Forbidden-Set Labeling on Graphs

B. Courcelle
LaBRI
Bordeaux University
Bruno.Courcelle@labri.fr

C. Gavoille
LaBRI
Bordeaux University
Cyril.Gavoille@labri.fr

M. Kanté
LaBRI
Bordeaux University
Mamadou.Kante@labri.fr

A. Twigg
Thomson Research, Paris
Computer Laboratory, Cambridge University
Andrew.Twigg@cl.cam.ac.uk

Abstract

We describe recent work on a variant of a distance labeling problem in graphs, called the forbidden-set labeling problem. *Given a graph $G = (V, E)$, we wish to assign labels $L(x)$ to vertices and edges of G so that given $\{L(x) \mid x \in X\}$ for any $X \subset V \cup E$ and $L(u), L(v)$ for $u, v \in V$, we can decide if a property holds in the graph $G \setminus X$, or compute a value like the distance between u, v in $G \setminus X$. The problem is motivated by routing in networks where some nodes or edges may fail, or where nodes may decide to route on paths avoiding some 'forbidden' set of nodes or edges.*

Keywords: Planar graphs, connectivity, forbidden-set routing, labeling schemes, compact routing.

1 Introduction

Routing is one of the basic functions in a network, and there is a large body of work on how to efficiently route data on shortest, or approximately shortest paths. Routers have only a limited amount of storage space, so the goal of routing with minimal storage space in each router is an important one.

In this paper we study a variant of shortest-path routing called forbidden-set routing, originally introduced by Feigenbaum et al. [6] in the context of finding stable routes in networks of autonomous systems, and later studied by Twigg [13] and Courcelle et al. [4] in the context of compact routing. The problem is motivated by routing in networks of autonomous systems such as the Internet, where nodes (representing routers) can independently set 'routing policies', assigning costs to paths and so making the shortest path not necessarily the most desirable. Indeed, a policy may specify that node u wishes to use paths avoiding a set $X \subset V$, while v wishes to use paths avoiding $Y \subset V$. In this case we have the forbidden-set routing problem.

2 Background

The idea of *compact routing* has proved to be extremely useful in achieving the goal of low memory requirements for shortest path routing. The underlying idea is due to Santoro and Khatib [9] who described how to assign labels to nodes so that given only the labels for u, v, one can determine quickly if they are adjacent. A compact routing scheme allows one to retrieve from the labels for u and v, the distance $d(u, v)$ and the next-hop on the shortest path to v. A scheme having short labels therefore imposes only small memory requirements on routers.

Many results on compact routing have been obtained, including optimal label sizes for various families of n-vertex graphs. For trees, Gavoille et al. [7] have given an optimal distance labeling scheme using $\Theta(\log^2 n)$ bits per label. For general graphs, it is

known that labels of size $\Omega(n)$ bits are required for exact distances. For stretch-3 distances, Thorup and Zwick [12] have given an almost-optimal scheme using roughly $\sqrt{n}$ bits per label up to poly-logarithmic facators.

For planar graphs, the situation is much better – Thorup [11] has given a scheme using $O(\epsilon^{-1}\log^2 n)$ bits per label that routes on paths of stretch $(1+\epsilon)$ for every $\epsilon > 0$. This has been extended by Gavoille et al. [1] to graphs excluding a fixed minor, and simplified for undirected graphs by Klein [8].

Feigenbaum et al. [6] studied a 'subjective' model of routing where each node u has a cost function c_u that assigns costs to vertices, and the cost of a path P from u is the sum of the costs on P using $c_u(\cdot)$. Motivated by the fact that the current Internet routing algorithm (BGP) tries to construct a tree rooted at each destination, they studied the feasibility of constructing such trees given different cost functions.

A tree T is said to be *stable* if no node u in T can change its parent so that it has a cheaper path to the root (using c_u) without creating a cycle. The motivation for using stable trees is that, without incentive to route on a suboptimal path, a node will always choose the best available path. They showed that even if the costs $c_u(\cdot)$ are in $\{0,1\}$ (so-called *forbidden-set costs* since given a set $X \subset V$, one sets $c_u(v) = 1$ if $v \in X$ and 0 otherwise), then it is NP-complete to decide if there exists a collection of trees, exactly one rooted at each destination, which are stable.

The memory requirements of using routing trees are quite high– to store for each destination the parent node in that tree uses $O(n)$ bits per node, and it is not clear if this can be compressed. Twigg [13] showed that we can do better if we relax the assumption of using stable routing trees– in fact, there exists a compact routing protocol for tree-width-k graphs that routes on lowest-cost paths using forbidden-set costs using labels of size $O(k^2 \log n)$. Furthermore, this protocol always routes data on lowest-cost paths, regardless of whether there exists a collection of stable trees for the given cost functions.

In this paper, we describe the formulation of the forbidden-set routing problem, give some known results and show how to solve a version of the problem on planar graphs, answering an open question in [13].

2.1 Definition

Let $\mathcal{F}$ be a family of graphs, and P be a graph property defined on vertex pairs of a graph of $\mathcal{F}$. For example, *adjacency* can be defined using $P(u,v,H) = \{1\}$ if $uv \in E(H)$ and $\{0\}$ otherwise, and *connectivity* can be defined using $P(u,v,H) = \{1\}$ if u, v are in the same connected component of H and $\{0\}$ otherwise. *Distance* can be defined using $P(u,v,H) = \{d_H(u,v)\}$, and *α-approximate distance* as $P(u,v,H) = [d_H(u,v), \alpha d_H(u,v)]$. The property P can also be a set of vertices, for example a *shortest-path routing* labeling can be defined using

$$P(u,v,H) = \{w \in N_H(u) \mid d_H(w,v) < d_H(u,v)\}$$

where $N_H(u)$ is the set of neighbors of u in the graph H (and similarly for approximate shortest path routing).

Observe that $P(u,v,H)$ is formally described as a set in order to capture in the same definition a wide variety of values including booleans, sets of vertices, or a range of values (say for distance approximation). This saves us from requiring a specific definition for a specific type of value.

Definition 1 *A pair (L,f) is a* forbidden-set P-labeling *of size k for $\mathcal{F}$ if for every $G = (V,E) \in \mathcal{F}$*

1. *$\forall x \in V \cup E$, $L(x,G)$ is a binary string of length $\leqslant k$*
2. *$\forall u, v \in V, X \subseteq V \cup E$, we have*

   $$f(L(u,G), L(v,G), L(X,G)) \in P(u,v,G \setminus X)$$

   where $L(X,G) = \{L(x,G) \mid x \in X\}$.

L is called the labeling*, f the* decoder*, and X the* forbidden-set.

An important feature of our problem is that the labels $L(\cdot)$ must be computed by preprocessing the input graph G *without* the knowledge of the forbidden set X. The labeling must therefore encode the necessary structure of the graph required to answer a query involving any forbidden set X. This generalized the classical P-labeling problem in the sense that it corresponds to the case $X = \varnothing$.

Our goal is to design, for a given family $\mathcal{F}$ and a property P, some efficient and compact forbidden-set P-labeling schemes, i.e., a labeling scheme using short labels whose labeling can be computed efficiently. Intuitively, the label length is a measure of how much information must be transmitted to a vertex in order to update its local data-structure supporting queries for property P. For most practical applications a poly-logarithmic label length (in n) is desirable.

3 Related results

It is easy to see that by storing the adjacency matrix of G in each label, we can obtain a forbidden-set P-labeling scheme for all fixed properties P and family $\mathcal{F}$ of n-vertex graphs using $O(n^2)$-bit labels.

If the property P can be formulated in monadic second-order logic (such as adjacency and connectivity), a result of Courcelle et al. [5] gives labels of size $O(\log n)$ bits for graphs of bounded clique-width. The notion of clique-width is more general than tree-width, since any graph of bounded tree-width has bounded clique-width, but not the other way round (cliques have clique-width 2 but unbounded tree-width). However, the hidden constant in the asymptotic result is a tower of exponentials whose depth depends on the logical formula, and so for even simple properties and small clique-width the bound is huge and impractical.

Twigg [13] showed that for tree-width k graphs, we can solve connectivity using $O(k^2 \log n)$ bits and distance and shortest paths using $O(k^2 \log^2 n)$ bits per label. This was extended by Courcelle et al. [4] to graphs of multiple clique-width, a variant generalizing both clique-width and tree-width (in the sense that the multiple clique-width of a graph is always at most a constant factor larger than both its clique-width and tree-width). For graphs of clique-width k, they get forbidden-set distance labels of size $O(k^2 \log^2 n)$ bits.

Taking $X = \varnothing$ gives the same lower bounds as for distance labeling. Hence for stretch-3 distances in general graphs we have an $\Omega(\sqrt{n})$ bits lower bound [12] and for planar graphs, an $\Omega(n^{1/3})$ bound for exact distances due to Gavoille et al. [7]. Twigg [13] gave an $\Omega(k \log(n/k))$ lower bound for any forbidden-set connectivity scheme that can detect all separators of size at most k, i.e. on k-connected graphs.

4 Trees

As an introduction, we shall describe how to construct forbidden-set labelings for trees. These results are already known [13]. However, this construction will illustrate the flavour of the problems. Assume that $\mathcal{F}$ is the family of n-vertex trees[1], and that P is defined by, for all vertices u, v and forest H:

$$P(u, v, H) = \begin{cases} \{1\} & \text{if } u, v \text{ are in the same tree of } H \\ \{0\} & \text{otherwise.} \end{cases}$$

The question is to design a compact forbidden-set P-labeling scheme for $\mathcal{F}$. For such property P, called connectivity, we shall give a solution with $O()$-bit labels. This can be extended to forbidden-set distance labeling in trees with $O(\log^2 n)$-bit labels. By the lower bounds for connectivity and distance labeling on trees [7], these bounds are tight.

The solution uses the following nearest common ancestor (NCA) labeling result.

Lemma 1 (Alstrup et al. [2]) *Every n-vertex rooted tree has a labeling of its vertices with $O(\log n)$-bit labels such that the label of the nearest-common ancestor between any two vertex u, v can be extracted in constant time from the labels of u and of v only.*

So, let us fix a tree $T \in \mathcal{F}$. We root the tree at an arbitrary vertex, and subdivide each edge by adding a new degree-2 vertex. This forms a tree T' with $2n - 1$ vertices. We denote by $\ell(u)$ the label assigned by the above NCA-labeling on T' (Lemma 1), and by $\mathsf{nca}(\ell(u), \ell(v))$ the label of the nearest-common ancestor between u and v in T'. Note that $\mathsf{nca}(\ell(u), \ell(v))$ is the label of a vertex of T (and not an edge).

For every $x \in V(T) \cup E(T)$, the label of x just consists in $\ell(x)$ which is of size $O(\log n)$. For every subgraph X of T, the decoder simply checks whether there exists at least one vertex or edge x of X that is on the path between u and v in T'. Observe that x corresponds to a vertex of T'. More precisely, we return $\{0\}$ (i.e., u and v are not in the same connected component in $T \setminus X$) if and only if there exists a label

[1] This can be extended to forests by adding a $\log n$ bit field to each vertex label indicating its tree.

$\ell(x)$ of X such that:

$$\mathsf{nca}(\ell(u),\ell(v)) = \mathsf{nca}(\mathsf{nca}(\ell(u),\ell(v)),\ell(x)) \text{ and}$$
$$[\ell(x) = \mathsf{nca}(\ell(x),\ell(u)) \text{ or } \ell(x) = \mathsf{nca}(\ell(x),\ell(v))].$$

For distances, we can augment the labels $\ell(u)$ with labels from a distance labeling scheme for trees, such as the one given by Gavoille et al. [7] with $O(\log^2 n)$ bits per label. The decoder returns $d_{G\setminus X}(u,v) = d_G(u,v)$ if u, v are connected in the forest $G \setminus X$, and infinity otherwise.

5 Planar graphs

In this part we ask the same question P, i.e., connectivity, for the family $\mathcal{F}$ of n-vertex planar graphs. We aim at solving the problem with optimal $O(\log n)$-bit labels. To simplify the presentation, we shall give a simple solution for forbidden edge-set for all planar graphs, and then for arbitrary edge/vertex forbidden-set for 3-connected planar graphs, the general case combines NCA-labeling and is more technical.

5.1 Connectivity with forbidden edges

The key observation is the following simple fact about planar graphs. Given a planar graph G and an embedding of G in the plane, we construct its dual G^* by interchanging vertices and faces, and for each edge e of G we have exactly one edge of G^* between the two faces of G on either side of e.

Fact 2 *A set of edges is a cut set in G iff their corresponding edges in the dual G^* form a cycle.*

The idea is to associate with each edge e of G the coordinates of both endpoints of the corresponding edge in the dual. We then check whether X encircle u but v. For that, we can count the parity of the number intersection between a segment from u to v and all the polygons in X.

More precisely, we construct a planar graph M with 4 types of vertices. The graph M is built from G and from its dual G^* (cf. Fig. 1). So we assume that G and its dual are drawn on the same picture in the usual way (plot a vertex of G^* in the corresponding face of G). The vertices of G are type-V vertices, and the vertices of G^* are type-F vertices. We then subdivide each edge of G with a degree-2 vertex (it forms type-E vertices). The type-E vertices are actually the intersection between edges of G and of G^*. Finally, the new segments linking type-E vertices to type-F are subdivided into degree-2 vertices, called type-EF vertices.

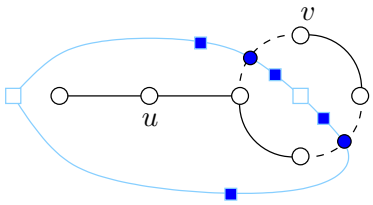


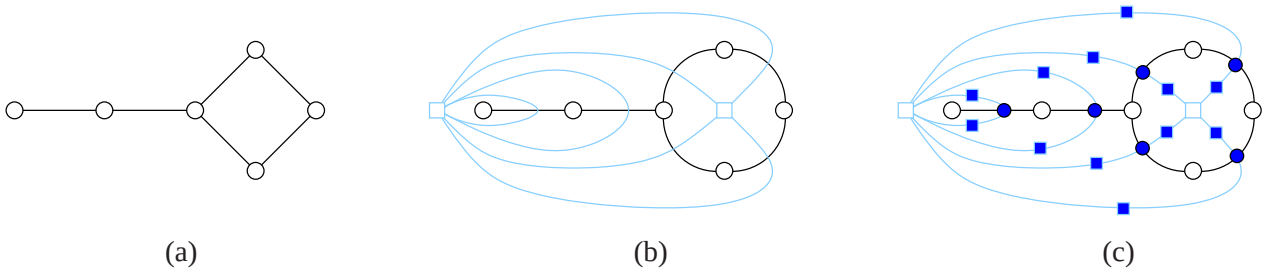
Figure 2. Removing two edges from G (dotted line), disconnecting u from v.

The graph M is planar, simple (no loops and multi-edges) and has at most $3|E(G)| + n + f < 12n$ vertices (by Euler's formula $n + f - e = 2$). A result of Schnyder [10] then implies that M has a Fáry embedding (a straight line drawing) in an $12n \times 12n$ grid. Then, we associate with each edge e of G the coordinates of at most five vertices: the vertices corresponding to the edge e (the degree-2 vertex of type-E), its two adjacent type-EF vertices, and the next type-F vertices (there are 1 or 2 such vertices since e belongs to one (in which case it is a bridge) or two faces of G). See Fig. 2 for an example. The labels contain at most $10\log(12n)$ bits.

Given the labels of u, v and edges in X, we check that the graph formed by the collection of edges in X (and their associated 5 points) contains a cycle separating u, v, i.e. u, v lie on different faces. This can be done in time polynomial in $|X|$ using the planar point location scheme of Arya et al. [3].

5.2 Connectivity with forbidden vertices

Let G_0 be any 3-connected planar graphs with n_0 vertices, and let $X_0 \subseteq V(G_0) \cup E(G_0)$ be any forbidden set containing a mixture of vertices and edges. Observe that removing an edge of G_0 is equivalent in removing a node the graph G obtained from G_0 by subdividing every edge by adding a degree-2 vertex.



(a) (b) (c)

Figure 1. (a) a planar graph G; (b) G and its dual G^* (with loops and multi-edges); (c) the simple planar graph M and its four types of vertices.

Therefore we shall assume that the input graph is a planar graph G with $n = O(n_0)$ vertices, and that $X \subset V(G)$.

We construct from G a planar graph M, which is similar in spirit to the dual G^*. For a face f, denote by $\mathrm{border}(f)$ the sequence of vertices encountered by traversing the border of f. (If a face f contains a cut vertex u, then u appears twice in $\mathrm{border}(f)$, and for a tree $\mathrm{border}(f)$ is an Euler tour.)

For each face f, we add a vertex in f (as in G^*) and add an edge between this vertex and each vertex in $\mathrm{border}(f)$ (with corresponding multiplicity). Each of these new edges is subdivided with a degree-2 vertex to ensure that the resulting graph is simple.

The graph M is planar and triangulated (each face is a triangle). It has at most $2n_0 + e_0 + f_0$ vertices where e_0 and f_0 are the number of edges and faces of any embedding of G_0. By Euler's formula $n_0 + f_0 = 2 + e_0$, and thus $|V(M)| = n_0 + 2e_0 + 2 \leqslant n + 2(3n_0 - 6) + 2 < 7n_0$. A result of Schnyder [10] implies that M has a straight-line embedding in a grid of size $7n_0 \times 7n_0$.

Ignoring the degree-2 vertices used to subdivide edges between f and $\mathrm{border}(f)$, there are two other kinds of vertices in M: the original vertices of G and the face vertices.

Let us define now the following labeling problem, called *common-face labeling*: label the vertices of M such that given the label of any two non-face vertices u, v, return the set $F(u, v)$ of coordinates of the face-vertices that are connected (at distance 2 via the degree-2 vertices) to both u and v, and also the connecting degree-2 vertices. Due to the structure of M, each set $F(u, v)$ is either empty, $\{(x, y)\}$, or $\{(x, y), (x', y')\}$ depending whether u and v lie on 0, 1, or 2 faces of G' because G was 3-connected.

Let us assume that the common-face labeling can be solved with $O(\log n)$-bit labels (we show later how to do this), and let K_u be the label of such a scheme given to a vertex u.

For every vertex u of M, denote by $p(u) \in \mathbb{R}^2$ its grid coordinates in M. Given two points a and b of $\mathbb{R}^2$, denote by $[a, b]$ the segment (straight-line edge) in $\mathbb{R}^2$ between a and b.

We label each $u \in V(G)$ by the pair $(p(u), K_u)$. Given a set X of vertices, define the *barrier* $\mathrm{bar}(X)$ of X as the set of points $\{p(u) \mid u \in X\}$ plus the subset of edges (u, f) of M such that $\exists v \in X$ and a face $f \in M$ such that u, v share f, i.e.

$$\mathrm{bar}(X) = \left(\bigcup_{u \in X} p(u) \right) \cup \left(\bigcup_{\substack{u,v \in X \\ q \in F(u,v)}} [p(u), q] \cup [q, p(v)] \right)$$

We can then use the following fact, which is similar to the cycle-cut duality:

Fact 3 *For a set of vertices $X \subset V(G)$, vertices u and v are in the same connected component of $G \setminus X$ if and only if the points $p(u)$ and $p(v)$ are in the same connected component of $\mathbb{R}^2 \setminus \mathrm{bar}(X)$.*

Clearly, given the set of labels of all the vertices of X, one can compute $\mathrm{bar}(X)$, in particular by the use of sub-labels K_u. This computation takes a time polynomial in $|X|$. The test "u, v are in the same component of $\mathbb{R}^2 \setminus \mathrm{bar}(X)$", once $\mathrm{bar}(X)$ is determined, can be also done in polynomial time in $|X|$.

5.2.1 Common-face labeling

It remains to solve now the problem of deciding when two vertices lie on the same face. Here is a solution. We construct two graphs from G: G_1 and G_2.

The graph G_1 is constructed from G by adding a vertex in each face (called face-vertex) connected to all the vertices around the face, and by removing all the edges of G. We observe that two vertices of G lie on the same face iff there are at distance two in G_1, because G_1 is bipartite and one part correspond to the face-vertex. We orient each edge of G_1 according to a 3-orientation (as in Schnyder's scheme [10]). This is possible since G_1 is clearly planar. Let us denote by $p_i(u)$ the i-th out-neighbor of u in G_1, $i \leqslant 3$ (note that $p_i(u)$ may not exist).

The graph G_2 has vertex-set $V(G)$ and we add an edge between any two distinct vertex u, v if there is a face-vertex f in G_1 such that $u = p_i(f)$ and $v = p_j(f)$ for some $i, j \leqslant 3$. Again, G_2 is planar, since roughly speaking G_2 is similar to G where each face is replaced by a triangle. Let us denote by $q_i(u)$ the i-th out-neighbor of u in G_2, $i \leqslant 3$ (note that $p_i(u)$ may not exist).

A vertex u of G stores in its label the vertices $p_i(u)$, $q_i(u)$, and $p_j(p_i(u))$ for all $i, j \leqslant 3$, that is 15 integers in $[1, O(n)]$. Two vertices u, v lie on the same face iff there are $i, j \leqslant 3$ such that:

- $q_i(u) = v$ or $q_i(v) = u$; or
- $p_i(u) = p_j(v)$; or
- $p_j(p_i(u)) = v$ or $p_j(p_i(v)) = u$.

We can therefore construct the subgraph of the common face graph induced by X, and test if it contains a cycle that separates u and v.

6 Conclusion and Open problems

We have an $O(\log n)$-bit forbidden-set labeling scheme for connectivity, for graphs of bounded clique-width (a side result of [4]) and for planar graphs. It would be good to have an efficient forbidden-set α-approximate distance labeling scheme for planar graphs using a poly-logarithmic number bits per label. The only lower bounds we know for this problem are those for *exact* distance labeling, i.e., $\Omega(n^{1/3})$ for planar graphs.

References

[1] I. Abraham and C. Gavoille. Object location using path separators. In *PODC '06: Proc. of the twenty-fifth annual ACM symposium on Principles of distributed computing*, pages 188–197, 2006.

[2] S. Alstrup, C. Gavoille, H. Kaplan, and T. Rauhe. Nearest common ancestors: a survey and a new distributed algorithm. In *SPAA '02: Proc. of the fourteenth annual ACM symposium on Parallel algorithms and architectures*, pages 258–264, 2002.

[3] S. Arya, T. Malamatos, and D. M. Mount. Entropy-preserving cuttings and space-efficient planar point location. In *SODA '01: Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*, pages 256–261, Philadelphia, PA, USA, 2001. Society for Industrial and Applied Mathematics.

[4] B. Courcelle and A. Twigg. Compact forbidden-set routing. In *Proc. STACS 2007*, volume 4393 of *LNCS*, pages 37–48., 2007.

[5] B. Courcelle and R. Vanicat. Query efficient implementation of graphs of bounded clique-width. *Discrete Appl. Math.*, 131(1):129–150, 2003.

[6] J. Feigenbaum, D. Karger, V. Mirrokni, and R. Sami. Subjective-cost policy routing. In *WINE*, pages 174–183, 2005.

[7] C. Gavoille, D. Peleg, S. Perennes, and R. Raz. Distance labeling in graphs. In *SODA '01: Proc. of the twelfth annual ACM-SIAM symposium on Discrete algorithms*, pages 210–219, 2001.

[8] P. Klein. Preprocessing an undirected planar network to enable fast approximate distance queries. In *SODA '02: Proc. of the thirteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 820–827, 2002.

[9] N. Santoro and R. Khatib. Labelling and implicit routing in networks. *Comput. J.*, 28(1):5–8, 1985.

[10] W. Schnyder. Embedding planar graphs on the grid. In *SODA '90: Proc. of the first annual ACM-SIAM symposium on Discrete algorithms*, pages 138–148, 1990.

[11] M. Thorup. Compact oracles for reachability and approximate distances in planar digraphs. *J. ACM*, 51(6):993–1024, 2004.

[12] M. Thorup and U. Zwick. Compact routing schemes. In *SPAA '01: Proc. of the thirteenth annual ACM symposium on Parallel algorithms and architectures*, pages 1–10, 2001.

[13] A. Twigg. Compact forbidden-set routing (PhD thesis). Technical Report UCAM-CL-TR-678, University of Cambridge, Computer Laboratory, Dec. 2006.