3-2 基于滑动窗口机制的可靠数据传输(基于3-1)

> 1811464 郑佶

通信参数设置

见实验3-1报告

功能实现

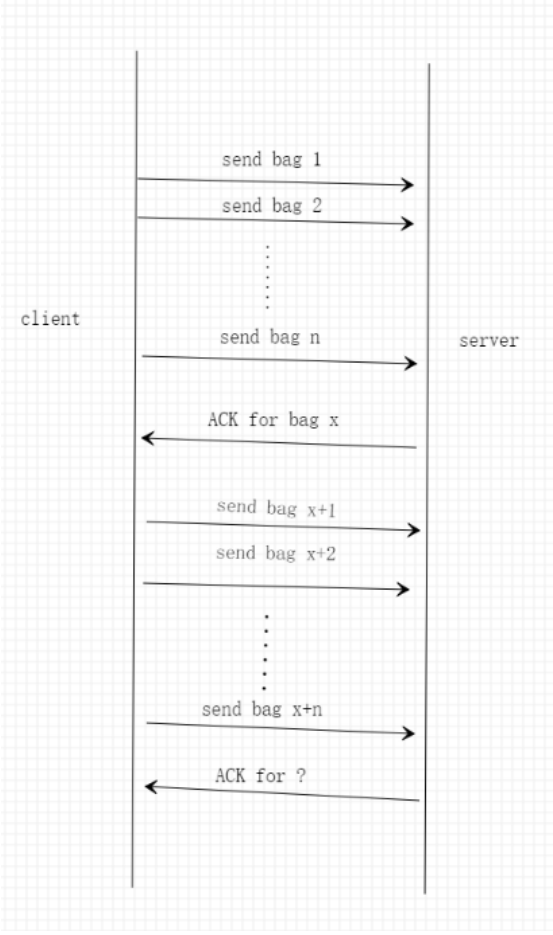
可靠连接的建立和断开/差错检验/确认重传

见实验3-1报告

滑动窗口机制

原理

滑动窗口机制,顾名思义,就是将等同于滑动窗口的大小的多个数据包一同发出,并在接收到`ACK`数据包后再决定滑动窗口的移动(在全体待发数据包队列中的移动).



如上图,是一个窗口大小为`n`的滑动窗口机制的简要描述.

在滑动窗口机制的实现中,总是需要根据接收到的`ACK`数据包中的指明的数据包序号来决定窗口的移动(即`累积确认`),这便需要对数据包的编码和数据包中的数据包序号变化机制做出要求.并且,为实现滑动窗口机制,也需要一定的数据结构来加以管理.

数据结构与重要变量规定

首先,滑动窗口采用先进先出的队列,即`C++`标准库中的`queue`类来实现.

该类使用`pop()`弹出队首元素,`push()`在队尾加入元素.函数`front()`返回队首元素,函数`back()`返回队尾元素.

队列中存储的元素,是一个包含包序号信息的`bag_elem`类.以`bag_elem`为队列元素的`queue`类实例`list`,用于存储目前处于发送窗口的数据包信息.

而数据包的编码方式,按照先后顺序是`校验和`->`包结束信号LAST`->`包序号`->`包长度`->`数据`.抑或是`校验和`->`包结束信号NOTLAST`->`包序号`->`数据`.(若不是最后一个包,则包长度固定为`MAXLEN`,无需存储).包序号以`char`型变量存储,以`模256`的方式递增.

变量`base`用于存储当前滑动窗口最左端的包序号.

变量`send_ok`用于存储当前滑动窗口最左端元素的前一个元素包序号,即被认为传输完成的最后一个包.

变量`send`用于存储当前滑动窗口最右端元素的包序号.

变量`next`用于存储当前滑动窗口最右端元素的下一个元素的包序号.

实现过程

- 检查窗口大小,右端不到末尾且窗口大小未满,则滑动窗口向右扩展.
  - 窗口拓展即指将`next`指代元素推入队列
- 组装数据包(即在数据内容前加上包序号/包结束信号/校验和).
- 发送滑动窗口队列`list`中数据包
- 循环接收`ACK`数据包,接收顺序从窗口首部开始按序号依序进行.被成功接受的`ACK`数据包对应的`bag_elem`型变量,依次从队列`list`中被弹出.
- 循环

具体代码

```
send = 0;
next = base;
send_ok = 0;
num = len / MAXLEN + (len % MAXLEN != 0);
int time_begin = clock();
while(1)
{
    if(send_ok == num)
        break;
    if(list.size() < WINDOW_MAXSIZE && send != num)
    {
        bag_send(storage+send*MAXLEN,send==num-1?
                    len-(num-1)*MAXLEN:
                    MAXLEN,
                 next % ((int) UCHAR_MAX + 1),send==num-1);
        list.push(bag_elem(next % ((int) UCHAR_MAX + 1)));
        bag_is_ok[next % ((int) UCHAR_MAX + 1)] = 1;
        next++;
        send++;
    }
    char recv[3];
    int len_tmp = sizeof(server_addr);
```

```
    if (recvfrom(client, recv, 3, 0, (sockaddr *) &server_addr, &len_tmp) !=
SOCKET_ERROR && check_sum(recv, 3) == 0 && recv[1] == ACK && bag_is_ok[(unsigned
char)recv[2]])
    {
        while (list.front().order != (unsigned char) recv[2])
        {
            send_ok++;
            base++;
            bag_is_ok[list.front().order] = 0;
            list.pop();
        }
        bag_is_ok[list.front().order] = 0;
        send_ok++;
        base++;
        list.pop();
    }
}
```

以上是在基于滑动窗口来传输文件数据包中的具体例子(不包括数据包组装过程).