College of Engineering Trivandrum

Compiler Design Lab

Abhishek Manoharan
S7 CSE Roll No:2

TVE17CS002

Department of Computer Science

November 5, 2020

CS431 - Compiler Design Lab · 2020 ·

Exp 12

1 Loop Unrolling

1.1 Aim

Write a program to perform loop unrolling

1.2 Theory

Loop Unrolling

Loop unrolling is a loop transformation technique that helps to optimize the execution time of a program. We basically remove or reduce iterations. Loop unrolling increases the program's speed by eliminating loop control instruction and loop test instructions.

1.2.1 Advantages

- Increases program efficiency.
- Reduces loop overhead.
- If statements in loop are not dependent on each other, they can be executed in parallel.

1.2.2 Disadvantages

- Increased program code size, which can be undesirable.
- Possible increased usage of register in a single iteration to store temporary variables which may reduce performance.
- Apart from very small and simple codes, unrolled loops that contain branches are even slower than recursions.

1.3 Algorithm

Algorithm 1: Algorithm for Loop Unrolling

```
Read the loop
Store initial, terminal condition and variable
unroll the loop with modifying initial and terminal condition
change variable name accordingly
```

1.4 Code

```
#include <bits/stdc++.h>
using namespace std;

string beautify(string s) // to remove unneccessery space , () etc in the loop
{
    string new_s = "";
    int n = s.size();
    int flag = 0;
    for (int i = 1; i < n; ++i)
    {

        if (s[i - 1] == '(')
        {
            flag = 1;
        }
```

```
        else if (s[i] == ')')
        {
            break;
        }
        if (flag == 0)
        {
            continue;
        }
        if (s[i] != ' ')
        {
            new_s += s[i];
        }
    }
    return new_s;
}
void get_det(string s, int *start, int *end, int *cond, char *var, string *relop) // find the variable start and end condition etc
{
    s = beautify(s);
    //cout << s << " trimmed string " << endl;
    *var = s[0]; // variable returned
    int first = 0, second = 0, n = s.size();
    for (int i = 0; i < n; ++i)
    { // finding index of ;
        if (s[i] == ';')
        {
            first = second;
            second = i;
        }
    }
    string init = s.substr(2, first - 2);
    //cout << init << " initial value" << endl;
    *start = stoi(init);
    //cout << s[first + 2] << " " << s[first + 3] << endl;
    if (s.substr(first + 2, 2) == "<=")
    {
        *relop = "<=";
        init = s.substr(first + 4, second - first - 4);
        //cout << init << " terminal value" << endl;
    }
    else if (s[first + 2] == '<')
    {
        *relop = "<";
        init = s.substr(first + 3, second - first - 3);
        //cout << init << " terminal value" << endl;
    }
    else if (s.substr(first + 2, 2) == ">=")
    {
        *relop = ">=";
        init = s.substr(first + 4, second - first - 4);
        //cout << init << " terminal value" << endl;
    }
    else
    {
        *relop = ">";
        init = s.substr(first + 3, second - first - 3);
        //cout << init << " terminal value" << endl;
    }
    *end = stoi(init);
    if (s[second + 1] == '+')
    {
        *cond = 0;
    }
    else
    {
        *cond = 1;
    }
}
void print_with_newval(vector<string> lines, vector<pair<int, int>> variable, string replace)
{
    int rep_count = variable.size();
    int n = lines.size(), curr = 0, flag;
    if (curr == rep_count)
    {
        flag = 1;
    }
```

```
    else
    {
        flag = 0;
    }
    for (int i = 2; i < n - 1; ++i)
    {
        if (flag == 1 || variable[curr].first != i)
        { // print thr line if falg = 1 or the line is free of variable
            // cout << "no variable in line " << i << endl;
            cout << lines[i] << endl;
        }
        else
        {
            int pos = 0;
            while (variable[curr].first == i)
            { // repeat untill the line has the  loop variable
                //cout << "line found";
                cout << lines[i].substr(pos, variable[curr].second - pos);
                pos = variable[curr].second + 1;
                cout << replace;
                curr++;
                if (curr == rep_count)
                {
                    flag = 1;
                    break;
                }
            }
            if (pos < lines[i].size())
            { // print the rest of the line
                cout << lines[i].substr(pos, lines[i].size() - pos) << endl;
            }
        }
    }
}
int main()
{
    vector<string> lines;
    string s, relop;
    ifstream file("loop.c");
    cout << "Reading from input.c" << endl;
    while (getline(file, s))
    {
        cout << s << endl;
        lines.push_back(s);
    }
    int start, end, cond; //cond = 0 for < , 1 for <= , 2 for >,  3 for >=
    char var;
    get_det(lines[0], &start, &end, &cond, &var, &relop);
    cout << "variable is " << var << " initial,terminating values are = " << start << "," <<
    end << endl;
    cout << "Unrolled Loop" << endl;
    cout << "**************************" << endl
         << endl;
    vector<pair<int, int>> variable;
    for (int i = 2; i < lines.size() - 1; ++i)
    {
        for (int j = 0; j < lines[i].size(); ++j)
        {
            if (lines[i][j] == var)
            {
                if (j == 0 && !isalnum(lines[i][j + 1]))
                {
                    variable.push_back({i, j});
                }
                else if (j == lines[i].size() - 1 && !isalnum(lines[i][j - 1]))
                {
                    variable.push_back({i, j});
                }
                else if (!isalnum(lines[i][j + 1]) && !isalnum(lines[i][j - 1]))
                {
                    variable.push_back({i, j});
                }
            }
        }
    }

```

```
    // for (auto x : variable)
    // {
    //     cout << x.first << " " << x.second << endl;
    // }
    string i_d;
    i_d = cond == 0 ? '+' : '-';
    cout << "for (" << var << " = " << start << "; " << var << i_d << "4"
         << " " << relop << " " << end / 4 << "; ";

    cout << var << " " << i_d << "= 4)" << endl;
    cout << "{" << endl;
    print_with_newval(lines, variable, var + i_d + '0');
    print_with_newval(lines, variable, var + i_d + '1');
    print_with_newval(lines, variable, var + i_d + '2');
    print_with_newval(lines, variable, var + i_d + '3');
    cout << "}" << endl
         << endl;
    cout << "**************************" << endl;
}
```

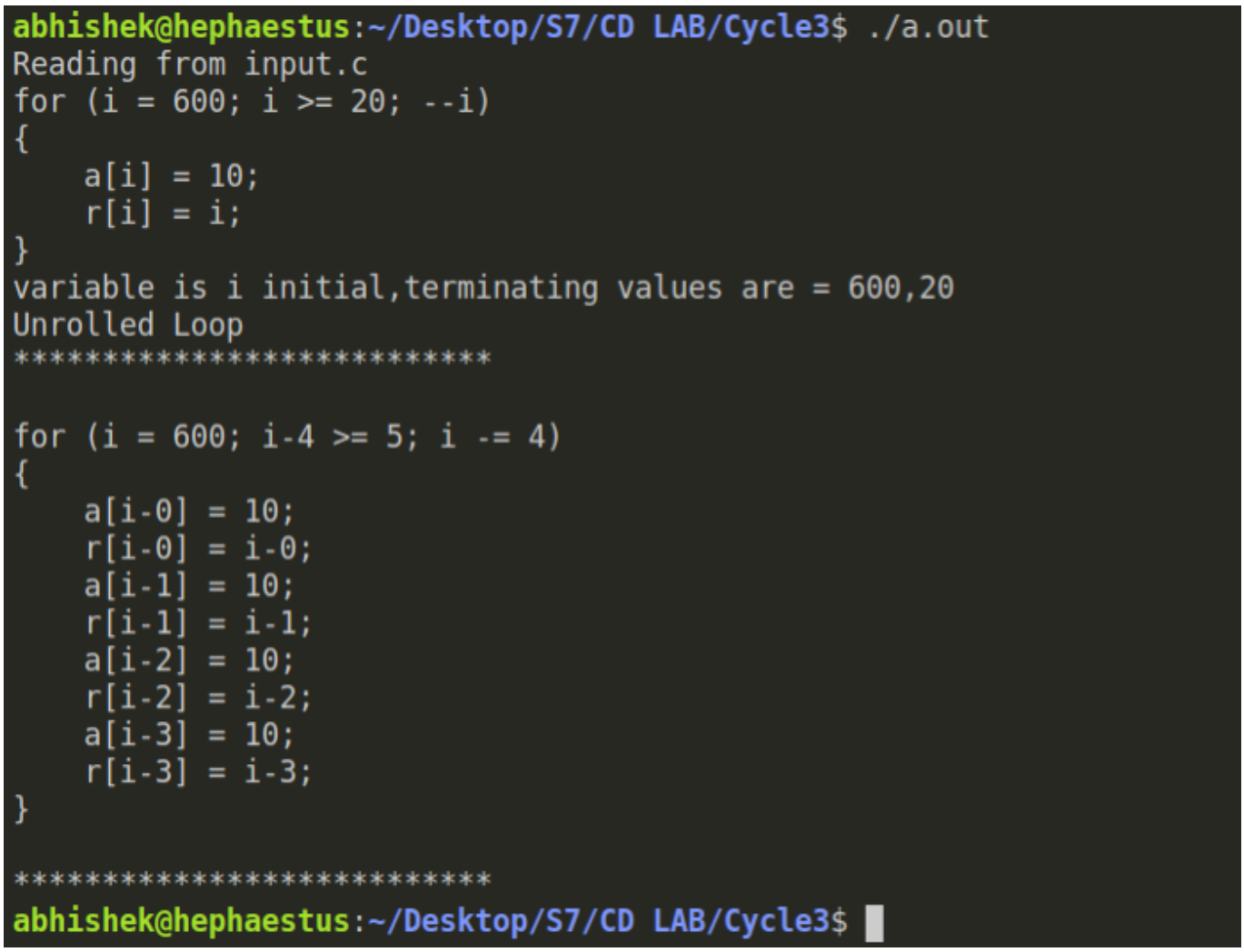
Code for SR Parser

1.5 Output

```
abhishek@hephaestus:~/Desktop/S7/CD LAB/Cycle3$ ./a.out
Reading from input.c
for (i = 600; i >= 20; --i)
{
    printf("Hello world\n");
}
variable is i initial,terminating values are = 600,20
Unrolled Loop
**************************

for (i = 600; i-4 >= 5; i -= 4)
{
    printf("Hello world\n");
    printf("Hello world\n");
    printf("Hello world\n");
    printf("Hello world\n");
}

**************************
```




```
abhishek@hephaestus:~/Desktop/S7/CD LAB/Cycle3$ ./a.out
Reading from input.c
for (i = 600; i >= 20; --i)
{
    printf("Hello world\n");
}
variable is i initial,terminating values are = 600,20
Unrolled Loop
**********

for (i = 600; i-4 >= 5; i -= 4)
{
    printf("Hello world\n");
    printf("Hello world\n");
    printf("Hello world\n");
    printf("Hello world\n");
}

**********
abhishek@hephaestus:~/Desktop/S7/CD LAB/Cycle3$ g++ loop_unroll.cpp
abhishek@hephaestus:~/Desktop/S7/CD LAB/Cycle3$ ./a.out
Reading from input.c
for (i = 600; i >= 20; --i)
{
    a[i] = 10;
    r[i] = i;
}
variable is i initial,terminating values are = 600,20
```

```
Unrolled Loop
**************************

for (i = 600; i-4 >= 5; i -= 4)
{
    a[i-0] = 10;
    r[i-0] = i-0;
    a[i-1] = 10;
    r[i-1] = i-1;
    a[i-2] = 10;
    r[i-2] = i-2;
    a[i-3] = 10;
    r[i-3] = i-3;
}

**************************
abhishek@hephaestus:~/Desktop/S7/CD LAB/Cycle3$
```

1.6 Result

Implemented the program for loop unrolling. It was compiled using g++ version 9.3.0, and executed in Ubuntu 20.04 and the above output was obtained.