External labeling and algorithms for hierarchic networks

Aarni Perko
Department of Mathematics, University of Jyväskylä
SF-40100 Jyväskylä, Finland

Abstract

Algorithms for generating internal data structures for networks are given. Data bases for networks can be partitioned hierarchically. Nodes of lower class networks may also be in the interior of a higher class arc which will be split if it is included into the final network. Naming is based either on nodes or on arcs.

1. Introduction

Big networks are often hierarchical. Road networks, for example, consist of highways, regional roads and local roads. Data bases may be partitioned accordingly. They may be even maintained by different officials. In applications, however, the used subnetwork may contain roads from several hierarchy classes.

In road networks especially the regional roads may have plenty of junctions with local roads. When roads from all classes are used together then each section of a higher level road between junctions with lower level roads should be a separate arc (Fig. 1 (a)). Thus number of arcs becomes large. When only higher level roads are considered in some application then it is inconvenient if they are divided into small sections. Maintainers of high level networks may also insist that they define their network data base in a way which is most suitable for them. So the arcs of the high level network may be longer sections between main junctions, without taking into account junctions with the lower level roads (Fig. 1 (b)). However, the final network used in computations must have the normal graph structure. When some low level arcs are included which meet a high level arc in its interior then this arc must be split correspondingly (Fig. 1 (c)).

We discuss algorithms and data structures which may be used when the final network is gathered from separate networks. Databases for networks may even have different formats. We make following assumptions:

- Let hierarchy classes be $C_1, C_2, \ldots, C_h$ by increasing quality.
- Data bases for classes $C_i, \ldots, C_h$ need not have any references to lower classes $C_1, \ldots, C_{i-1}$.
- The final network used in a particular application may contain arcs from different classes. These arcs may be selected by any application dependent
- Let \( a \) be an arc which is incident in its interior point \( X \) with some set \( L(X) \) of lower level arcs. Then \( X \) must be an end node for the arcs in \( L(X) \). If the arc \( a \), and at least one arc from \( L(X) \) is included into the final network, then the arc \( a \) will be split at \( X \).

- Arcs in the same class \( C_i \) can meet only in nodes from classes \( C_i, \ldots, C_h \).

Fig. 1. (a) Each section between junctions forms an arc. (b) A section between high level junctions forms an arc in the high level network and junctions with lower level arcs are neglected. (c) Some low level arcs are included and the higher level arc is split correspondingly.

External names for nodes and arcs are needed for human communications and maintenance of data bases. We discuss naming schemes which make possible efficient one-pass algorithms for generating the internal data structures. Two such algorithms are given which take into account splitting of arcs in junctions with lower level arcs. Arcs can also be split at certain specific points in order to force these points to be nodes in the final network [2].

2. Data structures and naming schemes

The data base for each hierarchy class consists naturally of nodes and arcs. Both parts have their own search and update methods which use unique names for items. Storing arcs is straightforward because hierarchy classes form just a