Discovering Polarization Niches via Dense Subgraphs with Attractors and Repulsers

Adriano Fazzone
CENTAI Institute
Turin, Italy
adriano.fazzone@centai.eu

Tommaso Lanciano
Sapienza University
Rome, Italy
lanciano@diag.uniroma1.it

Riccardo Denni
Sapienza University
Rome, Italy
denni@diag.uniroma1.it

Charalampos E. Tsourakakis
Boston University, USA
ISI Foundation, Turin, Italy
ctsourak@bu.edu

Francesco Bonchi
CENTAI Institute, Turin, Italy
Eurecat, Barcelona, Spain
bonchi@centai.eu

ABSTRACT

Detecting niches of polarization in social media is a first step towards deploying mitigation strategies and avoiding radicalization. In this paper, we model polarization niches as close-knit dense communities of users, which are under the influence of some well-known sources of misinformation, and isolated from authoritative information sources. Based on this intuition we define the problem of finding a subgraph that maximizes a combination of (i) density, (ii) proximity to a small set of nodes A (named *Attractors*), and (iii) distance from another small set of nodes R (named *Repulsers*).

Deviating from the bulk of the literature on detecting polarization, we do not exploit text mining or sentiment analysis, nor we track the propagation of information: we only exploit the network structure and the background knowledge about the sets A and R, which are given as input. We build on recent algorithmic advances in supermodular maximization to provide an iterative greedy algorithm, dubbed *Down in the Hollow* (DITH), that converges fast to a near-optimal solution. Thanks to a novel theoretical upper bound, we are able to equip DITH with a practical device that allows to terminate as soon as a solution with a user-specified approximation factor is found, making our algorithm very efficient in practice. Our experiments on very large networks confirm that our algorithm always returns a solution with an approximation factor better or equal to the one specified by the user, and it is scalable. Our case-studies in polarized settings, confirm the usefulness of our algorithmic primitive in detecting polarization niches.

PVLDB Reference Format:
Adriano Fazzone, Tommaso Lanciano, Riccardo Denni, Charalampos E. Tsourakakis, and Francesco Bonchi. Discovering Polarization Niches via Dense Subgraphs with Attractors and Repulsers. PVLDB, 15(13): 3883 - 3896, 2022.
doi:10.14778/3565838.3565843
PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/tlancian/dith.

Proceedings of the VLDB Endowment, Vol. 15, No. 13 ISSN 2150-8097.
doi:10.14778/3565838.3565843

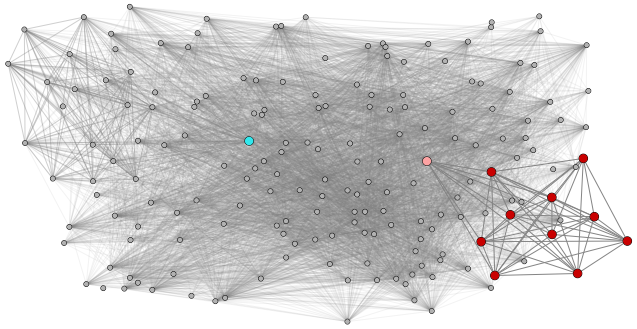

Figure 1: In the Greek parliament Twitter-sphere network (see Section 5.4 for dataset description) we search for far-right polarization niches, by querying for a dense cluster that is far from the light blue node (repulser), representing the center-left newspaper *Efimerida Syntakton* and close to the pink node (attractor), representing the far-right tabloid *Makeleio*. The solution found by our method is a dense cluster of ten *Golden dawn* leaders (red nodes), which, in October 2020, have been charged with running a criminal organization.

1 INTRODUCTION

Social media have become the main stage for societal debates in recent times. This emerging participatory environment, where everyone can easily access information and participate in the societal debate by expressing their own opinion, is heavily employed by authorities, official organizations, old-media outlets, as well as all sort of low-quality information, or even misinformation, sources. Hand in hand with the quality of the information propagating in social media, stands another important issue: the increase of polarization and partisanship around controversial issues. Users tend to interact with like-minded individuals and consume partisan information which reinforces their own ideological viewpoint [7, 24]: this is the so-called *"echo chamber"* effect [14]. Detecting misinformation campaigns and echo chambers, blocking bots, and limiting excessive polarization, have thus become urgent societal and technological problems, which are witnessing an uptake of the research on developing detection and intervention methods [6, 23, 25, 28, 41, 42, 47, 63].

In this paper, as a prerequisite towards deploying interventions, we study the problem of identifying niches of users which are well connected among them, exposed to low-quality information, and

isolated from authoritative sources. Consider, as a concrete example, the vaccination debate around COVID-19, where hesitancy builds up in communities around presumed authoritative sources which instead spread conspiracy-theory-based misinformation, and in which official medical authorities and academic sources have no penetration. We tackle the challenge of detecting these niches of users as a dense subgraph discovery problem. Differently from the literature, we do not exploit text mining or sentiment analysis, nor we track the propagation of information: we only use the network structure and the background knowledge about a small set of nodes A (Attractors), to which we want our solution to be close, and a small set of nodes R (Repulsers) from which we want our solution to be far. In our setting, the attractors A might be nodes that are known to be misinformation sources, while the repulsers R might be authoritative information sources. We aim at finding a close-knit dense community of users which are socially close to A and far from R: this might be a polarization niche which is strongly influenced by radicalizing content sources and isolated from authoritative sources. Figures 1 and 2 show examples of application of our algorithmic primitive to the Greek Twitter-sphere network.

Based on this intuition, we define the Dense Subgraphs with Attractors and Repulsers Problem (DSAR) whose objective is to maximize a combination of density, proximity to A, and distance from R, regulated by two scalars (λ_1 and λ_2) which control the importance of proximity and distance, respectively. Our problem is, at the same time, a generalization of the Densest Subgraph Problem (DSP)[1], and a special instance of a particular generalization of DSP on node-weighted graphs. Such a variant, that we dub Heavy and Dense Subgraph Problem (HDSP), was proposed by Goldberg in his seminal 1984 paper [27, Section 6]: although it has been used in applications (e.g., [32]), not much algorithmic progress has been made on HDSP since Goldberg's 1984 paper. Specifically, Goldberg shows that for HDSP, as for DSP, an exact solution can be computed in polynomial time by solving the maximum-flow problem. However, being based on max-flow computations, the algorithm is too slow to run on medium/large instances.

We thus focus our attention on devising an iterative greedy algorithm that converges fast to a near optimal solution, exploiting very recent algorithmic advances in supermodular maximization [13]. In order to explain in detail our technical contributions, we first need to provide some technical background.

Technical Background. DSP is a foundational formulation of dense subgraph discovery [26] that maximizes the average degree over all possible subgraphs. Goldberg [27] gave one of the first polynomial-time algorithms for the densest subgraph problem, via a reduction to $O(\log n)$ instances of maximum flow, where each candidate value of maximum average degree yields a unique maximum flow instance. Gallo, Grigoriadis, and Tarjan [22] solved DSP with a single Parametric-MaxFlow computation. Charikar [12] gave an alternative polynomial-time solution for the problem by formulating a linear program which solves it exactly. Charikar also gave a linear-time, $\frac{1}{2}$-approximation algorithm (known as "peeling") that greedily removes the node with the smallest degree, and then reports the maximum density seen among these subgraphs. This algorithm was also studied in [5].

[1]When $\lambda_1 = \lambda_2 = 0$ DSAR corresponds exactly to DSP.

Recently, Boob et al. [11] proposed Greedy++, an algorithm inspired by multiplicative weights update [4]. The algorithm simply runs Charikar's peeling routine for several iterations, and updates the vertex priorities based on the results from the previous iteration.

Very recently, Chekuri, Quanrud, and Torres [13], building over [11], showed that Greedy++ converges to a solution with an arbitrarily small approximation factor, and that it naturally extends to a broad class of supermodular functions. In the remaining of the paper we shall refer to the algorithm in [13] as Super-Greedy++.

Our contributions. We exploit the recent algorithmic advances of [13] for devising our *Down in the Hollow* (dith) algorithm for HDSP (and thus for our DSAR problem). We prove an upper bound which enables a termination criterion that makes dith much more efficient in practice. Specifically, we employ a user-specified input parameter γ, that allows our algorithm to terminate first if a solution that is a $(1-\gamma)$ approximation of the optimum is found. To prove the correctness of this modification, we define an LP program for HDSP, thus providing another exact algorithm alternative to Goldberg's [27].

More in details, the contributions of this paper can be summarised as follows:

- We define the novel Dense Subgraphs with Attractors and Repulsers Problem (DSAR), where the solution is required to be dense, close to a given set of nodes A, and far from another set of nodes R. We show that our problem can be seen as an instance of HDSP, thus solvable in polynomial time.
- We dust off the HDSP problem which, to the best of our knowledged, hasn't been studied since Goldberg's 1984 paper [27], and revive it through the lenses of very recent advances in supermodular maximization, obtaining an iterative greedy algorithm that converges fast to a near-optimal solution.
- We prove an upper bound which allows our algorithm to terminate as soon as a solution with a user-specified approximation factor is found, making our algorithm very efficient in practice. This also represents an advancement over the state of the art [11, 13] for the classic Densest Subgraph Problem: in fact, before our work it was possible to know the approximation factor only in an imprecise way (in terms of big-O notation).
- To prove the correctness of the bounds required by the early-stop device, we define an LP program for HDSP: thus, as a side effect, we provide another exact algorithm for this classic problem. We also show that the standard peeling algorithm has an approximation factor better than $\frac{1}{2}$ for any instance of HDSP where all weights on nodes are strictly positive.
- Our experiments on large networks confirm that our algorithm always returns a solution with an approximation factor better or equal to the one specified by the user, and it is scalable. Our case-studies in polarized settings, confirm the usefulness of our algorithmic primitive in detecting polarization niches.

2 BACKGROUND AND RELATED WORK

Detecting polarization in social media. Several researchers have studied the problem of detecting polarization in social media. Some approaches use content information and adopt text-analysis approaches [15, 43, 51], while others use graph-based approaches [2,

9, 16, 24, 46, 48, 52]. Among the graph-based approaches, different types of information are taken into consideration to create the network for further analysis: some researchers use (positive or negative) interactions among the users codified in signed edges of a *signed network* [2, 9, 52], others exploit *information cascades* generated by re-tweets [44, 46], others use multiple data sources such as interactions, content and re-tweets [16, 24].

Our work departs from this literature, as we are developing algorithmic primitives that provide insights into the polarization niches using two targeted sets of nodes that are either (i) strong proponents of diametrically opposed opinions, or (ii) misinformation vs. authoritative and reliable information sources. Contrarily to other graph-based approaches, we do not consider any content or interactions information.

Local Community Search. Given a graph $G = (V, E)$ and a query set of nodes $Q \subseteq V$, the community search problem requires to find a cluster of nodes S which contains (completely or partially) Q and whose induced subgraph optimizes some measures of cohesiveness, e.g., density [8, 18, 33, 37, 54, 56, 61] or conductance [58, 64]. Our problem can also be seen as a variant of the community search problem in which, instead of having as input only one set of nodes, we have two sets: one to which we want the solution to be close, and one to which we want the solution to stay away.

Many authors have adopted random-walk-based approaches [3, 10, 36, 37, 57, 61] for community search: indeed, the problem of finding other vertices related to a given seed of vertices is the basic idea of *Topic Sensitive PageRank* (*TSPR*) [31, 34]. In our experiments in Section 5.3, we are inspired by this literature to devise two non-trivial baselines for our problem, to compare with our method.

Submodular minimization. A function $f : 2^V \rightarrow \mathbb{R}$ defined on a ground set V is submodular if for any $S, T \subseteq V$: $f(S \cup T) \leq f(S) + f(T) - f(S \cap T)$. Alternatively, for every $A \subset B \subset V$ and $x \in V \setminus B$, f is submodular if $f(A \cup \{x\}) - f(A) \geq f(B \cup \{x\}) - f(B)$.

A function f is supermodular if and only if $-f$ is submodular, i.e., when the inequalities in the above definitions are reversed.

If f is both supermodular and submodular, then f is modular, and the inequalities become equalities. We refer to a set function $f : 2^V \rightarrow \mathbb{R}$ as normalized if, and only if, $f(\emptyset) = 0$, and as non-negative if, and only if, $f(S) \geq 0, \forall S \subseteq V$. A set function $f : 2^V \rightarrow \mathbb{R}$ is monotone non-decreasing (resp. non-increasing) if $\forall S \subseteq T \subseteq V$, $f(S) \leq$ (resp. $\geq$) $f(T)$. It is straightforward to show that if $f : 2^V \rightarrow \mathbb{R}$ is a normalized, non-negative, and supermodular set function, then it is also monotone non-decreasing. Furthermore, if $f, g : 2^V \rightarrow \mathbb{R}$ are two supermodular, non-negative, normalized and monotone non-decreasing functions defined on the same ground set V, then any non-negative linear combination $c_1 f(S) + c_2 g(S), c_1, c_2 \in \mathbb{R}_{\geq 0}$ satisfies the same properties.

In general, submodular functions may require a representation that is exponential in size, and therefore we assume that we are given access to a *value oracle* which given a set S returns $f(S)$. The classic problem of maximizing a monotone (i.e., $S \subseteq T \implies f(S) \leq f(T)$) submodular function under a cardinality constraint is solvable with a greedy algorithm which iteratively adds the element with largest marginal contribution into the solution obtains a $1 - 1/e$ approximation, which is optimal unless using exponentially-many queries or P=NP [20, 49]. Maximizing

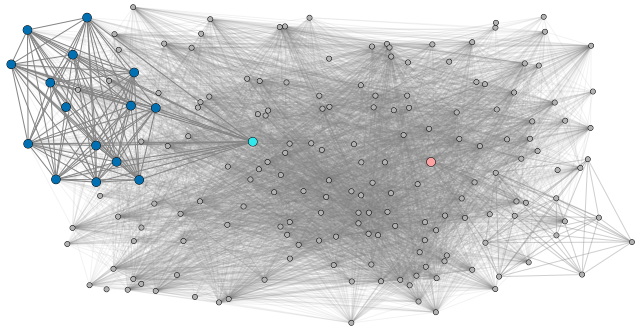

Figure 2: Again the Greek parliament Twitter-sphere network as in Figure 1, but switching the roles of *Efimerida Syntakton* (light blue node), which now is the attractor, and *Makeleio* (pink node), which now is the repulser. In this case the solution (dark blue nodes) is a very dense subgraph of 15 nodes composed by the main news outlets of Greece.

submodular functions appears in a variety of domains and applications, including combinatorial auction theory [19], sensor network placement [30], influence maximization [35], data streams [45], and machine learning [38].

Similarly to other problems in dense subgraphs discovery [13, 62], the problem we tackle in this paper is an instance of *submodular minimization*, which is solvable in strongly polynomial time, under the value oracle model [55]. However, the runtime of such an exact polynomial time algorithm is $O(n^6)$ [55].

3 PROBLEM STATEMENT

We are given an undirected graph $G = (V, E)$ and two distinct sets of nodes $A, R \subset V$, $A \cap R = \emptyset$, representing cognitively-grounded nodes which have a clear and diametric stance on a controversial topic [1]. For example, there exists news media that are well known to be right- or left-leaning in terms of politics; similarly, there exist Twitter accounts that clearly state their stance with respect to a controversial topic, e.g., abortions. The two sets A and R, can also represent sources of information such that are either trustworthy, or spread misinformation respectively. An extreme special case of such anchored nodes are propaganda accounts, which are not interested at all in a civil exchange of arguments for the sake of truth but are rather propagators of a perspective that typically serves their interests, using even spread of misinformation [29].

While polarization has been studied in a variety of recent works, we lack the algorithmic tools to discover polarization niches. We introduce an algorithmic primitive whose goal is to find a dense cluster of nodes S, where each node in S is simultaneously close to the set A of *Attractors* and far from the set R of *Repulsers*.

As the function of distance between one node $s \in S$ and the set of nodes R we consider $d : V \times 2^V \rightarrow \mathbb{N}$, such that:

$$d(s, R) = \begin{cases} 0, & \text{if } s \in R \\ \min_{r \in R} d_{sp}(s, r), & \text{otherwise} \end{cases}$$

where $d_{sp}(s, r)$ represents the classic shortest-path distance between two nodes. As the function of proximity between one node $s \in S$ and the set of nodes A we adopt $p : V \times 2^V \rightarrow \mathbb{N}$, such that $p(s, A) = \Delta(A) - d(s, A)$ where $\Delta(A) = \max_{v \in V} d(v, A)$.

We are now ready to define the problem we tackle in this paper.

Problem 1 (Dense Subgraphs with Attractors and Repulsers Problem (DSAR)). *Given a graph $G = (V, E)$, two sets of nodes $A, R \subset V$, $A \cap R = \emptyset$, and scalars $\lambda_1, \lambda_2 \in \mathbb{R}^+$, find*

$$S^* = \arg\max_{S \subseteq V} \frac{e(S) + \sum_{s \in S}(\lambda_1\, p(s, A) + \lambda_2\, d(s, R))}{|S|}$$

where $e(S) = |E(S)|$ is the number of edges in the subgraph induced by S, and $d, p : V \times 2^V \to \mathbb{N}$ are respectively distance and proximity functions as defined above.

Discussion on the objective function. Our objective function is a linear combination of the density, the proximity to the attractors A, and the distance from the repulsers R. In addition to the input sets A and R, each problem instance is defined by the scalars $\lambda_1, \lambda_2 \in \mathbb{R}^+$, which control the relative importance of proximity and distance, respectively. The quantitative relation between λ_1 and λ_2 directly affects the characteristics of the solution.

Since the distance function $d(s, R)$ considers the minimum distance of s from any node in R, our objective function favours nodes that are far from all the repulsers. If instead of the minimum we would have used the sum of distances from all repulsers, we could ended up, e.g., selecting nodes that are far from most of the repulsers but very close to one of them. This would not be adequate for our motivating problem of finding niches of users which are isolated from authoritative information sources. On the other hand, for what concerns proximity to A, our objective function favors nodes that are close to at least one of the attractors: again, this is consistent with our motivating problem of finding niches of users which are close, and thus influenceable, from a source of misinformation.

In the definition of the proximity function $p(s, A)$, we use the value $\Delta(A)$ to ensure non-negativity. We could have used any upper bound to $\Delta(A)$ for the same purpose (e.g., the diameter of G), without obtaining any effect on the definition of the problem. We use $\Delta(A)$ for a simple opportunistic reason: as we will see later in Section 4, we need to compute for each node in the graph its distance from A in any case.

To aggregate among all the nodes $s \in S$, we adopt the average of the distance and the proximity functions, (weighted by λ_1 and λ_2), which is consistent with the density term, defined as half the average degree within the solution S. The use of the average makes that there is no need to provide the size of the solution as input, given that the optimum is found balancing between the three functions at the numerator and the size of the solution itself.

4 COMPLEXITY AND ALGORITHMS

We first show that our problem is a special instance of a generalization of DSP first introduced by Goldberg in his seminal 1984 paper [27, Section 6], that we dub Heavy and Dense Subgraph Problem.

Problem 2 (Heavy and Dense Subgraph Problem (HDSP)). *Given an undirected graph (G, V, E, w_V, w_E) with no self-loops, where $w_V : V \to \mathbb{R}^+$ and $w_E : E \to \mathbb{R}^+$, find $S^* \subseteq V$ such that*

$$S^* = \arg\max_{S \subseteq V} \frac{e(S) + w_V(S)}{|S|},$$

where $e(S) = \sum_{e \in E(S)} w_E(e)$ and $w_V(S) = \sum_{s \in S} w_V(s)$.

Algorithm 1: Goldberg for DSAR

Input : $G(V, E)$, $w_E : E \to \mathbb{R}^+$, $\lambda_1 \geq 0$, $\lambda_2 \geq 0$, $A \subset V$, $R \subset V$
Output: $S^* \subseteq V$

```
w_V ← Vertex-Weights-Calculator(G, w_E, λ1, λ2, A, R);
S* ← Goldberg–HDSP(G, w_E, w_V);
return S*;
```

Algorithm 2: Vertex-Weights-Calculator

Input : $G(V, E)$, $w_E : E \to \mathbb{R}^+$, $\lambda_1 \geq 0$, $\lambda_2 \geq 0$, $A \subset V$, $R \subset V$
Output: $w_V : V \to \mathbb{R}^+$

```
V ← V ∪ {x};
foreach a ∈ A do E ← E ∪ {(a, x)}, w_E[(a, x)] ← 0;
Run Dijkstra Algorithm using x as source;
Δ(A) ← max_{v∈V} d_sp(v, x);
foreach v ∈ V \ {x} do w_V[v] ← λ1 · (Δ(A) − d_sp(v, x));
V ← (V \ {x}) ∪ {y} ;
foreach r ∈ R do E ← E ∪ {(r, y)}, w_E[(r, y)] ← 0;
Run Dijkstra algorithm using y as source;
foreach v ∈ V \ {y} do w_V[v] ← w_V[v] + λ2 d_sp(v, y);
return w_V;
```

The HDSP generalization of the DSP is solvable exactly in polynomial time [27]. Thus, we obtain the same result for DSAR. We state this as the next lemma.

Lemma 1. *The DSAR problem (Problem 1) is a special instance of the HDSP problem (Problem 2).*

Proof. It is easy to see that we can obtain Problem 1 as a special case of Problem 2, by setting $w_E(e) = 1$ for every $e \in E$, and $w_V(v) = \lambda_1 p(v, A) + \lambda_2 d(v, R)$ for every $v \in V$. The objective of Problem 1, for unweighted graphs, is clearly lower bounded by the objective value obtained for $S = V$, and it is (trivially) upper-bounded by $\frac{\binom{n}{2} + \lambda_1 n^2 + \lambda_2 n^2}{1} \leq (1 + \lambda_1 + \lambda_2) n^2 = O(n^2)$ for any constant values of λ_1, λ_2. Furthermore, it is straight-forward to prove that any two distinct values are separated by $\Omega(\frac{\min(1, \lambda_1, \lambda_2)}{n^2}) = O(n^{-2})$. □

We use Goldberg's algorithm, that performs $O(\log n)$ queries of the form $\exists S : e(S) + w_V(S) \geq \theta |S|$ using binary search on θ, to find an optimal set S^*. Goldberg's logarithmic upper bound on the number of queries holds also for non-negative weighted graphs with polynomially bounded edge weights, i.e., $\frac{1}{\text{poly}(n)} \leq w(e) \leq \text{poly}(n)$ for all edges $e \in E$.

Therefore it is possible to solve Problem 1 in polynomial time using Algorithm 1, that first computes the node weights using Algorithm 2, and then runs Goldberg's HDSP max-flow algorithm. The next proposition states the complexity of our algorithm.

Proposition 1. *Problem 1 is solvable in $O(T_{\text{max-flow}}(n, m) \cdot \log n)$ time, where $T_{\text{max-flow}}(n, m)$ is the runtime of a maximum flow computation in the standard RAM model.*

Notice that the total run time is dominated by the exact maximum flow computation; the best known exact algorithm for the maximum flow problem runs in $O(nm)$ [50], which makes the exact solution prohibitive for even medium-scale datasets.

This motivates us to develop an iterative greedy algorithm that converges fast to a near-optimal solution.

4.1 The *Down in the Hollow* (DITH) algorithm

Chekuri, Quanrud, and Torres [13] recently introduced the following problem.

Problem 3 (Densest Supermodular Subset problem (DSS)). *Given a normalized, non-negative, monotone, and supermodular set function defined over a ground set V ($f : 2^V \to \mathbb{R}^+$), find $S^* \subseteq V$ such that*

$$S^* = \arg\max_{S \subseteq V} \frac{f(S)}{|S|}.$$

Our next lemma shows that the HDSP problem is a special case of the DSS problem. This is a direct consequence of properties of supermodular functions, as discussed in Section 2.

Lemma 2. *HDSP (Problem 2) is a special instance of DSS (Problem 3).*

Proof. It suffices to prove that the functions defined in Problem 2, $e : 2^V \to \mathbb{R}_{\geq 0}$ and $w_V : 2^V \to \mathbb{R}_{\geq 0}$ are supermodular, normalized and non-negative set functions. By definition, both e and w_V are non-negative. Note that if $S = \emptyset$, $e(S) = w_V(S) = 0$. Furthermore, function e is a supermodular function. We conclude the lemma showing that w_V is modular. In fact, $\forall S, T \subseteq V$ it holds that

$$w_V(S) + w_V(T) = \sum_{s \in S} w_V(s) + \sum_{t \in T} w_V(t) =$$
$$= \sum_{x \in S \cup T} w_V(x) + \sum_{y \in S \cap T} w_V(y) = w_V(S \cup T) + w_V(S \cap T).$$

□

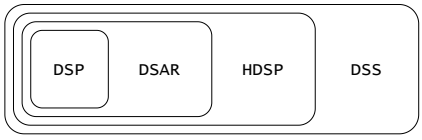

Figure 3: Problem hierarchy.

Figure 3 visualizes the hierarchy of problems in this work. To solve DSAR we can rely on any algorithm for HDSP: however, neither the max-flow algorithm, nor the $O(n^6)$ algorithm for supermodular maximization [55], scale to large real-world networks. Thanks to Lemma 2, we can adapt the framework by Chekuri et al. [13] for DSS, whose main result is reported next, to our setting. We use the following notation: we denote the marginal gain $f(S \cup \{v\}) - f(S)$ with $f(v|S)$, while OPT denotes the maximum of $f(S)/|S|$ over all $S \subseteq V$.

Theorem 4.1 (Chekuri-Quanrud-Torres). *Let $f : 2^V \to \mathbb{R}_{\geq 0}$ be a normalized, non-negative, and monotone supermodular function over the ground set $V = [n]$. Let $\epsilon \in (0, 1)$.* Super-Greedy++ *outputs a $(1 - \epsilon)$-approximate solution for DSS after $T > O\left(\frac{\Delta_f \log n}{OPT \epsilon^2}\right)$ iterations where, $\Delta_f = \max_{v \in V} f(v|V \setminus \{v\})$.*

Algorithm 3: *Down in the Hollow* (DITH)

Input : $G(V, E)$, $w_E : E \to \mathbb{R}^+$, $\lambda_1 \geq 0$, $\lambda_2 \geq 0$, $A \subset V$, $R \subset V$, $\gamma \in (0, 1)$, $T \in \mathbb{N} \setminus \{0\}$
Output: $S^* \subseteq V$

1 $w_V \leftarrow$ Vertex-Weights-Calculator$(G, w_E, \lambda_1, \lambda_2, A, R)$;
2 $S^* \leftarrow$HDSP-Super-Greedy++(G, w_E, w_V, γ, T);
3 **return** S^*;

Algorithm 4: HDSP-Super-Greedy++.

Input : $G(V, E)$, $w_E : E \to \mathbb{R}^+$, $w_V : V \to \mathbb{R}^+$, $\gamma \in (0, 1)$, $T \in \mathbb{N} \setminus \{0\}$
Output: $S^* \subseteq V$

1 **foreach** $v \in V$ **do** $\ell_v^{(0)} \leftarrow 0$;
2 $S^* \leftarrow V$, $n \leftarrow |V|$;
3 $LB \leftarrow \frac{e(S^*) + \sum_{s \in S^*} w_V(s)}{n}$, $UB \leftarrow +\infty$;
4 $t \leftarrow 0$;
5 **while** $\frac{LB}{UB} < (1 - \gamma) \wedge t < T$ **do**
6 $t \leftarrow t + 1$;
7 $S_{t,1} \leftarrow V$;
8 **for** $i \leftarrow 1$ *to* n **do**
9 **if** $\frac{e(S_{t,i}) + \sum_{s \in S_{t,i}} w_V(s)}{|S_{t,i}|} > LB$ **then**
10 $S^* \leftarrow S_{t,i}$;
11 $LB \leftarrow \frac{e(S^*) + \sum_{s \in S^*} w_V(s)}{|S^*|}$;
12 $v_{t,i} \leftarrow \arg\min_{v \in S_{t,i}} \ell_v^{(t-1)} + \delta_{S_{t,i}}(v) + w_V(v)$;
13 $\ell_{v_{t,i}}^{(t)} \leftarrow \ell_{v_{t,i}}^{(t-1)} + \delta_{S_{t,i}}(v_{t,i}) + w_V(v_{t,i})$;
14 $S_{t,i+1} \leftarrow S_{t,i} \setminus \{v_{t,i}\}$;
15 $UB \leftarrow \min\left\{UB, \frac{\max_{v \in V} \ell_v^{(t)}}{t}\right\}$;
16 **return** S^*;

We develop the algorithm DITH for the DSAR problem, that is shown as pseudocode in Algorithm 3. Our algorithm uses, as a blackbox, our version of Super-Greedy++ adapted to work for HDSP. In the pseudocode of Algorithm 4, $S_{t,i}$ (line 7) is the set of vertices at the i-th step of the t-th iteration. In particular, for every subset $S \subseteq V$ and for every $v \in S$, the marginal gain $f(v \mid S \setminus \{v\}) := f(S) - f(S \setminus \{v\})$ equals to $f(v \mid S \setminus \{v\}) = \delta_S(v) + \lambda_1 \cdot p(\{v\}, A) + \lambda_2 \cdot d(\{v\}, B)$, where $\delta_S(v)$ is the degree of v in the graph induced by S.

Algorithm 3 runs in near-linear time per iteration, and this is stated as the next lemma.

Lemma 3. *Algorithm 3 can be implemented to run in $O((m + n \log n)T)$ in the standard RAM model. Here, n, m, T are the number of nodes, edges, and rounds respectively.*

Proof. Algorithm 3 invokes two subroutines. The computational complexity of Algorithm 2 is dominated by the cost of computing all the needed shortest-path distances by means of Dijkstra's algorithm, which is $O(m+n \log n)$ [21]. For Algorithm 4 we implement a priority queue to find efficiently the vertex that gives the minimum marginal gain at each iteration of the peeling process. Before the beginning of the t-th peeling process, where $t = [T] := \{1, \ldots, T\}$, each vertex is inserted in an empty priority queue with the key

$f(v \mid V \setminus \{v\}) = \delta(v) + \lambda_1 \cdot p(v, A) + \lambda_2 \cdot d(v, B) + \ell_v^{(t-1)}$. At each iteration of the peeling process, the vertex with the smallest key is extracted from the priority queue. Consequently, to this extraction, the keys of all vertices in the priority queues that are adjacent to the extracted vertex are decreased by precisely the edge weight connecting the two vertices.

We observe that the computational cost of the algorithm is dominated by the cost of the operations on the priority queue. In particular, in a complete peeling process, the algorithm will perform n insert operations, n minimum extraction operations, and m key decrement operations. The number of key decrement operations follows from the observation that one key decrement is performed for each edge at most. By using a Fibonacci heap as priority queue, a complete peeling process terminates in $O(m + n \log n)$ computational steps. Since the number of times the peeling process is performed is bounded by T, we have the lemma. □

The next corollary on the performance of Algorithm 4 is obtained from Theorem 4.1.

Corollary 4.2. *Given any instance of Problem 2, Algorithm 4 outputs a solution that is at least at a factor* $\min\{(1-\gamma), (1-\epsilon)\}$ *from the optimum, where* $\epsilon = O\left(\sqrt{\frac{\Delta \log n}{OPT \cdot T}}\right)$, *with* $\Delta = \max_{v \in V} \delta(v) + w_V(v)$, *and OPT the value of the optimal solution.*

Proof. The corollary follows from Theorem 4.1, and the following observation.

Case I: If the algorithm terminates because $t \geq T$ then, according to Theorem 4.1, S^* is a $(1-\epsilon)$-approximate solution for the problem.

Case II: If the algorithm terminates because $\frac{LB}{UB} \geq (1-\gamma)$, then $LB \geq (1-\gamma)UB \geq (1-\gamma)OPT$. Note that LB is the value of the set S^*, which is the output of the algorithm. The algorithm returns a $(1-\gamma)$ solution. □

The correctness of the lower bound LB computed by Algorithm 4 is immediate; any subset $S \subseteq V$ is a feasible solution. In the following we prove the correctness of the upper bound computed by Algorithm 4. We define as UB_t the value of the variable UB at the end of the t-th peeling iteration in Algorithm 4, for $t \in [T]$. Our proof is based on linear programming duality. We follow Charikar's LP formulation for the DSP [11, 12], where the only variation is the addition of nodes' weights in the objective function. It is straightforward to notice that this LP formulation gives an exact polynomial-time algorithm for HDSP, which is an alternative to Goldberg's [27]. For sake of brevity we provide here only the dual, where $f_e(u)$ is the dual variable associated with the first $2m$ constraints of the form $y_e \leq x_u$. We refer to the following LP as Dual($HDSP$), and its optimum is equal to the optimal value ρ^*_{HDSP} of the primal.

$$
\begin{aligned}
\text{minimize} \quad & D \\
\text{subject to} \quad & f_e(u) + f_e(v) \geq w_E(e), && \forall e = uv \in E \\
& \ell_v := w_V(v) + \sum_{e \ni v} f_e(v) \leq D, && \forall v \in V \\
& f_e(u), f_e(v) \geq 0 && \forall e = uv \in E
\end{aligned}
$$

Theorem 1. *UB_t is an upper bound to the optimal value of Problem 1 $\forall t \in [T]$.*

Proof of Theorem 1. We will prove the theorem by showing that for every t, there exists a feasible solution of the dual that has value equal to UB_t. We will prove this by induction.
When $t = 1$, we have that

$$UB_1 = \max_{v \in V} \left(\sum_{j \in N(v)} w_E((v, j)) + w_V(v) \right),$$

where $N(v)$ is the set of neighbors of a vertex v. When a node u is removed, for all the remaining edges $e = (u, v)$ incident to it, we set $f_e(u) = w_E(e)$. After the end of the peeling, we set equal to 0 all variables $f_e(u)$ to which a value has not been assigned. We set the dual variable $D = \max_{v \in V} \left(\sum_{j \in N(v)} w_E((v, j)) + w_V(v) \right)$. The assigned values cause the variables to satisfy the constraints. The value of the solution is just equal to UB_1.

As we see, in line 14 of Algorithm 4, UB_{t+1} is equal to $\min\{UB_t, \max_{v \in V} \frac{\ell_v^{t+1}}{t+1}\}$.
Case I: If $UB_{t+1} = UB_t$, then the claim follows trivially from the inductive hypothesis.
Case II: If $UB_{t+1} = \max_{v \in V} \frac{\ell_v^{t+1}}{t+1}$, by the convexity structure of the polytope representing the feasible space of the dual problem, we have that any convex combination of dual feasible solutions is still a feasible solution for the dual. Note that the average is a convex combination. For each $i \in \{1, \ldots, H\}$ consider the dual feasible solutions $(D_i, f_e^i(v), \forall e \in E, \forall v \in e)$. Now, let us focus on this average: $\forall e \in E, \forall v \in e$, $\frac{f_e^i(v)}{H}$. Let us set D equal to the v that maximizes the following quantity: $w_V(v) + \sum_{v \ni e} \sum_{i \in \{1,\ldots,H\}} \frac{f_e^i(v)}{H}$.

Let's call the value assigned to D with D^*. The solution $\left(D^*, \frac{f_e^i(v)}{H} \ \forall e \in E, \forall v \in e\right)$ is feasible since $\forall e \in E, \forall v \in e \ f_e(v) \geq 0$ and $w_V(v) + \sum_{v \ni e} \sum_{i \in \{1,\ldots,H\}} \frac{f_e^i(v)}{H} \leq D^*$.

Observe that, setting $H = t + 1$,

$$
\begin{aligned}
&\max_{v \in V} \left(w_V(v) + \sum_{v \ni e} \sum_{i \in \{1,\ldots,t+1\}} \frac{f_e^i(v)}{t+1} \right) = \\
&= \max_{v \in V} \left(\frac{(t+1) w_V(v)}{(t+1)} + \sum_{v \ni e} \sum_{i \in \{1,\ldots,t+1\}} \frac{f_e^i(v)}{t+1} \right) = \\
&= \max_{v \in V} \left(\frac{(t+1) w_V(v) + \sum_{v \ni e} \sum_{i \in \{1,\ldots,t+1\}} f_e^i(v)}{t+1} \right) = \max_{v \in V} \frac{\ell_v^{t+1}}{t+1}
\end{aligned}
$$

The inductive step is proved. □

Corollary 4.2 implies the following corollary for DSAR (Problem 1).

Corollary 4.3. *Given any instance of Problem 1, Algorithm 3 outputs a solution that is at least at a factor* $\min\{(1-\gamma), (1-\epsilon)\}$ *from the optimum, where* $\epsilon = O\left(\sqrt{\frac{\Delta \log n}{OPT \cdot T}}\right)$, *with* $\Delta = \max_{v \in V} \delta(v) + \lambda_1 p(\{v\}, A) + \lambda_2 d(\{v\}, R)$, *and OPT the value of the optimal solution.*

Combining the above, we obtain the following result concerning the performance of our algorithm:

Fact 1. *Given any instance of Problem 1, if Algorithm 3 terminates before T iterations, then it returns a $(1-\gamma)$-approximate solution for the problem.*

It is worth recalling that all the results obtained for DSAR (Problem 1) hold also for the classic Densest Subgraph Problem (DSP), as it corresponds to the case $\lambda_1 = \lambda_2 = 0$.

Remarks. Algorithm 4 is an adaptation of the Super-Greedy++ for the HDSP problem with an efficient termination criterion. Our algorithm provides at each iteration a valid certificate of the quality of the best solution found for the HDSP instance in input. This certificate is obtained by executing a small number of computations, that compute the value of a feasible solution for the dual of the HDSP input instance. Notice that this enriches Super-Greedy++, whose approximation guarantee $O\left(\sqrt{\frac{\Delta_f \log n}{OPT \cdot T}}\right)$ depends on the unknown value OPT, and the constants are hidden by the big-O notation. The presence of the condition $\frac{LB}{UB} < (1-\gamma)$ in line 5 of Algorithm 4, together with the presence of the input parameter γ, allows the algorithm to stop its execution once the provided level of quality (represented by γ) is reached, avoiding useless further iterations. This is in contrast with Super-Greedy++, where the total number of iterations is always performed and no quantitative numeric evaluation of the solution quality is provided.

In all the performed experiments, our Algorithm 3 terminated by providing in output solutions with an approximation factor no worse than the one requested in input (through the parameter γ), before reaching the maximum number of T iterations.

4.2 Single-iteration peeling algorithm

As already mentioned in Section 1, Charikar's peeling algorithm is well-known for providing an efficient $\frac{1}{2}-$approximation for DSP [11, 12]. We next present an analysis of its application to HDSP. This corresponds to running only one iteration of our Algorithm 3 (i.e., $T = 1$), except for the computation of the upper bound to the input problem instance (that requires only constant time). This algorithm is also used in the experiments in Section 5, where it is named dith-1.

Lemma 4. *Charikar's greedy peeling algorithm for the HDSP (i.e., Algorithm 3 with $T = 1$) achieves an approximation factor*

$$\frac{1}{\min\left\{1 + \frac{\rho^*}{w_V^{MIN}}, 2 - \frac{w_V^{MIN}}{\rho^*_{HD}}\right\}},$$

where $\rho^* = \max_{S\subseteq V} \frac{e(S)}{|S|}$, $\rho^*_{HD} = \max_{S\subseteq V} \frac{e(S)+w_V(S)}{|S|}$, and w_V^{MIN} is the smallest weight that has a vertex.

Proof. Chekuri et al. [13] show that if f is a non-negative monotone supermodular set function such that $f(\emptyset) = 0$, then the adaption of Charikar's algorithm to the DSS problem has an approximation ratio of at least $\frac{1}{c_f}$, where $c_f := \max_{S\subseteq V} \frac{\sum_{v\in S} f(v|S\setminus\{v\})}{f(S)}$. Since HDSP problem is a special version of DSS, let us calculate c_f for HDSP. We have that

$$c_f = \max_{S\subseteq V} \frac{\sum_{v\in S}\left(\delta_{G(S)}(v) + w_V(v)\right)}{e(S)+w_V(S)} = \max_{S\subseteq V} \frac{2e(S)+w_V(S)}{e(S)+w_V(S)} =$$
$$= \max_{S\subseteq V}\left(1 + \frac{e(S)}{e(S)+w_V(S)}\right) = \max_{S\subseteq V}\left(1 + \frac{e(S)}{e(S)+w_V(S)}\frac{|S|}{|S|}\right) \leq$$
$$\leq \max_{S\subseteq V}\left(1 + \frac{e(S)}{w_V(S)}\frac{|S|}{|S|}\right) \leq 1 + \frac{\rho^*}{w_V^{\text{MIN}}},$$

where $\rho^* = \max_{S\subseteq V} \frac{e(S)}{|S|}$ and w_V^{MIN} is the smallest weight that has a vertex. For the other direction, we obtain

$$c_f = \max_{S\subseteq V} \frac{2e(S)+w_V(S)}{e(S)+w_V(S)} = \max_{S\subseteq V} \frac{2e(S)+2w_V(S)-w_V(S)}{e(S)+w_V(S)} =$$
$$= \max_{S\subseteq V}\left(2 - \frac{w_V(S)}{e(S)+w_V(S)}\right) = \max_{S\subseteq V}\left(2 - \frac{w_V(S)}{e(S)+w_V(S)}\frac{|S|}{|S|}\right) \leq$$
$$\leq \max_{S\subseteq V}\left(2 - \frac{|S| w_V^{\text{MIN}}}{e(S)+w_V(S)}\right) = 2 - \frac{w_V^{\text{MIN}}}{\rho^*_{\text{HD}}},$$

where $\rho^*_{\text{HD}} = \max_{S\subseteq V} \frac{e(S)+w_V(S)}{|S|}$. □

Note that, for every $S \subseteq V$ such that $S \neq \emptyset$, $w_V(S) \geq |S| w_V^{\text{MIN}}$, therefore $0 \leq \frac{w_V^{\text{MIN}}}{\rho^*_{\text{HD}}} \leq \frac{|S| w_V^{\text{MIN}}}{e(S)+|S| w_V^{\text{MIN}}} \leq 1$.

Equivalently, $1 \leq \min\left\{1 + \frac{\rho^*}{w_V^{\text{MIN}}}, 2 - \frac{w_V^{\text{MIN}}}{\rho^*_{\text{HD}}}\right\} \leq 2$.

Given to the relation between HDSP and DSAR, we can state the following corollary.

Corollary 4.4. *Consider an instance $(G, w_E, A \subseteq V, R \subseteq V, \lambda_1 \geq 0, \lambda_2 \geq 0)$ for the DSAR problem. Let $v \in V$ be the node that minimizes $h(v) := \lambda_1 p(v, A) + \lambda_2 d(v, R)$.*

The extension of the Charikar's algorithm for the DSAR problem has an approximation factor of

$$\frac{1}{\min\left\{1 + \frac{\rho^*}{h(v)}, 2 - \frac{h(v)}{\rho^*_{DSAR}}\right\}},$$

where ρ^* is the density of the densest subgraph and ρ^*_{DSAR} is the value of the optimum solution.

By setting $T = 1$ in Lemma 3, we see that the run time is $O(m + n \log n)$.

5 EXPERIMENTS

In this section we provide an extended experimental analysis that examines multiple aspects of our problem statement. In particular, in §5.1 we present characteristics of the solutions for different problem instances (i.e., varying λ_1 and λ_2), in §5.2 we analyze scalability and convergence of our algorithms, in §5.3 we compare the performance of our algorithms against a variety of non-trivial baselines. Finally, in §5.4 we present two case studies on real-word datasets.

Settings. All the experiments run on an Intel Xeon 2.3GHz with 48GB of RAM and a Linux Ubuntu 20.04 LTS Operating System. We exclude from the reported runtimes of all algorithms and baselines the time required by the subroutine Vertex-Weights-Calculator. Furthermore, in order to provide an intuitive comparison between

different problem instances, in the whole section we report the min-max normalized values for Avg. Degree, Avg. Proximity and Avg. Distance, taking respectively as maximum values the Avg. Degree of the Densest Subgraph, the Proximity of any node in A, and the Distance of the furthest node in the graph from R. Finally, with $\rho_{\text{DITH.01}}$ we are indicating the average degree of a 0.99-approximation of the densest subgraph computed by DITH ($\gamma = 0.01$) on the considered network.

Datasets. Table 1 lists the datasets used with the subsection in which they are used. In §5.1 and §5.3, where the characteristics of the solutions is an important factor, we use datasets related to the motivating application of this work: the detection of polarized niches. Specifically, we take the largest connected component from five networks from Garimella et al. [24], that represent the follow-networks on Twitter among users debating on controversial topics over a specific period of time: two nodes (users) are connected by an undirected edge if there is at least one follow relationship between them in Twitter. Garimella et al. partition the set of users into two parts [24], that have an opposite opinion around each topic of interest; we use these partitions to guide our selection of attractors and repulsers, by choosing nodes from different parts respectively. In §5.2, to prove the efficiency of our method and its fast convergence, we take the largest connected component of the undirected and unweighted network employed in [11]: we take those for which Goldberg's algorithm failed to run in order to show that our method can provide a near-optimal solution with a certificate of its approximation quality, and the others to show that DITH scales efficiently as the size of networks increases. For what concern the latter, we report results only for those networks whose runtime of Algorithm 1 was lower than 2 hours. Finally, in the use cases in §5.4, we employ two small social networks for which the identity of the nodes is known. We exploit this information in order to provide meaningful sets A and R, showing how the polarized niches can be effectively found.

5.1 Characteristics of the solutions

We next discuss the effect of λ_1 and λ_2 on the characteristics of the solutions of DSAR. Notice that different values of λ_1 and λ_2 define different problem instances, and thus are not directly comparable. Also, the actual values of λ_1 and λ_2 do not affect the computational complexity of the problem as it remains solvable in polynomial time, but they affect the actual runtime. We create 1 000 random instances for any possible combination of λ_1 and λ_2, with $\lambda_1, \lambda_2 \in \{i \cdot \frac{\lambda^{max}}{25} : i \in \{0, 1, \ldots, 25\}, \lambda^{max} = 6\rho_{\text{DITH.01}}\}$ and $|A|, |R| \in \{1, 5, 10, 20, 50, 100\}$, and we solve them with Algorithm 3, setting $\gamma = 0.01$ and $T = 10\,000$. A and R are chosen uniformly at random from the two parts of the partitions provided by [24]. We report in each cell of the heatmap (representing a single combination of λ_1 and λ_2) the average values over the relative 36 000 different instances for the following metrics: approximation factor, runtime, Norm. Avg. Degree, Norm. Avg. Proximity and Norm. Avg. Distance. The results are similar for all datasets, and thus we report the representative results for the BALTIMORE dataset (Figure 4).

In all of its executions DITH terminated before $T = 10\,000$ iterations, therefore, according to Fact 1 each provided solution is a $(1 - \gamma)$-approximation of the optimal solution. The algorithm is extremely fast, taking at most 13msec on average for solving a single instance; in addition, increasing λ_1 or λ_2 decreases the total runtime. We report also results in terms of normalized Avg. Degree, Avg. Proximity and Avg. Distance. By ranging the values of λ_1 and λ_2, we observe how Problem 1 gives different importance to the terms of the objective: the three heatmaps related to these metrics are complimentary with each other, showing both the existence of solutions that excel in only one of them, and solutions that try to combine the 3 different components. This flexibility can be of utmost importance to satisfy the desired characteristics of the output by the user.

Table 1: Datasets used in the experimental analysis.

	Id	Dataset	$\lvert V\rvert$	$\lvert E\rvert$
§5.2	W1	WEBTRACKERS [39]	27665729	140613747
	O1	ORKUT [39]	3072441	117184899
	L1	LIVEJOURNAL-AFFILIATIONS [39]	7489073	112305407
	W2	WIKI-TOPCATS [40]	1791489	25444207
	C1	CIT-PATENTS [40]	3764117	16511740
	W3	WEB-STANFORD [40]	255265	1941926
	E1	EGO-TWITTER [40]	81306	1342310
	C2	COM-DBLP [40]	317080	1049866
	C3	COM-AMAZON [40]	334863	925872
	S1	SOC-SLASHDOT0902 [40]	82168	582533
	S2	SOC-SLASHDOT0811 [40]	77360	546487
	S3	SOC-EPINIONS [40]	75877	405739
	E2	EMAIL-ENRON [40]	33696	180811
	E3	EGO-FACEBOOK [40]	4039	88234
	P1	PPI [60]	6944	42774
§5.1, §5.3	L2	LEADERSDEBATE [24]	9566	344088
	G1	GUNSENSE [24]	1821	103840
	B1	BALTIMORE [24]	1441	28291
	R1	RUSSIA_MARCH [24]	1189	16471
	B2	BEEFBAN [24]	799	6026
§5.4	G2	GREEK_PARLIAMENT [59]	185	17185
	V1	VAXNOVAX [17]	200	5806

5.2 Scalability and convergence

Scalability. Results of Section 4 prove theoretically the greater efficiency of our method w.r.t. the state-of-the-art. In order to show it empirically, we perform experiments either over the real-world large graphs used in [11] and syntectic graphs.

For what concerns the synthetic graphs, we generate them according to a simple Stochastic Block Model made of 2 blocks, with parameters set such that they result densely connected inside, and sparsely connected between them (further details in caption of Figure 5). This model mimics controversial graphs made of two groups of users with a different stance on a specific topic, and allows us to place A and R separated in the 2 different blocks of the graph, generating problem instances more coherent with our motivating application. We generate 10 different graphs for each value of $|V|$ (ranged from 10 to 50000) and we sample A and R at random with the constraint that they must belong to different blocks of the Stochastic Block Model.

For what concerns the real-world graphs, we employ each graph in 10 different problem instances, with A and R sampled at random. In both class of graphs, for any single problem instance we set $\lambda_1 = \lambda_2 = 1$, since according to our analysis reported in Section 5.1 it was the most expensive setup in terms of runtime for our algorithms, and $|A| = |R| = 1$. We compare DITH (setting $\gamma = 0.01$ and $T = 10\,000$) with Algorithm 1, that solves Problem 1 to the optimum.

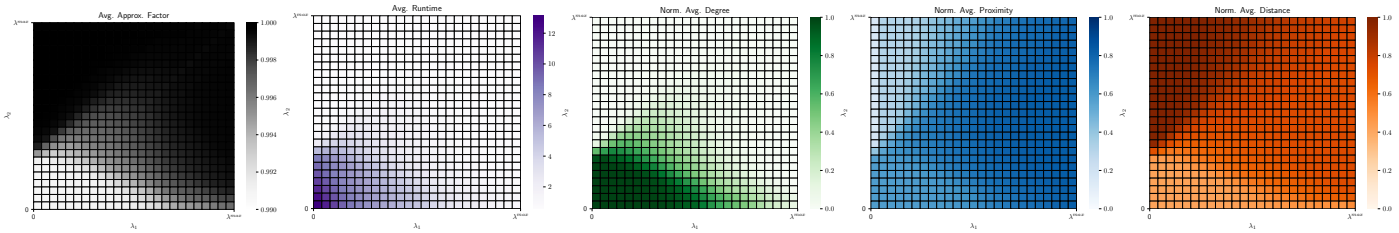

Figure 4: Average values of Approximation Factor, Running Time in milliseconds, Norm. Avg. Degree, Norm. Avg. Proximity and Norm. Avg. Distance, for different combinations of λ_1 and λ_2, for the BALTIMORE dataset.

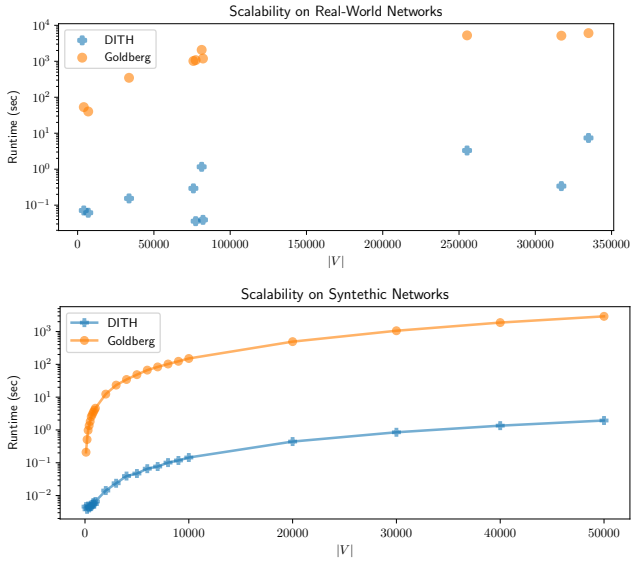

Figure 5: Scalability of DITH for real-world networks (top) and syntethic networks (bottom), y-axes are reported in log-scale. Real-world networks are those employed in [11], and results are showed only for those networks whose average runtime was less than 2 hours for Goldberg's algorithm. Syntethic networks are generated according to Stochastic Block Model with probability of an edge within a block of $4\frac{n}{2}/\binom{\frac{n}{2}}{2}$ and between blocks of $0.1/\frac{n}{2}$.

Figure 5 summarizes our experiment, and confirms evidently the results coming from the theoretical analysis. All the solutions returned by our method were at most at 0.01 far apart from the optimum, since for any problem instance DITH terminated in less than T iterations: allowing the output to be almost-optimal, the runtime of DITH stays stable either in real-world network and synthetic network as the number of nodes grows, while it increases rapidly for Algorithm 1.

Convergence As pointed out in Section 4.1, DITH guarantees both for DSAR and DSP a lower-bound on the quality of the solution computed at any iteration, and consequently on the returned one. We stress particularly this result, since DITH can embed a termination condition, according to the desired quality of the output. In order to showcase the practical relevance of the early termination device, we use the five largest graphs from Boob et al. [11] for which the maximum-flow approach fails due to memory consumption and study over them the performance of DITH for both DSAR and DSP.

To analyze it over DSAR problem, we generate 100 different problem instances for any dataset: each instance is defined by sets A and R sampled at random in the graph ($|A| \in \{1,2,3,4,5\}$, $|R| \in \{1,2,3,4,5\}$), and λ_1 and λ_2 chosen at random ($\lambda_1 \in (0, \lambda^{max}]$, $\lambda_2 \in (0, \lambda^{max}]$, $\lambda^{max} = \rho_{\text{DITH.01}}$). We solve all of them with three different variants of Algorithm 3: in the first we set $\gamma = 0.01$ (dubbed DITH.01), in the second we set $\gamma = 0.1$ (dubbed DITH.1) and in the last setting $T = 1$ (dubbed DITH-1). Results of this experiment are summarized in Table 2. We can see how the computation of the lower bound is efficient in early stopping DITH, letting it performing at most 3 iterations in all datasets for achieving a solution whose quality is at least the 90% w.r.t. the optimal one. For the datasets whose computation of DITH.01 takes more iterations, we can notice either that the running time of a single iterations does not exceed 35 seconds in any of the dataset and that the performance achieved by DITH-1 is almost identical in terms of quality, making also in this scenario the computation extremely handy.

Table 2: Convergence analysis: values reported refer to average value and standard deviation over 100 different instances. On the left of the table, the dataset Id as defined in Table 1.

	Obj. value	$\frac{LB(\gamma)}{UB(\gamma)}$	$\frac{LB(\gamma)}{UB(\gamma=0.01)}$	Iterations	RunTime (sec)	
W1	458.60 ± 58.76	0.99 ± 0.00	0.99 ± 0.00	29.66 ± 5.61	1009.32 ± 185.20	DITH.01
	458.15 ± 57.69	0.91 ± 0.01	0.99 ± 0.00	3.03 ± 0.17	101.94 ± 5.92	DITH.1
	457.58 ± 56.36	0.78 ± 0.01	0.98 ± 0.00	1.00 ± 0.00	35.04 ± 0.77	DITH-1
O1	335.42 ± 44.35	0.99 ± 0.00	0.99 ± 0.00	7.84 ± 1.58	87.89 ± 16.69	DITH.01
	335.05 ± 43.80	0.93 ± 0.01	0.99 ± 0.00	1.00 ± 0.00	11.89 ± 0.37	DITH.1
	335.05 ± 43.80	0.93 ± 0.01	0.99 ± 0.00	1.00 ± 0.00	11.58 ± 0.36	DITH-1
L1	159.83 ± 21.67	0.99 ± 0.00	0.99 ± 0.00	4.50 ± 0.87	55.77 ± 9.79	DITH.01
	159.83 ± 21.67	0.97 ± 0.00	0.99 ± 0.00	1.00 ± 0.00	13.21 ± 0.45	DITH.1
	159.83 ± 21.67	0.97 ± 0.00	0.99 ± 0.00	1.00 ± 0.00	13.05 ± 0.43	DITH-1
W2	102.08 ± 12.27	0.99 ± 0.00	0.99 ± 0.00	21.74 ± 5.60	56.03 ± 14.25	DITH.01
	101.45 ± 11.65	0.92 ± 0.01	0.98 ± 0.00	2.75 ± 0.43	7.88 ± 1.20	DITH.1
	99.64 ± 11.85	0.76 ± 0.02	0.96 ± 0.01	1.00 ± 0.00	2.71 ± 0.14	DITH-1
C1	57.00 ± 7.00	0.99 ± 0.00	0.99 ± 0.00	22.93 ± 5.68	133.23 ± 31.23	DITH.01
	56.85 ± 6.78	0.92 ± 0.01	0.98 ± 0.00	2.03 ± 0.17	12.68 ± 0.99	DITH.1
	56.79 ± 6.69	0.70 ± 0.02	0.98 ± 0.01	1.00 ± 0.00	6.40 ± 0.21	DITH-1

For the classic DSP problem (corresponding to DSAR with $\lambda_1 = \lambda_2 = 0$), we summarize in Figure 6 the performance of DITH: the x, y-axes correspond to the number of iterations t and the ratio $LB(\gamma)/UB(\gamma)$ respectively. Algorithm 3 achieve an high quality solution for all the datasets after few iterations.

This result allows us to assess the approximation quality of the output of the iterative algorithm, *without* knowing the exact solution, over some of the largest graphs tested so far in the literature for this classic problem.

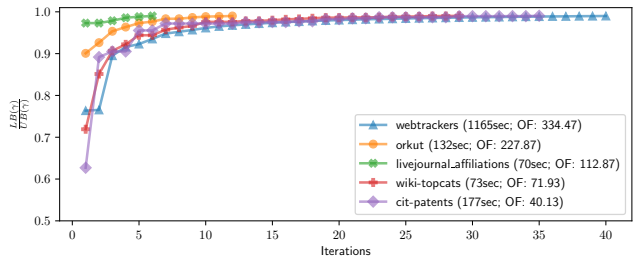

Figure 6: Convergence analysis for DSP. Plot shows the growth of the ratio $\frac{LB(\gamma)}{UB(\gamma)}$ vs. the number of iterations. The legend reports the runtime required to obtain a solution for DSP with a guarantee on its 0.99-approximation quality, and its relative objective function value (i.e., avg. degree).

5.3 Performance analysis

We next report the performance of our algorithms (i.e., DITH and DITH-1) in comparison with the following non-trivial baselines:

BRW: This baseline associates to each node in the graph a score that is the result of a random walk biased on the node weights, as described in [53], where at each node $u \in V$ is associated a weight equal to the following quantity: $\lambda_1\ p(\{v\}, A) + \lambda_2\ d(\{v\}, R)$. For each value of $k \in [|V|]$, the induced subgraph by the top-k nodes according to that score is considered as a solution for Problem 1. Among all these $|V|$ solutions, the one with maximum objective function value is returned as best solution for Problem 1.

SUBTRACTION-PR: This baseline associates to each node in the graph a score that is the difference of two topic-sensitive PageRank [31] scores: the one considering the set A as a topic, minus the one considering the set R as a topic. For each value of $k \in [|V|]$, the induced subgraph by the top-k nodes according to the just computed score is considered as a solution for Problem 1. Among all these $|V|$ solutions, the one with maximum objective function value is returned as best solution for Problem 1.

EGO-PROX-DS: As first step, this baseline simply extracts the subgraph induced by all nodes connected by at least one node in A, together with all nodes in A itself. Given this subgraph, the baseline returns the approximated densest subgraph executing Algorithm 4, with $\lambda_1 = \lambda_2 = 0$ and $\gamma = 0.01$.

Furthermore, we provide as benchmarks, results from the following methods based on DITH:

DS: Execution of Algorithm 3, with $\lambda_1 = \lambda_2 = 0$ and $\gamma = 0.01$.

DITH-NO-DIST: Execution of Algorithm 3, with $\lambda_2 = 0$ and $\gamma = 0.01$.

DITH-NO-PROX: Execution of Algorithm 3, with $\lambda_1 = 0$ and $\gamma = 0.01$.

In order to perform an exhaustive comparison, we generated for each dataset 1 000 different problem instances, taking λ_1 and λ_2 randomly in the interval $[0, \lambda^{max}]$, $\lambda^{max} = 3\rho_{\text{DITH.01}}$, and A and R randomly from the two distinct parts of the graph, with $|A|, |R| \in \{1, 2, 3, 4, 5\}$. Table 3 summarizes the performances by reporting for each method across all datasets the mean ± the standard deviation for the objective function, the runtime, the three terms involved in the objective function, and the number of connected components.

In addition to the afore-mentioned baselines, we run also the exact Algorithm 1: for sake of space we do not report these results which are almost-identical to those of DITH. Our algorithm provides the optimal solution for the 96% of the instances tested, achieving at least a 0.99-approximation for the remaining ones. The only difference is in terms of runtime, which for DITH is reduced by the 99.9% with respect to Algorithm 1. Overall, we observe that our proposed methods can effectively optimize the objective function, outperforming all of the other baselines even with just 1 iteration. DITH runs on average 2.5× faster than all of the other methods. The only method with competitive runtimes to ours is EGO-PROX-DS, which is also based on the early stopping condition provided in line 5 of Algorithm 4, but that only achieve on average a 0.8-approximation of the optimal solution. We also observe that DITH's runtime compared to DITH-1 is on average 7× greater, suggesting that our termination criterion has a significant effect in practice.

Looking at the three terms composing the objective function, we can see how DITH and DITH-1 are able to handle well the trade-off between all the three quantities. Comparing our solutions to the benchmarks (i.e., DS, DITH-NO-DIST and DITH-NO-PROX), we see how our algorithm works efficiently in practice, providing solutions that on average are able to have high values for all the 3 quantities. Our method exhibit solutions whose Distance on average is greater with respect to those of the 3 baselines (i.e, BRW, DITH-NO-DIST and DITH-NO-PROX). BRW returns in output on average denser solutions lacking of Distance, while the other 2 baselines base their search more only on Proximity. We provide a final remark over the connectivity of the solutions in output: one advantage in running DITH instead of DITH-1 lies exactly in the sensible reduction on the number of connected components given in output, that become comparable to those of the other methods, and close to 1 in general for almost all the baselines.

5.4 Case studies

We next present two case studies, over smaller Twitter datasets, on which the identity of the nodes is known. In both networks nodes represent users, while an edge between two users is weighted according to the Jaccard coefficient of the followers of the two users. This strategy has been successfully adopted in the work by Stamatelatos et al. [59], from which we borrow the GREEK_PARLIAMENT dataset, where nodes corresponds to accounts of either authorities of the Greek Parliament or newspapers. The other dataset, VAXNOVAX, is borrowed from [17], and contains the nodes active in the debate around kids vaccination in the Italian Twitter-sphere. As the edges of these graphs are weighted, we consider $1/w(u, v)$, as the length[2] of the edge (u, v), or the distance between the two connected nodes u and v.

We present the solutions found by our algorithm, by selecting meaningful A and R based on our prior knowledge of these two domains. Given A and R we can still have many different problem instances, and thus different solutions, by varying the importance of the three objectives. We next details how the solution presented in this section were selected: a potential user interested in finding polarization niches can follow the same approach. First, we perform a grid search over different instances varying λ_1 and λ_2, with $\lambda_1, \lambda_2 \in \{i \cdot \frac{\lambda^{max}}{200} : i \in \{0, 1, \ldots, 200\}, \lambda^{max} = 10\rho_{\text{DITH.01}}\}$

[2]It is trivial to notice that all findings reported in this manuscript are valid also when the network is positively weighted on the edges.

Table 3: Results of the comparative analysis between DITH, DITH-1, baselines and benchmarks. We report average and standard deviation over 1000 different problem instances.

	Obj. Function	RunTime (msecs)	Norm. Avg. Deg.	Norm. Avg. Prox.	Norm. Avg. Dist.	Conn. Comp.	
BALTIMORE	**120.31 ± 34.97**	4.37 ± 3.27	0.56 ± 0.49	0.69 ± 0.25	0.56 ± 0.24	1.18 ± 0.93	DITH
	119.58 ± 35.02	**0.66 ± 0.63**	0.69 ± 0.45	0.69 ± 0.20	0.50 ± 0.22	1.52 ± 1.86	DITH-1
	111.78 ± 28.73	13.06 ± 0.90	1.00 ± 0.00	0.63 ± 0.06	0.36 ± 0.06	1.00 ± 0.00	DS
	111.82 ± 29.07	17.16 ± 0.7	0.98 ± 0.02	0.64 ± 0.06	0.37 ± 0.06	1.1 ± 0.34	BRW
	110.3 ± 39.25	10.76 ± 0.81	0.35 ± 0.39	0.83 ± 0.18	0.52 ± 0.14	1.22 ± 0.69	SUB-PR
	94.15 ± 36.37	2.29 ± 3.50	0.29 ± 0.22	0.82 ± 0.03	0.39 ± 0.10	1.0 ± 0.05	EGO-PROX-DS
	113.29 ± 30.71	7.53 ± 2.8	0.92 ± 0.26	0.67 ± 0.11	0.38 ± 0.08	1.02 ± 0.22	DITH-NO-DIST
	105.79 ± 25.2	6.43 ± 5.17	0.54 ± 0.50	0.39 ± 0.28	0.66 ± 0.32	2.71 ± 3.52	DITH-NO-PROX
BEEFBAN	**50.72 ± 14.67**	1.8 ± 1.32	0.66 ± 0.46	0.7 ± 0.27	0.47 ± 0.24	1.05 ± 0.27	DITH
	50.38 ± 14.7	**0.26 ± 0.44**	0.78 ± 0.4	0.7 ± 0.23	0.42 ± 0.21	1.21 ± 0.65	DITH-1
	47.76 ± 13.68	4.76 ± 0.87	1.00 ± 0.0	0.69 ± 0.09	0.31 ± 0.09	1.0 ± 0.0	DS
	42.04 ± 12.91	3.86 ± 0.37	0.91 ± 0.05	0.7 ± 0.08	0.33 ± 0.08	1.1 ± 0.39	BRW
	40.63 ± 15.71	4.45 ± 0.6	0.50 ± 0.40	0.83 ± 0.13	0.43 ± 0.1	1.19 ± 0.76	SUB-PR
	41.18 ± 16.69	0.51 ± 0.8	0.28 ± 0.23	0.88 ± 0.02	0.38 ± 0.09	1.0 ± 0.07	EGO-PROX-DS
	48.44 ± 14.26	2.85 ± 1.21	0.93 ± 0.24	0.73 ± 0.11	0.32 ± 0.1	1.04 ± 0.29	DITH-NO-DIST
	43.89 ± 11.09	2.44 ± 2.14	0.52 ± 0.50	0.37 ± 0.35	0.65 ± 0.34	1.01 ± 0.21	DITH-NO-PROX
GUNSENSE	**259.93 ± 78.86**	12.64 ± 9.38	0.57 ± 0.47	0.71 ± 0.24	0.59 ± 0.2	1.13 ± 0.72	DITH
	258.69 ± 79.2	**1.68 ± 0.72**	0.65 ± 0.45	0.71 ± 0.21	0.55 ± 0.18	1.18 ± 0.86	DITH-1
	236.8 ± 67.82	40.2 ± 1.32	1.00 ± 0.0	0.57 ± 0.13	0.43 ± 0.14	1.0 ± 0.0	DS
	237.97 ± 69.43	61.51 ± 1.57	0.81 ± 0.22	0.61 ± 0.1	0.49 ± 0.11	1.14 ± 0.43	BRW
	249.65 ± 81.18	28.7 ± 2.0	0.52 ± 0.40	0.78 ± 0.17	0.56 ± 0.11	1.19 ± 0.57	SUB-PR
	224.51 ± 79.77	11.37 ± 11.83	0.46 ± 0.25	0.76 ± 0.04	0.5 ± 0.11	1.0 ± 0.0	EGO-PROX-DS
	248.7 ± 75.38	21.46 ± 10.57	0.79 ± 0.39	0.68 ± 0.19	0.46 ± 0.14	1.11 ± 0.54	DITH-NO-DIST
	229.43 ± 63.28	19.51 ± 14.14	0.61 ± 0.48	0.4 ± 0.27	0.68 ± 0.28	1.43 ± 2.23	DITH-NO-PROX
LEADERSDEBATE	**271.86 ± 79.47**	54.71 ± 38.69	0.57 ± 0.49	0.69 ± 0.24	0.57 ± 0.18	1.3 ± 1.5	DITH
	270.17 ± 79.21	**7.56 ± 1.15**	0.7 ± 0.44	0.66 ± 0.21	0.53 ± 0.15	1.78 ± 4.99	DITH-1
	254.16 ± 65.28	176.74 ± 9.45	1.00 ± 0.0	0.55 ± 0.08	0.45 ± 0.08	1.0 ± 0.0	DS
	241.79 ± 64.84	253.67 ± 11.47	0.85 ± 0.01	0.57 ± 0.05	0.45 ± 0.05	1.0 ± 0.0	BRW
	250.35 ± 90.01	116.31 ± 5.1	0.32 ± 0.37	0.83 ± 0.19	0.54 ± 0.1	1.4 ± 5.33	SUB-PR
	211.22 ± 82.77	8.86 ± 18.44	0.23 ± 0.20	0.79 ± 0.03	0.45 ± 0.07	1.0 ± 0.03	EGO-PROX-DS
	262.35 ± 73.6	94.78 ± 45.65	0.78 ± 0.40	0.66 ± 0.19	0.47 ± 0.08	1.22 ± 0.76	DITH-NO-DIST
	245.12 ± 57.47	88.26 ± 57.11	0.65 ± 0.47	0.41 ± 0.23	0.65 ± 0.26	2.57 ± 7.01	DITH-NO-PROX
RUSSIA_MARCH	**70.64 ± 21.19**	3.33 ± 2.26	0.6 ± 0.45	0.72 ± 0.22	0.62 ± 0.15	1.1 ± 0.47	DITH
	70.32 ± 21.15	**0.45 ± 0.51**	0.69 ± 0.41	0.72 ± 0.17	0.59 ± 0.13	1.25 ± 1.18	DITH-1
	63.02 ± 18.43	12.42 ± 1.0	1.00 ± 0.0	0.54 ± 0.12	0.46 ± 0.12	1.0 ± 0.0	DS
	64.12 ± 17.55	9.91 ± 0.42	0.90 ± 0.11	0.6 ± 0.07	0.49 ± 0.08	1.1 ± 0.3	BRW
	68.71 ± 20.75	7.32 ± 0.5	0.65 ± 0.39	0.75 ± 0.16	0.57 ± 0.1	1.13 ± 0.53	SUB-PR
	60.06 ± 22.03	1.1 ± 1.23	0.38 ± 0.23	0.8 ± 0.03	0.53 ± 0.08	1.0 ± 0.0	EGO-PROX-DS
	68.34 ± 19.91	5.77 ± 2.7	0.85 ± 0.31	0.67 ± 0.14	0.52 ± 0.1	1.07 ± 0.41	DITH-NO-DIST
	63.98 ± 16.75	5.18 ± 3.28	0.69 ± 0.44	0.46 ± 0.25	0.66 ± 0.23	1.5 ± 2.63	DITH-NO-PROX

and execute Algorithm 3 ($\gamma = 0.01, T = 10\,000$) for all of them. Then, to reduce the volume of the possible solutions, we remove any duplicates and filter them by only taking the solutions that are not dominated by any other solution in terms of Avg. Degree, Avg. Proximity and Avg. Distance (see Figure 7). Finally, we perform a manual inspection to select qualitatively one solution among all the provided ones; for these particular case studies, the selected solutions are characterized by large values of Avg. Proximity and Avg. Distance and an ample Avg. Degree. All the executions of Algorithm 3 ended before T iterations, thus according to Fact 1 all the solutions analyzed are 0.99-approximation to the optimal ones for any specific problem instances solved. The runtime requested for the grid search in GREEK_PARLIAMENT were respectively 17 and 24 seconds for the 2 different configurations of A and R, while 8 and 10 seconds for VAXNOVAX.

Greek parliament. In the Greek parliament Twitter-sphere network, we leverage the different political orientations of two distinct news outlets: the tabloid *Makeleio*, closer to the far-right and known for spreading racist, antisemitic, and homopohobic content, and the center-left newspaper *Efimerida Syntakton*, one of the main information sources of Greece, producing pro-government content during the years of Tsipras' govern. The solutions produced for this network, were already presented in Figures 1 and 2 at the beginning of the paper, and they reveal niches with opposite characteristics: 10 former politicians of *Golden Dawn* in Figure 1, and 15 of the main newspapers available in Greece in Figure 2. Looking at this result, it is worth discussing an emerging sociological aspect. Recall that edges in this graph reflect the similarity of Twitter's audience of 2 users, thus a dense structure in this context indicates the presence of a cluster of users that found interest the same Twitter accounts. The two solutions in Figures 1 and 2 show that while Twitter's audience around the tabloid *Makeleio* has an information diet mostly based on content coming from individuals that reflect their ideas and beliefs, users interested in *Efimerida Syntakton* also follow other authoritative media outlets, thus being exposed to a wider spectrum of opinions and higher quality information.

VaxNoVax. In 2017 a state law (Lorenzin, by the name of the proposer) made compulsory 10 different vaccinations in Italy for kids in school-age, generating a large societal debate [17]. Following the approach of [59], we reduce the dimension of the graph, by projecting it onto the 200 nodes with the highest in-degree (i.e., number of followers) in the original graph. Differently from [59], this graph was built without a prior personal knowledge on the users, thus with possibly more noisy accounts/edges. In order to better filter

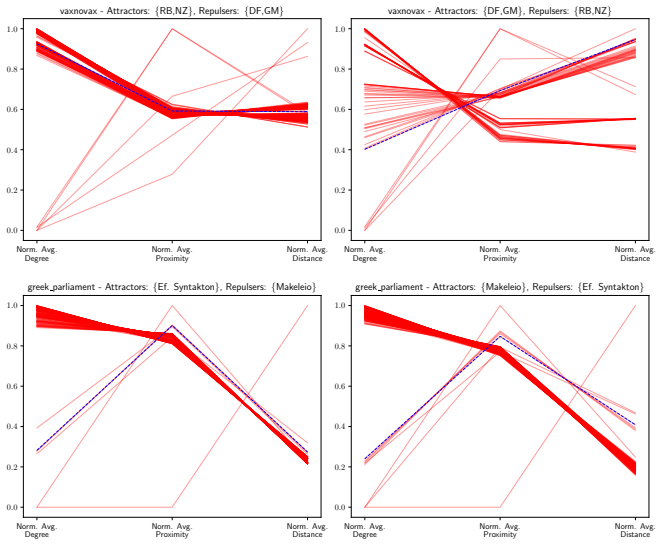

Figure 7: Variety of solutions obtained by performing a grid search over different values of λ_1 and λ_2. Each red line represents a single solution, for which we report its values of Norm. Avg. Degree, Norm. Avg. Proximity and Norm. Avg. Distance. Blue dashed lines represent the solutions discussed in Section 5.4.

out the noisy information, we manually remove specific users that are well-known to be satirical account, and set a threshold for edges' weights according to the maximum weight for which the removal of the edges does not produce a disconnected graph. For what concerns this graph, we select two nodes for A and R. The rationale behind the choice of the nodes is that we would like to observe the polarization around this topic combining 2 different points of view: that of the political representatives, and that of influential authorities in this debate. The first query set is composed by Nicola Zingaretti (NZ, Twitter username: nzingaretti), former leader of the main left-wing Italian party that showed support to this law, and Roberto Burioni (RB, Twitter username: RobertoBurioni), MD that takes part in many tv talk-show and via social media to promote the vaccination. The other is composed by Giorgia Meloni (GM, Twitter username: GiorgiaMeloni), leader of the main right-wing Italian party known for their position against restrictions, and Diego Fusaro (DF, Twitter username: DiegoFusaro), one of the main influential member of the NoVax movement in Italy.

The two solutions are reported together in Figure 8, and show specific characteristics. As for the size, the red niche (closer to GM and DF, far from NZ and RB) is more compact (28 users) than the blue one (closer to NZ and RB, far from GM and DF, 87 users). The results reported in Figure 7 confirm a larger polarization of the red niche, that is either closer from the *Attractor* and further from the *Repulser*, with respect to the blue one. Taking a look at the users inside the niche, we can find similarities with the previous analysis on the Greek Parliament, with different typologies of users: the red one is made principally limited to individuals, whose posts are effectively supporting both the NoVax movement (still in these days w.r.t. COVID-19 vaccine), and right-wing ideologies. There are

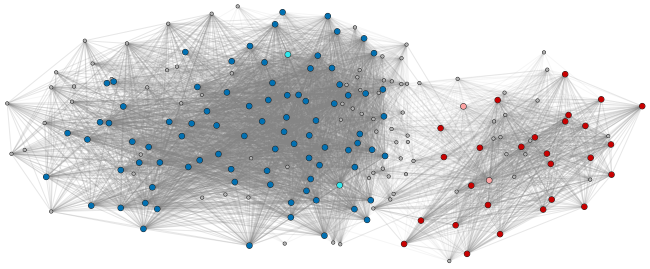

Figure 8: Polarized niches in VAXNOVAX. Dark blue nodes represent the niche closer to pro-vax sources (cyan nodes: Roberto Burioni, Nicola Zingaretti) and far from NoVax movement main supporters (pink nodes: Giorgia Meloni, Diego Fusaro), red nodes represent the opposite polarized niche. The former is composed by a variety of authorities (politicians, journalists, tv-shows and newspapers), while the latter contains normal individuals supporting the NoVax movement on Twitter.

neither specific newspapers, nor national authorities: as expected in such polarized niche, the misinformation spread through individual agents that has an audience, and their followers adequate their own beliefs based on their positions. For what concerns the blue subgraph, its composition is effectively less homogeneous: among the users we can find either politicians (mostly from the left-wing, including Beatrice Lorenzin, the proposer of the law), journalists, newspapers, tv-shows and some individual. Given the composition, it is also more difficult to be exposed to a specific propaganda, information confirmed by the Avg. Proximity and Avg. Distance of the subgraph reported in Figure 7.

6 CONCLUSIONS

In this work we propose an algorithmic primitive for discovering polarization niches. Our primitive leverages the prior knowledge of two sets of nodes that hold diametrically opposed opinions on a given controversial topic, for finding a dense cluster that is close to one set, and far from the other. Our formulation resembles the margin triplet loss used commonly in machine learning, yet it is a combinatorial function of shortest path distances.

Building on top of recent advances in combinatorial optimization, our proposal is DITH, an iterative algorithm that converges fast to a near-optimal solution. Our experiments show that our method can find near-optimal solutions within few greedy passes across a variety of very-large datasets. As the classic Densest Subgraph Problem, is a special case of our problem, we also show that our method can compute efficiently a certified high-quality approximation for DSP, over some of the largest graphs tested so far in the literature.

Our extensive evaluation on real-world networks extracted from Twitter on various controversial topics, shows that our proposed method can find polarization niches successfully, suggesting that these societal issues boosted by social media, such as the formation of echo-chambers with the consequential increase of polarization and partisanship around controversial issues, might be effectively detected and mitigated by algorithmic means.

REFERENCES

[1] Rediet Abebe, T.-H. Hubert Chan Chan, Jon Kleinberg, Zhibin Liang, David Parkes, Mauro Sozio, and Charalampos E. Tsourakakis. 2021. Opinion Dynamics Optimization by Varying Susceptibility to Persuasion via Non-Convex Local Search. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 16, 2 (2021), 1–34.

[2] Leman Akoglu. 2014. Quantifying Political Polarity Based on Bipartite Opinion Networks. *Proceedings of the International AAAI Conference on Web and Social Media* 8, 1 (2014), 2–11.

[3] Reid Andersen and Kevin J. Lang. 2006. Communities from Seed Sets. In *Proceedings of the 15th International Conference on World Wide Web* (Edinburgh, Scotland) *(WWW '06)*, 223–232.

[4] Sanjeev Arora, Elad Hazan, and Satyen Kale. 2012. The multiplicative weights update method: a meta-algorithm and applications. *Theory of computing* 8, 1 (2012), 121–164.

[5] Yuichi Asahiro, Kazuo Iwama, Hisao Tamaki, and Takeshi Tokuyama. 2000. Greedily Finding a Dense Subgraph. *Journal of Algorithms* 34, 2 (2000), 203–221.

[6] Cigdem Aslay, Antonis Matakos, Esther Galbrun, and Aristides Gionis. 2018. Maximizing the Diversity of Exposure in a Social Network. In *2018 IEEE International Conference on Data Mining (ICDM)*, 863–868.

[7] Eytan Bakshy, Solomon Messing, and Lada A. Adamic. 2015. Exposure to ideologically diverse news and opinion on Facebook. *Science* 348, 6239 (2015), 1130–1132.

[8] Nicola Barbieri, Francesco Bonchi, Edoardo Galimberti, and Francesco Gullo. 2015. Efficient and Effective Community Search. *Data Min. Knowl. Discov.* 29, 5 (2015), 1406–1433.

[9] Francesco Bonchi, Edoardo Galimberti, Aristides Gionis, Bruno Ordozgoiti, and Giancarlo Ruffo. 2019. Discovering Polarized Communities in Signed Networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management* (Beijing, China) *(CIKM '19)*, 961–970.

[10] Francesco Bonchi, Raffaele Perego, Fabrizio Silvestri, Hossein Vahabi, and Rossano Venturini. 2012. Efficient Query Recommendations in the Long Tail via Center-Piece Subgraphs. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, *(SIGIR '12)*, 345–354.

[11] Digvijay Boob, Yu Gao, Richard Peng, Saurabh Sawlani, Charalampos Tsourakakis, Di Wang, and Junxing Wang. 2020. Flowless: Extracting Densest Subgraphs Without Flow Computations. In *Proceedings of The Web Conference 2020* (Taipei, Taiwan) *(WWW '20)*, 573–583.

[12] Moses Charikar. 2000. Greedy Approximation Algorithms for Finding Dense Components in a Graph. In *Approximation Algorithms for Combinatorial Optimization*, Klaus Jansen and Samir Khuller (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 84–95.

[13] Chandra Chekuri, Kent Quanrud, and Manuel R. Torres. 2022. *Densest Subgraph: Supermodularity, Iterative Peeling, and Flow.* Society for Industrial and Applied Mathematics, , 1531–1555.

[14] Matteo Cinelli, Gianmarco De Francisci Morales, Alessandro Galeazzi, Walter Quattrociocchi, and Michele Starnini. 2021. The echo chamber effect on social media. *Proceedings of the National Academy of Sciences* 118, 9 (2021), e2023301118.

[15] Raviv Cohen and Derek Ruths. 2021. Classifying Political Orientation on Twitter: It's Not Easy! *Proceedings of the International AAAI Conference on Web and Social Media* 7, 1 (2021), 91–99.

[16] Michael D. Conover, Bruno Goncalves, Jacob Ratkiewicz, Alessandro Flammini, and Filippo Menczer. 2011. Predicting the Political Alignment of Twitter Users. In *2011 IEEE Third Int'l Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third Int'l Conference on Social Computing*, 192–199.

[17] Alessandro Cossard, Gianmarco De Francisci Morales, Kyriaki Kalimeri, Yelena Mejova, Daniela Paolotti, and Michele Starnini. 2020. Falling into the Echo Chamber: The Italian Vaccination Debate on Twitter. *Proceedings of the International AAAI Conference on Web and Social Media* 14, 1 (2020), 130–140.

[18] Wanyun Cui, Yanghua Xiao, Haixun Wang, and Wei Wang. 2014. Local Search of Communities in Large Graphs. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data (SIGMOD '14)*, 991–1002.

[19] Shahar Dobzinski, Noam Nisan, and Michael Schapira. 2010. Approximation algorithms for combinatorial auctions with complement-free bidders. *Mathematics of Operations Research* 35, 1 (2010), 1–13.

[20] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. 1978. An analysis of approximations for maximizing submodular set functions—II. In *Polyhedral Combinatorics: Dedicated to the memory of D.R. Fulkerson.* Springer Berlin Heidelberg, Berlin, Heidelberg, 73–87.

[21] Michael L. Fredman and Robert Endre Tarjan. 1987. Fibonacci Heaps and Their Uses in Improved Network Optimization Algorithms. *J. ACM* 34, 3 1987), 596–615.

[22] Giorgio Gallo, Michael D. Grigoriadis, and Robert E. Tarjan. 1989. A fast parametric maximum flow algorithm and applications. *SIAM J. Comput.* 18, 1 (1989), 30–55.

[23] Kiran Garimella, Aristides Gionis, Nikos Parotsidis, and Nikolaj Tatti. 2017. Balancing information exposure in social networks. In *Advances in Neural Information Processing Systems*, Vol. 30.

[24] Kiran Garimella, Gianmarco De Francisci Morales, Aristides Gionis, and Michael Mathioudakis. 2018. Quantifying Controversy on Social Media. *Trans. Soc. Comput.* 1, 1, Article 3 (2018), 27 pages.

[25] Kiran Garimella, Gianmarco De Francisci Morales, Aristides Gionis, and Michael Mathioudakis. 2018. Reducing Controversy by Connecting Opposing Views. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, 5249–5253.

[26] Aristides Gionis and Charalampos E. Tsourakakis. 2015. Dense Subgraph Discovery: KDD 2015 Tutorial. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, *(KDD '15)*, 2313–2314.

[27] Andrew V. Goldberg. 1984. *Finding a maximum density subgraph.* University of California Berkeley,

[28] Eduardo Graells-Garrido, Mounia Lalmas, and Daniele Quercia. 2014. People of Opposing Views Can Share Common Interests. In *Proceedings of the 23rd International Conference on World Wide Web* (Seoul, Korea) *(WWW '14 Companion)*, 281–282.

[29] Andrew M. Guess and Benjamin A. Lyons. 2020. Misinformation, Disinformation, and Online Propaganda. In *Social Media and Democracy*, Nathaniel Persily and Joshua A.Editors Tucker (Eds.). Cambridge University Press, , 10–33.

[30] Carlos Guestrin, Andreas Krause, and Ajit Paul Singh. 2005. Near-Optimal Sensor Placements in Gaussian Processes. In *Proceedings of the 22nd International Conference on Machine Learning* (Bonn, Germany) *(ICML '05)*, 265–272.

[31] Taher H. Haveliwala. 2002. Topic-Sensitive PageRank. In *Proceedings of the 11th International Conference on World Wide Web (WWW '02)*, 517–526.

[32] Bryan Hooi, Hyun Ah Song, Alex Beutel, Neil Shah, Kijung Shin, and Christos Faloutsos. 2016. FRAUDAR: Bounding Graph Fraud in the Face of Camouflage. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*, 895–904.

[33] Xin Huang, Laks V. S. Lakshmanan, and Jianliang Xu. 2019. *Community Search over Big Graphs.* Springer International Publishing, .

[34] Glen Jeh and Jennifer Widom. 2003. Scaling Personalized Web Search. In *Proceedings of the 12th International Conference on World Wide Web (WWW '03)*, 271–279.

[35] David Kempe, Jon Kleinberg, and Éva Tardos. 2003. Maximizing the Spread of Influence through a Social Network. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Washington, D.C.) *(KDD '03)*, 137–146.

[36] Isabel M. Kloumann and Jon M. Kleinberg. 2014. Community Membership Identification from Small Seed Sets. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14)*, 1366–1375.

[37] Yehuda Koren, Stephen C. North, and Chris Volinsky. 2007. Measuring and Extracting Proximity Graphs in Networks. *ACM Trans. Knowl. Discov. Data* 1, 3 (2007), 12–es.

[38] Andreas Krause and Carlos Guestrin. 2005. *Optimal nonmyopic value of information in graphical models: efficient algorithms and theoretical limits.* Carnegie Mellon University. Center for Automated Learning and Discovery, .

[39] Jérôme Kunegis. 2013. KONECT: The Koblenz Network Collection. In *Proceedings of the 22nd International Conference on World Wide Web (WWW'13 Companion)*, 1343–1350.

[40] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.

[41] Q. Vera Liao and Wai-Tat Fu. 2014. Can You Hear Me Now? Mitigating the Echo Chamber Effect by Source Position Indicators. In *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing, (CSCW '14)*, 184–196.

[42] Q. Vera Liao and Wai-Tat Fu. 2014. Expert Voices in Echo Chambers: Effects of Source Expertise Indicators on Exposure to Diverse Opinions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, *(CHI '14)*, 2745–2754.

[43] Richard A. Mills. 2018. Pop-up political advocacy communities on reddit. com: SandersForPresident and The Donald. *AI & SOCIETY* 33, 1 (2018), 39–54.

[44] Marco Minici, Federico Cinus, Corrado Monti, Francesco Bonchi, and Giuseppe Manco. 2022. Cascade-based Echo Chamber Detection. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, October 17-21, 2022.* 1511–1520.

[45] Baharan Mirzasoleiman, Stefanie Jegelka, and Andreas Krause. 2018. Streaming Non-Monotone Submodular Maximization: Personalized Video Summarization on the Fly. *Proceedings of the AAAI Conference on Artificial Intelligence* 32, 1 (2018).

[46] Corrado Monti, Giuseppe Manco, Cigdem Aslay, and Francesco Bonchi. 2021. Learning Ideological Embeddings from Information Cascades *Proceedings of the 30th ACM International Conference on Information and Knowledge Management, (CIKM'21)*,1325–1334.

[47] Sean Munson, Stephanie Lee, and Paul Resnick. 2021. Encouraging Reading of Diverse Political Viewpoints with a Browser Widget. *Proceedings of the International AAAI Conference on Web and Social Media* 7, 1 (2021), 419–428.

[48] Cameron Musco, Christopher Musco, and Charalampos E. Tsourakakis. 2018. Minimizing Polarization and Disagreement in Social Networks. In *Proceedings of the 2018 World Wide Web Conference* (Lyon, France) *(WWW '18)*, 369–378.

[49] George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. 1978. An analysis of approximations for maximizing submodular set functions—I. *Mathematical programming* 14, 1 (1978), 265–294.

[50] James B Orlin. 2013. Max flows in O (nm) time, or better. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*. 765–774.

[51] Marco Pennacchiotti and Ana-Maria Popescu. 2011. Democrats, Republicans and Starbucks Afficionados: User Classification in Twitter. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, (KDD '11)*, 430–438.

[52] John Pougué-Biyong, Akshay Gupta, Aria Haghighi, and Ahmed El-Kishky. 2022. Learning Stance Embeddings from Signed Social Graphs. *CoRR* abs/2201.11675 (2022).

[53] A P Riascos and José L Mateos. 2021. Random walks on weighted networks: a survey of local and non-local dynamics. *Journal of Complex Networks* 9, 5 (2021).

[54] Natali Ruchansky, Francesco Bonchi, David García-Soriano, Francesco Gullo, and Nicolas Kourtellis. 2015. The Minimum Wiener Connector Problem. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, (SIGMOD '15)*, 1587–1602.

[55] Alexander Schrijver. 2000. A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *Journal of Combinatorial Theory, Series B* 80, 2 (2000), 346–355.

[56] Mauro Sozio and Aristides Gionis. 2010. The Community-Search Problem and How to Plan a Successful Cocktail Party. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, (KDD '10)*, 939–948.

[57] Daniel A. Spielman and Shang-Hua Teng. 2004. Nearly-Linear Time Algorithms for Graph Partitioning, Graph Sparsification, and Solving Linear Systems. In *Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing, (STOC '04)*, 81–90.

[58] Daniel A. Spielman and Shang-Hua Teng. 2013. A local clustering algorithm for massive graphs and its application to nearly linear time graph partitioning. *SIAM Journal on computing* 42, 1 (2013), 1–26.

[59] Giorgos Stamatelatos, Sotirios Gyftopoulos, George Drosatos, and Pavlos S. Efraimidis. 2020. Revealing the political affinity of online entities through their Twitter followers. *Information Processing & Management* 57, 2 (2020), 102172.

[60] Chris Stark, Bobby-Joe Breitkreutz, Teresa Reguly, Lorrie Boucher, Ashton Breitkreutz, and Mike Tyers. 2006. BioGRID: a general repository for interaction datasets. *Nucleic Acids Research* 34 (01 2006), D535–D539.

[61] Hanghang Tong and Christos Faloutsos. 2006. Center-Piece Subgraphs: Problem Definition and Fast Solutions. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, (KDD '06)*, 404–413.

[62] Charalampos E. Tsourakakis. 2014. A Novel Approach to Finding Near-Cliques: The Triangle-Densest Subgraph Problem. *CoRR* abs/1405.1477 (2014).

[63] Vinod Vydiswaran, ChengXiang Zhai, Dan Roth, and Peter Pirolli. 2015. Overcoming bias to learn about controversial topics. *Journal of the Association for Information Science and Technology* 66, 8 (2015), 1655–1672.

[64] Hao Yin, Austin R. Benson, Jure Leskovec, and David F. Gleich. 2017. Local Higher-Order Graph Clustering. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, (KDD '17)*, 555–564.