Comment on "Subgraphs in random networks"

Oliver D. King*
Department of Biological Chemistry and Molecular Pharmacology, Harvard Medical School, 250 Longwood Avenue, SGMB-322, Boston, Massachusetts 02115, USA
(Received 22 October 2003; published 18 November 2004)

We point out biases in the algorithms used by Itzkovitz *et al.* [Phys. Rev. E **68**, 026127 (2003)] to assess their approximate formulas for the average number of occurrences of certain subgraphs in random graphs with prescribed degree sequences.

DOI: 10.1103/PhysRevE.70.058101 PACS number(s): 89.75.Hc, 89.75.Fb

Itzkovitz *et al.* [1] give approximate theoretical formulas for the average number of occurrences of certain small subgraphs in random graphs with prescribed degree sequences. To assess the accuracy of these approximate formulas, they randomly generate 1000 graphs with prescribed degrees and directly count the number of occurrences of the subgraphs. They do this twice, in the first case using the algorithm of Newman *et al.* [2] to generate multigraphs with prescribed degrees, and in the second case using a modified version of this algorithm, as described in Milo *et al.* [3], to generate simple graphs with prescribed degrees [4]. (In a multigraph, multiple edges are allowed between a pair of nodes; in a simple graph they are not.) But they do not mention that the algorithm of Newman *et al.* does not generate multigraphs with prescribed degrees uniformly at random, and the algorithm of Milo *et al.* does not generate simple graphs with prescribed degrees uniformly at random [5].

The stub-pairing algorithm given in Newman *et al.* proceeds as follows: (1) Each node i in the graph is given j_i inward-pointing edge stubs (*in-stubs*) and k_i outward-pointing edge stubs (*out-stubs*), where j_i and k_i are the prescribed in- and out-degrees of node i. (2) Each out-stub is randomly paired with a distinct in-stub to produce a directed graph. Newman *et al.* make no mention of what to do if the resulting graph has edges from a node to itself (*self-loops*) or multiple edges from one node to another (*multiedges*). It is possible that they intended to allow their construction to produce graphs with self-loops and multiedges, since the only restriction they note for the prescribed degrees is that the sum of the out-degrees must be equal to the sum of the in-degrees. (There are additional constraints on degrees of graphs with no self-loops or multiedges [6–8].) But this algorithm does not generate multigraphs with prescribed degrees uniformly (regardless of whether one allows self-loops)—it essentially generates directed "configurations" [9] uniformly at random, but (as has already been noted for both directed and undirected graphs [10–12]) a graph with multiedges has fewer configurations as preimages than a simple graph, by a factor of $k!$ for each k-fold multiedge. Thus, there can be arbitrarily large deviations from uniformity in the generation of multigraphs. (Newman *et al.* do not explicitly claim that this algorithm uniformly generates directed graphs with prescribed degrees, but they do make this claim for an analogous algorithm for undirected graphs, in which the same caveats apply [11,12].) Or, perhaps Newman *et al.* had in mind an implicit third step: (3) If the resulting graph contains any self-loops or multiedges, it is rejected; otherwise, it is accepted. In this case, the algorithm does generate simple graphs with the prescribed degrees uniformly, but the accep-

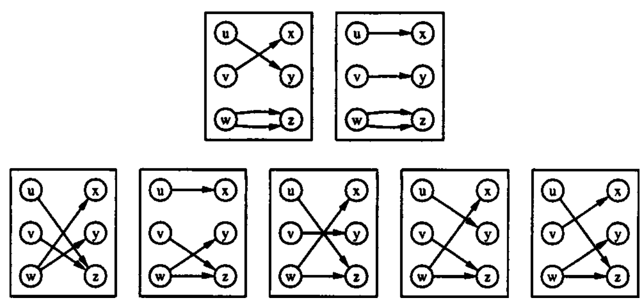

FIG. 1. Shown are the seven graphs with out-degree sequence (1, 1, 2, 0, 0, 0) and in-degree sequence (0, 0, 0, 1, 1, 2) for the nodes u,v,w,x,y,z (in that order). There are $4!=24$ "configurations," or ways to pair up the four out-stubs with the four in-stubs. The two upper graphs are each obtained from 2 of these 24 pairings, while the five lower graphs (the simple graphs) are each obtained from 4 of the 24 pairings. Thus while the stub-pairing algorithm of Newman *et al.* [2] generates configurations uniformly, it does not generate graphs with multiedges allowed uniformly, as the upper two graphs are each generated with probability $1/12$ and the lower five each with probability $1/6$. This stub-pairing algorithm is uniform when restricted to simple graphs, however, while the modified stub-pairing algorithm of Milo *et al.* [3] is not. In this example, the algorithm of Milo *et al.* generates the leftmost simple graph with probability $1/6=0.167$, and each of the other four simple graphs with probability $157/864=0.182$. (With probability $23/216=0.106$ the algorithm reaches a dead end.) These numbers were calculated exactly, by considering all possible orders in which random stub pairs can be selected, but the same effect can be seen numerically by running the algorithm of Milo *et al.* many times. Note that, unlike the leftmost simple graph, the other four simple graphs can each be converted to a graph with a multiedge by replacing some pair of edges $a\rightarrow b$ and $c\rightarrow d$ with the pair $a\rightarrow d$ and $c\rightarrow b$. (Note also that, while the top two graphs are isomorphic, as are the four rightmost simple graphs, the goal throughout this comment is to uniformly generate random *labeled* graphs with prescribed degrees.)

*Electronic address: oliver_king@hms.harvard.edu

tance rate might be too small for the algorithm to be practical.

In what may have been an attempt to increase the acceptance rate when generating simple graphs, Milo *et al.* ([3], supplementary web material) modified the algorithm of Newman *et al.* slightly. Instead of pairing up all the in- and out-stubs at once, and then rejecting the graph if it is not simple, Milo *et al.* take an incremental approach in which one stub pair is chosen at a time; if the addition of the edge between these stubs would create a self-loop or multiedge, a new stub pair is chosen; otherwise, the edge is added to the graph. If at some stage there is no stub pair that can be added without creating a multiedge or self-loop, the partial graph is rejected and the process is started from scratch. The approach of Newman *et al.* can be recast as an incremental algorithm, but one in which a partial graph is rejected as soon as *the first* stub pair is chosen that would create a self-loop or multiedge. Because the modified algorithm of Milo *et al.* does extra exploration in the vicinity of nonsimple partial graphs, it does not generate the simple graphs with prescribed degrees uniformly. (This bias is not mentioned in Milo *et al.* [3] or in Iztkovitz *et al.* [1].) An example is shown in Fig. 1 [13].

It may be that the generation of nonsimple graphs in the stub-pairing algorithm was inconsequential for the purposes of Newman *et al.* [2], as asymptotic properties of simple graphs can often be inferred from asymptotic properties of configurations, particularly when the degrees of the nodes do not grow too fast as the number of nodes increases (e.g. [9–11,14,15]).

Likewise, it may be that the bias in the modified stub-pairing algorithm was inconsequential for the particular graphs considered by Milo *et al.* [3]; they report that the same "network motifs" were identified as statistically significant when using a Markov chain Monte Carlo algorithm to generate simple graphs with prescribed degrees [16,17]. But it should be noted that while Kannan *et al.* [16] have shown that a similar Markov chain is rapidly mixing for near-regular degree sequences [18], they have not shown this for scale-free degree sequences, such as those considered by Milo *et al.* and Itzkovitz *et al.*—Milo *et al.* offer only that they simulate their Markov chains "until the network is well randomized" ([3], supplementary web material).

Finally, it may be that the biases in the algorithms used by Itzkovitz *et al.* [1] as a standard by which to assess their approximate formulas were not significant for the particular graphs they considered. But even if so, and even if no entirely satisfactory algorithm is available, it should nonetheless be noted for the benefit of others who may wish to generate graphs with prescribed degrees uniformly, that the algorithm of Newman *et al.* does not generate multigraphs uniformly and the modified algorithm of Milo *et al.* does not generate simple graphs uniformly.

[1] S. Itzkovitz, R. Milo, N. Kastan, G. Ziv, and V. Alon, Phys. Rev. E **68**, 026127 (2003).

[2] M. E. J. Newman, S. H. Strogatz, and D. J. Watts, Phys. Rev. E **64**, 026118 (2001).

[3] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon, Science **298**, 824 (2002).

[4] Milo *et al.* [3] describe two algorithms for generating simple graphs with prescribed degrees, one a modified version of the stub-pairing algorithm of Newman, Strogatz, and Watts [2], and the other, a Markov-chain Monte Carlo (MCMC) algorithm for which they cite Refs. [16,19]. According to the caption of Fig. 3 in Itzkovitz *et al.* [1], the simple graphs "were constructed using the algorithm of Newman, Strogatz, and Watts modified so that only a single edge in a given direction is allowed between any two nodes." But it may be that the MCMC algorithm was also used for some of the results in Itzkovitz *et al.*, since the caption to Table I says that the direct enumeration results were computed "using the algorithms described in [Milo *et al.*]," without specifying which.

[5] Itzkovitz *et al.* refer to "random networks" without explicitly stating with respect to which probability distribution, so it may be that their intended null models were not those in which all multigraphs with prescribed degrees are equiprobable, or in which all simple graphs with prescribed degrees are equiprobable. In the case of simple graphs, there is no mention of the intended probability distribution, only references to the algorithms used to generate the graphs. In the case of multigraphs, Itzkovitz *et al.* refer to "random networks which allow for multiple edges …as in the well-studied configuration model" and to simulations "when multiple edges are allowed, as in the configuration model." Should these be interpreted as observations that multiple edges can occur in configurations, or as assertions that when multiple edges are allowed, by "random network" they mean with respect to the uniform distribution on configurations (which is different than the uniform distribution on multigraphs)?

[6] D. Gale, Pac. J. Math. **7**, 1073 (1957).

[7] H. J. Ryser, Can. J. Math. **9**, 371 (1957).

[8] D. R. Fulkerson, Pac. J. Math. **10**, 831 (1960).

[9] B. Bollobás, Eur. J. Comb. **1**, 311 (1980).

[10] E. A. Bender, Discrete Math. **10**, 217 (1974).

[11] E. A. Bender and R. E. Canfield, J. Comb. Theory, Ser. A **24**, 296 (1978).

[12] N. C. Wormald, in *Surveys in Combinatorics*, edited by J. D. Lamb and D. A. Preece, LMS Lecture Note Series Vol. 267 (Springer, Berlin, 1999), pp. 239–298.

[13] During the review process, Itzkovitz alerted us to a paper by Rao *et al.* [20], in which it is shown that two variants of an algorithm attributed to Pramanik, for the closely related problem of uniformly generating (0,1) matrices with prescribed row and column sums, are biased. While these algorithms are similar in flavor to the modified stub-pairing algorithm of Milo *et al.* [3], they have an additional step of forcing edges, and give different results. In fact the nonuniform probabilities com-

puted in Rao *et al.* can be directly contrasted with the nonuniform probabilities computed in Fig. 1—Pramanik's algorithms have the same bias for (0,1) matrices with row and column sums (2,1,1) as they do for (0,1) matrices with row sums (2,1,1,0,0,0) and column sums (0,0,0,2,1,1), which after reordering the nodes correspond exactly to the adjacency matrices of the simple directed graphs in Fig. 1. (Moving to the 6×6 matrices forces the diagonal to be zero so that self-loops are not an issue.) Also, a preprint by Chen *et al.* [21] shows that an algorithm due to Sanderson [22] for generating (0,1) matrices with prescribed row and column sums is nonuniform, again by computing the probabilities with which it generates matrices with row and column sums (2,1,1). (Sanderson's algorithm differs from both variants of Pramanik's algorithm and from the algorithm of Milo *et al.*) Chen *et al.* also propose several importance sampling algorithms for computing expectations of random variables over the spaces of (0,1) matrices and contingency matrices with prescribed row and column sums. Here, the matrices are not generated uniformly, but they are weighted to compensate for the nonuniformity when expectations are computed. (See also Snijders [23].)

[14] M. Molloy and B. Reed, Random Struct. Algorithms **6**, 161 (1995).

[15] Caution should be exercised when considering graphs with fat-tailed degree distributions—see, for example, Refs. [24,17,25]. (Reference [24] also discusses an alternative strategy for generating graphs, in which the degrees are allowed to fluctuate.)

[16] R. Kannan, P. Tetali, and S. Vempala, Random Struct. Algorithms **14**, 293 (1999).

[17] S. Maslov, K. Sneppen, and A. Zaliznyak, e-print cond-mat/0205379.

[18] The results of Kannan *et al.* [16] for bipartite graphs and (0,1) matrices can be adapted to directed simple graphs provided self-loops are allowed. There are other Markov chains that are known to be rapidly mixing for simple graphs with nearly regular degree sequences [26], and for contingency tables (closely related to directed multigraphs) in which the minimum row and column sums are large [27].

[19] S. Maslov and K. Sneppen, Science **296**, 910 (2002).

[20] A. R. Rao, R. Jana, and S. Bandyopadhyay, Sankhya, Ser. A **58**, 225 (1996).

[21] Y. Chen, P. Diaconis, S. Holmes, and J. S. Liu, Sequential Monte Carlo methods for statistical analysis of tables. Discussion Paper 03-22, Institute of Statistics and Decision Sciences, Duke University, 2003 (unpublished) [J. Am. Stat. Assoc. (to be published).

[22] J. G. Sanderson, Am. Sci. **88**, 332 (2000).

[23] T. A. B. Snijders, Psychometrika **56**, 397 (1991).

[24] Z. Burda and A. Krzywicki, Phys. Rev. E **67**, 046118 (2003).

[25] M. Boguñá, R. Pastor-Satorras, and A. Vespignani, e-print cond-mat/0311650.

[26] M. R. Jerrum and A. J. Sinclair, Theor. Comput. Sci. **73**, 91 (1990).

[27] F. R. K. Chung, R. L. Graham, and S.-T. Yau, Random Struct. Algorithms **9**, 55 (1996).