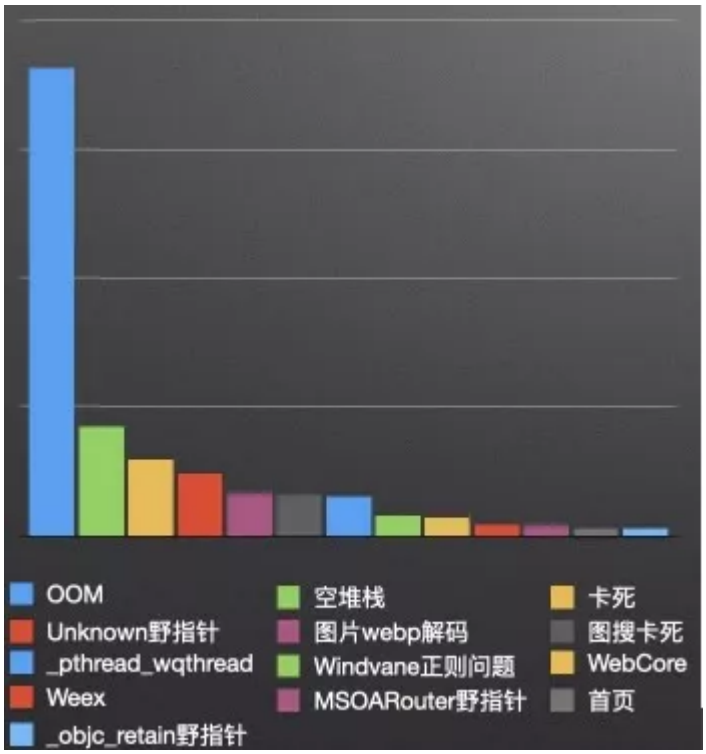
阿里巴巴淘系技术 Lv3
2020年08月06日 阅读 3679

关注

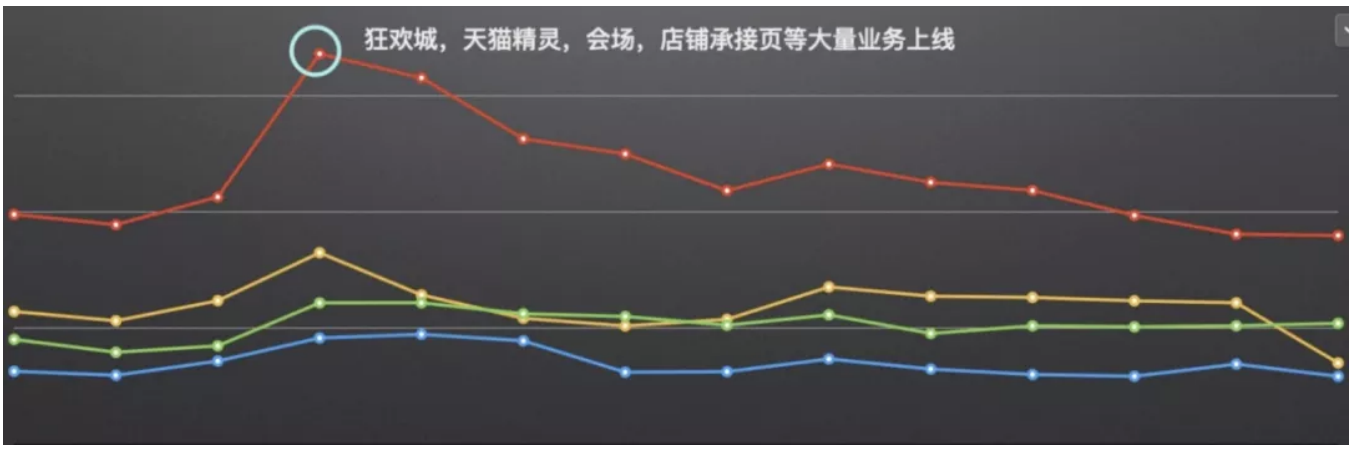
客户端稳定性优化实战，Crash率最高下降40%

大促一直是技术和产品的练兵场，每到大促，各种丰富的富媒体，如直播，视频，3D，游戏互动，AR等竞相上线，在淘宝的大航母战略下，都集中在万千宠爱于一身的淘宝App上，在这样的大促场景下，开始触碰到端侧系统资源上限的天花板。在17年双11大促期间，端侧的内存问题尤为突显，OOM的高居所有问题的榜首。内存问题成为了这几年大促端侧稳定性最大的挑战。

17年双11 Crash问题分类



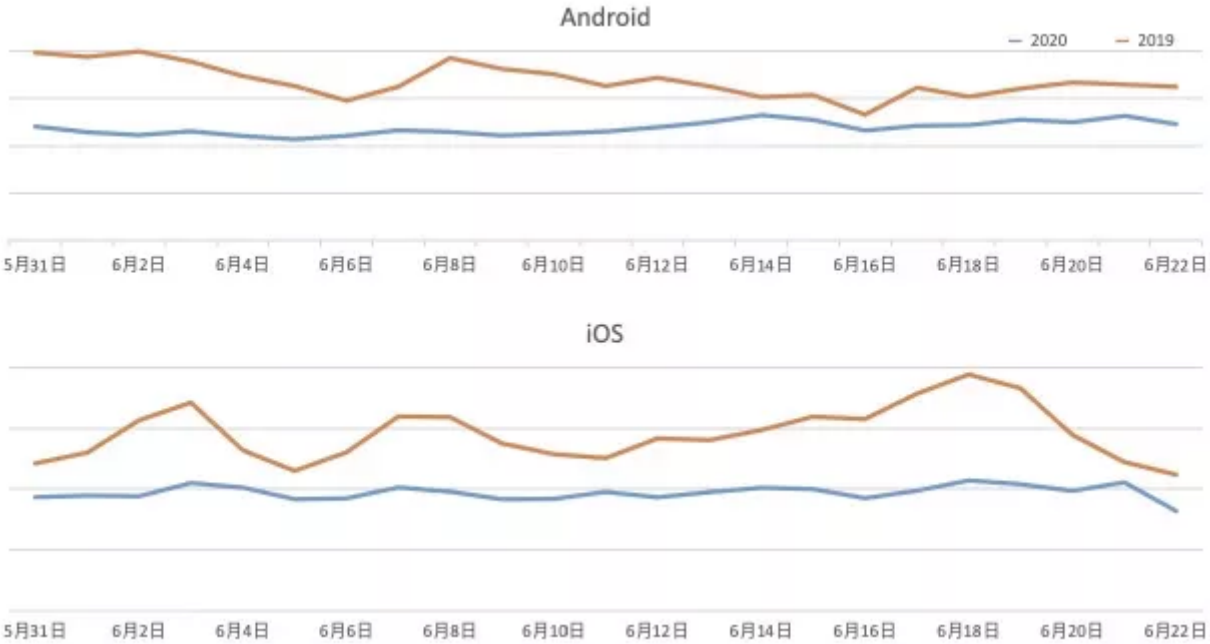
17年双11 Crash走势与业务上线关系



放飞业务

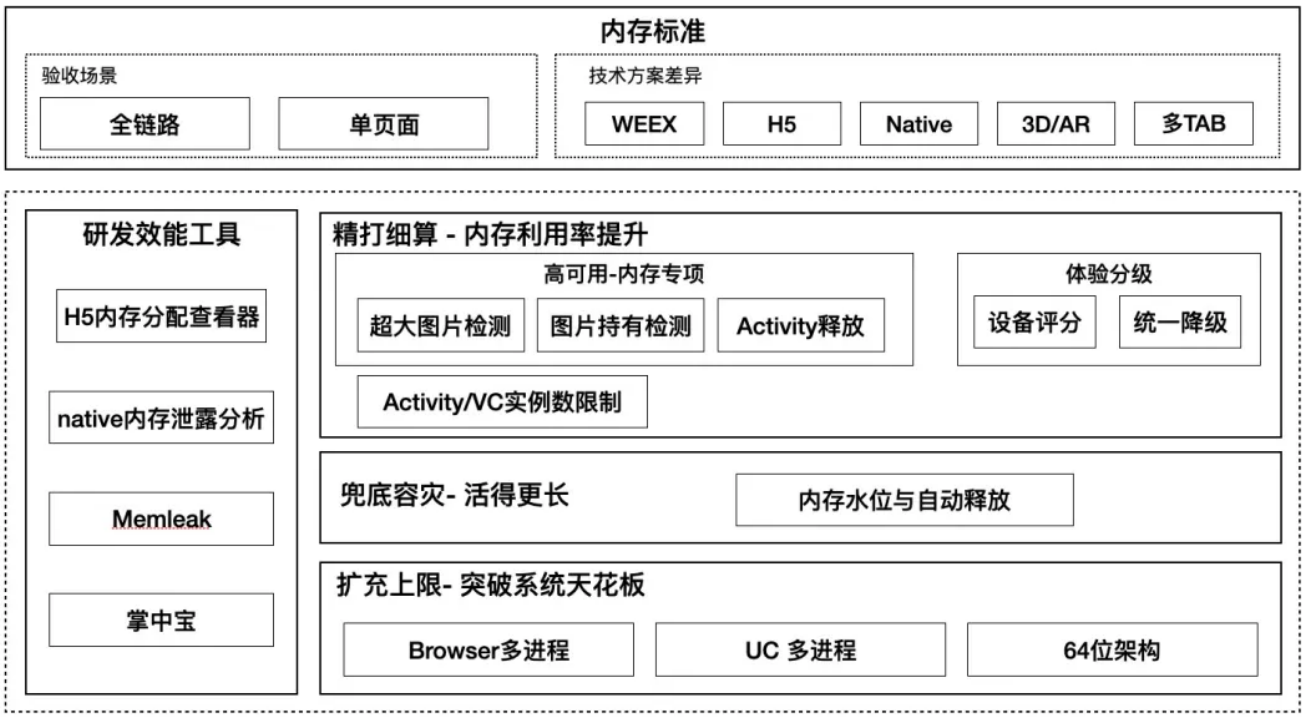
两年后的今天，通过我们持续的技术挖掘与治理，内存问题不再是影响大促稳定性的最主要的因素，本次618大促前所未有的支持了猜你喜欢无限坑位，支持了丰富的直播和小视频玩法，支持了会场上百个运营坑位，支持了互动业务的不降级策略，各种业务花式上线的同时，我们的端侧稳定性还进一步提升，crash率远好于去年同期。

618期间今年与去年对比crash走势



迎难而上，各显神通

面对内存带来的挑战，我们2年以来，一直在摸索中前行，沉淀一套内存治理的经验。



面向大促，当出现了问题后，我们要去思考当前的机制与规范，于是我们制定了内存标准与业务的上线验收。同时，提供了内存分析的一套工具，方便很快很准找到问题。同时，我们制定了三套内存化策略：

2. 兜底容灾，尽量让应用延长生命
3. 提升内存上限，突破系统的天花板

验收标准-----一夫当关

由于内存天花板的存在，从稳定性角度综合考虑，引入了大促的验收标准。标准的制定过程中，我们统计了发生OOM时的水位内位，分析出了高危，危险，正常水位线，以此为内存标准的制定指引。

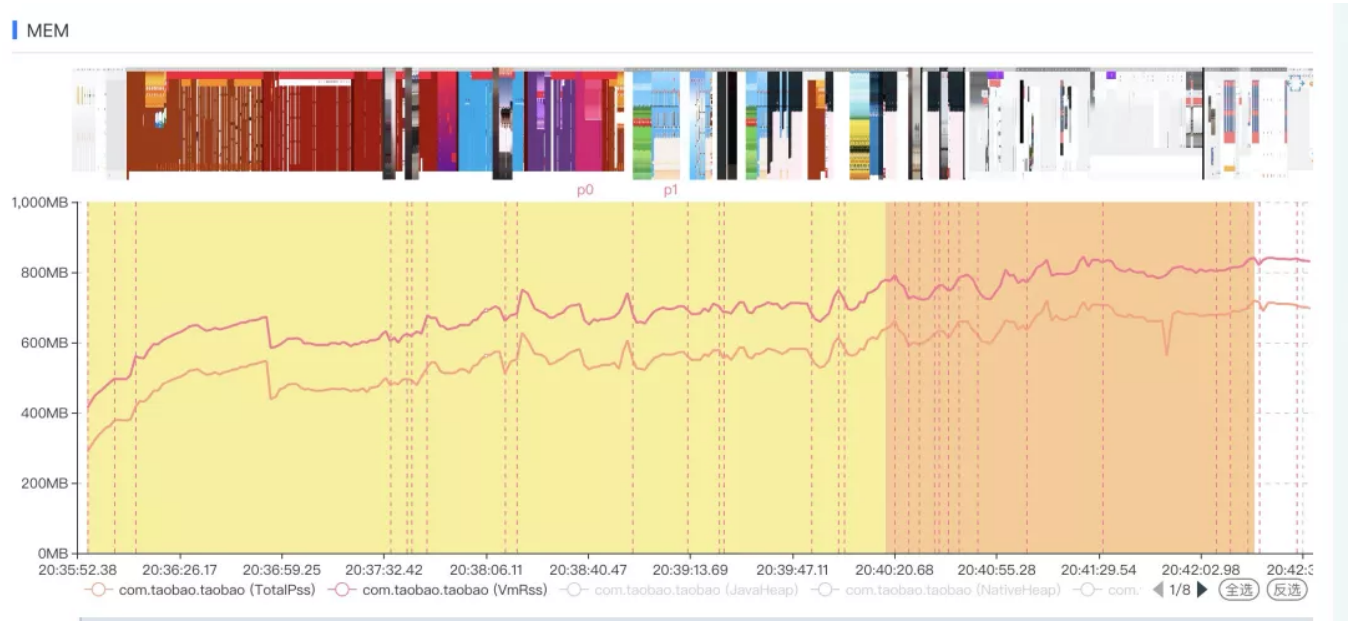
内存问题之所以复杂，是因为内存是一个全局共享池子，当出现溢出问题时，在没有明显问题时，很难去界定哪个业务存在问题，因此，在考虑标准的时候，我们定义了两种场景。单页面及链路。

单页面场景主要是为了减少单个业务过多的占用内存引发的风险。前面提到内存池子是全局且有限的，当单页面占据内存过多，就会导致系统整体可用的内存大幅减少，在浏览相同页面次数的情况，增加整体内存风险。

链路场景是对常见浏览链路的内存检测，比如从首页-会场-互动-店铺-详情-下单这样的常规玩法进行多页面叠加检测，判断用户正常场景下的内存风险。

同时，在制度内存标准时，也考虑了不同技术栈之间的差异。比如H5,weex,native，包括多tab的会场形式及直播，3D等。

测试同学研发的TMQ自动化测试工具



内存优化三板斧

精打细算-提升内存的利用率

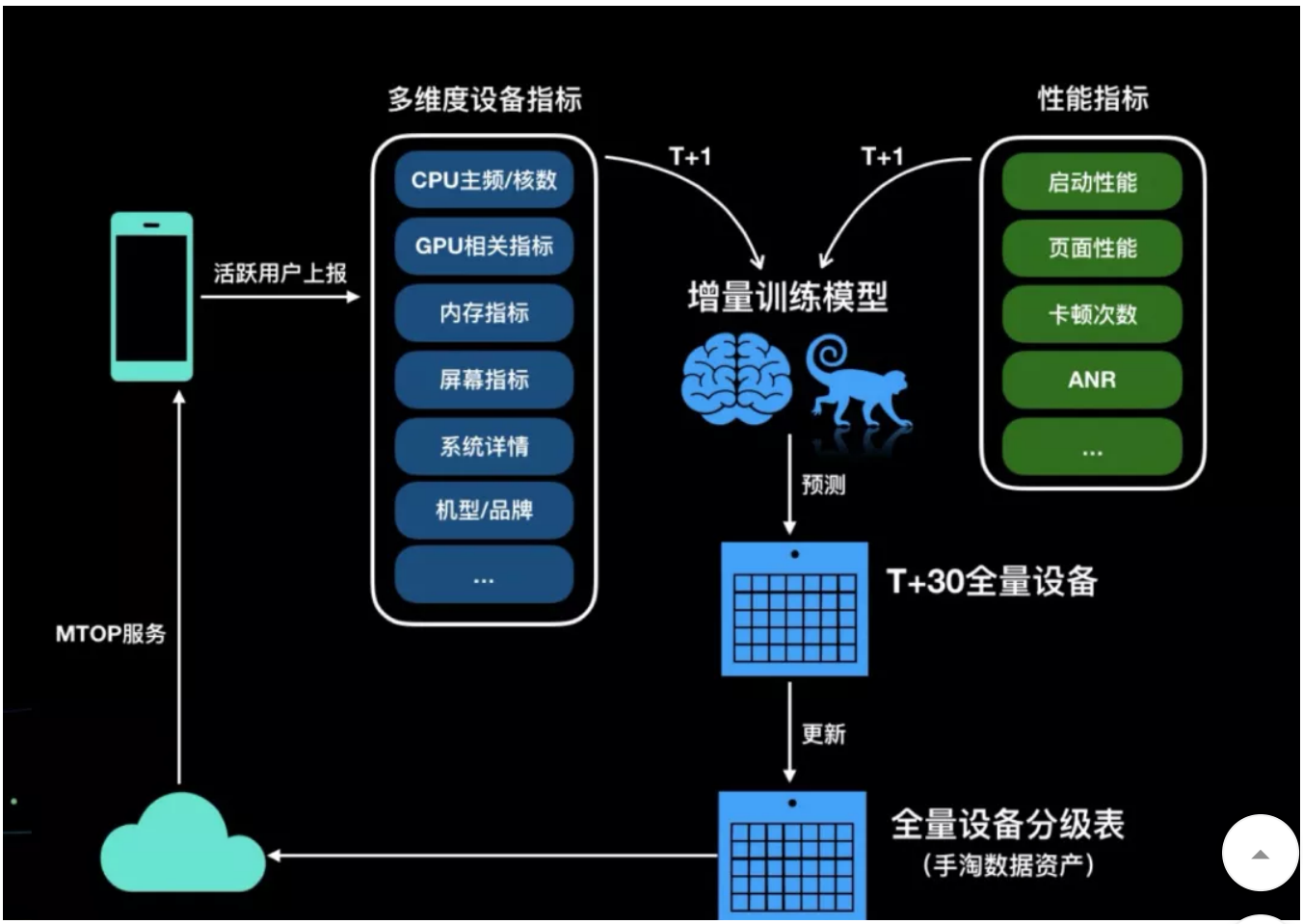
在业务屡屡触及内存天花板的情况下，每1KB的内存，都显得非常珍贵。

在实际对内存占用的分析中，我们偶尔会发现有些场景加载的图片远大于视图的大小，造成内存的浪费。或者在某些场景下，图片在内存中持有过久，比如在后台或是压栈很久后，图片所持有的空间仍不能释放出来给当前界面使用，面对这样的场景，我们在高可用体系中引用了对应的功能，能够检测出这些case，以便把内存交给用户正在使用的组件，以此来提升内存的利用率。

从图片库的数据流转以及View生命周期出发，来设计图片自动回收和恢复的实现，即当View不可见的时候，自动释放图片到图片库缓存，只保留图片的key值；当View可见的时候，又通过key恢复图片。图片片自带三级缓存策略，回收后的图片如果还在缓存，能立马恢复，对体验几乎无损。

同时，针对一些内存大户，也和各业务方约定一些实例数限制，比如详情页，有大图，还带视频，webview等，内存使用相对较大，这种情况下会对实例数做要求。目前有限制包括详情页，播放器实例等。

为了更好的体验，在降级策略上我们也做了一些优化，不再一刀切，而是根据各设备自身的能力，有选择的进行降级。要更好的达成目标，我们首先对设备进行分级，依赖于创建的智能分级。



统一降级在设备评分的基础上，提供默认的高中低端机型的设备分级能力，增加了配置化能力，为每个核心业务分配一个orange，支持业务对系统、品牌、机型、设备分、应用版本、生效时间等多个维度进行配置化降级。

依赖于统一降级，可以做到精准的体验分级，高端机型，可以采用各种特效和高清图片，能保障最优体验。中端机型，降级掉一部分特效，可以获得较好效果，低端机型，保障稳定性和基础体验。实现“高端设备最炫体验，低端设备流畅优先，紧急问题快速降级”

统一降级后的效果

兜底容灾--尽量延长生命周期

在应用内存最危险的时候，也许下一次的内存申请即面临崩溃，在这最危险的时候，我们是否有能力缓解一下，让用户多下一单呢，为此我们设计了内存容灾SDK。

具体原理是基于gc和lowmemorykiller原理实现（Android的OOM要区分jvm heap内存不足和native的内存不足），通过监听系统的gc和lowmemorykiller，去计算系统当前所处的内存状态，当内存不足的时候，销毁掉优先级较低的Activity，从而保障用户可见面Activity能尽可能多的使用内存而不出现稳定性问题。

内存容灾基本原理

扩充上限-突破系统天花板

内存容量，才是解决内存问题的最终解法。

多进程

多进程的使用是突破系统天花板的方法之一。由于大促态的变化新增以H5的页面居多，所以我们重点希望在webview上能有一些突破。这时苹果的WKWebivew被纳入到研究范围，关于WKWebview在内存的优势，经过我们的分析结论如下：

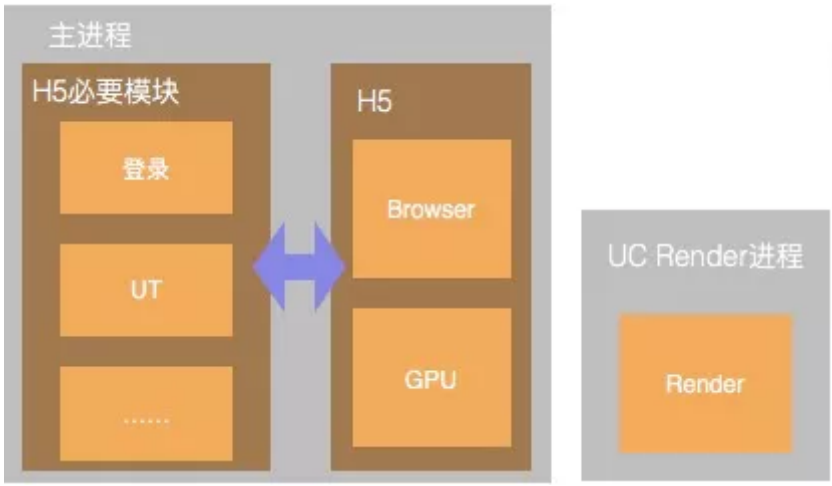
WKWebView的内存并不计算在主应用的内存中，而是作为独立进程单独进行计算，因此对于应用来说使用WKWebView相比UIWebView，应用整体可以使用更多的内存，因为Web的内存都在WKWewbView的Web进程中，并不影响主应用的内存上限。

对于Android来说，平台本身则支持的多进程方式，因此，我们最初的设计中，是依赖于Activity的独立进程方式，即使BrowserActivity独立出来。

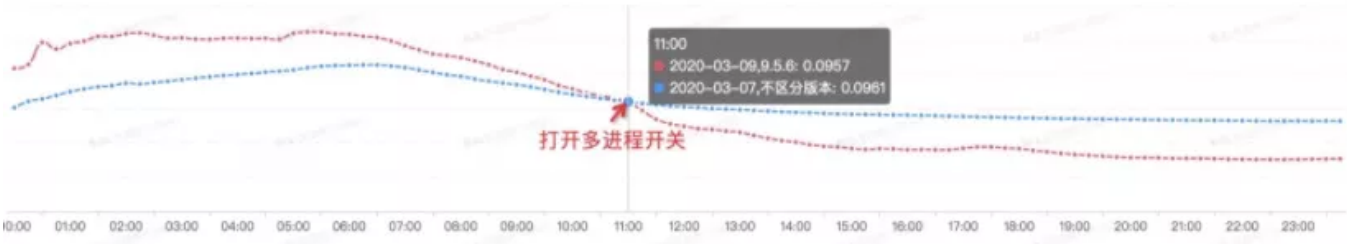
在99大促的AB实验中，对比对照组，在访问过淘金币/互动的用户中，主进程native crash率下降15%-18%，Crash计数(主+子) 下降1万次以上。在所有用户中，下降3%-5%，对内存的优化效果还是比较明显。

但是考虑到很多基础SDK在设计之初并没有考虑到多进程的方式，且多进程下应用的生命周期也有一些变化，整体方案不确认的风险较大。最终采用的是UC内核的多进程方案，它将整个H5页面的解析、排版、js执行等实现抽离封装到一个独立的进程中，分担了主进程一部分内存压力，从而实现突破单进程内存容量天花板的目标。

UC多进程示意图



uc多进程对crash率的影响



根据严格AB实验的结果评估，手淘开启UC多进程之后，Crash率能有30-40%的下降收益。

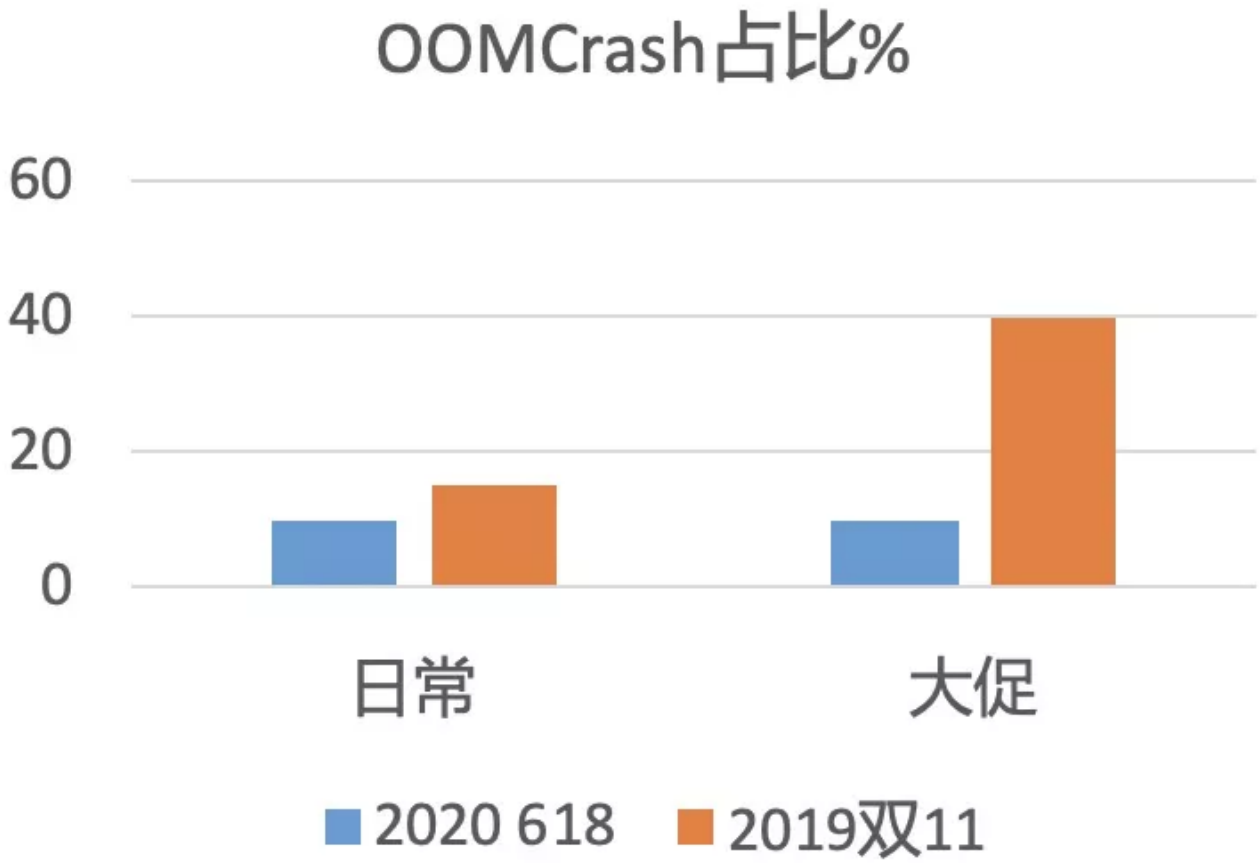
64位升级

一般说来，现在使用的程序，都是在32位的指令集下编译出来的结果，在32位子系统下，内存地址的大小只有4个字节，理论上的最大寻址空间只有4G。前面提到，在当前手淘的业务容量下，4G的内存地址已经不能满足，在今年开始力推手淘andorid架构从32位升级到64位。

说到64位，在硬件上，arm v8及以后的cpu都升级到了64位的架构，在软件上，android 5.0以后的系统开始支持了64位子系统。我们做过比较准确统计埋点，在市面上的手机，约有95%是支持64位运算的，也就是说64位升级带来的收益可以覆盖到绝大多数的用户。另一方面，我们也要看到64位升级带来的风险，所有的C/C++代码都需要重新编译到64位指令集，可能的风险点包括：

- 指针长度是从32位升到64位，对一些HardCode的写法可能计算出错，产生稳定性问题。
- 自定义so的加载逻辑（如服务端远程下载）可能没有考虑多cpu abi的情况，加载错误so，产生稳定性问题。
- 储存的数据，看看有没有因为64位和32位不同导致不兼容，特别是一些二进制数据，导致覆盖安装或升级后原数据不可用

为了应对这些风险，自3月份起就开始针对手淘中的120多个so进行回归，包括看32位与64位相互覆盖的升级场景，另一方面，针对so的加载逻辑，进行全手淘代码扫描，分析和查看自定义加载so的场景确认是否支持多cpuabi。经过几个月的灰度和迭代，最终在618版本前，完成了64位的上线。



在本次的618大促中，可以明显看到，以往大促中，OOM的占比，最高的时候，可以占到近40%，经过64位升级与多进程手段叠加后，可以看到看大促态的OOM占比，已经降到了10%左右。这里面还包括了5%的32位用户，对OOM的治理效果非常显著。

展望

- 新技术形态的挑战

内存问题一直是大促端侧稳定性最大的挑战，在今天已经得到比较好的解决，当然，系统资源终归是有限的，我们仍然需要有效合理的使用系统资源。更重要的是，面向未来，新的技术形态像flutter等入淘，多个虚似机的并存，对系统资源仍然会有较大的挑战。

- 从稳定性到流畅体验

对用户来说，稳定性只是最基础要求，后续我们会在体验上持续优化，带给手淘用户真正的如丝般顺滑的体验。

手淘客户端团队

一年一度的应届生秋招正式开始，岗位丰富，欢迎扫描下面二维码了解详情。同时大量移动端社招岗位opening 简历投放地址📮：difei.zdf@alibaba-inc.com

阿里巴巴2021届校招简历投递

> **关注下面的标签，发现更多相似文章**

iOS

砖厂小工 Lv1 程序员 @ 浦东砖厂

标题起的好，绩效少不了 👏👏👏

8天前 1 回复

free46000 Lv3 Android @ 去哪儿网

64位升级和解决oom是什么关系，没理解

9天前 1 回复

zlyBear Lv1 iOSer

宁可卡死也不能crash

11天前 1 回复

zjw-swun Lv2 Android退役选手

写的不错啊，改改名字就能给我老板汇报了，kpi到手

8天前 回复

limuyang2 Lv2 Android开发? iOS开发...

865都流畅不起来...

12天前 回复

九元道祖 划水的工程师 @ BUG司

我说我打开淘宝 完了手机就很卡，原来淘宝疯狂的请求内存

12天前 回复

OnClickListener110... Android Developer @ ...

太强大了

13天前 回复

methodInvoke 程序员 @ 无业游民

手淘内部各种跨平台框架比如奥创，阿里全家桶...这些占了多少内存..... 🐶

13天前 回复

相关推荐

橘子不酸丶 · 2天前 · iOS

iOS优化篇之App启动时间优化

32 1

老司机技术周报 · 1天前 · iOS / SwiftUI

【WWDC20】10027 - SwiftUI 中的 App 要领

首页 探索掘金 查询

FengyunSky · 4天前 · iOS

一文读懂iOS线程调用栈原理

12

ZenonHuang · 7天前 · iOS

iOS 的自动构建流程

61 7

JeremyHuang37 · 1天前 · iOS

【译】自定义 Collection View Layout -- 一个简单的模板

2

掘金酱 · 27天前 · iOS / Android / 前端 / 后端 / 程序员

🏆 掘金征文 | 2020与我的年中总结

94 109

Chouee · 3天前 · iOS

造轮子 - UITableView字母索引条

7 1

路过看风景 · 2天前 · iOS

CocoaPods原理 及 组件化

3

阿里巴巴淘系技术 · 5天前 · iOS

Apple Widget：下一个顶级流量入口?

9

路过看风景 · 1天前 · iOS

iOS事件处理 UIResponder

2