Succinct Representation of Labeled Graphs*

Jérémy Barbay[1], Luca Castelli Aleardi[2], Meng He[1], and J. Ian Munro[1]

[1] Cheriton School of Computer Science, University of Waterloo, Canada, {jbarbay, mhe, imunro}@uwaterloo.ca
[2] LIX (Ecole Polytechnique) and IGM (University of Marne-la-Vallée), France, amturing@lix.polytechnique.fr

Abstract. We consider the problem of designing succinct representations of labeled graphs (we consider vertex labeled planar triangulations, as well as edge labeled planar graphs and the more general k-page graphs) The goal is to support various label queries efficiently, while using an amount of additional space to store the labels which is essentially the information-theoretic minimum. As far as we know, our representations are the first succinct representations of labeled graphs. First, we define three new traversal orders on the vertices of a planar triangulation, allowing to design a succinct representation which supports efficiently labeled based navigation operators. Second, we design a succinct representation for a k-page graph when k is large supporting vertex adjacency in $O(\lg k \lg\lg k)$ time (while previous work uses $O(k)$ time [9, 11]).

1 Introduction

We address the designing of *succinct data structures*, i.e. data structures that occupy space close to the information-theoretic lower bound while supporting efficient navigational operations. The problem of designing space-efficient data structures to represent graphs has recently attracted a great deal of attention [4, 5, 6, 7, 10, 11], because of the number of modern applications often processing large graphs. Previous work focused on succinct graph representations which support efficiently testing the adjacency between two vertices and listing the edges incident to a vertex [4, 5, 11]. Here we address this problem by designing succinct representations of *labeled graphs*, where labels from an alphabet $[\sigma]$ [3] are associated with edges or vertices. These representations efficiently support label-based connectivity queries, such as retrieving the neighbors associated with a given label of a vertex. We investigate three classes of graphs: k-page graphs (used in several areas, such as sorting with parallel stacks [14], fault-tolerant processor arrays design [12], and VLSI design [8]), planar triangulations and graphs (corresponding to the information underlying surface meshes).

Related Work. Jacobson [10] first proposed the problem of representing unlabeled graphs succinctly using the concept of *book embedding*. A *k-page embedding* is a topological embedding of a graph with the vertices along the spine and edges distributed across k pages, each of which is an outerplanar graph. The minimum number of pages, k, for a particular graph has been called the *pagenumber* or *book thickness*. Jacobson showed how to represent a k-page graph using $O(kn)$ bits to support adjacency test in $O(\lg n)$ bit probes, and listing the neighbors of a vertex in $O(d \lg n + k)$ bit probes, where d is the vertex degree. Munro and Raman [11] improved his results on the word RAM model by showing how to represent a graph using asymptotically $2kn + 2m$ bits (m being the number of edges). Others solutions, with different tradeoffs, have been also proposed [9]. As any planar graph can be embedded in at most 4 pages [15], these results can be applied directly to planar graphs allowing to design an encoding requiring $8n+2m$ bits. These space requirements have been improved using a different approach based on the properties of canonical orderings of planar graphs [6, 7]. In particular in the case of a triangulation Chuang *et al.* [7] showed how to represent it using $2m + 2n$ bits. Using an approach based on a partition algorithm, Castelli Aleardi *et al.* [5] further improved previous works obtaining the first optimal succinct encodings for some classes of planar maps (in the case of triangulations $1.62n$ bits are needed, matching the entropy of this class).

* An extended version of this work will be presented to the *Int. Symp. on Algorithms and Computation, ISAAC'07*.
[3] We use $[\sigma]$ to denote the set $\{1, 2, \ldots, \sigma\}$ of references to arbitrary labels, as indeed the alphabet of labels.

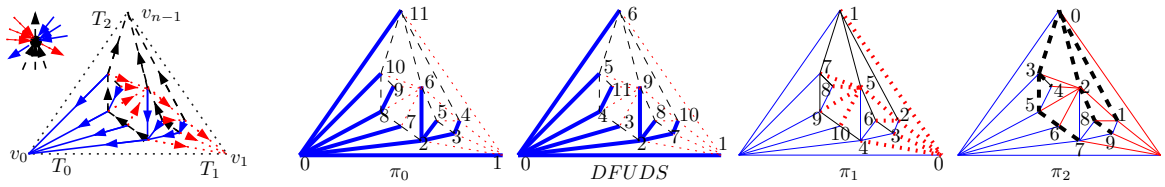

Fig. 1. A planar triangulation induced with one realizer (on the left). The three orders π_0, π_1 and π_2, as well as the order induced by a DFUDS traversal of $\overline{T_0}$ are also shown.

2 Vertex Labeled Planar Triangulations

Realizers and Planar Triangulations A key notion in this work is that of realizers:.

Definition 1 ([13]). *A **realizer** of a planar triangulation $\mathcal{T}$ is a partition of the set of the internal edges into three sets T_0, T_1 and T_2 of directed edges, s.t. for each internal vertex v we have:*

- *vertex v has exactly one outgoing edge in each of the three sets T_0, T_1 and T_2;*
- ***local condition:** the edges incident to v in ccw order are: one outgoing edge in T_0, zero or more incoming edges in T_2, one outgoing edge in T_1, zero or more incoming edges in T_0, one outgoing edge in T_2, and finally zero or more incoming edges in T_1.*

A fundamental property of realizers [13] is that any planar triangulation $\mathcal{T}$ of n vertices, with exterior face (v_0, v_1, v_{n-1}), admits a realizer $R = (T_0, T_1, T_2)$. Moreover each set of edges in T_i is a spanning tree of all internal vertices (see Figure 1 for an example).

One first result consists in defining three orders π_0, π_1 and π_2 on the vertices of a planar triangulation, based on the combinatorial properties of realizers. The major novelty is in the way we show how to efficiently implement the transformation between π_0, π_1 and π_2 (Theorem 1) and how to design a succinct index for labeled graphs (Theorem 2). Since the underlying combinatorial properties we exploit (realizers of triangulations) have been extended also for different classes of graphs, this section if a first step in the design of succinct indexes for labeled more general graphs.

2.1 Three New Traversal Orders on a Planar Triangulation

A key notion in the development of our results is that of three new traversal orders of planar triangulations based on realizers. Let $\mathcal{T}$ be a planar triangulation of n vertices and m edges induces with one realizer (T_0, T_1, T_2). We consider the *canonical spanning tree* of $\mathcal{T}$, denoted $\overline{T}_0$, obtained by adding the edges (v_0, v_1) and (v_0, v_{n-1}) to T_0 (as done in [7]). Each vertex is referred to using its number in *canonical ordering*, which corresponds to the ccw preorder number in $\overline{T}_0$.

Definition 2. ***The zeroth order** π_0 is defined on all the vertices of $\mathcal{T}$ and is simply given by the preorder traversal of $\overline{T_0}$ starting at v_0 in* counter clockwise order *(ccw order).*

***The first order** π_1 is defined on the vertices of $\mathcal{T} \setminus v_0$ and corresponds to a traversal of the edges of T_1 as follows. Perform a preorder traversal of the contour of $\overline{T_0}$ in a ccw manner. During this traversal, when visiting a vertex v, we enumerate consecutively its incident edges $(v, u_1), \ldots, (v, u_i)$ in T_1, where v appears before u_i in π_0. The traversal of the edges of T_1 naturally induces an order on the nodes of T_1: each node (different from v_1) is uniquely associated with its parent edge in T_1.*

***The second order** π_2 is defined on the vertices of $\mathcal{T} \setminus \{v_0, v_1\}$ and can be computed in a similar manner by performing a preorder traversal of T_0 in* clockwise order *(cw order). When visiting in cw order the contour of $\overline{T_0}$, the edges in T_2 incident to a node v are listed consecutively to induce an order on the vertices of T_2.*

Note that the orders π_1 and π_2 do not correspond to previously studied traversal orders on the trees T_1 and T_2, as they are dependent on $\overline{T_0}$ through π_0 (see Fig. 1). The following lemma is crucial:

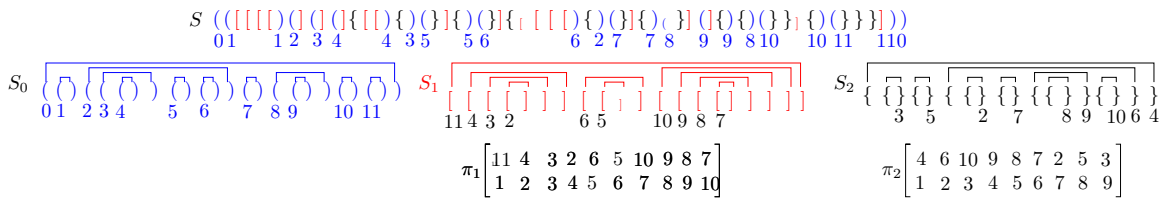

Fig. 2. An example of multiple parenthesis string encoding of the triangulation in Figure 1.

Lemma 1. *For any node x, its children in T_1 (or T_2), listed in ccw order (or cw order), have consecutive numbers in π_1 (or π_2). In the case of $\overline{T_0}$, children are listed consecutively by a* DFUDS *(or* Depth First Unary Degree Sequence *[3]) traversal of $\overline{T_0}$.*

2.2 Representing planar triangulations

We consider the following operations on unlabeled planar triangulations:

- `adjacency`(x, y) says whether vertices x and y are adjacent;
- `degree`(x) returns the degree of vertex x;
- `select_neighbor_ccw`(x, y, r): returns the r^{th} neighbor of vertex x starting from vertex y in ccw order if x and y are adjacent, and ∞ otherwise;
- `rank_neighbor_ccw`(x, y, z): returns the number of neighbors of vertex x between (and including) the vertices y and z in ccw order if y and z are both neighbors of x, and ∞ otherwise.
- $\Pi_j(i)$: given the number of a node v_i in π_0 it returns the number of v_i in π_j;
- $\Pi_j^{-1}(i)$: given the number of a node v_i in π_j it returns its rank in π_0.

Given a planar triangulation $\mathcal{T}$ and a realizer (T_0, T_1, T_2), we encode the three trees T_0, T_1 and T_2 using a multiple parenthesis sequence S of length $2m$. S is obtained by performing a preorder traversal of $\overline{T}_0$ and using three types of parentheses to describe the edges of $\overline{T}_0$ and edges in $\mathcal{T} \setminus \overline{T}_0$. We use parentheses of the first type, namely $'('$ and $')'$, to encode $\overline{T}_0$, while other types of parentheses, $'['$, $']'$, $'\{'$, $'\}'$ to encode the edges of T_1 and T_2 as follows (see Figure 2 for an example).

Let $v_0, \ldots, v_{n-1}$ be the ccw preorder of the vertices of $\overline{T}_0$. Then the string S_0 is simply the balanced parenthesis encoding of the tree $\overline{T}_0$ [11]: S_0 can be obtained by performing a ccw preorder traversal of the contour of $\overline{T}_0$, writing down an opening parenthesis when an edge of $\overline{T}_0$ is traversed for the first time, and a closing parenthesis when it is visited for the second time. During the traversal of $\overline{T}_0$, we insert in S a pair of parentheses $'['$ and $']'$ for each edge of T_1, and a pair of parentheses $'\{'$ and $'\}'$ for each edge in T_2. More precisely, when visiting in ccw order the edges incident to a vertex v_i, we insert:

- A $'['$ for each edge (v_i, v_j) in T_1, where $i < j$, before the parenthesis $')'$ corresponding to v_i;
- A $']'$ for each edge (v_i, v_j) in T_1, where $i < j$, after the parenthesis $'('$ corresponding to v_j;
- A $'\}'$ for each edge (v_i, v_j) in T_2, where $i > j$, after the parenthesis $'('$ corresponding to v_i;
- A $'\{'$ for each edge (v_i, v_j) in T_2, where $i > j$, before the parenthesis $')'$ corresponding to v_j.

Thus S is of length $2m$, consisting of three types of parenthesis. It is easy to observe that substrings S_1 and S_2 are balanced parenthesis sequences of length $2(n-1)$ and $2(n-2)$, respectively.

The following theorem shows how to support the navigational operations on triangulations. While the space used here is a little more than that of [6], the explicit use of the three parenthesis sequences seems crucial to exploiting the realizers to provide an efficient implementation supporting $\Pi_j(i)$ and $\Pi_j^{-1}(i)$ (for $j \in \{1, 2\}$).

Theorem 1. *A planar triangulation $\mathcal{T}$ of n vertices and m edges can be represented using $2m \log_2 6 + o(m)$ bits to support* `adjacency`, `degree`, `select_neighbor_ccw`, `rank_neighbor_ccw` *as well as the $\Pi_j(i)$ and $\Pi_j^{-1}(i)$ operators (for $j \in \{1, 2\}$) in $O(1)$ time.*

Vertex Labeled Planar Triangulations In addition to unlabeled operators, we present a set of operators that allow efficient navigation on a labeled graph :

- `node_label`(i, x), returns the i^{th} label associated to vertex x;
- `lab_degree`(α, x), the number of neighbors of vertex x in G associated with label α;
- `lab_select_ccw`(α, x, y, r), the r^{th} vertex labeled α among neighbors of vertex x after vertex y in ccw order, if y is a neighbor of x, and ∞ otherwise;
- `lab_rank_ccw`(α, x, y, z): the number of the neighbors of vertex x labeled α between vertices y and z in ccw order if y and z are neighbors of x, and ∞ otherwise.

As Barbay *et al.* did for multi-labeled trees [2], we reduce the designing of a succinct representation of a labeled triangulation to the support of succinct indexes for binary relations.

Observe that the sequence of the vertices referred by their numbers in three different orders, namely the `DFUDS` order of the nodes of $\overline{T_0}$, π_1 and π_2, form three binary relations, R_0, R_1 and R_2, with their associated labels. Using Theorem 1 for the conversion between the ranks of the vertices between π_0, π_1 and π_2 and adopting the approach of Barbay *et al.* [1] to encode each binary relation using $t \lg \sigma + O(t)$ bits in order to support `node_label` in $O(1)$ time, we can prove

Theorem 2. *A multi-labeled planar triangulation $\mathcal{T}$ of n vertices, associated with σ labels in t pairs ($t \geq n$) can be represented using $t \lg \sigma + t \cdot o(\lg \sigma)$ bits to support* `node_label` *in $O(1)$ time, and* `lab_degree`, `lab_select_ccw` *and* `lab_rank_ccw` *in $O((\lg \lg \lg \sigma)^2 \lg \lg \sigma)$ time.*

3 Edge Labeled Graphs with Pagenumber k

Graphs with Pagenumber k for large k. In this section, on unlabeled graphs with page number k, we consider the operators `adjacency` and `degree` defined in Section 2.2, and the operator `neighbors`(x), returning the neighbors of x. Previous results on succinctly representing k-page graphs [9, 11] support `adjacency` in $O(k)$ time. The lower-order term in the space cost of the result of Gavoille and Hanusse [9] is $o(km)$, which is dominant when k is large. Thus previous results mainly deal with the case when k is small. We prove the following

Theorem 3. *A k-page graph of n vertices and m edges can be represented using $n + 2m \lg k + o(m \lg k)$ bits to support* `adjacency` *in $O(\lg k \lg \lg k)$ time,* `degree` *in $O(1)$ time, and* `neighbors`(x) *in $O(d(x) \lg \lg k)$ time where $d(x)$ is the degree of x. Alternatively, it can be represented in $n + (2 + \epsilon)m \lg k + o(m \lg k)$ bits to support* `adjacency` *in $O(\lg k)$ time,* `degree` *in $O(1)$ time, and* `neighbors`(x) *in $O(d(x))$ time, for any constant ϵ such that $0 < \epsilon < 1$.*

Edge Labeled Graphs with Pagenumber k. We consider the following operations:

- `lab_adjacency`(α, x, y), whether there is an edge labeled α between vertices x and y;
- `lab_degree_edge`(α, x), the number of edges of vertex x that are labeled α;
- `lab_edges`(α, x), the edges of vertex x that are labeled α.

We first show how to design succinct representation for an edge labeled graph with one page.

Lemma 2. *An outerplanar graph of n vertices and m edges in which the edges are associated with σ labels in t pairs ($t \geq n$) can be represented using $n + t(\lg \sigma + o(\lg \sigma))$ bits to support* `lab_adjacency` *and* `lab_degree_edge` *in $O(\lg \lg \sigma (\lg \lg \lg \sigma)^2)$ time, and* `lab_edges`(α, x) *in $O($*`lab_degree_edge`$(\alpha, x) \lg \lg \sigma \lg \lg \lg \sigma)$ time.*

To support an edge labeled graph with k pages, we can use Lemma 2 to represent each page and combine all the pages represented in this way to support navigational operations. Alternatively, we can use the result of Theorem 3 and a similar approach to Lemma 2 to achieve a different tradeoff to improve the time efficiency when k is large.

Theorem 4. *A k-page graph of n vertices and m edges in which the edges are associated with σ labels in t pairs ($t \geq n$) can be represented using $kn + t(\lg\sigma + o(\lg\sigma)$ bits to support* `lab_adjacency` *and* `lab_degree_edge` *in $O(k \lg\lg\sigma(\lg\lg\lg\sigma)^2)$ time, and* `lab_edges`(α, x) *in $O($*`lab_degree_edge`$(\alpha, x) \lg\lg\sigma \lg\lg\lg\sigma + k)$ time. Alternatively, it can be represented using $n + (2m + \epsilon)\lg k + o(m \lg k) + m(\lg\sigma + o(\lg\sigma))$ bits to support* `lab_adjacency` *in $O(\lg k \lg\lg\sigma(\lg\lg\lg\sigma)^2)$ time,* `lab_degree_edge` *in $O(\lg\lg\sigma(\lg\lg\lg\sigma)^2)$ time, and* `lab_edges`(α, x) *in $O($*`lab_degree_edge`$(\alpha, x) \lg\lg\sigma \lg\lg\lg\sigma)$ time, for any constant ϵ such that $0 < \epsilon < 1$.*

Corollary 1. *An edge-labeled planar graph of n vertices and m edges in which the edges are associated with σ labels in t pairs ($t \geq n$) can be represented using $4n + t(\lg\sigma + o(\lg\sigma))$ bits to support* `lab_adjacency` *and* `lab_degree_edge` *in $O(\lg\lg\sigma(\lg\lg\lg\sigma)^2)$ time, and* `lab_edges`(α, x) *in $O($*`lab_degree_edge`$(\alpha, x) \lg\lg\sigma \lg\lg\lg\sigma)$ time.*

4 Concluding Remarks

We present in this work a framework for succinctly representing the properties of the graphs in the form of labels. We expect that our approach can be extended to support other types of graphs, which is an open research topic. In particular the fact that combinatorial properties of realizers have been generalized to more general planar graphs (e.g. 3-connected graphs and quadrangulations) it should be possible to take advantage of our new traversal orders on the vertices, as in the case of triangulations. Another open problem is to design succinct representation of vertex labeled graphs with pagenumber k.

References

[1] J. Barbay, A. Golynski, J. I. Munro, and S. S. Rao. Adaptive searching in succinctly encoded binary relations and tree-structured documents. In *Proc. of CPM*, pages 24–35. LNCS 4009, 2006.

[2] J. Barbay, M. He, J. I. Munro, and S. S. Rao. Succinct indexes for strings, binary relations and multi-labeled trees. In *SODA*, pages 680–689, 2007.

[3] D. Benoit, E. D. Demaine, J. I. Munro, R. Raman, V. Raman, and S. S. Rao. Representing trees of higher degree. *Algorithmica*, 43(4):275–292, 2005.

[4] L. Castelli-Aleardi, O. Devillers, and G. Schaeffer. Succinct representation of triangulations with a boundary. In *Proc. 9th Workshop on Algorithms and Data Structures*, volume 3608 of *LNCS*, pages 134–145. Springer, 2005.

[5] L. Castelli-Aleardi, O. Devillers, and G. Schaeffer. Optimal succinct representations of planar maps. In *Proc. of 22nd ACM Annual Symposium on Computational Geometry (SoCG)*, pages 309–318, 2006.

[6] Y.-T. Chiang, C.-C. Lin, and H.-I. Lu. Orderly spanning trees with applications to graph encoding and graph drawing. *SODA*, pages 506–515, 2001.

[7] R.-N. Chuang, A. Garg, X. He, M.-Y. Kao, and H.-I. Lu. Compact encodings of planar graphs via canonical orderings and multiple parentheses. *Automata, Laguages and Programming*, pages 118–129, 1998.

[8] F. R. K. Chung, F. T. Leighton, and A. L. Rosenberg. Embedding graphs in books: a layout problem with applications to VLSI design. *SIAM J. Algebr. Discrete Methods*, 8(1):216–227, 1987.

[9] C. Gavoille and N. Hanusse. On compact encoding of pagenumber k graphs. *Discrete Mathematics & Theoretical Computer Science*, 2004. To appear.

[10] G. Jacobson. Space-efficient static trees and graphs. In *FOCS*, pages 549–554, 1989.

[11] J. I. Munro and V. Raman. Succinct representation of balanced parentheses and static trees. *SIAM J. Comput.*, 31(3):762–776, 2001.

[12] A. L. Rosenberg. The DIOGENES design methodology: toward automatic physical layout. In M. Cosnard, editor, *Parallel algorithms and architectures*, pages 335–348. 1986.

[13] W. Schnyder. Embedding planar graphs on the grid. In *SODA*, pages 138–148, 1990.

[14] R. E. Tarjan. Sorting using networks of queues and stacks. *J. Assoc. Comput. Mach.*, 19:341–346, 1972.

[15] M. Yannakakis. Four pages are necessary and sufficient for planar graphs. In *STOC*, pages 104–108, 1986.