International Journal of Research in Engineering, Science and Management
Volume-2, Issue-5, May-2019
www.ijresm.com | ISSN (Online): 2581-5792

A Review on Scopes and Issues in Green Complier, Solving Synonym, Homonym, Hyponym and Polysemy Problems and Translation of English Algorithm in C Program using SDT

Sini Anna Alex[1], Rithika Bellad[2], Ankita Sumod[3], Soumya P. Sawkar[4]
[1]*Professor, Dept. of Computer Science and Engg., M. S. Ramaiah Institute of Technology, Bengaluru, India*
[2,3,4]*Student, Dept. of Computer Science and Engg., M. S. Ramaiah Institute of Technology, Bengaluru, India*

***Abstract*: The paper "Scopes and Issues in Green Compiler" focuses on energy conservation as there has been a high increase in energy consumption due to the emergence of cloud computing. Cloud computing use large datacenters which continuously consume energy at both the hardware and software level. Controlling the energy consumption at the hardware level can be tedious so green compilers are our best chance at controlling energy consumption at the software level. In this paper, some suggestions on how to control the energy consumption are discussed here.**

In the paper, "Improved Unsupervised Framework for solving Homonym, Synonym, Hyponym and Polysemy Problems from Extracted Keywords and Identify Topics in Meeting Transcripts", it says that keyword is the important item in a document that provides systematic access to the contents of documents. Keywords can be used for information retrieval and text categorization. In the present system, hyponym, homonym, synonym and polysemy problems were solved by trained keywords from meeting transcripts.

In the paper, "Translation of English Algorithm in C program using Syntax Directed Translation Schema", applications that translate English written algorithm to other programming languages such as C/C++ is very useful for individuals who are interested in the programming field but are unable to code due to lack of knowledge in programming languages. This translation has been achieved by implementing the rule based approach that uses the syntax directed translation schema. The input given to the system are algorithms that are written in regular English language and the output generated will be in the respective programming language.

***Keywords*: Green Complier, Solving Synonym, Homonym, Hyponym, Polysemy**

1. Introduction

In recent years cloud computing has sky rocketed worldwide and changed the way the world approaches computing. However, Clouds consume high measures of energy due to their datacenters hosting many services and applications. Global warming has become an issue in recent years with use the use of cloud computing adding on to the detrimental effect of global warming. So in order to minimize the pernicious effect of technology, green compilers have been suggested. It is seen that replacing high capacity machines with lower capacity machines would be beneficial as high capacity machines don't utilize resources completely. But this method doesn't yield a satisfactory decrease in energy consumption. This is where green compilers come into the picture. Energy optimization through the use of green computing can help reduce consumption of energy.

It is said that query expansion in information retrieval will help solve the synonym problem. By mapping a vector space model to a compact space with mathematical tool by Latent Semantic Analysis (LSA) we can solve dimension reduction and synonym problem. The general framework is to gather a large amount of data set, and then construct a model on a small set of data and a rich set of hidden topics which have been found from the universal data set. Semantic topics hidden in the documents are emphasised and guided by the topics inferred from a global data collection in order to handle synonym problem.

Natural Language Processing is usually approached by limiting the user to a compressed and low level English, but the disadvantage of this approach is that although a working code is produced, the natural language that follows is cumbersome. The two major factors to be achieved in a translation system are translation accuracy as well as speed. When it comes to accuracy, smart tools to handle translation standard that include sentences, expressions, phrases etc., need to be developed. In order to obtain an accurate parse and hence a translated output the grammar must be optimised with a vision. In terms of speed, systematic use of corpus analysis, systematic parsing algorithm and design of sufficiently correct data structure are required, which are implemented accurately by the Bison LALR and

GLR parsing algorithms.

2. Literature Survey

In existing systems, some of the methods used to decrease consumption of energy are:

1. Reduction of logic voltages can help in reducing the usage of energy, but the output frequency decreases significantly. This can cause the circuits to slow down and degradation of performance is seen.
2. COFFEE Compiler has also been used, which is capable of combining both modified hardware and software in order to conserve energy at compile time.
3. Clustering instructions has been used by implementing the program as a cluster of identical signals, and compiles them in to one run. This has shown to save energy from 26% to 47%.
4. By changing the order of execution of the programs, it can help in energy preservation.
5. Since, compilers convert recursion into iteration internally, compilers execute recursive processes as stack which tends to save some space, causing deterioration of performance and therefore, consumption of energy.
6. Green scheme for compilers:
   a. Using cache-skipping reduces consumption of power
   b. Clustering Instructions
   c. Use of Energy Cost Database
   d. Instruction reordering and Memory Addressing
   e. Resource Hiberation
   f. Loop Optimization

Sahami and Heilman calculated the correlation between text excerpts using search engines and kernel function to check similarity. For short queries from the web search query logs, Metzeler calculated a wide range of similarity measures. McCallum and Baker made an attempt to condense dimensionality based on class-distribution clustering. Modha and Dhillon discovered a spherical K-means for clustering of sparse text data. Cai and Hoffman introduced text categorization by automatically amplifying from the extracted concepts. Cai and Xuan-Hieu proposed finding hidden topics using LDA model, which was a similar work which used features based on topics to make the word-sense disambiguation more efficient.

Programming Languages Supplemented By Natural Language. Several programming languages are making high usage of natural languages to install commands. Such programming is called Natural Language Supplemented Programming.

Some of the instances of such programming languages are COBOL, BASIC and FORTRAN. Latest instances are AppleScript and KlarDeutsch.

KlarDeutsch:

The purpose of this system is to control machine and equipment with very low levela md precise Natural language sentences that are written in German.

AppleScript:

Despite AppleScript making an extensive use of natural language, AppleScript programs are much more of a linguistic mask that reflects the structures of an ordinary programming languages any other programming languages supplemented by natural languages.

Pure Naturalistic Programming:

The best approach towards a pure naturalistic programming is NLC, which has been extensively mentioned in the paper of Ballard a Biermann, of 1979. The paper proposes a naturalistic programming language for matrix operations specific to domains.

Meta for explained in the articles of Liu and Liebermann of 2005, that consequently

designs an interface, in which one can elaborate a program idea in natural language, such as using simple English sentences.

Later, Knöll, Roman, and Mira Mezini described Pegasus in their paper of 2006. Pegasus

was viewed globally as an increment of languages such as NLC, whose linguistic base is broad so that semantic descriptions can be included that Metafor also incorporates.

3. Discussion

In the paper, “Scope and Issues in Green Compiler” it presents a standard architecture of a proposed distributed green compiler and a high-level workflow. DGC is green compiler which makes use of many green schemes during the ICG in order to reform the source code. A classic green compiler would need a good amount of time to compile source codes for energy conservative executables. This might lead to poor performance. So, DGC can reduce the compile time by spreading the source code over a network of physical or virtual machines. Compiler isn’t able to mould all source code in energy preserving executables like for instance elimination of recursion. DGC suggests green techniques for software developers shining a spotlight of certain areas of code, which isn’t able to be moulded by the compiler for optimising energy during intermediate code conversion. Energy consumption statistics are provided to tell the programmer how much energy could be saved in an executable. The figure shows a pseudocode for DGC.

```
Input:
 1. A C program.
 2. Switches D for distribution
 3. Machine IPs for distribution.
Output:
    1. An energy conservative executable.
    2. Green suggestion.
    3. Energy statistics report of program.
Begin
    while(End of file){
          pre-process source code.}
    If(D){
          Whilse(End of file){
               Distribute on network}
    }
    while(End of file){
        Loop optimization, Dead code and recursion elimination, Un-optimized block
        identification, Energy statistics calculation
        Other Compilation process for intermediate code generation}
    If(D){While(End of file){
                         Integrate}
                }
 Generate output
End
```

We use C program with some other arguments such as a switch which tells the compiler that the code should be distributed over a network, IP address to specify network machines available for compilation etc. DGC uses distcc, an open source distributed C/C++ compiler using GCC compiler that sends preprocessed source codes over the network and volunteer machines in compilation of the source code.

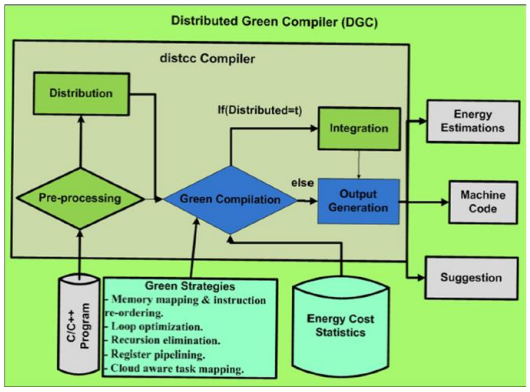

Fig. 1. Work flow of Distributed Green Compiler

The above figure shows the workflow of the DGC. Green Compilation and output generation modules of ditcc are modified to perform energy cost statistics and green strategies. DGC acquires the source file from C source project, then it pre-processes the files by including needed header files and libraries . Next step is to distribute the source code, this can be done by spreading the files on various slaves for compilation. The request will be sent to the machine which is available first . In the first run, the source code is selected block by block and the green strategies are applied on the selected code.

The features of DGC are:

1. *Loop Optimization:* DGC is capable of loop unrolling through funroll-all loops and fvariable-expansion-in-un roller switches of GCC compiler. Funroll-all-loops switches conserves energy by reducing the number of iterations of a loop, while fvariable-expansion-in- unroller copies local variables during loop unrolling for dependency elimination.
2. Dead code Elimination: The process of removing code resulting in no change in program result. This saves energy by saving the clock cycles by shrinking the programs which helps in avoiding implementation of unnecessary operations.
3. Software Pipelining: This can be used for loop optimization when overlapping iterations occur. DGC uses Modulo Scheduling to perform software pipelining.
4. Elimination of recursion: DGC converts recursion into iterations as needed. If conversion is not do-able then the DGC recommends green techniques by emphasising the specific areas of code that come under the same category.
5. Cloud Aware Task Mapping: DGC uses many clusters of virtual machines for distributed compilation process by using hardware and software resources to process a compilation.
6. *Energy Cost Statistics:* DGC maintains energy cost table for every executed instruction in a database. Using energy cost statistics database we can create different parse tree of selected code. In this paper, implementation details of DGC are not a part of this research. For the proposed concept, a conservation of 40 to 60 % energy can be reached If we implement green techniques.

The paper “Improved Unsupervised Framework for Solving Synonym, Homonym, Hyponym and Polysemy Problems from Extracted Keywords and Identify Topics in Meeting Transcripts” discusses Unsupervised Hidden Topic Framework which consists of the following steps:

A. Meeting transcripts

These are the text file consisting of the meeting speech in readable format. Nuance Dragon Naturally Speaking conversion software tool takes the audio dialogue from the meeting as an input and speech is converted readable (written) text file. The software is trained with some data, before conversion to text.

B. Data Pre-processing

Here text file is taken as an input, in the file stem and stop words are removed to give only the words that have meaning. Then using tf-idf frequency of each word is evaluated.

C. LDA Model

Latent Dirichlet allocation (LDA) is a generative probabilistic model of a corpus. LDA documents are displayed as random mixtures over latent topics, in which each topic is characterized by a distribution over words.

D. Synonym Problem

From LDA model similar words are grouped and their topic inference is made. Each similar group of words contains single topic and that topic is extracted as the output of this problem. By extracting topics, the synonym problem was solved.

E. Hyponym Problem

Hyponymy problem means one word denoting subclass that is considered and super class keyword is extracted. The word’s subclass is considered and its super class is extracted as the output. By extracting super class of each word, Hyponymy problem is solved.

F. Homonym Problem

In LDA model, keywords are collected under hidden topics. These topics are labelled with generalised concept. Homonym keywords are discovered by comparing with hidden topics keywords. The topic names give context of keywords and later calculated the frequency that is used to extract. The outcome of this are keywords and various meaning words.

G. Polysemy problem

Different keywords are presented in the meeting transcripts.

Related meaning keywords are identified by comparing with hidden keywords. These identified keywords are used for MaxEnt classifier.

H. MaxEnt Classifier

Maximum Entropy is a general technique for estimating probability distribution from data. The distribution is done as evenly as possible when nothing is known and will have Maximal Entropy. Labelled training data can be utilised to set constraints. Max Entropy uses single observation; it extracts features and groups to one set. It is very fast in both training and inference. Max Entropy classifier is trained and on the basis of probability estimation, high probability keywords are extracted from the meeting transcript.

I. Topic Extraction

Topic Extraction means extracting overall topic of the transcript. First, labeled test data has been prepared that will contain topic name and keywords. This labeled data is used for topic extraction that is, labeled data compared with transcript keywords. The topic extraction can be done using LDA model. Most of the keywords in transcript are presented in a particular topic. That topic can be extracted as overall topic of the transcripts.

J. Trained Dataset

Trained Dataset is used to collect all the types of words, their actual, related meaning, super class meaning to the particular word from the dictionary and to train all the types of words in the dataset. In this proposed framework, it will compare any type of untrained extracted keywords to the dataset and then it automatically solves the problem.

In the paper, "Translation of English Algorithm in C program using Syntax Directed Translation Schema", the application has been developed with the following system modules

1. A Flex scanner for comparing the input string and to generate similar tokens.
2. A Bison parser generator to input string syntax and generate similar semantic action.
3. A Gcc Compiler link Flex and Bison generated lex.yy.c and Y.tab.c to generate the final executable file.

   An executable file to generate the output.

A flex program basically comprises of a list of regexps with instructions on how to proceed when the given input matches the ones stored in the system.

A parser is a program which determines if its input has valid syntax and thereby decides its structure. Bison is a general-purpose parser generator that converts an annotated context-free grammar into a deterministic LR or generalised LR (GLR) parser employing LALR (1) parser tables.

The initial section of the Bison file contains optional, ordinary C subroutine declaration part, a list of tokens (other than single characters) that are expected by the parser and the specification of the start symbol of the grammar. The next section of the Bison file consists of the context-free grammar for the language.

4. Conclusion

In conclusion, Green Compiler is a software level technique to save energy. This uses several techniques to mould source code during intermediate code conversion to create energy conservative executables. In this paper, several methods for green compiler are highlighted to adopt energy conservation programs. By using the DGC we can distribute the source code over the network of physical or virtual machines. Analysis of the performance shows DGC clock cycles by 30% to 40% when green strategies are utilised. So future works in this field can be to optimise the compiling techniques for better performance analysis and detailed study of performance analysis of DGC with existing compilers, after completion of its prototype. A future study will be to search for prospects and use of green strategies to make the green compilers greener and more efficient so that in addition of efficient utilisation of resources, power consumption and Carbon dioxide emission can also be reduced.

Topic and keywords can be extracted more accurately by solving problems of Synonym, Hyponym, Homonym and Polysemy. In the future the problems will solve even untrained occurring keywords using proposed trained dataset. Using the proposed trained dataset, it can solve problems from any type of trained/untrained occurrence of keywords.

Evaluation of the system is cumbersome as there is no standard benchmark suite that would permit the comparison of both the systems. And hence, the system is evaluated with a simple test

suite. In this application, the system was tested with basic C programming concepts. The system successfully returned accurate and meaningful translations in all of the cases. The system handles ambiguity of similar meaning words based on lexical category. Case sensitivity of the words is not handled by the system since the compressed dictionary is considered and no morphological phase is added. The system output can be enhanced by including larger dictionary and morphological phases.

References

[1] F . Fakhar, B . Javed, R . u. Rasool, O . Malik, K . Zulfiqar: Software level green computing for large scale systems
[2] Raghavan P, Lambrechts A, Absar J, Jayapala M, Catthoor F, Verkest D (2008) COFFEE: Compiler Framework for Energy-Aware Exploration. HiPEAC'08 Proc 3rd Int Conference High Perform Embedded Architectures Compilers.
[3] Cloud Computing., June 2011
[4] Naik K (2010) A Survey of Software Based Energy Saving Methodologies for Handheld Wireless Communication Devices. Tech. Report No. 2010-13. Dept. of ECE, University of Waterloo
[5] Sheeba J.I, Vivekanandan K : Improved Unsupervised Framework for Solving Synonym, Homonym, Hyponym and Polysemy Problems from Extracted Keywords and Identify Topics in Meeting Transcripts, IJCSEA, October (2012).

[6] Feifan Liu, Deana Pennell, Fei Liu: Unsupervised Approaches for Automatic keyword extraction, Boulder, Colorado, ACM, June (2009).
[7] C.D. Manning, P. Raghavan, and H. Schutze,: Introduction to Information Retrieval, Cambridge Univ. Press, Springer (2008).
[8] S. Deerwester, G. Furnas, and T. Landauer: Indexing by Latent Semantic Analysis, J. Am. Soc. for Information Science, 1990
[9] T.A. Letsche and M.W. Berry: Large-Scale Information Retrieval with Latent Semantic Indexing, Information Science, ACM, 1997
[10] Su C-L, Tsui C-Y, Despain AM, 2002 Low Power Architecture Design and Compilation Techniques for High-Performance Processors. Compcon Spring '94, Digest of Papers.
[11] Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman "Compilers: Principles, Techniques, and Tools" Publisher: Addison Wesley, 1986.
[12] Biermann, Alan W., and Bruce W. Ballard. "Toward natural language computation." Computational Linguistics 6, 1980.
[13] Ballard, Bruce W., and Alan W. Biermann. "Programming in natural language: "NLC" as a prototype." In Proceedings of the 1979 annual conference, 1979.
[14] Carlos, Cohan Sujay. "Natural Language Programming Using Class Sequential Rules." In IJCNLP, 2011.
[15] Christiansen, Morten H., and Nick Chater. "Connectionist natural language processing: The state of the art." Cognitive science 23, 1999
[16] Cozzie, Anthony, and Samuel T. King. "Macho: Writing programs with natural language and examples." Technical report, University of Illinois at Urbana-Champaign, 2012.
[17] Cozzie, Anthony, Murph Finnicum, and Samuel T. King. "Macho: Programming with man pages." Proceedings of the 2011 Workshop on Hot Topics in Operating Systems, 2011.
[18] D.Jurafsky, J.H Martin. Speech and natural language processing. India: Pearson Education, 2000.
[19] Dan W. Patterson, Introduction to Artificial Intelligence and Expert System, PHI, 2001.
[20] Donnely, C., and Richard Stallman. "Bison: The Yacc-compatible parser generator, 2006." 2008.
[21] Elaine and Kevin Knight, Artificial Intelligence, McGraw Hill companies Inc., 2006.
[22] Igo, Sean, and Ellen Riloff. "Learning to Identify Reduced Passive Verb Phrases with a Shallow Parser." In AAAI, 2008.