The efficient time algorithm for isomorphism of block cactus graphs

2330022 Issei Munetomo

The graph isomorphism problem for block cactus graphs with n vertices and m edges can be solved in $O(n+m)$ time. Two graphs $G=(V,E)$ and $G'=(V',E')$ are isomorphic if and only if there exists a bijection $f:V\rightarrow V'$ such that any two vertices $u,v\in V$ have an edge in G if and only if the vertices $f(u),f(v)\in V'$ have an edge in G'. The graph isomorphism problem is to determine if G and G' are isomorphic. The computational complexity of the general graph isomorphism problem is not known. However, it is known only to belong to the complexity class NP, because if a bijection between the vertices of two graphs is given, it is verifiable in polynomial time whether that bijection actually defines a graph isomorphism. On the other hand, it is known that the graph isomorphism problem can be solved with efficient time complexity for certain graph classes. In particular, for graph classes that can be represented by a type of tree structure, efficient solutions are known by reducing the original problem to the graph isomorphism problem for trees.

Block cactus graphs can be represented by a tree structure using a PQ-Tree. For this tree representation, the graph isomorphism problem for block cactus graphs is reduced to the graph isomorphism problem for PQ-Trees. A block cactus graph is a graph in which all blocks are either cliques or cycles. A block cactus graph can be converted into a type of tree structure by representing its blocks as star graphs and connecting them at the cut vertices. We call this tree structure a Block-star tree. There is an important property that the vertices in a clique can be arbitrarily permuted, but the vertices in a cycle can only be permuted in their cyclic order or its reverse order. To represent this property of the block cactus graph in the Block-star tree, the vertices of the star graph representing a cycle can only be permuted in the original cyclic order or its reverse. To achieve this, we use a PQ-Tree which can assign constraints on the permutation of children to the vertices of a rooted tree. P-node in a PQ-Tree is a vertex that allows arbitrary permutations of its children. Q-node in a PQ-Tree is a vertex that only allows permutations of its children in forward or reverse order. By assigning P-node to the vertex of the star graph representing a clique and Q-node to the vertex of the star graph representing a cycle, the properties of the block cactus graph can be represented by the PQ-Tree.

The graph isomorphism problem for a PQ-Tree representing a block cactus graph is reduced to a string comparison problem. We call this string that represents the graph isomorphism the GI-Canonical representation. For a block cactus graph with n vertices and m edges, the GI-Canonical representation string can be constructed in $O(n+m)$ time. It uses algorithms for finding the lexicographically smallest cyclic permutation of a set of cyclic strings and the Longest Common Extension.

First, we find the center vertex of the Block-star tree and represent it as a rooted tree with that vertex as the root. Next, we perform a depth-first search on the rooted Block-star tree, simultaneously constructing the PQ-Tree and creating the GI-Canonical representation string. The process is simple if the root vertex becomes a P-node. The GI-Canonical representation of the PQ-Tree can be created in $O(n)$ time from an n vertices block cactus graph. However, if the root vertex becomes a Q-node, there is the most significant problem. We have to convert vertices of the cycle into a canonical sequence of vertices. To solve this problem, we define the canonical sequence of vertices that makes the lexicographically minimal list of GI-Canonical representation strings that can be created from the vertices of the cycle. We call this process root normalization. Finally, we introduce a data structure for the Longest Common Extension problem which enables lexicographical comparison of two GI-Canonical representation strings of length $O(n)$ in $O(1)$ time to improve the time complexity of the root normalization process. After $O(n)$ preprocessing, the GI-Canonical representation of a PQ-Tree with a Q-node root can be created in $O(n)$ time by using this data structure.
Once the GI-Canonical representation is constructed, the graph isomorphism problem for block cactus graphs can be solved simply by comparing the strings.