作业要求：基于给定的实验测试环境，通过改变延时和丢包率

➢ 控制变量法：对比时要控制单一变量（算法、窗口大小、延时、丢包率）

➢ Router：可能会有较大延时，传输速率不作为评分依据，也可自行设计

➢ 延时、丢包率对比设置：要有梯度（例如 30ms,50ms, …；5%，10%，…）

➢ 测试文件：必须使用助教发的测试文件（1.jpg、2.jpg、3.jpg、helloworld.txt）

➢ 性能测试指标：时延、吞吐率，要给出图、表并进行分析

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；

控制单一变量，延时、丢包率相同，
滑动窗口机制中窗口大小为 10，分别测试 1.jpg、2.jpg、3.jpg、helloworld.txt
为使结论更具有普遍性，重复五次实验，最后取平均值，
当不设置丢包率、延迟时间时，以 1.jpg 测试数据为例，
测试指标：时延、吞吐率

本机传输本机不设置丢包和时延时				
次数	吞吐率（bytes/ms）		时延（ms）	
	停等机制	滑动窗口	停等机制	滑动窗口
1	124.7	3312.7	15160	558
2	110.4	3582.2	17528	525
3	108.3	3530.7	17304	522
4	117.9	3267.4	16325	553
5	114.4	3250.6	16520	566
平均值	115.2	3388.7	16567.4	544.8

在不设置延迟时间以及丢包率时，滑动窗口机制的传输效率与停等机制相比大大提升了。
猜测原因：差距悬殊的原因主要是由于实验是本机向本机传输文件，没有延迟时间，没有丢包率，UDP 向下封装发送的过程较耗时，停等机制不仅发送时而且接收时也要等待这样的阶段。

下为在丢包率梯度增大，对每次实验进行五次重复实验取平均值，计时器超时重传时间为 200m 控制变量，丢包率分别为 1%，3%,5%,10%,15%,20%

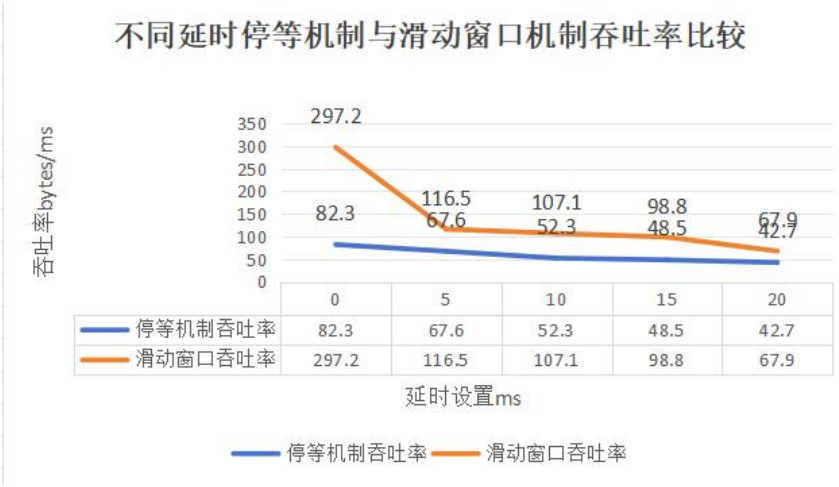
实验数据如下：

计时器超时重传时间为 200ms控制变量，丢包率分别为1%，3%, 5%, 10%, 15%, 20%				
丢包率	吞吐率（bytes/ms）		时延（ms）	
	停等机制	滑动窗口	停等机制	滑动窗口
1%	102.2	758.2	18740	2433
3%	95.8	644.1	19837	2815
5%	84.1	292.2	21006	6249
10%	68.3	197.5	26859	9983
15%	52.7	110.8	34558	16832
20%	44.9	77.4	40061	24416

滑动窗口在的丢包率逐渐变大时，性能下降，当丢包率达到 20% 时基本与停等机制的效率相同。
滑动窗口要重新发送窗口内的数据分组，如果丢包率进一步扩大，可能出现滑动窗口的效率更低，因为重传的过程之中也可能继续丢包。

下为延迟时间梯度增大的实验，丢包率为 5%控制变量，改变延迟时间分别设置0ms,5ms,10ms,15ms,20ms，每个时延五次重复实验取平均值，实验数据如下：

丢包率为 5%控制变量，改变延迟时间分别设置0ms, 5ms, 10ms, 15ms, 20ms				
时延(ms)	吞吐率（bytes/ms）		时延（ms）	
	停等机制	滑动窗口	停等机制	滑动窗口
0	82.3	297.2	22870	6320
5	67.6	116.5	29912	16836
10	52.3	107.1	36420	19883
15	48.5	98.8	39605	23105
20	42.7	67.9	43550	28522



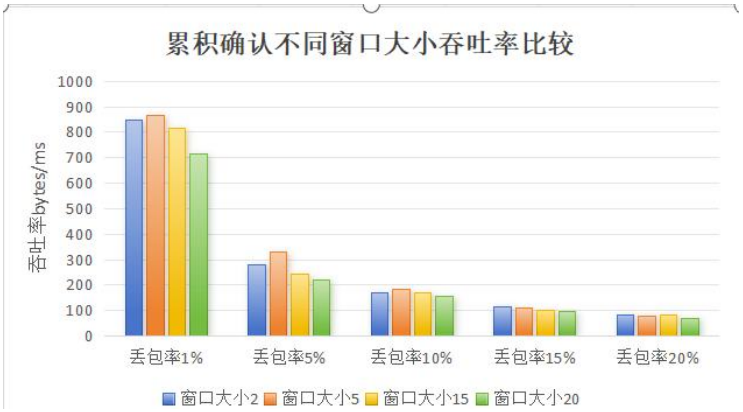
滑动窗口在 0ms 到 5ms 时斜率大，即传输效率变化大，猜测：UDP 向下封装传输占据了大部分的时间，停等机制需要一直等 ACK 响应；
5ms 到 20ms 的滑动窗口的吞吐率斜率较为平缓，猜测：发送每个分组到达的时间差不多，但是滑动窗口可能会重传更多的数据分组有更长的时延。

（2）滑动窗口机制中不同窗口大小对性能的影响（累计确认和选择确认两种情形）；

滑动窗口太小，无法充分发挥滑动窗口的优势快速地传输文件，滑动窗口太大，收发两方的缓冲区过大，占用硬件资源 ，而且 router 缓冲区固定大小，会因为缓冲区已满丢弃多余的分组导致性能下降。
窗口大小分别为 2,5,10,20 时，分别在不同的丢包率和延迟时间下，
累计确认，选择确认，
分别测试 1.jpg、2.jpg、3.jpg、helloworld.txt，以 1.jpg 为例，
测试指标：时延、吞吐率

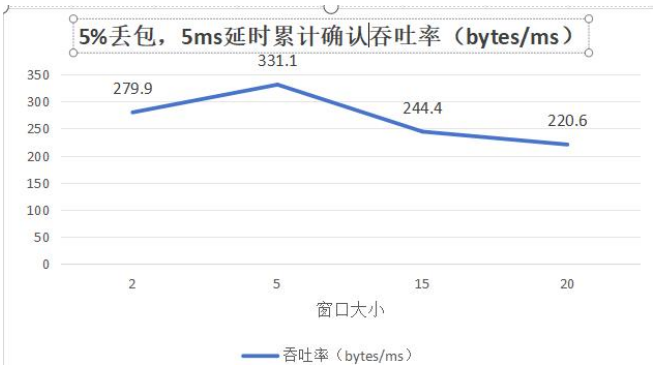
在不同的丢包率和不同的延迟时间下,设置滑动窗口的大小分别为 2,5,10,20，五次重复实验取平均值，
实验数据如下：

累计确认（GBN）窗口大小分别为2，5，15，20下在丢包率分别为1%，5%，10%，15%的吞吐率				
	吞吐率（bytes/ms）			
丢包率	2	5	15	20
1%	850.2	864.8	814.2	715.8
5%	279.9	331.1	244.4	220.6
10%	168.4	184.2	170.6	155.6
15%	113.3	112.4	102.4	94.5
20%	84.2	76.7	80.9	70.7



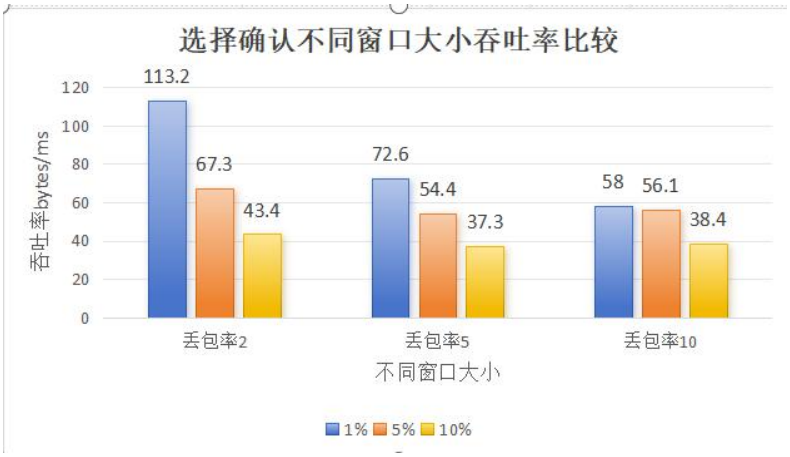
丢包率小于等于 10% 时，对比窗口大小为 2 和 5 的数据，可以发现此时增大数据窗口能够获得一定的传输效率提升，但是窗口大小超过 10 之后反而没有获得传输效率的提升，这是因为窗口大小变大之后，对 GBN 每次丢包需要重传的数据分组就会增加，重传的数据分组量增加就带来更多的时间消耗，并且多重传的数据分组也会带来多的丢包情形，导致传输效率的下降，我们知道不同的丢包率下的最佳的滑动窗口大小是不相同的。

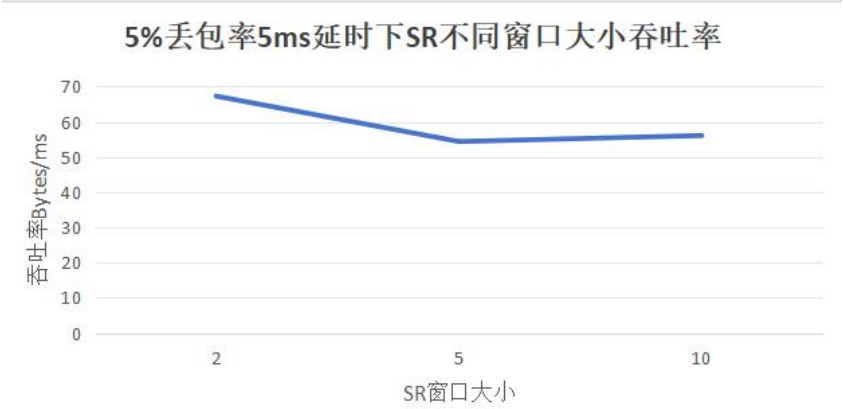
当丢包率大于 10% 时，不同窗口大小的传输效率几乎相同了， 因为丢包而产生的时间消耗占据了大多时间。



延时 5ms 不变，丢包率分别为 1%，5%,10%下选择重传（SR）窗口大小分别为 2，5，10 条件下，测试数据如下：

选择重传（SR）窗口大小分别为2，5，15下在丢包率分别为1%，5%，10%的吞吐率及延时						
	吞吐率（bytes/ms）			延时（s）		
丢包率	2	5	10	2	5	10
1%	113.2	72.6	58	16.4	25.5	31.8
5%	67.3	54.4	56.1	27.6	34.1	33.2
10%	43.4	37.3	38.4	42.8	49.8	48.4





当窗口大小较小时，发送方一次只能发送少量的数据段，这会导致发送方发送速率较慢，从而降低了吞吐率。窗口较大时，发送方可以同时发送多个数据段，这提高了发送速率和吞吐率。接收方可以并行接收多个数据段，并及时确认已接收的数据段，

但是窗口大小的选择应该与网络状况相匹配。如果网络状况较差，丢包率较高，选择较小的窗口大小可以减少重传次数，提高传输成功率。

（3）滑动窗口机制中相同窗口大小情况下，累计确认和选择确认的性能比较。

控制单一变量，窗口大小 16 时

不同延时 5ms，10ms,30ms,50ms，

不同丢包率 5%,10%,15%,20%,

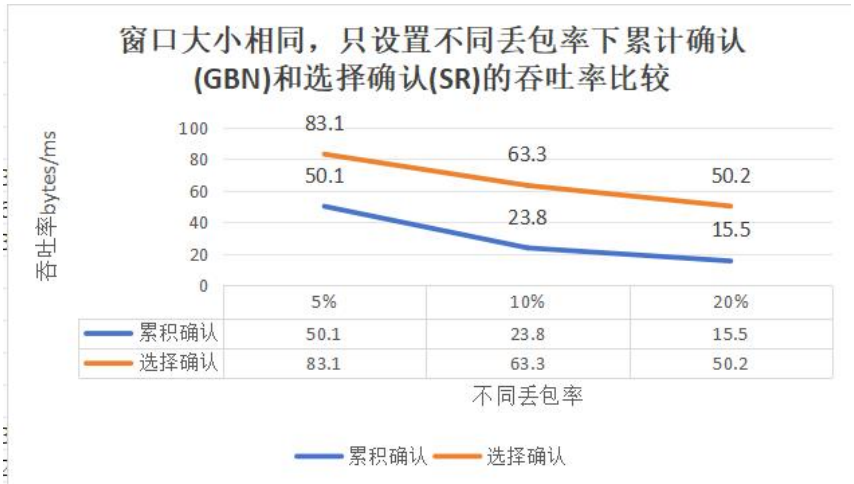
累计确认(GBN),选择确认(SR),

分别测试 1.jpg、2.jpg、3.jpg、helloworld.txt

以 1.jpg 为例，五次重复实验取平均值，

测试指标：时延、吞吐率

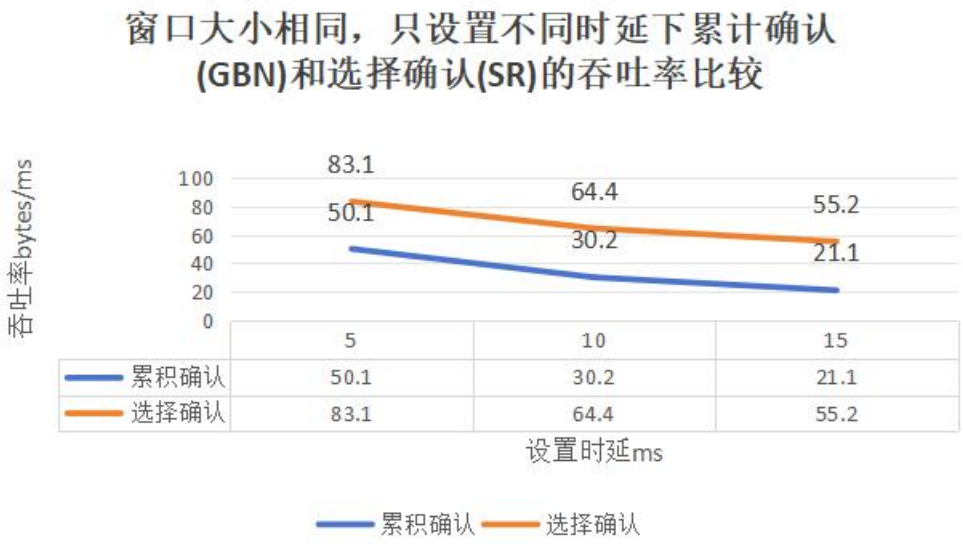
窗口大小相同时，只设置不同丢包率下累计确认(GBN)和选择确认(SR)的吞吐率时延				
	吞吐率（bytes/ms）		时延（s）	
	累积确认	选择确认	累积确认	选择确认
5%	50.1	83.1	37.2	22.3
10%	34.4	64.9	45.7	28.6
20%	22.8	48.4	69.6	38.3



当丢包率增加时，GBN 协议的吞吐率会受到较大的影响。因为 GBN 协议需要等待超时后才能重新发送丢失的数据包，而超时的时间通常比较长。在这段时间内，发送方无法发送新的数据包，导致吞吐率下降。另一方面，SR 协议可以更快地检测到丢失的数据包，并只重新发送丢失的数据包，因此在高丢包率下，SR 协议的吞吐率相对较高。

当丢包率增加时，GBN 协议的吞吐率下降更明显，而 SR 协议的吞吐率相对较高。SR 协议能够更有效地利用网络带宽，减少不必要的等待时间

窗口大小相同时，只设置不同时延累计确认(GBN)和选择确认(SR)的吞吐率时延				
时延ms	吞吐率（bytes/ms）		时延（s）	
	累积确认	选择确认	累积确认	选择确认
5	50.1	83.1	37.2	22.3
10	65.8	61.4	49.4	30.2
15	59.9	54.8	55.3	33.8



当传输时延较小时，选择确认(SR)的吞吐率较高。这是因为传输时延小的情况下，选择确认(SR)可以更快地发送和确认多个数据包，从而提高数据传输的效率。

当传输时延较大时，选择确认(SR)的吞吐率仍然较高。相比于累计确认(GBN)，选择确认(SR)可以根据接收方的确认情况动态调整发送窗口的大小，从而更好地适应传输时延的变化，提高数据传输的效率。选择确认(SR)相比于累计确认(GBN)具有更好的吞吐率性能，尤其在传输时延较大的情况下。