COMBINATORICA
Bolyai Society – Springer-Verlag

COMBINATORICA **38** (3) (2018) 619–664
DOI: 10.1007/s00493-016-3236-x

LOCAL ALGORITHMS, REGULAR GRAPHS OF LARGE GIRTH, AND RANDOM REGULAR GRAPHS

CARLOS HOPPEN*, NICHOLAS WORMALD†

Received February 15, 2014
Revised January 17, 2016
Online First April 17, 2018

We introduce a general class of algorithms and analyse their application to regular graphs of large girth. In particular, we can transfer several results proved for random regular graphs into (deterministic) results about all regular graphs with sufficiently large girth. This reverses the usual direction, which is from the deterministic setting to the random one. In particular, this approach enables, for the first time, the achievement of results equivalent to those obtained on random regular graphs by a powerful class of algorithms which contain prioritised actions. As a result, we obtain new upper or lower bounds on the size of maximum independent sets, minimum dominating sets, maximum k-independent sets, minimum k-dominating sets and maximum k-separated matchings in r-regular graphs with large girth.

1. A brief introduction

The effect of large girth on other graph parameters has been of interest at least since Erdős [9] showed that, for any given positive integers k and g, there is a graph with girth at least g and chromatic number at least k, evincing the global character of the chromatic number of a graph. In this paper, we consider r-regular graphs, where $r \geq 2$ is fixed. For such bounded degree graphs, naturally the chromatic number is bounded. Still, various studies have considered the effect of increasing girth on graph properties such

Mathematics Subject Classification (2000): 05C35, 05C80, 05C69, 05C85

* Supported by FAPERGS (Proc. 2233-2551/14-0), CNPq (Proc. 448754/2014-2 and 308539/2015-0) and FAPESP (Proc. 2013/03447-6).

† Research supported by the Canada Research Chairs Program, NSERC and the Australian Laureate Fellowship program of the ARC.

as chromatic number or the size of the largest independent set or smallest dominating set.

We introduce a new approach to such questions by considering a general class of algorithms and showing a relationship between their behaviours on random regular graphs and on regular graphs of large girth. By suitably choosing the algorithm to produce an appropriate structure, we then obtain upper or lower bounds on a variety of well studied graph parameters for r-regular graphs of large girth. For some of these parameters, such as the size of the maximum independent set or minimum dominating set, the new bound comes directly using known results on random regular graphs. Broadly speaking, we show that, for an algorithm $\mathcal{A}$ belonging to a quite large class of algorithms, the size of the structure produced by $\mathcal{A}$ is almost the same for r-regular graphs of very large girth, as it is for a random r-regular graph. This can be viewed as a partial converse of existing results which translate properties of regular graphs of large girth into highly likely properties of random regular graphs. Throughout this paper, a statement holding for graphs 'of sufficiently large girth' means that there exists g such that the statement holds when the girth of the graphs under consideration is at least g.

Our methods, relating to random regular graphs, are very new, and have their basis in a much less sophisticated one introduced by Lauer and Wormald [21]. There, it was shown that a simple greedy algorithm for finding independent sets behaves essentially the same on regular graphs with large girth as on random regular graphs analysed by Wormald [27]. However, stronger bounds were also obtained for random regular graphs by analysing more sophisticated algorithms. The challenge then was obvious, to show that the stronger bounds also hold for all regular graphs of large girth. For this, a simple extension of the methods of [21] does not suffice, as will be explained in the next section. In this paper, we define a specific class of algorithms which we call *local deletion algorithms* and use a system of differential equations to track their progress. Using the fact that the differential equation system is the same for both large girth and random regular graphs, we eventually obtain the desired results.

In order to encompass the applications dealt with here and in the follow-up paper [16], we supply a number of general results useful for analysing local deletion algorithms. We define local deletion algorithms to a level of generality that is sufficient for our results to apply to many problems, establishing many new bounds, some of which improve upon existing ones. In particular, we obtain new bounds on the size of maximum independent and k-independent sets, minimum and k-dominating dominating sets, and

maximum k-separated matchings in r-regular graphs with large girth. In this paper we restrict applications to results that follow easily by combining our main theorems with existing analysis of algorithms on random regular graphs, deferring new analysis of algorithms to the follow-up paper [16], which treats maximum cuts, minimum and maximum bisection and minimum connected and weakly-connected dominating sets, as well as improvements in the case $r=3$ for independent sets. (See also [15], which is a preprint of both papers combined.) In future applications, our general results will allow the analysis of algorithms in a more or less automated manner.

Hopkins and Staton [13] gave lower bounds on the size of independent sets in cubic (i.e., 3-regular) graphs with large girth, superseded by Shearer's bounds [26] for r-regular graphs in general. For cubic graphs on n vertices (and sufficiently large girth), this bound was $0.4139n$. Shearer's bounds were improved in [21] for all $r \geq 7$, and basically all (up to a value of r determined by computations) are further improved in the present paper. These bounds first appeared in the first author's thesis [14] (which was supervised by the second author) in a chapter forming an embryonic version of the present work. Our theory now permits a much more economic derivation (see Section 7). In the cubic case this bound is $0.4328n$. Kardoš, Král and Volec [17] followed key ideas in [14] to improve the bound to $0.4352n$. This result was yet again improved, to $0.4361n$, very recently by Csóka, Gerencsér, Harangi and Virág [4] using invariant Gaussian processes on the d-regular tree. A further improvement for cubic graphs comes in [16], where the lower bound $0.4375n$ is established.

Regarding dominating sets, Král, Škoda and Volec [20] showed that a cubic graph of sufficiently large girth has a dominating set of size at most $0.299871n$. We improve this here, and for higher degrees we give the first bounds specifically obtained for large girth. These improve on a general upper bound due to Reed [25] and even on refinements thereof obtained by Kawarabayashi, Plummer and Saito [19] for graphs with a 2-factor. Additionally, our upper bounds are stronger in the sense that they also hold for minimum *independent* dominating sets. Strangely perhaps, relaxing the independence condition does not produce any easy significant improvement.

2. Introduction to the general results

In this work, we often consider some given property P that a set of vertices of an input graph G might have, and consider the function f_P and a constant $c(r,P)$ such that

$$f_P(G) := \max\{|U| \colon U \subseteq V(G) \text{ and } U \text{ satisfies } P\} \geq c(r,P)n$$

for every n-vertex r-regular graph G with sufficiently large girth. Given positive integers $n>r$ (where nr is even for feasibility), consider the probability space $\mathcal{G}_{n,r}$ of all r-regular graphs with vertex set $V=\{1,\dots,n\}$ with uniform probability distribution. It is well known (see Bollobás [3], Wormald [29] for example) that, for fixed integers r and g, as $n\to\infty$, the probability that a random graph in $\mathcal{G}_{n,r}$ has girth at least g tends to a positive constant determined by r and g. Immediately then, if a random r-regular graph a.a.s. has no sets U satisfying P of cardinality at least $c_u(r,P)n$, then $c(r,P)$ must be less than $c_u(r,P)$. (A sequence of events A_n occurs *asymptotically almost surely* (a.a.s.) if $\lim_{n\to\infty}\mathbf{P}(A_n)=1$.)

Furthermore, the expected number of vertices in $\mathcal{G}_{n,r}$ that lie in cycles of length at most g is bounded. A coarse consequence of this is the following. Assume that $|f_P(G)-f_P(G')|<C$ (C constant) if G' comes from G by deleting a bounded number of vertices and edges. Then, for every $\delta>0$, a.a.s. $f_P(G')\geq(c(r,P)-\delta)n$ for $G'\in\mathcal{G}_{n,r}$. As a partial converse (also following from [3], [29]), if $G'\in\mathcal{G}_{n,r}$ a.a.s. satisfies a property, then for every fixed $g>0$, a random r-regular graph with girth at least g a.a.s. satisfies Q. However, Q holding a.a.s. in $\mathcal{G}_{n,r}$ does not imply that Q holds for *all* r-regular graphs with girth sufficiently large. For instance, consider connectedness. See [29] for basic results on random regular graphs.

The class of local deletion algorithms considered here was motivated by a series of algorithms used to study functions f_P for random regular graphs. These algorithms often follow the general description of [30]: they proceed by rounds, where, at each round, some basic operation is performed. A basic operation may be of several types, called Op_i ($1\leq i\leq r$), consisting of selecting a vertex v of degree i u.a.r., and then applying a specified sequence of randomised tasks (including deletion of v). These operations need to satisfy some additional technical conditions in order for the analysis in [30, Theorem 1] to go through. For later reference, algorithms to which [30, Theorem 1] applies will be called *degree-governed query algorithms*, or DGQ algorithms for short. Examples of these include the following two randomised procedures, for large independent sets and for small dominating sets. (An *independent set* in a graph G is a set $S\subset V(G)$ such that no edge joins two vertices in S. A *dominating set* in G is a set $T\subset V$ such that every $v\in V(G)$ lies in T or has a neighbour in T.)

The procedure P_{ind} looks for a large independent set. It inductively defines a *survival graph* G_t for $t\geq 1$, starting with $G_0=G$. The procedure chooses a vertex v uniformly at random among the vertices G_{t-1}, and adds v to the independent set. Define G_t by removing v and all its neighbours from G_{t-1}. The procedure continues until G_t is empty. A more sophisticated

version of this procedure gives priorities to some vertices; for instance, we might restrict the choice of v to vertices *of minimum degree* in G_{t-1}. Such procedures are called *prioritised* algorithms in [30]. For dominating sets, consider the prioritised procedure P_{dom}, starting with $G_0 = G$. At every step $t \geq 1$, it chooses a vertex v uniformly at random (u.a.r.) among the vertices of minimum degree, i, in G_{t-1}. If $i = 0$, add v to the dominating set and remove it from G_{t-1}. Otherwise choose a vertex w u.a.r. among all neighbours of v with largest degree, and add w to the dominating set. The survival graph G_t is obtained by removing w and all its neighbours from G_{t-1}.

A powerful method for analysing random processes, known as the "differential equation" method, was presented by the second author in [27,28] and applied DGQ algorithms on random regular graphs in [30]. To apply this method to such algorithms, one needs to compute the expected changes of some variables associated with the survival graph in a single step of the algorithm conditional on the values of the variables and on the operation used. This leads to a system of differential equations whose solutions track the progress of the algorithm. From these solutions, one may derive bounds on graph functions that hold a.a.s. for random regular graphs. We show in the present paper that the same bounds apply deterministically for regular graphs of sufficiently large girth, for a broad class of algorithms which includes all cases where the main result in [30] has been used to date.

In the approach introduced in [21] for the large girth problem, a randomised algorithm is defined that outputs a set S with a property P when applied to a fixed r-regular input graph G of girth at least g, such that the expected size of S does not depend on the choice of G. This gives bounds on the maximum and minimum cardinalities of a set satisfying P. Here, and in the ensuing papers such as [17], expected output sizes are computed via recurrence relations and using 'independence lemmas': in the neighbourhood of a given vertex, certain events are independent. In some cases, finding appropriate independent events is difficult.

Our main goal is to adapt the idea in [21] to the much more powerful algorithms treated in [30]. The main difficulty is that they use different operations in different steps, prioritising the choice of the vertices in the current survival graph according to their degree. For this reason, they are called *prioritised algorithms*. On the other hand, the analysis in [21] requires a well defined operation at a given time. To deal with this discrepancy, we essentially 'deprioritise' the algorithms as in [30]. After this transformation, each step is well defined in advance, consisting of an operation chosen at random under a given probability distribution. Additionally, we avoid the independence lemmas by proving a relationship with random regular graphs,

where a much stronger independence property holds (see Lemma 5.3). This eliminates the need to find suitable independent events on a case by case basis, enabling easy evaluation of the relevant probabilities. There is a catch: since random regular graphs contain short cycles, we replace the simple first moment method of [21] by sharp concentration arguments.

The main results in the present paper provide the desired direct connection between random regular graphs and graphs with large girth, showing how to transfer some particular results for random graphs into deterministic ones for all regular graphs with sufficiently large girth. This lets us widen the class of algorithms studied to much more powerful ones, as detailed in the next section.

Before stating our main methodological results, we put the algorithms P_{ind} and P_{cut} into a more general perspective. They may be viewed as examples of *local deletion algorithms*, defined formally in Section 3. Informally, their main features are as follows, where G is the input graph.

1. Let $G_0 = G$. The algorithm obtains a set $\mathcal{O}$ iteratively. At each step $t \geq 1$, there is a *selection step*, in which a set S_t of vertices is chosen at random in the survival graph G_{t-1} according to some distribution.
2. For each vertex v chosen in the selection step, there is an *exploration step*. This checks one by one the degrees of neighbours of a vertex, which we loosely call *exploring* the vertex and *exposing* its neighbours, and selects a new vertex to be explored, from those that have already been exposed, according to some randomised rule. The rule restricts exploration to vertices within some fixed distance of v.
3. The *insertion step* adds some subset of the vertices explored, or of the edges incident with them, to the set $\mathcal{O}$. All vertices explored are deleted from the survival graph.
4. Both the exploration step and the insertion step should depend only on the isomorphism type of the explored neighbourhood.

Since many vertices might be chosen in a selection step, large output sets may be obtained in a bounded number of steps. This can result in 'clashes', where nearby vertices are both selected in the same round, leading to conflicts in the insertion step which must be resolved.

We obtain stronger results in some cases by weakening the requirement that the entire neighbourhood of an explored vertex is exposed in the exploration step. Vertices for which the entire neighbourhood is exposed will be called *totally explored*, while vertices whose neighbourhood has been partially exposed will be called *partially explored*.

We also extend the scope so as to apply to vertex-coloured graphs. (In general these are not proper colourings.) The *type* of a vertex is the ordered

pair consisting of its degree and its colour. Where 'degree' is mentioned above, we may read 'type.' The insertion step becomes a *recolouring step*, where each totally explored vertex is assigned one of a finite set of 'output colours', and the colours of their neighbours in the survival graph may be changed. If an output set is desired, of course one output colour can be used to designate it. A local deletion algorithm with no vertex colours (as above) is *native*. For instance, the algorithms for independent and dominating sets described above are both native. An example of a non-native algorithm, using colours, is given at the end of this section, along with one that relies for its power on exploring a large neighbourhood of the selected vertex.

A local deletion algorithm is *chunky* if, at each round $t \geq 1$, the selection step chooses each vertex of G_{t-1} independently with a probability $p_{t,i}$ depending on its degree i. Our main results analyse chunky local deletion algorithms for regular graphs with large girth and tie their performance to that of algorithms previously considered in the random regular setting. Indeed, we prove (see Theorem 4.3) that the expected size of the output set of a chunky local deletion algorithm is the same for every r-regular graph G whose girth is sufficiently large. Here 'sufficiently' depends on the number of steps N taken by the algorithm. Moreover, we demonstrate (see Lemma 4.4) that the algorithm performs almost the same when the input graph contains a small number of short cycles. Consequently, the expected performance of a chunky local deletion algorithm for a fixed input graph with large girth is tied to its expected performance on $\mathcal{G}_{n,r}$ (see Theorem 4.5).

In light of this equivalence, we concentrate on $\mathcal{G}_{n,r}$. The values of a set of variables associated with an application of a chunky local deletion algorithm a.a.s. track the solutions of a d.e. system (see Theorem 5.4), which may be obtained explicitly using the ideas in the proof of Lemma 5.2. In Theorem 5.5 we describe a general strategy for analysing some special chunky algorithms that will eventually mimic deprioritised algorithms. This analysis uses $\mathcal{G}_{n,r}$ and then transfers via Theorem 4.5 to the large girth case.

In Section 6, we show that algorithms in a large class considered previously in the random regular setting can be 'chunkified' into chunky local deletion algorithms with similar performance (see Theorem 6.2). This is the final ingredient for deriving deterministic bounds for graphs with large girth from previous analysis of $\mathcal{G}_{n,r}$. In particular, we extend [30, Theorem 1] to the large girth context. This determines an algorithm's performance in terms of the solutions of differential equations (see Theorem 7.1). The rest of Section 7 gives applications of this result, translating known results on random regular graphs into results on regular graphs of sufficiently large girth. Finally, we make some comments on extensions of our results in Section 8.

We close this section by describing two more examples of local deletion algorithms.

The first, P_{cut}, uses colours, and is inspired by the algorithm in [18] for finding a large cut in a cubic graph. Given $A \subseteq V$, the *(edge) cut* induced by A is the set of all edges in E with one endpoint in A and the other in $V \setminus A$. We need two output colours, red and blue, which indicate the vertices assigned to each class of the bipartition. Throughout the algorithm, each vertex v of the survival graph will have a colour rb, short for (r, b), where r and b specify the numbers of neighbours of v in the input graph that have been coloured red and blue respectively. Hence a vertex of color rb has degree $3 - r - b$, and so its colour determines its type. There is a priority list of types. At step t, the algorithm selects u.a.r. a vertex v from those of highest priority type in the survival graph G_{t-1}. If $r > b$, v is coloured blue and deleted from G_{t-1}, and the types of its neighbours are updated with an additional blue neighbour. If $r < b$, the same thing happens with blue and red interchanged and if $r = b$, v is assigned red or blue uniformly at random and deleted from the survival graph. When the survival graph becomes empty, the output is the cut consisting of all edges in the original graph for which one endpoint is now red and the other is blue.

The second, P^*_{ind}, is a native local deletion algorithm for finding an independent set in a cubic graph, defined for illustration purposes only. The selection step chooses a vertex v with minimum positive degree in G_{t-1}. Unless v has degree 2 and has some degree 2 neighbour, the algorithm explores the neighbours of v, adds v to the independent set and deletes it along with its neighbours from the survival graph. Otherwise, v has degree 2 and has some degree 2 neighbour. The algorithm picks one such neighbour, u, to explore. If u's other neighbour, w, has degree 2, the algorithm explores w and adds u to the independent set (whilst the second neighbour of v is left 'unexplored'). Otherwise v is added to the independent set and its second neighbour is explored. All explored vertices (including v) are deleted from the survival graph.

P_{cut} and an improved version of P^*_{ind} are analysed in [16] using the results of this paper.

3. Definition of local deletion algorithms

Local deletion algorithms are formally defined in Section 3.1, and the chunky ones in Section 3.2. Since our topic is graphs of large girth, in this section all graphs are simple, i.e. have no loops or multiple edges.

3.1. Definitions for the general case

Let D be a positive integer. We use $d(u,v)=d_G(u,v)$ for the distance between u and v in G, and $d(u,H)=d_G(u,H)=\min\{d_G(u,v)\colon v\in H\}$. The algorithms are defined after several preliminary definitions.

Definition (Transient, output and neutral colours, coloured graph, type of a vertex). Assume there are two sets of colours, the set $\mathcal{C}$ of *transient colours*, to be assigned to the vertices in the survival graph, and the set $\mathcal{E}$ of *output colours*, to be assigned to vertices when deleted. A special transient colour, called *neutral*, is initially assigned to all vertices. The sets $\mathcal{C}$ and $\mathcal{E}$ each have another special colour, denoted by $\propto'$ and $\propto$, respectively, which are used for clashes. A *coloured graph* refers to a graph whose vertices are assigned colours from $\mathcal{C}$. Given a coloured graph G and a vertex $v\in V(G)$, the *type* $\tau_G(v)$ of v is the ordered pair (c,d), where c is the colour and d is the degree of v in G.

In the selection step, $\pi_G(S)$ will be the probability of selecting a set S of vertices.

Definition (Selection rule). A selection rule is a function $\mathit{\Pi}$ that, for a nonempty coloured graph $G=(V,E)$, gives a probability distribution π_G on the power set of V with the properties that

(i) $\sum_{v\in S\subseteq V}\pi_G(S)=\sum_{w\in T\subseteq V}\pi_G(T)$ for every $v,w\in V$ such that $\tau_G(v)=\tau_G(w)$;
(ii) $\pi_G(S)=0$ if S contains any vertex whose colour is $\propto'$.

For the exploration step, we introduce the concept of query graph. The multiset ℓ_v denotes the list of types of neighbours of v as currently determined by the algorithm, where the as-yet undetermined types are signified by $\diamond$. The non-$\diamond$ types are called *vertex types*.

Definition (Query graph, depth, root). An h-vertex query graph is a nonempty coloured graph H with vertex set $\{1,\dots,h\}$ such that each vertex $v\in V(H)$ is associated with a finite (possibly empty) multiset ℓ_v, each of whose elements is a type (either vertex type or $\diamond$). A query graph has depth D if every vertex is at distance at most $D-1$ from the vertex with label 1, which is called its root.

The set of all query graphs is denoted by $\mathcal{Q}$, whilst the set of query graphs of depth D is denoted by $\mathcal{Q}_D$. To avoid ambiguity arising from automorphisms, a copy of a query graph H in a coloured graph G is formally defined as follows.

Definition (Copy). Given a coloured graph G and a query graph H, a copy of H in G rooted at $v \in V(G)$ is a (graph theoretical) colour-preserving isomorphism $\psi\colon V(H) \to V(G)$ from H to a subgraph of G with $\psi(1) = v$ and carrying the following additional information. For every $w \in V\big(H\big)$, each vertex type in ℓ_w is associated with a vertex of the same type that is adjacent to $\psi(w)$ via an edge of $G - E\big(\psi(H)\big)$. Moreover, the number of unassociated neighbours of $\psi(w)$ is required to equal the number of occurrences of $\diamond$ in ℓ_w.

A copy ψ of a query graph will be used to record the information obtained part-way through an exploration step. The names $1, 2, \ldots$ of the vertices in the query graph record the order in which the vertices are explored in the copy, the root being first. The exploration step performs repetitions of a basic query operation, defined as follows. Note that occurrences of $\diamond$ in a multiset ℓ_w are not yet associated with any vertex.

Definition (Open adjacencies, querying, and exposing an edge). Let G be a coloured graph, H a query graph and ψ a copy of H in G. Define $U_i = \{(i,\tau)\colon i \in V(H) \text{ and } \tau \in \ell_i\}$. An element $(i,\tau) \in U_i$ is called an *open adjacency* of i with type τ. If τ is a vertex type, *querying* the open adjacency (i,τ) consists of selecting u.a.r. a neighbour, u, of $\psi(i)$ that is associated with a copy of τ. We say that the edge $\psi(i)u$ of G has been *exposed*. *Querying* an open adjacency $(i,\diamond)$ consists of selecting u.a.r. a neighbour u of $\psi(i)$ that has not yet been associated with an element of ℓ_i.

The effect of querying is explained below in the definition of local rule. Figure 1 shows a query graph H of depth 2 and a possible embedding of H in a graph G. For simplicity, we assume that there is a single colour available, so that types are defined by their degrees. The edges of H are thick solid lines, the edges associated with the open adjacencies having a vertex type are dashed, and those with type $\diamond$ are dotted. Other edges of G are thin lines. In particular, one end of the edge cd is dashed for the open adjacency $(c,4)$ and the other end is dotted for the open adjacency $(d,\diamond)$.

Next we define the crucial randomised rule for choosing the next open adjacency to explore in a query graph. This uses $*$ to denote termination of the exploration step.

Definition (Local subrule, depth). A local subrule is a function ϕ that, for each $H \in \mathcal{Q}$, specifies a probability distribution ϕ_H on the set $U \cup \{*\}$, where $U = \{(i,\tau)\colon i \in V(H) \text{ and } \tau \in \ell_i\}$ is the set of open adjacencies. We require that $\phi_H(f) = 0$ if f is an open adjacency whose type has colour $\propto'$. Moreover, we require that there is a natural number D such that, given any

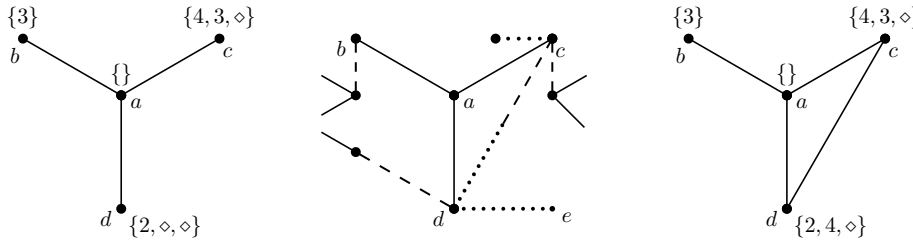

Figure 1. A query graph H (left) and a copy of it in a larger graph (middle). The query graph on the right is discussed after defining the local rule

query graph H, $\phi_H(f)=0$ for every open adjacency $f=(i,\tau)$ for which the distance from i to the root is at least $D-1$. The minimal such D is the *depth* of ϕ. Finally, we require that $\phi_H(*)=0$ if H is a single vertex u such that ℓ_u contains at least one $\diamond$, but no other type.

A local subrule is *normal* if $\phi_H(*)=0$ unless there is no open adjacency of the form $(i,\diamond)$, i.e., an open adjacency of the form $(i,\diamond)$ will always be queried if one exists in the query graph. Non-normal subrules, though apparently rare, can be advantageous, as in the example on k-independent sets in Section 7.

To conclude the description of the exploration step, we combine a local subrule and querying to build a copy of a query graph rooted at v.

Definition (Local rule). The local rule $L=L_\phi$ associated with a local subrule ϕ is a function that maps an ordered pair (v,G), where G is a coloured graph and $v\in V(G)$, to a copy ψ of a query graph H in G. The image of (v,G) under L_ϕ is defined inductively as follows. Let H_0 have a single vertex labelled 1 associated with the multiset ℓ_1 containing one copy of $\diamond$ associated with each neighbour of v in G. Let ψ_0 be the bijection that maps 1 to v. On subsequent steps $k>0$, the local rule applies the local subrule for H_{k-1} and then:

(i) if the outcome is $*$, the output ψ is defined to be ψ_{k-1};
(ii) if the outcome is $(i,\diamond)$, the local rule queries it, obtaining a vertex u adjacent to $w=\psi(i)$. Let τ be the type of u. A new query graph H_k is defined by replacing an occurrence of $\diamond$ in ℓ_i by an occurrence of τ, which is now associated with u, while the copy ψ_k of H_k in G is equal to ψ_{k-1};
(iii) if the outcome is an open adjacency (i,τ) with $\tau\neq\diamond$, then the local rule queries it, obtaining a neighbour u of w having type τ. By induction, $wu\notin E\big(\psi_{k-1}(H_{k-1})\big)$. If $u\notin V\big(\psi(H_{k-1})\big)$, define the query graph H_k from H_{k-1} by adding a new vertex labelled $j=|V(H_{k-1})|+1$. A new multiset ℓ_j is created, with as many copies of $\diamond$ as there are neighbours of u in

G other than w. Moreover, the multiset ℓ_i is updated by removing the occurrence of τ associated with u. The copy ψ_k of H_k in G is the extension of ψ_{k-1} obtained by mapping $|V(H_{k-1})|+1$ to u. On the other hand, if $u \in V\big(\psi(H_{k-1})\big)$, say $u = \psi_{k-1}(j)$, the query graph H_k is obtained from H_{k-1} by adding the edge ij and by removing the items corresponding to u and w from ℓ_i and ℓ_j respectively. The copy ψ_k of H_k in G is equal to ψ_{k-1}.

The copy ψ of the query graph H obtained when this process stops is called the *query graph obtained by the local rule*. The last condition in the definition of local subrule ensures that the local rule always explores some adjacency of the root vertex v unless it is an isolated vertex.

Furthermore, the local rule has *depth* D if the local subrule associated with the local rule has depth D, in which case the query graph clearly has depth at most D. Note also that querying (i,τ), where τ is a vertex type, always adds an edge to the query graph, and possibly a vertex, whilst querying $(i,\diamond)$ only changes ℓ_i. For example, in the situation depicted in Figure 1, querying $(c,4)$ in the embedding in the middle would lead to the query graph on the right. However, querying $(d,\diamond)$ would with probability $\frac{1}{2}$ select c, in which case the query graph structure would remain the same and ℓ_d would become $\{2,4,\diamond\}$. Otherwise it would select e, and ℓ_d would become $\{2,i,\diamond\}$, where i is the degree of e in G.

After applying the local rule, recolouring occurs. For many algorithms there is no need to colour edges, but it is convenient for some, so we include this as an option. Transient colours are not assigned to edges.

Definition (Recolouring rule). A recolouring rule is a (possibly randomised) function c that, given a query graph $H \in \mathcal{Q}$, assigns an output colour other than $\propto$ to each totally explored vertex in H and a transient colour other than $\propto'$ to each partially explored vertex in H. Moreover, it assigns a colour to any element $\tau \neq \diamond$, in a multiset ℓ_i for $i \in V(H)$, whose colour is not $\propto'$. Optionally, c also assigns an output colour to each edge of H.

Intuitively, the multiset element is recoloured in order to flag the desire to recolour the vertex of G associated with it. (The vertex itself cannot be recoloured yet, as explained below.)

For a sample application of a recolouring rule, consider the Algorithm P_{cut} described in Section 2. Assume that the local rule explores a vertex v of type 10 whose neighbours have types 00 and 01. Then the final query graph would consist of a single vertex called 1, with $\ell_1 = \{01,00\}$. The recolouring rule will recolour v with output colour blue since it has type $(1,0)$, while the

types 00 and 01 in the list ℓ_1 are replaced by 01 and 02, respectively (since they represent vertices that gained a blue neighbour).

Definition (Local deletion algorithm). This consists of a triple $(\mathbf{\Pi}, L, c)$ with $\mathbf{\Pi} = \{\Pi_t\}_{t=1}^{N}$ for some N, where each Π_t is a selection rule, L is a local rule and c is a recolouring rule. Applied to a graph $G = (V, E)$, the algorithm runs for N steps. Let $\mathcal{C}$ be the set of transient colours and $\mathcal{E}$ the set of output colours, disjoint from $\mathcal{C}$. The algorithm starts with $G_0 = G$, all of whose vertices are initially given neutral transient colour, and repeats the following step t, $t = 1, \dots, N$.

(i) (Selection step.) Obtain a set $S_t \subset V(G_{t-1})$ using the selection rule Π_t;

(ii) (Exploration step.) For each $v \in S_t$, obtain a copy ψ_v of a query graph $H \in \mathcal{Q}$ by applying the local rule $L = L_\phi$ to G_{t-1} and v;

(iii) (Clash step.) A vertex u is called a *clash* if at least one of the following occurs: (a) u lies in ψ_v for at least two vertices $v \in S_t$; (b) u lies in a single ψ_v but is adjacent to a vertex in some ψ_w, where $u \neq w$; (c) u lies in a single ψ_v but is adjacent to a vertex in ψ_v through an edge in $G_{t-1} - E(\psi_v(H_v))$. Let B be the set of all clashes. (Note that (a) and (b) are unlikely to occur often because of the bounded depth of the rules, whilst (c) will never occur when the girth is large enough.)

(iv) (Recolouring step.) The *survival graph* G_t is obtained from G_{t-1} as follows. At first, the vertices associated with open adjacencies whose type is not $\diamond$ are placed in a multiset W. All vertices and edges which are specified colours by c are recoloured accordingly, except that clash vertices and all incident edges are assigned $\propto$. Each vertex u in $G - \bigcup_{v \in S_t} \psi_v(H_v)$ placed in W is necessarily associated with at least one element of a list ℓ_i in a query graph. If in exactly one, recolour u with the colour assigned by c to that list element; otherwise, recolour u with $\propto'$. All vertices and edges receiving output colours, and all exposed edges, are deleted.

In each step, any random choices must follow the prescribed distributions conditional upon the graph G_t. The output of the algorithm is the vector ω whose jth component is the set of vertices receiving the jth output colour.

The *depth* of the algorithm is the depth of its local rule L. Actually, we could have used just one clash colour instead of both $\propto$ and $\propto'$, but the second (transient) clash colour used for clashes in the recolouring step is a useful signal to avoid querying their open adjacencies.

The algorithms described in Section 2 did not use colours, but such algorithms can be recast as local deletion algorithms as follows.

Definition (Native local deletion algorithm). This has only two transient colours, being neutral and $\propto'$, and three output colours: as usual, $\propto$

denotes clash vertices; the second colour is used on the output set $\mathcal{O}$; the third is used on the rest of the deleted vertices.

This agrees with the informal definition in Section 2 referring to an algorithm with no colours, since in the native case, the local and recolouring rules simply determine which vertices and edges are deleted and which are added to the output set.

3.2. Chunky local deletion algorithms

To facilitate analysis we can restrict the kind of algorithms under scrutiny, without sacrificing the power of the results to the accuracy we are interested in. A local deletion algorithm is said to be *chunky* if the probability distribution associated with the selection rule Π_t at step t is defined as follows for every $t \in \{1, \dots, N\}$:

(1) there are fixed real numbers $p_{t,j} \in [0,1]$ for every type j, called the governing probabilities of the chunky algorithm;
(2) the selection rule Π_t is such that the probability of a nonempty set S is given by $\pi_G(S) = \prod_{v \in S} p_{t,\tau_G(v)}$.

In other words, in a chunky local deletion algorithm, each vertex v of G_{t-1} with type j is added to the set S_t randomly, independently of all others, with probability $p_{t,j}$. Generally, we keep N fixed independently of the size of the input graph. If $\mathcal{T} = \{1, \dots, R\}$ denotes the set of types, we call the matrix $(p_{t,j})_{1 \le t \le N, 1 \le j \le R}$ a *matrix of probabilities* of the algorithm, and we define the *granularity* of the algorithm to be the maximum entry in this matrix. Fixed R suffices for any algorithm with a fixed number of colours acting on graphs with maximum degree r, as there is a bounded number of types in this case.

We illustrate this using the procedure for finding an independent set in a graph G discussed in Section 2. This can be turned into a (native) chunky local deletion algorithm if, instead of choosing a single vertex of minimum degree in the survival graph in each step, the selection rule produces a subset S_t by including each such vertex with some probability p. Assuming there are no clashes, the survival graph G_t is updated by incorporating the changes prescribed by each query graph separately. However, if there are clashes, such as when two vertices in S_t are adjacent, some recolouring is involved using $\propto$. In this way, the vertices in $\mathcal{O}$ will retain the property of actually being an independent set. This is a workable approach if we can ensure that clashes are rare by choosing small p.

4. Analysis of chunky local deletion algorithms

Recall that $r \geq 2$ is fixed. The aim of this section is to show that chunky local deletion algorithms give essentially the same performance, for the random variables of interest, when applied to any r-regular graph G of sufficiently large girth. This will be proved in two main steps. First we establish that the expected values of these random variables does not depend on the inputted n-vertex r-regular graph with girth at least g, provided that g is sufficiently large in terms of the number of steps of the algorithm. We then show that the variables are sharply concentrated around these same expected values as long as the number of short cycles is 'small,' which holds trivially for graphs with large girth and is a well known property of random regular graphs. This establishes the desired connection between the two types of input graphs.

It assists the analysis of the chunky algorithm if we make all random choices in advance, as follows. With each $t \in \{0, 1, 2, \ldots, N-1\}$ and each type j, we associate a random set $S_j(t) \subseteq V$, where a vertex v is placed in $S_j(t)$ with probability $p_{t,j}$. All these choices for various j, t and v are made independently of each other. In addition, for each copy ϕ_H of any query graph H in G, we select an open adjacency (or $*$) according to the probability distribution given by the local subrule ϕ, and also, if $*$ is specified and the recolouring rule c is randomised, the values of the colours assigned by c. This gives a function Υ defined on all copies of query graphs in G.

Given Υ and the sets $S_j(t)$, define $G_0 = G$ with all vertices of neutral transient colour, and determine G_t from G_{t-1} as follows. For a vertex v whose type in G_{t-1} is j, place v in the set S_t if and only if $v \in S_j(t)$. Then use the function Υ iteratively via the local subrule to obtain the query graphs (just as for the local rule in the chunky algorithm). Finally, apply steps (iii) and (iv) in the local deletion algorithm, using Υ to determine any random choices in the recolouring rule. This determines a 'survival' graph G_t and the colours of the vertices deleted. Note that the value of Υ at any particular point in its domain is used at most once. This is because for each query graph produced by the local rule, there are two possibilities. Either the selected vertex has degree at least 1 and, according to the local subrule, at least one open adjacency is queried, with the corresponding edges subsequently deleted, or it has degree 0 and is given an output colour and deleted according to the recolouring rule.

We now give an upper bound on the "speed" at which the effect of random choices made in the algorithm may propagate through the graph. First, before starting the algorithm, affix a label $\mathcal{L}^v = (\mathcal{L}_1^v, \mathcal{L}_2^v)$ to each vertex v where $\mathcal{L}_1^v = \{t \colon v \in S_j(t)\}$, and $\mathcal{L}_2^v$ consists of the action of Υ on all the copies of query graphs whose root is v.

Lemma 4.1. *Consider a chunky local deletion algorithm whose local rule has depth D. Let G be a graph, let G_t be the survival graph defined above, and let F be a subgraph of G. Then $F \cap G_t$, and the types of its vertices, are determined by the subgraph induced by the vertices whose distance in G from F is at most $2Dt$, together with the labels of those vertices, up to label-preserving isomorphisms.*

Proof. The nature of the change in type of a vertex w at step t, and whether w is deleted in step t, is dependent only on the copies of query graphs existing in G_{t-1} which w is in or adjacent to, together with the labels on the root vertices of such copies. As the depth is at most $D-1$, deleting vertices of the image of a query graph with root is mapped to v cannot affect the degree of any vertex of distance greater than D from v. So the set of such copies is determined by the graph $F' \cap G_{t-1}$, where $F' = G[\{v\colon d_G(v,F) \le 2D\}]$. The lemma now follows by induction on t and the triangle inequality. ∎

A *rooted graph* is a graph with one vertex distinguished, and called the *root*. If the graph is a tree T, we call it a *rooted tree*, and its height (i.e., maximum distance of a vertex from the root) is denoted by $h(T)$. Let $T_{r,h}$ denote the rooted tree in which every non-leaf vertex has degree r and every leaf is at distance h from the root.

For a vertex v in G, the *s-neighbourhood* of v is the subgraph of G induced by the set of vertices of distance at most s to v, viewed as a rooted graph with root v. A vertex not surviving in G_t has empty s-neighbourhood.

Corollary 4.2. *Fix positive integers t and h, and apply a chunky local deletion algorithm $\mathcal{A}$ of depth D to an r-regular graph G of girth greater than $4Dt+2h$.*

(i) Let F be a vertex-coloured version of a subgraph of $T_{r,h}$. Consider an embedding $\hat{T}$ of $T_{r,h}$ in G. Then $\mathbf{P}(G_t \cap \hat{T} = F)$ (where equality requires the colours of G_t to match the colours of F) is a constant depending on $\mathcal{A}$ and r but independent of the choice of the graph G and the embedding $\hat{T}$.

(ii) Let T be a coloured rooted tree of height at most h and fix a vertex $v \in V(G)$. Then the probability that the h-neighbourhood of v in G_t is isomorphic to T (where isomorphism requires the colours to be preserved) after t steps is a constant $p_{t,T,h} = p_{t,T,h}(\mathcal{A},r)$ independent of the choice of the graph G and the vertex v.

Proof. Define $G_{\hat{T}}$ to be the subgraph of G induced by all vertices within distance $2Dt$ of $\hat{T}$. By Lemma 4.1, for an embedding $\hat{T}$ of T in G, the

intersection of G_t and $\hat{T}$ is determined by the label-preserving isomorphism type of the subgraph induced by the vertices of $G_{\hat{T}}$. By the girth hypothesis, the graph $G_{\hat{T}}$ is a tree whose isomorphism type (ignoring the vertex labels) is independent of the choice of G and $\hat{T}$. The labels are assigned independently with the same probability distribution to each vertex. The result follows for (i). Then (ii) follows by applying (i) to all graphs F whose component containing the root of $T_{r,h}$ is isomorphic to T (with matching colours), and such that v is the root of $\hat{T}$, and summing the probabilities. ∎

Instead of proving facts about the number of vertices with each output colour, it will be convenient to generalise the concept as follows.

Definition (Active copy, output function). A copy ψ_v of H is active at step t of a local deletion algorithm if it is generated by the exploration part of this step. An output function W is a function determined by fixed numbers $c_{H,L,J}$ via the recurrence

$$W(t+1) = W(t) + \sum_{H,L,J} c_{H,L,J} W_{H,L,J}(t) \quad (t \geq 0), \quad W(0) = 0, \tag{1}$$

where $W_{H,L,J}(t)$ is the number of active copies ψ_v of $H \in \mathcal{Q}$ at step $t+1$ for which $\mathcal{L}_2^v(\psi) = L$ and $\psi(J)$ is the set of clash vertices of ψ_v.

It is simple to define $c_{H,L,J}$ so that the function $W(t)$ is precisely the number of vertices of a given output colour at step t. This will ease our analysis of the size of the components of the output vector.

Theorem 4.3. *Let G be an r-regular graph on n vertices with girth larger than $4DN + \max\{2, 6D\}$. For $0 \leq t \leq N$, let $Y_k(t)$ denote the number of vertices of type k in the survival graph G_t of a chunky local deletion algorithm $\mathcal{A}$ of depth D. Also let $W(t)$ be an output function. Then, for each such t, the vector $s_{r,\mathcal{A}}(t) = s_{r,\mathcal{A}}(t,n) = \big(\mathbf{E}Y_1(t), \ldots, \mathbf{E}Y_R(t), \mathbf{E}W(t)\big)$ is independent of the choice of G.*

Proof. Applying Corollary 4.2(ii) with $h = 1$, we may sum $p_{t,T,1}$ over all T with root of type k, to obtain the probability, independent of G, that any given vertex of G is of type k in G_t. Multiplying by n gives $\mathbf{E}Y_k(t)$, which proves the first part, as the girth is at least $4DN + 2$. The proof for $W(t)$ is similar, but requires girth at least $4DN + 6D$. The expected value of $W(t+1) - W(t)$ is fixed provided that, for each copy ψ of any query graph H, the probability that ψ is active and $L = \mathcal{L}_2^v$, where v is the root of ψ, and additionally J is the set of clash vertices in this copy in step $t+1$, is a fixed number given H, L and J. This is true for $t \leq N - 1$ by Corollary 4.2 with

$h = 3D$, since the intersection of the $(3D)$-neighbourhood of v with G_{t-1} determines all these things. ∎

We next show that the key random variables are concentrated near the values suggested by the constants given in the previous two results, provided the input graph has few short cycles.

Lemma 4.4. *Given a positive integer N, let G be an r-regular graph with n vertices, at most $\Theta < n^{2/3}$ of which are in cycles of length less than or equal to $4DN+6D+2$. Consider N steps of a chunky local deletion algorithm $\mathcal{A}$ of depth D applied to G. Then the following hold.*

(i) Given $t \in \{0,\dots,N\}$ and a coloured rooted tree T of height at most $h \le 2D(N-t)+3D+1$, let $A_{t,T,h}$ denote the number of vertices in G_t whose coloured h-neighbourhood in G_t is isomorphic to T. Then, for some positive constant C determined by the local and recolouring rules, the degree r and the number of steps N,

$$\mathbf{P}\left(|A_{t,T,h} - np_{t,T,h}| \ge Cn^{2/3}\sqrt{\Theta+1}\right) \le n^{-1/3},$$

where $p_{t,T,h}$ is defined in Corollary 4.2(ii).

(ii) Define $Y_k(t)$, $W(t)$ and $s_{r,\mathcal{A}}(t,n)$ as in Theorem 4.3. Set $\mathbf{s}(t) = \big(Y_1(t),\dots,Y_R(t),W(t)\big)$ and, given $C>0$, consider the event $F(C)$ that $||\mathbf{s}(t) - s_{r,\mathcal{A}}(t,n)||_\infty \ge Cn^{2/3}\sqrt{\Theta+1}$ for some $t \in \{0,1,\dots,N\}$. Then, for some constants C and C' determined by r, N and the local and recolouring rules,

$$\mathbf{P}\big(F(C)\big) \le C'n^{-1/3}.$$

Proof. We first prove part (i). A vertex in G is said to be *good* if its distance from every vertex lying in a cycle of length at most $4DN+6D+2$ is larger than $2D(N+1)$, and *bad* otherwise. Note that the set B of bad vertices of G has size at most $C\Theta$ where C is constant given r, D and N.

Let t and T be as in the hypothesis. By Corollary 4.2 and linearity of expectation,

$$\mathbf{E}A_{t,T,h} = (n-|B|)p_{t,T,h} + |B|O(1) = np_{t,T,h} + O(\Theta)$$

for fixed D and N. On the other hand,

$$\mathrm{Var}(A_{t,T,h}) = \sum_{v\in V}\mathrm{Var}(X_{v,T,h}) + \sum_{v\neq w\in V}\mathrm{Cov}(X_{v,T,h}, X_{w,T,h}),$$

where $X_{v,T,h}$ is the indicator random variable for the event that v has coloured h-neighbourhood T after t steps of the algorithm.

Being an indicator, $X_{v,T,h}$ has variance at most 1. If the distance between v and w is greater than $4D(N+2)$, then the variables $X_{v,T,h}$ and $X_{w,T,h}$ are independent, and hence their covariance is 0. For all other pairs, the covariance has absolute value at most 1. Thus

$$\mathrm{Var}(A_{t,T,h}) = O(n) + O(n|B|) = O(n(\Theta+1)).$$

By Chebyshev's inequality, $\mathbf{P}\big(|A_{t,T,h} - \mathbf{E}A_{t,T,h}| \geq \sqrt{n^{1/3}\,\mathrm{Var}(A_{t,T,h})}\big) \leq n^{-1/3}$, giving (i).

For part (ii), we argue in a fashion similar to the proof of Theorem 4.3. $Y_k(t)$ is just a sum of $A_{t,T,1}$ over appropriate trees T, so the required concentration of these components of $\mathbf{s}(t)$ follows from (a). For the final component, $W(t)$, one can define an indicator variable to denote that a copy ψ of a query graph H is active, with given value L of $\mathcal{L}_2^v$ where v is the root of ψ, and given set J of clash vertices in this copy in step $t+1$. Using the fact from (i) that the numbers of vertices with given neighbourhoods are concentrated, and again arguing as for (i) using Chebyshev's inequality, the sum of these indicators, which equals $W_{H,L,J}(t)$ as in (1), is concentrated, to the extent claimed, except for an event with probability $n^{-1/3}$. Summing over all H, L and J and applying the union bound shows that $W(t+1)-W(t)$ is similarly concentrated, up to a constant factor in the error, with probability $O(n^{-1/3})$. Doing the same for t gives the result required to complete (ii). ∎

We now apply Lemma 4.4(ii) to random regular graphs.

Theorem 4.5. *Let $\mathcal{A}$ be a chunky local deletion algorithm performing N steps applied to a random r-regular graph $G \in \mathcal{G}_{n,r}$. The vector $\mathbf{s}(t) = \big(Y_1(t),\dots,Y_R(t),W(t)\big)$ produced after t steps of the algorithm is a.a.s. within $C_1' n^{2/3}$ of $s_{r,\mathcal{A}}(t,n)$ in L^∞-norm, where C_1' is a constant for given N, R and D, uniformly over all $t \in \{0,\dots,N\}$.*

Proof. In $\mathcal{G}_{n,r}$, the expected number of cycles of length at most $4D(N+1)$ is bounded (see [3] or [29]). Now apply Lemma 4.4(ii). ∎

Theorem 4.5 implies that the output vector produced by a chunky local deletion algorithm $\mathcal{A}$ applied to a random n-vertex r-regular graph and the vector of expected output of $\mathcal{A}$ applied to an n-vertex r-regular graph G with sufficiently large girth a.a.s. differ by $o(n)$. (To interpret a statement mixing a.a.s. notation with other asymptotic notation, see [31].) So we may estimate the value of $s_{r,\mathcal{A}}(n)$ by analysing $\mathcal{A}$ applied to $\mathcal{G}_{n,r}$. Crucially, this allows us to make use of the powerful machinery already developed for analysing $\mathcal{G}_{n,r}$ (see the next section).

We still need to limit the number of clashes in order to obtain meaningful results from the algorithm. If $\mathcal{A}$ has depth D, we define a *pre-clash* to be a pair of vertices of distance at most $2D$ apart which are both included in the set S_t at the same step t. The number of clashes will be bounded above by a constant times the number of pre-clashes. Recall that the granularity of a chunky algorithm is the maximum entry in its matrix of probabilities.

Lemma 4.6. *The expected number of pre-clashes in step t of a chunky algorithm of depth D and granularity ϵ applied to an n-vertex graph with maximum degree r is $O(\epsilon^2 n)$. The implicit constant depends only on D and r.*

Proof. The number of pairs of vertices of distance at most $2D$ is $O(n)$, and the probability that both vertices are chosen in S_t is at most ϵ^2, so this follows immediately from the union bound. ∎

5. Explicit bounds from chunky algorithms

Theorem 4.3 produces bounds on the value of an output function when a local deletion algorithm is applied to a graph G with sufficiently large girth, in terms of $s_{r,\mathcal{A}}(t)$. Owing to Theorem 4.5, it suffices to consider $G \in \mathcal{G}_{n,r}$. In this section we develop this into part of the machinery which lets us translate some existing results on $\mathcal{G}_{n,r}$ to regular graphs of large girth.

To analyse $\mathcal{G}_{n,r}$, we use the configuration (or pairing) model of Bollobás [3]. Consider rn points in n buckets labelled $1,\dots,n$, with r in each bucket, and choose uniformly at random (u.a.r.) a *pairing* $P = a_1,\dots,a_{rn/2}$ of the points such that each a_i is an unordered pair of points, and each point is in precisely one pair a_i. We use $\mathcal{P}_{n,r}$ to denote this probability space of random pairings. Each pairing corresponds to an r-regular pseudograph (loops and multiple edges permitted) with vertex set $1,\dots,n$ and with an edge for each pair. A pair with points in buckets i and j gives rise to an edge joining vertices i and j. A straightforward calculation shows that the simple r-regular graphs (i.e., with no loops or multiple edges) on n vertices are produced u.a.r. The probability that a random pairing produces an r-regular graph tends to the positive constant $e^{(1-r^2)/4}$ as n tends to infinity (Bender and Canfield [2]). There is an obvious generalisation of $\mathcal{P}_{n,r}$ to $\mathcal{P}(\mathbf{r})$, where bucket i contains r_i points. Provided that the elements of $\mathbf{r}$ are bounded, the probability that the resulting graph is simple is bounded below. Conditioning on this event gives the model $\mathcal{G}(\mathbf{r})$ of uniformly random graphs with degree sequence $\mathbf{r}$. (See [29] for more details.)

To analyse algorithms, we choose the pairs sequentially: the first point in a pair can be selected using any rule that depends only on the choices made so far (possibly with external randomisation), as long as the second is chosen u.a.r. from the remaining points. We call this *exposing* the pair, and this property is the *independence property* of the model. In particular, for algorithms applied to the random graph, choosing the pairs can be done in the order in which the algorithm queries the edges of the graph. We call this a *pairing process*. For simplicity, we will often refer to a pairing as a (pseudo)graph and to a bucket as a vertex.

Local deletion algorithms can be redefined in an obvious way so as to apply to pairings, provided the set $\mathcal{Q}$ of query graphs includes all coloured pseudographs. We will assume this property of $\mathcal{Q}$ henceforth. Recall that, at every step t, once a vertex v is selected, the algorithm queries vertices in the survival graph G_{t-1} to build a query graph rooted at v. This query graph then determines (perhaps with randomisation) the deletions and recolourings which create G_t. For the pairings, querying an open adjacency $(i,\diamond)$ is equivalent to specifying, for an unpaired point j in bucket i, the type $\tau(j)$ of the bucket containing the mate of j, while querying an open adjacency (i,τ) corresponds to selecting any point j in bucket i with $\tau(j)=\tau$ and choosing an unpaired point k in a bucket of type τ to complete a pair with j. Here k is chosen randomly among all unpaired points in buckets that had type $\tau(j)$ at the moment that bucket i was queried. The random choice of v is similarly incorporated. The 'survival pairing' contains the unexposed pairs after t steps of the algorithm, and corresponds to the survival pseudograph G_t. The definition of clashes is obvious. In particular, loops create clashes.

It is easy to check that the above description for pairings corresponds to the local rule applied to the corresponding graph. Consequently, when applying local deletion algorithms to a pairing, we often speak of applying it to the associated pseudograph.

From the relation between pairings and graphs, we have the following.

Lemma 5.1. *Suppose that when a local deletion algorithm is applied to a random pairing with $n_k = n_k(n)$ vertices of type k for each k, the survival pseudograph G_t has property P with probability $p(n)$. If the same algorithm (i.e., selection and local rules and recolouring function) is applied to a random graph with $n_k = n_k(n)$ vertices of type k for each k, then the survival graph after t steps has property P with probability at most $cp(n)$, where c is a constant depending only on the maximum degree of the input graph.*

For a coloured pseudograph G on n vertices, let n_k denote the number of vertices of type k in G, define the vector $\tilde{\mathbf{n}} = (n_1/n, \ldots, n_R/n)$, and let the

degree of a vertex of type k be denoted by $d(k)$. The next result will let us describe how this vector is expected to change due to an application of the local rule to a single vertex in the survival graph.

Lemma 5.2. *Let r and D be positive integers. Given a local rule L_ϕ of depth D and a recolouring rule c, let R denote the number of types when a local deletion algorithm with these rules is applied to graphs with degrees bounded above by r. Let $i,k \in \{1,\dots,R\}$ be types and let H be a query graph. Then there exist functions $f_{k,i}$ and $g_{H,i}\colon \mathbb{R}^R_{\geq 0} \to \mathbb{R}$ which, for all $\epsilon > 0$, are Lipschitz continuous in $D(\epsilon) := \{(\tilde{n}_1,\dots,\tilde{n}_R) \in \mathbb{R}^R_{\geq 0}\colon \sum_{k=1}^R d(k)\tilde{n}_k \geq \epsilon\}$, such that the following is true. Let G be a random pseudograph with degree sequence $(r_1,\dots,r_n)$ generated by the pairing model, where $0 \leq r_k \leq r$ for all k, and fix a colouring of G. Let n_k denote the number of vertices of type k. Consider $i \in \{1,\dots,R\}$ such that $n_i > 0$ and fix a vertex v of type i in G. Let H_v be the (random) query graph obtained by one application of L_ϕ to v in G, and let G' be the survival graph obtained from G after the clash and recolouring steps. Assume that $M = \sum_{j=1}^n r_j = \sum_{k=1}^R d(k)n_k > \epsilon n$ for some $\epsilon > 0$. The following hold with the constants implicit in $O()$ terms independent of the r_k.*

(i) For any query graph H,

$$\mathbf{P}(H_v = H) = g_{H,i}(\tilde{\mathbf{n}}) + O(1/n).$$

(ii) The number Y_k of vertices of type k in G' satisfies

$$\mathbf{E}(Y_k) = n_k + f_{k,i}(\tilde{\mathbf{n}}) + O(1/n).$$

Moreover, the expected number of vertices that are assigned colour $\propto'$ in the recolouring step is $O(1/n)$.

Proof. Recall that the survival graph G' is derived from G by deleting all edges in the copy of the query graph H_v obtained via the local rule L_ϕ applied to v in G, and by recolouring or deleting vertices in this copy, and recolouring vertices associated with the open adjacencies of H.

To begin, we bound the (conditional) probability that either of the following two events occurs: (a) one of the open adjacencies in H_v is incident with a vertex within H_v (a clash); (b) two or more open adjacencies in H_v are incident with the same vertex (that vertex is then assigned colour $\propto'$). As noted before, when a pair is exposed, the mate of the starting point is distributed u.a.r. over the remaining points. Hence the probability of (a) is bounded above by s^2/S, where s is the number of points in the vertices of H_v and S is the number of unpaired points in $G - V(H_v)$. Clearly,

$s \le r|V(H_v)|$, $S \ge M - r|V(H_v)|$ and $|V(H_v)| \le m_{r,D-1} \le 1 + r^D$, where $m_{r,D-1}$ denotes the number of vertices in a balanced r-regular tree of height $D-1$. So $s^2/S \le r^2 m_{r,D-1}^2/(M - r m_{r,D-1}) = O(1/M) = O(1/n)$. Similarly, the probability of (b) is $O(1/n)$, as the number of points outside H_v is $\Omega(M)$ and so the probability that two open adjacencies lead to the same bucket is $O(1/M)$. Since the recolouring step assigns colour $\propto'$ only if (b) occurs, the expected number of vertices that are assigned colour $\propto'$ in the recolouring step is $O(1/n)$, which gives the final statement of the lemma.

Next, we show that part (ii) is a direct consequence of part (i). We argue at first that, conditional upon the query graph H_v, the quantity $\mathbf{E}(Y_k) - n_k$ is determined up to a $O(1/n)$ term. Indeed, consider the vertices that have type k in G. The type of one of these vertices may be different from k in G' if it is in the copy of the query graph H_v, or if it is the end of an open adjacency (whose type is necessarily k) in this copy of H_v. The expected number of these vertices that change type is determined by the action of the (possibly randomised) recolouring function c on H_v, independently of the degree sequence. On the other hand, consider the vertices that have type k (and whose colour is not $\propto'$) in G' but do not have this type in G. These vertices come from two sources: those that have degree $d(k)+1$ in G and are the end of an open adjacency of H_v, which in addition is assigned the colour of k by the recolouring function c, and those with the correct degree that lie in H_v and are assigned the colour of k by c. Again, the expected number is a function of the recolouring rule. In this discussion we have somewhat ignored events (a) and (b), but their probabilities are determined by the degree sequence of the graph, and the expected changes in the quantities due to their occurrence are similarly determined up to a $O(1/n)$ error.

Thus $\mathbf{E}(Y_k) - n_k = \Delta_k(H_v) + O(1/n)$ for a function Δ_k depending only on L_ϕ and c. Hence the expected value of Y_k is given by

$$\mathbf{E}(Y_k) = n_k + \sum_{H \in \mathcal{Q}_D} \mathbf{P}(H_v = H) \cdot \Delta_k(H) + O(1/n),$$

as the number of query graphs in $\mathcal{Q}_D$ with maximum degree bounded by r is a constant depending on r, D and the number of colours available. The claim of part (ii) thus follows from part (i).

Let $H_{v,j}$ denote the (random) query graph obtained after j steps of the local subrule when applying the local rule starting with a vertex v of type i in G. To prove (i), we show by induction on j that, for each H, $\mathbf{P}(H_{v,j} = H) = g_{H,i}^{(j)}(\tilde{\mathbf{n}}) + O(1/n)$, where $g_{H,i}^{(j)}$ is Lipschitz continuous on $D(\epsilon)$. After the local rule is initiated, but before any application of the local subrule, the only possible query graph H is a singleton $\{1\}$ with a multiset ℓ_1 containing

i copies of $\diamond$. This starts the induction with $j=0$. For the inductive step, we may assume that H is a tree; otherwise, at some point in the process, the query operation would have exposed an open adjacency incident with a vertex within the query graph, which has probability $O(1/n)$ as shown above. Hence, the query graph H has been derived in one of two possible ways. The first is from some H' by adding the vertex that has largest label in H, which we may denote by u. At the same time, the multiset associated with the unique neighbour w of u, is adjusted appropriately. There is a unique such H'. The second case is that H comes from some $H^\star$ by replacing an occurrence of $\diamond$ by a vertex type.

In the first case, by induction $\mathbf{P}(H_{v,j-1}=H')=g^{(j-1)}_{H',i}(\tilde{\mathbf{n}})+O(1/n)$. The probability that H was obtained from H' is the probability that the open adjacency $(w,\tau(u))$ is queried, where $\tau(u)$ is the type of the image of u in G, which is equal to $\phi_{H'}(w,\tau(u))$. In the second case, for each element in the multiset $S(H)=\{(w,\tau)\colon w\in H \text{ and } \diamond\neq\tau\in\ell_w\}$, let $H^\star_{w,\tau}$ be the same as H but with one occurrence of τ replaced by $\diamond$ in ℓ_w. Then $\mathbf{P}(H_{v,j-1}=H^\star_{w,\tau})=g^{(j-1)}_{H^\star_{w,\tau},i}(\tilde{\mathbf{n}})+O(1/n)$. Moreover, the probability that the open adjacency $(w,\diamond)$ is queried is $\phi_{H^\star_{w,\tau}}(w,\diamond)$, while the probability that the outcome of this query is τ is $d(\tau)n_\tau\big(\sum_{s=1}^R d(s)n_s\big)^{-1}+O(1/n)$ in the pairing model. This implies that the function $g^{(j)}_{H,i}$ defined by

$$g^{(j)}_{H,i}(\tilde{\mathbf{n}})=g^{(j-1)}_{H',i}(\tilde{\mathbf{n}})\phi_{H'}(w,\tau(u))+\sum_{(w,\tau)\in S(H)} g^{(j-1)}_{H^\star_{w,\tau},i}(\tilde{\mathbf{n}})\phi_{H^\star_{w,\tau}}(w,\diamond)\frac{d(\tau)n_\tau}{\sum_{s=1}^R d(s)n_s}$$

satisfies the required properties for the induction to go through. Regarding the Lipschitz property, note that the terms $\phi_F(\cdot,\cdot)$ are constants, the functions $g^{(j-1)}_{F,i}$ are Lipschitz continuous by induction and the function

$$\frac{d(\tau)n_\tau}{\sum_{s=1}^R d(s)n_s}=\frac{d(\tau)\tilde{n}_\tau}{\sum_{s=1}^R d(s)\tilde{n}_s}$$

is Lipschitz because of the assumption that $\sum_{k=1}^R d(k)n_k>\epsilon n$ for some $\epsilon>0$.

To conclude the proof of (i), note that

$$\mathbf{P}(H_v=H)=\sum_{j\geq 0}\mathbf{P}(H_{v,j}=H)\phi_H(*). \qquad \blacksquare$$

For later reference, note that the proof of Lemma 5.2 implies that the way a local rule acts on query graphs with cycles is irrelevant asymptotically.

Hence, every extension of a local deletion algorithm for graphs to one for pseudographs gives the same functions $f_{k,i}$.

The next lemma rests heavily on our restriction to deletion algorithms.

Lemma 5.3. *When a local deletion algorithm is applied to a random pairing as in Lemma 5.1, in each step the new survival pairing, conditional on history of the algorithm and the surviving vertices' degrees and transient colours, is distributed u.a.r.*

Proof. The algorithm can be formulated as a pairing process in which the pairs in the survival pairing consist precisely of the pairs that are not yet exposed in the process. So this lemma follows immediately from the independence property of the pairing process stated above. ∎

Lemma 5.2 refers to one step in a local deletion algorithm. We next extend this to analysing a complete algorithm. Assume that the selection step preceding the exploration and recolouring steps is that of a chunky algorithm where vertices of type i are chosen with probability p_i, and we are to perform step t in a local deletion algorithm with given local and recolouring rules. The expected number of vertices of degree i selected for the set S_t is $n\tilde{p}_i$ where $\tilde{p}_i = p_i n_i/n$, with n_i being the number of vertices of type i in the current graph. The expected change in n_i/n suggested by Lemma 5.2 is thus approximately $\sum_{i=0}^{r} \tilde{p}_i f_{j,i}(\tilde{\mathbf{n}})$ (for $0 \le i \le r$).

By choosing W, we mean choosing the constants $c_{H,L,J}$ in (1). Given W and the local, selection and recolouring rules, we define

$$f_{R+1,i}(\tilde{\mathbf{n}}) = \sum_{H \in \mathcal{Q}_i, L} c_{H,L,\emptyset}\, a_{H,L}\, g_{H,i}(\tilde{\mathbf{n}}), \tag{2}$$

where $\mathcal{Q}_i$ denotes the set of query graphs whose root vertex has type i, $a_{H,L}$ is the probability that $\mathcal{L}_2^{(v)} = L$ given that the query graph is H, and $g_{H,i}$ is the function given in Lemma 5.2(i). We restrict to the case $J = \emptyset$ because the influence of clashes is negligible, as observed in Lemma 5.2.

The next result shows that important variables in a chunky local deletion algorithm a.a.s. track the solution of the difference equation suggested by the expected changes estimated above. The error in the approximation depends on the maximum entry ϵ in the probability vectors. As the proof reveals, for fixed N an error of order $\epsilon^2 Nn$ is unavoidable to the first degree of approximation due to the occurrence of clashes. We impose the condition $N \le C/\epsilon$ to achieve eventually an error $O(\epsilon n)$.

Theorem 5.4. *Consider a chunky local deletion algorithm applied to $G \in \mathcal{G}(\mathbf{r})$, all of whose vertices initially have neutral transient colour, for N steps*

with matrix of probabilities $Q=(p_{t,j})$*, and let* W *be an output function. Let* n_k *denote the number of vertices of type* k *in* G*, and let* $\epsilon=\max_{t,j} p_{t,j}$*. For* C *a fixed constant and* $N\le C/\epsilon$*, consider the quantities* $z_j(t)$*, where* $j\in\{1,\dots,R+1\}$ *and* $t\in\{0,\dots,N\}$*, given iteratively by*

$$z_j(t) = z_j(t-1) + \sum_{i=1}^{R} p_{t,i} z_i(t-1) f_{j,i}(\mathbf{z}(t-1)),$$

where $z_j(0)=n_j/n$ *for* $1\le j\le R$*,* $z_{R+1}(0)=0$*, and the functions* $f_{j,i}$ *are those appearing in Lemma 5.2. Then, a.a.s. the number of vertices of type* j *in the survival graph is* $nz_j(t)+O(\epsilon n)$*, for all* $j\in\{1,\dots,R\}$*, and the value of the output function* $W(t)$ *at step* t *is* $nz_{R+1}(t)+O(\epsilon n)$*, uniformly for* $0\le t\le N$*. Here the constant implicit in* $O(\epsilon n)$ *depends on* C *and on the local and recolouring rules but is otherwise independent of* N *and of the* $p_{t,i}$ *(and in particular of* ϵ*). Furthermore, it is independent of* $\mathbf{r}$ *provided* $\max r_i\le r$ *for some fixed* r*.*

Part of the proof is deferred to the next section as it uses a deprioritised algorithm.

Suppose that the matrix of probabilities of the algorithm in Theorem 5.4 is such that, for each i, the quantities $p_{t,i}/\epsilon$ can be interpolated by a fixed piecewise Lipschitz continuous function $p_i(x)$ whose domain is rescaled to $[0,C]$, that is, $p_{t,i}/\epsilon = p_i(\epsilon(t-1))$ for all relevant t. Then the difference equation in Theorem 5.4 defines the approximate solution by Euler's method, with step size ϵ, of the d.e. system

$$y_j'(x) = \sum_{i=1}^{R} p_i(x) y_i(x) f_{j,i}(y_1,\dots,y_R) \quad (1\le j\le R+1) \tag{3}$$

on the interval $[0,C]$ with initial conditions

$$y_j(0) = n_j/n \quad (1\le j\le R), \quad y_{R+1}(0)=0, \tag{4}$$

where n_j is the number of vertices of type j in the input graph. Here and throughout our paper, derivatives are w.r.t. x.

The next result summarises our strategy for analysing local deletion algorithms. Starting with a local rule L_ϕ and recolouring rule c, for a system of d.e.'s (3) arising in the manner described above, there exists a chunky local deletion algorithm with these very same rules, whose performance on a large graph $G\in\mathcal{G}(\mathbf{r})$ is a.a.s. well approximated by this system. Although the functions p_i were motivated by analogy with probabilities, they may exceed 1. Moreover, if the algorithm is applied to any n-vertex r-regular graph

with sufficiently large girth, the expected value of the output function W at the end of the algorithm is close to the likely value achieved on $\mathcal{G}_{n,r}$.

Theorem 5.5. *Fix $r>0$. Consider a local rule L_ϕ, a recolouring rule c and a set of non-negative piecewise Lipschitz-continuous functions $p_i(x)$ $(1\le i\le R)$ on $[0,C]$ for some $C>0$. Define $f_{j,i}$ to be the functions in Lemma 5.2, and let $f_{R+1,i}$ be defined with respect to an output function W as in (2). Let the functions y_j be determined by the solution of (3) with initial conditions (4). Then the following hold.*

(i) For all $\epsilon'>0$, there is an $\epsilon>0$ with $\epsilon<\epsilon'$, and a chunky local deletion algorithm $\mathcal{A}(\epsilon')$ with local rule L_ϕ, recolouring rule c and of granularity at most ϵ' as follows. If $\mathcal{A}(\epsilon')$ is applied to a random n-vertex graph $G\in\mathcal{G}(\mathbf{r})$ with $\max r_i\le r$, then a.a.s. as $n\to\infty$, the number $Y_j(t)$ of vertices of type j in the survival graph G_t satisfies $|Y_j(t)-ny_j(\epsilon t)|<\epsilon' n$ and $|W(t)-ny_{R+1}(\epsilon t)|<\epsilon' n$, for all $1\le j\le R$ and $t\in\{0,\dots,\lfloor C/\epsilon\rfloor\}$.

(ii) For all $\epsilon'>0$, there exists a positive constant g such that the chunky local deletion algorithm $\mathcal{A}(\epsilon'/2)$ of (i) has the following property. Let G_n be an n-vertex, r-regular graph with girth at least g. Let $Y_j(t)$ denote the number of vertices of type j in the survival graph and $W(t)$ the value of the output function W, after t steps of the algorithm. Then, with ϵ as in (i), $|\mathbf{E}(Y_j(t))-ny_j(\epsilon t)|<\epsilon' n$ $(1\le j\le R)$ and $|\mathbf{E}(W(t))-ny_{R+1}(\epsilon t)|<\epsilon' n$, for all $t\in\{0,\dots,\lfloor C/\epsilon\rfloor\}$.

Proof. To prove part (i), choose ϵ such that $0<\epsilon<\epsilon'$, $\epsilon p_i(x)<1$ for all $0\le i\le r$ and $0\le x\le C$ and let $N=\lfloor C/\epsilon\rfloor$. Then define $p_{t,i}=\epsilon p_i(\epsilon t-\epsilon)$ for all $1\le t\le N$. This determines a chunky local deletion algorithm $\mathcal{A}$ with local rule L_ϕ and recolouring rule c. The difference equation in Theorem 5.4 gives precisely the solution of (3) by Euler's method with step size ϵ (apart from a final possible partial step). Hence, applying that theorem, we conclude that a.a.s. $Y_j(t)=ny_j(\epsilon t)+O(\epsilon n)$ $(1\le j\le R)$ and $W(t)=ny_{R+1}(\epsilon t)+O(\epsilon n)$, uniformly for $0\le t\le N$. Part (i) follows upon taking ϵ sufficiently small.

For part (ii), given $\epsilon'>0$, let $\mathcal{A}=\mathcal{A}(\epsilon'/2)$ defined in part (i). Apply Theorem 4.3 to find $g=g(\epsilon')$ for which the vector $s_{r,\mathcal{A}}(t,n)$ of expected values of the components of $\mathbf{s}(t)=\big(Y_1(t),\dots,Y_R(t),W(t)\big)$ is independent of the n-vertex r-regular input graph G whenever the girth of G is at least g. By Theorem 4.5 the size of each component of $\mathbf{s}(t)$, when applied to $\mathcal{G}_{n,r}$, is a.a.s. $s_{r,\mathcal{A}}(t,n)+o(n)$. Finally, part (i) implies that this is a.a.s. within $\epsilon' n/2$ of the corresponding component $ny_j(\epsilon t)$, which implies the desired result. ∎

6. Deprioritised algorithms and Theorem 5.4

The aim of this section is to prove Theorem 5.4 and some further useful results. We employ *deprioritised algorithms*. Recall that the algorithms for independent and dominating sets introduced in Section 2 are prioritised in that minimum degree vertices have priority over others in the selection step. To deprioritise them, a priority list is replaced by an appropriate set of probabilities. One may find an estimate $p_{t,j}$ for the probability that, at the t-th step of the algorithm, the minimum degree of G_{t-1} is equal to some fixed degree j. Then, instead of using prioritisation, one first chooses the degree i with probability $p_{t,i}$ and then selects a vertex v of degree i uniformly at random. In [30] it was shown that a certain class of algorithms acting on random regular graphs yield almost the same results when deprioritised appropriately.

This idea is easily generalized for algorithms on coloured graphs, where we focus on the following special class of chunky local deprioritised algorithms. In making this definition, key functions are expressed with parameters x representing t/n, where the graphs have n vertices, so that the algorithm scales with the graph size in a suitable way.

Definition (Amenable deprioritised algorithm). An algorithm applied to n-vertex graphs is said to be a deprioritised algorithm amenable in an interval $[0,M)$ if it is a local deletion algorithm in which the selection rule consists of the following at step t. Here $\tilde{p}_i\colon [0,M]\to[0,1]$ is some piecewise Lipschitz continuous function, called a *relative selection function*, for each possible type i $(1\le i\le R)$, such that $\sum_{i=1}^{R}\tilde{p}_i(x)=1$ for every $x\in[0,M]$.

(i) A number $i\in\{1,\dots,R\}$ is chosen with probability $p_i(t,n)=\tilde{p}_i(t/n)$;
(ii) a vertex v of type i is chosen uniformly at random.

If, in step (ii), there are no vertices of type i, then the algorithm is terminated; we say that it got *stuck*.

We now consider the random graph $G\in\mathcal{G}(\mathbf{r})$.

Theorem 6.1. *Let* $\mathbf{r}$ *have some probability distribution over the possible degree vectors of dimension* n *with maximum entry* r*. Consider an amenable deprioritised algorithm* $\mathcal{D}$ *with local rule* L_ϕ*, recolouring function* c*, and relative selection functions* $\tilde{p}_i$*, and apply it to* $G\in\mathcal{G}(\mathbf{r})$*. Let* N_j *be the (random) number of vertices of type* j *in* G $(1\le j\le R)$*, let* W *be an output function and set* $N_{R+1}=0$*. For fixed* $\delta'>0$*, let* T *be the first step of the algorithm for which there exists* j*,* $1\le j\le R$*, such that* $\tilde{p}_j(x)>0$ *for some* x

with $|x-T/n|<\delta'$ and the number of vertices of type j in the survival graph is less than $\delta' n$. Let $\mathbf{y}$ be given by the solution of the d.e. system

$$y_j'(x) = \sum_{i=1}^{R} \tilde{p}_i(x) f_{j,i}(y_1, \dots, y_R) \quad (1 \le j \le R+s), \tag{5}$$

with initial conditions $y_j(0) = N_j/n$ for $1 \le j \le R$ and $y_{R+1}(0) = 0$, where $f_{j,i}$ is defined as in Lemma 5.2 and equation (2). Then a.a.s. $W(t) = ny_{R+1}(t/n) + o(n)$, and the number $Y_j(t)$ of vertices of type j in the survival graph G_t is $ny_j(t/n) + o(n)$, uniformly for $0 \le t \le T$.

Proof. Let G_t denote the survival pseudograph obtained after t steps of the algorithm applied to the pairing model with degree sequence $\mathbf{r}$. Similarly, define $Y_j(t)$ and $W(t)$ for this random pseudograph, and set $\mathbf{Y}(t) = (Y_1(t), \dots, Y_R(t), W(t))$. We show that the random vector $\mathbf{Y}(t)$ is a.a.s. sharply concentrated near the solution of the differential equation system (5), using the differential equation method as presented in [27] or [28]. The proof is by induction over the intervals on which the relative selection functions $\tilde{p}_i$ are Lipschitz continuous. Let t_k denote the first value of t for which t/n lies in the kth such interval. The inductive hypothesis is that the claimed approximation holds for $t_{k-1} \le t \le \min\{T, t_k\}$. In particular, at $t = t_k$, each coordinate of the vector is a.a.s. within $o(n)$ of the differential equation solution. Initially, of course the two coincide precisely.

The inductive step concerns only the part of the process from x_k to x_{k+1}, where $x_k = t_k/n$ and $x_0 = 0$. We assume that the functions $p_j(t) = \tilde{p}_j(t/n)$ are Lipschitz continuous in the corresponding t-interval, since a discontinuity after t_{k+1}–1 does not affect the process before t_{k+1}. Also we may extend them to be continuous on intervals before and afterwards. Define $D \subseteq \mathbb{R}^{R+2}$ to be the interior of the set of all $(x, y_1, \dots, y_{R+1})$ satisfying $x_k - \delta' < x < x_{k+1} + \delta'$, $-1 < y_i < C$ for all $i \le R+1$ and x, and additionally $\delta'/2 < y_i$ for each $i \le R$ such that $\tilde{p}_i(x') > 0$ for some x' with $|x' - x| < \delta'/2$. Here C is large enough constant that Cn is a deterministic upper bound on the value of any of the variables Y_i. For $i \le R$ any $C > 1$ will suffice, whilst for the output function, some such C exists because this function is initially zero and has bounded change in one application of a local rule.

Then D is bounded, connected and open for $\delta' > 0$ sufficiently small. To apply [27, Theorem 1] requires verifying a few conditions, which we clarify below. We use the simple modification, given in [28, Theorem 6.1], in which these conditions are required only up to a stopping time T', which we define as $\min\{T, t_{k+1}\}$. See also [28, Theorem 5.1] for some minor variations we make use of here. Note that the vector $(t/n, \mathbf{Y}/n)$ must remain inside D

and cannot come within distance $\delta/2$ of its boundary, up until the time T'. We use the domain D to give 'elbow-room' within which the process behaves well.

The conditions of [27, Theorem 1], and the reasons that they hold, are the following after some suitable trivial modifications. (For instance we shift 0 to x_k, and since each step deletes at least one vertex, the process takes at most $n(x_{k+1}+\delta')$ steps so we ignore $m(n)$.) We define $H_t=(G_0,\dots,G_t)$, the history of the process to time t.

(a) There is a constant C' such that for all $t<T'$ and all $1\le j\le R+1$,

$$|Y_j(t+1)-Y_j(t)|<C'$$

always. This holds since a recolouring step affects a bounded number of vertices, and the resulting change in the output function is bounded.

(b) For all j and uniformly over all $t<T'$,

$$\mathbf{E}(Y_j(t+1)-Y_j(t)|H_t)=\tilde{f}_j(t/n,\mathbf{Y}(t)/n)+o(1)$$

for suitable functions $\tilde{f}_j$. As $t<T$, we have $Y_j(t)\ge\delta' n$ for all j such that $\tilde{p}_j(t/n)>0$. Hence, the algorithm does not terminate at this step. By Lemma 5.3, the survival pairing is distributed u.a.r. given the types of its vertices. Hence, conditioning on H_t, we may apply Lemma 5.2(ii) to conclude that, for $j\le R$, the expected change in $Y_j(t)$ resulting from the next step of the algorithm is $\tilde{f}_j(t/n,\mathbf{Y}(t)/n)+o(1)$ where $\tilde{f}_j(x,\mathbf{y})=\sum_{i=1}^{R}\tilde{p}_i(x)f_{j,i}(\mathbf{y})$. Moreover, the expected change in $W(t)$ is given by $\tilde{f}_{R+1}(x,\mathbf{y})=\sum_{i=1}^{R}\tilde{p}_i(x)f_{R+1,i}(\mathbf{y})$ with $f_{R+1,i}$ defined in (2).

(c) For each j the function f_j is Lipschitz continuous on D. This follows from Lemma 5.2, since amenability implies that the relative selection functions $\tilde{p}_i$ are Lipschitz continuous, and also the number M of Lemma 5.2 is at least $Y_r\ge\delta' n/2$ on D. Moreover, D contains the closure of

$$\{(0,y_1,\dots,y_{R+1})\colon \mathbf{P}(Y_j(0)=y_j n,0\le j\le R+1)\ne 0 \text{ for some } n\}.$$

The conclusion of the theorem is as follows.

(i) The system of differential equations

$$y_j'(x)=\tilde{f}_j(x,y_1,\dots,y_{R+1}),\qquad j=1,\dots,R+1$$

has a unique solution in D for $y_j\colon\mathbb{R}\to\mathbb{R}$, which we denote by $\tilde{y}_j$, passing through

$$\tilde{y}_j(x_k)=Y_j(t_k)/n\quad 1\le j\le R+1,$$

and which extends to points arbitrarily close (in Euclidean distance say) to the boundary of D. (Here $Y_j(t_k)/n$ is deterministic if $k=0$.)

(ii) Asymptotically almost surely

$$Y_j(t) = n\tilde{y}_j(t/n) + o(n)$$

uniformly for $t_k \le t \le \min\{\sigma n, T'\}$ and for each j, where $\sigma = \sigma(n)$ is the supremum of those x to which the solution can be extended.

Suppose that $Y_j(t)$ has not yet reached a point such that the condition in the definition of T' holds. Immediately, by the definition of T', if $Y_j(t)$ is sufficiently close to $n\tilde{y}_j(t/n)$, the point $\tilde{y}_j(t/n)$ has distance at least $\delta'/4$ (say) from the boundary of D. Thus, the approximation in (b) a.a.s. holds for $t_k \le t \le T'$.

By the inductive hypothesis, we have a.a.s.

$$Y_j(t_k)/n = y_j(x_k) + o(1) \quad 1 \le j \le R+1,$$

and hence $\tilde{y}_j(x_k) = y_j(x_k) + o(n)$ a.a.s. Since the derivatives in the differential equation system are Lipschitz, the standard property of solutions of differential equations implies that $\tilde{y}_j(x) = y_j(x) + o(n)$ uniformly for all $x_k \le x < \min\{\sigma, x_{k+1}\}$. Thus, from (b) above and the ensuing conclusions, a.a.s.

$$Y_j(t) = ny_j(t/n) + o(n)$$

uniformly for $t_k \le t \le \min\{T', t_{k+1}\} = \min\{T, t_{k+1}\}$ and for each $1 \le j \le R+1$. The differential equation system in (i) becomes that of (5). This proves the inductive hypothesis. ∎

Note. If part (ii) of the definition of an amenable deprioritised algorithm is altered so that the choice of the next vertex of type i depends (only) on the history H_t and the survival graph G_t, then the conclusion of the theorem still holds. This is because G_t occurs u.a.r. among the possible pseudographs generated by the pairing, given the types of its vertices, and Lemma 5.2 applies given any rule for choosing v of type i.

Proof of Theorem 5.4. Let $\mathcal{A}$ denote the given chunky local deletion algorithm. As discussed before, it is enough to apply the algorithm to $\mathcal{P}(\mathbf{r})$. We will define a deprioritised algorithm behaving similarly to $\mathcal{A}$, within errors $O(\epsilon n)$ in the significant variables. Define Y_j $(1 \le j \le R+1)$ as in the proof of Theorem 6.1. By induction on t, $0 \le t \le N-1$, we will show for each j that

$$Y_j(t) = nz_j(t) + O(\epsilon^2 tn) \quad \text{a.a.s.} \tag{6}$$

We may assume this holds for all t-values smaller than some $t \geq 1$. For $1 \leq i \leq R$, let S_i denote the set of vertices of type i in the survival graph G_{t-1} chosen by $\mathcal{A}$ in this step, and put $Z_i = |S_i|$. Then Z_i has a binomial distribution with expected value $p_{t,i}Y_i(t-1) \leq \epsilon n$, and so by Chernoff's bound, with probability $e^{-\Omega(n)}$,

$$|Z_i - p_{t,i}Y_i(t-1)| \leq \epsilon^2 n, \qquad |Z_i| < 2\epsilon n; \tag{7}$$

and we may assume these henceforth.

Now consider a process $\widetilde{\mathcal{A}}$ which consists of repeatedly applying the following step, beginning with G_{t-1}.

At each step, let S_i' denote the set of vertices of S_i that remain, and still have the degree $d(i)$ associated with type i, in the current survival graph. Let i be minimum such that S_i' is nonempty, select a vertex v u.a.r. from S_i', apply the local rule to v, and update the output sets and the survival graph according to the query graph obtained. We say that this step *processes* v.

Partition $\widetilde{\mathcal{A}}$ into R phases, where phase i consists of processing vertices in S_i' $(1 \leq i \leq R)$, until none remain. Vertices of S_i remaining after phase i must then have lost at least one neighbour each during the process. There is a caveat: if fewer than $\epsilon^2 n$ vertices lie in S_i' at the end of phase $i-1$, phase i is *skipped* and the process proceeds to (considering) phase $i+1$.

Phase i is essentially a deprioritised algorithm, the only difference being that it contains the following two variations. Firstly, the choice of the next vertex of type i is restricted to vertices in the set S_i'. Theorem 6.1 still applies, by the Note after it. Secondly, there is a stopping rule. The conclusion of the theorem still applies at this stopping time. (To see why, see [28, Section 4.2] and the proof of Theorem 6.1 of that paper.)

Let $\widetilde{Y}_j(\xi)$ denote the number of vertices of type j (or, in the case $j = R+1$, the value of the output function W) after ξ steps of the process $\widetilde{\mathcal{A}}$. Apply Theorem 6.1 as discussed above, with $\delta' = \epsilon^2$ and

$$\tilde{p}_j = \begin{cases} 1 & \text{if } j = i \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

Then phase i of $\widetilde{\mathcal{A}}$ a.a.s. finishes with

$$\widetilde{Y}_j(\xi_i)/n = \tilde{y}_j(\xi_i/n) + o(n) \quad 1 \leq j \leq R+1, \tag{9}$$

where ξ_i denotes the last step of phase i (which is random and we will examine shortly), $\xi_0 = 0$, and the $\tilde{y}_j$ satisfy (5) with initial values

$$\tilde{y}_j(\xi_{i-1}/n) = \widetilde{Y}_j(\xi_{i-1})/n.$$

Of course in the case $i=0$, $\widetilde{Y}_j(\xi_{i-1})$ is defined to be $Y_j(t-1)$, the random variable inputted to the process $\mathcal{A}$, which by induction on t satisfies $Y_j(t-1)=nz_j(t-1)+O(\epsilon^2 tn)$. Recall that $p_i=1$ in phase i, the functions $f_{j,i}$ are Lipschitz, and the length $\xi_i-\xi_{i-1}$ of phase i is $O(\epsilon n)$ by (7). Hence

$$\begin{aligned}\tilde{y}_j(\xi_i/n) &= \frac{\widetilde{Y}_j(\xi_{i-1})}{n} + \frac{\xi_i-\xi_{i-1}}{n} f_{j,i}\big(\tilde{\mathbf{y}}(\xi_{i-1}/n)\big) + O(\epsilon^2)\\ &= \frac{\widetilde{Y}_j(\xi_{i-1})}{n} + \frac{\xi_i-\xi_{i-1}}{n} f_{j,i}\big(\tilde{\mathbf{y}}(0)\big) + O(\epsilon^2),\end{aligned}$$

where $\tilde{\mathbf{y}}=(\tilde{y}_1,\dots,\tilde{y}_{R+1})$. Thus, by induction using (9), the final values at the end of the process $\widetilde{\mathcal{A}}$ are a.a.s. given by

$$\tilde{y}_j(\xi_R/n) = \tilde{y}_j(0) + \sum_{i=0}^{R} \frac{\xi_i-\xi_{i-1}}{n} f_{j,i}\big(\tilde{\mathbf{y}}(0)\big) + O(\epsilon^2) \tag{10}$$

for $1\le j\le R+1$.

Before proceeding, we need to bound the effect of the vertices being deleted during the process $\widetilde{\mathcal{A}}$. We call a vertex *susceptible* if it is a member of $S=\bigcup_{i=1}^{R} S_i$ and has distance at most $2D$ from some other vertex in S, that is, in the terminology of Section 4, it forms a pre-clash with some other vertex of S. Let A_t denote the set of susceptible vertices.

We next claim that, for some constant $C_0>0$ depending only on r and on the number of colours, conditional on the graph G_{t-1}, $\mathbf{P}(|A_t|>C_0\epsilon^2 n)=e^{-\Omega(n)}$. To establish this claim first note that by (7), the probability that a given vertex is contained in S and is furthermore susceptible is $O(\epsilon)$ (noting that D is fixed, determined by the local rule, and the maximum degree of G_{t-1} is at most r). Hence $\mathbf{E}|A_t|=O(\epsilon^2 n)$.

On the other hand, sharp concentration of $|A_t|$ can easily be proved as follows. Note that a simple switching (replacing two pairs on four points in the pairing by any two other pairs on the same points) can only change $|A_t|$ by an amount bounded by a function of D and r. For r-regular graphs, [29, Theorem 2.19] immediately implies that $\mathbf{P}(|A_t|>\mathbf{E}|A_t|+\epsilon^2 n)=e^{-\Omega(n)}$. Usually G_{t-1} is not regular, but the proof of that theorem still applies easily for graphs of maximum degree r. Hence, the claim holds, and we proceed assuming $|A_t|\le C_0\epsilon^2 n$.

We now turn to consideration of the lengths of the phases, i.e., $\xi_i-\xi_{i-1}$. Note that a non-susceptible vertex in S of type i will definitely retain degree $d(i)$ in the survival graph until at least the start of phase i. Hence, if the phase is not skipped, the number of vertices of degree $d(i)$ in S_i available

for processing must be between Z_i and $Z_i - A_t$, and the phase cannot finish until there are at most $\delta' n = \epsilon^2 n$ of them left. It follows that

$$\xi_i - \xi_{i-1} = Z_i + O(\epsilon^2 n). \tag{11}$$

On the other hand, if the phase is skipped then $\xi_i - \xi_{i-1} = 0$ and $Z_i - A_t < \epsilon^2 n$, so (11) holds in all cases. Then by (7) and (10)

$$\tilde{y}_j(\xi_R/n) = \tilde{y}_j(0) + \sum_{i=1}^{R} p_{t,i} \frac{Y_i(t-1)}{n} f_{j,i}\big(\tilde{\mathbf{y}}(0)\big) + O(\epsilon^2).$$

Recall that $\tilde{\mathbf{y}}(0) = \mathbf{Y}(t-1)/n$, so by the inductive claim (6) for $t-1$, we have $\tilde{y}_j(0) = z_j(t-1) + O_1(\epsilon^2(t-1))$ for each j. We use $O_1()$ to distinguish the constant here, which is bounding the inductive error. By the Lipschitz property of each $f_{j,i}$, it follows that $f_{j,i}\big(\tilde{\mathbf{y}}(0)\big) = f_{j,i}\big(\mathbf{z}(t-1)\big) + O(\epsilon^2(t-1)) = f_{j,i}\big(\mathbf{z}(t-1)\big) + O(\epsilon)$ since $t \le N = O(1/\epsilon)$ by hypothesis in the theorem. Hence, using $p_{t,i} \le \epsilon$ and $Y_i \le n$, together with (9) for $i = R$, and then (6) again for the second step, we obtain

$$\begin{aligned}
\tilde{Y}_j(\xi_R)/n &= z_j(t-1) + O(\epsilon^2(t-1)) + \sum_{i=1}^{R} p_{t,i} \frac{Y_i(t-1)}{n} \Big(f_{j,i}\big(\mathbf{z}(t-1)\big) + O(\epsilon) \Big) \\
&= z_j(t-1) + O_1(\epsilon^2(t-1)) + \sum_{i=1}^{R} p_{t,i} z_i(t-1) \Big(f_{j,i}\big(\mathbf{z}(t-1)\big) + O(\epsilon^2) \Big) \\
&= z_j(t) + O_1(\epsilon^2(t-1)) + O(\epsilon^2).
\end{aligned}$$

To establish (6) inductively, it only remains to show that $\tilde{Y}_j(\xi_R) = Y_j(t) + O(\epsilon^2 n)$ a.a.s. This fact is easy to see, because, in view of (11), the same vertices are processed in $\widetilde{\mathcal{A}}$ and $\mathcal{A}$ except for a set of $O(\epsilon^2 n)$ vertices, each of which has a bounded effect on the value of Y_j or $\tilde{Y}_j$. We now have (6), so it is true for $t = N$. Recalling that $\epsilon t \le \epsilon N = O(1)$, we deduce that the value of the output function at the end of the algorithm is equal to $n z_{R+1}(N) + O(\epsilon n)$, and the conclusion of the theorem follows. ∎

Before proceeding, we restate the conclusion of Theorem 6.1.

Alternative Conclusion for Theorem 6.1. Let $\delta > 0$ and let x_0 be the infimum of all $x \ge 0$ for which there exists j such that $\tilde{p}_j(x) > 0$ and $y_j(x) < \delta$. Then a.a.s. $Y_j(t) = n y_j(t/n) + o(n)$ and $W(t) = n y_{R+1}(t/n) + o(n)$ for all $t \le \lfloor (x_0 - \delta) n \rfloor$.

To see why this follows, define $\delta' = \delta/2$ and note that the conclusion of Theorem 6.1 implies that for $0 \le t \le T$ the value of $Y_j(t)$ differs from $n y_j(t/n)$ by at most $o(n)$, so that step T a.a.s. does not occur before $n(x_0 - \delta)$.

Theorem 6.2. *Assume the hypotheses of Theorem 6.1, let $\delta>0$ and define x_0 as in the alternative conclusion given above. Then there exists a number ϵ, $0<\epsilon<\delta$, and a chunky local deletion algorithm $\mathcal{A}$ with the same local and recolouring rules as $\mathcal{D}$, and granularity at most δ, such that, when both algorithms are applied to $\mathcal{G}_{n,r}$, a.a.s. $||\mathbf{Y}_{\mathcal{A}}(t)-\mathbf{Y}_{\mathcal{D}}(\lfloor \epsilon n t\rfloor)||\leq \delta n$ for all $0\leq t\leq \lfloor (x_0-\delta)/\epsilon\rfloor$. Here $\mathbf{Y}_{\mathcal{A}}(t),\mathbf{Y}_{\mathcal{D}}(t)\in\mathbb{R}^{R+1}$ denote the vectors whose components $j\leq R$ and $R+1$ are the number of vertices of type j in the survival graph, and the value of the output function, respectively, after t steps of the algorithm.*

Proof. Put $\delta'=\delta/2$ and let $\mathbf{y}=(y_1,\dots,y_{R+1})$ denote the solution of the differential equation system (5) in the domain D used in the proof of Theorem 6.1 up until the endpoint is reached.

Define $p_i(x)=\tilde{p}_i(x)/y_i(x)$, where y_i is precisely the solution function just determined, for all $0\leq x\leq x_0$. Then, on this interval, the differential equation system (5) is identical to (3), with the same initial conditions. Hence, this theorem follows from Theorem 5.5(i) applied with $\epsilon'=\delta$ and $C=x_0-\delta$, and the alternative conclusion of Theorem 6.1 with t replaced by $\lfloor \epsilon n t\rfloor$. Note that the fact that the functions $\tilde{p}_i$ are bounded implies that the floor function has no significant effect. ∎

Theorem 6.2 has an immediate consequence for regular graphs with sufficiently large girth which will be useful in the next section.

Theorem 6.3. *Fix integers $r,D\geq 1$. Consider an amenable deprioritised algorithm $\mathcal{D}$ with local rule L_ϕ of depth D, recolouring function c and relative selection functions $\tilde{p}_i$. Let W be an output function. Let $\mathbf{y}$ be given by the solution of the differential equation system (5) with initial conditions $y_1(0)=1$ and $y_j(0)=0$ for $j>1$, where $f_{j,i}$ is defined as in Lemma 5.2 and (2). Let $\delta>0$ and define $x_0=x_0(\delta)$ as the infimum of all $x\geq 0$ for which there exists j such that $\tilde{p}_j(x)>0$ and $y_j(x)<\delta$. Then there exists some $\epsilon>0$ with $\epsilon<\delta$, a constant g, and a chunky local deletion algorithm $\mathcal{A}$ with local rule L_ϕ, recolouring function c and granularity at most δ such that, for every n-vertex r-regular graph G with girth at least g,*

$$||s_{r,\mathcal{A}}(G,t)-n\mathbf{y}(\epsilon t)||_\infty<\delta n,$$

for $0\leq t\leq \lfloor (x_0-\delta)/\epsilon\rfloor$, where $s_{r,\mathcal{A}}(G,t)$ is the vector of expected values for algorithm $\mathcal{A}$ defined in Theorem 4.3.

Proof. Let $\delta>0$ and x_0 be as stated. Theorem 6.1 implies that the performance of $\mathcal{D}$ applied to a random r-regular graph is a.a.s. determined by the solution of (5) up until the step T defined there. Note that the definition of

x_0 ensures that $T \geq (x_0 - \delta)n$ a.a.s. By Theorem 6.2 (applied with δ replaced by $\delta/2$), there is a chunky local deletion algorithm $\mathcal{A}'$ whose performance in $\mathcal{G}_{n,r}$ is tracked by a vector $\mathbf{Y}$ that differs from its counterpart in an application of $\mathcal{D}$ by at most $(\delta/2)n$. The 'steps' of that algorithm are of size $\epsilon < \delta$. The algorithm $\mathcal{A}'$ is obtained in the proof of Theorem 6.2 by appealing to Theorem 5.5(i). The girth g and the algorithm $\mathcal{A}$ may be obtained from the further information given in Theorem 5.5(ii), using $\epsilon' = \delta/2$. ∎

7. Applications

In this section, we show that the analysis of deprioritised algorithms acting on random regular graphs in [30] may be carried over to the large girth setting, at least for a general class of algorithms including many to which it has been applied. Bounds already known to hold asymptotically for random regular graphs then immediately imply deterministic bounds for graphs whose girth is sufficiently large, and this is almost entirely by virtue of their proof via [30, Theorem 1]. After proving a general theorem to this effect, we give some applications in which Lemma 4.6 is used to show that the output set of the chunky algorithm is (for the properties being considered) within ϵn size of the set of interest.

In [30] the performance of DGQ algorithms (see Section 2) acting on $\mathcal{G}_{n,r}$ is determined a.a.s. Recall that for these algorithms each basic operation is one of several types, called Op_i $(1 \leq i \leq r)$, consisting of selecting a vertex v of degree i in the survival graph u.a.r., and then applying a specified sequence of randomised tasks (including deletion of v). The setting is $\mathcal{P}_{n,r}$; we transfer any operations from graphs to pairings in the obvious way; as explained in the comment just after Lemma 5.2, how this is done is immaterial. For appropriate operations, [30, Theorem 1] concludes that there is a randomised algorithm using these operations, such that a.a.s. at some point in the algorithm the output set size is asymptotic to ρn and, for $i = 1, \ldots, r$, the number of vertices of degree i is asymptotic to ρ_i. Here ρ and ρ_i are determined by solving d.e. system. At the end of this subsection is a special case of [30, Theorem 1]. We omit the general case here since its conditions have already been verified in the papers we refer to. Vertices of degree 0 are normally ignored because they can usually be forbidden in the survival graph by design of the operations.

Consider a local rule L and recolouring function c from a native local deletion algorithm. The sequence of random tasks for Op_i (after selection of v) consists obtaining a copy $\psi(H)$ of a query graph H via L, adding a subset of vertices of $\psi(H)$ to the output set $\mathcal{O}$ according to some pre-determined

rule, and then deleting the vertices in $\psi(H)$. Such an operation Op_i is a degree-governed query operation, and we say that it *arises* from L and c.

Given appropriate operations Op_i for $1 \le i \le r$, let $\rho(\mathrm{Op}_1, \dots, \mathrm{Op}_r)$ and $\rho_i(\mathrm{Op}_1, \dots, \mathrm{Op}_r)$ denote the constants ρ and ρ_i referred to above which are determined in the conclusion of [30, Theorem 1], in the guise of $\tilde{y}_{r+1}(x_m)$ and $\tilde{y}_{r-i}(x_m)$, respectively.

Theorem 7.1. *Fix $r \ge 1$, a local rule L and an recolouring function c. Suppose that Op_i, $1 \le i \le r$, are degree-governed query operations arising from L and c, and that the pairing versions of these operations satisfy the assumptions of [30, Theorem 1]. Let $\rho, \rho_1, \dots, \rho_r$ be the constants defined above. Then, for all $\delta > 0$, there exists a chunky native local deletion algorithm $\mathcal{A}$ with the very same local rule L and recolouring function c, and granularity at most δ, such that when $\mathcal{A}$ is applied to any n-vertex r-regular graph G of sufficiently large girth, there is a step T in which the expected size of the set $\mathcal{O}_1$ of vertices with output colour 1 is within δn of ρn, and where the expected number of vertices of degree i in the survival graph G_T is within δn of $\rho_i n$ for $1 \le i \le r$. Moreover, the number of vertices with colour $\propto'$ or $\propto$ at step T is at most δn.*

Proof. Here we assume the reader is somewhat familiar with the proofs in [30]. Before turning to the proof, we illustrate some of the main concepts with an example. Consider the DGQ algorithm P_{ind} introduced in Section 2: operation Op_i consists of choosing a vertex of degree i in the survival graph, which is added to the independent set and deleted, along with its i neighbours, from the survival graph. The prioritised algorithm P_{ind} determines, at each step, the minimum degree i of the survival graph and performs operation Op_i. The main contribution of [30] was to show that such an algorithm may be deprioritised: instead of adhering to the priorities, a probability vector $\mathbf{p} = \mathbf{p}(n, x) = (p_1, \dots, p_r)$ was prescribed for each survival graph G_t, where the vector depends only on the number n of vertices in the original graph and on a parameter x, which is related with the current step t of the algorithm. Now, instead of selecting a vertex with minimum degree, the algorithm would first choose a degree j according to this probability vector and then perform Op_j. There are some inherent complications, such as the need to ensure that G_t contains a vertex of degree j if the algorithm calls for one and the need to set the probabilities appropriately to ensure that its performance can be made arbitrarily close to the prioritised case. Chunkifying this algorithm means to find probability functions $p_i(x)$ (x is again related with the current step of the algorithm), for all $i \in \{1, \dots, r\}$, so that, instead of choosing the degree according to a probability vector

and then choosing a single vertex with this degree, we select each vertex of degree i with the corresponding probability, independently from the other vertices, and perform Op_i for each of them. As a consequence, a chunk of vertices is processed in each round. Here, one difficulty is again to fix probabilities in a way that there is no significant loss of performance. To prove Theorem 7.1, we combine results on deprioritisation in [30] with results on chunky algorithms in the current paper.

The conclusion of [30, Theorem 1] asserts the existence of a randomised algorithm that appropriately approximates the DGQ algorithm implicit in the hypotheses of the present theorem, and it is straightforward to check that the proof of [30, Theorems 1 and 2] consists of constructing an amenable deprioritised algorithm whose local rule and recolouring function are determined by Op_i. More precisely, it is the family of algorithms with parameters mentioned below. The first assumption of [30, Theorem 1] is that each operation Op_i satisfies [30, Eq. (2.2)] when (in present notation) the number, Y_r, of vertices of degree r, is at least ϵn (for fixed $\epsilon > 0$). This requires that for some fixed functions $f_{k,i}(x,\mathbf{y}) = f_{k,i}(x,y_1,\dots,y_{r+1})$, the expected increase in Y_k in one step of the algorithm involving Op_i is $f_{k,i}(t/n, Y_1(t)/n, \dots, Y_{r+1}(t)/n) + o(1)$ for $i = 1,\dots,r$, $k = 1,\dots,r+1$. Here Y_{r+1} denotes the size of the output set, and the convergence in $o(1)$ must be uniform over all states of the algorithm with $Y_r \geq \epsilon n$.

Since Op_i arises from L and c, we may apply Lemma 5.2(b) and (c), and we deduce that these functions $f_{k,i}$ exist and must be exactly the same as the functions appearing in Lemma 5.2 when all arguments are nonnegative and $Y_r > \epsilon n$. Theorem 6.3 refers to the same functions $f_{k,i}$.

We pause to summarise the proof of [30, Theorem 2]. Probability functions $\tilde{p}_i$ are constructed to complete the definition of a deprioritised algorithm that uses the operations Op_i mentioned above. The 'natural' choice of probabilities is altered in order to keep key variables strictly positive; this uses an arbitrary $\epsilon_1 > 0$ that is one of the parameters of the family of algorithms. The solution of the differential equations given by (5) (c.f. [30, Equation (3.5)]) is named $\tilde{\mathbf{y}}^{(\mathbf{1})} = (\tilde{y}_1^{(1)}, \dots, \tilde{y}_{r+1}^{(1)})$ to differentiate it from the solution $\tilde{\mathbf{y}}$ which arises when the natural probabilities are used. The initial condition in both cases is $(0,\dots,0,1,0)$, so in particular $\tilde{y}_r^{(1)}(0) = 1$. These two vector solution functions are shown to be arbitrarily close to each other up to a point x_m, when ϵ_1 is arbitrarily close to 0 (see [30, (4.25)]). The solution $\tilde{\mathbf{y}}$ is such that $(x,\tilde{\mathbf{y}})$ lies inside a domain $\mathcal{D}_\epsilon$ for $0 \leq x \leq x_m$ and $\tilde{y}_r \geq \epsilon$ at all points inside it.

Given $\delta > 0$, to apply Theorem 6.3 we need to determine x_0, which is the infimum of all $x \geq 0$ for which there exists j such that $\tilde{p}_j(x) > 0$ and

$\tilde{y}_j^{(1)}(x) < \delta$. We claim that for δ sufficiently small, the functions $\tilde{p}_j$ in the proof of Theorem 2 of [30] are defined in such a way that guarantees that x_0 is at least as large as the quantity $x_m^{(1)}$ defined in [30]. As stated at the end of that proof, $x_m^{(1)}$ can be taken arbitrarily close to x_m. To see why this claim is true, first observe that in Part 2 of the proof of [30, Theorem 2], the probability vector in the deprioritised algorithm is defined so that $\tilde{p}_r = 1$ and $\tilde{p}_i = 0$ for $i < r$ when $x \in [0, \epsilon_1]$ where ϵ_1 is as above. The fact that vertices of degree r are abundant (for sufficiently small ϵ_1) guarantees that the steps of the algorithm can (a.a.s.) be carried out as prescribed. On $[0, x_1]$, the function $\tilde{y}_r^{(1)}$ is bounded away from 0, and on $[\epsilon_1, x_m]$, the functions $\tilde{y}_j^{(1)}$ $(1 \le j \le r)$ are all bounded away from 0. These claims follow from some of the observations in the proof in [30], as we describe in more detail below. This then implies that $x_0 \ge x_m^{(1)}$ as desired.

The claim about $\tilde{y}_r^{(1)}$ follows directly from the facts mentioned above, which imply that $\tilde{y}_r(x) \ge \epsilon$ for $0 \le x \le x_m$, and that the difference between $\tilde{\mathbf{y}}^{(\mathbf{1})}$ and $\tilde{\mathbf{y}}$ is arbitrarily small by choice of $\epsilon_1, \ldots$. We next show that y_i remains positive on $[\epsilon_1, x_m]$ $(1 \le i \le r-1)$; being continuous, it is then bounded below by a positive constant. First, in the interval $[\epsilon_1, x_1]$, called phase 1, this is shown in [30, (4.16), (4.18), (4.19)]. The general argument for the later phases, covering the interval $[x_1, x_m^{(1)}]$, is a straightforward variation of the argument for the first phase, sketched as follows. In phase k, there is a preprocessing subphase which gets all $\tilde{y}_i$ strictly positive. Then, $\tilde{y}_{d-k}^{(1)}$ must remain positive by definition of the phases (i.e., the number x_k) except perhaps in the final phase, where it can reach 0 at x_m. The definition of $x_m^{(1)} < x_m$ excludes that case. By definition of the probabilities, $\tilde{y}_{d-k}^{(1)}$ has zero derivative in phase k and hence remains positive. For all other $1 \le i \le r-1$, $\tilde{y}_i^{(1)}$ cannot reach 0 because by hypothesis (B) in [30], its derivative is at least $-C_2 \tilde{y}_i^{(1)}$ for some constant C_2 (so it is bounded below by an exponentially decaying function). In the case $i = m$, $\tilde{y}_i^{(1)}$ is prevented from reaching 0 by the definition of the domain $\mathcal{D}_{\epsilon,M}$ of [30, (3.3)], which requires $y_d > \epsilon$, and the fact that x_m cannot exceed the first point that the solution leaves the domain $\mathcal{D}_{\epsilon,M}$.

Given this, we may apply Theorem 6.3 to deduce that, for all $\delta' > 0$, there is $\epsilon > 0$, a constant g and a chunky native local deletion algorithm $\mathcal{A}$ with local rule L_ϕ and granularity at most δ' such that

$$\left\| s_{r,\mathcal{A}}(G,t) - n\tilde{y}_i^{(1)}(\epsilon t) \right\|_\infty < \delta' n,$$

for $0 \le t \le \lfloor (x_0 - \delta')/\epsilon \rfloor$, where $s_{r,\mathcal{A}}(G,t)$ denotes the vector whose components $j \le r$ and $r+1$ are the expected number of vertices of type j in the survival graph and the expected cardinality of the output set of colour 1 when $\mathcal{A}$ is applied for t steps to an r-regular input graph G with girth at least g.

Since $x_m^{(1)} \le x_0$, we can replace x_0 by $x_m^{(1)}$ in this statement. By the fact mentioned above that $\tilde{\mathbf{y}}^{(\mathbf{1})}$ and $\tilde{\mathbf{y}}$ can be made arbitrarily close by appropriate choice of parameters, we can substitute one for the other in this conclusion, at the expense of replacing δn by $2\delta n$. Since $\tilde{y}_r$ has bounded derivative on $\mathcal{D}_\epsilon$, and $x_m^{(1)}$ can be made arbitrarily close to x_m, we deduce that by terminating $\mathcal{A}$ at an appropriate point corresponding to $x_m^{(1)}$, we can achieve the expected size of output colour 1 to be at most $C\delta' n$ different from $\tilde{y}_{r+1}(x_m)n$, for some constant C independent of δ'. Taking $\delta' < \delta/C$ gives the required result for output colour 1. A similar argument applies to all other variables. ∎

We next discuss some quick and direct applications of Theorem 7.1 using deprioritised algorithms whose applications to random regular graphs were previously analysed via [30, Theorem 1].

Maximum independent set

Asymptotic lower bounds on the size of a maximum independent set (see Section 2) in a random r-regular graph were obtained by Wormald [27]. The simpler of the two algorithms analysed in [27] was chunkified and then analysed directly for large-girth graphs in [21]. With Theorem 7.1 in hand, we can easily deduce that the stronger results of the second algorithm also carry over to the large-girth case. Slightly better results for the random case were obtained by Duckworth and Zito [8] using a prioritised native local deletion algorithm that outputs an independent set and achieves its improvement by looking more carefully into the neighbourhood of the selected vertices. It applies [30, Theorem 1], so by Theorem 7.1, for any fixed $\delta > 0$, there is $g > 0$ and a chunky local deletion algorithm with the same local and recolouring rules, whose output on r-regular graphs of girth at least g is a set whose expected size differs from the output of the original algorithm by at most δn. By nature of the algorithm it is easy to check that the output set of the chunky algorithm is independent. Letting $\delta \to 0$, we deduce that essentially the same bounds hold in the large girth case. See Table 1 for small r. For comparison, we provide the best known upper bounds $\alpha_u(r)$ on the best possible values of $\alpha(r)$. That is, $\alpha_u(r) \ge \alpha$ for all α such that there exists g for which all r-regular graphs of girth at least g have an independent set of cardinality at least αn. The values of $\alpha_u(r)$ were obtained by McKay [23] using random regular graphs as described for $c_u(r,P)$ in Section 2.

r	3	4	5	6	7
$\alpha(r)$	0.43475	0.39213	0.35930	0.33296	0.31068
$\alpha_u(r)$	0.45537	0.41635	0.38443	0.35799	0.33567
$\gamma(r)$	0.27942	0.24399	0.21852	0.19895	0.18329
$\gamma_\ell(r)$	0.2641	0.2236	0.1959	0.1755	0.1596

Table 1. Lower bounds α for maximum independent sets from [8], and upper bounds γ for minimum independent dominating sets [6]. Best possible values are α_u [23] and γ_ℓ [32]

The lower bounds $\alpha(r)$ are the best known for large girth r-regular graphs when $r \geq 4$. Improvements for the case $r = 3$ are mentioned in Section 1.[1]

Minimum independent dominating set

Duckworth and Wormald [6] gave upper bounds on the size of a minimum independent dominating set in a random r-regular graph using a deprioritised algorithm. These are still the best known upper bounds on the size of *any* dominating sets in these graphs. The bounds are based on the size ρn of the output set of this algorithm (which is an independent set) at the point when the number of vertices in the survival graph falls to ξn for some preselected $\xi > 0$. An independent dominating set may be obtained by judiciously adding some of these vertices greedily. Undominated clash vertices can also be treated greedily after we apply Theorem 7.1. Hence there is a deprioritised algorithm that, after adding a negligible number of vertices, a.a.s. produces an independent dominating set of size at most $\rho n + 2\xi n$. We deduce the same numerical upper bounds $\gamma(r)n$ on the minimum independent dominating set in an r-regular graph of sufficiently large girth. See Table 1 for some specific values of $\gamma(r)$ (more are supplied in [6]), and also lower bounds $\gamma_\ell(r)$ obtained via random regular graphs (see Zito [32]). The upper bound $\gamma(3)$ improves that obtained in [20] (0.299871).

***k*-independent sets, *k*-dominating sets and *k*-independent matchings**

For any positive integer k, a k-independent set of a graph is a set of vertices such that the minimum distance between any two vertices in the set is at least $k+1$. Duckworth and Zito [7], and Beis, Duckworth and Zito [1] obtained lower bounds on the size of these for random r-regular graphs through the analysis of an algorithm which satisfies the requirements for Theorem 7.1, and whose operations one can easily check to be degree-governed query operations. Whenever a vertex v is selected, the algorithm tries to find in G

[1] Using the results of the present paper, Csóka [Independent sets and cuts in large-girth regular graphs, arXiv:1602.02747] recently improved the lower bounds to 0.44533 for 3-regular graphs and to 0.40407 for 4-regular graphs.

one or more balanced r-regular trees of height $\lceil k/2 \rceil$ of which v is a leaf, in a precise way which we do not describe here. If it succeeds, it deletes all vertices in the tree and adds its root to the k-independent set. Otherwise, it deletes all edges corresponding to open adjacencies that were queried. In this case, the local rule is not normal, as it might be that only some edges of a vertex w were queried before finding a new vertex of degree less than r, at which point the local exploration is terminated. The advantage is that such a vertex w is not deleted at the end of the step, so that it may still be a leaf of a balanced r-regular tree found in a later step of the algorithm. Similarly to the case of independent sets above, Theorem 7.1 shows that the lower bounds given in [1] must also be valid lower bounds for the size of the largest k-independent set in r-regular graphs of sufficiently large girth.

For any positive integer k, a k-dominating set of a graph $G = (V, E)$ is a set of vertices S such that every vertex $v \in V$ is at distance at most k from S. Duckworth and Mans [5] analysed an algorithm that creates small k-dominating sets in random r-regular graphs, which can be checked to be essentially a native local deletion algorithm. It was analysed using [30, Theorem 1]. Implicit in their argument is the requirement that the leftover vertices do not affect their bounds, which must have been checked numerically as for the dominating set example above. Applying Theorem 7.1 as for the above examples, these are also valid bounds for r-regular graphs with large girth.

For any positive integer k, a k-separated matching in a graph is a set of edges such that the minimum distance between any two edges in the set is at least k. For $k \geq 2$, Beis, Duckworth and Zito [1] found lower bounds on the size of such an object in a random r-regular graph, and by the same remarks as above, these bounds carry over to every r-regular graph of sufficiently large girth.

8. Final remarks

In this paper, we used local algorithms to transfer several results proved for random regular graphs into (deterministic) results about all regular graphs with sufficiently large girth. Our methods have a basis in earlier, much less sophisticated ones, introduced in [21]. In order to establish our main results, we employ sharp concentration techniques. This should not be surprising, as they are often employed for related problems on triangle-free graphs. On the other hand, the arguments in [21] and the ensuing papers were able to make do with expectation alone. The switch in approach considerably reduced the difficulty of proving the relationship with random regular graphs. In

particular, the papers following [21] establish certain 'independence lemmas' using ad-hoc and sometimes tedious arguments. In our work here, this is replaced by a different argument, using the link to random regular graphs.

The original method in [21] lead to explicit bounds on the parameter involved for graphs of given girth, and was adapted by Gamarnik and Goldberg [10] to various paramaters in r-regular graphs. Similarly, in [17] whose method ultimately evolved from [21], explicit values of girth are given. Unfortunately, our argument has now lost the direct connection with explicit girth. However, with some work, bounds could still be obtained as a function of girth.

We also remark that the kinds of bounds we obtain for r-regular graphs with large girth sometimes carry over to graphs with large girth and *maximum* degree r. This is true, for instance, if P is *monotone*, that is, if H satisfies P, then every subgraph of H also satisfies P in G. Fix a monotone property P and suppose that there exists $g > 0$ such that every r-regular graph H with girth at least g contains an induced subgraph H_0 satisfying P such that $|H_0| \geq \gamma |V(H)|$ (and hence $f_P(H) \geq \gamma$). It is not hard to see that, given an n-vertex graph G with maximum degree r and girth at least g, we may construct a graph G' by taking copies of G and joining vertices in different copies so as to make G' r-regular. This can be done without decreasing the girth if sufficiently many copies of G are used. In particular, given a monotone property P, we have the inequality $f_P(G)/n \geq f_P(G')/|V(G')|$, since this holds for the copy of G containing the largest number of vertices in a maximum induced subgraph with property P in G'. Thus, bounds for r-regular graphs with large girth imply bounds for graphs with large girth and maximum degree r. For instance, the property of having chromatic number at most k is monotone, and for $k = 1$, this means that the graph is an independent set. Hence, we may deduce results about the size of a largest independent set in a graph in terms of its maximum degree, using the results of Section 7.

We note that our methods can readily be extended so as to apply to r-regular bipartite graphs, or indeed multipartite graphs, of large girth. Most of the technical results would go through with little effort. In particular, the sharp concentration result via switchings in the proof of Theorem 5.4 is easily modified appropriately.

A general open problem is to determine how much the definition of local deletion algorithms can be relaxed, such that they still behave asymptotically equivalently on random regular graphs and regular graphs of large girth.

Another general problem concerns the quality of the bounds that may be obtained through local algorithms. Recently, Gamarnik and Sudan [11] showed that, for sufficiently large r, local algorithms (which include our local deletion algorithms) cannot approximate the size of the largest independent set in an r-regular graph of large girth with an arbitrarily small multiplicative error. The approximation gap was improved by Rahman and Virág [24]. However, these results appear to say nothing about small r, say $r=3$.

Finally, we note that the relevant local structure of the graphs we consider is fixed – the r-regular tree. Further investigation of this area may be desirable given the recent interest in graph limits in the sparse case. See, for instance, Lovász [22], Hatami, Lovász and Szegedy [12].

Acknowledgment. We thank the referees for providing useful comments on the presentation of our results.

References

[1] M. Beis, W. Duckworth and M. Zito: Packing vertices and edges in random regular graphs, *Random Structures & Algorithms* **32** (2008), 20–37.

[2] E. A. Bender and E. R. Canfield: The asymptotic number of labeled graphs with given degree sequences, *J. Combinatorial Theory Ser. A* **24** (1978), 296–307.

[3] B. Bollobás: *Random graphs*, Academic Press, London, 1985.

[4] E. Csóka, B. Gerencsér, V. Harangi and B. Virág: Invariant Gaussian processes and independent sets on regular graphs of large girth, *Random Structures and Algorithms* **47** (2015), 284–303.

[5] W. Duckworth and B. Mans: Randomized greedy algorithms for finding small k-dominating sets of random regular graphs, *Random Structures and Algorithms* **27** (2005), 401–412.

[6] W. Duckworth and N. C. Wormald: On the independent domination number of random regular graphs, *Combinatorics, Probability and Computing* **15** (2006), 513–522.

[7] W. Duckworth and M. Zito: Large 2-independent sets of regular graphs, *Electronic Notes in Theoretical Computer Science* **78** (2003), 1–13.

[8] W. Duckworth and M. Zito: Large independent sets in random regular graphs, *Theoretical Computer Science* **410** (2009), 5236–5243.

[9] P. Erdős: Graph theory and probability, *Canadian Journal of Mathematics* **11** (1959), 34–38.

[10] D. Gamarnik and D. A. Goldberg: Randomized greedy algorithms for independent sets and matchings in regular graphs: exact results and finite girth corrections, *Combin. Probab. Comput.* **19** (2010), 61–85.

[11] D. Gamarnik and M. Sudan: Limits of local algorithms over sparse random graphs, http://arxiv.org/abs/1304.1831. *Proceedings of the 5th conference on Innovations in theoretical computer science* (2014), 369–376.

[12] H. Hatami, L. Lovász and B. Szegedy: Limits of locally-globally convergent graph sequences, *Geom. Funct. Anal.* **24** (2014), 269–296.

[13] G. Hopkins and W. Staton, Girth and independence ratio, *Canadian Mathematical Bulletin* **25** (1982), 179–186.

[14] C. Hoppen: *Properties of graphs with large girth*, Doctoral thesis, University of Waterloo, 2008.

[15] C. Hoppen and N. Wormald: Local algorithms, regular graphs of large girth, and random regular graphs, http://arxiv.org/abs/1308.0266v2.

[16] C. Hoppen and N. Wormald: Properties of regular graphs with large girth via local algorithms *Journal of Combinatorial Theory, Series B* **121** (2016), 367–397.

[17] F. Kardoš, D. Král and J. Volec: Fractional colorings of cubic graphs with large girth, *SIAM J. Discrete Math.* **25** (2011), 1454–1476.

[18] F. Kardoš, D. Král and J. Volec: Maximum edge-cuts in cubic graphs with large girth and in random cubic graphs, *Random Structures & Algorithms* **41** (2012), 506–520.

[19] K. Kawarabayashi, M. D. Plummer and A. Saito: Domination in a graph with a 2 factor, *J. Graph Theory* **52** (2006), 1–6.

[20] D. Král, P. Škoda and J. Volec: Domination number of cubic graphs with large girth, *J. Graph Theory* **69** (2012), 131–142.

[21] J. Lauer and N. Wormald: Large independent sets in random graphs with large girth, *Journal of Combinatorial Theory, Series B* **97** (2007), 999–1009.

[22] L. Lovász: *Large networks and graph limits*, American Mathematical Society Colloquium Publications, 60, American Mathematical Society, Providence (2012).

[23] B. D. McKay: Independent sets in regular graphs of high girth, *Ars Combinatoria* **23A** (1987), 179–185.

[24] M. Rahman and B. Virág: Local algorithms for independent sets are half-optimal, to appear in *Annals of Probability*, http://arxiv.org/abs/1402.0485.

[25] B. Reed: Paths, stars and the number three, *Combin. Probab. Comput.* **5** (1996), 277–295.

[26] J. B. Shearer: A note on the independence number of triangle-free graphs, II, *Journal of Combinatorial Theory, Series B* **53** (1991), 300–307.

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

[28] N. C. Wormald: The differential equation method for random graph processes and greedy algorithms, in: *Lectures on Approximation and Randomized Algorithms*, M. Karoński and H.J. Prömel (eds), 73–155, PWN, Warsaw, 1999.

[29] N. C. Wormald: Models of random regular graphs, *Surveys in Combinatorics, 1999*, London Mathematical Society Lecture Note Series **267** (J.D. Lamb and D.A. Preece, eds), Cambridge University Press, Cambridge, 239–298, 1999.

[30] N. C. Wormald: Analysis of greedy algorithms on graphs with bounded degrees, *Discrete Mathematics* **273** (2003), 235–260.

[31] N.C. Wormald: Random graphs and asymptotics, Section 8.2 in *Handbook of Graph Theory* (J. L. Gross and J. Yellen eds), 817–836, CRC, Boca Raton, 2004.

[32] M. Zito: Greedy algorithms for minimisation problems in random regular graphs, *Lecture Notes in Computer Science* **2161** 524–536, Springer-Verlag, 2001.

Carlos Hoppen

Instituto de Matemática e Estatística
Universidade Federal
do Rio Grande do Sul
Brazil
choppen@ufrgs.br

Nicholas Wormald

School of Mathematical Sciences
Monash University
Australia
nick.wormald@monash.edu