On Non-sequential Context Modeling with Application to Executable Data Compression

Wenrui Dai[1,2], Hongkai Xiong[1,2], Li Song[1,2]

[1]Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai, 200240, P.R. China

[2] Shanghai Key Laboratory of Digital Media Processing and Transmissions, Shanghai Jiao Tong University, Shanghai 200240, China

ABSTRACT

The sequential context modeling framework is generalized to a non-sequential one by context relaxation from consecutive suffix of the subsequences of symbols to the permutation of the preceding symbols as result of considering complex context structures in such sources as video and program binaries. Context weighting tree is also extended to a series of context trees which are built according to the "model tree", in which the descendent relationship in the formation of non-sequential context sets is described. Model redundancy and maximum a posteriori model in the framework are discussed and compared. A decision method based on the greedy algorithm is proposed to customize sets of models fitting the concrete sources. Brief description of application to executable data files incorporating with the semantics and syntax constraints are given and experiment are made accordingly as a validation.

1. Introduction

In classical sequential context modeling framework [1]-[3], one valid context set is required to satisfy two properties: i) disjoint property: no string in the context set is a suffix of any other string in this set; ii) exhaustive property: any subsequence of data symbols can find its suffix in the context set. Given a data sequence $x_1^n = x_1 x_2 \cdots x_n$, the subsequence associated with a context s in above valid context set S, is composed of the symbols whose preceding symbols are $s = x_{t-l_s}^{t-1}$, where l_s is the length of context s and $x_m^n = x_m x_{m+1} \cdots x_n$. Let A be the alphabet and let depth D denote the length of longest context available in the context set. It is well known that the two properties mentioned imply that the contexts in S can be represented as a tree structure, where each leaf node corresponds to one of its contexts respectively. In the tree, each node (excluding leaf nodes) has $|A|$ children and the height of the tree is D.

In most applications of context modeling, making inference for the context-induced subsequences results in redundancies, such as loss for not knowing the probability distribution or the underlying model. Two key problems are extensively concerned in

This work was supported in part under Intel-SJTU research program, and Grants NSFC No. 60632040, NSFC No. 60772099 from the National Science Foundation of China, and the National High Technology Research and Development Program of China (863 Program) (No. 2006AA01Z322).

Corresponding to Hongkai Xiong with Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai 200240, China. (xionghongkai@sjtu.edu.cn).

1068-0314/08 $25.00 © 2008 IEEE
DOI 10.1109/DCC.2008.6

sequential modeling and prediction: determining probability assignment to the context as a parameter and selecting models for least error in prediction. Prediction by Partial Match (PPM) and Context Tree Weighting (CTW) are two prevalent sequential modeling methods. PPM uses finite-context models of symbols to estimate the probability distribution for upcoming symbols, performing context switching from longer to shorter contexts [5][9]; whereas CTW imposes a sequential universal data compression procedure on binary tree sources, performing the "double mixture" over both parameters and models [4].

Sequential modeling has achieved success in such applications as text and speech data, but fails to perform well in source with complex context, e.g. image, video and executable files. The main problem is that contexts in such sources are not arranged in a sequential regularity. An attempt to the problem is multi-directional context modeling, in which the observations of contexts are multi-directional or multi-tracked under the classical sequential modeling framework [11]. Impressively, an effective code compression scheme is developed by analyzing syntax and semantics context in heterogeneous data to achieve superior performance [7] [8]. To achieve high prediction accuracy, a general compression software establishes various context models exploring either explicit or implicit correlations and mixes these models by neural network [12].

In this paper, the sequential context modeling framework is generalized to a non-sequential one in which context limitation, that all contexts are consecutive suffix of the subsequences of symbols, is relaxed to the permutation of the preceding symbols and "model tree" is introduced to describe the descendent relationship in the formation of context sets. In this non-sequential modeling framework, we propose a context weighting method for universal coding and discuss its model redundancy. Meanwhile, we apply the greedy algorithm for the model decision problem under *MDL* criterion [6]. For validation, the application of the framework to executable data compression is also described in consideration of the semantics and syntax constraints of the target source.

2. Non-Sequential Context Set

Given alphabet A, depth D and an individual data sequence x_1^n, we present the definition for non-sequential context set and model tree in this section. Contrary to sequential contexts, for a subsequence $x_{t-D}^{t-1} \in A^D$, the contexts in non-sequential modeling are not its consecutive suffix, but the permutation of symbols in x_{t-D}^{t-1}. For example, the non-sequential context with length three can be $x_{t-3}x_{t-2}x_{t-1}$, $x_{t-4}x_{t-2}x_{t-1}$, $x_{t-4}x_{t-3}x_{t-1}$ and $x_{t-4}x_{t-3}x_{t-2}$, while the context models are upper bounded by depth four. For simplicity, we discuss the binary source henceforth.

2.1 Definitions

For any symbol $x_t \in A, t = 1,2,\ldots,n$, its non-sequential context s with a length $l_s = m$ can be in the form of $x_{t-i_m} \cdots x_{t-i_2} x_{t-i_1}$, where $1 \le i_1 < \cdots < i_m \le D$. In sequential modeling i_m is the order of the context and we call it *bias* here. Specially, when $i_k = k$, s is as same as a context in sequential modeling. As a result, the context nodes in two context sets may be the same while their distribution on counts of zeroes and ones differs a lot. For a uniform

representation, a context set (or model) S is described as $\{(s, a_s, b_s)\}$, where s is the context node in S and a_s and b_s are its counts for zeroes and ones respectively. Consequently, the two properties of any context set S in sequential modeling shall be revised: i) for any context s in S, none of its children exists in S; ii) for any subsequence x_m^n, one permutation of its symbols can be found in S. For a non-sequential context s with length $l_s = m$ and bias i_m, the number of its children will be $D - i_m$ pairs. If its children exist, denote $s_j 0, s_j 1$ the j-th pair of its children with a length $m+1$ and bias $i_{m+1} = i_m + j$.

2.2 Model Tree

Since a context set is not uniform in sense of its context nodes, we construct a model tree including all composition of contexts to represent their successive relationship. Fig. 1 is a model tree for context sets with depth four. We can find that, in a model tree, each path from its root to its leaf node builds a context tree $T_i, i = 1 \ldots 2^{D-1}$ for context weighting. And consequently, we can deal with all the context trees (paths) with the context tree weighting methods.

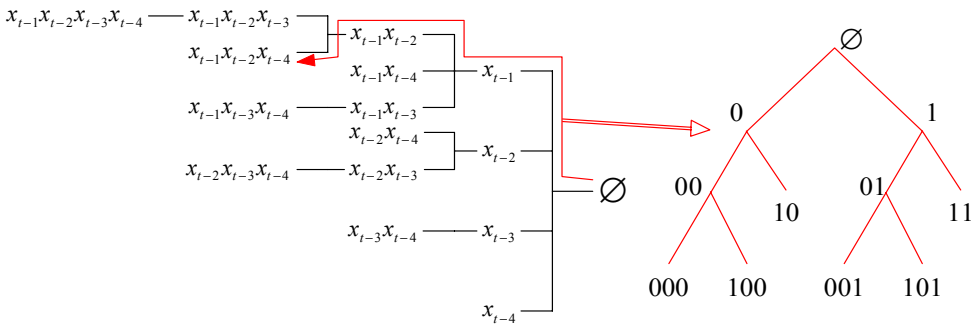

Fig. 1 Model tree for a model set with depth four

The height d of a context tree derived from the model tree is defined as the maximum length of its contexts. We can find that the context trees follow a binomial distribution with regard to its height. In detail, the number of context trees with a height d is $\binom{D-1}{d-1}$.

3. Non-Sequential Modeling and Weighting

The introduction of non-sequential models is meaningful, as sometimes the source to be estimated is not characterized by a sequential context. Fig. 2 shows the context trees of sequence 0100110100 for a third order sequential model (see [4]) and a corresponding model excluding the second symbol. In the Fig. 1, number a and b are the counts for zeroes and ones respectively, while P_e and P_w are the prediction under current context and the weighted prediction respectively. It is obvious that the latter costs less when encoded.

In this section, we propose a non-sequential weighting method for universal coding by balancing the estimated probability for all the context trees derived from the model tree. We also discuss the model redundancy and the maximum a posteriori model in the method and make a comparison with the results gained from the sequential modeling.

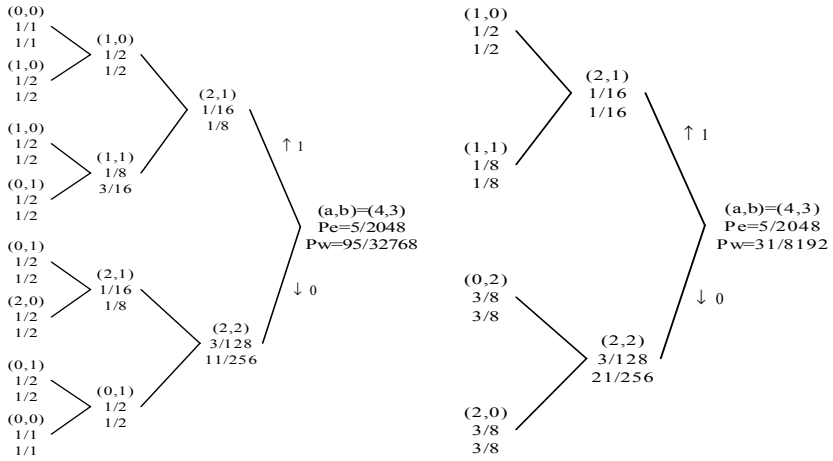

Fig. 2 Weighted context tree for $x_1^n = 0110100$ and $x_{1-D}^0 = \cdots 010$ with sequential model (left) and non-sequential model (right)

3.1 Non-sequential modeling for unknown source

For an unknown model and parameter source, we can make estimation by enumerating all the 2^D paths in the model tree with depth *D*. Let $P_e(a_s, b_s)$ and $P_{w,i}^s(d)$ denote the estimated and weighted estimated probability for context *s* in the *i*th tree with length *d* respectively. For a context tree T_i with height *d*, the estimated probability for the subsequence corresponding to the context *s* can be written as a weighted one according to the context tree weighting method:

$$P_{w,i}^s(d) = \begin{cases} P_e(a_s, b_s) & l_s = d \\ \frac{1}{2} P_e(a_s, b_s) + \frac{1}{2} P_{w,i}^{s_i 0} P_{w,i}^{s_i 1} & otherwise \end{cases} \tag{1}$$

where $s_j 0, s_j 1$ are the children of context *s* in this tree. Thus we can gain the weighted probability for the context tree T_i with a height *d*:

$$P_{w,i}^{\lambda}(d) = \sum_{S \in T_i} 2^{-L(S)} \prod_{s \in S} P_e(a_s, b_s) \tag{2}$$

with a weight $L(S) = 2|S| - 1 + |\{s \in S, l_s = d\}|$ for model *S* in T_i. We formulate the prediction in non-sequential modeling by weighting the models in all the context trees:

$$P_w^{\lambda} = \sum_i \frac{4^{d-1}}{5^{D-1}} \cdot P_{w,i}^{\lambda}(d) \tag{3}$$

In (3), a normalized weight $4^{d-1}/5^{D-1}$ is assigned to the context tree with a height *d*, such that contexts with same bias but different length are to play an equal role in weighting. In the weighting process we consider more context situations than the sequential modeling by diluting their weights with a normalized weight, such that non-sequential modeling can be applied to the contexts where complex correlation exists.

3.2 Model redundancy for non-sequential modeling

In this subsection, we discuss the model redundancy for non-sequential modeling. When considering sequential context sets only, the model redundancy [4] for any source with

model S_a is upper-bounded by:

$$\log\frac{\prod_{s\in S_a}P_e(a_s,b_s)}{P_w^{\lambda}(x_1^n)}\le\log\frac{1}{2^{-\Gamma_D(S_a)}}=\Gamma_D(S_a) \tag{4}$$

In non-sequential modeling, the model redundancy is analogically upper bounded by $L(S_a)-\log\left(4^{d-1}/5^{D-1}\right)$. When sequential modeling is applied, the model redundancy equals to the result in [4] if the underlying model is a sequential model. Otherwise, when the source is with a non-sequential model, the model redundancy is:

$$\log\frac{\prod_{s\in S_a}P_e(a_s,b_s)}{P_w^{\lambda}(x_1^n)}\le\min_S\log\frac{\prod_{s\in S_a}P_e(a_s,b_s)}{2^{-\Gamma_D(S)}\prod_{s\in S}P_e(a_s,b_s)}=\min_S\left(\Gamma_D(S)+\log\frac{\prod_{s\in S_a}P_e(a_s,b_s)}{\prod_{s\in S}P_e(a_s,b_s)}\right) \tag{5}$$

In most case, the sequential model close to S_a in structure is considered, e.g. with certain context s in non-sequential one split into s_1,s_2 in sequential model. If denote a,a_1,a_2 and b,b_1,b_2 the counts of zeroes and ones under the three contexts, the model redundancy is upper-bounded by:

$$\min_S\left(\Gamma_D(S)+\log\frac{\prod_{s\in S_a}P_e(a_s,b_s)}{\prod_{s\in S}P_e(a_s,b_s)}\right)=\min_S\left(\Gamma_D(S)+\log\frac{P_e(a,b)}{P_e(a_0,b_0)P_e(a_1,b_1)}\right)\le\min_S\left(\Gamma_D(S)+\log\frac{1}{\max\left(P_e(a_0,b_0),P_e(a_1,b_1)\right)}\right) \tag{6}$$

The equation infers that: because the more models are considered because of the relaxation of context limitations, the upper bound of model redundancy (the cost of not knowing the actual model) increases exponentially with regard to the depth *D* in non-sequential modeling. While applying sequential modeling here, if the underlying model is a sequential one, the upper bound is same to the result in [4]. Otherwise, the upper bound will increase with regard to the length of a semi-finite subsequence.

3.3 Maximum a posteriori model

To find the maximum a posteriori model, we apply the maximizing processing in the context tree weighting:

$$P_{m,i}^{s}=\begin{cases}P_e(a_s,b_s) & l_s=d\\ \frac{1}{2}P_e(a_s,b_s)+\frac{1}{2}\max_j\left(P_e(a_s,b_s),P_{w,i}^{s_j0}P_{w,i}^{s_j1}\right) & otherwise\end{cases} \tag{7}$$

Thus the maximum a posteriori probability is

$$P_m^{\lambda}=\max_i P_{m,i}^{\lambda}(d)=\max_i\max_{S\in T_i}2^{-L(S)}\prod_{s\in S}P_e(a_s,b_s) \tag{8}$$

Since the non-sequential modeling comprises all the sequential models, the maximum a posteriori probability in sequential modeling is not greater than the one in non-sequential modeling:

$$P_{m,seq}^{\lambda}\le P_m^{\lambda} \tag{9}$$

4. Non-sequential Model Decision

In this section we discuss the decision of non-sequential models for concrete sources.

We develop a sufficient condition on the distribution of counts, without gaining the actual code cost, for judging the adoption of a non-sequential context or a sequential one. When training sequence is available, we can apply the greed algorithm to customize the optimal combination of models under the minimum description length (*MDL*, see [6]) criterion.

4.1 Individual Decision for Non-sequential Context

We consider the estimated probability of sequence x_1^n under the non-sequential model $S_2=\{s=x_{t-1}\cdots x_{t-i+1}x_{t-i-1}\cdots x_{t-D},t=1\ldots n\}$, where the context consists of the previous D symbols excluding the ith one, and its corresponding sequential model $S_1=\{s=x_{t-D}^{t-1},t=1\ldots n\}$ respectively. We simply compare their estimated probabilities under certain node $s\in S_2$ and its corresponding node (s_0 for $x_{t-i}=0$ and s_1 for $x_{t-i}=1$) in S_1. Denote by a,a_0,a_1 and b,b_0,b_1 their counts for zeros and ones under contexts s,s_0,s_1. Define the ratio between correspondent counts as $u=a_0/a,v=b_0/b,t=(a_0+b_0)/(a+b)$. A sufficient condition for choosing a non-sequential rather than a sequential context can be gained.

Proposition: Under the KT-estimator (see [13]), the non-sequential context model S_2 is more economical than its corresponding sequential one S_1 in coding under the relevant contexts when the distributions of the counts for the contexts have a good tradeoff:

$$\sqrt{\frac{1}{t(1-t)(a+b)}}\cdot 2^{(a+b)(u-v)^2\min(u(1-u),v(1-v))}\le C \tag{10}$$

where C is a constant near the value $2^{-3/4}\sqrt{\pi}$, which depends on the counts distribution.

Proof: According to KT estimate, the ratio between the two estimated probabilities is:

$$\gamma=\frac{P_e^{s_0}(a_0,b_0)P_e^{s_1}(a_1,b_1)}{P_e^{s}(a,b)}=\frac{\left(a_0-\frac{1}{2}\right)\cdots\frac{1}{2}\cdot\left(b_0-\frac{1}{2}\right)\cdots\frac{1}{2}}{(a_0+b_0)!}\cdot\frac{\left(a_1-\frac{1}{2}\right)\cdots\frac{1}{2}\cdot\left(b_1-\frac{1}{2}\right)\cdots\frac{1}{2}}{(a_1+b_1)!}\Bigg/\frac{\left(a-\frac{1}{2}\right)\cdots\frac{1}{2}\cdot\left(b-\frac{1}{2}\right)\cdots\frac{1}{2}}{(a+b)!}$$

When the counts are great enough, we can compare the estimation of the two models according to Stirling's formula:

$$\gamma=\sqrt{\frac{2}{\pi}\cdot\frac{(a+b)}{(a_0+b_0)(a_1+b_1)}}\cdot\frac{a_0^{a_0}a_1^{a_1}b_0^{b_0}b_1^{b_1}}{a^ab^b}\cdot\frac{(a+b)^{a+b}}{(a_0+b_0)^{a_0+b_0}(a_1+b_1)^{a_1+b_1}}e^{\theta} \tag{11}$$

where θ is the parameter fading to zero when the counts increase., then

$$\begin{aligned}\log\left(\frac{a_0^{a_0}a_1^{a_1}b_0^{b_0}b_1^{b_1}}{a^ab^b}\cdot\frac{(a_0+b_0)^{a_0+b_0}(a_1+b_1)^{a_1+b_1}}{(a+b)^{a+b}}\right)&=\log\frac{u^{ua}v^{vb}(1-u)^{(1-u)a}(1-v)^{(1-v)b}}{t^{t(a+b)}(1-t)^{(1-t)(a+b)}}\\&=a\big(u\log u+(1-u)\log(1-u)\big)+b\big(v\log v+(1-v)\log(1-v)\big)-(a+b)\big(t\log t+(1-t)\log(1-t)\big)\end{aligned} \tag{12}$$

Let us consider the function $f(x)=x\log x+(1-x)\log(1-x),\ \ 0<x<1$. We can find that $f''(x)>0$ in the interval $(0,1)$. Thus according to the convexity we can find:

$$f(t)=f\left(\frac{a}{a+b}u+\frac{b}{a+b}v\right)\le\frac{a}{a+b}f(u)+\frac{b}{a+b}f(v) \tag{13}$$

On the other hand, we know that $-1\le f(x)<0$ and $f(x)$ is monotonously decreasing in the interval $(0,1/2)$ and monotonously increasing in the interval $(1/2,1)$. Without loss of generality, we assume that u is not less than v. Thus we can gain from Lagrange mean-value theorem:

$$af(u)+bf(v)-(a+b)f(t)=a\big(f(u)-f(t)\big)+b\big(f(v)-f(t)\big)\le\frac{1}{4}(a+b)(u-v)^2 f''(\zeta) \tag{14}$$

where $\xi\in(v,u)$. From (11) and (14) we can draw the proposition.

The proposition implies that the non-sequential model is more economical when i) the ratio difference $u-v$ is small and u,v is not close to zero or one; ii) the ratio *t* is close to $1/2$; iii) the sequence length $a+b$ is great enough that Stirling's formula can apply.

4.2 Customizing Non-sequential Models

If training sequence is available, we can find the models appropriate for the training sequence according to the *MDL* criterion. Once there exists an underlying model reflecting source distribution, we can take the maximum a posterior model under the training sequence as the objective; otherwise, the source might be described using a mixture of models which we can estimate in the *MDL* sense. The evaluation under *MDL* criterion for *m* models $S_1\ldots S_m$ is:

$$-\log p_{w,m}^{\lambda}+\sum C(S_m) \tag{15}$$

where $C(S_m)$ is the parameter complexity for model S_m, and $p_{w,m}^{\lambda}=\sum\alpha_i p_i^{\lambda}$ is the optimized weighted estimation made by the *m* models. We apply greedy algorithm here to find the optimized combination of models. For each step, the algorithm selects a model so that the MDL evaluation decreases the most. At step *m*, a candidate model S_{m+1} is chosen when:

$$S_{m+1}=\arg\min_{S}\Delta MDL=\min\left(-\log\frac{p_{w,m+1}^{\lambda}}{p_{w,m}^{\lambda}}+C(S_{m+1})\right)<0 \tag{16}$$

The process stops when $\Delta MDL>0$. In view of the property of MDL, the models selected are also optimal in a Bayesian sense.

5. Application to Executable Data Compression

5.1 Correlation and Modeling

The correlation in executable files is categorized as two kinds: vertical correlation and horizontal correlation and the split-stream algorithm and instruction rescheduling [7][8] are proposed for such correlations. The algorithms can be explained in the sense of non-sequential modeling. For split-stream algorithm, the sequential context *s* is split into two non-sequential ones s_1,s_2 ($s_1\cup s_2=s, s_1\cap s_2=\varnothing$), and whether it is split or not is due to the performance of the two kinds of models. Similarly, in instruction rescheduling, the non-sequential context s' is adjusted to sequential one *s* to align with the sequential prediction methods. Thus, besides comparing their actual code cost, the proposition in section 4.1 can be an alternative in judging whether adopting the algorithms.

5.2 Weighting Methods

Using a training sequence composed of assembly instructions in a size of 2MB, we apply the algorithm in section 4.2 to seek the optimal combination of context models and present them in the form of sequential models and non-sequential ones. It implies that

sequential models are corresponding to vertical models while non-sequential ones to horizontal ones. Table I shows the details.

Table I. Context Models in Executable Compression

(Symbol X is chosen to comprise contexts while Symbol O is omitted)

Sequential Model	XO; XXO; XXXO; XXXXO; XXXXXO
Non-sequential Model	SSXSXO; SXSSXO; SXSXSO; SXXSSO; XSSSXO; SSXXSO; XSSXSO; XSXSSO;SSSXSO; SXSSO;SXSSSO

To achieve the optimized estimated probability in section 4.2, we can apply the normalized LMS algorithm (see [10]) here. Normalized LMS algorithm has two distinct advantages: i) potentially-faster convergence speeds for both correlated and whitened input data; ii) stable behavior for a known range of parameter values independent of the input data correlation statistics. Choosing a destination function as $d=1$ for the optimal probability and error function as $e=1-p$, we can apply the algorithm as in Fig. 3. For the training sequence mentioned, we learn that the result will be optimal under the models selected when the initial weight ratio for sequential and non-sequential is set to 1:1 and the constant μ_n is set to 1.

1. Initialization

i) Initialization of weights. ii)Choose constant μ_n and γ

Do when not ended

2. Prediction

$$p = \sum w_i p_i$$

3. Update

i) $e(k) = d(k) - \sum w_i p_i$

ii) for i<max model number

$$w_i(k+1) = w_i(k) + \frac{\mu_n}{\gamma + \mathbf{p}^T(k)\mathbf{p}(k)} e(k) p_i(k)$$

Fig 3. Pseudo-code of Context Weighting Based on Normalized LMS

6 Experiment Results

Experiments are made to evaluate the performance of our scheme in executable compression. Some common executable files are taken as examples. Our result is designed to compare with compressors including WinRAR, PPMd, PPMonstr, and PAQ. PPMd and PPMonstr are the improved version of PPM, where the former emphasizes on

speed and the latter on modeling for non-stationary data sources (executable files are just one kind).

Compression ratio and speed are compared under the same limitation of memory. Table II elaborates the concrete experiment results. As we have analyzed above, complex context structure in executables hampers the two in performance. Our scheme is obviously better in performance — exceeding PPMd by 10% and PPMonstr by 4% in general. As for complexity in context weighting, our scheme is about 3 times the time cost of PPMonstr. However considering that the speed is about 100KB/s~110KB/s, it is enough for many applications such as network transmission with 1Mbps bandwidth.

Table II Compression Performance for Executable Files

	Our scheme	PAQ8	PPMd	PPMonstr	WinRAR
WINWORD 2002	62.58%	62.71%	53.93%	59.39%	55.74%
EXCEL 2002	61.52%	61.37%	52.63%	57.51%	52.89%
POWERPNT 2002	68.36%	68.49%	57.88%	62.81%	60.51%
MSACCESS	66.40%	66.25%	57.92%	62.69%	59.04%
MSNMSGR 8.0	74.45%	75.01%	65.23%	70.19%	69.25%
ACRORD32 6.0	68.12%	69.82%	59.27%	64.37%	63.58%
PHOTOSHOP 8.0	73.08%	74.51%	65.34%	70.16%	67.93%

PAQ8 combines probabilities with neural network, and thus, reaches a high compression ratio; however, it also leads to mass computation and memory occupation. While our scheme is slightly more efficient than PAQ8 in about half examples, its time cost is about one-third the latter. It can be found that our method performs well at most time, but has a discrepancy of approximately 2% in performance on ACRORD32.EXE compared with PAQ8, the reason for which is that Section Text, consisting of assembly instructions, only takes an approximate proportion of 60% in ACRORD32.EXE, but around 90% in others. Owing to the fact that we put an emphasis on executable compression (especially the dependency among assembly instructions), the performance demonstrates the efficiency of our methods.

Table III. Proportion of Section Text in Executable Files

	WINWORD	EXCEL	POWERPNT	MSACCESS	MSNMSGR	PHOTOSHOP	ACRORD32
RATIO	92.62%	89.31%	91.71%	86.26%	86.69%	78.00%	58.20%

7 Conclusion

Since the contexts in some sources do not obey a consecutive characterization, we generalize the sequential modeling framework to a non-sequential one by relaxing the

context from consecutive suffix of the subsequences of symbols to the permutation of the preceding symbols. We extend the weighting method for universal coding by introducing so called "model tree", which describes the descendent relationship in the formation of non-sequential context sets and discuss its model redundancy in the framework. For concrete sources, we propose a decision method based on the greedy algorithm to seek sets of suitable context models. The framework is also applied to executable data files incorporating with the semantics and syntax constraints for validating and the experiment result show its efficiency.

Reference

[1] J. Rissanen, "A universal data compression system," *IEEE Trans. Inform. Theory*, vol. 29, pp. 656-664, Sept. 1983.

[2] M.J. Weinberger, J. Rissanen, and M. Freder, "A universal finite memory source," *IEEE Trans. Inform. Theory*, vol. 41, pp.643-652, May 1995.

[3] J. Rissanen, "Universal Coding, Information, Prediction, and Estimation," *IEEE Trans. Inform. Theory*, vol. IT-30, No. 4, July 1984.

[4] F.M.J. Willems, Y.M. Shtarkov, and T.J. Tjalkens, "The context-tree weighting method: basic properties," *IEEE Trans. Inform. Theory*, vol. 41, no. 3, pp. 653–664, May 1995.

[5] J.G. Cleary and I.H. Witten, "Data Compression Using Adaptive Coding and Partial String Matching," *IEEE Trans. Communications*, COM-32(4):396-402, April 1984

[6] A. Barron, J. Rissanen, and B. Yu, "The Minimum Description Length Principle in Coding and Modeling," *IEEE Trans. Inform. Theory*, vol. 44, No. 6, October 1998.

[7] M. Drinic, and D. Kirovski, "PPMexe: PPM for compressing software," *Proc. Data Compression Conference 2002*, pp. 192–201, 2-4 April 2002.

[8] M. Drinic, D. Kirovski, and H. Vo, "PPMexe: Program compression," *ACM Trans. Programming Language and Systems*, vol. 29, no. 3, January 2007.

[9] A. Moffat "Implementing the PPM Data Compression Scheme," *IEEE Trans. Communications*, COM-38: 1917-1921, November 1990

[10] P.S.R. Diniz, *Adaptive Filtering: Algorithms and Practical Implementing*, Second Edition, Kluwer Academic Publishers.

[11] E. Ordentlich, M.J. Weinberger, and T. Weissman, "Multi-directional context sets with applications to universal denoising and compression," *IEEE Proc. International Symp. Inform. Theory*, pp. 1270-1274, Sept. 2005.

[12] Matthew V. Mahoney, "Adaptive weighing of context models for lossless data compression," http://www.cs.fit.edu/~mmahoney/compression/.

[13] R.E. Krichevsky and V.K. Trofimov, "The performance of universal coding," *IEEE Trans. Inform. Theory*, vol. 27, pp. 199-207, March 1981.