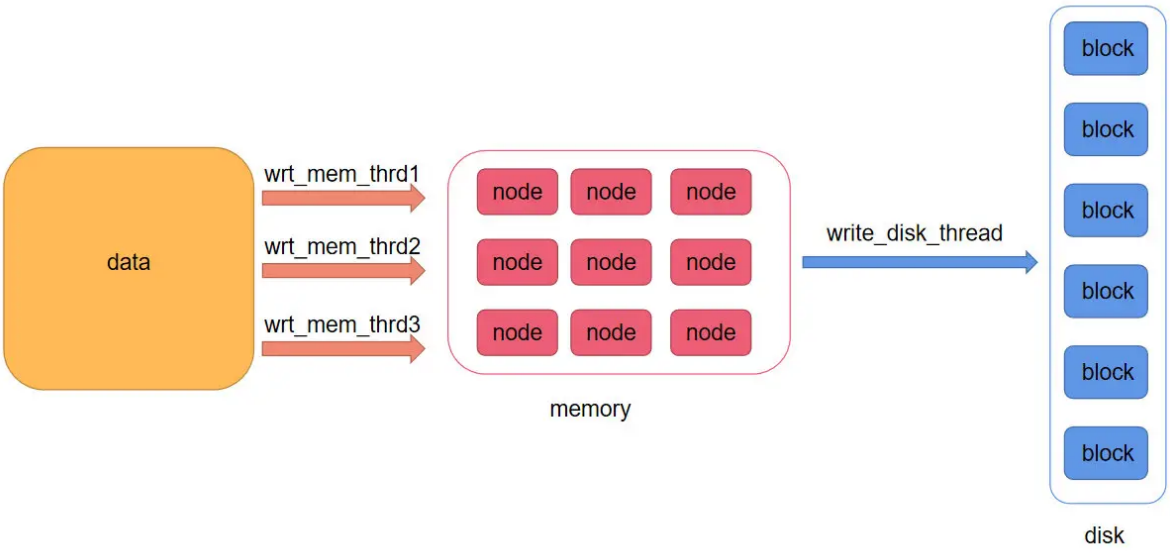
实验心得汇总

19309049 黄皓佳

此次大作业我们完成了B+树的并行批量加载，本人的工作主要是参与了实现思路的讨论，并负责并行思路的文档编写、结果评估代码的编写以及实验结果的可视化。此次小组的分工合作，我见识到队友的优秀——无论是思路的设计、代码的实现亦或文档的编写，都看到了许多值得学习之处。在小组讨论中，我也感受到了交流所碰撞出的火花，认识到积极的沟通交流能够使得整个项目快速有效地推进——第一次开会讨论，我们围绕如何实现并行讨论了两个多小时，从初步思路的提出，到慢慢挖出考虑欠缺之处——比如文件不能实现并行写、死锁问题等等，到思路的逐步完善，再到最后得到许多可尝试的实现方案，我们在曲折中逐步走上一条可执行的道路。在后续的开会中，我们又根据当下阶段进行工作的分配，合理的分工加快了我们的步伐。

在一开始实际上我们有多种实现方案，比如我提出过一个使用生产者消费者模型的思路——有多个线程进行节点的内存加载，一个线程负责将内存中的节点数据写到硬盘中，在共享内存中实际存放的是各个已经加载的节点的指针，大致的实现过程如下：



一开始也尝试过实现这种方式，但没有编码成功，出现了一些BUG没能调通，且经过讨论比较，发现这种实现方案效率并没能优于我们最终选择的实现方案——在一开始是想着使用生产者消费者模型相较于最终的方案可以不用互斥锁，但在真正编程的时候发现，若共享内存以一个环形队列组织，当多个生产者写入共享内存时还是需要用到锁，除非每个生产者有独立的生产空间，但这样的实现可扩展性较差。既然仍然需要用到锁，且相较于最终的实现方案多了进程间通信，所以我们认为不如直接由每个线程负责节点的内存写入和硬盘写入，而不进行分开，于是生产者消费者模型的方案就被弃用。

虽然看上去好像付出了一些无意义的劳动，但实际上在这个过程中，也让自己对整个项目有了更深入的理解，且我认为，在通往成功的道路上，总需要有试错，很多思路可能只有当真正去尝试落地时才能证实其可行性。当有多个思路的时候，就应该都去尝试实现，只有尝试了才知道哪些更好，在这个过程中也会有可能产生更优的实现思路。

总之，通过此次大作业，我收获了许多，对于B+树的结构、批量加载的原理有了更加深刻的理解，同时也提高了多线程编程的能力，懂得如何去设计一个并行程序并最终分析并行结果，认识到并行相较于串行的利弊。同时，也意识到自己的一些不足，比如对于代码的阅读理解以及编程能力还有待提高，希望接下来能够有机会继续提高。

19335008 曾家洋

在这次实验中，我们一起进行了如何进行并行设计和性能优化的讨论。我的工作主要是和何杰同学一起完成B+树的输出，以及编写了实验的测试报告部分。

在这次实验的过程中，我通过看代码，理解了`bulkLoad`的串行实现，深入理解了它的自底向上的实现，从叶子节点的key和value的构建，再到索引层的构建，最后到根root，每一层都按照相应的起始点的顺序处理。同时，我深刻理解了文件的层次结构，以及各个文件之间的作用，对给的代码的内存，块的分配和写入的实现有了深刻的理解。

在进行小组讨论后，我们确定了并行的思路，明确了分工，各个环节有条不紊的进行，使得我们对之后的优化也充满信心。我们采用的并行思路是采用先分配好块的空间，以及分配好块号，并行地去填充每个Block的内容。之后将内容写入文件，也就是写入硬盘。I/O的优化也使得我们收获了满满的成就感。我们使用了连续分配块和写入块的方式，使得I/O的瓶颈得以很好地解决，性能得到了明显的提升。

在这个过程中，我不仅对并行的概念有了更深层次的理解，还知道如何使用`p_thread`，使用连续地分配和写入硬盘，以及如何更好地去编写报告。接下来我会在更多的方面更好地提升自己。

19335018 陈俊熹

在这次课程设计中，我主要提出bulkload并行实现的算法基本思想并负责部分核心代码的实现，完成了`parallelBulkLoad`、`batchLoadLeaf`、`batchLoadIndex`，并实现了`append_blocks`、`init_noalloc`和`alloc_blocks`连续分配块的优化，同时完善实验报告。

在算法的实现过程中，多线程的实现思路其实比较简单，这主要是这学期的数据库和上学期操作系统分别对B+树和pthread多线程编程都有很深入的讲解和实验。在这一次课程设计中，对我来说最大收获就是对IO效率和开销的思考，如上网查fwrite的原码并思考如何调用效率最高，最后以调用fwrite一次性预分配存储一层节点所需的块和增大节点写回的粒度的方式来减少fwrite的调用次数，在实际运行中运行时间显著减少。这些思考和查资料过程让我对C库的文件I/O API有了更多的了解。

19335054 何杰

本课程的大作业为B+树bulkloading多核并行设计。

在此之前我只了解过B+树的相关知识，对于bulkloading 的内容充满未知，所以在实验开始之前，了解bulkloading 的知识乃当务之急。通过对源代码的详细阅读，我算是对bulkloading 的过程有了一定的了解。Bulkloading 即对有序的数据进行批量构建B+树的过程，有别于通常情况下树自顶向下的构建，bulkloading 是先从底层叶子节点构建，从左往右按顺序构建一个双向链表，从下往上，一层层构建索引节点，每一层也是从左往右构建索引节点。在串行情况下，先构建完叶子节点，然后一层层构建索引节点，逻辑清晰明了。如果想用并行实现，首先就要确定在什么步骤实现并行化，例如每个线程负责一层，但如果没有上一层的数据就无法构建下一层，故该方法无法实现。经过谈论后，我们小组决定提前串行分配好一层的空块，然后并行填充数据，每个线程将自己负责的数据填写入一定数量的节点中，线程数量和节点数量等数据可以根据数据大小和已知条件计算出来。然后每一层都设定一个数组保存节点的key值，下一层的构建也可依据此来进行，直至根节点。项目还有硬盘IO等方面的内容，我们小组的解决方法是设定一个锁，每当一个节点满时就申请这个锁，得到这个锁就将未写入的数据全部写入硬盘。

实验结果显示并行情况下的效率更高，用时更短。通过这次大作业，让我对并行设计和B+树有了更为深刻的理解。

我在此次课程设计参与了并行设计的讨论，和曾家洋同学一起讨论完成了B+树的遍历输出，并自己完成了算法流程图和并行思路可视化的绘制。

19335286 郑有为

在本次实验中，我负责编写（第1、2、4、5、7节）、整合实验报告、实验心得汇总、组织组员，并完成了优化二，即连续块写入部分的代码编写（`write_blocks`、`write_leaf_blocks`和`write_index_blocks`）。

通过本次实验，我们深入理解了B+树的结构，掌握了构建B+树的BulkLoading算法，理清了B+树实现数据索引中的一系列概念，包括块、数据项、索引项、键值等等，并应用以前在操作系统中学习的Pthread实现了并行的BulkLoading。

在算法设计过程中，我们研究算法耗时的瓶颈，在讨论的过程中，我们一致认为算法耗时的瓶颈在IO读写上，因为文件指针只有一个，每次只有一个线程在往文件中写数据。在俊熹的代码的基础上，我实现了连续写的优化，进一步缩短了运行耗时，加速率高达80%（8线程）。最后我们研究发现，相比于并行过程，优化IO对加速比的影响更显著。

我们使用的并行模式结合了非堵塞和堵塞，当每个线程内的块缓冲满了之后堵塞，其他情况下（如当前完成内存写的块小于MIN_BLOCK时）非堵塞竞争写入IO的锁。在此之前，我们想过使用生产者消费者模式，我为此也编写了一个Demo，即所有生产者线程只负责在内存中写块，一个单独的消费者线程通过一个类似于共享内存的结构体将生产者写完的块写入磁盘，但是这个过程不能很好地利用连续读写IO带来的好处，故没有继续研究。