Redes-De-Computadores\relatorio.md

Relatório – Análise do Impacto de Delay, Perda e Algoritmo TCP na Topologia Dumbbell

1. Introdução

Este relatório tem como objetivo analisar o impacto do **delay**, da **perda de pacotes (loss)** e dos **algoritmos de controle de congestionamento TCP** no desempenho da rede. A análise foi realizada em uma topologia do tipo **dumbbell**, que simula um ambiente de comunicação entre clientes e servidores através de um link central, onde os parâmetros de latência e perda de pacotes variaram.

---

2. Metodologia

A simulação foi realizada utilizando o **Mininet**, uma ferramenta amplamente utilizada para emulação de redes. A topologia é composta por:

- **2 clientes** (Host A e Host B)
- **2 roteadores** (R1 e R2)
- **2 servidores** (Server X e Server Y)

Os testes foram conduzidos em 27 cenários diferentes, seguindo:

- **Delays**: `0 ms`, `10 ms`, `100 ms`
- **Perda de pacotes (Loss)**: `0%`, `1%`, `5%`
- **Algoritmos TCP analisados**: `CUBIC`, `RENO`, `VEGAS`

Para cada cenário, foram coletadas as seguintes métricas:

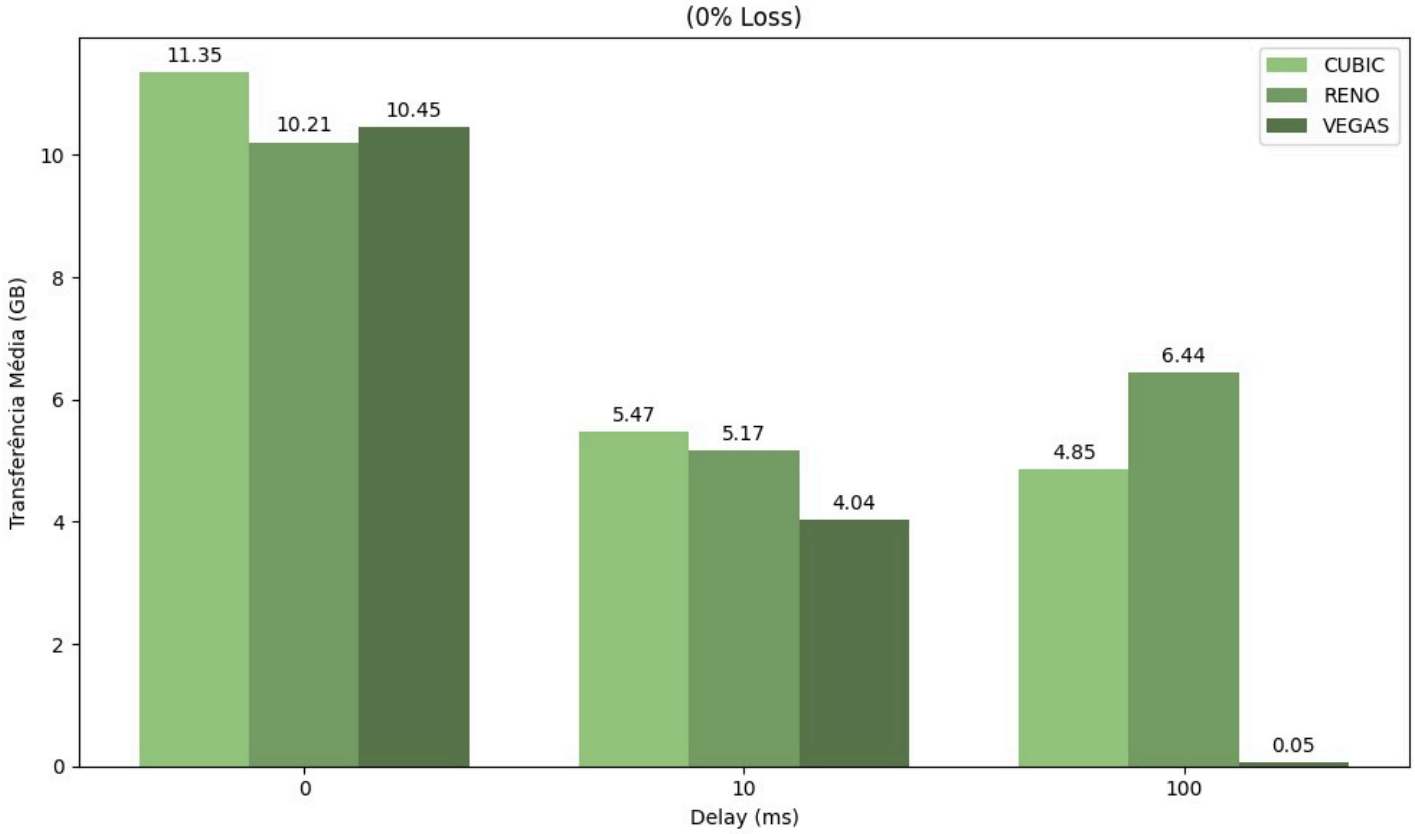
- **Transferência de dados (Transfer)**: Total de bytes transferidos.
- **Largura de banda (Bandwidth)**: Taxa de transferência em Mbps ou Gbps.

As medições foram realizadas utilizando o comando `iperf`.

---

3. Resultados

3.1 Desempenho com 0% de perda de pacotes



Algoritmo **CUBIC**

Delay (ms)	Transferência (min/méd/max)	Banda (min/méd/max)
0	10.5 / 11.35 / 12.2 GB	16.3 / 18.0 / 19.6 Gbps
10	5.12 / 5.47 / 5.82 GB	7.96 / 8.54 / 9.13 Gbps
100	4.64 / 4.85 / 5.05 GB	7.31 / 7.66 / 8.01 Gbps

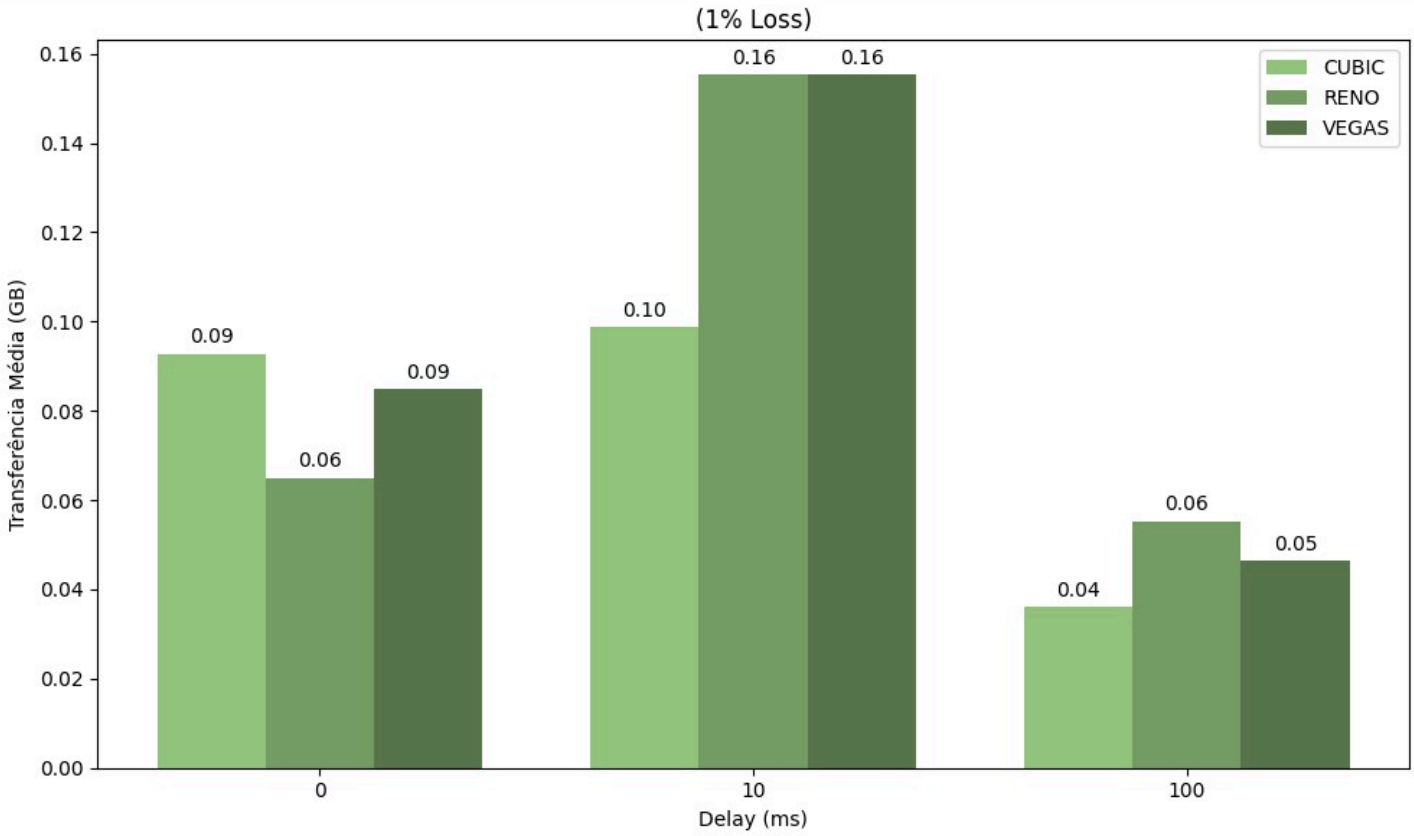
Algoritmo **RENO**

Delay (ms)	Transferência (min/méd/max)	Banda (min/méd/max)
0	8.21 / 10.21 / 12.2 GB	13.1 / 14.1 / 15.1 Gbps
10	5.12 / 5.17 / 5.22 GB	8.28 / 8.29 / 8.31 Gbps
100	5.85 / 6.44 / 7.03 GB	7.27 / 9.53 / 11.3 Gbps

Algoritmo **VEGAS**

Delay (ms)	Transferência (min/méd/max)	Banda (min/méd/max)
0	10.2 / 10.45 / 10.7 GB	16.5 / 16.6 / 16.7 Gbps
10	3.79 / 4.04 / 4.29 GB	6.05 / 6.37 / 6.70 Gbps
100	21.8 / 55.8 / 89.8 MB	33.9 / 87.0 / 140 Mbps

3.2 Desempenho com 1% de perda de pacotes



Algoritmo **CUBIC**

Delay (ms)	Transferência (min/méd/max)	Banda (min/méd/max)
0	36.1 / 95.05 / 154 MB	54.0 / 145.5 / 225 Mbps
10	98.2 / 101.1 / 104 MB	148 / 152.5 / 157 Mbps
100	26.9 / 36.85 / 46.8 MB	38.7 / 51.8 / 64.9 Mbps

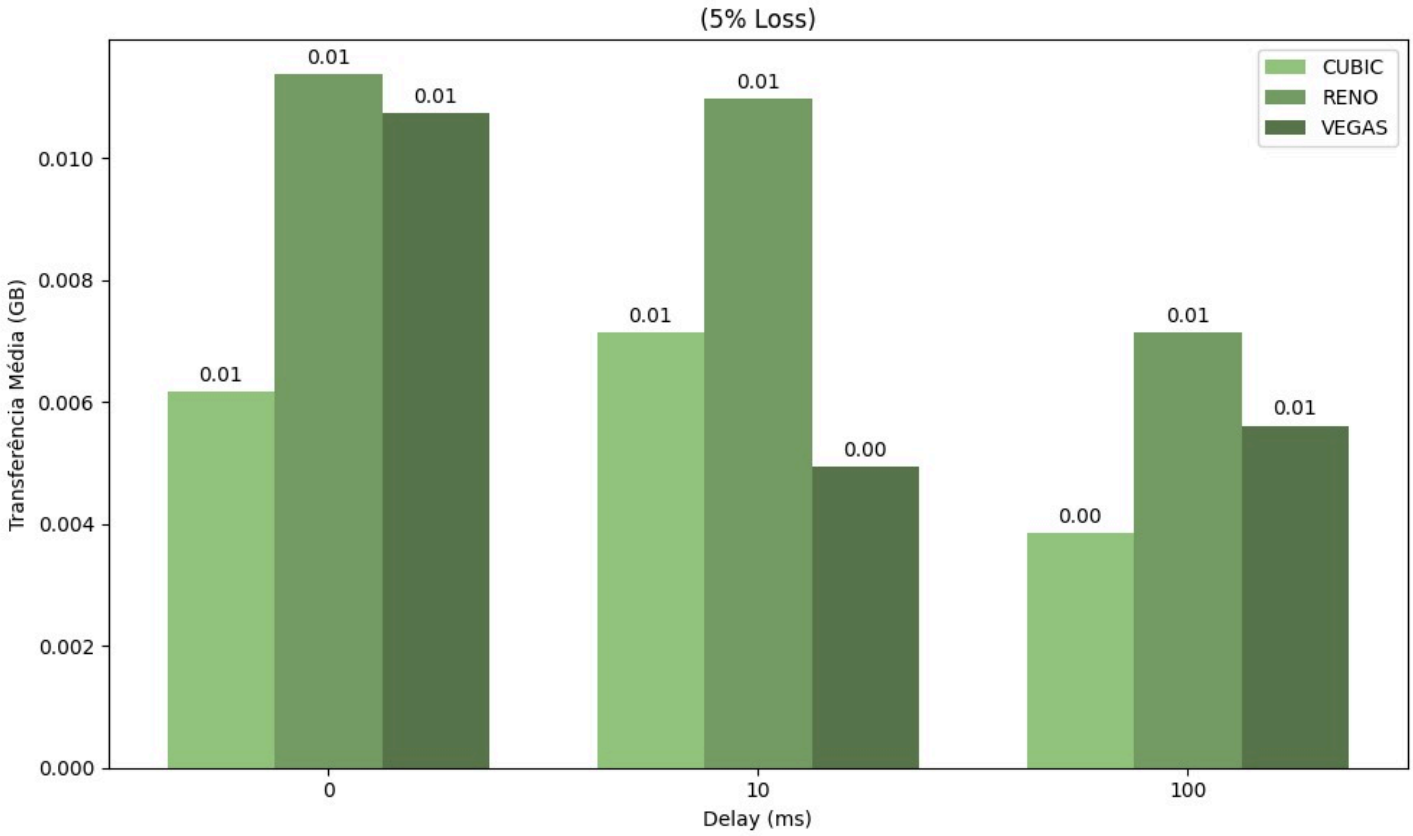
Algoritmo **RENO**

Delay (ms)	Transferência (min/méd/max)	Banda (min/méd/max)
0	61.0 / 66.55 / 72.1 MB	92.8 / 98.9 / 105 Mbps
10	144 / 159 / 174 MB	216 / 237.5 / 259 Mbps
100	42.2 / 56.65 / 71.1 MB	61.3 / 77.95 / 94.6 Mbps

Algoritmo **VEGAS**

Delay (ms)	Transferência (min/méd/max)	Banda (min/méd/max)
0	85.0 / 87.05 / 89.1 MB	129 / 130 / 131 Mbps
10	154 / 159 / 164 MB	220 / 232 / 244 Mbps
100	39.5 / 47.65 / 55.8 MB	60.9 / 72.5 / 84.1 Mbps

3.3 Desempenho com 5% de perda de pacotes



Algoritmo **CUBIC**

Delay (ms)	Transferência (min/méd/max)	Banda (min/méd/max)
0	6.25 / 6.32 / 6.38 MB	8.07 / 8.54 / 9.01 Mbps
10	5.38 / 7.31 / 9.25 MB	7.66 / 9.85 / 12.4 Mbps
100	2.75 / 3.94 / 5.12 MB	3.71 / 5.74 / 7.78 Mbps

Algoritmo **RENO**

Delay (ms)	Transferência (min/méd/max)	Banda (min/méd/max)
0	11.4 / 11.65 / 11.9 MB	14.9 / 15.35 / 16.8 Mbps
10	10.4 / 11.25 / 12.1 MB	14.6 / 15.95 / 17.3 Mbps
100	5.88 / 7.32 / 8.75 MB	8.67 / 10.69 / 12.7 Mbps

Algoritmo **VEGAS**

Delay (ms)	Transferência (min/méd/max)	Banda (min/méd/max)
0	10.1 / 11.0 / 11.9 MB	15.3 / 15.4 / 15.5 Mbps
10	4.38 / 5.06 / 5.75 MB	6.21 / 6.97 / 7.72 Mbps
100	5.38 / 5.75 / 6.12 MB	7.45 / 8.08 / 8.53 Mbps

4. Discussão

Qual algoritmo se comporta melhor em cenários com alta latência?

O algoritmo **CUBIC** um desempenho melhor em situações de alta latência (100 ms). Ele manteve uma largura de banda mais estável e consistente em comparação aos outros algoritmos.

O algoritmo **VEGAS**, embora tenha apresentado desempenho inferior ao **CUBIC**, continua sendo efetivo em baixa latência, mas sua performance caiu com o aumento do delay.

Qual algoritmo se adapta melhor à perda?

O algoritmo **RENO** demonstrou maior consistência em cenários com perda de pacotes, mantendo uma largura de banda relativamente alta mesmo em condições ruins.

O algoritmo **VEGAS** teve dificuldades em lidar com cenários de alta perda de pacotes, apresentando uma queda na largura de banda.

Como a combinação de delay e loss afeta a performance?

A combinação de **alta latência** e **alta perda de pacotes** resultou em uma piora grande no desempenho de todos os algoritmos. No entanto:

- **CUBIC** foi o mais afetado.
- **RENO** conseguiu manter um desempenho mais consistente.
- **VEGAS** apresentou o pior desempenho em cenários com alta perda e alta latência.

---

5. Conclusão

Com base nos testes realizados, conclui-se que:

- O algoritmo **CUBIC** é mais adequado para cenários com alta latência e baixa perda.
- O algoritmo **RENO** é mais robusto em cenários com alta perda de pacotes.
- O algoritmo **VEGAS** é eficiente em cenários de baixa latência, mas apresenta dificuldades em condições que não sejam ideais.
- A combinação de alta latência e alta perda impacta negativamente todos os algoritmos, mas **RENO** apresenta a melhor consistência.

Esses resultados destacam a importância de escolher o algoritmo TCP adequado para diferentes condições de rede.
