温兆和 1020550143 2. 计网编程作业三.

P3. 01010011 + 01100110 = 10111001

10111001 + 01110100 = 00101101

取反码. 得反码和为 11010010

UDP不直接使用该和. 而使用反码是为了检测时将3个字节与反码和相加. 如果相加的和全是1. 则未发生差错; 只要有一个0. 则发生了差错. 这种做法便于接收端检测传输中是否出现差错.

1个比特的差错不可能检测不出.

对于2比特的差错. 如果是两个字节(或一个字节与反码和)同一位上一个变成0. 一个0变成1. 则3个比特与反码和相加的和可能仍是1. 导致检测不出差错.

P11. 如果从"等待下层来自下层的1"自转换中删除. 没有影响. 因为此前下层必然经历了从"等待来自下层的0"到"等待来自下层的1"的转变. 这时已经生成了一个检验值为0的数据包. 当收到的包出差错时. 只要把原来生成的那个包再发一次. 不用再生成一个同样的数据包;

如果从"等待来自下层的0"自转换中删除这个步骤了就会有影响. 如果第一个接收到的数据包就出了差错. 此时中接收方并未在之前生成过检验值为1的包. 如果删除这步. 接收方将无包可发. 导致发送方不知道第一个包出了差错.

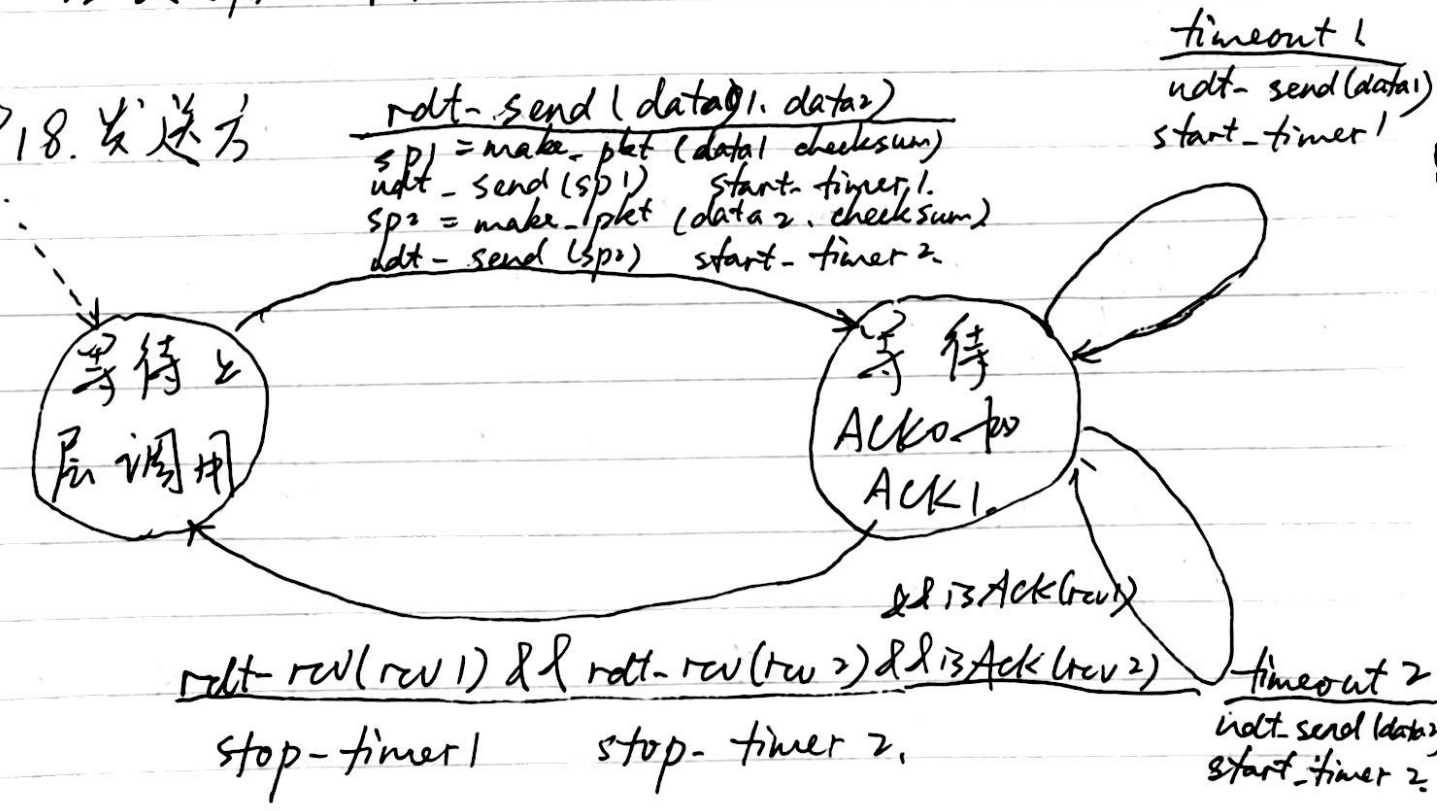
P14. 在只使用NAK的协议中，接收方仅在收到的数据包出差错时向发送方发送NAK，发送方在一定时间内未收到NAK则默认数据包无差错。

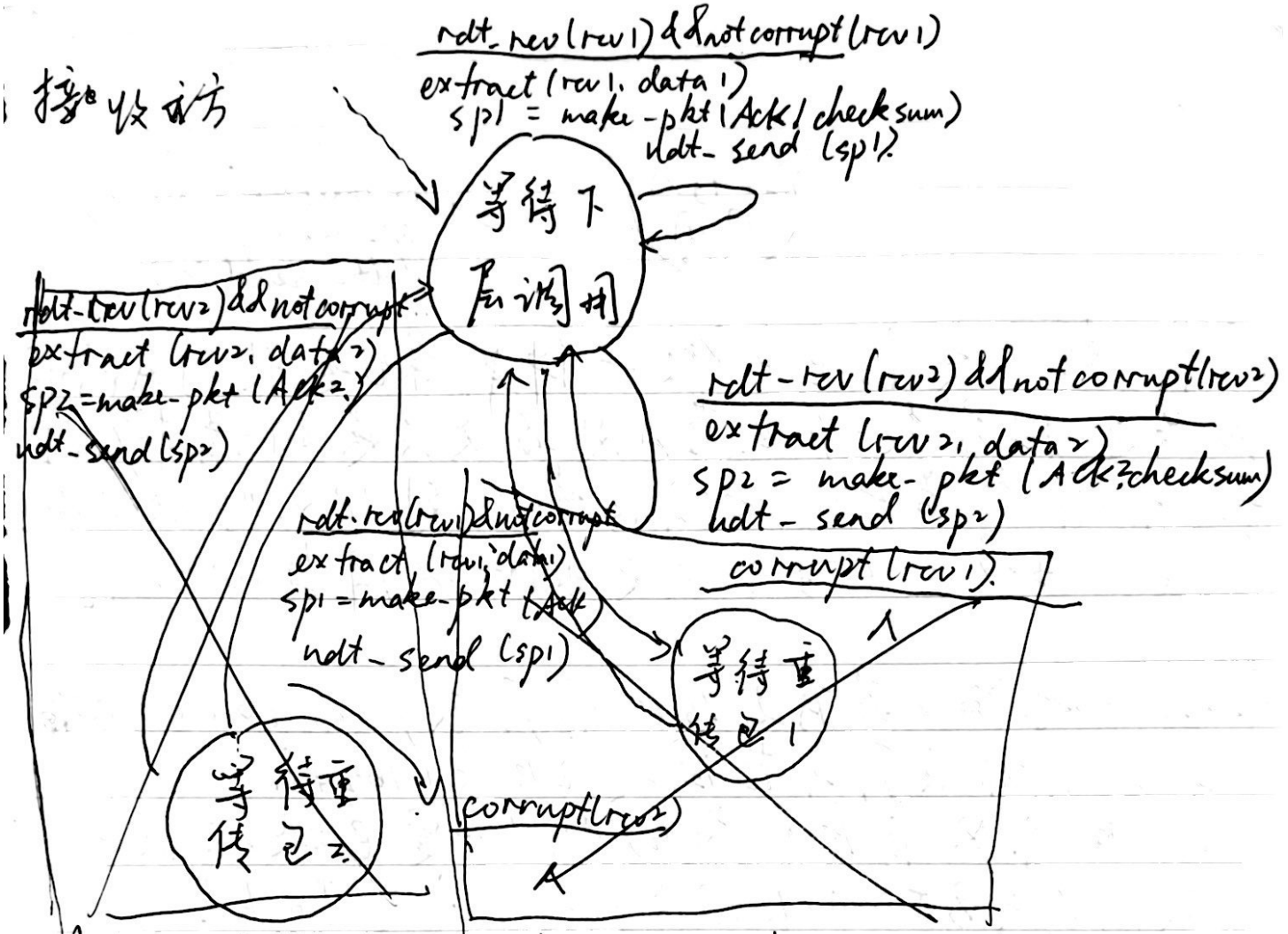
如果只是偶尔发送数据，则应使用ACK的协议更好。因为当发送的数据包无差错，就不用再发送ACK，从而缩短了链路在被占用的时间。

如果经常发送且大量数据且很少丢包，则使用ACK的协议更好。只使用NAK的协议好处不是很多，发送方只要在收到NAK后重传即可。

如果发送大量数据且很少丢包，则使用只含NAK的协议更好。因为大量的ACK会占用链路并造成拥塞，发送方只要不停发送数据包，在收到NAK后重传相应数据包即可。

P18. 发送方

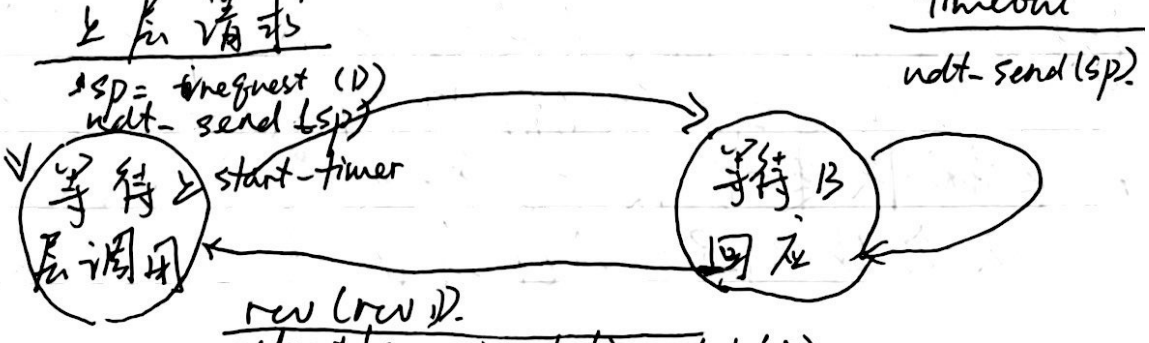




解释：发送方一次发送两个数据包，并分别为它们设置定时器，只有收到Ack1和Ack2后才会继续发送，哪个包一定时间内无Ack回复则重传，接收方再收到一个正确的包就发送一个相应的Ack，有任何一个包~~出错就等待重传那个包，重传后双方才会等待接收下一个对数据包~~，丢失就什么都不做，等发送方重传。

P21.

P21. A:



B

解释：整个过程B只做一件事，就是收到A请求后发送相应数据包。若A处每次发送请求后打开计时器，若一定时间内未收到报文，则默认A-B信道发生丢包，A重新发送和上一次同样的请求。

P22. a. 此时，窗口基序号为k，窗口长度为1024，故窗口内的报文序号范围是[k, k+1024]。

b. 此时，只有k以前的报文一定已被确认，故传回发送方的ACK报文序号范围是[0, k]。

P23. 窗口长度 $\leq \lceil \frac{n}{2} \rceil$

P26. a. 由于TCP序号字段为4字节，共有2^{32}种不同的TCP序号，故 $\frac{L}{536} \leq 2^{32}$，L最大为$536 \times 2^{32}$字节。

b. $t = \frac{(536+66)\times 2^{32}}{155\times 2^{20}} = 1598.33548$ s

P36. 也许有些时候发送方接收到冗余ACK并不是因为丢包，而是因为传输时间较长。如果接到3个冗余ACK就快速重传，可能还是会因为超时周期过短而一次次收到冗余ACK并快速重传，不仅没有解决

问题. 反而还加大了链路负担, 所以. 收到三个冗余 ACK 才默认发生丢包并快速重传是一个比较折中的做法.

P37. a. GBN: 收: ACK1. ACK1. ACK1. ACK1. (包2重传成功) ACK2. ACK3. ACK4. ACK5. 共9个
发: 包1. 包2. 包3 包4 包5 包2 包3 包4. 包5

SR: 收: ACK1. ACK3 ACK4 ACK5 ACK2 共5个
发: 包1 包2 包3. 包4 包5
窗口长度最大为2

TCP: 收: ACK1 ACK3 ACK4 ACK5 ACK2 共5个
发: 包1 包2 包3 包4 包5 包2.

b. TCP协议最快
GBN协议中超时前接收方不断发送 ACK1.
超时后需将包2、3、4、5全部重传.
SR协议中包2超时前不能发送包4包5. 只能等包2超时并重传成功后再发送
TCP协议中包2超时前仍可同时发送后续数据包. 只需在包2超时后重传包2即可.