Optimal Graph Orientation with Storage Applications

Oswin Aichholzer
Franz Aurenhammer
Institute for Theoretical Computer Science
Graz University of Technology
Klosterwiesgasse 32/2, A-8010 Graz, Austria
e-mail: {oaich,auren}@igi.tu-graz.ac.at

Günter Rote
Institut für Mathematik
Technische Universität Graz
Steyrergasse 30
A-8010 Graz, Austria
e-mail: rote@ftug.dnet.tu-graz.ac.at

Abstract

We show that the edges of a graph with maximum edge density d can always be oriented such that each vertex has in-degree at most d. Hence, for arbitrary graphs, edges can always be assigned to incident vertices as uniformly as possible. For example, in-degree 3 is achieved for planar graphs. This immediately gives a space-optimal data structure that answers edge membership queries in a maximum edge density-d graph in $O(\log d)$ time.

Keywords Graph orientation, edge density, Hall condition, balanced adjacency lists, edge membership queries

1 The Theorem

Let G be an undirected graph with n vertices and m edges. The parameter $\delta(G) = \frac{m}{n}$ is commonly called the *edge density* of G. The *maximum (edge) density* is the smallest integer d such that the edge density of any subgraph of G does not exceed d. More precisely,

$$d = \lceil \max\{\delta(G') \mid G' \text{ is a subgraph of } G\} \rceil .$$

For example, $d \leq 1$ for trees, $d \leq 3$ for planar graphs, $d = \lceil \frac{1}{2} \log_2 n \rceil$ for hypercubes [GG,AH], and $d \leq \lceil \frac{1}{2}(c-1) \log_c n \rceil$ for general Hamming graphs (cartesian products of cliques of size at most c, and their isometric subgraphs) [AFISW]. The complete graph K_n realizes the maximum $d = \lceil \frac{(n-1)}{2} \rceil$ over all graphs with n vertices. It is obvious that the number of edges in a graph with maximum edge density d can be as large as dn. Note also that a graph with low maximum density may still have vertices with high degree.

An *orientation* of a graph G is a directed graph which is obtained from G by selecting, for each edge of G, one of the two corresponding directed arcs. The purpose of this note is to show that every graph can be oriented in the following optimal way.

Theorem 1 *Let G be a graph with maximum edge density d. There exists an orientation of G such that each vertex has in-degree at most d.*

Proof. Let $G = (V, E)$, and let V_d denote the multiset that contains each vertex in V exactly d times. Construct a bipartite graph, B, between E and V_d by drawing an edge in B between e in E and each copy of $v \in V$ in V_d whenever v and e are incident in G. We will show that B contains a perfect matching. This matching immediately implies that — in G — each vertex gets assigned at most d of its incident edges (and no others).

To prove the existence of a perfect matching in B we show that B fulfills the Hall condition of the marriage theorem (see, e.g., [B]). This condition requires that, in B, each subset $E' \subset E$ has at least $|E'|$ adjacent vertices.

Let $|E'| = k$. The graph G has maximum density d, so any k of its edges are incident to at least $\frac{k}{d}$ of its vertices. By construction, V_d contains d copies of each such vertex, hence these k edges are incident to at least k vertices in V_d. This proves the Hall condition and hence the theorem. □

An orientation of G with vertex in-degrees of at most p will be called a *p-orientation* of G. The d-orientation predicted in Theorem 1 is best possible as, by the pigeon-hole principle, a graph having d times more edges than vertices cannot allow a $(d-1)$-orientation. In other words, Theorem 1 assigns edges to incident vertices as uniformly as possible, and thus optimally solves an interesting assignment problem.

Theorem 1 turns out to be related to a result by Nash-Williams [NW] on the *arboricity* of graphs, that is, the number of edge-disjoint spanning forests needed to cover a graph. In fact, a proof of Theorem 1 can be obtained by orienting these spanning forests accordingly. We were not aware of the result in [NW] when we found the more direct proof above based on the Hall theorem. Also, we recently learned that a general theorem on matroid partitioning by Edmonds and Fulkerson (see e.g. Lawler [L]) implies both the Hall theorem and Theorem 1. Different results on graph orientation have been obtained by Frank [F]. He gives conditions for a graph to allow an orientation with in-degrees *at least* some constant, which can be chosen individually for each vertex.

The proof of Theorem 1 is non-algorithmic, but we can find an optimal orientation for a given graph $G = (V, E)$ by computing maximum flows in the bipartite graph that reflects incidences between E and V. More precisely, when given a value d, one can decide in this way whether G has a d-orientation, and compute one if it exists. With the flow algorithm of Hopcroft and Karp [HK] (see also [T]) a decision takes $O(m\sqrt{m})$ time and $O(m)$ space if $|E| = m$. Standard exponential search is then used to determine the optimum value of d in $O(\log d)$ decisions. Note that, by optimality of Theorem 1, d actually is the maximum density of G. Hence the maximum density of a graph is determined as a byproduct.

If we do not insist on optimality but rather are content with a $2d$-orientation, a linear-time algorithm is possible. It relies on the following simple fact.

Lemma 1 *Each subgraph of a graph with maximum edge density d contains a vertex of degree at most $2d$.*

The algorithm now repeatedly finds a vertex v of small degree, orients the at most $2d$ incident edges towards v, and then removes v and its edges. It starts with the originally given graph G and, at any point in time, handles a subgraph of G. So by Lemma 1 it is evident that a $2d$-orientation of G is computed.

The algorithm maintains two lists L and L' of vertices, namely of current degree at most, and larger than, $2d$, respectively. In each step, a vertex of L is chosen and removed. The adjacent vertices have their degrees updated, and are moved from L' to L if necessary. It is clear that overall linear time is spent.

Theorem 2 *Let G be a (connected) graph with m edges and (unknown) maximum edge density d. A d-orientation of G can be computed in $O(m\sqrt{m}\log d)$ time and $O(m)$ space. Moreover, optimal $O(m)$ time and space suffices for computing a $2d$-orientation of G.*

2 Applications

We have shown that the edges of a graph can be assigned to their incident vertices in a uniform manner, and that optimal and near-optimal assignments can be found quickly. These results are exploited in [AARX] in a proof on length bounds for the greedy triangulation of a set of points in the plane.

Another application – though obvious but interesting – is a scheme for storing graphs: An *optimal adjacency list representation* where (1) each edge (x, y) is stored exactly once, either in the list of x or in the list of y, and (2) the length of the longest list is minimum.

For a graph with maximum density d, list lengths do not exceed d. This reduces the search time per list to $O(d)$, or to $O(\log d)$ if binary search is used. Moreover, if m is not much smaller than dn, an adjacency list representation with a fixed upper bound for list lengths may be simpler to manage than arbitrary-length lists.

Using an optimal adjacency list representation, edge membership queries can be answered quickly. To test whether edge xy is contained in the graph, we simply search for y in the list of x, and for x in the list of y. If required, lists can be sorted using bucket sort in $O(d)$ time each.

Theorem 3 *Let G be a graph with m edges and let d be its maximum edge density. There is an adjacency list representation for G with optimal preprocessing time (and storage) of $O(m)$ which answers edge membership queries in G in $O(\log d)$ time. Optimal list lengths can be achieved in $O(m\sqrt{m}\log d)$ preprocessing time.*

Theorem 3 will speed up graph algorithms which merely are based on edge tests. Although the result already follows from the simple trick of removing low-degree vertices in graphs (which is commonly used for planar graphs), it has not been paid attention in the literature. In particular, the achievement of minimal list lengths seems to be new.

Possible geometric applications of Theorem 3 include edge tests for polyhedra in 3-space. For instance, whether two points in a 3-dimensional point set determine an edge of the convex hull can be decided in constant time.

Theorems 1 and 2 apply particularly nicely to planar graphs which, by Euler's theorem, have maximum density 3 or less. In fact, the existence of a 3-orientation can be shown directly in this case. This was also observed by Chrobak and Eppstein [CE] who proved, among other related results, that each planar graph has a 5-orientation which is acyclic.

The induction proof below translates into a simple $O(n)$-time algorithm for 3-orienting planar n-vertex graphs. This improves over the $O(n\sqrt{n})$-time method based on flow algorithms. We will argue about graph triangulations, as being the planar graphs with the largest number of edges.

Theorem 4 *Every planar triangulation T has an orientation with in-degree ≤ 3. Moreover, if T has at least 6 vertices, then one can simultaneously guarantee that each vertex has in-degree ≥ 2.*

Proof. If T has at most 3 vertices then the assertion is certainly true. Assume now inductively that every triangulation with at most $n-1$ vertices has a 3-orientation, and let T have n vertices. It is well known that T always has a vertex v of degree $k \leq 5$; compare also Lemma 1. Removal of v (and re-triangulation of the k-cycle if $k \geq 4$) yields a triangulation T' which, by assumption, admits a 3-orientation. We show that, when putting back v again, T can be locally oriented as required.

If $k = 3$ we simply orient the k edges towards v. If $k = 4$ then there is an edge xy in T' that re-triangulates the 4-cycle and points to, say, x. Hence, in T, we can orient vx towards

x and the remaining three edges of v towards v. If $k = 5$ there are two edges xy and xz in T' that re-triangulate the 5-cycle. These point to vertices in one of the three ways displayed in Figure 1.

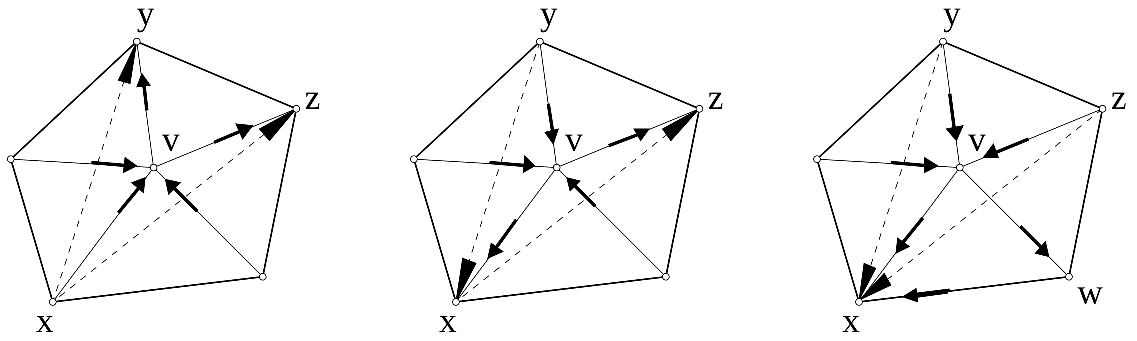

Figure 1: Triangulating and re-orienting 5-cycles

In the first two cases, the two edges point to y and z, or x and z. In T, we can thus orient two edges of v towards these vertices, and the remaining three towards v. In the third case, both edges point to x. But now at most one edge of x in the cycle can point to x in addition in T'. The other one, xw, points to w. Hence we can re-orient xw towards x. In addition, we orient vx towards x, vw towards w, and the three remaining edges of v towards v. This proves the first part of the theorem.

Note that, by construction, orienting the edges of a inserted vertex does not change the in-degree of present vertices. Furthermore, each inserted vertex gets in-degree 3. So, if T' can be oriented with minimum in-degree i, the same is true for T. Indeed, one can always achieve $i = 2$ if T' has 6 vertices, as can be seen by checking all such triangulations. The second part of the theorem follows by induction. $\square$

In order to actually construct a 3-orientation for a planar input graph G, we merely follow the lines of the proof above. Generally, G will be a subgraph of a triangulation rather than a complete one. Let us assume that G is connected, as each connected component can be treated separately, otherwise. We complete G in linear time by first invoking one of the known embedding algorithms (for example, [MM, CNAO]) and then triangulating the faces of the embedded graph.

The problem of *embedding* a planar graph consists of representing it as a *combinatorial map*, cf. [M]. This is a data structure which stores the edges incident to every vertex as a doubly-linked list in the circular order in which they appear in the embedding. This structure allows also to scan the edges bounding any given face, starting from any given boundary edge. See also the quad-edge data structure in [GS] for storing planar subdivisions.

Once a combinatorial embedding is known, all faces can be triangulated in linear time. For such a (combinatorial) planar triangulation one could then go on to compute a straight-line embedding with vertices on an $(n-1) \times (n-1)$ grid in linear time, see [S, FPP]. The grid embedding procedure is not needed for our orientation algorithm, however.

Now we repeatedly find and remove a vertex v of G of degree $k \leq 5$. Instead of orienting the k edges towards v (as did the 5-orientation algorithm of Section 1), we insert auxiliary edges if $k \geq 4$, in order to re-triangulate v's cycle. These local changes in the graph structure are stored in a chronological manner. Having ended up with a triangulation T_0 of constant size (which we 3-orient directly), we play the game in a backward fashion. Thereby, when putting back a vertex into the current triangulation, its at most 5 edges are oriented with the aid of the auxiliary edges that are removed. Re-orientation takes place as detailed in the proof above.

At last, each edge in the final triangulation which does not belong to the original graph G is removed, thereby decreasing the in-degree of one of its vertices.

Theorem 5 *A given planar n-vertex graph can be 3-oriented in $O(n)$ time and space.*

Chrobak and Eppstein [CE] showed that a linear time algorithm is possible even without the primary step of embedding the planar graph. Their algorithm solely is based on combinatorial properties of planar graphs but uses a more complicated orientation procedure.

We get the following optimal data structure out of Theorem 5:

Corollary 1 *A planar n-vertex graph can be stored in $O(n)$ time and space such that edge tests can be performed in constant time (at most 6 comparisons).*

Acknowledgements The authors would like to thank Raimund Seidel for pointing out the relation to graph arboricities, Gerhard Wöginger for drawing our attention to the paper by Chrobak and Eppstein, and Refael Hassin for mentioning the work by Frank.

This research was supported by the Spezialforschungsbereich F 003, *Optimierung und Kontrolle*. Research of Oswin Aichholzer was partially supported by the Jubiläumsfond der Österreichischen Nationalbank.

References

[AARX] F. Aurenhammer, O. Aichholzer, G. Rote, Y.-F. Xu, *Constant-level greedy triangulations approximate the MWT well*, submitted to J. Combin. Optimization.

[AH] F. Aurenhammer and J. Hagauer, *Recognizing binary Hamming graphs in $O(n^2 \log n)$ time*, Mathematical Systems Theory 28 (1995), 387–395.

[AFISW] F. Aurenhammer, M. Formann, R.M. Idury, A.A. Schäffer, F. Wagner, *Faster isometric embedding in products of complete graphs*, Discrete Applied Mathematics 52 (1994), 17–28.

[B] B. Bollobás, *Graph Theory. An Introductory Course*, Springer Verlag, 1979.

[CE] M. Chrobak, D. Eppstein, *Planar orientations with low out-degree and compaction of adjacency matrices*, Theoret. Computer Sci. 86 (1991), 243–266.

[CNAO] N. Chiba, T. Nishizeki, S. Abe, T. Osawa, *A linear algorithm for embedding planar graphs using PQ-trees*, J. Computer Syst. Sci. 30 (1985), 54–76.

[F] A. Frank, *On the orientation of graphs*, J. Combin. Theory B 28 (1980), 251–261.

[FPP] H. de Fraysseix, J. Pach, R. Pollack, *How to draw a planar graph on a grid*, Combinatorica 10 (1991), 41–51.

[GG] M.R. Garey and R.L. Graham, *On cubical graphs*, J. Combinatorial Theory B 18 (1975), 84–95.

[GS] L.J. Guibas and J. Stolfi, *Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams*, ACM Trans. Graph. 4 (1985), 74–123.

[HK] J. E. Hopcroft and R. Karp, *An $n^{\frac{5}{2}}$ algorithm for maximum matchings in bipartite graphs*, SIAM J. Comput. 2 (1973), 225–231.

[L] E. Lawler, *Combinatorial Optimization: Networks and Matroids*, Holt, Rinehart and Winston, New York, 1976.

[M] K. Mehlhorn, *Data Structures and Efficient Algorithms*, Springer-Verlag 1984, vol.2.

[MM] K. Mehlhorn and P. Mutzel, *On the embedding phase of the Hopcroft and Tarjan planarity testing algorithm*, Algorithmica, to appear.

[NW] C. Nash-Williams, *Edge-disjoint spanning trees of finite graphs*, J. London Math. Soc. 36 (1961), 445–450.

[S] W. Schnyder, *Embedding planar graphs on the grid*, Proc. 1st ACM-SIAM Symp. Discr. Alg. (SODA), San Francisco, 1990, 138–148,

[T] R. E. Tarjan, *Data Structures and Network Algorithms*, SIAM Press, Philadelphia 1987.