数理解析研究所講究録
第 731 巻 1990 年 195-201

On the Largest Common Subgraph Problem

Shigeru Masuyama, Yoshimasa Takahashi,
Tohru Okuyama*and Shin-ichi Sasaki†
Department of Knowledge-Based Information Engineering
Toyohashi University of Technology
Toyohashi 441, Japan

March 13, 1990

Abstract

The largest common subgraph problem (LCSG, for short) asks to find a common connected subgraph of the given two graphs G_1 and G_2, with the largest number of edges. In this paper, we develop polynomial time algorithms for LCSG when both G_1 and G_2 are trees.

1 Introduction

Given two graphs G_1 and G_2, *the largest common subgraph problem (LCSG,* for short) asks to find a common connected subgraph of both G_1 and G_2, with the largest number of edges. These problems appear in detection and recognition of the largest connected substructure possessed commonly by a plural number of chemical structures in the structure-activity studies, where exploring functional groups or particular structural fragments common to organic molecules which have the same biological or pharmacological function is one of the significant issues. Several computational methods for the problem of finding largest common substructures in two or more molecules were suggested (see e.g., [2, 4, 9, 10]). Generally, however, their algorithms arisen in such application are quite time-consuming as they are performed in exaustive manner

*Computer Center, Toyohashi University of Technology.
†Vice President, Toyohashi University of Technology.

based on atom-by-atom (i.e., vertex-by-vertex) comparisons. It is true that chemical structure diagrams are closely related to the graphs in their representation. Thus more efficient algorithms are now actually required for the LCSG problem in systematization of the computer-assisted design in chemistry.

LCSG in general graphs is obviously NP-complete as Hamiltonian circuit problem, which is known to be NP-complete [1], can easily be reduced to this problem where $G_1 = C_{|V|}$ (a cycle on $|V|$ vertices) and $G_2 = G(= (V, E))$. Efficient algorithms may, however, exist for some special cases.

In this paper, we develop polynomial time algorithms for LCSG when both G_1 and G_2 are trees.

2 LCSG Over a Rooted Tree

In this section, a polynomial time algorithm for LCSG is developed when both G_1 and G_2 are rooted trees (T_1, r_1) and (T_2, r_2), where r_1 (r_2) is the root of T_1 (T_2). For each vertex v in (T, r), let $(T(v), v)$ denote the subtree of (T, r) whose root is v and spanned by all the descendants of v. We first introduce *procedure LCS* for obtaining the largest common subtree (T, r) of two rooted trees (T_1, r_1) and (T_2, r_2) where r corresponds to both r_1 and r_2, respectively.

Procedure $LCS((T_1, r_1), (T_2, r_2), N((T_1, r_1), (T_2, r_2)))$
Input: Rooted trees (T_1, r_1) and (T_2, r_2).
Output: The number $N((T_1, r_1), (T_2, r_2))$ of edges of the largest common subtree (T, r) of (T_1, r_1) and (T_2, r_2) where r corresponds to both r_1 and r_2, respectively.

```
begin (Initialization)
For each leaf $l_i$ of $(T_1, r_1)$ and $l'_j$ of $(T_2, r_2)$,
    $N((T_1(l_i), l_i), (T_2(l'_j), l'_j)) \leftarrow 0.$
end

begin
Let $v_1, \dots, v_d$ and $w_1, \dots, w_{d'}$ be sons of the roots $r_1$ and $r_2$, respectively.
Recursively call procedure $LCS((T_1, r_1), (T_2, r_2), N((T_1(v_i), v_i), (T_2(w_j),$
```

$w_j)))$ for $i = 1, ..., d,\ j = 1, ..., d'$, and construct a bipartite graph $G = (V_1, V_2, E,\ c)$, where $V_1 = \{(T_1(v_i), v_i)|i = 1, ..., d\}$, $V_2 = \{(T_2(w_j), w_j)|j = 1, ..., d'\}$, $E = \{((T_1(v_i), v_i), (T_2(w_j), w_j))|i = 1, ..., d,\ j = 1, ..., d'\}$ and the weight $c((T_1(v_i), v_i), (T_2(w_j), w_j))$ of each edge $((T_1(v_i), v_i), (T_2(w_j), w_j))$ is

$$N((T_1(w_i), w_i), (T_2(v_j), v_j)) + 1\ (i = 1, ..., d,\ j = 1, ..., d').$$

$N((T_1, r_1), (T_2, r_2)) \leftarrow$ the value of the maximum weight matching [7] of G.

end □

Note 2.1. Throughout this paper, we introduce algorithms for obtaining the number of edges of the largest common subtree. It is easy, however, to modify such algorithms so that the actual largest common subtree is obtained, although we omit the details. □

Lemma 2.1. Procedure LCS correctly finds the number $N((T_1, r_1), (T_2, r_2))$ of the largest common subtree (T, r) of (T_1, r_1) and (T_2, r_2), here r corresponds to both r_1 and r_2, in $O(n_1^2 n_2)$ time, where n_1 (n_2) is the number of vertices in (T_1, r_1) $((T_2, r_2))$.

(Proof) We shall prove this lemma by induction on h, where h is the height of the tree which is heigher than or equal to the other among (T_1, r_1) and (T_2, r_2). This lemma is trivial when $h = 1$. Soppose that this lemma holds when $h \leq H - 1$, i.e., $N((T_1(v_i), v_i), (T_2(w_j), w_j))$ obtained by procedure LCS is the number of the largest common subtree of $(T_1(v_i), v_i)$ and $(T_2(w_j), w_j)$ for each pair $(T_1(v_i), v_i)$ and $(T_2(w_j), w_j)$, where $v_1, \ldots, v_d$ and $w_1, \ldots, w_{d'}$ are sons of the roots r_1 and r_2, respectively. Without loss of generality, we assume that the height of (T_1, r_1) $\leq$ the height of (T_2, r_2), i.e., the height of (T_2, r_2) is H. Let (T, r) be the largest common subtree of (T_1, r_1) and (T_2, r_2) where r corresponds to both r_1 and r_2, and let (T', r_1) $((T'', r_2))$ be the subtree in (T_1, r_1) $((T_2, r_2))$ isomorphic to (T, r). We consider a bipartite graph $G = (V_1; V_2, E, c)$, where $V_1 = \{(T_1(v_i), v_i)|i = 1, ..., d\}$, $V_2 = \{(T_2(w_j), w_j)|j = 1, ..., d'\}$, $E = \{((T_1(v_i), v_i), (T_2(w_j), w_j))|i = 1, ..., d, j = 1, ..., d'\}$ and the weight $c((T_1(v_i), v_i), (T_2(w_j), w_j))$ of each edge $((T_1(v_i), v_i), (T_2(w_j), w_j))$ is

$$N((T_1(v_i), v_i), (T_2(w_j), w_j))) + 1\ (i = 1, ..., d, j = 1, ..., d').$$

Consider the set of edges $M = \{((T_1(v_i), v_i), (T_2(w_j), w_j))|(T_1(v_i), v_i)$ and $(T_2(w_j), w_j)$ corresponds to the same subtree of $(T, r)\}$ $(\subset E)$. Then M

must be the maximum weight matching of G as, otherwise, let M' be a matching of G whose value is greater than that of M and we can construct a larger common subtree of (T_1, r_1) and (T_2, r_2) from $M' \subset E$, where r corresponds to r_1 and r_2. This, however, contradicts the fact that (T, r) is the largest common subtree of (T_1, r_1) and (T_2, r_2), proving that the common subtree obtained by the algorithm is also the largest one when $h = H$, as desired. Thus we have proved this lemma by induction.

The time complexity is now analyzed. Note that the most time-consuming part is the maximum weight matching. We assume here that $h_1 \leq h_2$ where h_1 (h_2) is the height of T_1 (T_2). The maximum weight matching is solved at vertices within distance h_1 from r_2 in T_2 and at each vertex v whose degree is k_2 matched with vertex in T_1 whose degree is k_1, the maximum weight matching is solved in $O(k_1^2 k_2)$ time by a well-known primal-dual type algorithm (see e.g., [7]). Thus the time complexity in total is

$$\sum O(k_1^2 k_2) = O(n_1^2 n_2). \quad \square$$

Now we turn to LCSG where the root of common subtree may correspond to any vertex in (T_1, r_1) $((T_2, r_2))$.

Note 2.2. The root r of the largest common subtree (T, r) must correspond to at least one of r_1 (of (T_1, r_1)) or r_2 (of (T_2, r_2)) as:

Let (T', v_1) $((T'', v_2))$ be the subtree isomorphic to (T, r) in (T_1, r_1) $((T_2, r_2))$ and consider path s_1 in (T_1, r_1) (s_2 in (T_2, r_2)) from v_1 to r_1 (from v_2 to r_2) and, without loss of generality, we assume that $s_1 \leq s_2$ holds. Let s' be the path in (T_2, r_2) from v_2 to an ancestor v' of v_2, whose length is s_1. Then by appending s_1 (s') to (T', v_1) $((T'', v_2))$, we have a larger common subtree of (T_1, r_1) and (T_2, r_2), contradicting the assumption. $\square$

Based on Note 2.2, we have the following algorithm LCRT for obtaining the largest common subtree of (T_1, r_1) and (T_2, r_2).

Algorithm LCRT($(T_1, r_1), (T_2, r_2), N$)
Input: Rooted trees (T_1, r_1) and (T_2, r_2).
Output: The number N of edges of the largest common subtree (T, r) of (T_1, r_1) and (T_2, r_2).

Step 1. For each vertex v in (T_1, r_1) call LCS$((T_1(v), v), (T_2, r_2), N((T_1(v), v), (T_2, r_2)))$.

Step 2. For each vertex v in (T_2, r_2) call LCS$((T_2(v), v), (T_1, r_1), N((T_2(v), v), (T_1, r_1)))$.

Step 3.

$$N \leftarrow max\{\max_{v \text{ in } (T_1,r_1)} N((T_1(v), v), (T_2, r_2)), \max_{v \text{ in } (T_2,r_2)} N((T_2(v), v), (T_1, r_1))\}$$

Step 4. Halt. □

Theorem 2.1. Algorithm LCRT solves LCSG over rooted trees in $O(n_1^2 n_2^2)$ time, where n_1 (n_2) is the number of vertices in (T_1, r_1) ((T_2, r_2)).

(Proof) The correctness is based on Note 2.2 and that of procedure LCS, hence that of Lemma 2.1.

Step 1 in algorithm LCRT is performed in $O(n_1 n_1 n_2^2) = O(n_1^2 n_2^2)$ time and Step 2 of algorithm LCRT is executed in $O(n_2 n_1^2 n_2) = O(n_1^2 n_2^2)$ time by Lemma 2.1. Thus the time complexity of this algorithm in total is $O(n_1^2 n_2^2)$. □

3 LCSG over an Undirectred Tree

In this section, we develop a polynomial time algorithm for the largest common subtree T of undirected trees T_1 and T_2. Let (T_1, r_1) ((T_2, r_2)) be a rooted tree obtained from T_1 (T_2) by choosing an arbitrary vertex r_1 in T_1 (r_2 in T_2) as a root. Then a rooted tree (T, r) isomorphic to rooted subtrees (T', v') in (T_1, r_1) and (T'', v'') in (T_2, r_2), both corresponding to T, are also the largest common subgraphs of (T_1, r_1) and (T_2, r_2). Conversely, let (T, v) be the largest common subtree of (T_1, r_1) and (T_2, r_2) and T' be the corresponding undirected tree obtained from T by neglecting the direction of edges and by not specifying any vertex as a root. Then T' is also the largest common subtree of T_1 and T_2. Based on this observation, we have the following algorithm LCUT whose time complexity is $O(n_1^2 n_2^2)$ where n_1 (n_2) is the number of vertices in T_1 (T_2).

Algorithm LCUT

Input: Undirected trees $T_1 = (V_1, E_1)$ and $T_2 = (V_2, E_2)$.

Output: The number N of the edges of the largest common subtree of undirected trees $T_1 = (V_1, E_1)$ and $T_2 = (V_2, E_2)$.

Step 1. Choose arbitrary vertices r_1 of T_1 and r_2 of T_2, respectively, and construct rooted trees (T_1, r_1) and (T_2, r_2). Execute algorithm LCRT($(T_1, r_1), (T_2, r_2), N$).

Step 2. Halt. □

4 Concluding Remarks

We can easily apply algorithm LCUT developed in this paper to the recognition of the largest common substructure of two chemical compounds with tree structures by specifying which vertex (edge) of T_1 may correspond to which vertex (edge) of T_2 according to the definition of structural similarity (see e.g., [10]) which is most suitable for the purpose of each research. Although applicability of the present algorithm is limited to acyclic molecules, it would be possible to apply it to wider class of molecules which have isolated simple rings by abstracting or devising the graph representation of their structures.

Finding some other special cases in which LCSG can be solved efficiently, providing good heuristic algorithms for general graphs and developing algorithms to find the largest common subgraph of more than two graphs (see e.g., [9, 10]) seem to deserve further research. The algorithms developed in this paper may be useful for these purposes.

References

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

[2] J. E. Armitage, J. E. Crowe, P. N. Evans, M. F. Lynch and J. A. McGuirk, "Documentation of Chemical Reactions by Computer Analysis of Structural Changes", J. Chem. Doc., 7, 209-215 (1967).

[3] C. Berge, *Graphs and Hypergraphs*, North-Holland, Amsterdam (1973).

[4] M. M. Cone, R. Venkataraghavan and F. W. McLafferty, "Molecular Structure Comparison Program for the identification of Maximal Common Substructures", J. Am. Chem. Soc. 99, 7668-7671 (1977).

[5] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman and Company, San Francisco (1979).

[6] F. Harary, *Graph Theory*, Addison-Wesley, Reading, Mass. (1969).

[7] E. L. Lawler, *Combinatorial Optimization: Networks and Matroids*, Holt Rinehalt and Winston, New York (1976).

[8] S. Masuyama, "On the minimum cost subgraph problem", *Proceedings of the International Workshop on Discrete Algorithms and Complexity*, 101-106 (1989).

[9] Y. Takahashi, Y. Sato, H. Suzuki and S. Sasaki, "Recognition of largest common structural fragment among a variety of chemical structures", *Analytical Sciences*, 3, 23-28 (1987).

[10] T. H. Varkony, Y. Shiloach and D. H. Smith, "Computer Assisted Examination of Chemical Compounds for Structural Similarlities", J. Chem. Inf. Comput. Sci. 19, 104-111 (1979)