Comparative Analysis of the Different Compilers

* Sindhuja. L ** Nirmala Gupta
*Sindhu.sin6@gmail.com **nirmalaguptha@revainstitution.org
School of computing and information technology, Reva University, Bangalore

Abstract: Recently there are lot many compilers Eg clang c++, Cygwin and lot many available. Users are not able to judge which compiler is best to analyze, we are aiming to explore best compiler by comparative analyst. We demonstrate this by taking the same code and demonstrating it with the different compilers. By the demonstration, we can observe that gnu/g++ gives better performance w.r.t other compilers.

1. Introduction

Despite the advent of new programming languages and technologies C++ is the workplace for many developers and is likely to remain so far a long time to come. The main reasons for C++ prominence are its flexibility, portability,efficiency and performance continues to be important. Aspect for benchmarking a compiler is not about the quality of the resulting code but also how long the compiler has taken to compile. That's get tricky as well, because as well, because there are so many compilers options that can skew the results.

While programs that compare different versions of the same source code file have been in widespread use for many years, very little focus has so far been placed on the importance of detecting and analyzing changes between two versions of the same executable.By just comparing the speed of executing a compiler will not decide which the best one to choose is.

To decide the best compiler some factors come into the main role:time to compiled code,size of compiled code,memory usage of compiled code bugs etc.In existing work [1],3compilers that is INTEL C++,GNU C++ and LLVM have been demonstrated than gnu c++ was among the three compilers.

2.Proposed work.

Compiler is a computer program that transforms source code from high level language into lower level language. Compiler includes better detection mechanisms, higher performance in terms of execution and enhancesoptimization. There are many lists of compiler [1],existing in the global technology of world.chossing and analyzation of the best and top most compiler have been made.More RAM, faster hard drives (including SSDs), and more CPUs/cores will all make a difference in compilation speed.

To choose a compiler, comparision of different compilers have been made. To decide which complier are best there are some factors which has to be considered such as time taken to compile code, size of compile code, memory usage of compiled code etc;

In this paper, some of compilers in the list have been chosen and made comparison between them. Compilers used are Borland c++, digital mars and gnu/g++.

This paper presents a methods to validate the compiler using random programs. Different compilers have been tested by the random programs. Common ways to compare compilers is by checking the functionalities of the compilers. Efficiency of programming language depends upon compiler and IDE which are going to be used.Effiecient and quick compilers which makes your code run faster is hard to search .All we need is to choose the best compilers among the list of the compilers. Analyzation of the top most compilers have been made as to Compare the compilers with different programs with no limits and also taking with the limits (using pointers).This requires a lot more analysis by running sample c/c++ code in all different compilers.Here we have taken a big c/c++ program which does the task for analysis.

3. Requirments.

Borland c/c++, dm, gnu c/c++ compiler are the compilers used here to compare with the different programs.dijistras program and also prime no program have been taken to compare the compilers.

3.1 Taking programs with different compilers with no limits.

Dijkstra's algorithm has been take taken to analyze the time.dijistras program has been executed in all three compilers. Snapshot have been displayed below which displays the time generated.
The command used in compilation for the Borland C++ is bcc new1.cpp and for execution it is new1.exe.In Borland C++ better interpretation of the warnings will be shown. Function related interpreter will not be handled properly.
Three files are generated after the execution of the code.Time generated when dijkstra's code is executed is 0.015 sec in GNU G++ compiler.

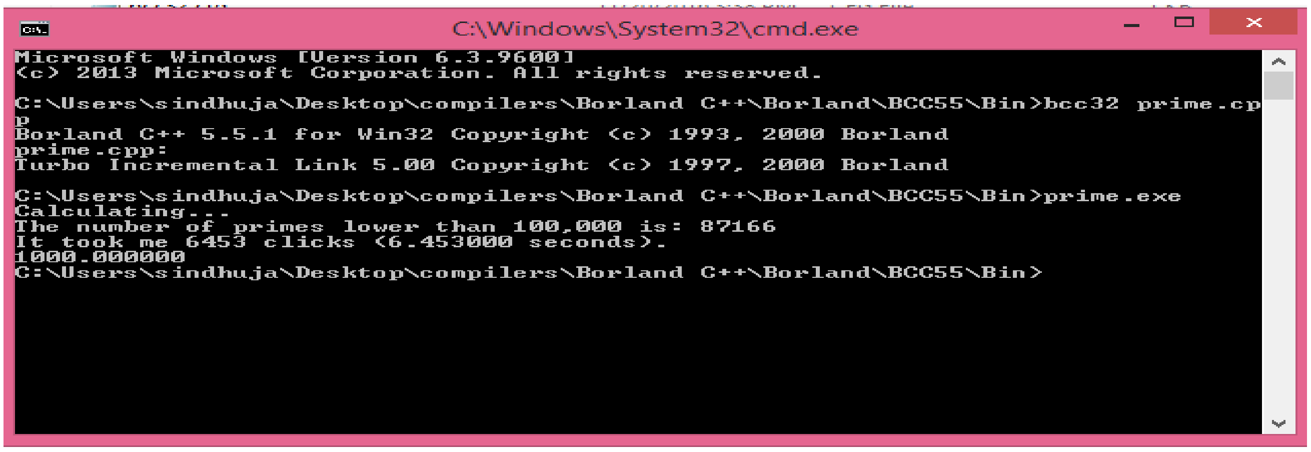
3.2:Taking programs with different compilers with limits (pointers) in the program.

When Dijkstra's program used using pointers in them in the code and when executed in the Borland c++ it takes 0.0015sec to execute.

When Dijkstra's program used using pointers in them in the code and when executed in the gnu/g++ it takes 0.002sec to execute. By this demonstration we can justify that gnu/g++ compiler has taken least time to compile.

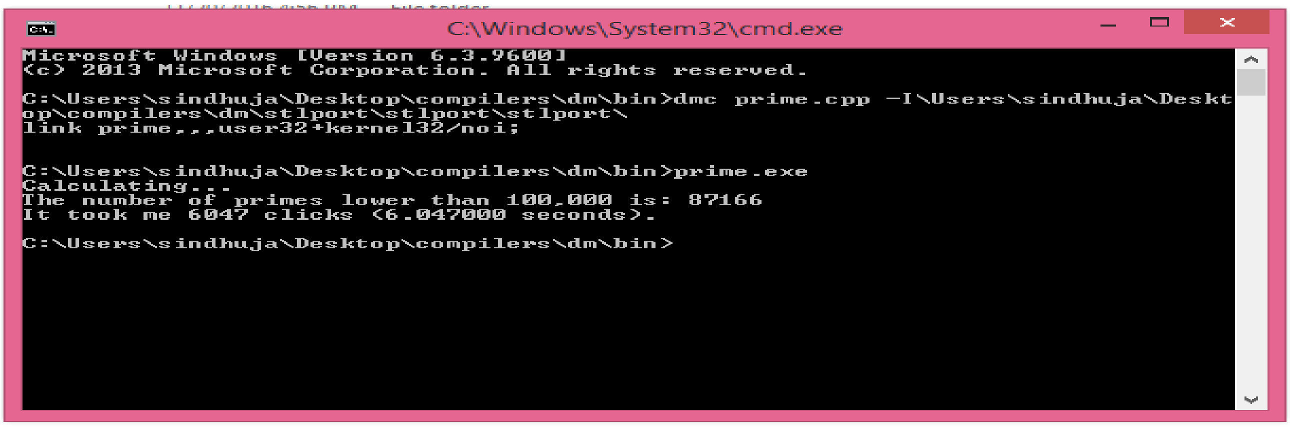
3.3 Taking a different program in all three compilers.

Prime no program is best example to optimize the time. Below are snapshots of the output of the code generated in all three compilers.



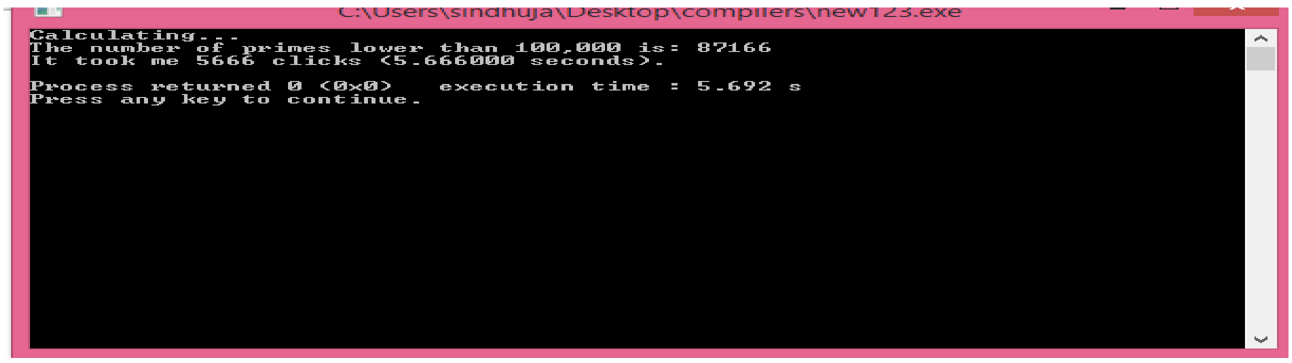
```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\sindhuja\Desktop\compilers\Borland C++\Borland\BCC55\Bin>bcc32 prime.cp
p
Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland
prime.cpp:
Turbo Incremental Link 5.00 Copyright (c) 1997, 2000 Borland

C:\Users\sindhuja\Desktop\compilers\Borland C++\Borland\BCC55\Bin>prime.exe
Calculating...
The number of primes lower than 100,000 is: 87166
It took me 6453 clicks (6.453000 seconds).
1000.000000
C:\Users\sindhuja\Desktop\compilers\Borland C++\Borland\BCC55\Bin>
```

Fig 1:Snapshot of prime no code when executed in the Borland c++ compiler.



```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\sindhuja\Desktop\compilers\dm\bin>dmc prime.cpp -I\Users\sindhuja\Deskt
op\compilers\dm\stlport\stlport\
link prime,,,user32+kernel32/noi;


C:\Users\sindhuja\Desktop\compilers\dm\bin>prime.exe
Calculating...
The number of primes lower than 100,000 is: 87166
It took me 6047 clicks (6.047000 seconds).

C:\Users\sindhuja\Desktop\compilers\dm\bin>
```

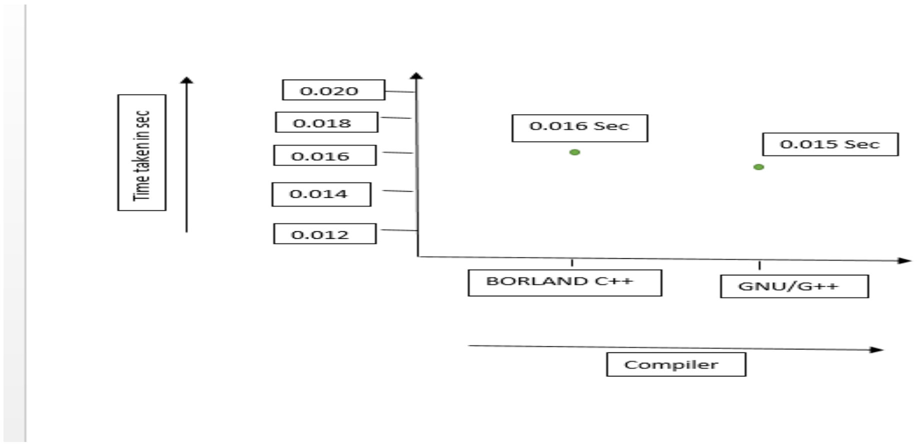
Fig 2: Snapshot of prime no code when executed in the Digital mars compiler.



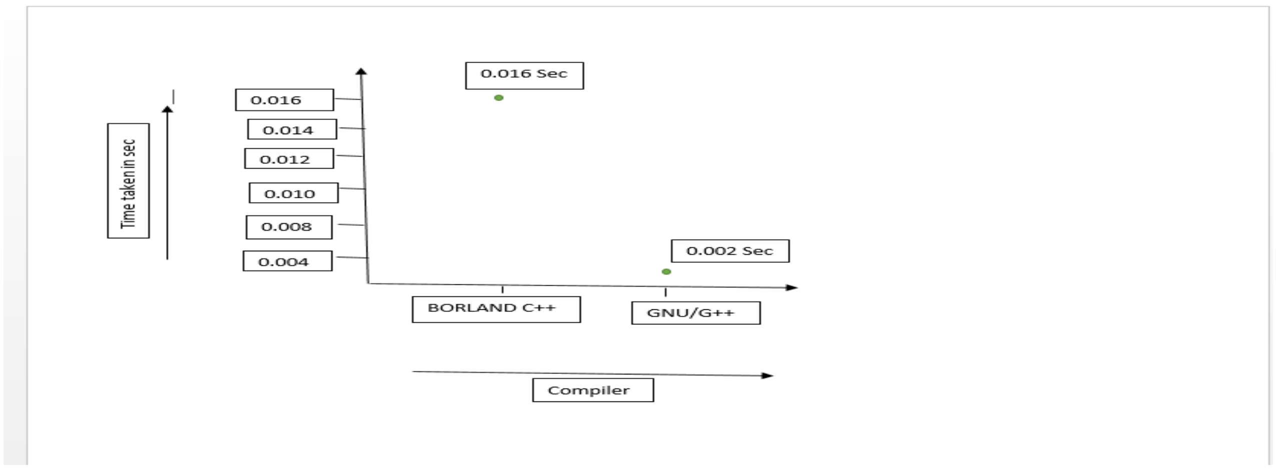
```
Calculating...
The number of primes lower than 100,000 is: 87166
It took me 5666 clicks (5.666000 seconds).

Process returned 0 (0x0)   execution time : 5.692 s
Press any key to continue.
```

Fig 3: Snapshot of prime no code when executed in the Gnu g++ compiler.

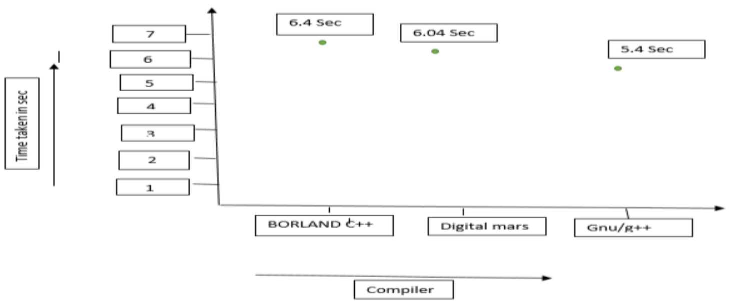
4.Results:



Graph 1: In the graph, time generated by 2 compilers have been plotted according to theseconds. Borland c++ generates 0.016 sec and gnu/g++ generates 0.015 sec(Dijkstra's without pointers)



Graph 2: In the graph, time generated by 2 compilers have been plotted according to theseconds. Borland c++ generates 0.015sec and gnu/g++ generates 0.002 sec.(Dijkstra's with pointers)



Graph 3. In the graph, time generated by 3 compilers have been plotted according to seconds.Borland generates 6.04sec,dm generates 6.04 sec and gnu/g++ generates 5.4 sec.

5.Comparision.

-In table 1.1 there is a comparision of the Borland c++ and gnu/g++ compiler with the dijistras algorithm (with no limits).

In table 1.2 there is a comparision of the Borland c++ and gnu/g++ compiler with the dijistras algorithm (with limits)eg:using pointers.
In the table 1.3 there is a comparision of the Borland c++,digital mars,gnu/g++ with prime no code taken to execute.

TABLE 1.1	Borland c++	Gnu/g++
Object file	9kb	2.8kb
Memory size	532kb	1mb
File generated after execution	4	2
Time taken to execute code	0.016sec	0.015sec

TABLE 1.2	Borland c++	Gnu/g++
Object file	2kb	3kb
Memory size	440kb	1mb
File generated after execution	3	2
Time taken to execute code	0.015sec	0.002sec

TABLE 1.3	Borland c++	Digital mars	Gnu/g++
Object file	2kb	1kb	2kb
Memory size	440kb	40kb	32kb
File generated after execution	3	3	3
Time taken to execute code	6.4sec	6.04	5.58sec

6. Conclusion:

When dijistras code is taken to compare the Borland c++ and gnu/g++ compiler.The time taken for the Borland compiler is 0.016 sec and for the gnu/g++ compiler it is 0.015 sec.The same digistras code when used pointers in the code generates 0.015 sec in Borland c++ and 0.002 sec in gnu/g++ compiler.Prime no program generates 6.40 sec in Borland c++ , 6.047 sec in digital mars and 5.4 sec in gnu/g++ compiler.By all these experiments it has been proved than gnu/g++ compiler has taken less time to execute the code.by all these experiments we can say that gnu/g++ is the best compiler.

References:

1.[http://insights.dice.com/2013/11/04/speed-test-comparing-intel-c-gnu-c-and-llvm-clang-compilers/]
2.[http://www.c4learn.com/c-programming/compiler-vs-interpreter/.]
3.[http://www.phoronix.com/scan.php?page=article&item=gcc-61-clang39&num=1]
4. Christian Menard, Andrés Goens, and Jeronimo Castrillon,"High-Level NoC Model for MPSoC Compilers" IEEE Dec 2016.