

## App 二进制文件重排已经被玩坏了

🕒 发表于 2019-09-01 By 杨萧玉

『二进制文件重排优化启动速度』本是一项上古 PC 时代就玩过的东东，前一阵子借助某宇宙大厂重新火了一把。不过令我惊讶的是：这么简单个事情竟然搞得如此复杂，而且还声称『开拓性的探索、在没有业界经验可供参考』。。。

说实话可能会得罪人，但是我怕过吗？我怂了，这段掐了。

其实二进制文件重排很简单啊，重点在于生成 order 文件。我基于 Clang SanitizerCoverage 和业界已有的经验，整了个 [AppOrderFiles](#)，一个调用搞定！Enjoy it！

```
1 AppOrderFiles(^NSString *orderFilePath) {
2     NSLog(@"OrderFilePath:%@", orderFilePath);
3 };
```

### 文章目录

1. 苹果官方文档的古老方案
2. Facebook 的 hfsort
3. 基于 Clang SanitizerCoverage 的方案
4. 后记

## 苹果官方文档的古老方案

苹果的官方文档很早就给了二进制文件重排的方案：[Improving Locality of Reference](#)，『早』到甚至被苹果提示这份文档已经年久失修，部分工具和链接失效了。文档的过时不仅体现在还是 GCC 时代，连工具链比如像 [gprof](#) 也不能用了，不过 Google 也给出了 macOS 上的替代品，有兴趣的可以去研究下。

## Facebook 的 hfsort

需要先用 hf-prod-collect.sh 收集数据，然后塞给 [hfsort](#) 生成 hotfuncs.txt 文件。很好很强大，不过对于编程小白来说有一定的使用成本。

PS：此方案来自于我写了这篇文章后，jmpews 大神丢给我了个链接，受益匪浅。（其实我啥都看不懂）

## 基于 Clang SanitizerCoverage 的方案

在 [Clang 10 documentation](#) 中可以看到 LLVM 官方对 SanitizerCoverage 的详细介绍，包含了示例代码。

简单来说 SanitizerCoverage 是 Clang 内置的一个代码覆盖工具。它把一系列以 `__sanitizer_cov_trace_pc_` 为前缀的函数调用插入到用户定义的函数里，借此实现了全局 AOP 的大杀器。其覆盖之广，包含 Swift/Objective-C/C/C++ 等语言，Method/Function/Block 全支持。

开启 SanitizerCoverage 的方法是：在 build settings 里的“Other C Flags”中添加 `-fsanitize-coverage=func,trace-pc-guard`。如果含有 Swift 代码的话，还需要在“Other Swift Flags”中加入 `-sanitize-coverage=func` 和 `-sanitize=undefined`。所有链接到 App 中的二进制都需要开启 SanitizerCoverage，这样才能完全覆盖到所有调用。

基于 Clang SanitizerCoverage 我写了个工具 AppOrderFiles。CocoaPods 接入，一行调用生成 Order File。啥也不说了，全在 GitHub 里了：<https://github.com/yulingtianxia/AppOrderFiles>

当然这也不完全是我的原创，对照着 Clang 文档的同时，还参考了 [Improving App Performance with Order Files](#) 这篇文章的代码。人家这篇文章虽然早就给出了，不过还是有一些 bug 和优化空间的。

原理就是在 SanitizerCoverage 的回调函数里将地址先收集到队列里，调用 `AppOrderFiles()` 后会停止收集，并将队列中的 PC 地址依次翻译符号，最后去重。反正代码也不多，直接贴核心代码：

```
1 static OSQueueHead queue = OS_ATOMIC_QUEUE_INIT;
2
3 static BOOL collectFinished = NO;
4
5 typedef struct {
6     void *pc;
7     void *next;
8 } PCNode;
9
10 // The guards are [start, stop).
11 // This function will be called at least once per DSO and may be called
12 // more than once with the same values of start/stop.
13 void __sanitizer_cov_trace_pc_guard_init(uint32_t *start,
14                                         uint32_t *stop) {
15     static uint32_t N; // Counter for the guards.
16     if (start == stop || *start) return; // Initialize only once.
17     printf("INIT: %p %p\n", start, stop);
18     for (uint32_t *x = start; x < stop; x++)
19         *x = ++N; // Guards should start from 1.
20 }
21
22 // This callback is inserted by the compiler on every edge in the
23 // control flow (some optimizations apply).
24 // Typically, the compiler will emit the code like this:
25 //   if(*guard)
26 //       __sanitizer_cov_trace_pc_guard(guard);
27 // But for large functions it will emit a simple call:
28 //   __sanitizer_cov_trace_pc_guard(guard);
29 void __sanitizer_cov_trace_pc_guard(uint32_t *guard) {
30     if (!*guard) return; // Duplicate the guard check.
31     if (collectFinished) {
32         return;
33     }
34     // If you set *guard to 0 this code will not be called again for this edge.
35     // Now you can get the PC and do whatever you want:
36     //   store it somewhere or symbolize it and print right away.
37     // The values of `*guard` are as you set them in
38     // __sanitizer_cov_trace_pc_guard_init and so you can make them consecutive
39     // and use them to dereference an array or a bit vector.
40     *guard = 0;
41     void *PC = __builtin_return_address(0);
42     PCNode *node = malloc(sizeof(PCNode));
43     *node = (PCNode){PC, NULL};
44     OSAtomicEnqueue(&queue, node, offsetof(PCNode, next));
45 }
```

## 后记

苹果官方也提供了 PGO 的详细文档，而且操作很简单。不过它跟二进制文件重排还是有区别的，这里不展开讲了。毕竟相对于对业务代码加载优先级的优化来说，PGO 对启动优化性价比没那么高，应该就是高频调用函数内联之类的（这句纯属瞎扯）。

我为啥过了这么久才发此文呢？猜猜原因是啥：

- A. 不爱蹭热度
- B. 喜欢炒冷饭
- C. 忙准备答辩
- D. 8 月发过文章了，这篇得等 9 月才能发，这样不浪费

碰到不会的题，我一般三短一长选最长。

### Github 名片



杨萧玉

55

REPOS

6

GISTS

2.0K

FOLLOWERS

### 友情链接

pbxprojHelper  
SimilarImageHunter  
Spiral  
ColorAtom  
养小鱼的水塘  
我的简历

### 标签云

ARC Algorithm App Extensions  
AppGroups  
AppleScript BlockHook C  
CocoaPods Core Data GitHub  
Java  
Machine Leaning Message Forwarding  
Messaging Metal Objective-C  
Octopress RAC Reference Counting  
Reverse Engineering Runtime  
Social Framework SpriteKit  
Swift UIKit Dynamics VPN Xcode  
iCloud iOS macOS 字体  
本地化 瞎折腾  
碰撞检测 翻译 设计模式 转载

RSS 订阅

### 新浪微博



杨萧玉HIT

+ 加关注

### 豆瓣秀



瞎折腾



下一篇：

[BlockHook with Invocation\(2\)](#) ➔



stay hungry, stay foolish  
Talk is cheap, show me the code.

