Avoiding a Giant Component II

Tom Bohman* and Alan Frieze†

Department of Mathematical Sciences,
Carnegie Mellon University
Pittsburgh PA 15213

Nicholas C. Wormald‡
Department of Mathematics and Statistics,
University of Melbourne
VIC 3010,
Australia.

June 23, 2003

Abstract

Let $e_1, e_2, \ldots$ be a sequence of edges chosen uniformly at random from the edge set of the complete graph K_n (i.e. we sample with replacement).

Our goal is to choose, for m as large as possible, a subset $E \subseteq \{e_1, e_2, \ldots, e_{2m}\}$, $|E| = m$, such that the size of the largest component in $G = ([n], E)$ is $o(n)$ (i.e. G does not contain a giant component). Furthermore, the selection process must take place *on-line*; that is, we must choose to accept or reject an e_i based on the previously seen edges $e_1, \ldots, e_{i-1}$.

We describe an on-line algorithm that succeeds **whp**[1] for $m = .9668n$. Furthermore, we find a tight threshold for the off-line version of this question; that is, we find the threshold for the existence of m out of $2m$ random edges without a giant component. This threshold is $m = c^*n$ where c^* satisfies a certain transcendental equation, $c^* \in [.9792, .9793]$. We also establish new upper bounds for more restricted *Achlioptas* processes.

1 Introduction

Let $e_1, e_2, \ldots$ be a sequence of edges chosen uniformly at random from the edge set of the complete graph K_n (i.e. we sample with replacement). We discuss an *on-line* algorithm which for some integer m, chooses m edges out of $\{e_1, e_2, \ldots, e_{2m}\}$, such that **whp** there is no "giant component" i.e. component of size $\Omega(n)$. (The latter

*Supported in part by NSF grant DMS-0100400. e-mail tbohman@andrew.cmu.edu

†Supported in part by NSF grant CCR-9818411. e-mail alan@random.math.cmu.edu

‡Research supported in part by the Australian Research Council and in part by Carnegie Mellon University Funds. e-mail nick@ms.unimelb.edu.au

[1] A sequence of events $\mathcal{E}_n$ is said to occur with high probability (**whp**) if $\lim_{n\to\infty} \mathbf{Pr}(\mathcal{E}_n) = 1$

denotes a function bounded below by a positive constant times n, for n sufficiently large.) We endeavor in this to make m as large as possible and achieve $m = .9668n$ rigorously, as well as a bound $m = .9760n$ which is nonrigorous only because of possible errors in floating point computations in solving a system of thousands of differential equations. We will also show that this is close to an upper bound for this type of process. The reader will recall the classic result of Erdős and Rényi [3] that selecting m random edges with no rejections, m can be at most $(.5 + o(1))n$ before the giant component appears.

This can be seen as a development along the lines of a problem posed by Achlioptas. Let $e_1, e'_1; e_2, e'_2; \ldots; e_i, e'_i; \ldots$ be a sequence of ordered pairs of edges chosen uniformly at random from the edge set of the complete graph K_n. This sequence is used to form a graph by choosing at stage i, $i = 1, 2, \ldots$, one edge from e_i, e'_i to be an edge in the graph, where the choice at stage i is based only on the observation of the edges that have appeared by stage i. It was shown in Bohman and Frieze [1] that **whp** at least $.545n$ edges could be chosen in this way without constructing a component of size more than $(\ln n)^A$ for some constant $A > 0$. This answered a question posed by Achlioptas as to whether or not such an on-line procedure existed, in which more than cn edges could be included **whp**, for some $c > 0.5$. We will refer to a process which makes the on-line choice of one edge from each presented pair as an *Achlioptas process*.

In both the problems discussed in this paper and the problem posed by Achlioptas, the number of edges chosen is equal to half the number of edges seen. But in the current setting there is more flexibility: we may reject all 'bad' edges, even if many of them occur consecutively. Given this increased flexibility, one would expect that the model we consider here could accommodate more edges before a giant component appears. We shall see below that this is indeed the case. That is, we give an upper bound **whp** on the number of edges handled by any Achlioptas process which is below the rigorous lower bound $m = .9668n$ mentioned above.

For ease of notation, we assume that each random edge e_t is an *ordered pair* (x_t, y_t) of vertices chosen uniformly at random from $[n] \times [n]$. Thus, we choose with replacement and we allow $x_t = y_t$. However, excluding loops and multiple edges will not change the result, as the probability that there are none in our model is bounded away from zero, noting that we only ever consider $O(n)$ edges. Hence, showing that our model satisfies a property **whp** implies that the same holds when restricted to no loops or multiple edges. In order to state our results we must also define a special constant c^*. For $c > 1/4$ define $t = t(c) < 1$ by

$$te^{-t} = 4ce^{-4c}.$$

Let c^* be the unique solution to the equation

$$L(c) := \frac{t^2}{8c} + 1 - \frac{t}{4c} - c = 0. \tag{1}$$

Observe that $L(c)$ is positive at $c = 1/4$, negative at $c = 1$ and monotone decreasing in between. Observe further that $c^* \in [.9792, .9793]$.

Theorem 1.

(a) There is an on-line algorithm that **whp** *selects at least* $m = \lfloor .96689n \rfloor$ *out of* $2m$ *sequentially presented random edges without creating a component of size more than* 200*.*

(b) Let $\eta > 0$ *be any positive constant, and define* c^* *as above.*

(1) If $m' = \lfloor (c^* + \eta)n \rfloor$ *and* X *is a collection of* $2m'$ *random edges then* **whp** all $Y \in \binom{X}{m'}$ *produce graphs* $H = ([n], Y)$ *that have components of size* $\Omega(n)$*.*

(2) There exists a constant C *(depending on* η*) such that there is a polynomial time algorithm which* **whp** *chooses* $m'' = \lfloor (c^* - \eta)n \rfloor$ *edges out of* $2m''$ *random edges without creating a component with more than* C *vertices.*

(c) If $e_1, e_1'; e_2, e_2'; \ldots; e_m, e_m'$ *is a sequence of* $m \geq 0.97765n$ *pairs of random edges then* **whp** *all edge sets of the form* $Y = \{f_1, \ldots, f_m\}$ *where* $f_i \in \{e_i, e_i'\}$ *for* $i = 1, \ldots, m$ *produce graphs* $H = ([n], Y)$ *that have components of size* $\Omega(n)$*.*

(d) Each Achlioptas process **whp** *creates a component of size* $\Omega(n)$ *before accepting* $0.964446n$ *edges.*

Note that parts (a) and (d) of Theorem 1 'separate' the process we introduce here from the Achlioptas processes. In other words, a small change in the rules for our choices results in a significant change in the maximum number of edges we can have in the generated graph before a giant component appears. A similar situation arises in the problem known as '20 questions with a liar.' In that problem, one player tries to determine which element from a set of n is being held by a second player by asking a series of yes/no questions, with the complication that the player answering the questions is allowed to lie some positive proportion of the time. It turns out that there are different thresholds for that problem depending on the rules imposed on the liar [5].

Our proof of Theorem 1(a) relies on the numerical solution of a large system of differential equations. We are confident that with the investment of more time (both computer time and attention to the analysis) our method can be used to achieve even better results. In fact, our calculations make it seem possible that the threshold described in part (b) can be achieved on-line. For the sake of brevity, however, we refrain from getting the best possible result out of our method. We state and prove a result sufficient to establish the separation mentioned above.

Of course, there are a number of interesting questions that remain open here. In addition to the question of whether or not the threshold described in part (b) can be achieved on-line, there are a number of other interesting questions we can ask about the behavior of these models around the phase transition.

We will prove the upper bounds in the next section. Section 3 deals with the algorithm we use for part (a).

2 Upper Bounds

Throughout this discussion we set $m = \lfloor cn \rfloor$ for some constant $c \leq 1$, and G is the random graph on vertex set $[n]$ consisting of $2m$ random edges (each edge is an ordered pair of randomly chosen vertices, as noted above).

We begin by proving a general density lemma. For constants $\epsilon, \delta > 0$ let $\mathcal{A}_{\epsilon,\delta}$ be the event that there exists $S \subseteq [n]$ such

(a) $|S| < \delta n$,

(b) the graph $G[S]$ contains more than $(1+\epsilon)|S|$ edges.

Lemma 1. *If $\epsilon > 0$ and $\delta = \delta(\epsilon) = 2\epsilon\,(4ce)^{-1-1/\epsilon}$ then $Pr\,(\mathcal{A}_{\epsilon,\delta}) = o(1)$.*

Proof. Since the property in question is monotone increasing we can work within the independent model $G' = G_{n,4c/n}$ (see Theorem 2.2 in [2]). Also, we can assume without loss of generality that $G[S]$ is connected.

We bound the probability of the existence of S in one of two ways, depending on $s := |S|$. First, assume $4 \leq s \leq (\log n)/6$ and let A_s be the event that there exists $S \in \binom{[n]}{s}$ such that $G'[S]$ is a spanning connected graph containing at least $s+1$ edges. We have

$$\begin{aligned}
\mathbf{Pr}(A_s) &\leq \binom{n}{s} s^{s-2} \binom{\binom{s}{2}}{2} p^{s+1} \\
&\leq \left(\frac{ne}{s}\right)^s s^{s-2} \frac{s^4}{8} \left(\frac{4c}{n}\right)^{s+1} \\
&= \frac{cs^2}{2n} (4ce)^s \\
&= o(n^{-1/2}).
\end{aligned}$$

For $\log n/6 < s < \delta n$ let A_s be the event that there exists $S \in \binom{[n]}{s}$ such that $G'[S]$

is a spanning connected graph containing at least $(1+\epsilon)s$ edges.

$$\begin{aligned}\mathbf{Pr}(A_s) &\le \binom{n}{s} s^{s-2} \binom{\binom{s}{2}}{\epsilon s} p^{s(1+\epsilon)} \\ &\le \frac{(ne)^s}{s^2} \left(\frac{se}{2\epsilon}\right)^{\epsilon s} \left(\frac{4c}{n}\right)^{(1+\epsilon)s} \\ &= \frac{1}{s^2}\left[4ce\left(\frac{4ce}{2\epsilon}\right)^{\epsilon}\left(\frac{s}{n}\right)^{\epsilon}\right]^s \\ &< \frac{1}{s^2}.\end{aligned}$$

Thus

$$\sum_{s=4}^{n} \mathbf{Pr}(A_s) = o(1)$$

as required. □

In the proofs of the upper bounds, we also make use of the following simple observations. Let V_1 be the set of isolated vertices in G, let V_2 be the set of vertices of degree 1 in G and let M be the set of isolated edges in G. It follows from straightforward mean and variance calculations that **whp** we have the following:

$$\alpha n := |V_1| = ne^{-4c} + \gamma n^{2/3}, \tag{2}$$

$$\beta n := |V_2| = 4ce^{-4c}n + \gamma n^{2/3}, \tag{3}$$

$$\nu n := |M| \ge 2ce^{-8c}n + \gamma n^{2/3}. \tag{4}$$

where $-1 < \gamma < 1$, different at each occurrence.

2.1 Proof of Theorem 1(b)

By elementary calculus, there exists an absolute constant $A > 0$ such that if $c = c^* + x$ then

$$L(c) = -Ax + O(x^2) \qquad \text{as } x \to 0. \tag{5}$$

First assume that $c = c^* + \eta$, and recall that $m' = \lfloor (c^* + \eta)n \rfloor = m$ in this case. Assume that $\mathcal{A}_{\epsilon,\delta}$ does not occur. Assume further that G has a unique giant component K such that

(i) K has $(1 - \frac{t}{4c})n + o(n)$ vertices, and

(ii) The rest of G consists of a forest with $\frac{t^2}{8c}n + o(n)$ edges and maximum tree size $O(\log n)$ together with $O(\log n)$ vertices in unicyclic components.

It is known, [2], [4], that G satisfies (i) and (ii) **whp**. Let Y be a set of m' edges of G and let $H = ([n], Y)$. Now, we apply Lemma 1, but letting S be the vertex set of a component of H that is contained in K. The number of edges of Y which also belong to to K is at least

$$\begin{aligned} cn - \frac{t^2}{8c}n - o(n) &= \left(1 + \frac{-L(c)}{1 - t/4c}\right)|K|(1 + o(1)) \\ &\sim \left(1 + \frac{A\eta + O(\eta^2)}{1 - t/4c}\right)|K| \\ &\geq (1 + A\eta)|K| \end{aligned} \tag{6}$$

since $t > 0$, for sufficiently small η and large n. Hence, S can be chosen so that it spans at least $(1 + A\eta)|S|$ edges of H. It follows by Lemma 1 that **whp** such S has size at least $\delta(A\eta)n$, and (b1) follows.

Now assume that $c = c^* - \eta$. Again, we may assume G has a unique largest component satisfying (i) and (ii). Moreover, we may assume that the forest in (ii) has at most ζn vertices in trees of size greater than C, for a certain function $\zeta \to 0$ as $C \to \infty$. (C will be chosen later to make ζ sufficiently small.)

We explain how to carefully choose the desired set Y of at least $m = \lfloor cn \rfloor$ edges. First of all, Y will contain all of the edges in trees of G of size less than C. By (ii), with the strengthening stated above, this contains all but $\zeta n + O(\log n)$ of the edges outside the largest component K. To complete Y we will need, by the argument leading to (6), to choose a further

$$cn - \frac{t^2}{8c}n + \zeta n - o(n) \leq (1 - A\eta)|K| + \zeta n \tag{7}$$

edges.

Note that the expected number of vertices of degree j in G is less than $n(4cn/(n-1))^j/j!$, for all j. Thus for any $\Delta > 0$ the expected value of the number, Z, of edges in G incident with vertices of degree more than Δ is at most

$$n\sum_{j>\Delta}(4cn/(n-1))^j/(j-1)! < \zeta'(\Delta)n$$

where $\zeta'(\Delta) \to 0$ as $\Delta \to \infty$. By a standard argument, the variance of this number of edges is $o(n^2)$. For fixed Δ to be chosen later, we delete all vertices of degree greater than Δ from K (as none of these will be used in Y), and choose an arbitrary spanning forest F of the resulting subgraph of K. Then by Chebyshev's inequality **whp** the number of edges incident with deleted vertices is less than $2\zeta'(\Delta)n$, and so

$$|E(F)| \geq |K| - 2\zeta'(\Delta)n \tag{8}$$

since $|E(K)| \geq |K| - 1$.

Claim 1. *If T is a tree with at least Δ^2 vertices and maximum degree at most Δ then we can delete edges of T to obtain a forest F' in which every subtree has between Δ and Δ^2 vertices.*

Proof For each edge of such a tree T, let $\rho(e)$ be the size of the smaller of the two components of $T-e$. Let $e^*=(x,y)$ maximize $\rho(e)$. If $\rho(e^*)\geq\Delta$ then each component of $T-e^*$ by induction has the required set of edges, and we are done. So we may assume $\rho(e^*)<\Delta$. Let the edges in the larger component of $T-e^*$ and adjacent to e^* be $e_1,e_2,\ldots,e_k$. Since $e^*=(x,y)$ maximizes $\rho(e)$, for each i, the smaller component of $T-e_i$ is the one not containing e^*, and has $\rho(e_i)\leq\Delta-1$ vertices. Hence T has at most $1+\Delta(\Delta-1)$ vertices, a contradiction. This proves the claim.

Applying the claim to the tree components in F containing more than Δ^2 vertices, we see that we can find at least $|E(F)|(1-1/\Delta)$ edges inside K which span a graph whose maximum component size is at most Δ^2. Now set $C=\Delta^2$ and recall that ζ and ζ' can be made arbitrarily small by choosing C sufficiently large. Thus for some C, by (8) we can find the edges we need to satisfy (7). □

2.2 Proof of Theorem 1(c)

If $c:=m/n>0.97765$ then there exists $\epsilon>0$ such that

$$1-c-e^{-4c}-2ce^{-8c}-c\left(2e^{-4c}-e^{-8c}\right)^2<-\epsilon. \tag{9}$$

As above, we set $\delta=2\epsilon\left(4ce\right)^{-1-1/\epsilon}$ and consider the graph $H:=([n],Y)$.

Suppose that H has no component having more than δn vertices, that (2), (3), (4) hold and that the event $\mathcal{A}_{\epsilon,\delta}$ does not (all of which is true **whp**, by Lemma 1 and the observations after it). Letting $S=[n]\setminus\{V_1\cup V_2\}$ and applying the falseness of $\mathcal{A}_{\epsilon,\delta}$ to the components of $H[S]$,

$$|E(H[S])|\leq(1+\epsilon)|S|. \tag{10}$$

We observe that if *both* e_i and e_i' are incident with vertices of degree 1 then one of these is *not* in Y, and this increases the minimum edge density of $H[S]$. Let M' be the set of edges that contain a vertex of degree 1 in G. Since $|M'|=\beta n-\nu n$, **whp** the number of indices i such that both e_i and e_i' are in M' is greater than $(\beta n-\nu n)^2/4m-n^{2/3}$. Assuming that this inequality holds, it follows from (10) that

$$\begin{aligned}(1+\epsilon)(n-\alpha n-\beta n)&\geq cn-(|M'|-|\{i:e_i,e_i'\in M'\}|)\\&=cn-\beta n+\nu n+|\{i:e_i,e_i'\in M'\}|\\&\geq cn-\beta n+\nu n+\frac{(\beta n-\nu n)^2}{4nc}-n^{2/3},\end{aligned}$$

and hence

$$1-c-e^{-4c}-2ce^{-8c}-c\left(2e^{-4c}-e^{-8c}\right)^2\geq-\epsilon+\epsilon e^{-4c}(1+4c)-O(n^{2/3}). \tag{11}$$

This violates (9) for n sufficiently large. □

Remark. We could make a slight improvement in Theorem 1(c) by considering edges that do not contain degree 1 vertices, but do lie in small components of G. Since this improvement is rather small, it is excluded for the sake of brevity. For the same reason we omit similar improvements of the bound given in Theorem 1(d).

2.3 Proof of Theorem 1(d)

We follow the density argument of the proof of Theorem 1(c); that is, for $Y = \{f_1, \ldots, f_m\}$ we consider the edge density of the graph $H[S]$ where $H = ([n], Y)$ and $S = [n] \setminus \{V_1 \cup V_2\}$, and obtain a contradiction to the assumption that H has no component having more than δn vertices, with $\delta = 2\epsilon\,(4ce)^{-1-1/\epsilon}$ with ϵ sufficiently small.

Our goal is to show that the edge density of $H[S]$ is large; in particular, we show that many of the edges of M' are *not* in Y. As in the proof of Theorem 1(c), we use the fact that for any index i such that $e_i, e_i' \in M'$ one edge in $\{e_i, e_i'\}$ is not in Y. We shall now get an additional improvement in $|M' \setminus Y|$ from a similar observation that uses the assumption that the edge set Y is chosen by an online algorithm. Let B_i be the set of vertices in the graph $G_i := ([n], \{e_1, e_1', \ldots, e_i, e_i'\})$ of degree 1. If both e_i and e_i' contain exactly one vertex in B_i then, conditional upon exactly one of the edges e_{i-1}, e_i winding up in M' (i.e. exactly one of the two degree 1 vertices in $e_i \cup e_i'$ 'surviving' as a degree 1 vertex) with probability 1/2 the edge chosen by the online algorithm will be the one that is not in M'. Informally, we may say that the probability that the online algorithm 'chooses the right edge' is 1/2, although this may be misleading since it takes the point of view that the remaining edges are not revealed, and yet the right choice is not known until they are. We shall take advantage of these as well as a large number of other 'mistakes' of similar types that any algorithm will make **whp**.

We need to quantify the difference made by the 'mistakes'. Let $\mathcal{I}$ denote the set of values of i such that both e_i and e_i' contain at least one vertex of B_i. This set is partitioned as follows. For $1 \le k \le 3$, let $\mathcal{I}_k$ denote the set of values of $i \in \mathcal{I}$ such that exactly $k+1$ vertices of e_i and e_i' are in B_i. These sets are further partitioned according to the edge pairs arriving after e_i, e_i': let $\mathcal{I}_k^{(j)}$ be the set of times $i \in \mathcal{I}_k$ that exactly j of the $k+1$ vertices in B_i and in e_i and e_i' are of degree 1 in G $(0 \le j \le k+1)$.

For each $i \in \mathcal{I}_1$, the algorithm chooses one edge e_i or e_i'. Conditioning on $i \in \mathcal{I}_1^{(1)}$, the remaining edges in the process cause exactly one of the two edges to be in M', and by the symmetry of the situation, with probability 1/2, this is the edge not chosen by the algorithm. Thus, for each $i \in \mathcal{I}_1^{(1)}$, this pair of edges produces an edge in $M' \setminus Y$ with probability 1/2. As in the proof of part (c) for each $i \in \mathcal{I}_1^{(2)}$, the pair of edges e_i, e_i' produces an edge in $M' \setminus Y$ with probability 1.

For $i \in \mathcal{I}_2$, the best edge for the algorithm to choose is the one containing two

vertices in B_i; if this is chosen, then, conditioning on $i \in \mathcal{I}_2^{(1)}$, the edges generated in the rest of the process are not incident with the other vertex in B_i (thereby giving an edge in $M' \setminus Y$) with probability 1/3. If the algorithm chooses the other edge, an even better bound is obtained; that is, 1/3 is a lower bound in all cases. For each $i \in \mathcal{I}_2^{(2)}$, we may similarly assume that the algorithm chooses the one hitting two vertices in B_i, and an edge results in $M' \setminus Y$ with probability 2/3. Finally, for each $i \in \mathcal{I}_2^{(3)}$, an edge results in $M' \setminus Y$ with probability 1. For $i \in \mathcal{I}_3^{(j)}$, a similar argument gives probabilities 1/2 when $j = 1$, 5/6 when $j = 2$, and 1 when $j = 3$ or 4.

Hence, the expected number of edges in $M' \setminus Y$ due to these events is at least $\mathbf{E}Q$ where, putting $I_k^{(j)} = |\mathcal{I}_k^{(j)}|$,

$$Q = \frac{1}{2}I_1^{(1)} + I_1^{(2)} + \frac{1}{3}I_2^{(1)} + \frac{2}{3}I_2^{(2)} + I_2^{(3)} + \frac{1}{2}I_3^{(1)} + \frac{5}{6}I_3^{(2)} + I_3^{(3)} + I_3^{(4)}.$$

In fact, we can say more: the probabilities of creating edges in $M' \setminus Y$ calculated above are valid even when conditioning on the history of the process up to time i. It follows that the total number of such edges resulting from $i \in \mathcal{I}_k^{(j)}$ is bounded below in distribution by the corresponding binomial random variable, and so **whp**

$$|\,|M' \setminus Y| - Q| < n^{2/3}. \tag{12}$$

If J_k denotes the total number of vertices of degree 1 in G that are in the edges e_i or e_i' for some $i \in \mathcal{I}_k$ then for $k = 1, 2, 3$,

$$J_k = \sum_{j=1}^{k+1} jI_k^{(j)}. \tag{13}$$

Let J_3' denote the number of unordered pairs of vertices of degree 1 in G and in e_i or e_i', summed over $i \in \mathcal{I}_3$. That is, $J_3' = \frac{1}{2}\sum_{j=2}^{4} j(j-1)I_3^{(j)}$. Then

$$3J_3' = I_3^{(2)} + 3I_3^{(3)} + 6I_3^{(4)}.$$

In combination with (13) we then obtain

$$Q = \frac{1}{2}J_1 + \frac{1}{3}J_2 + \frac{1}{2}J_3 - \frac{1}{6}J_3'.$$

By elementary calculations, for fixed $x > 0$ and integer r, the probability that xn pairs of randomly chosen edges from n vertices do not intersect r specified vertices is asymptotic to e^{-4rx} as $n \to \infty$. Insisting the edges are distinct makes negligible difference to this. Hence (analogous to (2), the probability that exactly s vertices of e_i and e_i' are in B_i is asymptotic to $\binom{4}{s}e^{-4si/n}(1 - e^{-4i/n})^{4-s}$. With $s - 1 = k \geq 2$, this gives $\mathbf{Pr}(i \in \mathcal{I}_k)$. For $k = 1$ a similar calculation holds, but in that case the requirement that the two vertices must be in different edges causes $\binom{4}{s}$ to be

replaced by 4. Thus, the contribution from i to J_1 is $4e^{-8i/n}(1-e^{-4i/n})^2 \cdot 2e^{-4(c-i/n)}$. The sum of this quantity over i is given asymptotically by an integral:

$$\mathbf{E}J_1/n \sim \int_0^c 4e^{-8x}(1-e^{-4x})^2 \cdot 2e^{-4(c-x)}\,dx \sim e^{-4c}(1-e^{-4c})^3,$$

and similarly

$$\begin{aligned}
\mathbf{E}J_2/n &\sim \int_0^c 4e^{-12x}(1-e^{-4x}) \cdot 3e^{-4(c-x)}\,dx \sim \frac{1}{2}e^{-4c} - \frac{3}{2}e^{-12c} + e^{-16c},\\
\mathbf{E}J_3/n &\sim \int_0^c e^{-16x} \cdot 4e^{-4(c-x)}\,dx \sim \frac{1}{3}e^{-4c} - \frac{1}{3}e^{-16c},\\
\mathbf{E}J_3'/n &\sim \int_0^c e^{-16x} \cdot 6e^{-8(c-x)}\,dx \sim \frac{3}{4}e^{-8c} - \frac{3}{4}e^{-16c},
\end{aligned}$$

from which

$$\mathbf{E}Q/n \sim \frac{2}{3}e^{-4c} - \frac{9}{8}e^{-8c} + \frac{1}{2}e^{-12c} - \frac{1}{24}e^{-16c}. \tag{14}$$

Each random variable $I_k^{(j)}$ is sharply concentrated, by a standard martingale argument (such as by [4, Corollary 2.27]). This gives sharp concentration of Q; that is, for some $\lambda \to 0$, $\mathbf{Pr}(|Q-\mathbf{E}Q| < \lambda n) \to 1$. So by (12), it follows that $|M' \setminus Y|/n$ is also given asymptotically by (14) **whp**.

We can now turn to the edge density argument on $H[S]$. Arguing as in the equations leading to (11), we have

$$\begin{aligned}
(1+\epsilon)(n-\alpha n-\beta n) &= (1+\epsilon)|S|\\
&\geq |E(G[S])|\\
&\geq cn - |M'| + |M' \setminus Y|\\
&= cn - \beta n + \nu n + |M' \setminus Y|.
\end{aligned}$$

We have a contradiction if

$$1 - c - e^{-4c} - 2ce^{-8c} - \frac{2}{3}e^{-4c} + \frac{9}{8}e^{-8c} - \frac{1}{2}e^{-12c} + \frac{1}{24}e^{-16c} < 0,$$

which holds for $c > 0.9644456$.

□

3 The Algorithm

In this section we present the algorithm which achieves the result claimed in Theorem 1(a), as well as the lower bound $.9760n$ nonrigorously. We set $E_t = \{e_1, e_2, \ldots, e_t\}$, and denote by A_t the set of edges from E_t that are actually chosen by the algorithm. Thus $A_0 = E_0 = \emptyset$. The algorithm we use runs in Phases

$k = 2, 3, \ldots$. The choice to transition from one phase to the next is governed by a function $g : \mathbb{N} \to \mathbb{R}$ such that $g \downarrow 1/2$, which can be viewed as a parameter of the algorithm. During phase k, the algorithm accepts those edges that form components of size at most k and transits to the next phase when the proportion of chosen edges drops below $g(k)$.

```
1.  Begin
2.      A ← ∅
3.      k ← 2
4.      t ← 1
5.      l ← 0
6.      repeat
7.          l ← size of largest component of A ∪ {e_t}
8.          if l ≤ k then
9.              A ← A ∪ e_t
10.             t ← t + 1
11.         else if |A|/t < g(k) then
12.             k ← k + 1
13.         else t ← t + 1
14.     until l = n
15. End
```

Note that at every step the algorithm has chosen at least half of the edges presented so far. Furthermore, at any step of the algorithm, the size of the largest component in the graph is at most the current phase.

We analyze this using the 'differential equations' method for concentration of random variables. We actually analyze a related algorithm which proceeds through a *bounded* number of phases, up to k_{final}. In the last phase the algorithm proceeds until the proportion of edges chosen drops to $1/2$. We keep track of the following set of random variables as the algorithm proceeds: $\mathbf{X}(t) = (X_i(t), i = 0, 1, 2, \ldots)$ where $X_0(t) = A(t)$ and for $i \geq 1$ the random variable $X_i(t)$ denotes the number of components with i vertices in the graph $\Gamma_t = ([n], A_t)$. Thus if t lies in Phase k, $X_i(t) = 0$ for $i > k$. During Phase k, for $1 \leq i \leq k$,

$$\mathbf{E}(X_i(t+1) - X_i(t) \mid \mathbf{X}(t)) = \sum_{j=1}^{i-1} \frac{j(i-j)X_j(t)X_{i-j}(t)}{n^2} - 2\sum_{j=1}^{k-i} \frac{ijX_i(t)X_j(t)}{n^2}. \quad (15)$$

(This equation is written for a process in which each edge is selected by choosing an order pair of vertices with replacement. With high probability, the number of times the same vertex is chosen twice for a given edge is at most $\log n$ say during this whole process. We can therefore extend the process by this many edges to obtain the same result for the true process. Alternatively, one can insert error terms of order n^{-1} and the rest of the argument still applies.) Furthermore,

calculating the probability that the edge e_{t+1} is accepted gives

$$\mathbf{E}(X_0(t+1) - X_0(t) \mid \mathbf{X}(t)) = \sum_{i=1}^{k-1} \sum_{j=1}^{k-i} \frac{ijX_i(t)X_j(t)}{n^2}. \tag{16}$$

This type of process can be closely approximated using differential equations. Setting $t = \tau n$ we consider the following sequence of systems of differential equations (where we set $x_i^{(k)} = x_i^{(k)}(\tau)$ and consider only $\tau \geq 0$):

System k:

$$\begin{aligned} \dot{x}_0^{(k)} &= \sum_{\ell=1}^{k-1} \sum_{j=1}^{k-\ell} \ell j x_\ell^{(k)} x_j^{(k)} \\ \dot{x}_i^{(k)} &= \sum_{j=1}^{i-1} j(i-j) x_j^{(k)} x_{i-j}^{(k)} - 2 \sum_{j=1}^{k-i} ij x_i^{(k)} x_j^{(k)}, \qquad i = 1, 2, \ldots, k. \end{aligned} \tag{17}$$

For Phase 2 (which is the first sensible phase) we use the boundary conditions $x_0^{(2)}(0) = 0, x_1^{(2)}(0) = 1$ and $x_i^{(2)}(0) = 0$ for $i \geq 2$. Our next task is to determine the times when we switch between phases. In the random process these will be $1 = t_2 \leq \cdots \leq t_{k_{\text{final}}} \leq t_{k_{\text{final}}+1}$. In the differential equations simulation we define $0 = \tau_2 \leq \cdots \leq \tau_{k_{\text{final}}} \leq \tau_{k_{\text{final}}+1}$ where we inductively define

$$\tau_{k+1} = \min \left\{ \tau \geq \tau_k : x_0^{(k)}(\tau)/\tau = g(k) \right\}. \tag{18}$$

When we switch between phases in the differential equations simulation we use the following boundary conditions at the start τ_{k+1} of Phase $k+1$,

$$x_i^{(k+1)}(\tau_{k+1}) = x_i^{(k)}(\tau_{k+1}),\ 0 \leq i \leq k \text{ and } x_{k+1}^{(k+1)}(\tau_{k+1}) = 0.$$

It follows directly from Theorem 5.1 of Wormald [6] that we have the following for a fixed Phase k: if $\lambda > 0$ then

$$X_i(t) = n x_i^{(k)}(t/n) + O(\lambda n), \tag{19}$$

uniformly in t, for $i = 0, \ldots, k$, with probability

$$1 - O\left(\lambda^{-1} e^{-n\lambda^3}\right).$$

It suffices to take $\lambda = n^{-1/4}$ here.

It remains to determine the exact values of the transition times $\tau_3, \cdots, \tau_{k_{\text{final}}+1}$. In particular, $\tau_{k_{\text{final}}+1}$ is the termination point for the algorithm. We calculate these transition times numerically. We should note that in so doing, we are changing the algorithm slightly. The algorithm that we are simulating is the one that transitions between phases at the value of τ that is given by the numerical calculations,

rather than at the value given by (18). In order to achieve the bound given in the statement of the theorem, we used Euler's method to solve the differential equations, and $g(k) = 1/2 + \sqrt{1/(2k)}$ and $k_{final} = 200$ (we arrived at these parameters through trial and error, we have no reason to believe that they are optimal). We bounded the error in these calculations using methods set out in the subsection below. Our goal here was not to determine the best possible result our algorithm can give, but rather to give a proof that if the parameters are chosen properly then **whp** the algorithm succeeds for m greater than the upper bound on the Achlioptas processes given in Theorem 1(d) (i.e. our goal was to establish the separation of problems discussed in the introduction). Furthermore, we attempt to achieve this in the simplest way we can manage. The program (written in C) we used for the numerical calculations is posted at `http:/www.math.cmu.edu/~af1p/nogiant.txt`.

Of course, there are other ways to solve the differential equation and bound the error. We are confident that with the investment of more time (both computer time and attention to the error analysis) results even closer to the upper bound given in Theorem 1(b) can be achieved. For example, by using a Runge-Kutta method (in the place of Euler's method, which is used below) without error analysis but with excellent convergence apparent, we obtain more than 0.976 with $k_{\text{final}} = 10^4$. This is pleasantly close to the upper bound in Theorem 1(c).

Error bounds

For simplicity we may take k and n fixed, and write the differential equation (17) as

$$\dot{x}_i = F_i(\mathbf{x}), \quad 0 \le i \le k$$

where $\mathbf{x} = (x_0, \dots, x_k)$. This is an autonomous system, i.e. F_i does not depend on t. Our goal in this section is to establish, in the simplest way we can manage, that the error in our numerical approximation to the solution of this differential equation is small.

We begin with a simple observation.

Claim 2. *For any vector* $\mathbf{y}$ *we have*

$$\sum_{i=1}^{k} iF_i(\mathbf{y}) = 0$$

Proof.

$$\begin{aligned}
\sum_{i=1}^{k} iF_i(\mathbf{y}) &= \sum_{i=1}^{k} i \left(\sum_{j=1}^{i-1} j(i-j) y_j y_{j-i} - 2 \sum_{j=1}^{k-i} j i y_i y_j \right) \\
&= \sum_{i=1}^{k} \sum_{j=1}^{k-i} ij y_i y_j \left(2(i+j) - 2i - 2j\right) \\
&= 0
\end{aligned}$$

□

Throughout this section we will make use of the following observation, that follows immediately from the differential equation (17) and Claim 2: For t in Phase k we have

$$\sum_{i=1}^{k} i x_i(t) = 1. \tag{20}$$

Of course, (20) expresses the simple fact that at every stage of the algorithm every vertex lies in exactly one component.

To solve the equations by Euler's method, set $\tilde{x}_i(0) = x_i(0)$ $(0 \le i \le k)$ and then, given $\tilde{\mathbf{x}}(t) = (\tilde{x}_0, \ldots, \tilde{x}_k)$, try to compute

$$\tilde{x}_i(t+h) = \tilde{x}_i(t) + hF_i(\tilde{\mathbf{x}}(t)), \quad 0 \le i \le k.$$

This is iterated for $t = 0, h, 2h, \ldots$. When computed by machine, we actually have

$$\tilde{x}_i(t+h) = \tilde{x}_i(t) + hF_i(\tilde{\mathbf{x}}(t)) + \rho_i(t) \tag{21}$$

where $\rho_i(t)$ is the rounding error due to floating point approximation in machine computation. In this discussion we assume

$$|\rho_i(t)| \le \eta(2 + 4k^2 h) \le 3\eta \tag{22}$$

where η is the maximum error in a single floating point computation (note that we can assume that we never do computations on numbers larger than 1). Of course, our goal in this section is to show that the differences

$$e_i(t) = x_i(t) - \tilde{x}_i(t), \quad 0 \le i \le k$$

remain small throughout the numerical computations. We have

$$\begin{aligned} e_i(t+h) &= x_i(t+h) - \tilde{x}_i(t+h) \\ &= x_i(t) + hF_i(\mathbf{x}(t)) + \tau_i(t) - (\tilde{x}_i(t) + hF_i(\tilde{\mathbf{x}}(t)) + \rho_i(t)) \\ &= e_i(t) + h(F_i(\mathbf{x}(t)) - F_i(\tilde{\mathbf{x}}(t))) + \tau_i(t) - \rho_i(t) \end{aligned} \tag{23}$$

where τ_i is the truncation error, i.e.

$$\tau_i(t) = x_i(t+h) - x_i(t) - hF_i(\mathbf{x}(t)).$$

Thus, our main tasks are in bounding $\tau_i(t)$ and the difference $F_i(\mathbf{x}(t)) - F_i(\tilde{\mathbf{x}}(t))$.

We begin with the truncation error. By Taylor's theorem,

$$\tau_i(t) = \frac{h^2}{2} \ddot{x}_i(\xi)$$

for some $t \le \xi_i \le t + h$.

Claim 3. *If ξ lies in phase k and $0 \le i \le k$ then*

$$|\ddot{x}_i(\xi)| \le 8k.$$

Proof. The key observation here is that the sum of the absolute values of the first derivatives is at most a constant.

$$\begin{aligned}
\sum_{i=1}^{k} |\dot{x}_i(\xi)| &\le \sum_{i=1}^{k} \left(\sum_{j=1}^{i-1} j(i-j) x_j(\xi) x_{i-j}(\xi) + 2 \sum_{j=1}^{k-i} ij x_i(\xi) x_j(\xi) \right) \\
&= \sum_{j=1}^{k-1} \sum_{\ell=1}^{k-j} 3j\ell x_j(\xi) x_\ell(\xi) \\
&\le 3 \left(\sum_{j=1}^{k} j x_j(\xi) \right) \left(\sum_{\ell=1}^{k} \ell x_\ell(\xi) \right) \\
&= 3.
\end{aligned}$$

Now, we consider the second derivatives.

$$\begin{aligned}
|\ddot{x}_0(\xi)| &\le \sum_{\ell=1}^{k-1} \sum_{j=1}^{k-\ell} \left(j\ell \, |\dot{x}_j(\xi)| \, x_\ell(\xi) + j\ell x_j(\xi) \, |\dot{x}_\ell(\xi)| \right) \\
&\le 2 \left(\sum_{j=1}^{k} j x_j(\xi) \right) \left(\sum_{\ell=1}^{k} \ell \, |\dot{x}_\ell(\xi)| \right) \\
&\le 6k
\end{aligned}$$

For $1 \le i \le k$ we have

$$\begin{aligned}
|\ddot{x}_i(\xi)| &\le \sum_{j=1}^{i-1} \left(j(i-j) \, |\dot{x}_j(\xi)| \, x_{i-j}(\xi) + j(i-j) x_j(\xi) \, |\dot{x}_{i-j}(\xi)| \right) \\
&\qquad + 2 \sum_{j=1}^{k-i} \left(ij \, |\dot{x}_i(\xi)| \, x_j(\xi) + ij x_i(\xi) \, |\dot{x}_j(\xi)| \right) \\
&\le 2 \left(\sum_{j=1}^{k} j x_j(\xi) \right) \left(\sum_{\ell=1}^{k} \ell \, |\dot{x}_\ell(\xi)| \right) + 2i^2 x_i(\xi) \, |\dot{x}_i(\xi)| \, \delta_{i \le k/2} \\
&\le 8k
\end{aligned}$$

where

$$\delta_{i \le k/2} = \begin{cases} 1 & \text{if } i \le k/2 \\ 0 & \text{otherwise.} \end{cases}$$

Note that we use the easily verified fact that $|\dot{x}_i(\xi)| \le 2$ in the last inequality. □

It follows from the Claim that, for t in phase k we have

$$|\tau_i(t)| \leq 4h^2 k. \tag{24}$$

Now we consider $F_i(\mathbf{x}(t)) - F_i(\tilde{\mathbf{x}}(t))$. Here we resort to the numerical computation itself to verify that the error remains small (i.e. we are actually doing interval arithmetic). We first define

$$f(t) = \sum_{i=1}^{k} i e_i(t) = \sum_{i=1}^{k} i x_i(t) - i\tilde{x}_i(t) = 1 - \sum_{i=1}^{k} i\tilde{x}_i(t)$$

for t in Phase k. We shall see that during the course of our simulations $f(t)$ is small (this is verified numerically). To show that $F_i(\mathbf{x}(t)) - F_i(\tilde{\mathbf{x}}(t))$ is small we take advantage of all possible cancellation in the sum $e_i(t) + h(F_i(\mathbf{x}(t))) - F_i(\tilde{\mathbf{x}}(t))$. We have

$$\begin{aligned}
e_i(t) + h(F_i(\mathbf{x}(t))) - F_i(\tilde{\mathbf{x}}(t))) &= e_i(t) + h\left(\sum_{j=1}^{i-1} j(i-j)(\tilde{x}_j e_{i-j} + \tilde{x}_{i-j} e_j + e_j e_{i-j}) \right. \\
&\qquad \left. -2\sum_{j=1}^{k-i} ij(\tilde{x}_i e_j + \tilde{x}_j e_i + e_i e_j)\right) \\
&= e_i(t)\left(1 - 2hi\sum_{j=1}^{k-i} j\tilde{x}_j\right) && (25) \\
&\quad + 2h\sum_{j=1}^{i-1} j(i-j) e_j \tilde{x}_{i-j} - 2h\sum_{j=1}^{k-i} ij\tilde{x}_i e_j && (26) \\
&\quad + h\sum_{j=1}^{i-1} j(i-j) e_j e_{i-j} - 2h\sum_{j=1}^{k-i} ij e_i e_j. && (27)
\end{aligned}$$

In our computations, we simply add in the absolute values of the errors in lines (25) and (27) (note that line (25) will actually give a decrease in the error). We do something slightly more sophisticated with the error in line (26). First note that

$$\begin{aligned}
&\left|\sum_{j=1}^{i-1} j(i-j) e_j \tilde{x}_{i-j} - \sum_{j=1}^{k-i} ij\tilde{x}_i e_j\right| \\
&\quad \leq \sum_{j=1}^{\min\{k-i,i-1\}} |je_j||(i-j)\tilde{x}_{i-j} - i\tilde{x}_i| + \sum_{j=\min\{k-i,i-1\}+1}^{i-1} j(i-j)|e_j|\tilde{x}_{i-j} \\
&\qquad + \sum_{j=\min\{k-i,i-1\}+1}^{k-i} ij\tilde{x}_i|e_j|. && (28)
\end{aligned}$$

This gives us some cancellation when k is small with respect to i. when k is large with respect to i, we invoke the fact that $f(t)$ is small:

$$\begin{aligned}\left|\sum_{j=1}^{i-1} j(i-j)e_j\tilde{x}_{i-j} - \sum_{j=1}^{k-i} ij\tilde{x}_i e_j\right| \\ \leq \sum_{j=1}^{i-1} j(i-j)|e_j|\tilde{x}_{i-j} + i\tilde{x}_i\left|\sum_{j=1}^{k} je_j - \sum_{j=k-i+1}^{k} je_j\right| \\ \leq \sum_{j=1}^{i-1} j(i-j)|e_j|\tilde{x}_{i-j} + i\tilde{x}_i\left(f(t) + \sum_{j=k-i+1}^{k} j|e_j|\right). \quad (29)\end{aligned}$$

The only remaining issue is the error in the $\tilde{x}_0(t)$. Note that this error has no impact on the errors in $\tilde{x}_1(t), \ldots, \tilde{x}_k(t)$. It only has an impact on the termination time of the process. As above, we use the fact that f remains small here:

$$\begin{aligned}e_0(t+h) &\leq \sum_{j=1}^{k}\sum_{i=1}^{k-j} ij\left(\tilde{x}_i\,|e_j| + \tilde{x}_j\,|e_i| + |e_ie_j|\,\right) \\ &\leq \sum_{j=1}^{k}\sum_{i=1}^{k-i} ij\,|e_ie_j| + 2\sum_{j=1}^{k} j\tilde{x}_j \min\left\{\sum_{i=1}^{k-j} i\,|e_i|\,, f + \sum_{i=k-j+1}^{k} i\,|e_i|\right\}\end{aligned}$$

We terminate the algorithm when

$$\frac{\tilde{x}_0(t) - e_0(t)}{t} \leq \frac{1}{2}.$$

By stopping when the error in x_0 could take the proportion of accepted edges below $1/2$ we get a rigorous lower bound on how long the actual process lasts.

References

[1] T. Bohman and A.M. Frieze, Avoiding a Giant Component, *Random Structures and Algorithms* 19 (2001) 75-85 (addendum, *Random Structures and Algorithms* 20 (2002) 126-130).

[2] B. Bollobás, *Random Graphs,* Academic Press, 1985 (Second Edition 2001).

[3] P. Erdős and A. Rényi, *On the evolution of random graphs*, Publ. Math. Inst. Hungar. Acad. Sci. 5 (1960) 17-61.

[4] S. Janson, T. Łuczak, and A. Ruciński, Random Graphs, Wiley - Interscience Series, New York, 2000.

[5] J. Spencer, P. Winkler, Three thresholds for a liar, *Combinatorics, Probability and Computing* 1 (1992) 81-93.

[6] N.C. Wormald, The differential equation method for random graph processes and greedy algorithms, in *Lectures on Approximation and Randomized Algorithms* (M. Karonski and H.J. Proemel, eds) (1999) 73-155.