The Graph Burning Problem under Constrained Diffusion

Enrico Iurlano[1], Günther R. Raidl[1], and Marko Djukanović[2]

Abstract Relying on a simplified model of fire spread, the Graph Burning Problem is an NP-hard combinatorial optimization problem that yields a metric of social contagion or vulnerability of a network. It concerns a discrete-time process on a simple undirected graph, with, in each timestep, a diffusion phase of fire towards all neighbors of "burned" vertices and a second phase, in which a next not-yet-burned vertex is ignited by an external actor. The aim is to find the minimum number of timesteps together with suitable choices of vertices for the actor such that the whole set of vertices gets burned. The applicability of the problem becomes more relevant when one takes into account that diffusion might realistically be suppressed by given natural obstacles and limitations as well as implemented countermeasures. Therefore, this paper proposes the Constrained Diffusion Graph Burning Problem, where vertex-specific thresholds are considered that determine the maximum number of neighbors the vertex can ignite. We consider the aspect of expiration, i.e., burned vertices are permitted to diffuse fire only immediately after becoming burned, but not to a later timestep. These assumptions not only appear meaningful in the context of fire spread but also within dynamics of viral contagion and rapid information dissemination. A time-indexed (mixed) integer linear programming approach is proposed in two versions: The first version one-hot encodes the time, and the second one handles the time using a Miller-Tucker-Zemlin formulation. Finally, a greedy heuristic is proposed and compared to the previous formulations.

Enrico Iurlano and Günther R. Raidl
Algorithms and Complexity Group, TU Wien, Favoritenstraße 9-11/192-01, Vienna, 1040, Austria
e-mail: {eiurlano|raidl}@ac.tuwien.ac.at

Marko Djukanović
Faculty of Natural Sciences and Mathematics, University of Banja Luka, Mladena Stojanovića 2, Banja Luka, 78000, Bosnia and Herzegovina e-mail: marko.djukanovic@pmf.unibl.org

This document is a preprint. The contribution was presented at the Optimization and Wildfire Conference (O&W), Luso, Portugal, October 1-4, 2024.

1 Introduction

In 2014, Bonato et al. [3] proposed the discrete-time process of *graph burning* as a model of social contagion. This process relies on a highly simplified model of fire spread and originally was used to provide insights into the achievability of rapid information dissemination in social networks under certain temporal assumptions on the release of information. Due to similarities, e.g., to the *Target Set Selection Problem* [14] or the *Least Cost Influence Maximization Problem* (LCIMP) [12], the *Graph Burning Problem* (GBP) additionally provides an interesting alternative model for viral marketing or opinion-making. For a formal description of the graph burning process, we consider a simple and undirected input graph $G = (V, E)$ whose vertices $v \in V$ carry a time-dependent binary status indicating either *unburned* or *burned* [3]. Assuming (discrete) timesteps indexed by $t \in \mathbb{N} \cup \{0\}$, consider the following process.

- Initially, all vertices are unburned.
- In timestep $t = 0$ the status of a single vertex is changed to burned.
- In each timestep $t \geq 1$, the neighbors of all vertices that have the status burned in timestep $t - 1$ become burned as well. Note that some of them may already be burned.
- Within the same timestep t, an unburned vertex is picked and made burned (if such a one exists).

The process is iterated and stops when a timestep is reached after which all the vertices of the graph are burned. Each such process is called an admissible burning process. The *burning number* $b(G)$ of G is defined as the minimum number of timesteps over all admissible burning processes, i.e., $b(G) := \min\{T+1 : \text{there is an admissible burning process on } G \text{ terminating with index } T\}$. Returning to the perspective of a social contagion model, a burned vertex could represent—within the setting of opinion-making or viral marketing—the scenario in which an individual has been convinced to adopt a certain opinion either by a direct influence of some external actor or by propagation from a neighboring individual that is already affected by that opinion. A small burning number for a social network reveals that it is possible to influence all individuals in a short amount of time.

Several combinatorial aspects of the graph burning number have been examined, e.g., it is known that its calculation is NP-hard even on trees of maximum degree three [1] or on caterpillars of maximum degree three [13, 16]; several other graph classes are mentioned in the literature in this context, see [2] for a partial overview. The central open problem in the field is stated in [3] claiming that for any graph $G = (V, E)$ the bound $b(G) \leq \lceil |V|^{1/2} \rceil$ holds; partial progress towards this so-called *burning number conjecture* has been recently obtained; in [19] it is proved that this conjecture holds asymptotically, and in [18] it is shown that the conjecture holds for any tree without vertices of degree two. The survey of Bonato [2] describes several further interesting computational findings on $b(G)$, in particular, established $b(G)$-numbers respectively approximation algorithms [4]; for general graphs, a 3-approximation algorithm is presented together with a 2-approximation algorithm

specifically designed for trees. A randomized 2.314-approximation algorithm for computing the burning number of a general graph is presented in [15]. In [9], an approximation algorithm termed "burning farthest-first" is proposed with a tighter approximation ratio of $3 - 2/b(G)$. It relies on the idea to iteratively incorporate locally optimal vertices into an initially empty set of candidate vertices. The chosen vertex lies here among the vertices that maximize the distance to the set of burning vertices. A similar approach has been designed earlier in [6] but from a purely heuristic perspective. Several further heuristics for this problem are proposed in [20, 11] based on the eigencentrality as a measure for attractiveness of a next vertex to be ignited. More recently in [8], based on the relationship between the GBP and the so-called *Clustered Maximum Coverage Problem*, the authors propose a greedy heuristic that effectively solves instances with up to hundreds of thousands of vertices. An integer linear programming formulation and a constraint programming approach are proposed in [10]. Recently, in [5] considerable progress has been achieved towards solving large solution instances optimally by a row generation approach.

The main contribution of this paper is the introduction and solution of a generalization of GBP where vertices can ignite a certain maximum number of neighbors. The generalization enlarges the search space of the original problem and requires the representation of feasible solutions to carry additional information. We propose two Mixed Integer Linear Programming (MILP) formulations and evaluate their computational advantages and limitations by running experiments on instances from the literature commonly used for the classical GBP. Furthermore, we propose a greedy approach which can be executed in one mode by solving subproblems via these formulations or, alternatively, by using a degree- and eigencentrality-based scoring function.

Notation and preliminaries. If not stated otherwise, simple and undirected graphs $G = (V, E)$ are considered; the *order* of a graph G refers to $|V|$, the number of vertices. By $N(v)$, for $v \in V$, we denote the set of vertices that are adjacent to v, i.e., $N(v) := \{w \in V : \{v, w\} \in E\}$. Moreover, let us denote by $\hat{G}$ the digraph resulting from bidirecting each edge of G, i.e., $\hat{G} := (V, \hat{E})$ with $\hat{E} := \{(u, v), (v, u) \in V \times V : \{u, v\} \in E\}$. In case some graph $G = (V, E)$ is a directed graph (digraph), the *in-degree* and *out-degree* of v refer to the quantities $|\{u \in V : (u, v) \in E\}|$ and $|\{w \in V : (v, w) \in E\}|$ respectively. The number of edges on a shortest path between v and w is denoted by $\mathrm{dist}(v, w)$. An *arborescence* is a directed acyclic graph $A = (V, E)$ with at most one vertex of in-degree zero—called the *root*—having in-degree one for all remaining vertices. Denote by $\mathrm{leaves}(A)$ the set of all leaves, i.e., vertices with out-degree zero, by $\mathrm{depth}(v)$ the *depth*, i.e., the number of edges on the unique directed path from the root of A to a vertex v, and by $\mathrm{height}(A)$ the maximum depth over all vertices contained in A. Sequences (or unions) of pairwise vertex-disjoint arborescences are called *forests of arborescences*. Denote by $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ the *floor* and *ceil* function, respectively.

The remainder of the paper is organized as follows. In Section 2 we motivate our generalization of graph burning. Section 3 provides two MILP formulations to solve

the introduced problem. Section 4 proposes a heuristic approach whose performance is in Section 5 compared to the MILP approaches. Conclusions and open questions are given in Section 6.

2 Constraints on the diffusion process

The classical GBP despite being practically motivated, assumes for its diffusion process that all neighbors of burned vertices get burned in the subsequent timestep. For many scenarios this assumption may seem too unrealistic: In general, people can successfully influence only a certain maximum number of other people from their circle of contacts (e.g., due to different individual resources, skills, or ambition for persuasion). Therefore, two constraints on the diffusion process shall be imposed to make the model more realistic and more flexible. We propose to limit, by a vertex-dependent threshold, the number of neighbors that can be ignited by a burned vertex.

Moreover, we want to incorporate the assumption that people become active in influencing other people immediately after they have been influenced; we motivate this by referring to [21] which demonstrates that the spread of social contagion is reduced by the decay of novelty. In the language of fire diffusion, the latter model is explained as follows: Vertices represent regions and a comparable behavior of the fire spread arises when fire can be contained to spread to a maximum number of neighboring regions, e.g., by having a specific local resource of firefighter units. Moreover, in this context, we assume that an already burned region cannot propagate fire at a later time point.

For a formalization, let us consider the graph $G = (V, E)$ with imposed thresholds $\theta_v \in \mathbb{N} \cup \{0\}$, $v \in V$; these thresholds are collected in a vector of parameters $\theta = (\theta_v)_{v \in V}$. Apart from the aforementioned time-dependent status burned or unburned for each vertex, we implicitly track a second time-dependent, irreversible binary status of a vertex, called the *expiration* status, which prevents vertices that have this status from transmitting their burned status to their neighbors. Assume that we are given a maximum time index $T \in \mathbb{N} \cup \{0\}$. Initially, for $t = 0$, burn a single vertex. Afterwards, execute the following actions for $t = 1, \dots, T$:

(i) *Constrained diffusion from unexpired vertices:* For each vertex v that has the status burned in timestep $t - 1$ but not in timestep $t - 2$, burn at most θ_v of its—according to the timestep $t - 1$—unburned neighbors. (By convention, in the (-1)-th timestep all vertices are unburned.)
(ii) *Burn:* Pick at most one currently unburned vertex and make it burned.

We call an execution of the latter procedure a *T-terminating θ-burning process (with constrained diffusion)* and denote the set of all such processes as $\mathcal{B}_{\theta,T}(G)$. For $P \in \mathcal{B}_{\theta,T}(G)$, the number of vertices with the status burned after termination of both phases in the final timestep T is called the *penetration* of P, adopting the terminology of the LCIMP [12]. If T^* is the minimum number that ensures a process

$P \in \mathcal{B}_{\theta,T^*}(G)$ of penetration $|V|$ exists, we define $b_\theta(G) := T^* + 1$ and call the latter *(constrained diffusion) θ-burning number of G*. The problem of determining this optimal value T^* is henceforth called the *Constraint Diffusion Graph Burning Problem* (CDGBP, or θ-GBP). If we want to provide more information on the burning process, we denote it as $\mathcal{B}_{\theta,T}(G, (s_0, \ldots, s_T), y)$ where s_t is the chosen vertex, henceforth called *seed*[1] during the *burn*-phase in timestep t. Moreover, for each directed edge $(u, v) \in \hat{E}$ we define

$$y_{u,v} := \begin{cases} 1 & \text{if } u \text{ ignited } v \text{ in some timestep } t \in \{1, \ldots, T\}, \\ 0 & \text{otherwise.} \end{cases}$$

The subsequent Proposition 1, in particular, gives a preparatory viewpoint for the MILP formulations stated later by directly lifting the considerations concerning $b(\cdot)$ in the proof of [3, Theorem 5] to θ-burning.

Proposition 1 *Let $T \in \mathbb{N} \cup \{0\}$ and $X \subseteq V$ for a given graph $G = (V, E)$ with thresholds θ. Then, the following assertions are equivalent.*

(i) There is a θ-burning process $\mathcal{B}_{\theta,T}(G, (s_0, \ldots, s_T), y)$ whose set of burned vertices after termination is X.

(ii) In G there is a subforest consisting of arborescences $T_{s_0}, T_{s_1}, \ldots, T_{s_T}$ with respective roots $s_0, \ldots, s_T$ and heights $h_0, \ldots, h_T$ such that, firstly, the set of all vertices contained in some arborescence corresponds to X; secondly, $h_t \leq T - t$ for $t = 0, \ldots, T$; and thirdly, each $v \in X$ is a member of a single arborescence $T_{s(v)}$ where the out-degree of v inside $T_{s(v)}$ is at most θ_v.

Proof. (i) $\Longrightarrow$ (ii): Let $P \in \mathcal{B}_{\theta,T}(G, (s_0, \ldots, s_T), y)$ and X be the set of vertices burned after termination of P. During the process, a vertex v might have become burned because one vertex (or even multiple vertices) "chose" to make their neighbor v burned. In the following, consider the digraph $\hat{G} := (V, \hat{E})$, i.e., the bidirected version of G. Note that if $s \in \{s_0, \ldots, s_T\}$ is a seed, we have $y_{u,s} = 0$ for each $u \in N(s)$, whereas $y_{s,u}$ can in fact attain the value 1 for $u \in N(s)$. If $\hat{E}|_y := \{(u, v) \in \hat{E} : y_{u,v} = 1\}$, then $\hat{G}|_y := (V, \hat{E}|_y)$ contains no directed cycles (not even of length two): In fact, the existence of a vertex having an incoming and an outgoing edge contradicts the property that only unburned vertices can be turned to burned by a burned neighbor. Facing a directed acyclic graph, we can therefore locate a set of sources (i.e., vertices of in-degree zero) that here must coincide with $\{s_0, \ldots, s_T\}$.

We run now the following procedure which returns an updated edge-status $y'_{u,v} \in \{0, 1\}$ satisfying for all $(u, v) \in \hat{E}$ that $y'_{u,v} \leq y_{u,v}$: Initially set $y'_{u,v} := y_{u,v}$. Then, iteratively, for $d = 1, \ldots, T$, execute the following: For each vertex $v \in V$ being reachable from a seed s_t via a directed length-d path do the following: Pick the seed s_{t^*} with the minimum index t^* permitting such a directed path $(s_{t^*}, u_1, u_2, \ldots, u_{d-1}, v)$. Then, overwrite $y'_{u,v} := 0$ for all $u \in N(v) \setminus \{u_{d-1}\}$. After termination of this procedure, we obtain that $\hat{G}|_{y'} := (V, \hat{E}|_{y'})$ is a forest of

[1] Potentially a seed reflects the empty vertex selection ε and this will manifest itself in our models in that no activation from an external source vertex is present.

arborescences $(A_{s_0}, \ldots, A_{s_T})$ whose roots are given by the seeds of the process. As all finally-burned vertices in X got burned in timestep T the latest, the distance from s_t to any vertex in A_{s_t} can be at most $T - t$. Moreover, y meets the θ-thresholds for each vertex $v \in V$, i.e., $\sum_{w \in N(v)} y_{v,w} \leq \theta_v$, and hence y', as a thinned-out version of y, does so as well.

(ii) $\implies$ (i): An admissible θ-burning process can be reconstructed by simply choosing the sequence of seeds $(s_0, \ldots, s_T)$ and in timestep t, for each burned vertex v, choosing as next vertices to be burned those being the children of v when v is seen as a member of its associated arborescence. Here the threshold-respecting out-degrees of the vertices in the arborescence will automatically imply compatibility with the thresholds for an admissible θ-burning process. The number of timesteps needed for this burning process is bounded from above by T, the height-restriction on the arborescence rooted in s_0. By construction, all $v \in X$ are therefore burned at most after termination of timestep T. □

Although equipped with further details, Fig. 1 in the later Sect. 3 is useful for visualizing the last characterization.

Observation 1. *For $G = (V, E)$ and $\theta_v := \deg(v)$, $v \in V$, we have $b_\theta(G) = b(G)$.*

Given $G = (V, E)$, θ, and T, finding the maximum penetration over all $P \in \mathcal{B}_{\theta,T}(G)$ may clearly be seen as a special case of the problem asking, for a pre-specified *arbitrary* parameter vector of heights $H := (h_1, \ldots, h_Q)$, $Q \leq |V|$, to find a sequence of Q arborescences maintaining these heights, respecting the thresholds θ and yielding maximum penetration. For the 1-tuple $H = (h)$ we call it the problem of finding a *Maximum Coverage Arborescence with θ-bounded Degree and h-bounded Height*, abbreviated MCADH(G, θ, h).

3 Mixed integer linear programming formulations

In the following, given the maximum time-index T, we address the problem of finding a sequence of seeds $(s_0, \ldots, s_T)$ for a θ-burning process leading to maximum penetration in a given graph $G = (V, E)$ with threshold parameter θ. The idea is to make use of Proposition 1 and to find a collection of pairwise vertex-disjoint arborescences $A^0, \ldots, A^T$ of the graph meeting all height and threshold constraints and maximizing the number of vertices covered by the collection.

An Integer Linear Program (ILP) purely relying on binary variables can be derived in the fashion of a time-indexed approach as done, e.g., in [7, 9] for related problems. We present it using an equivalent arborescence-based perspective: First, an external source vertex p with out-going directed edges pointing to all vertices in V is added to the graph. Introduce for all $t \in \{0, \ldots, T\}$ and $v \in V$ a depth-tracker $z_v^t \in \{0, 1\}$ indicating whether the arborescence covering vertex v, say its index is $s \in \{0, \ldots, T\}$, fulfills $s + \text{depth}(v) = t$. Similarly, for each $(v, w) \in \hat{E} \cup (\{p\} \times V)$ the variable $c_{v,w}^t \in \{0, 1\}$ indicates that $s + \text{depth}(v) = t - 1$ and, simultaneously, $s + \text{depth}(w) = t$

(using the convention $\text{depth}(p) = -1$). Here, the shifted depth-index $t = s + \text{depth}(v)$ agrees precisely with the timestep in which vertex v acquires the burned status (via diffusion or via a new ignition). Shifted depth-counters ensure that arborescences with a higher index are forced to attain lower heights. Based on these variables we derive the following formulation which we call "One-Hot encoded Depth" (OHD):

$$\max \sum_{t\in\{0,\dots,T\}} \sum_{v\in V} z_v^t \tag{1}$$

$$\text{s.t. } z_v^0 = c_{p,v}^0 \qquad \forall v \in V \tag{2}$$

$$\sum_{v\in V} c_{p,v}^t \leq 1 \qquad \forall t \in \{0,\dots,T\} \tag{3}$$

$$\sum_{t\in\{0,\dots,T\}} z_v^t \leq 1 \qquad \forall v \in V \tag{4}$$

$$\sum_{v\in N(w)} c_{v,w}^t \leq 1 \qquad \forall t \in \{0,\dots,T\}, \forall w \in V \tag{5}$$

$$\sum_{t\in\{0,\dots,T\}} c_{v,w}^t \leq 1 \qquad \forall (v,w) \in \hat{E} \tag{6}$$

$$\sum_{t\in\{0,\dots,T\}} \sum_{v\in V\cup\{p\}} c_{v,w}^t \leq 1 \qquad \forall w \in V \tag{7}$$

$$z_w^t \leq \sum_{v\in N(w)\cup\{p\}} c_{v,w}^t \qquad \forall t \in \{1,\dots,T\}, \forall w \in V \tag{8}$$

$$\sum_{w\in N(v)} c_{v,w}^t \leq \theta_v z_v^{t-1} \qquad \forall t \in \{1,\dots,T\}, \forall v \in V \tag{9}$$

$$c_{v,w}^t \leq z_v^{t-1} \qquad \forall t \in \{1,\dots,T\}, \forall (v,w) \in \hat{E} \tag{10}$$

$$c_{v,w}^t \in \{0,1\} \qquad \forall t \in \{0,\dots,T\}, \forall (v,w) \in \hat{E} \cup (\{p\}\times V) \tag{11}$$

$$z_v^t \in \{0,1\} \qquad \forall t \in \{0,\dots,T\}, \forall v \in V \tag{12}$$

Formulation (1)–(12) works as follows. The objective function (1) counts all vertices that are assigned a depth t and which thus must be part of one of the arborescences. Constraint (2), eliminable by preprocessing, associates depth 0 to a vertex v iff the 0-indexed activation from the source p to a vertex v is present. Constraint (3) ensures that for any activation index t, at most one vertex is activated by the source p. Constraint (4) forbids the assignment of more than one depth to each vertex. Constraint (5) guarantees that no matter which time index t is considered, the respective incoming activations to a vertex $w \in V$ from neighbors $v \in N(w)$ are at most one—uniqueness of predecessors in the arborescences is achieved in this way. The fact that an edge is active in at most one time step is represented by (6). Inequalities (7) enforce that no activations from potentially differing time indices point to a given vertex, as this is not forbidden solely by (5). The fact that for an index $t \geq 1$ and any vertex w no incoming t-indexed activation from any neighbor v (or the source p) is present implies that depth t is not assignable to w. To make sure that a vertex

v does not activate more than θ_v neighbors, constraint (9) is added. Constraint (10) is a strengthening inequality affirming that activation at timestep t via an outgoing edge of v requires that v has been assigned a shifted depth of $t-1$.

Let us add a formulation by adapting the well-known Miller-Tucker-Zemlin (MTZ) approach [17]. The goal is here to employ only a single binary activation variable per directed edge instead of $T+1$ associated binary variables. We introduce $T+1$ external source vertices $p_0, \dots, p_T$ all having out-going edges pointing to each vertex of the graph. For each $v \in V$ and $t \in \{0, \dots, T\}$ we let $\ell_v^t \in \{0,1\}$ indicate if v lies on A^t ($\ell_v^t = 1$ in the affirmative case, otherwise $\ell_v^t = 0$). Moreover, let the fractional variable $z_v \in [0, T]$ track the shifted depth of v with respect to its covering arborescence, and let $y_{v,w} \in \{0,1\}$ indicate if vertex w is a successor of vertex v in their covering arborescence. We declare the following formulation as MTZ:

$$\max \sum_{t\in\{0,\dots,T\}} \sum_{v\in V} \ell_v^t \tag{13}$$

$$\text{s.t.} \sum_{t\in\{0,\dots,T\}} \ell_v^t \le 1 \qquad \forall v \in V \tag{14}$$

$$\sum_{v\in V} y_{p_t,v} \le 1 \qquad \forall t \in \{0,\dots,T\} \tag{15}$$

$$y_{p_t,v} \le \ell_v^t \qquad \forall t \in \{0,\dots,T\}, \forall v \in V \tag{16}$$

$$y_{v,w} \le \sum_{t\in\{0,\dots,T\}} \ell_v^t \qquad \forall (v,w) \in \hat{E} \tag{17}$$

$$y_{v,w} \le \sum_{t\in\{0,\dots,T\}} \ell_w^t \qquad \forall (v,w) \in \hat{E} \tag{18}$$

$$\ell_w^t \le y_{p_t,w} + \sum_{v\in N(w)} y_{v,w} \qquad \forall t \in \{0,\dots,T\}, \forall w \in V \tag{19}$$

$$\ell_v^t - 1 + y_{v,w} \le \ell_w^t \qquad \forall t \in \{0,\dots,T\}, \forall (v,w) \in \hat{E} \tag{20}$$

$$z_v \ge t\ell_v^t \qquad \forall t \in \{0,\dots,T\}, \forall v \in V \tag{21}$$

$$z_v + 1 \le z_w + (1 - y_{v,w})(T+1) \qquad \forall (v,w) \in \hat{E} \tag{22}$$

$$0 \le z_v \le T \qquad \forall v \in V \tag{23}$$

$$\sum_{w\in N(v)} y_{v,w} \le \theta_v \qquad \forall v \in V \tag{24}$$

$$y_{v,w} \in \{0,1\} \qquad \forall (v,w) \in \hat{E} \cup (\{p_0,\dots,p_T\} \times V) \tag{25}$$

$$\ell_v^t \in \{0,1\} \qquad \forall t \in \{0,\dots,T\}, \forall v \in V \tag{26}$$

$$z_v \in \mathbb{Q} \qquad \forall v \in V \tag{27}$$

This formulation relies on a fractional counter variable that follows the rule of incrementation, see (22), up to a maximum height (23). If vertices are covered by the t-indexed arborescence, then their depths will be enumerated starting from t—this is imposed by (21). Constraint (24) implements the local bounds θ_v. The significant difference to the formulation OHD appears in (15)–(20), where the arborescence-

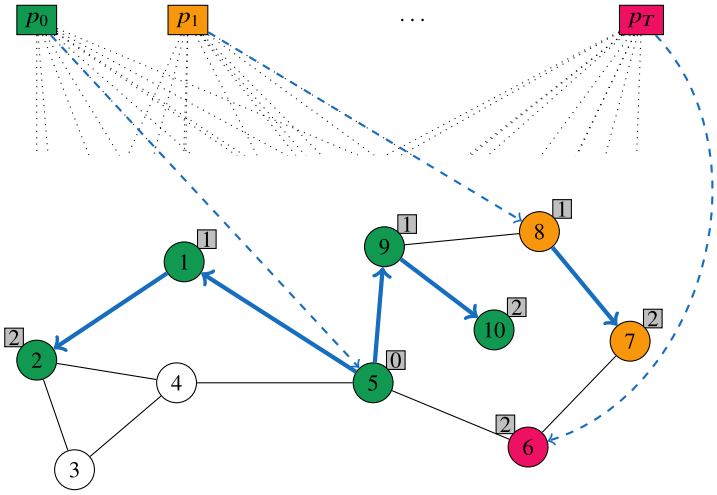

Fig. 1 An illustration of the MTZ formulation for $V = \{1, \ldots, 10\}$, $\theta_v := \lfloor \deg(v)/2 \rfloor$, and $T = 2$. Depth-trackers are displayed as attached rectangular labels. Association to the same arborescence is indicated by the same vertex color, activating edges by arrows.

indicating label resulting from ℓ must be inherited by activated neighbors; in fact, y is a variable shared by all arborescences, therefore propagating the right arborescence-labels is crucial. An illustration for MTZ is given in Fig. 1. The MTZ formulation can be strengthened by distance-based constraints: Vertices separated by a distance greater than the arborescence's height cannot both belong to the arborescence; a precalculation of the distances between all pairs of vertices is therefore required and can be achieved by Johnson's algorithm. Such an approach has been previously pursued in the setting of [10]. More precisely, we can impose

$$y_{p_t,v} + \ell_w^t \leq 1 \quad \forall t,\ \forall \{v, w\} \in \{\{v', w'\} \subseteq V : \operatorname{dist}(v', w') > T - t\}. \tag{28}$$

Remark 1 The proposed formulations rely on a fixed maximum time-index T. If we are interested in calculating b_θ, or more precisely in finding a shortest sequence of arborescences yielding full penetration, in principle, we can start with $T = 0$ and determine by a MILP solver if full penetration is already achievable. If this is not the case we iteratively repeat the procedure with incremented value of T, until a desired sequence of arborescences is found. Technically, we only need to solve a feasibility MILP where the original objective function is replaced with the constraint enforcing that all $|V|$ vertices must be burned. To reduce the number of MILP solver calls, it is conceivable to perform a binary search on the minimum value for T as in [10, Algorithm 1].

4 A heuristic approach

As we will see in the computational results, even the most adequate MILP approach reached its time limit on θ-GBP instances with thousands of vertices due to the high number of activation and depth-encoding variables. Thus, we propose an alternative heuristic approach relying on a simplified assumption. It will be able to produce solutions of solutions of satisfactory quality for large-scale instances requiring shorter runtimes. A typical greedy approach in the setting of the θ-GBP would be to iteratively find arborescences of decreasing heights and let them cover the input graph. After the placement of an arborescence, all used vertices are eliminated from the working copy of the input graph before the next arborescence is derived with the bound on its height decremented by one. The procedure terminates when the height bound reaches zero. Note that it may happen that all vertices are already covered earlier, in which case we assume here that all remaining arborescences are empty. The core task in this heuristic framework concerns the solution of the intermediate one-arborescence problem MCADH(G, θ, h), as introduced in Section 2.

The first approach relies on a straightforward adaptation of the ILP (1)–(12) addressing the search of a single arborescence; here, we can eliminate many activation variables per edge. We refer to this approach as *Greedy One Arborescence at a Time via Integer Linear Programming* (GOAT-ILP).

Recalling that many heuristic approaches in the literature, e.g., [20, 11] rely on the eigencentrality, we sketch a respective approach in our setting. The idea is to concurrently achieve the placement of a single arborescence via the iterative extension of an initial arborescence A (initially consisting of solely a root vertex) by finding the most promising current leaf w of A and strategically appending to it as many further vertices as the threshold of w allows. This growing process is guided by a scoring function quantifying how useful it is to append to a leaf w of A a neighbor $z \in N(w)$ (according to the adjacency of the original graph G) which is not yet present in the arborescence. We use the scoring function

$$\mathsf{LN}(A, w; z) := d(w) \cdot \frac{\theta'(z)}{\max_{y \in V} \theta'(y)} + (T - 1 - d(w)) \cdot \mathrm{ec}_{G-(A \cup \{w,z\})}(z),$$

where $d(w)$ denotes the depth of w inside A. The value $\theta'(z)$ refers to the remaining effective threshold of vertex z taking into account that some neighbors of z may already be incorporated into the arborescence; thus this threshold for the count of neighbors that might be ignited via diffusion can already be smaller. Lastly, $\mathrm{ec}_{G-(A \cup \{w,z\})}(z)$ denotes the eigencentrality of z after excluding the edge $\{w, z\}$ and the vertices of the so-far built arborescence A from G.

The (renormalized) value of $\theta'(z)$ and the eigencentrality of z—both attaining values in the interval $[0, 1]$—are linearly combined by weights assuring that leaves whose current height is far from the maximum height T are increasingly rewarded by a high eigencentrality value, and gradually, as the depths of the leaves increase, their (renormalized) θ'-value receives more importance. In the situation just before the

depth reaches the bound on the arborescence's height, by design, the eigencentrality does not even contribute to LN.

Eventually we want to find a leaf w of A for which

$$\mathsf{LeafScore}(A, w) := \max_{\substack{P \subseteq N(w) \setminus A \\ |P| \leq \theta'(w)}} \sum_{z \in P} \mathsf{LN}(A, w; z) \tag{29}$$

is maximal among the leaves of A. When this maximizer w is found, the arborescence A is extended towards all vertices in the set P that was itself responsible for maximizing (29).

We start the whole construction by initializing A with a vertex having highest eigencentrality, apply aforementioned procedure as long as an extension is possible, and return this maximal arborescence as result for the MCADH instance. We refer later to this score-based approach carried out iteratively for $t = T, T-1, \ldots, 1, 0$, simply as *Greedy One Arborescence at a Time* (GOAT).

5 Computational results

The MILP and GOAT approaches proposed in Section 3 have been implemented in `Julia 1.11.1`. The MILP solver `Gurobi`[2] in version 10.0.3 is used to solve the MILPs. A cluster with an Intel(R) Xeon(R) E5-2640 v4 CPU with 2.40GHz and 160GB RAM running Ubuntu 18.04.6 LTS was used to run all experiments on a single thread.

We rely on the benchmark instances from [10, Table 3]. Here, we show results only for every fourth instance from this benchmark set due to space limitations. The results for all instances can be found online[3]. As thresholds we use $\theta_v := \lfloor v/2 \rfloor$, $v \in V$, comparable to the so-called *majority thresholds* for the Target Set Selection Problem [14]. The following approaches are considered in the comparisons: OHD; MTZ; MTZ-s, i.e., MTZ strengthened by the distance-based constraints (28); GOAT-ILP with a time limit of 1000 seconds per placement; and GOAT which will be let run until termination without a set timelimit. A time limit of 1000 seconds is given to `Gurobi` to solve OHD, MTZ, MTZ-s. No time limit is set for the original GOAT approach. For each approach after termination an independently written feasibility-checker is run.

Table 1 is structured as follows: each column corresponds to a selected instance, labeled with its respective name. The first three rows provide key characteristics of each instance, including the number of vertices, the number of edges, and the standard deviation of vertex degrees. The subsequent rows present the results for OHD, MTZ, MTZ-s, GOAT-ILP, and GOAT, evaluated for $T \in \{2, 3\}$. For the OHD and the GOAT approach, the runtimes in seconds are additionally documented;

[2] `https://www.gurobi.com`

[3] `https://www.ac.tuwien.ac.at/research/problem-instances/#Graph_Burning_Problem`

Table 1 Comparison of approaches for $T \in \{2, 3\}$. Rows OHD report optimal values except for $T = 3$ with encountered gaps of 426% and 12% for `c-fat500-1` and `TVshow`, respectively.

	karate-club	polbooks	ia-enron-only	sphere3	c-fat500-1	web-polblogs	DD199	lattice3D	lattice2D	ia-fb-messages	TVshow
$\|V\|$	34	105	143	258	500	643	841	1000	1089	1266	3892
$\|E\|$	78	441	623	768	4459	2280	1902	2700	2112	6451	17262
sd(deg)	3.88	5.47	6.08	0.30	1.31	11.46	1.47	0.69	0.34	13.24	12.6
$T = 2$											
OHD	34	76	122	18	47	369	30	18	11	778	753
MTZ	34	68	90	18	41	293	19	12	11	163	84
MTZ-s	34	69	118	18	46	190	9	12	10	5	0
GOAT-ILP	31	76	122	18	47	369	30	18	11	761	753
GOAT	25	62	109	18	47	283	26	18	11	697	204
OHD time[s]	0.06	4.41	1.20	0.66	127.2	20.41	1.17	2.37	1.66	41.17	231.0
GOAT time[s]	0.01	0.01	0.02	0.01	0.03	0.24	0.02	0.01	0.02	21.73	7.52
$T = 3$											
OHD	34	105	143	52	95	612	68	58	26	1237	1861
MTZ	34	105	143	46	82	513	36	33	26	629	100
MTZ-s	34	105	143	48	86	587	23	43	16	1108	0
GOAT-ILP	34	105	139	52	96	583	67	58	26	1235	1807
GOAT	21	99	138	47	89	402	55	56	26	1062	626
OHD time[s]	0.01	0.15	0.29	8.19	1000	21.68	20.13	0.24	11.98	67.13	1000
GOAT time[s]	0.01	0.03	0.06	0.02	0.08	0.67	0.08	0.07	0.07	42.88	30.73

see rows OHD-time[s] and GOAT-time[s], respectively. Gaps in percent shown in Table 1 indicate the difference between the dual bound and the primal bound, further renormalized by the primal bound. The gaps for MTZ and MTZ-s, are omitted here, as they often arise just from the trivial dual bound, i.e., the number of vertices. The following observations can be made from the reported results in Table 1.

For $T = 2$ the OHD approach is able to solve all instances to optimality where the used time is shorter than the time limit of 1000 seconds—often under a fraction of a second. We notice that none of the other exact approaches, i.e., MTZ and MTZ-s, is able to return results of comparable quality; they just can solve the two sparsest instances `sphere3` and `lattice2D` (almost) optimally. When examining the slightly denser instances, already for `polbooks` with only 105 vertices both MTZ approaches do not find a primal bound of at least 90% of the optimal one computed by OHD despite a 200 times longer runtime. This trend is even more evident on larger, dense instances where the incumbent solution of the MTZ approach can be smaller by (almost) an order of magnitude (see `ia-fb-messages` and `TVshow`). On the instances with up to 500 vertices, the additional distance-based constraints in MTZ-s seem indeed to help finding better solutions than those obtained via the

MTZ formulation. On the other hand, for larger graphs the numerous strengthening inequalities negatively impact, apparently, on the solution quality yielding solutions smaller by even two orders of magnitude for the two largest graphs. The results for the GOAT-ILP indicate that the successive arborescence placement seems to be a promising heuristic. Despite the considerably simplifying nature of the approach, primal bounds on par with (or just slightly inferior to) the optimal primal bounds from OHD can be obtained. On smaller graphs the GOAT heuristic is slightly inferior to the exact approaches. However, on graphs with more than 800 vertices it outperforms the MTZ approaches. While the quality determined by the OHD approach cannot be obtained by GOAT, the computation time of the latter is in general shorter.

When $T = 3$ we notice that runtimes for OHD are mostly longer than for $T = 2$, which is explainable by the increased number of variables in the respective formulation. On the three smallest instances the runtime is, however, shorter than for $T = 2$, which is most likely due to the fact that a solution burning the entire graph is now quickly encounterable, making the computation of tight dual bounds unnecessary. Apparently, for the instance `c-fat500-1` with $T = 3$, the calculation of tight dual bounds is particularly difficult in the modality of OHD; we notice that GOAT-ILP is in this case the preferable approach returning a solution better by one unit. On the remaining instances GOAT-ILP has again a quality on par or slightly inferior to OHD.

For $T \in \{2, 3\}$ the GOAT approach does not appear to fully exploit the quality that can be obtained according to GOAT-ILP in the greedy setting, but it is superior to the MTZ approaches on the largest instance; the MTZ approaches seem to be unsuitable for processing such large graph sizes in a meaningful way. In contrast to the exact ILP-driven routine in GOAT-ILP, the score function driven placement in GOAT risks leading the arborescence towards dead ends or locally unpromising vertices. However, in particular if the need is to quickly generate a solution, this seems a viable approach given the faster—and apparently better estimable—computation times.

It is an important insight that, when dealing with a large number of binary variables in OHD, the solver performs better than in the setting of MTZ where many of these variables are replaced by fewer fractional variables.

6 Conclusion

We introduced a generalized version of the well-studied graph burning problem that more realistically enables control of the diffusion process via local threshold parameters. During diffusion it limits the number of neighboring vertices to be possibly affected by a burned vertex in the subsequent iteration. This problem is linked to the task of finding a shortest sequence of vertex-disjoint arborescences with upper-bounded heights and local upper bounds on their out-degrees.

Following this observation, we proposed two MILP models, one of which was clearly ahead on the instances tested. Empirically shown, the addition of a greedy

assumption has proven its worth with only minimal loss of quality counterbalanced by a clear simplification of the problem. The availability of a suitable local search would possibly add the finishing touches to near-optimal solutions, in particular for GOAT-ILP.

Our greedy approaches still neglect the aspect of "communication" between the different arborescences. In future work, one may address their simultaneous construction. We plan to let a score function also consider the connectivity, in particular the number and topology of the arising connected components, to enhance the approach.

Furthermore, several open problems arise in the context of the introduced θ-burning process: From the perspective of fire spread, the θ-burning number expresses a worst-case that we are interested in precalculating. However, it might be also relevant to consider the setting where a number of at most θ_v unburned neighbors picked uniformly at random are chosen to receive the status burned—at the same time, we might want to assume a random choice of the seeds. This yields a different measure of vulnerability and coming up with estimates for the expected penetration over all processes obeying these assumptions seems an interesting challenge.

The following bilevel optimization problem may be a relevant extension: Given a budget $B \in \mathbb{N}$, find a vector of "threshold-reducers" $(r_v)_{v \in V}$ with $0 \leq r_v \leq \theta_v$ and $\sum_{v \in V} r_v \leq B$ such that for $\theta' := (\theta_v - r_v)_{v \in V}$ the respective value of $b_{\theta'}$ is minimum, i.e., find a budget-maintaining way to contain the worst-case of fire diffusion as much as possible.

Acknowledgements We thank an anonymous referee and Johannes Varga for valuable inputs.

This work is partially supported by the VGSCO (FWF [10.55776/W1260-N35]), the Ministry of Civil Affairs of Bosnia and Herzegovina [1259074], and OeAD [BA05/2023]; M. Djukanović further by the Ministry of Scientific and technological development and higher education, Serb Republic ["Utilizing deep learning to enhance the efficiency of optimization algorithms"].

References

1. Bessy, S., Bonato, A., Janssen, J., Rautenbach, D., Roshanbin, E.: Burning a graph is hard. Discrete Applied Mathematics **232**, 73–87 (2017)
2. Bonato, A.: A survey of graph burning. Contributions to Discrete Mathematics **16**(1), 185–197 (2021)
3. Bonato, A., Janssen, J., Roshanbin, E.: Burning a graph as a model of social contagion. In: Algorithms and Models for the Web Graph: 11th International Workshop. LNCS, vol. 8882, pp. 13–22. Springer (2014)
4. Bonato, A., Kamali, S.: Approximation algorithms for graph burning. In: Theory and Applications of Models of Computation: 15th Annual Conference. LNCS, vol. 11436, pp. 74–92. Springer (2019)
5. de Carvalho Pereira, F., de Rezende, P.J., Yunes, T., Morato, L.F.B.: A row generation algorithm for finding optimal burning sequences of large graphs. In: 32nd Annual European Symposium on Algorithms. LIPIcs, vol. 308, pp. 94:1–94:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik (2024)

6. Farokh, Z.R., Tahmasbi, M., Tehrani, Z.H.R.A., Buali, Y.: New heuristics for burning graphs. arXiv preprint arXiv:2003.09314 (2020)
7. Finbow, S., MacGillivray, G.: The firefighter problem: a survey of results, directions and questions. Australasian Journal of Combinatorics **43**, 57–78 (2009)
8. García-Díaz, J., Cornejo-Acosta, J.A., Trejo-Sánchez, J.A.: A greedy heuristic for graph burning. Computing **107**(3), 91 (2025)
9. García-Díaz, J., Pérez-Sansalvador, J.C., Rodríguez-Henríquez, L.M.X., Cornejo-Acosta, J.A.: Burning graphs through farthest-first traversal. IEEE Access **10**, 30395–30404 (2022)
10. García-Díaz, J., Rodríguez-Henríquez, L.M.X., Pérez-Sansalvador, J.C., Pomares-Hernández, S.E.: Graph burning: Mathematical formulations and optimal solutions. Mathematics **10**(15), 2777 (2022)
11. Gautam, R.K., Kare, A.S.: Faster heuristics for graph burning. Applied Intelligence **52**(2), 1351–1361 (2022)
12. Günneç, D., Raghavan, S., Zhang, R.: Least-cost influence maximization on social networks. INFORMS Journal on Computing **32**(2), 289–302 (2020)
13. Hiller, M., Koster, A.M., Triesch, E.: On the burning number of p-caterpillars. In: Graphs and Combinatorial Optimization: from Theory to Applications, AIRO, vol. 5, pp. 145–156. Springer (2021)
14. Kempe, D., Kleinberg, J., Tardos, É.: Maximizing the spread of influence through a social network. In: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 137–146. Association for Computing Machinery (2003)
15. Lieskovský, M., Sgall, J., Feldmann, A.E.: Approximation algorithms and lower bounds for graph burning. In: Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques. LIPIcs, vol. 275, pp. 9:1–9:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik (2023)
16. Liu, H., Hu, X., Hu, X.: Burning number of caterpillars. Discrete Applied Mathematics **284**, 332–340 (2020)
17. Miller, C.E., Tucker, A.W., Zemlin, R.A.: Integer programming formulation of traveling salesman problems. Journal of the ACM **7**(4), 326–329 (1960)
18. Murakami, Y.: The burning number conjecture is true for trees without degree-2 vertices. Graphs and Combinatorics **40**(4), 82 (2024)
19. Norin, S., Turcotte, J.: The burning number conjecture holds asymptotically. Journal of Combinatorial Theory, Series B **168**, 208–235 (2024)
20. Šimon, M., Huraj, L., Dirgová Luptáková, I., Pospíchal, J.: Heuristics for spreading alarm throughout a network. Applied Sciences **9**(16), 3269 (2019)
21. Wu, F., Huberman, B.A.: Novelty and collective attention. Proceedings of the National Academy of Sciences **104**(45), 17599–17601 (2007)