南開大學
Nankai University

计算机学院

计算机网络实验报告

3-4: 基于 UDP 服务设计可靠传输协议并编程实现

姓名：李欣

学号：2011165

专业：计算机科学与技术

2022 年 12 月 30 日

目录

1 实验要求

基于给定的实验测试环境，通过改变延迟时间和丢包率，完成下面 3 组性能对比实验：

1. 停等机制与滑动窗口机制性能对比；
2. 滑动窗口机制中不同窗口大小对性能的影响；
3. 有拥塞控制和无拥塞控制的性能比较。

以下实验数据均为传送 1.jpg 时程序的表现，表格内数值为多次实验 (>=3) 的平均值。

2 停等机制与滑动窗口机制

2.1 延时相同，改变丢包率

在**延时相同 =0ms** 的情况下，逐渐增大丢包率，比较停等机制和滑动窗口机制的性能。测试结果如表格所示。

下表的单位为秒 (s)，记录了时延，即文件的传输时间。

丢包率	停等机制	滑动窗口机制
0.0%	3.212	5.614
2.0%	16.904	14.886
4.0%	20.471	19.626
6.0%	25.696	25.796
8.0%	39.136	32.741

表 1: 时延/丢包率 (停等 vs GBN 延时 =0ms)

将表格绘制成折线图，观察起来更加清晰明了。

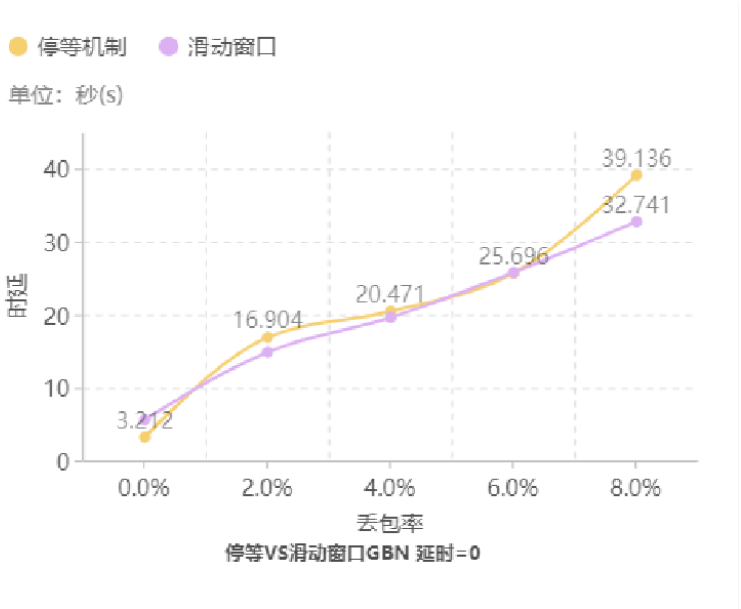

图 2.1: 时延/丢包率 (停等 vs GBN 延时 =0ms)

丢包率	停等机制	滑动窗口机制
0.0%	4689.250	2682.910
2.0%	891.025	1011.820
4.0%	735.767	767.445
6.0%	586.157	583.884
8.0%	384.860	460.031

表 2: 吞吐率/丢包率 (停等 vs GBN 延时 =0ms)

下表的单位为 kbps，记录了吞吐量，即网络的传输能力。

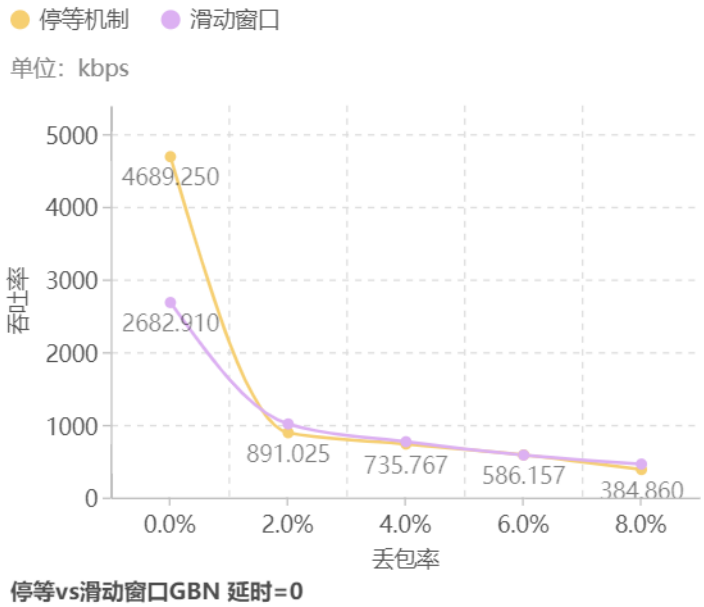

图 2.2: 吞吐率/丢包率 (停等 vs GBN 延时 =0ms)

分析: 当延时固定为 0ms 时，逐渐增加丢包率，在丢包率较小的时候，停等和滑动窗口机制的性能相当，当丢包率较大时，如丢包率为 10% 时，滑动窗口传送 1.jpg 的时延明显小于停等，且网络的吞吐率更大，此时，滑动窗口的性能更好一些。

2.2 丢包率相同，改变延时

为了测试程序的鲁棒性，选取了两个固定的丢包率 (0% 和 4%), 分别在这两个固定的丢包率下测试不同延时对停等机制和滑动窗口机制的影响，比较二者的性能。

2.2.1 固定丢包率为 0%

在**丢包率固定为 0%** 的情况下，逐渐增大延时，比较停等机制和滑动窗口机制的性能。测试结果如表格所示。

延时	停等机制	滑动窗口机制
0ms	3.297	5.262
3ms	30.796	29.479
30ms	85.641	84.676
60ms	141.622	141.232
120ms	254.465	253.857
240ms	473.831	450.926

表 3: 时延/延时 (停等 vs GBN 丢包率 =0%)

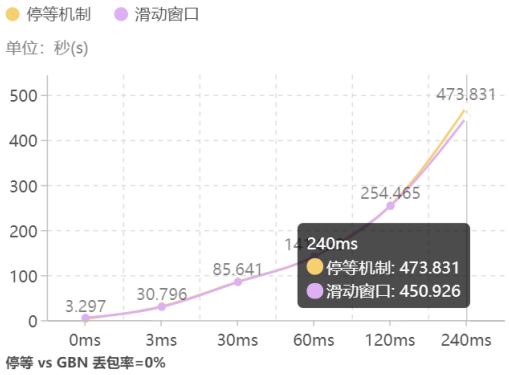

图 2.3: 时延/延时 (停等 vs GBN 丢包率 =0%)

延时	停等机制	滑动窗口机制
0ms	4568.36	2862.39
3ms	489.086	510.936
30ms	175.872	177.877
60ms	106.353	106.646
120ms	59.1904	59.3321
240ms	31.7875	33.4021

表 4: 吞吐率/丢包率 (停等 vs GBN 丢包率 =0%)

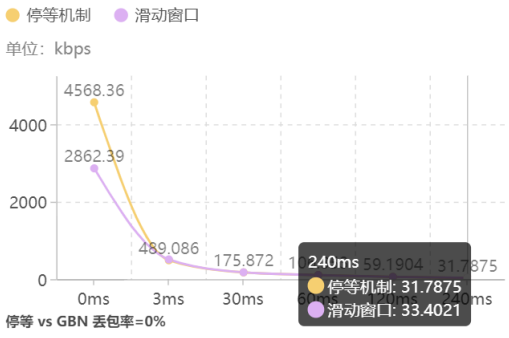

图 2.4: 吞吐率/丢包率 (停等 vs GBN 丢包率 =0%)

分析： 当固定丢包率为 0% 时，随着延时的增加，停等机制和滑动窗口性能表现相当，在延时达

到 240ms 时，滑动窗口机制表现得性能略优于停等机制。

2.2.2 固定丢包率为 4%

在**丢包率固定为 4%** 的情况下，逐渐增大延时，比较停等机制和滑动窗口机制的性能。测试结果如表格所示。

延时	停等机制	滑动窗口机制
0ms	18.915	22.385
3ms	56.82	64.681
30ms	94.155	109.131
60ms	151.159	179.313
120ms	270.692	320.504

表 5: 时延/延时 (停等 vs GBN 丢包率 =4%)

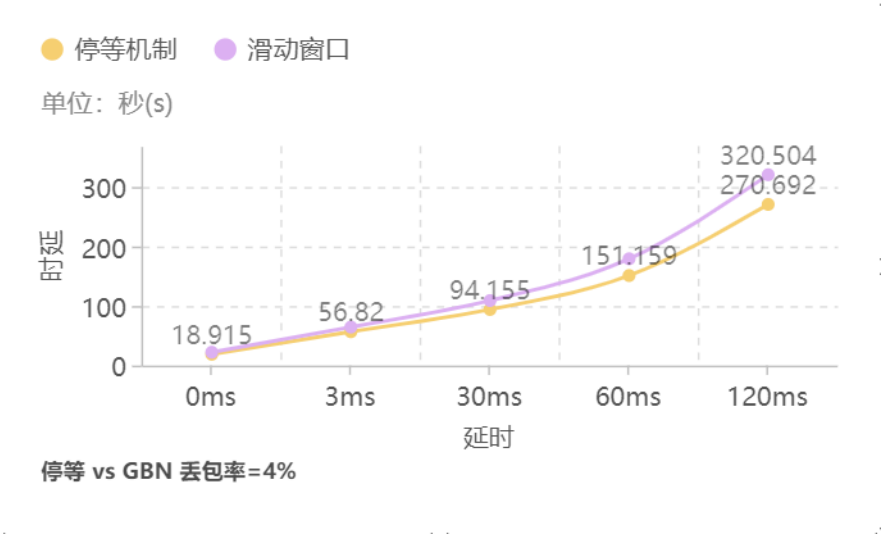

图 2.5: 时延/丢包率 (停等 vs GBN 丢包率 =4%)

延时	停等机制	滑动窗口机制
0ms	796.293	672.856
3ms	265.081	232.864
30ms	159.969	138.017
60ms	99.6426	83.9977
120ms	55.6421	46.9944

表 6: 吞吐率/丢包率 (停等 vs GBN 丢包率 =4%)

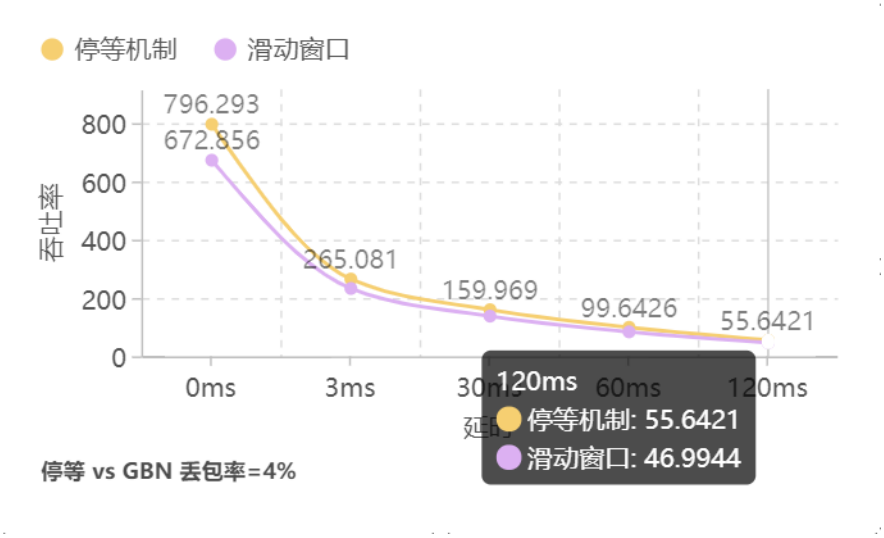

图 2.6: 吞吐率/丢包率 (停等 vs GBN 丢包率 =4%)

分析：当固定丢包率为 4% 时, 增大延时，滑动窗口 GBN 得表现逊色于停等，即在网络条件较差得情况下，滑动窗口 GBN 得效果并不如预期得好。

2.3 总结

1. 在延时较低，丢包率较低的情况下，二者性能相当；
2. 在延时较低，丢包率较大的情况下，滑动窗口表现得性能更佳 (时延更小，吞吐率更大)，虽然更大的窗口代表了更大的重传代价，得到这样的实验结果是因为，我们实验开的窗口是 4 个，并不是很大，重传开销的影响远没有多线程 (接收 ack 节省的时间) 的影响大。
3. 在延时较大，丢包率较小的情况下，滑动窗口机制性能表现更好 (时延更小，吞吐率更大)，滑动窗口机制开启了多线程，且能同时发送多个数据包，减少了 RTT 的影响。
4. 在网络状况较差，即延时较大、丢包率较大得情况下，二者表现出性能相当。

3 滑动窗口机制中不同窗口大小

3.1 延时相同，改变丢包率

丢包率	窗口 =4	窗口 =6	窗口 =8
0.0%	20.408	19.798	18.121
2.0%	37.655	38.338	37.587
4.0%	49.925	44.645	45.24
6.0%	73.551	92.794	115.846
8.0%	76.281	93.096	123.857

表 7: 时延/丢包率 (4 vs 6 vs 8，延时 =0ms)

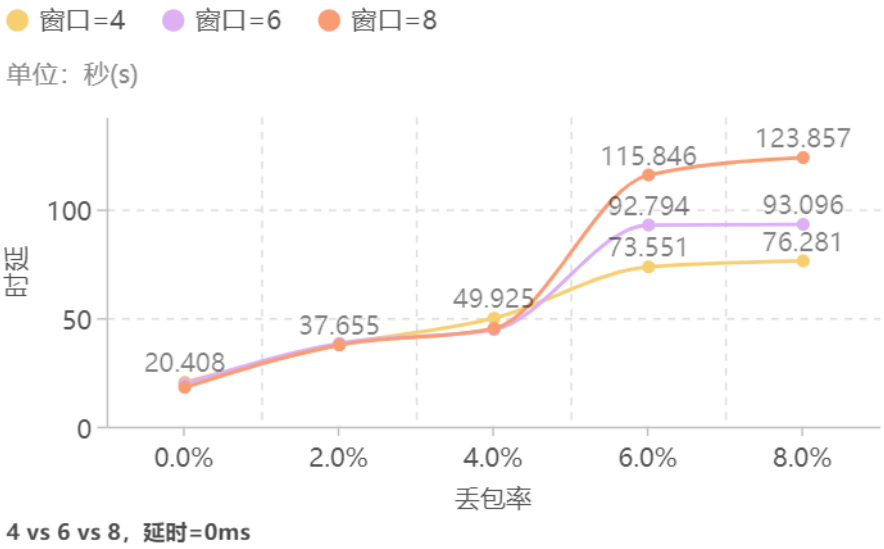

图 3.7: 窗口：4 vs 6 vs 8，延时 =0ms

丢包率	窗口 =4	窗口 =6	窗口 =8
0.0%	738.038	760.778	831.184
2.0%	399.997	392.871	400.72
4.0%	301.690	337.370	332.933
6.0%	204.781	162.315	130.016
8.0%	197.453	161.789	121.607

表 8: 吞吐率/丢包率 (4 vs 6 vs 8，延时 =0ms)

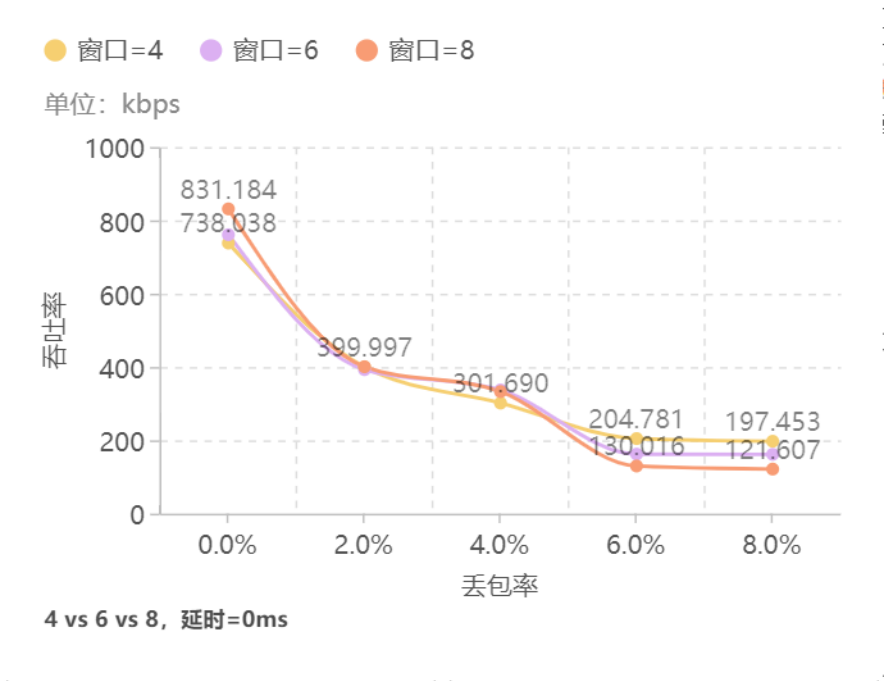

图 3.8: 窗口：4 vs 6 vs 8，延时 =0ms

分析：在延时为 0 时，当丢包率较小 (0%,2%) 时，窗口越大，延时越低，吞吐率越大，当丢包率较大时，窗口越大，重传开销越大，时延也相应的增加了。

3.2 丢包率相同，改变延时

3.2.1 固定丢包率为 0%

延时	窗口 =4	窗口 =6	窗口 =8
0ms	20.408	19.798	18.121
50ms	133.616	133.298	133.716

表 9: 时延/丢包率 (4 vs 6 vs 8，丢包率 =0%)

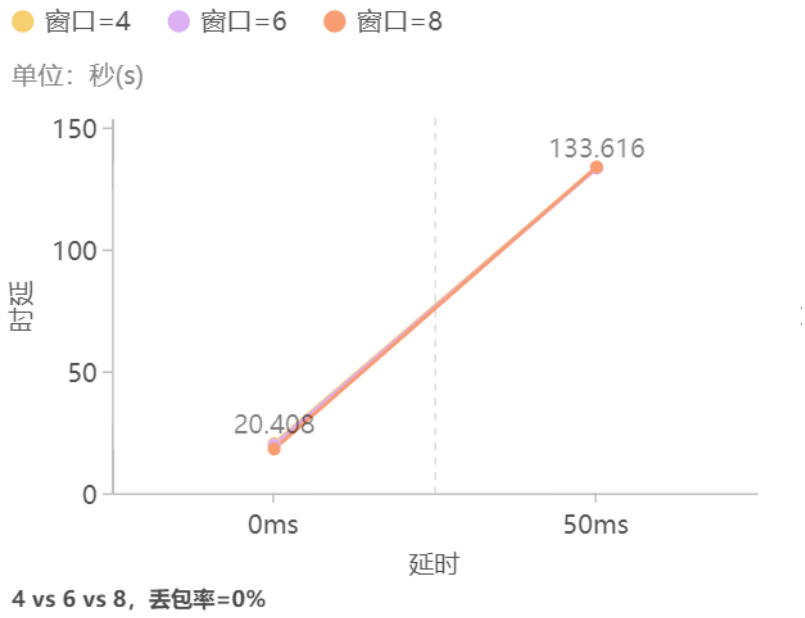

图 3.9: 窗口：4 vs 6 vs 8，丢包率 =0%

延时	窗口 =4	窗口 =6	窗口 =8
0ms	738.038	760.778	831.184
50ms	112.725	112.994	112.641

表 10: 吞吐率/丢包率 (4 vs 6 vs 8，丢包率 =0%)

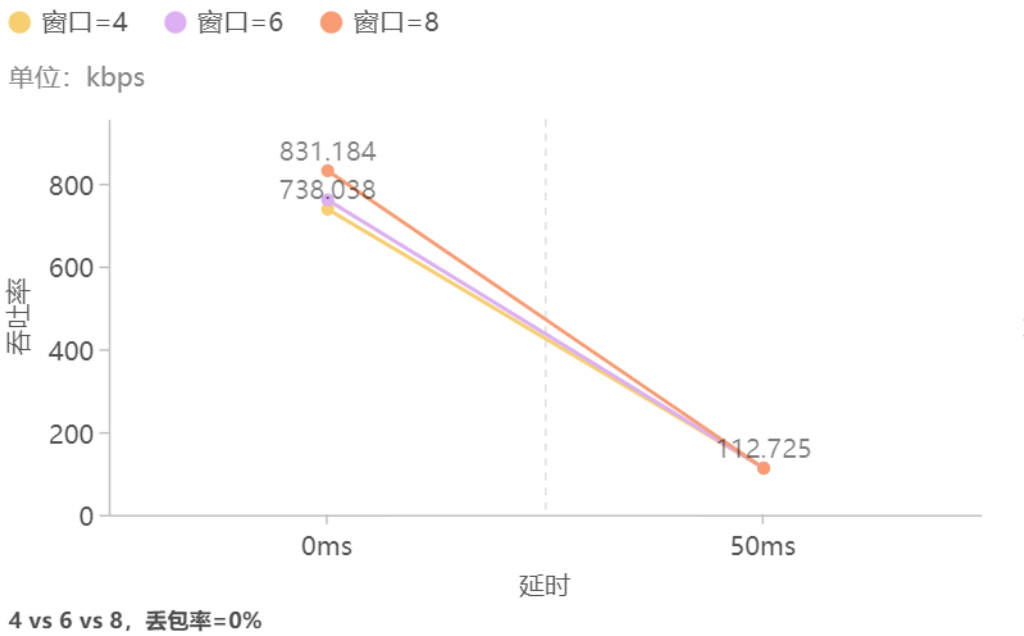

图 3.10: 窗口：4 vs 6 vs 8，丢包率 =0%

分析：当固定丢包率为 0% 时，窗口对时延和吞吐率的影响不是很大。

3.2.2 固定丢包率为 2%

延时	窗口 =4	窗口 =6	窗口 =8
0ms	38.264	39.023	37.313
50ms	231.472	239.354	311.441

表 11: 时延/丢包率 (4 vs 6 vs 8，丢包率 =2%)

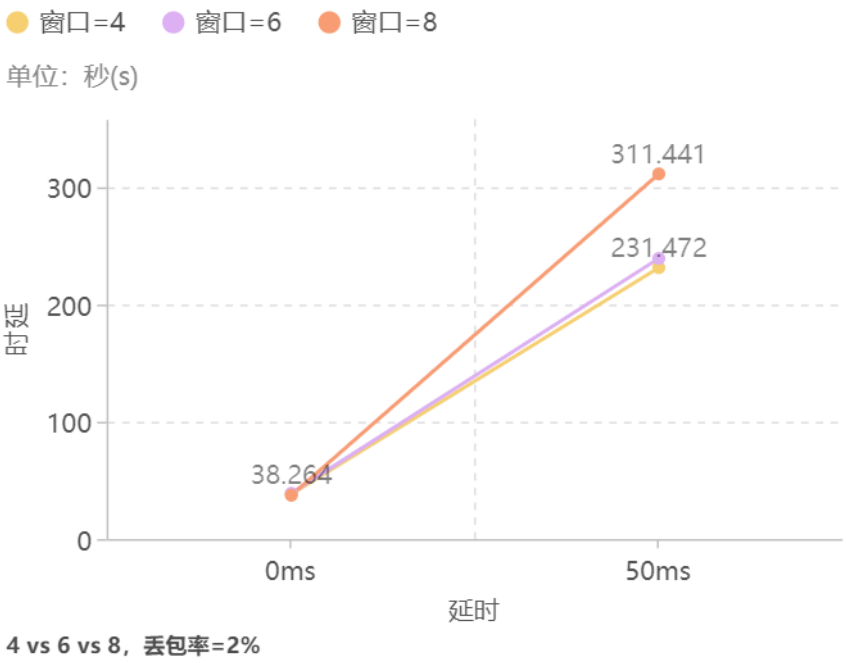

图 3.11: 窗口：4 vs 6 vs 8，丢包率 =2%

延时	窗口 =4	窗口 =6	窗口 =8
0ms	393.631	385.974	403.663
50ms	65.070	62.9272	48.3619

表 12: 吞吐率/丢包率 (4 vs 6 vs 8，丢包率 =2%)

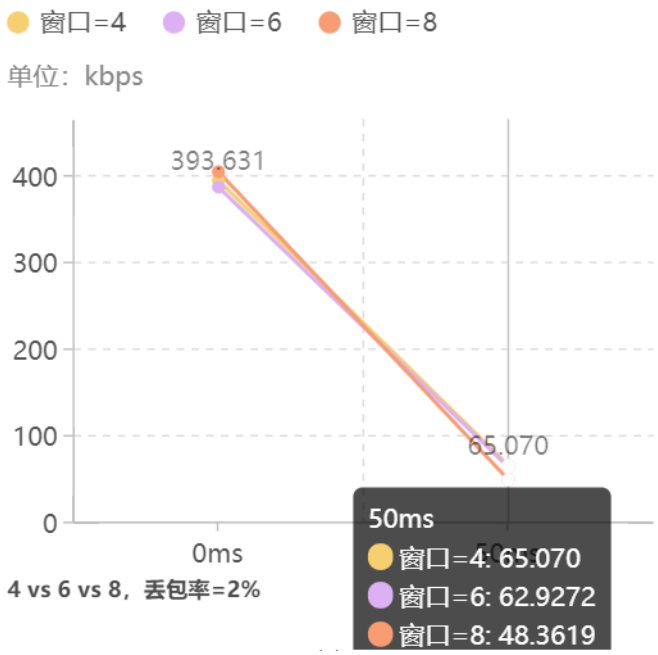

图 3.12: 窗口：4 vs 6 vs 8，丢包率 =2%

分析：当固定丢包率为 2% 时，当延时较小时，窗口越大，时延越小，网络吞吐率越大。当延时较大时，窗口越小，时延越小，网络吞吐率越大。

3.3 总结

1. 在延时较小，丢包率较小的情况下，由于开启了多线程，窗口越大效率越高，时延越小，吞吐率越大。
2. 在网络情况较差，即延时较大或丢包率较大的情况下，实验结果显示窗口显示窗口越小效率越高，即时延越小，吞吐率越大，这是由于滑动窗口重传开销比较大。
3. 窗口大小虽然不同，对于网络情况的变化，整体的变化趋势是相同的。

4 有拥塞控制和无拥塞控制

4.1 延时相同，改变丢包率

丢包率	无拥塞控制	有拥塞控制
2.0%	29.77	31.649
4.0%	54.641	79.335
6.0%	86.227	116.41
8.0%	123.262	127.706
10.0%	162.83	149.573
12.0%	238.133	180.757

表 13: 时延/丢包率 (有拥塞控制 vs 无拥塞控制，延时 =0ms)

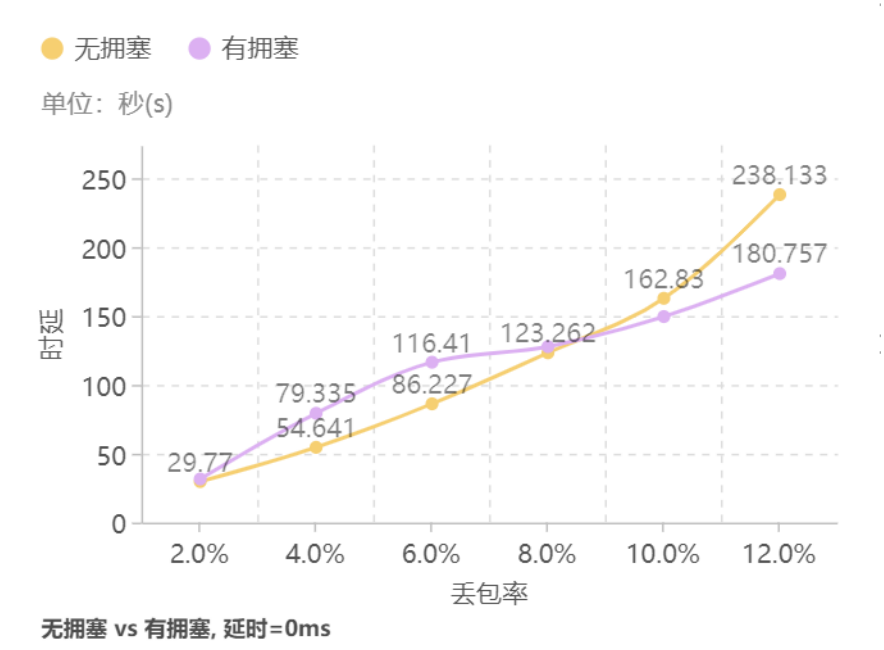

图 4.13: 有拥塞控制 vs 无拥塞控制，延时 =0ms

丢包率	无拥塞控制	有拥塞控制
2.0%	505.942	478.653
4.0%	275.652	190.949
6.0%	174.677	130.134
8.0%	122.194	118.623
10.0%	92.501	101.281
12.0%	63.250	83.808

表 14: 吞吐率/丢包率 (有拥塞控制 vs 无拥塞控制，延时 =0ms)

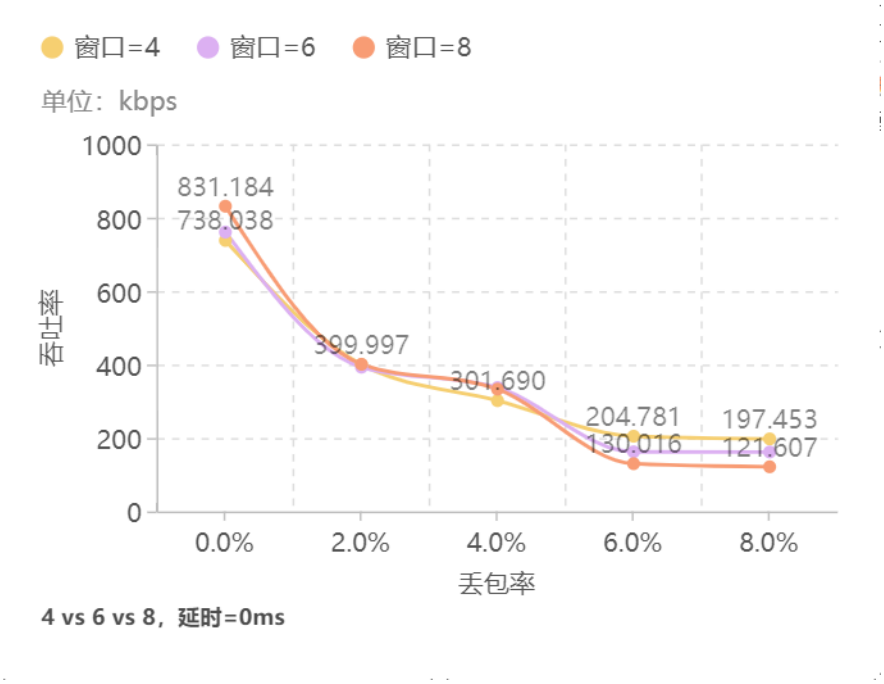

图 4.14: 窗口：4 vs 6 vs 8，延时 =0ms

分析：当延时 =0ms 时，改变丢包率，当丢包率较小 (<=8%) 时，无拥塞控制的性能优于有拥塞控制，当丢包率较大时 (>=10%)，有拥塞控制的性能明显优于无拥塞控制。

4.2 丢包率相同，改变延时

延时	无拥塞控制	有拥塞控制
0ms	31.649	29.77
30ms	149.012	125.543
60ms	226.764	180.873
120ms	282.185	431.316

表 15: 时延/丢包率 (无拥塞控制 vs 有拥塞控制，丢包率 =2%)

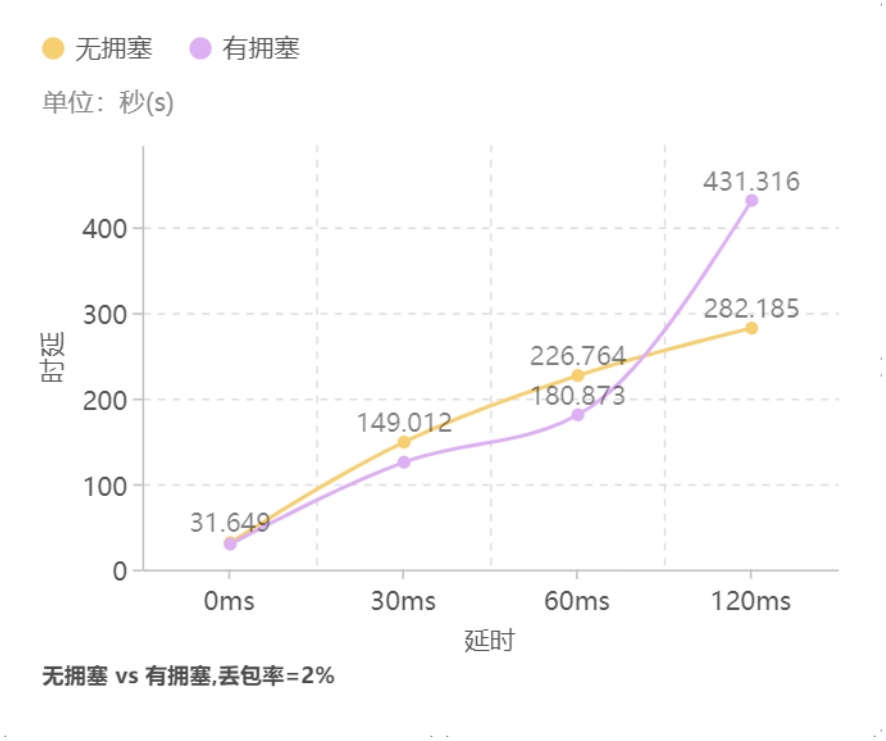

图 4.15: 有拥塞控制 vs 无拥塞控制，丢包率 =2%

延时	无拥塞控制	有拥塞控制
0ms	478.653	505.942
30ms	101.662	119.974
60ms	66.8047	83.2732
120ms	53.3759	35.1225

表 16: 吞吐率/丢包率 (有拥塞控制 vs 无拥塞控制，丢包率 =2%)

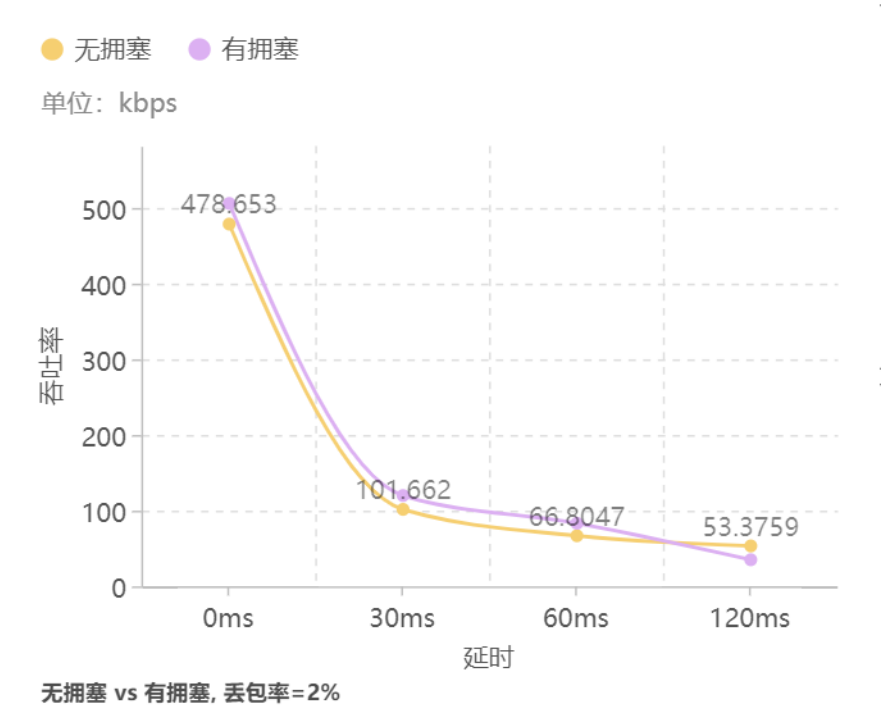

图 4.16: 有拥塞控制 vs 无拥塞控制，丢包率 =2%

分析：当固定丢包率为 2% 时，改变延时，当延时较小 (<=60ms) 时，二者效率相当，有拥塞效率略高于有拥塞，而当延时较大时，无拥塞控制的效率明显高于有拥塞。

4.3 总结

1. 当延时较小、丢包率较小时，有拥塞控制和无拥塞控制性能相当，无拥塞控制性能略好，此时网络情况良好，有无拥塞都能很好的进行传输。
2. 当延时较小、丢包率较大时，有拥塞控制的效率高于无拥塞控制，这是由于有拥塞控制能够根据网络丢包的状况调整窗口的大小，使得传输更加合理，合理的减小了开销。
3. 当网络不稳定，情况不太好时，即时延影响较大时，有拥塞控制可能就不如无拥塞控制效率高了，频繁变动窗口，窗口的大小变得不合理（窗口过大或过小）。

5 实验总结

由于时延的设置，在实验的过程中，需要根据实际情况调整超时等待时间，否则就会出现接收端回复了 ACK，但是发送端没有收到 ack 的情况。

现实网络情况变幻莫测，我们应该根据实际情况，选取合适的方法，努力提高网络传输效率。