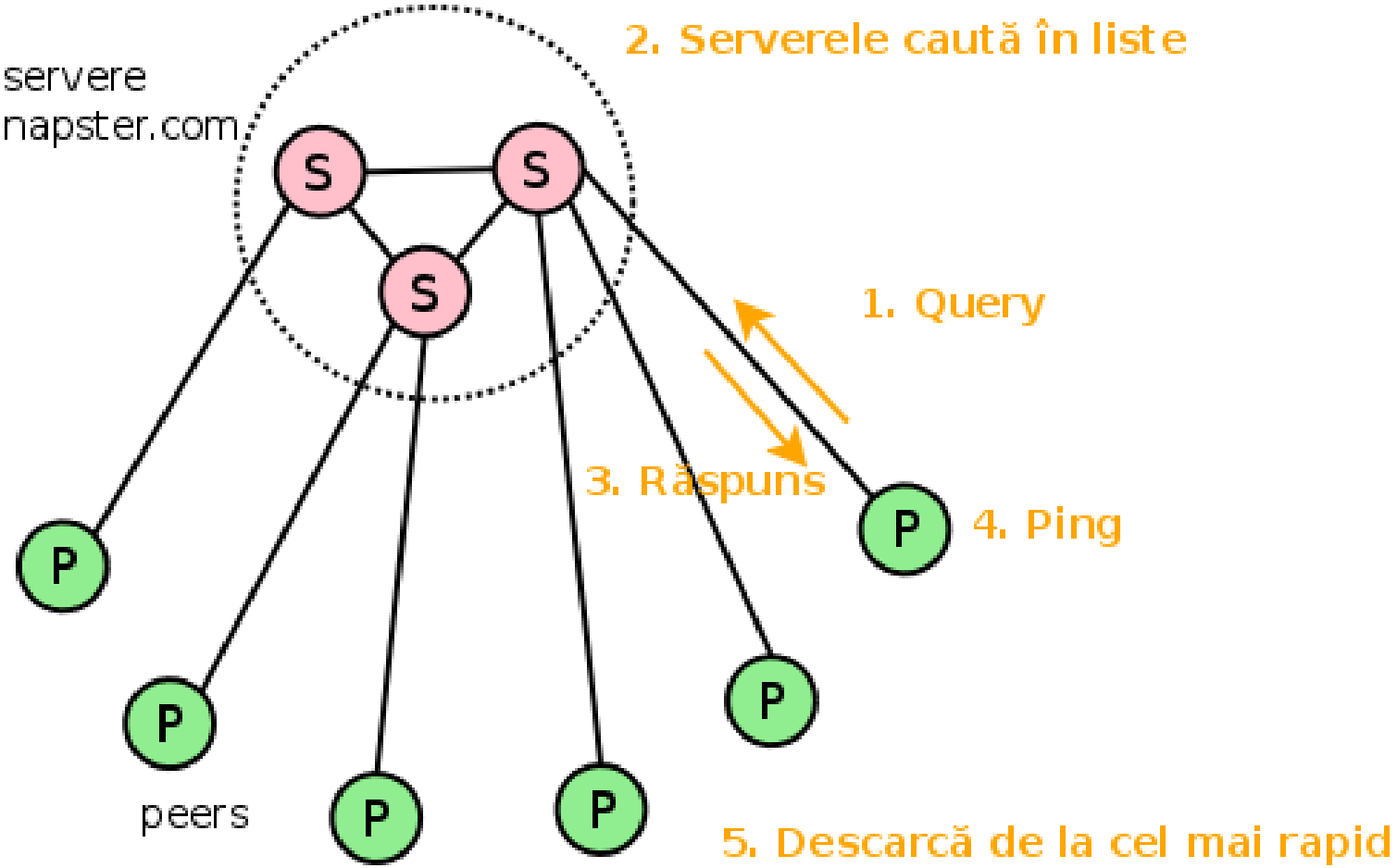
- sisteme P2P active pe scară largă:
  - Napser
  - Gnutella
  - Fasttrack (Kazaa, Kazaalite, Grokster)
  - BitTorrent
- sisteme P2P cu proprietăți verificabile
  - Chord
  - Pastry
  - Kelips

Sistemul Napster



- clienți Napster = peers
- fișierul stă pe mașina peer
- serverele Napster stochează informații despre fișiere și adresa de IP care îl conține
- clientul trimite la server o listă cu fișiere puse în comun
- serverul nu stochează fișiere
- pentru găsire, grupul de servere discută între ele
- clientul va face ping către toți clienții din listă; alege rata cea mai mare
- comunicație TCP

Sistemul Napster



Join-ul unui client

- trimite request la un nume de server cunoscut (ex. napster.com)
- după query-ul DNS, mesajul se trimite către server-ul ce ține evidența nodurilor din sistem
- cel care a făcut join va primi de la server lista vecinilor cei mai apropiați

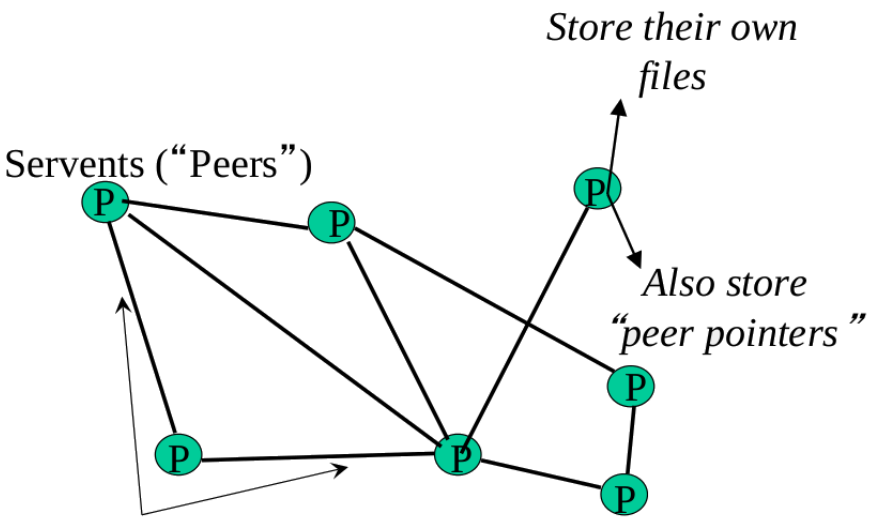
Probleme inițiale Napster

- serverul centralizat este o sursă de congestii
- serverul centralizat = point of failure
- nu există securitate: parole și mesaje în clar
- napster.com declarat ca fiind vinovat de încălcările de copyright din partea utilizatorilor (indirect infringement)
- un nou sistem: Gnutella, a rezolvat aceste probleme

Gnutella

- eliminarea serverelor
- mașinile client vor căuta între ele
- clienții se comportă și ca servere, denumiți servents
- în martie 2000 a fost pornit de AOL, retras imediat
- în 2003 avea 88K utilizatori
- design-ul original a suferit câteva modificări

Gnutella

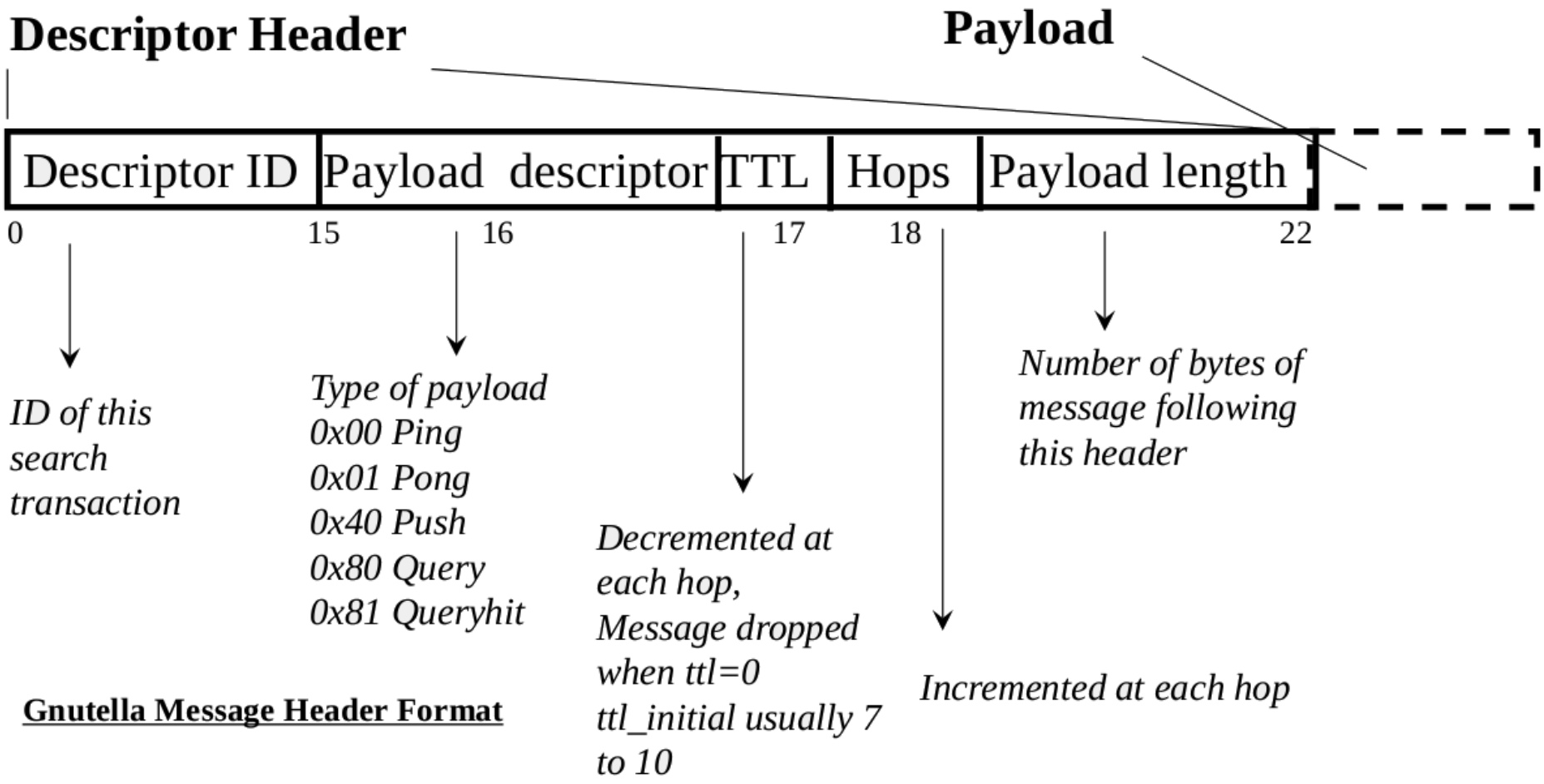


- peer-urile comunică direct
- stochează fișierele
- fiecare peer este legat de vecini
- overlay graph = fiecare link este o cale Internet

Căutare

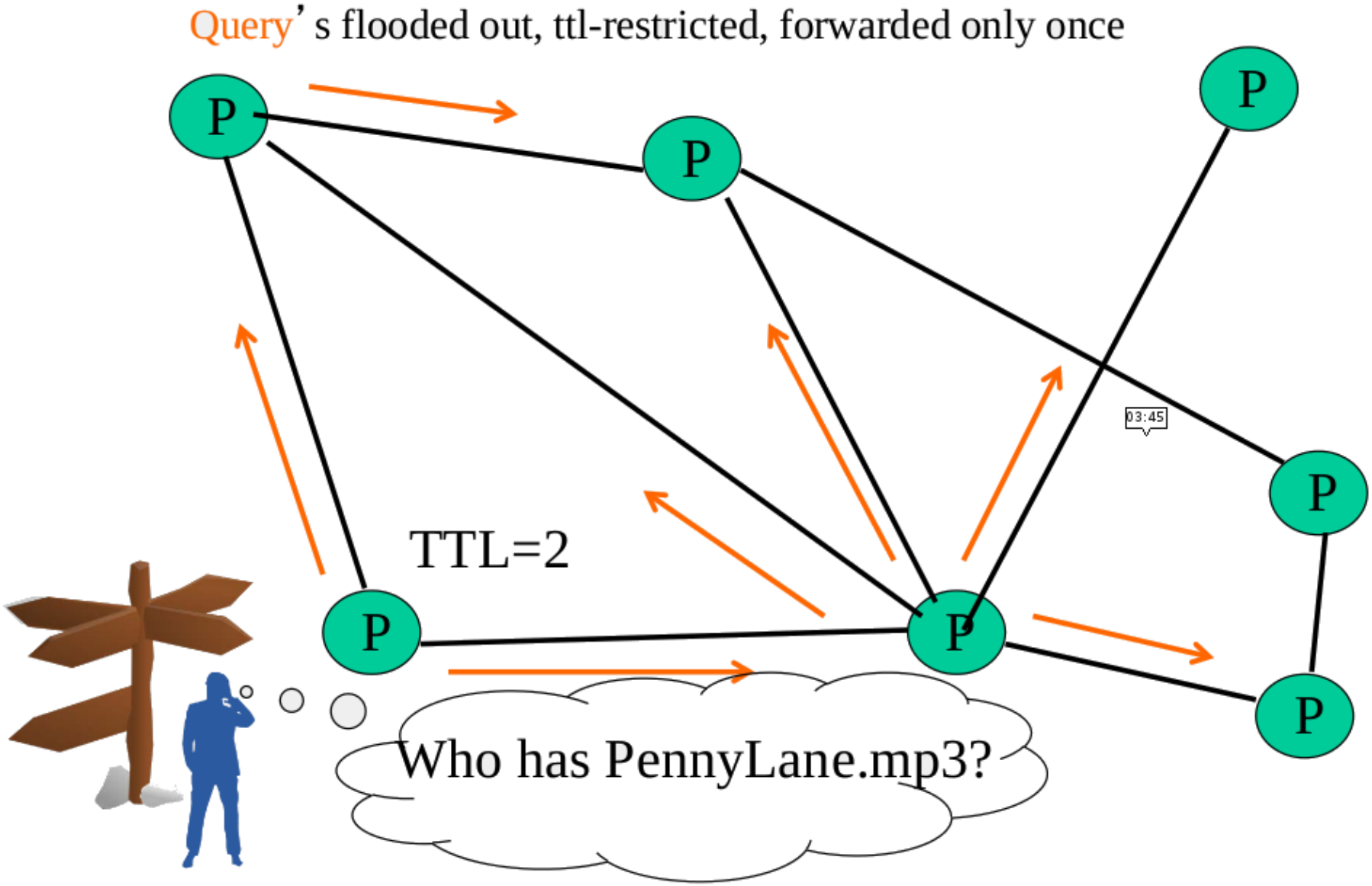
- Gnutella rutează mesaje în acest overlayed graph
- 5 tipuri de mesaje:
  - Query (search)
  - QueryHit (răspuns)
  - Ping (pentru a confirma existența vecinilor)
  - Pong (răspuns la ping, conține adresa altui peer)
  - Push (transfer de fișiere)
- în cele ce urmează se folosește notația little-endian

Căutare

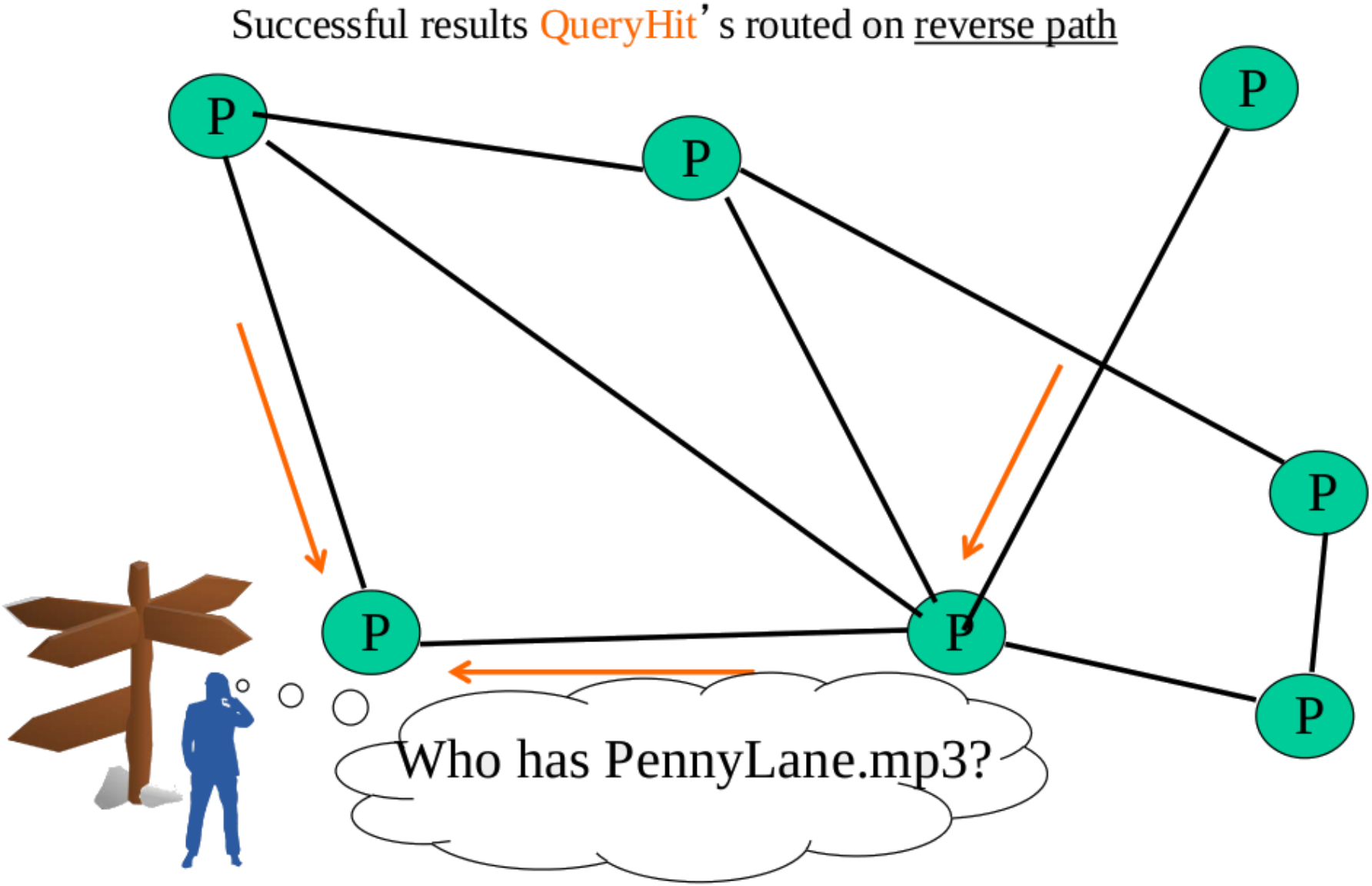


Gnutella Message Header Format

Căutare


Query's flooded out, ttl-restricted, forwarded only once
P
P
P
P
P
P
P
03:45
TTL=2
Who has PennyLane.mp3?

Căutare


Successful results QueryHit's routed on reverse path
P
P
P
P
P
P
P
Who has PennyLane.mp3?

Reducerea traficului excesiv

- pentru evitarea transmisiilor duplicat, fiecare peer păstrează o lisă cu mesajele transmise recent
- query-ul este trimis tuturor vecinilor cu excepția celui de la care l-a primit
- fiecare query (identificat de DescriptorID) este trimis doar o dată
- QueryHit este rutat înapoi doar către peer-ul care l-a emis, bazat pe DescriptorID
- duplicatele cu același DescriptorID și Payload descriptor sunt aruncate
- QueryHit cu DescriptorID pentru care nu este vizibil un query, este aruncat (când graful overlay se schimbă)

La primirea mesajelor QueryHit

- cel care cere selectează cel mai bun QueryHit care a răspuns
- inițiază request-ul HTTP direct către IP:port-ul celui ce a răspuns:

```
GET /get/<File Index>/<File Name>/HTTP/1.0\r\n
Connection: Keep-Alive\r\n
Range: bytes=0-\r\n
User-Agent: Gnutella\r\n
\r\n
```

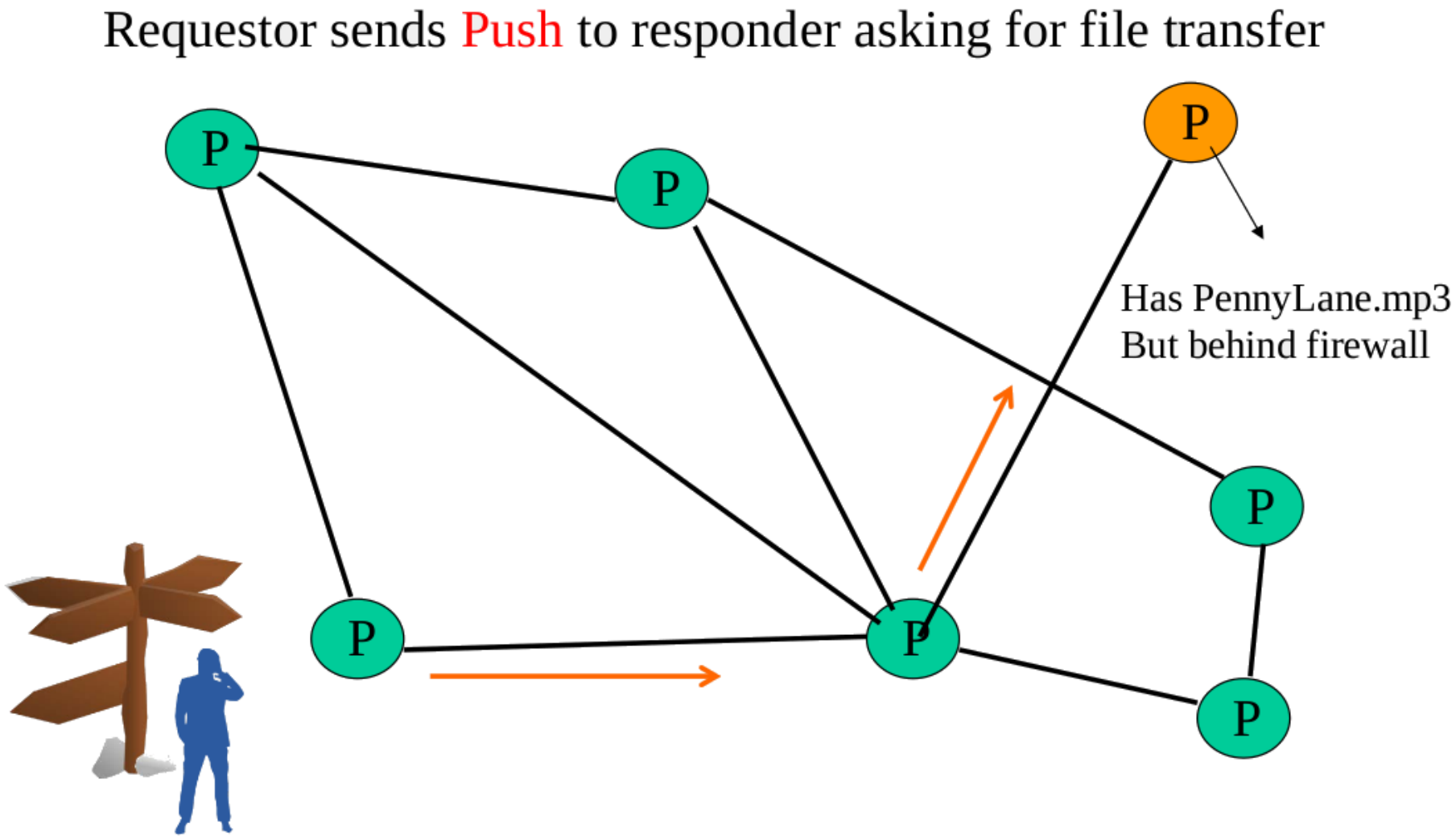
- respondentul replică cu pachete din fișier:

```
HTTP 200 OK\r\n
Server: Gnutella\r\n
Content-type: application/binary\r\n
Content-length: 1024\r\n
\r\n
```

QueryHit și posibile probleme

- HTTP frecvent folosit ca protocol de transfer
- parametrul range oferă suport pentru transferul parțial
- ce se întâmplă dacă peer-ul care răspunde este în spatele unui firewall care nu permite conexiuni incoming?

QueryHit și posibile probleme



Firewalls

- respondentul va stabili o conexiune TCP pe adresa și portul specificate
- cel care cere trimite un mesaj GET la responder, iar fișierul se va transfera pe această conexiune
- posibil ca și inițiatorul să fie în spatele unui firewall

Ping-pong

- ping este folosit de un nod pentru a-și afla vecinii
- peer-urile inițiază periodic ping-uri flood; TTL-restricted, < 7
- răspunsul vine în Pong, pe ruta reverse; conține adresa IP a peer-ului care a răspuns (QueryHit), numărul de fișiere respectiv de KB puse în comun
- folosit pentru a prefera vecini care au mai multe date de oferit
- rata de churn (join/leave) e foarte ridicată, lista de membership a vecinilor trebuie menținută up-to-date

Gnutella pe scurt

- nu există servere
- peer-urile formează un overlay graph și mențin liste cu veninii
- peer-urile își stochează fișierele
- Query-urile se transmit prin flooding, TTL-restricted
- răspunsurile QueryHit sunt routed pe calea inversă
- suportă transfere și în cazul în care responding peer e în spatele unui firewall
- dialog periodic de tip ping-pong pentru refresh-ul continuu al listei de vecini (churn ridicat)
- lista de vecini de dimensiune variabilă, eterogenă - anumite peer-uri pot avea mai mulți vecini

Probleme cu Gnutella

- traficul ping/pong însumează 50% din total; soluții:
  - multiplex, la primirea mai multor pong-uri, se forwardează unul singur; la fel pentru ping, forward doar 1 ping
  - cache: la primirea unui ping pentru un alt nod, dacă s-a păstrat un pong, se trimite acest pong cached
  - reducerea frecvenței ping/pong
- căutări repetate cu aceleași cuvinte cheie
  - cache-ul mesajelor Query și QueryHit
- peer-uri conectate prin modem-uri lente, bandwidth redus
  - folosirea unui server central ca proxy pentru peer-uri lente
  - folosirea nodurilor mai puternice din sistem (Fasttrack)

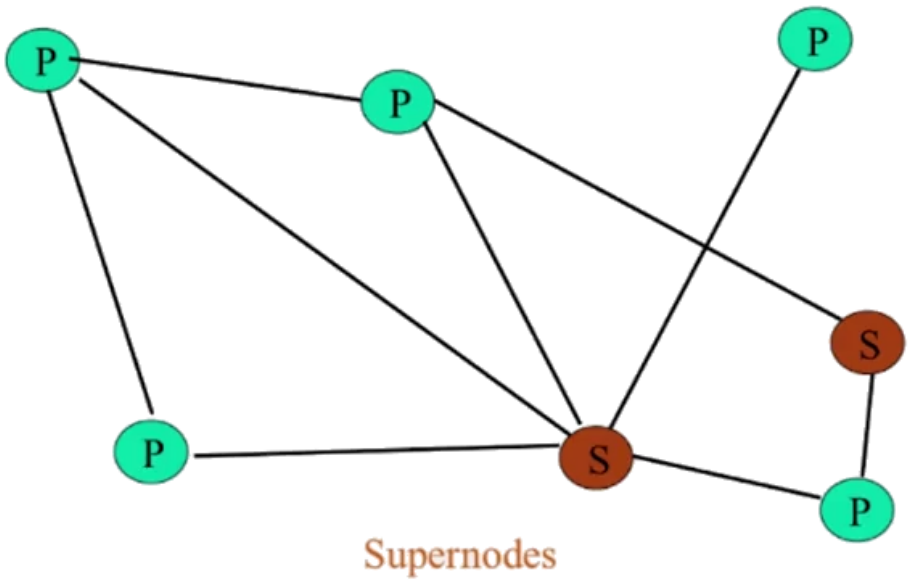
Probleme cu Gnutella

- număr mare de freeloaders (utilizator care doar downloadează)
  - 70% din utilizatori erau freeloaders (2000)
  - problemă de comportament, nu etică
  - doar 30% din noduri se vor comporta ca servere de fișiere, majoritatea responsabilităților (încărcare dezechilibrată)
- flooding-ul cauzează trafic excesiv
  - există o modalitate de a folosi meta-informație pentru un routing inteligent?
  - sisteme Peer-to-Peer structurate (sistemul Chord, structurat, studiat teoretic)

FastTrack

- sistem popular, ca și BitTorrent
- hibrid între Gnutella și Napster
- profită de participanții mai puternice din sistem pentru a face sistemul mai rapid
- tehnnologie folosită de Kazaa, KazaaLite, Grokster
- protocol proprietar, câteva detalii disponibile
- seamănă cu Gnutella (overlay graph), există însă peer-uri denumite supernodes

Un sistem de tip FastTrack

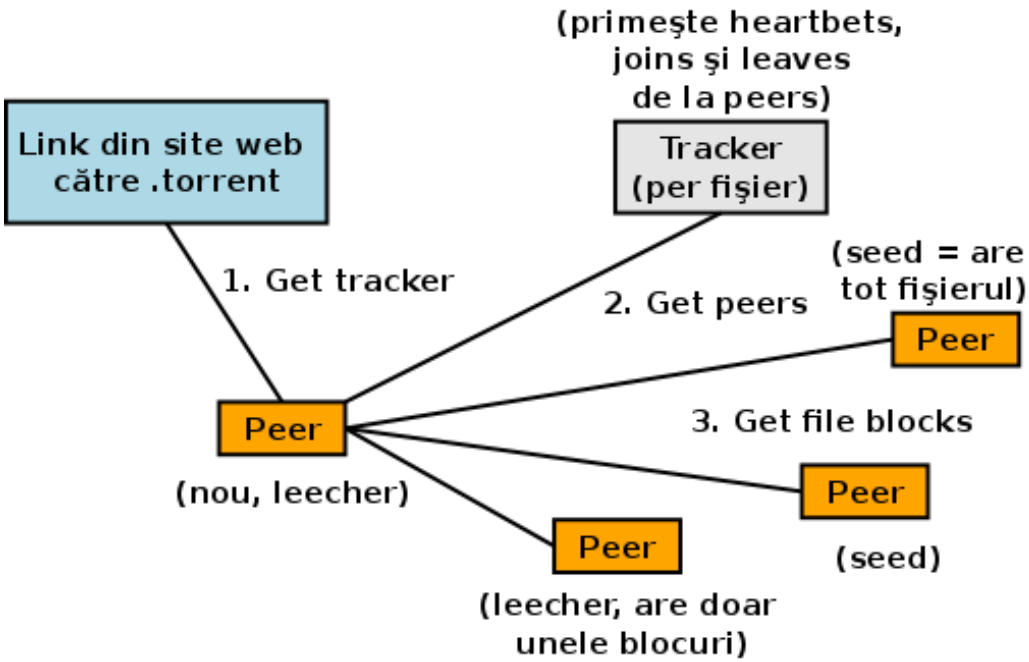


- supernodurile au un rol similar cu nodurile server din Napster
- preiau informația directory listing pentru a fi folosită apoi de vecini în căutări

Un sistem de tip FastTrack

- un supernod stochează un directory listing, un subset de tip (nume fișier, pointer spre peer)
- status-ul de tip supernod se schimbă în timp
- un nod nu se poate alege singur supernod
- orice nod poate deveni supernod dacă a câștigat suficientă **reputație** (contribuție în trecut):
  - KazaaLite: între 0 și 1000, reprezintă nivelul de participare; inițial este 10, este afectată de lungimea perioadei de prezență în sistem și numărul de upload-uri
  - la trecerea peste un anumit prag, peer-ul devine supernod
  - siteme de reputație mai complexe, bazate pe fenomente economice, vezi workshop-ul P2PEcon
- un peer caută prin inspectarea unui supernod vecin; se previne astfel flooding-ul
- multe query-uri devin astfel seach-uri locale, nu implică trafic de rețea (rapide)

BitTorrent



- introduce incentives, motivează peer-urile să nu fie egoiste
- tracker-ul are o listă de peer-uri (câteva) ce transferă fișierul
- block-uri de mărime egală

BitTorrent

- fișierul e împărțit în blocuri (32 KB - 256 KB)
- download policy: **cel mai rar bloc primul**; se preferă blocurile care au replicarea cea mai slabă printre vecini (ca să nu dispară)
  - excepție: unui nod nou îi este permis să aleagă un vecin la întâmplare (ajută la pornire, să nu se blocheze pe un nod)
- folosirea benzii: furnizează cu precădere blocuri vecinilor care au furnizat rata de download cea mai ridicată
  - motivație pentru noduri să furnizeze rată de download bună
  - la fel procedează și seed-urile
- **chocking**: limitarea numărului de vecini către care se face upload concurent (≤ 5)
  - ceilalți sunt "înecați" (sugrumați?); se controlează bandwidth-ul de upload
  - reevaluare periodică a setului (la 10 s)
  - periodic, se activează un vecin - se menține setul "proaspăt"

Distributed Hash Table (DHT)

- Chord - proiectat de mediul academic; aplicații în key-value stores
- hash table: structură de date in-process; insert, lookup și delete de obiecte pe baza cheii, în timp constant
- un hash table distribuit permite acest lucru într-un mediu distribuit, obiectele fiind fișiere (key = file name)
- paralelă cu stocarea obiectelor pe bucket-uri vs. stocarea fișierelor în noduri, distribuite în lume
- considerații de performanță:
  - load balancing: fiecare nod are cam același număr de obiecte
  - fault tolerance: s-ar putea ca unele noduri să cadă, dar nu se vor pierde obiecte
  - eficiența lookup-urilor și insert-urilor: rapide
  - locality: transferul de mesaje transmise în cluser să se facă mai mult între vecini
- Napster, Gnutella și FastTrack sunt un fel de DHT
- Chord: DHT "evoluat" din cele anterioare

Comparație a performanțelor

	Memorie (client / server)	Timp lookup	Număr mesaje pentru lookup
Napster	O(1) la client O(N) la server	O(1)	O(1)
Gnutella	O(N)	O(N)	O(N)

- N este numărul de clienți (milioane de noduri)
- încărcarea la server pentru Napster este mare (liniară)
- numărul de vecini pentru Gnutella poate fi chiar $N - 1$
- pentru o topologie degenerată, timpul de lookup este în O(N)

Comparație a performanțelor

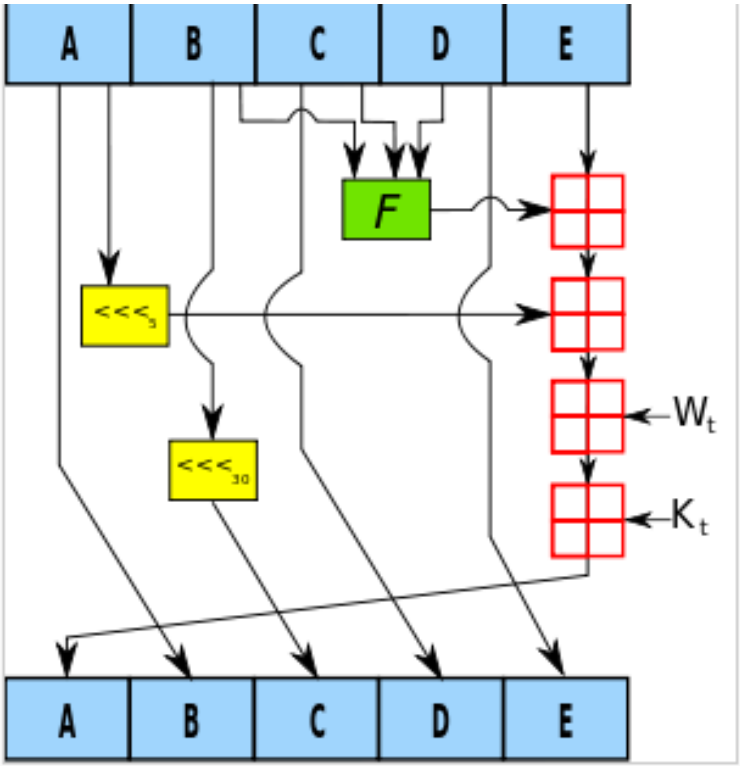
	Memorie (client / server)	Timp lookup	Număr mesaje pentru lookup
Napster	O(1) la client	O(1)	O(1)
Gnutella	O(N) la server O(N)	O(N)	O(N)
Chord	O(log(N))	O(log(N))	O(log(N))

- O(log(N)) în practică este aproape de constant

Chord

- autori: I. Stoica, D. Karger, F. Kaashoek, H. Balakrishnan, R. Morris, Berkeley și MIT (2001)
- alegerea inteligentă a vecinilor pentru a reduce latența și costul de routing al mesajelor (lookup-uri / insert-uri)
- folosește **consistent hashing** pentru adresa nodurilor:
  - SHA-1 (Secure Hashing Algorithm 1) pentru (IP, port) $\rightarrow$ string de 160 biți
  - trunchiere la m biți
  - denumit peer ID, număr între 0 și $2^m - 1$
  - nu e unic, dar ID-urile duplicate (collisions) sunt rare
  - poate mapa fiecare peer la unul din cele 2^m puncte de pe un cerc

SHA-1

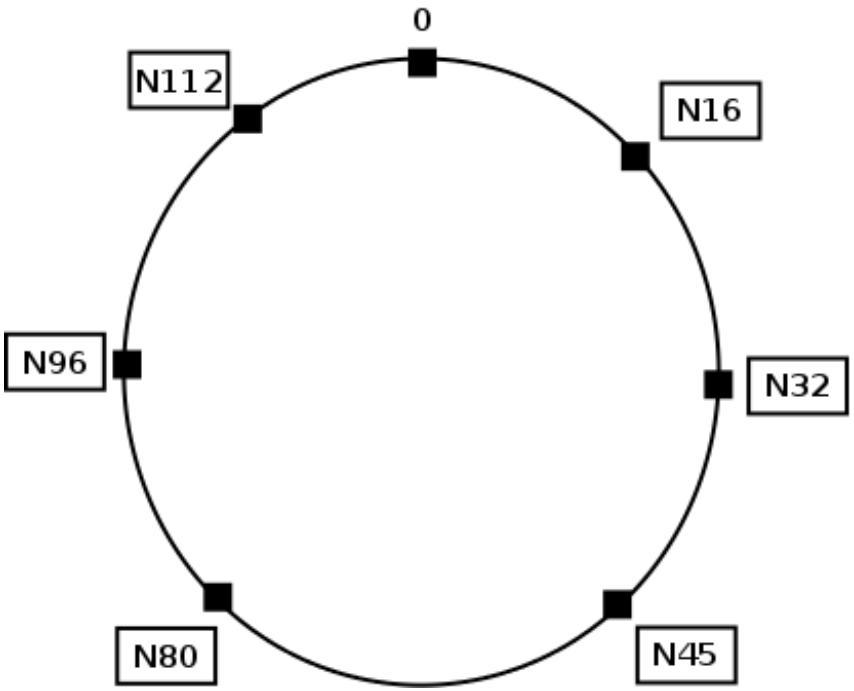


One iteration within the SHA-1 compression function:
A, B, C, D and E are 32-bit words of the state;
F is a nonlinear function that varies;
$\lll_n$ denotes a left bit rotation by n places;
n varies for each operation;
W_t is the expanded message word of round t;
K_t is the round constant of round t;
$\boxplus$ denotes addition modulo 2^{32}.

- sursa:

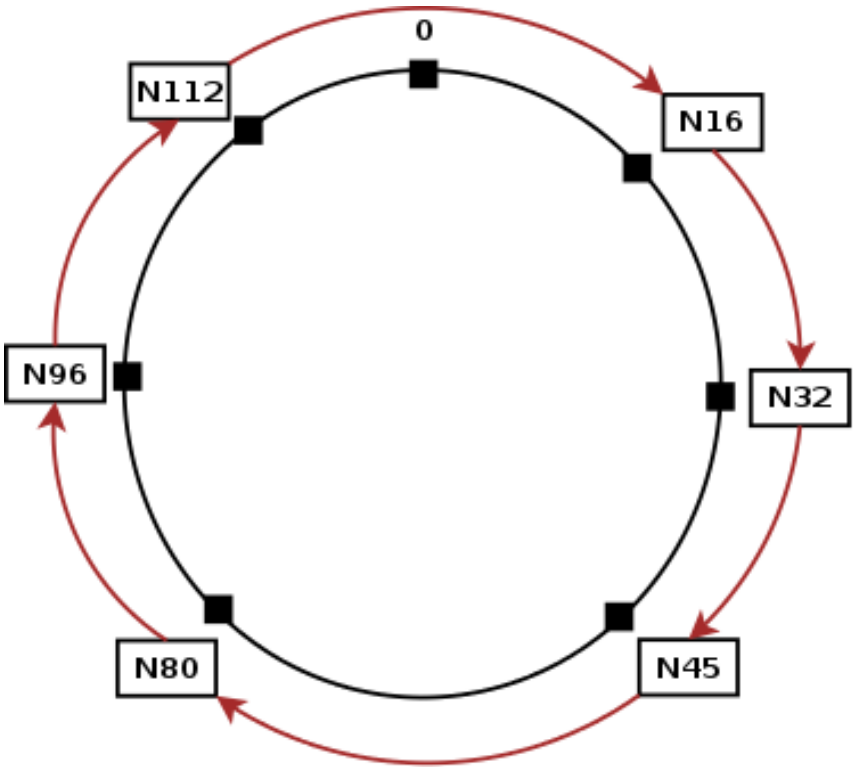
  `https://en.wikipedia.org/wiki/SHA-1`
- folosit în TLS, SSL, PGP, SSH
- în Git, Mercurial pentru detectarea coruperii datelor
- produce message digest (MD) de 160 biți (20 bytes)
- collision: două mesaje diferite ce produc același MD
- $2^{160} \approx 10^{48}$, jumătate din numărul atomilor din universul observabil
- rata de coliziune tipică sub 10^{-40}

Inel de peer-uri



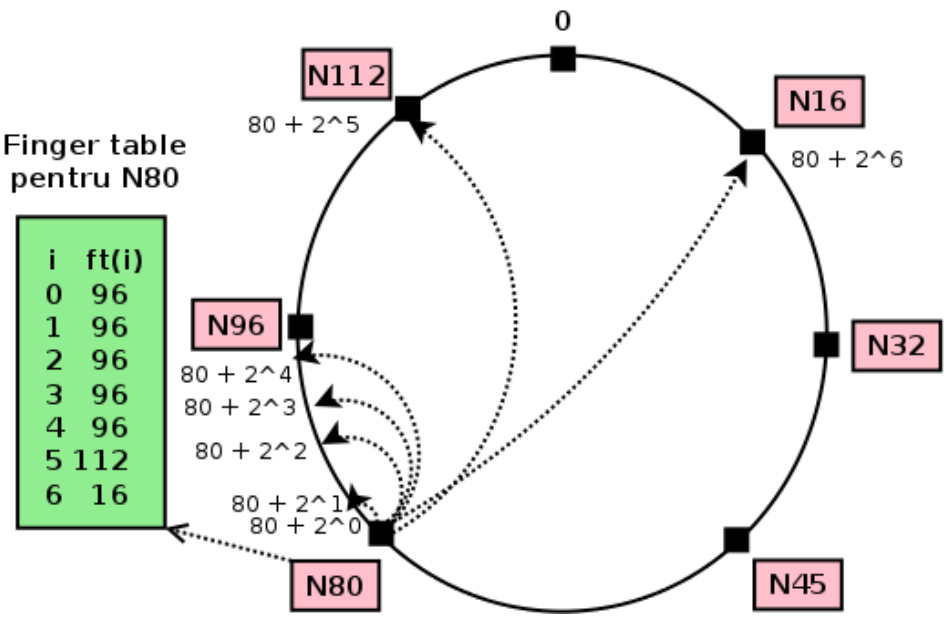
- m = 7 (127 poziții posibile), 6 noduri
- se stochează peer ID-urile
- fiecare peer își cunoaște peer-ul succesor:

Inel de peer-uri



- m = 7 (127 poziții posibile), 6 noduri
- se stochează peer ID-urile
- fiecare peer își cunoaște peer-ul succesor
- în mod similar pentru predecesori (mai rar folosit în practică)
- nodul **succesor** este primul tip de peer pointer

Finger tables



- m = 7
- pentru nodul cu ID-ul n, $ft(i)_n$ = primul peer cu ID-ul $\geq (n + 2^i)$ **modulo** 2^m
- folosit pentru routing-ul unui query foarte rapid

  n = 80 (N80)

  $n + 2^0 \rightarrow N96$

  . . .

  $n + 2^5 = 80 + 32 \rightarrow N112$

  $n + 2^6 = 80 + 64 \rightarrow N16$
- i crește cu 1, dar distanța se dublează

Sumar

- sisteme P2P utilizate pe scară largă
  - Napster
  - Gnutella
  - FastTrack (propriety for Kazaa, KazaaLite, Grokster)
  - BitTorrent
- sisteme P2P cu proprietăți demonstrate - problema DHT
  - Chord (MIT, Berkeley)