# Course Project Guidelines

Submission: 23:59 June 18, 2017

Report/Codes submission: yayugao@hust.edu.cn

Please pay attention to the following notes.

- The course project report and the related source codes must be submitted to yayugao@hust.edu.cn before 23:59 June 18, 2017. No late work will be accepted.
- In order to submit your document effectively, you are required to compress your course report and related documents, and name the compressed file as **StudentID_Name_Course_Project.xxx**.
- Please also rename the title of your email to be "Information Theory": Course_Project_StudentID_name.
- Replace StudentID_Name with your own personal details.
- The course project is conducted on the individual basis. If any plagiarism is found, your project report will not be graded at all.
- Two selected topics are provided in below for you. You may choose any one as your course project.

## 1 Implemenation of Huffman Coding and Shannon Coding

This programming project is designed for the memento to Steve Jobs, D.A. Huffman and C. E. Shannon. The objective of this programming assignment is to deeply understand the Huffman coding and Shannon coding method. In your source code, please write down proper comment on each line which you think is important. You may discuss ideas with others in the class, but you need to implement your own program. Do not copy from other's problem source code. When you finish, create a ZIP file with your student ID as the file name. In the ZIP file, please include the source files and the corresponding executable files. Submit your ZIP file to the course email address before the due date.

In the above zipped file, please provide a **readme.txt** file which explains each files in your zipped file and how to run your program.

Detailed requirements:

1. Read Steve Jobs' 2005 Stanford Commencement Address "You've got to find what you love" as in the file "Steve_Jobs_Speech.txt". The file can also be downloaded at the following URL.

   `http://itec.hust.edu.cn/~yayugao/assets/pdf/Information_Theory/Steve_Jobs_Speech.doc`

2. Collect the statistics of the letters, punctuation, space in "Steve_Jobs_Speech.txt".
3. Compute the entropy of "Steve_Jobs_Speech.txt".
4. Apply the Huffman coding method and Shannon coding method for "Steve_Jobs_Speech.txt". Output the letters/punctuation/space and their Huffman codewords and Shannon codes, respectively.
5. Compute the average code length of "Steve_Jobs_Speech.doc" using your Huffman codes and Shannon codes, respectively.

Output the above results into a report.

Hint:

```
**Input**: $r$: the number of the source symbols; $P$: the probability distribution of source
        symbols.
**Output**: Output the Huffman codewords $w_i$ corresponding to the source symbols $s_i$.
initialization;
if $r == 2$ then
    return $s_0 \longmapsto 0$, $s_1 \longmapsto 1$
else
    sort $\{p_i\}$ in Descending order;
    reduce source: create a new symbol $s'$ to replace $s_{r-1}$, $s_{r-2}$ with the probability
    $p' = p_{r-1} + p_{r-2}$;
    Call the Huffman algorithm recursively to obtain the codes of $s_0, \ldots, s_{r-3}, s'$ as
    $w_0, \ldots, w_{r-3}, w'$ with the corresponding probability distribution $p_0, \ldots, p_{r-3}, p'$;
    return $s_0 \longmapsto w_0$, $s_1 \longmapsto w_1$, …, $s_{r-3} \longmapsto w_{r-3}$, $s_{r-2} \longmapsto w'_0$, $s_{r-1} \longmapsto w'_1$.
end
```

**Algorithm 1:** The Binary Huffman Algorithm

```
**Input**: $r$: the number of the source symbols;
$P = \{p(s_i)\}, i = 1, \ldots, r$: the probability distribution of source symbols.
**Output**: Output the Shannon codewords $w_i$ corresponding to the source symbols $s_i$.
initialization;
sort $\{p(s_i)\}$ in Descending order; for $i = 1 \to r$ do
    $F(s_i) \leftarrow \sum_{k=1}^{i-1} p(s_k)$;
    $l_i \leftarrow \left\lceil \log \frac{1}{p(s_i)} \right\rceil$;
    Code $F(s_i)$ using binary;
    Take $l_i$个digits after the dot as the codeword for the source symbols $s_i$.
end
```

**Algorithm 2:** The Shannon Coding Algorithm

## 2 Markov Models for English Text Analysis

This project was motivated by the programming assignment in [**?**]. Use a Markov chain to create a statistical model of a piece of English text.

### 2.1 Perspective

In the 1948 landmark paper “A Mathematical Theory of Communication”, Claude Shannon founded the field of information theory and revolutionized the telecommunications industry, laying the groundwork for today’s Information Age [**?**]. In this paper, Shannon proposed using a Markov chain to create a statistical model of the sequences of letters in a piece of English text. Markov chains are now widely used in speech recognition, handwriting recognition, information retrieval, data compression, and spam filtering. They also have many scientific computing applications including: the genemark algorithm for gene prediction, the Metropolis algorithm for measuring thermodynamical properties, and Google’s PageRank algorithm for Web search.

### 2.2 Markov model of natural language

Shannon approximated the statistical structure of a piece of text using a simple mathematical model known as a Markov model. A Markov model of order 0 predicts that each letter in the alphabet occurs with a fixed probability. We can fit a Markov model of order 0 to a specific piece of text by counting the number of occurrences of each letter in that text, and using these counts as probabilities. For example, if the input text is ”agggcagcgggcg”, then the Markov model of order 0 predicts that each letter is ’a’ with probability 2/13, ’c’ with probability 3/13, and ’g’ with probability 8/13. The following sequence of letters is a typical example generated from this model.

```
a g g c g a g g g a g c g g c a g g g g ...
```

An order 0 model assumes that each letter is chosen independently. This does not coincide with the statistical properties of English text since there is a high correlation among successive letters in an English word or sentence. For example, the letters ’h’ and ’r’ are much more likely to follow ’t’ than either ’c’ or ’x’. We obtain a more refined model by allowing the probability of choosing each successive letter to depend on the preceding letter or letters. An Markov model of order k predicts that each letter occurs with a fixed probability, but that probability can depend on the previous k consecutive letters (k-gram). For example, if the text has 100 occurrences of ”th”, with 60 occurrences of ”the”, 25 occurrences of ”thi”, 10 occurrences of ”tha”, and 5 occurrences of ”tho”, the Markov model of order 2 predicts that the next letter following the 2-gram ”th” is ’e’ with probability 3/5, ’i’ with probability 1/4, ’a’ with probability 1/10, and ’o’ with probability 1/20.

### 2.3 A brute force solution

Shannon proposed an interesting scheme to generate text according to a Markov model of order 1.

To construct [an order 1 model] for example, one opens a book at random and selects a letter at random on the page. This letter is recorded. The book is then opened to another page and one reads until this letter is encountered. The succeeding letter is then recorded. Turning to another page this second letter is searched for and the succeeding letter recorded, etc. It would be interesting if further approximations could be constructed, but the labor involved becomes enormous at the next stage.

Your task is write an efficient program to automate this laborious task. Shannon's brute force approach yields a statistically correct result, but, as Shannon observed, it is prohibitively slow when the size of the input text N is large. An alternate approach is to create a "Markov chain" and simulate a trajectory through it. This approach is described below. Markov chain. For the purpose of this assignment, a Markov chain is comprised of a set of states, one distinguished state called the start state, and a set of transitions from one state to another. The figure below illustrates a Markov chain with 5 states and 14 transitions.

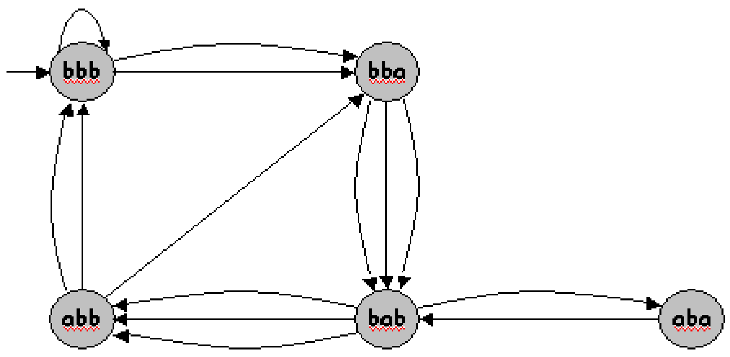

Figure 1: a Markov chain with 5 states and 14 transitions

### 2.4 Markov chain

Simulating a Markov chain. To simulate a trajectory through the Markov chain, begin at the start state. At each step select one of the leaving arcs uniformly at random, and move to the neighboring state. For example, if the Markov chain is in state bab, then it will transition to state abb with probability 3/4 and to state aba with probability 1/4. The following are the first ten steps of a possible trajectory beginning from bbb:

```
bbb -> bbb -> bba -> bab -> abb -> bbb -> bbb -> bba -> bab -> abb
```

### 2.5 Deliverables

1. Read Steve Jobs' 2005 Stanford Commencement Address "You've got to find what you love" as in the file "Steve_Jobs_Speech.doc". The file can also be downloaded from the course website.
2. Collect the statistics of the letters, punctuation, space in "Steve_Jobs_Speech.doc".
3. Compute the entropy of "Steve_Jobs_Speech.doc".
4. Construct the 0th, 3-th, 5-th order Markov models for "Steve_Jobs_Speech.doc". Output the model states and transition probabilities.
5. Submit your source codes for constructing the Markov chain models.