

# Hybrid congestion control for high-speed networks

Martin Skala

Fakulta informatiky Masarykovy univerzity

9. listopadu 2011



# Obsah

## 1 Úvod

- Základní termíny
- Průlet historií
- Členění protokolů pro řízení zahlcení

## 2 HCC TCP

- Související protokoly
- Koncept
- Architektura
- Implementace
- Nastavení testovací prostředí
- Výsledky testování

## 3 Závěr

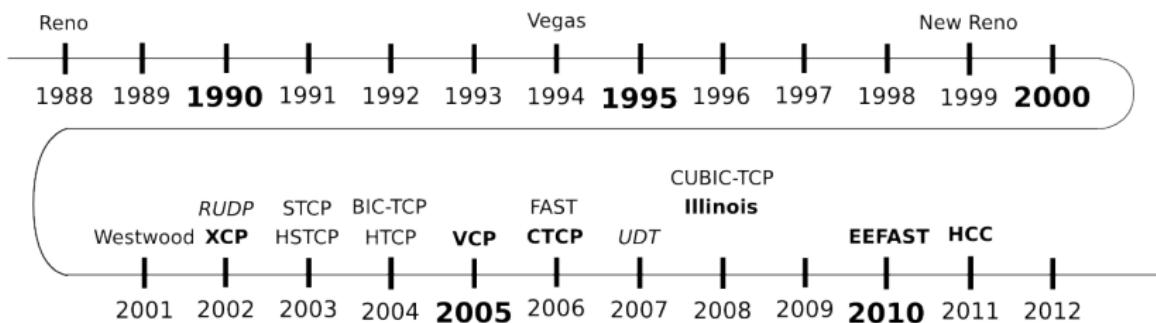


## Základní termíny

- ▶ Vlastnosti vztahující se k přenosovým protokolům<sup>1</sup>
    - ▶ **aggressiveness** - využití dostupné linky (maximální zvýšení rychlosti odesílání během jednoho  $RTT$  [ $pkts/s$ ])
    - ▶ **responsiveness** - schopnost reagovat na zahlcení (počet  $RTT$  potřebných ke snížení rychlosti odesílání na polovinu při trvajícím zahlcení)
    - ▶ **fairness** - férové rozdělení sítové kapacity mezi více proudů
      - ▶ **RTT-fairness** - férové rozdělení sítové kapacity v rámci proudů s různými  $RTT$
      - ▶ **TCP-friendliness** - férovost daného transportního protokolu vůči klasickému  $TCP$
    - ▶ **robustness** - odolnost vůči náhodným výpadkům
    - ▶ **convergence time** - čas potřebný ke zkonzvergování do férového stavu mezi existujícími proudy a nově začínajícím

<sup>1</sup>Zdroj: viz [2] a [3]

# Průlet historií



Obrázek 1: Zachycení vývoje vybraných TCP variant na časové ose

## Členění protokolů pro řízení zahlcení

- ▶ **router-based**
    - ▶ strategie zvolena na základě explicitní zpětné vazby z routerů
    - ▶ přidání podpory do existujících routerů ve světě je nepraktické
    - ▶ **protokoly:** *XCP, VCP, ..*
  - ▶ **end-to-end**
    - ▶ **loss-based CC**
      - ▶ ztráta paketu jako indikátor zahlcení
      - ▶ zvětšení okna po každém ACK, zmenšení při ztrátě paketu
      - ▶ **protokoly:** *HSTCP, STCP, BIC, CUBIC, HTCP, ..*
    - ▶ **delay-based CC**
      - ▶ zpoždění ve frontách routerů / *RTT* jako indikátor zahlcení
      - ▶ vytěžování front routerů
      - ▶ **protokoly:** *FAST, Vegas*
    - ▶ **loss-delay-based CC**
      - ▶ snaha o kombinaci předchozích přístupů
      - ▶ někdy jeden přístup primární a druhý jako *fallback*
      - ▶ **protokoly:** *CTCP, Illinois, EEFAST, HCC*

# Obsah

## 1 Úvod

- Základní termíny
- Průlet historií
- Členění protokolů pro řízení zahlcení

## 2 HCC TCP

- Související protokoly
- Koncept
- Architektura
- Implementace
- Nastavení testovací prostředí
- Výsledky testování

## 3 Závěr



## Související protokoly

## **Loss-based**

Mezi společné problémy patří oscilace mezi plným využitím kapacity sítě a podním, což zvyšuje ztrátovost paketů.

## ► HSTCP

- ▶ využití krokovacích funkcí pro zvětšování / zmenšování okna
  - ▶ umožňuje rychlé využití volné kapacity
  - ▶ **Problém:** velké problémy s *RTT-fairness*, rychlé zvětšování okna i v případě blížícího se zahlcení

## ► STCP

- ▶ hodnota pro zvětšení / zmenšení nastavována proporcionalně vzhledem k aktuálnímu oknu
  - ▶ **Problém:** viz *HSTCP*

## ▶ HTCP

- ▶ doba od posledního výpadku pro růstovou hodnotu
  - ▶ při zahlcení použití adaptivní *back-off* strategie
  - ▶ **Problém:** rychlé zvětšování okna i v případě blížícího se zahlcení



## Související protokoly II.

## ► BIC-TCP

- ▶ přibližování k refer. hodnotě metodou binárního vyhledávání
  - ▶ parametry  $W_{min}$ ,  $W_{max}$ ,  $S_{max}$
  - ▶ v počáteční fázi roste okno lineárně a poté logaritmicky  
 $(M = \frac{W_{max} - W_{min}}{2}; M \leq S_{max} ? M : S_{max})$
  - ▶ **Problém:** lepší než předchozí, ale má problémy s *RTT-fairness*

## Delay-based

Společným problémem je provoz na zpětné cestě, jenž snižuje propustnost na cestě dopředné (tzv. *reverse traffic problem*). Nepříjemný je požadavek na velikost bufferů, který by měl být větší než celkové množství paketů aktuálně zpracovávaných na routerech.

#### ► FAST TCP

- ▶ odvozen od *TCP Vegas*
  - ▶ udržuje obsazení front routerů na malé nenulové hodnotě
  - ▶ umožňuje plné využití sítové kapacity
  - ▶ umožňuje dosáhnout vyšší průměrné propustnosti
  - ▶ rychlá konvergence do rovnovážného stavu, *RTT-fairness*

## Související protokoly III.

## Loss-delay-based

Podobně jako v předchozím případě je propustnost některých protokolů z této kategorie negativně ovlivněna provozem na zpětné cestě. Stále zůstává požadavek na velikost bufferů jako v předchozím případě.

- ▶ **CTCP**
    - ▶ *delay-based* komponenta odvozená od *TCP Vegas*
    - ▶ zpočátku použití tradiční Slow-Start fáze
    - ▶ indikátor zpoždění paketů použit ve fázi *congestion avoidance*
  - ▶ **TCP-Illinois**
    - ▶ indikátor ztráty paketů je primární, zpoždění paketů sekundární
    - ▶ ztrátovost udává směr změny velikosti okna
    - ▶ zpožďování udává velikost/tempo této změny
    - ▶ konkávnost křivky velikosti okna dána parametry  $\alpha, \beta$
    - ▶ nehrozí-li zahlcení, pak  $\alpha \gg \beta$ ; v opačném případě  $\beta \gg \alpha$
  - ▶ **EEFAST**
    - ▶ navržen pro odstranění vlivu provozu na zpětné cestě
    - ▶ dynamicky mění okno vzhledem k jednocestnému zpoždění

# Koncept

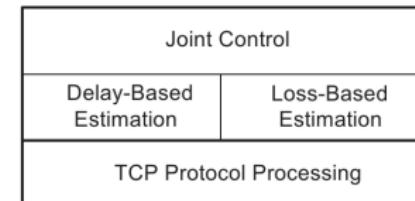
## Základní charakteristika HCC TCP

- ▶ spadá do kategorie *loss-delay-based* protokolů
  - ▶ zpožďování jako primární indikátor, ztrátovost jako sekundární
  - ▶ funguje dobře při provozu na zpětné cestě, měří pouze jednocestné zpoždění stejně jako *EEFAST*
  - ▶ nevyžaduje explicitně větší velikost bufferů (díky sekundárnímu indikátoru)
  - ▶ další vlastnosti ověřeny při testovacích měření



## Architektura

Z architektury znázorněné na obrázku 2 je patrné, že se protokol v podstatě skládá ze 3 komponent - komponenta odhadu na bázi zpoždění (**delay-based estimation**), komponenta odhadu na bázi ztrátovosti (**loss-based estimation**) a komponenta společné kontroly (**joint control**).



Obrázek 2: Architektura HCC

## Značení

- ▶  $\alpha_i$  - počet paketů v rovnovážném stavu ve frontách routerů
  - ▶  $w_i, w_i^a(t), \Delta w_i$  - velikost okna, resp. předpokl. velikost okna zdroje  $i$  v čase  $t$ , resp. rozdíl mezi aktuální a předpokl. velikostí okna
  - ▶  $D_i^f, d_i$  - dopředné jednocestné resp. propagační zpoždění zdroje  $i$
  - ▶  $baseD_i^f$  - minimální dosud pozorované  $D_i^f$
  - ▶  $Q_i^f, Q_i^r$  - dopředné resp. zpětné zpoždění ve frontách zdroje  $i$ ,  
 $Q_i^f = D_i^f - baseD_i^f$
  - ▶  $avgQ_i^f(t), avgD_i(t)$  - průměrné hodnoty  $Q_i^f(t)$  resp. RTT
  - ▶  $I_{min}, I_{max}$  - minimální resp. maximální změna velikosti okna,  $I_{max} = \gamma \alpha_i$

## Komponenty I.

## Komponenta delay-based estimation

- ▶ využívá mechanismů *EEFAST* pro odhad zahlcení v síti
  - ▶ rozšíření o nové kontrolní postupy řídící velikost okna
  - ▶ v rovnovážném stavu by mělo být ve frontách routerů  $\alpha_i$  paketů bez negativního efektu provozu na zpětné cestě
  - ▶ velikost okna lze zapsat jako  $w_i = \frac{\alpha_i}{Q_i^f} D_i = \frac{\alpha_i}{Q_i^f} (Q_i^f + Q_i^r + d_i)$
  - ▶ předpokládaná velikost okna v čase  $t$  odpovídá  

$$w_i^a(t) = \frac{\alpha_i}{Q_i^f(t)} D_i(t) = \alpha_i \left( 1 + \frac{Q_i^r(t) + d_i}{Q_i^f(t)} \right)$$
  - ▶ v iniciální fázi obvykle  $Q_i^f$  velmi malé a  $w_i^a(t)$  naopak velmi vysoké (než  $w_i^c(t)$ )
  - ▶ s rostoucí  $w_i^c(t)$  roste také  $D_i^f \Rightarrow$  se  $w_i^a(t)$  snižuje

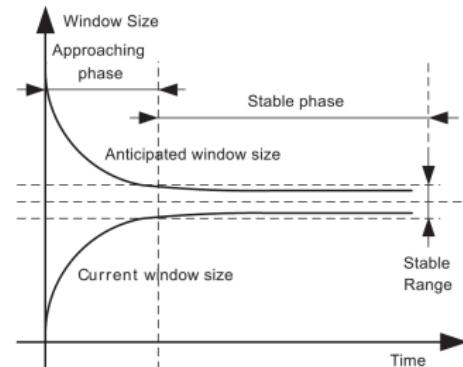
## Komponenty II.

Pokud se vzdálenost mezi předpokl. a aktuální velikostí okna nachází v oblasti *stable range* (viz obr. 3), můžeme říct, že je plně využito dostupné kapacity sítě.

$$w_i^a(t) = \frac{\alpha_i}{\text{avg}Q_i^f(t)} \text{avg}D_i(t).$$

Poté rozdíl  $\Delta w_i$  mezi přepodkl. a aktuální velikostí okna  $\Delta w_i(t) = w_i^a(t) - w_i^c(t)$ .

Pokud je  $\Delta w_i > 0$ , pak by se mělo aktuální okno zvětšit. V opačném případě by mělo dojít ke změnšení, které povede k přiblížení se k přepodkl. oknu.



Obrázek 3: Změna velikosti okna na základě *delay-based* strategie

## Komponenty III.

Před změnou velikosti okna musí dojít ke kontrole, zda změna velikost o  $\Delta w_i$  není větší než  $I_{max}$ .  $|\Delta w_i| \geq I_{max} ? I_{max} : |\Delta w_i|$ . Příliš velké změny by mohly zapříčinit výkyvy, které by měly za následek degradaci výkonu protokolu. Z předešlých výzkumů v oblasti protokolu *FAST TCP* bylo zjištěno, že ideální hodnota  $I_{max} = \gamma \alpha_i$ .

- ▶ protokol aktualizuje velikost okna jednou za  $RTT$
  - ▶ pozvolné přibližování k  $w_i^a(t)$ , velikost se mění pouze o  $\Delta w_i/2$

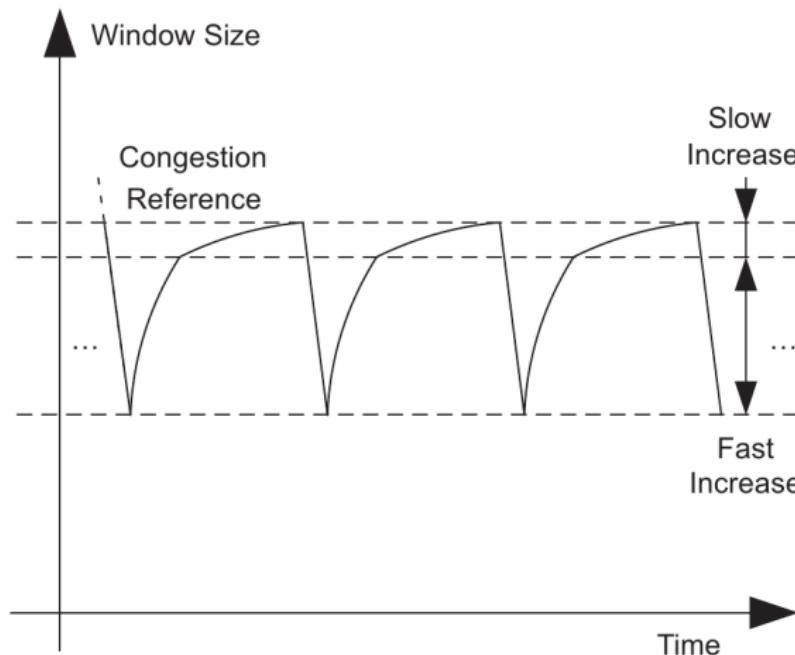
## Komponenty IV.

## Komponenta loss-based estimation

*Delay-based* algoritmy vyžadují ve frontách routerů určený počet paketů, aby mohly udržovat průměrnou propustnost okolo maximálního možné hodnoty. Proto by velikost bufferů na routerech měla být větší než tento určený počet paketů. V opačném případě může docházet k výrazným ztrátám paketů v síti.

- ▶ rychlé získání volné kapacity sítě a prodloužení doby mezi posobě následujícími zahlceními díky změnám velikosti okna v prim. komponentě (lineární → logaritmický růst - viz obr. 4)
  - ▶ změna velikosti okna vzhledem k parametrům  $I_{max}$  a  $I_{min}$  stejně jako v primární komponentě
  - ▶ bere také pouze polovina vypočítané hodnoty, která by měla určovat změnu okna

## Komponenty V.



Obrázek 4: Lineárně → logaritmicky se zvyšující schéma

## Komponenty VI.

- ▶ při detekci ztraceného paketu dojde k zaznamenání aktuální velikosti okna (*loss-reference*) a k nastavení okna na polovinu
  - ▶ poté fáze zotavení, kde  $w_i^{max} = loss\text{-reference}$ ,  $w_i^{min} =$  aktuální velikost okna a cílové zvětšení okna ( $w_i^{tar}$ ) bude  $w_i^{tar} = (w_i^{max} - w_i^{min}) / 2$
  - ▶ není-li po navýšení okna ve fázi zotavení detekována ztráta, nastaví se  $w_i^{min}$  na akt. velikost okna a celý proces se opakuje
  - ▶ je-li detekována ztráta, dojde k zaznamenání aktuální velikosti okna na *loss-reference* a  $w_i^{max}$

## Implementace

- ▶ při startu protokolu se pro odhad zahlcení na dopředné cestě nastaví *threshold* hodnota *mi\_threshold*
  - ▶ pokud  $avgQ_i^f(t) < mi\_threshold$ , pak indikuje, že dopředné zpoždění ve frontách je nízké - lze použít schéma multiplikativního navýšení (MI)
  - ▶ v opačném případě se předá řízení zahlcení primární komponentě
  - ▶ dále se nastaví *loss-reference* na svou maximální (implicitní) hodnotu (dokud není detekována ztráta)

#### **Delay-based congestion control function ( ):**

Estimate anticipated window size (*delay-reference*):

$$w_i^a(t) = \alpha_i \cdot avgD_i(t) / avgQ_i^f(t)$$

Calculate the change for current window size:  $\Delta w_i(t) = w_i^d(t) - w_i^e(t)$ ;

If  $\Delta w_i(t) > I_{max}$ , then  $\Delta w_i(t) = I_{max}$

If  $\Delta w_i(t) < -I_{max}$ , then  $\Delta w_i(t) = -I_{max}$ .

Update congestion window:  $W = W_{old} + (Aw_i/2)$  per RTT:

Upon detected loss packet

### update loss-reference

back off to  $W=W/2$

turn to loss-based congestion control function ( );

### Loss-based congestion control function ( ):

Calculate increment for current window size:

$$W_i^{\text{tar}} = (W_i^{\max} - W_i^{\min})/2$$

If  $w_i^{\text{tar}} \geq l_{\max}$ , then  $w_i^{\text{tar}} = l_{\max}$

If  $w_{\text{tar}}^{\text{tar}} \leq l_{\text{tar}}$ , then  $w_{\text{tar}}^{\text{tar}} = l_{\text{tar}}$

Update congestion window:  $W = W_{\text{initial}} + (w^{\text{fair}}/2)$  per RTT;

If  $W > loss\_reference \&$  no packet loss is detected

in W > loss-reference

turn to **delay-based congestion control function** ( )

Upon detected loss packet

Upon detected loss packet  
update loss reference

update loss reference  
back off to  $W-W/2$

(a) část 1.

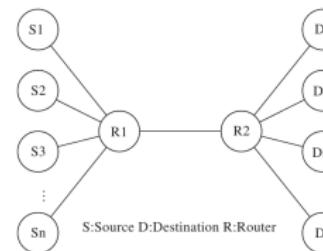
(b) část 2

### Obrázek 5: Algoritmus - pseudokód

## Nastavení testovací prostředí

## Nastavení testovací prostředí

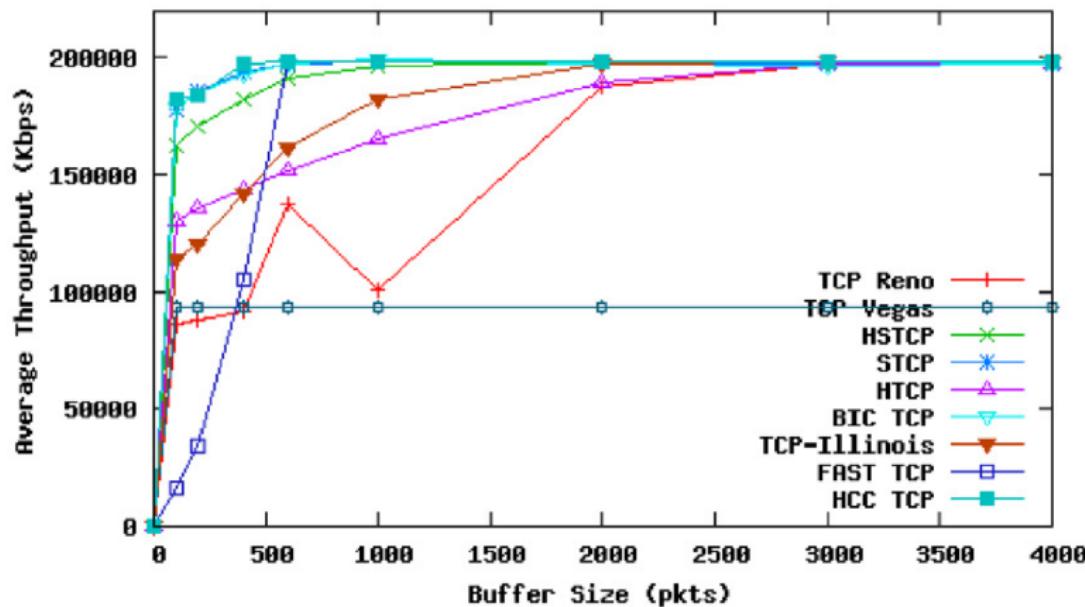
- ▶ simulace prostřednictvím *The Network Simulator (ns-2)*
- ▶ cílem porovnat a vyhodnotit vlastnosti *HCC TCP* s dalšími protokoly<sup>2</sup>
- ▶ zvolena topologie na obr. 6, jedna sdílená *200 Mbps* linka ( $d_i = 20ms$ )
- ▶ routery R1 i R2 využívají FIFO fronty, velikost datových paketů 1000 bajtů
- ▶ parametrizované protokoly (*FAST, HCC*) implicitně  $\alpha = 400, \gamma = 0.5$
- ▶ standardní doba měření každé simulace je *300 s*



Obrázek 6: Síťová topologie zvolená k testování

<sup>2</sup> TCP Reno, TCP Vegas, HSTCP, STCP, HTCP, BIC-TCP, TCP-Illinois a FAST TCP

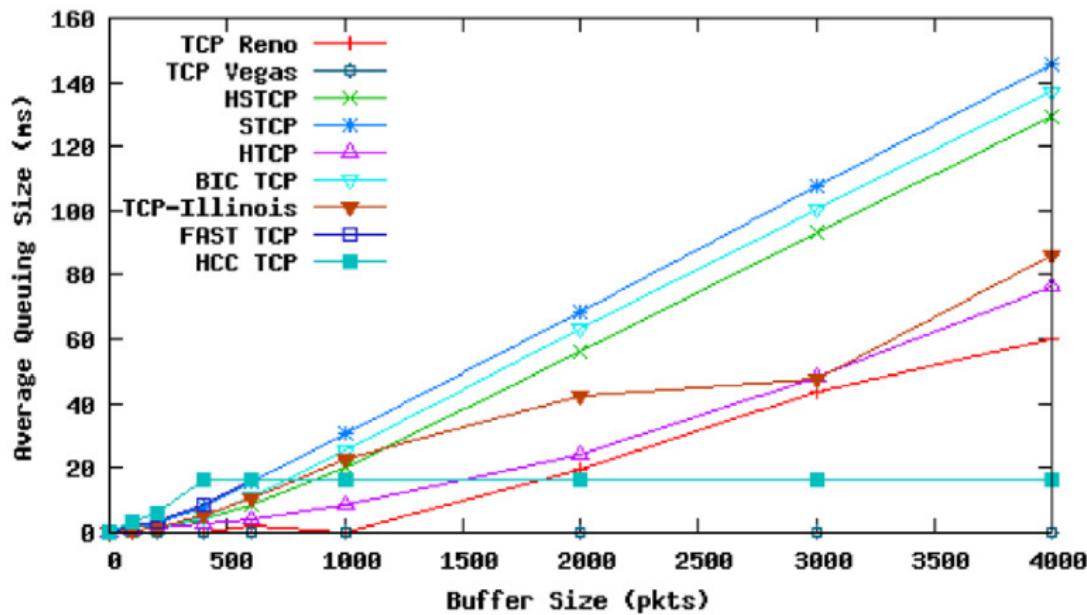
## Účinnost jednoho proudu



Obrázek 7: Průměrná propustnost vs. velikost bufferů

## Výsledky testování

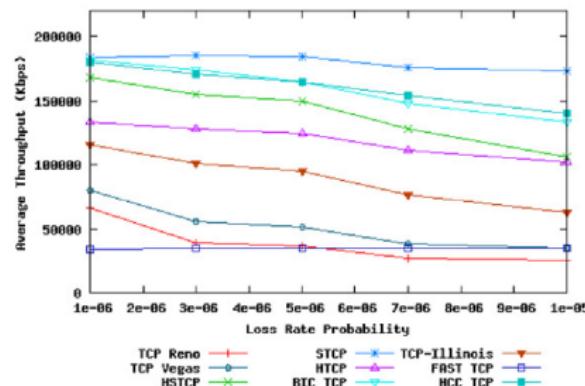
## Účinnost jednoho proudu II.



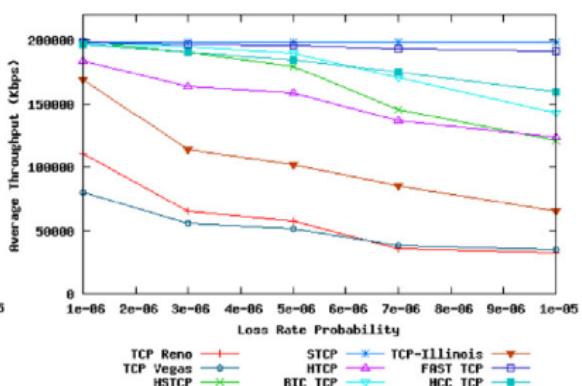
Obrázek 8: Průměrná frontovací velikost vs. velikost bufferů

## Výsledky testování

## Účinnost jednoho proudu III.



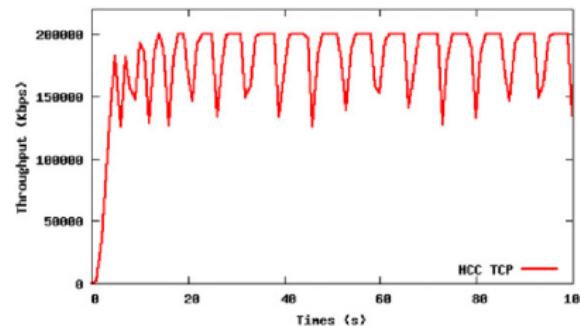
(a) velikost bufferu 200 paketů



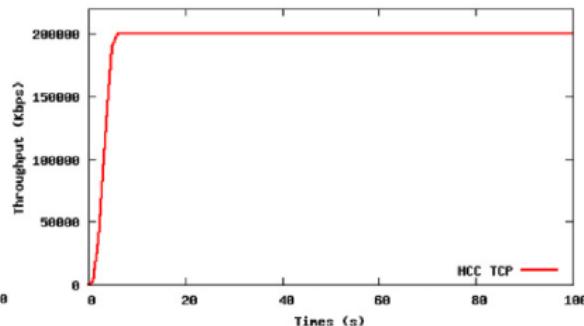
(b) velikost bufferu 2000 paketů

Obrázek 9: Průměrná propustnost vs. pravděpodobnost ztráty paketu

## Účinnost jednoho proudu IV.



(a) velikost bufferu 200 paketů

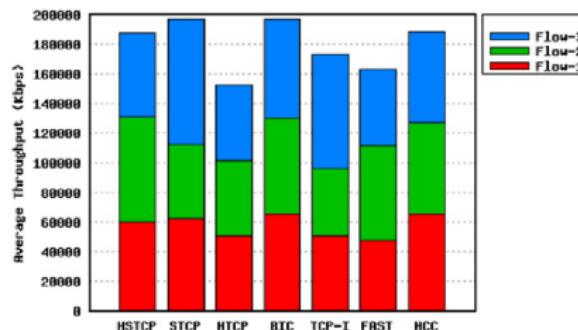


(b) velikost bufferu 600 paketů

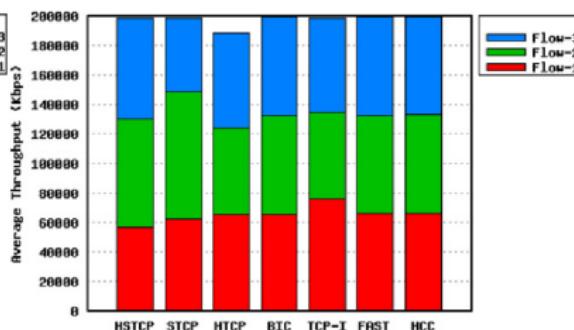
Obrázek 10: Dynamičnost HCC TCP

## Výsledky testování

## Fairness



(a) velikost bufferu 600 paketů



(b) velikost bufferu 2000 paketů

Obrázek 11: Průměrná propustnost 3 homogenních RTT (120 ms) přenosů

*Jain's fairness index (FI)*, počet paralelních proudů  $n$ , průměrná prop. proudu  $i = \bar{x}_i$

$$FI \equiv \frac{(\sum_{i=1}^n \bar{x}_i)^2}{n}$$

## Výsledky testování

## Fairness II.

*Efficiency index (EI)* odpovídá součtu propustností všech paralelních proudů

RTT ratio	1.5				2				6					
	Protocols		T1	T2	EI	FI	T1	T2	EI	FI	T1	T2	EI	FI
HSTCP		197.91	1.68	199.59	<b>0.5085</b>		196.75	2.76	199.51	<b>0.5140</b>	198.57	0.66	199.23	<b>0.5033</b>
STCP		199.53	0.08	199.61	<b>0.5004</b>		199.24	0.29	199.53	<b>0.5015</b>	199.54	0.02	199.56	<b>0.5001</b>
HTCP		175.96	9.38	185.34	<b>0.5532</b>		184.64	6.09	190.73	<b>0.5329</b>	166.89	9.31	176.20	<b>0.5556</b>
BIC TCP		199.02	0.62	199.64	<b>0.5031</b>		199.00	0.62	199.62	<b>0.5031</b>	199.31	0.34	199.65	<b>0.5017</b>
TCP-Illinois		156.56	42.28	198.84	<b>0.7517</b>		173.75	23.15	196.90	<b>0.6309</b>	195.81	2.93	198.74	<b>0.5150</b>
FAST TCP		187.13	12.43	199.56	<b>0.5661</b>		190.11	9.44	199.55	<b>0.5495</b>	186.76	6.21	192.97	<b>0.5332</b>
HCC TCP		177.03	22.45	199.48	<b>0.6248</b>		179.19	20.26	199.45	<b>0.6116</b>	182.33	16.18	198.51	<b>0.5880</b>

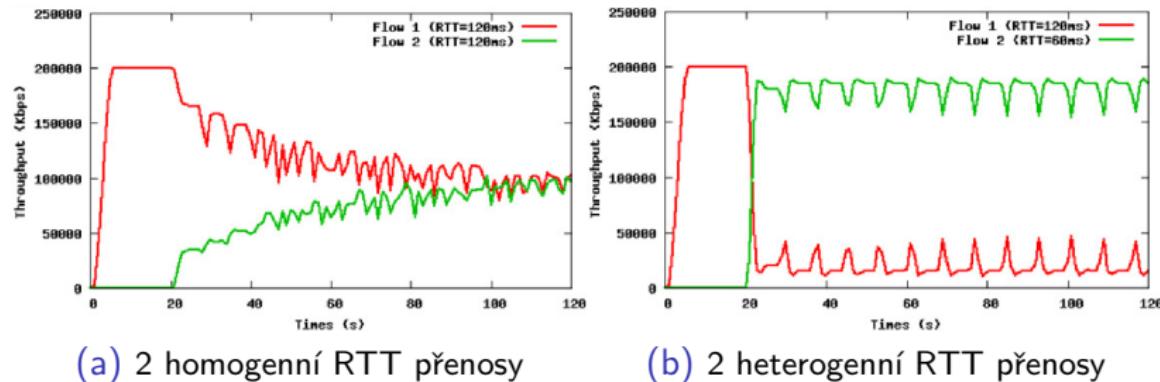
(a) velikost bufferu 600 paketů

RTT ratio	1.5				2				6					
	Protocols		T1	T2	EI	FI	T1	T2	EI	FI	T1	T2	EI	FI
HSTCP		196.82	2.84	199.66	<b>0.5144</b>		199.35	0.23	199.58	<b>0.5012</b>	199.01	0.32	199.33	<b>0.5016</b>
STCP		198.74	0.93	199.67	<b>0.5047</b>		199.45	0.12	199.57	<b>0.5006</b>	197.69	1.26	198.95	<b>0.5064</b>
HTCP		189.87	9.79	199.66	<b>0.5514</b>		189.55	10.00	199.55	<b>0.5526</b>	168.39	30.28	198.67	<b>0.6742</b>
BIC TCP		199.33	0.31	199.64	<b>0.5016</b>		198.17	1.33	199.50	<b>0.5067</b>	199.03	0.63	199.66	<b>0.5032</b>
TCP-Illinois		130.19	69.27	199.46	<b>0.9147</b>		161.90	37.56	199.46	<b>0.7201</b>	168.30	31.13	199.43	<b>0.6788</b>
FAST TCP		100.18	99.40	199.58	<b>1.0000</b>		100.35	99.22	199.57	<b>1.0000</b>	102.77	96.77	199.54	<b>0.9991</b>
HCC TCP		99.94	99.55	199.49	<b>1.0000</b>		100.13	99.33	199.46	<b>1.0000</b>	103.31	96.11	199.42	<b>0.9987</b>

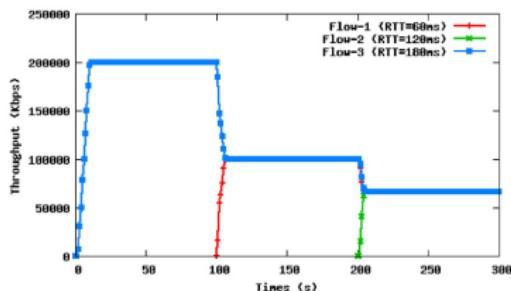
(b) velikost bufferu 2000 paketů

Obrázek 12: Výsledky měření indexů EI a FI

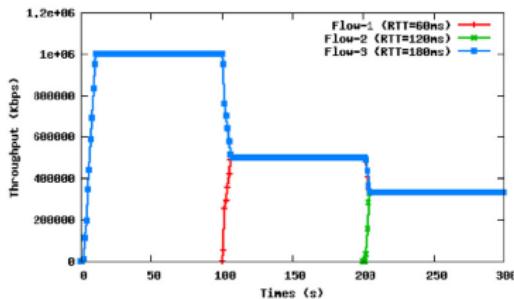
# Fairness III.



Obrázek 13: Dynamičnost HCC TCP s velikostí bufferu 600 paketů

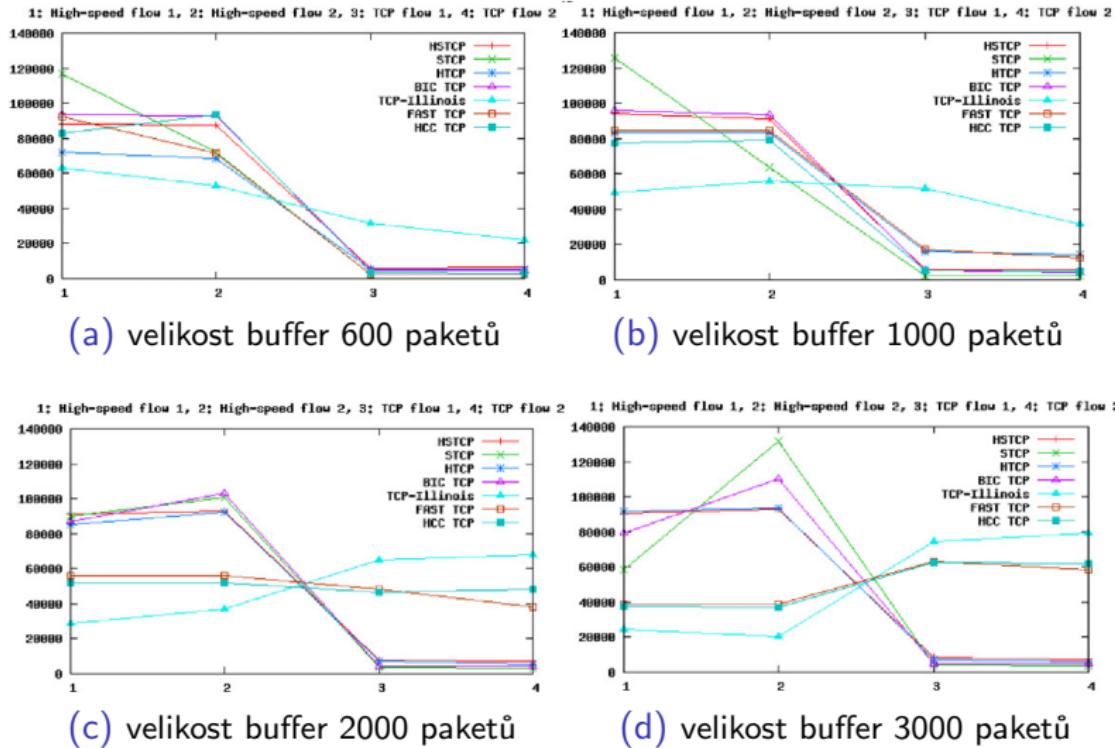


Obrázek 14: Dynamičnost HCC TCP s velikostí bufferu 2000 paketů

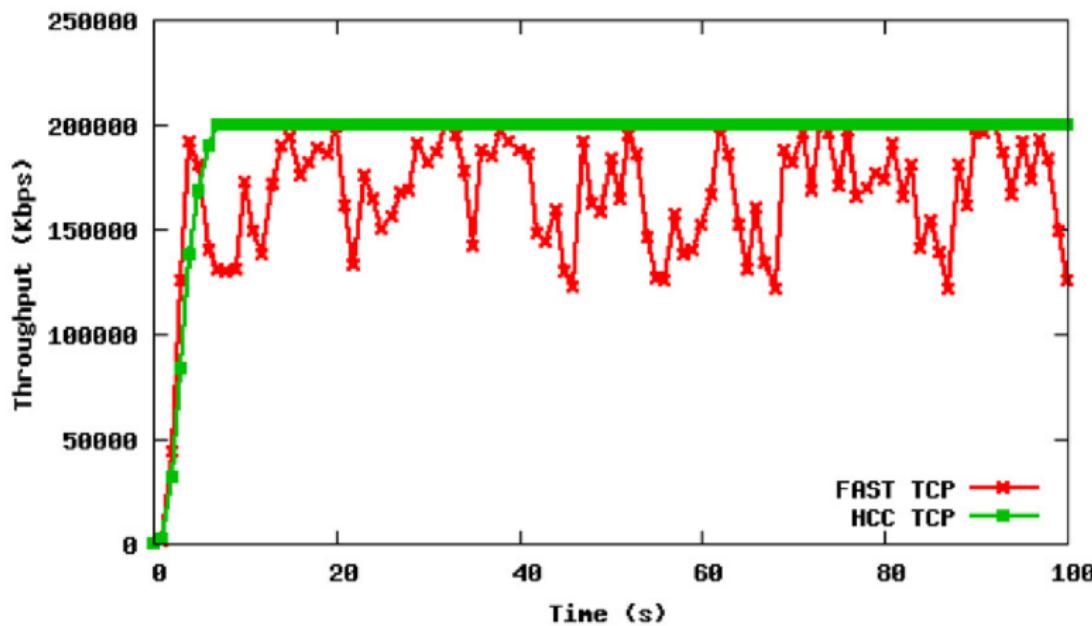


Obrázek 15: Dynamičnost HCC TCP na 1 Gbps síti

## Výsledky testování

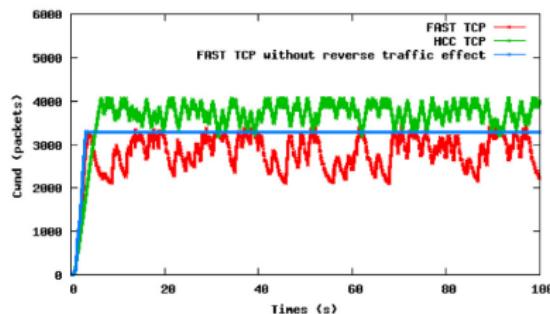
Obrázek 16: Průměrná prop. 2 *high-speed* TCP variant a 2 *TCP Reno*

## *Robustness I.*

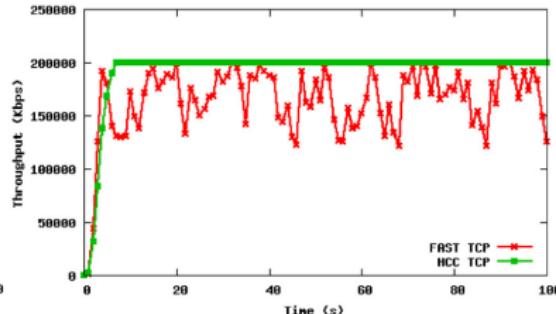


Obrázek 17: Průměrná prop. vs. velikost bufferu s přítomností provozu na zpětné cestě

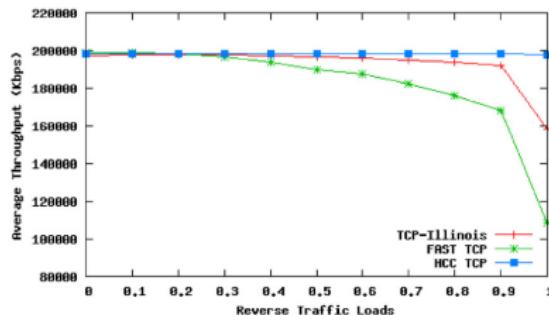
## Výsledky testování



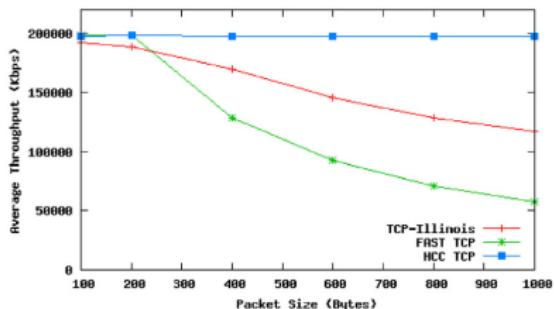
(a) porovnání křivek oken



(b) s provozem na zpětné cestě



(c) průměrná prop. vs. proměnná zátěž na zpětné cestě



(d) průměrná prop. vs. proměnná velikost paketů na zpětné cestě

# Obsah

## 1 Úvod

- Základní termíny
- Průlet historií
- Členění protokolů pro řízení zahlcení

## 2 HCC TCP

- Související protokoly
- Koncept
- Architektura
- Implementace
- Nastavení testovací prostředí
- Výsledky testování

## 3 Závěr



Shrnutí

- ▶ V případě, že primární *delay-based* algoritmus pracuje neefektivně (tj. jsou detekované ztráty), využívá *HCC* růstovou funkci založenou na měření ztrát, která zpočátku roste lineárně a ke konci logaritmicky.
  - ▶ *HCC* dokáže velmi rychle a efektivně využít volnou kapacitu sítě, přičemž vždy dosahuje lepší průměrné propustnosti než ostatní testované *high-speed TCP* varianty s různou velikostí bufferů na routerech.
  - ▶ Pracuje dobře v prostředích s homogenními i heterogenními *RTT* proudy. Může se ovšem chovat *RTT-unfriendly* v případě, že velikost bufferů je nízká (viz obr. 13b).
  - ▶ Protokol se chová *TCP-friendly* (viz obr. 16), robustně (dosahuje dobrých výsledků v přítomnosti výrazného zpětného provozu, viz obr. 18b).

## Reference a užitečné odkazy

-  *Wenjun Xu et al.: Hybrid congestion control for high-speed networks, 2011, <http://dx.doi.org/10.1016/j.jnca.2011.03.025>.*
  -  *Holub P.: Výlet za hranice TCP: protokoly pro síť s velkým součinem latence a šířky pásma, 2006, <[http://is.muni.cz/el/1433/jaro2006/PA160/um/fast\\_network\\_protocols.pdf](http://is.muni.cz/el/1433/jaro2006/PA160/um/fast_network_protocols.pdf)>.*
  -  *Floyd S.: Metrics for the Evaluation of Congestion Control Mechanisms, 2008, <<http://www.ietf.org/rfc/rfc5166.txt>>.*

Dotazy

## Nějaké dotazy?



Dotazy

## Nějaké dotazy?

Děkuji za Vaši pozornost

