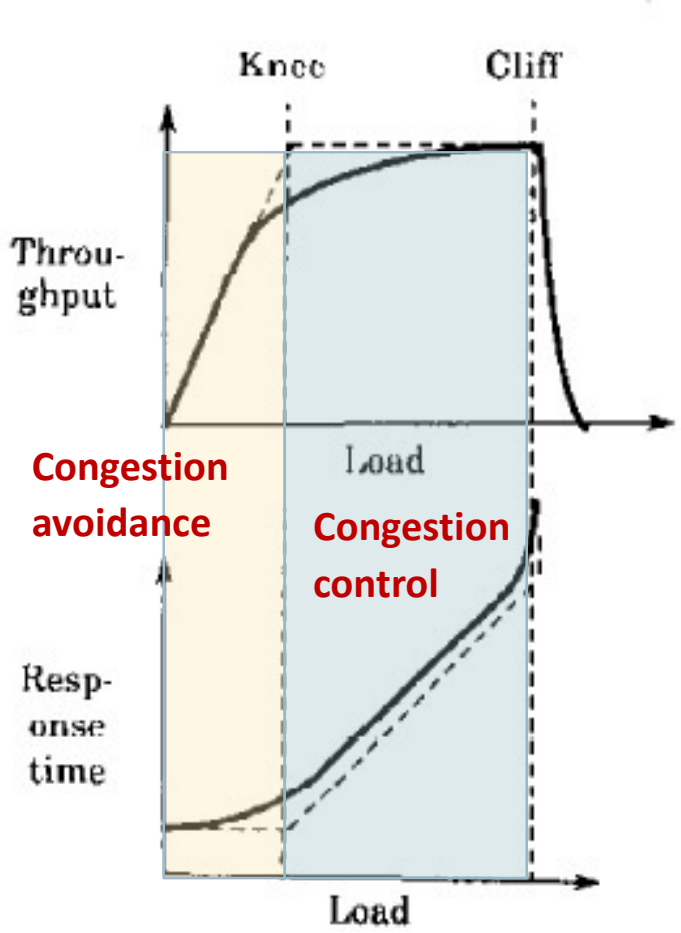
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA ELEKTROTECHNICKÁ
KATEDRA TELEKOMUNIKAČNÍ TECHNIKY

Pokročilé síťové technologie

Transmission Control Protocol - TCP a jeho funkce, principy řešení přetížení v síti, AIMD, vyhýbání se přetížení u TCP

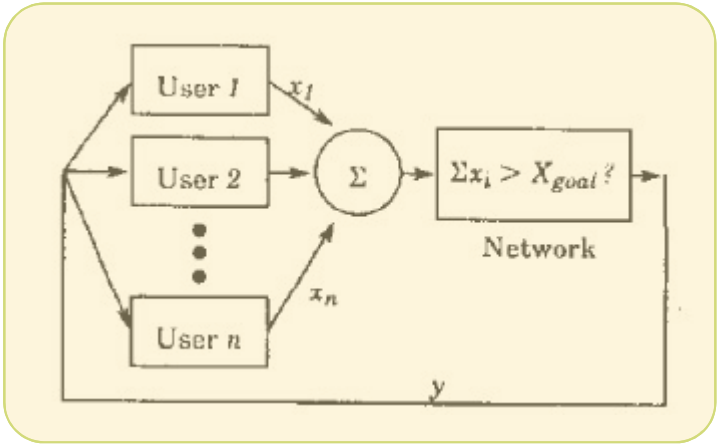
Doc. Ing. Leoš Boháč, Ph.D.

Propustnost sítě při zvětšující se zátěži



- při rostoucí nabídce rostě propustnost nejprve lineárně až do oblasti kolene (< Knee)
- v okamžiku, kdy naroste agregovaná nabídka kapacity sítě, začne se propustnost saturovat a začne narůstat zpoždění (stav ukládání do vyrovnávací paměti a postupné zaplňování front) (Knee – Cliff)
- pokud se bude nabídka nadále zvětšovat, začnou se vyrovnávací paměti přeplňovat a pakety budou zahazovány, dojde k silnému poklesu propustnosti a značnému nárůstu zpoždění (>Cliff)

Metody řízené zátěže



$$y(t) = \begin{cases} 0 \rightarrow \text{Increase load}, \\ 1 \Rightarrow \text{Decrease load}. \end{cases}$$

$$x_i(t+1) = x_i(t) + u_i(t). \qquad u_i(t) = f(x_i(t),\ y(t)).$$

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

$x_i(t+1)$

$$= \begin{cases} a_I + b_I x_i(t) & \text{if } y(t) = 0 \Rightarrow \text{Increase}, \\ a_D + b_D x_i(t) & \text{if } y(t) = 1 \Rightarrow \text{Decrease}. \end{cases}$$

(1) *Multiplicative Increase/Multiplicative Decrease*:

$$x_i(t+1) = \begin{cases} b_I x_i(t) & \text{if } y(t) = 0 \Rightarrow \text{Increase}, \\ b_D x_i(t) & \text{if } y(t) = 1 \Rightarrow \text{Decrease}. \end{cases}$$

Here, $b_I > 1$ and $0 < b_D < 1$.

(2) *Additive Increase/Additive Decrease*:

$x_i(t+1)$

$$= \begin{cases} a_I + x_i(t) & \text{if } y(t) = 0 \Rightarrow \text{Increase}, \\ a_D + x_i(t) & \text{if } y(t) = 1 \Rightarrow \text{Decrease}. \end{cases}$$

Here, $a_I > 0$ and $a_D < 0$.

(3) *Additive Increase/Multiplicative Decrease*:

$x_i(t+1)$

$$= \begin{cases} a_I + x_i(t) & \text{if } y(t) = 0 \Rightarrow \text{Increase}, \\ b_D x_i(t) & \text{if } y(t) = 1 \Rightarrow \text{Decrease}. \end{cases}$$

(4) *Multiplicative Increase/Additive Decrease*:

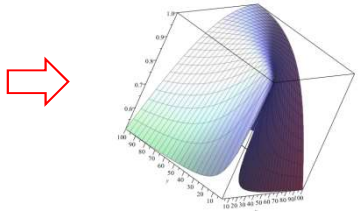
$x_i(t+1)$

$$= \begin{cases} b_I x_i(t) & \text{if } y(t) = 0 \Rightarrow \text{Increase}, \\ a_D + x_i(t) & \text{if } y(t) = 1 \Rightarrow \text{Decrease}. \end{cases}$$

Criteria for Selecting Controls

- Efficiency
  - Closeness of the total load on the resource to the knee point
- Fairness
  - Users have the equal share of bandwidth

$$Fairness = \frac{(\sum x_i)^2}{n(\sum x_i^2)}$$



- Distributedness
  - Knowledge of the state of the system
- Convergence
  - The speed with which the system approaches the goal state from any starting state

Responsiveness and Smoothness of Binary Feedback System

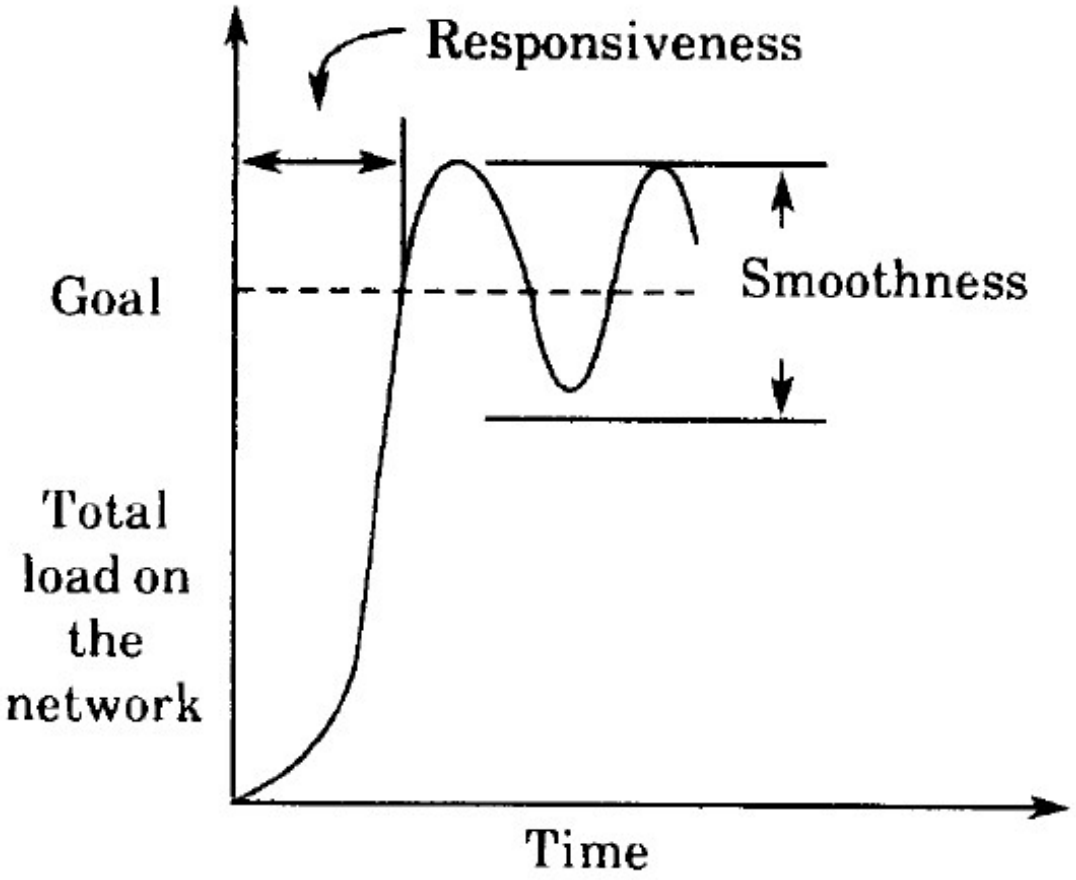


Fig. 3. Responsiveness and smoothness.

Vector Representation of the Dynamics

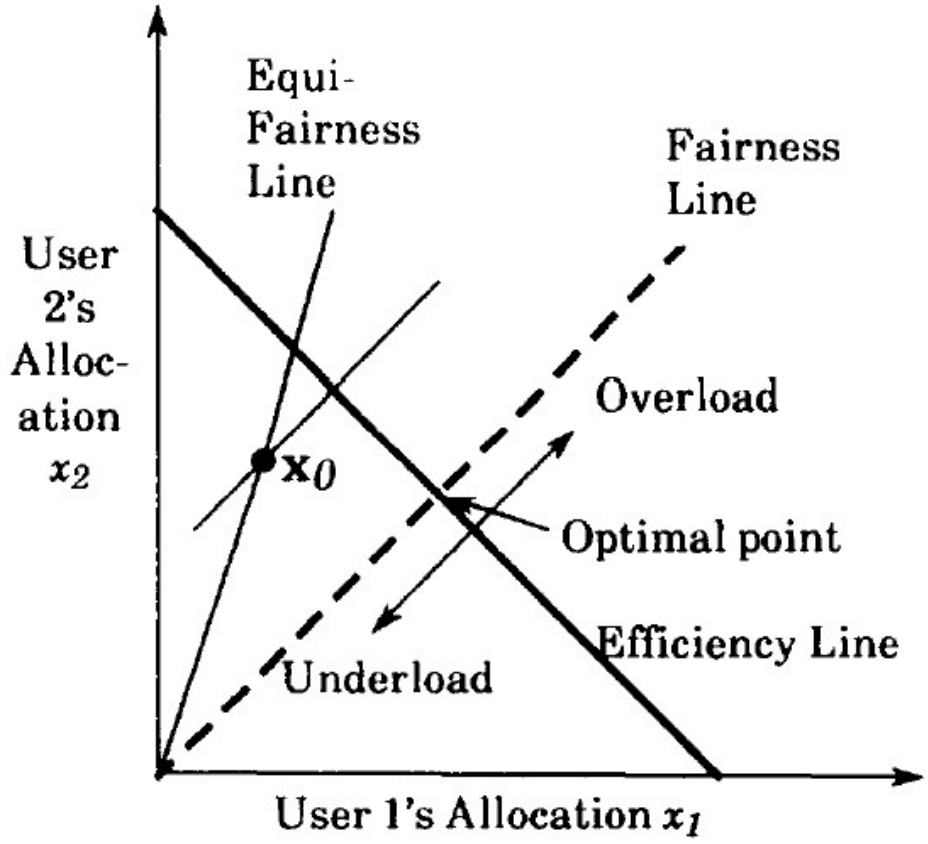
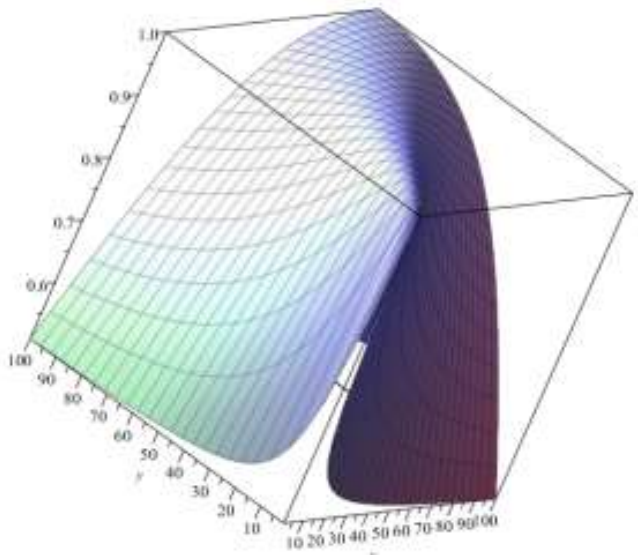


Fig. 4. Vector representation of a two-user case.

$$Fairness = \frac{(x_1 + x_2)^2}{2(x_1^{\,2} + x_2^{\,2})}$$



Example of Additive Increase/ Additive Decrease Function - AIAD

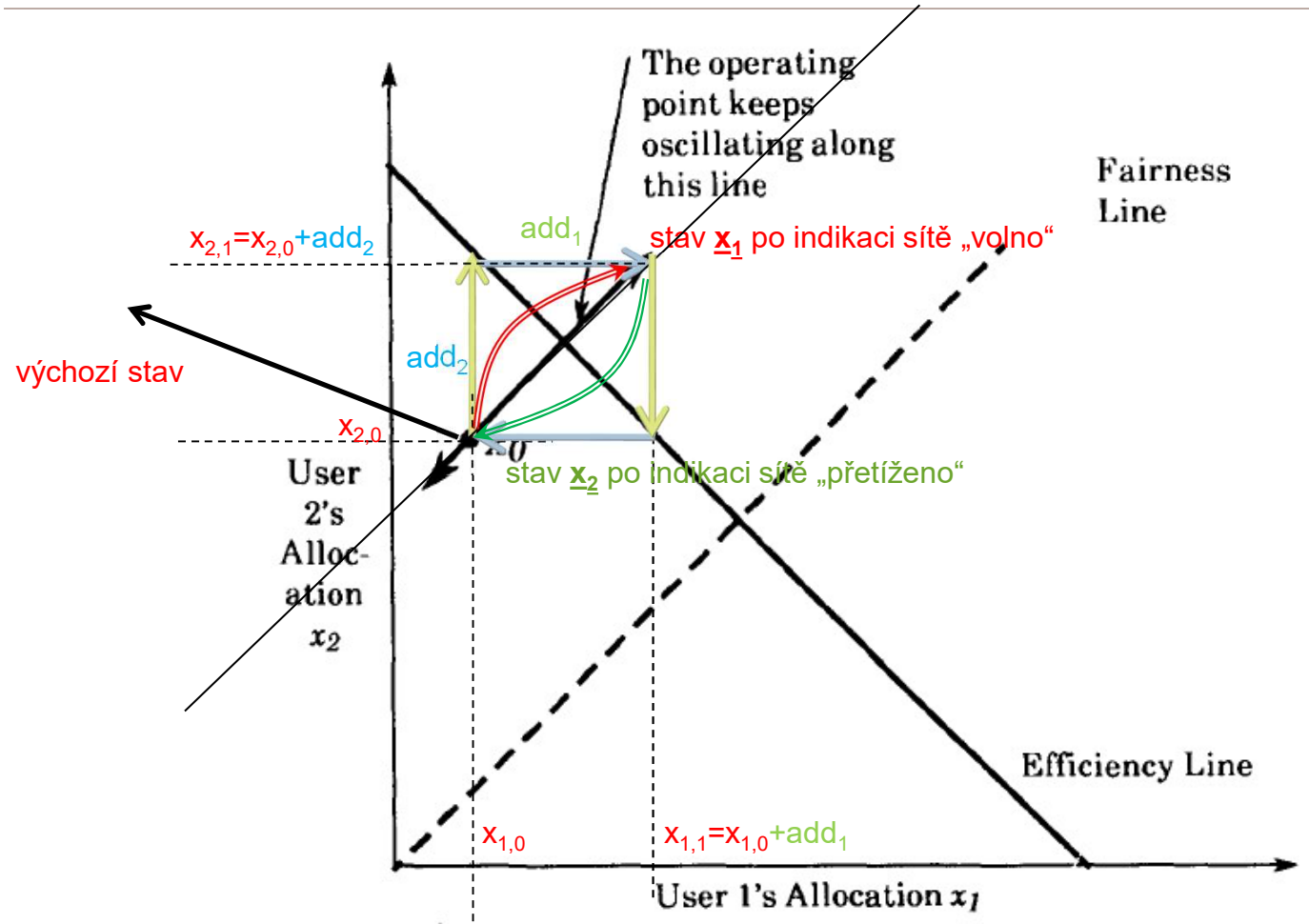


Fig. 6. Additive Increase/Additive Decrease does not converge.

Example of Additive Increase/ Multiplicative Decrease Function

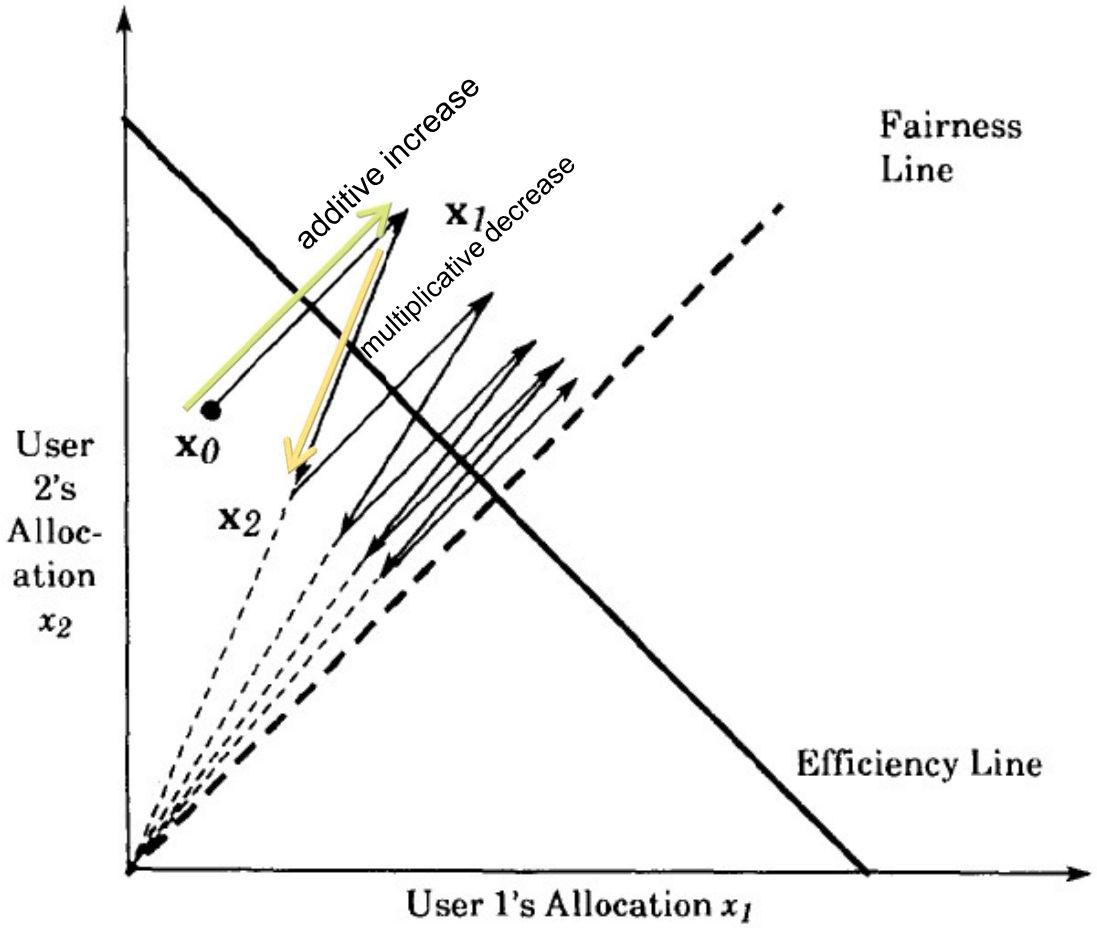


Fig. 5. Additive Increase/Multiplicative Decrease converges to the optimal point.

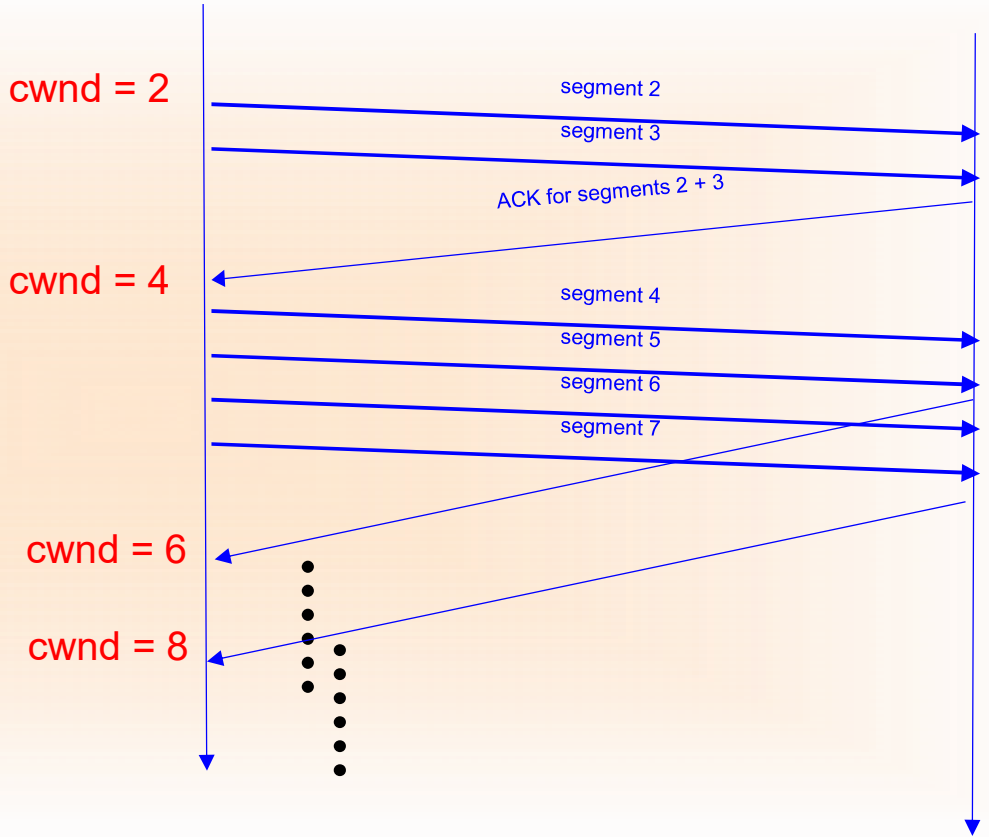
Řízení přetížení u TCP

- prochází dvěma fázemi:
  1. pomalý start (slow-start)
  2. vyhýbání se přetížení (congestion avoidance)
- proměnné, které se přidávají pro každé TCP spojení :
  - cwnd: velikost okna přetížení (congestion window)
  - ssthresh: mez velikosti okna přetížení, kdy se přechází z pomalého startu do fáze vyhýbání se přetížení (slow start threshold)
- více metod, jak řešit problém s přetížením sítě u TCP:
  - TCP/Tahoe: méně optimalizovaná verze
  - TCP/Reno: většina implementací používá tento typ
  - TCP/Vegas: není dnes používán - budoucnost

Fáze pomalého startu

inicializace:
cwnd = 2 (IW – initial window);
ssthresh = rcv_window;
pro každé nové potvrzení ACK segmentu:
if (cwnd < ssthresh)
/* pomalý start*/
cwnd = cwnd+ 1;

Triple Duplicate ACKs:
/*fáze vyhýbání se přetížení*/



!! cwnd – vyjadřuje se zde pro jednoduchost jako počet segmentů, ve skutečnosti je však v bajtech !!

Fáze vyhýbání se přetížení - RENO

Initially:

cwnd = 2;

ssthresh = infinite (64K);

For each newly ACKed segment:

if (cwnd < ssthresh)

/* pomalý start

cwnd = cwnd + 1;

else

/* vyhýbání se přetížení; cwnd se zvětšuje o hodnotu jednoho TCP segmentu při každém uplynutém intervalu intervalu RTT */

cwnd += 1;

nebo **cwnd+=SMSS*SMSS/cwdn** při každém příchodu neduplikovaného potrvrzení od přijímče TCP

/* tři duplikované potvrzení pro jeden segment – na straně přijímače je v posloupnosti dat „díra“:

cwnd = ssthresh = cwnd/2; **- jen u metody RENO; Tahoe toto nemá !!**

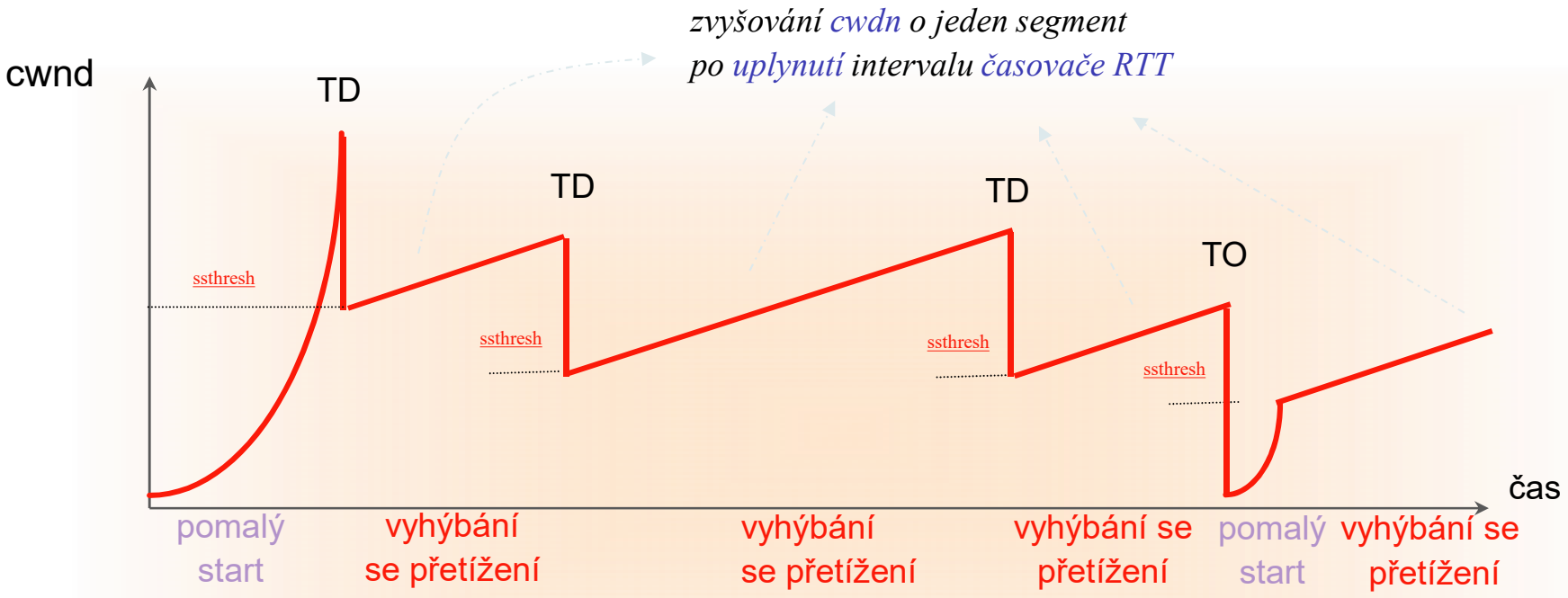
/* vypšení RTT časovače:

ssthresh = cwnd/2;

cwnd = 2;

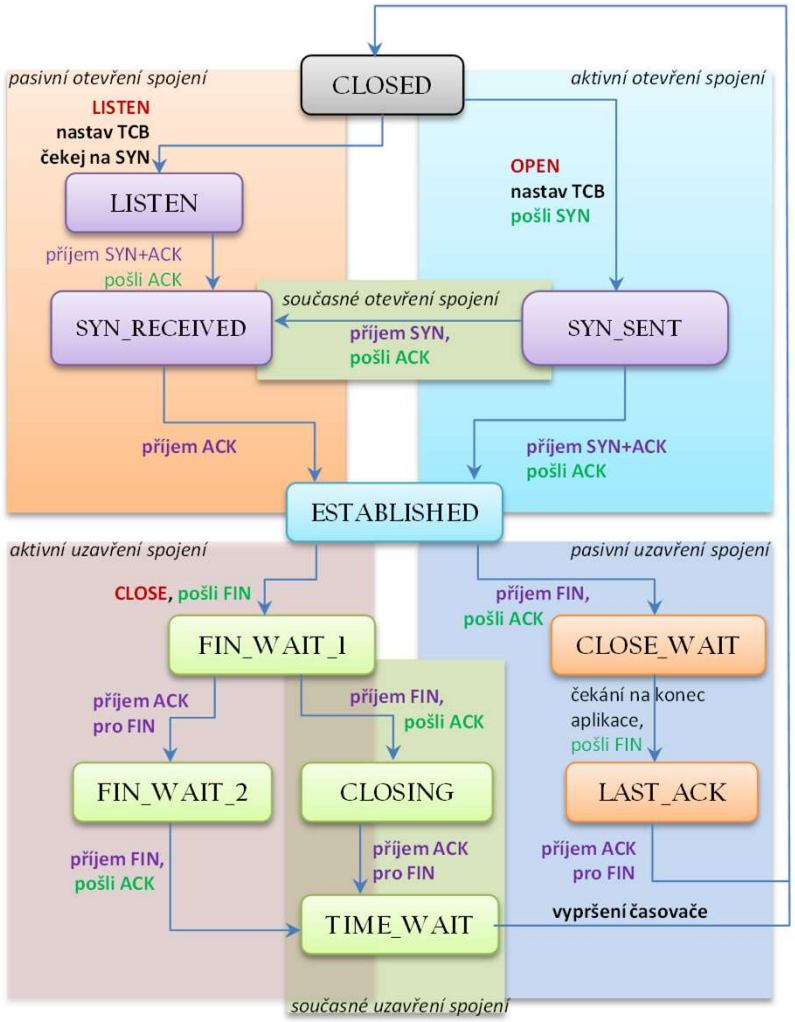
!! cwnd – vyjadřuje se zde pro jednoduchost jako počet segmentů, ve skutečnosti je však v bajtech !!

Změna okna přetížení v čase - RENO



TD: příjem tří duplikovaných potvrzení pro jeden segment na straně vysílače
TO: vypršení časovače RTT pro libovolný z již odvysílaných segmentů

TCP konečný automat



TCP protokol - záhlaví