

Benchmark graphs for testing community detection algorithms

Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi

Complex Systems Lagrange Laboratory (CNLL), Institute for Scientific Interchange (ISI), Viale S. Severo 65, 10133, Torino, Italy

(Received 30 May 2008; revised manuscript received 11 September 2008; published 24 October 2008)

Community structure is one of the most important features of real networks and reveals the internal organization of the nodes. Many algorithms have been proposed but the crucial issue of testing, i.e., the question of how good an algorithm is, with respect to others, is still open. Standard tests include the analysis of simple artificial graphs with a built-in community structure, that the algorithm has to recover. However, the special graphs adopted in actual tests have a structure that does not reflect the real properties of nodes and communities found in real networks. Here we introduce a class of benchmark graphs, that account for the heterogeneity in the distributions of node degrees and of community sizes. We use this benchmark to test two popular methods of community detection, modularity optimization, and Potts model clustering. The results show that the benchmark poses a much more severe test to algorithms than standard benchmarks, revealing limits that may not be apparent at a first analysis.

DOI: [10.1103/PhysRevE.78.046110](https://doi.org/10.1103/PhysRevE.78.046110)

PACS number(s): 89.75.Hc, 89.75.Kd

I. INTRODUCTION

Many complex systems in nature, society, and technology display a modular structure, i.e., they appear as a combination of compartments that are fairly independent of each other. In the graph representation of complex systems [1,2], where the elementary units of a system are described as nodes and their mutual interactions as links, such modular structure is revealed by the existence of groups of nodes, called communities or modules, with many links connecting nodes of the same group and comparatively few links joining nodes of different groups [3,4]. Communities reveal a non-trivial internal organization of the network, and allow us to infer special relationships between the nodes that may not be easily accessible from direct empirical tests. Communities may be groups of related individuals in social networks [3,5], sets of Web pages dealing with the same topic [6], biochemical pathways in metabolic networks [7,8], etc.

Detecting communities in networks is a big challenge. Many methods have been devised over the last few years, within different scientific disciplines such as physics, biology, computer, and social sciences. This race towards the ideal method aims at two main goals, i.e., improving the accuracy in the determination of meaningful modules and reducing the computational complexity of the algorithm. The latter is a well defined objective: in many cases it is possible to compute analytically the complexity of an algorithm, in others one can derive it from simulations of the algorithm on systems of different sizes. The main problem is then to estimate the accuracy of a method and to compare it with other methods. This issue of testing is in our opinion as crucial as devising powerful algorithms, but until now it has not received the attention it deserves.

Testing an algorithm essentially means analyzing a network with a well-defined community structure and recovering its communities. Ideally, one would like to have many instances of real networks whose modules are precisely known, but this is unfortunately not the case. Therefore, the most extensive tests are performed on computer generated networks, with a built-in community structure. The most fa-

mous benchmark for community detection is a class of networks introduced by Girvan and Newman (GN) [3]. Each network has 128 nodes, divided into four groups with 32 nodes each. The average degree of the network is 16 and the nodes have approximately the same degree, as in a random graph. At variance with a random graph, nodes tend to be connected preferentially to nodes of their group: a parameter k_{out} indicates what is the expected number of links joining each node to nodes of different groups (external degree). When $k_{\text{out}} < 8$ each node shares more links with the other nodes of its group than with the rest of the network. In this case, the four groups are well defined communities and a good algorithm should be able to identify them.

This benchmark is regularly used to test algorithms. However, there are several caveats that one has to consider: all nodes of the network have essentially the same degree, the communities are all of the same size, and the network is small. The first two remarks indicate that the GN benchmark cannot be considered a proxy of a real network with community structure.

Real networks are characterized by heterogeneous distributions of node degree, whose tails often decay as power laws. Such heterogeneity is responsible for a number of remarkable features of real networks, such as resilience to random failures/attacks [9], and the absence of a threshold for percolation [10] and epidemic spreading [11]. Therefore, a good benchmark should have a skewed degree distribution, similar to real networks. Likewise, it is not correct to assume that all communities have the same size: the distribution of community sizes of real networks is also broad, with a tail that can be fairly well approximated by a power law [8,12–14]. A reliable benchmark should include communities of very different sizes. A variant of the GN benchmark with communities of different size was introduced in Ref. [15]. Finally, the GN benchmark was a network of a reasonable size for most existing algorithms at the time when it was introduced. Nowadays, there are methods able to analyze graphs with millions of nodes [14,16,17] and it is not appropriate to compare their performances on small graphs. In general, an algorithm should be tested on benchmarks of variable size and average degree, as these parameters may

seriously affect the outcome of the method, and reveal its limits, as we shall see.

In this paper we propose a realistic benchmark for community detection, that accounts for the heterogeneity of both degree and community size. Detecting communities on this class of graphs is a challenging task, as shown by applying well known community detection algorithms.

II. THE BENCHMARK

We assume that both the degree and the community size distributions are power laws, with exponents γ and β , respectively. The number of nodes is N , the average degree is $\langle k \rangle$.

In the GN benchmark a node may happen to have more links outside than inside its community even when $k_{\text{out}} < 8$, due to random fluctuations, which raises a conceptual problem concerning the natural classification of the node. The construction of a realization of our benchmark proceeds through the following steps.

(1) Each node is given a degree taken from a power law distribution with exponent γ . The extremes of the distribution k_{min} and k_{max} are chosen such that the average degree is $\langle k \rangle$. The configuration model [18] is used to connect the nodes so to keep their degree sequence.

(2) Each node shares a fraction $1 - \mu$ of its links with the other nodes of its community and a fraction μ with the other nodes of the network; μ is the mixing parameter.

(3) The sizes of the communities are taken from a power law distribution with exponent β , such that the sum of all sizes equals the number N of nodes of the graph. The minimal and maximal community sizes s_{min} and s_{max} are chosen so to respect the constraints imposed by our definition of community: $s_{\text{min}} > k_{\text{min}}$ and $s_{\text{max}} > k_{\text{max}}$. This ensures that a node of any degree can be included in at least a community.

(4) At the beginning, all nodes are homeless, i.e., they are not assigned to any community. In the first iteration, a node is assigned to a randomly chosen community; if the community size exceeds the internal degree of the node (i.e., the number of its neighbors inside the community), the node enters the community, otherwise it remains homeless. In successive iterations we place a homeless node to a randomly chosen community: if the latter is complete, we kick out a randomly selected node of the community, which becomes homeless. The procedure stops when there are no more homeless nodes.

(5) To enforce the condition on the fraction of internal neighbors expressed by the mixing parameter μ , several rewiring steps are performed, such that the degrees of all nodes stay the same and only the split between internal and external degree is affected, when needed. In this way the ratio between external and internal degree of each node in its community can be set to the desired share μ with good approximation (see Fig. 1).

The prescription we have given leads to fast convergence. In Fig. 2 we show how the time to completion scales with the number of links of the graphs. The latter is expressed by the average degree, as the number of nodes of the graphs is kept fixed. The curves clearly show a linear relation between the

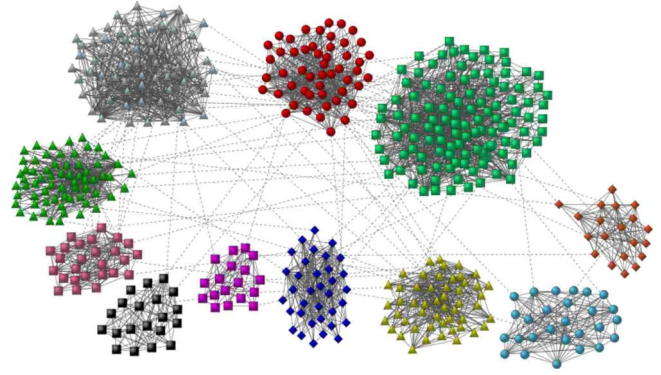


FIG. 1. (Color online) A realization of the new benchmark, with 500 nodes.

computer time and the number of links of the graph. Therefore our procedure allows one to build fairly large networks (up to $10^5 - 10^6$ nodes) in a reasonable time.

Due to the strong constraints we impose to the system, in some instances convergence may not be reached. However,

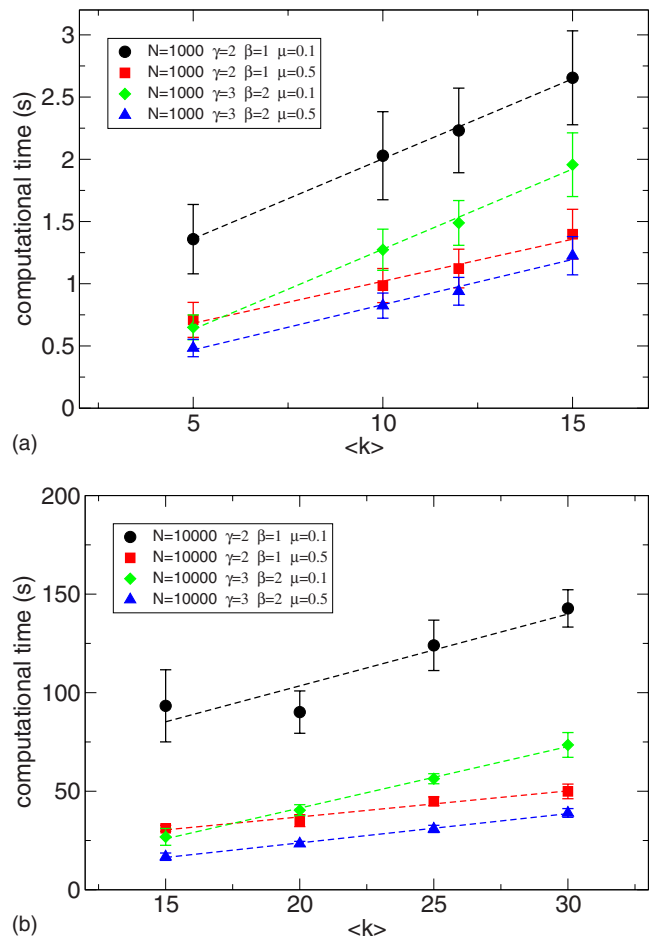


FIG. 2. (Color online) Study of the complexity of our algorithm. The plots show the scaling of the computer time (in s) with the average degree of the graph. The curves correspond to different choices for the exponents γ and β and the value of μ . The two panels reproduce graphs with 1000 (a) and 10000 nodes (b). The calculations were performed on Opteron processors.

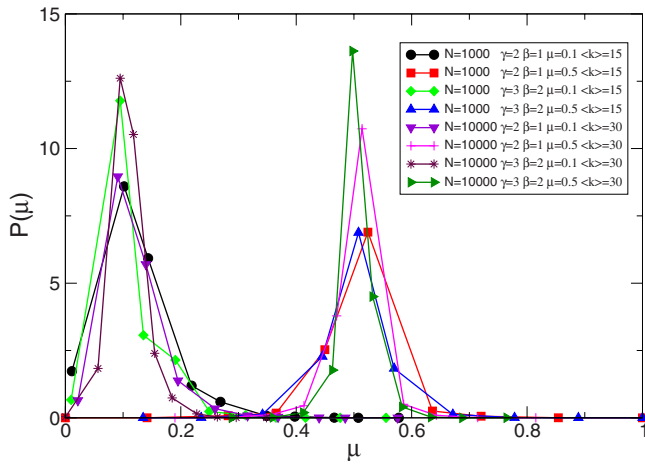


FIG. 3. (Color online) Distribution of the μ values for benchmark graphs obtained with our algorithm for different choices of the exponents and system size.

this is very unlikely for the range of parameters we have used. For the exponents we have taken typical values of real networks: $2 \leq \gamma \leq 3$, $1 \leq \beta \leq 2$.

Our algorithm tries to set the μ value of each node to the predefined input value, but of course this does not work in general, especially for nodes of small degree, where the possible values of μ are just a few and clearly separated. So, the distribution of μ values for a given benchmark graph cannot be a δ function, but it will have a bell-shaped curve, with a pronounced peak (Fig. 3).

III. TESTS

We have used our benchmark to test the performance of two methods to detect communities in networks, i.e., modularity optimization [7,19,20], probably the most popular method of all, and the algorithm based on the Potts model introduced by Reichardt and Bornholdt [21]. For modularity, the optimization was carried out through simulated annealing, as in Ref. [7], which is not a fast technique but yields

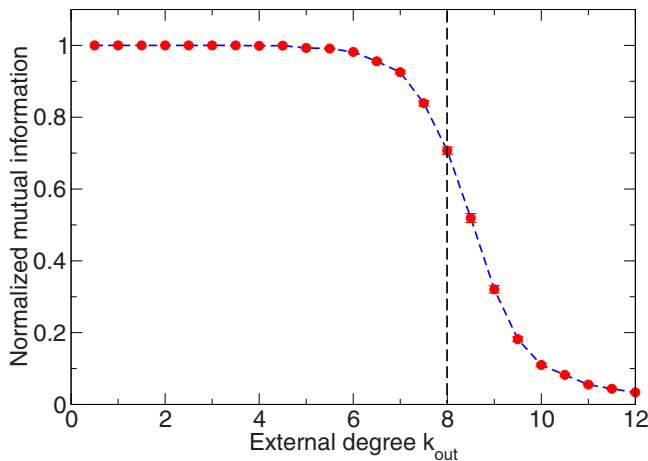


FIG. 4. (Color online) Test of modularity optimization on the benchmark of Girvan and Newman.

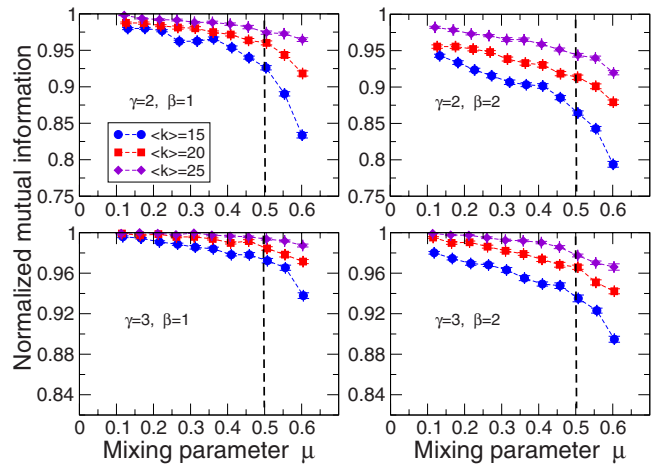


FIG. 5. (Color online) Test of modularity optimization on the new benchmark. The number of nodes $N=1000$. The results clearly depend on all parameters of the benchmark, from the exponents γ and β to the average degree $\langle k \rangle$. The threshold $\mu_c=0.5$ (dashed vertical line in the plots) marks the border beyond which communities are no longer defined in the strong sense, i.e., such that each node has more neighbors in its own community than in the others [23]. Each point corresponds to an average over 100 graph realizations.

good estimates of modularity maxima. In Fig. 4 we plot the performance of the method as a function of the external degree of the nodes for the GN benchmark.

To compare the built-in modular structure with the one delivered by the algorithm we adopt the normalized mutual information, a measure of similarity of partitions borrowed from information theory, which has proved to be reliable [22]. As we can see from the figure, the natural partition is always found up until $k_{out}=6$, then the method starts to fail, although it finds good partitions even when communities are fuzzy ($k_{out} \geq 8$). Meanwhile, many algorithms are able to achieve comparable performances, so the benchmark can hardly discriminate between different methods. As we can

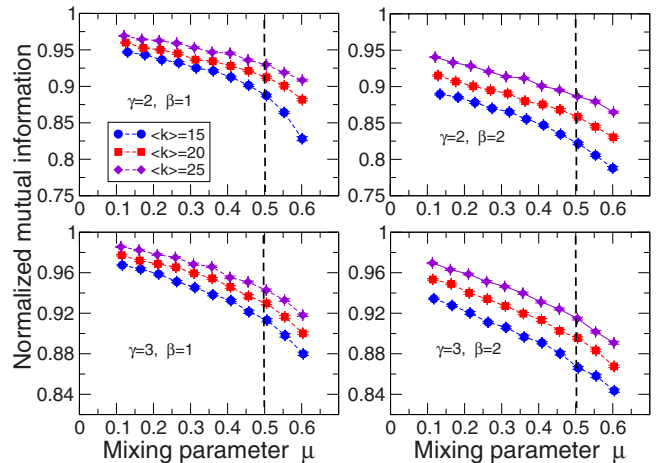


FIG. 6. (Color online) Test of modularity optimization on the new benchmark. The number of nodes is now $N=5000$, the other parameters are the same as in Fig. 5. Each point corresponds to an average over 25 graph realizations.

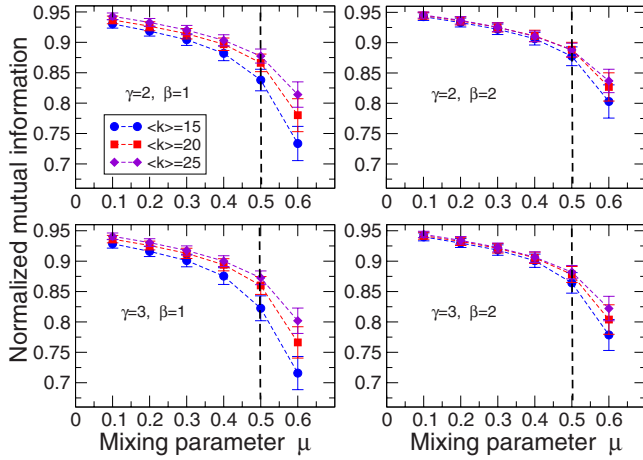


FIG. 7. (Color online) Test of Potts model clustering on the new benchmark. The number of nodes $N=1000$. The results clearly depend on all parameters of the benchmark, from the exponents γ and β to the average degree $\langle k \rangle$. Each point corresponds to an average over 100 graph realizations.

see from the figure, for $k_{\text{out}} < 8$ we are close to the top performance and there seems to be little room for improvement.

In Fig. 5 we show what happens if one optimizes modularity on the new benchmark, for $N=1000$. The four panels correspond to four pairs for the exponents $(\gamma, \beta) = (2, 1), (2, 2), (3, 1), (3, 2)$. We have chosen combinations of the extremes of the exponents' ranges in order to explore the widest spectrum of graph structures. For each pair of exponents, we have used three values for the average degree $\langle k \rangle = 15, 20, 25$. Each curve shows the variation of the normalized mutual information with the mixing parameter μ .

In general, from Fig. 5 we can infer that the method gives good results. However, we find that it begins to fail even when communities are only loosely connected to each other (small μ). This is due to the fact that modularity optimization has an intrinsic resolution limit that makes small communities hard to detect [24]. Our benchmark is able to disclose this limit. We have explicitly verified that the modularity of the natural partition of the graph is lower than the maximum obtained from the optimization, and that the partition found by the algorithm has systematically a smaller number of clusters, due to the merge of small communities into larger groups.

We also see that the performance of the method is better the larger the average degree $\langle k \rangle$, whereas it gets worse when the communities are more similar to each other in size (larger β). To check how the performance is affected by the network size, we have tested the method on a set of larger graphs (Fig. 6). Now $N=5000$, whereas the other parameters are the same as before. Curves corresponding to the same parameters are similar, but shifted towards the bottom for larger systems. We conclude that the performance of the method worsens if the size of the graph increases. If we consider that

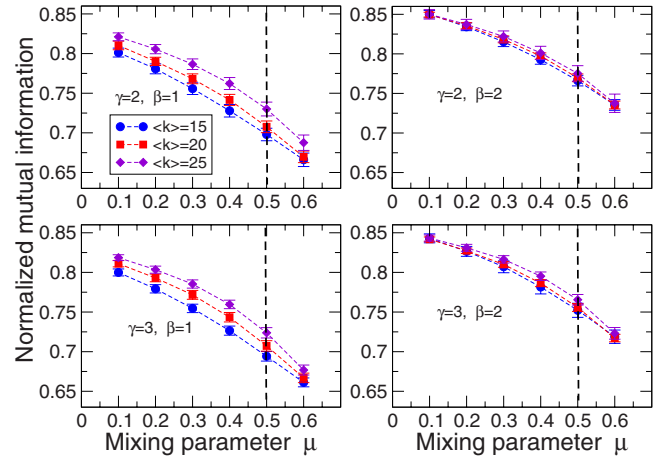


FIG. 8. (Color online) Test of Potts model clustering on the new benchmark. The number of nodes $N=5000$, the other parameters are the same as in Fig. 7. Each point corresponds to an average over ten graph realizations.

networks with 5000 nodes are much smaller than many graphs one would like to analyze, modularity optimization may give inaccurate results in practical cases, something which could not be inferred from tests on existing benchmarks.

We have repeated the same analysis for the Potts model algorithm. We closely followed the implementation suggested by the authors of Ref. [21]: we set the number of spin states equal to the number of nodes of the network, the ferromagnetic coupling J was set to 1, whereas the antiferromagnetic coupling γ equals the density of links of the network. The results are shown in Figs. 7 and 8. The performance of the method is fair, and it worsens for larger system sizes, as for modularity optimization, which proves superior.

IV. SUMMARY

We have introduced a class of graphs to test algorithms identifying communities in networks. These graphs extend the GN benchmark by introducing features of real networks, i.e., the heterogeneity in the distributions of node degree and community size. We found that these elements pose a harder test to existing methods. We have tested modularity optimization and a clustering technique based on the Potts model against the benchmark. From the results the resolution limit of modularity emerges immediately. Furthermore, we have seen that the size of the graph and the density of its links have a sizeable effect on the performance of the algorithm, so it is very important to study this dependence when testing a new algorithm. The new benchmark is suitable for this type of analysis, as the graphs can be constructed very quickly, and one can span several orders of magnitude in network size [25].

- [1] M. E. J. Newman, *SIAM Rev.* **45**, 167 (2003).
- [2] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D.-U. Hwang, *Phys. Rep.* **424**, 175 (2006).
- [3] M. Girvan and M. E. J. Newman, *Proc. Natl. Acad. Sci. U.S.A.* **99**, 7821 (2002).
- [4] S. Fortunato and C. Castellano, in *Encyclopedia of Complexity and System Science*, edited by B. Meyers (Springer, Heidelberg, 2009).
- [5] D. Lusseau and M. E. J. Newman, *Proc. R. Soc. London, Ser. B* **271**, S477 (2004).
- [6] G. W. Flake, S. Lawrence, C. Lee Giles, and F. M. Coetzee, *Comput. Sci. Eng.* **35**, 66 (2002).
- [7] R. Guimerà and L. A. N. Amaral, *Nature (London)* **433**, 895 (2005).
- [8] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, *Nature (London)* **435**, 814 (2005).
- [9] R. Albert, H. Jeong, and A.-L. Barabási, *Nature (London)* **406**, 378 (2000).
- [10] R. Cohen, K. Erez, D. ben-Avraham, and S. Havlin, *Phys. Rev. Lett.* **85**, 4626 (2000).
- [11] R. Pastor-Satorras and A. Vespignani, *Phys. Rev. Lett.* **86**, 3200 (2001).
- [12] R. Guimerà, L. Danon, A. Díaz-Guilera, F. Giralt, and A. Arenas, *Phys. Rev. E* **68**, 065103 (R) (2003).
- [13] L. Danon, J. Duch, A. Arenas, and A. Díaz-Guilera, in *Large Scale Structure and Dynamics of Complex Networks: From Information Technology to Finance and Natural Science*, edited by G. Caldarelli and A. Vespignani (World Scientific, Singapore, 2007), pp. 93–114.
- [14] A. Clauset, M. E. J. Newman, and C. Moore, *Phys. Rev. E* **70**, 066111 (2004).
- [15] L. Danon, A. Díaz-Guilera, and A. Arenas, *J. Stat. Mech.: Theory Exp.* 2006 P11010.
- [16] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, e-print arXiv:0803.0476.
- [17] A. Lancichinetti, S. Fortunato, and J. Kertész, e-print arXiv:0802.1218.
- [18] M. Molloy and B. Reed, *Combinatorics, Probab. Comput.* **6**, 161 (1995).
- [19] M. E. J. Newman, *Phys. Rev. E* **69**, 066133 (2004).
- [20] J. Duch and A. Arenas, *Phys. Rev. E* **72**, 027104 (2005).
- [21] J. Reichardt and S. Bornholdt, *Phys. Rev. Lett.* **93**, 218701 (2004).
- [22] L. Danon, A. Díaz-Guilera, J. Duch, and A. Arenas, *J. Stat. Mech.: Theory Exp.* 2005 P09008.
- [23] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi, *Proc. Natl. Acad. Sci. U.S.A.* **101**, 2658 (2004).
- [24] S. Fortunato and M. Barthélemy, *Proc. Natl. Acad. Sci. U.S.A.* **104**, 36 (2007).
- [25] A software package to generate the benchmark graphs can be downloaded from <http://santo.fortunato.googlepages.com/benchmark.tgz>