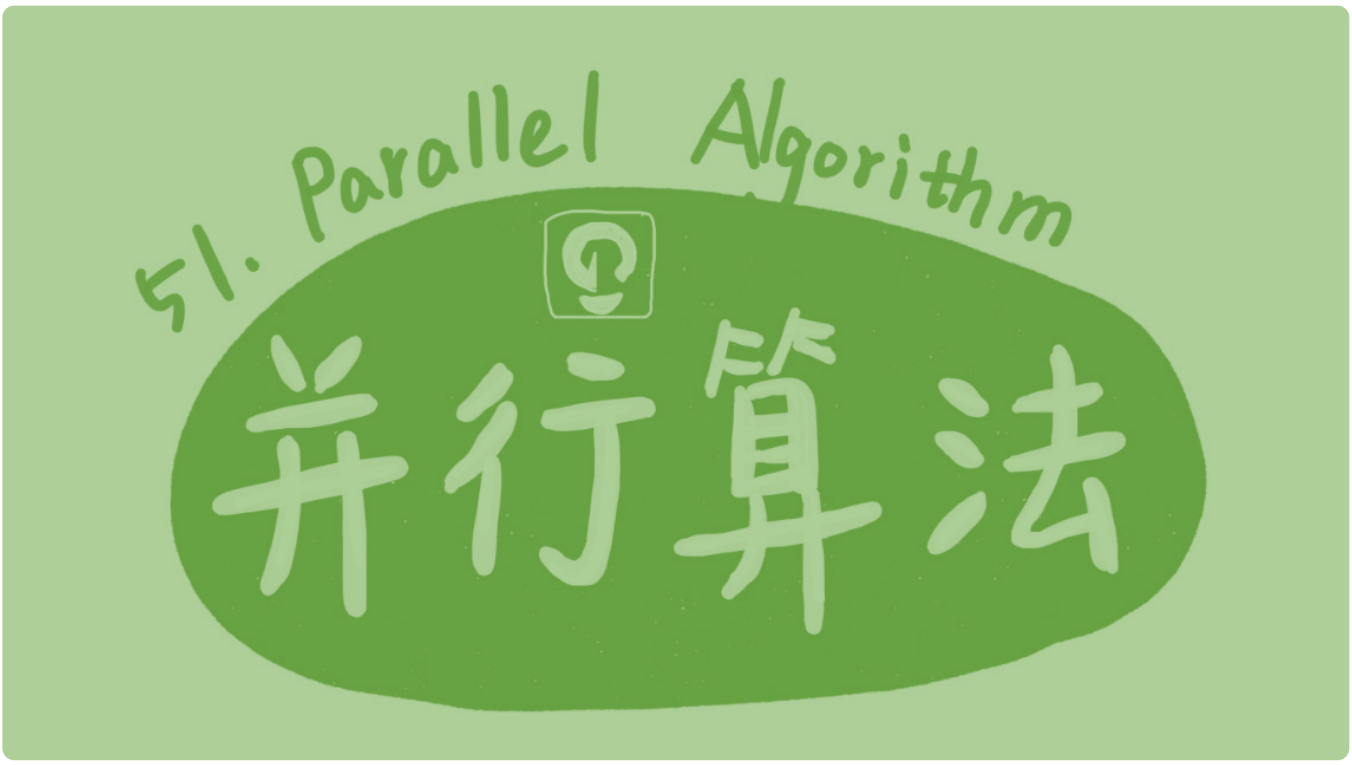
51 | 并行算法：如何利用并行处理提高算法的执行效率？

2019-01-23 王争



朗读人：修阳
时长09:55 大小9.10M

时间复杂度是衡量算法执行效率的一种标准。但是，时间复杂度并不能跟性能划等号。在真实的软件开发中，即便在不降低时间复杂度的情况下，也可以通过一些优化手段，提升代码的执行效率。毕竟，对于实际的软件开发来说，即便是像 10%、20% 这样微小的性能提升，也是非常可观的。

算法的目的就是为了提高代码执行的效率。那**当算法无法再继续优化的情况下，我们该如何来进一步提高执行效率呢**？我们今天就讲一种非常简单但又非常好用的优化方法，那就是并行计算。今天，我就通过几个例子，给你展示一下，**如何借助并行计算的处理思想对算法进行改造？**

并行排序

假设我们要给大小为 8GB 的数据进行排序，并且，我们机器的内存可以一次性容纳这么多数据。对于排序来说，最常用的就是时间复杂度为 O(nlogn) 的三种排序算法，归并排

序、快速排序、堆排序。从理论上讲，这个排序问题，已经很难再从算法层面优化了。而利用并行的处理思想，我们可以很轻松地将这个给 8GB 数据排序问题的执行效率提高很多倍。具体的实现思路有下面两种。

第一种是对归并排序并行化处理。我们可以将这 8GB 的数据划分成 16 个小的数据集合，每个集合包含 500MB 的数据。我们用 16 个线程，并行地对这 16 个 500MB 的数据集合进行排序。这 16 个小集合分别排序完成之后，我们再将这 16 个有序集合合并。

第二种是对快速排序并行化处理。我们通过扫描一遍数据，找到数据所处的范围区间。我们把这个区间从小到大划分成 16 个小区间。我们将 8GB 的数据划分到对应的区间中。针对这 16 个小区间的数据，我们启动 16 个线程，并行地进行排序。等到 16 个线程都执行结束之后，得到的数据就是有序数据了。

对比这两种处理思路，它们利用的都是分治的思想，对数据进行分片，然后并行处理。它们的区别在于，第一种处理思路是，先随意地对数据分片，排序之后再合并。第二种处理思路是，先对数据按照大小划分区间，然后再排序，排完序就不需要再处理了。这个跟归并和快排的区别如出一辙。

这里我还要多说几句，如果要排序的数据规模不是 8GB，而是 1TB，那问题的重点就不是算法的执行效率了，而是数据的读取效率。因为 1TB 的数据肯定是存在硬盘中，无法一次性读取到内存中，这样在排序的过程中，就会有频繁地磁盘数据的读取和写入。如何减少磁盘的 IO 操作，减少磁盘数据读取和写入的总量，就变成了优化的重点。不过这个不是我们这节要讨论的重点，你可以自己思考下。

并行查找

我们知道，散列表是一种非常适合快速查找的数据结构。

如果我们是给动态数据构建索引，在数据不断加入的时候，散列表的装载因子就会越来越大。为了保证散列表性能不下降，我们就需要对散列表进行动态扩容。对如此大的散列表进行动态扩容，一方面比较耗时，另一方面比较消耗内存。比如，我们给一个 2GB 大小的散列表进行扩容，扩展到原来的 1.5 倍，也就是 3GB 大小。这个时候，实际存储在散列表中的数据只有不到 2GB，所以内存的利用率只有 60%，有 1GB 的内存是空闲的。

实际上，我们可以将数据随机分割成 k 份（比如 16 份），每份中的数据只有原来的 1/k，然后我们针对这 k 个小数据集合分别构建散列表。这样，散列表的维护成本就变低

了。当某个小散列表的装载因子过大的时候，我们可以单独对这个散列表进行扩容，而其他散列表不需要进行扩容。

还是刚才那个例子，假设现在有 2GB 的数据，我们放到 16 个散列表中，每个散列表中的数据大约是 150MB。当某个散列表需要扩容的时候，我们只需要额外增加 150*0.5=75MB 的内存（假设还是扩容到原来的 1.5 倍）。不管从扩容的执行效率还是内存的利用率上，这种多个小散列表的处理方法，都要比大散列表高效。

当我们要查找某个数据的时候，我们只需要通过 16 个线程，并行地在这 16 个散列表中查找数据。这样的查找性能，比起一个大散列表的做法，也并不会下降，反倒有可能提高。

当往散列表中添加数据的时候，我们可以选择将这个新数据放入装载因子最小的那个散列表中，这样也有助于减少散列冲突。

并行字符串匹配

我们前面学过，在文本中查找某个关键词这样一个功能，可以通过字符串匹配算法来实现。我们之前学过的字符串匹配算法有 KMP、BM、RK、BF 等。当在一个不是很长的文本中查找关键词的时候，这些字符串匹配算法中的任何一个，都可以表现得非常高效。但是，如果我们处理的是超级大的文本，那处理的时间可能就会变得很长，那有没有办法加快匹配速度呢?

我们可以把大的文本，分割成 k 个小文本。假设 k 是 16，我们就启动 16 个线程，并行地在这 16 个小文本中查找关键词，这样整个查找的性能就提高了 16 倍。16 倍效率的提升，从理论的角度来说并不多。但是，对于真实的软件开发来说，这显然是一个非常可观的优化。

不过，这里还有一个细节要处理，那就是原本包含在大文本中的关键词，被一分为二，分割到两个小文本中，这就会导致尽管大文本中包含这个关键词，但在这 16 个小文本中查找不到它。实际上，这个问题也不难解决，我们只需要针对这种特殊情况，做一些特殊处理就可以了。

我们假设关键词的长度是 m。我们在每个小文本的结尾和开始各取 m 个字符串。前一个小文本的末尾 m 个字符和后一个小文本的开头 m 个字符，组成一个长度是 2m 的字符串。我们再拿关键词，在这个长度为 2m 的字符串中再重新查找一遍，就可以补上刚才的漏洞了。

并行搜索

前面我们学习过好几种搜索算法，它们分别是广度优先搜索、深度优先搜索、Dijkstra 最短路径算法、A* 启发式搜索算法。对于广度优先搜索算法，我们也可以将其改造成并行算法。

广度优先搜索是一种逐层搜索的搜索策略。基于当前这一层顶点，我们可以启动多个线程，并行地搜索下一层的顶点。在代码实现方面，原来广度优先搜索的代码实现，是通过一个队列来记录已经遍历到但还没有扩展的顶点。现在，经过改造之后的并行广度优先搜索算法，我们需要利用两个队列来完成扩展顶点的工作。

假设这两个队列分别是队列 A 和队列 B。多线程并行处理队列 A 中的顶点，并将扩展得到的顶点存储在队列 B 中。等队列 A 中的顶点都扩展完成之后，队列 A 被清空，我们再并行地扩展队列 B 中的顶点，并将扩展出来的顶点存储在队列 A。这样两个队列循环使用，就可以实现并行广度优先搜索算法。

总结引申

上一节，我们通过实际软件开发中的“索引”这一技术点，回顾了之前学过的一些支持动态数据集合的数据结构。今天，我们又通过“并行算法”这个话题，回顾了之前学过的一些算法。

今天的内容比较简单，没有太复杂的知识点。我通过一些例子，比如并行排序、查找、搜索、字符串匹配，给你展示了并行处理的实现思路，也就是对数据进行分片，对没有依赖关系的任务，并行地执行。

并行计算是一个工程上的实现思路，尽管跟算法关系不大，但是，在实际的软件开发中，它确实可以非常巧妙地提高程序的运行效率，是一种非常好用的性能优化手段。

特别是，当要处理的数据规模达到一定程度之后，我们无法通过继续优化算法，来提高执行效率 的时候，我们就需要在实现的思路上做文章，利用更多的硬件资源，来加快执行的效率。所以，在很多超大规模数据处理中，并行处理的思想，应用非常广泛，比如 MapReduce 实际上就是一种并行计算框架。

课后思考

假设我们有 n 个任务，为了提高执行的效率，我们希望能并行执行任务，但是各个任务之间又有一定的依赖关系，如何根据依赖关系找出可以并行执行的任务？

欢迎留言和我分享，也欢迎点击“请朋友读”，把今天的内容分享给你的好友，和他一起讨论、学习。

© 版权归极客邦科技所有，未经许可不得转载

上一篇 50 | 索引：如何在海量数据中快速查找某个数据？

下一篇 52 | 算法实战（一）：剖析Redis常用数据类型对应的数据结构

精选留言

写留言

失火的夏天
2019-01-23

9

思考题用一个有向图来存储任务之间的依赖关系，然后用拓扑排序的思想来执行任务，每次都找到入度为0的，放在队列里，启动线程池开始执行，队列里的任务并行执行完毕，再次调用拓扑排序找到入度为0的人，放入队列，直到所以任务跑完

hua168
2019-01-23

1

老师，我就问一个题外问题：
大专学历，想直接自学考本科或研究生，自考学历IT类公司承认的吗？
很多都要求全日制本科~~

Jerry银银
2019-01-23

1

一看到依赖，就想到了拓扑。

这种感觉好是还是不好呢？

子嘉
2019-01-23

1

思考题是：拓扑排序么。。

茴香根
2019-01-23

1

思考题讲的够直白了，n个任务有互相依赖。那么并行处理的方法就要采用流水线的思想了。创建n个线程，每个线程完成一个任务。每个线程在它的上游线程结束输出结果后启动，完成之后把结果传递给下游任务线程继续流程。整个工作场景像工厂里面的流水线...
展开

睡痴儿😑
2019-01-27

0

使用的是拓扑排序，刚开始还以为太简单。不可能是这样

才才
2019-01-25

0

打卡，继续学习。

小苏饼
2019-01-24

0

spark，oozie的有向无环图

国富
2019-01-24

0

既然有依赖关系，我条件反射想到拓扑排序算法，根据依赖关系把任务分组，各组任务按照依赖关系排序。没有依赖关系的任务组可以并行执行，有依赖关系的任务组内则按依赖关系有序的执行。

2019-01-23

0

计算机不一定都是n核的，怎么实现性能提升n倍呢

作者回复: 即便是n核也不一定提高n倍，毕竟还共享内存呢：）

我这里本就是粗略的表示:)

Smallfly
2019-01-23

0

并行搜索只用一个队列不可以么?

作者回复: 也可以的。不过就有可能导致没法找到最短路径了。

纯洁的憎恶
2019-01-23

0

并行与分治的区别是什么？前者偏工程，后者偏算法么？还是前者在并发环境中，后者在单核串行环境中？

展开

作者回复: 并行是一种工程思路。分治是一种算法思想。感觉差不多哈；）你理解就好，不要太纠结；）

纯洁的憎恶
2019-01-23

0

用Dag图

feifei
2019-01-23

0

我想到了图，讲依赖关系抽象成边，使用图排序就可以找出依赖关系，然后将每一层的任务放入线程池执行，当一层完成后，继续下一层处理

展开

farFlight
2019-01-23

0

各个任务之间有依赖关系的话可以按照前面讲的拓扑排序来决定任务的执行顺序吧

虫儿飞
2019-01-23

0

回答问题：我认为可以考虑按依赖关系分组，组外并行，组内串行。但若是生产者消费者模型，组内都可以并行计算。老师您觉得呢？

展开