并行程序设计与算法

Assignment 1

2025-03-10

1: 简答题

Problem 1.1:

当 CPU 将数据写入缓存时，缓存中的值可能与主存中的值不一致，有哪两种解决策略？请简要描述这两种策略的优缺点。

Problem 1.2:

请简要解释时间局部性和空间局部性的概念，对于以下的代码片段，请简述代码中存在的这两种局部性。

```
int a[1000][1000];
int sum = 0;
for (int i = 0; i < 1000; i++) {
  for (int j = 0; j < 1000; j++) {
    sum += a[i][j];
  }
}
```

Problem 1.3:

请简述 SIMD 和 MIMD 的概念，以及这两种并行计算模型的区别。

2: 全局求和问题

全局求和问题是课本中的一个经典问题，我们假设有一个长度为 `n` 的数组 `a`，我们需要求出所有 `a[i]` 的和。现在假设我们有 `p` 个处理器核心，有以下的伪代码实现：

```python
def main(a, p):
  sum = 0
  for i in range(p):
    sum += local_sum(a, i) # 将该任务分配给处理器核心 i 进行计算
  return sum

def local_sum(a, i):
  # 此函数在处理器核心 i 上计算数组 a 的和，内部变量均为该核心上的局部变量
  my_sum = 0, my_left = ???, my_right = ???
  for j in range(my_left, my_right): # 包含 my_left，不包含 my_right
    my_sum += a[j]
  return my_sum
```

Problem 2.1:

假设各个处理器核心的计算能力相同，我们需要使得每个处理器核心的计算任务尽可能均衡，同时保证算法的时间复杂度尽可能低。请推导出对于 $\forall i \in \{0, 1, ..., p-1\}$，`local_sum` 函数中 `my_left` 和 `my_right` 的取值计算表达式。

Problem 2.2:

对于数组 `a` 来说，当 $n = 10^5$ 时，我们使用 $p = 1,000$ 个处理器核心进行计算。假设每一次加法运算的总耗时为 1，每一次乘法运算的总耗时为 10，不考虑比较函数、内存、寄存器访问时延和流水线并行等一切其他操作耗时，假设 `main` 函数需要等待所有所有处理器核心的计算结果后才能开始计算全局和，请计算出使用上述伪代码计算数组 `a` 的和所需的总耗时。

Problem 2.3:

假设串行处理器核心的计算能力为 1，即数组 `a` 的和的计算耗时为 10^5，请计算并行导致的加速比。这里的加速比理想吗？ 如果不理想，你认为如何改进？

3: 计算题

Problem 3.1:

假设有一个应用程序，其中 85 % 的代码可以并行化，而剩下的 15 % 代码是串行的。你有一个 10 核处理器，假设你可以完美地并行化代码。在当前的环境下，该应用可以达到 4 倍的加速比吗？ 如果可以，你需要多少处理器核心？此时的加速比是多少？如果不可以，你认为的最大加速比是多少？

Problem 3.2:

假定 $T_{\text{串行}} = n$, $T_{\text{并行}} = \frac{n}{p} + \log_2^p$。如果以倍率 k 增加 p 的值，为了保证**效率（The parallel efficiency**）不变，需要如何增加 n 的值？该并行程序是可扩展的吗？是强可扩展还是弱可扩展？

其中，T 为总时间，n 为问题规模，p 为处理器核心数（进程/线程数）。