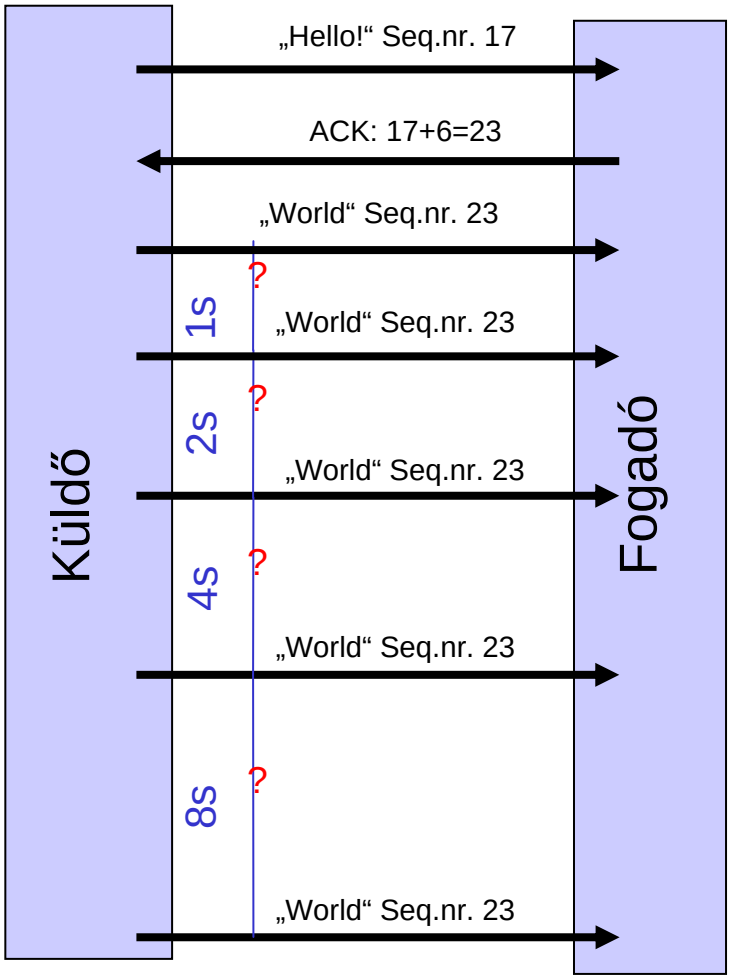
# Hálózatok II
# 2006

## 8. Szállítói réteg – TCP
## Tahoe, Reno, AIMD, hatékonyság, fairness

# Exponenciális visszavétel (exponential backoff)

- Retransmission Timout (RTO)
  - szabályozza az időközt a küldés és egy duplikátum újraküldése között, ha egy nyugta kimarad
- Mikor nem kerül nyugtázásra egy TCP-csomag?
  - Ha a nyugta lényegesen több időt vesz igénybe, mint az átlagos „round trip time” (RTT)
    - 1. Probléma: RTT mérése
    - 2. Probléma: Csak a nyugta jön túl későn
  - Küldő
    - Vár az RTO-nak megfelő ideig
    - Ha nem érkezett nyugta, újraküldi a csomagot és növeli

      RTO ← 2 RTO (RTO = 64 másodpercig)
- RTO újraszámolása, ha a csomagok nyugtázódnak



# TCP – Nagle algoritmusa

- Hogyan biztosíthatjuk,
  - hogy kis csomagok időben egymáshoz közel kerüljenek kiszállításra
  - és hogy sok adat esetén nagy csomagok előnyben részesüljenek?

- **Nagle algorithmusa:**
  - Kis csomagok nem kerülnek addig küldésre, amig nyugták hiányoznak
    - egy csomag kicsi, ha az adathossz < MSS
  - Ha a korábban küldött csomag nyugtája megérkezik, küldi a következőt
- Tulajdonsága
  - Önmagát ütemező: Gyors kapcsolat = sok kis csomag

# A Round Trip Time (RTT) becslése

- A TCP-csomag nem nyugtázottnak számít, ha a nyugta „lényegesen” tovább tart, mint az RTO
- RTT nem számítható on-line (csak visszatekintve)
- RTT erősen ingadozik
- Ezért: Retransmission Timeout Value nagyvonalú becsléssel:
  - RFC 793: (M := utoljára mért RTT)

    $R \leftarrow \alpha R + (1- \alpha) M,$ ahol $\alpha = 0{,}9$

    $RTO \leftarrow \beta R,$ ahol $\beta = 2$
  - Jacobson 88: a becslés nem elég robosztus, ezért

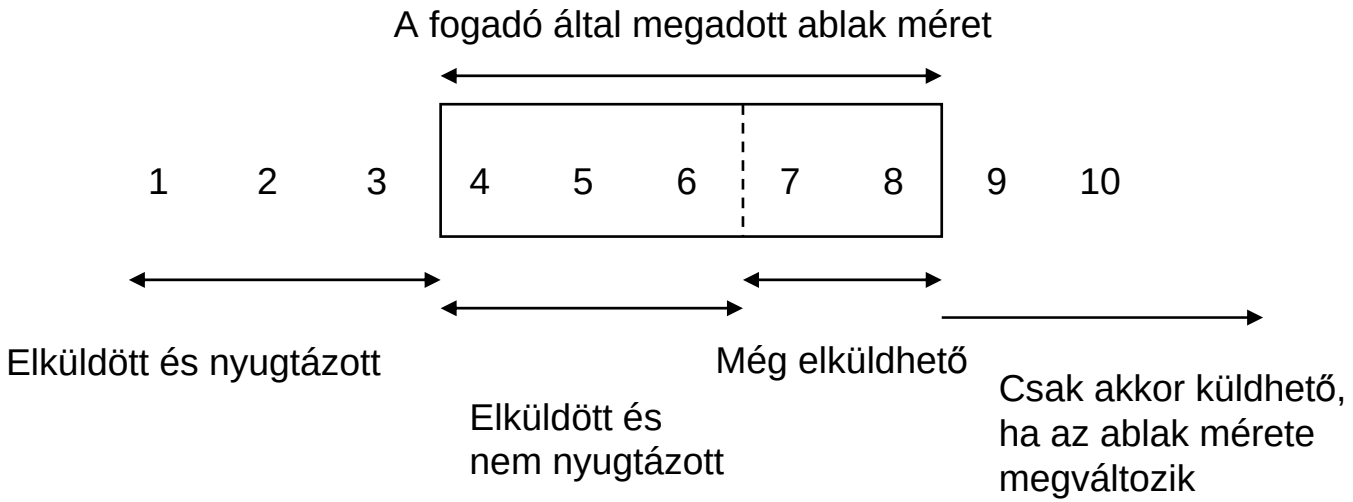
    $A \leftarrow A + g (M - A)$ , ahol $g = 1/8$

    $D \leftarrow D + h (|M - A| - D)$ , ahol $h = 1/4$

    $RTO \leftarrow A + 4D$
- Többszörösen elküldött csomagoknál nem aktualizálunk

# Csúszó Ablakok (sliding windows)

- Adatátráta szabályozása ablak segítségével
  - A fogadó meghatározza az ablak méretet (wnd) az ACK-szegmensek TCP-fejlécében
  - Ha a fogadó fogadási puffere tele van, akkor wnd=0 -t küld
  - Máskülönben a fogadó wnd>0 -t küld
- A küldőnek be kell tartani:
  - Az elküldött nem nyugtázott szegmensek száma ≤ ablak mérete



# Lassú Start (slow start)

- A küldőnek nem szabad a fogadó által felajánlott ablakméretet azonnal kihasználni
- Második ablak: Congestion-ablak (cwnd: congestion window)
  - A küldő választja (FSK)
  - Az ablak amiben küld: min {wnd,cwnd}
  - Kezdetben:

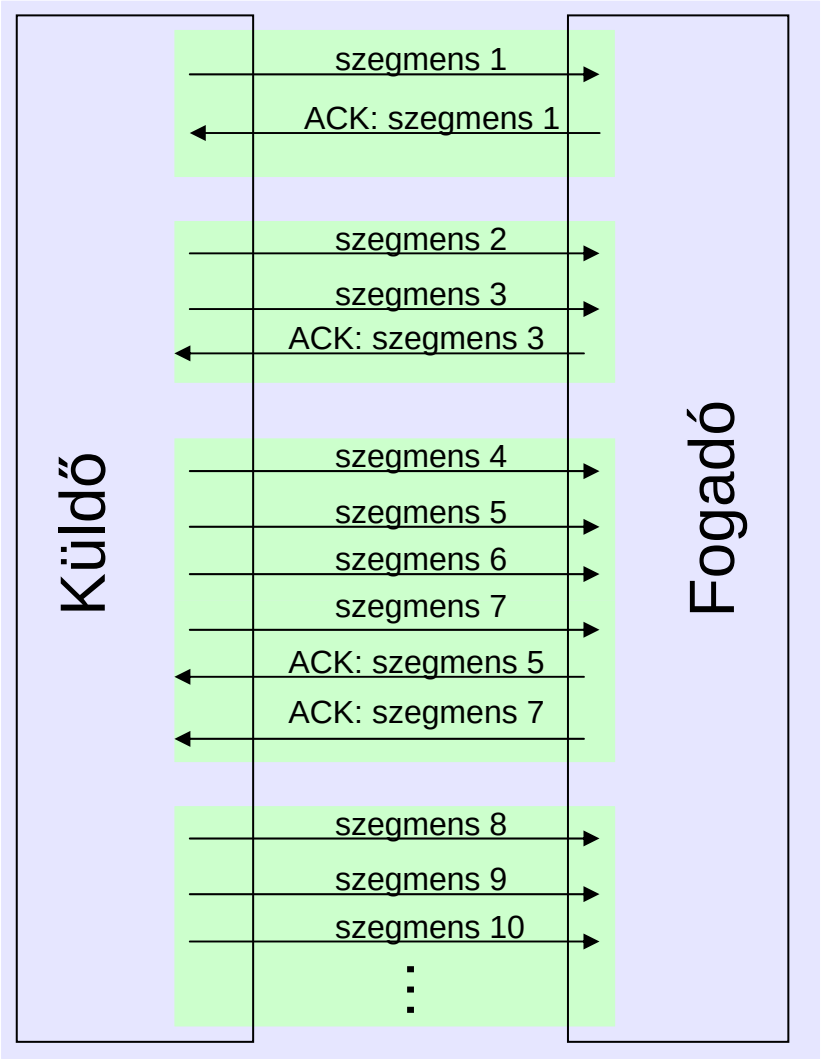
    $$cwnd \leftarrow MSS$$

  - Minden csomagnál a megkapott nyugta után nő

    $$cwnd \leftarrow cwnd + MSS$$

    (azaz megduplázódik minden RTT után)
  - Addig, amíg egyszer egy nyugta kimarad

Slow start = exponenciális növekedés
(hisztórikus elnevezés: korábban még aggresszívebb sémák)



# Torlódás elkerülés (congestion avoidance) TCP Tahoe

- Jacobson 88:
  - Paraméter: cwnd és ssthresh (= slow-start-küszöb, slow start threshold)

1. Kapcsolatfelépítés:

   cwnd ← MSS  ssthresh ← 65535

2. Csomagvesztésnél, azaz nyugta ideje > RTO: multiplicatively decreasing

   cwnd ← MSS  ssthresh ← $\max\left\{2\ \mathrm{MSS}, \frac{\min\{cwnd, wnd\}}{2}\right\}$

3. Nyugta jön a szegmenshez és cwnd ≤ ssthresh: slow start

   cwnd ← cwnd + MSS

4. Nyugta jön a szegmenshez és cwnd > ssthresh: additively increasing

   cwnd ← cwnd + MSS $\frac{\mathrm{MSS}}{cwnd}$

# TCP Tahoe

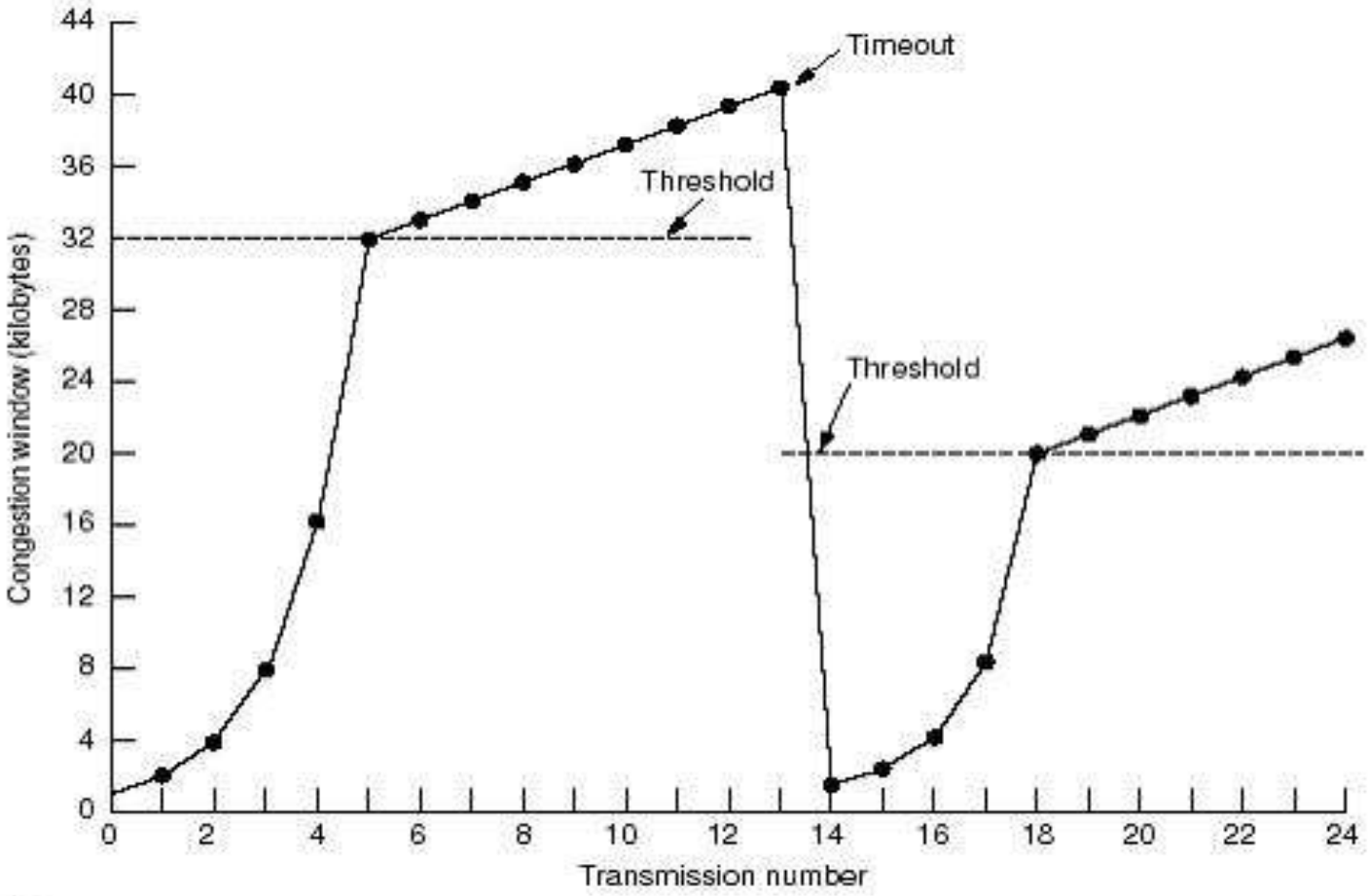



Fig3

pictures from TANENBAUM A. S. *Computer Networks 3rd edition*

# Fast Retransmit és Fast Recovery

- TCP Tahoe [Jacobson 1988]:
  - Ha csak egy csomag veszik el, akkor
    - a csomagot + a fennmaradó ablakot újraküldi
    - és egyidejűleg slow start
  - Fast retransmit
    - ha ugyanazon csomaghoz 3 nyugta-duplikátum (azaz 4 azonos nyugta) érkezik (triple duplicate ACK),
    - újraküldi az elveszet csomagot, egyidejűleg slow start
- TCP Reno [Stevens 1994]
  - Fast retransmit után:
    - ssthresh ← max( min(wnd,cwnd)/2, 2 MSS )
    - cwnd ← ssthresh + 3 MMS
  - Fast recovery a fast retransmit után
    - Miden további nyugta-duplikátum után növeli a
    - cwnd ← cwnd + MMS
  - Congestion avoidance: amikor új adat nyugtája megérkezik (ujraküldés ACK-ja) :
    - cwnd ← ssthresh

# Torlódás elkerülési elv: AIMD

- A TCP a „fast recovery” mechanizmussal lényegében a következőképp viselkedik:

  x: csomagok száma per RTT

- Kapcsolatfelépítés:

  **x ← 1**

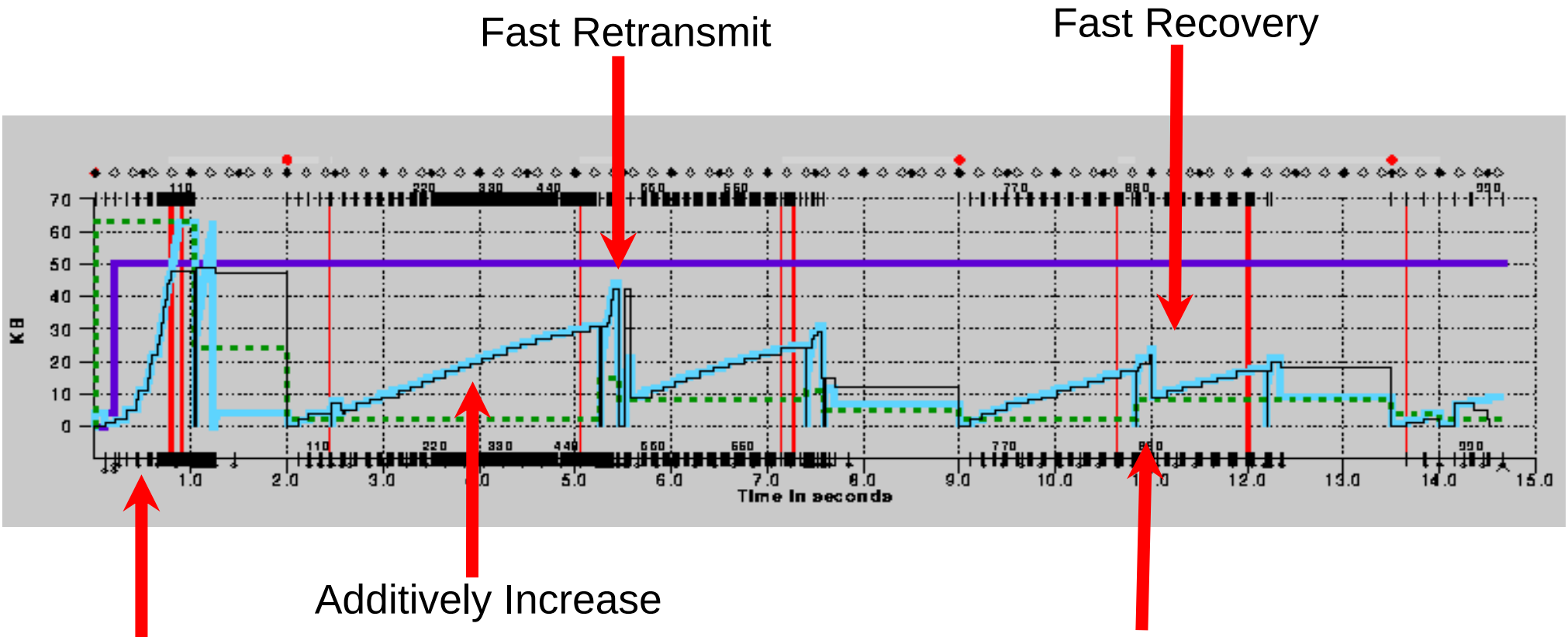
- Csomagvesztésnél, MD: multiplicative decreasing

  **x ← x/2**

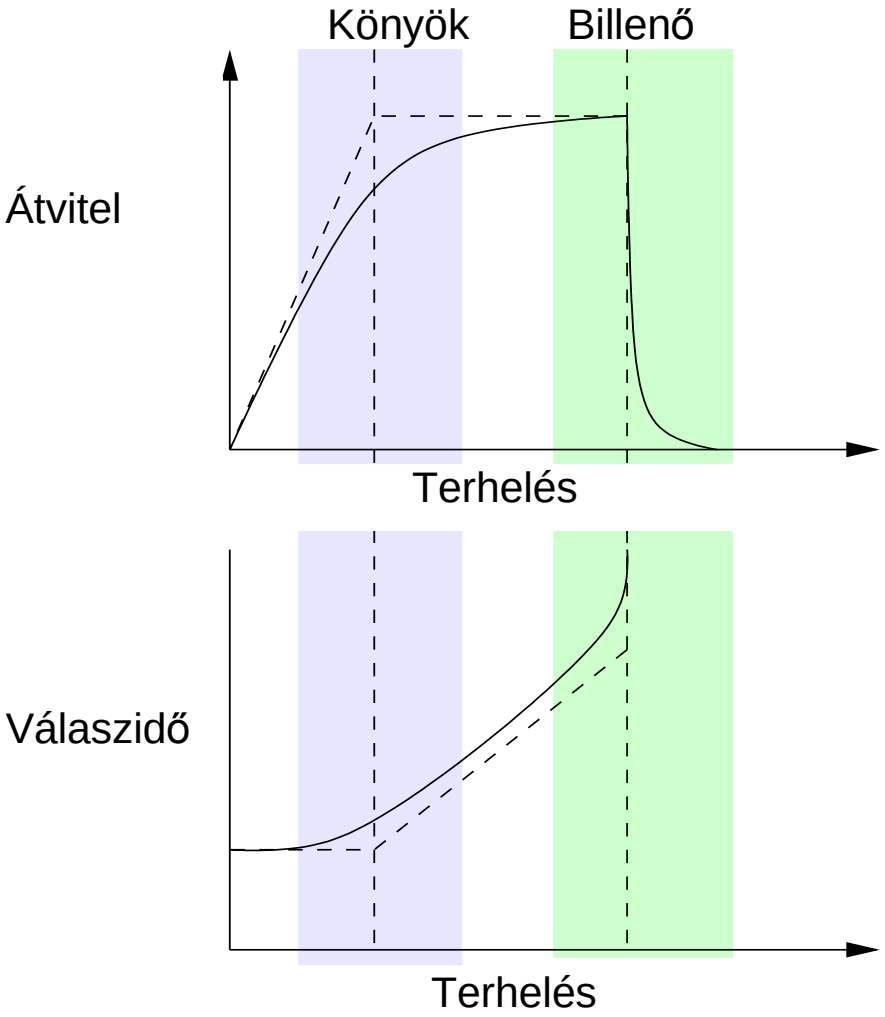
- Nyugtázott szegmenseknél, AD: additive increasing

  **x ← x +1**

# Példa: TCP Reno „in aktion”



# Additive Increase Multiplicative Decrease (AIMD): Fairness és Hatékonyság




A hálózati terhelés az átvitellel és a válasziővel kölcsönösen hat egymásra.

- Az átvitel maximális, ha a terhelés a hálózat kapacitását majdnem eléri.
- Ha a terhelés tovább nő, túlcsordulnak a pufferek, csomagok vesznek el, újra kell küldeni, drasztikusan nő a válaszidő. Ezt a toródást congestion-nak nevezzük.
- Ezért a maximális terhelés helyett, ajánlatos a hálózat terhelését a könyök közelében beállítani. Itt a válaszidő csak lassan emelkedik, míg az adatátvitel már a maximum közelében van.
- Egy jó torlódáselkerülési (*congestion avoidance*) stratégia a hálózat terkelését a könyök közlében tartja: hatékonyság. Emellett fontos, hogy minden résztvevőt egyforma rátával szolgáljunk ki: fairness.

# AIMD Fairness és Hatékonyság – A Modell

- n résztvevő, forduló-modell
- résztvevő i adatrátája a t-eik fordulóban $x_i(t)$
- Kezedeti adatráták: $x_1(0)$, …, $x_n(0)$
- A visszacsatolás (feedback) forduló t után: y(t) = 0, ha $\sum_{i=1}^{n} x_i(t) \leq K$

  y(t) = 1, ha $\sum_{i=1}^{n} x_i(t) > K$
- Minden résztvevő aktualizálja az adatrátáját a t+1-edik fordulóban:

$$x_i(t+1) = f(x_i(t),y(t))$$

  - Increase-stratégia $f_0(x) = f(x,0)$
  - Decrease-stratégia $f_1(x) = f(x,1)$

- Tekintsük a következő lineáris függvényeket :

$$f_0(x) = a_I + b_I x, \qquad f_1(x) = a_D + b_D x$$

# AIMD Fairness és Hatékonyság– A Modell

- A következő lineáris függvényeket vizsgáljuk:

$$f_0(x) = a_I + b_I x, \qquad f_1(x) = a_D + b_D x$$

- Érdekes speciális esetek:
  - MIMD: Multiplicative Increase/Multiplicative Decrease

    $f_0(x) = b_I x, \quad f_1(x) = b_D x, \quad$ ahol $\quad b_I > 1,\ b_D < 1.$

  - AIAD: Additive Increase/Additive Decrease

    $f_0(x) = a_I + x, \quad f_1(x) = a_D + x, \quad$ ahol $\quad a_I > 0,\ a_D < 0.$

  - AIMD: Additive Increase/Multiplicative Decrease

    $f_0(x) = a_I + x, \quad f_1(x) = b_D x, \quad$ ahol $\quad a_I > 0,\ b_D < 1.$

# AIMD Fairness és Hatékonyság

- Hatékonyság
  - Terhelés: $X(t) := \sum_{i=1}^{n} x_i(t)$
  - Mérték: $|X(t) - K|$

- Fairness: x=($x_1$, …, $x_n$) esetén:

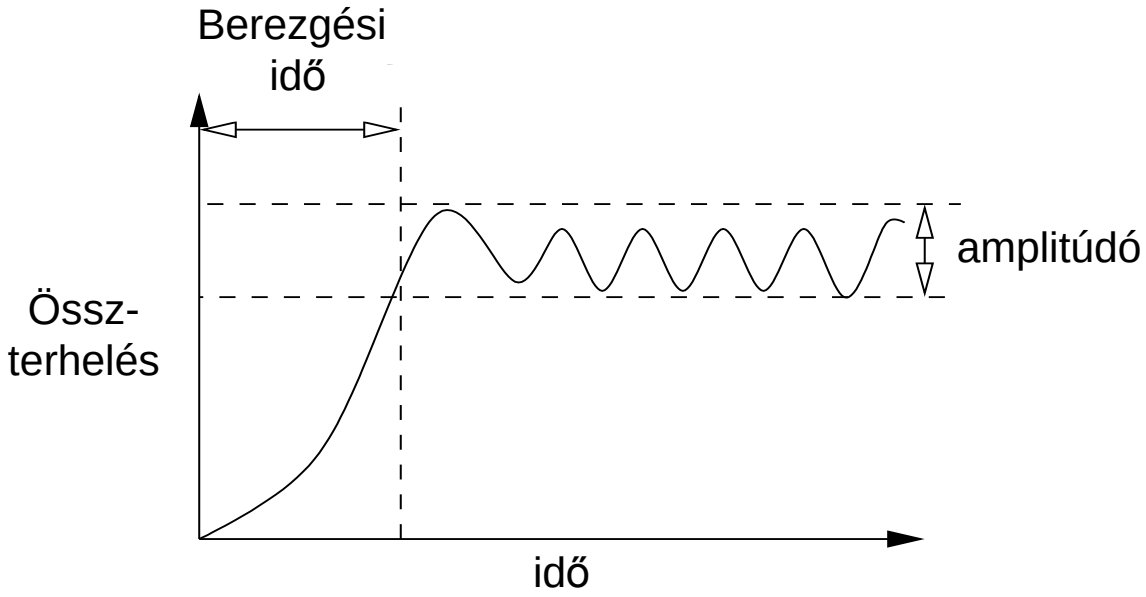
$$F(x) = \frac{(\sum_{i=1}^{n} x_i)^2}{n \sum_{i=1}^{n} (x_i)^2} .$$

  - 1/n ≤ F(x) ≤ 1
  - F(x) = 1 ↔ absolut Fairness
  - skálázástól független
  - Folytonos, differenciálható
  - Ha n közül k fair, a többi 0, akkor F(x) = k/n

# Konvergencia

- Konvergencia nem lehetséges
- Legjobb esetben oszcilláció az optimális érték körül
  - Az oszcilláció amplitúdója A

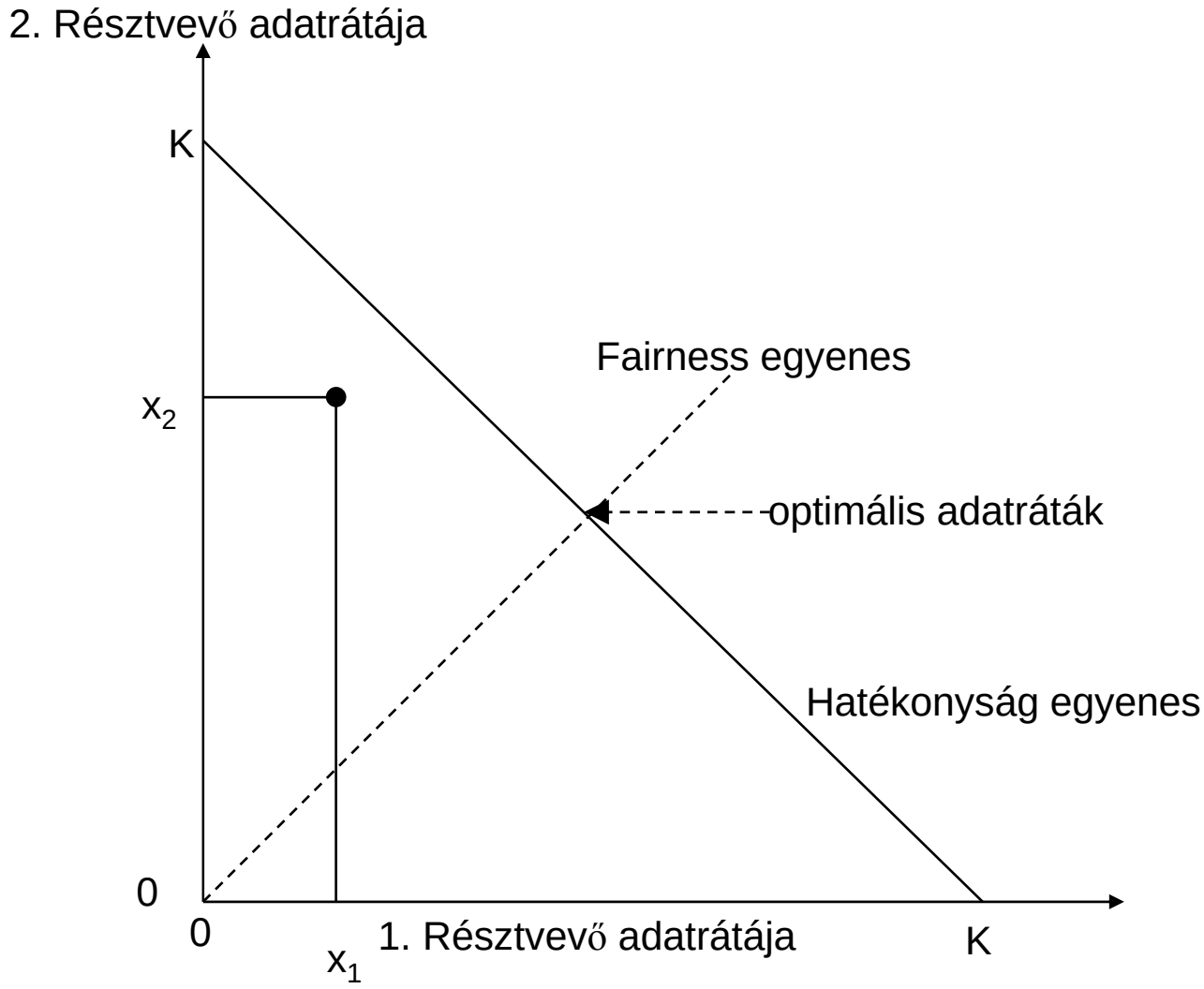
$$A = \inf_{t_0 \geq 0} \sup_{t \geq t_0} |X(t) - K| \ .$$



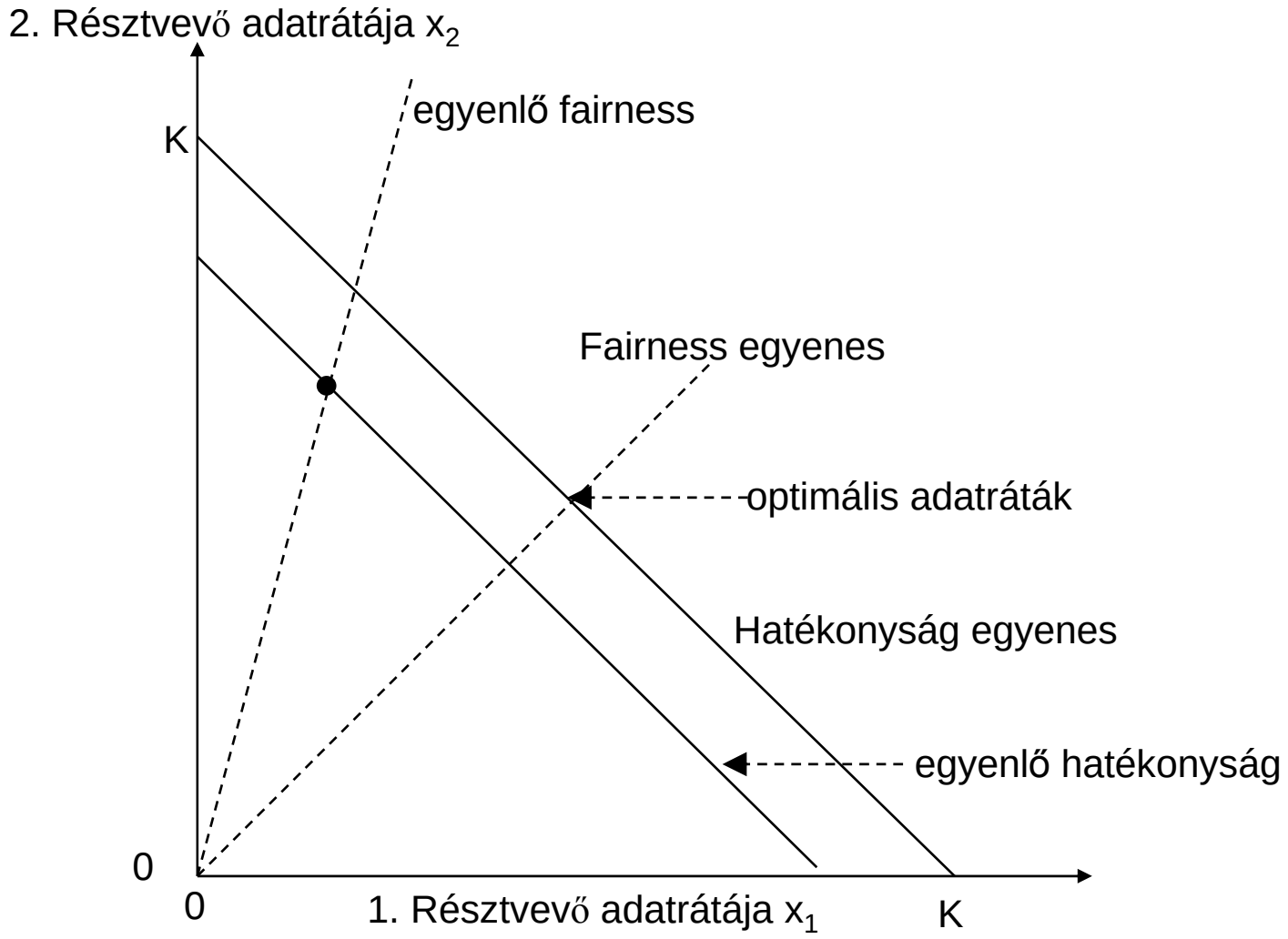

  - Berezgési idő T

$$T = \min\{t_0 \mid \forall t \geq t_0 \ : \ |X(t) - K| \leq A\} \ .$$

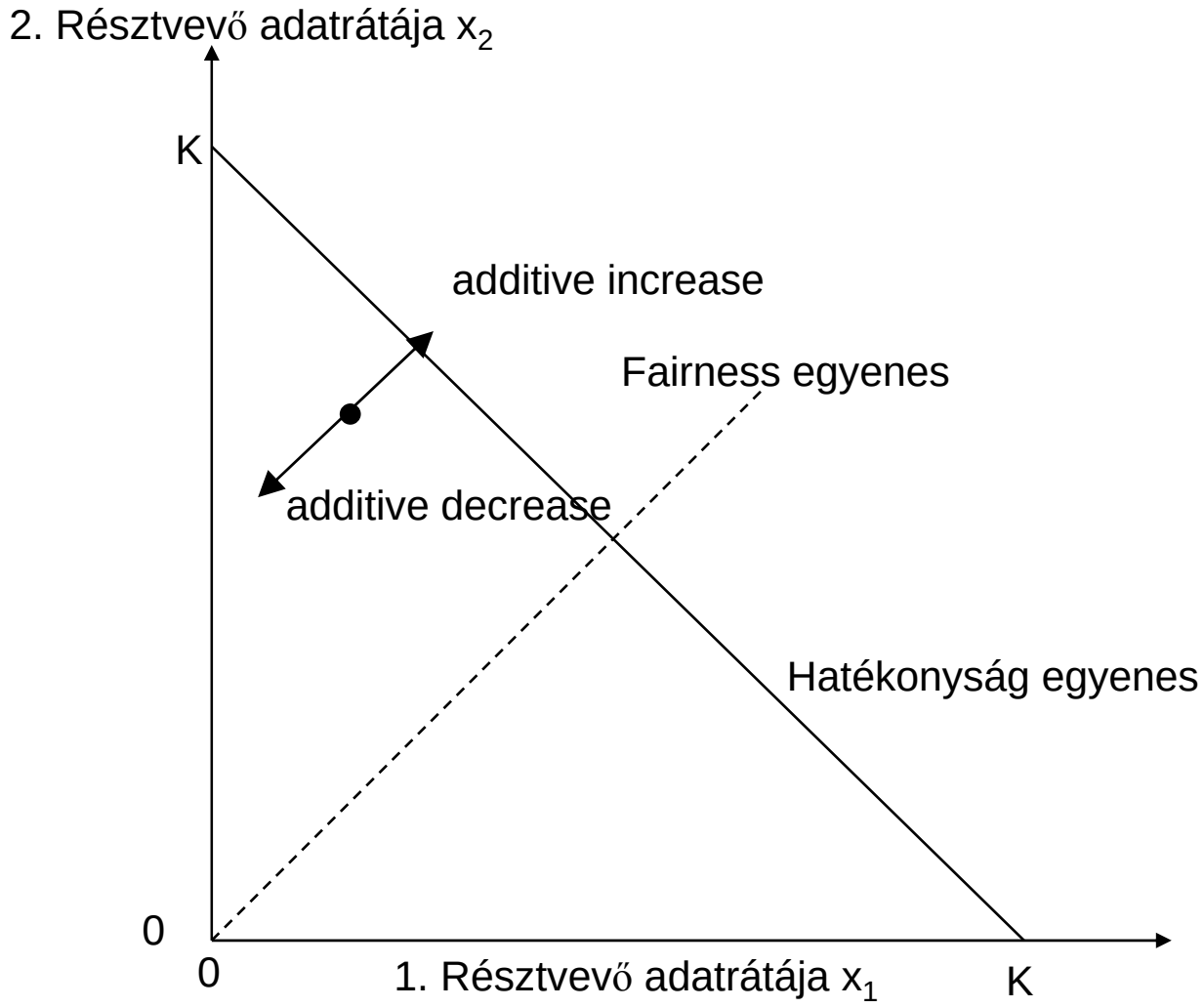
# Vektor Ábrázolás (I)



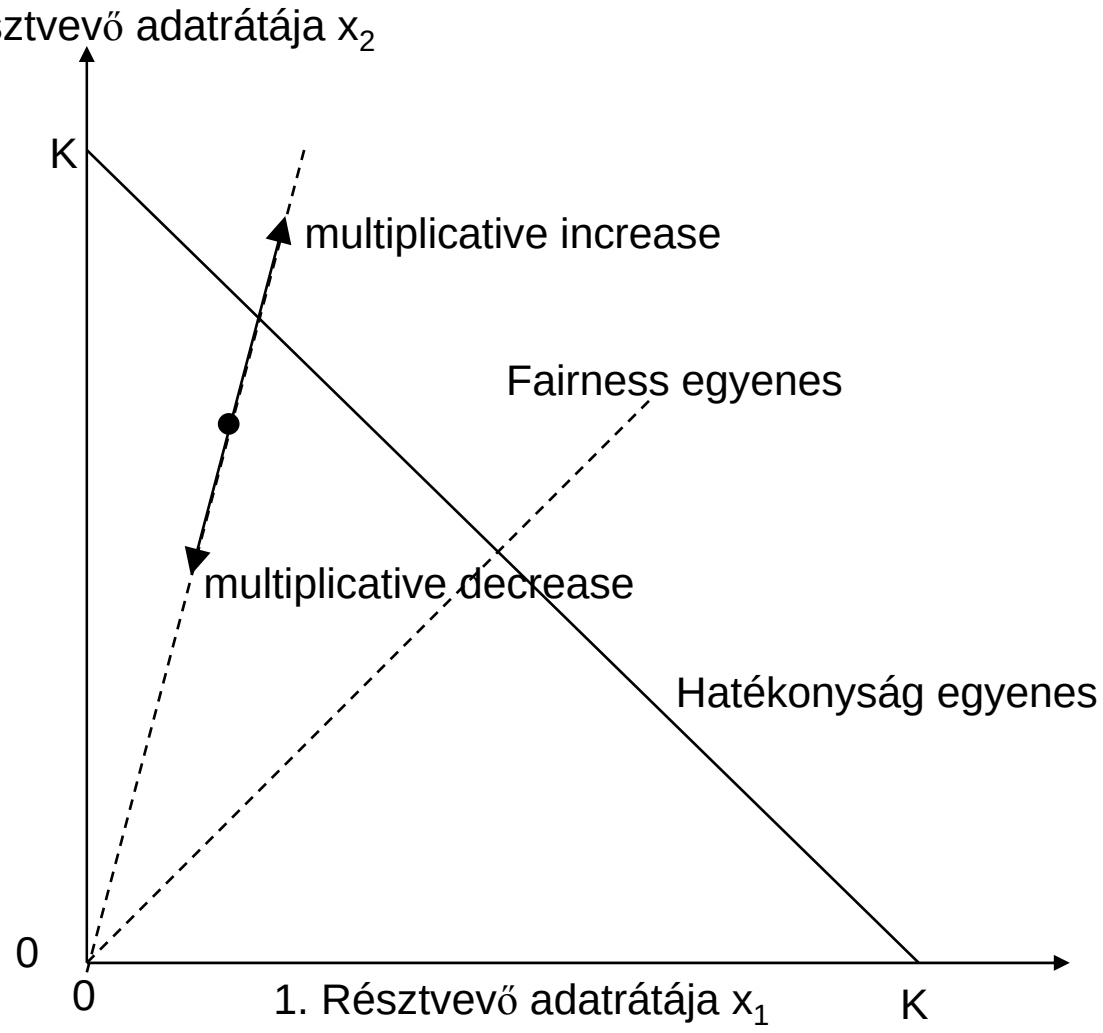
# Vektor Ábrázolás (I)



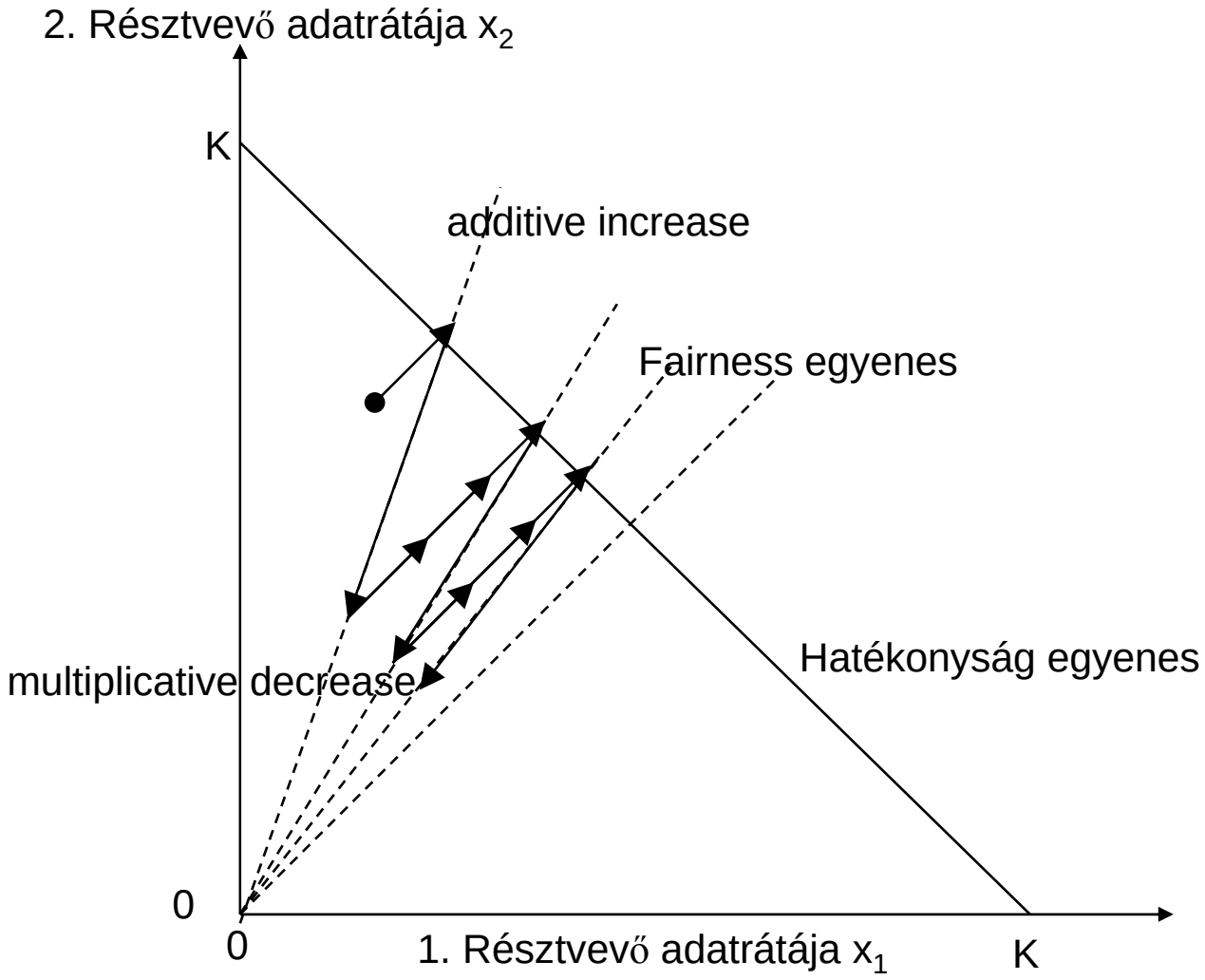
# Vektor Ábrázolás (I)



# Vektor Ábrázolás (I)



# Vektor Ábrázolás (I)



# Hatékony Lineáris Függvények

- X(t) > K

$$X(t+1) < X(t) \iff \sum_{i=1}^{n} x_i(t+1) < \sum_{i=1}^{n} x_i(t)$$

$$\iff \sum_{i=1}^{n} a_D + b_D x_i(t) < \sum_{i=1}^{n} x_i(t)$$

$$\iff na_D + (b_D - 1)X(t) < 0$$

$$\iff b_D < 1 - \frac{na_D}{X(t)} .$$

  - $a_D \leq 0 \rightarrow b_D \leq 1$
  - $a_D > 0 \rightarrow b_D < 0$

- X(t) < K

$$X(t+1) > X(t) \iff \sum_{i=1}^{n} x_i(t+1) > \sum_{i=1}^{n} x_i(t)$$

$$\iff \sum_{i=1}^{n} a_I + b_I x_i(t) > \sum_{i=1}^{n} x_i(t)$$

$$\iff na_I + (b_I - 1)X(t) > 0$$

$$\iff b_I > 1 - \frac{na_I}{X(t)} .$$

  - $a_I \geq 0 \rightarrow b_I \geq 1$
  - $a_I < 0 \rightarrow$ nem lehet

# Fairness (I)

- Az x(t) = ($x_1(0)$, …, $x_n(0)$) fairnesse konvergál 1-hez, azaz

$$\lim_{t \to \infty} F(x(t)) = 1 \ .$$

ahol

$$F(x) \ = \ \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n \sum_{i=1}^{n} (x_i)^2} \ .$$

- Teljesül c=a/b esetén:

$$F(x(t+1)) = F(x(t)) + (1 - F(x(t))) \left(1 - \frac{\sum_{i=1}^{n} (x_i(t))^2}{\sum_{i=1}^{n} (c + x_i(t))^2}\right)$$

- Bizonyítás?

# Bizonyítás! (1)

- Teljesül:

$$F(x(t+1)) = \frac{\left(\sum_{i=1}^{n} x_i(t+1)\right)^2}{n\sum_{i=1}^{n}(x_i(t+1))^2} = \frac{\left(\sum_{i=1}^{n} a + bx_i(t)\right)^2}{n\sum_{i=1}^{n}(a+bx_i(t))^2}$$

$$= \frac{\left(\sum_{i=1}^{n} \frac{a}{b} + x_i(t)\right)^2}{n\sum_{i=1}^{n}\left(\frac{a}{b}+x_i(t)\right)^2} = \frac{\left(\sum_{i=1}^{n} c + x_i(t)\right)^2}{n\sum_{i=1}^{n}(c+x_i(t))^2}$$

- Behelyettesítve

$$X = \sum_{i=1}^{n} x_i(t)\,, \qquad Q = \sum_{i=1}^{n} (x_i(t))^2\,,$$

$$L = \sum_{i=1}^{n} (x_i(t)+c)\,, \quad R = \sum_{i=1}^{n} (x_i(t)+c)^2\,.$$

- Ekkor: $F(x(t)) = \frac{X^2}{nQ}$ és $F(x(t+1)) = \frac{L^2}{nR}$

# Bizonyítás! (2)

- Meg kell mutatni: $F(x(t+1)) = F(x(t)) + (1 - F(x(t)))\left(1 - \frac{Q}{R}\right)$

$$\iff \frac{L^2}{nR} = \frac{X^2}{nQ} + \left(1 - \frac{X^2}{nQ}\right)\left(1 - \frac{Q}{R}\right)$$

$$\iff QL^2 = RX^2 + \left(nQ - X^2\right)(R - Q)$$

$$\iff QL^2 = RX^2 + nQR - nQ^2 - RX^2 + QX^2$$

$$\iff L^2 = nR - nQ + X^2$$

$$\iff L^2 - X^2 = n(R - Q)\ .$$

- Miért érvényes ez az egyenlőség?

# Bizonyítás! (3)

- Miért érvényes ez az egyenlőség?

$$L^2 - X^2 = n(R - Q)$$

ahol

$$X = \sum_{i=1}^{n} x_i(t)\,, \qquad Q = \sum_{i=1}^{n} (x_i(t))^2\,,$$

$$L = \sum_{i=1}^{n} (x_i(t) + c)\,, \quad R = \sum_{i=1}^{n} (x_i(t) + c)^2\,.$$

$$n(R-Q) = n\sum_{i=1}^{n}(x_i(t)+c))^2 - x_i(t)^2 = n\sum_{i=1}^{n}(2cx_i(t)+c^2) = 2cnX + c^2n^2$$

$$L^2 - X^2 = \sum_{i,j\in[1..n]} (x_i(t)+c)(x_j(t)+c) - x_i(t)x_j(t)$$

$$= \sum_{i,j\in[1..n]} cx_i(t) + cx_j(t) + c^2 = 2cnX + c^2n^2$$

# Fairness (II)

- Teljesül, hogy:

$$F(x(t+1)) = F(x(t)) + (1 - F(x(t))) \left( 1 - \frac{\sum_{i=1}^{n} (x_i(t))^2}{\sum_{i=1}^{n} (c + x_i(t))^2} \right)$$

- Azaz a fairness növekszik, ha c=a/b növekszik.
  - c=0, akkor F(x(t+1))= F(x(t))
  - c>0, akkor F(x(t+1))> F(x(t)), ha F(x(t)) ≠ 1
  - c<0 ist F(x(t+1))< F(x(t))
- Így $a_I/b_I \geq 0$ és $a_D/b_D \geq 0$
  - tehát, $a_I, b_I, a_D, b_D \geq 0$
- A hatákonyságból:
  - $a_D \leq 0 \rightarrow b_D \leq 1$, azt jelenti tehát, hogy $a_D = 0 \rightarrow b_D \leq 1$,
  - $a_D > 0 \rightarrow b_D < 0$ kiesik.
  - $a_I \geq 0 \rightarrow b_I \geq 1$,
    - Teljesülni kell, hogy $a_I > 0$, máskülönben nem növekszik a fairness (lásd MIMD)
- Ez AIMD-hez vezet.

# Fairness (III)

- Tétel 1: Ha $f_0$ és $f_1$ lineáris függvények, akkor a fairnesshez és a hatékonysághoz $x \geq 0$ esetén teljesülni kell, hogy:

$$f_0(x) \geq x + a_I, \quad \text{ahol } a_I > 0 \text{ és}$$
$$f_1(x) = b_D x, \quad \text{ahol } 0 < b_D < 1.$$

# TCP Adatráta

- AIMD probáló stratégia $\ell$ csomagvesztés-rátát okoz
- $\ell$ = elvesztett szegmensek / küldött szegmensek
- Közepes adatráta B = byte/sec. és a hibaráta $\ell$ kölcsönösen kihatnak egymásra:
  - Az adatráta növelése növeli a vesztésrátát
  - Magasabb vesztésráta csökenti az adatrátát
- Kisérletek mutatják, hogyha a szegmenshossz MSS és a „round trip time“ RTT:

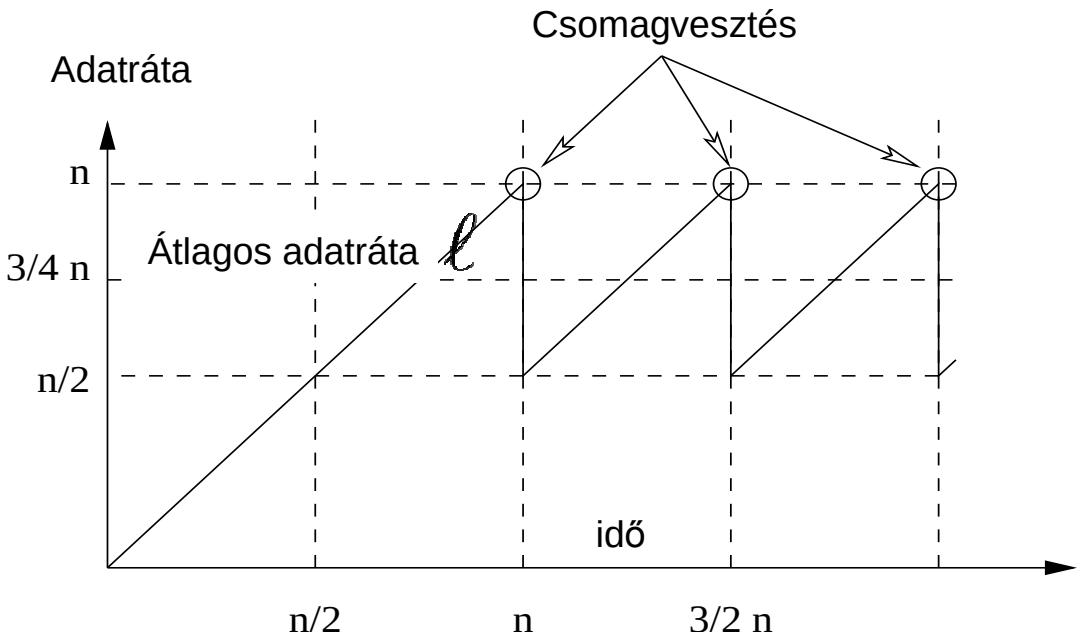
$$B = 1,3 \ \frac{\mathrm{MSS}}{\mathrm{RTT}\sqrt{\ell}}$$

# TCP-Adatráta – A Statikus Eset

- n: rendelkezésre álló sávszélesség
- Átlagos adatráta:

  B = 3/4 n
- n/2 forduló után a vesztés: 1
- Így

$$\ell = \frac{1}{\frac{3}{4}n \cdot \frac{1}{2}n} = \frac{8}{3}\frac{1}{n^2}$$




- tehát

$$n = \sqrt{\frac{8}{3\ell}}$$

$$B(\ell) = \frac{3}{4}\sqrt{\frac{8}{3\ell}}\frac{\mathrm{MSS}}{\mathrm{RTT}} = 1,22\ldots \cdot \frac{\mathrm{MSS}}{\mathrm{RTT}\sqrt{\ell}}$$

# TCP-Adatráta – Egy Sztochasztikus Modell

- Egy csomag p valószínűséggel veszik el
- A hiba nélkül átvitt csomagok száma X exponenciális eloszlású:

$$\mathrm{P}[X = i] = p(1-p)^i$$

- A hiba nélkül küldött csomagok mennyiségének várható értéke : O(1/p)
  - A fordulók száma az adatráta felezéséig: $O(1/p^{1/2})$
  - Várható adatráta: $O(1/p^{1/2})$
  - De: Mi a konstans faktor?
- Kisérletileg: (rokon modellre bizonyított):

$$B(p) = 1,3\ldots \cdot \frac{\mathrm{MSS}}{\mathrm{RTT}\sqrt{p}}$$

# Irodalom


- Van Jacobson: Congestion Avoidance and Control. In: *ACM SIGCOMM*, 314-329, 1988.
- Dah-Ming Chiu, Raj Jain: Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks. *Computer Networks and ISDN Systems*, Vol.: 17, 1-14, 1989.