GRAPH EMBEDDINGS 1988:
Recent Breakthroughs, New Directions

Arnold L. Rosenberg

Computer and Information Science Department
University of Massachusetts

COINS Technical Report 88-28

GRAPH EMBEDDINGS 1988: Recent Breakthroughs, New Directions

Arnold L. Rosenberg
Department of Computer and Information Science
University of Massachusetts
Amherst, MA 01003

March 31, 1988

Abstract

The past few years have seen a number of results on graph embeddings that deserve to be called *breakthroughs* because of their settling hard open problems and/or their raising important issues that promise to alter the direction of subsequent research on graph embeddings. We describe and discuss several such results.

1. THE FORMAL SETTING

1.1. Background

Except where otherwise noted, we operate with the following basic notion of graph embedding. Let G and H be simple undirected graphs. An *embedding* of G in H is a one-to-one association of the vertices of G with the vertices of H, together with a specification of paths in H connecting the images of the endpoints of each edge of G. The *dilation* of the embedding is the maximum length of any of these G-edge-routing paths. The *expansion* of the embedding is the ratio $|H|/|G|$ of the number of vertices in H to the number of vertices in G. (We need to consider both dilation and expansion since, somewhat counterintuitively, one can sometimes decrease dilation by embedding G in a larger relative of H [HMR].) The *congestion*

of the embedding is the maximum number of edges of G that are routed through a single edge (or vertex) of H.

Sections 2-4 of our tour of the frontiers of graph-embedding theory deal with graph embeddings used to model *logical* or *inherent* phenomena: issues like incompatibilities in structure or intersimulatability of graph families. Sections 5 and 6 deal with *physical* phenomena: tolerance to faults and the physical mapping problem.

1.2. Graphs of Interest

The *d-dimensional Hypercube* $Q(d)$ is the graph whose vertex-set is the set $\{0,1\}^m$ of length-m binary strings and whose edges connect just those pairs of vertices/strings that differ in precisely one bit-position.

The *m-level Butterfly graph* $B(m)$ has vertex-set

$$V_m = \{0, 1, \cdots, m-1\} \times \{0,1\}^m.$$

The edges of $B(m)$ form *butterflies* between consecutive levels of vertices, with wraparound in the sense that level 0 is identified with level m. Each butterfly connects vertices

$$\langle \ell,\ \beta_0\beta_1\cdots\beta_{\ell-1}0\beta_{\ell+1}\cdots\beta_{m-1}\rangle \text{ and } \langle \ell,\ \beta_0\beta_1\cdots\beta_{\ell-1}1\beta_{\ell+1}\cdots\beta_{m-1}\rangle$$

($0 \le \ell < m$; each $\beta_i \in \{0,1\}$) with vertices

$$\langle \ell+1 (\text{mod } m),\ \beta_0\beta_1\cdots\beta_{\ell-1}0\beta_{\ell+1}\cdots\beta_{m-1}\rangle$$

and

$$\langle \ell+1 (\text{mod } m),\ \beta_0\beta_1\cdots\beta_{\ell-1}1\beta_{\ell+1}\cdots\beta_{m-1}\rangle$$

The *height-h complete binary tree* $T(h)$ is the graph whose vertices comprise all binary strings of length at most h and whose edges connect each vertex x of length less than h with vertices $x0$ and $x1$. The ℓ^{th} *level* of $T(h)$ $(0 \le \ell \le h)$ consists of all vertices/strings of length ℓ.

The *height-h X-tree* $X(h)$ is obtained from the complete binary tree $T(h)$ by adding edges that connect the vertices at each level of $T(h)$ in a path (with the vertices in lexicographic order).

The $s \times s$ *mesh* $M(s)$ is the graph whose vertex-set is

$$\{1, 2, \cdots, s\} \times \{1, 2, \cdots, s\}$$

and whose edges connect vertices $\langle a, b\rangle$ and $\langle c, d\rangle$ just when $|a-c| + |b-d| = 1$.

2. EMBEDDING GRAPHS IN BUTTERFLIES

2.1. Background

It has been known for many years (cf. [PV]) that a large class of algorithms run as fast on the 4-valent Butterfly network $B(m)$ as on the m-valent Hypercube $Q(m)$. Indeed, this observation motivates much of the interest in butterfly-like parallel architectures. Although it was widely believed that the Hypercube was strictly more powerful than the Butterfly – because of its interconnection structure rather than just its larger valence – no one had been able to prove that this was the case. Sandeep Bhatt, Fan Chung, Jia-Wei Hong, Tom Leighton, and I [BCHLR] have recently found such a proof. The following parameters of a graph are central to our result.

- A *1/3-2/3 (vertex-)separator* of G is a set of vertices whose removal partitions G into subgraphs, each having $\geq |G|/3$ vertices; we denote by $\Sigma(G)$ the size of the smallest 1/3-2/3 vertex-separator of G.
- When G is planar and we are given a witnessing planar embedding ϵ, we denote by $\Phi_\epsilon(G)$ the number of vertices in G's largest interior face in the embedding. When ϵ is clear from context, we omit the subscript.

2.2. A Nontrivial Lower Bound

Theorem 1 [BCHLR] *Any embedding of a nontree planar graph G in a Butterfly graph has dilation $\Omega\left(\frac{\log \Sigma(G)}{\Phi(G)}\right)$. This bound cannot be improved in general.*

Theorem 1 holds, in fact, for a wide variety of butterfly-like graphs. Two instantiations of this result are particularly interesting.

Corollary 1 *Any embedding of the height-h X-tree $X(h)$ in a Butterfly graph must have dilation $\Omega(\log h) = \Omega(\log \log |X(h)|)$.*

Corollary 2 *Any embedding of the $s \times s$ mesh $M(s)$ in a Butterfly graph must have dilation $\Omega(\log s) = \Omega(\log |M(s)|)$.*

Techniques in [BCLR] and [Gr] show that both of these graphs can be embedded very efficiently - simultaneous dilation $O(1)$ and expansion $O(1)$ - in the Hypercube. Thus, these graphs yield the desired examples.

2.3. Remaining Challenges

The bounds in Corollaries 1 and 2 are tight, to within constant factors [BCHLR], but the method of demonstration raises an important issue. The embeddings that witness the upper bounds embed the subject guest graphs in a complete binary tree, with the indicated dilations, and then embed the tree in the Butterfly, with simultaneous dilation $O(1)$ and expansion $O(1)$. The embeddings thus do not exploit the structure of the Butterfly and, as a consequence, have horrendous congestion. The moral of this story is:

> *Graph-embedding theory is going to have to start paying more attention to the issue of congestion.*

The second question raised Theorem 1 concerns extending the result to other networks. In just the same way that butterfly-like graphs can often simulate Hypercubes with no time loss, *coset graphs* of butterfly-like graphs – the Shuffle-Exchange graph, for instance – can often simulate butterfly-like graphs with little or no time loss [ABR]. Almost certainly, butterfly-like graphs are more powerful than such coset graphs in general; but, no proof to this effect has yet appeared. In its strongest form the problem is:

Problem 1 *Do there exist n-vertex graphs that are embeddable in Butterflies with dilation $O(1)$ but that require dilation $\Omega(\log n)$ when embedded in the Shuffle-Exchange graph?*

3. DYNAMIC GRAPH EMBEDDINGS

3.1. Background

In 1986, Sandeep Bhatt, Fan Chung, Tom Leighton, and I [BCLR] proved, via a very complicated embedding strategy, that every binary tree could be embedded efficiently – with simultaneous dilation $O(1)$ and expansion $O(1)$ – in the Hypercube. One major motivation for seeking this result was to be able to argue that Hypercube networks could efficiently execute any divide-and-conquer algorithm. The complication of our embedding was due to three stringent demands, that are inherent in our notion of graph embedding, but that one might argue are "overkill," given our motivation. We insisted

1. that we assign tasks to processors using *global* knowledge of how each execution of the algorithm unfolds

2. that the execution time of the algorithm, as measured by dilation, be $O(1)$

3. that the entire algorithm/tree reside in the network throughout execution, with each task occupying its own private processor.

3.2. A Dynamic Embedding

Sandeep Bhatt and Jin-Yi Cai [BC] have abandoned these demands and have discovered a very simple algorithm that dynamically maps a growing (and shrinking) binary tree onto the Hypercube, with small dilation; however, their mapping need not be an embedding: it may map several tree vertices onto the same Hypercube vertex, but only boundedly many.

Theorem 2 [BC] *Any n-vertex binary tree can be dynamically mapped into the Hypercube $Q(\lceil \log n \rceil)$, with dilation $O(\log \log n)$; there is a constant c such that, with probability $1 - n^{-c}$, only $O(1)$ tree vertices are mapped to any single Hypercube vertex. No randomized algorithm produces a mapping with smaller dilation.*

3.3. Remaining Challenges

Most obviously, one would like to generalize Theorem 2 to guest graphs other than trees and host graphs other than Hypercubes. It is not clear that the Bhatt-Cai strategy of mapping via random walks will admit either generalization.

I have recently begun to look at a genre of "embedding" that abandons only the third of the above demands. A similar but technically distinct relaxation appears in [Fe] and, with different motivation, in [Be, GH]. None of these approaches has yet produced any truly sophisticated embeddings; but the motivation is great, and I remain optimistic.

4. NEW RESULTS ON BOOK-EMBEDDINGS

4.1. Background

A *book* is a finite set of half-planes (the *pages*) that share a common boundary (the *spine*). One *embeds a graph in a book* by ordering the vertices of the graph along the spine and assigning the edges of the graph to pages in such a way that each edge lies on just one page, and edges that lie on the same page do not cross. One's goal is to embed one's graph in a book having as few pages as possible, and as small a cutwidth as possible on each page. Thus, we refer to the *pagenumber* of a graph G, i.e., the number of pages in the smallest book in which G can be embedded, and the *pagewidth* of G, i.e., the smallest possible maximum cutwidth on any page in a book in which G can be embedded.

4.2. Pagenumber and Graph Size

In an attempt to find a nontrivial predictor of the pagenumber of a graph, Fan Chung, Tom Leighton, and I [CLR] proved the following.

(a) *Every n-vertex d-valent graph has pagenumber $O(d\sqrt{n})$.*
(b) *For every $d \geq 2$ and all large n, there are n-vertex d-valent graphs whose pagenumber is at least*

$$\Omega\left(\frac{\sqrt{n}}{n^{1/d}\log^2 n}\right).$$

Except when d is in the neighborhood of $\log n$, these upper and lower bounds are quite far apart.

Seth Malitz [Ma] has narrowed the gap considerably; indeed, when the valence $d \geq \log n$, Malitz's bounds are tight to within constant factors.

Theorem 3 [Ma] (a) *Every e-edge graph has pagenumber $O(\sqrt{e})$.*
(b) *For every $d \geq 2$ and all large n, there are n-vertex d-regular graphs whose pagenumber is at least*

$$\Omega\left(\frac{\sqrt{dn}}{n^{1/d}}\right).$$

4.3. Pagenumber and Graph Structure

Also seeking a nontrivial predictor of the pagenumber of a graph, Mihalis Yannakakis [Ya] and Lenny Heath and Sorin Istrail [HI] have made a giant stride toward correlating the pagenumber of a graph with its genus:

> [Ya] *Every planar graph has pagenumber at most* 4. *There exist planar graphs whose pagenumber is* 4.
>
> [HI] *Every graph of genus* $g \geq 1$ *has pagenumber* $O(g)$. *For each such* g, *there exist graphs of genus* g *whose pagenumber is* $\Omega(\sqrt{g})$.

Seth Malitz [Ma] has built a sophisticated superstructure on top of the framework of [HI] to lower the upper bound in the latter result to within a constant factor of the lower bound.

Theorem 4 [Ma] *Every graph of genus* $g \geq 1$ *has pagenumber* $O(\sqrt{g})$.

4.4. Pagenumber vs. Pagewidth

From my vantage point, the greatest remaining challenge in the area of book-embeddings resides in the following pairs of facts, which suggest that it is generally impossible to minimize pagenumber and pagewidth simultaneously.

1. *Every* d*-valent* n*-vertex graph with pagenumber* 1 *is outerplanar* [BK], *hence has cutwidth* $O(d \log n)$.
2. *There exist trivalent* n*-vertex outerplanar graphs every* 1*-page book-embedding of which has pagewidth* $\Omega(n)$ [CLR].
3. *Every* d*-valent* n*-vertex graph with pagenumber* 2 *is planar* [BK], *hence has cutwidth* $O(d\sqrt{n})$.
4. *There exist* 4*-valent* n*-vertex planar graphs every* 2*-page book-embedding of which has pagewidth* $\Omega(n)$ [CLR].

Lenny Heath [He] has taken the sting out of the first pair of assertions, by showing that increasing pagenumber by 1 allows one to approach minimum cutwidth.

Theorem 5 |He| *Every d-valent n-vertex outerplanar graph admits a 2-page book-embedding with pagewidth* $O(d \log n)$.

The challenge alluded to above is to find an analog of Heath's result that takes the sting out of the second pair of facts also.

Problem 2 *Does there exist a constant c such that every d-valent n-vertex 2-page-embeddable graph admits a c-page book-embedding with pagewidth* $O(d\sqrt{n})$*?*

Indeed it is not even known that every 2-page-embeddable d-valent n-vertex graph admits a $(\log n)$-page book-embedding with pagewidth $O(d\sqrt{n})$.

If the open problem has an affirmative solution, then the next challenge will be to show that one can *always* find a close-to-minimum-cutwidth layout of a graph G via a book-embedding using only modestly more than pagenumber(G) pages.

5. SALVAGING HYPERCUBES

5.1. Background

The many techniques that have been proposed for rendering a parallel architecture tolerant to faults suffer from one of two deficiencies that render them inapplicable to "dense" interconnection networks, such as Butterflies and Hypercubes.

1. Some strategies merely maintain a degree of connectivity in the network. This is far too modest a goal, since parallel algorithms typically exploit the structure of the network, not just its degree of connectivity.

2. Strategies that preserve network structure typically have such high overhead that they are useful only for sparse networks, like meshes and trees.

To the extent that this assessment is valid, the only viable approach to fault tolerance for dense networks is to try to *salvage* the network, in the sense that |GE, LL| salvage meshes, i.e., to embed a copy of a smaller version of the network in the surviving portion of the current version.

5.2. Salvaging a Faulty Hypercube

Johan Hastad, Tom Leighton, and Mark Newman [HLN] have recently discovered a disarmingly simple technique for salvaging the Hypercube efficiently. We paraphrase only the simplest of their results, which differ in the assumed degree of debilitation of the Hypercube due to the faults.

Theorem 6 [HLN] *Let the vertices of $Q(d)$ be colored* red *and* green *at random, independently, with $p = Pr(\text{red}) < 1/2$. There is a deterministic algorithm that, with probability $1 - 2^{-c_p d}$, embeds $Q(d-1)$ in the* green *vertices of $Q(d)$ with dilation 3 and congestion $O(\log n)$, where c_p is a constant depending only on p.*

5.3. Remaining Challenges

Given the structural kinship of the Butterfly to the Hypercube, and of the Shuffle-Exchange graph to the Butterfly, one would hope that a strategy similar to that in [HLN] would yield the following.

Problem 3 *Can one find an analog of Theorem 6 that embeds $B(m-1)$ in $B(m)$ (or, the order-$(n-1)$ Shuffle-Exchange graph in the order-n Shuffle-Exchange graph) with dilation $O(\log \log n)$?*

The conjectured dilation in this problem is due to the 4-valence of the sparse graphs, in contrast to the $(\log n)$-valence of the Hypercube.

6. THE PHYSICAL MAPPING PROBLEM

6.1. Background

Even when techniques for tolerating faults in networks do not incur excessive overhead, they create a problem that has received little attention in the literature. The problem is exemplified by the following scenario. Say that we wish to realize a complete binary tree on our idealized architecture and that after fabrication and testing, we find that 510 of the processors of our physical architecture are free of faults. Since the largest complete binary tree we can realize on 510 PEs has only 255 nodes, we must decide which 255 fault-free processors to use. It is quite likely

that our choice of processors will have an effect on the run-time efficiency of the tree; hence, we do not want to make the choice haphazardly. We need to find an efficient, efficiency-enhancing way to map the processors of our logical architecture on the surviving processors of our physical architecture.

6.2. An Instance of Physical Mapping

Motivated by the relationship between the book-embedding problem and the DIOGENES methodology for designing fault-tolerant processor arrays [Ro], Lenny Heath, Bruce Smith, and I [HRS] studied the physical mapping problem for graphs that are embedded in books, considering both the average cost of an embedding and the maximum-wire-run cost *MCOST*. For the latter case, we established the following.

Say that we are given a book-embedding Λ of an m-vertex graph G, together with a linear sequence Π of "processors," $n \geq m$ of which are fault-free, hence available to "receive" the vertices of G. We wish to *assign* the vertices of Λ in an order-preserving manner to the fault-free "processors" in Π. The *MCOST* of an assignment is just the maximum distance in the sequence Π between the endpoints of any edge of G.

Theorem 7 [HRS] *Given a book-embedding Λ and a sequence of "processors" Π, there is an algorithm that finds an MCOST-optimal assignment of the vertices of Λ to the "processors" of Π, that operates in time*

$$O(m \cdot (n - m) \cdot \log(m \cdot (n - m)) \cdot \log M),$$

where M is the largest inter-"processor" distance.

When one allows "shortcuts" in the sequence Π, the general problem of finding *MCOST*-optimal assignments becomes *NP*-complete, although there are efficient approximation algorithms; cf. [HRS].

The obvious remaining challenge is to identify other genres of layout strategies where one can attack the physical mapping problems.

ACKNOWLEDGMENT. This work was supported in part by NSF Grant DCI-87-96236. It is a pleasure to acknowledge the contribution – to the field as well as to this paper – of all the authors whose work is cited here.

7. REFERENCES

[**ABR**] F. Annexstein, M. Baumslag, A.L. Rosenberg (1987): Group-action graphs and parallel architectures. Submitted for publication.

[**Be**] F. Berman (1983): Parallel computation with limited resources. *Johns Hopkins Conf. on Information Sciences and Systems.*

[**BK**] F. Bernhart and P.C. Kainen (1979): The book thickness of a graph. *J. Comb. Th. (B) 27*, 320-331.

[**BC**] S.N. Bhatt and J.-Y. Cai (1987): Take a walk, grow a tree. Typescript, Yale Univ.

[**BCLR**] S.N. Bhatt, F.R.K. Chung, F.T. Leighton, A.L. Rosenberg (1986): Optimal simulations of tree machines. *27th IEEE Symp. on Foundations of Computer Science*, 274-282.

[**BCHLR**] S.N. Bhatt, F.R.K. Chung, J.-W. Hong, F.T. Leighton, A.L. Rosenberg (1988): Optimal simulations by Butterfly networks. *20th ACM Symp. on Theory of Computing*, to appear.

[**CLR**] F.R.K. Chung, F.T. Leighton, A.L. Rosenberg (1987): Embedding graphs in books: A layout problem with applications to VLSI design. *SIAM J. Algebr. Discr. Meth. 8*, 33-58.

[**Fe**] M.R. Fellows (1985): *Encoding graphs in graphs.* Ph.D. Dissertation, Univ. California at San Diego.

[**FP**] J. Friedman and N. Pippenger (1986): Expanding graphs contain all small trees. Typescript, IBM Almaden Research Center.

[**Gr**] D.S. Greenberg (1987): Minimum expansion embeddings of meshes in hypercubes. Tech. Rpt. DCS/RR-535, Yale Univ.

[**GE**] J.W. Greene and A. El Gamal (1984): Configuration of VLSI arrays in the presence of defects. *J. ACM 31*, 694-717.

[**GH**] A.K. Gupta and S.E. Hambrusch (1988): Embedding large tree machines into small ones. *MIT Conf. on Advanced Research in VLSI.*

[**HLN**] J. Hastad, F.T. Leighton, M. Newman (1987): Reconfiguring a hypercube in the presence of faults. *19th ACM Symp. on Theory of Computing.*

[**He**] L.S. Heath (1987): Embedding outerplanar graphs in small books. *SIAM J. Algebr. Discr. Meth. 8*, 198-218.

[**HI**] L.S. Heath and S. Istrail (1987): The pagenumber of genus g graphs is $O(g)$. *19th ACM Symp. on Theory of Computing.*

[HRS | L.S. Heath, A.L. Rosenberg, B.T. Smith (1988): The physical mapping problem for parallel architectures. *J. ACM*, to appear.

[HMR | J.-W. Hong, K. Mehlhorn, A.L. Rosenberg (1983): Cost tradeoffs in graph embeddings. *J. ACM 30*, 709-728.

[LL | F.T. Leighton and C.E. Leiserson (1985): Wafer-scale integration of systolic arrays. *IEEE Trans. Comp., C-34*, 448-461.

[Ma | S.M. Malitz (1988): Embedding graphs in small books. Manuscript, MIT.

[PV | F.P. Preparata and J.E. Vuillemin (1981): The cube-connected cycles: a versatile network for parallel computation. *C. ACM 24*, 300-309.

[Ro | A.L. Rosenberg (1983): The Diogenes approach to testable fault-tolerant arrays of processors. *IEEE Trans. Comp., C-32*, 902-910.

[Ya | M. Yannakakis (1986): Four pages are necessary and sufficient for planar graphs. *18th ACM Symp. on Theory of Computing*, 104-108.