轻舟已过万重山——编译器优化

> 班级：212113
>
> 姓名：田乐
>
> 学号：21371362

~~以下内容总体来说比较逆天，充斥着厚重的历史包袱所带来的无奈的妥协和退让，但毫无疑问的，从本质上来说是一次成功的优化。从存在主义的角度出发，一切为了存在，存在就是一切；类似的，在这里，只要能有优化的效果，哪怕是微乎其微的渺小的效果，也是有意义的。故曰：宜将剩勇追穷寇，轻舟已过万重山~~

寄存器分配

由于本次的编译器开发中，寄存器分配是相当关键的一步，早在代码生成二刚刚通过的时候，便尝试性地提交了竞速排序地测试，结果发现最后一个点没有通过，当此之时，智慧的开发者便高瞻远瞩地提出了一个具有伟大历史意义的猜想：**这个点没过就是因为没有分配寄存器，全部都存到栈里面了**，事实证明，这个猜想不能说完全正确，只能说没什么鬼用。

接下来从下面两个方面开始试图解决寄存器分配的难题

全局寄存器与局部变量

在已有的架构下，智慧的开发者选择了先划分函数块（~~这是多么富有创新意义的壮举啊~~），再在函数块中划分基本块的方式，随后对于跨基本块的变量，使用最常规的活跃变量分析和图着色算法即可。在具体的开发中，`$s`系列寄存器用来承载这个内容

局部寄存器与临时变量

由于中间代码部分的历史遗留问题，出现了较多的临时变量，为此，智慧的开发者依旧使用了图着色算法。算法实现的具体方式和全局寄存器那边简直是一模一样。因此不再赘述。

有意思的地方是，如何绘制一个基本块内的临时变量的冲突图。在此，智慧的开发者选择使用猪的战术——大力处奇迹。逐行遍历以确定每一个临时变量的活跃范围，再据此来找出活跃范围有重合的临时变量，判定其为冲突

> 怎么样，是不是很残暴（

但不管怎么说，虽然很逆天，这个优化确实是有效果的.....