アルゴリズム 88－5
（2003．1．20）

酵素反応式からの構造変換規則の抽出アルゴリズム

阿久津 達也

酵素反応式などの化学反応式における入出力関係から、どの部分構造がどの部分構造に移ったかを推定する問題は、薬剤設計、トレーサー実験のシミュレーション、代謝パスウェイデータベースのデータチェックなどに応用できる可能性がある。従来、この問題に対しては最大共通部分グラフ検出アルゴリズムによる手法が用いられてきたが、本研究ではグラフ分割とグラフ同型性判定に基づく新しい方法を提案する。本稿では、この問題が一般には NP 困難であるが、多くの実用規模の問題のクラスに対し多項式時間アルゴリズムがあることを示す。また、理論的正当性はないが実用的なアルゴリズムを、計算機実験による結果とともに示す。

Efficient Extraction of Mapping Rules of Atoms from Enzymatic Reaction Data

Tatsuya Akutsu [1]

Extraction of mapping rules of atoms from enzymatic reactions is useful for drug design, simulation of tracer experiments and consistency checking of pathway databases. Most of previous methods for this problem are based on maximal common subgraph algorithms. In this article, we propose a novel approach based on graph partition and graph isomorphism. We show that this problem is NP-hard in general, but can be solved in polynomial time for wide classes of enzymatic reactions. We also present practical polynomial time algorithms. The results of computational experiments suggest that practical algorithms are useful in many cases.

1 Introduction

Knowledge discovery from biological pathway databases is becoming an important topic in Bioinformatics because understanding of chemical/metabolic mechanisms in living organisms is important for understanding of functions of genes. Several databases on biological pathways are being developed such as KEGG/LIGAND [9] and BioCyc/EcoCyc [7], and a number of enzymatic reaction data are stored in these databases. But, limited facilities are provided for analysis of enzymatic reactions and biological pathways in KEGG and BioCyc. Therefore, more useful tools for analysis of enzymatic reactions and biological pathways should be developed.

Various kinds of problems can be considered for analysis of enzymatic reactions and biological pathways. Among them, *similar compound search* is well studied [12]. Most of these studies are based on *substructure search* or *maximal/maximum common substructure search*. Since these problems are NP-hard in general, most of the proposed algorithms are heuristic.

In this article, we do not consider substructure search or maximal common substructure search. Instead, we focus on the following problem [4]: given an enzymatic reaction, identify which atom in an input compound is transfered to which atom in an output compound. We call this problem as *extraction of enzymatic mapping rules*. Extraction of enzymatic mapping rules

[1] 京都大学 化学研究所 バイオインフォマティクスセンター
Bioinformatics Center, Institute for Chemical Research, Kyoto University,
Gokasho, Uji, Kyoto-Fu 611-0011, Japan. takutsu@kuicr.kyoto-u.ac.jp

has various applications such as drug design, simulation of *tracer experiments*, and consistency checking of pathway databases [4].

Most of previous studies on extraction of enzymatic mapping rules employ a maximal (or maximum) common subgraph based approach [4, 12]. But, such an approach has the following drawbacks.

- There is no guarantee that the correct mappings are always found. As shown by Arita [4], the maximum common subgraph approach sometimes fails to find correct mappings.
- The maximum common subgraph problem is NP-hard even for planar graphs of bounded degree 3. Approximation is also known to be very hard [6].
- It is difficult to find mapping rules consistent with multiple reaction formulas.

Therefore, we propose another approach based on *graph partition* and *unique naming algorithms* ($\approx$ graph isomorphism algorithms) [8, 11], and develop theoretical algorithms and practical algorithms. Our approach has the following advantages.

- Theoretical algorithms, which use elaborated graph isomorphism algorithms [8, 10], work in polynomial time for reasonably wide class of enzymatic reactions.
- Practical algorithms, which use a practical graph isomorphism algorithm [11], are easy to implement and are fast enough.
- The proposed algorithms can be modified for coping with multiple enzymatic reactions.
- The proposed algorithms can be modified for outputting all possible mapping rules consistent with given reactions (see Fig. 1). Therefore, the algorithms do not miss correct mappings (under chemically reasonable assumptions).

Though the algorithms in this article have some drawbacks (e.g., reaction rules that modify ring structures can not be handled), these drawbacks are not crucial and shall be overcome.

2 Basic Framework

We first define a chemical graph. A *chemical graph* is an undirected connected graph of bounded degree D (usually $D \leq 6$), where each node v has a label $atom(v)$ and each edge e may have a label $bond(e)$ (it is sometimes better to ignore $bond(e)$). In this article, we do not consider multiedges and thus information about chemical bonds should be stored in $bond(e)$ if it is required. Since chemical graphs are of bounded degree, the number of edges is $O(n)$.

A *chemical reaction formula* has the form of $I_1 + I_2 + \ldots + I_p \longleftrightarrow O_1 + O_2 + \ldots + O_q$, where $I_1, \ldots, I_p$ and $O_1, \ldots, O_q$ are chemical graphs and the multiset of atoms in $I_1, \ldots, I_p$ must be equal to the multiset of atoms in $O_1, \ldots, O_q$ (i.e., the *law of conservation* must be satisfied). For convenience of explanation, we call $I_1, \ldots, I_p$ *input compounds* and $O_1, \ldots, O_q$ *output compounds*. We define a *chemical cut* of size C as a partition of a graph G into connected components obtained by removing at most C edges, where the following condition must be satisfied. Let $\tilde{V}$ be the multiset of the connected components. There is an edge $\{c_i, c_j\} \in \tilde{E}$ iff. there is a removed edge between c_i and c_j. Then, a graph $\tilde{G}(\tilde{V}, \tilde{E})$ must be a star. We use $\hat{G}$ to denote $\tilde{V}$.

Problem 1. (see Fig. 1)
Given a chemical reaction formula $I_1 + \ldots + I_p \longleftrightarrow O_1 + \ldots + O_q$ and an integer C, find a chemical cut of size C for each of $I_1, \ldots, I_p$ and $O_1, \ldots, O_q$ such that the multiset of connected components $\hat{I}_1 \cup \ldots \cup \hat{I}_p$ is equal to the multiset of connected components $\hat{O}_1 \cup \ldots \cup \hat{O}_q$, where we identify isomorphic components as the same element (i.e., $x = y$ if $x \in \hat{I}_1 \cup \ldots \cup \hat{I}_p$ and $y \in \hat{O}_1 \cup \ldots \cup \hat{O}_q$ are isomorphic).

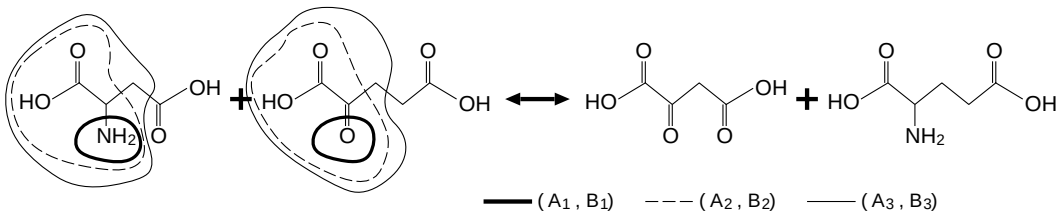

Figure 1: Example of a reaction catalyzed by Transaminase, where this reaction has the form of X-A + Y-B $\longleftrightarrow$ X-B + Y-A ('-' corresponds to a cut). In this case, there are three possible pairs for (A, B), where (A_1, B_1) is the most plausible. It should be noted that hydrogen atoms and types of bonds are ignored in this case. Algorithm 3c in Section 5 outputs all three pairs.

In Problem 1, which edges are removed is not taken into account. But, in some cases, it is better to mind which edges are removed. Thus, we consider a variant (**Problem 1A**) of Problem 1 by modifying the *cut operation* as follows: we replace an edge $e = \{v_i, v_j\}$ by two edges $e' = \{v_i, v'\}$ and $e'' = \{v_j, v''\}$ whenever we remove an edge e, where v' and v'' are newly created vertices (per edge) and all of v' and v'' have a special label showing that these are newly created vertices (i.e., $atom(v') = atom(v'') = new$).

Problem 1 can be modified for the case that multiple reaction formulas are given, where we assume that the *reaction center* is preserved. We use $\tilde{G}^c$ to denote the center of the star $\tilde{G}$ (if it is ambiguous, either one can be the center), where the center corresponds to the reaction center.

Problem 2.
Given K chemical reaction formulas $I_{i,1}+I_{i,2}+\ldots+I_{i,p} \longleftrightarrow O_{i,1}+O_{i,2}+\ldots+O_{i,q}$ $(i = 1, \ldots, K)$ and an integer C, find a chemical cut of size C for each of $I_{i,1}, \ldots, I_{i,p}$ and $O_{i,1}, \ldots, O_{i,q}$ such that (i) Each reaction formula satisfies the condition of Problem 1, (ii) For all $i \neq j$ and for all h, $\tilde{I}^c_{i,h}$ (resp. $\tilde{O}^c_{i,h}$) is isomorphic to $\tilde{I}^c_{j,h}$ (resp. $\tilde{O}^c_{j,h}$).

Problem 2 can be modified for the case of maximizing the number of chemical formulas satisfying the conditions of (i)-(ii). This case is denoted by **Problem 3**. **Problem 2A** and **Problem 3A** are defined in the same way as in Problem 1A.

Proposition 1. Problem 1 and Problem 1A can be solved in polynomial time if C, p, q are constants.
(Proof) For all combinations $(\hat{I}_1, \ldots, \hat{I}_p, \hat{O}_1, \ldots, \hat{O}_q)$ obtained by chemical cuts of size C, we examine whether $\hat{I}_1 \cup \ldots \cup \hat{I}_p$ is equal to $\hat{O}_1 \cup \ldots \cup \hat{O}_q$, where we identify isomorphic graphs as the same element. Since D is a constant and isomorphism of graphs of bounded degree can be tested in polynomial time [8, 10], comparison of two multisets can be done in polynomial time. Since the number of combinations is $O((n^C)^{(p+q)})$, the algorithm works in polynomial time. □

Proposition 2. Problem 2, Problem 2A, Problem 3 and Problem 3A can be solved in polynomial time if C, p, q are constants.

3 Theoretical Results

3.1 NP-hardness Results

The algorithms in Section 2 use exhaustive search procedures. Moreover, all algorithms presented in this article are more or less based on exhaustive search. It is reasonable to ask whether or not exhaustive search is inevitable. The following theorems suggest that the answer is YES.

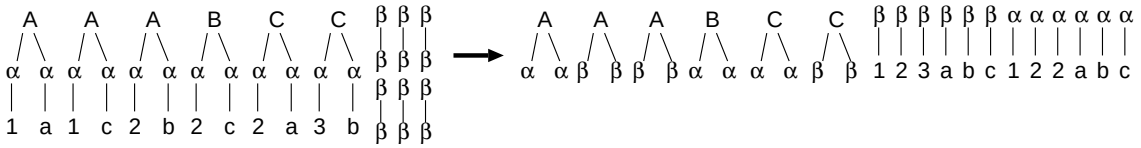

Figure 2: Reduction from 3DM to Problem 1(A), where $X = \{A, B, C\}$, $Y = \{1, 2, 3\}$, $Z = \{a, b, c\}$ and $M = \{\{A, 1, a\}, \{A, 1, c\}, \{A, 2, b\}, \{B, 2, c\}, \{C, 2, a\}, \{C, 3, b\}\}$.

However, NP-hardness results do not imply that efficient or practical algorithms can not be developed because p, q, C are usually very small.

Theorem 1. Problem 1 and Problem 1A are NP-complete for fixed C.
(Proof) We show a polynomial time reduction from 3DM (3-dimensional matching) for the case of $C = 2$. The other parts can be proved easily. Let X, Y, Z be mutually disjoint sets of size m. Let $M = \{\{x_{i_t}, y_{j_t}, z_{k_t}\} \mid t = 1, \ldots, n\}$ be an instance of 3DM, where x_{i_t}, y_{j_t}, z_{k_t} are elements of X, Y, Z, respectively. Recall that 3DM is to find a subset $M' \subseteq M$ of size m such that $\bigcup_{s \in M'} s = X \cup Y \cup Z$.

Input compounds are constructed in the following way (see Fig. 2). Though we assume that we can use $O(n)$ labels here, this assumption can be removed. In the following, α and β denote nodes with labels α and β, respectively. From each element $\{x_{i_t}, y_{j_t}, z_{k_t}\}$ of M, we construct an undirected tree defined by the edge set $\{\{x_{i_t}, \alpha\}, \{x_{i_t}, \alpha'\}, \{\alpha, y_{j_t}\}, \{\alpha', z_{k_t}\}\}$, where α' is a node having label α (we use two symbols α and α' for denoting two distinct nodes with label α). We also construct n trees, each of which is defined by the edge set $\{\{\beta, \beta\}\}$.

Output compounds are constructed in the following way (see Fig. 2). For each element $x_i \in X$ $(i = 1, \ldots, m)$, we construct a tree defined by the edge set $\{\{x_i, \alpha\}, \{x_i, \alpha'\}\}$. Let $\#x$ be the number of appearance of an element x in M. For each element $x_i \in X$, we also construct $\#x_i - 1$ identical trees, each of which is defined by the edge set $\{\{x_i, \beta\}, \{x_i, \beta'\}\}$. For each element $w \in Y \cup Z$, we construct a tree defined by the edge set $\{\{\beta, w\}\}$. For each element $w \in Y \cup Z$, we construct $\#w - 1$ identical trees, each of which is defined by the edge set $\{\{\alpha, w\}\}$.

It is straight-forward to see the correctness of the reduction. □

Theorem 2. Problem 1 and Problem 1A are NP-complete even for $p = q = 2$.

3.2 Efficient Algorithms for a Special Case

In this subsection, we present efficient algorithms for a special case of Problem 1A and Problem 2A in which $p = q = 2$, $C = 1$ and all compounds have tree structures. Though it is a special case, *the techniques introduced in this subsection can be applied to other cases.*

First we consider Problem 1A. Since we consider the case of $p = q = 2$ and $C = 1$, we focus on reactions of the form: X-A + Y-B $\longleftrightarrow$ X-B + Y-A, that A, B, X, Y are rooted trees. We ignore the case that some of A, B, X, Y are empty for the sake of simplicity.

In order to identify X, Y, A, B from a given reaction, we use a unique naming function [11] ($\approx$ a normal form [8]). Suppose that a set of (directed or undirected) graphs $\mathcal{G} = \{G_1, \ldots, G_t\}$ is given. Then, a function $id(G_i)$ from $\mathcal{G}$ to $[1, 2, \ldots, \gamma t]$ (γ is a constant) is called a *unique naming function* if $id(G_i) = id(G_j)$ holds iff. G_i is isomorphic to G_j. We identify the graph with its unique name (i.e., isomorphic graphs are identified as the same object). But, we distinguish X, Y, A, B (i.e., these are not identified even if these are isomorphic). Assuming a unique naming function, we have Algorithm 1.

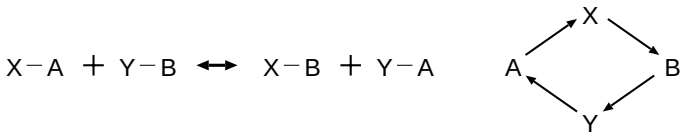

Figure 3: Relationship between a reaction and a directed cycle of length 4.

Algorithm 1

1. For all partitions (X_i, A_i) of I_1, compute unique names of X_i and A_i.
2. For all partitions (Y_j, B_j) of I_2, compute unique names of Y_j and B_j.
3. For all partitions (X_k, B_k) of O_1, compute unique names of X_k and B_k.
4. For all partitions (Y_h, A_h) of O_2, compute unique names of Y_h and A_h.
5. Construct a directed graph $G(V, E)$ defined by the vertex set $V = \{X_i, A_i, Y_j, B_j, X_k, B_k, Y_h, A_h | i = 1, \ldots, |I_1| - 1, \ \ldots \ , h = 1, \ldots, |O_2| - 1\}$ and the edge set $E = \{(A_i, X_i), (X_k, B_k), (B_j, Y_j), (Y_h, A_h)\}$.
6. Examine whether or not there exists a directed cycle of length 4 in $G(V, E)$.

Theorem 3. Algorithm 1 correctly finds chemical cuts of size 1 in $O(n^2)$ time.
(Proof) Since the correctness is almost trivial (see Fig. 3), we prove that it works in $O(n^2)$ time.

First note that the number of partitions is $O(n)$ and thus the number of rooted subtrees appearing in the algorithm is $O(n)$. For each rooted subtree, we need to compute a unique name. A linear time algorithm is known for testing isomorphism of two trees [1]. Slightly modifying the algorithm, we can obtain a unique naming function for all rooted subtrees appearing in Algorithm 1. The total time required to compute unique names for all subtrees is $O(n)$. Therefore, the time required for Steps 1-4 is $O(n)$.

Since the number of partitions and the number of subtrees are $O(n)$, the size of $G(V, E)$ is $O(n)$. Thus, the time required for Step 5 is $O(n)$.

In order to find a cycle of length 4, we employ DFS (depth first search) from each vertex A_i. Since DFS takes $O(n)$ time per A_i and there are $O(n)$ vertices, Step 6 takes $O(n^2)$ time. □

Finding given length cycles is a well-studied graph theoretic problem. Alon *et al.* developed an $O(n^{1.5})$ time algorithm for finding a directed cycle of length 4 in a directed graph of size n [3].

Corollary 1. Using an efficient cycle finding algorithm [3], Algorithm 1 correctly finds chemical cuts of size 1 in $O(n^{1.5})$ time.

Algorithm 1A can be modified for enumerating all pairs of (A, B). But, $O(n^2)$ time would be required in such a case.

Next, we consider the case of $K = 2$ of Problem 2A. We should identify a pair of rooted subtrees (A, B) from two reactions: X-$A + Y$-$B \leftrightarrow X$-$B + Y$-A, X'-$A + Y'$-$B \leftrightarrow X'$-$B + Y'$-A. We construct a directed graph as in Algorithm 1 from both reactions, where X, Y, X', Y' must have mutually distinct unique names. Then, we find a subgraph isomorphic to the graph H shown in Fig. 4. Algorithm 2 finds such a subgraph in $o(n^2)$ time, where a vertex is called a *high degree vertex* if its degree is at least Δ. Algorithm 2 is based on the cycle finding algorithm developed by Alon *et al.* [3] though a new idea is introduced here.

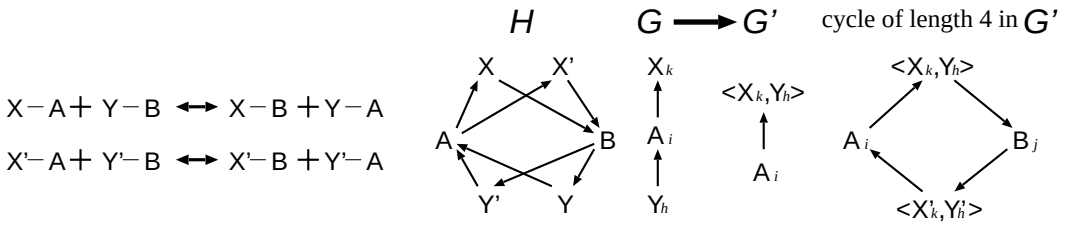

Figure 4: Explanation of Algorithm 2.

Algorithm 2

1. Construct a directed graph $G(V, E)$ using a similar procedure as in Algorithm 1.
2. For each high degree vertex A_i, examine whether or not there exist four paths $A_i \rightarrow X_p \rightarrow B_j$, $A_i \leftarrow Y_q \leftarrow B_j$, $A_i \rightarrow X'_r \rightarrow B_j$, and $A_i \leftarrow Y'_s \leftarrow B_j$. If such paths exist, output cuts corresponding to (A_i, B_j).
3. For each high degree vertex B_j, execute a similar procedure as Step 2.
4. Remove all high degree vertices A_i, B_j from the graph.
5. Let $E' = \{(A_i, \langle X_k, Y_h\rangle)|$ there exists a path $Y_h \rightarrow A_i \rightarrow X_k\} \cup \{(\langle X_k, Y_h\rangle, B_j)|$ there exists a path $X_k \rightarrow B_j \rightarrow Y_h\}$. Construct E'' be in the same way using X'_k, Y'_h except that the directions of edges are reversed. Let G' be a directed graph defined by the edge set $E' \cup E''$.
6. Find a cycle of length 4 in G'.

Lemma 1. Algorithm 2 correctly finds chemical cuts of size 1 in $O(n^{2-\frac{1}{5}})$ time.
(Proof) Since the correctness is almost trivial (see Fig. 4), we analyze the time complexity.

Step 1 and Step 4 take $O(n)$ time. Step 2 and Step 3 take $O(n^2/\Delta)$ time, where we omit details here. Since G' have $O(n\Delta)$ vertices and $O(n\Delta)$ edges, Step 5 and Step 6 take $O(n\Delta)$ time and $O((n\Delta)^{1.5})$ time, respectively. Solving $(n\Delta)^{1.5} = n^2/\Delta$, we have $\Delta = n^{1/5}$. Therefore, Algorithm 2 works in $O(n^{2-\frac{1}{5}})$ time. □

Using Algorithm 2 repeatedly, we have the following [2].

Theorem 4. Let K be a constant and the power of 2 ($K = 2^L$). Then, we can solve the special case of Problem 2A in $O(n^{2-\frac{1}{\frac{3}{2}\cdot K^{\log 3}+\frac{1}{2}}})$ time.

4 Practical Algorithms

We can develop practical algorithms by replacing unique naming algorithms with the Morgan algorithm [11]. The Morgan algorithm iteratively computes an integer label $i_G(v)$ for each node v in G such that two vertices v and v' have the same integer label iff. there exists an isomorphic mapping from G to G that maps v to v'. Though this property is not strictly satisfied, it holds for almost all chemical structures. The Morgan algorithm works in $O(n^2)$ time for a chemical structure of size n. In order to obtain the unique name of G, we simply use the maximum of $i_G(v)$ among all vertices v in G. Using (a variant of) the Morgan algorithm, we develop Algorithm 3 for a special case of $p = q = 2$, $C = 1$ of Problem 1 (or Problem 1A), where compounds are not limited to trees.

Algorithm 3

1. For all partitions (X_i, A_i) of I_1, for all partitions (Y_j, B_j) of I_2, for all partitions (X_k, B_k) of O_1, and for all partitions (Y_h, A_h) of O_2, compute the Morgan names of $X_i, A_i, Y_j, B_j, X_k, B_k, Y_h, A_h$.
2. For each A_i, examine whether there exists A_h having the same Morgan name. If so, create an object A_i having the Morgan name as a label.
3. Execute a similar procedure for B_j.
4. For all pairs of objects (A_i, B_j), let $M[A_i, B_j] = 0$.
5. For all pairs (X_i, X_k) having the same Morgan name, let $M[A_i, B_k] = 1$.
6. For all pairs (Y_j, Y_h) having the same Morgan name, let $M[A_h, B_j] = 2$ if $M[A_h, B_j] = 1$.
7. Output (A_i, B_j) such that $M[A_i, B_j] = 2$.

It should be noted that Algorithm 3 outputs (A_i, B_j). But, it is not difficult to obtain a mapping rule or a chemical cut from (A_i, B_j) though one (A_i, B_j) may corresponds to multiple mapping rules or chemical cuts.

It is easy to see that Algorithm 3 works in $O(n^3)$ time. Though the correctness of Algorithm 3 is not guaranteed, it correctly works for almost all cases under the condition that $p = q = 2$ and $C = 1$. Algorithm 3 does not search cycles of length 4. Instead, it uses a two-dimensional matrix. As shown in Section 5, only a small number of objects (A_i and B_j) are created in most practical cases. Thus, Steps 4-7 can be completed very quickly.

Algorithm 3 can be modified for the other p, q, C, where we omit details here.

5 Computational Experiments

We performed computational experiments on Algorithm 3 using enzymatic reaction data in the KEGG/LIGAND database, release 20.0 (http://www.genome.ad.jp/ligand/) [9]. This version contains 5238 enzymatic reactions. But, Algorithm 3 cannot handle all reactions because Algorithm 3 accepts reactions with $p = q = 2$. Thus, we used 2346 reactions satisfying this condition. We used a standard PC with an Intel Pentium-4 2.2GHz CPU and 2GB main memory.

We examined three variants of Algorithm 3.

Algorithm 3a. All hydrogen atoms are reconstructed from reactions because hydrogen atoms are omitted in the LIGAND database. Types of bonds are taken into account. Removed edges are also taken into account (this case corresponds to Problem 1A).

Algorithm 3b. Hydrogen atoms and types of bonds are ignored. But, removed edges are taken into account (this case corresponds to Problem 1A too).

Algorithm 3c. Hydrogen atoms and types of bonds are ignored. Removed edges are not taken into account (this case corresponds to Problem 1).

Ignoring hydrogen atoms might be reasonable from a chemical point of view [4] because some hydrogen atoms (and some oxygen atoms) might come from solvent.

The results are summarized in Table 1. For each algorithm, the total CPU time for 2346 reactions, the number of successful reactions (i.e., the number of reactions for which at least one pair (A, B) is output), the average number of output pairs (A, B) per reaction, and the average number of the created objects (see Algorithm 3) per reaction are shown. From the table, it is seen that all algorithms are fast enough. Each algorithm took less than 0.1 sec. per reaction on the average. It is also seen that both the number of output pairs and the number of created

objects are small for all algorithms. Table 1 suggests that Algorithm 3c is better than Algorithm 3a and Algorithm 3b. It is reasonable since Algorithm 3c is more flexible than Algorithm 3a and Algorithm 3b. For other details of computational experiments, see [2].

Table 1: Result of computational experiment on Algorithm 3.

	Total CPU time	#Successful reactions	$\#(A, B)$ per reaction	#Created objects per reaction
Algorithm 3a	150.4 sec.	1104	2.50	8.35
Algorithm 3b	68.0 sec.	1071	2.18	8.45
Algorithm 3c	66.7 sec.	1912	2.15	8.82

6 Acknowledgements

The author is grateful to Masanori Arita in Computational Biology Research Center for stimulating discussions. The author would like to thank Yasushi Okuno and Masahiro Hattori in Kyoto University for helpful discussions. This work was supported in part by Grants-in-Aid "Genome Information Science" and #13680394 from MEXT of Japan.

References

[1] A. V. Aho, J. E. Hopcroft and J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, 1974.

[2] T. Akutsu, Efficient extraction of mapping rules of atoms from enzymatic reaction data, *7th Int. Conf. Computational Molecular Biology* (RECOMB 2003, to appear).

[3] N. Alon, R. Yuster and U. Zwick, Finding and counting given length cycles, *Proc. 2nd European Symp. Algorithms*, 354–364, 1994.

[4] M. Arita, *Automated Metabolic Reconstruction: Theory and Experiments*, Ph. D Thesis, University of Tokyo, 2000.

[5] D. Eppstein, Subgraph isomorphism in planar graphs and related problem, *J. Graph Algorithms and Applications*, 3:1–27, 1999.

[6] V. Kann, On the approximability of the maximum common subgraph problem, *Lecture Notes in Computer Science*, 577:377–388, 1992.

[7] P. D. Karp *et al.*, The EcoCyc database, *Nucleic Acids Res.*, 30:56–58, 2002.

[8] M. Fürer, W. Schnyder and E. Specker, Normal forms for trivalent graphs and graphs of bounded valence, *Proc. 15th. ACM Symp. Theory of Computing*, 161–170, 1983.

[9] S. Goto *et al.*, LIGAND: database of chemical compounds and reactions in biological pathways, *Nucleic Acids Res.*, 30:402–404, 2002.

[10] E. M. Luks, Isomorphism of graphs of bounded valence can be tested in polynomial time, *Proc. 21st. IEEE Symp. Foundations of Computer Science*, 42–49, 1980.

[11] H. L. Morgan, The generation of a unique machine description for chemical structures - A technique developed at chemical abstracts service, *J. Chemical Documentation*, 5:107–113, 1965.

[12] T. Wang and J. Zhou, EMCSS: a new method for maximal common substructure search, *J. Chemical Information and Computer Sciences*, 37:828–834, 1997.