Abstract

In [L 85a] a hierarchical graph model is defined that allows the exploitation of the hierarchy for the more efficient solution of graph problems on very large graphs. The model is motivated by applications in the design of VLSI circuits.

We show how to efficiently find minimum spanning forests in this graph model. We solve decision problems pertaining to minimum spanning forests in almost linear time in the length of the hierarchical description, query problems in time linear in the depth of the hierarchy and we construct minimum spanning forests in space linear in the length of the hierarchical description.

1. Introduction

In [L 85a] a hierarchical graph model has been introduced that allows to speed up the solution of graph problems by exploiting the hierarchy. The model is motivated by applications of graph algorithms in the design of hierarchically specified integrated circuits.

The exploitation of the hierarchy is necessary even for simple, e.g., linear time graph problems, because graphs arising from hierarchical descriptions can be too large to fit into the main memory of a computer. Thus running a non-hierarchical graph algorithm on graphs of this size amounts to extraordinary amounts of page swapping. For practical purposes it is important to decrease the space requirement for processing such graphs. This is done by modifying the algorithms such that they exploit the hierarchy of the graph description and need only store the hierarchical description of the graph instead of the fully expanded graph. [L 85a] presents a special method for doing this, the so-called bottom-up method, and applies the method to the solution of connectivity problems on directed and undirected graphs. [L 85b] applies the same method for testing planarity of hierarchically defined graphs.

In this paper, we will apply the bottom-up method to the problem of finding minimum spanning forests of hierarchically defined graphs. Specifically we deal with three
problems pertaining to minimum spanning trees, a decision problem, a query problem, and a construction problem that requires the generation of large output.

Thus for the decision and query problem substantial time savings are achieved (over the non-hierarchical solutions up to exponential, depending on the hierarchy). The construction problem can be solved using up to exponentially less space than with a non-hierarchical algorithm.

The paper is organized as follows. Section 2 defines the hierarchical graph model. Section 3 introduces the bottom-up method. Section 4 discusses the decision problem. Section 5 considers the query problem. The construction problem is solved in Section 6. Section 7 gives conclusions.

2. Basic definitions

Definition 1: A hierarchical undirected graph \( G = (G_1, \ldots, G_k) \) is a tuple of undirected graphs \( G_i \) called cells. Here the graph \( G_i \) has \( n_i \) vertices and \( m_i \) edges. \( p_i \) of the vertices are distinguished and called pins. The other \( n_i-p_i \) vertices are called inner vertices. \( r_i \) of the inner vertices are distinguished and called nonterminals or hierarchical vertices. The other \( n_i-r_i \) vertices are called terminals or proper vertices.

Each pin has a unique label, its name. W.l.o.g. we can assume that the pins are named with numbers between 1 and \( p_i \). Each nonterminal has two labels, a name and a type. The type is a symbol from \( \{G_1, \ldots, G_{i-1}\} \). If a nonterminal \( v \) has type \( G_j \) then \( v \) has degree \( p_j \) and each proper vertex that is a neighbour of \( v \) has a label \( (v, \xi) \) such that \( 1 \leq \xi \leq p_j \). We say that the neighbour of \( v \) labeled \( (v, \xi) \) matches the \( \xi \)-th pin of \( G_j \). (All neighbours of a nonterminal must be terminals.)

We assume that \( r \) is irredundant in the sense that each \( G_j \) is the type of some nonterminal \( v \) in some \( G_j, j > i \).

The size of \( r \) is \( n := \sum_{1 \leq i \leq k} n_i \), the edge number is \( m := \sum_{1 \leq i \leq k} m_i \).

Note that with \( r = (G_1, \ldots, G_k) \) also each prefix \( r_i = (G_1, \ldots, G_i), i < k \) is a hierarchical graph.

A hierarchical edge labelled graph is a graph such that each edge between terminals in a cell is labelled with a positive number, its length.

Definition 1 essentially describes a context-free graph grammar with axiom \( G_k \).