Interactive Compression of Books

BRUNO CARPENTIERI
Dipartimento di Informatica ed Applicazioni "R. M. Capocelli"
Università di Salerno
Via S. Allende – 84081 Fisciano (SA)
ITALY
bc@dia.unisa.it http://www.dia.unisa.it/professori/bc

Abstract: In this paper we study interactive data compression and present experimental results on the interactive compression of textual data (books or electronic newspapers) in Italian or English language.
The main intuition is that when we have already compressed a large number of similar texts in the past, then we can use this previous knowledge of the emitting source to increase the compression of the current text and we can design algorithms that efficiently compress and decompress given this previous knowledge. By doing this in the fundamental source coding theorem we substitute entropy with conditional entropy and we have a new theoretical limit that allows for better compression. Moreover, if we assume the possibility of interaction between the compressor and the decompressor then we can exploit the previous knowledge they have of the source. The price we pay is a very low possibility of communication errors.

Key-Words: - Data Compression, Interaction, Dictionary based compression, Fingerprinting.

1 Introduction

Data Compression is the coding of data to minimize its representation. This process is called lossless compression (also reversible or noiseless coding or redundancy reduction) if the original can be exactly reconstructed from the compressed copy; otherwise it is called lossy compression (also irreversible or fidelity-reducing coding or entropy reduction.

The theoretical background of the data compression techniques is strong and well established. It dates back to the seminal work of Shannon who, more than half a century ago, gave precise limits on the performance of any lossless compression algorithm: this limit is the entropy of the source we want to compress.

Data compression techniques are specifically dependent on the type of data that has to be compressed and on the desired performance. Typical measures of performances are time and space complexity, fidelity, compression ratio. Typical data are text, images, speech, video.

Today state of the art lossless compressors are efficient. While it is not possible to prove that they always achieve the entropy limit, their effective performances for specific types of data are often very close to this limit.

One option we have to increase compression is to use the knowledge of similar messages from the same source that the two transmitting/compressing sides have compressed in the past and to design algorithms that efficiently compress and decompress given this previous knowledge.

By doing this in the fundamental source coding theorem we can substitute entropy with conditional entropy and we have a new theoretical limit that allows for better compression.

Moreover, if we assume the possibility of interaction between the compressor and the decompressor then we can exploit the previous knowledge they both have of the source. The price we might accept to pay is a very low possibility of communication errors.

In this paper we review recent work that applies previous knowledge and interactive approaches to data compression and discuss this possibility.

In the next Section we remind the relationship between compression and entropy. Section 3 reviews the dictionary based compression approaches. Section 4 introduces the interactive compression paradigm, Section 5 shows how to use interactive compression to compress books and Section 6 presents our conclusions.

2 Compression and Entropy

The amount of digital data being produced after the beginning of the second millennium is increasing at an exponential rate. To deal with digital data we need data compression: without compression we cannot store and/or transmit the huge amount of data we treat every day.

For example consider digital images and consider the number of bits per image resulting from typical sampling and quantization rates: a 24 x 35 mm negative photograph scanned at 12 µm, 3000 x 2000 pixels/color, 8 bits / pixel, 3 colors, uses about 144.000.000 bits and a LANDSAT Thematic Mapper scene, 6000 x 6000 pixels / spectral band, 8 bits / pixel, and 6 nonthermal spectral bands uses 1,7 x 109 bits. These images are often transmitted in groups, where each group contains sequences of images that are strictly correlated.
Data compression is therefore motivated by the economic and logistic needs to save space in storage media and bandwidth in communication: in compressed form we can rapidly and efficiently store and transmit data.
The theoretical background of data compression techniques is strong and established. It dates back to the seminal work of Shannon. This theoretical background gives precise limits on the performance of a compression algorithm.
Any process that generates information can be seen as a source that emits a sequence of symbols from a finite alphabet. For example we might consider an n-bit image as generated by a source with alphabet of 2n symbols representing the possible values for every n-bits combination.
The source output can be viewed as a sequence of: finite length sequences of alphabet symbols, often called *words*. For example, a word of a source that emits English sentences can be a sequence of one or more common English words, including blanks, punctuation marks, etc..
If we consider a Discrete Memoryless Source (DMS) S, in which successive symbols are statistically independent, Shannon and Weaver in [1] showed that, given the set of probabilities P = $\{p_1, p_2, ..., p_n)$ for the source symbols, the optimal expected number of code bits is:

$$\sum_{i=1}^{n} -p_i \log_2 (p_i)$$

a quantity which they called the Entropy of the source S, usually denoted by H(S).
The entropy is a measure of uncertainty about an event: it has a zero value, if and only if, there is absolute certainty and it is maximum when all the events are equiprobable: i.e. there is absolute uncertainty.
The conditional entropy $H(S_1|S_2)$ is the natural analogue of the entropy H(S) but it is based on the conditional probability. It can be shown that: $H(S_1|S_2) \leq H(S_1)$.
Because of the fundamental source coding theorem (see Storer [2]) entropy is a limit to the length of the string that we can use to lossless code a message.
Today lossless compressors are very efficient, the performance of the state of the art compressors for specific data types are often close to this theoretical limit.

3 Dictionary based compression

In this paper we are going to deal with lossless, dictionary based, compression algorithms.
We can use static dictionary methods when the source is known in advance.
When we use data compression to communicate data, the Sender and the Receiver shall use the same (static) dictionary and at the beginning of the communication the Sender shall send the dictionary to the Receiver.
For example in vector quantization algorithms the compressor builds up a dictionary on a training set of images and it uses this dictionary to compress the new data. The decompressor needs the same dictionary to decompress, therefore the compressor shall transmit this dictionary to the decompressor.
The cost of making this dictionary available depends on its dimensions. Generally this cost is not considered in the analysis of a static dictionary compression method by using the argument that this constant cost can be amortized over time by compressing a large amount of data.
In real life situations this is not always true: in many cases the cost of sending the dictionary might have an important impact on the total cost of the communication making static compression methods unpractical.
Dynamic dictionary methods build up the Sender and the Receiver dictionary at run time, starting with empty dictionaries and building up the dictionaries in terms of the already compressed data.
In real life applications often we have as a compression target small or medium size files, that represent computer programs, images, text data, music, videos, etc..
It would be obviously more effective to start up the compression/decompression process with an appropriate, complete, dictionary instead of an empty one.
Mainstream methods that are based on dictionaries are almost always dynamic dictionary methods: this adaptive technique is flexible and can be easily used in practical situations.

4 Interactive Compression

If the compressor and decompressor have already compressed a large number of source messages (but not necessarily the same messages, see for example Carpentieri [3]) and we can assume some degree of interaction between the two communication parties, then we can exploit the compression process when data compression is used for data transmission. The price we pay might be a low possibility of errors that derives by the usage that the two sides make of the knowledge they have of the source.

We can design interactive protocols that allow a Sender and a Receiver to take advantage of the knowledge they have of the source and that could exploit the interaction to minimize the communication cost.

This situation occurs in every day life. As an example consider an internet user that has download frequently a file (a newsletter, an image, a report, etc.) from the same source, or a system manager that has to download repetitively an update file for a program or an antivirus, or an ftp mirroring site that has to be periodically brought up to date, etc.

The compression algorithms involved might be static or dynamic dictionary methods, Vector Quantization in the case of images, etc..

El Gamal and Orlistky in [4] consider the following problem: “Given two random variables X and Y with entropies H(X) and H(Y) and joint entropy H(X,Y), and two persons PX and PY, such that PX knows X and PY knows Y, suppose that the two persons wish to communicate over a noiseless two-way channel so that at the end of the communication process they both know X and Y . How many bits on the average must they exchange and what are the optimal codes?”

They prove that at least $H(X|Y) + H(Y|X)$ bits must be exchanged on the average and that $H(X,Y) + 2$ bits are sufficient and that if the joint probability $p(x,y)$ is uniform then the average number of bits needed is close to $H(X|Y) + H(Y|X)$.

They also discuss randomized protocols that can reduce the amount of communication if some probability of communication error is acceptable.

In Carpentieri [5] it is presented a communication protocols that allow a “learned” Sender and a “learned” Receiver to communicate, and to compress files, with dictionaries that are initially independently built, starting by previous (and may be discordant) examples of communication messages each of the two sides has available.

This communication protocol results in an improvement in compression paid with a negligible or almost null possibility of communication errors

5 Interactive compression of books.

In this section we consider the interactive compression/transmission of natural language textual data, as electronic newspapers or books, which have to be sent from a remote source to a client destination.

We might improve the transmission/compression process by taking into account the common rules of the specific natural language (i.e. the set of common words of the natural language, the syntax, etc.).

All these rules might be considered as shared knowledge between a Sender (that sends the book) and a Receiver (that receives the data).

Another option, if the Receiver has already received a consistent number of messages form the source, is to use those messages to build, independently from the Sender that might not have the same messages available, a dictionary to be used in the decompression process.

Otherwise, when the Receiver receives the book it can use a standard on line dictionary of the language in which the book is written to decode the Sender’s messages.

The Sender does not know which (static) dictionary the Receiver is using.

The Sender can send the message words as pointers that the Receiver could try to decode by using its dictionary.

We have experimented with this dictionary based approach by digitizing six books: three of them are in the Italian language and the other three are in English.

We have used two standard, on line, dictionaries (an English dictionary and an Italian dictionary) in the decompression process.

In our experiments the Sender sends a book to the Receiver a word at a time.

If the word is long enough, instead of the raw word, the Sender sends a token including the initial character of that word, the length of the word and a hash value (in our implementation a Karp and Rabin hash, see Karp and Rabin [6]) for that word.

The Receiver will receive the token and will try to decode the word by using its local dictionary that the Receiver has already organized in terms of the hashing function used.

If the token sent by the Sender cannot be decoded by the Receiver because for the hash value it has received there is a collision in the Receiver’s dictionary, then the Receiver asks the Sender to send a fresh new token for the same word with a different letter (may be the middle letter of the word) and a different hash.

If the Receiver finds a unique word that matches the token in the dictionary then it decodes the word and eventually acknowledges the Sender.

If the Receiver does not find any word that matches the token in the dictionary, then it sends a request to

have the word (entropy coded and) sent directly as raw data to the Sender.
There is a possibility of communication errors, i.e. situations in which the Receiver believes it can decode correctly the word but the word that it finds in its dictionary it is not the correct one.
If the probability of an error is very low then the Receiver might accept to have a few words incorrectly decoded: it will be still possible to read and understand the book.
For example this might be the situation of an electronic newspaper that is constantly updated at the Receiver's side. If a few words are misspelled but the text is still readable it is not a big deal for the reader (that is already used to find a few misspelled words in the paper newspapers…).
If we assume that the dictionary used by the Receiver is the correct dictionary for that specific book language, then the probability of an error depends on the choice and length of the hash value.
Figure 1 outlines our communication model, where there is a communication line that goes from the Sender to the Receiver and an acknowledgment line that goes from the Receiver to Sender.
In many real life situations the communication line is bidirectional: there is the possibility of interaction between Sender and Receiver in communication via modem on a telephone line, on a radio or satellite link, etc.. Moreover often the bandwidth of the line is not fully used in a single direction but it is automatically divided in two separate lines of half bandwidth each to allow messages in both directions; for example this was the case of the previous generation modems.
In these cases it might even be considered that the acknowledgment line of our model, and all the tra±c on that line, comes almost for free, because it does not slow up the communication on the communication line. However in what we will assume that the cost of the communication between the Sender and the Receiver is the sum of the cost of the traffics on the two lines, assuming that the messages on the acknowledgment line have the same cost of the ones on the communication line.
In Table 1 we show the results obtained by compressing six books and by using Karp and Rabin hashes of length 12 bits.
The results obtained are compared with the "off of the shelves" standard dictionary based compressors Zip and Gzip.
These results are very encouraging: with this approach we have implicitly set a strong limit on the length of any matched string that is copied into the output stream (the maximum length of a match will be the length of the longest word in the static Receiver's dictionary) but nevertheless the results are competitive with respect to the standard zip and gzip compressors.
Possible improvements are therefore foreseen if we allow the matches to have longer lengths.
The price we pay is the possibility of small communication errors.
These communication errors shall not propagate as in standard dictionary based compression, because here we use a static dictionary.
With the above settings we have a very limited communication error: i.e. only a very few words (often less than ten) are mismatched in a whole book which therefore maintains its readability.
There is a strong relationship between the length of the hash value, the compression results, and the compression errors.
With a longer hash we have less compression errors but a worst compression, with a shorter hash we improve compression but we pay the price of more compression errors.
Figures 2, 3 and 4 picture out the relationship between the length of the hash value, the size of the compressed file and the error percentage for the books "Decamerone", "Promessi Sposi" and "Gomorra".
In the (a) part of the pictures we have on the x-axis the lengths of the hash value in bits and on the y-axis the corresponding sizes in Kb of the compressed file.
In the (b) part of the pictures on the x-axis we have again the lengths of the hash value in bits, but on the y-axis we have the corresponding error percentages.
Figure 2 shows that, in the case of "Decamerone" compression results are better for a hash value that has a length of 9 bits, but also that this length is not good enough to cope with transmission errors.
Instead a hash of length 15 bits gives a compression that is almost lossless but the compression obtained is not competitive.
The right compromise for the hash length here is 12 bits.
The behavior for the other books is very similar to the one in Figure 1.

6 Conclusion
If we assume the possibility of interaction between the compressor and the decompressor then we can exploit the previous knowledge they might have of the source. The price we pay is a very low possibility of communication errors.
In this paper we study interactive data compression and present experimental results on the interactive compression of textual data.

Future work will focus on the improvement of the algorithms presented and on a wider testing of the approaches described.

Acknowledgements

I would like to thank Prof. James A. Storer for fruitful discussions we had on interactive data compression, and my students Mario Immobile Molaro and Vincenzo Viola for carrying out some of the experimental work described in this paper.

References:

[1] C. S. Shannon and W. Weaver, The mathematical theory of communication. University of Illinois Press, Urbana, IL., 1949.

[2] J. A. Storer, Data compression: methods and theory. Computer Science Press, 1988.

[3] B. Carpentieri, Conditional data compression: the lossless case. *WSEAS Trans. on Systems*, Vol. 2, n. 4, 2003, pp. 856-860.

[4] A. El Gamal, A. Orlitsky, Interactive data compression. In: *Proceedings 25th Ann. Symp. Foundations Computer Science*, 1984, pp. 100–108.

[5] B. Carpentieri, Sending compressed messages to a learned receiver on a bidirectional line. *Information Processing Letters*, Vol. 83, n.2, 2002, pp. 63-70.

[6] R. M. Karp, M. O. Rabin, Efficient randomized pattern-matching algorithms. *IBM J. Res. Develop.* 31 (2), 1987, pp. 249–260.

[7] B. Carpentieri and J. A. Storer, Video Compression and The Complexity of Aligning Vectors, *International Journal of Foundations of Computer Science*, vol. 5, N. 2, 1994, pp. 165-177.

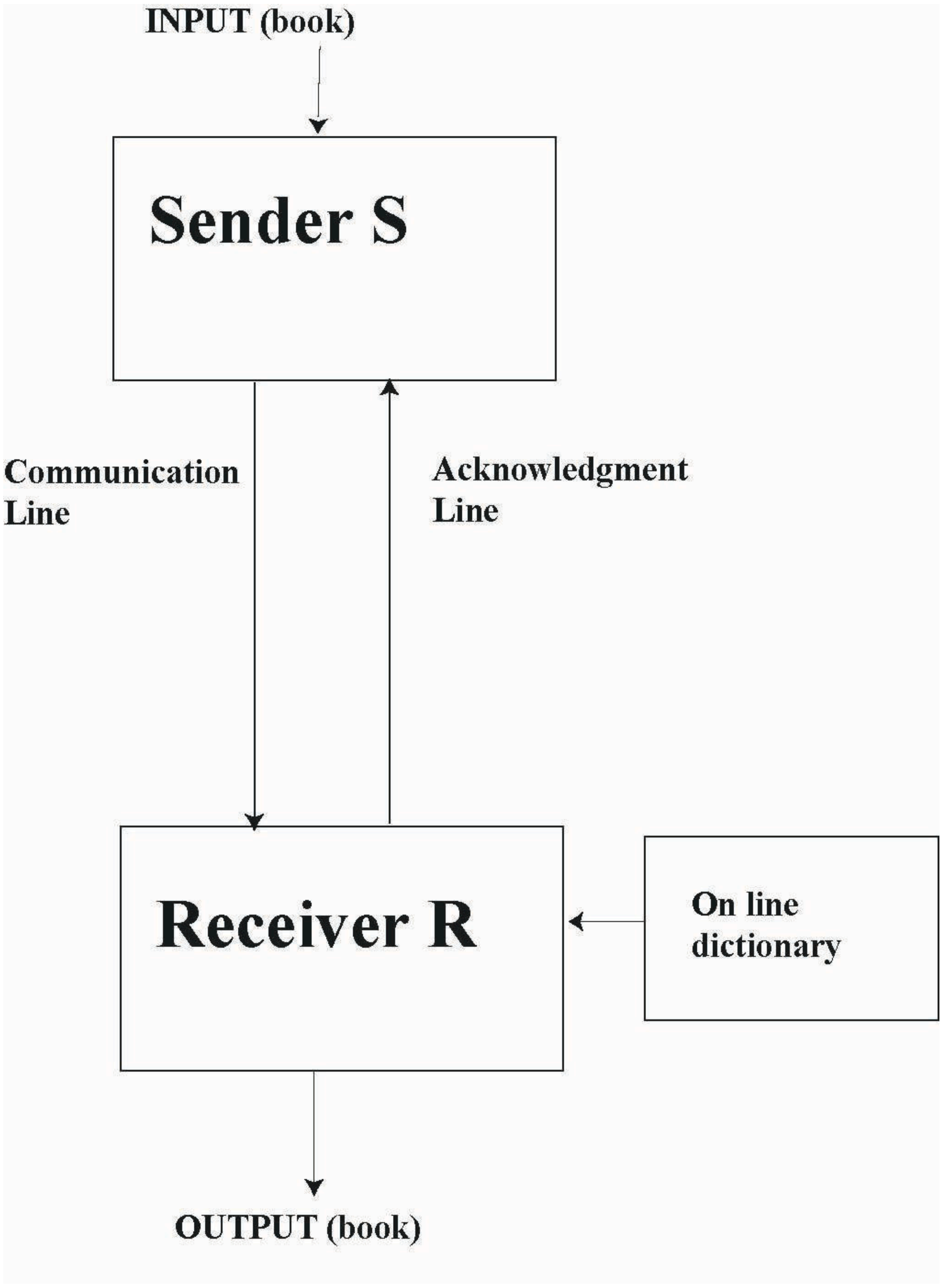
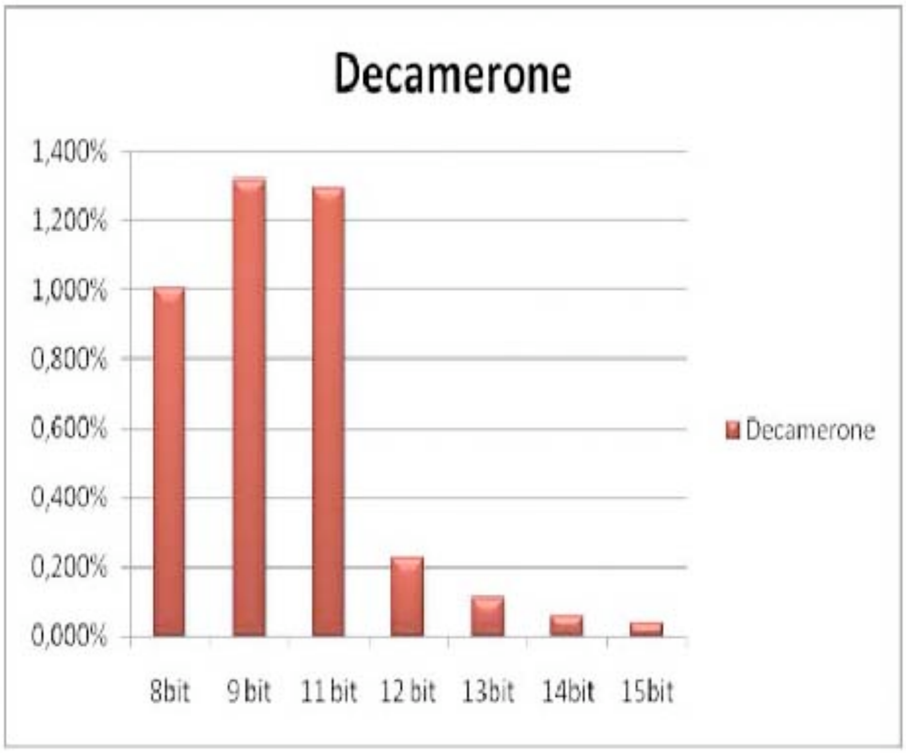


Figure 1: The Communication Model

Book title	Original Dimensions	Interactive Protocol (12 bits hash)		Zip		Gzip	
		Compresed size	Compression ratio	Compresed size	Compression ratio	Compresed size	Compression ratio
Decamerone	1638,2 KB	544,6 KB	3,00	577,2 KB	2,83	551,7 KB	2,96
Gomorra	663,0 KB	240,2 KB	2,76	251,4 KB	2,63	241,3 KB	2,74
Promessi Sposi	1394,9 KB	490,1 KB	2,84	520,2 KB	2,68	498,5 KB	2,79
20000 Legues Under The Sea	875,5 KB	306,2 KB	2,85	336,1 KB	2,60	323,4 KB	2,70
The Wealth Of Nations	2273,1 KB	602,9 KB	3.77	688,1 KB	3,30	652,8 KB	3.48
For Whom The Bell Tolls	937,4 KB	288,2 KB	3,25	326,3 KB	2,87	310,1 KB	3,02

Table 1. Compression results



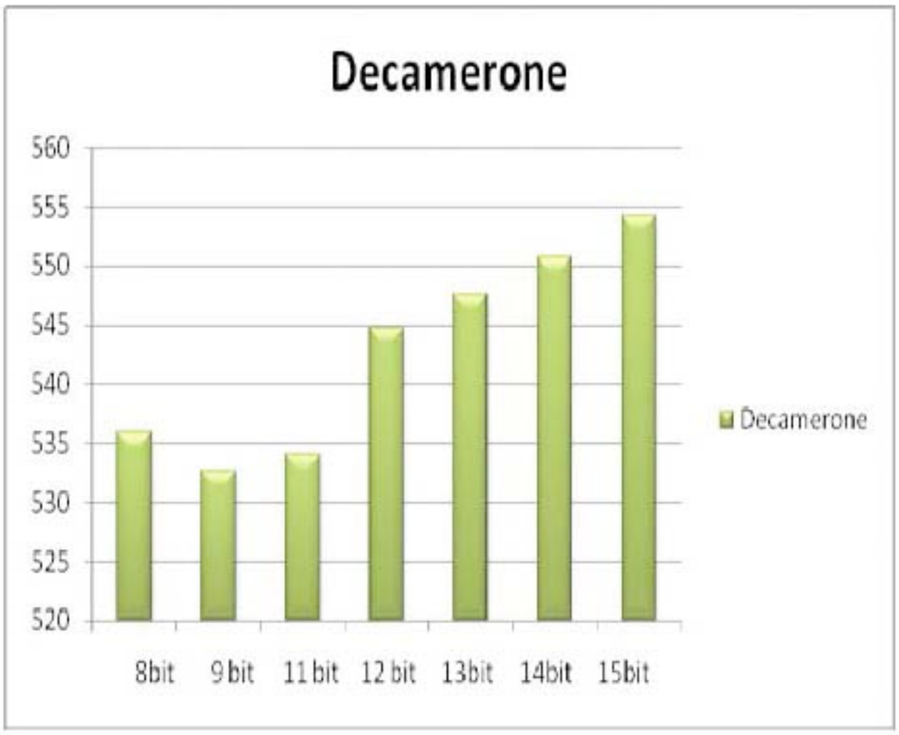
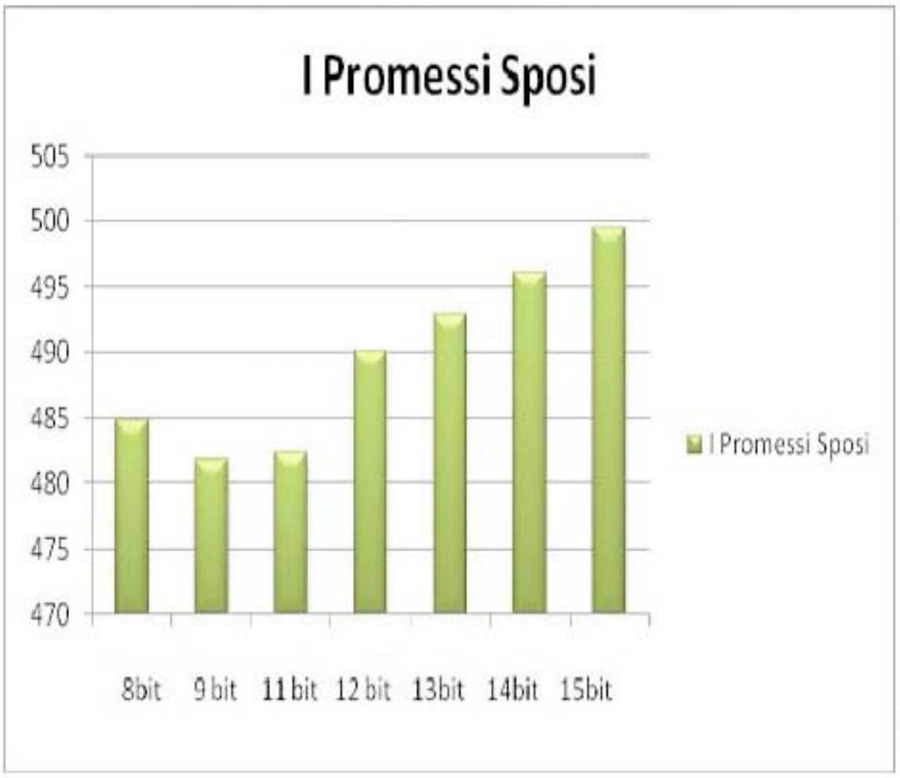


Figure 2 (a)-(b): Compression and hash length for "Decamerone"



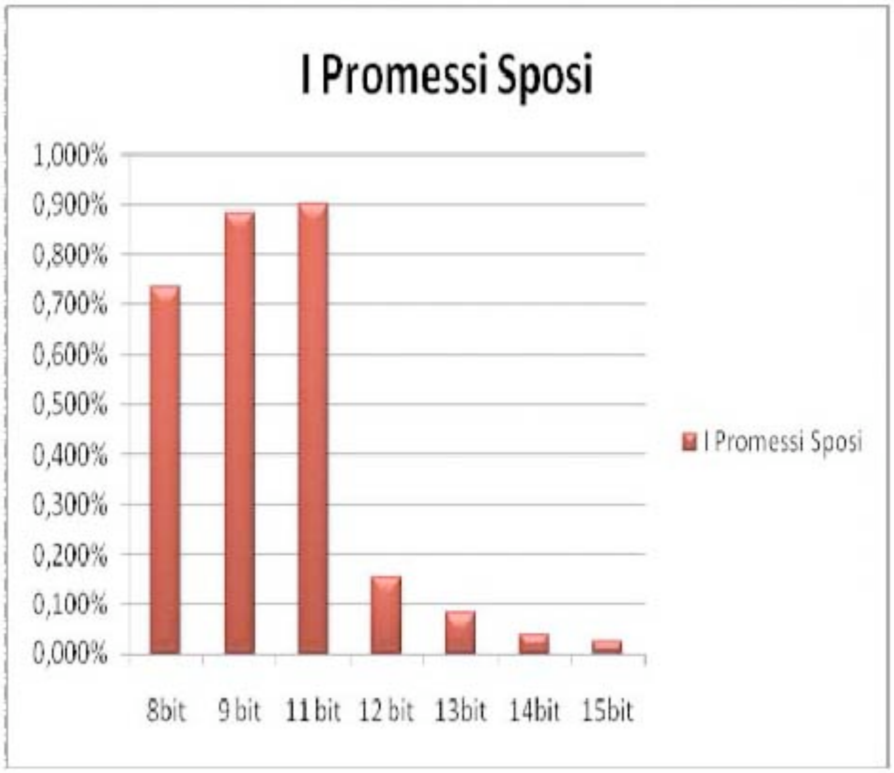
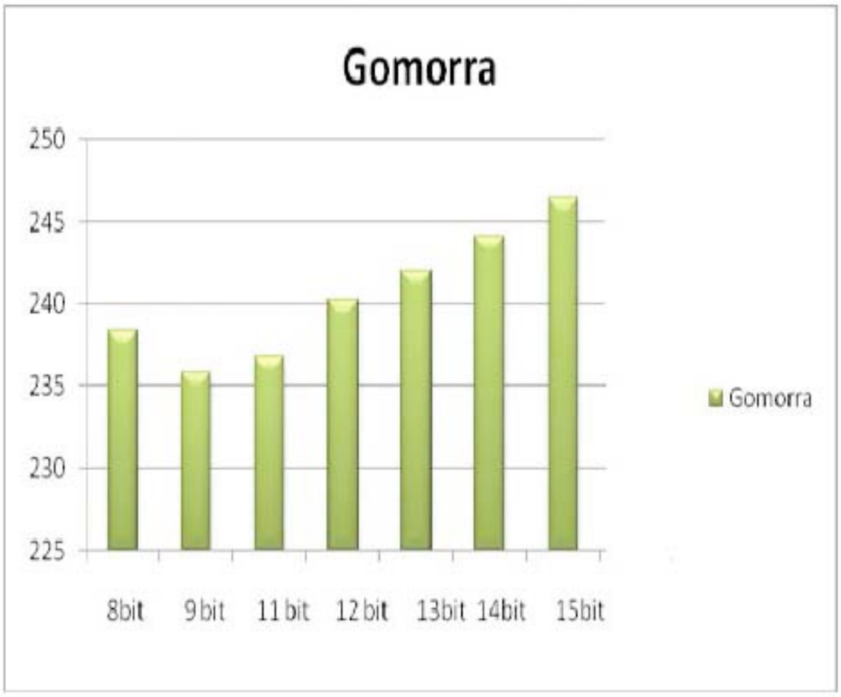


Figure 3 (a)-(b): Compression and hash length for "Promessi Sposi"



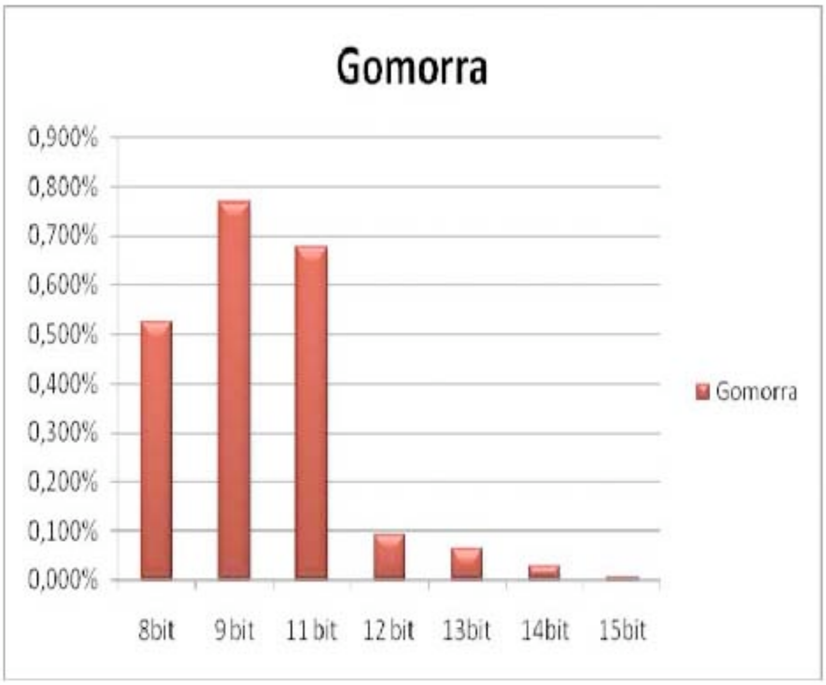


Figure 4 (a)-(b): Compression and hash length for "Gomorra"