International Journal of Advanced Technology in Engineering and Science www.ijates.com
Volume No 03, Special Issue No. 01, May 2015 ISSN (online): 2348 – 7550

HOW TO REDUCE EXECUTION TIME

Amit Bansal[1], Vipin Babbar[2]

[1,2]*Department of Computer Sc. Govt. College for Women, Hisar, Haryana, (India)*

ABSTRACT

Complexity of the software increases as the dependence on the computer is increasing. The importance and use of computer can be enhanced by reducing execution time of software. Writing a program is a good skill to be a programmer. But writing a bug free program makes a programmer better. Programmer can be the best if he can write programs which can produce results in no time. Testing can be used to check the time taken by program by applying different types of methods to solve a particular problem. We can also test our program by applying different technique in a block which are taking most of the processing time so that technique which are using less time without changing the goal of program can be incorporated in our program. This paper explains some of the techniques to reduce execution time in basic programming.

I. INTRODUCTION

Testing is a process

- To find out errors in a program.
- Try to fail the program by testing on some inputs which can generate errors in a program.
- Spend some time on executing the program by giving different inputs in the whole domain.
- To understand the working of a program.
- To understand different inputs and their corresponding outputs.
- It is a way to makes program work in adverse conditions.
- It is a way to trust on a program to work in a critical situation.

Testing may also be used to check the time taken by program to produce required output. If the time taken by program to produce required output is in valid range than time parameter can be accepted. If the time is not in the valid range, it can be put in that range by doing some of the very basic changes in the program as given below.

II. DIFFERENT METHODS TO REDUCE EXECUTION TIME

2.1 To find some repetitive process and think whether it can be converted in recursive process. Because recursive process takes less time to execute than looping and easy to understand for a programmer.

2.2 You can use formulas instead executing the whole process of the formula for ex.

Use (N*(N+1))/2 to calculate sum of natural numbers instead of using for loop like

```
Sum=0;
For(i=1;i<N;i++)
        Sum=sum+I;
Printf("%d",sum);
```

Using the formula instead of detailed calculation may takes less time to execute.

2.3 Use faster algorithm for data structure operations in the application you are working. This can be achieved by comparing complexity. For example in searching in an array, binary search is less time consuming than linear search. So using binary search instead of linear search might takes less time to execute in most of the cases. But selection of technique may depend upon the behavior of problem.

DS operation	Worst case complexity	Best case complexity	Avg. case complexity
Linear searching	O(n)	O(1)	O(n)
Binary searching	$\log_2 n$	O(1)	$\log_2 n$
Bubble Sort	$O(n^2)$	O(n)	$O(n^2)$
Insertion Sort	$O(n^2)$	O(n)	$O(n^2)$
Selection sort	$O(n^2)$	$O(n^2)$	$O(n^2)$
Quick sort	$O(n^2)$	$O(n\log_2 n)$	$O(n\log_2 n)$
Merge sort	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n\log_2 n)$
Heap sort	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n\log_2 n)$
Radix sort	$O(n^2)$	$O(n\log_2 n)$	$O(n\log_2 n)$

2.4 Avoid unnecessary declaration of a variable.

2.5 If we have functions in our programs, then we have to call them by reference instead of call by value. But the danger of changing the original value can be avoided by declaring the parameter as a const.

2.6 Use of macros for smaller functions may reduce the time to execute.

2.7 Try to maximize the step value in for loop. So that no of iteration should be minimum for a particular problem.

For Ex consider the for loop to calculate sum of odd numbers from 1 to 10.

```
Sum=0;
For (i=1;i<10;i++)
{
      If(i%2!=0)
              Sum-sum+i;
}
```

Instead of using this for loop we can do this as follows:

```
Sum=0;
For (i=1;i<10;i+=2)
```

```
{
          Sum-sum+i;
}
```

In the first method there are 10 iterations as well as extra condition checking makes this method more time consuming than the second method.

2.8 Include only those files in your program which are necessary for your program. Inclusion of header Files which are not used in C/C++ Programs may increase compilation time[2].

III. CONCLUSION

The methods explained above looks to be promising to reduce the total execution time of a program in any programming language. All these methods are explaind on the basis of theoretical knowledge and some programming techniques . But still I need experimental proof to prove these facts. The main difficulty for taking experimental data is that the time taken by the smaller programs in basic programming is not measurable as these statements takes approximately no time to execute. We have to find the smallest time unit to measure the time.