Linear-Time Compression of Bounded-Genus Graphs into Information-Theoretically Optimal Number of Bits

Hsueh-I Lu*

1 Introduction

This extended abstract summarizes a new result for the *graph compression* problem, addressing how to *compress* a graph G into a binary string Z with the requirement that Z can be *decoded* to recover G. Graph compression finds important applications in 3D model compression of Computer Graphics [12, 17–20] and compact routing table of Computer Networks [7]. For brevity, let a *π-graph* stand for a graph with property π. The *information-theoretically optimal* number of bits required to represent an n-node π-graph is $\lceil \log_2 N_\pi(n) \rceil$, where $N_\pi(n)$ is the number of distinct n-node π-graphs. Although determining or approximating the close forms of $N_\pi(n)$ for nontrivial classes of π is challenging, we provide a linear-time methodology for graph compression schemes that are *information-theoretically optimal with respect to continuous super-additive functions* (abbreviated as *optimal* for the rest of the extended abstract).[1] Specifically, if π satisfies certain properties, then we can compress any n-node m-edge π-graph G into a binary string Z such that G and Z can be computed from each other in $O(m+n)$ time, and that the bit count of Z is at most $\beta(n)+o(\beta(n))$ for any continuous super-additive function $\beta(n)$ with $\log_2 N_\pi(n) \leq \beta(n)+o(\beta(n))$. Our methodology is applicable to general classes of graphs; this extended abstract focuses on graphs with sublinear genus.[2] For example, if the input n-node π-graph G is equipped with an embedding on its genus surface, which is a reasonable assumption for graphs arising from 3D model compression, then our methodology is applicable to any π satisfying the following statements:

F1. The genus of any n-node π-graph is $o(\frac{n}{\log^2 n})$;
F2. Any subgraph of a π-graph remains a π-graph;
F3. $\log N_\pi(n) = \Omega(n)$;[3] and
F4. There is an integer $k = O(1)$ such that it takes $O(n)$ time to determine whether an $O(\log^{(k)} n)$-node graph satisfies property π.[4]

For instance, π can be the property of being a directed 3-colorable simple graph with genus no more than ten.[5] The result is a novel application of planarization algorithm for bounded-genus graphs [5] and separator decomposition tree of planar graphs [9]. Rooted trees were the only known nontrivial class of graphs with linear-time optimal coding schemes. He, Kao, and Lu [11] provided $O(n \log n)$-time compression schemes for planar and plane graphs that are optimal. Our results significantly enlarge the classes of graphs that admit efficient optimal compression schemes. More results on various versions of graph compression problems or succinct graph representations can be found in [1–4, 6, 8, 10, 14, 15] and the references therein.

2 Compressing bounded-genus graphs

The *genus* of a graph is the smallest integer g such that G can be embedded on an orientable surface with g handles without edge crossings. Determining the genus of a general graph is NP-complete [21], but it takes linear time to determine whether a graph is of genus g [13] for any $g = O(1)$.

Our compression algorithm is a subroutine call $\text{encode}_\pi(G_0)$, where G_0 is an input n_0-node π-graph. Let λ be a function with $\lambda(n_0) = \omega(1)$ such that the indexing scheme of all $\lambda(n_0)$-node π-graphs can be obtained in $O(n_0)$ time. (Statement F4 implies the existence of such a function λ.) The indexing scheme of $\lambda(n_0)$-node π-graphs can be implemented by precomputing a table M of $N_\pi(\lambda(n_0))$ entries, where the

*Institute of Information Science, Academia Sinica, Taipei 115, Taiwan. http://www.iis.sinica.edu.tw/~hil.

[1] A function $\beta(n)$ is *super-additive* if $\beta(n_1)+\beta(n_2) \leq \beta(n_1+n_2)$. A function $\beta(n)$ is *continuous* if $\beta(n+o(n)) = \beta(n)+o(\beta(n))$. For instance, $\beta(n) = c_0 n^{c_1} \log^{c_2} n$ is continuous and super-additive, for any real numbers c_0, c_1, c_2 with $c_0, c_2 \geq 0$ and $c_1 \geq 1$. The continuity and super-additivity are closed under additions.

[2] The bottleneck of 3D model compression is the so-called "connectivity encoding" [20], which encodes a graph embedded on the surface of a 3D object. To achieve good resolution for the 3D representation of an object, it is reasonable to assume the number of "holes" in the 3D object to be sublinear in the number of nodes of the graph.

[3] Note that this condition holds even for trees: $\log_2 N_\pi(n) = 2n - o(n)$ for π being a rooted tree.

[4] By $\log^{(k)} n$ we mean $\overbrace{\log\log\log\cdots\log}^{k} n$.

[5] Observe that determining whether an input graph satisfies such a π is NP-complete.

i-th entry contains an adjacency matrix representing the i-th $\lambda(n_0)$-node π-graph. The subroutine encode$_\pi(G)$ is described as follows, where $|G|$ denotes the number of nodes in G:

Step 1. If $|G| > \lambda(n_0)$, then go to Step 2. Otherwise, let code$_\pi(G)$ be the index of G in M, update the node labels of G according to the adjacency matrix of G stored in M, and finally return code$_\pi(G)$.
Step 2. Let $n = |G|$. Decompose G in $O(n)$ time into edge-disjoint π-subgraphs $G_0, G_1, \ldots, G_\ell$ such that $\ell = o(n)$, $\sum_{i=0}^{\ell} |G_i| = n + o(\frac{n}{\log n})$, G_0 has $o(\frac{n}{\log n})$ edges, and $|G_i| = O(\log^{O(1)} n)$ holds for each $1 \le i \le \ell$.
Step 3. Recursively call encode$_\pi(G_i)$ for each $1 \le i \le \ell$. Then, update the label of each node of G according to its labels in its residing subgraphs G_i determined by encode$_\pi(G_i)$. Based upon the new labels, create an $o(n)$-bit string S sufficient for recovering G from $G_1, G_2, \ldots, G_\ell$. Let code$_\pi(G)$ be the concatenation of S and code$_\pi(G_i)$ with $1 \le i \le \ell$, appended by an $o(n)$-bit string [16] with which all these $o(n)$ concatenated strings can be recovered from code$_\pi(G)$. Return code$_\pi(G)$.

We can prove that the output bit count of our compression algorithm is optimal. Moreover, our compression algorithm and the corresponding decoding algorithm take linear time. Our framework is equipped with the mechanism of node relabeling, and thus is more powerful than that of He et al. [11].

It remains to show how to implement Step 2 for any π-graph satisfying Statements F1–F4. For decomposition, we resort to Djidjev and Venkatesan's $O(n+g)$-time planarization algorithm for n-node genus-g graphs [5] and Goodrich's $O(n)$-time algorithm for separator decomposition trees of n-node planar graphs [9]. From these two algorithm, one can obtain in $O(n)$-time a set S_0 of $o(\frac{n}{\log n})$ nodes whose removal decomposes G into $o(\frac{n}{\log n})$ disjoint $O(\log^{O(1)} n)$-node subgraphs. Let G_0 be the subgraph of G induced by S_0. We then partition the nodes of $G - S_0$ into $S_1, S_2, \ldots, S_\ell$ according to the embedding of G such that each set is either of size $w(\log n)$ or adjacent to at most one node in S_0. For each $i \ge 1$, let G_i be the subgraph of G induced by S_i and the neighbors of S_i in G. We can prove that these $G_0, G_1, \ldots, G_\ell$ indeed satisfy the required conditions.

References

[1] S. R. Arikati, A. Maheshwari, and C. D. Zaroliagis. Efficient computation of implicit representations of sparse graphs. *Discrete Applied Mathematics*, 78:1–16, 1997.

[2] Y.-T. Chiang, C.-C. Lin, and H.-I. Lu. Orderly spanning trees with applications to graph drawing and graph encoding. In *Proceedings of the 12th SODA*, pages 506–515, 2001.

[3] R. C.-N. Chuang, A. Garg, X. He, M.-Y. Kao, and H.-I. Lu. Compact encodings of planar graphs via canonical ordering and multiple parentheses. In *Proceedings of the 25th ICALP*, pages 118–129, 1998.

[4] N. Deo and B. Litow. A structural approach to graph compression. In *Proceedings of MFCS'98 Satellite Workshop on Communications*, pages 91–101, 1998.

[5] H. N. Djidjev and S. M. Venkatesan. Planarization of graphs embedded on surfaces. In *Proceedings of the 21st Workshop on Graph-Theoretic Concepts in Computer Science*, LNCS 1017, pages 62–72. Springer, 1995.

[6] T. Feder and R. Motwani. Clique partitions, graph compression and speeding-up algorithms. *Journal of Computer and System Sciences*, 51(2):261–272, 1995.

[7] C. Gavoille and N. Hanusse. Compact routing tables for graphs of bounded genus. In *Proceedings of the 26th ICALP*, pages 351–360, 1999.

[8] C. Gavoille and N. Hanusse. On compact encoding of pagenumber k graphs. Submitted for publication, 2001.

[9] M. T. Goodrich. Planar separators and parallel polygon triangulation. *Jouranl of Computer and System Sciences*, 51(3):374–389, 1995.

[10] X. He, M.-Y. Kao, and H.-I. Lu. Linear-time succinct encodings of planar graphs via canonical orderings. *SIAM J. Discrete Mathematics*, 12(3):317–325, 1999.

[11] X. He, M.-Y. Kao, and H.-I. Lu. A fast general methodology for information-theoretically optimal encodings of graphs. *SIAM J. Computing*, 30(3):838–846, 2000.

[12] D. King. `http://www.3dcompression.com`.

[13] B. Mohar. A linear time algorithm for embedding graphs in an arbitrary surface. *SIAM Journal on Discrete Mathematics*, 12(1):6–26, 1999.

[14] J. I. Munro and V. Raman. Succinct representation of balanced parentheses, static trees and planar graphs. In *Proceedings of the 38th FOCS*, pages 118–126, 1997.

[15] J. I. Munro, V. Raman, and A. Storm. Representing dynamic binary trees succinctly. In *Proceedings of the 12th SODA*, pages 529–536, 2001.

[16] R. Pagh. Low redundancy in static dictionaries with constant query time. *SIAM Journal on Computing*, 31(2):353–363, 2001.

[17] J. Rossignac. Edgebreaker: Connectivity compression for triangle meshes. *IEEE Transactions on Visualization and Computer Graphics*, 5(1):47–61, 1999.

[18] J. Rossignac, A. Safonova, and A. Szymczak. 3D compression made simple: Edgebreaker on a corner-table. In *Proceedings of the Shape Modeling International Conference*, 2001.

[19] J. Rossignac and A. Szymczak. Wrap&Zip decompression of the connectivity of triangle meshes compressed with Edgebreaker. *Computational Geometry: Theory and Applications*, 14(1-3):119–135, 1999.

[20] G. Taubin and J. Rossignac. Geometric compression through topological surgery. *ACM Transactions on Graphics*, 17(2):84–115, 1998.

[21] C. Thomassen. The graph genus problem is NP-complete. *Journal of Algorithms*, 10(4):568–576, 1989.