Vliv velikosti vysílacího okna metody *Selective Repeat*

Jakub Adamec – adamej14@fel.cvut.cz
Daniel Petránek – petrada6@fel.cvut.cz

1. Princip

Protokol selektivního opakování (Selective Repeat) je protokol zpětné vazby s automatickým opakováním pro zajištění spolehlivé komunikace v síti. Je velmi podobný protokolu Go-Back-N, avšak s rozdílem, že se po ztrátě nějakého paketu posílá pouze daný paket a ne celé okno. Tímto způsobem zajišťujeme spolehlivou komunikaci na nespolehlivých linkách.

Odesílatel posílá několik paketů určených velikostí okna. Pakety si očísluje od jedné do n, kde n je velikost okna. Jakmile odešle celé okno (n paketů), čeká na potvrzení (ACK) od příjemce. Příjemce musí mít stejně veliké okno. Jestliže paket dorazí k příjemci, odesílá ihned ACK s číslem paketu. Zároveň si posouvá svoje okno. Tohle provádí pro celé okno. Jakmile odesílatel přijme ACK se správným číslováním paketů, posouvá si svoje okno a pokračuje v posílání.

1.1 Možné chyby při komunikaci

Jestliže paket či ACK nedorazí, okno se neposouvá. Pokud nedorazí i-tý paket, okno příjemce by se zastavilo na i-té pozici, přičemž pakety $i + 1$ až n si bude muset někde uložit. Pokud však nedorazí ACK, odesílatel okno neposouvá a čeká na timeout. Po uplynutí času timeout se chybějící paket znovu odešle.

2. Velikost okna

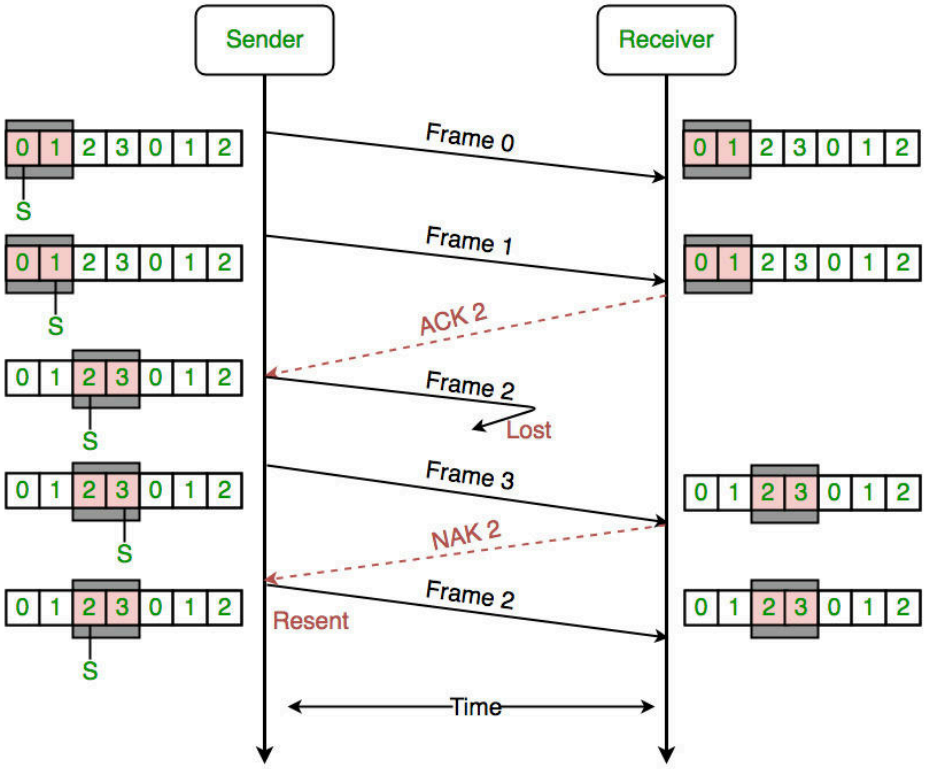
Maximální velikost okna by neměla přesáhnout polovinu rozsahu indexování paketů. Například pokud sender bude číslovat pakety pomocí čísel 0-5 (receiver stejně), a velikost okna by byla 4, mohou nastat situace, kdy sender bude posílat nějaké pakety znovu, protože mu nepřišla potvrzení, a receiver by mohl považovat tyto pakety za nové.

2.1 Příklad

Sender má okno, ve kterém ukládá pakety na poslání a indexuje je čísly 0, 1, 2, 3. Jakmile by poslal 4. paket, znovu ho indexuje číslem 0. Receiver má také okno, ve kterém čísluje pakety 0, 1, 2, 3. Jakmile mu přijde paket s indexem 0, celé okno se posouvá a po čísle 3 se indexuje opět od 0.

V tomto příkladu se posílají 2 pakety najednou (tedy okno spojení má velikost 2). Sender pošle 2 pakety a receiver pošle jedno potvrzení o přijetí celého okna. Při odesílání dalšího okna se ztratí první paket okna, což dá receiver najevo tím, že zašle negativní potvrzení (NAK) s číslem paketu, který je vadný, v tomto případě paket, kterému vypršel čas na doručení. Sender tuto chybu napraví znovuodesláním chybějícího paketu. Až po validním přijetí se sliding window receivera posune dále.

Praha, léto 2024



Obrázek 1: Příklad funkce sliding window

2.2 Minimální velikost okna pro dané zpoždění

Velikost okna by neměla být příliš velká, protože by si receiver musel rezervovat velké množství paměti. Zároveň by ale neměla být příliš malá, protože by se rychlost posílání příliš nelišila od metody *Stop & Wait*.

Mějme zpoždění $R_{\mathbb{T}}$, které lze spočítat jako:

$$R_{\mathbb{T}} = \text{čas přijetí odesílatelem} - \text{čas odeslání}$$

Zpoždění signálu (propagation delay, P) je tedy:

$$P = \frac{R_{\mathbb{T}}}{2}$$

Zpoždění přenosu (transmission delay, T) je doba, kterou trvá odeslání všech bitů paketů v rámci jednoho okna při dané šířce pásma:

$$T = \frac{N}{B}$$

kde N je velikost okna v bitech a B je šířka pásma v bitech za sekundu (bps).

Abychom maximalizovali využití šířky pásma, chceme, aby doba odeslání všech dat v rámci jednoho okna T byla rovna zpoždění signálu P:

$$T = P$$

$$\frac{N}{B} = P$$

$$N = B \cdot P$$

Toto je výsledná velikost okna v bitech pro dané zpoždění.

Praha, léto 2024

3. Ukázka testování pomocí NetDerperu

Nastavení testu

Pro testování vlivu velikosti vysílacího okna na datovou propustnost budeme používat program NetDerper. Simulujeme různé hodnoty zpoždění a velikostí oken, přičemž sledujeme dosaženou datovou propustnost.

Průběh testování

1. Nastavíme simulované zpoždění mezi koncovými uzly.
2. Ověříme minimální velikost vysílacího okna pro dané zpoždění pomocí výše uvedeného výpočtu.
3. Postupně zvyšujeme velikost vysílacího okna a zaznamenáváme dosaženou datovou propustnost.
4. Analyzujeme výsledky a ukážeme, že další zvyšování velikosti okna již nemá vliv na dosaženou datovou propustnost.

Ukázka komunikace ve Wiresharku

Pro každou testovanou konfiguraci zaznamenáme průběh komunikace pomocí programu Wireshark. To nám umožní vizualizovat potvrzovací mechanismus protokolu Selective Repeat a identifikovat případné chyby či opakovaná přenosy.

noindent{} Na obrázku níže je příklad zachyceného přenosu ve Wiresharku: TODO

Závěr

Na základě výsledků našich testů pomocí NetDerperu a analýzy komunikace ve Wiresharku lze demonstrovat, že po dosažení optimální velikosti vysílacího okna (pro dané zpoždění a šířku pásma) další zvyšování této velikosti již nevede k nárůstu datové propustnosti. Tento závěr je důležitý pro efektivní konfiguraci síťových protokolů v prostředích s různými hodnotami zpoždění.

Praha, léto 2024