区分Java堆限制与物理内存，并测量MSFragger搜索峰值

完整操作教程 · 参考固定 · 往返恢复验收

MSFragger搜索报内存不足，先区分Java堆耗尽、操作系统杀进程以及搜索空间过大。把-Xmx写成服务器总RAM，可能挤占非堆、本地库及其他任务。本篇固定MSFragger4.4.1、Java11及以上的独

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。

MSFragger搜索报内存不足，先区分Java堆耗尽、操作系统杀进程以及搜索空间过大。把-Xmx写成服务器总RAM，可能挤占非堆、本地库及其他任务。本篇固定MSFragger4.4.1、Java11及以上的独立DDA mzML入口，记录固定参数下的完整搜索峰值。

先冻结搜索空间和软件条件

准备已验证的fragger.params、目标加诱饵FASTA及一份代表mzML。酶切、错切、肽段范围、修饰组合、质量窗口和搜索模式都会影响候选数量，不能只按FASTA文件MB数选择RAM。软件及使用范围按本项目适用许可确认；服务器租赁不自动提供MSFragger商业使用授权。

现象	先查证据	处理方向
Java heap space	JVM堆与GC日志	合理堆额度和候选规模
Killed或退出137	作业/系统OOM记录	进程总RAM及并发
Cannot allocate/ native错误	非堆、运行库、容器限额	不能只看-Xmx
运行慢但RAM够	索引、CPU、I/O与搜索模式	固定方法后做基准

检查参数所指数据库和输出

将下段保存为check_params.py。本例明确num_threads=2、output_format=pepXML、output_file_extension=pepXML，并使用数据库绝对路径；这些只固定试跑口径，正式线程可按获配资源调整。参数模板必须来自相同版本，其余搜索参数已经过方法验证，不在脚本里自动删修饰或改酶切。

```
from pathlib import Path
params={}
for line in Path("fragger.params").read_text().splitlines():
    line=line.split("#",1)[0].strip()
    if not line:continue
    key,value=line.split("=",1);key=key.strip()
    assert key not in params;params[key]=value.strip()
assert params["num_threads"]=="2"
assert params["output_format"]=="pepXML" and params["output_file_extension"]=="pepXML"
db=Path(params["database_name"])
assert db.is_absolute() and db.is_file() and db.stat().st_size>0
print("database",db,"bytes",db.stat().st_size)
print("threads",params["num_threads"],"mode must be checked from full parameter file")
```

给堆与总内存分别留证据

FRAGGER_JAR填写4.4.1整套发行中的绝对路径，Java运行库已验证。mzML复制到新工作目录，避免向原始数据目录写搜索产物，复制副本也要计入空间。-Xmx16g只是教学堆设置，必须小于本任务可用RAM并保留实际所需余量，不是推荐所有数据库都用同一堆。

```
set -euo pipefail
: "${FRAGGER_JAR:?■■■■■■MSFragger JAR■■■■■}"
```

```
[[ "$FRAGGER_JAR" = /* ]] || exit 2
test -s "$FRAGGER_JAR"
test -s /data/S1.mzML
python3 check_params.py
mkdir fragger-probe-001
cp fragger.params /data/S1.mzML fragger-probe-001/
cd fragger-probe-001
java -version > java-version.txt 2>&1
sha256sum "$FRAGGER_JAR" fragger.params S1.mzML > inputs.sha256
/usr/bin/time -v -o resources.txt java -Xmx16g \
  '-Xlog:gc*:file=gc.log:time,level,tags' -jar "$FRAGGER_JAR" \
  fragger.params S1.mzML > search.log 2>&1
test -s S1.pepXML
```

读搜索日志及实际输出

检查启动报告的版本、参数、数据库、肽段与fragment index规模，以及处理谱图数、终止状态和输出。pepXML应能由后续兼容工具读取并对应输入谱图；文件存在不表示鉴定通过FDR。统计建模、目标诱饵过滤和蛋白推断另按完整流程执行，不把原始搜索候选直接当可靠蛋白清单。

堆异常结合GC日志观察是否频繁回收仍无法容纳必要索引；退出137还需核调度器和容器内存事件，GNU Time不能独自解释所有结束原因。提高堆前确认主机可用RAM及其他任务，不要同时扩大堆和并发，导致总占用超过获配限额。

优化方法必须保留科学含义

先去除不合理的重复/无关数据库条目，核注释版本与目标诱饵来源；必要时按研究问题重新论证候选空间。减少修饰、错切或改搜索模式会改变可检出的结果范围，应作为方法变更记录，不能不说明就称为同任务内存优化。分库搜索也可能影响统计与合并规则，需另行验证。

多份MS/MS文件在同次兼容批量搜索中可复用fragment index，官方建议优先评估顺序处理而不是启动很多独立JVM。先测固定库与参数的完整代表任务，再扩批记录有效吞吐、堆、RSS和共存盘。确实存在必要物理内存缺口时再选更大RAM；仅CPU或I/O不足应按对应瓶颈调整。

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

MSFragger4.4.1当前变更和Java要求 · 来源资料

官方JVM启动及索引复用说明 · 来源资料

官方线程、数据库与输出参数 · 来源资料

MSFragger官方使用条件 · 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。