**山东大学计算机科学与技术学院  
《数据结构与算法》课程设计报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 学号： 201700301147 | 姓名：杜瀛川 | | 班级： 17.4 |
| 上机学时：4 | | 日期： 2019.3.22 | |
| 课程设计题目：外排序 | | | |
| 软件开发环境：Qt Creater | | | |
| 项目最终效果图      报告内容：  1.需求描述  1.1 问题描述  应用竞赛树结构模拟实现外排序。  1.2 基本要求  1.设计实现最小竞赛树结构。  2.设计实现外排序，外部排序中的生成最初归并串以及K路归并都应用竞赛树结构实现；  3.随机创建一个较长的文件；设置归并路数以及缓冲区的大小；获得外排序的访问磁盘的次数并进行  分析。可采用小文件来模拟磁盘块。  4.用某种内部排序法生成最初归并串，然后进行K路归并，给出实验结果，比较访问磁盘次数  1.3 输入说明  外排序程序与用户交互的过程采用了简单的GUI界面，用户可选择下拉菜单栏上功能，根据对话框上的文字提示实现数据文件的形成、文件的打开、数据排序等操作。如图1所示。      图1  实现的功能如下：  1.3.1 打开文件  用户点击选择要打开的文件按钮，会出现如图2所示的对话框，用户通过点击选择要打开的文件。    图2  1.3.2 数据  用户点击放到菜单栏中的数据上，触发下拉菜单，点击新建数据文件，创建排序所需的随机数文件，用户可以自定义所需的规模大小。如图3所示。        图3  1.3.3 排序  用户可对选择的数据文件进行排序，排序方式有两种，分别是：   1. 竞赛树生成初始归并段，竞赛树进行归并段归并。 2. 快速排序生成归并段，竞赛树进行归并段归并。   用户可以设置限制内存的大小，实现外排序。  如图4所示。        图4  1.3.4 清空  用户点击清空按钮，文件显示将会关闭，进度条将会清0，状态栏的测试信息将会清空。如图5所示。    图5  1.4 输出说明  1.4.1 打开文件，文件格式为QT标准数据格式(.stm)，文件中的数据会显示在editText中，如图6所示。    图6  1.4.2 数据生成，程序会将产生用户自定义规模的qint64类型的数据存放在用户指定的位置。文件类型为QT标准数据格式(.stm)。  1.4.3 排序输出，在子线程中进行数据排序，并根据子线程进度不断在主线程中更新UI，通过ProgressBar显示进度，以免用户因不耐烦而关闭程序。排序过程分为两步，分别是初始归并段的合成和归并段的归并，通过Label进行阶段区分。  当归并段归并完成后，显示信息框，提示排序已完成，并在状态栏显示排序总计的内存访问次数。点击OK后，将显示经过排序的结果文件。如图7所示。        图7  1.4.4 清空操作，用户根据自身需要将主窗口中的显示信息全部清除。以便进行下一步操作。如图8所示。    图8  2.设计  2.1 系统结构设计    2.2 设计思路  内部排序要求待排序的元素全部放入计算机内存。但是，当待排序的元素所需要的空间超出内存容量时，内部排序法就需要频繁地访问外部存储介质（如磁盘），那里存储着部分或全部待排的元素。这使得排序效率大打折扣。于是我们需要引入外部排序法。外部排序一般包括两个步骤：   1. 需要根据内存大小，将外存中的数据文件划分成若干段，每次把其中一段读入内存并用内排序方法进行排序，这些已排序的段或有序的子文件成为顺串或归并段。生成一些初始归并段，每个归并段都是有序集： 2. 将这些归并段合并成一个归并段。   外排序所需要的时间由3部分组成：  1）内部排序所需要的时间。  2）外存信息读写所需要的时间  3）内部归并所需要的时间。  减小外存储信息的读写次数是提高外部排序效率的关键。对同一个文件而言，进行外排序所需的读写外存的次数与归并趟数有关系。假设有m个初始顺串，每次对k个顺串进行归并，归并趟数为m对K的对数。为了减少归并趟数，可以从两个方面着手：   1. 减少初始归并段m的数量。 2. 增加归并顺串的顺串数量k。 3. 减少k路合并的时间。   2.3 数据及数据类(型)定义  赢者树：有n个选手的一颗赢者树是一颗完全二叉树，它由n个外部节点和n-1个内部节点，每个内部节点记录的是在该节点比赛的赢者。  简单的说就是有一颗完全二叉树，从叶节点开始，左右两个子节点谁大（小），就把谁的值赋值给他们的父节点，然后父节点在和其兄弟节点进行比较将较大（小）者赋值给其父节点直到根节点，那么根节点的数据就是冠军。 注意：现实的竞赛所对应的树不一定都是完全二叉树，但是用完全二叉树能使比赛的场次最少  首先定义赢者树虚基类，其中定义了构造函数、析构函数、初始化函数、重赛函数。      定义completeWinnerTree类，继承自winnerTree父类，有如下方法。    2.4.算法设计及分析  2.4.1初始化  赢者树实现原理：  假如用完全二叉树的数组表示来表示赢者树。一颗赢者树有n名选手，需要n-1个内部节点tree[1:n-1]。选手（或外部节点）用数组palyer[1:n]表示，因此tree[i]是数组player的一个索引，类型为int。在赢者树的节点i对应比赛中，tree[i]代表赢者，为实现这种对应关系，我们必须能够确定外部节点player[i]的父节点tree[p]。当外部节点的个数为n时，内部节点的个数为n-1。最底层最左端的内部节点，其编号为s，且s=2^[log2(n-1)] (这里中括号代表向下取整)。因此，最底层内部节点的个数是n-s，最底层外部节点个数lowExt是这个数的2倍。倒数第二层最左端的外部节点号为lowExt+1。令offset=2\*s-1。对于任何一个外部节点player[i]，其父节点tree[p]由以下公式给出：    赢者树的初始化：我们计算比赛时是从左往右，所以当当前计算的节点为某个父节点的右子节点时，我们就能组织一场比赛，因为左节点的胜者一定产生了，否则不会轮到右节点，我们是从左往右开始计算的。      2.4.2 节点比赛  递归实现赢者树自下而上比赛，确地节点值。    2.4.3 重赛  当选手 thePlayer 的值改变，在从外部节点 player[ thePlayer ] 到根节点 tree[1] 的路径上，一部分或者全部比赛都需要进行重赛。为简单起见，我们要路径上的全部比赛重赛。具体的实现方案如下：      2.4.4文件io操作  使用QT 中的qFile qDataStream类，实现文件io，为外排序提供底层支持。具体方法实现如下。        2.4.5 排序  本程序提供两种排序，均为外排序。两种排序生成初始归并段的方法不同。  一、第一种使用胜者树结构生成初始归并段，归并段的平均大小约为内存限制的两倍，可以减少初始归并段的数量，进而提高排序速度。部分实现如下。      生成初始归并段后，进行各归并段的归并操作。同样使用竞赛树结构进行归并。       1. 首先将文件按内存限制分为多个小文件。在各小文件内部使用内部排序方法（快速排序）生成初始归并段，部分实现如下。     2.4.5 ui线程进度显示  由于外排序规模可能较大，为防止用户因等待时间过长、长时间没有收到反馈而关闭程序，故在窗口中添加了progressBar组件进行进度显示。利用qt信号与槽机制，在子线程中进行排序，同时将进度以信号形式发送到主线程，在主线程中更新ui。实现进度展示。      2.4.7 随机测试文件生成  在子线程中通过随机数生成函数生成用户规定规模的随机数，并通过io操作写入到用户指定的文件中。随机数生成器实现如下。    3. 测试结果  首先进行随机数文件的生成操作，随机生成的1000个64位随机数如下：    使用两种排序方法对该文件进行排序，内存限制均为1000，所得结果如下：    同时统计访存次数。其中第一种方法访存次数为 24次    第二种方法的访存次数为 40次    进行第二次试验，使用更大的数据规模，为100000个64位随机数，内存规模限制为5000，实验结果如下。    第一种排序：访存次数为44次    第二种排序访问次数为 80次    可见在相同的数据规模下和内存限制下，经由竞赛树生成初始归并段的方法可以明显减少访存次数。  4. 分析与探讨  4.1算法复杂度分析  利用赢者树进行k路归并，复杂度约为，首先，用O(k)的时间初始化含k个选手的赢者树。这k个选手都是k个被合并的顺串的一个头元素，然后将赢者树移入输出顺串中相应的输入顺串中的下一个元素代替之。如果在该输入顺串下无一元素，则需要用一个key值很大的元素代替之。k次移入和代替赢家共需耗时，因此采用赢者树进行k路合并的总时间为。  4.1.2访存次数分析  设内存空间所容纳的外部节点为p个，实验一个含有p个选手的赢者树。每个选手对应输入集合中的一个元素。每个选手有一个值，和一个顺串号。赢者规则：具有较小顺串号的元素获胜，具有较小元素值的元素获胜。  从输入集合中输入前p个元素，初始这个元素的顺串号均为1。建立这p个选手的最小赢者树。  重复如下操作：将最终赢w移入它的顺串号所对应的顺串中；若输入集合中有下一个输入元素，则n等于下一个输入元素，否则，n等于无穷大。如果n的值大于w的值，则n的顺串号等于w的顺串号，否则为w的顺串号为n的顺串号加1。直到所有元素都输出到顺串中。  使用这种方法生成初始归并段时，顺串的平均长度约为2p,特别是，当输入集合几乎有序时，效率会特别高。  而第二种方法中归并段的平均长度为p,所以会生成更多更多的归并段，增大k路归并的k值，增大访存次数，和实验结果相符。   1. 附录：   因项目文件和源码过多，故将源码和可运行程序打包附在报告之后。项目图如图所示。 | | | |