用构象清单安排Rosetta对接、随机任务与结果汇总

完整操作教程 · 参考固定 · 往返恢复验收

Rosetta对接多个起始构象，应同时保留“起始构象”和“随机重复”两层身份。两个结构各跑五个decoy，与同一结构改变五次种子是不同的采样设计。本篇限定已经prepack、两组伙伴位置接近候选界面的蛋白—蛋白局部对接，按独立任务运行标准docking_protocol。

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。

Rosetta对接多个起始构象，应同时保留“起始构象”和“随机重复”两层身份。两个结构各跑五个decoy，与同一结构改变五次种子是不同的采样设计。本篇限定已经prepack、两组伙伴位置接近候选界面的蛋白—蛋白局部对接，按独立任务运行标准docking_protocol。

先验收输入构象与伙伴定义

使用条件、二进制和数据库版本先确认，环境已通过该版本官方示例。PDB中的链ID、缺失残基、非标准残基和伙伴相对位置应经过处理。官方要求起始结构prepack，以准备界面外侧链；不能跳过后直接将分数与另一套准备流程比较。本例不是膜蛋白、抗体专用或配体对接协议。

字段	本例含义	错误风险
job	唯一构象—重复任务ID	不同随机任务共用输出
pdb	已prepack的绝对PDB路径	原始与处理后结构混用
partners	如A_B或LH_A	定义运动的伙伴组，非任意链列表
seed / nstruct	正整数种子 / 5个decoy	随机状态与实际数量都需保存

按清单顺序运行完整任务

dock-jobs.tsv无表头，四列为job、PDB绝对路径、partners和seed。DOCK填本版docking_protocol可执行文件绝对路径，ROSETTA_DB填匹配数据库。下面固定-dock_pert 3 8为教学局部扰动，未加randomize1/2；改变扰动和全局随机化会改变搜索问题，不作为纯资源开关。

```
set -euo pipefail
: "${DOCK:?■■docking_protocol■■■■■■■■■■}"
: "${ROSETTA_DB:?■■■■■■■■■■■■}"
test -x "$DOCK"
test -d "$ROSETTA_DB"
mkdir rosetta-dock-001
root="$PWD/rosetta-dock-001"
sha256sum "$DOCK" > "$root/binary.sha256"
completed=0
while IFS=$'\t' read -r job pdb partners seed || [[ -n "$job$pdb$partners$seed" ]]; do
  [[ "$job" =~ ^[A-Za-z0-9_-]+$ ]]
  [[ "$partners" =~ ^[A-Za-z0-9]+_[A-Za-z0-9]+$ ]]
  [[ "$seed" =~ ^[1-9][0-9]*$ ]]
  [[ "$pdb" == /* ]]
  test -s "$pdb"
  out="$root/$job"
  mkdir "$out"
  sha256sum "$pdb" > "$out/input.sha256"
  (
    cd "$out"
    /usr/bin/time -v -o run.time "$DOCK" -database "$ROSETTA_DB" -s "$pdb" \
      -partners "$partners" -dock_pert 3 8 -ex1 -ex2aro -nstruct 5 \
      -run:constant_seed -run:jran "$seed" -out:file:scorefile score.sc \
      > run.log 2>&1
    test -s score.sc
    shopt -s nullglob
    models=(*.pdb)
    test "${#models[@]}" -eq 5
```

```
  )
  completed=$((completed+1))
done < dock-jobs.tsv
test "$completed" -gt 0
```

按列名读取分数，不猜固定列号

保存为check_scores.py，以某个任务目录作参数执行。检查器只核对一份标准scorefile的记录、数值与PDB对应；不将最低总分自动宣布为真实结合构象。若本版额外输出或格式改变，先解释差异再修改验收协议。

```
import sys,math
from pathlib import Path
d=Path(sys.argv[1]);header=None;rows=[]
for line in (d/"score.sc").read_text().splitlines():
    fields=line.split()
    if not fields or fields[0]!="SCORE:":continue
    if fields[-1]=="description":
        if header is None:header=fields[1:]
        else:assert header==fields[1:]
    else:
        assert header is not None and len(fields[1:])==len(header)
        rows.append(dict(zip(header,fields[1:])))
assert len(rows)==5 and len({r["description"] for r in rows})==5
for r in rows:
    assert (d/(r["description"]+".pdb")).is_file()
    assert all(math.isfinite(float(v)) for k,v in r.items() if k!="description")
print("DECOY_FILES_AND_SCORES_CHECKED",len(rows))
```

比较采样与结果，而不是只取最小值

汇总表写入起始构象SHA、伙伴定义、种子、decoy编号和分数列。按适用协议同时评价界面、碰撞、构象聚类与独立证据；总分、界面分数和实验结合能力不能互相替代。不同输入准备或分数函数下的数值也不能直接排序成同一榜单。

同一种子重试应保留原任务身份，并另开attempt目录；新增独立采样使用新种子。固定种子只帮助复查该软件与配置，不保证跨编译版本逐位一致。抽查若干高低分构象，核对伙伴链和界面方向，避免因链定义错误得到看似完整的分数文件。

本例先顺序运行，每次标准调用不等于会自动使用整机多核。根据实际编译和日志测单任务，再由调度器增加外层任务并发，合计输入载入、RSS与落盘需求。失败保留日志，缺PDB或记录不足标记失败；不要把先结束的少量decoy当作全批完成，也不使用旧scorefile补齐计数。

任务表中还应保存prepack使用的协议与结构处理版本，同一构象的不同随机重复分别占一行。恢复失败前确认scorefile和PDB都来自该任务；若仅完成部分decoy，另开重试目录并写明原任务的完成数，不把两个目录无记录拼成一份已完成结果。每个decoy仍保留原任务来源。

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

RosettaDock官方输入、prepack与参数 · 来源资料

Rosetta当前全参数定义 · 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。