HAIL: A Hardware-Accelerated Algorithm for Language Identification;
by Charles M. Kastner, G. Adam Covington, Andrew A. Levine, John W. Lockwood;
15th Annual Conference on Field Programmable Logic and Applications (FPL); August, 2005

HAIL: A HARDWARE-ACCELERATED ALGORITHM FOR LANGUAGE IDENTIFICATION

Charles M. Kastner, G. Adam Covington, Andrew A. Levine, John W. Lockwood

Applied Research Laboratory
Washington University in St. Louis
One Brookings Drive, Campus Box 1045
St. Louis, MO 63130-4899 USA
email:{cmk2, gac1, aal1, lockwood}@arl.wustl.edu
`http://www.arl.wustl.edu/projects/fpx/reconfig.htm`

ABSTRACT

A hardware-accelerated algorithm has been designed to automatically identify the primary languages used in documents transferred over the Internet. The algorithm has been implemented in hardware on the Field programmable port extender (FPX) platform. This system, referred to as the Hardware-Accelerated Identification of Languages (HAIL) project, identifies the primary languages used in content transferred over Transmission Control Protocol (TCP) / Internet Protocol (IP) networks that operate at rates exceeding 2.4 Gigabits/second. We demonstrate that this hardware accelerated circuit, operating on a Xilinx XCV2000E-8 FPGA, far outperforms software algorithms running on modern personal computers while maintaining extremely high levels of accuracy.

1. INTRODUCTION

Although modern microprocessors continue to increase in performance, they are not increasing in performance as fast as the rate of streaming Internet data. As the limits of Moore's Law are reached, it is quite possible that the difference in performance improvements will increase further.

Reconfigurable logic can process network traffic at rates much faster than what is achievable with microprocessor-based systems. Systems created from Field Programmable Gate Arrays (FPGAs) are flexible, as they can be easily modified to provide new functionality.

This paper presents the design of HAIL, a system implemented in FPGA hardware that classifies documents travelling across a network based on their language. The FPX implementation of HAIL is capable of identifying up to four languages used in the same document, and can be programmed to recognize up to 255 different languages. Through experimentation, we will show that this implementation of HAIL is capable of processing data at OC48 rates (2.488 Gigabits/second) while maintaining accuracy levels rivaling existing language classification methods.

This research was sponsored by the Air Force Research Laboratory, Air Force Materiel Command, USAF, under Contract Number MDA972-03-9-0001. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of AFRL or the U.S. Government.

1.1. Motivation

As of 2004, nearly two-thirds of the world's Internet users speak a language other than English as their primary language [1], and nearly one-third of the pages available on the World Wide Web are written in a language other than English [2]. As the rate that data is transferred over the Internet increases, the rapid identification of languages becomes a more difficult problem. A system capable of quickly identifying the primary language or languages used in documents can be useful as a preprocessor for document classification and translation services. It can also be used as a mechanism to winnow documents written in unwanted languages from a data stream.

1.2. Related Work

Several methods can be used to classify document characteristics using principles from linguistics and artificial intelligence. Some existing methods of classifying documents use dictionary-building [3], Markov Models, tri-gram frequency vectors [4], and/or n-gram based text categorization [5, 6]. While these methods are capable of achieving high degrees of accuracy, most require floating-point mathematics, large amounts of memory, and/or generous amounts of processing time.

Commercial software applications have been developed that can identify a document's language. One such tool is the Lextek Language Identifier [7], which can categorize a document into one of 260 different languages. Although Lextek is useful, it is computationally intensive, performs all computations in software using a microprocessor, and cannot process data at the rates found on high-speed networks.

2. ALGORITHM

Some language-identification algorithms, including the algorithm implemented in HAIL, use *n-grams* to determine the language of a document. N-grams are sequential patterns of exactly *n* characters that are found in written documents. By using n-grams as indicators of language, the primary language or languages of a document can be reliably determined. HAIL could use any n-gram length, although experiments have shown that n-grams of length three (tri-grams) and length four (tetra-grams) provide the most accurate results.

Prior to processing data with HAIL, the target system must be trained with information on various languages. This is accomplished by training on a set of documents in the languages of interest; if an n-gram appears significantly more frequently in documents of one language than any other, it is associated with that language. After training has established which n-grams best correspond to particular languages, memory modules on the hardware platform implementing HAIL must be programmed. This is performed by using a hash to map each n-gram to a particular memory location. The memory location that corresponds to a particular n-gram is labelled with the associated language. Once data processing begins, n-grams are sampled from the data stream and used as addresses into memory to discern the language associated with the n-gram.

2.1. N-gram Extraction

N-grams can be subsampled if memory bandwidth is small compared to that of the incoming data link. The subsampling scheme used by HAIL ignores n-grams that cross word boundaries, and extracts only those n-grams that begin on certain letter offsets within a word. For instance, if one memory access can be made for every two characters that arrive in the system, the n-grams that begin on a word's first, third, fifth, etc. letters would be extracted. Likewise, if one memory access could be made for every three characters that arrive in the system, the n-grams that begin on each word's first, fourth, seventh, etc. letters would be extracted. This subsampling scheme causes a given word to be processed in the same manner whenever it appears.

When working with a single character set, several other optimizations can be made to reduce the amount of memory needed to store n-grams. First, case conversion can be performed; in addition to reducing the amount of required memory, this practice allows n-grams to be identified consistently, regardless of the capitalization of individual letters. The amount of memory required can be reduced further by mapping accented characters (such as ñ and é) to their non-accented forms (n and e). By using these two methods, the Latin alphabet can be compressed into twenty-seven characters (twenty-six letters plus a character representing a delimiter) for use in HAIL. This allows each character to be represented with five bits.

As an example of HAIL's operation, consider an implementation in which n-grams of length four (tetra-grams) are used, one memory access can be performed for every two characters that enter the system, and case conversion and accent mapping are performed. If the Spanish phrase *Hablamos Español* were passed into the system, HAIL would extract the tetra-grams *HABL*, *BLAM*, *AMOS*, *ESPA*, and *PANO*. From a statistical analysis, we found that the tetra-grams *HABL*, *AMOS*, and *PANO* are strong indicators of the Spanish language, while *BLAM* is a strong indicator of the English language. Since the count of Spanish tetra-grams exceeds that of English tetra-grams, this phrase could be correctly classified as Spanish. We will show in Section 4 that HAIL provides a high level of accuracy for large numbers of languages and for documents of many sizes.

2.2. Counting

As mentioned in Section 1, our implementation of HAIL can tag packets and flows with up to four language IDs, while drawing from a pool of up to 255 different languages. To efficiently track which four of the 255 languages appear in the document, HAIL uses a *trend register*. HAIL implements four on-chip counters for language identification. The trend register determines which languages should utilize these counters.

This trend register tracks successive n-grams from the same language. If several such n-grams arrive (three successive n-grams were found to be the optimal number), the corresponding language is granted use of one of the four language counters. Once a language is associated with one of these counters, any occurrence of a tetra-gram in that language increments the counter. Thus, up to four languages that establish trends within a given document are represented.

3. IMPLEMENTATION

HAIL has been implemented on the Field-programmable Port Extender (FPX), an open hardware platform that allows circuits to be rapidly prototyped and implemented in reconfigurable hardware [8]. The FPX is used extensively for the fast processing of streaming network data.

Circuits developed for the FPX consist of VHDL specified modules that include a common infrastructure as well as user-defined logic that performs customized content processing [9]. FPX circuits exchange data over the network using a set of layered Internet Protocol wrappers [10]. The Transmission Control Protocol / Internet Protocol wrappers allow

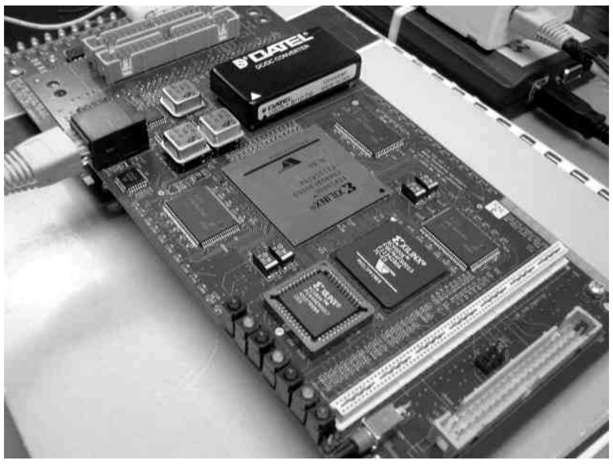

Fig. 1. The FPX platform used to implement HAIL

the FPX to directly process streaming TCP/IP network traffic flows in hardware [11].

Once an application has been implemented for use on the FPX, a bitfile is uploaded to the hardware over a network. A Xilinx XCV2000E-8 FPGA implements the user-defined logic on the FPX. A photograph of the FPX system is shown in Fig. 1. The HAIL module runs on the large FPGA in the middle of the FPX circuit board.

Information enters the HAIL system as IP packets transmitted across a network link. Documents sent across the link are split into one or more packets, which can arrive in the system interleaved with packets from other documents. Packets entering the system first pass through an FPX module that performs TCP/IP stream assembly and protocol annotation [11]. This module maintains the state of each active network traffic flow in off-chip SDRAM, annotates packets with flow state information, and ensures that data entering the HAIL module is properly ordered.

This implementation of HAIL also uses SDRAM to maintain TCP flow context information. Each time that the system processes a packet of a TCP flow that has not yet terminated, intermediary language IDs and counters are stored in off-chip memory for retrieval when the next packet from the flow arrives.

3.1. Engineering Trade-offs

The FPX is designed to accept four bytes of data per clock cycle. When processing streaming data on a high-speed network link, it is assumed that new data can arrive on every clock cycle. In order to reliably process all n-grams that appear at every byte offset in a document, the system would need to be capable of performing four memory lookups per cycle.

The Xilinx XCV2000E-8 FPGA on the FPX contains 160 small, single-cycle latency, dual-port memories called BlockRAMs. Each BlockRAM contains four kilobits of storage, providing a total of 640 kilobits of on-chip RAM. Although BlockRAMs have been used in other FPX circuits to efficiently perform string matching operations [12, 13, 14], HAIL must store more strings than could fit into BlockRAM.

The FPX platform is also equipped with a pair of two Megabyte, zero-bus turnaround (ZBT) SRAMs. Each of these memory devices can perform a single read or write operation during each clock cycle. By configuring the two SRAMs with identical dictionaries of n-grams, two parallel lookups can be performed per clock cycle.

Since two n-gram lookups can be performed per clock cycle, and data enters the FPX at the rate of four bytes per clock cycle, only half of the n-grams in words of length greater than *n* can be looked up.

Using the scheme described in section 2.1, tetra-grams (n-grams with a length of four) can represent each character in the Latin alphabet with five bits. Thus, a tetra-gram can be represented with twenty bits; since each SRAM on the FPX is two megabytes in size, every possible tetra-gram could fit into SRAM. Furthermore, we experimentally determined that tetra-grams performed approximately 1.5% better than n-grams with a length of three, and significantly better than n-grams of other lengths.

3.2. Architecture

The architecture of the HAIL system as implemented in the FPX is shown in Fig. 2. It consists of eleven discrete components, which are pipelined to provide high throughput. A brief description of major components appears below.

3.2.1. Tetra-gram Generator

This module processes bytes from the TCP data stream that represent letters in the extended ASCII character set. It converts all letters to their uppercase unaccented form, then compresses the letters into a five-bit format. The stream of letters is inserted into a shift register, and the circuit extracts tetra-grams that do not cross word boundaries and that begin on odd-numbered character offsets within a word.

3.2.2. SRAM Reader

This module uses tetra-grams as addresses into the SRAM dictionaries. Up to two lookups into the dictionaries are performed each clock cycle; the data returned from each lookup is an eight-bit language identifier, which is then passed to the count and score module described next.

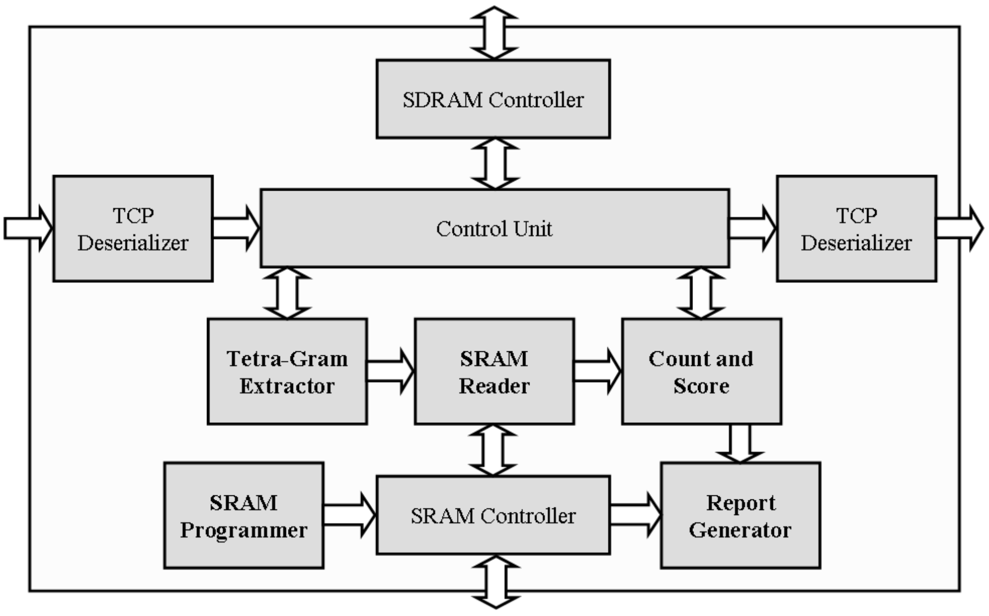

Fig. 2. Block diagram of HAIL's basic architecture

3.2.3. Count and Score

This module uses the method described in Section 2.2 to track the trends of languages seen in tetra-grams. After processing the contents of a packet, the module annotates the packet data with its primary languages and passes the data to the TCP reserializer. Once the content of an entire TCP flow is analyzed, the module transmits information regarding the flow's language or languages to the report generator described next.

3.2.4. Report Generator

HAIL records the source and destination IP addresses, source and destination ports, and the counters of the four primary languages used in a TCP flow. Upon termination of a flow, the report generator formats this data into a UDP packet and transmits it to a PC.

3.2.5. SRAM Programmer

Communication between the FPX and external devices is performed via control packets transmitted over an IP network. In order to program dictionaries into SRAM and to set other options within HAIL, an external computer transmits formatted UDP/IP packets to the FPX. This module decodes the commands contained in the UDP packets and uses them to program the SRAM dictionaries.

4. EXPERIMENTS

We performed several experiments in which the FPX implementation of HAIL was compared to three other algorithms. A set of documents written in extended ASCII was obtained from the LDC corpora. This set contained documents written in Arabic, English [15], Dutch [16], French [17], German [18], Italian [19], Norwegian [20], and Spanish [21].

For the experiments, we measured the accuracy of the algorithms as a function of the amount of training data. In each experiment, the number of words used for training was varied from five hundred to five thousand words from each language of interest. The training documents were chosen at random from the set of documents in each language.

Test sets in each experiment consisted of one hundred documents in each of the eight languages; testing data was chosen at random from the set of documents not used in training. Data presented is the average over ten runs of testing and training.

The four algorithms used in the experiments are outlined below.

HAIL: The algorithm outlined in this paper.

Dictionary: All words found in the training data were associated with the respective language. Testing data was labelled as the language with the highest number of words in the document.

Tetragrams: All sets of four letters that do not cross word boundaries were extracted from the training data. The probability that a tetra-gram identified a particular language was computed for all tetra-grams across all languages. A

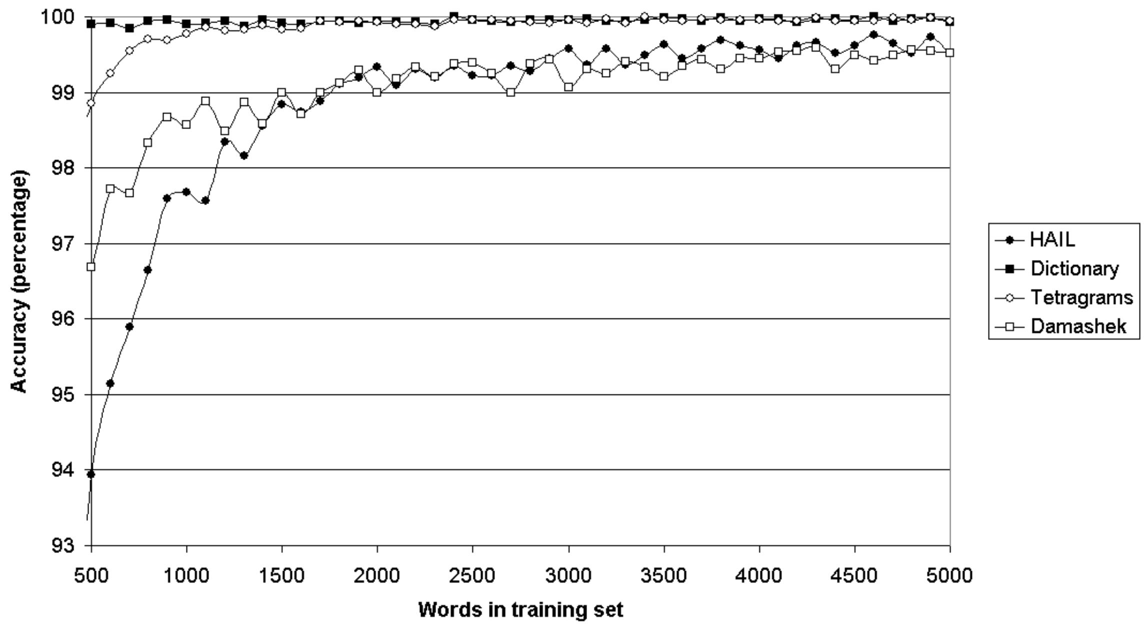

Fig. 3. Experimental results for language identification schemes using documents of one hundred words each.

testing document was scored based on the sum of probabilities of the tetra-grams it contained.

Damashek: This is an implementation of Damashek's topic and language identification algorithm [5]. Training was performed on n-grams of various lengths throughout a document, and n-grams were given scores based on their contribution to each language. Scores from the n-grams found in testing documents were used to compute the probability that a document is written in a specific language.

Experiments were run in which the average document size was varied from one hundred words to one thousand words. For document sizes of five hundred words or more, all four algorithms in consideration provided accuracy greater than 99 percent for training sets as small as one thousand words.

For testing document sizes of one hundred words, the algorithm used in HAIL was over 94 percent accurate for training sets of one thousand words. However, when the number of training words grew to five thousand, HAIL's accuracy was nearly identical to that of the other algorithms. The results of this experiment are shown in Fig. 3.

4.1. Performance and Device Utilization

The circuit implementing HAIL was placed and routed at 90 MHz for the Xilinx XCV2000E-8 found on the FPX board. As detailed in Table 1, HAIL requires approximately one-quarter of the logic slices on the Xilinx XCV2000E-8. This figure also includes infrastructure for memory controllers and modules for communication between HAIL and external devices.

The circuit implementing HAIL requires an overhead of 30 clock cycles per packet, during which data for the incoming packet's TCP flow is retrieved from SDRAM. A typical TCP/IP packet contains 40 bytes of header data. Data enters the FPX four bytes at a time, so the 40 bytes of packet overhead consume 10 clock cycles of processing time. In total, each packet that enters the FPX implementation of HAIL utilizes 40 clock cycles to retrieve flow context and to process the header.

For 1500 byte packets (the MTU on most Internet links), HAIL can process 324 Megabytes (2.5 Gigabits) of payload data per second, a rate that exceeds the maximum throughput of a fully-utilized OC48 link.

For comparison, a program was written using an application programming interface (API) provided by Lextek [7], a company that provides a language identification tool. The API is accompanied by files that the provided functions use to identify the languages used in a document. The API allows the programmer to utilize multiple language identification files; however, the tool's throughput decreases as the number of language identification files increases.

The program was executed on a PC containing an Athlon 64 3200+ processor and one gigabyte of PC2100 DDR SDRAM. The number of language identification files used was varied from one to ten, and the software was run on 4.5 Megabytes of text. Measurements were taken only after the data had been loaded into memory.

Table 1. Device utilization for HAIL

Resources	XCV2000E-8 Utilization	Utilization Percentage
Slices	5262 out of 19200	27%
4-input LUTS	7157 out of 38400	14%
Flip Flops	5379 out of 38400	18%
Block RAMs	88 out of 160	55%

The software utilizing Lextek had an average throughput of 3.8 Megabytes (30.4 Megabits) per second when only one language identification file was open. As the number of these files was increased to ten, performance decreased linearly to 1.2 Megabytes (9.6 Megabits) per second. For 1500-byte TCP/IP packets, HAIL is 85 to 270 times faster (depending on the number of language modules loaded into Lextek). Furthermore, HAIL's throughput does not change when more languages are programmed into it.

5. CONCLUSION

HAIL, a hardware-accelerated algorithm that identifies the language used in documents sent over the Internet, has been implemented and tested on the FPX platform. This implementation of HAIL processes content in a TCP/IP network traffic flow and tracks the four most-used languages from a set of up to 255 languages. With only a few thousand words of training data, HAIL achieves accuracy exceeding 99 percent for documents as small as one hundred words. As document size increases, HAIL is capable of identifying document languages over 99.95 percent of the time. HAIL processes network payload data at up to 2.5 Gigabits per second, a rate much higher than that of algorithms designed for execution in software. The circuit uses 14% of the lookup tables and 55% of the on-chip Block RAMs in Virtex XCV2000E-8 FPGA on the FPX platform. The additional logic can be used for future expansions, including support for processing additional character sets.

6. REFERENCES

[1] Global Reach, "Global Internet Statistics By Language," Online: `http://www.glreach.com/globstats-/index.php3`, Dec. 2004.

[2] ——, "Global Internet Statistics: Sources and References," Online: `http://www.glreach.com/globstats-/refs.php3`, Dec. 2004.

[3] R. Paulsen and M. Martino, "Word Counting Natural Language Determination," U.S. Patent 6,704,698, 1996.

[4] J. Schmitt, "Trigram-Based Method of Language Identification," U.S. Patent 5,062,143, 1990.

[5] M. Damashek, "Method of Retrieving Documents that Concern the Same Topic," U.S. Patent 5,418,951, 1994.

[6] R. Shaner, "Method of identifying data type and locating in a file," U.S. Patent 5,991,714, 1998.

[7] Lextek International, "Lextek Language Identifier," Online: `http://www.languageidentifier.com`, Dec. 2004.

[8] J. W. Lockwood, "An open platform for development of network processing modules in reprogrammable hardware," in *IEC DesignCon'01*, Santa Clara, CA, Jan. 2001, pp. WB–19.

[9] J. Lockwood, J. Turner, and D. Taylor, "Field Programmable Port Extender (FPX) for Distributed Routing and Queuing," in *ACM International Symposium on Field Programmable Gate Arrays (FPGA)*.

[10] F. Braun, J. Lockwood, and M. Waldvogel, "Layered Protocol Wrappers for Internet Packet Processing in Reconfigurable Hardware," *IEEE Micro*, vol. Volume 22, no. Number 3, pp. 66–74, Feb. 2002.

[11] D. Schuehler and J. Lockwood, "A Modular System for FPGA-based TCP Flow Processing in High-Speed Networks," in *14th International Conference on Field Programmable Logic and Applications (FPL)*, Antwerp, Belgium, Aug. 2004, pp. 301–310.

[12] S. Dharmapurikar, P. Krishnamurthy, T. Sproull, and J. Lockwood, "Deep Packet Inspection Using Parallel Bloom Filters," in *11th Annual IEEE Symposium on High Performance Interconnects (HotI)*, Stanford, CA, Aug. 2003, pp. 44–51.

[13] S. Dharmapurikar, P. Krishnamurthy, and D. Taylor, "Longest Prefix Matching Using Bloom Filters," in *ACM Special Interest Group on Data Communications (SIGCOMM)*, Karlsruhe, Germany, Aug. 2003.

[14] M. Attig, S. Dharmapurikar, and J. Lockwood, "Implementation Results of Bloom Filters for String Matching," in *IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM)*, Napa, CA, Apr. 2004.

[15] "Arabic English Parallel News Part 1," Philadelphia: Linguistic Data Consortium, 2005.

[16] Groningen and The Institute of Dutch Lexicology, "Dutch ECI Multilingual Text," Philadelphia: Linguistic Data Consortium, 2005.

[17] Le Monde, "French ECI Multilingual Text," Philadelphia: Linguistic Data Consortium, 2005.

[18] University of Munster, Frankfurter Rundschau, "German ECI Multilingual Text," Philadelphia: Linguistic Data Consortium, 2005.

[19] A. Wilkin and E. Burr, "Italian ECI Multilingual Text," Philadelphia: Linguistic Data Consortium, 2005.

[20] Bergen, "Norwegian ECI Multilingual Text," Philadelphia: Linguistic Data Consortium, 2005.

[21] Moreno, "Spanish ECI Multilingual Text," Philadelphia: Linguistic Data Consortium, 2005.

[22] Aggelos Bletsas, "Physical limitations on the expansion of Internet," Online: `http://web.media.mit.edu-/∼aggelos/861.html`, 1999.