比较串行和并行的内存占用，再选择运行环境

完整操作教程 · 参考固定 · 往返恢复验收

用固定输入和输出验收比较串行与并行，正确区分单进程RSS和任务组内存，再制定CPU与内存方案。

运行边界：示例代码需按安装版本和真实项目输入复核；未提供真实生物数据，语法或小样本检查不代表科学结果和性能验收。

并行能缩短一批独立任务的总墙钟时间，但常会增加同时驻留的R对象和worker开销。比较串行与并行时，必须使用同一输入、代码、包版本、输出目标和资源上限；否则无法判断差异来自worker数量还是数据/环境变化。

建立串行基线

先在sequential模式下跑一个代表性任务，记录启动和结束时间、退出码、峰值内存、输出完整性。确认脚本无并行依赖且失败时会返回非零状态。对计算较短的任务，可重复测量并保留所有结果，避免偶然缓存或共享负载造成误判。

```
library(future)
plan(sequential)
start <- Sys.time()
result_serial <- run_one_sample(sample_id = "S01")
elapsed_serial <- difftime(Sys.time(), start, units = "secs")
print(elapsed_serial)
# run_one_sample■■■■■■■■■■■■■■■■
```

逐级增加worker

在确认单任务有效后，再用multisession开两个worker试跑同样的任务清单。multisession在独立R会话中运行worker，worker会有启动开销，也可能导出或复制对象。不要一开始就设置成CPU核数；先观察global导出量、并发内存和I/O，再决定是否增加。

```
plan(multisession, workers = 2)
# ■■■■■■■■■■■■■■■■■run_one_sample
# ■■■■worker■■■■■■■■■■■■■■
# ■■■■■■■■■■■■■■■■worker
```

对比指标	串行	并行
有效输出数	是否与输入样本数一致	是否无缺失或重复
总墙钟	从启动到全部完成	包括worker启动与等待
内存峰值	R进程/作业口径	明确是否统计所有worker
失败重跑	记录样本和原因	只重跑失败样本并隔离目录

解释结果并控制失败影响

如果并行总时间变短但峰值超出额度，应减少worker或分批执行；若时间没有明显改善，可能受磁盘I/O、任务不均衡或worker启动影响。GNU Time主进程的最大RSS不一定包含所有worker，优先使用调度器或容器的作业组统计。检查每个样本独立日志，不能仅以主R进程正常退出作为全批成功。

把future plan、worker数量、globals阈值、输入清单、软件版本、日志路径和输出验收写进项目记录。更换数据规模、存储位置、并行后端或同机负载后重新试跑。并行方案只有在结果正确、资源可承受且重跑路径清楚时才适合扩展到正式批次。

用记录表判断并行是否值得

每轮只改worker数，至少保存样本级退出状态、开始结束时间、主进程及作业组内存口径、CPU时间、I/O等待和输出校验结果。并行耗时要按“全部任务从启动到最后完成”计算，不是把各worker用时求平均。并行任务若有快慢差异，最后一个慢任务会决定总墙钟时间；因此多次复跑应保持同一批样本和任务分配规则。

若同一配置重复运行时结果差异很大，先比较节点负载、文件缓存和共享盘读写，不要只保留最快的一次。只有并行结果在预定资源上限内稳定完成、结果与串行基线通过同一验收，且节省的时间对项目有意义，才采用并行配置。记录不出作业组峰值时，应写“未测聚合峰值”，不要用单个worker数据代替。

把并行失败恢复纳入计划

每个样本使用独立输出目录和日志，已成功的样本写入完成清单，失败项保留退出码与最后日志。重跑前先确认旧worker已退出、文件没有被占用，并只重启失败任务。结束并行试验后按所用future后端关闭worker，再检查R会话与调度作业是否仍有子进程，避免上一轮资源残留影响下一次测量。

配套练习包

本PDF对应的ZIP包含本文代码块、主题练习模板或合成数据、README和SHA256校验文件。请从同一教程页面下载练习包；先核对校验值，再在独立目录复现。

参考资料

future options · 来源资料

future multisession · 来源资料

R Memory documentation · 来源资料

操作建议：先在合成小样本上验证流程，再依据项目参考、标签合同、存储策略与获配资源验收真实数据。