Compact Routing Methods: A Survey

J. van Leeuwen and R.B. Tan

UU-CS-1995-05
February 1995

Compact Routing Methods: A Survey

J. van Leeuwen and R.B. Tan

Technical Report UU-CS-1995-05
February 1995

Department of Computer Science
Utrecht University
P.O.Box 80.089
3508 TB Utrecht
The Netherlands

ISSN: 0924-3275

Compact Routing Methods: A Survey*

Jan van Leeuwen[1] and Richard B. Tan[1,2]

[1] Department of Computer Science, Utrecht University
Padualaan 14, 3584 CH Utrecht, the Netherlands
(jan@cs.ruu.nl)

[2] Department of Computer Science
University of Sciences & Arts of Oklahoma
Chickasha, Oklahoma 73018, U.S.A.
(rbtan@cs.ruu.nl, rtan@mercur.usao.edu)

Abstract. We give a short survey on various compact routing methods used in communication networks. The routing schemes considered are Interval Routing, Prefix Routing and Boolean Routing. Various known characterizations of networks that have such routing schemes are presented. A few open problems in this area also are given.

1 Introduction

In a parallel or distributed system, as more processors are added to increase the overall computing power, the underlying communication network needs to *scale* favorably along with the expansion. As the amount of storage space at each processor is limited, the expansion of the network should not put undue burden locally by requiring excessive space for communication purposes. The *routing* methods used should also be simple and dynamically adjustable with the expansion. The underlying network structure can be quite arbitrary, so the routing methods should not rely on any fixed *topology.* More and more emphasis is given to this type of *universal routing* on arbitrary networks (see, for example, [MT90], [I91], [HKR91]). This gives rise to a need of simple *compact routing* methods that are scalable with the growth of networks and independent of any underlying topology. For instance, the C104 Router Chip used in the INMOS T9000 Transputer design [I91] uses one such method called *Interval Routing*, which was introduced in [SK82] and [LT83]. In this paper we survey a few of the available methods.

* This research was partially supported by EC Cooperative Action IC-1000 (project ALTEC: *Algorithms for Future Technologies*) and by the Netherlands Organization for Scientific Research (NWO) under contract NF 62-376 (NFI project ALADDIN: *Algorithmic Aspects of Parallel and Distributed Systems*).

1.1 Communication Model

We shall model the interconnection network as a (finite) *graph* and phrase the terminologies accordingly. Let $G =< V, E >$ be a *connected* graph with *vertex set* V of size n and *edge set* E of size e. Vertices (*nodes*) carry unique *identifiers (addresses)* taken from some *ordered* domain, and edges (*links*) are assumed to be *bidirectional.* We shall concentrate on *distributed* models, where processors have access only to their own local memory and communicate with each other by sending *messages.* Each message contains headers that typically include the *source* and *destination* addresses of the processors. In order to route a message m from processes i to j, a *path* from i to j must be identified to transport m. Traditionally a path of shortest distance or cost is used, but there are other variants. The path information must be stored somehow at each intermediate node to allow progress of the message from source to destination. Typically the necessary information is stored in a *routing table* with n entries, one entry for every possible destination and one table at every node. As the message m arrives at an intermediate node, it checks if it is indeed the intended destination target. If so the message is processed, otherwise the local routing table is consulted for an appropriate link to further relay the message. The message m thus travels in a series of hops until it reaches its final destination. Normally the routing is implemented by a special unit termed the *router* that is associated with each processor. It is thus the function of the router to decide whether it should keep the message for local processing or pass it on further to a neighboring router via an appropriate link. The routing method can be described as follows:

```
procedure SEND(id, dest, m)
{m is a message to be sent from current node id to node dest}
  if id = dest
  then process m
  else
    find the appropriate link x out of node id to be used towards dest
    send m over the link x
    id := the node that receives m over link x
    SEND(id, dest, m)
  end if
end{procedure SEND }
```

Figure 1

The original sender *source* of a message m will then invoke the protocol SEND(*source, dest, m*) to send the message to its proper destination *dest.*

1.2 Compact Routings

With the expansion of the networks, the above method of keeping routing tables becomes untenable, as at each node the table size is $O(n)$ (measured in $\log n$-size words). A more compact way of representing the tables is needed. Now, in practice, the *degree* d (the maximum number of edges incident to a node) of a graph is usually much smaller than the size of the graph. Thus, if we arrange the routing table according to the edges, then we may hope to get a more compact table, with only d rows of entries, though each entry may now consist of more than one node-label. Searching the table for a node-label will be less efficient however, as there is no special ordering in the entries in general.

We can make the tables more compact if we can represent the group of nodes associated with an edge in a simple way. Some kind of a *relation* needs to be established among the nodes that belong to the same group. In this survey, we look at some of the relations that have been used to compactify such groups of nodes. The first relation we study is the *Interval Labeling Scheme.* The idea here is to label the nodes in the graph such that all the nodes belonging to the same edge form a (cyclic) interval modulo n. This is presented in section 2. Section 3 covers a special case of Interval Labeling called *Linear Interval Labeling.* In section 4, we look at *Prefix Labeling*, where nodes having a maximum common prefix in their string addresses belong to the same group. Section 5 introduces *Boolean* routing, which uses certain boolean predicates to define its relations. Finally in section 6 we discuss extensions of the above relations to *multi-label* relations. Throughout we discuss the known results for each scheme and list some open problems.

We assume that the nodes and links do not fail and that messages eventually arrive over the link over which they are sent. We also concentrate mainly on results pertaining to networks with shortest path or minimum cost routing.

A preliminary version of this paper was presented in the Colloquium on Structural Information and Communication Complexity at Carleton University ([LT94]).

2 Interval Routing

The idea behind an Interval Labeling Scheme (ILS) is to group the destination nodes belonging to the same outgoing link in a *cyclic* interval (modulo n). This is done by first labeling all the nodes in the graph with some unique integer in $[0..n-1]$. Each link is then labeled with a unique interval $[a, b)$. Wrap-around of intervals is allowed, so if $a > b$ then $[a, b) = \{a, a+1, ..., n-1, 0, ..., b-1\}$. The set of all such intervals associated with the edges of a node must form a *partition* of the cyclic interval $[0..n)$. The routing protocol is exactly as procedure SEND presented in Figure 1, with line 4 modified to :

```
4       find the link $x$ at node $id$ with label $[a, b)$ such that $dest \in [a, b)$
```

An ILS is *valid* if for all nodes i and j of graph G, messages sent from i to j by means of the procedure SEND eventually reach j. Figure 2 gives an example of a valid ILS. It is not a priori clear that there is a valid ILS for every graph. Of course, one cannot just label the graph arbitrarily, as the route may contain a cycle and any message caught in the cycle on the way will never reach its proper destination. There is an $O(n^2)$ algorithm for checking whether a given ILS is valid ([LT83]). A valid ILS is termed an Interval *Routing* Scheme (*IRS*) for short.

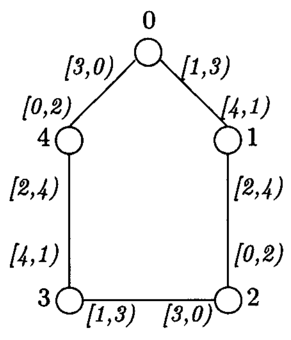

Figure 2

Interval Routing was introduced by Santoro and Khatib [SK82]. They showed that any *tree* and *ring* admits a valid ILS. By constructing a valid ILS on a *spanning tree* of a graph, one can thus impose an IRS on an arbitrary graph also, but then 'most' links are not being used. Van Leeuwen and Tan [LT83], who introduced the term 'interval routing', showed the fundamental result that interval routing can be done on general graphs in such a way that *all links* are actually being used for routing (in *both* directions) and identified further uses of it. In either case however, the IRS constructed is not necessarily *optimal* (shortest path).

As the union of all the disjoint intervals spans the cyclic interval $[0..n)$, we can reduce the size of the routing table further, by listing only the left end-points of the intervals. Also, as there is no overlapping of intervals, the path specified by the IRS is unique. This gives rise to *deterministic* schemes and allows for efficient searching. Sort the interval labels and arrange the tables accordingly to the sorted order. One can then use a *modified binary search* to search quickly the desired link for each *dest*.

We now mention some known results.

2.1 Uniform Cost Links

Suppose by optimum we mean *shortest path*. This case can be considered as assigning positive *uniform cost* to each link and then finding the *minimum cost* routes.

The following graphs (with uniform cost links) have optimum IRS.

Santoro and Khatib [SK82]
Trees
Rings

van Leeuwen and Tan [LT85]

Meshes
Complete bipartite graphs
Complete graphs
Grids with column-wrap-around

Frederickson and Janardan [FJ86]
Graphs whose biconnected components are either outerplanar or K_4

Hofestädt, Klein and Reyzl [HKR91]
'Clos-like' multi-stage networks
Modified Butterfly networks

Fraigniaud and Gavoille [FG94]
Unit circular graphs

Chordal Rings have been studied by Flammini, Gambosi and Salomone [FGS94]. They present some positive and negative results concerning the existence of optimal (shortest path) IRS for these networks.

Not unexpectedly, there also is a negative result.

Ružička [R88] There are networks with *no* optimum (shortest path) IRS.

On the other hand, as mentioned earlier, we know at least the following.

van Leeuwen and Tan [LT83] Every network has a valid (but not necessarily optimum) ILS which uses all edges of the graph and each edge in both directions.

It is not clear exactly what type of graphs admits an optimum IRS.

OPEN PROBLEM: Characterize the graphs with uniform cost links that admit an optimum IRS.

It has been conjectured that it is actually NP-Complete to decide whether a graph has an optimum IRS. Things look more positive if we abandon the restriction to uniform cost links.

2.2 Dynamic Cost Links

Suppose a graph is given and one is allowed to label the nodes appropriately. It is reasonable to assume that the names of the nodes remain *fixed* over time, to avoid confusion, for example. But over time, it is to be expected that the cost of the links may vary (*dynamic cost links*). Assume that the cost of the links are *non-negative* numbers that vary over time. Can one always relabel the links of the graphs accordingly (with no change of the labels of the nodes) to allow for an optimum (minimum cost) IRS?

Frederickson and Janardan [FJ86] came up with a very nice way of characterizing the type of graphs with dynamic cost links that allow optimum IRS. But they did this over a stricter class of IRS. An ILS is termed *strict* if no interval assigned to the edges of a node contains the label of the node itself. Thus, for example, a node with label 5 cannot have an edge that is labeled with the interval $[3, 7)$, as the interval contains the node label. (Note however, that the given example is a legal interval for the regular ILS as we have defined it earlier.) Of course, this is only a conceptual difference and in no way affects the size of the routing table. It can be shown that every graph admits a strict (static) IRS in which each edge is used in at least one direction for routing.

Frederickson and Janardan [FJ86] A graph G with dynamic cost links has an optimum *strict* IRS if and only if G is an *outerplanar* graph.

The corresponding result for a general IRS is almost similar, with one extra graph included in the characterization.

Bakker, van Leeuwen and Tan [BLT94] A graph G with dynamic cost links has an optimum IRS if and only if G is an *outerplanar* graph or K_4.

2.3 Dynamic Cost Links with Dynamic Node Names

In analyzing the adaptability of an optimum IRS in the case of dynamic cost links in a fixed topology, it was assumed that the names of the nodes should remain fixed over time. A different situation arises if we allow the nodes to be *renamed* every time a change in link costs arises (*dynamic node names*). We can expect more topologies to admit an optimum IRS no matter how link costs vary, if we can change the node names after every change in link costs. We note the following result.

Frederickson and Janardan [FJ86] A graph G with dynamic cost links has an optimum strict IRS with dynamic node names if and only if its biconnected components are either outerplanar or K_4.

OPEN PROBLEM: Characterize graphs with dynamic cost links and dynamic node names that have optimum (regular) IRS.

From a networking point of view, the changing edge and node labels may be hard to maintain distributively as the changes in link costs arise.

OPEN PROBLEM: Develop efficient distributed algorithms for maintaining the optimum IRS in the classes of dynamic networks discussed above.

2.4 Other Results and Directions

We now state some further results and areas of research concerning IRS (see also section **6**).

Nondeterministic IRS We introduced ILS as a deterministic scheme by insisting that the intervals assigned to the edges incident to a node do not overlap. In this way we could restrict the routing labels to the left end-points of the intended intervals, leaving their full specification implicit. In practice it might be profitable to assign complete intervals $[a, b)$ to the link instead of just their left end-points and allow intervals to overlap. With this *nondeterministic* scheme, a destination label may belong to several links and a message for a destination j may be routed by sending it over a *random* link whose interval contains j. This gives greater flexibility in distributing the traffic in the network. This is a suggestion for further study mentioned in [LT83]. We state it here as an open problem.

OPEN PROBLEM: Study the above *nondeterministic* IRS's.

Clearly the existence of an *optimum* (shortest path) nondeterministic IRS implies the existence of an optimum IRS in the ordinary sense: whenever two intervals at a node overlap, omit the part of one of the intervals that is covered by the other interval (although this may lead to links without intervals).

3 Linear Interval Routing

A *Linear Interval Labeling Scheme* (LILS) is an ILS in which no intervals 'wraps around'. Thus an interval such as $[5,1)$ is definitely not allowed as a label in an LILS. It is a special case of a *Prefix Labeling Scheme*, to be discussed in the next section. As expected, the class of graphs that have a Linear Interval Routing Scheme (LIRS) is much smaller than IRS. Yet it contains some interesting networks, such as the hypercubes. This, together with the simplicity of the schemes, explains why some implementors have shown considerable interest in linear IRS.

3.1 Uniform Cost Links

The following classes of graphs with uniform cost links have optimum (shortest path) LIRS.

Bakker, van Leeuwen and Tan [BLT91]

Complete Graphs
Hypercubes
d-Dimensional Grids
Rings of size ≤ 4
d-Dimensional Tori $\Pi_{i=1}^{d} d_i$ with $d_i \leq 4$ for each i
Trees which contain no *T-graph* (see Figure 3) as a subgraph

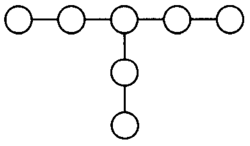

Figure 3

Kranakis, Krizanc and Ravi [KKR93]

Complete r-Partite Graphs[3] $K_{n_1,n_2,\ldots,n_r}$ with $r \geq 2, n_i \geq 1$
The product-graph $\Pi_{i=1}^{n} G_i$ if G_i has an optimum LIRS for each i

Fraigniaud and Gavoille [FG94]
Unit Interval Graphs

It is known that there are many types of interconnection networks that do not have optimum (shortest path) LIRS, such as the Cube-Connected-Cycle, the Star-graph and so on (see [BLT91] and [FG94] for a list). A complete characterization is still elusive.

OPEN PROBLEM: Characterize the graphs with uniform cost links that have optimum LIRS or *strict* LIRS.

3.2 Dynamic Cost Links

Again the situation is more pleasant for networks with dynamic cost links.

Bakker, van Leeuwen and Tan [BLT91] A graph with dynamic cost links has an optimum LIRS if and only if it is a *centipede.*

A *centipede* is defined recursively as follows: it is one of the following two graphs (see Figure 4):

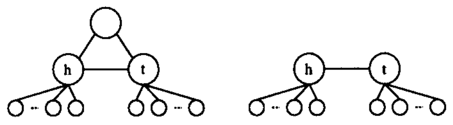

Figure 4

or a centipede joined by another centipede, where by joining we mean that the head (h) of the centipede is identified with the tail (t) of another centipede that is "attached" to it. Figure 5 gives an example of a centipede.

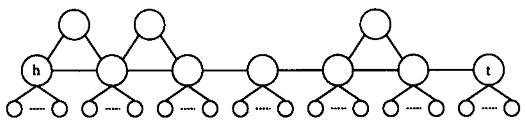

Figure 5

Bakker, van Leeuwen and Tan [BLT94] A graph with dynamic cost links has an optimum strict LIRS iff it is a *line* (a tree of degree 2).

[3] A complete r-partite graph is a graph with r groups of nodes of sizes n_1 through n_r in which every pair of distinct groups of nodes is connected as a complete bipartite graph.

3.3 Dynamic Cost Links and Dynamic Node Names

Similar characterization results for dynamic cost links with dynamic node names are also known for optimum LIRS.

Bakker, van Leeuwen and Tan [BLT94] A graph with dynamic cost links and dynamic node names has an optimum LIRS iff it is a centipede, a $K_3 - Star$ or a $K_4 - Star$. (A $K_n - Star$ graph is the complete graph K_n with zero or more leaf-nodes attached to each node.)

Bakker, van Leeuwen and Tan [BLT94] A graph with dynamic cost links and dynamic node names has an optimum *strict* LIRS iff it is a line or a ring of size 3 or 4.

4 Prefix Routing

A *Prefix Labeling Scheme* (PLS) is based on the notion of *source* or *path* routing. This type of routing assumes that the address in each message explicitly or implicitly specifies a particular path, for example *cuny!mcsun!ruuinf!...*. Thus the address is a sequence of names consisting of strings of characters with suitable separators. When a message is to be relayed, the next name in the sequence, i.e. a *prefix* of the address, is extracted and the routing table is consulted for the next link in the path.

The idea behind PLS is to group nodes together on a link by the *maximum common prefix.* This is done by labeling each node with a string, over some alphabet Σ, which serves as name. Each link also is labeled with a unique string, possibly by ϵ, the *empty* string. When a message arrives at a node with destination *dest*, the routing table is consulted for a link label that is the *maximum length prefix* of the address. For example, if the destination address is *abc* and the link labels available are ϵ, a, ab, ca, then the link label ab will be selected, as it is a prefix of *abc* of maximum length. Of course, each link must be properly labeled so for any address there is always a maximum length prefix. The routing protocol is also similar to procedure SEND (Figure 1), with appropriate modification to line 4.

4 find the link x at *id* with a label that is the maximum length prefix of *dest*

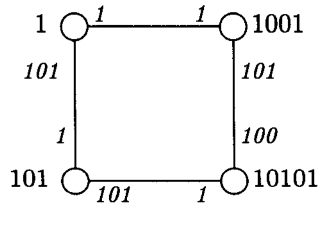

Figure 6

A PLS is *valid* if procedure SEND works correctly, i.e. if it makes all messages arrive at their proper destination eventually. Figure 6 shows a valid PLS. Again for short, we call a valid PLS a Prefix *Routing* Scheme (*PRS*). The feasibility of such a scheme is shown in the following result.

Bakker, van Leeuwen and Tan [BLT90] There is a valid PLS for any *dynamically growing* network. Insertions of links and nodes require an *adaptation cost* of $O(1)$.

We now look at known results on optimality.

4.1 Uniform Cost Links

Unfortunately most graphs do not have an optimum (shortest path) PRS. The following graphs (with uniform cost links) have optimum (shortest path) PRS.

Bakker, van Leeuwen and Tan [BLT90]
Trees
Rings of size ≤ 4
Complete Graphs
Complete Bipartite Graphs
Hypercubes
d-Dimensional Grids
d-Dimensional Tori $\Pi_{i=1}^{d} d_i$ with $d_i \leq 4$ for each i

As with all previous routing schemes, we have the following unsatisfactory situation.

OPEN PROBLEM: Characterize the graphs (with uniform cost) that have optimum (shortest path) PRS.

4.2 Dynamic Cost Links

Again assume that the labels of the nodes remain fixed but allow that the cost of the links may vary non-negatively. In the previous schemes (ILS and LILS) we only considered *fixed* networks, i.e. there were no insertions and deletions of nodes or links. We now consider the dynamic cost case for PRS first.

Bakker, van Leeuwen and Tan [BLT90] A *fixed* (with no insertion or deletion of nodes and links) network with dynamic cost links has an optimum PRS iff its *biconnected components* are of size ≤ 4.

As PRS is a *dynamic* scheme, with arbitrary insertions and deletions of nodes and links allowed (as long as this does not cause the network to be disconnected), we also consider dynamic networks.

Bakker, van Leeuwen and Tan [BLT90] A *dynamic* (with insertion and deletion of nodes and links and no disconnection of the network) network (with dynamic cost links) of more than 4 nodes has an optimum PRS iff it contains no cycle of length > 3.

Nothing much is known about the type of graphs that admit optimum PRS with dynamic cost links and dynamic node names.

OPEN PROBLEMS: Characterize static and dynamic graphs with dynamic cost links and dynamic node names that admit optimum PRS.

4.3 Remarks

A PLS is a naturally dynamic scheme. Insertion and deletion of nodes and links (without disconnecting the network) are easy to do with *constant adaptation cost.* Only the neighboring nodes and links that are affected need to adjust their routing tables. There is no massive update of the whole network involved. Unfortunately, in general the address label can be quite large, up to $O(Diameter \cdot \log(Degree))$ in length. We note also that if the alphabet Σ consists of only one symbol, then PLS is exactly LILS. Thus LILS is a special case of PLS. There are quite a few variants of PLS. For example, the *Path Labeling Scheme*, where the destination address is shortened as it is relayed by stripping off the prefix. (This is fairly similar to the technique used in UUCP.) See [BLT90] for more details.

5 Boolean Routing

In an IRS and a PRS, to route a message we check for a link label that satisfies a certain condition. This relation can be expressed as a *boolean predicate.* In more general terms, this leads to the idea behind *Boolean Labeling Schemes.* In a Boolean Labeling Scheme (BLS) destinations in the network are grouped together to share the same link at a node if they satisfy a certain boolean predicate on their name labels. Strings of bits are assigned as labels to nodes and predicates are attached to the links based on these labels. Elementary boolean functions such as $\neg, \vee, \wedge, \leq, \ldots$ are used to form the predicates. For example, one can represent a LILS by using the predicate $p_i(dest) \equiv (a_i \leq dest) \wedge \neg(a_{i+1} \leq dest)$ at a link l_i that has label $[a_i, a_{i+1})$ in the LILS, for each link l_i. When a message arrives, each predicate is checked in turn until a predicate is found that is satisfied. The appropriate line 4 in procedure SEND (Figure 1) becomes:

```
4    find link $x$ at node $id$ with predicate $p_x$ such that $p_x(dest)$ is **true**
```

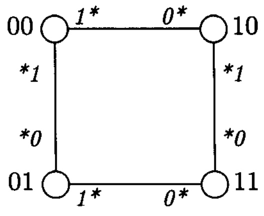

Figure 7

Figure 7 gives an example of a *valid* BLS (a Boolean Routing Scheme or *BRS*), where *0** is the predicate $bit_1(v) = 0$, i.e. check if the *first* bit of v is 0, **1* is the predicate $bit_2(v) = 1$ to check for the *second* bit, and so on. Note that a destination label may satisfy more than one predicate, in which case either one can be used. Thus the above scheme represents all shortest paths in the network. This is an advantage over a *deterministic* ILS; searching becomes less efficient however. Boolean Routing was introduced by Flammini, Gambosi and Salomone [FGS93] to represent schemes with *all pair shortest paths.* They show that with no more than $2 \log n$ bits of strings as labels for the nodes, one can design predicates such that there are *optimum* (all pair shortest paths) BRS for the following networks:

Flammini, Gambosi and Salomone [FGS93]

Rings
Trees
Hypercubes
d-Dimensional Grids
Complete Bipartite Graphs
Complete Graphs

See [FGS93] for the exact predicates used for each type of graph.

6 Multi-Label Routing

All labeling schemes considered so far assumed a unique label per link. If we relax this condition and allow multiple labels (up to n) to be associated with every link, then we have a *multi-label* scheme. The routing decisions at the nodes are made in a straightforward way, where we observe that we have again the choice of allowing the intervals to overlap or not. If we are only interested in optimum schemes, it makes no difference what we impose here, for the same reason as given in section **2.3**. A scheme is termed optimal if and only if all implied source-to-destination paths are optimal.

Essentially, a labeling scheme LS is a *k-LS* if there are at most k labels per link ($k > 0$). Thus the standard ILS is just 1-LS. Multi-label schemes were considered by van Leeuwen and Tan in [LT85] to handle certain graphs that do not have 1-ILS. For example, 1-LILS is quite restrictive: a ring of more than 4 nodes has no optimum 1-LIRS; but a ring of any size has an optimum 2-LIRS. Thus the class of graphs that admits an optimum scheme expands sharply as we increase the admissible number of labels. Of course, trivially any graph has an optimum (n-1)-LS for any scheme LS (ILS, LILS, PLS, BLS): just label a link with all the node labels that need to be sent optimally via this link. This boils down to the traditional complete routing table.

Unfortunately, nothing much is known about the graphs that admit an optimum k-label Routing Scheme (k-RS) with $k > 1$. Flammini, Gambosi and Salomone [FGS95] have shown that it is NP-hard to determine whether a graph with an arbitrary $k > 0$ admits an optimal multi-label IRS (however the k is of order n in the construction). Further results along this line are few. Frederickson and Janardan [FJ86] consider the design of (optimum) k-RS with small k for planar graphs and, more generally, for graphs of a certain genus. We can reformulate the open problem from section **2.1.** as follows.

OPEN PROBLEM: Is there an efficient algorithm to decide, for each k and each graph G with uniform link costs, whether G has an optimum k-RS.

Recent work of Flammini, van Leeuwen and Marchetti-Spaccamela [FLM94] suggests that 'most' networks need a very large k in order to admit an optimum k-RS. This would imply that even multi-label RS can imply a saving over full routing tables *only* for very structured networks, in the case of optimal routing.

6.1 Hierarchies of Graphs

As characterization results on optimum (shortest paths) multi-label RS on graphs with uniform cost links are non-existent, we change our focus and consider the routing problem for graphs with dynamic cost links.

Let $k-\mathcal{RS}$ be the class of graphs with dynamic cost links that have optimum (minimum cost) k-RS. Then obviously we have $k-\mathcal{RS} \subseteq (k+1)-\mathcal{RS}$ for any k-RS and $k-\mathcal{LIRS} \subseteq k-\mathcal{IRS}$. Only a little bit more is known.

Frederickson and Janardan [FJ86]
$k-\mathcal{IRS} \subset (k+1)-\mathcal{IRS}$

Bakker, van Leeuwen and Tan [BLT91]
$k-\mathcal{LIRS} \subset (k+1)-\mathcal{LIRS}$
$k-\mathcal{IRS} \subset (k+1)-\mathcal{LIRS}$
$1-\mathcal{LIRS} \neq 1-\mathcal{IRS}$

Bakker, van Leeuwen and Tan [BLT90]
$k-\mathcal{LIRS} \subset k-\mathcal{PRS}$
$k-\mathcal{IRS} \subseteq (k+1)-\mathcal{PRS}$
$1-\mathcal{IRS} \neq 1-\mathcal{PRS}$

OPEN PROBLEM: Characterize $k-\mathcal{RS}$ for $k>1$ for IRS, LIRS and PRS.

6.2 Further Results and Directions

We now state some further results and areas for further research dealing with *compact* routing.

Near-optimal routing In most of the results in this survey we insisted on optimal routing. This may be overly restrictive considering that, for example, many interesting networks do not admit an optimal ILS (i.e., a 1-label RS). It would be interesting to study the potential of Interval Routing and consider the same characterization questions as above, if it is sufficient to guarantee that the implied source-to-destination routes are within a certain (fixed) factor of optimality. Very little is known here, but the question has been considered in combination with different techniques for *space-efficient* (or: *compact*) routing. For example, Frederickson and Janardan [FJ89] have designed compact routing techniques for planar networks that can guarantee routes that are within 'a small constant factor' (like 3) from optimal. In [FJ90] they have shown that a similar result holds for all networks which have recursive decompositions using 'small' (bounded size) separators. This includes e.g. all series-parallel graphs and all k-outerplanar graphs (for any constant k). A fair number of results is known showing different kinds of trade-off between the quality of the implied routes and the compactness of the routing scheme (measured in bits per node).

Notions of compact routing In section **2** we indicated that Interval Routing is only one of many options one has for defining a compact routing technique. Its simplicity and applicability in useful cases has made it interesting for implementation in scalable processor networks. The variants of the basic scheme that have emerged and that have been surveyed here, clearly suggest the need for a more general combinatorial framework for compact routing. One possible approach would be to use *boolean circuits* for realizing the routing decisions at the network nodes, and to relate the complexity of the routing problem in a network to the complexity of the necessary circuits. It can be shown that every network of n nodes and e edges admits a specification of its optimal routes in these terms in no more than $O(n^2 \log \frac{e}{n})$ bits total [FLM94]. This means, for example, that in every planar network the shortest path information can be represented and employed for optimal routing in an average of $O(n)$ bits per node. The study of other compact schemes seems to open many interesting questions of an algorithmic and graph-theoretic nature.

References

[B91] E. M. Bakker, *Combinatorial Problems in Information Networks and Distributed Data-structuring*, Ph. D. Thesis, Dept. of Computer Science, Utrecht University, 1991.

[BLT94] E. M. Bakker, J. van Leeuwen and R. B. Tan, *manuscript*, 1994.

[BLT90] E. M. Bakker, J. van Leeuwen and R. B. Tan, Prefix Routing Schemes in Dynamic Networks, *Tech. Rep. RUU-CS-90-10*, Dept. of Computer Science, Utrecht University, 1990. Also in: *Computer Networks and ISDN Systems* **26** (1993) pp. 403-421.

[BLT91] E. M. Bakker, J. van Leeuwen and R. B. Tan, Linear Interval Routing Schemes, *Tech. Rep. RUU-CS-91-7*, Dept. of Computer Science, Utrecht University, 1991. Also in: *Algorithms Review* **2** (2), (1991) pp. 45-61.

[FGS93] M. Flammini, G. Gambosi and S. Salomone, Boolean Routing, in: A. Schiper (Ed.), *Distributed Algorithms (WDAG'93), Proceedings 7th International Workshop, Lecture Notes in Computer Science* **725**, Springer-Verlag, 1993, pp. 219-233.

[FGS94] M. Flammini, G. Gambosi and S. Salomone, Interval Labeling Schemes for Chordal Rings, *Tech. Rep. No. 52*, Department of Pure and Applied Mathematics, University of L'Aquila, 1994.

[FGS95] M. Flammini, G. Gambosi and S. Salomone, Interval Routing Schemes, *to appear*, 1995.

[FG94] P. Fraigniaud and C. Gavoille, Interval Routing Schemes, *Tech. Rep. No. 94-04*, Laboratoire de l'Informatique du Parallélisme, Ecole Normale Supérieure de Lyon, 1994.

[FJ86] G. N. Frederickson and R. Janardan, Optimal Message Routing Without Complete Routing Tables, *Proc.* 5th *Annual ACM Symposium on Principles of Distributed Computing*, ACM Press, 1986, pp. 88-97. Also as: Designing Networks with Compact Routing Tables, *Algorithmica* **3** (1988) pp. 171-190.

[FJ89] G. N. Frederickson and R. Janardan, Efficient Message Routing in Planar Networks, *SIAM Journal on Computing* **18** (1989) pp. 843-857.

[FJ90] G. N. Frederickson and R. Janardan, Space Efficient Message Routing in c-Decomposable Networks, *SIAM Journal on Computing* **19** (1990) pp. 164-181.

[FLM94] M. Flammini, J. van Leeuwen and A. Marchetti-Spaccamela, *to appear*, 1994.

[HKR91] H. Hofestädt, A. Klein and E. Reyzl, Performance Benefits from Locally Adaptive Interval Routing in Dynamically Switched Interconnection Networks, in: A. Bode (Ed.) *Distributed Memory Computing, Proc.* 2nd *European Conference, Lecture Notes in Computer Science* **487**, Springer-Verlag, Berlin, 1991, pp. 193-202.

[I91] *The T9000 Transputer Products Overview Manual*, Inmos (1991).

[KKR93] E. Kranakis, D. Krizanc and S. S. Ravi, On Multi-Label Linear Interval Routing Schemes, in: J. van Leeuwen (Ed.), *Graph-Theoretic Concepts in Computer Science (WG'93), Proceedings* 19th *International Workshop, Lecture Notes in Computer Science* **790**, Springer-Verlag, Berlin, 1993, pp. 338-349.

[LT83] J. van Leeuwen and R. B. Tan, Routing with Compact Routing Tables, *Tech. Rep. RUU-CS-83-16*, Dept. of Computer Science, Utrecht University, 1983. Also as: Computer Networks with Compact Routing Tables, in: G. Rozenberg and A. Salomaa (Eds.) *The Book of L*, Springer-Verlag, Berlin, 1986, pp. 298-307.

[LT85] J. van Leeuwen and R. B. Tan, Interval Routing, *Tech. Rep. RUU-CS-85-16*, Dept. of Computer Science, Utrecht University, 1985. Also in: *Computer Journal* **30** (1987) pp. 298-307.

[LT94] J. van Leeuwen and R. Tan, Compact Routing Methods: A Survey, *Proceedings Colloquium on Structural Information and Communication Complexity (SICC '94)*, Carleton University Press, 1994.

[MT90] D. May and P. Thompson, *Transputers and Routers: Components for Concurrent Machines*, Inmos, 1990.

[R88] P. Ružička, On Efficiency of Interval Routing Algorithms, in: M.P. Chytil, L. Janiga, V. Koubek (Eds.), *Mathematical Foundations of Computer Science 1988*, Springer-Verlag *Lecture Notes in Computer Science* **324**, Springer-Verlag, Berlin, 1988, pp. 492-500.

[SK82] N. Santoro and R. Khatib, Routing Without Routing Tables, *Tech. Rep. SCS-TR-6*, School of Computer Science, Carleton University, 1982. Also as: Labelling and Implicit Routing in Networks, *Computer Journal* **28** (1), (1985) pp. 5-8.