Ants Crawling to Discover the Community Structure in Networks

Mariano Tepper and Guillermo Sapiro[⋆]

Department of Electrical and Computer Engineering, Duke University, USA.
mariano.tepper,guillermo.sapiro@duke.edu

Abstract. We cast the problem of discovering the community structure in networks as the composition of community candidates, obtained from several community detection base algorithms, into a coherent structure. In turn, this composition can be cast into a maximum-weight clique problem, and we propose an ant colony optimization algorithm to solve it. Our results show that the proposed method is able to discover better community structures, according to several evaluation criteria, than the ones obtained with the base algorithms. It also outperforms, both in quality and in speed, the recently introduced FG-Tiling algorithm.

1 Introduction

Networks are frequently used to describe many real-life scenarios were units interact with each other (e.g., see [1,2] and references therein). A seemingly common property to many networks is *community structure*, i.e., that is, networks can be divided into groups such that intra-group connections are denser than inter-group ones. The ability to find and analyze these communities sheds light on important characteristics of a network. However, the best way to establish the community structure is still disputed. Addressing this is the topic of this work.

Let $\mathcal{G} = (\mathcal{U}, \mathcal{E})$ be the graph to analyze, where $\mathcal{U}$ is the set of nodes and $\mathcal{E}$ is the set of edges (in the following we indistinguishably use the terms graph and network). Generically, we consider that a community-detection algorithm provides a set $\mathcal{X}$ of candidate communities ($\mathcal{X}$ can be a partition of the node set $\mathcal{U}$ or a hierarchy of subsets in $\mathcal{U}$, from which the best partition may latter be extracted). Let us consider that a pool of C such algorithms provides a universe of candidates $V = \bigcup_{i=1}^{C} \mathcal{X}_i$. We build a new graph $G = (V, E, \omega)$, where $(u, v) \in E$ if and only if communities $u, v \in V$ do not overlap, and $\forall u \in V \; \omega(u)$ is a measure associated with the quality of partition u. In this work, we do not address the case in which communities overlap, while our technique could be considered for this as well.

We formulate the problem of finding the best community structure in a network as a *patchwork* algorithm: instead of building a community set $\mathcal{X}$ from scratch, we browse through V and build a new solution by combining the best

[⋆] Work partially supported by NSF, ONR, NGA, ARO, DARPA, and NSSEFF. Work partially done while the authors were with the Department of Electrical and Computer Engineering, University of Minnesota.

communities in V in such a way that no overlap exists among them. In other words, the proposed *meta* algorithm selects the set of communities with maximum total weight among the ones that are fully interconnected by E. This is a maximum-weight clique problem on the graph G. This type of formulation was simultaneously introduced in [3,4] for image segmentation. In this work, we propose to follow and extend this approach for community structure discovery.[1]

Let us first formally pose the maximum-weight clique (MWC) problem. Let $G = (V, E, \omega)$ be a weighted graph where V is a set of nodes, $E \subseteq V \times V$ is a set of edges (no self-loops are allowed, that is $\forall u \in V, (u, u) \notin E$), and $\omega : V \to \mathbb{R}^+$ is a node weighting function. For convenience, given a set $C \subseteq V$ we write $\omega(C) = \sum_{u \in C} \omega(u)$. A clique $\mathcal{C} \subseteq V$ is a set such that $\forall u, v \in \mathcal{C}, u \neq v, (u, v) \in E$. The maximum clique problem is to find a clique of maximum cardinality in G. The MWC problem is to find a clique $\mathcal{C}$ of maximum weight $\omega(\mathcal{C})$. Both problems are known to be NP-Hard.

A popular approach for solving hard discrete combinatorial problems such as MWC is the use of metaheuristics. Ant Colony Optimization (ACO) is a variant of swarm intelligence in which the system is made of a population of simple agents interacting locally with one another and with their environment. Although each agent builds a solution following an extremely simple set of rules, cooperation among the population leads to the emergence of intelligent behavior. When looking for food, real ants deposit pheromones on the ground; then, the probability that other ants follow a particular path is proportional to the level of pheromones in that path. Similarly, the solution construction process in ACO is stochastic and is biased by a "pheromone" model. Specifically, artificial ants will explore more thoroughly regions of the solution space where good solutions were previously found.

In this work we propose an ACO algorithm for the MWC problem and, as already discussed, we apply it to the problem of selecting the best community structure from a set of community candidates. We propose then a double collaboration structure, where multiple algorithms collaborate to propose community structures, and the ants collaborate to find the best one from all the possibilities. The algorithm presents a good balance between the amount of exploration and computational efficiency. We provide experiments that show the pertinence of the proposed method.

The remainder of the work is organized as follows. In Section 2 we present our ACO algorithm for the MWC problem. In Section 3 we discuss the experimental results and finally we provide some closing remarks in Section 4.

2 ACO for the MWC problem

A few variants of ACO have been proposed for the maximum clique problem [6,7]. We adapt Solnon and Fenet's algorithm [7] for solving the MWC problem. Being a metaheuristic, the change is minimal: instead of searching for the clique with maximum cardinality, we seek the clique with maximum weight.

[1] An approach based on metaheuristics was indepently and simultaneously developed for clustering in [5], solving the related maximum-weight independent set problem using simulated annealing.

We associate a pheromone level $\tau(v)$ to each node $v \in V$, and use an ACO variant, called Max-Min Ant System (MMAS), where pheromone levels are bounded, that is $\forall v \in V,\ \tau(v) \in [\tau_{\min}, \tau_{\max}]$. These bounds ensure that no region in the solution space will be excessively over or under sampled.

Given a clique $\mathcal{C}$, we define the set of possible additions to $\mathcal{C}$ as

$$\mathrm{PA}(\mathcal{C}) = \{u \mid u \in (V \smallsetminus \mathcal{C}) \wedge \forall v \in \mathcal{C}, (u,v) \in E\}\,. \tag{1}$$

That is, given a clique $\mathcal{C}$, $\forall u \in \mathrm{PA}(\mathcal{C}), \mathcal{C} \cup \{u\}$ is also a clique.

The overall ACO procedure is described by Algorithm 1. In each iteration, a certain number K of ants explore the solution space. Each ant builds a randomized solution, favoring nodes with higher pheromone levels.

The ACO algorithm regulates its overall behavior through pheromone trails: cooperation between ants rises from the optimality of previous experiences. When adding a new node to a partial solution, the pheromone level in each node is used to bias the election, i.e., the higher $\tau(v)$, the more probably a node v will be chosen. An ant then selects a node v with probability

$$p_\alpha(v \mid \mathcal{C}) = [\tau(v)]^\alpha \left(\textstyle\sum_{u \in \mathrm{PA}(\mathcal{C})} [\tau(u)]^\alpha\right)^{-1}, \tag{2}$$

where α is a parameter of the algorithm.

As in every ACO algorithm, pheromones evaporate over time. This ensures that exploration will not get stuck around good previous solutions and will diversify over time. We set the evaporation rule

$$\forall u \in V,\ \tau(u) = \max(\tau_{\min}\ ,\ \rho \cdot \tau(u)), \tag{3}$$

where ρ is a parameter of the algorithm.

As previously stated, pheromone levels must be increased for those nodes that belong to good solutions. Let $\mathcal{C}^*$ be the best solution in the current iteration and let $\mathcal{C}_{\text{best}}$ be the best solution in all previous iterations, including the current one. We only update those nodes that belong to $\mathcal{C}^*$, according to the rule

$$\tau(u) = \min\left(\tau_{\max}\ ,\ \tau(u) + [1 + \omega(\mathcal{C}_{\text{best}}) - \omega(\mathcal{C}^*)]^{-1}\right). \tag{4}$$

2.1 Local search

The ACO algorithm uses local search as a post-processing to improve the best solution found by the ants. Adding this intelligence to every ant would be computationally costly and would hinder the stochastic search. Notice that the ACO algorithm uses the local search method as a black box and, as such, any suitable technique can be employed. Many local search schemes have been proposed for the maximum clique problem. Their goal is to avoid local minima by exploring the neighborhood (in the solution space) of an initial solution.

Katayama et al. [8] propose to examine the k-opt neighborhood of an initial solution, defined as the set of neighbors that can be obtained by a sequence of several add and drop moves that are adaptively changed in the feasible search space. For this task, they introduce an efficient algorithm called k-opt local search (KLS). KLS has proven capable of finding satisfactory cliques with reasonable running times. In this work, we extend KLS for the MWC problem. Pseudocode of the weighted-KLS is presented in Algorithm 2.

Algorithm 1: ACO algorithm for the MWC problem.

```
$\mathcal{C}_{\text{best}} \leftarrow \emptyset$;   $\forall u \in V, \tau(u) \leftarrow \tau_{\max}$;
while number of iterations is lower than $T$ do
    for ant $k$ such that $1 \leq k \leq K$ do
        Choose a seed $u_k \in V$ at random;
        $\mathcal{C}_k \leftarrow \{u_k\}$;
        while $\text{PA}(\mathcal{C}_k) \neq \emptyset$ do
            Choose a vertex $v \in \text{PA}(\mathcal{C}_k)$ with probability $p_\alpha(v \mid \mathcal{C}_k)$;
            $\mathcal{C}_k \leftarrow \mathcal{C}_k \cup \{v\}$;
    $\mathcal{C}_{\max} \leftarrow \text{argmax}_{\mathcal{C}_k}\ \omega(\mathcal{C}_k)$;
    $\mathcal{C}^* \leftarrow$ improve solution $\mathcal{C}_{\max}$ using local search (see Sec. 2.1 and Algorithm 2);
    if $\omega(\mathcal{C}^*) > \omega(\mathcal{C}_{\text{best}})$ then $\mathcal{C}_{\text{best}} \leftarrow \mathcal{C}^*$;
    Evaporate pheromone levels (Eq. (3));
    Update pheromone levels of vertices in $\mathcal{C}^*$ (Eq. (4));
return $\mathcal{C}_{\text{best}}$
```

Algorithm 2: Local search.

```
repeat
    $P \leftarrow V$;   $\mathcal{C}_{\text{prev}} \leftarrow \mathcal{C}$;   $D \leftarrow \mathcal{C}$;   $g \leftarrow 0$;   $g_{\max} \leftarrow 0$;
    while $D \neq \emptyset$ do
        if $\text{PA}(\mathcal{C}) \cap P \neq \emptyset$ then // add phase
            $v^* \leftarrow \underset{v \in \text{PA}(\mathcal{C})}{\text{argmax}}\ \text{input}_\omega(\mathcal{C}, v)$;
            $\mathcal{C} \leftarrow \mathcal{C} \cup \{v^*\}$;   $g \leftarrow g + \omega(v^*)$;
            $P \leftarrow P \setminus \{v^*\}$;
            if $g > g_{\max}$ then
                $g_{\max} \leftarrow g$;   $\mathcal{C}_{\text{best}} \leftarrow \mathcal{C}$
        else // drop phase
            $v^* \leftarrow \underset{v \in (\mathcal{C} \cap P)}{\text{argmax}}\ \omega(\text{PA}(\mathcal{C} \setminus \{v\}))$;
            $\mathcal{C} \leftarrow \mathcal{C} \setminus \{v^*\}$;   $P \leftarrow P \setminus \{v^*\}$;
            $g \leftarrow g - \omega(v^*)$;
            if $v^* \in \mathcal{C}_{\text{prev}}$ then $D \leftarrow D \setminus \{v^*\}$
until $g_{\max} > 0$;
return $\mathcal{C}$
```

KLS is a greedy algorithm, each decision of adding or removing a node from the current solution is made by maximizing the immediately obtained reward:

- When adding a node, the most desirable candidate in $\text{PA}(\mathcal{C})$ is picked. In the unweighted case, the desirability of a node is given by its degree; in the weighted case, we define it as
$$\forall u \in \text{PA}(\mathcal{C}),\ \text{input}_\omega(\mathcal{C}, u) = \omega(u) + \textstyle\sum_{v \in \text{PA}(\mathcal{C})} \omega(v). \tag{5}$$
- When removing a node, we pick the node whose removal will produce the most desirable set of candidates $\text{PA}(\mathcal{C})$ for future additions. In the absence of weights, the appeal of $\text{PA}(\mathcal{C})$ is determined by its size $|\text{PA}(\mathcal{C})|$; when weights are present, the appeal is defined by $\omega(\text{PA}(\mathcal{C}))$.

KLS uses the set P as a mechanism to avoid incurring in loops of addition/removal of the same set of nodes. Also note that the functions PA and input_ω can be computed and updated very efficiently [9], and do not present a significant computational overhead.

Complexity. Let us begin by analyzing the KLS algorithm. Due to its parameterless nature, computing its time complexity is not an easy task. We estimate the overall complexity as $O(Ln^2h)$, where $n = |V|$, h is the size of the initial (input) clique, and L is the number of executions of the outer cycle, although a more realistic *practical* bound would be $O(h)$. In the ACO algorithm, each ant k can build its own solution in $O(\deg(u_k)^2)$, where $\deg(u_k)$ denotes the degree of seed u_k. Thus the total complexity of the proposed algorithm is $O(T(Kd^2 + h))$, where $d = \max_{u \in V} \deg(u)$, if we use the aforementioned $O(h)$ for KLS.

The complexity of the ACO algorithm is extremely lower than the $O(Nd^3)$ of FG-Tiling [4]. This can be observed in practice, where our algorithm systematically outperformed FG-Tiling in running-time. Notice that FG-Tiling is deterministic, thus its worst case complexity provides a tight bound.

Community quality measures. Many different measures have been used for assessing the quality of a given community structure. We work with the standard modularity [2] and a new measure called surprise [10]. Our method is of course completely agnostic to the measure's nature and it can directly profit from any improvement in this respect.

Modularity is by far the most popular measure for assessing the quality of a partition. We will denote by $Q(\mathcal{P})$ the modularity of partition $\mathcal{P}$. Because of space constraints, we do not provide its formal definition.

The surprise S of partition $\mathcal{P}$ is defined [10] as $S(\mathcal{P}) = -\log H(F, N, m, b)$, where H is the tail of the hypergeometric distribution, $F = n(n-1)/2$, $N = \sum_{P \in \mathcal{P}} |P|(|P|-1)/2$, and $b = \sum_{P \in \mathcal{P}} e_P$. This represents the probability of obtaining b intra-community edges in m draws, without replacement, from a finite population of size F containing N successes. We can use surprise in our framework assuming that the subsets in a random partition are i.i.d., that is, $S(\mathcal{P}) \approx \sum_{P \in \mathcal{P}} -\log H(F, N_P, m, e_P)$, where $N_P = |P|(|P|-1)/2$.

3 Experimental results

In this section we evaluate the community structure discovery results of the proposed ACO algorithm for the MWC problem. We use two different base algorithms for community detection: Walktrap [11] and Jerarca [1]. Both algorithms provide a hierarchy of communities and then globally threshold the hierarchy at different levels, thus obtaining a partition per level, and finally select the best partition among them. Let $\mathcal{H}_W$ and $\mathcal{H}_J$ be the hierarchies provided by Walktrap and Jerarca, respectively. As already explained, we create the graph $G = (V, E)$, where $V = \left(\bigcup_{C \in \mathcal{H}_W} C\right) \cup \left(\bigcup_{C \in \mathcal{H}_J} C\right)$ and $(C, C') \in E$ if and only if $C \cap C' = \emptyset$. We then look for the MWC in G. We compare our results with FG-Tiling [4], a recently introduced and successful MWC solver in the context of image segmentation. For all experiments, unless specifically indicated, we set the ACO parameters to $\alpha = 1$, $\tau_{\min} = 0.01$, $\tau_{\min} = 10$, $\rho = 0.98$, $K = 100$, and $T = 1000$.

In Table 1 we show the community structure discovery results on different networks used in the literature. We first observe that the proposed approach, finding an approximate solution of the MWC problem, outperforms the hierarchical approach. Indeed, FG-Tiling and ACO outperform Jerarca and Walktrap. Of course, this comes at the cost of extended running times, Jerarca and Walktrap being very fast algorithms. On a finer observation level, ACO consistently obtains better solutions than the deterministic FG-Tiling, confirming in practice the ability of adaptive stochastic algorithms for exploring the solution space. We provide graph plots in Fig. 1 to show that the differences in the measures shown in Table 1 actually correspond to different community structures. Note that in no experiment we judge the quality of the obtained partition from the graph plots, the assessment is solely based on the selected standard measure.

Table 2 presents running-time examples of both algorithms (implemented in Python). Note that all of our current implementations can be further optimized. These times do not reflect the best times that can be achieved with these methods, but they serve to corroborate the previously presented complexity bounds.

Table 1: Modularity $Q(\mathcal{P})$ and surprise $S(\mathcal{P})$ values of the best partition $\mathcal{P}$ found by each tested method on different networks (WT stands for Walktrap).

	Modularity				Surprise			
	Jerarca	WT	FG-Tiling	ACO	Jerarca	WT	FG-Tiling	ACO
1dc.64[1]	0.212	0.251	0.251	**0.251**	148.65	146.06	156.4	**158.95**
Les Misérables [12]	0.490	0.403	0.497	**0.512**	361.14	325.98	378.91	**378.91**
Chesapeake [13]	0.356	0.102	0.339	**0.356**	41.26	14.16	45.59	**46.00**
Dolphins [14]	0.246	0.445	0.434	**0.461**	144.40	90.00	122.55	**152.79**
Aegean34 [15]	0.529	0.571	0.571	**0.571**	130.97	126.78	134.18	**134.18**

[1] http://www2.research.att.com/~njas/doc/graphs.html

Table 2: Running time (in seconds) of FG-Tiling and ACO for different networks.

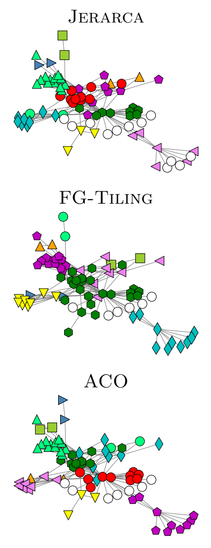
	Aegean34	Chesapeake	Les Misérables	1dc.64
FG-Tiling	975	1200	37790	48630
ACO	109	130	748	930

We also ran an experiment comparing three different versions of the proposed ACO algorithm: one that disallows cooperation and uses local search, ones that allows cooperation but does not use local search, and the proposed one which allows cooperation and uses local search. The emergence of cooperation can be prevented by setting $\alpha = 0$, see Eq. (2). We ran this comparison on a random graph $\mathcal{G}$ where all edges are i.i.d. and belong to $\mathcal{G}$ with a fixed probability. In average there should be no communities in $\mathcal{G}$, and hence no trivial solution can be prematurely found. This is observed in practice since all modularity values are close to zero. The results are shown in Fig. 2. Clearly, the version that allows both collaboration and local search outperforms the other variants.

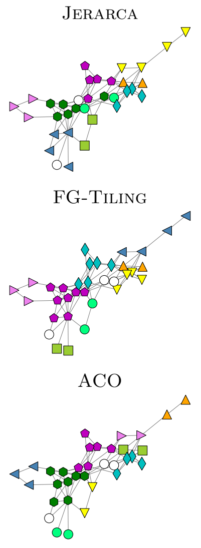
To show the generality of the proposed approach, we include some basic results on image segmentation in Fig. 3. Briefly, each region in a segmentation can be viewed as a "community" and we compute its weight using a basic and untuned version of the method in [3]. Let $\mathcal{H}_{\text{UCM}}$ be the hierarchy produced by the state-of-the-art UCM image segmentation algorithm [16]. We select the global threshold on $\mathcal{H}_{\text{UCM}}$ such that the resulting partition has maximum total weight. When G is built from $\mathcal{H}_{\text{UCM}}$ in our framework, the proposed algorithm is able to improve the segmentation, obtaining a partition with higher total weight. More tuned or sophisticated weighting functions would further improve the obtained results, these are obtained directly from the proposed general technique.

4 Conclusions

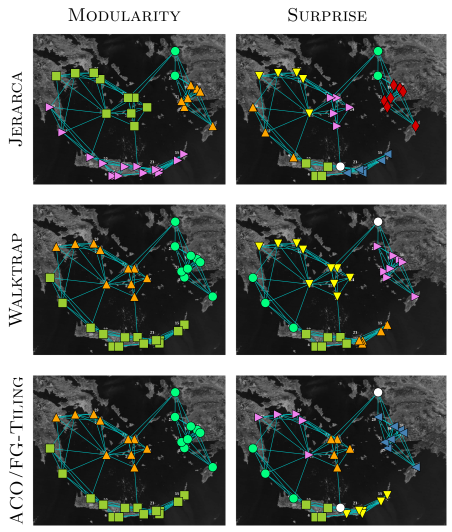
We proposed an algorithm for community structure discovery. Instead of building our own custom community structure from scratch, we take the output of several community detection algorithms (they can be the same algorithm executed with different parameters, and also provide hierarchies or partitions), and compose a new structure by combining the best communities in each. In this way, we can make use of multiple algorithms that provide globally suboptimal solutions which contain some optimal communities. The combination of these optimal communities leads to the creation of a new and globally better solution.



(a) Les Misérables (with modularity).



(b) Chesapeake (with surprise).



(c) Aegean34 network

Fig. 1: Different communities are represented by nodes of different colors and shapes. All singleton communities (composed of one node) are depicted with white circles. Differences in the quality measure (see Table 1) create different community structures.

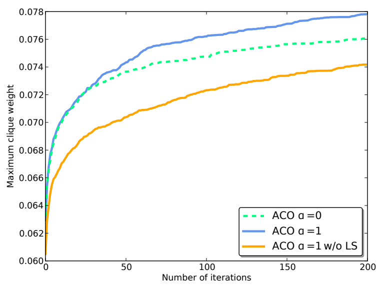


Fig. 2: The proposed algorithm benefits from both the collaborative nature of ACO and the local search (weighted KLS). It outperforms the non-cooperative ACO algorithm ($\alpha = 0$) and a version that does not use local search (termed "w/o LS"). Weights correspond to modularity values.

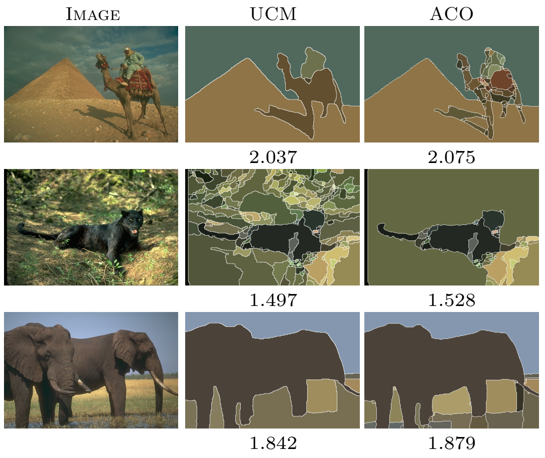


Fig. 3: Image segmentation results. We show the weight of the obtained partition. The proposed method is able to compose a better result by searching through the UCM hierarchy.

We showed results that confirm in practice the theoretical benefits of the proposed algorithm. Namely, the stochastic nature of ACO helps in handling the non-convex nature of the MWC problem's solution space, by exploring it "fairly." The collaborative aspect of ACO, ensures that "fairness" is distributed smartly: potentially appealing zones of the solution space attract more attention and are thus explored more thoroughly. We further strengthen the ACO algorithm by using a local search heuristic. We adapt the KLS algorithm to handle weighted graphs, obtaining both efficient and solid performances.

As future work, we plan on adding support for weighted edges in the model, which could be used to model relationships between communities. We also plan to extend our current model to the case of overlapping communities, for which suitable quality measures are needed. Additionally, a C++ implementation will allow to analyze larger networks. Furthermore, the ant exploration phase in the proposed ACO algorithm can be parallelized with ease.

References

1. Aldecoa, R., Marín, I.: Jerarca: efficient analysis of complex networks using hierarchical clustering. PLoS ONE **5**(7) (July 2010) e11585+
2. Newman, M.E.J., Girvan, M.: Finding and evaluating community structure in networks. Phys. Rev. E **E 69**(026113) (2004)
3. Brendel, W., Todorovic, S.: Segmentation as maximum-weight independent set. In: NIPS. (2010)
4. Ion, A., Carreira, J., Sminchisescu, C.: Image segmentation by figure-ground composition into maximal cliques. In: ICCV. (2011)
5. Li, N., Latecki, L.J.: Clustering aggregation as maximum-weight independent set. In: NIPS. (2012)
6. Leguizamon, G., Michalewicz, Z., Schutz, M.: An ant system for the maximum independent set problem. In: CACIC. Volume 2. (2001)
7. Solnon, C., Fenet, S.: A study of ACO capabilities for solving the maximum clique problem. J. Heuristics **12**(3) (May 2006) 155–180
8. Katayama, K., Hamamoto, A., Narihisa, H.: An effective local search for the maximum clique problem. Inf. Process. Lett. **95**(5) (September 2005) 503–511
9. Battiti, R., Protasi, M.: Reactive local search for the maximum clique problem. Algorithmica **29**(4) (April 2001) 610–637
10. Aldecoa, R., Marín, I.: Deciphering network community structure by surprise. PLoS ONE **6**(9) (September 2011) e24195+
11. Pons, P., Latapy, M.: Computing communities in large networks using random walks. J. Graph Algorithms Appl. **10**(2) (2004) 284–293
12. Knuth, D.E.: The Stanford GraphBase: A Platform for Combinatorial Computing. ACM, New York, NY, USA (1993)
13. Baird, D., Ulanowicz, R.E.: The seasonal dynamics of the Chesapeake bay ecosystem. Ecol. Monogr. **59**(4) (1989) 329–364
14. Lusseau, D., Schneider, K., Boisseau, O.J., Haase, P., Slooten, E., Dawson, S.M.: The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations. Behav. Ecol. Sociobiol. **54**(4) (2003) 396–405
15. Evans, T.S., Rivers, R.J., Knappett, C.: Interactions in space for archaeological models. Adv. Complex Syst. **15**(September) (2011) 1150009+
16. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: From contours to regions: an empirical evaluation. In: CVPR. (2009)