A Survey on Graph Layout Problems *

Josep Díaz[†] Jordi Petit[†] María Serna[†]

October 17, 2000

1 Introduction

In this paper we survey results on several graph layout problems from an algorithmic point of view. Graph layout problems are a particular class of combinatorial optimization problems whose goal is to find a linear layout of an input graph in such way that a certain objective cost is optimized. A linear layout is a labeling of the nodes of a graph with distinct integers. A large amount of relevant problems in different areas can be formulated as graph layout problems. These include network optimization, VLSI circuit design, information retrieval, numerical analysis, computational biology, graph theory, scheduling, archaeology, etc. On the other hand, the minimal values of some layout costs are also related to interesting graph theoretic invariants of graphs. Most of the graph layout problems are **NP**-complete, but, in many of their applications, feasible solutions with an almost optimal cost are sufficient and, thus, approximation algorithms or effective heuristics can be used.

There exist different other surveys that deal with several aspects of graph layout problems; see for instance [9, 18, 20, 75]. We have tried to give a complete view of the current state of the art with respect to layouts problems

The paper is organized as follows. In Section 1, we define several layout problems and state some basic properties. In Section 2, we present motivations and applications for this family of problems. Negative and positive complexity results are reviewed in Section 4. Section 5 presents methods to get lower bounds for layout problems. In Sections 6 and 7, we present work done in approximation algorithms and heuristics, respectively. Probabilistic results on different models of random graphs for layout problems are presented in Section 8. We close the paper with a selection of open problems.

2 The problems

A *(linear) layout* φ of an undirected graph $G = (V, E)$ with $n = |V|$ nodes is a bijective function $\varphi : V \to \{1, 2, \ldots, n\}$. We denote by $\Phi(G)$ the set of all layouts of the graph G. A layout is also called a *linear arrangement* or a *labeling* or a *numbering* of the nodes of a graph.

*This research was partially supported by the IST Programme of the EU under contract number IST-1999-14186 (ALCOM-FT).

[†]Departament de Llenguatges i Sistemes Informàtics. Universitat Politècnica de Catalunya. Campus Nord C6. c/ Jordi Girona 1-3. 08034 Barcelona, Catalonia. {diaz,jpetit,mjserna}@lsi.upc.es

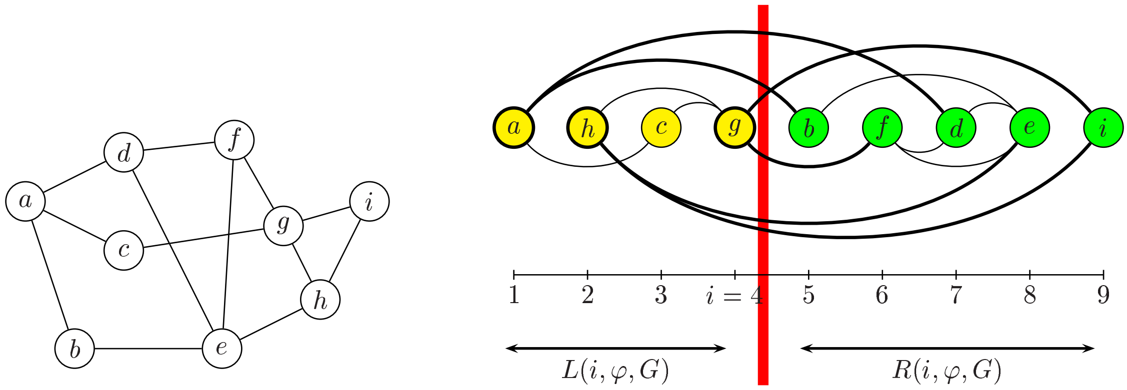

Figure 1: A graph G and a graphical representation of a layout φ of G.

Given a layout φ of a graph $G = (V, E)$ and an integer i, we define the sets

$$L(i, \varphi, G) = \{u \in V \ : \ \varphi(u) \leqslant i\} \quad \text{and} \quad R(i, \varphi, G) = \{u \in V \ : \ \varphi(u) > i\}.$$

The *(edge) cut* at position i of φ is defined as

$$\theta(i, \varphi, G) = |\{uv \in E \ : \ u \in L(i, \varphi, G) \wedge v \in R(i, \varphi, G)\}|$$

and the *modified (edge) cut* at position i of φ as

$$\zeta(i, \varphi, G) = |\{uv \in E \ : \ u \in L(i, \varphi, G) \wedge v \in R(i, \varphi, G) \wedge \varphi(u) \neq i\}|.$$

The *vertex cut* or *separation* at position i of φ is defined as

$$\delta(i, \varphi, G) = |\{u \in L(i, \varphi, G) \ : \ \exists v \in R(i, \varphi, G) : uv \in E\}|.$$

Given a layout φ of G and an edge $uv \in E$, the *length* of uv on φ is

$$\lambda(uv, \varphi, G) = |\varphi(u) - \varphi(v)|.$$

A natural way to represent a layout φ of a graph G is to align its nodes on a line, mapping node u to position $\varphi(u)$, as shown in Fig. 1.

A *layout cost* is a function F that associates to each layout φ of a graph G an integer $F(\varphi, G)$. Let F be a layout cost; the (optimization) layout problem associated with F consists in determining some layout $\varphi^* \in \Phi(G)$ of an input graph G such that

$$F(\varphi^*, G) = \min_{\varphi \in \Phi(G)} F(\varphi, G) = F(G).$$

In Table 1 we show, the particular layout costs and the corresponding problems.

At this point it is relevant to point out some basic but relevant observations:

Observation 1. The graph layout problems we have formulated explicitly ask for an optimal layout rather than the cost of an optimal layout: *"Given a graph G, find a layout $\varphi^* \in \Phi(G)$ such that $F(\varphi^*, G) = F(G)$."* All the problems can however be restated as decisional problems, where the task is to decide whether or not a graph admits a layout with cost not greater than a certain input integer: *"Given a graph G and an integer K, is there some layout $\varphi \in \Phi(G)$ such that $F(\varphi, G) \leqslant K$?"* In the following, we will not insist too much on which version of the problem we are considering (optimization or decisional), because they will be differentiated by the context.

Problem	Name	Cost
Bandwidth	Bandwidth	$\textsc{bw}(\varphi, G) = \max_{uv\in E} \lambda(uv, \varphi, G)$
Min Linear Arrangement	MinLA	$\textsc{la}(\varphi, G) = \sum_{uv\in E} \lambda(uv, \varphi, G) = \sum_{i=1}^{n} \theta(i, \varphi, G)$
Cutwidth	Cutwidth	$\textsc{cw}(\varphi, G) = \max_{i=1}^{n} \theta(i, \varphi, G)$
Modified Cut	ModCut	$\textsc{mc}(\varphi, G) = \sum_{i=1}^{n} \zeta(i, \varphi, G)$
Vertex Separation	VertSep	$\textsc{vs}(\varphi, G) = \max_{i=1}^{n} \delta(i, \varphi, G)$
Sum Cut	SumCut	$\textsc{sc}(\varphi, G) = \sum_{i=1}^{n} \delta(i, \varphi, G) = \textsc{pr}(\varphi^R, G)$
Profile	Profile	$\textsc{pr}(\varphi, G) = \varphi(u) - \min_{v\in\Gamma^*(u)} \varphi(v)$
Edge Bisection	EdgeBis	$\textsc{eb}(\varphi, G) = \theta(\lfloor n/2 \rfloor, \varphi, G)$
Vertex Bisection	VertBis	$\textsc{vb}(\varphi, G) = \delta(\lfloor n/2 \rfloor, \varphi, G)$

Table 1: Layout problems for a graph $G = (V, E)$ with $|V| = n$.

Observation 2. For any graph $G = (V, E)$ and any layout φ of G, the total edge length equals the sum of all edge cuts: $\sum_{uv\in E} \lambda(uv, \varphi, G) = \sum_{i\in[|V|]} \theta(i, \varphi, G)$. This fact was first noticed by Harper [50]. It follows from the observation that any edge $uv \in E$ with $\varphi(u) < \varphi(v)$ contributes $\varphi(v) - \varphi(u)$ to the left hand side and 1 to each one of the terms $\theta(\varphi(u), \varphi, G), \theta(\varphi(u) + 1, \varphi, G), \ldots, \theta(\varphi(v), \varphi, G)$ in the right hand side.

Observation 3. For any graph $G = (V, E)$ and any layout φ of G, it is the case that $\textsc{pr}(\varphi, G) = \textsc{sc}(\varphi^R, G)$ where φ^R denotes the layout φ reversed (i.e., $\varphi^R(u) = |V| - \varphi(u) + 1$ for all $u \in V$). The reason is that each node $u \in V$ contributes $\varphi(u) - \min_{v\in\Gamma^*(u)} \varphi(v)$ times one unit to the sum cut in the reversed layout. As a consequence, Profile and SumCut are equivalent problems!

Observation 4. Let G be any graph with n nodes and m edges. The following results are a basic consequence of the previous definitions.

$$\textsc{la}(G) \leqslant n \cdot \textsc{cw}(G), \qquad \textsc{la}(G) \leqslant m \cdot \textsc{bw}(G), \qquad \textsc{sc}(G) \leqslant n \cdot \textsc{vs}(G),$$
$$\textsc{mc}(G) \leqslant \textsc{la}(G), \qquad \textsc{vb}(G) \leqslant \textsc{vs}(G) \leqslant \textsc{bw}(G), \qquad \textsc{eb}(G) \leqslant \textsc{cw}(G).$$

Observation 5. For any layout cost F in the set $\{\textsc{bw}, \textsc{cw}, \textsc{vs}, \textsc{la}, \textsc{mc}, \textsc{sc}\}$, it is possible to obtain an optimal layout for F of a graph just by computing optimal layouts of its connected components. Observe, however, that the bisection costs (EB and VB) do not share this property.

3 Motivations and applications

Historically, the first layout problem was Bandwidth, as a means to speed up several computations on sparse matrices during the fifties. The Bandwidth problem for graphs was first posed as an open problem during a graph theoretical meeting in 1967 by Harary [48].

The Minimum Linear Arrangement problem was first stated in 1964 by Harper in the same way that the Bandwidth problem but replacing “maximum” by “sum” [49]. Harper's aim was to design error-correcting codes with minimal average absolute errors on certain classes of graphs [49, 50]. More recently, this problem was considered by Mitchison and Durbin as an over-simplified model of some nervous activity in the cortex [74]. MinLA also has applications in single machine job scheduling [2, 91]. A generalization of the MinLA is the storage-time product minimization problem [91]. MinLA has also received some alternative names, as *Optimal Linear Ordering*, *Edge Sum problem* or *Minimum*-1-*sum*.

The Cutwidth problem was first used as a theoretical model for the number of channels in an optimal layout of a circuit in the seventies [3, 72]. In general, the cutwidth of a graph times the order of the graph gives a measure of the area needed to represent the graph in a VLSI layout when nodes are laid out in a row [68]. More recent applications of this problem include network reliability [59], automatic graph drawing [79], information retrieval [17] and as a subroutine for the cutting plane algorithm to solve the TSP [57].

The Vertex Separation problem was originally motivated by the general problem of finding good separators for graphs [69] and it is strongly connected with several other important **NP**-complete problems: Gate Matrix Layout, Pathwidth and Node Search Number. The Gate Matrix Layout is a well studied problem with application in CMOS circuit design [22], the Pathwidth problem has received an enormous interest in recent years because of its relation with the Robertson–Seymour theory [92], and the Node Search Number problem is related to strategy and search in graphs [67].

The Sum Cut and Profile problems were independently defined in [24] and [64], and both problems turn to be equivalent to the Interval Graph Completion problem [91] and have well-known applications in archaeology [62] and clone fingerprinting [60].

After this brief historical introduction, let us mention some of the particular applications of layout problems.

Layout problems in numerical analysis. In the area of numerical analysis, it is desirable to reorder the rows and columns of very large sparse symmetric matrices in such a way that their non-zero entries lie as much close as possible to the diagonal (a matrix is sparse when it has very few non-zero entries). Specifically, the bandwidth of a symmetric matrix M is the largest integer b for which there is a non-zero entry at $M[i, i+b]$ and the profile of M is $\sum_{i=1..n}(i - p_i)$ where p_i is the index of the first non-zero entry of row i. Reducing the bandwidth and/or the profile of a matrix leads to a reduction of the amount of space needed for some storage schemes and to an improvement of the performance of several common operations such as Choleski factorization of non-singular systems of equations. The problem of reducing the bandwidth or the profile of a matrix M consists in finding a permutation matrix P such that the bandwidth or the profile of $M' = P \cdot M \cdot P^T$ is minimal.

Observe that if we identify the non-zero entries of a symmetric matrix with the edges of a graph and the permutations of rows and columns with flips of the node labels, then the bandwidth of the graph equals the bandwidth of the matrix, and the profile of the graph equals the profile of the matrix.

Many commercial packages still offer functions to reduce the bandwidth or the profile of sparse matrices as a preprocessing step. Thus, improvements in these methods can be ported to this software without a complete reorganization of their architecture [36]. Efficient algorithms to perform several operations on matrices with small bandwidth can be found, for instance, in [93]. Information retrieval to browse hypertext is a recent area where bandwidth and profile reduction techniques are used [17].

Layout problems in VLSI. Given a set of modules, the VLSI layout problem consists in placing the modules on a board in a non-overlapping manner and wiring together the terminals on the different modules according to a given wiring specification and in such a way that the wires do not interfere among them. There are two stages in VLSI layout: placement and routing. The *placement problem* consists in placing the modules on a board; the *routing problem* consists in wiring together the terminals on different modules that should be connected. A VLSI circuit can be modeled by the means of a graph, where the edges represent the wires and the nodes represent modules. Of course, this graph is an over-simplified model of the circuit, but understanding and solving problems in this simple model can help to obtain better solutions for the real-world model [10, 94, 96].

One possible approach to solve the placement problem consists in finding recursively minimal cuts with minimal capacity among all cuts that separate the graph in two components of equal size. The EdgeBis we have presented use this approach. A different approach to solve the placement phase is to use the MinLA in order to minimize the total wire length [3, 51].

Layout problems in graph drawing. Perhaps the most important goal in graph drawing is to produce aestethic representations of graphs. Reducing the number of crossing edges is a way to improve their readability and comprehension. A bipartite drawing (or 2-layer drawing) is a representation of a bipartite graph where nodes are placed in two parallel lines and the edges are drawn with straight lines. The bipartite crossing number of a bipartite graph is the minimal number of edge crossings over all bipartite drawings. Shahrokhi *et al.* [82] prove that for a large class of bipartite graphs, reducing the bipartite crossing number is equivalent to reducing the total edge length, that is, to the MinLA. Moreover, an approximate solution of MinLA can be used to generate an approximate solution to the Bipartite Crossing Number problem.

Layout problems in parallel processing Many parallel computers in use today are made up of a set of processors with their own private memory that exchange messages by the way of a communication network. In order to get good speed-ups when using such a system, it is important to distribute the total amount of work between the processors as evenly as possible to minimize idle times. It is also important to reduce the amount of communication between the processors, because communicating through the network is generally much slower than the speed of the processors. Several mapping and load balancing techniques have been developed to address these situations. For certain cases, these techniques lead to graph partitioning problems (see for example [66]).

The Graph Partitioning problem consists in partitioning the nodes of a given graph in k sets of nearly same size in such a way that the number of cutting edges between the k sets is as minimal as possible. The Edge Bisection problem (EdgeBis) is a particular case of Graph Partitioning where $k = 2$. Recursive bisection is a popular technique to obtain partitions when k is a power of 2 (see [97] for an analysis of recursive bisection).

Edge Bisection can be of use when solving partial differential equations (PDEs) and finite elements methods (FEMs) in parallel systems. Simplifying, in these problems a particular iterative computational task has to be carried out in every node of a grid (or general graph) and its computation involves data from this node and in its neighbors. A way to distribute the total amount of computation between two processors is to assign to each one half of the nodes in the grid. But as border nodes need to communicate in order to get their operands, it is necessary to reduce the cut of the bisection. More information on applications of graph bisection can be found, e.g., in [54, 66].

4 Complexity results

The following theorem indicates the difficulty of the considered layout problems on arbitrary graphs:

Theorem 1. The decisional versions of the following layout problems are **NP**-complete: Bandwidth, MinLA, Cutwidth, ModCut, VertSep, SumCut, Profile, EdgeBis, VertBis.

It is possible to show that many layout problems remain **NP**-complete even for certain restricted classes of graphs. Table 2 summarizes these known negative results and also includes references for the proofs of Theorem 1.

Fixed parameter results. In many applications, practical problems are often characterized by fixed-parameter instances. The parameter may represent, for example, the number of processing elements to be employed, the maximum allowed wire length in a placement problem, or the number of crossings in a planar drawing of a graph. Parameterized complexity has been primarily motivated by concrete computational problems and can be used to analyze layout problems, by considering the decisional versions, and using the corresponding layout measure as the parameter.

A parameterized problem is *fixed-parameter* tractable if there is a constant α and an algorithm that decides whether an instance x with parameter k has a solution in time $f(k)|x|^\alpha$, where f is an arbitrary function. The class of fixed parameter tractable problems is denoted **FPT**. Downey and Fellows [30] introduced a framework from complexity theory and derived a hierarchy of complexity classes $\mathbf{W}[t]$ (the W hierarchy), whose definition is based on logical circuit families and fixed-parameter reducibility. The parameterized analog of the well known **NP** is the class $\mathbf{W}[1]$, and $\mathbf{W}[1]$ hardness is the basic evidence that a a parameterized problem is not likely to be fixed-parameter tractable.

For a few layout problems some parameterized complexity results are known. Fellows and Langston [34] analyzed several layout problems with respect to their parameterized complexity, in particular they obtained: The VertSep is closed under minor containment, and therefore fixed-parameter tractable. The Cutwidth is not closed under minors, but it is closed under the immersion order, therefore by application of results on well-partial-orderings of graphs it is fixed-parameter tractable. In contrast, the ModCut is neither minor closed nor immersion closed, however it can be reduced to a fixed parameter tractable problem, and therefore it is fixed-parameter tractable. In [13] it is shown that the Bandwidth is hard for the class $\mathbf{W}[t]$ for any t. Table 3 summarizes the best complexity of the constructive algorithms. For other problems, their fixed parameter complexity remains open.

Positive results. Recall that **NP**-completeness results do not rule out the existence of efficient algorithms to get optimal layouts on particular classes of graphs. Table 4 summarizes

Problem	NP-complete	Ref.
Bandwidth	in general	[83]
	for trees with maximum degree 3	[37]
	for caterpillars with hairlength $\leqslant 3$	[76]
	for caterpillars with $\leqslant 1$ hair per backbone node	[76]
	for grid graphs and unit disk graphs	[26]
MinLA	in general	[39]
	for bipartite graphs	[38]
Cutwidth	in general	[40]
	for graphs with maximum degree 3	[73]
	for planar graphs with maximum degree 3	[77]
	for grid graphs and unit disk graphs	[26]
ModCut	for planar graphs with maximum degree 3	[77]
VertSep	in general	[67]
	for planar graphs with maximum degree 3	[77]
	for chordal graphs	[45]
	for grid graphs and unit disk graphs	[26]
	for bipartite graphs	[42]
SumCut	in general	[24, 64]
	for cobipartite graphs	[101]
EdgeBis	in general	[39]

Table 2: Review of **NP**-completeness results for graph layout problems.

Problem	Complexity	Ref.
Bandwidth(2)	$O(n)$	[37]
Cutwidth(2)	$O(n)$	[37]
Cutwidth(k)	$O(n^2)$	[35]
ModCut(k)	$O(n^2)$	[35]
VertSep(k)	$O(n)$	[14]

Table 3: Complexity results for layout problems with fixed parameter (n denotes the number of nodes in the graph and k the parameter).

Problem	Class of graph	Complexity	Ref.
BANDWIDTH	Caterpillars with hairlength $\leqslant 2$	$O(n \log n)$	[5]
	Interval graphs	$O(n+m)$	[71]
	Chain graphs	$O(n \log n)$	[63]
	Complete k-level t-ary tree	O(1)	[20]
MINLA	Rooted trees	$O(n \log n)$	[3]
	Complete k-level 3-ary tree	$O(n)$	[19]
	Trees	$O(n^{\log 3/ \log 2})$	[20]
	Rectangular meshes	$O(n)$	[78]
	Square meshes	$O(n)$	[74]
	Hypercubes	$O(n)$	[49]
	de Bruijn graph of order 4	$O(n)$	[51]
	d-dimensional c-ary cliques	$O(n)$	[80]
CUTWIDTH	Trees	$O(n \log n)$	[100]
	Complete k-level t-ary tree	O(1)	[68]
	Hypercubes	$O(n)$	[49]
	d-dimensional c-ary cliques	$O(n)$	[80]
VERTSEP	Trees	$O(n \log n)$	[31]
SUMCUT	Trees	$O(n^{1.722})$	[64]
	Trees	$O(n)$	[24]
EDGEBIS	Hypercubes	$O(n)$	[80]
	d-dimensional c-ary arrays	$O(n)$	[80]
	d-dimensional c-ary cliques	$O(n)$	[80]

Table 4: Review of classes of graphs optimally solvable in polynomial time. (n denotes the number of nodes in the graph, m the number of edges and d its maximal degree)

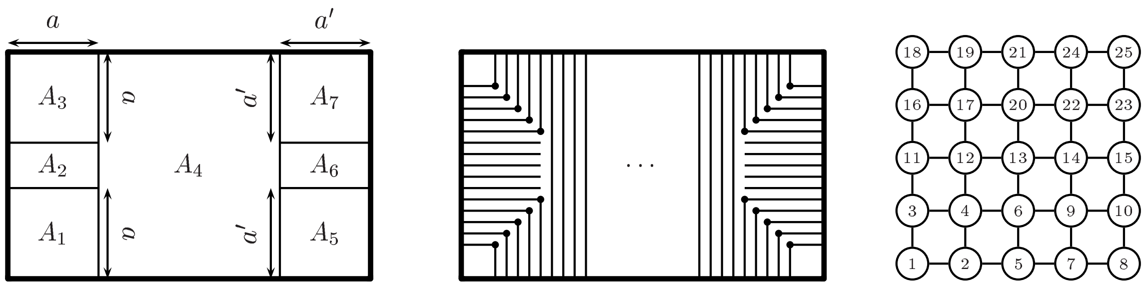

Figure 2: Schematic representation of the optimal layout of a rectangular mesh for MINLA.

which classes of graphs are known to have exact solution in polynomial time. Let us comment some of those results.

In the case of MinLA, Harper computed the optimal value for the de Bruijn graph of order four [51] and for hypercubes [49]. In the case of a d-dimensional hypercube Q_d, $\textsc{la}(Q_d) = 2^{d-1}(2^d - 1)$.

MinLA on square or rectangular grids has a peculiar history: The problem was first solved in a paper written in Russian by Muradyan and Piliposjan for the general case of rectangular grids in 1980 [78]. Latter, Mitchison and Durbin published the solution only for square grids in 1986 [74]. In a paper published in 1993, Nakano [80] references a 1981 paper by Niepel and Tomasta [81] that incorrectly conjectures that the lexicographic layout is optimal for a square grid. Indeed, the optimal layout for a $n \times m$ rectangular mesh $L_{n,m}$ has an interesting solution [9, 74, 78]. The optimal layout is schematically shown in Fig. 2. The numbering starts with the left lower corner of the grid and then fills the areas $A_1, A_2, \ldots, A_7$, where A_1, A_3 are $a \times a$ squares and A_5, A_7 are $a' \times a'$ squares. The values of a and a' must satisfy

$$a, a' \in \left\{ \left\lceil n - \tfrac{1}{2} - \sqrt{\tfrac{1}{2}n^2 - \tfrac{1}{2}n + \tfrac{1}{4}} \right\rceil, \left\lfloor n + \tfrac{1}{2} - \sqrt{\tfrac{1}{2}n^2 - \tfrac{1}{2}n + \tfrac{1}{4}} \right\rfloor \right\}.$$

Each square must be numbered sequentially. A_1 must be filled row after column. A_2 must be filled by consecutive rows, from bottom to top and from left to right. A_3 must be enumerated with the reverse order with respect to A_1. A_4 must be numbered by colums from bottom to top and from left to right. Finally A_5, A_6 and A_7 are filled in the same way. Using this solution, $\textsc{la}(L_{n,m}) = -\frac{2}{3}a^3 + 2na^2 - \left(n^2 + n - \frac{2}{3}\right) a + m(n^2 + n - 1) - n$.

The following theorem gathers the optimal values of layout problems for square meshes:

Theorem 2 ([26, 66, 74, 78]). *Let L_m be an square mesh of side m. Then,*

$$\textsc{vs}(L_m) = m, \qquad \textsc{cw}(L_m) = m + (\text{odd } m), \qquad \textsc{sc}(L_m) = \tfrac{2}{3}m^3 + \tfrac{1}{2}m^2 - \tfrac{7}{6}m,$$
$$\textsc{vb}(L_m) = m, \qquad \textsc{eb}(L_m) = m + (\text{odd } m), \qquad \textsc{la}(L_m) = \tfrac{1}{3}(4 - \sqrt{2})m^3 + O(m^2).$$

With respect to parallel algorithms, Díaz *et al.* proved that MinLA and SumCut can be solved for trees in **NC** [23, 24]. Recall that **NC** is the class of problems that can be solved by a PRAM in polylog number of steps and using a polynomial number of processors. Also, Cutwidth restricted to trees with bounded degree belogs to the **NC** class [23]. An interesting open problem is whether Cutwidth is in **NC** for *unbounded* degree trees.

5 Lower bounds

As all the layout problems presented so far are extremely difficult to solve for general graphs (even if their size is small), it is important to have good lower bounds on the cost of their optimal solutions.

Bounds based on spectral properties. Let $G = (\{1, 2, \ldots, n\}, E)$ be a graph and let L_G be its Laplacian matrix, defined by $L_G[u, v] = -1$ if $uv \in E$, $L_G[u, v] = 0$ if $uv \notin E$ and $L_G[u, v] = \deg(u)$ if $u = v$. The sequence of real eigenvalues $\lambda_1 \leqslant \lambda_2 \leqslant \ldots \leqslant \lambda_n$ is known as the *spectrum* of the graph G. The following theorem states some lower bounds involving spectral properties:

Theorem 3 ([52, 58, 75]). Let G be a connected graph with n nodes and let λ_2 and λ_n denote the second smallest and the largest eigenvalue of the Laplacian of G, respectively. Let α be the largest integer smaller than $n\lambda_2/\lambda_n$. Then,

$$\textsc{la}(G) \geqslant \lambda_2 \frac{n^2-1}{6}, \qquad \textsc{eb}(G) \geqslant \begin{cases} \lambda_2 \cdot n/4 & \text{if } n \text{ is even,} \\ \lambda_2 \cdot n^2 - 1/4n & \text{if } n \text{ is odd,} \end{cases}$$

$$\textsc{cw}(G) \geqslant \lambda_2 \frac{\lfloor n/2 \rfloor \lceil n/2 \rceil}{n}, \qquad \textsc{bw}(G) \geqslant \begin{cases} n-1 & \text{if } \alpha \geqslant n-2, \\ \alpha+1 & \text{if } \alpha \leqslant n-2 \text{ and } n \text{ is even,} \\ \alpha & \text{otherwise.} \end{cases}$$

Bounds based on fundamental cuts. Let $G=(V,E)$ be a graph with n nodes. Gomory and Hu showed that a matrix f storing the maximal flow between two distinct nodes i and j can simply be represented by a weighted spanning tree T of G where each edge represents a *fundamental cut* of G and has weight equal to the corresponding minimal cut [43]. The maximum flow $f[i,j]$ between any pair of nodes i,j can be obtained by finding the path between i and j in T and finding the minimal weight over all its edges. In next theorem, the LA result is due to Adolphson and Hu [3] and the remaining are from the authors of the survey,

Theorem 4. Let $G=(V,E)$ be a graph and $T=(V,E',w)$ its Gomory–Hu tree. Then,

$$\textsc{cw}(G) \geqslant \max_{e\in E'} w(e), \qquad \textsc{la}(G) \geqslant \sum_{e\in E'} w(e), \qquad \textsc{bw}(G) \geqslant \left\lceil \frac{1}{n} \sum_{e\in E'} w(e) \right\rceil.$$

See [70, 88] for an evaluation of these and other lower bounds for the MinLA.

6 Approximation algorithms

One of the classical methods to deal with intractable problems is to dessign an approximation algorithm that in polynomial time will give a solution "close" to the optimal one [6, 38].

Given a minimization problem Π, recall that an $r(n)$-approximation algorithm is an algorithm that for any input x of size n it finds a solution to Π whose cost is at most $r(n)$ times the cost of an optimal solution to the problem's instance. When a problem Π has some $r(n)$-approximation algorithm, it is said to be $r(n)$-*approximable.* When it has an algorithm such that, for all $\varepsilon < 1$, A_ε returns a feasible solution σ such that the ration between the obtained value and the optimal value is less than $+1\varepsilon$ and runs in polynomial time with respect to $|x|$, A_ε is said to be a *polynomial time approximation scheme.* Moreover, when A_ε runs in polynomial time with respect to $|x|$ and $1/\varepsilon$, A_ε is said to be a *fully polynomial time approximation scheme.* A combinatorial optimization problem belongs to the class **APX** if it is ε-approximable for some constant $\varepsilon > 1$, to the class **PTAS** if it admits an approximation scheme and to the class **FPTAS** if it admits a fully approximation scheme. It is known that **FPTAS** $\subseteq$ **PTAS** $\subseteq$ **APX**, where the inclusions are strict if and only if **P**$\neq$**NP**.

It is know that it is **NP**-complete to find a 1.5 approximation for Bandwidth in the case the input are general graphs and trees, therefore Bandwidth doesn't belong to **PTAS** [11]. In fact, it is **NP**-complete to find any k-approximation even for cartepillars, where k is any constant [99]. There are polylogarithmic approximation algorithms, running in polylogarithmic time for general graphs [12, 33]. For some particular kinds of graphs, there are different approximations algorithms. Recall that a graph G with n vertices is called δ-dense iff the minimum degree of G is at least δn. For δ-dense graphs there exists a 3-approximation in polynomial time, moreover it is in **RNC** [61]. For an introduction to parallel approximability, see [29]. There are also polynomial time $O(\log n)$-approximation algorithms for caterpillars [47], and for a more large class of trees, denoted as GHB-trees, which are characterized as trees such that for any node v, the depth difference of any two non-empty subtrees rooted at v is bounded by a constant k [46]. In the case that the input graph is chain-free, the exists a polynomial time $O(\log n)$-approximation algorithm for Bandwidth [63]. For MinLA the obtained polynomial time approximation algorithms are a $O(\log n)$-approximation for general graphs and a $O(\log \log n)$-approximation for planar graphs. Both results are due to Rao and Richa and use the spreading metric technique [90]. Their technique works for the general case of graphs with weighted edges. In the case of dense graphs, MinLA belongs to the class **PTAS** [4]. Very similar results can be obtained for Cutwidth: it belongs to **PTAS** in the case that the input is dense [4] and it is $O(\log n \log \log n)$-approximable for general graphs, using the spreading metrics technique [32].

7 Heuristics

An heuristic is a rule of thumb, simplification or guess that reduces or limits the search for solutions in domains that are difficult and poorly understood. Unlike approximation algorithms, heuristic solutions do not provide a theoretical guarantee but are often used in practice. The assessment of the effecitivity and efficiency of heuristics is therefore inherently empirical.

Due to their importance in engineering applications, many heuristics have been developed to reduce the bandwidth and/or profile of sparse matrices. The most well-known heuristics for bandwidth/profile reduction are King's algorithm, the CutHill–McKee algorithm [21] and the Gibbs–Poole–Stockeyer algorithms [41]. On the other hand, the area of parallel computing has given rise to many heuristics for the Bisection problem. For instance, the Party [89] and Chaco [53] partitioning libraries include a variety of different partitioning methods.

Most heuristics are based on general methods, widely used to solve other combinatorial problems. Adapting these general methods to layout problems is easy in principle, but many decisions that have a great effect on their behavior have to be taken. Under the *Successive Augmentation* approach (also called greedy), a partial layout is extended, node by node, until all nodes have been enumerated, at which point the arrangement is output without any attempt to improve it. At each step, the better possible free label is assigned to the current node. The basic principle of *local search* [1] is to iteratively improve a given solution by performing local changes on its combinatorial structure. Changes that improve the solution are accepted, whereas that changes that worse the solution are rejected or accepted with a probability that decreases as the quality worsens: In Hill Climbing, changes that worse the solution are always rejected; in Metropolis, changes that worse the solution by Δ are accepted with probability $e^{\Delta/t}$ where t is some fixed parameter called temperature; Simulated Annealing consists in a series of runs of Metropolis with a progressively lowering of the temperature. *Spectral Sequencing* [58], first computes the eigenvector $x^{(2)}$ corresponding to the second smallest eigenvalue λ_2 of the Laplacian matrix $L(G)$ of the input graph G and

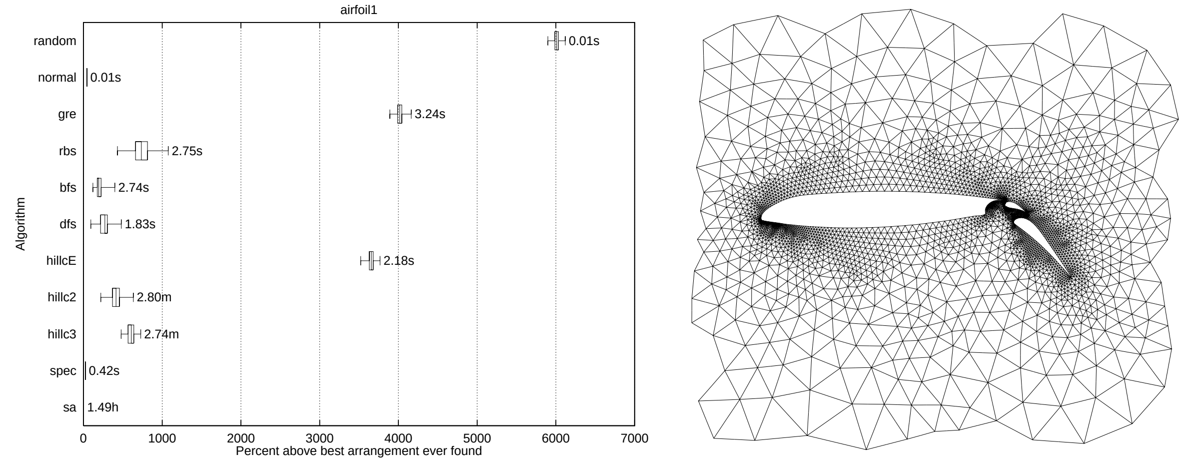

Figure 3: Performance of diverse heuristics (left) on the `airfoil1` graph (right) for the MinLA problem. random: random layout, normal: normal layout, gre, rbs, bfs, dfs: successive augmentation, hillcE, hillc2, hillc3: hill climbing, spec: spectral sequencing, sa: simulated annealing.

then ranks each position of $x^{(2)}$. Thus, the heuristic returns an arrangement φ satisfying $\varphi(u) \leqslant \varphi(v)$ whenever $x^{(2)}_u \leqslant x^{(2)}_v$.

We have undertaken an experimental case of study on heuristics for the MinLA problem [87, 88]. The empirical results are based on two random models (uniform random graphs and random geometric graphs— see Section 8), "real life" graphs (including graphs arising from finite element discretizations, VLSI design and graph-drawing) and graphs with known optimal (trees, hypercubes, meshes). The first goal was to obtain an evaluation on the behavior of different families of heuristics through experimental results. The second goal was to use in a systematic way this experimental knowledge as a fundamental method to guide the design of new heuristics.

In [88] we presented approximation heuristics and methods to find lower bounds for MinLA and we evaluated and compared experimentally their behavior when applied to sparse graphs. The methods belong to the families of Successive Augmentation heuristics, Local Search heuristics (Hill Climbing, Metropolis and Simulated Annealing) and Spectral Sequencing. The conclusion is that the best approximations are obtained using Simulated Annealing, which involves a large amount of computation time. However, solutions found by Spectral Sequencing are also good and can be found in radically less time. Fig. 3 is a sample of the obtained measurements.

Motivated by the long running time of Simulated Annealing compared to Spectral Sequencing, in [87] we presented and analyzed sequential and parallel versions of a new heuristic to approximate the MinLA problem on graphs embedded and clustered in some geometry. The heuristics consist on obtaining a first global solution using Spectral Sequencing and improving it locally through Simulated Annealing. The measurements obtained on an IBM SP2 computer with 8 processors show that the new parallel heuristic maintains solution quality, decreases the running time and offers an excellent speedup when ran in parallel without sacrificing much solution quality.

Our experimental results show, however, a big gap between the lower and the upper bounds.

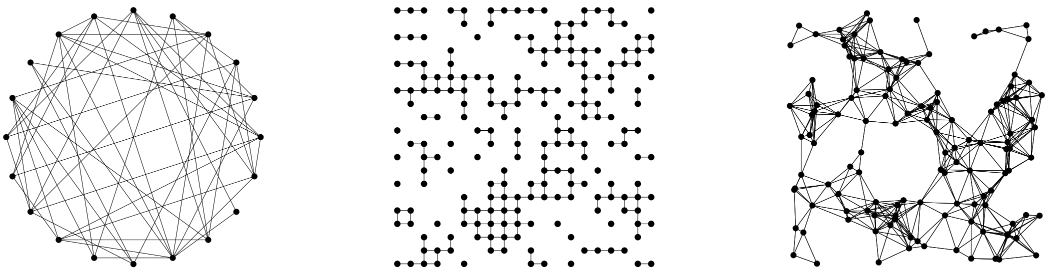

Figure 4: Different models of random graphs: Uniform random graph, random lattice graphs, geometric random graph.

8 Probabilistic analysis

Worst-case analysis does not always catch the real difficulty of a problem. A standard way of evaluating (from a practical point of view) the *real* efficiency of an algorithm or heuristic is to evaluate its performance on random instances. In this section, we survey probabilistic results on graph layout problems which we have obtained together with M.D. Penrose and L. Trevisan.

Recall that a sequence of events $(\mathcal{E}_n)_{n\geqslant 1}$ occurs *with high probability* if $\mathbf{Pr}[\mathcal{E}_n] \to 1$ and *with overwhelming probability* if $\mathbf{Pr}[\mathcal{E}_n] \geqslant 1 - 2^{-\Omega(n)}$ for all n. Also, if $(X_n)_{n\geqslant 1}$ is a sequence of random variables and X is a random variable, it is said that X_n converges *in probability* to X ($X_n \xrightarrow{\Pr} X$) if $\mathbf{Pr}[|X_n - X| > \epsilon] \to 0$, for all $\epsilon > 0$.

Uniform random graphs. Given a positive integer n and a probability p_n, the class of *uniform random graphs*, $\mathcal{G}(n, p_n)$, is a probability space over the set of undirected graphs $G = (V, E)$ on the node set $V = \{1, 2, \ldots, n\}$ $\mathbf{Pr}[uv \in E] = p_n$ with these events mutually independent. This well studied model offers a natural starting point to analyze approximability properties of layout problems in a probabilistic setting.

It is was already known that Bandwidth and EdgeBis can be approximated within a constant on uniform random graphs [15, 16, 98]. Recently, we have extended this kind of result to all layout problems considered in this suvey using a unique framework involving "mixing graphs" (a concept close to expander graphs):

Theorem 5 ([28]). Let $\epsilon \in (0, \frac{1}{6})$, $\gamma \in (0, 1)$ and define $C_{\epsilon,\gamma} = 3(1 + \ln 3)(\epsilon\gamma)^{-2}$. Consider a sequence $(c_n)_{n\geqslant 1}$ such that $C_{\epsilon,\gamma} \leqslant c_n \leqslant n$ for all $n \geqslant n_0$ for some natural n_0. Then, with overwhelming probability, Bandwidth, MinLA, Cutwidth, ModCut, SumCut, VertSep, EdgeBis and VertBis, can be approximated within $O(\epsilon + \gamma)$ on uniform random graphs $\mathcal{G}(n, p_n)$ where $p_n = c_n/n$.

Actually, the proof of the above theorem shows that *any* algorithm computing a feasible layout, no matter how good or bad, will perform rather well on random graphs, pointing out that evaluating heuristics on uniform random graphs may be unworthy for layout problems.

Random lattice graphs. Given an integer m and a probability p, the class of *random lattice graphs*, $\mathcal{L}(m, p)$, is a probability space over the set of lattice graphs $G = (V, E)$ on the node set $V \subseteq [m]^2$ determined by $\mathbf{Pr}[u \in V] = p$ with these events mutually independent and $E = \{uv\ :\ u, v \in V \wedge \|u - v\|_1 = 1\}$.

The study of random lattice graphs is based on *percolation* theory [44]. Consider a *site percolation* process, where nodes from the infinite lattice ($\mathbb{Z}^2$) are selected with some probability p. Let C_0 be the connected component where the origin belongs. A basic question in percolation theory is whether or not C_0 can be infinite. Let $\vartheta(p)$ denote the probability that $|C_0| = \infty$ and set $p_c = \inf\{p : \vartheta(p) > 0\}$, the *critical value* of p. In the *subcritical* limiting regimes $p \in (0, p_c)$, all components are almost surely finite.

Theorem 6 ([26]). Let $p \in (0, p_c)$; then there exist constants $0 < c_1 < c_2$ such that

$$\lim_{m\to\infty} \mathbf{Pr}\left[c_1 \leqslant \frac{\textsc{vs}(\mathcal{L}(m,p))}{\sqrt{\log m}}, \leqslant \frac{\textsc{cw}(\mathcal{L}(m,p))}{\sqrt{\log m}} \leqslant c_2\right] = 1$$

and there exist two constants $\beta_{\textsc{la}}(p) > 0$ and $\beta_{\textsc{sc}}(p) > 0$ such that, as $m \to \infty$,

$$\textsc{la}(\mathcal{L}(m,p))/m^2 \xrightarrow{\Pr} \beta_{\textsc{la}}(p) \qquad \text{and} \qquad \textsc{sc}(\mathcal{L}(m,p))/m^2 \xrightarrow{\Pr} \beta_{\textsc{sc}}(p).$$

Results for the case $p \in (p_c, 1)$ are derived by Penrose in [86].

Random geometric graphs. *Random geometric graphs* $\mathcal{G}(n; r_n)$ are graphs whose n nodes are n points uniformly distributed in the unit square and whose edges between any pair of nodes exist when their distance is smaller than some parameter r_n. Random geometric graphs have been proposed as a possible model to take into account the structural characteristics of instances that appear in many practical applications [8, 55, 65].

Random geometric graphs also exhibit a phase transition [85]: When $nr_n^2 \to \lambda$ there exists a critical parameter λ_c such that if $\lambda < \lambda_c$, graphs $\mathcal{G}(n; r_n)$ are likely to have $O(\log n)$ points in each connected component, while if $\lambda > \lambda_c$, there is likely to be a component with $\Theta(n)$ nodes. The next couple of theorems characterize the behavior of several layout costs in the subcritical and supercrital phases.

Theorem 7 ([26]). Suppose $nr_n^2 \to \lambda \in (0, \lambda_c)$. Then, there exist constants $0 < c_1 < c_2$, $0 < c_3 < c_4$, $\tilde{\beta}_{\textsc{la}}(\lambda) > 0$ and $\tilde{\beta}_{\textsc{sc}}(\lambda) > 0$ such that

$$\textsc{eb}(\mathcal{G}(n; r_n)) \xrightarrow{\Pr} 0, \qquad \textsc{vb}(\mathcal{G}(n; r_n)) \xrightarrow{\Pr} 0,$$

$$\textsc{la}(\mathcal{G}(n; r_n))/n \xrightarrow{\Pr} \tilde{\beta}_{\textsc{la}}(\lambda), \qquad \textsc{sc}(\mathcal{G}(n; r_n))/n \xrightarrow{\Pr} \tilde{\beta}_{\textsc{sc}}(\lambda),$$

$$\mathbf{Pr}\left[c_1 \leqslant \frac{\textsc{vs}(\mathcal{G}(n; r_n))}{\log n/\log\log n} \leqslant c_2\right] \to 1, \qquad \mathbf{Pr}\left[c_3 \leqslant \frac{\textsc{cw}(\mathcal{G}(n; r_n))}{(\log n/\log\log n)^2} \leqslant c_4\right] \to 1.$$

Theorem 8 ([86]). Suppose $nr_n^2 \to \lambda \in (\lambda_c, \infty)$. Then,

$$\textsc{cw}(\mathcal{G}(n; r_n)) = \Theta(n^2 r_n^3), \qquad \textsc{eb}(\mathcal{G}(n; r_n)) = \Theta(n^2 r_n^3), \qquad \textsc{sc}(\mathcal{G}(n; r_n)) = \Theta(n^2 r_n),$$
$$\textsc{la}(\mathcal{G}(n; r_n)) = \Theta(n^3 r_n^3), \qquad \textsc{vs}(\mathcal{G}(n; r_n)) = \Theta(n r_n), \qquad \textsc{bw}(\mathcal{G}(n; r_n)) = \Theta(n r_n).$$

In the case that the radius is slightly greater, it is possible to obtain nice approximations:

Theorem 9 ([27]). Let $(r_i)_{i\geqslant 1}$ be a sequence of positive numbers with $r_n \to 0$ and $nr_n^2/\log n \to \infty$. Then, with high probability, Bandwidth, MinLA, Cutwidth, ModCut, SumCut, VertSep, EdgeBis and VertBis can be approximated within a constant on random geometric graphs $\mathcal{G}(n; r_n)$. Moreover, for Bandwidth and VertSep, the approximation factor can be arbitrarily close to 1.

The proofs of the above results involve the use of isoperimetric inequalities to get lower bounds and the analysis of two simple heuristics using geometrical information to get matching upper bounds. In the case of the Bandwidth and Vertex Separation problems, the solutions returned by either of the heuristics are asymptotically optimal. In these cases, this result is an analog of the seminal BHH theorem [7] on the TSP tour of a random set of points in the unit square.

These results on random geometric graphs can be applied to give empirical evidence of the goodness of several well-known heuristics for layout and partitioning problems [25]. These heuristics include global methods (such as spectral, multilevel and greedy methods) and local methods (such as simulated annealing or Kernighan-Lin).

9 Open problems

In this paper we have presented a current view on the main known results about graph layout problems. The reader has surely observed that plenty of problems remain open: In general, everything that in this paper is not refered as done might turn into an interesting research problem. For instance little is known about the MinLA for sparse graphs, it is not known if it belongs to **NP**-complete or it is in the class **P**.

A task to carry out would be to continue the research of other probabilistic classes of graphs for which good estimations of their optimal values could be known easily (random uniform graphs and random geometric graphs were an example of such a class). These classes could serve as a source of new graph generators to help in the benchmark and analysis of approximation heuristics as well as to the design of networks with some wanted properties. A concrete example of this class of graphs are *random linear graphs*. A random linear graph with n nodes is a graph $G = (V, E)$ where $V = \{1, \dots, n\}$ and where edge $ij \in E$ with probability $1/(j-i)^c$ for all $1 \leqslant i < j \leqslant n$ and some constant $c \geqslant 1$. Our experimental results (and the intuition behind the definition of this class) suggest that the "canonical" arrangements of these graphs (that is, $\varphi(i) = i$, for all i) may be very close to the optimum. Currently, the authors are investigating the behaviour of layout problems with respect to unreliable random geometric graphs, that is, random geometric graphs whose edges or nodes may fail at random independently.

The classification of local search for layout problems with respect to the **PLS** and **P** classes is another interesting line of reserach. The class **PLS** was introduced by Johnson, Papadimitriou and Yannakakis [56, 84] and chapter 2 of citeLSinCO in order to formalize the behavior of Local Search heuristics in the worst case. **PLS** (*polynomial-time local search*) contains the problems for which the local optimality of the feasible solutions can be computed in polynomial time. This class has an associated notion of reduction that gives rise to complete problems. Some famous problems as the TSP, MaxCut and MaxSat are **PLS**-complete [56, 95]. It is therefore reasonable to think that linear arrangement problems also might be complete for **PLS**. In any way, the previous problems are only **PLS**-complete for their weighed versions. Therefore we suppose that it will also be necessary to consider weighted layout problems in order to prove their hipothethical completeness.

When non weighted **PLS**-complete problems are considered (and the objective function is polynomially bounded), a local optimum can always be found in polynomial time. Therefore, it is a natural question to ask whether this computation may be accelerated by the use of massive parallelism. For the previous problems (TSP, MaxCut and MaxSat) it is known that obtaining a local optimum (under usual and natural neighborhood relations) is a **P**-complete problem [95] and therefore without much chances to be parallelized. Thus, it would be interesting to try to investigate whether local search is also **P**-complete for layout problems.

References

[1] E. Aarts and J. K. Lenstra, editors. *Local Search in Combinatorial Optimization.* Wiley, 1997.

[2] D. Adolphson. Single machine job sequencing with precedence constraints. *SIAM Journal on Computing*, 6:40–54, 1977.

[3] D. Adolphson and T. C. Hu. Optimal linear ordering. *SIAM Journal on Applied Mathematics*, 25(3):403–423, Nov. 1973.

[4] S. Arora, A. Frieze, and H. Kaplan. A new rounding procedure for the assignment problem with applications to dense graphs arrangements. In *37th IEEE Symposium on Foundations of Computer Science*, 1996.

[5] S. F. Assman, G. W. Peck, M. M. Syslo, and J. Zak. The bandwidth of caterpillars with hair of lengths 1 and 2. *SIAM Journal on Algebraic and Discrete Methods*, 2:387–393, 1981.

[6] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, and A. M.-S. M. Protasi. *Complexity and Approximation.* Springer-Verlag, 1999.

[7] J. Beardwood, J. Halton, and J. M. Hammersley. The shortest path through many points. *Proceedings of the Cambridge Philos. Society.*, 55:299–327, 1959.

[8] J. W. Berry and M. K. Goldberg. Path Optimization for Graph Partitioning Problems. Technical report TR: 95-34, DIMACS, 1995.

[9] S. L. Bezrukov. Edge isoperimetric problems on graphs. In *Graph theory and combinatorial biology (Balatonlelle, 1996)*, pages 157–197. János Bolyai Math. Soc., Budapest, 1999.

[10] S. N. Bhatt and F. T. Leighton. A framework for solving VLSI graph layout problems. *Journal of Computer and System Sciences*, 28:300–343, 1984.

[11] G. Blache, M. Karpinski, and J. Wirtgen. On approximation intractability of the bandwidth problem. Technical Report TR98-014, Electronic Colloquium on Computational Complexity, 1998.

[12] A. Blum, G.Konjevod, R. Ravi, and S. Vempala. Semi-definite relaxations for minimum-bandwith and and other vertex-ordering problems. In *30th ACM Symposium on Theory of Computing*, pages 284–293, 1998.

[13] H. Bodlaender, M. R. Fellows, and M. T. Hallet. Beyond NP-completeness for problems of bounded width: hardness for the W-hierarchy. In *26th. ACM Symposium on Theory of Computing*, pages 449–458, 1994.

[14] H. L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing*, 25(6):1305–1317, 1996.

[15] B. Bollobas. *Random Graphs.* Academic Press, London, 1985.

[16] R. Boppana. Eigenvalues and graph bisection: An average case analysis. In *28th. IEEE Symposium on Foundations of Computer Science*, pages 280–285, 1987.

[17] R. A. Botafogo. Cluster analysis for hypertext systems. In *16th Annual ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 116–125, 1993.

[18] P. Chin, J. Chvatalova, A. Dewdney, and N. Gibbs. The bandwidth problem for graphs and matrices: A survey. *Journal of Graph Theory*, 6:223–254, 1982.

[19] F. R. K. Chung. Some problems and results in labelings of graphs. In *The theory and applications of graphs*, pages 255–264. Wiley, New York, 1980.

[20] F. R. K. Chung. Labelings of graphs. In *Selected topics in graph theory, 3*, pages 151–168. Academic Press, San Diego, CA, 1988.

[21] E. Cuthill and J. Mckee. Reducing the bandwidth of sparse symmetric matrices. In *24th. ACM National Conference*, pages 157–172, 1969.

[22] N. Deo, M. S. Krishnamoorthy, and M. A. Langston. Exact and approximate solutions for the gate matrix layout problem. *IEEE Transactions on Computer Aided Design*, CAD-6(1):79–84, Jan. 1987.

[23] J. Díaz, A. Gibbons, G. Pantziou, M. Serna, P. Spirakis, and J. Torán. Parallel algorithms for the minimum cut and the minimum length tree layout problems. *Theoretical Computer Science*, (181):267–287, 1997.

[24] J. Díaz, A. M. Gibbons, M. S. Paterson, and J. Torán. The minsumcut problem. In F. Dehen, R. J. Sack, and N. Santoro, editors, *Algorithms and Datastructure*, volume 519 of *Lecture Notes in Computer Science*, pages 65–79. Springer-Verlag, 1991.

[25] J. Díaz, M. D. Penrose, J. Petit, and M. Serna. Approximating layout problems on random geometric graphs. Report de recerca, Departament de Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya, 2000.

[26] J. Díaz, M. D. Penrose, J. Petit, and M. J. Serna. Layout problems on lattice graphs. In T. Asano, H. Imai, D. Lee, S. Nakano, and T. Tokuyama, editors, *Computing and Combinatorics*, volume 1627 of *Lecture Notes in Computer Science*, pages 103–112. Springer-Verlag, Berlin, 1999.

[27] J. Díaz, M. D. Penrose, J. Petit, and M. J. Serna. Linear orderings of random geometric graphs. In P. Widmayer, G. Neyer, and S. Eidenbenz, editors, *Graph Theoretic Concepts in Computer Science*, volume 1665 of *Lecture Notes in Computer Science*, pages 291–302. Springer-Verlag, 1999.

[28] J. Díaz, J. Petit, M. Serna, and L. Trevisan. Approximating layout problems on random graphs. *Discrete Mathematics. Elsevier North–Holland.* To appear 2000.

[29] J. Díaz, M. J. Serna, P. Spirakis, and J. Torán. *Paradigms for fast parallel approximability.* Cambridge University Press, Cambridge, 1997.

[30] R. G. Downey and M. R. Fellows. Fixed-parameter tractability and completeness II: On completeness for $W[1]$. *Theoretical Computer Science*, 141(1-2):109–131, 1995.

[31] J. Ellis, I. H. Sudborough, and J. Turner. The vertex separation and search number of a graph. *Information and Computation*, (113):50–79, 1979.

[32] G. Even, J. Naor, S. Rao, and B. Schieber. Divide and conquer approximation algorithms via spreading metrics. In *36th IEEE Symposium on Foundations of Computer Science*, pages 62–71, 1995.

[33] U. Feige. Approximating the bandwidth via volume respecting embeddings (extended abstract). In *29th. ACM Symposium on Theory of Computing*, pages 23–26, 1998.

[34] M. R. Fellows and M. A. Langston. Layout permutation problems and well-partially-ordered sets. In *Advanced research in VLSI (Cambridge, MA, 1988)*, pages 315–327. MIT Press, Cambridge, MA, 1988.

[35] M. R. Fellows and M. A. Langston. On well-partial-order theory and its application to combinatorial problems of VLSI design. *SIAM Journal of Discrete Mathematics*, 5(1):117–126, 1992.

[36] M. R. Fellows and M. A. Langston. On search, decision, and the efficiency of polynomial-time algorithms. *Journal of Computer and System Sciences*, 49(3):769–779, 1994.

[37] M. R. Garey, R. L. Graham, D. S. Johnson, and D. Knuth. Complexity results for bandwidth minimization. *SIAM Journal on Applied Mathematics*, 34:477–495, 1978.

[38] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, San Francisco, 1979.

[39] M. R. Garey, D. S. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1:237–267, 1976.

[40] F. Gavril. Some NP-complete problems on graphs. In *Proc. 11th. Conference on Information Sciences and Systems*, pages 91–95, John Hopkins Univ., Baltimore, 1977.

[41] N. E. Gibbs, W. G. Poole, Jr., and P. K. Stockmeyer. An algorithm for reducing the bandwidth and profile of a sparse matrix. *SIAM Journal Numerical Analysis*, 13(2):236–250, 1976.

[42] P. W. Goldberg, M. C. Golumbic, H. Kaplan, and R. Shamir. Four strikes against physical mapping od DNA. *Journal of Computational Biology*, 2(1):139–152, 1995.

[43] R. E. Gomory and T. C. Hu. Multi-terminal flows in a network. pages 172–199. Studies in Math., Vol. 11, 1975.

[44] G. Grimmett. *Percolation. (2nd edition).* Springer-Verlag, Heidelberg, 1999.

[45] J. Gustedt. On the pathwidth of chordal graphs. *Discrete Applied Mathematics. Combinatorial Algorithms, Optimization and Computer Science*, 45(3):233–248, 1993. ARIDAM IV and V (New Brunswick, NJ, 1988/1989).

[46] J. Haralambides and F. Makedon. Approximation algorithms for the bandwidth minimization problem for a large class of trees. *Theory of Computing Systems*, (30):67–90, 1997.

[47] J. Haralambides, F. Makedon, and B. Monien. Bandwidth minimization: an approximation algorithm for cartepillars. *Mathematical Systems Theory*, (24):169–177, 1991.

[48] F. Harary. Problem 16. In M. Fiedler, editor, *Theory of Graphs and its Applications*, page 161. Czech. Acadamy of Sciences, Prague, 1967.

[49] L. H. Harper. Optimal assignments of number to vertices. *SIAM Journal*, 12(1):131–135, 1964.

[50] L. H. Harper. Optimal numberings and isoperimetric problems on graphs. *Journal of Combinatorial Theory*, 1(3):385–393, 1966.

[51] L. H. Harper. Chassis layout and isoperimetric problems. Technical Report SPS 37–66, vol II, Jet Propulsion Laboratory, 1970.

[52] C. Helmberg, F. Rendl, B. Mohar, and S. Poljak. A spectral approach to bandwidth and separator problems in graphs. *Linear and Multilinear Algebra*, 39(1-2):73–90, 1995.

[53] B. Hendrickson and R. Leland. The Chaco User's Guide, 1995.

[54] J. Hromkovič and B. Monien. The bisection problem for graphs of degree 4 (configuring transputer systems). In *Mathematical Foundations of Computer Science*, volume 516, pages 211–220. Springer-Verlag, Lecture Notes in Computer Science, 1991.

[55] D. S. Johnson, C. R. Aragon, L. A. McGeoch, and C. Schevon. Optimization by simulated annealing: an experimental evaluation; part I, graph partitioning. *Operations Research*, 37(6):865–892, Nov. 1989.

[56] D. S. Johnson, C. Papadimitriou, and M. Yannakakis. How easy is local search? *Journal of Computer and System Sciences*, 37:79–100, 1988.

[57] M. Junguer, G. Reinelt, and G. Rinaldi. The travelling salesman problem. In C. L. M. M. Ball, T. Magnanti and G. L. Nemhauser, editors, *Handbook on Operations Research and Management Sciences*, volume 7, pages 225–330. North–Holland, 1995.

[58] M. Juvan and B. Mohar. Optimal linear labelings and eigenvalues of graphs. *Discrete Applied Mathematics*, (36):153–168, 1992.

[59] D. R. Karger. A randomized fully polynomial approximation scheme for all terminal network realiability problem. In *27th ACM Symposium on Theory of Computing*, pages 11–17, 1996.

[60] R. M. Karp. Mapping the genome: some combinatorial problems arising in molecular biology. In *24th ACM Symposium on the Theory of Computing*, pages 278–285, 1993.

[61] M. Karpinski, J. Wirtgen, and A. Zelikovsky. An approximating algorithm for the bandwidth problem on dense graphs. Technical Report TR 97-017, ECCC, 1997.

[62] D. G. Kendall. Incidence matrices, interval graphs, and seriation in archeology. *Pacific Journal of mathematics*, 28:565–570, Dec. 1969.

[63] T. Kloks, D. Kratsch, and H. Müller. Bandwidth of chain graphs. *Information Processing Letters*, 68(6):313–315, 1998.

[64] D. Kuo and G. J. Chang. The profile minimization problem in trees. *SIAM Journal on computing*, 23:71–81, 1994.

[65] K. Lang and S. Rao. Finding Near-Optimal Cuts: An Empirical Evaluation. In *4th. ACM-SIAM Symposium on Discrete Algorithms*, pages 212–221, 1993.

[66] F. T. Leighton. *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes*. Morgan Kaufmann, San Mateo, CA., 1993.

[67] T. Lengauer. Black-white pebbles and graph separation. *Acta Informatica*, 16:465–475, 1981.

[68] T. Lengauer. Upper and lower bounds on the complexity of the min-cut linear arrangements problem on trees. *SIAM Journal on Algebraic and Discrete Methods*, 3:99–113, 1982.

[69] R. Lipton and E. Tarjan. A separator theorem for planar graphs. *SIAM Journal of Applied Mathematics*, 36:177–189, 1979.

[70] W. Liu and A. Vannelli. Generating lower bounds for the linear arrangement problem. *Discrete Applied Mathematics*, (59):137–151, 1995.

[71] R. Mahesh, C. Pandu, and A. Srinivasan. On finding the minimum bandwidth of interval graphs. *Information and Computation*, 95:218–224, 1991.

[72] F. Makedon and I. H. Sudborough. On minimizing width in linear layouts. *Discrete Applied Mathematics*, 23(3):243–265, 1989.

[73] F. S. Makedon, C. H. Papadimitriou, and I. Sudborough. Topological bandwidth. *SIAM Journal on Algebraic and Discrete Methods*, 6(3):418–444, 1985.

[74] G. Mitchison and R. Durbin. Optimal numberings of an $n \times n$ array. *SIAM Journal on Discrete Mathematics*, 7(4):571–582, 1986.

[75] B. Mohar and S. Poljak. Eigenvalues in Combinatorial Optimization. In R. A. Brualdi, S. Friedland, and V. Klee, editors, *Combinatorial and Graph-Theoretical Problems in Linear Algebra*, volume 50, pages 107–151, Springer-Verlag, 1993. IMA Volumes in Mathematics and its Applications.

[76] B. Monien. The bandwidth minimization problem for caterpillars with hair length 3 is NP-complete. *SIAM Journal on Algebraic and Discrete Methods*, 7(4):505–512, 1986.

[77] B. Monien and I. H. Sudborough. Min cut is NP-complete for edge weighted trees. *Theoretical Computer Science*, 58(1–3):209–229, June 1988.

[78] D. O. Muradyan and T. E. Piliposjan. Minimal numberings of vertices of a rectangular lattice. *Akad. Nauk. Armjan. SRR*, 1(70):21–27, 1980. In Russian.

[79] P. Mutzel. A polyhedral approach to planar augmentation and related problems. In P. Spirakis, editor, *European Symposium on Algorithms*, volume 979 of *Lecture Notes in Computer Science*, pages 497–507. Springer-Verlag, 1995.

[80] K. Nakano. Linear layouts of generalized hypercubes. In J. van Leewen, editor, *Graph-theoretic concepts in computer science*, volume 790 of *Lecture Notes in Computer Science*, pages 364–375. Springer-Verlag, 1994.

[81] L. Niepel and P. Tomasta. Elevation of a graph. *Czechoslovak Math. Journal*, 31(106):475–483, 1981. In Russian (English abstract at MathSciNet).

[82] J. Pach, F. Shahrokhi, and M. Szegedy. Applications of the crossing number. *Algoritmica*, 16:111–117, 1996.

[83] C. Papadimitriou. The NP-completeness of the bandwidth minimization problem. *Computing*, 16:263–270, 1976.

[84] C. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. *Journal of Computer and System Sciences*, 43:425–440, 1991.

[85] M. Penrose. Single linkage clustering and continuum percolation. *J. of Multivariate Analysis.*, 53:94–109, 1995 POSAR LA VERSIO DE RANDOM STRUCTURES.

[86] M. D. Penrose. Vertex ordering and partition problems for random spatial graphs. Technical report, University of Durham, 1999. POSAR VERSIO BONA.

[87] J. Petit. Combining Spectral Sequencing with Simulated Annealing for the MinLA Problem: Sequential and Parallel Heuristics. Technical Report LSI-97-46-R, Departament de Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya, http://www.lsi.upc.es/~jpetit/MinLA/SS+SA, 1997.

[88] J. Petit. Approximation Heuristics and Benchmarkings for the MinLA Problem. In R. Battiti and A. A. Bertossi, editors, *ALEX'98 — Building bridges between theory and applications* . Università di Trento, 1998.

[89] R. Preis and R. Diekmann. *The PARTY Partitioning Library User Guide.* University of Paderborn, 1996. http://www.uni-paderborn.de/fachbereich/AG/monien/RESEARCH/PART/party.html.

[90] S. Rao and A. W. Richa. New Approximation Techniques for Some Ordering Problems. In *9th. ACM-SIAM Symposium on Discrete Algorithms*, pages 211–218, 1998.

[91] R. Ravi, A. Agrawal, and P. Klein. Ordering problems approximated: single-processor scheduling and interval graph completition. In M. R. J. Leach, B. Monien, editor, *18th. International Colloquium on Automata, Languages and Programming*, volume 510 of *Lecture Notes in Computer Science*, pages 751–762. Springer-Verlag, 1991.

[92] N. Robertson and P. D. Seymour. Graph minors—a survey. In *Surveys in combinatorics 1985 (Glasgow, 1985)*, pages 153–171. Cambridge Univ. Press, Cambridge, 1985.

[93] Y. Saad. *Iterative methods for sparse linear systems.* PWS Publishing Company, Boston, 1996.

[94] A. Sangiovanni-Vincentelli. Automatic layout of integrated circuits. In A. De Michelli, Sangiovanni-Vincentelli, editor, *Design Systems for VLSI circuits*, pages 113–195. Martinus Nijhoff Publishers, 1987.

[95] A. A. Schaffer and M. Yannakakis. Simple local search problems that are hard to solve. *SIAM Journal of Computing*, 20:56–87, 1991.

[96] M. T. Shing and T. C. Hu. Computational complexity of layout problems. In T. Ohtsuki, editor, *Layout design and verification*, pages 267–294, Amsterdam, 1986. North-Holland.

[97] H. D. Simon and S.-H. Teng. How good is recursive bisection? *SIAM Journal on Scientific Computing*, 18(5):1436–1445, 1997.

[98] J. S. Turner. On the probable performance of heuristics for bandwidth minimization. *SIAM Journal on Computing*, 15:561–580, 1986.

[99] W. Unger. The complexity of the approximation of the bandwidth problem. In *39th IEEE Symposium on Foundations of Computer Science*, pages 82–91, 1998.

[100] M. Yannakakis. A polynomial algorithm for the min-cut linear arrangement of trees. *Journal of the Association for Computing Machinery*, 32:950–988, 1985.

[101] J. Yuan, Y. Lin, Y. Liu, and S. Wang. NP-completeness of the profile problem and the fill-in problem on cobipartite graphs. *Journal of Mathematical Study. Shuxue Yanjiu*, 31(3):239–243, 1998.