Finding the Closed Partition of a Planar Graph\(^1\)

Vijaya Ramachandran\(^2\) and Honghua Yang\(^2\)

**Abstract.** In this paper we consider the problem of finding a *closed partition* in a directed graph. This problem has applications in concurrent probabilistic program verification. The best sequential algorithm known for this problem runs in $O(mn)$ time where $m$ is the number of directed edges and $n$ is the number of vertices in the given digraph. In this paper we present a linear-time sequential algorithm to solve the closed partition problem for planar digraphs that are compact. We then build on this algorithm to obtain an $O(n^{1.5})$-time sequential algorithm to solve the closed partition problem for a general planar digraph.

**Key Words.** Closed partition, Compact digraph, Directed graph, Dual graph, Planar graph, Sequential algorithm, Strongly connected component.

1. Introduction. In this paper we consider the *closed partition problem* \([13], [2]\). Let $G = (V, E)$ be a digraph in which the vertices in a set $V' \subseteq V$ are colored blue. A *closed component* in $G$ is a maximal induced subgraph $S$ of $G$ such that either $S$ is a single vertex or $S$ is strongly connected and no blue vertex in $S$ has an outgoing edge in $G$ to a vertex not in $S$. A closed component is *nontrivial* if it contains more than one vertex. Note that, in general, a closed component is a subgraph of a strongly connected component. The *closed partition problem* is to find the collection of closed components in $G$. This problem has been widely studied in concurrent probabilistic program verification \([2], [8], [12]\). It is also interesting as a graph-theoretical problem.

The following results are known about the complexity of the closed partition problem \([13]\). The problem can be solved by the following straightforward polynomial-time algorithm: find the strongly connected components of the input graph, and repeatedly refine each strongly connected component by deleting a blue vertex with an edge outgoing from the component (if such a vertex exists) and recomputing the strongly connected components in the resulting graph. If the input graph has $n$ nodes and $m$ edges, then this algorithm runs in $O(mn)$ time. A reduction from the monotone circuit value problem \([4]\) shows that the problem is $P$-complete and hence unlikely to have an efficient highly parallel algorithm.

A *compact digraph* is an embedding of a planar digraph in which no strongly connected component encloses any other strongly connected component. In this paper we present a linear-time sequential algorithm to solve the closed partition problem for compact digraphs. We then extend this algorithm to obtain an

\(^1\) This work was supported in part by NSF Grant CCR 89-10707.

\(^2\) Department of Computer Sciences, The University of Texas at Austin, Austin, TX 78712, USA.
An $O(n^{1.5})$-time sequential algorithm to solve the closed partition problem for a general planar digraph where $n$ is the number of vertices in the input digraph.

The main idea in our algorithms is to take advantage of the dual digraph of a plane digraph. As in the sequential algorithm described above, our algorithms work with a strongly connected subgraph of the input graph and repeatedly deletes a blue vertex that has an edge outgoing from the subgraph until no such blue vertex can be found. In general, it is difficult to decide, when deleting a blue vertex from a digraph, whether a strongly connected subgraph is separated. However, the decision becomes easier when dealing with the dual of a plane digraph. We make use of the fact that a plane digraph is strongly connected if and only if its dual digraph is acyclic (see, e.g., [6]), and an extension of this result that the edges that are not in any strongly connected component of a plane digraph are exactly those edges whose dual edges are in the strongly connected components of the dual digraph.

We do not know if planar graphs are likely to appear in the applications cited in [2], [8], and [12]. However, planar digraphs form a natural subclass of directed graphs, and our algorithm can be viewed as a first step toward obtaining more efficient algorithms for the closed partition problem. Further, our technique of moving between the primal and dual of a plane embedding in order to obtain an efficient algorithm is one that may have applications in other problems on planar directed graphs.

The rest of this paper is organized as follows: Section 2 defines the terminology used in this paper. Section 3 gives the main lemmas that establish the relationship between the operations in a primal digraph and the operations in its dual digraph. Section 4 gives a high-level description of the algorithm for compact digraphs. Section 5 presents a detailed description of this algorithm and its complexity analysis. Using the algorithm for compact digraphs, we give an algorithm to solve the closed partition problem for a general planar digraph in Section 6. We discuss some open problems in Section 7. An example is included in the Appendix to illustrate how the first algorithm works on a compact digraph. Figures 5 and 11 in the Appendix show a compact digraph $G$ and its closed partition, respectively.

2. Preliminaries. A (weakly) connected component in a digraph $G$ is a connected component in the undirected graph obtained from $G$ by making the edges in $G$ undirected. A digraph is strongly connected if every two vertices in the digraph are reachable from each other. A strongly connected component (scc for short) of a digraph is a maximal subgraph that is strongly connected. An scc which contains only one vertex is a trivial scc. An scc which contains more than one vertex is a nontrivial scc.

The start-point $u$ of a directed edge $(u, v)$ is called the tail of the edge and the end-point $v$ is called the head of the edge. The directed edge $(u, v)$ is said to be outgoing from $u$ and incoming to $v$. An edge is connected to a vertex if it is either incoming to or outgoing from the vertex. A directed edge is outgoing from a subgraph if its tail is a vertex of the subgraph and its head is not. An edge that is outgoing from an scc is called a link edge.