Scrapy 参数优化

settings 参数调优

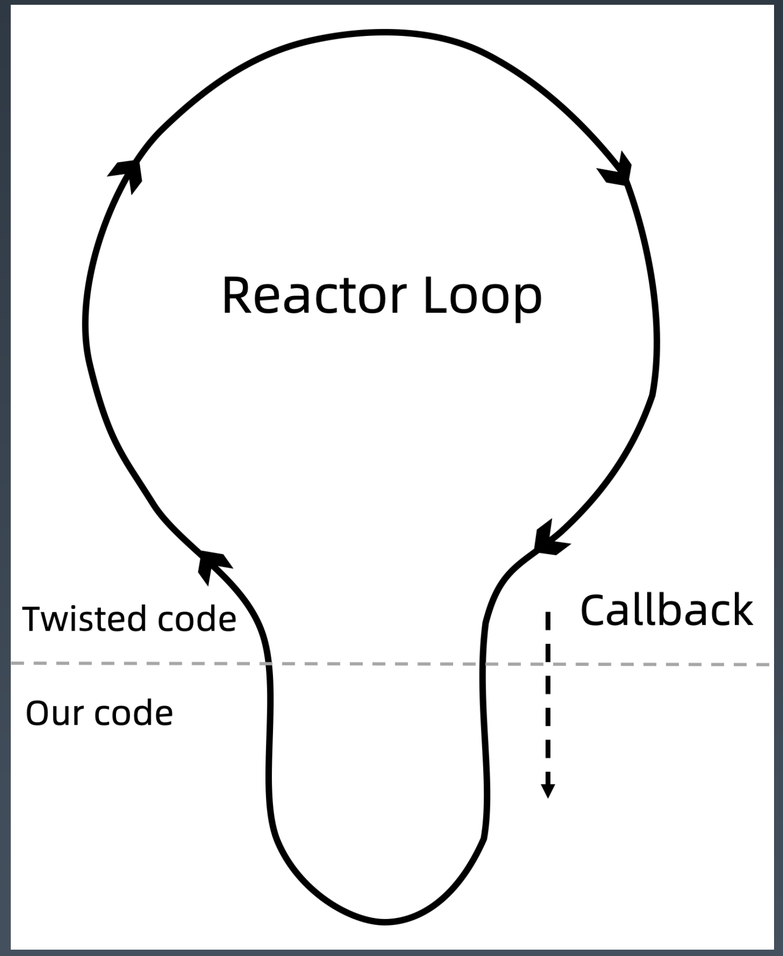
```
# Configure maximum concurrent requests performed by Scrapy (default: 16)
# CONCURRENT_REQUESTS = 32

# Configure a delay for requests for the same website (default: 0)
DOWNLOAD_DELAY = 3

# The download delay setting will honor only one of:
# CONCURRENT_REQUESTS_PER_DOMAIN = 16
# CONCURRENT_REQUESTS_PER_IP = 16
```

基于 twisted 的异步 IO 框架

多任务模型分为同步模型和异步模型

Scrapy 使用的是 Twisted 模型

Twisted 是异步编程模型，任务之间互相独立，用于大量 I/O 密集操作。



多进程与多线程

threading 多线程模型

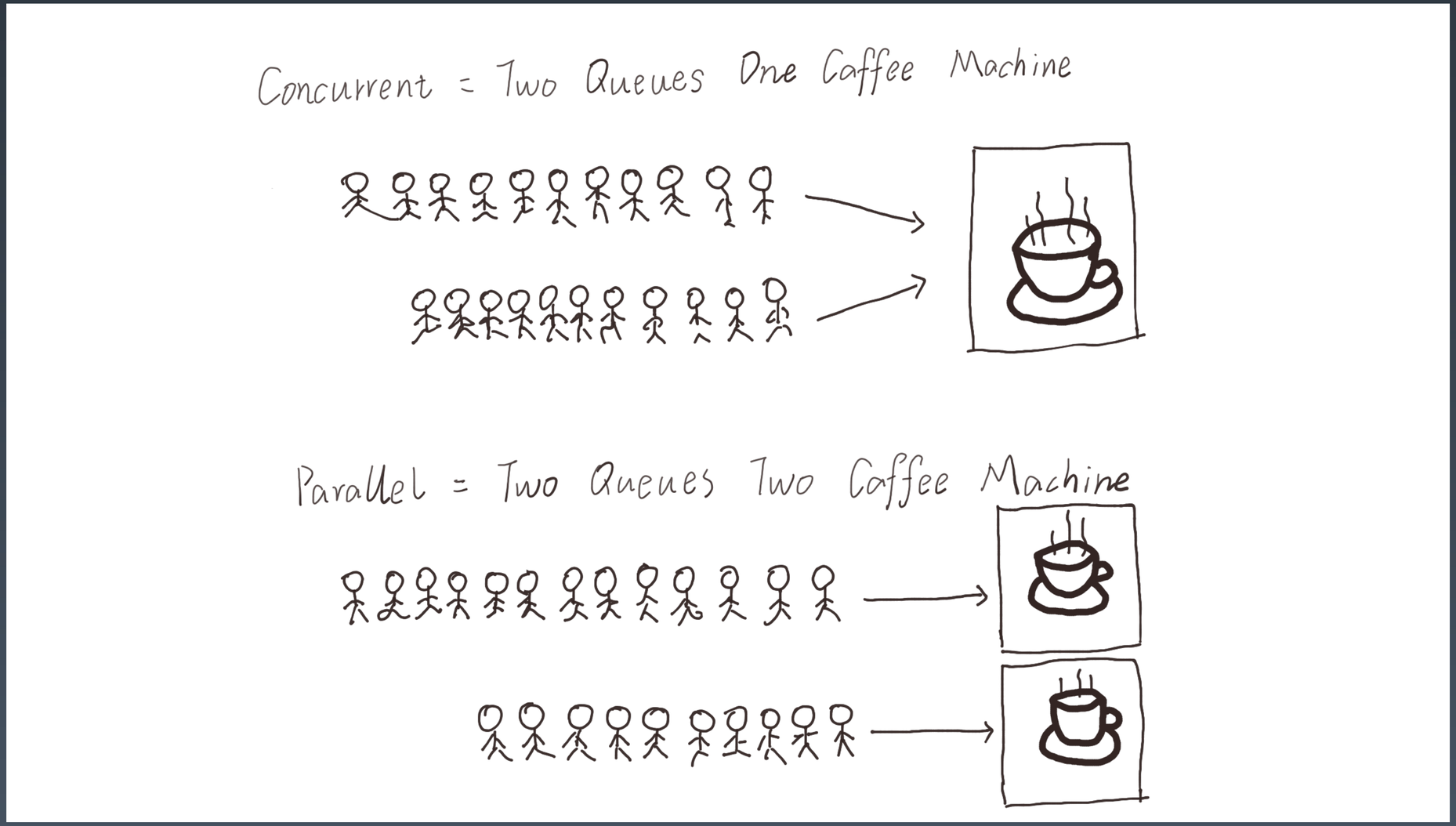
线程同步

多进程模型

队列与线程池

使用 requests 和 threading 库实现 miniScrapy 多线程爬虫框架

并发和并行


Concurrent = Two Queues One Caffee Machine
Parallel = Two Queues Two Caffee Machine

多进程模型

多进程、多线程、协程的目的都是希望尽可能多处理任务

多进程模块使用 multiprocessing 模块

多进程的第一个问题：进程的父子关系

进程间的通信

为什么不能再使用变量作为进程间共享数据了？

主要共享方式：

- 队列
- 管道
- 共享内存

资源的抢占：

- 加锁机制

进程池

使用进程池可以完成更方便地并发任务

pool.map(functions, range(10))

多线程模型

进程与线程的区别是什么？

经常提到的同步和异步、阻塞和非阻塞是什么意思？

为什么有多进程还要有多线程？反过来也是？

为什么要产生协程？

threading 模块

使用函数创建多线程模型

```
threading.Thread(target=run, args=("thread 1",))
```

使用类创建多线程模型

```
class MyThread(threading.Thread)
```

线程锁

线程锁 Lock 与 RLock 实战

其他的线程同步机制介绍：

- 信号量
- 事件
- 条件
- 死锁

队列

线程队列

优先级队列

线程池

一般的线程池
from multiprocessing.dummy import Pool as ThreadPool

并行任务的高级封装（python3.2 版本以后支持）
from concurrent.futures import ThreadPoolExecutor

线程安全问题

单例模式使用双重锁机制保证线程安全

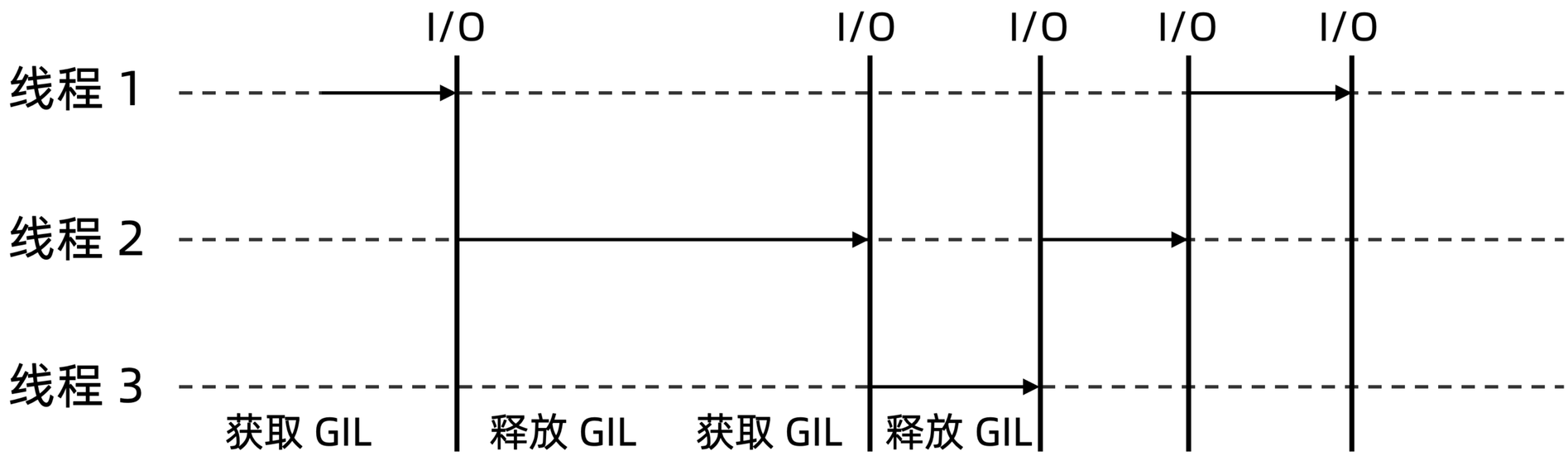
```
class Singleton(object):
    _instance_lock = threading.Lock()

    def __new__(cls, *args, **kwargs):
        if not hasattr(Singleton, "_instance"):
            with Singleton._instance_lock:
                if not hasattr(Singleton, "_instance"):
                    Singleton._instance = object.__new__(cls)
        return Singleton._instance
```

多线程与 GIL

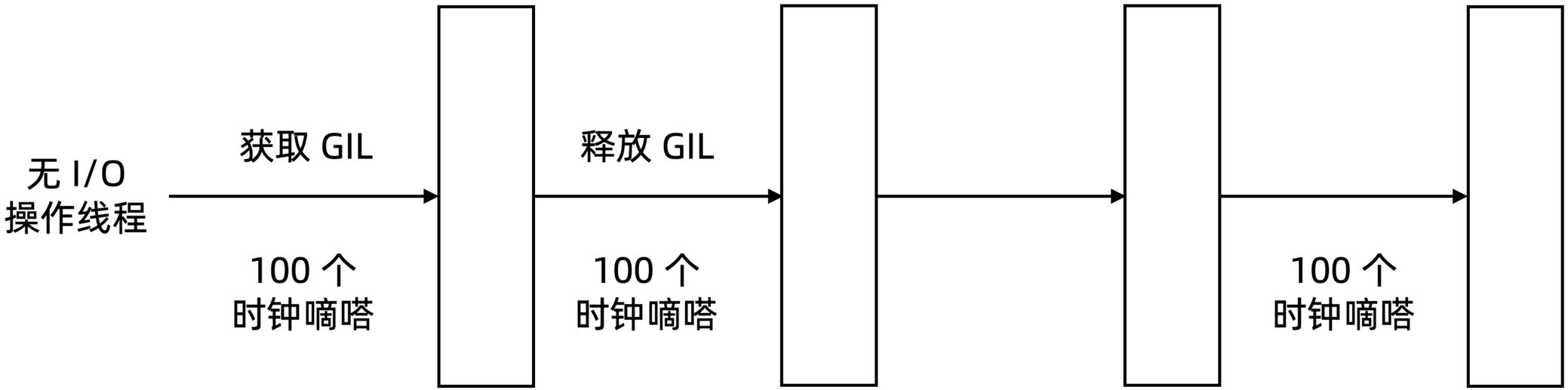
进程与线程性能对比

GIL 对性能的影响

I/O 操作中 GIL 的变换过程



无 I/O 操作时 GIL 的变换过程



无 I/O 操作时 GIL 的变换过程

miniScrapy 多线程爬虫框架

使用 requests 和 threading 库实现 miniScrapy 多线程爬虫框架

THANKS!