Comparing Common Programming Languages to Parse Big XML File in Terms of Executing Time, Memory Usage, CPU Consumption and Line Number on Two Platforms

Ahmed Mahdee Abdo
Sardar Hasan Alali
Computer Science department, Zakho University, Kurdistan, Iraq

doi: 10.19044/esj.2016.v12n27p325 URL:http://dx.doi.org/10.19044/esj.2016.v12n27p325

Abstract

XML files are used widely to save the information especially in the field of bioinformatics about the whole genome. There are many programming languages and modules used to parse XML files in different platforms such as Linux, Macintosh and Windows. The aim of this report is to reveal and determine which common programming language to use and on which platform is better to parse XML files in terms of memory usage, CPU time consumption and executing time.

Keywords: XML, memory usage, CPU consumption and executing time.

Introduction

As many kinds of methods such as next generation technologies are used to sequence the whole genome, enormous amount of date became available. These dates are usually saved as text files. Extensible Markup Language (XML) files are kind of text files which are used to save much information about the whole genome.XML is a flexible text-based language that was developed by Standard Generalized Markup Language (SGML) to represent the information on the web server. XML is a document that represents the data as a structure which involves tags or entities and elements that are subset of these entities. Each element contains one or more attributes that define the method to process these elements. Moreover, XML files describe and define the relationship between the entities, elements and attributes that help the software to parse which part of the file (Bryan, 1998). XML files can be read, processed, written in any operating system which makes it easy to transfer and exchange XML files between different platforms. In addition, XML files can be easily converted to another format

types such as Comma Separated Values (CSV) file in order to be used for other purposes (Soo & Wei, 2002). Moreover, it boosts Unicode that make it easy and reliable to presents the majority kind of information and connect them together. Furthermore, XML files have the power to explore the most common types of data-structure that are used in computer science such as trees, lists and records. XML files can be used offline and online for storing and processing data which make it uncomplicated to parse it either directly from the web or from the computer (Alnaqeib *etal.,* 2010). Consequently, many online resources have used XML format to represent their data such as dbSNP of NCBI ftp. Many bioinformatics labs use the XML files to store and read the information. Moreover, different types of operating systems are used in these labs due to the necessity of various kind of software. Almost all programming languages can parse XML files. These languages use specific modules and libraries to handle these files. However, each one of these languages takes a period of time, memory space, CPU usage and lines number to parse them, differently from another one. When it comes to big XML file such as 2, 3 gigabytes or more, execution time, memory consumption and CPU usage, do matter when XML files are parsed. In addition, performance of these languages differs from one platform to another. The aim of this report is to spot the light on the most suitable programming language that takes less time, memory usage, CPU consumption and lines number to parse big XML files with the appropriate platform.

Methods

Six common programming languages were used and they are C++, C#, PHP, JAVA, Python and Perl as these are the most common programming languages used in bioinformatics labs. The programming languages were executed on two platforms, Windows and Linux Minta. The languages were executed using DOS in Windows and Terminal in Linux to consume less memory. There are many modules and libraries for each language to parse the XML files. Chosen the suitable module or library depending on how the module works. Since the size of the chosen XML file is very big, the module that does not load the whole file to the memory was chosen for each programming language. In addition to the size, the modules that work well on all platforms were taken into consideration when appropriate modules were chosen. Table 1 shows the modules used for each programming language.

The language	The module
C#	XMLReader
C++	XmlInspector
Java	SAX
PHP	SAX
Python	SAX
Perl	XMLTwig

Table 1: Programming languages and their chosen modules to parse XML files

As there are many programs to compile these languages, table 2 reveals the compiler name of each one besides its version for each platform.

Table 2: The compiler software for each language with the version number for each platform

Windows		Linux	
The Compiler	**The Version**	**The Compiler**	**The Version**
Visual C# 2008	3.5	Mono	4.0.12
g++	5.3.0	GUN g++	4.8.4
Jdk javac/java	1.8.0_73	Jdk javac/java	1.7.0_95
PHP	5.6.15	PHP	5.5.9
Python	3.4.4	Python	2.7.6
Perl	5.22.1	Perl	5.18.2

To calculate the executing time for each one, different libraries are used such as DataTime in C#, ctime in C++ and Date in Java. In addition, memory usage and CPU consumption were calculated using top command and system monitor in Linux and task manager in Windows. The XML file ds_chY.xml was used as a test file which contains on human genetic variations for the chromosome Y. Its size is 674 Megabyte. Nine attributes were extracted from this file. The locations of these attributes are distributed among root element, element and child element in the XML file. Moreover, some of these strings are manipulated as this usually happens when XML files are parsed. One of these attributes is capitalized and a small string (two letters) was added to other attribute. These attributed were saved in a tsv file as a text file. The features of the laptop that was used in this research are, HP EliteBook, 8 Gigabyte RAM and Intel (R) Core(TM) i5-4310U CPU @ 2.00GHz 2.60 GHz. The operating systems that were used are Windows 10 Pro 32-bit and Linux Minta 17.3.

Results

The results of the research uncovered a huge variation among these programming languages when XML file is parsed in terms of executing time and memory usage and slight difference of CPU consumption. Here we will

divide the results into four sections, execution time, memory usage, CPU consumption and lines number.

Execution Time: As showed in figure 1, C# is the fastest one on both platforms meanwhile Perl was the slowest one. C# took only 19 seconds to parse the XML file on Linux while it took 7 seconds on Windows. As Perl was the slowest one, it took 3.42 minutes on Linux and 4.25 minutes on Windows to parse the file. The speed of other languages was also different. For example, Python needed 26 seconds on Linux while it needed 34 seconds on Windows. PHP also needed 1.35 minutes on Linux and 48 seconds on Windows to parse the file. Moreover, Java needed little time on Windows which was 12.5 seconds close to C# execution time and 40 seconds on Linux. And finally, C++ had also good results, 38 seconds on Linux and 57 seconds on Windows.

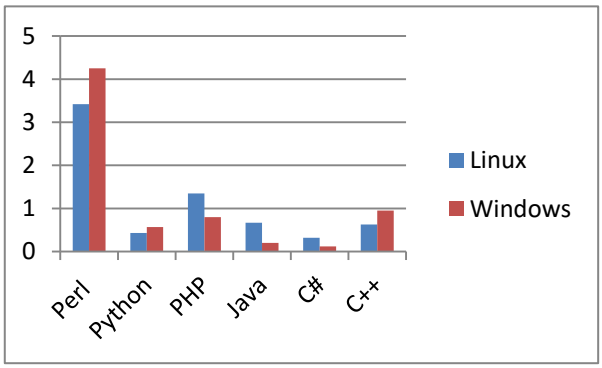

Figure 1: Execution time for each programming language on both platforms. It shows that C# is the faster one while Perl was the slowest one on both operating systems.

Memory usage: Memory usage is important when parsing very big XML files. Figure 2 illustrates the memory usages when the XML file was parsed using all the six languages on both platforms. C++ was the best one, it used less memory on both Linux and Windows. It used only one megabyte on Windows meanwhile it needed 0.37 megabyte on Linux. On the other side, Java needed a lot of memory on Linux which was 57 megabyte and 11.2 megabyte on Windows. The other languages as the following, PHP needed 4.3 megabytes on Linux and 5.9 megabytes on Windows. Also, Python used 4.6megabytes on Linux and 6 megabytes on Windows. In addition, Perl also used more memory on both Linux and Windows. It consumed 12 megabytes on Windows and 13.9 megabytes on Linux. Finally, C# consumed less memory on windows compared to Linux. It used only 2.8 megabytes on Windows while it consumed 15.2 megabytes on Linux.

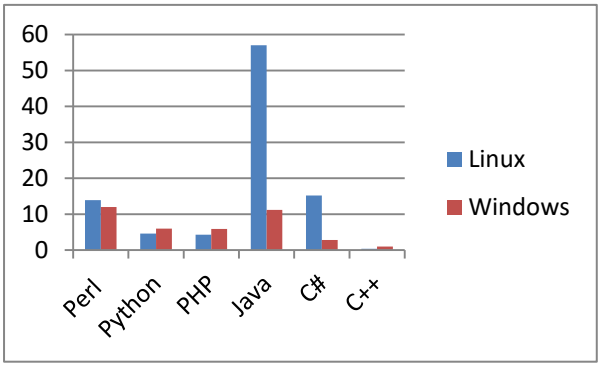

Figure 2: It shows the memory usage for each language on both platforms in Megabytes.

CPU consumption: There was not so much difference in CPU time consumption among all languages when the XML file was parsed on both Linux and Windows except java which consumed slightly more CPU time than other languages on Linux as it is showed in Figure 3. Perl, Python, C# and C++ consumed 25% of the CPU time meanwhile PHP needed only 20% of the CPU and Java consumed 44% on Linux. On the other hand, Perl, Python, Java, PHP and C++ consumed 29% of the CPU time, while C# consumed less CPU time about 24%. Accordingly, PHP was the better on Linux while Java was the worst. In addition, C# was the best on Windows and the rest were the same.

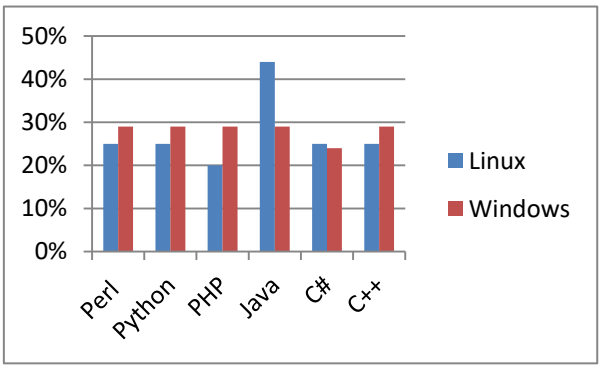

Figure 3: It displays CPU time consumption of each language on both operating systems.

Lines Number: As the XML files might be parsed by non computer science specialist, lines number is an important factor to determine which language needs less code lines and consequently less experience in a specific task. Figure 4 shows the lines number needed to parse the XML file. Moreover, any extra empty lines and comments lines were excluded. Java and C++ needed more lines than other languages. Java and C++ used 108 and 109 lines respectively. The other three languages used fewer lines. Perl, Python, C# and PHP used 46, 46, 56 and 66 lines correspondingly.

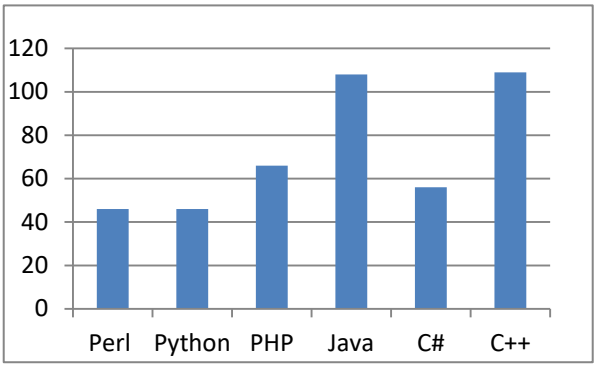

Figure 4: It shows lines number of each programming languages needed to parse the XML file

Discussion

All the scripts of the six languages were written by the same programmer with a different experience level in the programming languages. The main purpose of this research is to catch the most suitable programming language to parse an XML file with an operating system. The dilemma to choose one of these six languages has increased after this research.

Perl

Perl is one of the most widely used languages to parse XML files specifically for those whose background is not computer science. Perl is easy to learn. Perl is a script programming language for text processing and web application that was designed, maintained and updated by Larry Wall to be run in UNIX environment (Till, 1996). Perl is widely used in biological area and labs since it can solve and fix biological tasks in a very reasonable way. Bioinformaticians are broadly using Perl to analyze and process data of DNA and amino acids sequences. Perl has many exceptional features that make programmers heavily use it. One of the main marvelous Perl's features is that, Perl could be learned quickly even if the programmer does not have appropriate information background about it. Moreover, Perl could be written to process a specific data in only few script lines while it might need more script lines if another programming language is used such as Java and C (Tisdall, 2001). In this research, Perl used fewer lines to parse the XML file and did not consume much CPU time. The CPU time consumption for Perl was likely to be close to the other languages CPU time consumption on both platforms. However, it needed 3.42 minutes on Linux and 4.25 minutes on Windows to parse the file which make it the slowest one among the languages. In addition, Perl used a lot of memory (RAM) which makes it the third one to consume big amount of memory after C# and Java on Linux.

Moreover, Perl also used huge amount of memory on Windows by using 12 megabytes that make it the second bigger memory user after Java.

Python

Python is one of the script language used widely in computational biology to analyze biological data. Normally, it needs fewer lines and less knowledge to write a script for a specific task compared to other languages. Moreover, it has a library called BioPython designed to perform most bioinformatics tasks (Timand & Wayne, 2015). It was the second faster one to parse the XML file after C# on both operating systems. Moreover, Python was the third better one in terms of less memory usage by consuming 4.6 gigabytes on Linux and 6 gigabytes on Windows. As other languages, Python CPU time consumption was very close to other languages. It also did not need a lot of lines to do the task and the lines number was the same of Perl lines number.

PHP

PHP is a script programming language that is used to design website, embed SQL queries, parse data from HTML forms, embed tags of HTML and many other futures (Leon, 2000**).** PHP was a collection of Perl scripts at the beginning back to 1995, which was very basic and lack of many important features such as for loops (Andi, *et al.*2004). PHP presents acceptable results in all examined issues in this research. It needed 1.3 minutes on Linux and 0.8 minutes on Windows to parse the file. In terms of memory, PHP did not use much memory on both platforms. It used 4.3 gigabyte son Linux which make it the second one after C# and 5.9 gigabyte son Windows that make it the third one after C# and C++. In addition, the line number was also few comparing to Java and C++ and close to others.

Java

Java is an object oriented language (OOP) and among those languages called high level language. It was developed based on C++ and C in USA at Sun Microsystems by James Gosling and his team. In addition, it is considered as the easier language to be learned among objected oriented languages (Thomas, 2012).One of the good features of Java language is that, the same java script would run on many platforms such as Windows, Linux or Macintosh without any manipulating. However, Java might need more experience and much time to learn it as it was not designed for students (Thomas, 2012)**.** This research shows that Java was not good choice to parse the XML file in terms of memory usage as it used huge amount of memory specifically on Linux. It used 57 megabyte son Linux that make it the first one to use much memory and the second one on Windows after Perl by using

11.2 megabytes. In addition, Java was very close to other languages when it comes to CPU time consumption as it used 29% of the whole CPU time on Windows but it used more CPU time than other languages by using 44% on Linux. On the other hand, Java has good results in terms of execution time. It needed only 12.5 seconds to parse the XML file, which makes it the second faster after C# on windows. Furthermore, Java used 40 seconds on Linux which makes it the fourth, after C#, Python, and C++. In terms of lines number, Java needed a lot of lines to write the code that parsed the XML file by using 108 lines to be the second one after C++.

C#

C# is another high level common programming language and one of object oriented languages group that was designed by Anders Hejlsberg and his team when they developed .Net framework at Microsoft. C# is a well-known programming language for .NET Common Language Runtime. In both platforms, C# was the better one among all the programming languages **(**Svetlin, 2013**).** C# was superfast and parsed the XML file in just 19 seconds on Linux and 7 seconds on windows. However, it consumed much memory on Linux by consuming 15.7 gigabytes that makes it the second biggest memory consumer after Java. In contrast, C# was the second best one, consuming less memory on Windows after C++. In addition, CPU time consumption for C# was very close to other languages. Another notable feature is that, C# did not need many lines to write the code to parse the file. It just needed 58 lines which are very close to script languages such as Perl and Python.

C++

C++ is another powerful object oriented language that was developed based on C. It supports many OOP features such as handling exceptions, multiple inheritances, and over loading. C++ is a complicated language. C++ supports many kinds of variables such as array, pointers, and I/O facilities, and the conversion between variable types. C++ has many compliers, GCC might be the most notable one (Bjarne, 2013).In this research, C++ parsed the XML file in good time. It was the best third one after C# and Python on Linux and the fourth one on Windows. In addition, C++ was the best one by consuming less memory among the programming languages on both operating systems. CPU consumption for C++ was the same as other languages. However, its code used 109 lines which make it on the top of languages that needed a lot of lines. Moreover, C++ is a complicated language and the user might face some obstacles to learn it in a short period of time. (Bjarne, 2013).

Conclusion

The investigated programming languages gave different results for every examined issue on every platform. Table 3 shows the results and the performance of each programming language on both operating systems.

Time			Memory in MB			CPU			Time in Min			Line number		
Pro.Lang	Linux	Windows	Pro.Lang	Linux	Windows	Pro.Lang	Linux	Windows	Pro.Lang	Linux	Windows	Pro.Lang	Linux	Windows
Perl	3.42 min	4.25 min	Perl	13.9	12	Perl	25%	29%	Perl	3.42	4.25	Perl	46	46
Python	26 sec	34 sec	Python	4.6	6	Python	25%	29%	Python	0.43	0.57	Python	46	46
PHP	1.35 min	0.8 min	PHP	4.3	5.9	PHP	20%	29%	PHP	1.35	0.8	PHP	66	66
Java	40 sec	12.5 sec	Java	57	11.2	Java	44%	29%	Java	0.67	0.2	Java	108	108
C#	19 sec	7 sec	C#	15.2	2.8	C#	25%	24%	C#	0.32	0.12	C#	56	56
C++	38 sec	57 sec	C++	0.37	1	C++	25%	29%	C++	0.63	0.95	C++	109	109

Table 3: The comparison table shows the performance of each examined language for each issue on both platforms

Some of them showed good results in terms of run time, CPU time consumption, memory usage and lines number. However these languages might need more time to be learned and written. Script programming languages such as Perl, PHP Python are easy to be Learned and compiled programming languages such as Java, C++ and C are much more difficult(Prechelt, 2000).Consequently, many factors could determine which language might be better choice to parse XML files. Here we suggest some tips to succeed, to choose the most suitable language.

- If the running time is the main important factor, then C# might be the best one since it was the fastest one. However, C# consumed a lot of memory on Linux. In addition, the user might need more time to learn and to write a script using C#.
- C++ is the best one when it comes to memory usage. The amount of consumed memory was very little on both platforms compared to other languages. In addition, C++ presented good results in terms of run time. However, C++ needed twice lines number compared to scripts languages and C#. Moreover, C++ needs prior experience in programming to write a program to parse an XML file. In result, C++ might need more time to be learned and get confident to parse a text file.
- There was a not a big difference between the languages when CPU time consumption was compared on both platforms except Java, which consumed about twice CPU time compared to other languages on Linux. In this case, it depends on the user to choose which language to parse XML files.
- If the user does not have prior knowledge about programming, scripting languages might be a better choice. Script languages are easy to be learned and even according to this research it just needed half line number comparing to other languages. Specifically, among script languages, Python was the best one since it just needed 46 lines

to write the script and consumed less memory compared to Perl and even, presented a good runtime.

- Java was also fast, however, it was the biggest memory consumer among all the languages on both platforms. Moreover, Java is a high level language and might need twice time to write a script compared to script languages. (Prechelt, 2000). In addition, Java consumed more CPU time than other languages on Linux. In this case, Java might not be a successful choice for this task. Finally, PHP results ranged between best and worst which make it neither the best one nor the worst one.

References:

Alnaqeib, R., Alshammari, F.H., Zaidan, M., Zaidan, A., Zaidan, B., Hazza, Z.M., 2010. An Overview: Extensible Markup Language Technology. *ArXiv Preprint arXiv:1006.4565.*

Leon, A., 2000, *Core PHP Programming Using PHP to Build Dynamic Web Sites,* Prentice Hall PTR: USA.

Bryan, M., 1998. An introduction to the extensible markup language (XML). *Bulletin of the American Society for Information Science and Technology.* 25, 11-14.

Soo, F.M. & Wei, L.M., 2002. Introduction to the Extensible Markup Language (XML). *XML Programming Using the Microsoft XML Parser.* Springer. 1-40.

Andi, G., Bakken, S., and Rethans, D., 2004, *PHP 5 Power Programming*,John Wait: USA.

Svetlin, N., 2013. *FUNDAMENTALS OF COMPUTER PROGRAMMING WITH C#.* Sofia**:** Bulgaria.

Prechelt, L., 2000. An empirical comparison of C, C++, Java, Perl, Python, Rexx, and Tcl. *IEEE*.

Timand, S., Wayne, B., 2015 *Python Programming for Biology.* Cambridge University Press: United Kingdom.

Bjarne, S., 2013, *The C++ Programming* Language. Addison Wesley: Michigan.

Till, D., 1996. *Teach yourself Perl 5 in 21 days.* Sams.

Tisdall, J., 2001. *Beginning Perl for bioinformatics.* " O'Reilly Media, Inc.".

Thomas, W., 2010, *An Introduction to Object-Oriented Programming with JavaTM.* McGraw-Hill: New York