Boost 攻关总结

1. 问题背景

项目中 odps 集群服务调用 folly 和 boost 库时报错 coredump，该问题直接影响 odps 联调，进一步阻塞 base、bcc、pai 等众多大数据产品线联调，导致无法按预期完成项目中 POC 测试。boost 报错扼住了项目进度的喉咙，导致后续工作无法开展，而项目验收时间迫在眉睫。

2. 问题根源

这次攻关的问题最终根源包含两方面：

1、boost 协程汇编实现过程中，由于 sw 架构底层操作差异性，导致保存和恢复的协程上下文信息丢失；

2、阿里 odps 联合编译时隐含的 Optimizing compilation 处理，导致 folly 功能代码缺失，指令优化异常。

3. 攻关难点

难点 1：报错信息少，查错只能基于汇编，无法反推上层代码逻辑

boost 报错表象单一，仅为 core dump 报错，无其它关键提示信息。定位出 odps 在唤起服务，调用 folly 和 boost 时产生的报错。调试相关源码未发现有用信息，只能从汇编角度跟踪分析。定位到报错指令后，发现并不是错误根源，报错指令只是错误的表现位置，需要往上追溯错误根源；但是结合上下文汇编代码反推上层源码逻辑难度较大。

难点 2：阿里 odps 集群服务逻辑复杂、依赖众多，无法直接联调

基于阿里 odps 应用，沿着代码逻辑依次往下分析过程中，由于 odps 属于大数据的集群应用，服务启动后会源源不断地产生新线程，进程会调用各种各样的第三方库，代码逻辑极其复杂；odps 正向联调工作量庞大，排查难度增加。

难点 3：模块解耦验证过程中，无全覆盖的测试用例

对 odps+folly+boost 整体进行拆分、单点验证，依次对每个模块进行代码分析，梳理代码逻辑并未发现不合理的之处。若想发现隐藏的细节问题，必须进行功能测试，此外要保证测试覆盖范围要广。然而 boost 和 folly 中，并没有这种符合要求、直接可用的测试用例，无相关参考文献，需要自行完成测试用例编写。

难点 4：解耦后的模块修改并验证无误后，整合到 odps 应用仍然报错

在对 boost 和 folly 模块功能完成正确性验证后，汇总到 odps 集群编译环境却报错。依次验证解耦后的 boost 和 folly 源码及编译规则，未有不同之处。对于 odps 服务中模块化编译参数引入和编译规则定义隐匿性较强的特点，容易导致关注度缺失。

4. 攻关突破

突破 1：编写 boost 测试用例，定位并修复协程上下文保存和恢复缺陷

针对 boost 源码，重点分析和 sw 架构强相关的两个关键汇编文件。由于单纯分析无法发现细节错误，继而参考 boost 源码，经过反复修改验证，最终编写出用来验证协程功能的全覆盖测试用例，同时避免了其他模块干扰。通过测试用

例发现了问题：由于 sw 和其他架构的底层操作差异性，保存和恢复的上下文信息不全，导致协程创建和切换时访问非法地址。随后参考其他架构，在草稿纸反复推导 sw 架构寄存器保存和恢复过程，终于修复了两个汇编文件中的多处错误。

突破 2：编写 folly 测试用例，通过分析、编译和调试，定位出 folly 库报错

boost 报错问题修复后，经过 odps 集群验证，又出现新的 coredump 报错，这次不再报错到 boost 里，而是报错到 folly 库。随后联合阿里经过几天夜以继日的攻关，编写出 folly 测试用例，通过测试用例准确触发了 folly coredump 报错场景。通过对 folly 源码进行分析、编译和调试，经过多次对比验证发现了 odps 联合编译的 folly 库和我们自行编译的 folly 库不同之处，从而导致报错。

突破 3：模拟 odps 集群环境，通过反汇编对比分析，定位并修复剩余错误

基于前期攻关成果，剩余的错误发生在 odps 联合编译的 folly 库里。继而从 odps 里将 folly 模块进行剥离，对该模块进行单点验证，模拟 odps 依赖环境。将 folly 源码和编译规则成功剥离后，便趁着五一假期加班加点验证起来。验证发现，剥离后的 folly 和测试用例均能通过编译。最后，针对联合编译和剥离编译的 folly 库分别进行反汇编分析，发现了 FiberManager 接口里一些被异常优化后的指令代码，定位出 GCC Optimizing compilation 的问题，是 odps 从其他模块隐含传递的优化选项并进行了强制设定。

5. 中途存在的问题

在攻关过程中，因疫情影响，只能在酒店采用远程调试；阿里相关技术人员、

项目负责人等分布在上海、杭州、北京多个城市，沟通只能通过线上电话会议进行，沟通难度较大，沟通等待时间较长；阿里调试环境复杂，众多产品在环境中相互交叉，调试环境节点复杂，测试文件传输和迭代验证需要完全依靠阿里完成；周末和节假日期间阿里对应响应迟缓等问题都或多或少影响攻关进度。

6. 攻关建议和感想

从本次攻关中发现部分不足之处，对于后续类似相关查错问题应汲取教训和吸收相关经验，遂提出以下几点建议：

1、前期服务器环境准备工作应该优化，查错环境整洁易操作十分有必要；

2、软件自测测试用例和异地复现问题的测试用例存在的必要性需要提高重视；

3、多点并发处理问题的有效性和时效性也至关重要。需要摒弃单从问题表象介入排查问题的单一性思想，不应等到表象问题处理后问题未解决才确立新的入手点。

4、对待临近关键时间节点的问题应提高重视度，不要因为假期原因影响沟通问题和处理问题的时效性。

回顾起来，在攻关的二十九天里虽然每天都在忙碌，也没有五一假期和周末时间，全身心投入到工作中，同时让人收获满满、倍感充实。经过这么多天查错分析，感觉到了自身技术能力的提升，相信更利于接下来的工作开展，为申威生态建设添砖加瓦！