编译原理 Lambda 表达式 LLVM 修改

在实现编程语言的 lambda 表达式的过程中，如何实现外部变量的捕获（编译到LLVMIR）？ 修改

我想到一个办法，就是作为额外的参数传进去，传引用那种。不知道有没有“正统”的做法。

修改

关注问题 写回答 邀请回答 2 条评论 分享 邀请回答 举报 ···

关注者 85 被浏览 5,525

他们也关注了该问题

查看全部 7 个回答

Felis sapiens
函数式编程、编程语言、编程 话题的优秀回答者

圆角骑士魔理沙、考古学家千里冰封、彭飞、开源哥、刘雨培等 15 人赞同了该回答

ghc runtime 的内存布局中，堆上的对象都是 info pointer + payload 的模式，info pointer 指向一个 info table，info table 维护一些与该对象相关的元数据（对象类型、payload 的布局、用于触发该对象求值的 entry code 等等）。如果这个对象是一个函数闭包或者 thunk，那么就可能有自由变量，这些自由变量的指针保存在 payload 的 pointer 字段里。

基于 ghc 衍生的 haskell to js 编译器中，ghcjs 采取类似 ghc 的做法，用 js 的普通对象表示一个 stg machine 的堆对象，自由变量就是 js 对象的成员，而保留成员 obj.f 则是对应 entry code 的函数。而 haste 则直接使用 js 函数实现 haskell 函数，haskell 中 lambda 表达式的自由变量直接对应 js 函数捕获的自由变量。

Commentary/Rts/Storage/HeapObjects - GHC

编辑于 2017-08-29

▲ 赞同 15 ▼ 添加评论 分享 收藏 感谢 ···

更多回答

SuperSodaSea
造轮子，正在填坑中

11 人赞同了该回答

举个栗子吧，比如说C++中

```
int a = ...;
auto func = [a](int b) { return a + b; };
```

其实相当于是

```
int a = ...;
class XXX {
private:
    int a;
public:
    XXX(int a_) : a(a_) {}
    int operator ()(int b) const { return a + b; }
};
```

展开阅读全文

关于作者

Felis sapiens

函数式编程、编程语言、编程 话题的优秀回答者

电影旅行敲代码、Antokha Yuuki、暮无井见铃也关注了她

回答 328 文章 40 关注者 14,871

已关注 发私信

被收藏 3 次

收藏夹 273 人关注
圆角骑士魔理沙 创建

博闻强识 0 人关注
灯语 创建

计算机 0 人关注
tomsu 创建

相关问题

▲ 赞同 11 ▼ 4 条评论 分享 收藏 感谢 ···

匿名用户

10 人赞同了该回答

展开阅读全文

▲ 赞同 10 ▼ 添加评论 分享 收藏 感谢 ···

查看全部 7 个回答

怎样给想自制编译型程序语言的有志青年推荐后端·runtime? 11 个回答

为什么程序语言会存在解释型或编译型的限制? 11 个回答

任何编程语言都可以编译为原生的机器码吗? 13 个回答

编译器是如何编译自己的? 24 个回答

编译器是怎么编译出来的? 6 个回答

相关推荐

淼懂物理学：理解世界的极简指南

共 31 节课 ▶ 试听

自己动手构造编译系统：编译、汇编与链接

7 人读过 阅读