# 适配指导

## 1. 运行测试数据

1.1 在phasesubmit/config.json中predictdirjsonpath字段指定输入的json文件。“spath”字段指定生成的中间文件路径，默认为null不生成。这个主要是根据中间文件来进行调试。

1.2 设置spath字段来调试，并且设置“isExistFaultFlag”字段为True，使用默认参数运行phasesubmit/runTestCase.py, 这个文件将会读取同一目录下的config.json的信息，然后进行将中间文件和检测结果存储到spath中。

1.3 由于使用默认参数，效果可能不好，需要修改一些参数，（这些参数应该是可以全自动实现的，只要给定相同应用的一组算例数据，目前只能手动修改了），修改参数的建议在下面的指导中。

## 2. 内存泄漏异常

如果内存泄露的异常效果不好，可能是设置的每分钟内存泄露的强度过大，可以将config.json中的“memleakpermin”，默认是大于等于700M的都会被检测到。

## 3. 内存带宽适配指导

主要问题点在于

1. pgfree的平均值问题

2. mflops的平均值问题

3. 内存带宽异常影响该指标的幅度

4. 补偿pgfree时可以只对内存带宽变化的幅度进行补偿，这也是一个阈值

在效果不好的情况下，进行如下步骤：

1.判断mflops确定的正常稳定值是否符合预期。

使用可视化工具，查看spath/abnormalInfo/memory\_bandwidth50/pgfree.csv文件查看两个指标 mflops和mflops\_mean，如下图：判断平均值是否选择恰当。

图表

中度可信度描述已自动生成

如果平均值选择恰当，那么进行下一步判断，否则需要在config.json中文件中设置normalpath路径，会自动从正常文件中解析mflops的稳定值。目前这个功能还在抓紧实现中，手工做法是在输入文件中，即config.json中predictdirjsonpath对应的文件中RequestData/normalDataMean/topdown 中设置mflops的正常稳定值。

2. 判断是pgfree的正常稳定值确定的是否正常。

使用可视化工具，查看spath/abnormalInfo/memory\_bandwidth50/pgfree.csv文件

查看两个指标 pgfree\_smooth和pgfree\_mean，如下图：判断平均值是否选择恰当。

图表

描述已自动生成

如果平均值选择恰当，那么进行下一步判断，否则需要在config.json中文件中设置normalpath:null路径，会自动从正常文件中解析pgfree的稳定值。目前这个功能还在抓紧实现中，手工做法是在输入文件中，即config.json中predictdirjsonpath对应的文件中RequestData/normalDataMean/server 中设置的pgfree的正常稳定值。

3. 内存带宽异常影响该指标的幅度大小

该指标重点在与内存带宽泄露时pgfree的变化幅度，由于不同HPC应用可能导致的变化幅度是不一样的，从wrf和grapes数据中可以发现，同一应用不同算例情况下，相同异常的变化幅度是一样的。此变化幅度可以由同一应用下的不同算例（训练数据）得到。目前需要手动指定，确定数值方式如下：

使用可视化工具，查看spath/abnormalInfo/memory\_bandwidth50/pgfree.csv文件查看两个指标 pgfree\_change\_mflops和pgfree\_mean，查看内存带宽异常时pgfree\_change\_mflops的数值高于pgfree\_mean的数值大小，如下图：

日程表

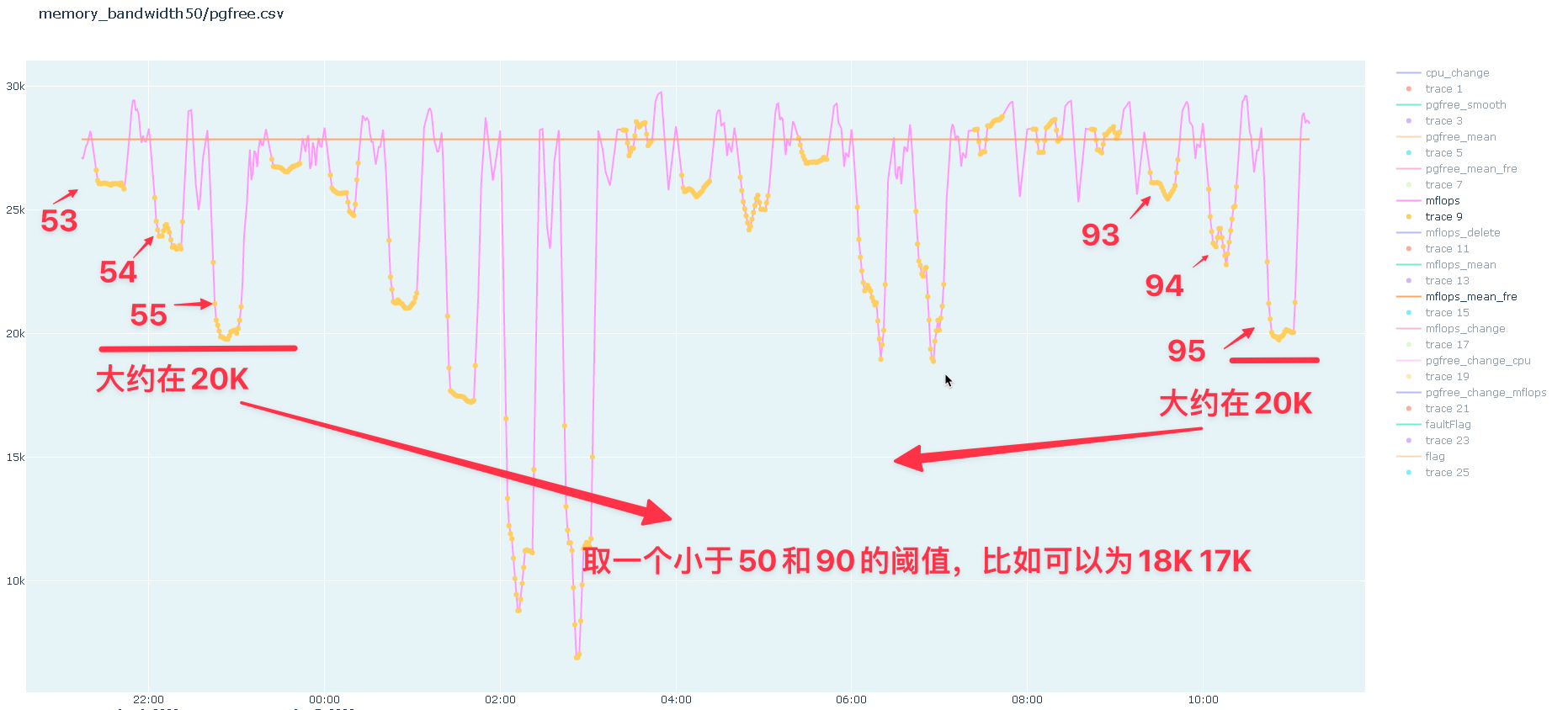
中度可信度描述已自动生成

具体设置config.json中pgfree\_thread字段为2000000

4. 补偿pgfree时可以只对内存带宽变化的幅度进行补偿

补偿的主要目的是使得其他内存带宽异常的pgfree指标凸显出来，依据mflops进行补偿，可以对mflops补偿选择一个阈值。

使用可视化工具，查看spath/abnormalInfo/memory\_bandwidth50/pgfree.csv文件查看两个指标mflops和mflops\_mean，如下图：



具体设置方式为config.json maxflopsinio字段为18000。这个阈值目前为手工指定，还在完善代码，由训练数据得到。

## 4. cache异常适配指导

主要问题点在于

1. mflops的平均值以及读写的平均值问题

2. cache抢占异常影响该topdown读写指标的幅度

4. 补偿读写指标时可以只对cache抢占这个范围进行补偿 – 已经在3.3中解决。

**1. mflops的平均值问题**

和3.1中的方式一样，观察mflops的正常稳定值是否取得正确。否则修改

**2. topdown文件中ddrc\_wr指标平均值是否正常**

使用可视化工具，查看spath/abnormalInfo/cacheGrab/ debug90.csv文件查看两个指标 ddrc\_wr和ddrc\_wr\_means，如下图：判断平均值是否选择恰当。

图表, 折线图

描述已自动生成

如果平均值自动识别的不正确，手工做法是在输入文件中，即config.json中predictdirjsonpath对应的文件中RequestData/normalDataMean/topdown中设置的ddrc\_wr的正常稳定值。

**3. topdown文件中ddrc\_rd指标平均值是否正常**

使用可视化工具，查看spath/abnormalInfo/cacheGrab/ debug90.csv文件查看两个指标 ddrc\_rd和ddrc\_rd\_means，如下图：判断平均值是否选择恰当。

图表

描述已自动生成

如果平均值自动识别的不正确，手工做法是在输入文件中，即config.json中predictdirjsonpath对应的文件中RequestData/normalDataMean/topdown中设置的ddrc\_rd的正常稳定值。

**4. cache抢占异常影响读写指标的幅度**

使用可视化工具，查看spath/abnormalInfo/cacheGrab/ debug90.csv文件

查看两个指标 ddrc\_ddwr\_sum和ddrc\_ddwr\_sum\_mean，如下图：判断平均值是否选择恰当。

图表

中度可信度描述已自动生成

具体设置方式为config.json ddrc\_ddwr\_sum\_max字段为6000。这个阈值目前为手工指定，还在完善代码，由训练数据得到。