**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：2019.3.15 | |
| 课程设计题目：跳表实现与分析 | | | |
| 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  增加了向前指针的链表叫作跳表。跳表全称叫做跳跃表，简称跳表。跳表是一个随机化的数据结构，实质就是一种可以进行二分查找的有序链表。跳表在原有的有序链表上面增加了多级索引，通过索引来实现快速查找。跳表不仅能提高搜索性能，同时也可以提高插入和删除操作的性能。  跳表是一个随机化的数据结构，可以被看做二叉树的一个变种。它采用随机技术决定链表中哪些节点应增加向前指针以及在该节点中应增加多少个指针。跳表结构的头节点需有足够的指针域，以满足可能构造最大级数的需要，而尾节点不需要指针域。  本问题旨在实现模拟跳表，并且模拟跳表增删查操作，实现跳表的功能。并实验过程和实验结果分析跳表的结构。  1.2 基本要求  1、构造并实现跳表ADT，跳表ADT 中应包括初始化、查找、插入、删除指定关键字的元素、删除关键字最小的元素、删除关键字最大的元素等基本操作。  2、分析各基本操作的时间复杂性。  3、针对一个实例实现跳表的动态演示(鼓励使用图形演示)。  4、能对跳表维护动态数据集合的效率进行实验验证，获得一定量的实验数据，如给定随机产生1000 个数据并将其初始化为严格跳表，在此基础上进行一系列插入、删除、查找操作（操作序列也可以随机生成），获得各种操作的平均时间（或统计其基本操作个数）；获得各操作执行时间的变化情况，应该是越来越大，当大到一定程度后应该进行适当的整理，需设计相应的整理算法，并从数量上确定何时较为合适；能和其他简单线性数据结构，如排序数组上的折半查找进行各类操作效率上的数量对比。  1.3 输入说明  跳表与用户交互过程采用了简单的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自己所需跳表的初始化和各种模拟操作。如图1所示。      图1  实现的功能如下：  1.3.1初始化   1. 根据用户输入初始化跳表。点击初始化按钮。会出现如图2所示对话框，用户可按图中的输入格式，依次输入Key和Value。合理设置跳表最大关键字的大小，进行初始化。     图2   1. 随机初始化跳表。点击随机初始化按钮。会出现如图3所示对话框，用户可根据需求设置最大关键字和所需跳表的元素数。程序随机初始化跳表。     图3  1.3.2 删除  1、删除特定元素，点击删除指定元素，会出现如图4所示对话框，用户输入要删除元素的关键字即可删除。    图4  2、删除最大、最小元素。  3、删除整个跳表。  1.3.3 插入  1、插入单个元素，点击插入单个元素，会出现如图5所示对话框，用户输入要插入元素的Key和Value即可插入该元素。    图5   1. 插入多个元素，点击插入多个元素，会出现如图6所示对话框，用户按初始化格式输入，即可插入多个元素。     图6  1.3.4 查找  查找指定元素，点击查找指定元素，会出现如图7所示对话框，用户输入要查找元素的关键字，即可查找指定元素。    图7  1.3.5 测试  1、开始时间测试，点击开始时间测试，会出现如图8所示对话框，用户设置随机测试数据的规模，即可测试跳表各种操作的用时。    图8   1. 结束时间测试，子线程停止，输出经过测试后的跳表。   1.3.6 调整  重新调整跳表，将跳表中的数据重新进行初始化。整理跳表。  1.3.7 清空  清空跳表显示窗口或测试结果显示窗口。  1.4 输出说明  1.4.1 查询结果通过对话框形式显示。如图9所示。    图9  1.4.2 跳表结构通过跳表输出窗口显示，测试结果通过测试结果输出窗口显示，如图10所示。    图10  2.设计  2.1 系统结构设计      2.2 设计思路  2.2.1 数据结构选择  本问题是跳跃链表的实现及分析，所以首先应找一种合适的数据结构模拟跳跃链表。自然而然，经过改造的链表是一种合适的数据结构。  对n个数对而言，0级链表包括所有数对，1级链表每2个数对取一个，2级链表每4个数对取一个，i级链表每2的i次方个数取一个。一个数对属于i级链表，当且仅当它属于0到i级链表，但不属于i+1级链表。这样，可以实现二分查找，极大提高查找的效率，从而极大提高链表插入、删除的效率。  2.2.2 插入与删除  在插入和删除时，要保持跳表的规则结构，需要耗时O(n)。在规则的跳表结构中，i级链表有n/2i个记录，在插入时要尽量逼近这种结构。插入的新数对属于i级链表的概率为1、2。在插入时，要为新数分配一个级，分配过程通过随机数生成器来完成。若新数对属于i级链表，则插入结构仅影响0到i级链表指针。  对于删除操作，我们无法控制结构，只在原有跳表的结构上删除指定节点，对每一级链表进行维护即可。  2.2.3 级的分配  在规则的跳表结构中，i-1级链表的树对个数与i级链表的数对个数之比是一个分数p，因此，属于i-1级链表的数对同时属于i级链表的概率为p,假设用一个统一的随机数生成器产生0和1之间的实数，产生的随机数<=p的概率为p。若下一个随机数<=p，则新数对应的应在1级链表上，依次确认，直到一随机数>p为止。  这种方法有潜在的缺点，某些数对被分配的级数可能特别大，远远超过所需。没避免这种情况，可以根据数对数取p的对数，确认级数的上限。即使采用了级数上限，还应控制链表级数依次增加，降低链表复杂性，减少不必要的空间占用。  2.3 数据及数据类(型)定义  2.3.1 结构skipNode  首先实现跳表节点skipNode,结构体重有pair类型的成员，用来存储Key和Value。跳表的头结点需有足够的指针域，以满足最大链表级数的构建需要，而尾节点不需要指针域。每个存有数对的节点都有一个个数大于自身级数的指针域。    2.3.2 类skipList  跳表类的数据成员有    跳表类成员函数有    2.4.算法设计及分析  2.4.1 find()函数  find函数从最高级链表开始查找，知道0级链表，在每一级链表中，从左边尽可能逼近要找的记录。虽然在找到关键字等于theKey的数对时，可能在i级就终止搜索，但是用来检验是否相等的额外操作是不必要的，因为大部分这样的数对都只出现在0级链表中。当for循环退出时，指针正好处在要查找的数对的左边。与0级链表的下一个比较，即可确定要找的数对是否在链表中。    2.4.2 分级函数  在规则的跳表结构中，i-1级链表的树对个数与i级链表的数对个数之比是一个分数p，因此，属于i-1级链表的数对同时属于i级链表的概率为p,假设用一个统一的随机数生成器产生0和1之间的实数，产生的随机数<=p的概率为p。若下一个随机数<=p，则新数对应的应在1级链表上，依次确认，直到一随机数>p为止。  这种方法有潜在的缺点，某些数对被分配的级数可能特别大，远远超过所需。没避免这种情况，可以根据数对数取p的对数，确认级数的上限。即使采用了级数上限，还应控制链表级数依次增加，降低链表复杂性，减少不必要的空间占用。    2.4.3 search()函数    2.4.4 插入函数  插入函数和删除函数都是在查找的基础上，对链表进行操作，具体操作请参考search函数。      2.4.5 删除函数    3. 测试结果  测试跳表为通过随机关键字和字符串生成程序生成的，也可以是用户自行输入的数据。本过程为了方便，觉得使用关键字最大值为32000，规模为20000的随机生成的跳表进行测试。为了不阻塞主线程，随机跳表生成过程在子线程myThread中进行。只在主线程中进行UI更新和跳表展示等工作。  测试数据示例    测试过程示例    测试过程在子线程testThread中进行。一次测试依次进行5000次插入操作，5000次查找操作，5000次删除操作，依次记录各种类操作时间和总时间。将时间结果发送回主线程进行UI更新、跳表更新、测试结果展示。  经过约5000次测试实验，能反应出群体特点的部分样本实验结果如下：               |  |  |  |  |  | | --- | --- | --- | --- | --- | | **测试次数** | **插入用时** | **查找用时** | **删除用时** | **总用时** | | 16 | 15.82 | 4.98 | 5.002 | 25.802 | | 147 | 17.946 | 5.985 | 5.979 | 29.91 | | 523 | 18.105 | 5.891 | 6.753 | 30.759 | | 1095 | 22.282 | 4.991 | 6.981 | 34.254 | | 2035 | 26.544 | 4.975 | 7.156 | 38.675 | | 5395 | 24.931 | 6.982 | 8.977 | 40.89 |   显然可见，随实验次数的增加，实验用时呈递增趋势。跳表通过增加链表层数实现二分查找，从而提高了查找、插入、删除等操作的效率。但多次操作会破坏严格的跳表结构，降低二分查找的效率。  为了提高经过多次操作后跳表各项操作的效率，实验设计出整理算法，也是在子线程中进行。即将原始跳表中的数据取出，再次初始化到新的跳表中。实现严格跳表结构的重新实现。从而提高跳表效率。    对调整后的跳表重新进行测试。能反应出群体特点的部分样本实验结果如下：               |  |  |  |  |  | | --- | --- | --- | --- | --- | | **测试次数** | **总用时** | **插入用时** | **查找用时** | **删除用时** | | 9 | 25.97 | 15.995 | 4.951 | 5.024 | | 102 | 27.929 | 16.958 | 4.988 | 5.983 | | 500 | 28.93 | 16.95 | 4.967 | 7.013 | | 1099 | 29.886 | 18.964 | 4.939 | 5.983 | | 2044 | 32.909 | 19.942 | 5.985 | 6.982 | | 5353 | 46.843 | 34.875 | 5.018 | 6.95 |  1. 分析与探讨   由上述实验数据粗略绘制出操作时间变化的折线图。第一阶段，对跳表进行，插入、删除、查找操作，随操作次数增加，严格跳表结构被破坏，插入、删除、查找操作所需时间呈递增趋势。第二阶段，使用调整函数对跳表进行调整，经过调整后，操作时间出现了明显降低，回到了初始水平。第三阶段，对跳表进行插入、删除、查找操作，随着严格跳表的结构再次被破坏，所需时间再次呈现递增趋势。  但继续加大实验规模，跳表操作的时间会维持在一个较高水平后不再增加，经过对此时跳表结构的分析可知，此时跳表结构以及非常随机，随机的查找、插入、删除等操作以及很难再使跳表的结构更加混乱。和实验现象相符合。  时间复杂度：在字典中有n个数对使，查找、插入、删除等操作的时间复杂度均为O(n+maxLevel)。在最坏的情况下，可能只有一个maxLevel级数的数对，余下所有数对均在0级链表上。I>0时，在i级链表上花费的时间为O(maxlevel),在0级链表上花费的时间为O(n)。尽管最坏情况下的性能较差，但跳表仍不失为一种有价值的数据描述方法，因为查找、插入、删除的时间复杂度均为O(logn)，这是由二分查找的性质决定的。  空间复杂度：在最坏的情况下，每一个记录都可能有maxLevel级，都需要maxLevel+1个指针。因此，除了需要存储n个数对的空间，还需要存储O(n\*maxLevel)个指针的空间。不过，在一般情况下，1级链表有n\*p个数对，2级链表有n\*p2个数对，i级链表有n\*pi个记录。虽然最坏情况下的空间需求比较大，但平均的空间需求不大。当p = 0.5时，平均空间需求大约是2n个指针空间。   1. 附录：实现源代码   项目文件如图所示,因程序和源代码量较大，故打包成一个单独文件夹。 | | | |