检查checkpoint并演练GROMACS停止、转移与续算

完整操作教程 · 参考固定 · 往返恢复验收

GROMACS续算依赖完整模拟状态，不依赖SSH会话是否还在。到队列时限或租期前，先按安全时间窗口停止并检查checkpoint，再复制已停止任务的完整结果，演练续算。本篇限定同一体系、同一TPR、同版本兼容环境的中断恢复，不保证任意版本或任意修改参数都能无缝续接。

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。

GROMACS续算依赖完整模拟状态，不依赖SSH会话是否还在。到队列时限或租期前，先按安全时间窗口停止并检查checkpoint，再复制已停止任务的完整结果，演练续算。本篇限定同一体系、同一TPR、同版本兼容环境的中断恢复，不保证任意版本或任意修改参数都能无缝续接。

先明确原运行与停止方式

假设原任务使用-deffnm run，目录中有run.tpr及对应输出。启动时可用-cpt设置定期checkpoint间隔，用-maxh在墙钟时限前安排停止；数值根据真实队列额度和留存余量填写。不能等到强制断电才假定最后状态完整，也不反复发送强杀信号来代替正常停止。

文件	作用	迁移要求
run.tpr	体系、算法与原计划	保留原件及SHA
run.cpt / run_prev.cpt	坐标、速度和其他续算状态	检查内容，前一份可作为恢复证据
log/edr/轨迹等	按原名续写的输出	默认append需校验原输出
版本与编译信息	兼容性依据	记录原机与目标环境，不只写软件名

检查停止后的状态包

SOURCE填已停止、无其他进程写入的原任务绝对目录，以下以run前缀为例。先审阅原日志停止位置，确认cpt确由该TPR的任务产生；gmx dump能读取不等于它属于正确任务。只复制cpt而丢掉原输出，可能无法按原方式append。

```
set -euo pipefail
: "${SOURCE:?■■■■■■■■■■■■■■■}"
test -s "$SOURCE/run.tpr"
test -s "$SOURCE/run.cpt"
mkdir checkpoint-review-001
gmx --version > checkpoint-review-001/target-version.txt 2>&1
gmx dump -cp "$SOURCE/run.cpt" > checkpoint-review-001/checkpoint.txt 2>&1
gmx dump -s "$SOURCE/run.tpr" > checkpoint-review-001/tpr.txt 2>&1
(
  cd "$SOURCE"
  find . -type f -print0 | LC_ALL=C sort -z | xargs -0 -r sha256sum
) > checkpoint-review-001/source-files.sha256
test -s checkpoint-review-001/source-files.sha256
mkdir continuation-001
cp -a "$SOURCE/." continuation-001/
(
  cd continuation-001
  sha256sum -c ../checkpoint-review-001/source-files.sha256
)
```

在副本中演练同一任务续算

先比较原日志和目标环境的GROMACS版本、编译与设备兼容性，再运行。MAXH填本次允许的小时数并提前留出退出和复制时间，-cpt 5表示周期分钟数教学值。原始任务必须尚有计划步数可继续；若已达到TPR终点，应另用convert-tpr按研究计划延长，不把零进展的正常退出误判为续算成功。

```
set -euo pipefail
: "${MAXH:?████████████████████}"
cd continuation-001
test -s run.tpr
test -s run.cpt
test ! -e continuation-001.time
test ! -e continuation-001.console.log
test ! -e continuation-001.checkpoint.txt
/usr/bin/time -v -o continuation-001.time gmx mdrun -s run.tpr \
  -deffnm run -cpi run.cpt -cpt 5 -maxh "$MAXH" \
  > continuation-001.console.log 2>&1
test -s run.cpt
gmx dump -cp run.cpt > continuation-001.checkpoint.txt 2>&1
```

用步数和文件连续性验收

比较续算前后checkpoint中的step/time及日志起点，确认接着原状态前进，并检查能量、轨迹时间和原子数。默认append会根据checkpoint内的校验信息检查原输出，不能为解决校验失败而直接编辑日志或改名。查清缺失或变化文件，恢复正确原件后再尝试。

确需-noappend时会产生带part编号的新输出段，必须记录并在后续分析中核对时间、重叠帧和顺序；它不是跳过checkpoint科学兼容性的开关。恢复原任务时通常不重新gen-vel，改温度、拓扑或积分设置属于另一个受控重启问题，应依官方grompp/continuation规则处理。

cp -a会保留符号链接，迁移前盘点外部依赖和链接目标，确保它们也被冻结且在目标环境可访问；这份目录副本不自动等于独立归档。跨机器传输同样先冻结写入，再按相对清单校验内容，完成之前保留原任务。

最终记录原末步、恢复起步、新末步、使用checkpoint、运行版本和各部分路径。只要这些证据尚不一致，就保留为待复核；命令退出零、进程重新出现或轨迹文件变大，都不能单独证明续算状态正确。先用复制任务演练，再把同一流程用于真实时限交接。

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

GROMACS长期模拟、append与checkpoint · 来源资料

mdrun checkpoint与墙钟接口 · 来源资料

读取checkpoint的gmx dump · 来源资料

GROMACS 官方 mdrun checkpoint 与恢复说明 · 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。