Michgan Tech

Reference group detection and Loop cost

Project IV

Writen by: liang YAN

Computer Science

Last modified: May 2014

0.1 Introduction

1. compile the source file

clang -O1 -emit-llvm -c loops.c

2. run tests

opt -analyze -ds –refgroup=true –cls=64 loops.bc

0.2 Implementation

0.2.1 Basic Algorithm

```
class InstrInfo{
public:
  unsigned pairSize;
  Instruction *instruction;
  unsigned currentLoopLevel;
  std::string controlVariable;
  APInt UpperBound[10];
  unsigned PairLevel[10];//each subscript is at each level
  APInt stride[10];
  unsigned CacheLineCount;
  bool used;
};

std::map<Instruction *, InstrInfo> instrInfoList;

class RefGroupBase {
public:
  Instruction* groupLeader;
  APInt loopCost;
  int CalculateCost();
  std::set<Instruction *> groupList;
};

class RefGroup {
public:
  unsigned currentLoopLevel;
  std::string controlVariable;
  std::vector<RefGroupBase > groupForLoop;
};


APInt totalBound[10];

std::map<unsigned, RefGroup> refGroupResult;
```

First, use the project 3 part, to collect the instruction we need here, and save the information to map structure instrInfoList;

then, analysis this structure by using the function depends(),and RefGroup algorith mentioned in Mckinley's paper, divide the instruction to different group based on different loop control Variable.

then analysis the instruction in each RefGroup, find the leader.

then use the leader to Calculate the loop cost based on different loop control Variable.

Finally, output all information to screen.

0.2.2 Loop Nest Number

Find the outermost loop (no parent) of each loopnest, save it to a currentLoop, then , find another outermost, check if it is same as the currentLoop, if same, do nothing, not same, let loopNest++, and make currentLoop = current outermost loop.

0.2.3 Get the variable type

Need to know the value type of the array, because different type comes with different layout size. Int would be 4, and doulbe is 8, which means a cacheline with size 64 can hold 16 integers but 8 doulbe variable. This is one thing we need to notice, if we get a pointer value, we need to find the element it points to.

0.2.4 Get the upperbound

Upperbound is necessary for a[i][k][k] and a[k][j][k], this kind of situations. We also need to get all level upperbound in advance.

0.2.5 Get the dependence

if two instructions are loopinvariant, then in the same group; else check the outtermost leveal if existing group Spatial, then check if group temp based on different loop control variable. when insert a new instruction, I will check it with the exist loop, if has dependence, then insert it to the group, after that, check it with the rest instruction, if connected, insert to the same group.

0.2.6 Get the leader

I use the simple idea, to check the distance in differnt loop level from outermost to innermost, choose the bigger one to be the leader, when finishing the iteration, get the leader of the group.

0.2.7 Calculate the cost

check the subscript of the groupleader, if the loop control variable level equals to the subscript level, and subscript belongs to a same loop level, then self Spatial exist, Calculate it with UpperBound*stride/cachelinecount, else, let it be 1, then multiple this value with the UpperBound in other level.

0.2.8 Difficulty

did not deal with nest loop since use a different method comparing with project 3, could not work well with some spacial cases. Still need to familiar with scalar evaluation class.