Evaluating the Relationship Between Running Times and DNA Sequence Sizes using a Generic-Based Filtering Program.

O.O. Oluwagbemi, M.Sc. [1*] **and A.C. Omonhinmin, M.Sc.**[2]

[1]Department of Computer and Information Sciences, College of Science and Technology,
Covenant University. PMB 1023, Ogun State, Nigeria.
[2]Department of Biological Sciences, College of Science and Technology, Covenant University
PMB 1023, Ogun State, Nigeria.

*E-mail: gbemiseun@yahoo.com

ABSTRACT

Generic programming depends on the decomposition of programs into simpler components which may be developed separately and combined arbitrarily, subject only to well-defined interfaces. Bioinformatics deals with the application of computational techniques to data present in the Biological sciences. A genetic sequence is a succession of letters which represents the basic structure of a hypothetical DNA molecule, with the capacity to carry information. This research article studied the relationship between the running times of a generic-based filtering program and different samples of genetic sequences in an increasing order of magnitude. A graphical result was obtained to adequately depict this relationship. It was also discovered that the complexity of the generic tree program was O ($\log_2$ N). This research article provided one of the systematic approaches of generic programming to Bioinformatics, which could be instrumental in elucidating major discoveries in Bioinformatics, as regards efficient data management and analysis.

(Keywords: generic programming, DNA sequences, trend analysis, computer science, bioinformatics)

INTRODUCTION

One of the major problems with modern data management rests in some biological databases which contain redundant DNA sequences. The major disadvantage of this is that, it slows down the pace of research activities on such raw data, thus making it difficult to obtain an optimal and quality interpretation of the data. Thus, such data has to be filtered out to eliminate sequences that have multiple occurrences. Programming, as one of the sub-fields in Computer Science, finds relevance in solving this particular problem.

The knowledge of generic programming is thus deployed in filtering some typical samples of biological DNA sequence data and the running times analyzed in order to study the effects of the generic filtering program on different samples of DNA sequence. This will better prepare and assist biological scientists to undertake research work which involves large volumes of DNA sequence data. It will also give them an understanding about the minimum time required to get such data ready for thorough and in depth research work. Finally, it will help to minimize error occurrences in the final research results.

The purpose of this research article is to develop a generic program for filtering redundant DNA sequence data from a database and study the relationship between the number of characters present in each data sample and the total running time taken by the generic filtering program to filter meaningful data from such sequence.

A DNA sequence constitutes the heritable genetic information in nuclei, plasmids, mitochondria, and chloroplasts that forms the basis for the developmental programs of all living organisms The possible letters of a DNA sequence are A, C, G, and T, representing the four nucleotide subunits of a DNA strand - adenine, cytosine, guanine, and thymine bases covalently linked to phospho-backbone.

For instance, the sequences are printed abutting one another without gaps, as in the sequence AAAGTCTGAC, going from 5' to 3' from left to right. A succession of any number of nucleotides greater than four is liable to be called a sequence. These sequences can be derived from the biological raw material through a process called DNA sequencing.

DNA sequencing is an aspect of Bioinformatics which encompasses the biochemical methods for determining the order of the nucleotide bases, adenine, guanine, cytosine, and thymine, in a DNA oligonucleotide.

Determining the DNA sequence is therefore important and useful in basic research studying the fundamental biological processes in different organisms, as well as in applied fields such as forensic research.

The rate of biological research and discovery has been significantly accelerated by the advent of DNA sequencing. The rapid speed of sequencing attainable with modern DNA sequencing technology has been instrumental in the large-scale sequencing of the human genome. Furthermore, related projects, often by scientific collaboration across continents, have helped a great deal in generating the complete DNA sequences of many animals, plants, and microbial genomes.

Generic programming is a programming technique which aims to make programs more adaptable by making them more general. Thus, generic programming is an attractive paradigm for scientific numerical components [5].

Generic programming has wide applicability in the fields of Bioinformatics. Bioinformatics refers to the application of information technology to the management and analysis of biological data; which has implications in diverse areas, ranging from artificial intelligence and robotics to genome analysis. [7] Presently, one of the most famous and relevant generic library in bioinformatics is the BTL (Bioinformatics Template Library), which is a very useful tool in Bioinformatics.

A research work conducted by W.R. Pitt et al., helped to describe an extension of the C++ standard library to include additional software components that are of use in the areas of bioinformatics and molecular modeling. [6]. Another research work was carried out by Andreas Döring et al., in which a new efficient generic C++ software library known as SeqAn was developed which was characterized with an efficient and generic design that addresses a wide range of problems in sequence analysis. SeqAn is under active development and it is hoped that it will become one of the standard platforms for algorithmic engineering at the interface of stringology, algorithm design, and computational biology [1].

Several other works have been done as regards generic programming applications; Kevin L. Howe et al., implemented a method in a program, called GAZE which was used for assembling arbitrary evidence for individual gene components (features) into predictions of complete gene structures. The system used was generic in that both the features themselves, and the model of gene structure against which potential assemblies were validated and scored, were external to the system and supplied by the user [4].Another research work was conducted by Benny Shomer [2], which helped in describing a generic, object oriented data submission system, entirely based on the Python programming language. This system could be easily accommodated to serve several data submission schemes with a relatively short development and Implementation time.

The research work carried out by Kenneth J. B et al., reported the complete thermodynamic library of all 10 Watson-Crick DNA nearest-neighbor interactions. The relevant thermodynamic data from calorimetric studies on 19 DNA oligomers and 9 DNA polymers was obtained. It was shown how these thermodynamic data could be used to calculate the stability and predict the temperature-dependent behavior of any DNA duplex structure from knowledge of its base sequence [3].

Our research work helped to study the effects of a generic program in extracting or filtering out specific DNA sequence thus, eliminating redundant occurrences of such sequence in a typical biological database.

This research paper is structured as follows; the introductory aspect gives a proper description of the DNA sequence, the generic programming concept, and a brief literature review about the varied applications of generic programming to bioinformatics related data. Secondly, the materials and methods section includes data collection, data processing and the various tools employed in processing and analyzing the different set of genetic sequence samples. Thirdly, as a means of contributing to existing knowledge, this research paper also highlights the implementation of a generic program on different samples of genetic sequence in order to study the relationship between the number of character sequence in each data sample and the running time taken by the generic program in filtering out the relevant data. The fourth section presents the results and discussions. The final section of this paper provides the conclusion, acknowledgements, and recommendations.

MATERIALS AND METHODS

Data Collection

The data used in this research work was self-generated but took the format of typical DNA sequence in some DNA sequence database. It was self-generated in order not to discredit any DNA database. Fifteen different samples were used for this study. Samples 1, 2, 3, 4, 5, 6, 7,……..15 were used for data collection (different samples of DNA sequences).

Implementation

The programming language used in implementing the generic-based filtering program was the Java programming language. The Net bean 5.5.1 Integrated development environment was used to compile and run the program at different instances of sample data.

The corresponding reading for each sample data was taken. This includes the compile time and the running time for each DNA sequence sample.

Java Programming language (Java Code)

```
/*
 * GenericDnaSequenceFilter.java
 * Created on April 26, 2008, 5:51 AM
 */

/**
 * @author Oluwagbemi Olugbenga Oluseun
 * Covenant University
 * College of Science and Technology
 * Department of Computer and Information Science
 * PMB 1023, Ota, Ogun State
 * Nigeria
 * West Africa
 */
import java.util.*;

public class GenericDnaSequenceFilter {

    public static void main(String[] args) {
      SortedSet<Item> origin = new TreeSet<Item>();
      origin.add(new Item("", 1));
      origin.add(new Item("", 2));
      origin.add(new Item("", 3));
      origin.add(new Item("", 6));
      origin.add(new Item("", 27));
      origin.add(new Item("", 80));
      origin.add(new Item("",49));
      origin.add(new Item("",95));
      origin.add(new Item("",1002));
      origin.add(new Item("",2095));
      origin.add(new Item("",3449));
      origin.add(new Item("",5095));
      System.out.println(origin);

      SortedSet<Item>    sortByDescription    =    new
TreeSet<Item>(new Comparator<Item>()
      {
          public int compare(Item a, Item b)
            {
                String descrA = a.getDescription();
                String descrB = b.getDescription();
                return descrA.compareTo(descrB);
            }
      });
      sortByDescription.addAll(origin);
      System.out.println(sortByDescription);
    }
}
class Item implements Comparable<Item>
  {
      public Item(String aDescription, int aPartNumber)
     {
        description = aDescription ;
         origin = aPartNumber;
      }
          public String getDescription()
                {
                    return description;
     }
    public String toString()
      {
      return "[description=" + description + ", origin=" +
origin + "]";
         }
    public boolean equals(Object otherObject)
         {
       if (this == otherObject) return true;
       if (otherObject == null) return false;
```

```
          if (getClass() != otherObject.getClass()) return false;
          Item other =(Item) otherObject;
          return description.equals(other.description)
          && origin == other.origin;
                    }

      public int hashCode()
              {
         return 13 * description.hashCode() + 17 * origin;
                    }

      public int compareTo(Item other)
               {
        return origin - other.origin;
           }
       private String description;
      private int origin;
    }
```

RESULTS

This section shows the output generated by the Java-based generic extraction and filtering program for each of the DNA sequence sample used in this research work.

The results for the first seven samples ,Sample 1, Sample 2, Sample 3, Sample 4, Sample 5, Sample 6, Sample 7, were generated here as a concrete proof . The other extra 13 results were tabulated in the results section.

Results for Sample 1 (Compile time result)

```
   init:
   deps-jar:
   Compiling 1 source file to C:\Documents and
Settings\Gbenga\My Documents\Sample Java
Programs\GenericDnaSequenceFilter\build\classes
   compile-single:
   BUILD SUCCESSFUL (total time: 3 seconds)
   Results for Sample 1(Run time result)
   init:
   deps-jar:
   compile-single:
   run-single:
   [[description=GACTGACTGACTGACTGACT, origin=1],
    [description=ACTGACTGACTGACTGACTGACTG,
origin=2],
[description=AAAAGGGGAAAAGGGGTTTTCCCC,
origin=3],
   [description=GACTGACTGACTGACTGACT, origin=6],
   [description=ACTGACTGACTGACTGACTGACTG,
origin=27],
   [description=GACTGACTGACTGACTGACT, origin=49],
   [description=AAAAGGGGAAAAGGGGTTTTCCCC,
origin=80],
[description=ACTGACTGACTGACTGACTGACTG,
origin=95],
[description=AAAAGGGGAAAAGGGGTTTTCCCC,
origin=1002]]
```

Extracted or filtered DNA sequence

```
   [[description=AAAAGGGGAAAAGGGGTTTTCCCC,
origin=3],
[description=ACTGACTGACTGACTGACTGACTG,
origin=2],
   [description=GACTGACTGACTGACTGACT, origin=1]]
   BUILD SUCCESSFUL (total time: 5 seconds)
```

Results for Sample 2 (Compile time result)

```
   init:
   deps-jar:
   Compiling 1 source file to C:\Documents and
Settings\Gbenga\My Documents\Sample Java
Programs\GenericDnaSequenceFilter\build\classes
   compile-single:
   BUILD SUCCESSFUL (total time: 2 seconds)
```

Results for Sample 2 (Run time result)

```
   init:
   deps-jar:
   compile-single:
   run-single:
   [[description=GGAATTCCGGAATTCCGGAATTCCGGAA
TTCC,                                origin=1],
[description=GAGATATACACAGAGACACATATAAAA,
origin=2],
[description=GATTGATTGATTGATTGATTGATTGATTGAT
T,                                   origin=3],
[description=GGAATTCCGGAATTCCGGAATTCCGGAATT
CC,                                  origin=6],
[description=GAGATATACACAGAGACACATATAAAA,
origin=27],
[description=GGAATTCCGGAATTCCGGAATTCCGGAATT
CC,                                 origin=49],
[description=GGAATTCCGGAATTCCGGAATTCCGGAATT
CC,                                 origin=55],
[description=GATTGATTGATTGATTGATTGATTGATTGAT
T, origin=80],
[description=GAGATATACACAGAGACACATATAAAA,
origin=95],
[description=GAGATATACACAGAGACACATATAAAA,
origin=105],
[description=GATTGATTGATTGATTGATTGATTGATTGATT
, origin=1002],
[description=GATTGATTGATTGATTGATTGATTGATTGATT
, origin=1500]]
```

Extracted or filtered DNA sequence

```
   [[description=GAGATATACACAGAGACACATATAAAA,
origin=2],
[description=GATTGATTGATTGATTGATTGATTGATTGATT
, origin=3],
[description=GGAATTCCGGAATTCCGGAATTCCGGAATTC
C, origin=1]]
   BUILD SUCCESSFUL (total time: 4 seconds)
   Results for Sample 3 (Compile time result)
   init:
   deps-jar:
   Compiling 1 source file to C:\Documents and
Settings\Gbenga\My Documents\Sample Java
Programs\GenericDnaSequenceFilter\build\classes
   compile-single:
   BUILD SUCCESSFUL (total time: 3 seconds)
```

Results for Sample 3 (Run time result)

```
   init:
   deps-jar:
   compile-single:
   run-single:
   [[description=GGGGTTTTTTAAAACCCCTACGTACG,
origin=1],
[description=AAGTCAAGTCTAAGTCAACTCAACTC,
origin=2],
[description=ACTGACTCACTGTTTCCCGGGAAATTT,
origin=3],
[description=GGGGTTTTTTAAAACCCCTACGTACG,
origin=6],
[description=AAGTCAAGTCTAAGTCAACTCAACTC,
origin=27],
[description=GGGGTTTTTTAAAACCCCTACGTACG,
origin=49],
[description=GGGGTTTTTTAAAACCCCTACGTACG,
origin=55],
[description=CGCGCGCGAAACGCGCGAATTTCGCG,
origin=80],
[description=GGGGTTTTTTAAAACCCCTACGTACG,
origin=85],
[description=AAGTCAAGTCTAAGTCAACTCAACTC,
origin=95],
[description=AAGTCAAGTCTAAGTCAACTCAACTC,
origin=105],
[description=AAGTCAAGTCTAAGTCAACTCAACTC,
origin=207],
[description=AAGTCAAGTCTAAGTCAACTCAACTC,
origin=501],
[description=ACTGACTCACTGTTTCCCGGGAAATTT,
origin=1002],
[description=CGCGCGCGAAACGCGCGAATTTCGCG,
origin=1500],
[description=GGGGTTTTTTAAAACCCCTACGTACG,
origin=2000],
[description=ACTGACTCACTGTTTCCCGGGAAATTT,
origin=3000],
[description=CGCGCGCGAAACGCGCGAATTTCGCG,
origin=15000]]
```

Extracted or filtered DNA sequence

```
   [[description=AAGTCAAGTCTAAGTCAACTCAACTC,
origin=2],
[description=ACTGACTCACTGTTTCCCGGGAAATTT,
origin=3],
[description=CGCGCGCGAAACGCGCGAATTTCGCG,
origin=80],
[description=GGGGTTTTTTAAAACCCCTACGTACG,
origin=1]]
   BUILD SUCCESSFUL (total time: 5 seconds)
```

Results for Sample 4 (Compile time result)

```
   init:
   deps-jar:
   Compiling 1 source file to C:\Documents and
Settings\Gbenga\My Documents\Sample Java
Programs\GenericDnaSequenceFilter\build\classes
   compile-single:
   BUILD SUCCESSFUL (total time: 5 seconds)
```

Results for Sample 4 (Run time result)

```
   init:
   deps-jar:
   compile-single:
   run-single:
   [[description=AAAGGGTTCCAAAGGGTTCCAAATG,
origin=1],
[description=GTGTGTGTAACCGTGTAACCGTGTAACGT,
origin=2],
[description=ACTGACTGACTGACTGACTGACTGACTG,
origin=3],
[description=AGTCAGTCAGTCAGTCAGTCAGTCAGTCAGT
CAGTCAGTCAGGTC,                    origin=6],
[description=GGGGGTTTTTAAAAATTTTTCCCCCCAGAGA
TATACACATGACTA,                   origin=27],
[description=AAAGGGTTCCAAAGGGTTCCAAATG,
origin=49],
[description=AGTCAGTCAGTCAGTCAGTCAGTCAGTCAGT
CAGTCAGTCAGGTC,                   origin=55],
[description=AGCTTTTTCCCCGGGGGAGAGAGAGAGAGA
GAGAGAG,                          origin=80],
[description=AGCTTTTTCCCCGGGGGAGAGAGAGAGAGA
GAGAGAG,                          origin=85],
[description=GTGTGTGTAACCGTGTAACCGTGTAACGT,
origin=95],
[description=GGGGGTTTTTAAAAATTTTTCCCCCCAGAGA
TATACACATGACTA,                  origin=105],
[description=AGTCAGTCAGTCAGTCAGTCAGTCAGTCAGT
CAGTCAGTCAGGTC,                  origin=207],
[description=ACTGACTGACTGACTGACTGACTGACTG,
origin=1002],
[description=AGCTTTTTCCCCGGGGGAGAGAGAGAGAGA
GAGAGAG,                        origin=1500],
[description=ACTGACTGACTGACTGACTGACTGACTG,
origin=2000],
[description=GGGGGTTTTTAAAAATTTTTCCCCCCAGAGA
TATACACATGACTA, origin=3000]]
```

Extracted or filtered DNA sequence

```
    [[description=AAAGGGTTCCAAAGGGTTCCAAATG,
origin=1],
[description=ACTGACTGACTGACTGACTGACTGACTG,
origin=3],
[description=AGCTTTTTCCCCGGGGGAGAGAGAGAGAGA
GAGAGAG,                          origin=80],
[description=AGTCAGTCAGTCAGTCAGTCAGTCAGTCAGT
CAGTCAGTCAGGTC,                    origin=6],
[description=GGGGGTTTTTAAAAATTTTTCCCCCCAGAGA
TATACACATGACTA,                   origin=27],
[description=GTGTGTGTAACCGTGTAACCGTGTAACGT,
origin=2]]
BUILD SUCCESSFUL (total time: 5 seconds)
```

Results for Sample 5 (Compile time result)

```
init:
deps-jar:
Compiling 1 source file to C:\Documents and
Settings\Gbenga\My Documents\Sample Java
Programs\GenericDnaSequenceFilter\build\classes
compile-single:
BUILD SUCCESSFUL (total time: 3 seconds)
```

Results for Sample 5 (Run time result)

```
init:
deps-jar:
compile-single:
run-single:
[[description=GGGGTTTTTTAAAACCCCTACGTACG,
origin=1],
[description=AAGTCAAGTCTAAGTCAACTCAACTC,
origin=2],
[description=CGCGCGCGAAACGCGCGAATTTCGCG,
origin=3],
[description=ACTGACTGACTGACTGACTGACTGACTG,
origin=6],
[description=AGTCAGTCAGTCAGTCAGTCAGTCAGTCAG
TCAGTCAGTCAGGTC,                  origin=27],
[description=AGCTTTTTCCCCGGGGGAGAGAGAGAGAG
AGAGAGAG,                         origin=49],
[description=CGCGCGCGAAACGCGCGAATTTCGCG,
origin=55],
[description=GGGGGTTTTTAAAAATTTTTCCCCCCAGAG
ATATACACATGACTA,                  origin=80],
[description=GGGGGTTTTTAAAAATTTTTCCCCCCAGAG
ATATACACATGACTA,                  origin=85],
[description=GGGGTTTTTTAAAACCCCTACGTACG,
origin=95],
[description=ACTGACTGACTGACTGACTGACTGACTG,
origin=105],
[description=AGCTTTTTCCCCGGGGGAGAGAGAGAGAG
AGAGAGAG,                        origin=207],
[description=AAGTCAAGTCTAAGTCAACTCAACTC,
origin=1002],
[description=AGTCAGTCAGTCAGTCAGTCAGTCAGTCAG
TCAGTCAGTCAGGTC,                origin=1500],
[description=GGGGGTTTTTAAAAATTTTTCCCCCCAGAG
ATATACACATGACTA,                origin=2000],
[description=AGTCAGTCAGTCAGTCAGTCAGTCAGTCAG
TCAGTCAGTCAGGTC,                origin=3000],
[description=AGCTTTTTCCCCGGGGGAGAGAGAGAGAG
AGAGAGAG,                       origin=5000],
[description=GGGGGTTTTTAAAAATTTTTCCCCCCAGAG
ATATACACATGACTA, origin=6000],
```

Extracted or filtered sequence

```
    [description=AAGTCAAGTCTAAGTCAACTCAACTC,
origin=2],
[description=ACTGACTGACTGACTGACTGACTGACTG,
origin=6],
[description=AGCTTTTTCCCCGGGGGAGAGAGAGAGAGA
GAGAGAG,                              origin=49],
[description=AGTCAGTCAGTCAGTCAGTCAGTCAGTCAGT
CAGTCAGTCAGGTC,                       origin=27],
[description=CGCGCGCGAAACGCGCGAATTTCGCG,
origin=3],
[description=GGGGGTTTTTAAAAATTTTTCCCCCCAGAGA
TATACACATGACTA,                       origin=80],
[description=GGGGTTTTTTAAAACCCCTACGTACG,
origin=1]]
    BUILD SUCCESSFUL (total time: 15 seconds)
```

Results for Sample 6 (Compile time results)

```
    init:
    deps-jar:
    Compiling 1 source file to C:\Documents and
Settings\Gbenga\My Documents\Sample Java
Programs\GenericDnaSequenceFilter\build\classes
    compile-single:
    BUILD SUCCESSFUL (total time: 9 seconds)
```

Results for Sample 6 (Run time results)

```
    init:
    deps-jar:
    compile-single:
    run-single:
    [[description=GTGTGTGTGTGTGTGTGAGAGAGCGCGC
GCGCGCGCGCGCGCG,                       origin=1],
[description=ATGATAGATAGATAGATAGATAGATAGATAG
ATAGATAGATAG,                          origin=2],
[description=CGCTCGCTCGCTCGAGCTCGCTAGCTCGCT
AGCTCGCTAGCTAGC,                       origin=3],
[description=TAGTAGATAGATAGATAGATAGATAGATAGA
TAGATAGATCCCT,                        origin=27],
[description=ACGTAGTCGTAGCTAGTAGCTAGCTAGCTA
GTAGCTAGTAGCTAG,                      origin=49],
[description=GTGTGTGTGTGTGTGTGAGAGAGCGCGCGC
GCGCGCGCGCGCG,                        origin=55],
[description=TTGGAACCGGTTAACCGGTTCCAAGGTTCC
AAGGTTCCAAGGTTA,                      origin=80],
[description=CTCTCGAAAGTCTCGGGATTCTCTCGAGAG
CTCTCTAGAGCTAGC,                      origin=85],
[description=CTCTCGAAAGTCTCGGGATTCTCTCGAGAG
CTCTCTAGAGCTAGC,                      origin=95],
[description=ATGATAGATAGATAGATAGATAGATAGATA
GATAGATAGATAG,                       origin=105],
[description=TTGGAACCGGTTAACCGGTTCCAAGGTTCC
AAGGTTCCAAGGTTA,                     origin=207],
[description=GATCCGATTCGGATTCGGACCCGGATTCGA
TGATGCTAGGCTGAT,                    origin=1002],
[description=CGCTCGCTCGCTCGAGCTCGCTAGCTCGC
TAGCTCGCTAGCTAGC,                   origin=1500],
[description=TAGTAGATAGATAGATAGATAGATAGATAG
ATAGATAGATCCCT,                     origin=2000],
[description=ACGTAGTCGTAGCTAGTAGCTAGCTAGCTA
GTAGCTAGTAGCTAG,                    origin=3000],
[description=GATCCGATTCGGATTCGGACCCGGATTCGA
TGATGCTAGGCTGAT,                    origin=5000],
[description=CGCTCGCTCGCTCGAGCTCGCTAGCTCGC
TAGCTCGCTAGCTAGC,                   origin=5500],
[description=GTGTGTGTGTGTGTGTGAGAGAGCGCGCG
CGCGCGCGCGCGCG,                     origin=6000],
[description=TAGTAGATAGATAGATAGATAGATAGATAG
ATAGATAGATCCCT,                     origin=6500],
[description=ATGATAGATAGATAGATAGATAGATAGATA
GATAGATAGATAG,                      origin=7000],
[description=CTCTCGAAAGTCTCGGGATTCTCTCGAGAG
CTCTCTAGAGCTAGC,                    origin=8000],
[description=ACGTAGTCGTAGCTAGTAGCTAGCTAGCTA
GTAGCTAGTAGCTAG,                    origin=8500],
[description=GATCCGATTCGGATTCGGACCCGGATTCGA
TGATGCTAGGCTGAT,                    origin=9000],
[description=TTGGAACCGGTTAACCGGTTCCAAGGTTCC
AAGGTTCCAAGGTTA, origin=9500]]
```

Extracted or filtered DNA sequence

```
[[description=ACGTAGTCGTAGCTAGTAGCTAGCTAGCTA
GTAGCTAGTAGCTAG, origin=49],
[description=ATGATAGATAGATAGATAGATAGATAGATA
GATAGATAGATAG,                         origin=2],
[description=CGCTCGCTCGCTCGAGCTCGCTAGCTCGC
TAGCTCGCTAGCTAGC,                      origin=3],
[description=CTCTCGAAAGTCTCGGGATTCTCTCGAGAG
CTCTCTAGAGCTAGC,                      origin=85],
[description=GATCCGATTCGGATTCGGACCCGGATTCGA
TGATGCTAGGCTGAT,                    origin=1002],
[description=GTGTGTGTGTGTGTGTGAGAGAGCGCGCG
CGCGCGCGCGCGCG,                        origin=1],
[description=TAGTAGATAGATAGATAGATAGATAGATAG
ATAGATAGATCCCT,                       origin=27],
[description=TTGGAACCGGTTAACCGGTTCCAAGGTTCC
AAGGTTCCAAGGTTA, origin=80]]
BUILD SUCCESSFUL (total time: 16 seconds)
```

Results for Sample 7 (Compile time results)

```
init:
deps-jar:
Compiling 1 source file to C:\Documents and
Settings\Gbenga\My Documents\Sample Java
Programs\GenericDnaSequenceFilter\build\classes
compile-single:
BUILD SUCCESSFUL (total time: 22 seconds)
```

Results for Sample 7 (Run time results)

```
init:
deps-jar:
compile-single:
run-single:
[[description=GAGAGAGAGAGAGAGTAGTAGGAGAGATC
GTCGATCGATCGATC,                        origin=1],
[description=GCACGACAGACAGACACGACACTTTTTGTGT
ACGACAGACAGAC,                          origin=2],
[description=CGACTCGTAGCTAGCTAGCTAGCTAGCTAG
CTAGCTAGCTAGCTA,                        origin=3],
[description=GCAAAGTTTTAGGGGCAGCTTAGCCAGCTA
GCTAGCTAGCTAGCT,                       origin=27],
[description=GCACGACAGACAGACACGACACTTTTTGTGT
ACGACAGACAGAC,                         origin=49],
[description=GCAAAGTTTTAGGGGCAGCTTAGCCAGCTA
GCTAGCTAGCTAGCT,                       origin=55],
[description=GAGAGAGAGAGAGAGTAGTAGGAGAGATCG
TCGATCGATCGATC,                        origin=80],
[description=CGACTCGTAGCTAGCTAGCTAGCTAGCTAG
CTAGCTAGCTAGCTA,                       origin=85],
[description=CGACTCGTAGCTAGCTAGCTAGCTAGCTAG
CTAGCTAGCTAGCTA,                       origin=95],
[description=GCACGACAGACAGACACGACACTTTTTGTGT
ACGACAGACAGAC,                        origin=105],
[description=GAGAGAGAGAGAGAGTAGTAGGAGAGATCG
TCGATCGATCGATC,                       origin=207],
[description=GCAAAGTTTTAGGGGCAGCTTAGCCAGCTA
GCTAGCTAGCTAGCT,                     origin=1002],
[description=CGACTCGTAGCTAGCTAGCTAGCTAGCTAG
CTAGCTAGCTAGCTA,                     origin=1500],
[description=GCAAAGTTTTAGGGGCAGCTTAGCCAGCTA
GCTAGCTAGCTAGCT,                     origin=2000],
[description=GCACGACAGACAGACACGACACTTTTTGTGT
ACGACAGACAGAC,                       origin=3000],
[description=GAGAGAGAGAGAGAGTAGTAGGAGAGATCG
TCGATCGATCGATC,                      origin=4000],
[description=GCACGACAGACAGACACGACACTTTTTGTGT
ACGACAGACAGAC,                       origin=4500],
[description=GCAAAGTTTTAGGGGCAGCTTAGCCAGCTA
GCTAGCTAGCTAGCT,                     origin=5000],
[description=CGACTCGTAGCTAGCTAGCTAGCTAGCTA
GCTAGCTAGCTAGCTA,                    origin=5500],
[description=CGACTCGTAGCTAGCTAGCTAGCTAGCTAG
CTAGCTAGCTAGCTA,                     origin=5700],
[description=GAGAGAGAGAGAGAGTAGTAGGAGAGATC
GTCGATCGATCGATC,                     origin=6000],
[description=GCAAAGTTTTAGGGGCAGCTTAGCCAGCTA
GCTAGCTAGCTAGCT,                     origin=6500],
[description=GCAAAGTTTTAGGGGCAGCTTAGCCAGCTA
GCTAGCTAGCTAGCT,                     origin=6700],
[description=GCACGACAGACAGACACGACACTTTTTGTG
TACGACAGACAGAC,                      origin=7000],
[description=CGACTCGTAGCTAGCTAGCTAGCTAGCTAG
CTAGCTAGCTAGCTA,                     origin=8000],
[description=GCACGACAGACAGACACGACACTTTTTGTG
TACGACAGACAGAC,                      origin=8500],
[description=GCAAAGTTTTAGGGGCAGCTTAGCCAGCTA
GCTAGCTAGCTAGCT,                     origin=9000],
[description=GAGAGAGAGAGAGAGTAGTAGGAGAGATC
GTCGATCGATCGATC,                     origin=9500],
[description=GAGAGAGAGAGAGAGTAGTAGGAGAGAT
CGTCGATCGATCGATC,                    origin=9900],
[description=CGACTCGTAGCTAGCTAGCTAGCTAGCTAG
CTAGCTAGCTAGCTA,                    origin=18000],
[description=GCACGACAGACAGACACGACACTTTTTGTG
TACGACAGACAGAC,                     origin=18500],
[description=GCAAAGTTTTAGGGGCAGCTTAGCCAGCTA
GCTAGCTAGCTAGCT, origin=19000]]
```

Extracted or filtered DNA sequence

```
[[description=CGACTCGTAGCTAGCTAGCTAGCTAGCTA
GCTAGCTAGCTAGCTA,                       origin=3],
[description=GAGAGAGAGAGAGAGTAGTAGGAGAGATC
GTCGATCGATCGATC,                        origin=1],
[description=GCAAAGTTTTAGGGGCAGCTTAGCCAGCTA
GCTAGCTAGCTAGCT,                       origin=27],
[description=GCACGACAGACAGACACGACACTTTTTGTG
TACGACAGACAGAC, origin=2]]
BUILD SUCCESSFUL (total time: 6 seconds)
```

DISCUSSION

Table 1 shows the results generated from the generic program. It shows the samples types, the number of string characters in the DNA sequence, the compile times, running times and the total running times of the generic program on each sample of DNA sequence.

Increasing sizes of DNA sequence was fed into the generic filtering program. It was observed that the total running times of the generic program on different DNA sequence samples differed. The total running time also increased with corresponding increase in the size of each DNA sequence sample. This continued until a certain point which formed the peak of the running time readings; immediately after the peak, the running time started decreasing, with increase in the size of DNA sample sequences. The results from the generic program output are summarized below in Table 2.

Table 1

DNA Samples	DNA sequence character nos	Compile time (seconds)	Running Time (Seconds)	Total Running Time (Seconds)
1	204	3	5	8
2	364	2	4	6
3	471	3	5	8
4	573	5	5	10
5	638	3	15	18
6	1065	9	16	25
7	1424	22	6	28
8	1474	12	19	31
9	1720	4	36	40
10	2035	4	23	27
11	2105	11	9	20
12	2370	18	6	24
13	2541	6	4	10
14	2784	3	6	9
15	2803	14	7	21

Table 2

DNA Sample	DNA sequence character nos	Total Running Time(Seconds
1	204	8
2	364	6
3	471	8
4	573	10
5	638	18
6	1065	25
7	1424	28
8	1474	31
9	1720	40
10	2035	27
11	2105	20
12	2370	24
13	2541	10
14	2784	9
15	2803	21

Graphical analyses to depict the relationship between the running times of the generic program and the DNA sequence sizes was undertaken using Microsoft Excel (Figures 1 and 2).

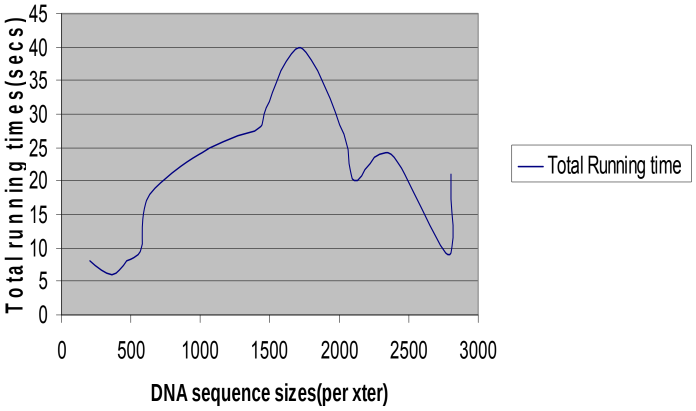

Figure 1: Graphical Analysis Depicting the Relationship between Running Times of the Generic-Based Filtering Program and the DNA Sequence Sizes (O.O. Oluwagberri).

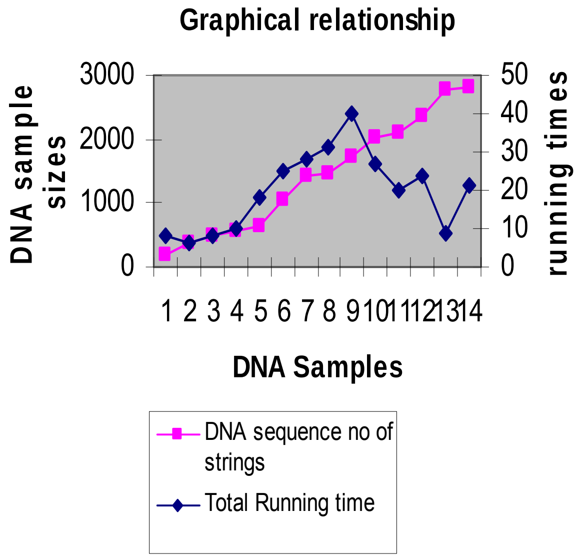

Figure 2: A Classic Combination Chart to show the Relationship Between the Generic-Based Filtering Program and the DNA Sequence Sizes.

Complexity of the generic-based tree program

From the results of the total running times generated from the experiment carried out in this research work, it can be inferred that the running time of this generic program is approximately O(log2 N). This means that if the generic tree program contains N elements, then an average of log2 N comparisons (iterations) are required to find the correct position for the new element. Thus, the complexity is O (log2 N).

Proof of the asymptotic bounds for the generic tree program

A generic tree which contains n internal nodes has a height of O (log2 (n)).

Definitions:

h (v) = height of subtree rooted at node v
bh (v) = the number of black nodes (not counting v if it is black) from v to any leaf in the subtree (called the black-height).

Lemma: A subtree rooted at node v has at least 2bh (v) − 1 internal nodes.

Proof of Lemma (by induction height):

Basis: h (v) = 0

If v has a height of zero then it must be null, therefore bh (v) = 0. So:

2bh (v) − 1 = 20 − 1 = 1 − 1 = 0

Inductive Step: v such that h (v) = k, has 2bh (v) − 1 internal nodes implies that v' such that h (v') = k+1 has 2bh (v') − 1 internal nodes.

Since v' has h (v') > 0 it is an internal node. As such it has two children which have a black-height of either bh (v') or bh (v')-1 (depending on whether v' is red or black). By the inductive hypothesis each child has at least 2bh (v') − 1 − 1 internal nodes, so v' has:

2bh (v') − 1 − 1 + 2bh (v') − 1 − 1 + 1 = 2bh (v') − 1 internal nodes.

Using this lemma, we can now show that the height of the tree is logarithmic. Since at least half of the nodes on any path from the root to a leaf are black, the black-height of the root is at least h (root)/2.

By the lemma we get:

$n >= 2^{\underline{h\ (root)\ /2}}$ -1 <---> $\log_2 (n+1) >= \overset{2}{\underline{h\ (root)}}$
←→ $h\ (root) <= 2\log_2 (n+1)$

2
Therefore, the height of the root is O (log 2 (n)) [8].

CONCLUSION

In conclusion, the results obtained from this research (as shown in Tables 1 and 2) illustrate that the running times in extracting or filtering out redundant DNA sequences from a typical database increased with a corresponding increase in the size of each data sample. At peak performance, the running time started decreasing with an increase in DNA sample size. These results can be a very important tool in elucidating major research interests in the field of Bioinformatics, especially efficiency in the area of time taken to process a given amount of biological data, thus eliminating redundancies and minimizing the time spent on the overall research.

ACKNOWLEDGEMENT

The authors would like to acknowledge the Chancellor of Covenant University, Bishop Dr. David Oyedepo, for his stimulating words of encouragement to promote research with new and positive discoveries. Thanks also go to Professor E.F. Adebiyi for his good mentoring role as a Ph.D. supervisor.

REFERENCES

1. Döring, A,. David, W., Tobias, R., and Knut, R. 2008. “SeqAn: An Efficient, Generic C++ Library for Sequence Analysis”. *BMC Bioinformatics*. 9(11). http://www.biomedcentral.com/1471-2105/9/11.

2. Shomer, B. 1997. “A Generic Data Collection System Through WWWforms, Based on a Python OOD Program, EMBL Outstation”. The European

Bioinformatics Institute, Hinxton Hall, Hinxton: Cambridge, UK,.http://www.ebi.ac.uk/

3. Breslauer, K.J., R. Franks, H. Blockers, and L.A. Marky. 2004. “Predicting DNA Duplex Stability from the Base Sequence”. *Proc. Natl. Acad. Sci (Biochemistry)*. 83(June):3746-3750.

4. Howe, K.L., T. Chothia, and R. Durbin. 2002. “GAZE: A Generic Framework for the Integration of Gene-Prediction Data by Dynamic Programming”, *Genome Research*. 12:1418-1427. http://www.sanger.ac.uk/Software/analysis/GAZE

5. Lie-Quan, Lee and A. Lumsdaine. 2005. “Generic Programming for High-Performance Scientific Applications, Research Articles, Concurrency and Computation: Practice & Experience,” *Conference Proceedings of 2002 ACM Java Grande–ISCOPE Conference Part II*. 17(7-8): 941 – 965. ISSN: 1532-0626.

6. [6] Pitt W. R., Williams M. A. , Steven M., Sweeney B., Bleasby A. J. and Moss D. S., 2001 “The Bioinformatics Template Library—generic components for biocomputing”, BMC Bioinformatics Vol. 17 no. 8, pp 729–737.

7. Attwood, T.K. and D.J. Parry-Smith. 1999. *Cell and Molecular Biology in Action Series: Introduction to Bioinformatics*. Pearson Education Limited: Essex, U.K.

8. Cormen, T.H., C.E. Leiserson, R.L. Rivest, and C. Stein. 2001. *Introduction to Algorithms, Second Edition*. MIT Press: Boston, MA. .273–301

ABOUT THE AUTHORS

O.O Oluwagbemi holds a B.Sc. in Computer Science from the University of Ilorin, Ilorin, Nigeria. He also holds an M.Sc. degree in Computer Science from the University of Ibadan, Ibadan , Nigeria. He is presently pursuing a Ph.D. program in Computer Science and also serves as a lecturer at Covenant University, Ota, Ogun State, Nigeria. He is a member of the American Association for the Advancement of Science, the International Society for Computational Biology, and the African Society for Bioinformatics and Computational Biology. He is also a member of the Nigerian Society of Computer Science. His research areas include among others, modeling and simulations, software modeling and engineering, bioinformatics, and neural networks.

A.C Omonhinmin holds a B.Sc. degree in Botany from the University of Benin, Benin, Nigeria. He also holds am M.Sc. degree in Biosystematic and Plant Genetic Resource Management from the University of Benin. He is presently a Ph.D. candidate at the University of Benin and also serves as a lecturer at Covenant University, Ota, Ogun State, Nigeria.

SUGGESTED CITATION

Olwuagbemi, O.O. and A.C. Omonhinmin. 2008. “Evaluating the Relationship Between Running Times and DNA Sequence Sizes using a Generic-Based Filtering Program”. *Pacific Journal of Science and Technology*. 9(2):656-666.

Pacific Journal of Science and Technology