编译Project2报告

1700012943 周迈

1700012845 刘易鑫

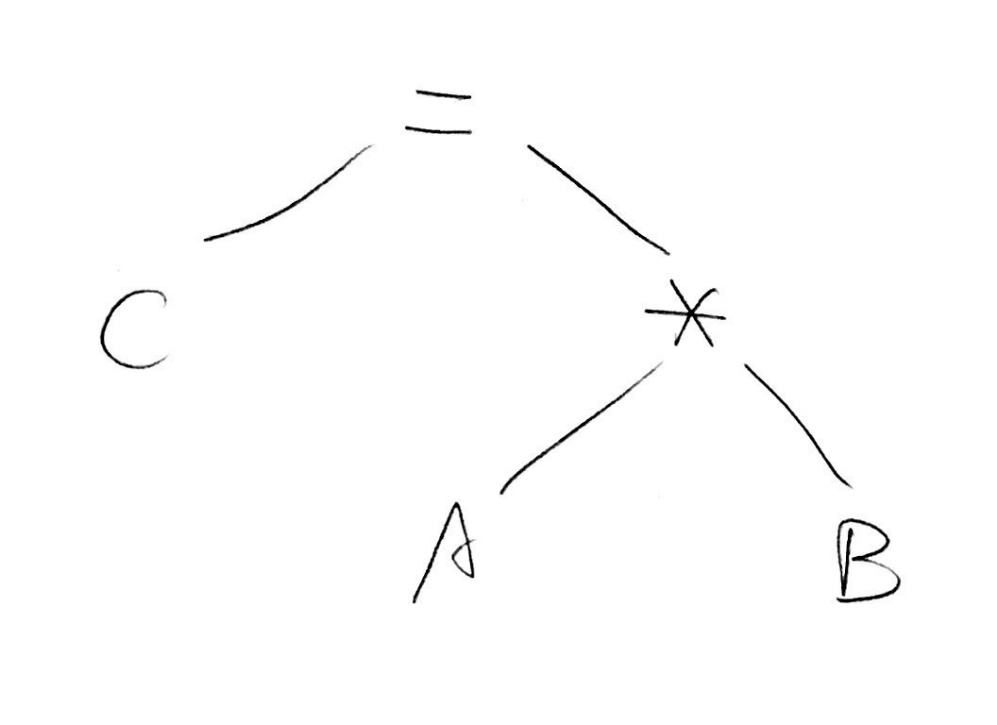
1700012757 杨扬

1. 小组分工

算法设计：全体

算法实现：周迈

1. 算法思想
2. 通过Project1实现的功能得到要求导语句的语法树；
3. 自顶向下递归扫描语法树，在合适的结点输出响应内容，得到梯度树
4. 再利用Project1的内容将梯度树打印出来即可。
5. 具体实例

以case3的“C<4, 16>[i, j] = A<4, 16>[i, k] \* B<16, 16>[k, j];“为例，首先我们得其语法树 。

然后，从=结点开始，首先访问其左子结点C，在输入中我们可以得知，C是输出变量，而最终结果中=左侧应为dA，故我们将其替换。但此时我们还不能确定dA的情况，故先空着。此外，我们还应将C的情况记录下来。

然后访问到\*结点，进而访问其两个子结点。访问到A时，我们知道其为要求导的变量，且知道其处在A\*B这样一个子式中，于是可以将A的情况记录下来，然后对其进行求导，将其直接替换为dC。

到了B结点，由于其处于乘法子式A\*B中，我们可以直接将其保留。

最后是递归的返回过程，将刚刚得到的东西正常结合起来即可。

1. 具体实现

我们根据给的提示，自己写了一个MyMutator来对语法树进行遍历，再将遍历的结果输出。

除此以外，还需要调整函数的参数的顺序。此处我们对函数的情形作了简单假设，即待求导参数只有一个，且其导数中不含有自身。此时，只需将输入变量中待求导的那个直接放到变量的最后即可。

我们的实现可以很好地求出一些简单的导数，但对复杂情况的求导能力还有待加强

1. 其它

本次Project的实现比较类似于SDT的实现过程，加深了我们对编译知识的理解。