International Journal of Intelligent Computing and Information Sciences

MULTITHREAD IN NAMED ENTITY RECOGNITION

Z. M. Rabea M. A. Abu Elsoud M. Z. Rashed

Faculty of Computer and Information Sciences, Mansoura University, - Egypt
carwan22@yahoo.com mohamed_hossieny@yahoo.com magdi_z2011@yahoo.com

Abstract: *According to Gordon Earle Moore, Every two years, the number of core on a CPU chip is doubling. So we change our program to use threads for different reasons, program will run faster and make better use of the multiple CPU/core architecture that you are running on. This paper introduces a multithreading named entity recognition (NER) approach in Open American National Corpus. Named entity can be name of person, location, organization, number, date, e-mail, and so on .Using the pooling technique and get benefits of its computation time in the NER.*

Keywords*: Named entity recognition; multithreading; pooling; Gate.*

1. Introduction

The information extraction (IE) is a subfield that lies under the natural language processing (NLP) umbrella. The ultimate goal of IE is to deal with natural text in order to produce readable output through extracting useful information from such text [1]. Various application domains were benefited of IE. For example, biomedical domain uses IE to extract the names of diseases, genes, proteins, cell type, etc. [2] Another example is opinion mining where IE is used to extract product review opinions as well as comparisons from reviews. Also search engines applies IE to extract meta-data to retrieve the required quires [3].

The task of IE is divided to [1]: Named entity recognition (NER), co-reference resolution, relation detection and identification of roles. NER is a first step of IE that responsible for identifying and classifying the entity in the text, entity include proper noun, number, dates, e-mail address, etc. Co-reference resolution used to collect all expression that refer to the same entity. Relation detection task of finding relation between pairs of entity such as organization and location. Identification of entity roles usually deals with filling in a pre-defined event frame slots. Machine translation, information retrieval and question answering used named entity as pre-processing task. Also NER can used to improve Search engines by using it to rewrite quires into more formed quires. Its better searching for more information. To say that NER is good required to extract knowledge. Not only required determine the type of named entity(NE) which can be noun, adverb, prepositions, adjective, and so on but also classified each phrase e.g. noun phrase can classified to people, organization, product, companies and so on . The earlier approaches to recognize NER is handcrafted rule which use in system that involve the Message Understanding Conference (MUC) shared task. Requiring human intuition and large number of rule to write simple approach for recognize named entity. System that involve the Conference on Natural Language Learning (CoNLL) use machine learning approach. There are many different techniques in machine learning approach. They are supervised learning (SL) techniques, unsupervised

learning techniques, Semi-supervised learning techniques. SL is a very popular approach for NER but unsupervised learning isn't. Some systems use Semi-supervised learning [2].

SL include Hidden Markov Models (HMM) that is not success in NER but success in other NLP problems such as speech recognition and part-of-speech tagging, Maximum Entropy Models is a good choice for NER. It can handle a large number of features and Very successful in general for integrating information, Support Vector Machines (SVM) applying to NLP task such as chunking and text classification and performing a non-linear classification, Decision Trees is adequately but the transformation-based learner is better on NER task (CoNLL 2002) …, and other.

The data need first to transfer into format fit for previous approaches another words extract necessary feature from data. Feature selection is the process of determining which subset of features use. Feature selection useful for redundant variables. Reducing number of features reduce execution time and the cost of recognition. There are many type of feature. Some feature depend on the word itself such as morphology features (prefix - suffix), statistical feature (word length), orthographic features (all uppercase letter-only digit-contain digit-is punctuation-…). Other feature depends on the position of word such as part of speech tag, and chunking tag. Some system use gazetteer list that containing a list of location. organizations, days, car, etc. [4, 5].

Many Workshops and conference about NER and IE were held such as MUC, CoNLL, and the Japanese Multilingual Entity Tasks (MET). Seven MUC conference have been organized by the Defense Advanced Research Projects Agency (DARPA) since1987 to 1997. A lot of developments in the field of (English) NER discussed in MUC-6 and MUC-7 conference. Temporal expressions and numerical expressions and named entity had to be recognize in MUC-6 and MUC-7 conference. The MET conference look alike the MUC conference but MET for Japanese NER [4]. The first event of CoNLL organized in July 1997 by SIGNLL (ACL's Special Interest Group on Natural Language Learning). Since that time until now, every year held a conference to discuss the evolution of the NER. CoNLL Shared Task on NP chunking in 1999 to Shared Task on Shallow Discourse Parsing in 2015[6].

2. NER existing systems

Three are many existing systems for NER [7] such as BANNER, Stanford NER, Illinois Named Entity Tagger and GATE and other. BANNER [8] is intended for biomedical text, it is based on conditional random fields. Stanford NER or CRFClassifier [9] is used particularly for person, location, organization classes. It is based on conditional random fields. Illinois Named Entity Tagger [10] has two style. The first tags plain text with named entities such as people, locations, organizations and Misc. The second uses 18-label type based on the OntoNotes corpus. Illinois Named Entity Tagger uses seed list extracted from Wikipedia [11].

General Architecture for Text Engineering (GATE) [12] used for all types of computational task enclose human language. It is used to solve text processing problem. It use machine learning approaches to recognize NER. It use Maximum Entropy Models and Support Vector Machines. It also use Weka. Weka is open source software, it consider a collection of machine learning algorithms such as Naive Bayes, k-Nearest Neighbors and decision tree algorithm. GATE also used JAPE (Java Annotation Patterns Engine) that is based on regular expressions. GATE contain too set of plugins include, POS Tagger, Noun Phrase Chunker, parser, Gazetteer list, Ontology Editor, Tokenizer, Sentence Splitter, etc.. GATE support also plugins for several languages: Arabic, French, German, Italian, Chinese, Romanian, Hindi, Russian, and Cebuano. GATE read data from the following type of document, Plain Text, HTML, SGML, XML, RTF, Email, CoNLL/IOB, UIMA, and other. The output can save as XML file. It is

important to mention here that GATE system include a complete information extraction system known as A Nearly-New Information Extraction System (ANNIE) that uses some features such as orthography features, part of speech tagging, gazetteer list, and others.

3. Multithreading

Before going through the description of the proposed framework, it is important to introduce some background related to the multithreading due to its effective role in our proposed work. Multithreading (MT) is a way to allow one program to do multiple task at same time in other mean introduce parallelism in the program to maximum the utilization of the CPU time [13]. Using multithread can result in significant performance gains especially when working with multiple core machines. MT aims to divide works to more thread that execute concurrently. On single processor, thread in java provide concurrent operation. There are two ways in java to create thread. Thread may be in another class or subclass of java.long.thread [14].

It is hard to write multithread application because of it is difficult to make parallel algorithm. Other difficult are deadlock, race condition, and tearing and cache coherency [15]. When the number of task is more than the number of thread, use the thread pool. Thread pool give every thread one task, other tasks wait until any thread finish its job. Thread pool service tasks in efficient way by using the java.util.concurrent package [16]. Thread pool provide the time of creating and destroying thread and its associated resource for each task, the performance and system stability is better. Thread pool provide memory, instead of make number of thread equal the task, make little number of thread equal the potential of machine. Number of threads you can use at the same time is limited.

4. Proposed framework

The aim of this section is to present the proposed framework that relied on the usage of pooling technique and get benefits of its computation time in the context the NER. Mainly, the proposed framework is divided into two systems. In the first system, the gate was embedded in the java program on the other hand, the second system applied pooling which helps in enhancing computation time as shown in the experimental results. Figure. 1 illustrates the steps of the proposed works while the following subsections describe these systems in much more details.

After open Gate program installing Gate plugins shown in figure. 1 and all requested files, creating a new corpus and then adding MASC files to corpus, loading the processing resources that is need "ANNIE". Running this application, C1 take a lot of time. C2 cannot perform this task as show in figure.2, the size of Corpus is too big, and the computer is idle. So Using java to embedded gate library in the program that show in algorithm. 1 but it also take a lot of time so used multithreaded that show in algorithm. 2.

4.1 Sequence System Steps

1. Adding all gate library and gate.jar to the library of Java, so we can added it to our project. Gate home, gate plugins and user configuration file should be determined so we can initialized gate.
2. Using PersistenceManager.loadObjectFromFile() to load ANNIE and other plugging need to use by using .gapp file which making it by developing our GATE processing pipeline in GATE Developer.
3. Defined the path (p1) for data base on computer for each folder in p1 add all text files in this folder to corpus (array of files).

4. Using class ConditionalSerialAnalyserController which apply processing resource methods to each documents in the corpus.
5. Create an instance of a resource to implementation corpus and document and save original content and the analysis information about document.

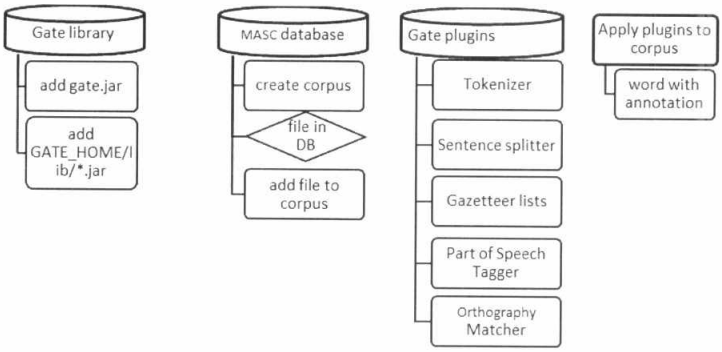


Figure 1: The steps of the proposed work

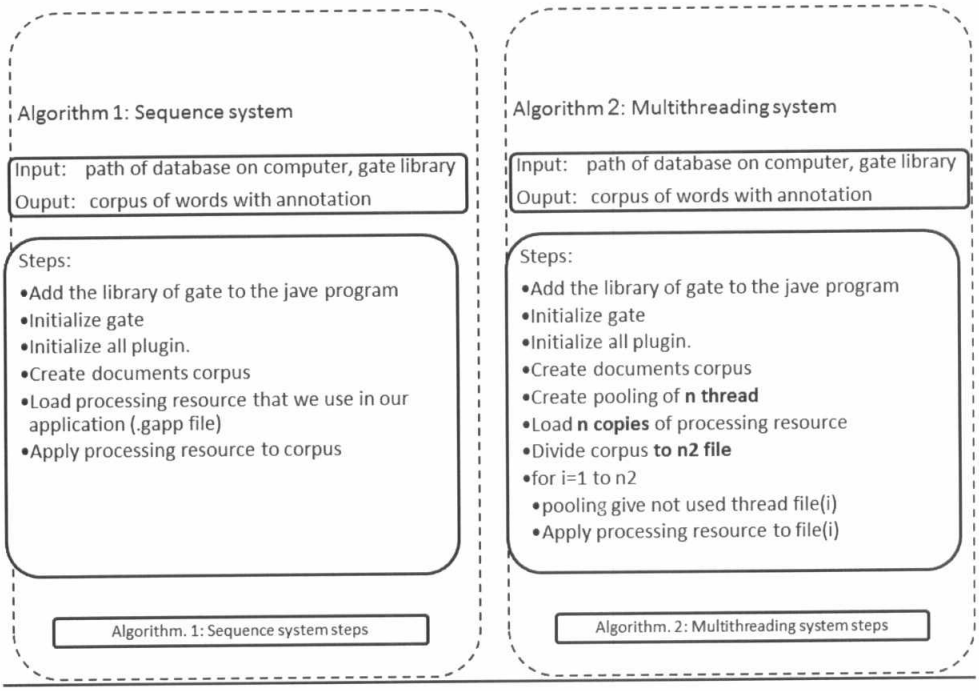
4.2 Sequence System Steps

6. Adding all gate library and gate.jar to the library of Java, so we can added it to our project. Gate home, gate plugins and user configuration file should be determined so we can initialized gate.
7. Using PersistenceManager.loadObjectFromFile() to load ANNIE and other plugging need to use by using .gapp file which making it by developing our GATE processing pipeline in GATE Developer.
8. Defined the path (p1) for data base on computer for each folder in p1 add all text files in this folder to corpus (array of files).
9. Using class ConditionalSerialAnalyserController which apply processing resource methods to each documents in the corpus.
10. Create an instance of a resource to implementation corpus and document and save original content and the analysis information about document.

4.3 Multithread System Steps

In order to create a multithreaded code must follow the following steps

1. Use synchronized way for methods, that shared by thread to avoid a race condition error.
2. Transfer static variable to a thread-safe way.
3. Load n copies of processing resource store them in a pooling.
4. Divide corpus to thread by give every thread one file, when any thread finish its job it take another file. Processing text require get a copy of processing resource from the pool, when processing is finished return it to the pool.



5.. Experimental results and discussion

5.1 Applied database

The proposed paper was applied MASC database that contains 506768 words from the Open American National Corpus (OANC) [17]. It is contain 376 different files from newspaper, journal, email, twitter, jokes, movie-script, technical, etc.

5.2 Experiments

Experiments were performed on two computer. Computer 1 (C1), Computer 2 (C2). C1 was worked with-Genuine Intel(R) CPU 000@ 2.40 GHz 2.40 GHz and installed memory (RAM) of 4.00 GB. C2 was worked with AMD A6-5200 APU with Radeon(TM), 2.00 GHz and installed memory (RAM) of 4.00 GB (3.47GB usable).

5.3 Evaluation metrics

Using Stopwatch or calculate the difference between system current time in begin and end of code are not a good way for measuring performance of our system. Because it measures Total time in seconds from the beginning of running the program until the appearance of the results on the screen. The time affected with any user working on the computer or anything run in the background at this time. This caused a significant difference in the outcome in case we have a lot of programs that operate on the device at this time.

5.3. 1 Evaluation Metrics In Multithreading System

For each thread we use methods that return the total CPU time for this thread in nanoseconds (cpu_time) and returns the time that this thread has executed in user mode in nanoseconds (user_time).
Calculate cpu_time and user_time for the main method (c_m_time, u_m_time).
Then compare between the time of thread and then select the maximum time (max_cpu,max_user).
The CPU time of the multithreading system (C_M_S) = max_cpu+ c_m_time.
The time that the multithreading system has executed in user mode (U_M_S) = max_user +u_m_time.
The difference between C_M_S and U_M_S determine the time spent running on OS code.
In sequence system
Calculate cpu_time and user_time for the main method (c_m_t, u_m_t).
S_m_t= c_m_t - u_m_t.

5.4 Experiment Result

In the figure below we compare gate program, sequence system and multithread system in both computer C1 and C2.

Table 1 show the total time in second as a user watching the computer screen until the program finishes through gate program and sequence system and multithread system using 2, 4, 6, 8, 10 and 12 thread. We made the experiment more time and take the average result.
Figure 2 show the total time in second in CPU mode through gate program and sequence system and multithread system using 2, 4, 6, 8, 10 and 12 thread.

In the Figure 3 we compare sequence system in C1 and C2, compare the time each computer take in CPU mode, user mode and system mode. We made the experiment five time and take the average. C1 take approximately 87.2 second to display the result in CPU mode and take approximately 79.6 second in user mode, the different between the average time of CPU mode and user made is 7.6 second. C2 take approximately 313.6 second to display the result in CPU mode and take approximately 289.1 second in user mode, the different between the average time of CPU mode and user made is 24.5 second. It is the time in system mode in all experiments. In system mode, the time is approximately fixed in all experiments. C1 give the best result three and half time from the C2.

Figure 4 show the time each thread take in CPU mode when using 4 thread in C1 and C2. The main thread takes the same time in each experiment, but the time each thread take difference the distribution of work to thread different each time because of using pool technique.

Figure 5 show the speed for each system when using 2, 4, 6, 8, 10 and 12 threads in system mode between C1 and C2. We made the experiment four time. There is a slight difference between the times each experiment take. In C1, the maximum time when using two thread is 5.7 second and the minimum time is 4.5 second. The maximum and minimum time when using four thread is 3.1 second, 2.8 second, it is better than two thread. The maximum and minimum time when using six thread is 3 second, 2.7 second, it is better than four thread. The maximum and minimum time when using eight thread is 2.1 second, 1.8 second, it is better than six thread. The maximum and minimum time when using ten thread is 2.3 second, 2.1 second, the minimum time when using ten thread approximately equal the maximum time in eight thread, so it is better to use 8 thread in C1 to provide memory. In C2, we also determine the maximum and minimum time for each thread, when using two thread the time is 23.3 second and 19.1 second, when using four thread the time is 14.6 second and 11.3 second, when using six thread the time is 12.1 second and 10.9 second. The minimum time when using 4 thread is better than maximum

time when using six thread. It is better to use 4 thread in C2. The other thread gives convergent result time. It can be seen that the multithreaded system, sequence system and GATE produce similar result. Efficiency of each of them is equal. Using multithread is better than using sequence system. Execution time decreases with increase the number of threads. The speed of the program does not increase linearly.

Table 1: the time gate program, sequence system and multithread system take in second in C1 and C2

	gate	sequence	2 thread	4 thread	6 thread	8 thread	10 thread	12 thread
C1	114.514	87.29688	53.24921	31.72917	26.15379	19.95313	18.34917	15.57292
C2	ideal	313.6494	187.8644	113.6404	96.00872	87.16447	72.98025	65.02902

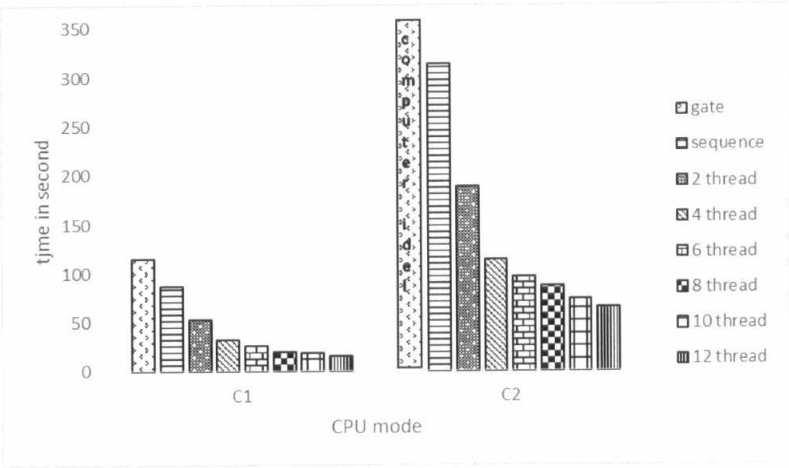


Figure 2: the different between gate program, sequence system and multithread system in C1 and C2

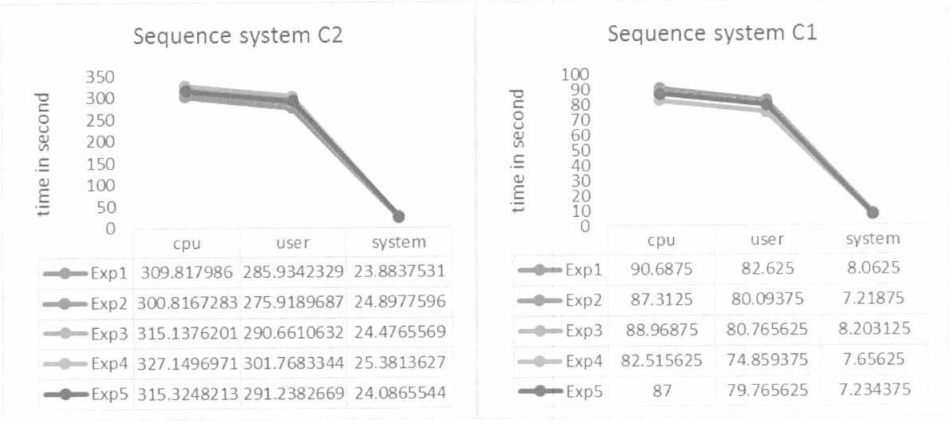


Figure 3: compare the difference of time between C1 and C2 in sequence system

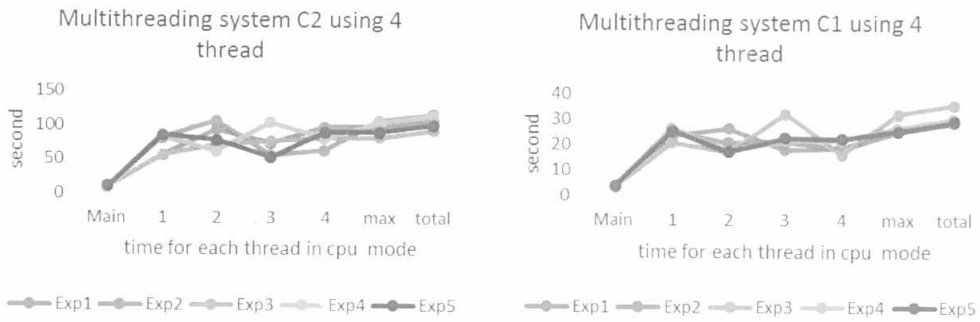


Figure 4: the time each thread use when using 4 thread in C1 and C2

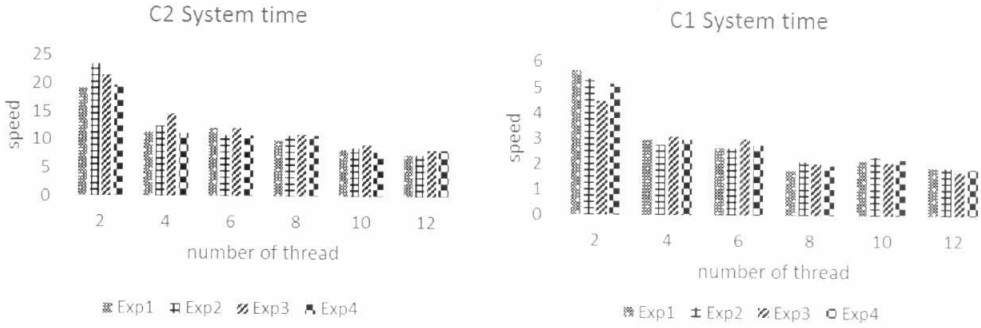


Figure 5: compare the difference of time between C1 and C2 when using 2, 4, 6, 8, 10 and 12 thread

6. Conclusions

We have use multiple threads to present a system for running GATE. Our model requires to Use synchronized way methods to transform GATE into thread-safe way before it can be multithreaded.
Using pooling give better performance, it saves time because there is no need to create new thread, saving the cost of creating new threads. The implementation of the system show that the speed increases with increasing number of threads as shown in figure 2, The speed of system when using 2 thread increase 40% about using sequence system, increased 40% when using 4 thread about using 2thread, increase 16% when using 6 thread about using 4thread, increase 16% when using 8 thread about using 6 thread, increase 12% when using 10 thread about using 8 thread, increase 12% when using 12 thread about using 10 thread. The increases not linear as expected. In pooling, the difference in time that main methods take to divide the tasks to system that use (2, 4, 6, etc.) thread, almost not mentioned with the time used in the total process as shown in figure 4.

On a single-core CPU, threading can actually slow you down, the CPU spend time to switch between threads. When using two computer, C1 faster than C2 by 3.5% when using sequence system as shown in figure 3, this percentage increase with increase the number of thread, C1 faster than C2 by 4.1% when using 12 thread as shown in figure 5. Better multi-core CPU, better result.

References

1. R. Farkas, "Machine Learning techniques for applied Information Extraction", Ph.D, University of Szeged, 2009.
2. N. Suakkaphong, Z. Zhang and H. Chen, "Disease named entity recognition using semisupervised learning and conditional random fields", J. Am. Soc. Inf. Sci., vol. 62, no. 4, pp. 727-737, 2011.
3. Y. Zhang, *Automatic extraction of outbreak information from news*. 2008.
4. T. Bogers, "DutchNamedEntityRecognition: OptimizingFeatures,Algorithms,and Output", 2004.
5. S. AbdelRahman, M. Elarnaoty, M. Magdy and A. Fahmy, "Integrated Machine Learning Techniques for Arabic Named Entity Recognition", *International Journal of Computer Science Issues*, vol. 7, no. 4, pp. 27:36, 2010.
6. "Proceedings of the 52nd Annual Meeting of the Association for Computational", 2016.
7. B. Sun, "Named entity recognition Evaluation of Existing Systems", Master in Information Systems, Norwegian University of Science and Technology, 2010.
8. http://sourceforge.net/projects/banner/, viewed on 2/1/2016.
9. http://nlp.stanford.edu/software/CRF-NER.shtml, viewed on 15/1/2016.
10. https://cogcomp.cs.illinois.edu/page/download_view/NETagger, viewed on 6/2/2016.
11. M. Ciglan and M. Laclavik, "Evaluation of named entity recognition tools on microposts", IEEE 17th International Conference on Intelligent Engineering Systems (INES), 2013.
12. *Developing Language Processing Components with GATE Version 8 (a User Guide)*, available at https://gate.ac.uk/ , 2016.
13. X. Dong, G. Cooperman and J. Apostolakis, "Multithreaded Geant4: Semi- automatic Transformation into Scalable Thread-Parallel Software", College of Computer Science, Northeastern University, Boston, MA 02115, USA, 2010.
14. J. Cowell, *Essential Java Fast*. London: Springer London, 1997.
15. M. Gregoire, *Professional C++*. .
16. B. Kurniawan, *Java 7*. [S.l.]: Brainy Software, 2014.
17. http://www.anc.org/data/masc/downloads/data-download/ viewed on 16/9/2015.