Alcom-FT Technical Report Series
ALCOMFT-TR-03-44

Computation of the bisection width for random d-regular graphs

J. Díaz*† M.J. Serna*† N.C. Wormald‡

Abstract

In this paper we provide an explicit way to compute asymptotically almost sure upper bounds on the bisection width of random d-regular graphs, for any value of d. We provide the bounds for $5 \leq d \leq 12$. The upper bounds are obtained from the analysis of the performance of a randomized greedy algorithm to find bisections of d-regular graphs. We also give empirical values of the size of bisection found by the algorithm for some small values of d and compare it with numerical approximations of our theoretical bounds. Our analysis also gives asymptotic lower bounds for the size of the maximum bisection.

1 Introduction

Given a graph $G = (V, E)$ with $|V| = n$ and n even, a *bisection* of V is a partition of V into two parts each of cardinality $n/2$, and its *size* is the number of edges crossing between the parts. A *minimum bisection* is a bisection of V with minimal size. The decisional problem related to finding a minimum bisection is known to be NP-complete [GJ79], even for 3-regular graphs [BCLS87]. (See for example [DPS02] for further results an applications on graph bisection). The size of a minimum bisection is called the *bisection width* and the *min bisection problem* consists of finding a minimum bisection in a given G. In the present paper, we give a family of randomized algorithms which give asymptotic upper bounds as $n \to \infty$ on the bisection width of almost all d-regular graphs, where d is fixed.

Plenty of results are known on bisection width. With respect to lower bounds, in 1975 Fiedler gave a spectral lower bound of $\lambda_2 n/4$ applicable for any graph, where λ_2 is the second eigenvalue of the Laplacian of the graph [Fie75]. In 1984, Bollobás provided a lower bound of $(\frac{d}{4} - \frac{\sqrt{d \ln 2}}{2})n$, for almost all d-regular graphs [Bol84]. Later Kostochka and Melnikov proved that almost all cubic graphs have bisection width greater than $0.101n$ [KM92]. Using spectral techniques, Bezrukov et al. gave lower bounds of $0.082n$ for the bisection width of cubic Ramanujan graphs, and of $0.176n$ for the case of 4-regular Ramanujan graphs [BEM+00].

Regarding upper bounds, Kostochka and Melnikov proved that asymptotically as $n \to \infty$, all d regular graphs have bisection width of at most $\frac{d-2}{4}n + O(d\sqrt{n}\log n)$ [KM92]. Later, Alon proved that for $n > 40d^9$, all d-regular graphs have bisection width at most $(\frac{d}{4} - \frac{3\sqrt{d}}{32\sqrt{2}})n$ [Alo97]. More recently, Monien and Preis [MP01] gave upper bounds on the bisection width of

*Dept. Llenguatges i Sistemes, Universitat Politecnica de Catalunya, Jordi Girona Salgado 1–3, 08034 Barcelona, Spain

†Supported by the IST programme of the EU under contract IST-1999-14186 (ALCOM-FT)

‡Dept. Combinatorics and Optimization, University of Waterloo, Canada N2L 3G1. Supported by the Canada Research Chairs Program and partially by the Australian Research Council when this author was at the University of Melbourne.

$(\frac{1}{6}+\epsilon)n$ for 3-regular graphs and of $(0.4+\epsilon)n$ for 4-regular graphs, for any ϵ, when n is larger than some function of the chosen ϵ. To the best of our knowledge, the most recent result on bisection width was given in [DDSW03], where it was proved that the bisection width of a random 4-regular graph on n vertices is asymptotically smaller than $(\frac{1}{3}+\epsilon)n$, with probability tending to 1 (a.a.s.). This result was proved by analysing a simple greedy algorithm, a variant of which yielded a bisection of a cubic random graph on n vertices of width asymptotically almost surely smaller than $0.174039n$.

The problem of finding the maximum bisection size has also received considerable attention. This problem is again NP-hard even for planar graphs [JKLS01]. It is known to be solvable in polynomial time for graphs of bounded treewidth [JKLS01].

The maximum bisection size is a lower bound on the maximum size (number of edges) of a bipartite subgraph. Locke [Loc82] showed that a d-regular graph which is not complete or a cycle has a bipartite subgraph with at least $(nd/4)d/(d-1)$ edges if d is even, and at least $(nd/4)((d+1)/d+2/d^2)$ edges if d is odd. Shearer [SJ92] improved this result to $(nd/4)+n\sqrt{d}/8\sqrt{2}$ for triangle-free graphs, a property which a positive fraction of random regular graphs have. Our lower bounds for maximum bisection in random d-regular graphs easily exceed these bounds.

In Section 2 we present a basic randomized algorithm to find a (small) bisection of a graph by 2-colouring its vertices in a greedy way. The next vertex to be coloured is chosen according to a prioritisation scheme. The priority depends on to the status of a vertex with respect to the number of neighbours it has of either colour. Many different priority schemes were considered, each specified by a list of the types of vertices (i.e. their possible status with respect to colours of neighbours).

This prioritisation scheme is significant both as a simplification and as a generalisation of the method in [DDSW03], where only 3-regular and 4-regular graphs were considerd. It is a generalisation because, to each of the algorithms given there, there are corresponding algorithms of the general type considered in this paper which have equivalent asymptotic performance (although the algorithms do not give identical results). It is a simplification, because the method in [DDSW03] was to specify one or two main phases of the algorithm. In each main phase, two types of vertex were coloured, with one of the specified types having priority. Such detailed control of the algorithm is difficult to generalise to higher d because of the difficulty of knowing which types of vertices might be available when one phase ends and a new phase starts. The key idea in the present paper is to specifiy which types of vertices should have priority over which others, throughout the whole algorithm. The transitions between phases then become automatic. It is hard to substantiate this claim without looking at the algorithm in more detail, since even the definition of a phase becomes more delicate with this approach. A similar effect occurred in the analysis of greedy algorithms for finding independent sets in random regular graphs [Wor95], but the situation there was considerably simpler. In that case, the prioritisation was merely according to the degree of a vertex during a deletion algorithm, whilst in the present case, the best prioritisation list is much harder to determine. Moreover, in the present case, the algorithm in some sense returns to phase which it visited earlier, and this did not happen in [Wor95].

In Section 3, we *sketch* the analysis of the performance using the differential equation method. For any given d, we choose the appropriated priority list, set the equations and solve them numerically to find the asymptotic bisection width for the random d-regular graphs under consideration. In the same section, we produce empirical evidence indicating that

there are two types of optimal priority lists of the vertices: one for even values of d and the other for odd values of d. In Section 4, we give empirical results comparing the values obtained numerically from the differential equations with the bisection width obtained by the randomized greedy algorithm. In Section 5, we discuss the maximum bisection results. It should be emphasised that the main contribution of this paper is to give better asymptotic bounds for the bisection width of d-regular graphs ($d > 4$), and the algorithm produced in Section 2 is only of methodological value.

2 The priority-greedy algorithm for random d-regular graphs

In this section, we describe a family of randomized greedy procedures to find a bisection for d-regular graphs. We also introduce some generic notation to be used later.

Given a graph, and given a partial assignment of colours red (R) and blue (B) to its vertices, we classify the uncoloured vertices according to the colours of their neighbours:

> An uncoloured vertex is of *Type* (r, b), if it has r neighbours coloured R and b neighbours coloured B.

For $r \leq b$, we say that a pair of uncoloured vertices is a *symmetric pair* if their types are (r, b) and (b, r) for some r and b. We then call this an (r, b)-*symmetric* pair, or a symmetric pair of *type* (r, b).

The greedy procedure works by colouring vertices chosen randomly in symmetric pairs, to maintain balancedness, and repeatedly uses the *majority operation* (Maj), that colours each vertex of an (r, b)-symmetric pair, $r < b$, with the majority colour among its neighbours, and, given an (r, b)-symmetric pair with $r = b$, randomly colours one vertex of the pair R and the other B .

We assume that the symmetric pair types have the priorities $0, 1, 2, \ldots$ associated with them (a larger number denotes higher priority). The priority-greedy algorithm for random d-regular graphs is given in Figure 1.

Initial step: input `prio`(r, b) for all $r \leq b \leq d$;
select two non-adjacent vertices u.a.r., colour one with R and the other with B.

Main iteration:

while there is at least one uncoloured symmetric pair
  do
    let (r, b) denote the highest priority type of an uncoloured symmetric pair;
    select u.a.r. an (r, b)-symmetric pair and perform Maj;
end while

Clean up: colour any remaining uncoloured vertices, half of them R and half B, in any manner, and output the bisection R, B.

Figure 1: Algorithm priority-greedy for obtaining a bisection of a d-regular graph

This algorithm takes as input a predetermined priority list assigning a distinct priority, `prio`(r, b), to each symmetric pair type (r, b). We impose the conditions on all priority lists

that $\texttt{prio}(0,0) = 0 < \texttt{prio}(r,b)$ whenever $(r,b) \neq (0,0)$. Note that the priority of pairs (r,b) with $r+b=d$ is immaterial since colouring vertices in such pairs cannot affect the remainder of the algorithm. So for simplicity we assume that all such vertices have negative priority, and only those with $r+b<d$ need to be specified.

3 Analysis: The differential equation system derived from the priority-greedy algorithm

We follow the description in [DDSW03], extending it to the d-regular setting for arbitrary d. The algorithms considered there give equivalent results in special cases of the priority-greedy algorithm, for particular priority lists and for $d = 3$ and 4. (Notice, that the algorithms described in [DDSW03] for $d = 4$ and $d = 4$ are different than the *general* algorithm presented in this paper)

One method of analysing the performance of a randomised algorithm is to use a system of differential equations to express the expected changes in the variables describing the state of the algorithm during its execution. An exposition of this method can be found in [Wor99a], which includes various examples of graph-theoretic optimisation problems.

We use the pairing model to generate n-vertex d-regular graphs u.a.r. Briefly, to generate such a random graph, it is enough to begin with dn points in n cells, and choose a random perfect matching of the points, which we call a *pairing*. The corresponding pseudograph (possibly with loops or multiple edges) has the cells as vertices and the pairs as edges. Any property a.a.s. true of the random pseudograph is also a.a.s. true of the restriction to random graphs, with no loops or multiple edges, and this restricted probability space is uniform (see for example [Bol01, Wor99b] for a full description).

We consider the priority-greedy algorithm applied directly to the random pairing. As discussed in [Wor99a], the random pairing can be generated pair by pair, and at each step a point p can be chosen by any rule whatsoever, as long as the other point in the pair is chosen u.a.r. from the remaining unused points. We call this step *exposing* the pair containing p. At each step of the priority-greedy algorithm in which a vertex is coloured, we expose all pairs containing points in that vertex.

We now give an analysis of the algorithm which is not rigorous but will presumably yield the same bounds rigorously by introducing technical arguments as in [DDSW03]. Informally speaking, in a typical part of the algorithm, there will be symmetric pairs of one particular type, (r_0, b_0), which are plentiful in the graph but are quite regularly chosen in the main iteration of the algorithm. Symmetric pairs of types with higher priority may also be regularly chosen, but will be rare and regularly be used up entirely (at which point another pair of type (r_0, b_0) will be used). In this situation, we say that (r_0, b_0) is the *basic* type. The algorithm will typically pass through *phases*, determined by points at which, roughly speaking, the basic type changes. A phase finishes when either symmetric pairs with higher priorities than the current basic type become plentiful, or those with the current basic type become very scarce. The boundaries of the phases are best defined precisely in terms of the solution of a set of differential equations which we now proceed to derive.

At each point in the algorithm, let Z_{rb} represent the number of uncoloured vertices of type

(r, b), and let W denote the number of points not yet involved in exposed pairs. Then

$$W = \sum_{r+b<d} (d - r - b) Z_{rb}.$$

From this point onwards, we assume the reader is thoroughly familiar with the argument in [DDSW03], and omit any justifications are identical to those appearing there. Let $d_{r,b}$ denote the expected contribution to ΔZ_{rb}, the increment of Z_{rb}, due to exposing the pair containing a point in a vertex u which has just been coloured red by the priority-greedy algorithm. Then the probability that the other point chosen in the pair is in a vertex v of type (i, j) is $(d - i - j)Z_{i,j}/(W - 1)$ (except for a correction of size $O(1/W)$ due to the change in status of u). Hence, ignoring terms of size $O(1/W)$,

$$d_{r,b} = \frac{\alpha_{d+1,r+b} Z_{r-1,b} - \alpha_{d,r+b} Z_{r,b}}{W}$$

for $r + b \leq d$, where

$$\alpha_{x,y} = \begin{cases} x - y & \text{if } x > y, \\ 0 & \text{otherwise.} \end{cases}$$

In the following we continue to ignore terms of size $O(1/W)$. The equations due to the case that u is coloured blue are

$$d_{r,b} = \frac{\alpha_{d+1,r+b} Z_{r,b-1} - \alpha_{d,r+b} Z_{r,b}}{W}$$

for $r + b \leq d$. Let $\bar{d}_{r,b}$ the expected increment due to the colouring of a symmetric pair. Making the assumption of having *rb-symmetry* (for all i and j, $Z_{ij} = Z_{ji}$), and adding the effects from a point in the red vertex and a point in the blue, gives

$$\bar{d}_{r,b} = \frac{\alpha_{d+1,r+b}(Z_{r,b-1} + Z_{r-1,b}) - 2\alpha_{d,r+b} Z_{r,b}}{W},$$

for $r + b \leq d$.

Let $\phi_{r,b}$ denote the probability of processing an (r, b)-symmetric pair at some step in a given phase. This will be examined non-rigorously, since we to some extent ignore the history and current state of the process. (The equations which this heuristic argument culminates in can be used to define another process which can be analysed rigorously; see [DDSW03] for details.) Assume that at the beginning of a new phase, (r_0, b_0) is the basic type. Then these vertices are plentiful, and thus for a considerable part of the algorithm, no vertices of lower priority will be chosen for Maj. We calculate the ϕ's for that symmetric pair type and all others with higher priority.

Let B' denote the set of types of symmetric pairs with higher priority than the basic, (r_0, b_0), and let $B' = B \cup \{(r_0, b_0)\}$.

Given the assumption about the ϕ's, the expected number of points in a blue (or red) vertex when Maj is perfomed is

$$c = \sum_{(r',b') \in B} (d - r' - b') \phi_{r',b'}.$$

This leads to

$$\phi_{r,b} = c\,\bar{d}_{r,b}, \quad (r,b) \in B' \tag{1}$$

and of course we also have

$$\sum_{(r,b)\in B} \phi_{r,b} = 1. \tag{2}$$

These three equations are easy to solve, and we find

$$c = \frac{1-\phi_{r_0,b_0}}{S}, \text{ with } \quad S = \sum_{(r',b')\in B'} \bar{d}_{r',b'},$$

which yields

$$c = \frac{d-r_0-b_0}{T}, \text{ with } \quad T = 1 + \sum_{(r',b')\in B'} (r'+b'-r_0-b_0)\bar{d}_{r',b'}.$$

Now $\phi_{r,b}$ is determined from (1) for $(r,b) \in B'$, and a little computation produces from (2)

$$\phi_{r_0,b_0} = 1 + \sum_{(r',b')\in B'} (r'+b'-d)\bar{d}_{r',b'}.$$

Assuming rb-symmetry, and assuming validity of the equations for the ϕ's, the expected increments of the random variables $Z_{r,b}$ at each iteration is given by

$$\mathbf{E}\left[\Delta(Z_{r,b})\right] = c\bar{d}_{r,b} - (1+\delta_{rb})\phi_{r,b}$$

where δ_{rb} is the Kronecker delta (1 if $r=b$, 0 otherwise). The terms subtracted are due to the change in types of the symmetric pair of vertex being coloured; in the case $r=b$, two vertices of type (r,r) are lost.

As done in [DDSW03] for the case $d=4$, we may express the above expected increments as a set of differential equations, where each $\mathbf{E}\left[\Delta(Z_{r,b})\right]$ is expressed as the differential $Z'_{r,b}$ (all as functions of the number t of iterations). We scale both time and the variables by dividing by n, and denote $Z_{r,b}/n$ by $z_{r,b}$ t/n by x and W/n by w. Then the equations are

$$z'_{r,b} = (\alpha_{d+1,r+b}(z_{r,b-1}+z_{r-1,b}) - 2\alpha_{d,r+b}z_{r,b})\frac{C}{w} - (1+\delta_{rb})\theta_{r,b},$$

where $w = w(x) = \sum_{r+b\le d} \alpha_{d,r+b}z_{r,b}$, $\theta_{r,b} = \theta_{r,b}(x)$ represents $\phi_{r,b}(t/n)$ and can be defined as in (1) but with $Z_{r,b}$ replaced by $z_{r,b}(x)$ (and the same goes for $\bar{d}_{r,b}$), and $C = C(x) = \sum_{(r',b')\in B}(d-r'-b')\theta_{r',b'}$ (after manipulation of the equations above).

The increase in the size of the bisection due to a vertex of type (r,b) being coloured red is r, and the symmetric vertex being coloured blue, and of type (b,r), also increases the bisection by r. Thus, the expected increase per algorithm step is $2\sum_{(r,b)\in B} r\,\phi_{r,b}$. Letting z denote the bisection size (divided by n), this suggests the equation

$$z' = 2\sum_{(r,b)\in B} r\,\phi_{r,b}.$$

These are the differential equations for a phase with (r_0,b_0) being the basic type. The phase will end when either $\theta_{r_0,b_0} = 0$ (in which case, the basic type will have priority $\texttt{prio}(r_0,b_0)+1$)

d	5	6	7	8	9	10	11	12
list A	0.5028	0.6675	0.8502	1.0391	1.2317	1.4278	1.624	1.823
list B	0.5247	0.6674	0.8590	1.0386	1.2318	1.4278	1.624	1.823
min(A,B)	0.5028	0.6674	0.8502	1.0386	1.2317	1.4278	1.624	1.823

Table 1: Results for lists A and B, rounded up

or when z_{r_0,b_0} begins to go negative (in which case, the basic type (r, b) in the next phase will be whichever type has highest priority among those with $z_{r,b} > 0$). There is the possibility of a phase of zero length, if these criteria immediately apply to the new basic type; then the next basic type can be determined using the same rule. The variables' initial conditions at the start of a phase are just their values at the end of the previous phase. The whole calculation begins with basic type $(0, 1)$ and with all variables equal to 0, except for $z_{0,0} = 1$. The size of the bisection is represented by the value of z (scaled up by a factor n) when the values of all the $z_{r,b}$ reach 0 simultaneously. The last few phases are those in which the basic type (r_0, b_0) has $r_0 + b_0 = d$, and in practice, these may be skipped if the appropriate quantity is added to z.

After trying many different priority lists, solving the resulting system of differential equations using a Runge-Kutta method of order 2, we focussed on two priority lists which appear to give the best results. The following determine the order amongst those (r, b) with $r + b < d$:

List A: $\texttt{prio}(i, j) > \texttt{prio}(k, l)$ iff $j - i < l - k$ or $(j - i = l - k$ and $i > k)$.

List B: Same as List A but swapping $\texttt{prio}(0, 2)$ with $\texttt{prio}(\lfloor d/2 \rfloor - 1, \lfloor d/2 \rfloor)$.

For example, with $d = 5$, List A places the types in the following order: (0,0), (1,1), (2,2), (0,1), (1,2), (0,2), (1,3), (0,3), (0,4); List B puts (0,2) before (1,2) but retains all other relative rankings.

List A appears, from the results of the calculations, to perform better for d odd and List B performs better for d even. However, for larger d, this is not clearly demonstrated to the accuracy with which we can confidently quote the results (due to errors inherent in numerical solution of the differential equations). The bounds obtained with Lists A and B for $d \leq 12$ are given in Table 1. Machine power was too limiting to go much further than this with sufficient accuracy, but the further digits obtained, which we do not report here, suggested that the ranking of Lists A and B according to the parity of d continues, at least up to $d = 12$.

As mentioned above, this argument is not rigorous, and in particular the concept of the probability measured by $\phi_{r,b}$ was not well specified. In the remainder of this section we sketch how the results could be turned into rigorous upper bounds using the type of argument in [DDSW03].

A major complicating factor is that the rate of change of variables is not smooth: when a vertex of one type is coloured, the effects are different from that of another type being processed. The values of the ϕ's can be estimated as in [Wor95] by breaking the process up into pieces called *clutches* in [DW02], separated by the steps in which pairs of basic type are processed. Alternatively, the number of pairs of each type in a clutch can be estimated and an expression found for $\phi_{r,d}$. The heuristic argument above can be justified in a more

direct way by considering a different, deprioritized algorithm as in [Wor03]. At the beginning of the deprioritized algorithm, the first ϵn steps each randomly choose a pair of type $(0,0)$ and applies Maj. This produces a plentiful supply of vertices of all types. Then, during the algorithm proper, the type of symmetric pair to be coloured in a given step is chosen randomly with probability $\phi_{r,b}$ for type (r,b), where ϕ is calculated usings equations (1) and (2). The expected changes in the values of the $Z_{i,j}$ in one step can then be calculated (asymptotically), given their values at the beginning of the step. The differential equations we have derived above apply, but with the slightly different initial conditions determined by ϵ. When the variable z_{r_0,b_0} corresponding to the basic type of symmetric pair reaches 0, or the corresponding θ_{r_0,b_0} reaches 0, the phase ends. Inductively, one may apply the differential equation method (see [Wor99b]) to show that the variables of the process with probability $1+o(1)$ follow the solution of the differential equations in each phase; that is, $Z_{r,b}(t) = nz_{r,b}(t/n) + o(n)$ for all relevant t. Finally, it can be shown that letting $\epsilon \to 0$, the differential equation solution trajectory for the deprioritized algorithm approaches arbitrarily close to the trajectory for the original algorithm. The desired result follows from this.

4 The experimental upper bounds

We have also generated a set of d-regular graphs for each $d = 5$ to 12, following the method described in [StWo99]. We repeated the algorithm 10 times on each of the graphs, with priorities given by List A for d odd and List B for d even. The results, for five graphs with 10^5 vertices and two graphs with 2×10^5 vertices, for each $d = 5,\ldots,12$ are summarized in Table 2. The left column of the table includes the bound via the differential equations. The mean, max and min of the bisection values obtained using Algorithm 1 are given for each graph, and the means are also averaged over all graphs of each of the two sizes. For any reader interested in checking the experiment, the graphs generated can be found at:
http://www.lsi.upc.es/~mjserna/dregraphs.html

5 Maximum Bisection

Let us consider the variation of the priority-greedy algorithm obtained replacing the majority operation (Maj) with the minority operation, that asigns to a vertex the color minority among its colored neighbors. Let us call this variation max priority-greedy

Define an edge to be *fully colored* when both its ends are finally colored. A fully colored edge is *monocolored* if both ends have the same color and *bicolored* if both ends have different color. So the edges monocolored by priority-greedy get bicolored by max priority-greedy and vice versa, whenever the vertices of the graph are treated in the same order (which happens with the same probability, in both cases). That is, every edge that counts in the bisection for one algorithm does not count in the other and vice versa. So, taking into account that the total number of edges in a d-regular graph is $dn/2$, we have the following complementary bounds for the maximum bisection: the size of the maximum bisection in a random d-regular graph is a.a.s. at least $dn/2 - c_d n$, where c_d is the min value given in Table 1 in column d.

		e5-1	e5-2	e5-3	e5-4	e5-5	av:e5	2e5-1	2e5-2	av:2e5
d=5	avg	0.5046	0.5087	0.5042	0.5046	0.5038	0.5052	0.5036	0.5040	0.5038
	max	0.5060	0.5463	0.5060	0.5061	0.5060		0.5045	0.5052	
0.5028	min	0.5026	0.5026	0.5019	0.5035	0.5025		0.5032	0.5031	
d=6	avg	0.6692	0.6710	0.6695	0.6690	0.6689	0.6695	0.6687	0.6691	0.6689
	max	0.6719	0.6849	0.6720	0.6724	0.6711		0.6692	0.6703	
0.6674	min	0.6675	0.6682	0.6670	0.6672	0.6675		0.6677	0.6683	
d=7	avg	0.8517	0.8515	0.8517	0.8522	0.8516	0.8517	0.8511	0.8516	0.8513
	max	0.8536	0.8534	0.8541	0.8530	0.8545		0.8533	0.8524	
0.8502	min	0.8443	0.8482	0.8502	0.8511	0.8498		0.8496	0.8509	
d=8	avg	1.0410	1.0397	1.0403	1.0406	1.0404	1.0407	1.0407	1.0402	1.0404
	max	1.0438	1.0420	1.0426	1.0427	1.0422		1.0432	1.0416	
1.0386	min	1.0388	1.0378	1.0384	1.0379	1.0378		1.0392	1.0384	
d=9	avg	1.2340	1.2339	1.2336	1.2333	1.2335	1.2336	1.2325	1.2342	1.2333
	max	1.2374	1.2366	1.2353	1.2352	1.2357		1.2331	1.2364	
1.2317	min	1.2312	1.2306	1.2313	1.2308	1.2309		1.2316	1.2326	
d=10	avg	1.4292	1.4303	1.4289	1.4296	1.4300	1.4296	1.4291	1.4288	1.4289
	max	1.4326	1.4327	1.4299	1.4315	1.4333		1.4300	1.4301	
1.4278	min	1.4265	1.4293	1.4275	1.4280	1.4281		1.4283	1.4272	
d=11	avg	1.6257	1.6256	1.6256	1.6256	1.6262	1.6258	1.6251	1.6254	1.6254
	max	1.6276	1.6276	1.6286	1.6273	1.6288		1.6264	1.6270	
1.624	min	1.6227	1.6234	1.6236	1.6240	1.6239		1.6243	1.6240	
d=12	avg	1.8245	1.8248	1.8241	1.8241	1.8257	1.8246	1.8241	1.8244	1.8244
	max	1.8264	1.8273	1.8270	1.8260	1.8273		1.8248	1.8254	
1.823	min	1.8231	1.8214	1.8226	1.8226	1.8223		1.8237	1.8236	

Table 2: Size of the bisection obtained by the greedy algorithm for five graphs with $n = 10^5$ (e5-*) and two graphs with $n = 2 \times 10^5$ (2e5-*). The asymptotic almost sure upper bound from the differential equation analysis is given in the left column.

6 Conclusions and open problems

As mentioned above, in [DDSW03] there is an analytic expression for the bound obtained on the bisection width of a random 4-regular graph, obtained from differential equations corresponding to the priority-greedy algorithm. When run for $d = 4$, Lists A and B give the same theoretical result, because the types $(0, 2)$ and $(1, 2)$ never become basic: there is only one phase, with $(0, 1)$ basic (see [DDSW03], where the algorithm is expressed in a different way but gives the same differential equations).

In this paper we have proposed a randomized greedy procedure which bounds the bisection width of any d-regular graph, and analyzed its typical performance on random d-regular graphs. The algorithm uses a predefined list of priorities. Furthermore, a related algorithm shows complementary bounds for the maximum bisection size. We sketch a proof that for any given list, and any $d \geq 3$, the values of the size of the bisection obtained by the algorithm are concentrated around the value determined by the solution of a set of differential equations.

Experimentally, we notice that a good list of priorities is given by List A for $d \geq 6$ even and List B for $d \geq 5$ odd. It remains an open problem to search for other possible lists of priorities that improve the outcome of the algorithm. In Table 1, we get the asymptotic bisection width as solution to the differential equations (the tables reflect the constant to be multiplied by n). We may compare with the asymptotic lower bound of Bollobás and the asymptotic value of the upper bound of Alon, which is a deterministic one. For instance, for $d = 5$, Bollobás' lower bound yields $0.31917n$, Alon's upper bound yields $1.15118n$, while our upper bound is $0.5028n$. Furthermore, the complementary lower bounds we get on max bisection are well above the known lower bounds for all d-regular graphs, triangle-free or not.

Moreover, as can bee seen from Table 2, even for rather small values of n, the size of the bisection obtained by the algorithm is close to the solution determined by the differential equations. As n grows, this phenomenon strengthens.

Several open problems remain, the first being to improve the upper bound. One way to do this may be to find better priority lists. Another question is whether there is some simpler way to analyse these greedy algorithms rigorously.

References

[Alo97] Noga Alon. On the edge-expansion of graphs. *Combinatorics, Probability and Computing*, (6):145–152, 1997.

[BEM+00] S.L. Bezroukov, R. Elsasser, B. Monien, R. Preis, and J.P. Tillich. New spectral lower bounds on the bisection width of graphs. In U. Brandes and D. Wagner, editors, *26th. Graph-Theoretic Concepts in Computer Science*, volume 1928, pages 23–34. Springer, Lecture Notes in Computer Science, 2000.

[Bol84] B. Bollobás. The isoperimetric number of random regular graphs. *European Journal of combinatorics*, 9:241–244, 1984.

[Bol01] B. Bollobás. *Random Graphs. Second Edition.* Combridge University Press, 2001.

[BCLS87] T. Bui, S. Chaudhuri, T. Leighton, and M. Sipser. Graph bisection algorithms with good average case behavior. *Combinatorica*, 7:171–191, 1987.

[DDSW03] J. Díaz, N. Do, M. J. Serna, and N. Wormald. Bounds on the max and min bisection of random cubic and random 4-regular graphs. *Theortical Computer Science*, to appear, 2003.

[DPS02] J. Díaz, J. Petit, and M. J. Serna. A survey on graph layout problems. *ACM Computing Surveys*, 34:313–356, 2002.

[DW02] W. Duckworth and N.C. Wormald Minimum independent dominating sets of random cubic graphs. *Random Structures and Algorithms*, 21:147–161, 2002.

[Fie75] M. Fiedler. A property of the eigenvectors of nonnegtive symetric matrices and its application to graph theory. *Czechoslovak Mathematical Journal*, (25):619–633, 1975.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, San Francisco, 1979.

[JKLS01] K. Jansen, M. Karpinski, A. Lingas, and E. Seidel. Polynomial time approximation schemes for MAX-BISECTION on planar and geometric graphs.

[KM92] A.V. Kostochka and L.S. Melnikov. On bounds of the bisection width of cubic graphs. In J. Nesetril and M. Fiedler, editors, *Fourth Czechoslovakian Symposium on Combinatorics, Graphs and Complexity*, pages 151–154. Elsevier Science Publishers, 1992.

[Loc82] S. C. Locke. Maximum k-colorable subgraphs. *J. Graph Theory* 6(2):123–132, 1982.

[MP01] B. Monien and R. Preis. Upper bounds on the bisection width of 3- and 4-regular graphs. In A. Pultr J. Sgall and P. Kolman, editors, *Mathematical Foundations of Computer Science*, volume 2136, pages 524–536. Springer, Lecture Notes in Computer Science, 2001.

[SJ92] Shearer, James B.(1-IBM) A note on bipartite subgraphs of triangle-free graphs. *Random Structures Algorithms* 3(2):223–226, 1992.

[StWo99] A. Steger and N.C. Wormald. Generating random regular graphs quickly. *Combinatorics, Probabability and Computing*, 8:377–396, 1999.

[Wor95] N.C. Wormald. Differential equations for random processes and random graphs. *Annals of Applied Probability*, 5:1217–1235, 1995.

[Wor99a] N.C. Wormald. The differential equation method for random graph processes and greedy algorithms. In M. Karoński and H. Prömel, editors, *Lectures on Approximation and Randomized Algorithms*, pages 73–155. PWN, Warsaw, 1999.

[Wor99b] N.C. Wormald. Models of random regular graphs. In *Surveys in Combinatorics*, pages 239–298. Cambridge University Press, 1999.

[Wor03] N.C. Wormald. Analysis of greedy algorithms on graphs with bounded degree. *Discrete Mathematics*, 2003.