Compiler Design Assignment

Name: Laxman Chandrasali
Reg. No.: 201700491
Sec.: C
Roll No.: 40

Peephole Optimization

A straightforward yet successful procedure for improving the objective code is peephole optimizer, a technique for attempting to improving the presentation of the objective program by analysing a short arrangement of target guidelines (called the peephole) and supplanting these directions by a more limited or quicker grouping, at whatever point conceivable.

Peephole optimization is a type of Code Optimization performed on a small part of the code. It is performed on the very small set of instructions in a segment of code.

It basically works on the theory of replacement in which a part of code is replaced by shorter and faster code without change in output. Peephole is the machine dependent optimization.

Objectives of Peephole Optimization:

i. To improve performance
ii. To reduce memory footprint
iii. To reduce code size

Peephole Optimization Techniques:

1. Redundant Instruction Elimination:
   At compilation level, the compiler searches for instructions redundant in nature. Multiple loading and storing of instructions may carry the same meaning even if some of them are removed. For example:

```
MOV x, R0
MOV R0, R1
```

We can delete the first instruction and re-write the sentence as:

```
MOV x, R1
```

2. Removal of unreachable code:

   Inaccessible code is a piece of the program code that is never gotten to considering programming builds. Software engineers may have accidently composed a bit of code that can never be reached.

3. Flow of control optimization:

   There are instances in a code where the program control jumps back and forth without performing any significant task. These jumps can be removed.

4. Algebraic Simplification:

   There are occasions where algebraic expressions can be made simple. For example, the expression a = a + 0 can be replaced by itself and the expression a = a + 1 can simply be replaced by INC a.

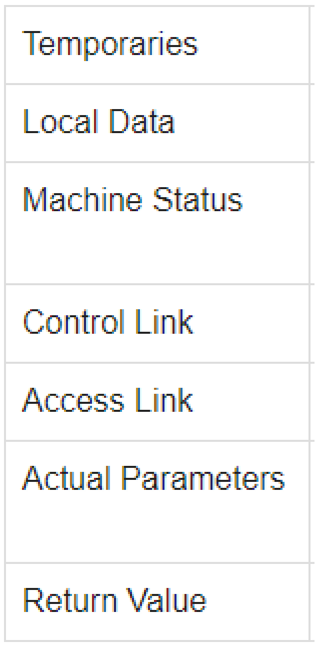
5. Machine Idioms:

   The objective machine can send more advanced guidelines, which can have the ability to perform explicit activities much productively. On the off chance that the objective code can oblige those guidelines straightforwardly, that will not just improve the nature of code, yet additionally yield more productive outcomes.

Activation Records:

A program is a grouping of guidelines joined into various techniques. Directions in a method are executed successively. A method has a beginning and an end delimiter and everything inside it is known as the body of the system. The system identifier and the succession of limited directions inside it make up the body of the technique.

The execution of a procedure is called its activation. An activation record contains all the necessary information required to call a procedure. An activation record may contain the following units:



Temporaries
Local Data
Machine Status
Control Link
Access Link
Actual Parameters
Return Value

Return Value: It is used by calling procedure to return a value to calling procedure.

Actual Parameter: It is used by calling procedures to supply parameters to the called procedures.

Control Link: It points to activation record of the caller.

Access Link: It is used to refer to non-local data held in other activation records.

Saved Machine Status: It holds the information about status of machine before the procedure is called.

Local Data: It holds the data that is local to the execution of the procedure.

Temporaries: It stores the value that arises in the evaluation of an expression.

DAG representation of basic blocks:

A DAG for a basic block is a directed acyclic graph with the following labels on nodes:

1. Leaves are labeled by unique identifiers, either variable names or constants.

2. Interior nodes are labeled by an operator symbol.

3. Nodes are also optionally given a sequence of identifiers for labels to store the computed values.

DAGs are useful data structures for implementing transformations on basic blocks. It gives a picture of how the value computed by a statement is used in subsequent statements. It provides a good way of determining common sub - expressions.

Algorithm for construction of DAG

Input: A basic block

Output: A DAG for the basic block containing the following information:

1. A label for each node. For leaves, the label is an identifier. For interior nodes, an operator symbol.

2. For each node a list of attached identifiers to hold the computed values.

Case (i) x : = y OP z

Case (ii) x : = OP y

Case (iii) x : = y