

**计算机网络**

**课程实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 实验名称 | 可靠数据传输协议-GBN 协议的设计与实现 | | | | | |
| 姓名 | 董琦 | | 院系 | 数据科学与大数据 | | |
| 班级 | 2003501 | | 学号 | 120L020701 | | |
| 任课教师 | 刘亚维 | | 指导教师 | 刘亚维 | | |
| 实验地点 | G001 | | 实验时间 | 2022/10/13 | | |
| 实验课表现 | 出勤、表现得分(10) |  | 实验报告  得分(40) |  | 实验总分 |  |
| 操作结果得分(50) |  |
| 教师评语 | | | | | | |
|  | | | | | | |

****

|  |
| --- |
| 实验目的： |
| 理解滑动窗口协议的基本原理；掌握 GBN 的工作原理；掌握基于 UDP 设计并实现一个 GBN 协议的过程与技术。 |
| 实验内容： |
| 1) 基于 UDP 设计一个简单的 GBN 协议，实现单向可靠数据传输（服务器到客户的数据传输）。  2) 模拟引入数据包的丢失，验证所设计协议的有效性。  3) 改进所设计的 GBN 协议，支持双向数据传输；  4）将所设计的 GBN 协议改进为 SR 协议。 |
| 实验过程： |
| 1. 报文结构   一个MSS共长1500B，第0B是本端发送报文的序号，第1B是确认对方发送报文的序号，2、3B合起来用以标志报文内容的长度，剩下的1496B为报文内容部分。当序号为-1时，代表发送文件结束。  由于考虑到后续扩展为双向通信的协议，故客户端和服务端的发送和接收报文格式一致，均为上述格式。  例：若本次发送的包在文件中序号为3，而且接收到对方序号为2的包，且此次包内容大小800B，则报文第0B内容为3，第1B内容为2，第2、3B合并内容为800，余下内容长度为800B，总长度为804B。   1. 报文超时与丢失的实现：在SW协议中已叙述，这里不再赘述。 2. GBN协议FSM的流程图      1. GBN协议的交互示例：   查看源图像   1. GBN协议的实现（写代码时构思不好，没做好模块化）    1. 服务器端：   各属性如下图：    当启动一个传输文件任务时，初始化变量、开辟缓存：    报文构建模块，当要发送新的报文时，执行此部分函数。    当文件发送完毕时，发送一个终止报文，序号设置为-1：    发送报文的函数，设置定时器等一系列属性：    当接收到ACK报文时，检测其返回的序号是否等于当前的Base，若是，则说明成功接收，base+1，发送新的报文；若否，则判定为冗余分组，不做处理。    当接收到冗余分组时，若是已发送未确认的报文，则重启计时器。当接收到正确分组时，则关闭计时器。  实现文件双向传输，要对ACK报文的内容进行解析，步骤类似于停等协议（这里我从简写了）：    当有报文超时时，则执行全部重发。  检测是否有报文超时：    重发所有未确认的缓存报文：    定时器类的实现：    当双方文件传输完毕后，结束任务：     * 1. 客户端   客户端在收到一个服务器报文时，需检测其头部所带序号。若为期望的序号，则返回ACK并且将报文内容保存，同时更新序号；若不是期望的序号，则依然返回ACK而不保存报文，接收到报文什么序号，则返回的确认报文就是什么信号，交由服务器来判断执行对应动作。  向服务器传输文件的代码则类似SW协议，较为简单，不做赘述。  一次接收任务的参数初始化：    对接收分组的内容解析：    构建发送的报文，若接收报文显示已经接收到最新的分组，则在发送报文内容中写入新的内容，若否则保持不变进行重发：    当双方文件传输结束后，则完成任务：    由于写代码时候双方的循环动作顺序不一样。。。所以客户端和服务端的代码结构不太一样，对称性不太好。   1. SR协议流程：        1. SR协议的实现   SR是在GBN协议上进行了更改所得的，故这里只展示具体改动地方。   * 1. 服务器端：   重发操作：只重发超时的报文，而不是所有缓存报文    确认收到操作：对于在窗口内的序号报文，收到ACK时则标记已经确认接收。  若等于Base，则进行窗口滑动，若不等于，则窗口不变。    此外还有一些辅助这俩个功能的代码，这里就不做赘述了，都是比较简单的逻辑。   * 1. 客户端   客户端新增了报文缓存，缓存大小等于客户端的窗口大小：    当接受到窗口范围内的报文时，确认接收并且缓存。当报文序号等于base时，则按序号将已经缓存的连续报文保存到磁盘，并且更新序号；而在其他情况下，则单纯返回一个ACK，表示时一个冗余分组：    这样就完成了所有SR协议的改动。更多细节请见代码。 |
| 实验结果： |
| 1. GBN协议：   一次文件传输任务中服务器和客户端的输出。由于文件较大，引入丢包、超时等机制，所以流程较为复杂，故不详细解释，只展示大概的输出。具体请助教可以自己运行然后分析一下过程。          最后可以看到文件双向传输成功实现：    从上面例子中可以看到分组丢失、网络延迟、确认报文丢失等情况，而服务器也相应地做出了重发的操作。  等等。   1. SR协议：         一次传输文件的部分控制台输出，可以看到服务端当前的窗口base位置、尚未确认接收的报文序号集合、窗口滑动变化、期望客户端传输文件报文序号等。也可以看到客户端成功接收报文后的处理，包括写入、base滑动等。最后可以验证，成功完成了文件的双向传输。 |
| 问题讨论： |
| Q: 相同条件下，GBN协议和SR协议哪个更快一些？  A：SR协议更快一些，SR协议的接收方可以将乱序到达的数据序列缓存起来，直到顺序正确即可向上层交付，而GBN协议要重传base及以后的一系列分组，浪费了大量的时间。 |
| 心得体会： |
| (1)掌握了滑动窗口协议的基本原理；并通过具体的实践感受到了滑动窗口协议在 GBN 与 SR 中的体现；  (2)掌握了基于 UDP 设计并实现一个 GBN 协议的过程与技术，同时也体会 到了如何实现模拟上层数据到来以及模拟丢包事件的产生；  (3)掌握了基于 UDP 设计并实现一个 SR 协议的过程与技术。 |