High-Order Entropy-Based Population Diversity Measures in the Traveling Salesman Problem

Yuichi Nagata nagata@is.tokushima-u.ac.jp
Graduate School of Technology, Industrial and Social Sciences, Tokushima University, 2-1 Minami Josanjima, Tokusima, 770-8506, Japan

https://doi.org/10.1162/evco_a_00268

Abstract

To maintain the population diversity of genetic algorithms (GAs), we are required to employ an appropriate population diversity measure. However, commonly used population diversity measures designed for permutation problems do not consider the dependencies between the variables of the individuals in the population. We propose three types of population diversity measures that address high-order dependencies between the variables to investigate the effectiveness of considering high-order dependencies. The first is formulated as the entropy of the probability distribution of individuals estimated from the population based on an m-th–order Markov model. The second is an extension of the first. The third is similar to the first, but it is based on a variable order Markov model. The proposed population diversity measures are incorporated into the evaluation function of a GA for the traveling salesman problem to maintain population diversity. Experimental results demonstrate the effectiveness of the three types of high-order entropy-based population diversity measures against the commonly used population diversity measures.

Keywords

Genetic algorithm, population diversity, entropy, high-order dependencies, traveling salesman problem.

1 Introduction

The maintenance of population diversity is recognized as an important factor for fully exercising the capability of evolutionary algorithms (EAs), and a wide variety of population management strategies for promoting diversity inside the population have been proposed. A survey of methodologies for promoting population diversity in EAs can be found in Squillero and Tonda (2016).

Several population diversity management methodologies utilize measures of population diversity, which can be used to analyze the behavior of EAs (Yao, 1993; Tsai et al., 2004; Wang et al., 2010), to select individuals to maintain population diversity in a positive manner (Maekawa et al., 1996; Zhang et al., 2006; Nagata, 2006; Nagata and Kobayashi, 2013), and as a trigger to activate diversification procedures (Tsujimura and Gen, 1998; Vallada and Ruiz, 2010). The pairwise Hamming distance (the average of the Hamming distance between all possible pairs of the population members) is the most commonly used measure of population diversity. Another commonly used population diversity measure is based on entropy. In information theory, entropy, defined as $-\sum_{s\in S} p_s \log p_s$, is a measure of the uncertainty of a probability distribution p_s $(s \in S)$,

Manuscript received: 4 November 2018; revised: 10 September 2019, 18 December 2019, 30 January 2020; accepted: 30 January 2020.

where S is a set of all possible events. This definition, however, cannot be directly used to measure population diversity because the population size is typically considerably smaller than the number of all possible solution candidates in the search space S. Therefore, to the best of our knowledge, the entropy-based population diversity measures proposed in previous works are all defined as the sum of the entropies of the univariate marginal distributions of all variables. For example, let the solution space S be defined as $(x_1, \ldots, x_n)$, where x_i is a variable taking values in a discrete set A_i. The entropy of the i-th variable is defined as $H_i = -\sum_{j \in A_i} p_{ij} \log p_{ij}$, where p_{ij} is the probability that x_i has a value j in the population. Then, the commonly used entropy-based population diversity measure is defined as $H = \sum_{i=1}^{n} H_i$. In this article, we refer to an entropy-based population diversity measure defined in this manner as an *independent entropy measure*. In previous works, the independent entropy measure was incorporated into EAs applied to the knapsack problem (Mori et al., 1996), binary quadratic programming problem (Wang et al., 2010), traveling salesman problem (Yao, 1993; Maekawa et al., 1996; Tsujimura and Gen, 1998; Tsai et al., 2004; Nagata, 2006; Nagata and Kobayashi, 2013), and others (Zhang et al., 2006).

The independent entropy measure (and other commonly used population diversity measures), however, is not able to consider the dependencies between the variables of the individuals in the population, which creates a situation where population diversity cannot be evaluated appropriately. For example, consider an extreme example on the n-dimensional binary solution space where half of the population members are "$00 \cdots 00$" and the other half are "$11 \cdots 11$." The value of the independent entropy measure of this population is virtually the same as that of a randomly generated population because $p_{i0} \simeq p_{i1} \simeq 0.5$ $(i = 1, \ldots, n)$ for both populations, even though "true" population diversity is extremely low in the former. Therefore, our motivation herein is to design a more appropriate entropy-based population diversity measure by considering dependencies between the variables of the individuals in the population. We refer to such a population diversity measure as a *high-order entropy measure*.

In this article, we propose several high-order entropy measures for the traveling salesman problem (TSP) to investigate the advantages of using entropy-based population diversity measures that consider the dependencies between the variables. We first formulate high-order entropy measures based on a fixed-order Markov model, where we assume that the probability of observing each vertex at a certain position in an individual (tour) of the population depends on the sequence of m (≥ 1) precedent vertices. We further extend this model into a variable-order Markov model, where the value of m varies depending on the situation.

We tested the proposed high-order entropy measures on a genetic algorithm (GA) developed by Nagata and Kobayashi (2013), which is known as one of the most effective heuristic algorithms for the TSP. In this GA, one important feature for achieving a top performance is to maintain population diversity by evaluating offspring solutions based on an evaluation function that incorporates a population diversity measure as well as the original evaluation function (tour length). An independent entropy measure is used for evaluating the population diversity. In this article, we perform this GA by replacing the original independent entropy measure with each of the proposed high-order entropy measures in the evaluation function.

Preliminary reports for the high-order entropy measures proposed in this article, were presented in previous works of the author (Nagata and Ono, 2013; Nagata, 2016). This article provides a full description and instructive analysis of the proposed high-order entropy measures; Section 6.3 and the Appendix are completely new and other

parts significantly extend the contents of the conference proceedings. The remainder of this article is organized as follows. In Section 2, we describe the background of this study. In Section 3, we propose two types of high-order entropy measures based on a fixed-order Markov model. In Section 4, we propose a high-order entropy measure based on a variable-order Markov model. In Section 5, the GA framework, where the proposed population diversity measures are incorporated, is described. Computational results are presented in Section 6. Finally, conclusions are provided in Section 7.

2 Background

We first consider commonly used population diversity measures (independent entropy measure and pairwise Hamming distance) in a general case and then describe the independent entropy measure for the TSP. We also refer to the difficulty of designing an entropy-based population diversity measure that considers the dependencies between the variables.

2.1 Commonly Used Population Diversity Measures

We first consider a general case where an individual is represented as a string of length n, consisting of symbols in a set L. Let the population consist of N_p individuals, denoted as $\boldsymbol{p_1}, \boldsymbol{p_2}, \ldots, \boldsymbol{p_{N_p}}$. Let X_i $(i = 1, \ldots, n)$ be a random variable representing the symbol in the i-th position of an individual (string) selected randomly from the population. The joint probability distribution $P(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n)$ then represents the probability of finding individual $(x_1, x_2, \ldots, x_n)$ when an individual is selected randomly from the population. We denote $P(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n)$ as $P(x_1, x_2, \ldots, x_n)$ for simplicity. The marginal probability distribution $P(X_i = x_i)$ or $P(x_i)$ then represents the probability of finding symbol x_i in the i-th position of an individual selected randomly from the population. Probability distribution $P(X_i = l)$ is estimated as $\frac{n_{il}}{N_p}$, where n_{il} is the number of individuals that have the symbol l at position i. The independent entropy measure, which we denote as H^{ind}, is then defined by

$$H^{ind} = -\sum_{i=1}^{n} \sum_{x_i \in L} P(x_i) \log P(x_i). \tag{1}$$

Another commonly used population diversity measure is the pairwise Hamming distance (the average of the Hamming distance between all possible pairs of the population members). Let a symbol in the i-th position of individual $\boldsymbol{p}$ be denoted as $(\boldsymbol{p})_i$. The Hamming distance between two individuals (strings) $\boldsymbol{p}$ and $\boldsymbol{q}$, denoted as $d(\boldsymbol{x}, \boldsymbol{y})$, is then defined by

$$d(\boldsymbol{p}, \boldsymbol{q}) = n - \sum_{i=1}^{n} I((\boldsymbol{p})_i, (\boldsymbol{q})_i), \tag{2}$$

where the function $I(a, b)$ returns 1 if a and b are the same, and 0 otherwise.

In fact, the pairwise Hamming distance can be expressed as a function of $P(x_i)$ (Wineberg and Oppacher, 2003). To derive this expression, we rewrite the Hamming distance as $d(\boldsymbol{p}, \boldsymbol{q}) = n - \sum_{i=1}^{n} \sum_{l \in L} I_l((\boldsymbol{p})_i, (\boldsymbol{q})_i)$, where $I_l(a, b)$ returns 1 if both a and b are symbol l, and 0 otherwise. The pairwise Hamming distance, which we denote as

D, is then calculated as follows:

$$
\begin{aligned}
D &= \frac{2}{N_p(N_p-1)} \sum_{j=1}^{N_p-1} \sum_{k=j+1}^{N_p} d(\boldsymbol{p_j}, \boldsymbol{p_k}) = \frac{1}{N_p(N_p-1)} \sum_{j=1}^{N_p} \sum_{k=1}^{N_p} d(\boldsymbol{p_j}, \boldsymbol{p_k}) \\
&\propto \sum_{j=1}^{N_p} \sum_{k=1}^{N_p} \{n - \sum_{i=1}^{n} \sum_{l \in L} I_l((\boldsymbol{p_j})_i, (\boldsymbol{p_k})_i)\} = {N_p}^2 n - \sum_{i=1}^{n} \sum_{l \in L} \sum_{j=1}^{N_p} \sum_{k=1}^{N_p} I_l((\boldsymbol{p_j})_i, (\boldsymbol{p_k})_i) \\
&= {N_p}^2 n - \sum_{i=1}^{n} \sum_{l \in L} n_{il}^2 = {N_p}^2 n - \sum_{i=1}^{n} \sum_{l \in L} \{N_p P(x_i = l)\}^2 \\
&\propto n - \sum_{i=1}^{n} \sum_{l \in L} P(x_i = l)^2 = n - \sum_{i=1}^{n} \sum_{x_i \in L} P(x_i)^2. \qquad (3)
\end{aligned}
$$

Note that $\sum_{j=1}^{N_p} \sum_{k=1}^{N_p} I_l((\boldsymbol{p_j})_i, (\boldsymbol{p_k})_i) = n_{il}^2$ because the number of individuals that have symbol l in the i-th position is n_{il}.

The similarity between the independent entropy measure H^{ind} and the pairwise Hamming distance D was discussed in Wineberg and Oppacher (2003). As suggested in Nagata and Kobayashi (2013), one advantage of the independent entropy measure over the pairwise Hamming distance is the sensitivity to the change of rare elements in the population. This feature makes H^{ind} a more appropriate population diversity measure than D.

2.2 TSP Case

Let an asymmetric TSP (ATSP) be defined on a complete directed graph (V, E) with a set of vertices $V = \{1, \ldots, n\}$ and a set of edges $E = \{(i, j) \mid i, j \in V\}$. In the asymmetric case, the distance (or cost) between two vertices depends on the travel direction. If the distance is the same in both directions, the TSP is called a symmetric TSP (STSP).

In the majority of GAs applied to the TSP, an individual is represented as the order of vertices on the tour. It is also possible to represent an individual such that the variable x_i represents the vertex subsequent to vertex i in the tour, which is well suited for the definition of population diversities D and H^{ind}. Using this expression, $P(X_i = l)$ $(l \in 1, 2, \ldots n)$ is defined as the probability distribution of the vertices subsequent to vertex i in the population, and the population diversity measures H_{ind} and D are defined according to Eqs. (1) and (3), respectively. Here, we need to give attention to the STSP case. In this case, we must consider both travel directions for each tour because the population diversity should not depend on the travel direction. Therefore, $P(X_i = l)$ $(l \in 1, 2, \ldots n)$ is defined as the probability distribution of the vertices linked to vertex i in the population. The population diversity measure H^{ind} defined in this manner is used in GAs (Maekawa et al., 1996; Tsai et al., 2004; Nagata, 2006; Nagata and Kobayashi, 2013) for the STSP to control the diversity of the population.

2.3 Difficulty in Considering Dependencies

The population diversity measures H^{ind} and D do not consider the dependencies between the variables of the individuals in the population because these are expressed

as functions of $P(x_i)$. The most naive entropy-based population diversity measure that considers the dependencies between the variables would be defined by

$$H = -\sum_{x_1 \in L} \cdots \sum_{x_n \in L} P(x_1, \ldots, x_n) \log P(x_1, \ldots, x_n). \tag{4}$$

Clearly, this population diversity measure is of no value in practical use (unless n is extremely small) because it is impossible to obtain a sufficient number of samples from the population necessary to estimate $P(x_1, \ldots, x_n)$. Therefore, the joint probability distribution $P(x_1, \ldots, x_n)$ must be estimated under a certain assumption(s). For example, when $P(x_1, \ldots, x_n)$ is modeled as $\prod_{i=1}^{n} P(x_i)$, the entropy H is equivalent to H^{ind} except for a constant factor, i.e., $H = nH^{ind}$.

A Bayesian network could be also useful to model the joint probability distribution. It is represented as $P(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} P(x_i \mid \boldsymbol{x}_{parent(i)})$, where each variable x_i is conditional only on its parent variables $\boldsymbol{x}_{parent(i)}$ (if it is empty, $P(x_i \mid \boldsymbol{x}_{parent(i)})$ means a prior probability distribution $P(x_i)$). However, it is typically difficult to detect an appropriate conditional dependency between the variables, which is represented as a directed acyclic graph (DAG), in advance. Moreover, it is difficult to compute Eq. (4) efficiently even if a DAG is given in advance.

3 High-Order Entropy Measures Based on a Fixed-Order Markov Model

We propose two high-order entropy measures based on a fixed-order Markov model to measure population diversity of GAs for the TSP.

3.1 High-Order Entropy Measure H_m

Let an individual (tour) be represented as a permutation of the vertices. We model the joint probability distribution of the population on the assumption that the probability of observing each vertex at a certain position in an individual of the population depends on the sequence of m (≥ 1) precedent vertices in the tour. This assumption is reasonable for the TSP. For example, if you are solving a TSP manually, the next vertex at each vertex strongly depends on the partial path created so far near this vertex and the sequence of m precedent vertices contains a lot of information about this. Let S_i ($i = 1, \ldots, n$) be a random variable representing the i-th vertex in an individual selected randomly from the population. Given that a tour has a cyclic structure, $P(S_1 = s_1, S_2 = s_2, \ldots, S_n = s_n)$, which we denote as $P(s_1, s_2, \ldots, s_n)$ for short, is represented by

$$P(s_1, s_2, \ldots, s_n) = \prod_{i=1}^{n} P(s_{i+m} \mid s_i, \ldots, s_{i+m-1}), \tag{5}$$

where index $i + n$ ($1 < i < m$) corresponds to i. The entropy H of this joint probability distribution is then calculated as follows:

$$\begin{aligned} H &= -\sum_{s_1} \cdots \sum_{s_n} P(s_1, \ldots, s_n) \log P(s_1, \ldots, s_n) \\ &= -\sum_{s_1} \cdots \sum_{s_n} P(s_1, \ldots, s_n) \sum_{i=1}^{n} \log P(s_{i+m} \mid s_i, \ldots, s_{i+m-1}) \end{aligned}$$

$$= -\sum_{i=1}^{n} \left\{ \sum_{s_1} \cdots \sum_{s_n} P(s_1, \ldots, s_n) \log P(s_{i+m} \mid s_i, \ldots, s_{i+m-1}) \right\}$$

$$= -\sum_{i=1}^{n} \left\{ \sum_{s_i} \cdots \sum_{s_{i+m}} P(s_i, \ldots, s_{i+m}) \log P(s_{i+m} \mid s_i, \ldots, s_{i+m-1}) \right\}. \quad (6)$$

Given that each tour can start from an arbitrary vertex, the joint probability distribution of any $m+1$ consecutive random variables should be invariant with respect to shifts in the index, that is, $P(s_i, \ldots, s_{i+m})$ and $P(s_{i+m} \mid s_i, \ldots, s_{i+m-1})$ should be equivalent to $P(s_1, \ldots, s_{m+1})$ and $P(s_{m+1} \mid s_1, \ldots, s_m)$, respectively, regardless of the value of i. This assumption is justified by considering n tours with all different start vertices for each individual (tour) in the population. Then, Eq. (6) can be simplified as follows:

$$\begin{aligned} H &= -n \sum_{s_1} \cdots \sum_{s_{m+1}} P(s_1, \ldots, s_{m+1}) \log P(s_{m+1} \mid s_1, \ldots, s_m) \\ &= -n \sum_{s_1} \cdots \sum_{s_{m+1}} P(s_1, \ldots, s_{m+1}) \log \frac{P(s_1, \ldots, s_{m+1})}{P(s_1, \ldots, s_m)} \\ &= -n \Big\{ \sum_{s_1} \cdots \sum_{s_{m+1}} P(s_1, \ldots, s_{m+1}) \log P(s_1, \ldots, s_{m+1}) \\ &\quad - \sum_{s_1} \cdots \sum_{s_m} P(s_1, \ldots, s_m) \log P(s_1, \ldots, s_m) \Big\} \\ &= n(\overline{H_{m+1}} - \overline{H_m}), \end{aligned} \quad (7)$$

where

$$\overline{H_k} = -\sum_{s_1} \cdots \sum_{s_k} P(s_1, \ldots, s_k) \log P(s_1, \ldots, s_k). \quad (8)$$

The average entropy per symbol of the probability distribution Eq. (5), which we denote as H_m, is then given by

$$H_m = \overline{H_{m+1}} - \overline{H_m}. \quad (9)$$

We use H_m as a high-order entropy measure for the TSP. Note that $H_1 (= \overline{H_2} - \overline{H_1})$ is essentially equivalent to H^{ind} because $P(S_1 = i, S_2 = j) \propto P(X_i = j)$ and $\overline{H_1}$ is a constant value for the TSP (because all symbols appear exactly once in each individual of the population). We have the following relation:

$$H_1 = \frac{1}{n} H^{ind} + const. \quad (10)$$

In information theory, H_m is known as the entropy rate of an m-th–order Markov information source modeled by the conditional probability distributions $P(s_{m+1} \mid s_1, \ldots, s_m)$, where the entropy rate of a data source is defined as the average information per symbol obtained from the data source. A central theorem of information theory states that the entropy rate of a data source indicates the average number of bits per symbol required to encode it. Therefore, the existence of the same sequence consisting of up to $m+1$ vertices in the population decreases the value of H_m; this effect is more prominent when the length of the high-frequency sequences increases.

To compute H_m, we must estimate $P(s_1, \ldots, s_k)$ for $k = m, m+1$ by sampling sequences of symbols (vertices) from individuals in the population; $P(s_1, \ldots, s_k)$ is estimated by $\frac{N(s_1, \ldots, s_k)}{N_{sample}}$, where $N(s_1, \ldots, s_k)$ is the number of a sequence of symbols

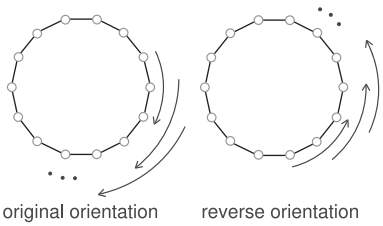

Figure 1: Illustration of method to sample sequences of length k (= 3) from a tour; sequences are sampled for the original (both) orientation(s) in the ATSP (STSP) case.

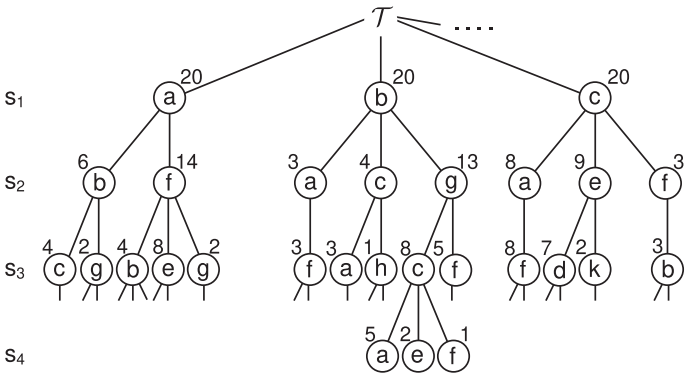

Figure 2: Tree representation of $N(s_1, \ldots, s_k)$ ($k \leq 4$).

$\{s_1, \ldots, s_k\}$ in the population and N_{sample} is the total number of all possible samples of the sequences in the population. Sequences of length k are sampled from each individual while shifting the start of the sequence one by one, resulting in N_p samples from each individual, i.e., $N_{sample} = nN_p$. Note that in the STSP case, sequences of length k are sampled from each individual for both travel directions, resulting in $2n$ samples from each individual, i.e., $N_{sample} = 2nN_p$. Figure 1 illustrates the method to sample the sequences from an individual (tour) in the ATSP and STSP cases.

We store the values of $N(s_1, \ldots, s_k)$ ($k \leq m + 1$) in the form of a tree because it is impractical to store all entries in a table for a large value of m. Let $\mathcal{T}$ be a tree for storing $N(s_1, \ldots, s_k)$ ($k \leq m + 1$). Figure 2 illustrates an example of the tree $\mathcal{T}$ for $k \leq 4$ in the ATSP case (because it is easier to understand). The tree $\mathcal{T}$ stores only $N(s_1, \ldots, s_k)$ with nonzero values; for example, $N(a, b) = 6$, $N(a, b, c) = 4$, and $N(a, c) = 0$ in this example. Note that in the TSP, $N(s_1)$ must be the same for all s_1 and the same symbol does not reappear in a sequence. For a population consisting of relatively high-quality solutions for the TSP (e.g., tours obtained by a local search algorithm with the 2-opt neighborhood), we have observed that the number of branches at each node is at most about ten (usually less than five).

As the value of m is increased, H_m captures higher-order dependencies in the sequences of symbols included in the population. In this sense, we want to increase the value of m. However, H_m is of no value if the value of m is overly large because it is unlikely to obtain a sufficient number of samples (sequences of symbols) from the population necessary to estimate the conditional probability distributions $P(s_{i+m} \mid$

$s_i, \ldots, s_{i+m-1}) = \frac{P(s_1,\ldots,s_{m+1})}{P(s_1,\ldots,s_m)} = \frac{N(s_1,\ldots,s_{m+1})}{N(s_1,\ldots,s_m)}$ for computing H_m; the estimated conditional probability distribution is unreliable if the number of samples of the denominator is small. Therefore, there is a tradeoff between the potential ability to capture higher-order dependencies and the estimate accuracy of the conditional probability distributions, and we need to determine an appropriate value of m.

3.2 High-Order Entropy Measure H_m^{adj}

One might think that $\overline{H_{m+1}}$ can also be used as a population diversity measure. This is equivalent to the entropy of the probability distribution $P(s_1, \ldots, s_{m+1})$ defined under the assumption that blocks of $m+1$ consecutive symbols $s_1, \ldots, s_{m+1}$ appear in the population according to this probability distribution and that these occurrences are independent of each other. However, these occurrences are actually correlated and the definition of $\overline{H_{m+1}}$ neglects such dependencies. In information theory, $\overline{H_{m+1}}$ is known as the entropy of the *adjoint source* of the $(m+1)$-th *extension* of the original Markov source. From this point onward, we call $\overline{H_{m+1}}$ the high-order entropy measure H_m^{adj}, to distinguish it from H_m.

Although the definition of H_m^{adj} $(= \overline{H_{m+1}})$ is somewhat ad hoc as a population diversity measure, we can find the following simple relationship between H_m^{adj} and H_k $(k = 1, \ldots, m)$:

$$H_1 + H_2 + \cdots + H_m = (\overline{H_2} - \overline{H_1}) + (\overline{H_3} - \overline{H_2}) + \cdots + (\overline{H_{m+1}} - \overline{H_m}) = H_m^{adj} - \overline{H_1}. \quad (11)$$

Because $\overline{H_1}$ is a constant value (for the TSP), H_m^{adj} is equivalent to $H_1 + H_2 + \cdots + H_m$. As can be predicted, the best value m for H_m^{adj} will be greater than that of H_m because H_m^{adj} mixes the high-order entropy measures H_k $(k = 1, \ldots, m)$ equally. We discuss the advantages of this population diversity in Section 6.3.

4 High-Order Entropy Measure Based on a Variable-Order Markov Model

We propose a high-order entropy measure based on a variable-order Markov model to measure population diversity of GAs for the TSP.

4.1 Motivation

The high-order entropy measure H_m defined in the previous section is derived from the assumption that the probability of occurrence of a symbol appearing in individuals in the population obeys a Markov process of order m. In the following equations, we use random variables S_i $(i = -m, \ldots, -2, -1, 0)$ to represent a Markov process of order m, where S_0 represents the symbol to be observed next and S_{-i} $(1 \le i \le m)$ represents the i-th preceding symbol. The expression of H_m is then given by the following formula:

$$H_m = -\sum_{s_{-m}} \cdots \sum_{s_{-1}} \sum_{s_0} P(s_{-m}, \ldots, s_{-1}, s_0) \log P(s_0 \mid s_{-m}, \ldots, s_{-1}). \quad (12)$$

In theory, the value of H_m is equivalent to the entropy rate of a Markov process of order k as long as $k \le m$ (i.e., $H_m = H_k$) because $P(s_0 \mid s_{-m}, \ldots, s_{-k}, \ldots, s_{-1}) = P(s_0 \mid s_{-k}, \ldots, s_{-1})$ in a Markov process of order k.

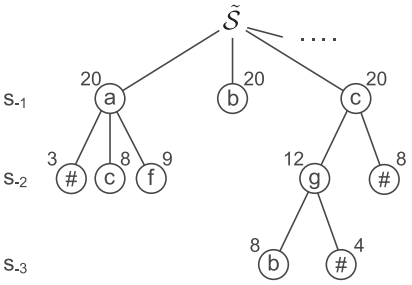

Figure 3: Context tree representation of $\tilde{S}$.

To capture higher-order dependencies without suffering from a lack of sufficient statistics, we model the probability of the occurrence of symbols (vertices) appearing in individuals in the population as a variable-order Markov process. The variable-order Markov model was first suggested in Rissanen (1983) for data compression purposes. Later variants of variable-order Markov models have been successfully applied to areas such as statistical analysis, classification, and prediction (Shmilovici and Ben-Gal, 2007; Begleiter et al., 2004; Ben-Gal et al., 2003). In a variable-order Markov process, the probability distribution of observing the next symbol s_0 depends on the preceding symbols of variable length k. The basic idea is to determine the value of k adaptively such that the number of samples $N(s_{-k}, \ldots, s_{-1})$ is a sufficient statistic for estimating the conditional probability distribution $P(s_0 \mid s_{-k}, \ldots, s_{-1}) = \frac{N(s_{-k},\ldots,s_{-1},s_0)}{N(s_{-k},\ldots,s_{-1})}$.

4.2 A High-Order Entropy Measure H_m^{vari}

A variable-order Markov process is characterized by a set of conditional probability distributions: $P(s_0|\boldsymbol{s_c})\, \boldsymbol{s_c} \in \mathcal{S}$, where $\mathcal{S}$ is a set of sequences of symbols for the conditioning variables. We set the upper limit m on the length of the sequences for the conditioning variables because it is impractical to store all conditional probability components if m is overly large (e.g., $m > 10$). For any sequence of symbols $\{s_{-m}, \ldots, s_{-2}, s_{-1}, s_0\}$, the length of the sequence assigned to the conditioning variables must be uniquely determined. To represent a set $\mathcal{S}$ that satisfies this requirement, a *context tree* (Rissanen, 1983) is useful. Let $\tilde{\boldsymbol{s_c}}$ be the reverse sequence of $\boldsymbol{s_c}$ and we define $\tilde{S} = \{\tilde{\boldsymbol{s_c}}|\boldsymbol{s_c} \in \mathcal{S}\}$. Then, the elements of $\tilde{S}$ are represented as the leaf nodes of a context tree $\tilde{S}$ (we use this symbol to refer to the context tree as well) as illustrated in Figure 3, where each number represents the number of the corresponding sequence existing in the population. Note that a symbol "#" means any symbol other than the symbols of its sibling nodes. For a given sequence $\{s_{-m}, \ldots, s_{-2}, s_{-1}\}$ (s_0 is observed next), the conditioning part is determined by tracing the context tree $\tilde{S}$ to the maximum extent possible from top to bottom according to $s_{-1}, s_{-2}, \ldots, s_{-m}$. For example, if $\{s_{-3}, s_{-2}, s_{-1}\} = \{h, c, a\}$ ($m = 3$), the conditioning part is determined as $\{s_{-2}, s_{-1}\} = \{c, a\}$ and the conditional probability distribution of observing the next symbol s_0 is given by $P(S_0 = s_0|S_{-2} = c, S_{-1} = a)$ or $P(s_0|c, a)$ for short. In another example, if $\{s_{-3}, s_{-2}, s_{-1}\} = \{d, b, a\}$, the conditioning part is determined as $\{s_{-2}, s_{-1}\} = \{\#, a\}$ ("#" means any symbol other than c and f) and the conditional probability distribution of observing the next symbol s_0 is given by $P(S_0 = s_0|S_{-2} = \#, S_{-1} = a)$ or $P(s_0|\#, a)$. This conditional probability distribution is estimated by $\frac{P(\#,a,s_0)}{P(\#,a)} = \frac{P(a,s_0)-P(c,a,s_0)-P(f,a,s_0)}{P(a)-P(c,a)-P(f,a)} = \frac{N(a,s_0)-N(c,a,s_0)-N(f,a,s_0)}{N(a)-N(c,a)-N(f,a)}$.

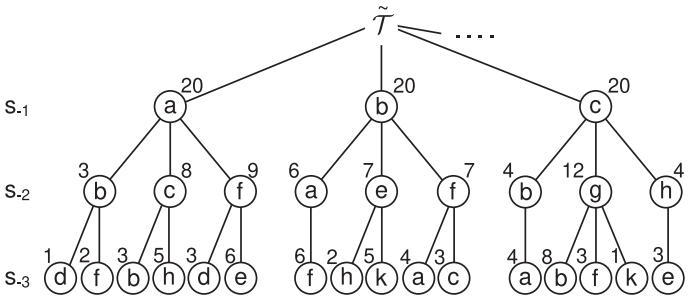

Figure 4: Tree representation of $\tilde{N}(s_{-1}, \ldots, s_{-k})$ $(k \leq 3)$.

The entropy rate of the variable-order Markov process, which we denote as H_m^{vari}, is then defined by

$$H_m^{vari} = -\sum_{\boldsymbol{s_c} \in \mathcal{S}} \sum_{s_0 \in L} P(\boldsymbol{s_c}, s_0) \log P(s_0 \mid \boldsymbol{s_c}) = -\sum_{\boldsymbol{s_c} \in \mathcal{S}} \sum_{s_0 \in L} P(\boldsymbol{s_c}, s_0) \log \frac{P(\boldsymbol{s_c}, s_0)}{P(\boldsymbol{s_c})}$$
$$= -\sum_{\boldsymbol{s_c} \in \mathcal{S}} \sum_{s_0 \in L} P(\boldsymbol{s_c}, s_0) \log P(\boldsymbol{s_c}, s_0) + \sum_{\boldsymbol{s_c} \in \mathcal{S}} P(\boldsymbol{s_c}) \log P(\boldsymbol{s_c}). \tag{13}$$

If the context tree $\tilde{\mathcal{S}}$ is represented as a perfect tree with depth m, H_m^{vari} is equivalent to H_m. Conversely, H_m^{vari} can be viewed as an approximation of H_m because H_m^{vari} is obtained from H_m by replacing $P(s_0 \mid s_{-m}, \ldots, s_{-k}, \ldots, s_{-1})$ with a lower-order conditional probability distribution $P(s_0 \mid s_{-k}, \ldots, s_{-1})$ for all $\{s_{-k}, \ldots, s_{-1}\} \in \mathcal{S}$.

Next, we describe the method to determine $\tilde{\mathcal{S}}$ (and equivalently $\mathcal{S}$), which is essentially equivalent to the learning algorithms used in Rissanen (1983), Ron et al. (1996), Mächler and Bühlmann (2004), and Schulz et al. (2008). Let $\tilde{N}(s_{-1}, \ldots, s_{-k})$ be the number of the reverse sequence of symbols $\{s_{-k}, \ldots, s_{-1}\}$ in the population, i.e., $\tilde{N}(s_{-1}, \ldots, s_{-k}) = N(s_{-k}, \ldots, s_{-1})$. As with tree $\mathcal{T}$, let $\tilde{\mathcal{T}}$ be a tree for storing the values of $\tilde{N}(s_{-1}, \ldots, s_{-k})$ $(k \leq m)$. Figure 4 illustrates an example of the tree $\tilde{\mathcal{T}}$, which corresponds to $\mathcal{T}$ presented in Figure 2 (e.g., $N(a, f, b) = \tilde{N}(b, f, a) = 4$). Note that in the case of the STSP, $\mathcal{T} = \tilde{\mathcal{T}}$ (except for the maximum depth of the tree), but the displayed example shows the case of the ATSP (because it is easier to understand). For a current population, the context tree $\tilde{\mathcal{S}}$ is constructed by the following procedure, where $ratio$ is a parameter taking a value between zero and one.

Construction of context tree $\tilde{\mathcal{S}}$

1. $\tilde{\mathcal{S}}$ is initialized as the perfect tree of depth one, i.e., $\tilde{\mathcal{S}} = \{s_{-1} | s_{-1} \in L\}$.
2. For each of the leaf nodes $\{s_{-1}, \ldots, s_{-k}\} \in \tilde{\mathcal{S}}$ with $k < m$, if there exists a symbol(s) $s'_{-(k+1)} \in L$ such that $ratio \times N_p \leq \tilde{N}(s_{-1}, \ldots, s_{-k}, s'_{-(k+1)})$, this node is expanded to generate a new leaf node(s) $\{s_{-1}, \ldots, s_{-k}, s'_{-(k+1)}\}$. Expansions of the leaf nodes are iterated until no further expansion is possible.
3. For every node that is already expanded, generate a child node with a symbol "#".

According to the above procedure, the context tree $\tilde{\mathcal{S}}$ presented in Figure 3 is obtained from $\tilde{\mathcal{T}}$ presented in Figure 4 (if $ratio \times N_p = 8$).

Algorithm 1 : Procedure GA

1: $\{x_1, \ldots, x_{N_p}\} :=$ GENERATE_INITIAL_POP();
2: **repeat**
3: $r(\cdot) :=$ a random permutation of $1, \ldots, N_p$;
4: **for** $i := 1$ to N_p **do**
5: $p_A := x_{r(i)}$, $p_B := x_{r(i+1)}$; // $r(N_p + 1) = r(1)$
6: $\{c_1, \ldots, c_{N_{ch}}\} :=$ CROSSOVER(p_A, p_B);
7: $x_{r(i)} :=$ SELECT_BEST$(c_1, \ldots, c_{N_{ch}}, p_A)$;
8: **end for**
9: **until** a termination condition is satisfied
10: **return** the best individual in the population;

The aim behind the expansion of a leaf node $\{s_{-1}, \ldots, s_{-k}\}$ (Step 2 of the above procedure) is to capture the higher-order dependency expressed as the conditional probability distribution $P(s_0|s'_{-(k+1)}, s_{-k}, \ldots, s_{-1})$ only when it is judged to have a sufficient statistic for estimating this conditional probability distribution. The parameter $ratio$ balances the tradeoff between the potential ability to capture higher-order dependencies and the estimate accuracy of the conditional probability distributions.

In Rissanen (1983), Ron et al. (1996), Mächler and Bühlmann (2004), and Schulz et al. (2008), the context tree constructed by the above algorithm is then pruned based on the magnitude of the effect on the stochastic model. However, we do not use this pruning procedure because we confirmed in a preliminary experiment that the performance of the GA deteriorated by introducing this pruning procedure.

5 GA Framework

To evaluate the ability of the proposed population diversity measures H_m, H_m^{adj}, and H_m^{vari}, we perform the GA proposed in Nagata and Kobayashi (2013) using each of the three types of the population diversity measures with different values of m and $ratio$. This GA is one of the most effective heuristic algorithms for the TSP. One important factor for achieving top performance is to maintain population diversity by evaluating offspring solutions based on the change in population diversity when they are selected to survive in the population as well as the tour length. The independent entropy measure H^{ind} was originally used for evaluating population diversity.

Algorithm 1 depicts the GA framework. The population consists of N_p individuals. The initial population is generated by a greedy local search algorithm with the *2-opt* neighborhood (Line 1). At each generation (Lines 3–8) of the GA, each of the population members is selected, once as parent p_A and once as parent p_B, in random order (Lines 3 and 5). For each pair of parents, edge assembly crossover (EAX) operator generates the N_{ch} (e.g., 30) offspring solutions (Line 6). Then, a best solution is selected from the generated offspring solutions and p_A in terms of a given evaluation function, and the selected individual replaces the population member selected as p_A (Line 7). Therefore, no replacement occurs if all offspring solutions are worse than p_A. Note that only parent p_A is replaced to better maintain population diversity because EAX typically generates offspring solutions similar to p_A. Iterations of generation are repeated until a termination condition is achieved (Line 9).

Figure 5 illustrates an outline of EAX. From a selected pair of parent solutions (tours), EAX can generate a number of offspring solutions (tours) through the two phases. In the first phase, intermediate solutions are constructed by assembling edges

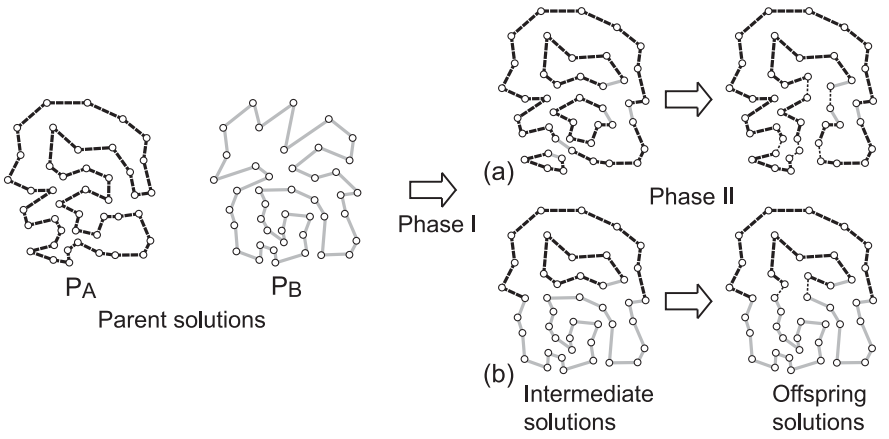

Figure 5: Outline of EAX for the TSP.

of the parent solutions (tours) under the relaxed constraint that exactly two edges are linked to every vertex; an intermediate solution typically consists of several subcycles. In the second phase, each intermediate solution is modified into a Hamilton cycle by merging the subcycles heuristically. There are several strategies for generating offspring solutions. For example, EAX with "single" strategy typically generates an intermediate solution like (a) in the figure, where the majority of the edges are inherited from parent p_A. Conversely, EAX with "block 2" strategy typically generates an intermediate solution like (b), where a number of edges are inherited from each parent. EAX with single strategy is used from the beginning to almost the end of the search and EAX with block 2 strategy is used only at the final stage of the search. The timing to switch from single strategy to block 2 strategy and the termination condition are determined based on the number of consecutive generations for which the best individual of the population is not updated. For more details, we refer the reader to the original paper (Nagata and Kobayashi, 2013).

We define the evaluation function for selecting the individual that replaces $x_{r(i)}(= p_A)$ (Line 7) in a somewhat different manner from the one used in Nagata and Kobayashi (2013) because this evaluation function is natural than the original. Let L be the average tour length of the population and H the population diversity measure selected (H_m, H_m^{adj}, or H_m^{vari}). First of all, (as in Nagata and Kobayashi, 2013) only offspring solutions that do not deteriorate the tour length of $x_{r(i)}$ can be selected because without this restriction, we need a careful cooling schedule for a parameter T (described later) to converge the population. The best one is then selected among them such that $L - TH$ is minimized after the replacement, where T is a parameter that controls the balance between the exploration (maximization of H) and exploitation (minimization of L) of the search. Therefore, offspring solutions are evaluated by the following evaluation function (a lower value is better):[1]

$$Eval(y) = \begin{cases} \Delta L(y) - T\Delta H(y) & (\Delta L \leq 0) \\ \infty & (\Delta L > 0) \end{cases}, \tag{14}$$

[1]This strategy is useful because many of offspring solutions generated by EAX improve (or do not change) the tour length of $x_{r(i)}$. Otherwise, offspring solutions should be evaluated simply by $\Delta L(y) - T\Delta H(y)$ while reducing the value of T in the course of the search.

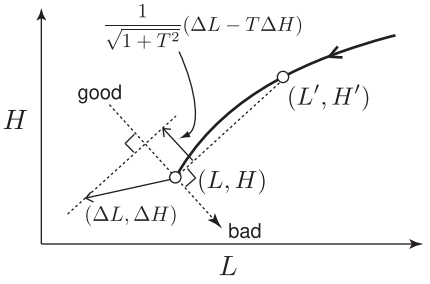

Figure 6: Meaning of the evaluation function $\Delta L(y) - T\Delta H(y)$.

where $\Delta L(y)$ and $\Delta H(y)$ denote the differences in L and H, respectively, when an offspring solution y replaces $x_{r(i)}$. After evaluating all offspring solutions, the one with the smallest evaluation value is selected to replace $x_{r(i)}$. Note that if all evaluation values of the offspring solutions are greater than zero, p_A is selected (i.e., no replacement occurs) because $Eval(p_A) = 0$. We update the parameter T in the following manner. At the beginning of the GA, T is set to ∞ (a sufficiently large value). The value of T is updated every time the replacement of $x_{r(i)}$ is performed N_p times. Let L' and H' be the values of L and H, respectively, when the value of T is last updated. The value of T is then updated by $T = \frac{L-L'}{H-H'}$. Figure 6 illustrates the meaning of this evaluation function. In this figure, the solid curve represents a trajectory of (L, H) in the L-H plane during the search. If an offspring solution y replaces $x_{r(i)}$, the current position (L, H) moves by the vector $(\Delta L(y), \Delta H(y))$ as illustrated in the figure (y is omitted in the figure). The value of $\Delta L(y) - T\Delta H(y)$ is proportional to the magnitude of the vector $(\Delta L(y), \Delta H(y))$ in the direction (dotted arrow in the figure) perpendicular to the vector $(L - L', H - H')$. Note that the value of $\Delta L(y) - T\Delta H(y)$ is negative in the example of the figure and a lower value is preferable.

For every offspring solution y, we must compute $\Delta L(y)$ and $\Delta H(y)$ to obtain the value of $Eval(y)$. However, the computational cost of $\Delta H(y)$ for H_m, H_m^{adj}, and H_m^{vari} is not negligible, especially for large values of m. Although this problem is partially alleviated by computing $\Delta H(y)$ only when $\Delta L(y) \leq 0$, we require an efficient algorithm for computing $\Delta H(y)$. An outline of the efficient computation of $\Delta H(y)$ is presented in the Appendix.

When the population diversity measure H_m^{vari} is used, the context tree $\tilde{\mathcal{S}}$ should be updated each time $x_{r(i)}$ is replaced. However, we decided to update the context tree at the beginning of each generation (before Line 3) because we confirmed that the results did not change significantly using this update method and an efficient algorithm for the immediate update of the context tree is complicated (although it can be implemented).

6 Experimental Results

We now present experimental results to analyze the ability of the proposed high-order entropy measures. The GA was implemented in C++ on Ubuntu 14.04 and the program code was executed on PCs with Intel Core i7-4790 CPU/3.60 GHz processor.

6.1 Experimental Settings

To investigate the ability of the three types of high-order entropy measures H_m, H_m^{adj}, and H_m^{vari}, we performed the GA described in Section 5 using each of the population diversity measures in the evaluation function (14). We used the default configuration

of the GA except for the population diversity measure (H^{ind} was used in the original GA), where $N_p = 300$ and $N_{ch} = 30$ in the default configuration (Nagata and Kobayashi, 2013). In addition, we tested $N_p = 50$ to investigate the effect of the population size. We also tested the GA using either the pairwise Hamming distance D (Eq. (3)) or no population diversity measure (set $T = 0$ in the evaluation function) to evaluate the baseline ability of H^{ind}. The population diversity measures tested are summarized as follows:

- Greedy (no population diversity measure), D, H^{ind}
- H_m ($m = 2, 3, 4, 5, 6, 8$)
- H_m^{adj} ($m = 2, 3, 4, 5, 6, 8$)
- H_m^{vari} ($m = 6;\ ratio = 0.02, 0.05, 0.1, 0.2, 0.4$)

Note that H_1 and H_1^{adj} are equivalent to H^{ind}. For H_m^{vari}, the value of m was set to six, and we show the results of H_6^{vari} with different values of the parameter $ratio$ because the amount of data for possible combinations of m and $ratio$ is very large.

For each population diversity measure, we performed 30 independent runs of the GA on instances in the following three well-known benchmark sets for the STSP, where we selected all 54 instances of sizes ranging from 4,000 to 30,000.[2]

TSPLIB A most widely-used collection of TSP instances drawn from industrial applications and from geographic problems featuring the locations of cities on maps (available at http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/).

National TSPs A collection of TSP instances that are based on the real-world locations of cities in selected countries (available at http://www.math.uwaterloo.ca/tsp/vlsi/index.html).

VLSI TSPs A collection of TSP instances that are based on VLSI circuit design (available at http://www.math.uwaterloo.ca/tsp/vlsi/index.html).

6.2 Results

Table 1 lists the results of the GA using different population diversity measures under the population size $N_p = 300$, where detailed results only for some selected population diversity measures including $Greedy$ and H^{ind} are presented to avoid displaying a huge amount of data; for each of H_m, H_m^{adj}, and H_6^{vari}, the parameter value of m or $ratio$ (=0.1) that achieved relatively good results is selected. Each line presents the instance name (instance), the optimal or best-known solution (Opt. or UB), the number of runs that succeeded in finding the optimal or best-known solution (Su), and the average error of the best tour lengths in the 30 runs from the optimal or best-known solutions (A-Err), where one unit in the column A-Err is 10^{-5}%. Full results for all population diversity measures are provided in the online supplementary file, available at https://www.mitpressjournals.org/doi/suppl/10.1162/evco_a_00268.

For each population diversity measure H, we performed the one-sided Wilcoxon rank sum test for the null hypothesis that the median of the distribution of the tour

[2] At present, several TSP generators are available and it is possible to make comparisons on hundreds or thousands of TSP instances (Kotthoff et al., 2015; Kerschke et al., 2018). However, it takes a huge amount of time to perform this for instances of similar size and we did not perform this.

Table 1: Solution quality of GA using different population diversity measures.

		Greedy		H^{ind}		H_3		H_6^{adj}		H_6^{vari}	
Instance	Opt.(UB)	Su	A-Err	Su	A-Err	Su	A-Err	Su	A-Err	Su	A-Err
fnl4461	182566	0	858†	30	0	30	0	30	0	30	0
rl5915	565530	2	1294†	30	0	30	0	30	0	23	82†
rl5934	556045	2	4130†	23	342	28	64*	26	164	27	109
pla7397	23260728	0	894†	3	251	0	585†	0	425†	0	502†
rl11849	923288	0	835†	24	12	25	12	28	2	28	5
usa13509	19982859	0	816†	14	18	23	9*	16	13	25	7*
brd14051	469385	0	820†	29	2	22	15†	26	7	24	12†
d15112	1573084	0	512†	18	8	14	11	23	4	16	4
d18512	645238	0	747†	18	7	21	12	25	3*	19	10
ca4663	1290319	1	555†	27	24	30	0*	30	0*	30	0*
pm4951	114855	0	4600†	24	87	9	255†	21	240	25	142
tz6117	394718	0	717†	12	99	18	72	14	99	19	72
ar6723	837479	0	1822†	25	19	24	23	23	27	25	19
ho7103	177092	0	717†	24	22	21	39	19	43	11	84†
eg7146	172386	0	446†	23	19	21	34	29	3*	30	0*
ym7663	238314	0	1759†	29	2	22	288†	27	6	28	92
ei8246	206171	0	994†	13	79	28	9*	30	0*	30	0*
ja9847	491924	0	2598†	6	83	12	43*	9	58*	15	48*
gr9882	300899	0	614†	12	163	13	55	16	46	19	37*
kz9976	1061881	0	1283†	27	4	29	3	29	1	30	0*
fi10639	520527	0	796†	15	33	25	10*	29	0*	25	10*
mo14185	(427377)	0	627†	13	28	18	18	24	8*	21	13*
it16862	557315	0	1065†	3	57	9	44	6	20*	6	24*
vm22775	569288	0	1330†	0	116	0	138	0	115	1	92
sw24978	855597	0	1014†	14	29	10	31	15	18	14	30
bgb4355	12723	3	4034†	30	0	30	0	30	0	30	0
bgd4396	13009	19	486†	27	76	25	128	30	0*	30	0*
frv4410	10711	9	2116†	30	0	30	0	30	0	30	0
bgf4475	13221	7	1159†	25	126	30	0*	30	0*	30	0*
xqd4966	15316	1	1654†	30	0	30	0	30	0	30	0
fqm5087	13029	7	1074†	30	0	29	25	29	25	30	0
fea5557	15445	17	841†	29	21	30	0	30	0	30	0
xsc6880	21535	0	3451†	22	216	27	46*	28	30*	25	77
bnd7168	21834	18	305†	30	0	30	0	30	0	30	0
lap7454	19535	4	784†	30	0	30	0	30	0	30	0
ida8197	22338	1	1910†	29	14	28	29	30	0	30	0
dga9698	27724	2	1262†	28	36	30	0	30	0	30	0
xmc10150	28387	1	1585†	15	258	24	82*	30	0*	25	58*
xvb13584	37083	1	1240†	21	80	27	26*	27	26*	29	8*
xrb14233	(45462)	0	1649†	2	461	4	381*	9	300*	15	227*
xia16928	(52850)	0	1255†	19	113	19	145	17	145	20	88
pjh17845	(48092)	0	1919†	12	138	19	97	16	97	16	97
frh19289	(55798)	1	1409†	27	23	29	5	30	0*	30	0*
fnc19402	(59287)	0	1388†	19	67	20	73	26	22*	21	50
ido21215	(63517)	0	1548†	20	104	22	68	28	15*	19	78
fma21553	(66527)	0	1332†	12	135	22	45*	14	95	23	35*

Table 1: Continued.

		Greedy		H^{ind}		H_3		H_6^{adj}		H_6^{vari}	
Instance	Opt.(UB)	Su	A-Err	Su	A-Err	Su	A-Err	Su	A-Err	Su	A-Err
lsb22777	(60977)	1	$754^{\dagger}$	19	65	26	21^{*}	26	27^{*}	29	5^{*}
xrh24104	(69294)	0	$1279^{\dagger}$	28	9	29	4	27	14	30	0
bbz25234	(69335)	0	$1413^{\dagger}$	23	38	28	14^{*}	29	4^{*}	28	9^{*}
irx28268	(72607)	0	$1221^{\dagger}$	28	9	27	13	26	18	27	13
fyg28534	(78562)	0	$1251^{\dagger}$	12	106	19	50^{*}	18	59^{*}	24	29^{*}
icx28698	(78088)	0	$1494^{\dagger}$	1	217	4	170^{*}	13	93^{*}	10	102^{*}
boa28924	(79622)	0	$1486^{\dagger}$	4	121	5	117	3	117	5	108
ird29514	(80353)	0	$2152^{\dagger}$	4	215	12	99^{*}	14	87^{*}	19	58^{*}
Statistical test		W=0: L=54		—		W=17: L=4		W=22: L=1		W=22: L=4	

length (of the best solution of each run) obtained with H is greater than that with H^{ind}. If the null hypothesis was rejected at a significance level of 0.05, the corresponding value in the column A-Err is indicated by the asterisk (indicating that H was significantly superior to H^{ind}). Conversely, if the opposite null hypothesis was rejected, the corresponding value is indicated by the dagger (indicating that H was significantly worse than H^{ind}). For each population diversity measure, the numbers of asterisks and daggers are presented in the bottom line (Statistical test) of the table, where "W=a: L=b" indicates that the numbers of asterisks and daggers are a and b, respectively.

We performed the same one-sided Wilcoxon rank sum test between H^{ind} and each of all population diversity measures under the population size $N_p = 50$ and 300, and the summarized results (W and L) are presented in Table 2. Note that when the GA using H^{ind} found optimal (or best-known) solutions in most trials of the 30 runs, we cannot find a statistically significant difference even if the GA using the population diversity measure H finds optimal (or best-known) solutions for all 30 runs. This situation is particularly noticeable when the number of vertices is small (e.g., $n < 10{,}000$). Therefore, the results in Table 2 are presented separately for all 54 instances, a set of the 27 instances with $n < 10{,}000$, and a set of the 27 instances with $10{,}000 \leq n$. When the population size is 300, the numbers of instances from which a statistically significant difference can be detected (if the optimal (best-known) solution is found in all runs) are 15 ($n < 10{,}000$) and 24 ($10{,}000 \leq n$). On the other hand, this situation does not occur when the population size is 50.

Table 2 indicates that the GA using the independent entropy measure H^{ind} clearly outperforms the GA using either no population diversity measure or the pairwise Hamming distance D. In the following, we first compare the results of the three types of the high-order entropy measures when the population size is 300.

Results of H_m Table 2 shows that the results of H_m gradually improves as the value of m increases from one (H_1 is equivalent to H^{ind}) to three or four. For greater values of m, however, the results of the statistical test gradually deteriorates with increasing the value of m. These results demonstrate that the ability of evaluating population diversity can be improved by considering high-order dependencies between consecutive symbols (vertices) in the population. As we expected, however, there is a tradeoff between the potential ability to capture higher-order dependencies and the estimate accuracy of

Table 2: Results of Wilcoxon rank sum test between H^{ind} and other population diversity measures.

		$N_p = 300$						$N_p = 50$					
		All		$n < 10^4$		$10^4 \leq n$		All		$n < 10^4$		$10^4 \leq n$	
Population diversity		W	L	W	L	W	L	W	L	W	L	W	L
Greedy	—	0	54	0	27	0	27	0	54	0	27	0	27
D	—	0	43	0	22	0	21	0	54	0	27	0	27
H_m	$m = 2$	11	4	3	2	8	2	11	0	3	0	8	0
	$m = 3$	17	4	6	3	11	1	6	10	3	3	3	7
	$m = 4$	15	3	7	3	8	0	6	14	5	4	1	10
	$m = 5$	13	11	5	8	8	3	3	31	3	10	0	21
	$m = 6$	14	13	7	6	7	7	1	42	1	15	0	27
	$m = 8$	9	21	4	7	5	14	0	49	0	22	0	27
H_m^{adj}	$m = 2$	11	1	4	0	7	1	13	0	5	0	8	0
	$m = 3$	17	2	5	1	12	1	26	0	10	0	16	0
	$m = 4$	23	1	8	1	15	0	29	0	13	0	16	0
	$m = 5$	22	1	6	0	16	1	33	0	17	0	16	0
	$m = 6$	22	1	7	1	15	0	24	0	15	0	9	0
	$m = 8$	18	4	7	4	11	0	20	1	14	0	6	1
H_6^{vari} ($r = ratio$)	$r = 0.02$	17	5	8	4	9	1	1	42	1	16	0	26
	$r = 0.05$	20	5	8	4	12	1	3	27	3	7	0	20
	$r = 0.1$	22	4	8	3	14	1	13	3	8	1	5	2
	$r = 0.2$	19	5	6	3	13	2	31	2	13	1	18	1
	$r = 0.4$	17	3	4	2	13	1	21	2	6	2	15	0

the conditional probability distributions required for computing H_m. The experimental results indicate that $m = 3$ or 4 achieves an appropriate tradeoff between these for this population size ($N_p = 300$).

Results of H_m^{adj} Table 2 shows that the results of H_m^{adj} gradually improves as the value of m increases from one (H_1^{adj} is equivalent to H^{ind}) to four, five, or six. However, the result of H_8^{adj} is worse than that of H_6^{adj}, though it is still better than that of H^{ind}. As can be predicted (see Section 3.2) and supported by the results, the best value of m for H_m^{adj} is greater than that for H_m. More importantly, the best result of H_m^{adj} (obtained with $m = 4, 5$, or 6) is superior to the best result of H_m (obtained with $m = 3$ or 4). This suggests that H_m^{adj} (with an appropriate value of m) is a better population diversity measure than H_m. We analyze the reason for this in the next subsection.

Results of H_6^{vari} In preliminary experiments for selected instances, better results were obtained when $m = 6$ than when $m = 4$, but there was no significant difference in results between $m = 6$ and $m = 8$. Table 2 shows that the GA using H_6^{vari} outperforms the GA using H^{ind} for all values of $ratio$ ranging from 0.02 to 0.4, where the best value of $ratio$ is around 0.1. When $H_6^{vari}(ratio = 0.1)$ is compared with H_m and H_m^{adj}, the result of H_6^{vari} ($ratio = 0.1$) is better than the results of H_3 and H_4 (the best results of H_m). This

indicates that the variable-order Markov model introduced to estimate H_m (see Section 4.1) works as expected. However, the result of H_6^{vari} ($ratio = 0.1$) is slightly worse than the results of H_4^{adj}, H_5^{adj}, and H_6^{adj} (the best results of H_m^{adj}).

Results on hard instances The result of H^{ind} is poor (e.g., Su < 5) for some instances. This is because there is a deceptive structure on these instances, i.e., some edges of the optimal solution are not included in a majority of near-optimal solutions. In general, finding the optimal solution for such instances is more difficult. Table 1 shows that the results of H^{ind} for these instances are more or less improved by using any of the high-order entropy measures H_3, H_6^{adj}, and H_6^{vari} except for instance pla7397.

Effect of the population size Next, we describe the results of the three types of the high-order entropy measures when the population size is 50, where we focus mainly on the effect of the population size. Table 2 shows that the use of H_m improves the result of H^{ind} only when $m = 2$, whereas the result of H^{ind} is improved by using H_m ($m = 2, 3, 4, 5$) under the population size of 300. This makes sense because for a smaller population size, it will be more difficult to obtain a sufficient number of samples (sequences of symbols) from the population necessary to estimate the conditional probability distributions for all values of m. On the other hand, the superiority of H_m^{adj} over H^{ind} is retained for large values of m even in the small population size ($N_p = 50$). As for the high-order entropy measure H_6^{vari}, the best value of $ratio$ is around 0.2, which is greater than that ($= 0.1$) when $N_p = 300$. Given the role of $ratio$ in constructing the context tree $\tilde{\mathcal{S}}$, this is an expected result. Therefore, the value of $ratio$ should be set appropriately depending on the population size.

Execution time Next, we discuss the difference in the computation time of the GA when using the different population diversity measures. Table 3 lists the average computation time in seconds for the GA using the different population diversity measures. Results are presented for six selected instances listed in the table to avoid displaying a huge amount of data, where two instances ($10{,}000 < n$) were randomly selected from each of the three benchmark sets. In addition, the bottom line (Ave. ratio) of the table shows the ratio of the execution time to that of the GA using H^{ini} averaged over the six instances. The table shows that the execution time tends to increase as the value of m increases in the population diversity measures H_m and H_m^{adj}. This is mainly because of the increase in the computational effort of calculating $\Delta H(y)$ in the evaluation function (14). Note that the execution times of H_m are smaller than those of H_m^{adj} (for the same value of m) even though the calculation of H_m^{adj} is simpler than that of H_m. The reason for this is that the number of generations of the GA required to complete the search was smaller when H_m was used than when H_m^{adj} was used. As for the population diversity measure H_6^{vari} with various values of $ratio$, the results are similar to that of H_6^{adj}.

6.3 Analysis

We analyze the behavior of the GA using different population diversity measures to investigate their influence on preserving population diversity. Here, we focus mainly on why the GA using H_6^{adj} achieves superior results compared with the GA using H_m with various values of m. For this purpose, we depict the typical behavior of a single run of the GA using each of six population diversity measures (Greedy, H^{ind}, H_3, H_4, H_6, and H_6^{adj}) on instance usa13509 under the population size of 300. We omit results on other instances because the same tendency was observed. The first graph in Figure

Table 3: Average execution time (in seconds) of the GA using different population diversity measures.

				$H_6^{vari}(ratio)$				
	H^{ind}	$Greedy$	D	(0.02)	(0.05)	(0.1)	(0.2)	(0.4)
usa13509	2429	1097	2442	3458	3704	3620	3475	3068
d15112	3482	1932	3640	5743	5543	5127	4841	4284
it16862	2958	1547	3009	5201	4803	4600	4341	4104
pjh17845	1636	947	1677	2803	2685	2600	2497	2351
fma21553	2053	1094	2084	3518	3354	3285	3154	2982
sw24978	5930	3351	6142	10388	9769	9386	8744	8199
Ave. ratio	—	0.53	1.02	1.69	1.61	1.55	1.47	1.36

	H_m						H_m^{adj}					
	$m=2$	$m=3$	$m=4$	$m=5$	$m=6$	$m=8$	$m=2$	$m=3$	$m=4$	$m=5$	$m=6$	$m=8$
usa13509	2402	2352	2451	2614	2726	3069	2619	2717	2951	3218	3564	4471
d15112	3562	3550	3634	3982	4161	4901	3718	3762	4106	4567	4967	6632
it16862	3077	3087	3208	3406	3720	4294	3162	3290	3649	4139	4614	5936
pjh17845	1675	1729	1866	1969	2108	2488	1759	1861	2014	2257	2443	3033
fma21553	2109	2133	2248	2385	2582	3037	2214	2294	2528	2789	3072	3821
sw24978	6259	6065	6433	6849	7340	8382	6295	6729	7504	8169	9168	11600
Ave. ratio	1.03	1.03	1.08	1.15	1.23	1.42	1.07	1.12	1.23	1.36	1.50	1.90

7 displays the plots of the average tour length L of the population against the number of generations of the GA. This graph is presented as a reference for the following four graphs. The remaining four graphs display the plots of H_1, H_3, H_4, and H_6 against the average tour length L.

Figure 7 indicates that each value of H_1, H_3, H_4, and H_6 is maintained at the highest value at each value of L when the same diversity measure is incorporated into the evaluation function (14) of the GA. However, maintaining the value of H_m at the highest level for a small value of m does not necessarily lead to maintaining H_k $(m < k)$ at a high level. For example, if H^{ind} (equivalently H_1) is incorporated into the evaluation function, the population is evolved such that duplication of sequences of length two in the population is suppressed without considering the increase of duplication of longer sequences. On the other hand, maintaining the value of H_m at the highest level for a large value of m (e.g., $m = 6$) leads to "overfitting" of the population specialized in maintaining the value of H_m high. For example, if it is possible to exclude any duplication of sequence of length $m + 1$ in the population, such a population is preferred to maintain the value of H_m as high as possible even if the duplication of a certain shorter sequence increases excessively.

To alleviate the overfitting problem of H_m for a large value of m, not only the values of H_m but also the values of H_k $(k = 1, \ldots, m-1)$ should also be maintained at a high level. The high-order entropy measure H_m^{adj} is suitable for this purpose because H_m^{adj} is equivalent to $H_1 + \cdots + H_m$, and therefore the population is evolved such that each of the values of H_k $(k \leq m)$ is maintained high although there is no guarantee that all values are maintained near their highest levels. Fortunately, as can be observed from Figure 7,

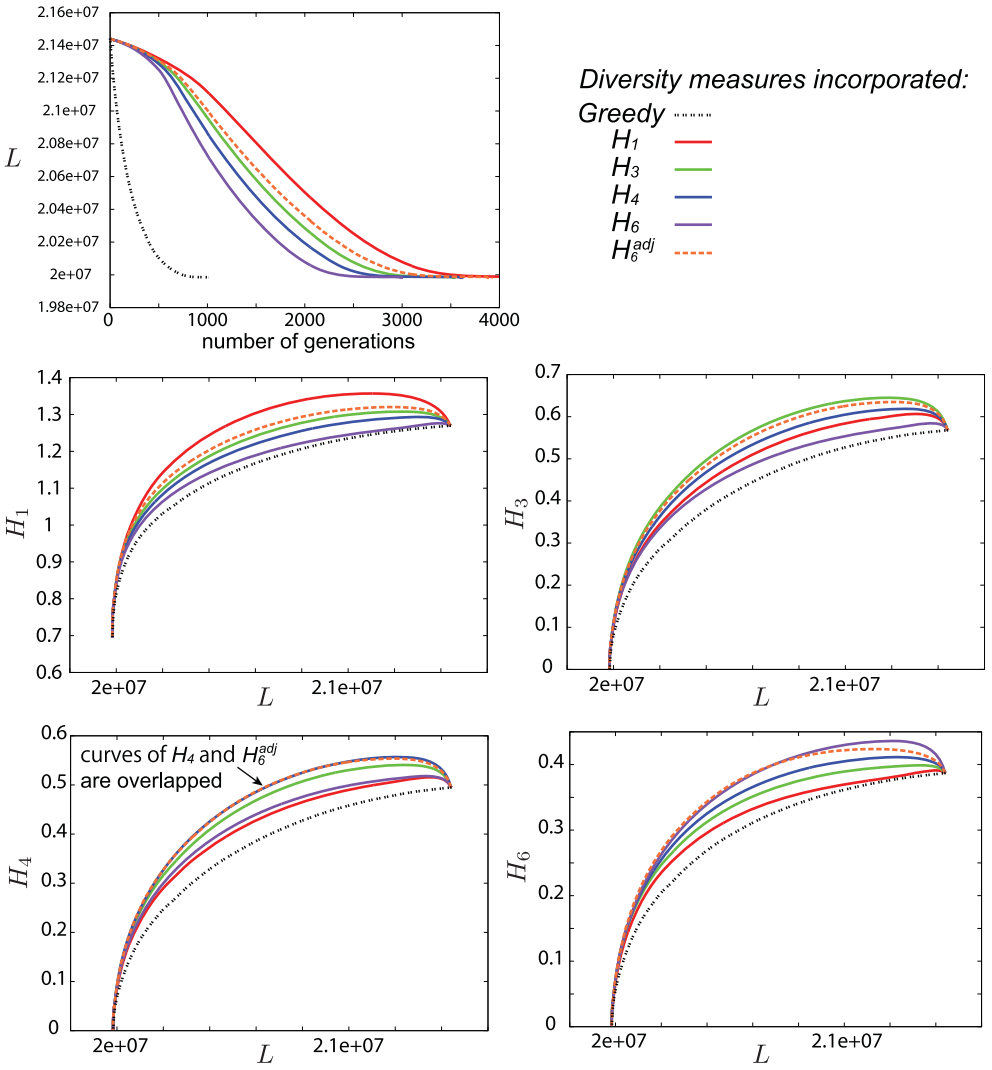

Figure 7: Behavior of GA using different population diversity measures on instance usa13509 ($N_p = 300$).

when H_6^{adj} is incorporated into the evaluation function, the values of H_k ($k \leq 6$) are all maintained near their highest level, especially for $3 \leq k$. Therefore, we conclude that this is a reason why the GA with a high-order entropy measure H_6^{adj} achieves superior results compared with the GA using each of H_m ($m = 1, \ldots, 6$).

7 Conclusions

We proposed three types of entropy-based population diversity measures to evaluate population diversity of a GA for the TSP. These measures consider high-order dependencies between variables of individuals in the population (high-order entropy measures). To derive these, we considered dependencies between consecutive variables and assumed that an individual is represented as a circular sequence of symbols, which is

well suited for the TSP. Under these conditions, the entropy of the probability distribution of individuals in the population (used as a population diversity measure) is defined as the entropy rate of a Markov process estimated from the sequences of symbols sampled from the population.

The high-order entropy measure H_m is equivalent to the entropy rate of the m-th–order Markov process. It has the potential ability to capture dependencies between consecutive variables of length up to $m+1$. We demonstrated that H_m with an appropriate value of m ($=3$ or 4) is significantly superior to H^{ind}, the commonly used entropy-based population diversity measure that does not consider dependencies between variables, in the ability to evaluate population diversity. Although the high-order entropy measure H_m^{adj} is defined in a somewhat ad hoc manner, it is essentially equivalent to $H_1 + H_2 + \cdots + H_m$. It reduces the overfitting problem of H_m for a large value of m (e.g. $m = 6$) while considering dependencies between consecutive variables of length up to $m+1$. Consequently, H_m^{adj} with an appropriate value of m ($= 4, 5,$ or 6) further improves H_m in the ability to measure population diversity. The high-order entropy measure H_m^{vari} is equivalent to the entropy rate of the variable-order Markov process. It also reduces the overfitting problem of H_m with an appropriate parameter setting (e.g., $m = 6$ and $ratio = 0.1$) and improves H_m. Overall, the high-order entropy measure H_m^{adj} with an appropriate value of m ($= 4, 5,$ or 6) is the best population diversity measure among all the population diversity measures tested.

We have demonstrated the effectiveness of considering high-order dependencies between variables of individuals in evaluating population diversity at least for the TSP. Development of other high-order entropy measures and their application to other combinatorial optimization problems remains as a future research direction.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Number 17K00342.

References

Begleiter, R., El-Yaniv, R., and Yona, G. (2004). On prediction using variable order Markov models. *Journal of Artificial Intelligence Research*, 22:385–421.

Ben-Gal, I., Morag, G., and Shmilovici, A. (2003). Context-based statistical process control: A monitoring procedure for state-dependent processes. *Technometrics*, 45(4):293–311.

Kerschke, P., Kotthoff, L., Bossek, J., Hoos, H. H., and Trautmann, H. (2018). Leveraging TSP solver complementarity through machine learning. *Evolutionary Computation*, 26(4):597–620.

Kotthoff, L., Kerschke, P., Hoos, H., and Trautmann, H. (2015). Improving the state of the art in inexact TSP solving using per-instance algorithm selection. In *Proceedings of the 9th International Conference on Learning and Intelligent Optimization*, pp. 202–217. Lecture Notes in Computer Science, Vol. 8994.

Mächler, M., and Bühlmann, P. (2004). Variable length Markov chains: Methodology, computing, and software. *Journal of Computational and Graphical Statistics*, 13(2):435–455.

Maekawa, K., Mori, N., Tamaki, H., Kita, H., and Nishikawa, Y. (1996). A genetic solution for the traveling salesman problem by means of a thermodynamical selection rule. In *Proceedings of the 3rd IEEE Conference on Evolutionary Computation*, pp. 529–534.

Mori, N., Kita, H., and Nishikawa, Y. (1996). Adaptation to a changing environment by means of the thermodynamical genetic algorithm. In *Proceedings of the 9th International Conference*

on Parallel Problem Solving from Nature, pp. 513–522. Lecture Notes in Computer Science, Vol. 4193.

Nagata, Y. (2006). Fast EAX algorithm considering population diversity for traveling salesman problems. In *Proceedings of the 6th International Conference on Evolutionary Computation in Combinatorial Optimization*, pp. 171–182. Lecture Notes in Computer Science, Vol. 3906.

Nagata, Y. (2016). Population diversity measures based on variable-order Markov models for the traveling salesman problem. In *Proceedings of the 14th International Conference on Parallel Problem Solving from Nature*, pp. 973–983. Lecture Notes in Computer Science, Vol. 9921.

Nagata, Y., and Kobayashi, S. (2013). A powerful genetic algorithm using edge assembly crossover for the traveling salesman problem. *Informs Journal on Computing*, 25(2):346–363.

Nagata, Y., and Ono, I. (2013). High-order sequence entropies for measuring population diversity in the traveling salesman problem. In *Proceedings of the 13th European Conference on Evolutionary Computation in Combinatorial Optimization*, pp. 179–190. Lecture Notes in Computer Science, Vol. 7832.

Rissanen, J. (1983). A universal data compression system. *IEEE Transactions on Information Theory*, 29(5):656–664.

Ron, D., Singer, Y., and Tishby, N. (1996). The power of amnesia: Learning probabilistic automata with variable memory length. *Machine Learning*, 25(2–3):117–149.

Schulz, M. H., Weese, D., Rausch, T., Döring, A., Reinert, K., and Vingron, M. (2008). Fast and adaptive variable order Markov chain construction. In *Proceedings of the 8th International Workshop on Algorithms in Bioinformatics*, pp. 306–317. Lecture Notes in Bioinformatics, Vol. 5251.

Shmilovici, A., and Ben-Gal, I. (2007). Using a VOM model for reconstructing potential coding regions in EST sequences. *Computational Statistics*, 22(1):49–69.

Squillero, G., and Tonda, A. (2016). Divergence of character and premature convergence: A survey of methodologies for promoting diversity in evolutionary optimization. *Information Sciences*, 329:782–799.

Tsai, H., Yang, J., Tsai, Y., and Kao, C. (2004). An evolutionary algorithm for large traveling salesman problems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 34(4):1718–1729.

Tsujimura, Y., and Gen, M. (1998). Entropy-based genetic algorithm for solving TSP. In *Proceedings of the 2nd International Conference on Knowledge-Based Intelligent Electronic Systems*, pp. 285–290.

Vallada, E., and Ruiz, R. (2010). Genetic algorithms with path relinking for the minimum tardiness permutation flowshop problem. *Omega*, 38(1):57–67.

Wang, Y., Lü, Z., and Hao, J. (2010). A study of multi-parent crossover operators in a memetic algorithm. In *Proceedings of the 11th International Conference on Parallel Problem Solving from Nature*, pp. 556–565. Lecture Notes in Computer Science, Vol. 6238.

Wineberg, M., and Oppacher, F. (2003). The underlying similarity of diversity measures used in evolutionary computation. In *Proceedings of the 2003 International Conference on Genetic and Evolutionary Computation*, p. 206.

Yao, X. (1993). An empirical study of genetic operators in genetic algorithms. *Microprocessing and Microprogramming*, 38:707–714.

Zhang, C., Su, S., and Chen, J. (2006). Efficient population diversity handling genetic algorithm for QoS-aware web services selection. In *Proceedings of the 6th International Conference on Computational Science*, pp. 104–111.

Appendix: Efficient Computation of $\Delta H(y)$

An efficient computation of $\Delta H(y)$ in the evaluation function (14) is crucial for the execution of the GA using the proposed high-order entropy measures (H_m, H_m^{adj}, and H_m^{vari}). We present an outline of the efficient computation of $\Delta H(y)$ in the ATSP case, which includes the STSP case as a special case (see Section 3.1).

Remember that $\mathcal{T}$ is a tree for storing the values of $N(s_1, \ldots, s_k)$ $(k \leq m+1)$ (see Figure 2). For an offspring solution y, let E_{re} (resp. E_{ad}) be a set of edges that are removed from (resp. added to) the parent solution p_A to generate this offspring solution. For each edge of E_{re}, p_A has m sequences of length $m+1$ including this edge, which disappear from the population if p_A is replaced with y. Figure 8 illustrates the sequences of length $m+1$ $(=4)$ to remove for E_{re}. In a similar manner, we can determine the sequences of length $m+1$ to add for E_{ad} as illustrated in Figure 8. For each sequence $\{s_1, \ldots, s_m, s_{m+1}\}$ to remove or add, we can know the values of $N(s_1, \ldots, s_k)$ $(h \leq k \leq m+1)$ affected by the replacement by tracing $\mathcal{T}$ according to the sequence, where the value of h depends on the situation. For example, let a sequence of length $m+1(=4)$ to remove be $\{b, g, c, a\}$ where only edge (g, c) is included in E_{re}. In this case, $N(b, g, c)$ and $N(b, g, c, a)$ are decremented (if p_A is replaced with y), whereas $N(b)$ and $N(b, g)$ remain the same, that is, $h = 3$.

The computation of $\Delta H_m(y)$ and $\Delta H_m^{adj}(y)$ are similar; hence, we only describe the method to compute $\Delta H_m(y)$. For every sequence $\{s_1, \ldots, s_{m+1}\}$ to remove and add, the changes to $N(s_1, \ldots, s_m)$ and $N(s_1, \ldots, s_{m+1})$ are accumulated in $\Delta N(s_1, \ldots, s_m)$ and $\Delta N(s_1, \ldots, s_{m+1})$, respectively. We store these values in the tree $\mathcal{T}$. Therefore, if an adding sequence $\{s_1, \ldots, s_k\}$ does not exist in $\mathcal{T}$, we must create a temporal new node of $\mathcal{T}$ with $N(s_1, \ldots, s_k) = 0$ to store the necessary data ($\mathcal{T}$ must be restored after computing $\Delta H_m(y)$).

Let S_m and S_{m+1} be sets of the sequences of length m and $m+1$, respectively, that are removed or added. Let $P'(\boldsymbol{s})$ be defined as $\frac{N(\boldsymbol{s})+\Delta N(\boldsymbol{s})}{N_{sample}}$ for $\boldsymbol{s} \in S_m \cup S_{m+1}$, where $P(\boldsymbol{s})$ is already defined as $\frac{N(\boldsymbol{s})}{N_{sample}}$ (see Section 3.1) . The value of $\Delta H_m(y)$ is then computed by

$$\Delta H_m(y) = -\sum_{\boldsymbol{s} \in S_{m+1}} \Big\{ P'(\boldsymbol{s}) \log P'(\boldsymbol{s}) - P(\boldsymbol{s}) \log P(\boldsymbol{s}) \Big\} + \sum_{\boldsymbol{s} \in S_m} \Big\{ P'(\boldsymbol{s}) \log P'(\boldsymbol{s}) - P(\boldsymbol{s}) \log P(\boldsymbol{s}) \Big\}. \quad (15)$$

The computation of $\Delta H_m^{vari}(y)$ is somewhat complicated. Before describing this, we first describe a method to compute H_m^{vari} (Eq. (13)). Remember that $\mathcal{S}$ is a set of the

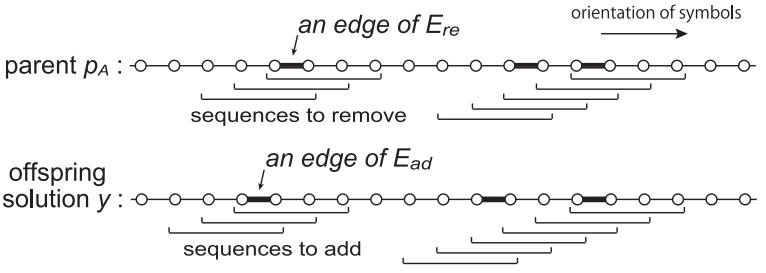

Figure 8: Illustration of sequences of length $m+1$ $(=4)$ to remove (add) for E_{re} (E_{ad}).

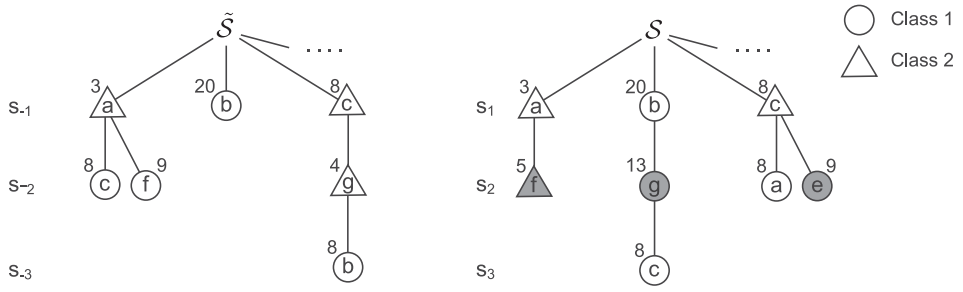

Figure 9: Context tree representation of $\tilde{\mathcal{S}}$ and $\mathcal{S}$. The numbers represent $\tilde{N}_I(\tilde{\boldsymbol{s}_c})$ and $N_I(\boldsymbol{s}_c)$.

sequences of symbols for the conditioning variables of the variable-order Markov process to define H_m^{vari} and that $\tilde{\mathcal{S}} = \{\tilde{\boldsymbol{s}_c} | \boldsymbol{s}_c \in \mathcal{S}\}$ with $\tilde{\boldsymbol{s}_c}$ being the reverse sequence of $\boldsymbol{s}_c$. The context tree $\tilde{\mathcal{S}}$ is then introduced to represent the set $\tilde{\mathcal{S}}$ (see Figure 3), where the elements of the set $\tilde{\mathcal{S}}$ are represented as the leaf nodes of the context tree $\tilde{\mathcal{S}}$. In the actual implementation, each node with a symbol "#" in the context tree $\tilde{\mathcal{S}}$ is removed and is associated with its parent node. Figure 9 (Left) shows a context tree obtained from the context tree $\tilde{\mathcal{S}}$ presented in Figure 3 according to this manner. For example, node a in the modified context tree represent a sequence $a\#$, where "#" means any symbol other than c and f. From this point onward, we refer to the modified context tree simply as the context tree $\tilde{\mathcal{S}}$ unless otherwise stated. We classify the nodes of the context tree $\tilde{\mathcal{S}}$ into two classes as follows (see also Figure 9): (Class 1) leaf nodes and (Class 2) internal nodes. Further, we define a tree $\mathcal{S}$ to represent the set $\mathcal{S}$, where the nodes of $\mathcal{S}$ correspond one-to-one with the nodes of $\tilde{\mathcal{S}}$. The nodes of the tree $\mathcal{S}$ are also classified into the two classes such that each node in $\mathcal{S}$ has the same class as the corresponding node in $\tilde{\mathcal{S}}$. Figure 9 (Right) shows the tree $\mathcal{S}$ that corresponds to the context tree $\tilde{\mathcal{S}}$ in the left side of the figure. For example, node "cgb" in $\tilde{\mathcal{S}}$ (i.e., a node reached by following $\tilde{\mathcal{S}}$ in this order) and node "bgc" in $\mathcal{S}$ both belong to Class 1. Note that only white nodes of $\mathcal{S}$ can be determined from the information of $\tilde{\mathcal{S}}$ presented in the figure because other nodes (gray nodes) depend on the nodes of $\tilde{\mathcal{S}}$ omitted in the figure. In the actual implementation, all data of the trees $\mathcal{S}$ and $\tilde{\mathcal{S}}$ are stored in the trees $\mathcal{T}$ and $\tilde{\mathcal{T}}$, respectively, because $\mathcal{S}$ (reps. $\tilde{\mathcal{S}}$) is a subtree of $\mathcal{T}$ (resp. $\tilde{\mathcal{T}}$).

Let S_1 and S_2 be sets of the nodes of Class 1 and Class 2, respectively, in the tree $\mathcal{S}$. Similarly, $\tilde{S}_1$ and $\tilde{S}_2$ are also defined for $\tilde{\mathcal{S}}$. For each node $\boldsymbol{s}_c \in \mathcal{S}$, we denote the number of the corresponding sequence of symbols existing in the population as $N_I(\boldsymbol{s}_c)$. The values of $N_I(\boldsymbol{s}_c)$ are given by

$$N_I(\boldsymbol{s}_c) = \begin{cases} \tilde{N}(\tilde{\boldsymbol{s}_c}) \ (= N(\boldsymbol{s}_c)) & (\boldsymbol{s}_c \in S_1) \\ \tilde{N}(\tilde{\boldsymbol{s}_c}) - \sum_{\substack{s \in L \\ (\tilde{\boldsymbol{s}_c}, s) \in \tilde{S}_1}} \tilde{N}(\tilde{\boldsymbol{s}_c}, s) & (\boldsymbol{s}_c \in S_2) \end{cases}. \tag{16}$$

We need to pay attention when $\boldsymbol{s}_c \in S_2$. For example, if $\boldsymbol{s}_c = a$ (equivalently $\tilde{\boldsymbol{s}_c} = a$) in the example, $\tilde{\boldsymbol{s}_c}$ represents sequences "$a\#$," but $\tilde{\boldsymbol{s}_c}$ represents "a" when it is an argument of the functions $N(\cdot)$ and $\tilde{N}(\cdot)$ because these functions are defined on the trees $\mathcal{T}$ and $\tilde{\mathcal{T}}$. The values of $N_I(\boldsymbol{s}_c)$ are stored in the tree $\mathcal{S}$.

Further, for $\boldsymbol{s_c} \in \mathcal{S}$ and $s_0 \in L$, we define the following function:

$$N_{I+}(\boldsymbol{s_c}, s_0) = \begin{cases} N(\boldsymbol{s_c}, s_0) & (\boldsymbol{s_c} \in S_1) \\ N(\boldsymbol{s_c}, s_0) - \sum_{\substack{s \in L \\ (s, \boldsymbol{s_c}) \in S_1}} N(s, \boldsymbol{s_c}, s_0) & (\boldsymbol{s_c} \in S_2) \end{cases}. \tag{17}$$

The values of $N_{I+}(\boldsymbol{s_c}, s_0)$ are not stored and are computed as necessary. Note that for $\boldsymbol{s_c} \in S_2$, a set of the nodes $(s, \boldsymbol{s_c}) \in S_1$ is obtained by introducing a function that returns the addresses of the corresponding nodes in $\tilde{\mathcal{S}}$ (resp. $\mathcal{S}$) to the nodes in $\mathcal{S}$ (resp. $\tilde{\mathcal{S}}$) because we can easily access the nodes $(\tilde{\boldsymbol{s_c}}, s) \in \tilde{S}_1$ in the context tree $\tilde{\mathcal{S}}$. We define $P_I(\boldsymbol{s_c}) = \frac{N_I(\boldsymbol{s_c})}{N_{sample}}$ $(\boldsymbol{s_c} \in \mathcal{S})$ and $P_{I+}(\boldsymbol{s_c}, s_0) = \frac{N_{I+}(\boldsymbol{s_c}, s_0)}{N_{sample}}$ $(\boldsymbol{s_c} \in \mathcal{S}, s_0 \in L)$. Then, H_m^{vari} is computed by

$$H_m^{vari} = -\sum_{\boldsymbol{s_c} \in \mathcal{S}} \sum_{s_0 \in L} P_{I+}(\boldsymbol{s_c}, s_0) \log P_{I+}(\boldsymbol{s_c}, s_0) + \sum_{\boldsymbol{s_c} \in \mathcal{S}} P_I(\boldsymbol{s_c}) \log P_I(\boldsymbol{s_c}). \tag{18}$$

Now, $\Delta H_m^{vari}(y)$ is computed by calculating the differences in $N_I(\boldsymbol{s_c})$ and $N_{I+}(\boldsymbol{s_c}, s_0)$ when p_A is replaced with y. We denote the differences in $N_I(\boldsymbol{s_c})$ and $N_{I+}(\boldsymbol{s_c}, s_0)$ as $\Delta N_I(\boldsymbol{s_c})$ and $\Delta N_{I+}(\boldsymbol{s_c}, s_0)$, respectively. Let S_I and S_{I+} be sets of the nodes $\boldsymbol{s_c}$ and $(\boldsymbol{s_c}, s_0)$ that change the values of $N_I(\boldsymbol{s_c})$ and $N_{I+}(\boldsymbol{s_c}, s_0)$, respectively. At the beginning, an initialization procedure is performed as follows: $S_I = \emptyset$, $S_{I+} = \emptyset$, and $\Delta N_I(\boldsymbol{s}) = \Delta N_{I+}(\boldsymbol{s}) = 0$ $(\boldsymbol{s} \in \mathcal{S})$. For every sequence $\{s_1, \ldots, s_{m+1}\}$ to remove or add (if p_A is replaced with y), the following procedure is performed.

1. Trace $\mathcal{S}$ according to the sequence $\{s_1, \ldots, s_{m+1}\}$, and for each sequence $\{s_1, \ldots, s_k\}$ $(h \le k \le m+1)$, perform procedures (2)–(3). The value of h is explained earlier in this appendix.

2. (a) If $\{s_1, \ldots, s_k\}$ is a Class 1 node of $\mathcal{S}$, add this sequence to S_I and increment (decrement) $\Delta N_I(s_1, \ldots, s_k)$ by one if this sequence is added (removed).
   (b) Else if $\{s_2, \ldots, s_k\}$ is a Class 2 node of $\mathcal{S}$, add $\{s_2, \ldots, s_k\}$ to S_I and increment (decrement) $\Delta N_I(s_2, \ldots, s_k)$ by one if $\{s_1, \ldots, s_k\}$ is added (removed).

3. (a) If $\{s_1, \ldots, s_{k-1}\}$ is a Class 1 node of $\mathcal{S}$, add $\{s_1, \ldots, s_k\}$ to S_{I+} and increment (decrement) $\Delta N_{I+}(s_1, \ldots, s_k)$ by one if $\{s_1, \ldots, s_k\}$ is added (removed).
   (b) Else if $\{s_2, \ldots, s_{k-1}\}$ is a Class 2 node of $\mathcal{S}$, add $\{s_2, \ldots, s_k\}$ to S_{I+} and increment (decrement) $\Delta N_{I+}(s_2, \ldots, s_k)$ by one if $\{s_1, \ldots, s_k\}$ is added (removed).

After completing the above procedure, we compute $P_I{}'(\boldsymbol{s_c}) = \frac{N(\boldsymbol{s_c}) + \Delta N_I(\boldsymbol{s_c})}{N_{sample}}$ $(\boldsymbol{s_c} \in S_I)$ and $P_{I+}{}'(\boldsymbol{s}) = \frac{N(\boldsymbol{s}) + \Delta N_{I+}(\boldsymbol{s})}{N_{sample}}$ $(\boldsymbol{s} \in S_{I+})$. The value of $\Delta H_m^{vari}(y)$ is then computed by

$$\begin{aligned} \Delta H_m^{vari}(y) = &- \sum_{\boldsymbol{s} \in S_{I+}} \left\{ P_{I+}{}'(\boldsymbol{s}) \log P_{I+}{}'(\boldsymbol{s}) - P_{I+}(\boldsymbol{s}) \log P_{I+}(\boldsymbol{s}) \right\} \\ &+ \sum_{\boldsymbol{s_c} \in S_I} \left\{ P_I{}'(\boldsymbol{s_c}) \log P_I{}'(\boldsymbol{s_c}) - P_I(\boldsymbol{s_c}) \log P_I(\boldsymbol{s_c}) \right\}. \end{aligned} \tag{19}$$

After the parent solution p_A is replaced with the selected offspring solution, we need to update the trees $\mathcal{S}$ and $\tilde{\mathcal{S}}$, accordingly. This can be done efficiently, but we omit the details because the procedure is complicated to explain. In the actual implementation, the values of $N_I(\boldsymbol{s_c})$ were updated immediately, but the structure of the trees were reconstructed at the beginning of each generation of the GA (see Section 5).