Semi-Supervised Overlapping Community Finding with Pairwise Constraints

Elham Alghamdi and Derek Greene

School of Computer Science & Informatics, University College Dublin
elham.alghamdi@ucdconnect.ie,derek.greene@ucd.ie

Abstract. In complex networks, we say that a network has community structure if subsets of its nodes form dense, highly-connected groups. Algorithms for detecting communities are generally unsupervised, relying solely on the network topology. However, such algorithms can often fail to uncover structure that reflects the underlying communities in the data, particularly when those communities are highly overlapping. One of the ways to improve accuracy is by harnessing additional background information (e.g. from domain experts), which can be used as a source of constraints to guide the community detection process. In this work, we explore the potential of semi-supervised strategies to improve algorithms for finding overlapping communities in networks. Specifically, we propose a greedy approach for finding communities using a limited number of pairwise constraints.

1 Introduction

Many applications of machine learning do not neatly correspond to the standard distinction between supervised and unsupervised learning [6]. In many domains, a limited degree of background knowledge will be available. Often supervision will take the form of pairwise constraints, which describe the relations between pairs of data objects. Such constraints have been used to guide and improve the usefulness of clustering algorithms [4]. The idea of semi-supervised learning also extends to the area of network analysis. Tasks such as community detection can potentially benefit from the introduction of limited supervision originating from individual domain experts or crowdsourcing, where this knowledge might be encoded as constraints indicating that a pair of nodes should always be assigned to the same community or should never be assigned to the same community. By harnessing this knowledge, we can potentially uncover communities of nodes which are difficult to identify when analyzing large or noisy networks.

Initial work in community detection focused on the development of algorithms which produce disjoint groups, where each node belongs one community [5]. However, in many real-world networks, nodes will naturally belong to multiple communities. In the case of both online and offline social networks, we can observe pervasive overlap where individuals belong to many highly-overlapping social groups [2]. More recently, overlapping community finding algorithms have been developed for application to these real networks [2, 16]. In contrast, work

on semi-supervised community finding continues to focus on cases where communities are required to be disjoint.

In this paper, we propose a semi-supervised approach for community finding, based on the concept of greedy clique expansion [16], which we refer to as Pairwise Constrained GCE (PC-GCE). We introduce must-link and cannot-link constraints to both the initialization phase of the process and to the subsequent community expansion process. Experimental evaluations on benchmark synthetic networks demonstrate that the introduction of a relatively small number of constraints can improve our ability to correctly uncover the underlying communities in these networks.

The remainder of this paper is structured as follows: Section 2 provides a summary of relevant work pertaining to semi-supervised learning and community finding. In Section 3 we describe the proposed approach for community finding. To demonstrate the effectiveness of the approach, in Section 4 we perform a benchmarking evaluation on several synthetic networks. Finally, Section 5 presents concluding remarks and suggestions for extending this work.

2 Related Work

2.1 Community Finding

Finding non-overlapping communities. Algorithms in this context can be broadly grouped into three types. *(1) Hierarchical algorithms* construct a tree of communities based on the network topology. These algorithms can be one of two types: divisive algorithms [10] or agglomerative algorithms [7]. *(2) Modularity-based algorithms* optimize the well-known modularity objective function to uncover communities in the network [21]. *(3) Other algorithms.* This category includes algorithms based on label propagation approach, spectral algorithms that make use of the eigenvectors of Laplacian matrix or standard matrix, and methods based on statistical models [9].

Finding overlapping communities. Existing algorithms in this context can be classified into four main categories. *(1) Node seeds and local expansion.* These algorithms detect communities by starting from a node or a group of nodes, then expanding them into a community using a quality function. OSLOM [13] is an example of such an algorithm, which uses a statistical function to evaluate the node value to expand it to a community. Another example is MOSES [20], which is an algorithm based on a statistical model and uses an objective function as a greedy optimization technique. *(2) Clique expansion.* This type of method uses a group of fully-connected nodes, called a clique, as the starting point for expansion. CFinder [1] and Greedy Clique Expansion (GCE) [16] are examples of this type of algorithm. *(3) Link clustering.* This category of algorithms detects communities by splitting the links rather than the nodes [3]. *(4) Label propagation.* This strategy classifies each node into a community based on its neighboring nodes affinities. An example is the COPRA algorithm [11].

2.2 Semi-Supervised Learning

Several forms of prior knowledge have been used to guide the community detection process. The most widely-used has been pairwise constraints *("must-link" or "cannot-link")*, which indicate that two nodes must be in the same community or must be in different communities. Such constraints have been implemented in several algorithms, including a modularity-based method [18], a spectral analysis method [12, 23], and methods based on matrix factorization [22, 23]. Instead of constraints, other algorithms use *node labels* as prior knowledge to improve the process of community detection [17]. In [19], the authors propose a method that uses a semi-supervised label propagation algorithm based on *node labels* and *negative information*, where a node does not belong to a specific community.

The clear majority of semi-supervised algorithms in this area aim at detecting disjoint communities, whereas many real-world networks contain overlapping communities [1]. To the best of our knowledge, very little work has been done in the context of finding overlapping communities context. In [8], a small set of nodes called *seed nodes* was used, whose affinities to a community is provided as prior knowledge to infer the rest of the nodes affinities in the network. In the remainder of this paper we focus on the problem of semi-supervised community finding suitable for application to networks containing overlapping communities.

3 Methods

3.1 Greedy Clique Expansion (GCE)

The GCE [16] community finding algorithm initially finds maximal cliques as seeds, and subsequently expands these seeds into larger communities in a greedy fashion, by optimizing a local fitness function. Given a network G, a user-specified minimum clique size k, and a minimum community distance e, the GCE algorithm involves the following steps:

1. Find the seeds, which are all maximal cliques in G with at least k nodes.
2. Choose the largest unexpanded seed and greedily expand it into a candidate community $C\prime$ by using a community fitness function FS until adding any node would not increase the fitness value.
3. Test the distance between $C\prime$ and all previously accepted communities. If the distance between $C\prime$ and an existing community C is $< e$, then C and $C\prime$ are deemed to be near-duplicates, so discard $C\prime$. Otherwise, accept $C\prime$.
4. Repeat steps 2 and 3 until no more seeds remain

GCE employs a fitness function defined [14] to expand each seed, which defines the fitness community of a given community S in terms of its internal and external degrees (k_{in} and k_{out}) as follows

$$F_S = \frac{k_{in}^S}{(k_{in}^S + k_{out}^S)^\alpha} \tag{1}$$

where the parameter α typically takes values in the range $[0.9, 1.5]$. Generally, we can summarize the greedy expansion step using the fitness function F_S as follows:

1. For each node v in the frontier of a given seed S, calculate v's fitness value, i.e., the amount by which the community fitness of S would change if the node v was added to S.
2. Choose the node that has the maximum fitness value, v_{max}.
3. If the fitness value v_{max} is positive, then insert the node v into S and go back to Step 1. Otherwise, terminate and return S.

To discard near-duplicate communities in Step 4 of the GCE algorithm, the authors proposed the use of an overlap-based measure of distance between two communities:

$$\delta_E(S, S') = 1 - \frac{|S \cap S'|}{min(|S|, |S'|)} \tag{2}$$

Given two communities S and $S\prime$, Eqn. 2 measures the number of nodes in the smaller community that are not included in the larger one. So, for a given set of communities, a near-duplicate community of a given community S would be all the communities that are within a distance e of S.

3.2 Pairwise Constraints in Community Detection

Given a network that contains a set of nodes V, semi-supervised pairwise constraints typically take two possible forms:

1. *Must-link constraints specify* that two nodes must be in the same community. Let C_{ML} be the must-link constraint set: $\forall\ v_i, v_j \in V$ where $i \neq j$, $(v_i, v_j) \in C_{ML}$ indicates that the two nodes v_i and v_j must be assigned to the same community.
2. *Cannot-link constraints* specify that two nodes must be in different communities. Let C_{CL} be the cannot-link constraint set: $\forall\ v_i, v_j \in V$ where $i \neq j$, $(v_i, v_j) \in C_{CL}$ indicates that the two nodes v_i and v_j must be assigned to two distinct communities.

The simplest approach to selecting this form of constraints is to naively select a pair of nodes (v_i, v_j) at random, and query the oracle about whether the pair should share a must-link or cannot-link constraint. This process can be repeated to select the required number of constraints or until some supervision budget is exhausted.

In non-overlapping community finding, must-link constraints have what is referred to as a *transitive property*, where a third must-link relationship can be inferred from two other associated must-link constraint pairs. So, if $(v_i, v_j) \in C_{ML}$, and $(v_i, v_k) \in C_{ML}$, then we can also infer that $(v_j, v_k) \in C_{ML}$ (see the first example in Fig. 1).

However, incorporating constraints into the context of overlapping communities is more challenging. This is because the transitive property does not hold

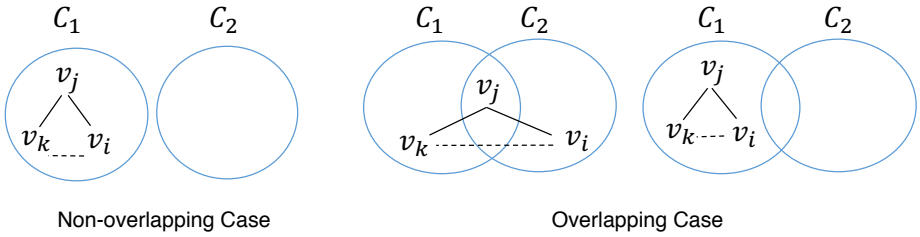

Fig. 1: In the non-overlapping context, the transitive property allows us to infer a third must-link constraint from two existing must-link constraints. However, this does not automatically apply in the overlapping context, where two possible situations exist.

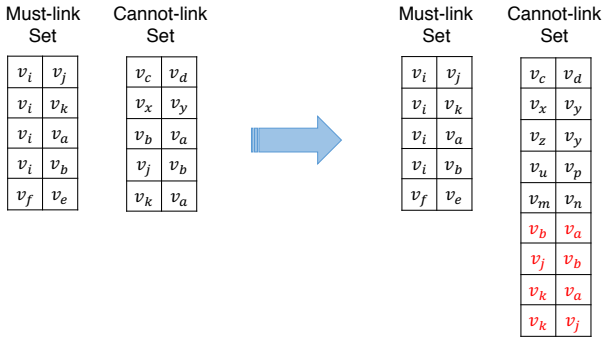

Fig. 2: Example of expanding 5 pairwise constraints. Inserting the derived cannot-link pairs from all connected must-link pairs will double the size of the cannot-link set.

here (see the second example in Fig. 1). Specifically, if $(v_i, v_j) \in C_{ML}$, and $(v_i, v_k) \in C_{ML}$, there are two possible scenarios for the pair (v_j, v_k). It can be the case that either $(v_j, v_k) \in C_{ML}$ or $(v_j, v_k) \in C_{CL}$. This is because an overlapping node v_j can have a must-link constraint with both v_i and v_i, yet these two nodes could belong to two different communities. However, it is also possible that all three nodes are in fact in the same community. Unless we explicitly inform the algorithm about whether a must-link or cannot-link constraint exists for the pair (v_j, v_k), we cannot reliably distinguish between the two cases. If the network has highly-overlapping communities, then this problematic situation will occur more frequently. If we naively attempt to incorporate pairwise constraints into overlapping community finding without taking this situation into account, it is likely that the quality of the resulting communities can potentially decrease even as more constraints are added.

To resolve this issue, after selecting pairwise constraints, we need to explicitly detect every cannot-link pair that derived from any two connect must-link pairs and insert it to the cannot link set. However, this will significantly increase the set of cannot-link constraints. For example, if we want to feed only five pairwise constraints, each as shown in Fig. 2, inserting the required cannot-link pairs

will double the size of the cannot-link set. In the next section, we introduce an approach to address this issue.

3.3 Semi-Supervised GCE

We now describe our proposed approach for overlapping community finding with limited supervision, referred to as Pairwise Constrained GCE (PC-GCE). This approach consists of two stages: The first stage includes selecting and pre-processing constraints and resolves the problem of the lack of the transitive property for must-link constraints in the context of overlapping communities. The second stage supplies the resulting constraints to the GCE algorithm to process them during community detection. In the following, these stages are described in more detail.

Stage 1: Selecting and pre-processing constraints. In this stage, we can treat the set of pairwise constraints as a new graph, where an edge exists between two nodes from the original network if they share a pairwise constraint (either must-link or cannot-link). Then we look for all possible *forbidden triads* among the nodes involved in the must-link set. Given three nodes A, B, C, a *forbidden triad* (or *open triad*) occurs when A is connected to B and C, but no edge exists between B and C. In our pre-processing step, we look for such cases — i.e. where we do not know whether a must-link or cannot-link exists between a pair of nodes B and C. To control the size of the constraints set, we greedily expand it until we reach a pre-defined maximum size. This stage can be summarized as follows (see also Fig. 3):

1. Choose a small initial random set of both must-link and cannot-link sets.
2. Find all possible forbidden triad cases in the must-link set to query the oracle about their relationship.
3. If their relationship is must-link insert it into must-link set, otherwise insert it into cannot-link set.
4. Repeat all steps until the set reaches the maximum size.

At the end of this stage, the pairwise constraints set is ready to be supplied to the GCE algorithm for community detection.

Stage 2: Pairwise Constrained GCE (PC-GCE). During the community detection phase, we incorporate only cannot-link constraints into the existing GCE algorithm as follows (see Fig. 4 for an illustration):

1. Find seeds, which are all maximal cliques in G with at least k nodes.
2. Choose the largest unexpanded seed and greedily expand it to a candidate community C$'$ by using a community fitness function (Eqn. 1) until adding any node no longer increases the fitness value. However, during this expansion process, do not add any node, which has a cannot link relationship with any existing node in the seed.
3. Check for the existence of any cannot-link constraints among all pairs of nodes in C$'$. If such a pair exists, calculate the fitness for both nodes relative to C$'$, and remove the one with the lower value.

4. Test the distance between the community C′ and all of the already accepted communities, using Eqn. 2. If the distance between C′ and any accepted community C is $<e$, then C and C′ are near-duplicates, then discard C′. Otherwise, accept C′.
5. Go back to Step 2 and repeat the process until no seeds remain.

The justification for using only cannot-link constraints in the community detection process is as follows: because of the greedy nature of the seed expansion step, incorporating must-link constraints into this step results in a smaller number of considerably larger communities. Mainly using the fitness function as a technique of greedy local optimization to expand clique to community already achieves some of the must-link relationships, and processing cannot-link set will mostly detect the pairs that derived from two connected must-link pairs. Thus, using must-link set to be explicitly processed by the algorithm will be as processing extra non-informative constraints which cause noise to the algorithm and reduction of the accuracy.

4 Evaluation

In this section, the performance of the Pairwise Constrained GCE algorithm (PC-GCE) is evaluated by running experiments on two groups of synthetic benchmark networks containing overlapping communities. Since, to the best of our knowledge, no work has been conducted in the literature regarding pairwise constrained algorithms for finding overlapping communities, for the sake of comparison, the PC-GCE results are compared with the following unsupervised overlapping community detection algorithms: standard GCE [16], OSLOM [13], MOSES [20], and COPRA [1].

4.1 Data

The synthetic networks used in our experiments is generated using the widely-used LFR benchmark generator [15], which can produce networks with properties

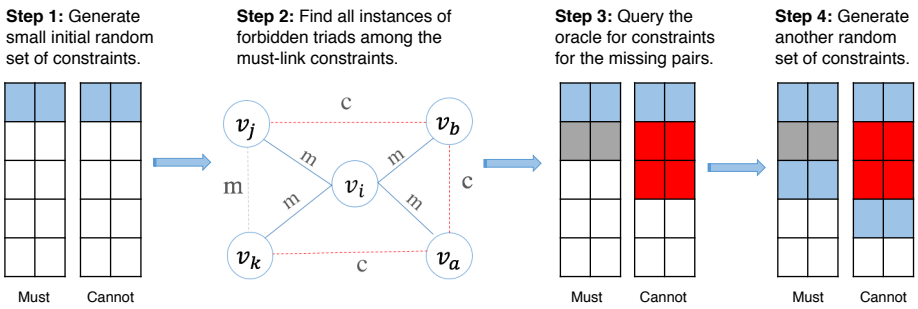

Fig. 3: An illustration of the four steps involved in Stage 1 of semi-supervised GCE.

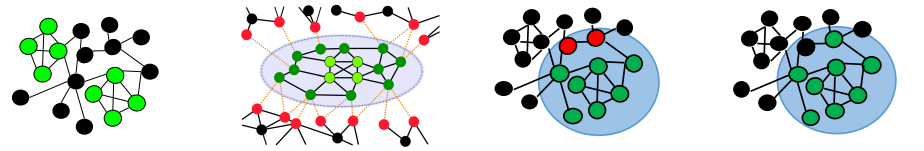

Fig. 4: An illustration of the four steps involved in Stage 2 of semi-supervised GCE.

similar to real-world networks, which also contain embedded ground truth communities. The full set of parameter values used to generate the networks is listed in Table 1.

In our experiments, we generate two different groups of networks, containing small and large communities respectively. Small communities have 10–50 nodes, while large communities have 20–100 nodes. Each group consists of 16 networks with different combinations of the two parameters O_m and O_n. The parameter O_m controls the number of communities per node, and O_n controls the number of overlapping nodes. For the first network in each set, all nodes belong to two communities ($O_m = 2$), then for each successive network this parameter increments in value by 1 until $O_m = 5$ is reached. For each value taken by the parameter O_m, we increase the fraction of overlapping nodes O_n by 25% until 100% of the nodes belong to more than one community.

Table 1: Parameter values used for the generation of LFR synthetic networks.

	Description	*Value*		*Description*	*Value*
N	Number of nodes	1000	t_1	Degree exponent	2
k	Average degree	20	t_2	Community exponent	2
K_{max}	Max degree	60	M	Mixing parameter	0.2
C_{min}	Min community size	10/20	O_n	Num of overlapping nodes	250-1000
C_{max}	Max community size	50/100	O_m	Communities per node	1-5

4.2 Experimental Setup

To compare the performance of the different algorithms in our experiments, we use the overlapping form of the standard Normalized Mutual Information (NMI) measure, as proposed in [14]. This measures the level of agreement between the communities produced by an algorithm on a network and the ground truth communities in that network. A value close to 1 indicates a high level of agreement, while a value close to 0 indicates that the algorithms communities are no better than random.

We have conducted two experiments. The first experiment aims to measure the current performance of the selected community detection algorithms. We use these values as a baseline for evaluating the performance of the proposed PC-GCE algorithm. The second experiment evaluates the performance of the PC-GCE on differing numbers of constraints, ranging from 1% to 5% of the total number of possible constraints pairs in each network. In this experiment, the initial pairwise constraints are selected at random. Therefore, we repeat the process over 20 runs and average the NMI scores. Finally, we compare the results obtained from PC-GCE with the selected benchmark algorithms.

Table 3: NMI scores of benchmark algorithms on small and large community networks.

Small Comm.		Comm. per node (O_m)			
		MOSES			
		2	3	4	5
Fraction of overlapping nodes (O_n)	250	0.8392	0.8127	0.6896	0.6057
	500	0.8357	0.6911	0.5836	0.4681
	750	0.7729	0.5915	0.4474	0.2527
	1000	0.7206	0.5188	0.2252	0.1171
		COPRA			
		2	3	4	5
Fraction of overlapping nodes (O_n)	250	0.7536	0.6138	0.4951	0.3595
	500	0.5664	0.0000	0.0000	0.0000
	750	0.0000	0.0000	0.0000	0.0000
	1000	0.0000	0.0000	0.0000	0.0000
		OSLOM			
		2	3	4	5
Fraction of overlapping nodes (O_n)	250	0.9733	0.9147	0.7963	0.7169
	500	0.9435	0.8496	0.6020	0.3172
	750	0.9240	0.6789	0.2299	0.0777
	1000	0.8953	0.3717	0.0098	0.0000

Large Comm.		Comm. per node (O_m)			
		MOSES			
		2	3	4	5
Fraction of overlapping nodes (O_n)	250	0.6140	0.5448	0.4595	0.4244
	500	0.5338	0.4497	0.3495	0.3198
	750	0.4789	0.3435	0.1795	0.0960
	1000	0.5741	0.1989	0.0191	0.0000
		COPRA			
		2	3	4	5
Fraction of overlapping nodes (O_n)	250	0.7121	0.6235	0.4844	0.4377
	500	0.5217	0.0000	0.0000	0.0000
	750	0.0000	0.0000	0.0000	0.0000
	1000	0.0000	0.0000	0.0000	0.0000
		OSLOM			
		2	3	4	5
Fraction of overlapping nodes (O_n)	250	0.9524	0.8601	0.7076	0.6425
	500	0.9078	0.7375	0.4646	0.2285
	750	0.9026	0.4909	0.1780	0.0725
	1000	0.8618	0.1383	0.0000	0.0000

4.3 Results and Discussion

We have two baselines for comparison and evaluation of the results. Firstly, we compare the accuracy of PC-GCE to standard GCE (Tables 2). Secondly, we compare the accuracy of PC-GCE algorithm to the other benchmark algorithms (Table 3).

As we observe from Tables 2, in most cases, regardless of the fraction of overlapping nodes in the networks, PC-GCE outperforms the standard GCE algorithm. To be more specific, for all measures of fraction of overlapping nodes, as the percentage of pairwise constraints increases, the accuracy of PC-GCE also improves. On the other hand, as we increase the fraction of overlapping nodes O_n from 25% to 100% of the total number of nodes, the NMI of GCE drops to 0 for almost 60% of the total set of results. For instance, in the case of small community networks, the NMI score of GCE drops from 0.764 to 0 for $O_m = 3$, and from 0.648 to 0 for $O_m = 4$. In contrast, the PC-GCE algorithm shows a

Table 2: NMI scores using $0-5\%$ of constraints on small and large community networks.

Small community networks		Comm. per node (O_m)					
		2					
		0%	1%	2%	3%	4%	5%
		GCE	PCGCE	PCGCE	PCGCE	PCGCE	PCGCE
Fraction of overlapping nodes (O_n)	250	0.9932	0.9935	0.9936	0.9941	0.9939	0.9940
	500	0.9660	0.9786	0.9858	0.9865	0.9896	0.9905
	750	0.9300	0.9504	0.9656	0.9715	0.9769	0.9815
	1000	0.9378	0.9508	0.9576	0.9650	0.9698	0.9723
		3					
		0%	1%	2%	3%	4%	5%
		GCE	PCGCE	PCGCE	PCGCE	PCGCE	PCGCE
Fraction of overlapping nodes (O_n)	250	0.7637	0.7866	0.8004	0.8139	0.8309	0.8432
	500	0.7070	0.7579	0.8064	0.8305	0.8562	0.8696
	750	0.6276	0.6856	0.7231	0.7586	0.7760	0.8002
	1000	0.0000	0.6239	0.6343	0.6538	0.6650	0.6721
		4					
		0%	1%	2%	3%	4%	5%
		GCE	PCGCE	PCGCE	PCGCE	PCGCE	PCGCE
Fraction of overlapping nodes (O_n)	250	0.6484	0.6167	0.6389	0.6612	0.6743	0.7037
	500	0.3812	0.4184	0.4921	0.5411	0.5754	0.6055
	750	0.0000	0.3156	0.3716	0.4074	0.4303	0.4536
	1000	0.0000	0.1747	0.1933	0.2062	0.2159	0.2275
		5					
		0%	1%	2%	3%	4%	5%
		GCE	PCGCE	PCGCE	PCGCE	PCGCE	PCGCE
Fraction of overlapping nodes (O_n)	250	0.4702	0.4751	0.4909	0.5085	0.5114	0.5256
	500	0.2015	0.2038	0.2429	0.2673	0.3007	0.3135
	750	0.0000	0.1053	0.1315	0.1486	0.1624	0.1732
	1000	0.0000	0.0160	0.0177	0.0190	0.0216	0.0218
Large community networks		Comm. per node (O_m)					
		2					
		0%	1%	2%	3%	4%	5%
		GCE	PCGCE	PCGCE	PCGCE	PCGCE	PCGCE
Fraction of overlapping nodes (O_n)	250	0.9693	0.9704	0.9713	0.9722	0.9721	0.9724
	500	0.9852	0.9902	0.9923	0.9930	0.9954	0.9957
	750	0.9443	0.9652	0.9750	0.9784	0.9816	0.9860
	1000	0.8209	0.8620	0.8843	0.8861	0.8895	0.8947
		3					
		0%	1%	2%	3%	4%	5%
		GCE	PCGCE	PCGCE	PCGCE	PCGCE	PCGCE
Fraction of overlapping nodes (O_n)	250	0.6542	0.6756	0.6760	0.6768	0.6773	0.6776
	500	0.6276	0.6660	0.6855	0.6932	0.7042	0.7087
	750	0.4953	0.5646	0.5865	0.5963	0.6224	0.6240
	1000	0.2577	0.2889	0.3075	0.3250	0.3315	0.3495
		4					
		0%	1%	2%	3%	4%	5%
		GCE	PCGCE	PCGCE	PCGCE	PCGCE	PCGCE
Fraction of overlapping nodes (O_n)	250	0.5590	0.5596	0.5606	0.5617	0.5611	0.5623
	500	0.4402	0.4623	0.4752	0.4862	0.4886	0.4954
	750	0.1031	0.1281	0.1587	0.1665	0.1754	0.1774
	1000	0.0000	0.0106	0.0110	0.0142	0.0153	0.0211
		5					
		0%	1%	2%	3%	4%	5%
		GCE	PCGCE	PCGCE	PCGCE	PCGCE	PCGCE
Fraction of overlapping nodes (O_n)	250	0.4667	0.4669	0.4672	0.4680	0.4686	0.4687
	500	0.1980	0.2083	0.2315	0.2489	0.2589	0.2693
	750	0.0000	0.0637	0.0784	0.0798	0.0762	0.0815
	1000	0.0000	0.0000	0.0000	0.0002	0.0000	0.0000

moderate decrease of accuracy as the value of On increases. This indicates that PC-GCE outperform the standard GCE with highly-overlapping communities. However, in general, both algorithms show better performance on the networks containing smaller communities.

We can see from Table 3 that PC-GCE algorithm outperforms COPRA algorithm on both types of networks. When considering the smallest fraction of overlapping nodes (i.e., $O_n = 250$), COPRA performs almost comparably with PC-GCE. However, as the value of O_n increases, the performance of COPRA drops to 0, whereas the performance of PC-GCE shows only a slight decrease in the accuracy. Thus, it can be concluded that PC-GCE outperforms COPRA on highly overlapping community networks. On the other hand, PC-GCE outperforms MOSES on large networks but shows almost similar performance on small networks. Finally, the NMI scores exhibit comparable overall performance with PC-GCE in the case of OSLOM, on both types of networks.

5 Conclusion

In this paper, we have explored the potential of semi-supervised strategies to improve existing algorithms for finding overlapping communities in networks, and a new algorithm for detecting overlapping communities with pairwise constraints (PC-GCE) is proposed. Extensive experiments were carried out, and the results show GCE algorithm with constraints (PC-GCE) outperforms unconstrained algorithms with highly-overlapping communities, and its performance improves with increasing number of pairwise constraints. This shows the potential of using semi-supervised strategies for finding overlapping communities. However, in most networks, only a small percentage of the nodes have informative pairwise constraints. Thus, using random selection of pairwise constraints could have adverse effects by decreasing the accuracy of the community detection process caused by the selection of non-informative nodes. Therefore, our future work will aim to apply ideas from active learning for selecting informative pairwise constraints. We will also explore the effect of incorrect "noisy" constraints and how they affect algorithm performance.

Acknowledgements. This publication has partly emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under Grant Number SFI/12/RC/2289.

References

1. Adamcsek, B., Palla, G., Farkas, I.J., Derényi, I., Vicsek, T.: Cfinder: locating cliques and overlapping modules in biological networks. Bioinformatics 22(8), 1021–1023 (2006)
2. Ahn, Y.Y., Bagrow, J.P., Lehmann, S.: Link communities reveal multiscale complexity in networks. Nature 466(7307), 761–764 (2010)
3. Amelio, A., Pizzuti, C.: Overlapping community discovery methods: a survey. In: Social Networks: Analysis and Case Studies, pp. 105–125. Springer (2014)

4. Basu, S., Bilenko, M., Mooney, R.J.: A probabilistic framework for semi-supervised clustering. In: Proc. 10th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining. pp. 59–68 (2004)
5. Blondel, V., Guillaume, J., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. J. Stat. Mech 10008 (2008)
6. Chapelle, O., Schölkopf, B., Zien, A. (eds.): Semi-Supervised Learning. MIT Press, Cambridge (2006)
7. Clauset, A., Newman, M.E., Moore, C.: Finding community structure in very large networks. Physical review E 70(6), 066111 (2004)
8. Dreier, J., Kuinke, P., Przybylski, R., Reidl, F., Rossmanith, P., Sikdar, S.: Overlapping communities in social networks. arXiv preprint arXiv:1412.4973 (2014)
9. Fortunato, S.: Community detection in graphs. Physics reports 486(3), 75–174 (2010)
10. Girvan, M., Newman, M.E.: Community structure in social and biological networks. Proceedings of the national academy of sciences 99(12), 7821–7826 (2002)
11. Gregory, S.: Finding overlapping communities in networks by label propagation. New Journal of Physics 12(10), 103018 (2010)
12. Habashi, S., Ghanem, N.M., Ismail, M.A.: Enhanced community detection in social networks using active spectral clustering. In: Proceedings of the 31st Annual ACM Symposium on Applied Computing. pp. 1178–1181. ACM (2016)
13. Lancichinetti, A., Radicchi, F., Ramasco, J., Fortunato, S., Ben-Jacob, E.: Finding statistically significant communities in networks. PLoS ONE 6(4), e18961 (2011)
14. Lancichinetti, A., Fortunato, S., Kertész, J.: Detecting the overlapping and hierarchical community structure in complex networks. New Journal of Physics 11(3), 033015 (2009)
15. Lancichinetti, A., Fortunato, S., Radicchi, F.: Benchmark graphs for testing community detection algorithms. Physical review E 78(4), 046110 (2008)
16. Lee, C., Reid, F., McDaid, A., Hurley, N.: Detecting highly overlapping community structure by greedy clique expansion. In: Workshop on Social Network Mining and Analysis (2010)
17. Leng, M., Yao, Y., Cheng, J., Lv, W., Chen, X.: Active semi-supervised community detection algorithm with label propagation. In: International Conference on Database Systems for Advanced Applications. pp. 324–338. Springer (2013)
18. Li, L., Du, M., Liu, G., Hu, X., Wu, G.: Extremal optimization-based semi-supervised algorithm with conflict pairwise constraints for community detection. In: Advances in Social Networks Analysis and Mining (ASONAM), 2014 IEEE/ACM International Conference on. pp. 180–187. IEEE (2014)
19. Liu, D., Duan, D., Sui, S., Song, G.: Effective semi-supervised community detection using negative information. Mathematical Problems in Engineering 2015 (2015)
20. McDaid, A., Hurley, N.: Detecting highly overlapping communities with model-based overlapping seed expansion. In: Advances in Social Networks Analysis and Mining (ASONAM), 2010 International Conference on. pp. 112–119. IEEE (2010)
21. Newman, M.E.: Modularity and community structure in networks. Proceedings of the national academy of sciences 103(23), 8577–8582 (2006)
22. Shi, X., Lu, H., He, Y., He, S.: Community detection in social network with pairwisely constrained symmetric non-negative matrix factorization. In: Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015. pp. 541–546. ACM (2015)
23. Zhang, Z.Y.: Community structure detection in complex networks with partial background information. EPL (europhysics letters) 101(4), 48005 (2013)