

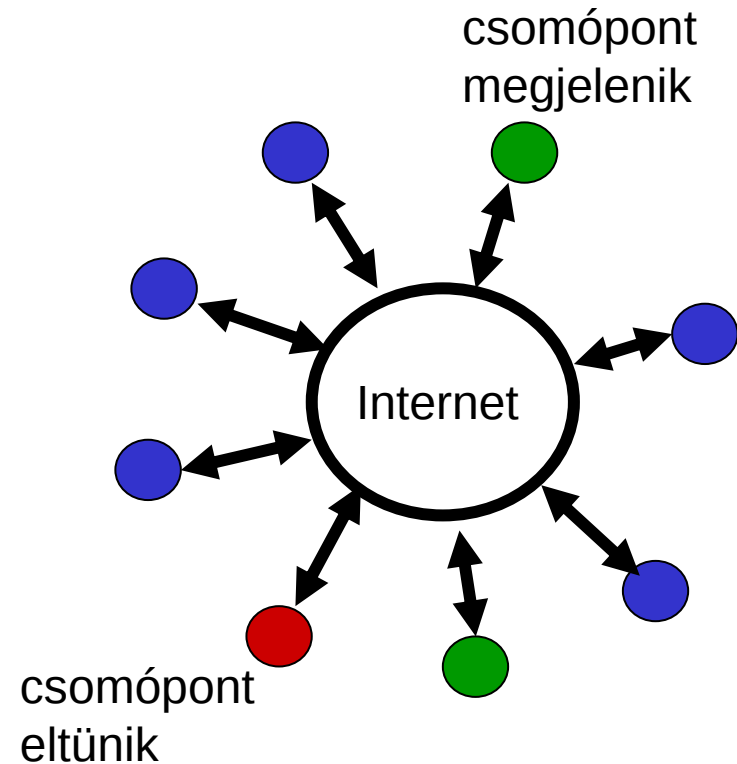


Hálózatok II 2006

13: Peer-To-Peer Hálózatok III

Peer-to-peer Hálózatok

- Peer-to-peer hálózatok elosztott rendszerek
 - Központi kontrol és hierarchikus struktúrák nélkül
 - Azonos szoftware-rel
 - Nagyfokú dinamikával, azaz csomópontok megjelennek és eltűnnek
 - Sok csomóponttal
 - Kevés hálózat-információval

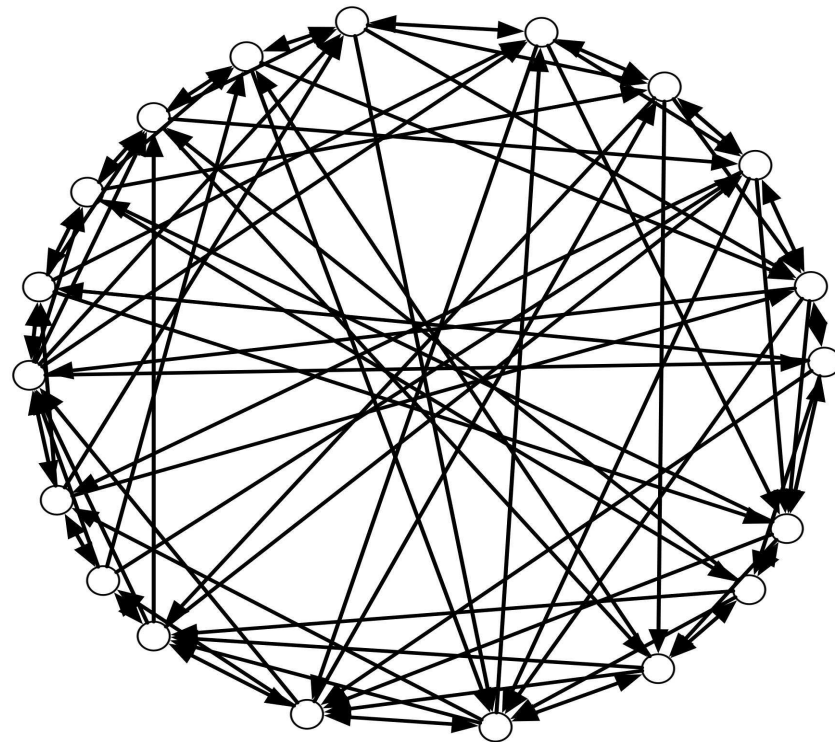


P2P hálózatok kritériumai

- Kezelhetőség
 - Milyen nehéz a hálózatot működtetni
- Információ-koherencia
- Mennyire jól osztja el az információt?
- Bővíthetőség
 - Milyen könnyen tud növekedni?
Mennyi idő alatt aktualizálódik a hálózat Peer-ek belépése/kilépése után
- Hibatűrés
 - Milyen könnyen hárítja el a hibákat?
- Biztonság
 - Mennyire nehezen rombolható szét tudatosan?
- Skálázhatóság
 - Milyen nagyra tud a hálózat növekedni?

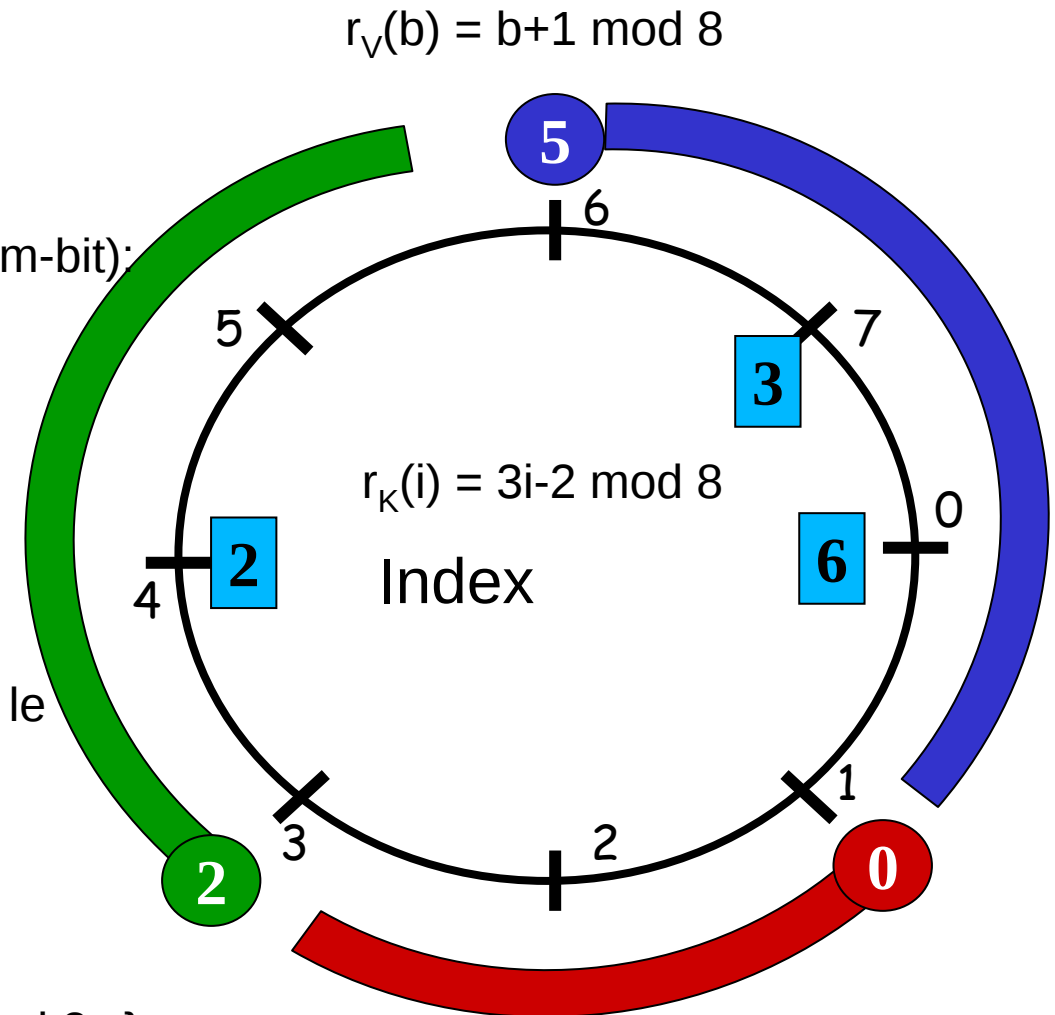
Chord

- DHT (elosztott hash-tábla), ahol a leképezési tartomány $\{0, \dots, 2^m - 1\}$
 - m elegendően nagy
- A Peer-ek egy gyűrűt formálnak
- A gyűrűben útrövidítések vannak bizonyos utódokhoz, melyek távolsága exponenciálisan növekszik



Chord mint DHT

- V : csomópontok halmaza, $|V| = n$
- K : kulcsok halmza (adatok), $|K| = k$
- m : hash-függvény értékének hossza (m-bit):
 $m \gg \log \max\{k, n\}$
- Két hash-függvény, ami $\{0, \dots, 2^m - 1\}$ -re képez
 - $r_V(b)$: $b \in V$ Peer-t képezi le $\{0, \dots, 2^m - 1\}$ -re véletlenül
 - $r_K(i)$: $i \in K$ kulcsot (adatot) képezi le $\{0, \dots, 2^m - 1\}$ -re véletlenül
- Egy i kulcs leképezése egy b Peer-re
 $b = f_V(i)$
 - $f_V(i) := \arg \min_{b \in V} \{ (r_K(i) - r_V(b)) \bmod 2^m \}$



Chord adatstruktúrája

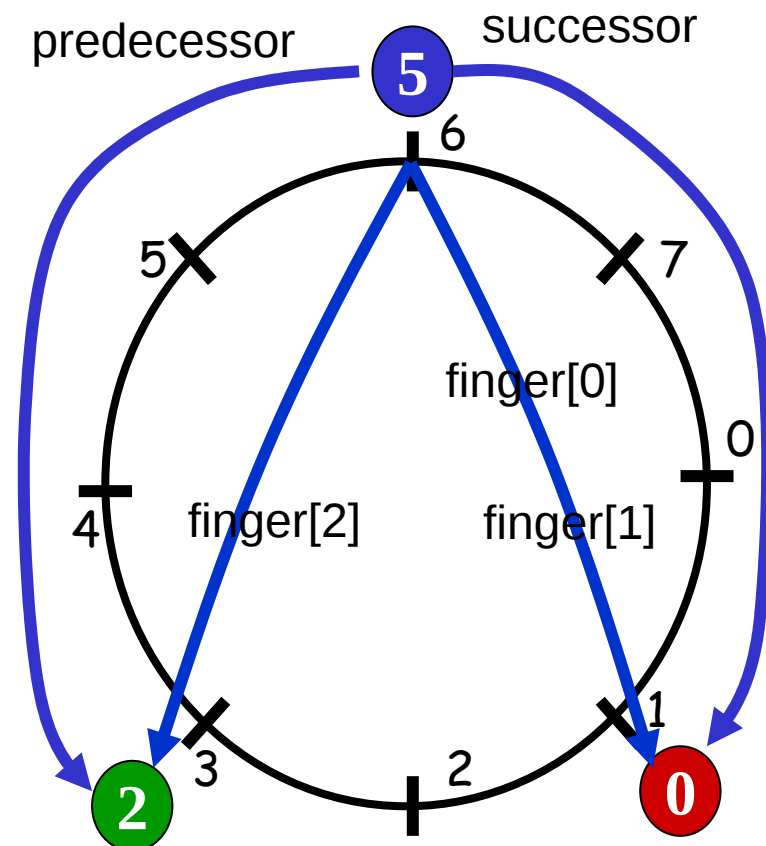
Minden b csomóponthoz tároljuk:

- **successor**: következő csomópont a gyűrűn
- **predecessor**: megelőző csomópont
- minden $i \in \{0, \dots, m-1\}$ -re
 - **finger[i]**: az a csomópont, amely képe $r_v(b) + 2^i \bmod 2^m$ értékét követi, azaz legalább $r_v(b) + 2^i \bmod 2^m$ és azok között a legkisebb
 - egyszerűbb jelölés miatt: $\text{finger}[m] := b$
- Kicsi i esetén a finger-bejegyzések gyakran azonosak
 - csak különböző finger-bejegyzéseket tárolunk

Lemma:

A különböző finger-bejegyzések száma b csomóponton $O(\log n)$ nagy valószínűséggel.

nagy valószínűséggel = $1 - n^{-c}$ valószínűséggel



Keresés a Chord-ban

Tétel 1:

Egy kulcs keresése $O(\log n)$ ugrást igényel nagy valószínűséggel.
Kulcs befűzése és törlése $O(\log n)$ üzenetcserét igényel nagy valószínűséggel.

Keresőalgorithmus k kulcshoz:

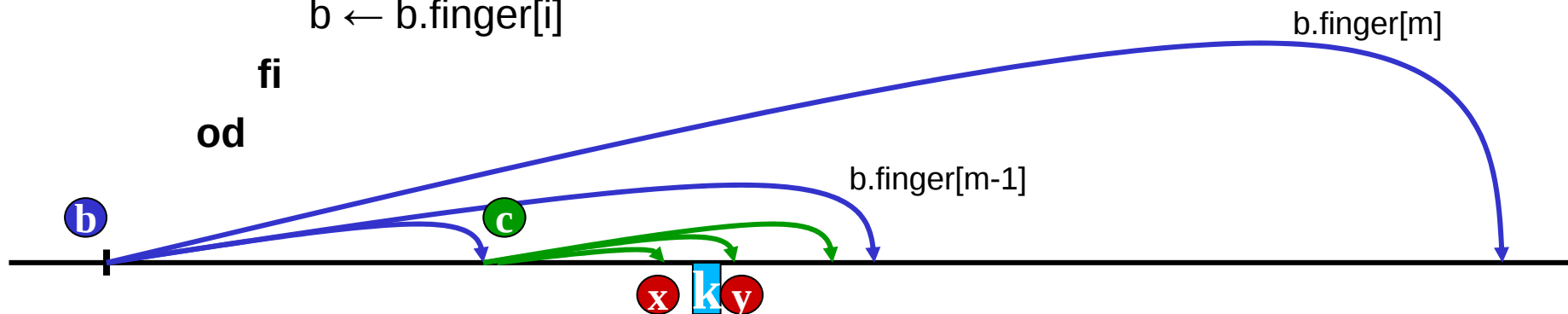
- Főrutin: Kezdjünk egy tetszőleges b csomópontnál

while not $r_k(k) \in [r_v(b), r_v(b.successor))$ **do**

for $i=m-1$ **downto** 0 **do**

if $r_k(k) \in [r_v(b.finger[i]), r_v(finger[i+1]))$ **then**

$b \leftarrow b.finger[i]$



Egyensúly a Chord-ban

- n : csomópontok száma a P2P-hálózatban
- k : kulcsok száma ≥ 1

Tétel 2:

A Chord adatstruktúra a következő tulajdonságokkal rendelkezik:

- 1. Egyensúly és Terhelés:** nagy valószínűséggel ($1-n^{-c}$) minden csomópont legfeljebb $O(\frac{k \log n}{n})$ kulcsot tárol.
- 2. Dinamika:** Ha egy új csomópont kapcsolódik a hálózathoz, vagy egy csomópont elhagyja a hálózatot, nagy valószínűséggel legfeljebb $O(\frac{k \log n}{n})$ kulcsot kell mozgatni (a szomszéd csomóponton tárolni).

Biz.:

- ...

A Cord adatstruktúra tulajdonságai

Lemma 1:

A $r_v(b.succ) - r_v(b) \bmod 2^m$ távolság

1. várható értéke $2^m/n$,
2. nagy valószínűséggel legfeljebb $O((2^m/n) \log n)$ és
3. nagy valószínűséggel legalább $(2^m/n) / n^c$ egy konstans $c > 0$ -ra.
4. Egy h $2^m/n$ hosszú intervallumban a csomópontok száma nagy valószínűséggel
 - a) $\Theta(h)$, ha $h = \Omega(\log n)$
 - b) legfeljebb $O(h \log n)$, ha $h = o(\log n)$

Lemma 2:

Azon csomópontok száma, melyek egy finger-mutatója b Peer-re mutat

1. várható érték $O(\log n)$,
2. nagy valószínűséggel legfeljebb $O(\log^2 n)$

Bizonyítások?

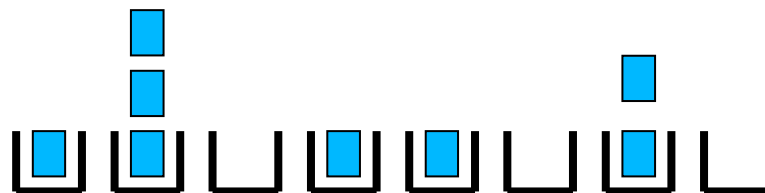
- Hogy lehet ezeket a tulajdonságokat bebizonyítani
- Abstrakt modell: labdák és kosarak (bins and balls)
- Chernoff-egyenlőtlenség felhasználása

Labdák és kosarak

Lemma 3:

Ha m labdát véletlenül n kosárba dobunk, akkor:

1. A labdák számának várható értéke egy kosaranként m/n .
2. Annak a valószínűsége, hogy k labda esik egy bizonyos kosárba: $\binom{m}{k} \left(\frac{1}{n}\right)^k \left(1 - \frac{1}{n}\right)^{m-k}$.



Lemma 4:

Ha $m=n$ labdát véletlenül n kosárba dobunk, akkor:

1. Annak a valószínűsége, hogy egy bizonyos kosár üres marad, konstans (konvergál $1/e$ -hez). Az üres kosarak számának várható értéke konvergál n/e -hez.
2. Annak a valószínűsége, hogy több mint $k \ln n$ labda esik egy bizonyos kosárba, legfeljebb $O(n^{-c})$ egy konstans k -ra és c -re.

Biz.: 1.: Lemma 3, 2. pontja szerint: $\binom{n}{0} \left(\frac{1}{n}\right)^0 \left(1 - \frac{1}{n}\right)^n = \left(1 - \frac{1}{n}\right)^n \approx \frac{1}{e}$.

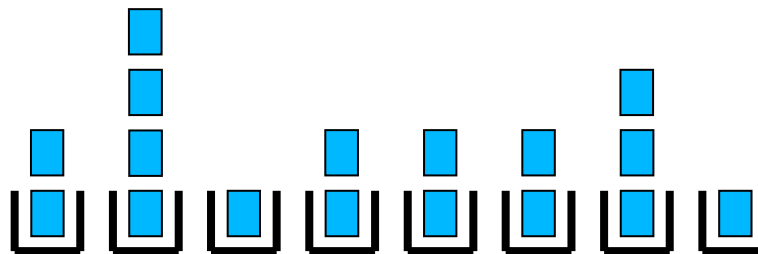
2. Chernoff-korlát

Labdák és kosarak

Lemma 5

Legyen $k > 0$ egy fix konstans. Ha $m = k n \log n$ labdát véletlenül n kosárba dobunk, akkor a következő érvényes:

1. Minden $c > k$ –ra annak a valószínűsége, hogy több mint $c \log n$ labda esik egy kosárba legfeljebb $O(n^{-c'})$, ahol $c' > 0$ egy konstans.
2. Minden $c < k$ –ra annak a valószínűsége, hogy egy kosárba kevesebb mint $c \log n$ labda esik, legfeljebb $O(n^{-c'})$, ahol $c' > 0$ egy konstans.



Biz.:

Chernoff-korlát

Kitérés: Valószínűségszámítás – Markov egyenlőtlenség

- Egy diszkrét valósz. változó X várható értéke: $\mathbf{E}[X] = \sum_{x \in \mathbb{R}} \mathbf{P}[X = x]x$
- **Markov egyenlőtlenség:** Legyen $k > 0$. Egy $X \geq 0$ diszkrét valósz. változóra, amire $\mathbf{E}[X] > 0$:

$$\mathbf{P}[X \geq k \cdot \mathbf{E}[X]] \leq \frac{1}{k}$$

- **Biz.:**
$$\begin{aligned} \mathbf{E}[X] &= \sum_{x \in \mathbb{R}} \mathbf{P}[X = x]x \\ &\geq \sum_{x \geq k\mathbf{E}[X]} \mathbf{P}[X = x]x \\ &\geq \sum_{x \geq k\mathbf{E}[X]} \mathbf{P}[X = x] \cdot k \cdot \mathbf{E}[X] \\ &= k \cdot \mathbf{E}[X] \cdot \sum_{x \geq k\mathbf{E}[X]} \mathbf{P}[X = x] = k \cdot \mathbf{E}[X] \cdot \mathbf{P}[X \geq k \cdot \mathbf{E}[X]]. \end{aligned}$$

Chebyshev egyenlőtlenség

- Erősebb korlát: **Chebyshev egyenlőtlenség** :

$$P[|X - \mathbf{E}[X]| \geq k] \leq \frac{V[X]}{k^2}$$

Ha a szórás (variance) ismert:

$$V[X] := \mathbf{E}[(X - \mathbf{E}[X])^2]$$

Chernoff egyenlőtlenség

- Bernoulli kísérlet
 - p valószínűséggel 1
 - 1-p valószínűséggel 0
- **Tétel 3. Chernoff egyenlőtlenség :**
Legyenek $x_1, \dots, x_n$ független Bernoulli kísérletek, melyekre

$$P[x_i = 1] = p, \quad P[x_i = 0] = 1 - p.$$

Legyen $S_n = \sum_{i=1}^n x_i$.

Ekkor:

1. $c > 0$: $P[S_n \geq (1 + c)E[S_n]] \leq e^{-\frac{\min\{c, c^2\}}{3}pn}.$
2. $c \in [0, 1]$: $P[S_n \leq (1 - c)E[S_n]] \leq e^{-\frac{c^2}{2}pn}.$

A Chord adatstruktúrájának tulajdonságai

Lemma 6

1. Két szomszédos Peer p és q közötti távolság a gyűrűn (azaz $r_V(p) - r_V(q) \bmod 2^m$)
 - a) várható érték: $2^m/n$,
 - b) $< O((2^m/n) \log n)$ (nagy valószínűséggel) és
 - c) $> 2^m/n^c$ (nagy valószínűséggel) egy konstans $c > 1$ -re
2. A gyűrű egy $h \cdot 2^m/n$ hosszú intervallumba (nagy valószínűséggel)
 - a) $O(\log n + h \log n)$ Peer esik, ha $h = O(\log n)$
 - b) $O(h)$ Peer esik, ha $h = \Omega(\log n)$

A Chord adatstruktúrájának tulajdonságai

Lemma 6

1. Két szomszédos Peer p és q közötti távolság a gyűrűn

- a) várható érték: $2^m/n$,
- b) $< O((2^m/n) \log n)$ (nagy valószínűséggel) és
- c) $> 2^m/n^c$ (nagy valószínűséggel) egy konstans $c > 1$ -re

Biz.:

1a) Az összes távolság összege 2^m , a Peerek száma n

1b) Tekintsünk egy $c \log n \cdot 2^m / n$ hosszú intervallumot a gyűrűn

- A valószínűség, hogy egy Peer ebbe az intervallumba esik: $c (\log n)/n$
- A valószínűség, hogy minden n Peer kívül esik az intervallumon:

$$\left(1 - \frac{c \ln n}{n}\right)^n \leq e^{-c \ln n} = n^{-c}$$

- Tehát egy ilyen intervallum nem marad üres és így a távolság két szomszédos Peer között nagy valószínűséggel $\leq 2c \log n \cdot 2^m / n$

1c) A valószínűség, hogy egy Peer egy adott $2^m/n^c$ hosszú intervallumba esik n^{-c}

- Tehát a Peer-ek nagy valószínűséggel nem esnek túl közel egy másik Peer-hez

A Chord adatstruktúrájának tulajdonságai

Lemma 6

2. A gyűrű egy $h \cdot 2^m/n$ hosszú intervallumába (nagy valószínűséggel)

- a) $O(\log n + h \log n)$ Peer esik, ha $h=O(\log n)$
- b) $O(h)$ Peer esik, ha $h=\Omega(\log n)$

Biz.: (Chernoff korláttal). Tekintsünk egy $h \cdot 2^m/n$ hosszú intervallumot

- A valószínűség, hogy egy Peer ebbe esik: $p = h / n$
- A Peer-ek számának várható értéke az intervallumban: $p \cdot n = h$

2a) 1.eset: $p \cdot n \geq 1, c > 1$: $P[X \geq (1 + c \ln n)pn] \leq e^{-\frac{1}{3}(c \ln n)pn} \leq n^{-\frac{1}{3}c}$

2.eset: $p \cdot n < 1, c > 1$: $P[X \geq pn + c \ln n] = P[X \geq (1 + \frac{c \ln n}{pn})pn] \leq e^{-\frac{1}{3} \frac{c \ln n}{pn} pn} \leq n^{-\frac{1}{3}c}$

2b) $p \cdot n > k \ln n, c > 1$: $P[X \geq (1 + c)pn] \leq e^{-\frac{1}{3}ck \ln n} = n^{-\frac{1}{3}ck}$

Egyensúly a Chord-ban

- n : Peer-ek száma a P2P hálózatban
- k : kulcsok száma ≥ 1 (a tárolt adatok kulcsainak száma)

Tétel 4.

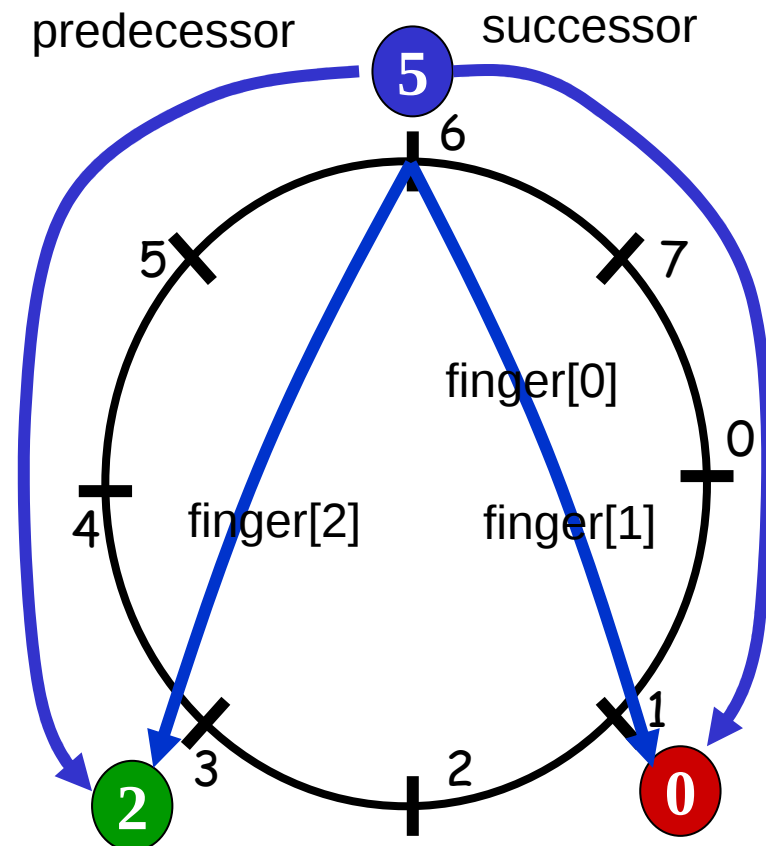
Az elemek eloszlására a Peer-eken a következő igaz:

- Ha $k=O(n \log n)$:
Minden csomópont legfeljebb $O(\log n + k/n \log^2 n)$ kulcsot tárol nagy valószínűséggel
- Ha $k=\Omega(n \log n)$:
Minden csomópont legfeljebb $O(k/n \log n)$ kulcsot tárol nagy valószínűséggel
- **Biz.:**
 - Chernoff korlát

Chord adatstruktúrája

Minden b csomóponthoz tároljuk:

- **successor**: következő csomópont a gyűrűn
- **predecessor**: megelőző csomópont
- minden $i \in \{0, \dots, m-1\}$ -re
 - az i -edik ujj: **finger[i]**: az a csomópont, amely képe $r_v(b) + 2^i \bmod 2^m$ értékét követi, azaz legalább $r_v(b) + 2^i \bmod 2^m$ és azok között a legkisebb
 - jelölje $\text{finger}[m] := b$
- Kicsi i esetén a finger-bejegyzések gyakran azonosak
 - csak különböző finger-bejegyzéseket tárolunk



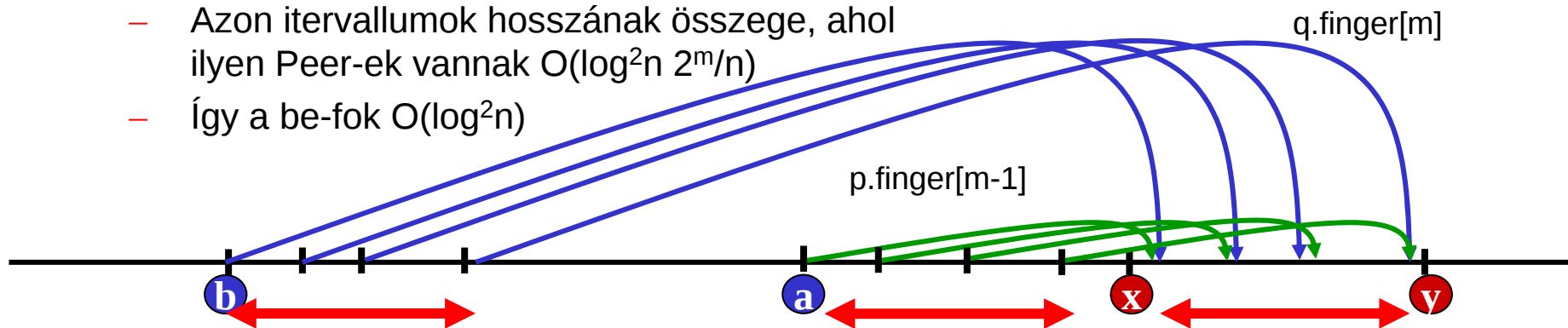
Az ujjak száma

Lemma 7

1. A CHORD hálózatban a ki-fok $O(\log n)$ nagy valószínűséggel
2. A CHORD hálózatban a be-fok $O(\log^2 n)$ nagy valószínűséggel

Biz.:

1. A minimális távolság két Peer között $2^m/n^c$ (nagy valószínűséggel)
 - Így a ki-fok legfeljebb $c \log n$ (nagy valószínűséggel)
2. A maximális távolság két szomszédos Peer x, y között $O(\log n \cdot 2^m/n)$
 - Minden p Peer, amelynek egy ujj $p.\text{finger}[i]$ erre az intervallumra mutat, növeli y be-fokát egyel.
 - Azon intervallumok hosszának összege, ahol ilyen Peer-ek vannak $O(\log^2 n \cdot 2^m/n)$
 - Így a be-fok $O(\log^2 n)$



Keresés a Chord-ban

Tétel 5

A keresés nagy valószínűséggel $O(\log n)$ ugrást tartalmaz

Kereső algoritmus s kulccsal:

- Főrutin: Legyen b egy csomópont (a keresés elindítója)

while not $r_K(s) \in [r_V(b), r_V(b.successor))$ do

for $i=m-1$ downto 0 do

if $r_K(s) \in [r_V(b.finger[i]), r_V(finger[i+1]))$ then

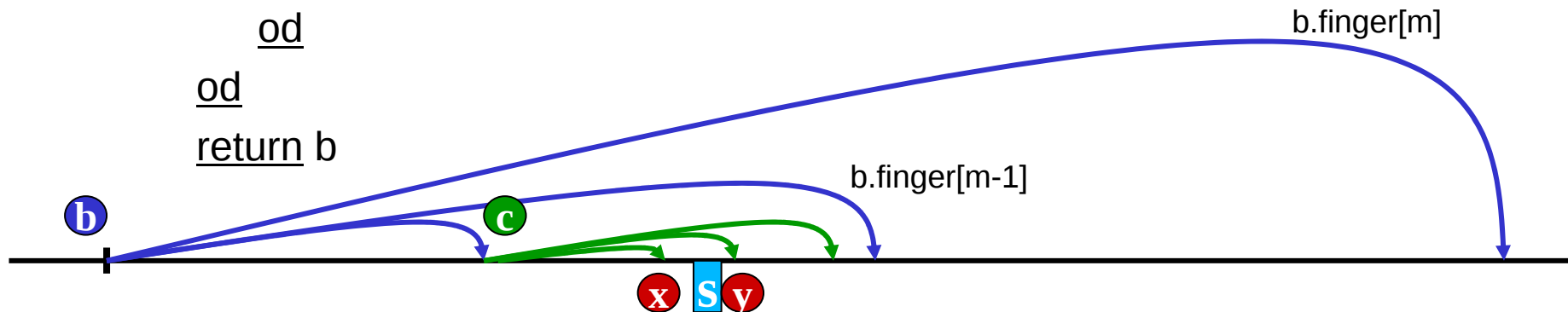
$b \leftarrow b.finger[i]$

fi

od

od

return b



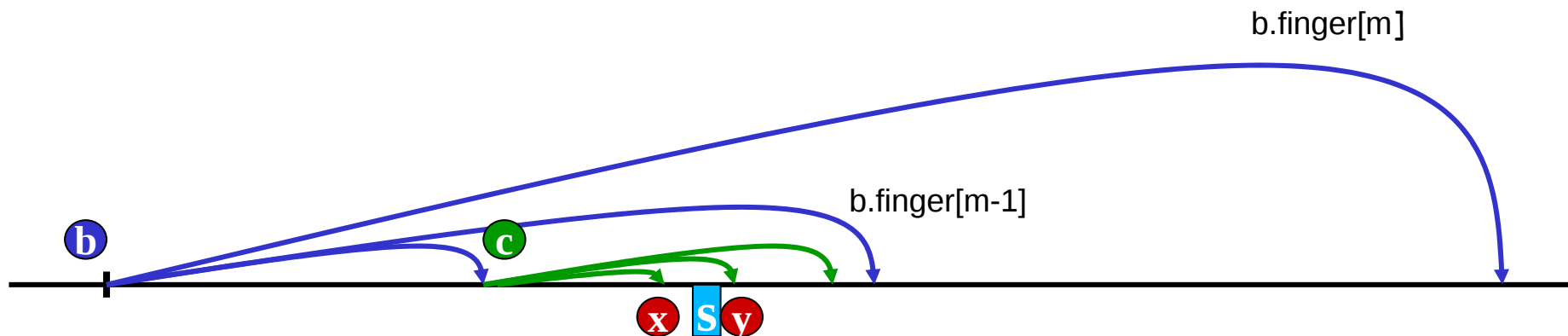
Keresés a Chord-ban

Tétel 5

A keresés nagy valószínűséggel $O(\log n)$ ugrást tartalmaz

Biz.:

- A távolság a célhoz minden ugrással legalább feleződik
- A távolság a keresés legelején legfeljebb 2^m
- A távolság két szomszédos Peer között legalább $2^m/n^c$ nagy valószínűséggel
- Így a keresés ideje legfeljebb $c \log n$



Peer-ek hozzáadása

Tétel 6

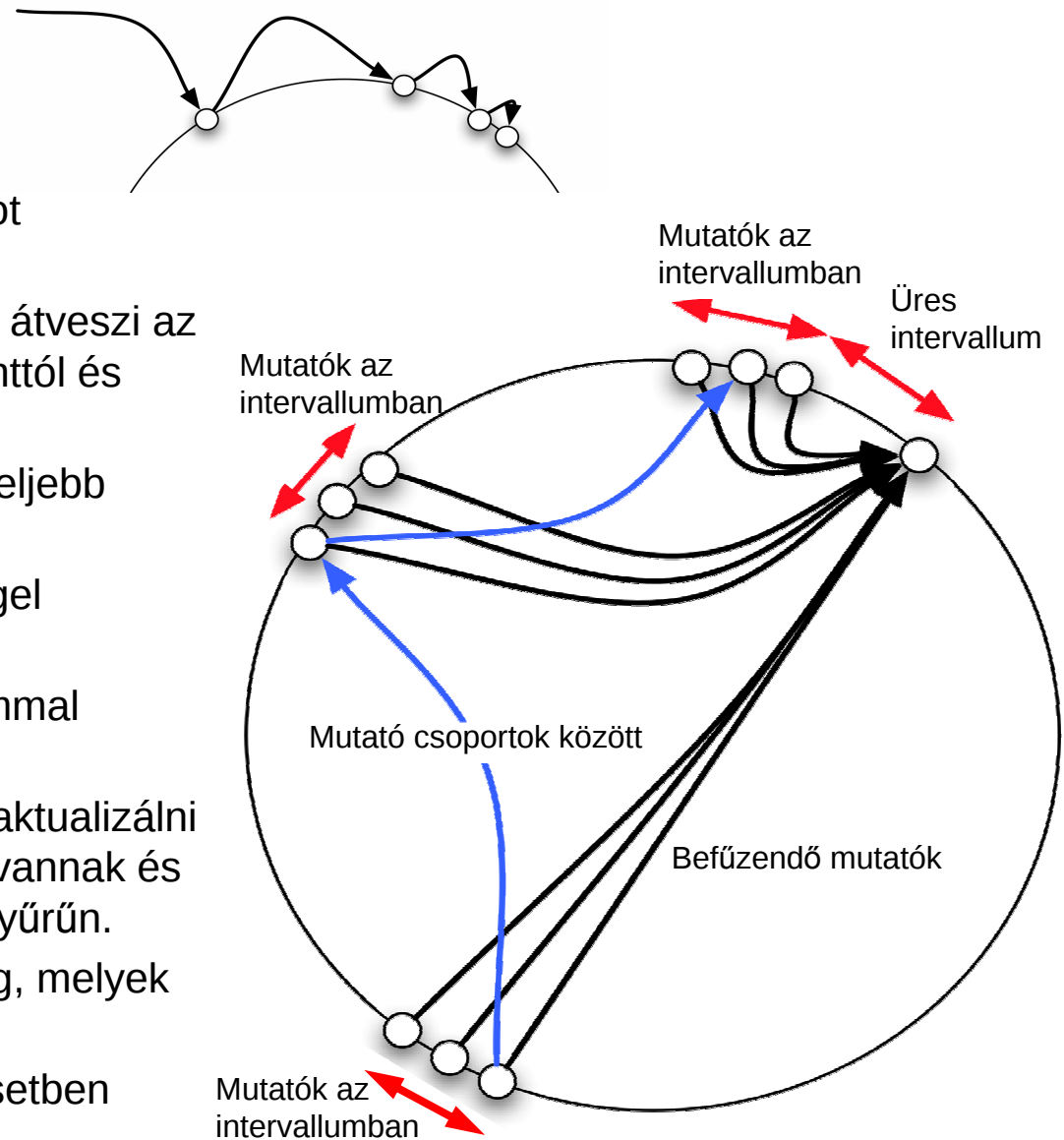
Nagy valószínűséggel $O(\log^2 n)$ üzenet elegendő egy új Peer-t a Chord-ba felvenni

Bizonyítási ötlet:

- A cél-intervallumot megkeressük $O(\log n)$ lépésben
- A mutató (finger) információkat átvesszi az új Peer az őt megelőző és az őt követő csomóponttól és ezeket aktualizálja
 - Minden ilyen mutatót a gyűrűn legfeljebb $O(\log n)$ lépésben aktualizálhatunk
- Az új Peer p be-foka nagy valószínűséggel $O(\log^2 n)$
 - A keresés költsége minden alkalommal $O(\log n)$
 - A Peer-ek, ahol a $\text{finger}[i]$ mutatót aktualizálni kell p -re, maximum $O(\log n)$ sokan vannak és ezek egymással szomszédosak a gyűrűn
 - Így $O(\log n)$ keresésre van szükség, melyek mindegyikének költsége $O(\log n)$
 - Az aktualizálás költsége minden esetben konstans

Peer-ek hozzáadása

- Először megkeressük a célintervallumot $O(\log n)$ lépésben
- Az új Peer a kimenő mutatókat (finger) átveszi az öt megelőző és az öt követő csomóponttól és ezeket aktualizálja
 - Minden ilyen mutatót a gyűrűn legfeljebb $O(\log n)$ lépésben aktualizálhatunk
- Az új Peer be-foka nagy valószínűséggel $O(\log^2 n)$
 - A keresés költsége minden alkalommal $O(\log n)$
 - A Peer-ek, ahol a $\text{finger}[i]$ mutatót aktualizálni kell p -re, maximum $O(\log n)$ sokan vannak és ezek egymással szomszédosak a gyűrűn.
 - Így $O(\log n)$ keresésre van szükség, melyek mindegyikének költsége $O(\log n)$
 - Az aktualizálás költsége minden esetben konstans



Belépés és kilépés várható költsége

Tétel 7

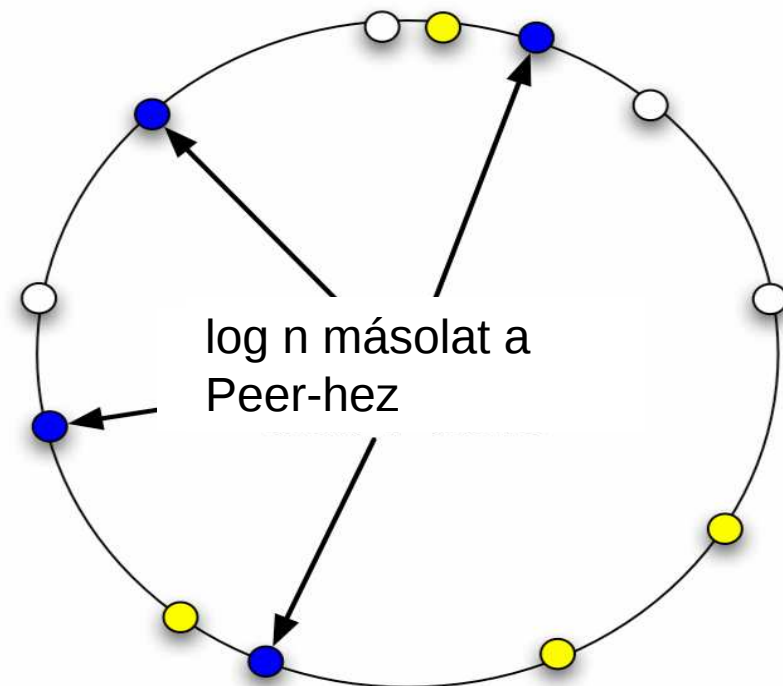
Egy Peer belépéséhez szükséges üzenetek számának várható értéke $O(\log n)$

Bizonyítási ötlet:

- Minden Peer ki-foka nagy valószínűséggel $O(\log n)$
 - Ezáltal a be-fok várható értéke ugyancsak $O(\log n)$
 - Az ugrások számának a várható értéke a belépés-operációban $O(1)$
-
- Egy Peer kilépésének (törlésének) költsége ugyanakkora

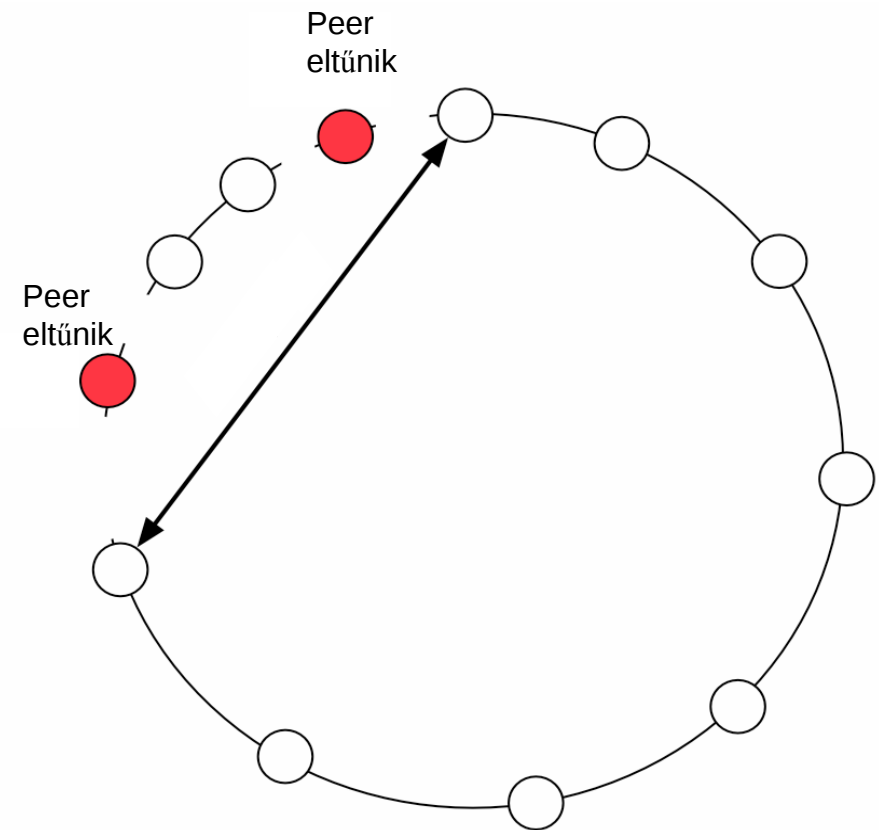
Egyensúly virtuális Peer-ek segítségével

- Minden Peer tart $O(\log n)$ virtuális Peer-t a gyűrűn
 - Ezáltal a ki-fok megnő $O(\log^2 n)$ -re
- Előnyök:
 - A be-fok továbbra is $O(\log^2 n)$ nagy valószínűséggel
 - Belépés/Törlés-Operáció végrehajtható $O(\log^2 n)$ üzenettel nagy valószínűséggel
 - Ha az adatok száma $k = \Omega(n \log n)$, akkor minden Peer nagy valószínűséggel $O(k/n)$ adatot tárol



Stabilizálás a CHORD-ban

- Csomópontok belépése és törlése párhuzamosan történhet
- Ha csomópontok magukat törlik (eltűnnek), nem értesítik a szomszédjaikat
 - A csomópontoknak rendszeresen tesztelni kell, hogy a szomszédai jelenvannak-e
 - Ha a szomszéd eltűnt, a finger-információ segítségével keresünk új szomszédot
 - Ha két csomópont egyidejűleg tűnik el, a gyűrű széteshet két részre és ez inkonzisztenciához vezethet
- Egyidejű belépés szintén inkonzisztenciához vezethet az adatstruktúrában
- Megoldás: speciális stabilizálás-operáció segítségével
 - Ötlet: egy kiesett csomópont szomszédainak a finger-információi „hasznosak” a kiesett Peer-éhez (nem kerül pontosabb bemutatásra)



Irodalom

- I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan: **Chord: A scalable peer-to-peer lookup service for Internet applications.** In *Proc. ACM SIGCOMM*, 149-160, 2001.