超级计算机原理与操作第四次作业
Pthread 编程

April 12, 2023

1 欧拉公式

并非所有和 π 有关的研究都旨在提高计算它的准确度。1735 年，欧拉解决了巴塞尔问题，建立了所有平方数的倒数和 π 的关系: $\frac{\pi^2}{6} = \sum_{i=1}^{\infty} \frac{1}{i^2}$

请使用 pthread 中的 semaphore 计算 $\frac{\pi^2}{6}$ 的值。

可以参考课件中的方法，在参考代码中提供了运行所需的主函数，也提供了串行代码供同学们参考；请同学们将并行的代码补充完整，需要补充的部分见注释 PLEASE ADD THE RIGHT CODES 部分。实验报告中请展示相应的运算结果，并分析加速比随 n 变化的关系。

2 生产者消费者问题

有一个生产者在生产产品，这些产品将提供给若干个消费者去消费，为了使生产者和消费者能并发执行，在两者之间设置一个有多个缓冲区的缓冲池，生产者将它生产的产品放入一个缓冲区中，消费者可以从缓冲区中取走产品进行消费，所有生产者和消费者都是异步方式运行的，但它们必须保持同步，即不允许消费者到一个空的缓冲区中取产品，也不允许生产者向一个已经装满产品且尚未被取走的缓冲区中投放产品。在本题中，我们只考虑一个生产者一个消费者的情况，具体流程如图:

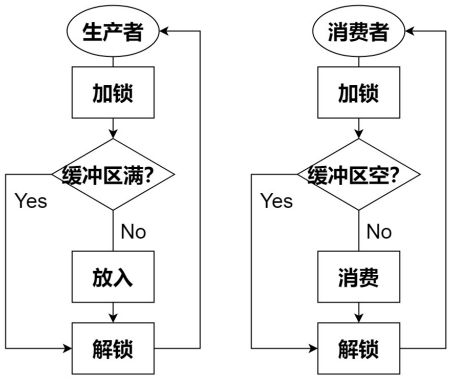

Figure 1: 一把锁实现的生产者消费者队列

1. 在参考代码中提供了运行所需要的主函数，请同学们将代码补充完整。

2. 参考代码只使用了一把锁来实现生产者消费者队列，但是这存在一个问题，极端情况下，生产者每次都加锁成功，那缓冲区会满，产品无法放入缓冲区。消费者会饥饿，因为他一直无法获得锁，请考虑如何解决饥饿问题。

实验报告中请展示相应的实验过程和运算结果。

3 线程池

线程池 (thread pool)：是一种线程的使用模式。在实际系统中，频繁的创建销毁线程会带来过多的调度开销，从而影响整体性能。线程池维护着多个线程，等待着监督管理者分配可并发执行的任务。这避免了在处理短时间任务时创建与销毁线程的代价。线程池不仅能够保证内核的充分利用，还能防止过分调度。

在本习题中，我们采用任务队列的模式来实现一个线程池：主线程负责将相应的任务放入任务队列中，工作线程负责从任务队列中取出相应的任务进行处理，如果任务队列为空，则取不到任务的工作线程将进入挂起状态。具体代码见 github:

https://github.com/Monaco12138/Threadpool-2023-DCS244-homework4

请根据要求填充完成空缺的部分，实现一个简单的线程池。

作业要求

1. 提交作业的时候需要提交一份报告（PDF 格式）和相应的源代码，将材料打包命名为姓名 _ 学号 _hw4.zip 提交
2. 提交地址：easyhpc 课程主页
3. 请在截止日期前提交
4. 禁止抄袭