编译大作业

设计思路

我们的设计思路分为三部分，其中第一部分为根据输入的Json文件生成Json文法树，第二部分为根据生成的Json文法树生成IR语法树，第三部分为根据IR语法树生成C++代码。

实现方法

生成json语法树

这里采用的算法类似于LR文法分析器的方式，即通过自底向上的方式来构造Json语法树，我们根据给定的Json文法设定了所有的Json语法树结点，这些结点的定义位于**include/JsonExpression.h**中。然后我们遍历Json中的Kernel串，使用**include/JsonParser.h**中定义的parser来处理该串。

我们使用类似于访问IR的visitor模式来访问Json语法树，其中**include/JsonVisior.h**用于调试时输出kernel，**include/JsonAbsiVisitor.h**用于生成IR语法树。

生成IR语法树

每一条语句用一个 Stmt 来表示， 其中包含了相关的 for 语句， 然后是条件的判断。 由于 IR 的语法树并没有括号内容， 所以括号交给了 IRPrinter 处理， 语法树上不处理 我们提取出每一个变量的范围，然后加入一个vector里面， 以方便 for 循环确定范围。 然后对于每一个约束我们用一个 if 实现，以保证数组下标不会越界。 由于 IRPrinter 的原因， 我们采取的 else 里面的内容是一个空的 LoopNest，方便后续代码使用。

生成C++代码

该部分的工作是：将已经建立好的IR节点树，通过深度遍历地方式，打印出与之对应C语言程序。

立即数

在对立即数进行处理时，关键要考虑不同类型在C中如何表示。C中整数默认为int，对浮点数默认为double。对于uint32, int64, float,可以直接通过 U, LL, F 等后缀标识有无符号、不同字节数的立即数。但对于 int16, int8 这种数据类型，必须通过显式类型转换，比如`(unsigned char)2`来突出其数据类型。

运算和表达式

每个运算和表达式都逐一对照翻译，但关键需要考虑运算优先级的问题。根据在我们IR树转换的实现方式，只可能存在二元算术运算会有运算优先级问题，我们需要给所有BinaryOp的两个运算数都添加括号，来确保运算不出错。

Dom、Index和LoopNest

由于在C中不能直接使用某种类似于Dom的东西，我们不直接处理Dom，而是将它和Index一起处理。考虑到所有Index基本都对应了for循环，我们将Index翻译成`(int i = beg; i < end; ++i;)`的形式，而`for`关键字通过LoopNest产生。这就将每个LoopNest翻译成一个标准的for循环。

条件语句

条件语句只需将它翻译程C语言中的if(),else形式。必须确保使用了{}包裹语句块，因为语句块内可能没有语句，可能导致后续其它语句被算作else语句。

Move

在本次实验中我们只需要实现MemToMem的移动，这种移动只需使用等号赋值语句即可。

分工

生成Json语法树：王嘉睿

生成IR语法树：侯锐泽

生成C++代码：余忠蔚、刘诗逸