中山大學 软件工程学院

SUN YAT-SEN UNIVERSITY SCHOOL OF SOFTWARE ENGINEERING

1924-2024 中山大學 世纪华诞 100th ANNIVERSARY SUN YAT-SEN UNIVERSITY

处理器调度：多核调度策略

SSE202/204: 操作系统原理

苏玉鑫

suyx35@mail.sysu.edu.cn

助教：龙玉丹 单诗雯 毛晨希 沈志轩 郑灿峰 胡伟峰

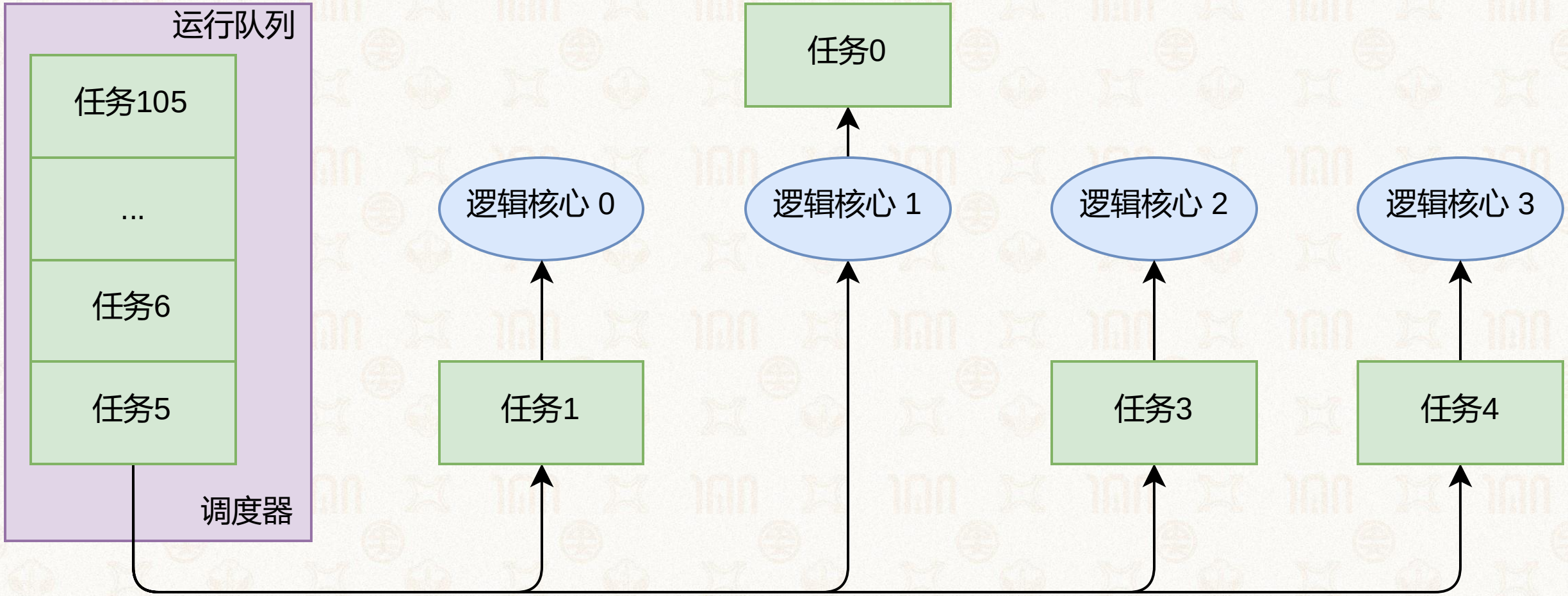
版权信息

- 部分内容来自：上海交通大学并行与分布式系统研究所操作系统课件
  - https://ipads.se.sjtu.edu.cn/courses/os/
- 其它参考资料：
  - 清华大学操作系统公开课
    - https://open.163.com/newview/movie/courseintro?newurl=ME1NSA351
    - 介绍标准内容，适合考研
  - 南京大学计算机软件研究所
    - http://jyywiki.cn/OS/2025/
    - https://space.bilibili.com/202224425/channel/collectiondetail?sid=192498
    - 比较有趣

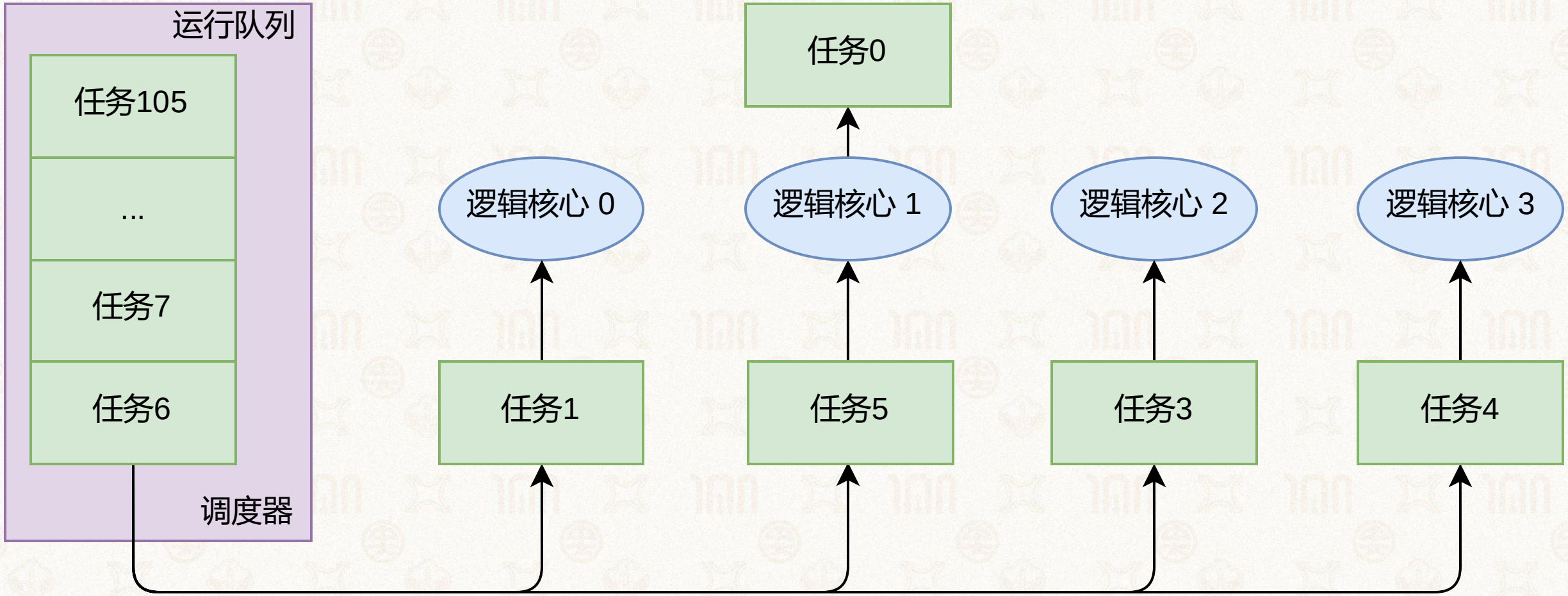
大纲

- 调度的含义
- 调度的机制
- 单核调度策略
  - 经典调度
  - 优先级调度
  - 公平共享调度
  - 实时调度
- 多核调度策略
- 调度进阶机制
  - 处理器亲和性
- 现代Linux调度器

多核下的负载分担

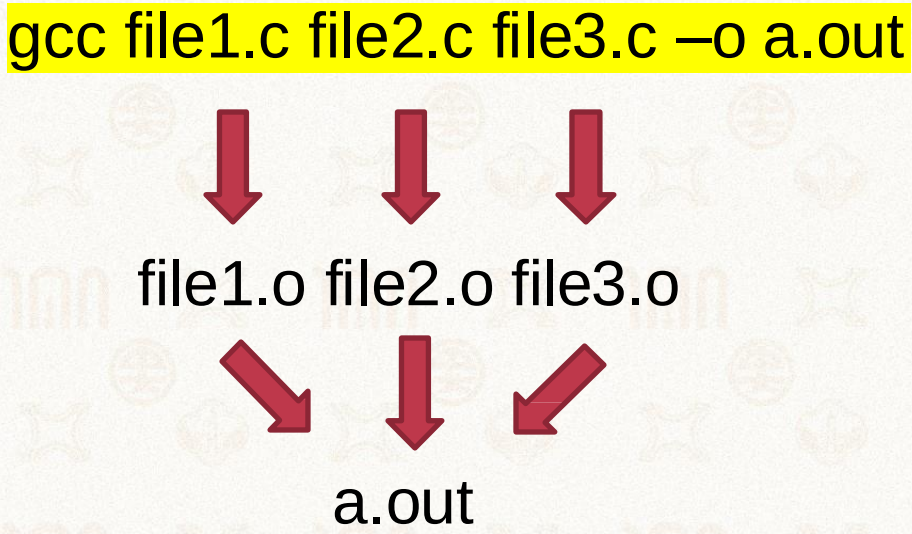

运行队列
任务105
...
任务6
任务5
调度器
任务0
逻辑核心 0
逻辑核心 1
逻辑核心 2
逻辑核心 3
任务1
任务3
任务4

多核下的负载分担


运行队列
任务105
...
任务7
任务6
调度器
任务0
逻辑核心 0
逻辑核心 1
逻辑核心 2
逻辑核心 3
任务1
任务5
任务3
任务4

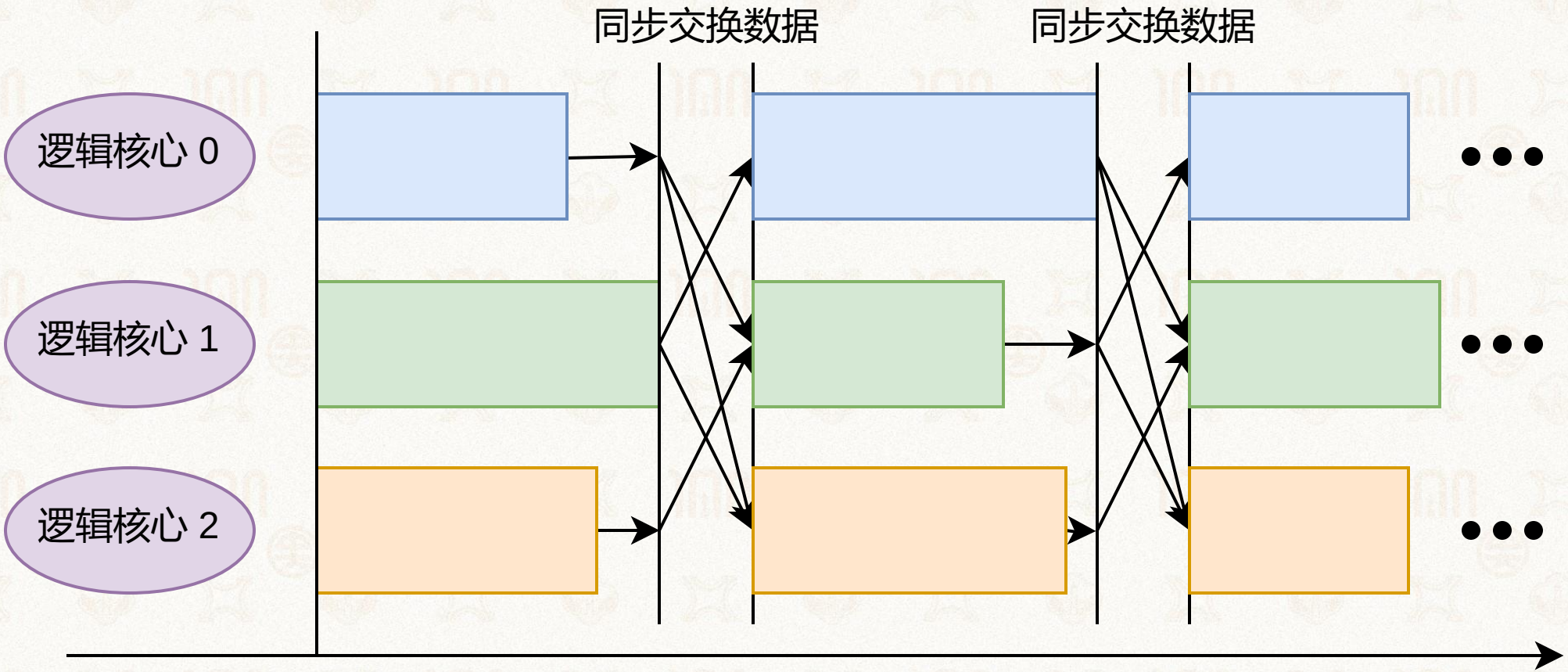
问题1：同一个进程的线程很可能有依赖关系

- 例：GCC 编译a.out文件
- 每个线程编译一个文件（.c到.o）
- 线程间依赖：
  - 所有.o生成完才能进入下一步操作



考虑依赖关系：协同调度

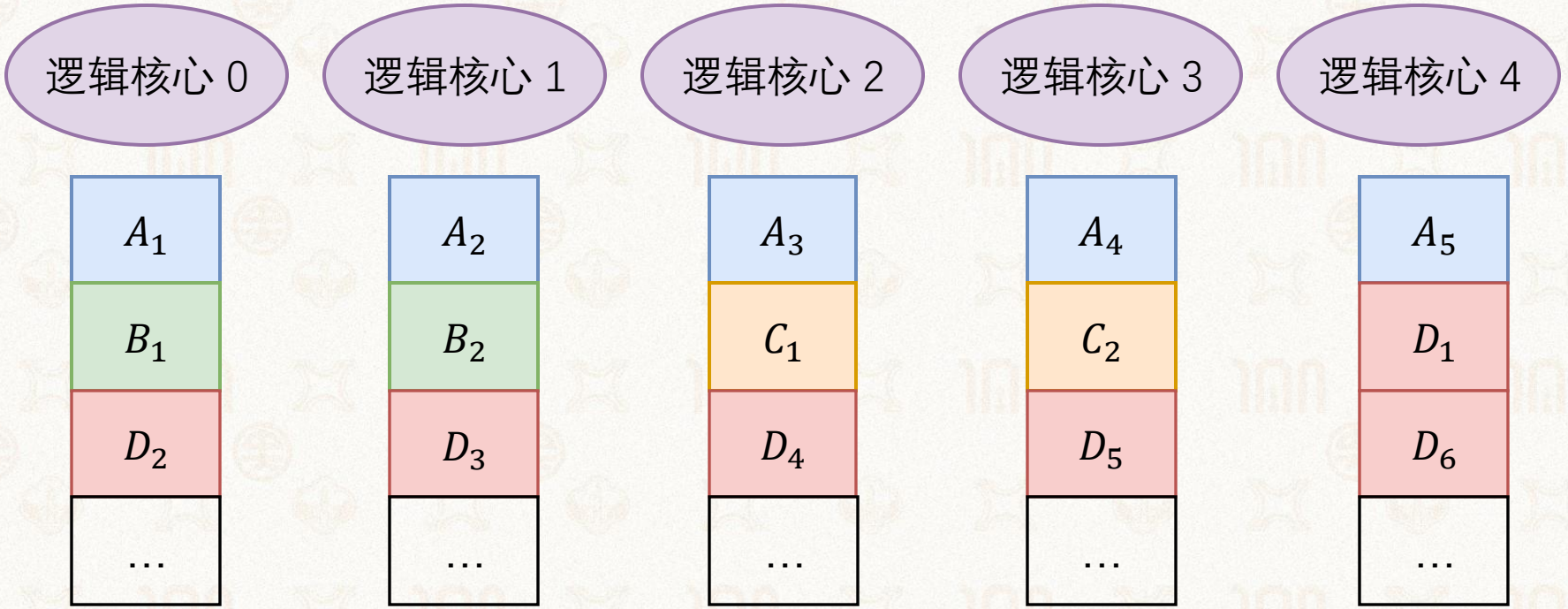
- 整体同步并行(Bulk Synchronous Parallelism, BSP) 计算模型
  - 没有依赖关系的并行执行，有依赖关系的等待下一轮运行



常见于机器学习、图计算、分布式数据处理平台

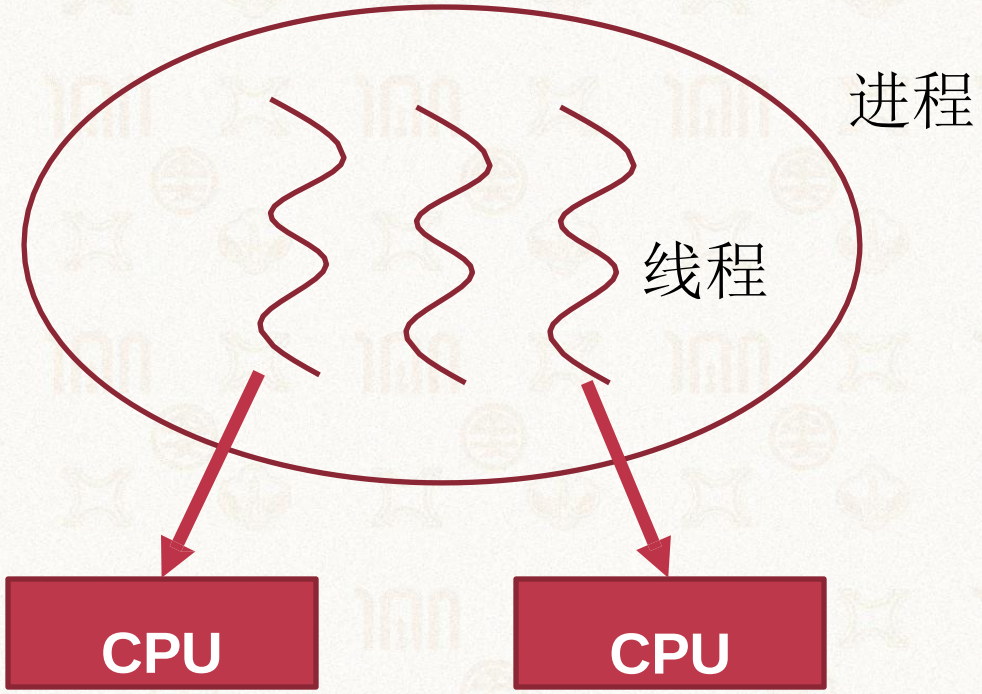
考虑依赖关系：群组调度(Gang Scheduling)

- 没有依赖关系的任务分在同一组，以组为单位进行调度
- 例：把所有任务分为A,B,C,D四组



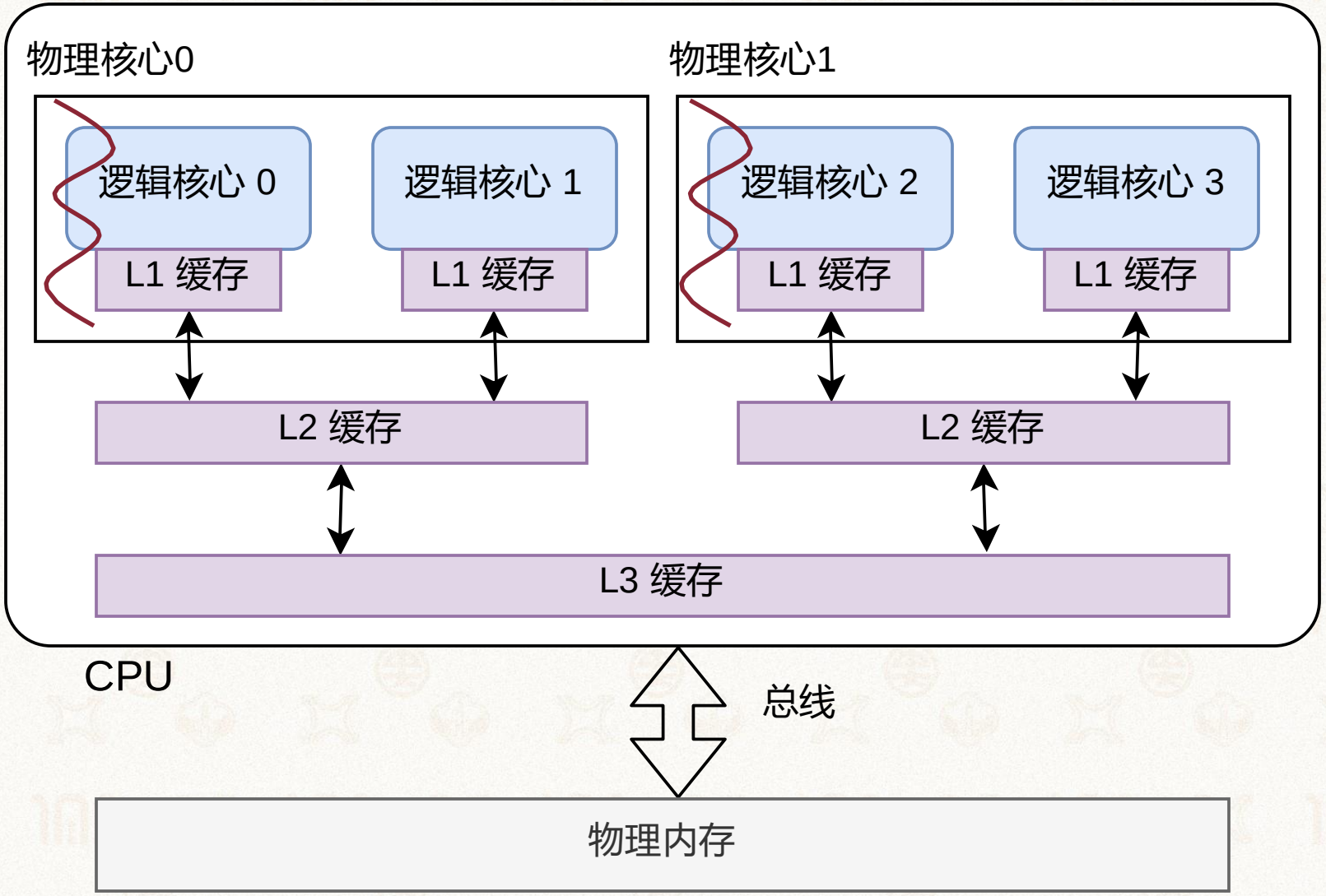
多核调度需要考虑的额外因素

- 一个进程的不同线程可以在不同CPU上同时运行



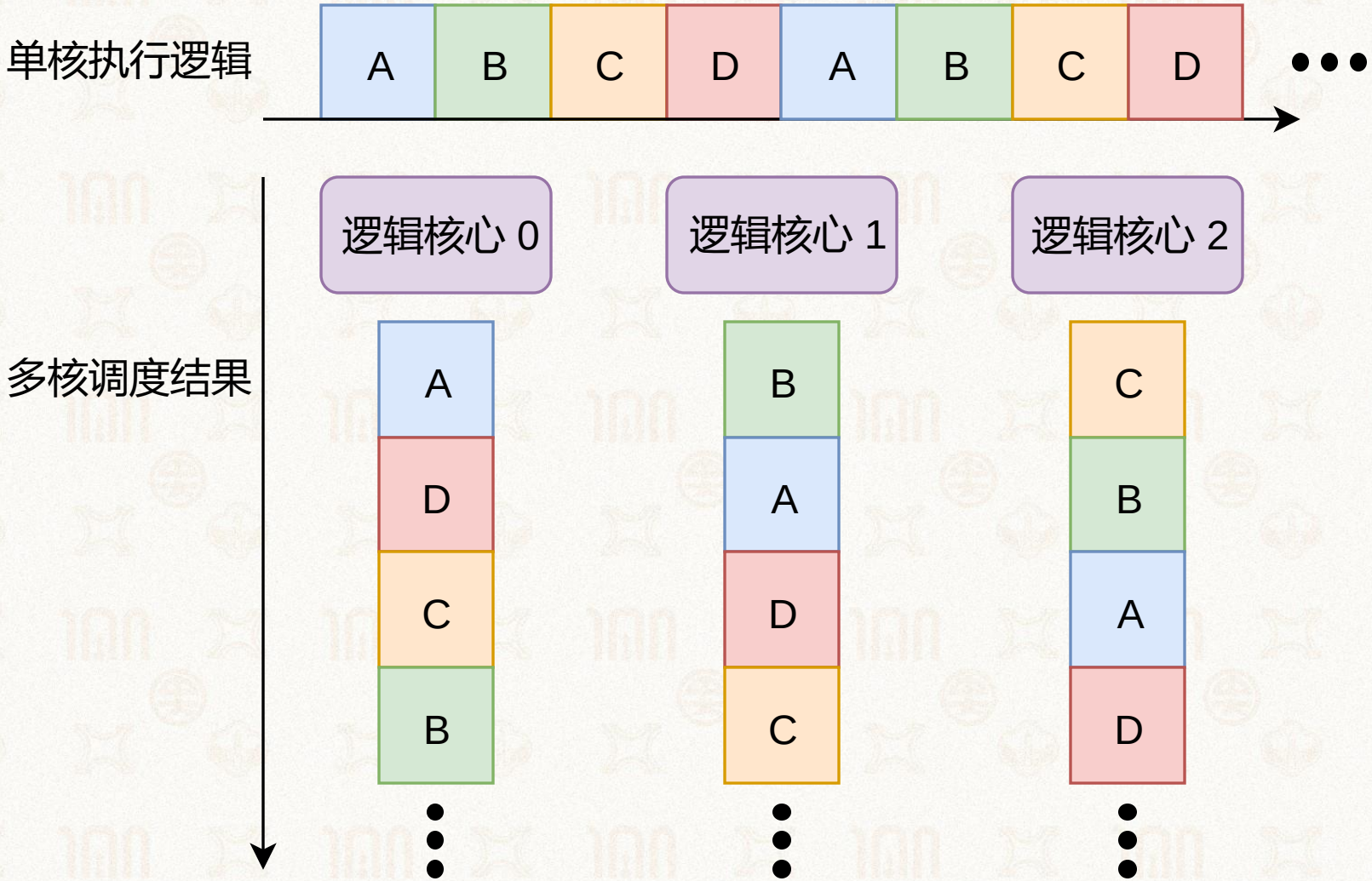
多核调度需要考虑的额外因素

- 一个进程的不同线程可以在不同CPU上同时运行

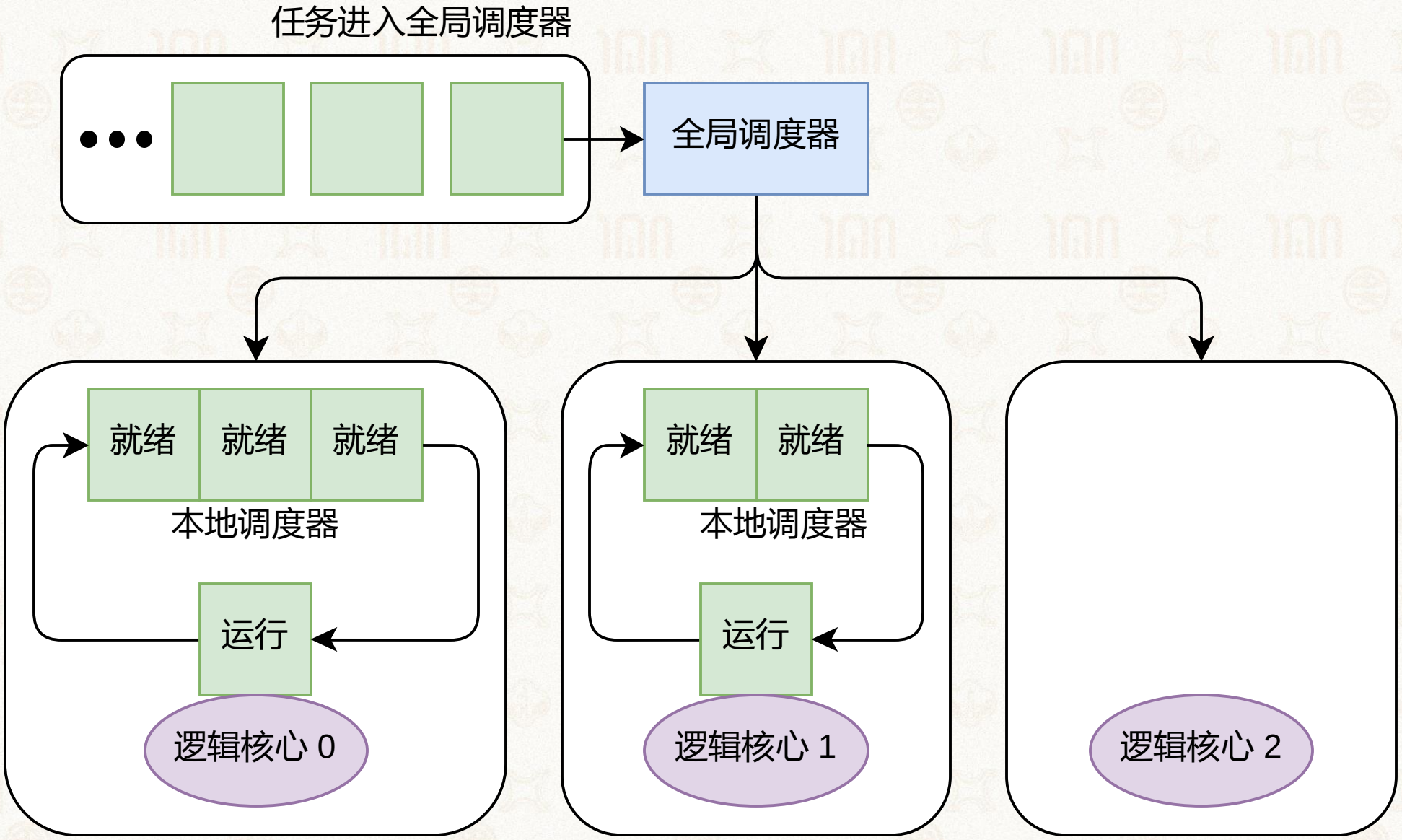


多核下的负载分担

- 问题2：任务在CPU核心间频繁地切换，对缓存不友好

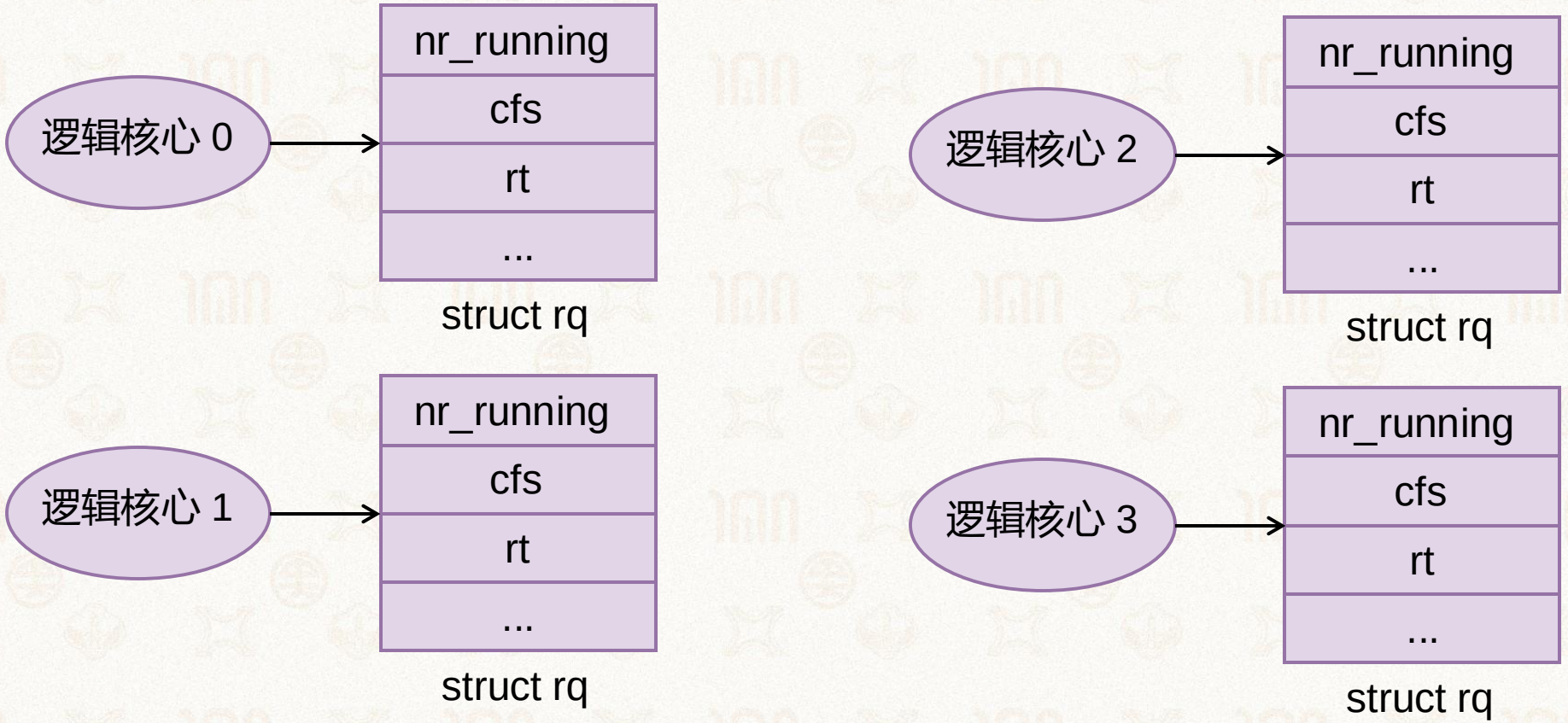


缓存友好型调度：两级调度



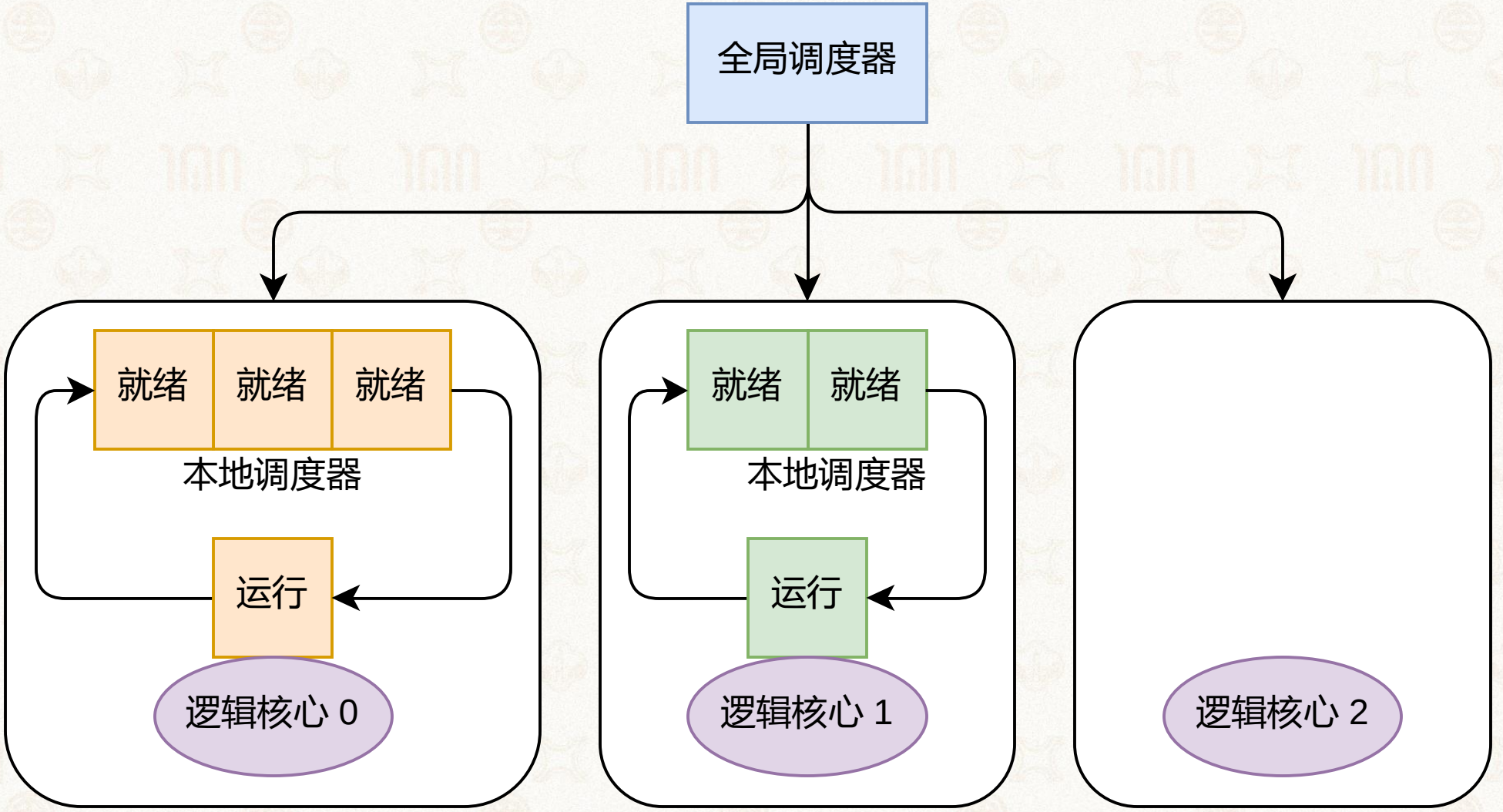
Linux的多核调度

- 标准的两级调度
  - 每个CPU有各自的本地调度器和调度队列(rq)



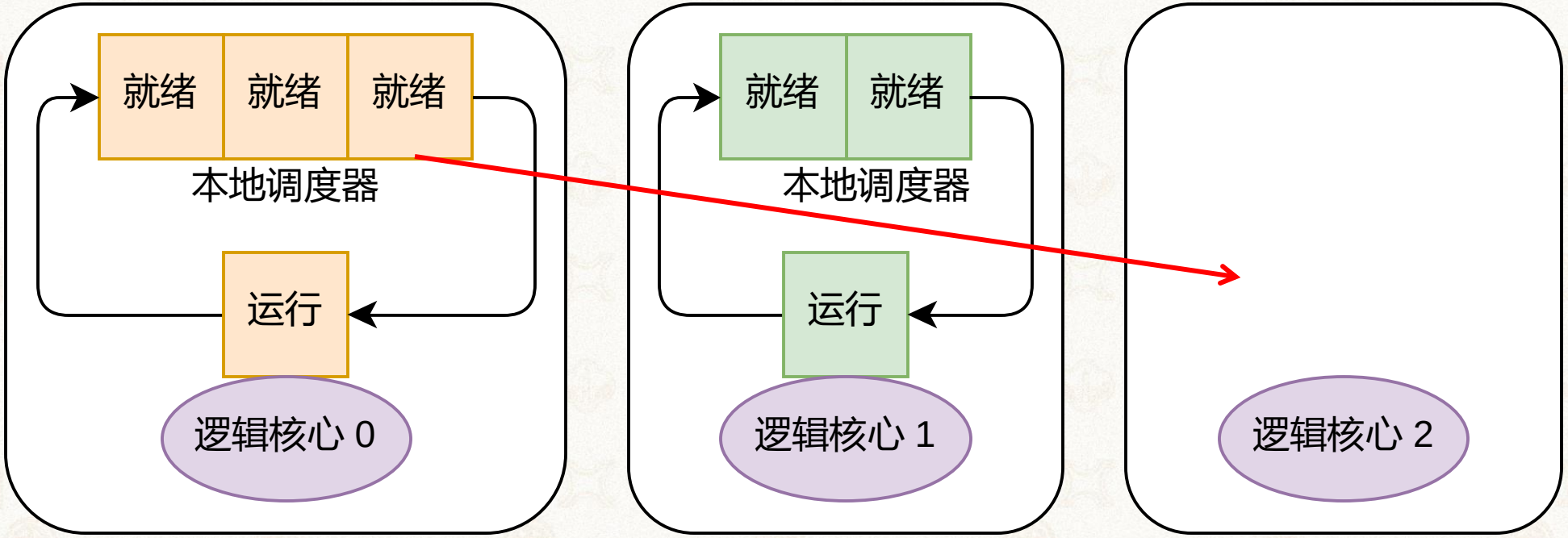
两级调度的问题

- 指定线程在某个CPU上运行容易负载不均衡



负载均衡（Load Balance）

- 每个任务特点不同，不能用任务数量代表真实负载
- 需要追踪CPU的负载情况
- 将任务从负载高的CPU迁移到负载低的CPU



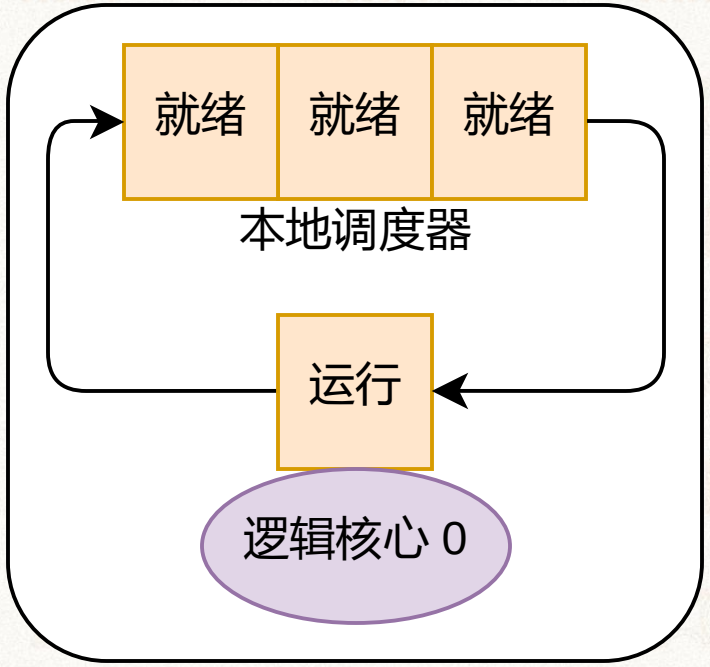
如何进行准确的负载追踪?

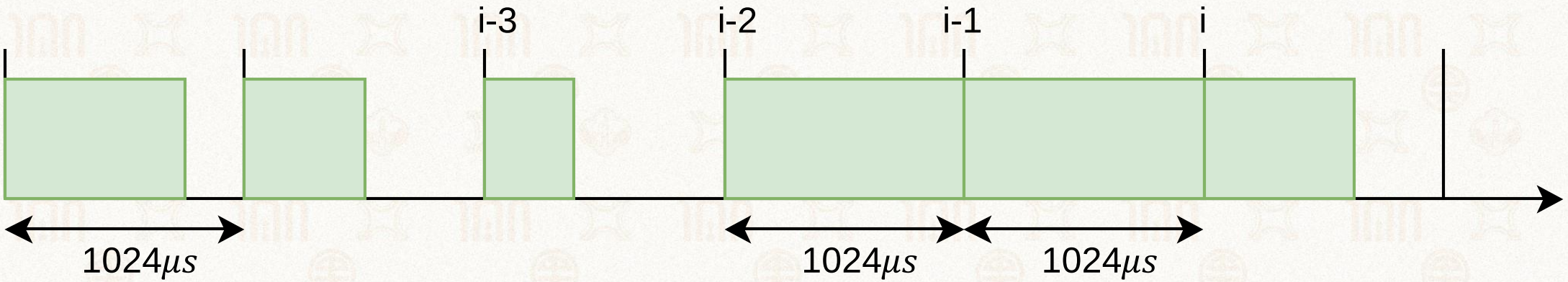
- 方法一：以运行队列为粒度追踪负载
  - 运行队列长，意味着负载高
  - 但从一个高负载的运行队列中，到底选取哪个任务迁移呢？没有
  - 相关信息，因此不够准确

- 方法二：以调度实体为粒度追踪负载
  - 以调度实体（单个任务）为粒度记录负载
  - Linux's PELT: Per Entity Load Tracking，从Linux 3.8开始

PELT: Per Entity Load Tracking

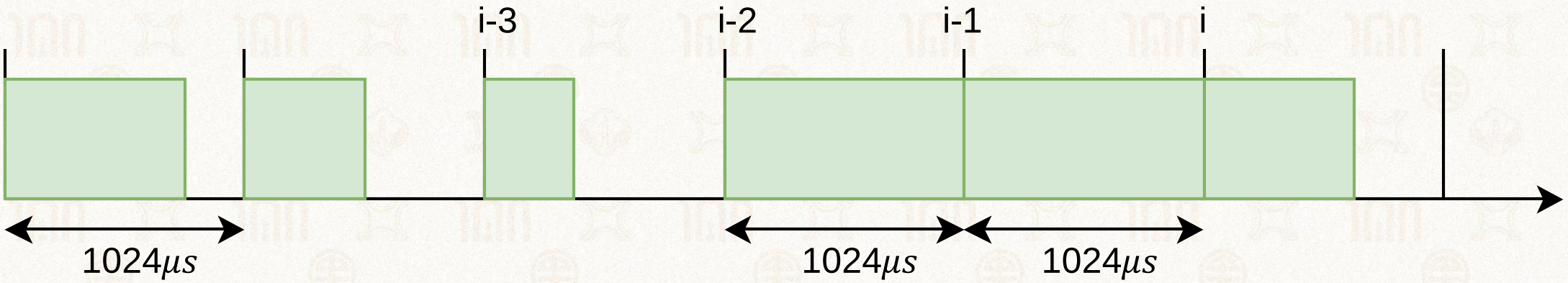
- 记录每个调度实体对负载的贡献(有多少时间在运行)
  - 每1024μs周期，记录线程处于可运行状态的时间x
  - 线程t在第i个周期内的负载为:
    - $L_{t,i} = \frac{x}{1024} \cdot \text{CpuScaleFactor}$
    - 为什么需要$\boldsymbol{CpuScaleFactor}$？
      - 负载是在每个CPU上独立计算的，需要有一个统一的标准
      - 性能高的CPU，Factor也更高





任务总负载的计算

- $\text{Load}_t = L_{t,i} + L_{t,i-1} + \cdots + L_{t,0}$
  - 不合适，任务的负载是变化的
  - 不能因为任务连续运行一个星期，就说它当前负载高
  - 最近的负载权重应该更大
- $\text{Load}_t = L_{t,i} + \gamma \cdot L_{t,i-1} + \gamma^2 \cdot L_{t,i-2} + \cdots + \gamma^i L_{t,0}$
  - Linux设置的衰减因子$\gamma^{32} = 0.5$
    - 当前负载在任务经过32个周期后减半
  - 如何快速计算高次多项式的值？
    - $\text{Load}_t = L_{t,i} + \gamma \cdot \text{Load}_t$



调度总结

- 策略 vs 机制
- 经典调度：先到先得、最短任务优先、时间片轮转
- 优先级调度：多级优先级队列、优先级的选取
- 公平共享调度：彩票调度、步幅调度
- 实时调度：最早截止时间优先(EDF)

大纲

- 调度的含义
- 调度的机制
- 单核调度策略
  - 经典调度
  - 优先级调度
  - 公平共享调度
  - 实时调度
- 多核调度策略
- 调度进阶机制
  - 处理器亲和性
- 现代Linux调度器

处理器亲和性(processor affinity)

- 老码农比内核更懂得该如何调度自己写的程序
- 操作系统允许程序主动选择用哪个逻辑核心来运行

```c
#include <sched.h>
#include <stdio.h>
#include <sys/sysinfo.h>

int main() {
    cpu_set_t mask;
    // 初始化 mask 的CPU集合为空
    CPU_ZERO(&mask);

    // 在 mask 的 CPU 集合中加入逻辑核心0和逻辑核心2
    CPU_SET(0, &mask);
    CPU_SET(2, &mask);

    // 根据 mask 设置当前任务的亲和性
    sched_setaffinity(0, sizeof(mask), &mask);
}
```

大纲

- 调度的含义
- 调度的机制
- 单核调度策略
  - 经典调度
  - 优先级调度
  - 公平共享调度
  - 实时调度
- 多核调度策略
- 调度进阶机制
  - 处理器亲和性
- 现代Linux调度器

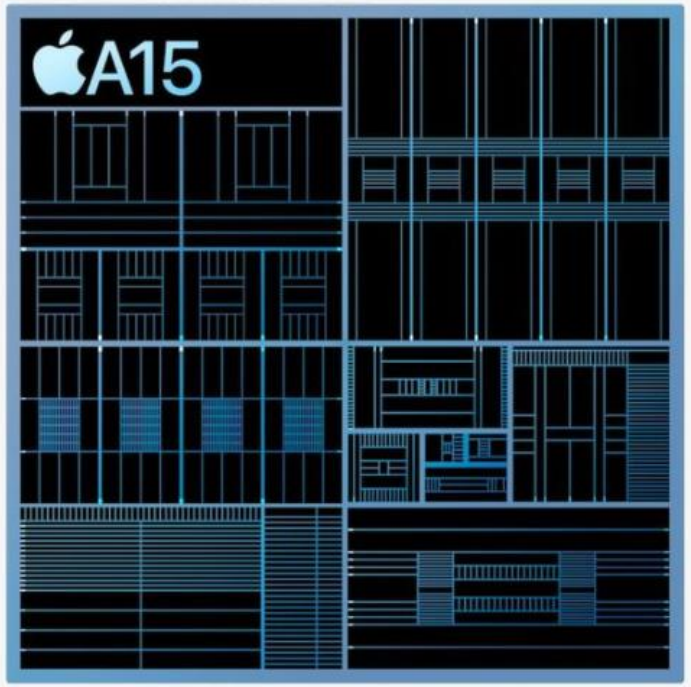
Linux能耗感知调度器

- 现代处理器的“大小”核设计

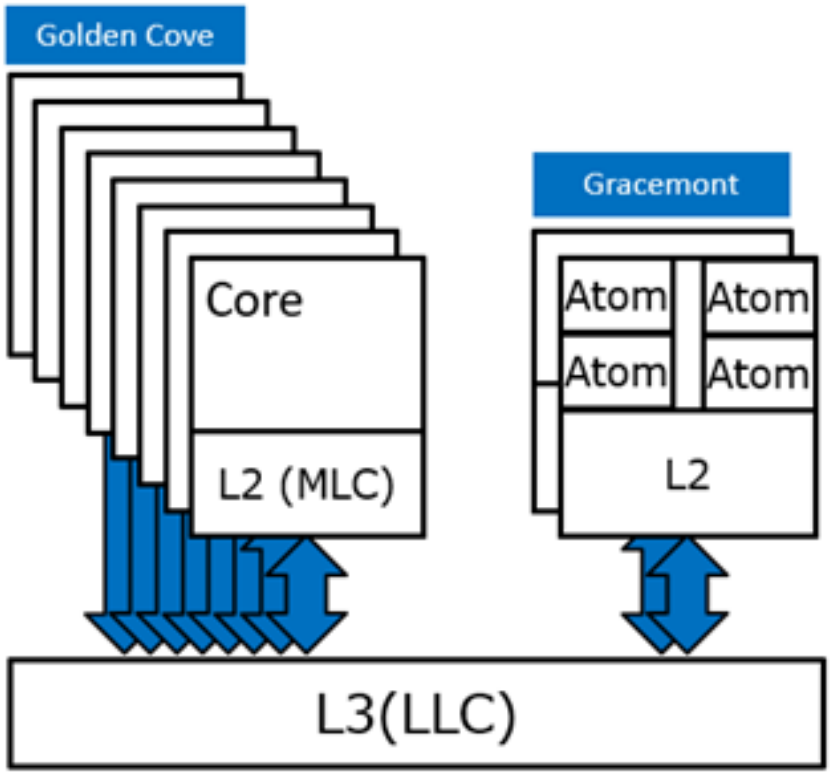
2 new high-performance cores

4 new high-efficiency cores

Faster Neural Engine



苹果2021年出品的A15芯片，采用Armv8架构，拥有2个性能核，4个节能核



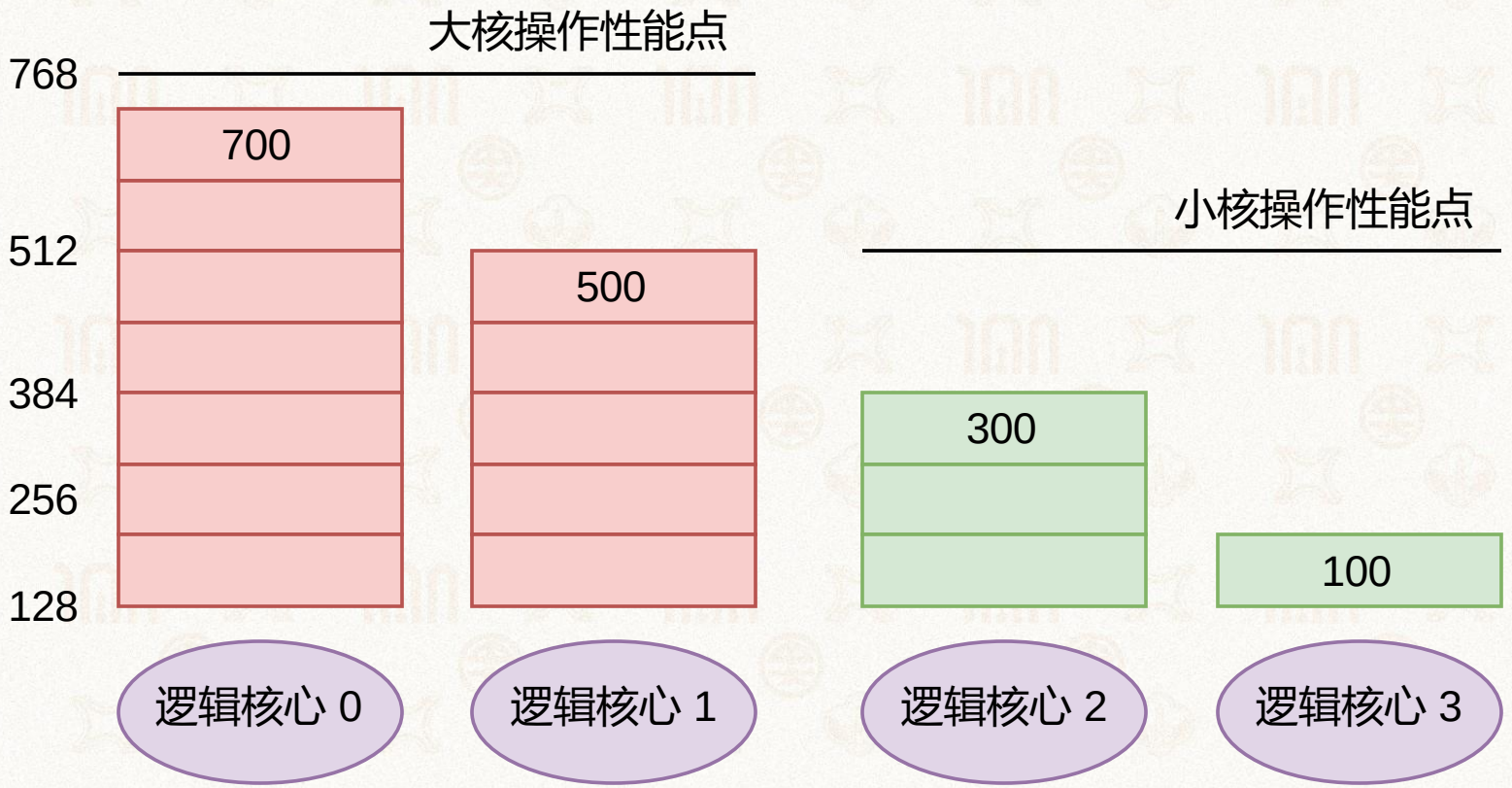
英特尔2021年末出品第12代酷睿处理器，多种性能大核、效能小核组合

Linux能耗感知调度器

- CPU核心的能耗模型
  - [数电] 数字电路元器件功率和哪些因素相关?
    - $W = \frac{1}{2} V f^2$
  - 在CPU中电压V是固定的，只能通过调节频率f来调整功耗
  - 频率降低，指令处理速度变慢，性能下降
  - 性能被标准化为0~1024之间

大核		小核	
性能容量	功率(mW)	性能容量	功率(mW)
384	300	128	80
768	900	256	180
1024	1800	512	440

Linux能耗感知调度器



大核		小核	
性能容量	功率(mW)	性能容量	功率(mW)
384	300	128	80
768	900	256	180
1024	1800	512	440

谢谢

微信：suyuxin
钉钉：苏玉鑫
B站：https://space.bilibili.com/502854403
软工集市课程专区：https://ssemarket.cn/new/course
匿名提问箱：https://suask.me/ask-teacher/106/苏玉鑫