Analysis of edge deletion processes on faulty random regular graphs

Andreas Goerdt[1] and Mike Molloy[2]

[1] Fakultät für Informatik, Technische Universität Chemnitz
09107 Chemnitz, Germany
e-mail: goerdt@informatik.tu-chemnitz.de
[2] Department of Computer Science, University of Toronto
Toronto, Canada
e-mail: molloy@cs.toronto.edu

Abstract. Random regular graphs are, at least theoretically, popular communication networks. The reason for this is that they combine low (that is constant) degree with good expansion properties crucial for efficient communication and load balancing. When any kind of communication network gets large one is faced with the question of fault tolerance of this network. Here we consider the question: Are the expansion properties of random regular graphs preserved when each edge gets faulty independently of other edges with a given fault probability? We improve previous results on this problem in two respects: First, expansion properties are preserved for much higher fault probabilites than known before. Second, our results apply to random regular graphs of any degree which is at least 4. Previous results apply only to degrees of at least 42. Moreover, the techniques used by us are more elementary than those used in previous work in this area.

Introduction

This paper draws its motivation from the theory of fault tolerance of communication networks. Here a communication network, like the hypercube or the butterfly network simply is a graph where we think of the nodes of this graph as processors of the network and of the edges as communication links. Each edge connects the 2 processors incident with it and allows them to communicate in 1 step with each other. It is a general assumption of the theory that the network proceeds in a synchronous fashion: In each step each processor performs a finite number of computation steps and communication steps with its neigbours. When networks get large we are faced with the question of fault tolerance: We want to use the network in spite of some components being faulty. Ideally the faulty network should be able to simulate efficiently the non-faulty network, that is it should simulate computation of the non-faulty network with running time increasing only by a constant factor, so that the time is essentially unchanged.

Two kinds of faults or fault models are known in the theoretical literature: Worst case faults and random faults which are our concern. The random fault model assumes that each component of the network gets faulty independently of other components with a given fault probability and we get a probability space of faulty networks. Our networks are parameterized with respect to the number of nodes which is usually denoted by n. Therefore the induced probability spaces of faulty networks are also parameterized by n. We are interested in properties which hold with probability tending to 1 when n goes to infinity. In this case we say that a property holds with high probability. Note that this is a standard notion in the theory of random structures.

For the hypercube network efficient simulations are possible with high probability if each node gets faulty with a given constant fault probability, cf. [Le et al. 92]. The degree of each node of the hypercube is $\log n$ and thus unbounded. For bounded degreee networks it is still an open question if efficient simulations under constant fault probabilities are possible. For the butterfly network this question is investigated intensely, see for example [CoMaSi 95].

We consider the situation where the underlying non-faulty network is of bounded degree. In particular we assume that it is itself a random d-regular graph, that is a random member of the set of all graphs with n nodes where each node has constant degree d. The reader unfamiliar with random regular graphs should consult [Bo 85]. Random regular graphs have some importance in the theory of communication networks because they are known to be expander graphs with high probability [Bo 88], this means: There is a constant c with $0 < c < 1$ such that each subset X of nodes has $\geq c \cdot |X|$ neighbours adjacent to X but not belonging to X (provided X contains only at most half of all nodes). Expansion properties are important in the theory of communication networks because they allow for efficient concurrent communication between arbitrary sets of processors. Motivated by the above mentioned open question for bounded degree networks we are interested in the efficient simulation of computation on non-faulty random regular graphs by faulty random regular graphs. The minimum requirement for such a simulation to be possible is that the faulty graph should have a connected component of linear size on which the simulation can take place. But we need more, we need a linear size subgraph which is an expander in order to be able to simulate the efficient communication properties of the non-faulty network.

Following the literature we look at random edge faults of random regular graphs. Note that the structures considered by us are random due to two probabilistic experiments, the choice of the graph and the random faults. A faulty edge is of no use at all and thus can be considered as being deleted. The investigation of random regular graphs with edge faults starts with the paper [Sp et al. 94]. The authors give sufficient conditions for the existence of a linear size component. In the subsequent paper [NiSp 95] attention is drawn to the preservation of expansion properties. Again some sufficient conditions are given in order that the faulty graph has a linear size expanding subgraph. In work by the first author [Go 97] a threshold result for the existence of a linear size component is proved: If each edge is *non*-faulty with probability $> 1/(d-1)$, d being the degree we have a linear size component, if the probability is below this bound we do not have a linear size component. In [Go 98] we give a sufficient condition on fault probability and degree such that we can *find* a linear size expander efficiently – a question not treated in the initial work on expansion [NiSp 95]. Crucial to this paper is the notion of a k-core of a graph: The k-core is the unique maximal subgraph where each node has a degree of k or more. Note that the k-core of a graph can easily be found efficiently by iteratively deleting all nodes of degree strictly less than k. In the paper [Go 98] it is noted first that the 3-core of a faulty random regular graph is an expander with high probability (this follows simply from randomness properties of the 3-core). Second, a probabilistic analysis of a simple edge deletion algorithm finding the 3-core is performed. It is shown that a linear size 3-core is found if $d \geq 42$ and each edge is non-faulty with probability at least $20/d$.

The present paper improves considerably on these results: We give a precise threshold on the fault probability for the existence of a linear size k-core for any d and k where $d > k \geq 3$. This improves the previous bounds for the existence of an expanding subgraph. For example when the degree is as low as $d = 4$ and each edge is faulty with probability $< 1/9$ we have a linear size 3-core and thus an expanding subgraph.

Our proof uses a proof technique originally developed for [MoPi]. It is technically quite simple in that it can be presented without any advanced concepts from probabilistic combinatorics. This is in sharp constrast to the previous proofs of the weaker results mentioned above. Due to its relative simplicity the technique used here applies to a wide range of similar problems, such as analyzing the behaviour of the pure literal rule for random 3-satisfiability instances [BrFrUp 93] and determining the threshold for the appearance of a k-core in a random graph from the common $G_{n,p}$ model, see [MoWo]. The technique is inspired by the original (more involved) proof of the k-core threshold for $G_{n,p}$ given in [PiSpWo 96]. Note that in Computer Science cores are not only interesting from the point of view of communication networks but are of wider relevance. In [Ha et al. 93] cores of random hypergraphs are used to obtain perfect hash functions.

1 Outline

The first problem when dealing with the probability space of regular graphs is the difficulty of calculating probabilities in this space. This difficulty is due to the fact that we do not have a natural probabilistic generation procedure of random regular graphs, like for example in the $G_{n,p}$ model where the generation algorithm throws each possible edge independently with probability p. Therefore random regular graphs are ususally treated with the help of the configuration model, cf. [Bo 85] for a detailed exposition.

Let $n \cdot d$ be even. A d-regular configuration with n classes, $\Phi = (\mathcal{W}, E)$, consists of a set $\mathcal{W} = \{W_1, \ldots, W_n\}$ where the W_i are disjoint d-element sets, the classes, and of a set E which is a partition of $\bigcup \mathcal{W}$ into 2-element sets, called edges. Identifying the d elements of W_i with one single node i each configuration induces a graph with n nodes and possibly multiple edges and loops in a natural manner. Each node of this graph has degree d when counting properly, that is for example loops contribute with 2 to the degree. The elements of $\bigcup W$ are called node copies or just copies.

We denote the set of all d-regular configurations over $\mathcal{W}$ as $Con(n, d)$ and consider it as a probability space endowed with the uniform distribution. There are $(n \cdot d)!! = (n \cdot d - 1) \cdot (n \cdot d - 3) \cdots 5 \cdot 3 \cdot 1$ d-regular configurations altogether. Note that the space of d-regular configurations allows for simple probabilistic generation algorithms as for example: The first edge is generated by first picking a copy deterministically and then picking a uniform random copy from the remaining copies. The algorithm continues in this way until no copy is available any more. We visualize this algorithm as a probability tree branching at the probalistic choices. Each leaf of this tree corresponds to a computation of the algorithm. The probability that the algorithm reaches a given leaf is $1/(dn!!)$ and it is easy to see that each configuration is generated by at most 1 computation. Therefore the algorithm generates each configuration with the required probability.

For $d \cdot n$ even a d-regular graph with n nodes is just a usual graph, that is without multiple edges or loops, where each node has degree d. By a random d-regular graph we mean a graph from the uniform distribution of all such graphs. The relevance of d-regular configurations is that the probability of those configurations which induce a d-regular graph without multiple edges or loops is bounded away from 0 by a constant (decreasing in d, but independent of n). Moreover each d-regular graph is generated by exactly $d!^n$ many configurations. For properties of configurations which can be translated in a natural way to graphs by identifying the node copies of one class with a single node this implies that properties holding with high probability for random d-regular configurations also hold with high probability for random d-regular graphs, cf. [Bo 85] for the details on this. A subconfiguration is a strucuture $\Phi = (\mathcal{V}, F)$ where $\mathcal{V}$ is a set of disjoint d-element sets, the classes, and F is a set of edges, that is disjoint 2-element sets of elements from $\bigcup \mathcal{V}$. Note that the classes of $\mathcal{V}$ still have d node copies each, however, some node copies may be loose in that they do not belong to edges.

The actual probability space of interest here is the space of random d-regular graphs with edge fault probability f. We always let $p = 1 - f$ be the probability that an edge is present. For $n \cdot d$ even this probability space of is determined by the following probabilistic experiment: First, draw a random d-regular graph. Second, delete each edge independently with probability f. We treat faulty graphs by faulty configurations. A random d-regular configuration with edge fault probability $f = 1 - p$ is a is a subconfiguration generated as follows: First, draw a random d-regular configuration. Second, delete each edge with probability f independently. We denote this space with $Con(n, d, p)$. Note that we neither delete node copies nor classes. As before we get that properties holding for faulty configurations with high probability also hold for faulty random regular graphs with high probability. This allows us to restrict attention to $Con(n, d, p)$ in the following. The paper [Go 98] has a more detailed discussion on the relationship between faulty graphs and configurations.

Let $\Phi = (\mathcal{V}, E)$ be a subconfiguration. The degree of the class $W \in \mathcal{V}$ in Φ, $Deg_\Phi(W)$ is the number of copies of W which belong to an edge. Note that loop edges like $\{x, y\}$ with $x, y \in W$ contribute with 2 to the degree of W. For a node copy x the degree of x, $Deg_\Phi(x)$ is the degree of the class to which x belongs. For $k \geq 1$ the k-core of a subconfiguration $\Phi = (\mathcal{V}, E)$ is the maximum subconfiguration $\Psi = (\mathcal{X}, F)$ where $\mathcal{X} \subseteq \mathcal{V}$ and $F \subseteq E$ such that each class of Ψ has degree $\geq k$. Let for $d \geq 1$ and $0 \leq \lambda \leq 1$ $Bin(d, \lambda)$ be a random variable following the binomial distribution with parameters d and λ.

To state our main result we need to make some preparatory remarks. Fix $d > k \geq 3$. Consider the real valued function defined over the reals

$$L(\lambda) = \frac{\lambda}{\sum_{i=k-1}^{d-1} \binom{d-1}{i} \cdot \lambda^i \cdot (1-\lambda)^{d-1-i}}.$$

Dividing by $\lambda > 0$ we see that $L(\lambda) \to +\infty$ for $\lambda \to 0$ as $k - 1 \geq 2$. Further we have $L(1) = 1$ and $L(\lambda) > 0$ for $0 < \lambda \leq 1$ as in this case the denominator is just

$Pr[Bin(d-1,\lambda) \geq k-1]$. As $L(\lambda) \to \infty$ for λ approaching 0 we have that

$$r(d,k) \;=\; \min_{0<\lambda\leq 1} L(\lambda) \tag{1}$$

is > 0, well defined, and ≤ 1 and we are ready to state our theorem.

Theorem 1. *(a) If $p > r(k,d)$ then a random $\Phi \in Con(n,\, d,\, p)$ has a k-core of linear size with high probability.*
(b) If $p < r(k,d)$ then a random $\Phi \in Con(n,\, d,\, p)$ has only the empty k-core with high probability.

The definition of $r(d,k)$ may appear mysterious at this point, but we will see that it has an intuitive combinatorial motivation. As an example note that $r(4,3) = 8/9$. Our proof consists in its major part of a probabilistic analysis of the following algorithm:

Algorithm 2. The Global Algorithm
Input: A subconfiguration Φ.
Output: The k-core of Φ.

while Φ has classes with degree $\leq k-1$ do
  Φ := the modification of Φ where all
    classes with degree $\leq k-1$ are deleted.
od.
Output Φ.

Note the technical point that the algorithm only deletes whole classes not single node copies from classes. Thus we always comply with our notion of subconfiguration. A class survives if it has not been explicitly deleted up to the time considered. When we delete a class we delete the class and the edges incident with it. Thus it canin the course of the algorithm there can be surviving classes of degree 0 belonging to the current Φ. Of course, these classes are deleted with the next execution of the loop. In the context of this and similar algorithms we call classes and node copies of degree $\geq k$ heavy, those of degree $\leq k-1$ light. At the end of the algorithm Φ has only heavy classes. The correctness of this follows with the invariant: The heavy classes of the current Φ are a superset of the classes of the k-core of the input.

In section 2 we analyze a modified version of this algorithm in that the main loop is executed only $\sqrt{\log_d n}$ many times. The point of this reduced number of executions is that it goes only slowly to infinity. First, we show that the expected number of heavy classes left by the modified algorithm is linear in n when $p > r(k,d)$ and the input configuration is from $Con(n,d,p)$. Second, we prove a concentration result for the number of heavy classes to the effect that it is linear in n with high probability with respect to the input space. It is noteworthy that this concentration result is simply proved by Tschebycheff's inequality. Third, we get that the number of light classes left over after $\sqrt{\log_d n}$ executions of the loop is only $o(n)$ with high probability. (Standard arguments show that this number initially is linear in n with high probability.) Fourth, in section 4 we present an additional argument showing that with high probability we get rid of these few light classes leaving us with a linear size k-core provided $p > r(k,d)$. On the other hand, if $p < r(k,d)$ we first show that after $\sqrt{\log_d n}$ rounds of the algorithm the expected number of classes still remaining is $o(n)$. Second, in section 3 we show that we now have no longer enough classes of degree > 0 to form a non-empty k-core.

2 Reduction of the number of light classes

Let for $d \geq t \geq 0$ and $j \geq 0$ the random variable

$$X_{t,j} \,:\, Con(n,\, d,\, p) \,\rightarrow\, \mathcal{N} \tag{2}$$

be the number of surviving classes having degree equal to t after j executions of the loop of the global algorithm. With $X_{W_i} = X_{W_i,t,j}$ being the indicator random variable of the event that the class W_i has degree t after j executions of the loop we can represent $X = X_{t,j}$ as the sum

$$X \,=\, X_{W_1} \,+\, \cdots \,+\, X_{W_n}. \tag{3}$$

(Recall that an indicator random variable assumes the value 1 in case that the event in question holds and 0 otherwise.) For $1 \leq i \leq n$ arbitrary we get by linearity of expectation and as $Pr[X_{W_i} = 1]$ is independent of i that

$$EX \,=\, n \cdot E[X_{W_i}] \,=\, n \,\cdot\, Pr[X_{W_i} = 1].$$

We will determine $Pr[X_{W_i} = 1]$ approximately, that is up to an additive error of $o(1)$ for $j = \sqrt{\log_d n}$ and any t with $0 \leq t \leq d$. The probability of the event: the class $W = W_i$ has degree equal to t after j executions of the loop, turns out to depend only on the $j+1$-neighbourhood of W. The notion of a neighbourhood is slightly different from the usual one:

Definition 3. Let $\Phi = (\mathcal{V}, E)$ be a subconfiguration, $j \geq 0$, and $W \in \mathcal{V}$. The j-neighbourhood of W in Φ, denoted as $\mathcal{N}_\Phi(W,j)$, is that subconfiguration of Φ which has as classes the classes U whose distance from W is $\leq j$. That is the length (= number of edges) of a shortest path from W to U is $\leq j$. The edges of $\mathcal{N}_\Phi(W,j)$ are those induced from Φ as long as they are incident with at least one class in distance j.

This notion in slightly different form the usual one in that we do not include edges between node copies which are both in distance j. For $j = j(n)$ going sufficiently slowly to infinity the structure of the $j+1$-neighbourhood of a fixed class is a tree with high probability. Note that this does not mean that the neighbourhood of any class is a tree.

Lemma 4. *Let W be a fixed class and let $j = j(n) = \sqrt{\log_d n}$, then with high probability the $j+1$-neighbourhood of W in a random fault free configuration from $Con(n,d)$ is a tree.*

Proof. We consider the following probabilistic algorithm to generate the $j+1$-neighbourhood of W in a random fault free configuration Φ. The algorithm is a breadth first generation process, and we describe it as a sequence of classes and edges. We let $S_0 = \{W\}$ be the initial front of the algorithm. The first edge e_1 is generated probabilistically: Pick a copy w from W according to some deterministic rule and expand it. That means, choose from the set of remaining node copies (inclusive $W \setminus \{w\}$) a uniform random copy v, set $e_1 = \{w, v\}$. Pick the next copy from W deterministically and form the edge e_2 in an analogous way. Continue in this way until all copies in W belong to an edge. Let $V_1, \dots, V_l$ be the classes now reachable from W but not equal to W. Set $S_1 = \{V_1, \dots, V_l\}$ and continue with S_1 in the same manner. The algorithm stops when all classes in distance j of W are expanded. The algorithm makes at most $j + 1$ loops of expanding the current front.

Now given a $j+1$-neighbourhood $\Gamma = (\mathcal{V}, E)$ of W we observe that there is exactly one possible computation of the algorithm generating Γ. If $|E| = m$ the probability for the algorithm to generate Γ is equal to

$$\frac{1}{dn-1} \cdot \frac{1}{dn-3} \cdot \dots \cdot \frac{1}{dn-2m+1}$$

which is also the probability that a random configuration fron $Con(n, d)$ has the edges from E and thus the $j + 1$-neighbourhood Γ of W. The number of nodes in classes which are incident with edges generated by the algorithm in the end is bounded above by

$$d \cdot \sum_{i=0}^{j+1} (d-1)^i \ \leq d^{j+3})$$

using the geormetric series. The number of probabilistic steps made sequentially by the algorithm is therefore certainly bounded above by this number. The algorithm generates no tree if at one point it randomly picks a node copy from a class already reachable from W. This probability is certainly bounded above by $d^{(j+3)\cdot 2}/(dn - d^{(j+3)} = o(1)$, as can again be easily seen with the help of the geometric series. □

We restrict attention to the class W of a subconfiguration Φ and execute the loop of the global algorithm only once. We see that effect on the class W is the same regardless whether we use Φ or $\mathcal{N}_\Phi(W, 2)$ as input to the algorithm. This holds in general.

Lemma 5. *Let $j \geq 0$. For any subconfiguration Φ and any class W of Φ holds: W survives with degree $t \geq 0$ after j executions of the loop of the global algorithm with Φ as input iff W survives j loops with degree t and $\mathcal{N}_\Phi(W, j + 1)$ as input.*

Proof. The algorithm has one of two possible effects on W: W is deleted or W survives with degree $t \geq 0$. We proceed by induction on j. For $j = 0$ the lemma holds because after 0 executions of the loop W survives with its initial degree with either input.

Now let $j > 0$ and assume W survives j executions of the loop with degree $t \geq 0$ when given the $j + 1$-neighbourhood as input. Then after $j - 1$ executions W has at least k neighbours, t of which are heavy. By induction hypothesis we get that precisely these neighbours of W survive $j - 1$ executions with the same degrees as before when using the j-neighbourhoods of these neighbours as input. Again by induction hypothesis (now the other direction of the equivalence) we get that precisely these neighbours survive $j - 1$ executions of the loop with the same degrees as before when we take the whole configuration as input. Therefore W survives with degree t when the whole configuration is the input.

Now assume that W survives j executions of the loop with degree t when we take the whole configuration as input. After $j - 1$ executions W has again at least k neighbours t of which are heavy. By induction hypothesis we have that precisely these neighbours survive $j-1$ executions with the same degrees when using their j-neighbourhoods as input. By induction hypothesis (now the other direction of the equivalence) precisely these neighbours survive $j-1$ loops with the same degrees when using the $j+1$-neighbourhood of W as input. Hence W survives j rounds with degree t when the $j+1$-neighbourhood is the input. $\square$

With respect to the fixed class W the following local algorithm has essentially the same effect as j executions of the global loop have, provided the $j+1$-neighbourhood of W is a tree.

Algorithm 6. The Local Algorithm.
Input: A $j+1$-neighbourhood of the class W, Γ, which is a tree. We let W be the root of this tree. The depth of this tree is $\leq j+1$.
for $i = j$ downto 0 do
Modify Γ by deleting all light classes in depth i.

Note that W in the end either is heavy or is deleted. The first execution of the loop of the algorithm deals with classes in depth j the second with classes in depth $j-1$ ... the $j+1$'th execution with W. The next lemma clarifies the exact relationship of the global and the local algorithm.

Lemma 7. *Let Φ be a subconfiguration such that the $j+1$-neighbourhood of the class W from Φ is a tree, then we have:*
After j executions of the loop of the global algorithm with input Φ the class W survives with degree $t \geq k$.
$\Leftrightarrow$
After running the local algorithm with $\mathcal{N}_\Phi(W, j+1)$ the class W survives with degree $t \geq k$.

Note that it can happen that W is deleted by the local algorithm but not deleted after j loops of the global one. This is of course due to the fact that the local algorithm executes its loop $j+1$-times. The lemma tells us that in this case W is light after j global loops and therefore is deleted by the next loop.

Proof. With Lemma 5 it is sufficient to show: W survives j executions of the global loop applied to the $\mathcal{N}_\Phi(W, j+1)$with degree $t \geq k \Leftrightarrow W$ survives the local algorithm with input $\mathcal{N}_\Phi(W, j+1)$ with degree $t \geq k$.

For the subsequent inductive argument we need the following statement: Let Γ and Γ' be two possible $j+1$-neighbourhoods of W which are trees and satisfy the following condition: Γ is obtained from Γ' by deleting edges which are incident with a light class in depth $\leq j$. (Note, not in depth $j+1$ as any class in depth $j+1$ has degree 1 and thus is light.) Then holds that the application of the local algorithm with input Γ and Γ' has the same effect for the class W. This is intuitively true because for $1 \leq l \leq j+1$ light classes in depth $j+1-l$ are deleted by the l'th execution of the local loop anyway. Nevertheless we present the induction: Let $j = 0$, then Γ is made from Γ' by deleting edges incident with W provided W is light in Γ'. In both cases W is deleted by the first execution of the loop of the local algorithm. For $j = 1$ Γ is obtained from Γ' by deleting edges which are in Γ' incident with light classes in depth ≤ 1. After the execution of the first loop of the local algorithm we get the 1-neighbourhoods Γ_1 and Γ_1' of W. All light classes in depth 1 are deleted. Accordingly Γ_1 and Γ_1' satisfy the condition above for $j = 0$ and the result holds. Now for $j > 2$ and with Γ and Γ' being $j+1$-neighbourhoods as described we get: After the execution of the first loop of the local algorithm with Γ and Γ' we get two j-neighbourhoods of W to which the induction hypothesis is applicable and the result follows. Note, that our neighbourhoods are trees, otherwise the last argument is not clear and possibly not true because Γ might have more classes in distance j due to edges being deleted.

Now, the claim of the lemma inductively on $j \geq 0$. If $j = 0$ and W is heavy in its 1-neighbourhood the degree of W is not changed in both cases. If W is light W gets deleted by the local algorithm and is not touched by the global one. The claim holds.

For the induction step let $j > 0$. After the first execution of the global loop and the local loop respectively we get two j-neighbourhoods of W to which the hypothesis of the preceding statement applies. We apply the induction hypothesis to the neighbourhood generated by the global algorithm and the claim follows recalling Lemma 5. □

Let Φ be a fault free configuration and let $j \geq 0$ such that the $j+1$-neighbourhood of the class W in Φ is a tree. This tree is of course the complete tree of depth $j+1$ where the root has d sons and the interior nodes have $d-1$ sons. Let Γ be the random $j+1$-neighbourhood of W when considering faulty versions with fault probability $f = 1-p$. We now turn our attention to the survival probability of W when considering the local algorithm applied to a random Γ.

We define $\phi_0 = 1$ and $\phi_{i+1} = Pr[Bin(d-1, \phi_i p) \geq k-1]$ for any $i \geq 0$. We see that the ϕ_i are monotously decreasing and all in the intervall $[0,1]$. Therefore

$$\phi = \phi(p) = \lim_{i \to \infty} \phi_i \tag{4}$$

is well defined. As all functions involved are continuous we get:
$\phi = Pr[Bin(d-1, p\phi) \geq k-1]$. Note that this is no definition of ϕ. The equation is always satisfied by $\phi = 0$.

$\phi_0 = 1$ is the probability that a class in depth $j+1$ of Γ survives the local algorithm because the local algorithm does not delete classes in depth $j+1$. The probability that a class in depth j survives is equal to $\phi_1 = Pr[Bin(d-1,p) \geq k-1]$ because a class survives iff it has at least $k-1$ surviving sons. Assume by induction that the probability that a class in depth $j+1-i \geq 2$ survives is ϕ_i and consider a class V in depth $j+1-(i+1) \geq 1$ in a random Γ. Again V survives iff it has $\geq k-1$ sons surviving the previous loop of the algorithm, which means that V has $l \geq k-1$ sons in Γ to begin with of which $\geq k-1$ survive the i'th loop of the algorithm. By independence and conditioning on the number of sons of V in Γ, l, we see that the probability that V has exactly h surviving sons is

$$\sum_{l=k-1}^{d} Pr[Bin(l, \phi_i) = h] \cdot Pr[Bin(d-1,p) \leq l] = Pr[Bin(d-1, p\phi_i) = h].$$

To show this the binomial identity

$$\binom{l}{h} \cdot \binom{d}{l} = \binom{d-h}{l-h} \cdot \binom{d}{h}$$

and the binomial theorem is required. One could also argue as follows: The probability that a son of V survives, given it is present in Γ, is ϕ_i by induction. The probability that the son is present to begin with is p. As the required events for the different sons are independent this implies that the distribution of the number of sons of V after the i'th execution of the loop of the local algorithm is as that of the binomial random variable $Bin(d-1, \phi_i p)$. We get that the survival probability of V is $\phi_{i+1} = Pr[Bin(d-1, p\phi_i) \geq k-1]$. The probability that the root W survives is slightly different: $Pr[Bin(d, \phi_j p) \geq k]$.

For $t \geq k$, $i \geq 0$ the λ_t's are the probability that the root of a tree survives with t sons the local algorithm with input the $i+1$-neighbourhood, therefore we set

$$\lambda_{t,i} = \lambda_{t,i}(p) = Pr[Bin(d, p\phi_i) \geq k]$$

The probability that W as above survives with t sons is $\lambda_{t,j}$. Again the $\lambda_{t,i}$ are monotonously decreasing in i and between 0 and 1 and

$$\lambda_t = \lim_{i \to \infty} \lambda_{t,i} \tag{5}$$

exists.

Now let $j = j(n)$ go to infinity and for each n let W be the root of the non-faulty tree-neighbourhood of depth $j+1$ and consider the family of probability spaces of faulty versions of these trees. Then the probability that W survives with degree t the local algorithm applied to the faulty neighbourhood is

$$\lambda_{t,j} = \lambda_t + o(1) \tag{6}$$

as j goes to infinity with n. Here is where our formula for $r(d,k)$ comes from, recall (1 ,4).

Lemma 8. *$\phi > 0$ iff $p \geq r(d,k)$.*

Proof. For ϕ we know that $\phi = Pr[Bin(d-1, p\phi) \geq k-1]$. If $\phi > 0$ then so is $Pr[Bin(d-1, p\phi) \geq k-1]$ and p. Therefore, setting $\lambda = p\phi$, we get

$$\frac{\lambda}{p} = Pr[Bin(d-1, \lambda) \geq k-1],$$

and so

$$p = \frac{\lambda}{Pr[Bin(d-1,\lambda) \geq k-1]} = \frac{\lambda}{\sum_{i=k-1}^{d-1} \binom{d-1}{i} \lambda^i (1-\lambda)^{d-1-i}}.$$

This establishes that if $\phi > 0$ then $p \geq r(d,k)$.

Now assume $p \geq r(d,k)$. Let λ_0 be such that

$$r(d,\, k) \;=\; \frac{\lambda_0}{\sum_{i=k-1}^{d-1} \binom{d-1}{i} \cdot \lambda_0^i \cdot (1 \,-\, \lambda_0)^{d-1-i}}.$$

We show by induction on i that $p \cdot \phi_i \geq \lambda_0$. For the induction base we get:

$$p \cdot \phi_0 = p \geq r(d, k) \geq \lambda_0$$

where the last estimate holds because the denominator in the definition of $r(d, k)$ always is ≤ 1. For the induction step we get:

$$p \cdot \phi_{i+1} = p \cdot Pr[Bin(d-1, p \cdot \phi_i) \geq k-1] \geq p \cdot Pr[Bin(d-1, \lambda_0) \geq k-1] \geq \lambda_0$$

where the last but one estimate uses the induction hypothesis and the last one follows from the assumption $p \geq r(d, k)$. $\square$

We now return to the analysis of the global algorithm. The next corollary follows directly with Lemma 4, Lemma 5, and (6).

Corollary 9. *Let W be a fixed class, $t \geq k$ and let $j = j(n) = \sqrt{\log_d n}$. In the space of faulty configurations we have: The probability that W survives j executions of the loop of the global algorithm with degree t is*

$$\lambda_{t,j} + o(1) = \lambda_t + o(1).$$

Next the announced concentration result:

Theorem 10. *Let $t \geq k$, $j = \sqrt{\log_d n}$, $X = X_t = X_{t,j}$ be the random variable defined as in (2), let $\lambda = \lambda_t$ then we have:*
(1) $EX = \lambda \cdot n + o(n)$.
(2) $|X - \lambda \cdot n| \leq o(n)$ with high probability.

Proof. (1) The claim follows with the representation of X as sum of indicator random variables, see(3), and with Corollary 9.
(2) We show further below that the variance of X, $VX = o(n^2)$. This implies the claim as follows:
Tschebycheff's inequality reads : $Pr[|X - EX| \geq a] \leq VX/a^2$ for $a > 0$. Putting $\sigma = \sqrt{VX}$ we get that

$$Pr[|X - EX| \geq a \cdot \sigma] \leq \frac{1}{a^2}.$$

We have $\sigma = \sigma(n) = o(n)$ as $VX = o(n^2)$. Hence we have a function $\omega(n) \to \infty$ with $\sigma = \sigma(n) = n/\omega(n)$. With $a = a(n) = \sqrt{\omega(n)}$ we have $\sigma \cdot a(n) = o(n)$ and

$$Pr[|X - EX| \geq a \cdot \sigma] \leq \frac{1}{a^2} = o(1).$$

This implies the claim as EX differs only by an additive term of $o(n)$ from λn.

We calculate the variance of X. We decompose $X = X_{W_1} + \cdots X_{W_n}$ see (3). This and (1) of the present theorem implies

$$VX = E[X^2] - (EX)^2 = E[X^2] - ((\lambda n)^2 + o(n) \cdot n).$$

Moreover,

$$E[X^2] = EX + n \cdot (n-1) \cdot E[X_U \cdot X_W] = \lambda \cdot n + o(n) + n \cdot (n-1) \cdot E[X_U \cdot X_W].$$

Here U and W are two arbitrary distinct indices of classes. We need to show that $E[X^2] = (\lambda \cdot n)^2 + o(n^2)$. This follows from $E[X_U \cdot X_W] = \lambda^2 + o(1)$.

Let $Distrees$ be the event that the $j+1$-neighbourhoods of U and W in the underlying fault free configurations are disjoint trees. We get

$$\begin{aligned}
&E[X_U \cdot X_W] = Pr[X_U \cdot X_W = 1] \\
&= Pr[X_U \wedge X_W \wedge Distrees] + Pr[X_U \wedge X_W \wedge \neg Distrees] \\
&= (\lambda + o(1))^2 \cdot Pr[Distrees] + Pr[X_U = 1 \wedge X_W = 1 | \neg Distrees] \cdot Pr[\neg Distrees],
\end{aligned}$$

where the last equation holds by conditioning and Corollary 9 as the survival probabilities are independent when the neighbourhoods of U and W are disjoint trees.

It remains to show that $Pr[\neg Distrees] = o(1)$. To this end we define two additional events, let $Notdis$ denote the event that the $j+1$-neighbourhoods of U, W a random fault free configuration are not disjoint and let $Trees$ be the event that both neighbourhoods are trees, which are not necessarily disjoint. We have by conditioning and Lemma 4 that

$$Pr[\neg Distrees] \leq Pr[\neg Trees] + Pr[Notdis | Trees] \leq Pr[Notdis | Trees] + o(1).$$

We condition the following calculation on the event $Trees$:

$$Pr[Notdis] = Pr[Notdis \wedge U \in \mathcal{N}_\Phi(W, j+1)] + Pr[Notdis \wedge U \notin \mathcal{N}_\Phi(W, j+1)]$$

where the probability refers to a random fault free configuration Φ. The first summand is easily seen to be $o(1)$ by analyzing the breadth first generation process starting with W as in the proof of Lemma 4:

$$Pr[U \in \mathcal{N}_\Phi(W, j+1)] \leq d \cdot \frac{d^{j+3}}{dn - \cdot d^{j+3}} = o(1).$$

The second summand is slightly more complex to deal with. Let $\Gamma = \mathcal{N}_\Phi(W, j+1)$. Then Γ has $\leq d^{j+3}$ node copies altogether. Given the $j+1$-neighoburhood Γ of W we generate the $j+1$-neighbourhood of U by a subsequent breadth first generation procedure starting with U. The number of copies not any more available for random choices in the end is $\leq 2 \cdot d^{j+3}$. To generate the $j+1$-neighbourhood of U we have to make $\leq d^{j+3}$ copy expansion steps. Accordingly:

$$\begin{aligned}
&Pr[Notdis \wedge U \notin \mathcal{N}_\Phi(W, j+1)] \\
&= Pr[\text{ When generating } \mathcal{N}_\Phi(U, j+1) \text{ we hit a copy from } \mathcal{N}_\Phi(W, j+1).] \\
&\leq \frac{d^{(j+3)2}}{dn - 2d^{j+3}} = o(1)
\end{aligned}$$

and we are done. □

3 When there is no k-core

The proof of Theorem 1(b) is now quite simple, let $p < r(d, k)$ for the rest of this section.

First we need the following fact showing that very small k-cores are not possible.

Lemma 11. *A random member of* $Con(n, d)$ *has no* k*-core on* $o(n)$ *classes with high probability.*

Proof. The lemma follows from the fact that with high probability a random member of $\mathrm{Con}(n, d)$ has no subconfiguration with average degree at least 3 on at most ϵn classes, where $\epsilon = \epsilon(d)$ is a small positive constant. Consider any $s \leq \epsilon n$. The number of choices for s classes, $1.5s$ edges from amongst those classes, and copies for the endpoints of each edge, is at most:

$$\binom{n}{s}\binom{\binom{s}{2}}{1.5s}d^{3s}.$$

Setting for even t $M(t) = t!! = t!/(2^{t/2}(t/2)!)$ to be the number of ways of pairing t copies, we have that for any such collection of $1.5s$ edges, the probability that those edges lie in our random member of $\mathrm{Con}(n, d)$ is

$$M((dn - 3s)/M(dn) < \left(\frac{e}{n}\right)^{1.5s}.$$

Therefore, the expected number of such subconfigurations is at most:

$$\begin{aligned}\binom{n}{s}\binom{\binom{s}{2}}{1.5s}d^{3s}\left(\frac{e}{n}\right)^{1.5s} &< \left(\frac{en}{s}\right)^{s}\left(\frac{e(s^2/2)}{1.5s}\right)^{1.5s}d^{3s}\left(\frac{e}{n}\right)^{1.5s} \\ &\leq \left(\frac{20d^6 s}{n}\right)^{0.5s} = f(s).\end{aligned}$$

Therefore, if $\epsilon = 1/40d^6$ then the expected number of such subconfigurations is less than $\sum_{s=2}^{n} f(s)$ which is easily verified to be $o(1)$ using the geometric series. □

Now, by Lemma 8 we have that ϕ cannot be strictly greater than 0 and therefore must be equal to 0 by definition (4). Therefore, since $\lim_{j\to\infty} \phi_j = \phi = 0$ and as $j(n) = \sqrt{\log_d n}$ grows with n, we have that $\lim_{n\to\infty} \phi_{j(n)} = 0$. Therefore, the expected number of classes which survive the first $j(n)$ executions of the global loop with degree $\geq k$ is $o(n)$. This implies by Lemma 10 that with high probability this number is $o(n)$. Therefore a possible k-core can only have $o(n)$ classes. But with high probability by Lemma 11 the surviving subconfiguration cannot have k-core any more and so the original faulty configuration has no k-core.

This establishes Theorem 1(b).

4 When there is a k-core

In this section, we prove Theorem 1(a). So we assume that $p > r(d, k)$. We start by showing that with high probability very few light clauses survive the first $j(n) = \sqrt{\log_d n}$ executions of the gobal loop.

Corollary 12. *In Con(n, d, p) we have with high probability: The number of light classes after $j(n) = \sqrt{\log_d n}$ executions of the loop of the global algorithm is reduced to $o(n)$.*

Proof. Let $\psi = \lambda_k + \cdots + \lambda_d$, see (5). Then $\psi + o(1)$ is the probability that a given class W survives $j(n)$ executions of the loop of the global algorithm with degree $\geq k$ (Corollary 9). The probability that a given class survives $j(n) - 1$ executions of the loop of the global algorithm with degree $\geq k$ is $\psi + o(1)$, too (with a different $o(1)$, note $j(n) - 1$ goes to infinity, too). Let H_i be random variable giving the number of classes which are heavy after the i'th execction of the loop of the global algorithm. Let L_i the number of classes which become light during the i'th loop . Then we have $H_j = H_{j-1} - L_j$. From Theorem 10 we know that for $t \geq k$ X_t differs from $\lambda_t \cdot n$ by an additive error of at most $o(n)$ with high probability. Hence we get that H_j differs from $\psi \cdot n$ by an additive error of $o(n)$ with high probability, too. The same applies to H_{j-1} with a different error of $o(n)$. Hence L_j the number of classes becoming light in the j'th loop of the global algorithm must be $o(n)$ with high probability. □

In order to show that the light classes still present after $j(n)$ executions of the global loop can be eliminated leaving us with a linear size k-core, we need to know something about the distribution of the subconfigurations after $j(n)$ loops. As usual in similar situations the uniform distribution needs to be preserved. For $\bar{n} = (n_0, n_1, n_2, \ldots, n_d)$ we let Con$(\bar{n})$ be the space of all subconfigurations with n_i classes of degree i. Each configuration is equally likely.

Lemma 13. *Conditioning the space $Con(n, d, p)$ on those subconfigurations which give a subconfiguration in $Con(\bar{n})$ after i loops of the global algorithm each subconfiguration from $Con(\bar{n})$ has the same probability to occur.*

A proof of this lemma can be found in [Go 98]. After running the global algorithm for $j(n) = \sqrt{\log n}$ loops we get a subconfiguration uniformly distributed in a space $Con(\bar{n})$ where $n_0 + n_1 + n_2 + ... + n_{k-1} = o(n)$ and $|n_t - \lambda_t \cdot n| \leq o(n)$ for $t \geq k$ with high probability. A probabilistic analysis of the following algorithm eliminating the light classes one by one shows that we obtain a linear size k-core with high probability.

Algorithm 14.
Input: A subconfiguration Φ.
Output: The k-core of Φ.
while There exist light classes in Φ do
 Choose uniformly at random a light class W
 from all light classes and delete W.
od
The surviving classes are of degree $\geq k$ and make up the k-core of Φ.

In order to perform a probabilistic analysis of this algorithm it is again important that the uniform distribution is preserved. A similar result is Proposition 1 in [PiSpWo 96] (for the case of graphs instead of configurations).

Lemma 15. *If we apply the algorithm above to a random $\Phi \in Con(\bar{n})$ for a given number of iterations we get: Conditional on the event in $Con(\bar{n})$ that the subconfiguration obtained is in $Con(n'_0, n'_1\, n'_2,\ n'_3, \ldots, n'_d)$ this subconfiguration is a uniform random configuration from this space.*

Lemma 16. *We consider probability spaces $Con(\bar{n})$ where the number of heavy classes is $\geq \delta \cdot n$ for a given δ. One execution of the loop of Algorithm 14 causes one light class to disappear and we get $\leq k-1$ new light classes. Let $Y : Con(\bar{n}) \to \mathcal{N}$ be the number of new light classes after one loop of Algorithm 14. Let $\nu = \sum_i i \cdot n_i$ and $\pi = (k \cdot n_k)/\nu$. Thus π is the probability to pick a copy of degree k when picking uniformly at random from all copies incident with an edge. Then:*
(a) $Pr[Y = l] \leq Pr[Bin(k-1, \pi) = l] + o(1)$.
(b) $EY \leq (k-1) \cdot \pi + o(1)$.

Proof. The probabilities in question can easily be calculated by analyzing the following two step procedure to generate configurations from $\mathrm{Con}(\bar{n})$ according to the uniform distribution. 1. Choose copies uniformly at random from the classes according to the degree constraints prescribed in $\bar{n}$. 2. Put the copies chosen in 1. into $\sum i \cdot n_i$ slots a follows: Pick the first copy deterministically and the second at random from the rest. Pick the next copy deterministically and the fourth copy randomly,... The pair of copies in slot $2 \cdot m - 1$ and $2 \cdot m$ represents one edge. The required probabilities are now easily calculated:

We condition on the event that W is the light class chosen by the algorithm and that its degree is $h \leq k-1$. Then Y can assume the values $0, \ldots, h$. We consider the generation procedure above where the copies picked deterministically are always those from W whereas the adjacent copies are randomly chosen from those copies which do not yet belong to edges. We only expand W. First observe that the probability that two copies of W are adjacent to two copies from the same class V is $o(1)$ because the number of all copies available is $\geq \delta \cdot n$. Also the probability that a copy from W is adjacent with a copy from W is $o(1)$. Thus the number of new light classes is dominated by the number of times we hit a copy from a class of degree k when expanding W. Now again as the number of copies altogether is linear in n this number follows the binomial dstribution with parameters h and π. □

Our next step is to bound the π form the preceding lemma.

Lemma 17. $\pi \leq \frac{1-\epsilon}{k-1}$ *for some* $\epsilon > 0$.

Proof. We will prove that when $p = r(d,k)$ then $\pi = \frac{1}{k-1}$. Since π is easily shown to be decreasing in p, this proves our lemma. Recall (1) that $r(d,k)$ is defined to be the minimum of a function $L(\lambda)$. Therefore, with $L(\lambda) = r(d,k)$, we have $L^{'}(\lambda) = 0$. Differentiating L, we get:

$$\sum_{i=k-1}^{d-1} \binom{d-1}{i} \lambda^i (1-\lambda)^{d-1-i} = \sum_{i=k-1}^{d-1} \binom{d-1}{i} \lambda^i (1-\lambda)^{d-2-i} (i - (d-1)\lambda)$$
$$= (k-1) \binom{d-1}{k-1} \lambda^{k-1} (1-\lambda)^{d-k}$$

(The second identity can be easily confirmed using Maple.)

Setting $j = i+1$, and multiplying by λd, the LHS comes to:

$$\sum_{j=k}^{d} d \binom{d-1}{j-1} \lambda^j (1-\lambda)^{d-j} = \sum_{j=k}^{d} j \binom{d}{j} \lambda^j (1-\lambda)^{d-j},$$

and the RHS comes to

$$d(k-1)\binom{d-1}{k-1}\lambda^k(1-\lambda)^{d-k} = k(k-1)\binom{d}{k}\lambda^{k-1}(1-\lambda)^{d-k}.$$

Now, since

$$\pi = \frac{k\binom{d}{k}\lambda^{k-1}(1-\lambda)^{d-k}}{\sum_{j=k}^{d} j\binom{d}{j}\lambda^j(1-\lambda)^{d-j}} + o(1),$$

this establishes our lemma. □

Theorem 18. *Algorithm 14 stops after $o(n)$ executions of the while loop with a linear size k-core with high probability (with respect to $Con(\bar{n})$).*

Proof. We define Y_i to be the number of light classes remaining after i loops of Algorithm 14. By assumption, $Y_0 = o(n)$. Furthermore, by Lemmas 16 and 17, we have $EY_1 \leq Y_0 - 1 + (k-1)\pi < Y_0 - \epsilon$. Furthermore, it is not hard to verify that, since there are $\Theta(n)$ classes of degree k, then so long as $i = o(n)$ we have

$$EY_{i+1} \leq Y_i - \frac{1}{2}\epsilon,$$

and in particular, the probability that at least ℓ new light vertices are formed during step i is less than the probability that the binomial variable $Bin(k-1, \pi)$ is at least ℓ.

Therefore, for any $t = o(n)$, $Y_0, Y_1, ..., Y_t$ is statistically dominated by a random walk defined as:

$$Z_0 = Y_0; Z_{i+1} = Z_i - 1 + Bin(k-1, \frac{1-\frac{1}{2}\epsilon}{k-1}),$$

where $Bin(d, p)$ simply stands for a random variable which follows the binomial distribution with parameters n and p. Since Z_i has a drift of $-\frac{1}{2}\epsilon$, it is easy to verify that with high probability, $Z_t = 0$ for some $t = o(n)$, and thus with high probability $Y_t = 0$ as well.

If $Y_t = 0$ then we are left with a k-core of linear size. □

Clearly Theorem 14 implies Theorem 1(a).

References

[Bo 85] Bela Bollobas. Random Graphs. Academic Press. 1985.

[Bo 88] –.The isoperimetric number of random regular graphs. European Journal of Combinatorics. 1988, 9, 241-244.

[BrFrUp 93] Andrei Z. Broder, Allan M. Frieze, Eli Upfal. On the satisfiability and maximum satisfiability of random 3-CNF formulas. In Proceedings of the 4th SoDA, 1993, ACM-SIAM, 322-330.

[CoMaSi 95] Richard Cole, Bruce Maggs, Ramesh Sitaraman. Routing on Butterfly networks with random faults. In Proceedings of FoCS 1995, IEEE, 558-570.

[Go 97] Andreas Goerdt. The giant component threshold for random regular graphs with edge faults. In Proceedings of MFCS 1997, LNCS 1295, 277-288. Theoretical Computer Science, to appear.

[Go 98] –. Random regular graphs with edge faults: expansion through cores. In Proceedings ISAAC 1998. LNCS 1533, 219-228. Theoretical Computer Science, to appear.

[Ha et al. 93] G. Havas, B. S. Majewski, N. C. Wormald, C. Z. Czech. Graphs, hypergraphs and hashing. In Proceedings 19th International Workshop on Graph Theoretical Concepts in Computer Science (WG 93), 1993, LNCS 790, Springer Verlag, 153-165.

[Le et al. 92] Tom Leighton, Bruce Maggs, Ramesh Sitaraman. On the fault tolerance of some popular bounded-degree networks. In Proceedings of FoCS 1992, IEEE, 542-552.

[PiSpWo 96] Boris Pittel, Joel Spencer, Nicholas Wormald. Sudden emergence of a giant k-core in a random graph. Journal of Combinatorial Theory B 67,1996,111-151.

[MoPi] Mike Molloy and Boris Pittel. Subgraphs with average degree 3 in a random graph. In preparation.

[MoWo] Mike Molloy and Nick Wormald.In preparation.

[Sp et al. 94] Sotiris Nikoletseas, K. Palem, Paul Spirakis, Moti Yung. Short vertex disjoint paths and multiconnectivity in random graphs: secure network computing. In Proceedings of ICALP 1994, LNCS 820, 508-519

[NiSp 95] Sotiris Nikoletseas, Paul Spirakis. Expander properties of random regular graphs with edge faults. In Proceedings STACS 1995, LNCS 900,421-432.