EPiC Series in Computing

Volume 81, 2022, Pages 418–428

Proceedings of 11th International Congress on Advanced Applied Informatics


# Using Label Information in a Genetic Programming Based Method for Acquiring Tag Tree Patterns with Vertex Labels and Wildcards

Shunsuke Yokoyama[1], Tetsuhiro Miyahara[1]
Yusuke Suzuki[1], Tomoyuki Uchida[1] and Tetsuji Kuboyama[2]

[1] Faculty of Information Sciences, Hiroshima City University, Hiroshima 731-3194, Japan
{b20210@e., miyares21@, y-suzuki@, uchida@}hiroshima-cu.ac.jp
[2] Computer Centre, Gakushuin University, Tokyo 171-8588, Japan
ori-eskm21winter@tk.cc.gakushuin.ac.jp

**Abstract**

Machine learning and data mining from tree structured data are studied intensively. In this paper, as tree structured patterns we use tag tree patterns with vertex and edge labels and wildcards in order to represent label connecting relation of vertices and edges in tree structured data. We propose an evolutionary learning method based on Genetic Programming for acquiring characteristic tag tree patterns with vertex and edge labels and wildcards from positive and negative tree structured data. By using label information, that is, label connecting relation of positive examples, as inappropriate individuals we can exclude tag tree patterns that do not satisfy label connecting relation of positive examples. We report experimental results on our evolutionary learning method and show the effectiveness of using label connecting relation of positive examples.

## 1 Introduction

Machine learning and data mining from tree structured data have much attention. Glycans are said to be the third major class of biomolecules next to DNA and proteins and have tree structures. Genetic Programming (GP) is an evolutionary method that is an extension of Genetic Algorithm (GA) and deals with tree structured objects such as formulas and programs [1, 4]. Obtaining characteristic tree structured patterns such as tag tree patterns, using Genetic Programming, from glycan data are proposed [2, 3, 9, 7]. Tag tree patterns are tree structured patterns with structured variables that can be substituted by tree structured data.

To represent a characteristic tree structured pattern, which explains many positive and few negative tree structured data, we use a *tag tree pattern* (Figure 1), which is a rooted tree pattern with ordered children and structured variables. A variable in a tag tree pattern can be substituted by an arbitrary tree. A tag tree pattern is a whole tree structured pattern that matches a whole structure of an example tree structure, and has rich expressiveness of a structured variable representing any subtree structure.

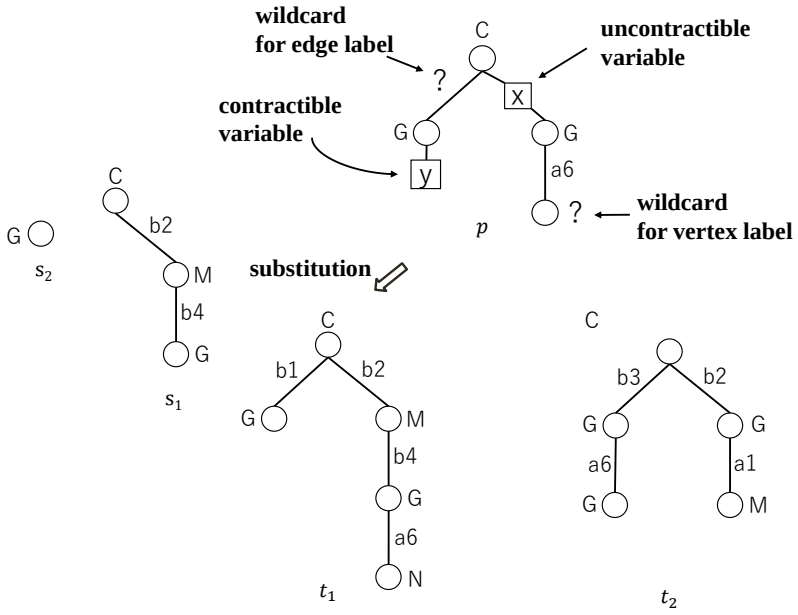



Figure 1: A tag tree pattern $p$ that matches a tree $t_1$ and does not match a tree $t_2$.

In our previous work [7], we use a tag tree pattern with only edge labels and wildcards for edge labels and converted glycan data with only edge labels. In this paper, we extend the representation of a tag tree pattern in our previous work. In order to represent characteristic tree structures of original glycan data with sugar names as vertex labels and bond names as edge labels, we introduce a new type of tag tree pattern, a tag tree pattern with vertex and edge labels and wildcards, which is simply called a *tag tree pattern with wildcards*. By using vertex and edge labels in a tag tree pattern, we can represent naturally label connecting relation of vertices and edges in tree structured data. By using a tag tree pattern with vertex and edge labels and wildcards for vertex and edge labels, we can use a tree structured pattern with rich representation power, since a wildcard for a vertex label or an edge label matches any vertex label or any edge label. In order to obtain highly characteristic tag tree patterns, by using label information of positive examples, that is, label connecting relation of positive tree examples, we propose a learning method for acquiring characteristic tag tree pattern with wildcards using Genetic Programming, from positive and negative tree structured data. The proposed method is considered an extension of GP approach [2, 7] for acquiring characteristic tag tree patterns from positive and negative tree structured data. As the glycan data $g_1$ and $g_2$ in Figure 2 show, the structure of a glycan is abstractly represented as a tree by representing single sugars as vertices and covalent bonds between sugars as edges. A vertex label and an edge label denote the name of a sugar and the name of a bond, respectively. Therefore, GP approach is very suited for learning of the structural features of glycan data with respect to a specific phenomenon in Bioinformatics.

The learning method of the previous work [2, 7] generates tag tree patterns without checking label information of positive examples in GP operations. The method can generate tag tree patterns that match little or no positive examples in a given example set. Therefore such tag tree patterns generated in the previous method are not appropriate for early convergence and

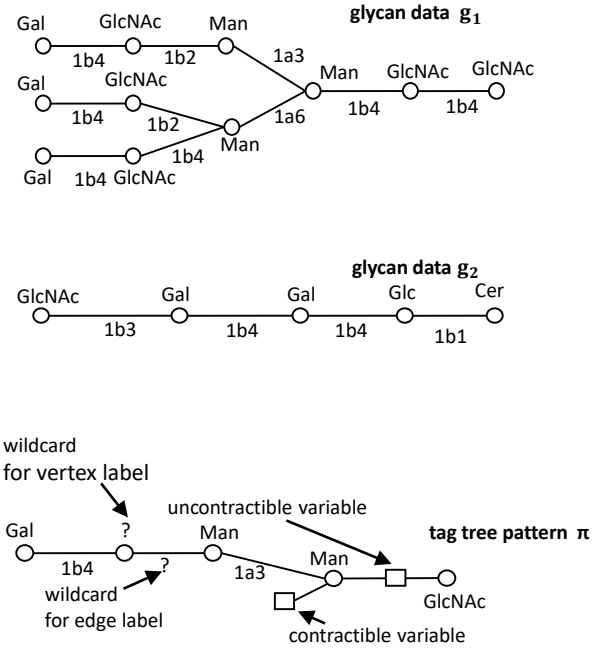



Figure 2: Tree structured glycan data $g_1$ and $g_2$. A tag tree patterns $\pi$ that matches $g_1$ but does not match $g_2$.

high fitness, although those tag tree patterns are useful in maintaining diversity of populations and avoiding local maximum solutions.

In our proposed method in this paper, we generate tag tree patterns with wildcards, by using label information of positive examples. The improved method generates only tag tree patterns that satisfy label information of positive examples and gains highly characteristic tag tree patterns. We report experimental results on our evolutionary learning method using newly introduced tag tree patterns with wildcards, and show the effectiveness of using the newly introduced tag tree patterns and label connecting relation of positive examples. Although the experimental results are on obtaining characteristic tree structured patterns from positive and negative real and synthetic data about glycans, our proposed method for obtaining characteristic tree structured patterns is applicable to classifying positive and negative tree structured data of general type and has a wide range of applicability.

This paper is organized as follows. In Section 2, we introduce tag tree patterns with vertex and edge labels and wildcards as new tree structured patterns. In Section 3, we give our learning problem of extraction of characteristic tag tree patterns, and present our Genetic Programming based learning method for the problem, using label information of positive examples. In Section 4, we report experimental results on some glycan data and synthetic data. In Section 5, we give concluding remarks.

## 2 Tag Tree patterns with Vertex and Edge Labels and Wildcards

In this section, we introduce a tag tree pattern with vertex and edge labels and wildcards, which is a tree structured pattern easy to represent characteristic structures of tree structured data. We first explain a term tree pattern [5, 6], then introduce a tag tree pattern with vertex and edge labels and wildcards, which is an extension of a term tree pattern.

Let $T = (V_T, E_T)$ be a rooted tree with ordered children (simply called a *tree*) that has a set $V_T$ of vertices and a set $E_T$ of edges. Let $E_g$ and $H_g^u$ be a partition of $E_T$, i.e., $E_g \cup H_g^u = E_T$ and $E_g \cap H_g^u = \emptyset$, and let $V_g = V_T$. Let $H_g^c$ be a multiset consisting of vertices in $V_g$. A 4-tuple $g = (V_g, E_g, H_g^u, H_g^c)$ is called a *term tree pattern*, and elements in $V_g$, $E_g$, $H_g^u$ and $H_g^c$ are called a *vertex*, an *edge*, an *uncontractible variable* and a *contractible variable*, respectively. We consider two types of variables, uncontractible variables and contractible variables. An uncontractible variable in a term tree pattern can be replaced with an arbitrary tree consisting of two or more vertices. A contractible variable in a term tree pattern can be replaced with an arbitrary tree consisting of one or more vertices. In figures, an uncontractible variable is represented by a box with two lines, and a contractible variable is represented by a box with one line. A term tree pattern with no variable is considered a tree.

A *substitution* $\theta$ is an operation that identifies the vertices of a variable $x_i$ with the vertices of a substituted tree $g_i$, and replaces the variables $x_i$ with the trees $g_i$, simultaneously. We assume that the parent of a variable is identified with the root of a substituted tree, and the child of a variable is identified with a leaf of a substituted tree. By $t\theta$ we denote the term tree pattern obtained by applying a substitution $\theta$ to a term tree pattern $t$.

Let $\Lambda$ be a set of edge labels and $\Delta$ a set of vertex labels. A *tag tree pattern with vertex and edge labels and wildcards* is an extended term tree pattern with vertex and edge labels such that each edge label on it is a keyword (a word in $\Lambda$) or a special symbol "?"($\notin \Lambda$) and each vertex label on it is a keyword (a word in $\Delta$) or a special symbol "?"($\notin \Delta$). A special symbol "?" represents a wildcard for a vertex or an edge label. A tag tree pattern with vertex and edge labels and wildcards is a tree structured pattern that has edge and vertex labels and is allowed to have wildcards for edge labels and wildcards for vertex labels, which is an extension of a tag tree pattern with only edge labels and wildcards used in the previous work [7]. A tag tree pattern with vertex and edge labels and wildcards is simply called a *tag tree pattern with wildcards*, or a tag tree pattern.

A tag tree pattern with no variable is called a *ground tag tree pattern*. We say that an edge $e$ with an edge label $L$ of a tag tree pattern *matches* an edge $e'$ with an edge label $L'$ of a tree if (1) $L$ and $L'$ are the same keyword, or (2) $L$ is a wildcard "?" and $L'$ is an arbitrary keyword in $\Lambda$. We say that a vertex $v$ with a vertex label $L$ of a tag tree pattern *matches* a vertex $v'$ with a vertex label $L'$ of a tree if (1) $L$ and $L'$ are the same keyword, or (2) $L$ is a wildcard "?" and $L'$ is an arbitrary keyword in $\Delta$.

A ground tag tree pattern $\pi$ *matches* a tree $T$ if (1) $\pi$ and $T$ are isomorphic ignoring vertex and edge labels, and (2) every edge of $\pi$ matches its corresponding edge of $T$, and (3) every vertex of $\pi$ matches its corresponding vertex of $T$. A tag tree pattern $\pi$ *matches* a tree $T$ if there exists a substitution $\theta$ such that $\pi\theta$ is a ground tag tree pattern and $\pi\theta$ matches $T$.

A matching relation of a tag tree pattern $\pi$ and a tree $T$ can be also explained by a replacement that replaces variables in $\pi$ with appropriate trees and replaces wildcards in $\pi$ with appropriate labels such that $T$ is obtained from $\pi$ by the replacement. For example, in Figure 1, from a tag tree pattern $p$ we obtain a tree $t_1$ by applying a replacement that replaces the variable having a label $x$ in $p$ with a tree $s_1$ and replaces the variable having a label $y$ in $p$

with a tree $s_2$ and replaces wildcard "?" for edge label with a label "b1" and replaces wildcard "?" for vertex label with a label "N". Then the tag tree pattern $p$ matches the tree $t_1$. But no such replacement exists for a tag tree pattern $p$ and a tree $t_2$. Then the tag tree pattern $p$ does not match the tree $t_2$. For example, in Figure 2, consider tree structured glycan data $g_1$ and $g_2$, and a tag tree pattern $\pi$. The tag tree pattern $\pi$ matches $g_1$ but does not match $g_2$.

# 3 Acquisition of Characteristic Tag Tree Patterns with Vertex and Edge Labels and Wildcards

In this section, we give our learning problem of acquiring characteristic tag tree patterns from positive and negative tree structured data. Then, we present our Genetic Programming based learning method for the problem, using label information of positive examples.

We first describe Problem of Acquiring Characteristic Tag Tree Patterns. Let $D$ be a finite set of positive and negative tree structured data. The *fitness* of a tag tree pattern $\pi$ w.r.t. $D$, denoted by $fitness_D(\pi)$, is defined as the balanced accuracy of $\pi$ w.r.t. $D$, that is, $fitness_D(\pi)$ = (the ratio of positive examples in $D$ that $\pi$ matches + the ratio of negative examples in $D$ that $\pi$ does not match) / 2. A tag tree pattern as an individual is a binary classifier of tree structured data. So a tag tree pattern of high fitness is considered a characteristic tag tree pattern, which matches many positive and few negative tree structured data. In this paper we consider the following learning problem [2].

**Problem of Acquiring Characteristic Tag Tree Patterns**
**Input:** A finite set $D$ of positive and negative tree structured data.
**Problem:** Find a tag tree pattern of high fitness w.r.t. $D$.

The learning method of the previous work [7] generates tag tree patterns without checking label information of positive examples in GP operations. The method can generate tag tree patterns that match few or no positive examples in a given example set $D$. Then, in this paper, using label connecting relation of positive examples we give an improved method for acquiring characteristic tag tree patterns. The improved method generates only tag tree patterns that satisfy label connecting relation of positive examples and gains highly characteristic tag tree patterns.

For a vertex $v$ of a positive tree example, the set of pairs of the label $L_v$ of the vertex $v$ and the multiset of the edge labels $L_1, \cdots, L_n$ of the edges $e_1, \cdots, e_n$ adjacent to the vertex $v$ is called *the label connecting relation* of the vertex $v$. We sum up the number of occurrences of the label connecting relations of all vertices of all positive examples and make *the label connecting relation* of the positive tree examples. For example, Figure 3 shows a set of positive examples of tree structured data and the label connecting relation of the set of positive examples.

As label information of positive examples, we use the label connecting relation of the positive examples. We apply the previous evolutionary learning method [8] using label information of positive examples for bpo-graph patterns to tag tree patterns. We give a method for the Problem of Acquisition of Characteristic Tag Tree Patterns, by using Genetic Programming, tag tree patterns with vertex and edge labels and wildcards as new representations, the label connecting relation including the number of occurrences of labels of positive examples, as follows.

**The GP method for the Problem of Acquisition of Characteristic Tag Tree Patterns with Wildcards, using label information of positive examples**

1. Let $s$ be the population size given by a user. Let $Pos$ be the set of all positive examples in $D$. Calculate the label connecting relation of $Pos$.

2. Generate the initial population $P = \{p_1, \ldots, p_s\}$, where $p_i$ is a tag tree pattern with wildcards, by using the label connecting relation of $Pos$.

3. Evaluate the fitness $fitness_D(p_i)$ of each tag tree pattern with wildcards $p_i (i = 1, \ldots, s)$ in $P$.

4. Generate the population $P' = \{p'_1, \ldots, p'_s\}$ of the next generation by repeating the following procedure: Perform genetic operations (crossover, mutation and reproduction) on tag tree patterns with wildcards selected according to their fitness, by using the label connecting relation of $Pos$, and generate new tag tree patterns with wildcards.

5. Evaluate the fitness $fitness_D(p'_i)$ of each tag tree pattern with wildcards $p'_i (i = 1, \ldots, s)$ in $P'$.

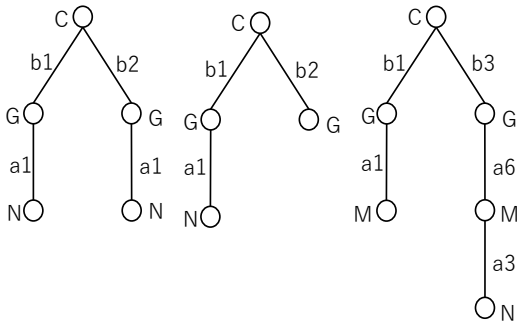
6. If the generation reaches the maximum number of generations (the only termination condition), then terminate the whole process. Otherwise, set $P = P'$ and the process returns to step 4.

In order to generate individuals (tag tree patterns with wildcards) that satisfy the label connecting relation of positive examples in our GP method, we newly define a label connecting relation for a tag tree pattern with wildcards as follows. We say that a vertex $v$ with label $\ell$ in a tag tree pattern with wildcards $p$ *satisfies* the label connecting relation $CR_{Pos}$ of a set $Pos$ of positive examples, if the pair of $\ell$ and a multiset of edge labels connecting a vertex labeled with $\ell$ is included in $CR_{Pos}$. If $v$ is labeled with a wildcard, then we check the relation after we replace the wildcard with any vertex label of a vertex of a positive example. If $v$ connects an edge labeled with a wildcard, then we check the relation after we replace the wildcard with any edge label of an edge of a positive example. If $v$ connects a variable, then we check the relation after we replace the variable with an appropriate tree. We say that a tag tree pattern with wildcards $p$ *satisfies* a label connecting relation of positive examples if all vertices of $p$ satisfy the label connecting relation of positive examples.

We give the GP operators that are applied to tag tree patterns with wildcards, in our method. We use crossover operator and five mutation operators (change-subpattern, add-subpattern, del-subpattern, change-vertex-label and change-edge-label). By using these GP operators we generate a tentative individual. If the tentative individual satisfies the label connecting relation at a vertex where GP operator is applied, then we adopt the tentative individual as an individual in a next generation. Otherwise we go back one step to a selection point in our GP method. Figures 4 and 5 show the case where the label connecting relation is satisfied and the case where the label connecting relation is not satisfied, respectively, after applying crossover operator. We confirm whether the vertices specified by arrows satisfy the label connecting relation, since edges incident on those vertices are changed after crossover operation. Figure 4 shows the resulting individuals $p'_1$ and $p'_2$ obtained by applying crossover operator, after confirming that the label connecting relation of positive examples in Figure 3 is satisfied.

# 4 Experimental Results

We have implemented our evolutionary method proposed in Section 3 for acquiring characteristic tag tree patterns with wildcards, from positive and negative tree structured data and have performed experiments on some glycan data. The implementation is in Java and all experiments




| vertex label | multiset of edge labels | the number of occurrences |
|---|---|---|
| C | {b1,b2} | 2 |
| | {b1,b3} | 1 |
| G | {b1,a1} | 3 |
| | {b2,a1} | 1 |
| | {b2} | 1 |
| | {b3,a6} | 1 |
| N | {a1} | 3 |
| | {a3} | 1 |
| M | {a1} | 1 |
| | {a6,a3} | 1 |

Figure 3: Set of positive examples of tree structured data and label connecting relation of positive examples.

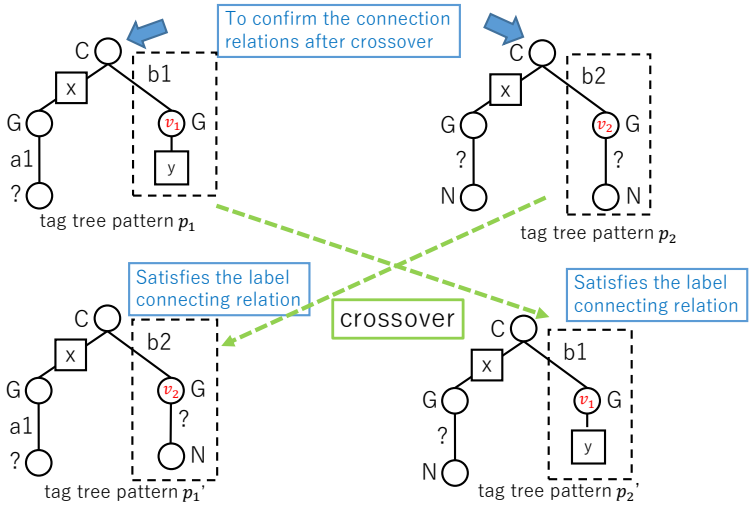



Figure 4: The case where the label connecting relation is satisfied, after applying crossover operator.

are performed on 4.0 GHz Intel Core i7 processor, macOS Mojave 10.14.6 OS. Table 1 shows the parameters of our GP setting.

We performed experiments of acquiring characteristic tag tree patterns with wildcards by our proposed method, using as inputs real data and 4 kinds of synthetic data (Tables 2 and 3). The real data are glycan data from KEGG GLYCAN database and consists of 177 positive examples and 302 negative examples. We make synthetic data as follows. We set a generation tag tree pattern, and randomly generate tree structured data, then we set tree structured data that match the generation pattern as positive examples, and set tree structured data that do

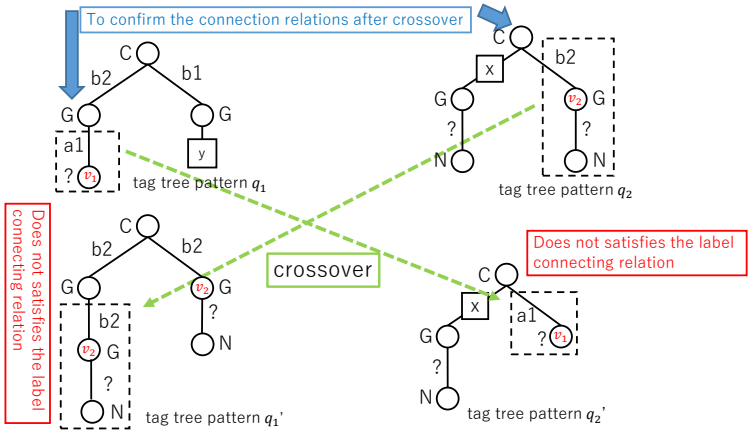



Figure 5: The case where the label connecting relation is not satisfied, after applying crossover operator.

not match the generation pattern as negative examples. For each generation pattern, we have 500 positive examples and 500 negative examples. Table 3 shows that we use the following 4 generation tag tree patterns:
the generation pattern(1) having no wildcards for vertex labels and no wildcards for edge labels,
the generation pattern(2) having wildcards for vertex labels and wildcards for edge labels,
the generation pattern(3) having wildcards for vertex labels and no wildcards for edge labels,
the generation pattern(4) having no wildcards for vertex labels and wildcards for edge labels.

Tables 2 and 3 show that for each set of experimental data, in four experimental settings we performed evolutionary learning methods for obtaining tag tree patterns as follows:
the setting (1) "usage of wildcards by proposed method: No, and usage of label information by proposed method: No" means that the method obtains tag tree patterns having no wildcards and does not use label information of positive examples,
the setting (2) "usage of wildcards by proposed method: No, and usage of label information by proposed method: Yes" means that the method obtains tag tree patterns having no wildcards and uses label information of positive examples,
the setting (3) "usage of wildcards by proposed method: Yes, and usage of label information by proposed method: No" means that the method obtains tag tree patterns allowed to have wildcards and does not use label information of positive examples,
the setting (4) "usage of wildcards by proposed method: Yes, and usage of label information by proposed method: Yes" means that the method obtains tag tree patterns allowed to have wildcards and uses label information of positive examples.

For each set of experimental data and each experimental setting, we performed 50 GP runs and calculated average fitness of the best individuals (the individuals with highest fitness in the final generation) in 50 GP runs. Table 2 shows the average fitness of the best individuals, for the real data and 4 settings. Table 3 shows the average fitness of the best individuals, for the 4 synthetic data and 4 settings. Figure 6 shows an example of the best individual by the proposed method using wildcards and label information of positive examples from real data.

Experimental results show that tag tree pattern with wildcards and usage of label information of positive examples by the proposed method are effective in obtaining characteristic

Table 1: GP parameters in this experiment.

| | |
|---|---|
| Population size | 50 |
| Reproduction probability | 0.05 |
| Crossover probability | 0.50 |
| Mutation probability | 0.45 |
| | (change-subpattern:0.30) |
| | (add-subpattern:0.30) |
| | (del-subpattern:0.30) |
| | (change-vertex-label:0.05) |
| | (change-edge-label:0.05) |
| Selection method | Roulette wheel selection |
| | tournament size 2 |
| | elite size 3 |
| Maximum number of generations | 200 |

tag tree patterns that are individuals with high fitness. Also experimental results show that introducing vertex labels in a tag tree pattern is effective in obtaining characteristic tag tree patterns that are individuals with high fitness, by using label connecting information of positive tree examples.

Table 2: Average fitness of the best individuals for real data by 4 settings of proposed method.

| usage of wildcards by proposed method | usage of label information by proposed method | average fitness of best individual |
|---|---|---|
| No | No | 0.784 |
| | Yes | 0.833 |
| Yes | No | 0.772 |
| | Yes | 0.830 |

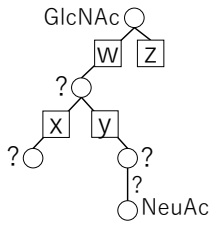



Figure 6: The best individual by the proposed method using wildcards and label information of positive examples from real data.

Table 3: Average fitness of the best individuals for 4 synthetic data by 4 settings of proposed method.

| types of input data | usage of wildcards by proposed method | usage of label information by proposed method | average fitness of best individual |
|---|---|---|---|
| synthetic data(1) | No | No | 0.887 |
| generation pattern(1) having | | Yes | 0.957 |
| no wildcards for vertex labels | Yes | No | 0.875 |
| no wildcards for edge labels | | Yes | 0.953 |
| synthetic data(2) | No | No | 0.849 |
| generation pattern(2) having | | Yes | 0.854 |
| wildcards for vertex labels | Yes | No | 0.898 |
| wildcards for edge labels | | Yes | 0.898 |
| synthetic data(3) | No | No | 0.865 |
| generation pattern(3) having | | Yes | 0.873 |
| wildcards for vertex labels | Yes | No | 0.896 |
| no wildcards for edge labels | | Yes | 0.925 |
| synthetic data(4) | No | No | 0.901 |
| generation pattern(4) having | | Yes | 0.936 |
| no wildcards for vertex labels | Yes | No | 0.888 |
| wildcards for edge labels | | Yes | 0.951 |

# 5 Concluding Remarks

In this paper we have proposed an evolutionary method for acquiring characteristic tag tree patterns with vertex and edge labels and wildcards from positive and negative tree structured data, by using Genetic Programming and using label information of positive examples. By using label information of positive examples, as inappropriate individuals we can exclude tag tree patterns that do not satisfy label connecting relation of positive examples. We have reported experimental results on our evolutionary learning method using newly introduced tag tree patterns with wildcards, and have shown the effectiveness of using the newly introduced tag tree patterns and label connecting relation of positive examples. Although the experimental results are on obtaining characteristic tree structured patterns from positive and negative real and synthetic data about glycans, our proposed method for obtaining characteristic tree structured patterns is applicable to classifying positive and negative tree structured data of general type. As future work, we consider a method that can obtain characteristic tag tree patterns with higher fitness, for example, a method for obtaining characteristic multiple tag tree patterns.

Acknowledgments. We would like to thank the anonymous referees for their helpful comments and detailed suggestions. This work is partly supported by Support Grant for Acquiring Grants-in-Aid for Scientific Research from Hiroshima City University and Grant-in-Aid for Scientific Research (C) (Grant Numbers JP19K12103 and JP21K12021) from Japan Society for the Promotion of Science.

# References

[1] J. R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.

[2] M. Nagamine, T. Miyahara, T. Kuboyama, H. Ueda, and K. Takahashi. A genetic programming approach to extraction of glycan motifs using tree structured patterns. *Proc. AI-2007, Springer-Verlag, Vol. 4830*, pages 150–159, 2007.

[3] M. Nagamine, T. Miyahara, T. Kuboyama, H. Ueda, and K. Takahashi. Evolution of multiple tree structured patterns from tree-structured data using clustering. *Proc. AI-2008, Springer-Verlag, Vol. 5360*, pages 500–511, 2008.

[4] R. Poli, W. Langdon, and N. McPhee. *A Field Guide to Genetic Programming*. Lulu Press, 2008.

[5] Y. Suzuki, T. Shoudai, S. Matsumoto, T. Uchida, and T. Miyahara. Efficient learning of ordered and unordered tree patterns with contractible variables. *Proc. ALT-2003, Springer-Verlag, LNAI 2842*, pages 114–128, 2003.

[6] Y. Suzuki, T. Shoudai, T. Uchida, and T. Miyahara. An efficient pattern matching algorithm for ordered term tree patterns. *IEICE Trans. Inf. Syst.*, E98-A(6):1197–1211, 2015.

[7] S. Tani, T. Miyahara, Y. Suzuki, and T. Uchida. Acquisition of multiple tree structured patterns by an evolutionary method using sets of tag tree patterns as individuals. *Proc. IIAI AAI 2015*, pages 213–218, 2015.

[8] F. Tokuhara, S. Okinaga, T. Miyahara, Y. Suzuki, T. Kuboyama, and T. Uchida. Using label information in a genetic programming based method for acquiring block preserving outerplanar graph patterns with wildcards. *Proc. IWCIA 2019*, pages 95–100, 2019.

[9] K. Yoshida, T. Miyahara, and T. Kuboyama. Evolution of multiple tree structured patterns using soft clustering. *Proc. ICCAE 2010*, pages 749–753, 2010.