Solving graph compression via optimal transport

Vikas K. Garg
CSAIL, MIT
vgarg@csail.mit.edu

Tommi Jaakkola
CSAIL, MIT
tommi@csail.mit.edu

Abstract

We propose a new approach to graph compression by appeal to optimal transport. The transport problem is seeded with prior information about node importance, attributes, and edges in the graph. The transport formulation can be setup for either directed or undirected graphs, and its dual characterization is cast in terms of distributions over the nodes. The compression pertains to the support of node distributions and makes the problem challenging to solve directly. To this end, we introduce Boolean relaxations and specify conditions under which these relaxations are exact. The relaxations admit algorithms with provably fast convergence. Moreover, we provide an exact $O(d \log d)$ algorithm for the subproblem of projecting a d-dimensional vector to transformed simplex constraints. Our method outperforms state-of-the-art compression methods on graph classification.

1 Introduction

Graphs are widely used to capture complex relational objects, from social interactions to molecular structures. Large, richly connected graphs can be, however, computationally unwieldy if used as-is, and spurious features present in the graphs that are unrelated to the task can be statistically distracting. A significant effort thus has been spent on developing methods for compressing or summarizing graphs towards graph *sketches* [1]. Beyond computational gains, these sketches take center stage in numerous tasks pertaining to graphs such as partitioning [2, 3], unraveling complex or multi-resolution structures [4, 5, 6, 7], obtaining coarse-grained diffusion maps [8], including neural convolutions [9, 10, 11].

State-of-the-art compression methods broadly fall into two categories: (a) sparsification (removing edges) and (b) coarsening (merging vertices). These methods measure spectral similarity between the original graph and a compressed representation in terms of a (inverse) Laplacian quadratic form [12, 13, 14, 15, 16]. Thus, albeit some of these methods approximately preserve the graph spectrum (see, e.g., [1]), they are oblivious to, and thus less effective for, downstream tasks such as classification that rely on labels or attributes of the nodes. Also, the key compression steps in most of these methods are, typically, either heuristic or detached from their original objective [17].

We address these issues by taking a novel perspective that appeals to the theory of optimal transport [18], and develops its connections to minimum cost flow on graphs [19, 20]. Specifically, we interpret graph compression as minimizing the transport cost from a fixed initial distribution supported on all vertices to an unknown target distribution whose size of support is limited by the amount of compression desired. Thus, the compressed graph in our case is a subgraph of the original graph, restricted to a subset of the vertices selected via the associated transport problem. The transport cost depends on the specified prior information such as importance of the nodes and their labels or attributes, and thus can be informed by the downstream task. Moreover, we take into account the underlying geometry toward the transport cost, unlike agnostic measures such as KL-divergence [21].

There are several technical challenges that we must address. First, the standard notion of optimal transport on graphs is tailored to directed graphs where the transport cost decomposes as a directed

flow along the edge orientations [22]. To circumvent this limitation, we extend optimal transport on graphs to handle both directed and undirected edges, and derive a dual that directly measures the discrepancy between distributions on the vertices. As a result, we can also compress *mixed* graphs that contain both directed and undirected edges [23, 24, 25, 26].

The second challenge comes from the compression itself, enforced in our approach as sparse support of the target distribution. Optimal transport (OT) is known to be computationally intensive, and almost all recent applications of OT in machine learning, e.g., [27, 28, 29, 30, 31, 32] rely on entropy regularization [33, 34, 35, 36, 37] for tractability. However, entropy is not conducive to sparse solutions since it discourages the variables from ever becoming zero [22]. In principle, one could consider convex alternatives such as enforcing ℓ_1 penalty [38]. However, such methods require iterative tuning to find a solution that matches the desired support, and require strong assumptions such as restricted eigenvalue, isometry, or nullspace conditions for recovery. Some of these issues were previously averted by introducing binary selection variables [39, 40] and using Boolean relaxations in the context of unconstrained real spaces (regression). However, they do not apply to our setting since the target distribution must reside in the simplex. We introduce *constrained* Boolean relaxations that are not only efficient, but also provide exactness certificates.

Our graph compression formulation also introduces new algorithmic challenges. For example, solving our sparsity controlled dual transport problem involves a new subproblem of projecting on the probability simplex Δ under a diagonal transformation. Specifically, let $D(\epsilon)$ be a diagonal matrix with the diagonal $\epsilon \in [0, 1]^d \setminus \{\mathbf{0}\}$. Then, for a given ϵ, the problem is to find the projection $x \in \mathbb{R}^d$ of a given vector $y \in \mathbb{R}^d$ such that $D(\epsilon)x \in \Delta$. This generalizes well-studied problem of Euclidean projection on the probability simplex [41, 42, 43], recovered if each ϵ_i is set to 1. We provide an exact $O(d \log d)$ algorithm for solving this generalized projection. Our approach leads to convex-concave saddle point problems with fast convergence via methods such as Mirror Prox [44].

To summarize, we make the following contributions. We propose an approach for graph compression based on optimal transport (OT). Specifically, we (a) extend OT to undirected and mixed graphs (section 2), (b) introduce constrained Boolean relaxations for our dual OT problem, and provide exactness guarantees (section 3), (c) generalize Euclidean projection onto simplex, and provide an efficient algorithm (section 3), and (d) demonstrate that our algorithm outperforms state-of-the-art compression methods, both in accuracy and compression time, on classifying graphs from standard real datasets. We also provide qualitative results that our approach provides meaningful compression in synthetic and real graphs (section 4).

2 Optimal transport for general edges

Let $\vec{G} = (V, \vec{E})$ be a directed graph on nodes (or vertices) V and edges $\vec{E}$. We define the signed incidence matrix $\vec{F}$: $\vec{F}(\vec{e}, v) = 1$ if $\vec{e} = (w, v) \in \vec{E}$ for some $w \in V$, -1 if $\vec{e} = (v, w) \in \vec{E}$ for some $w \in V$, and 0 otherwise. Let $c(\vec{e}) \in \mathbb{R}_+$ be the positive cost to transport unit mass along edge $\vec{e} \in \vec{E}$, and $\Delta(V)$ the probability simplex on V. The shorthand $a \preceq b$ denotes that $a(i) \leq b(i)$ for each component i. Let $\mathbf{0}$ be a vector of all zeros and $\mathbf{1}$ a vector of all ones. Let $\rho_0, \rho_1 \in \Delta(V)$ be distributions over the vertices in V. The optimal transport distance $\vec{W}(\rho_0, \rho_1)$ from ρ_0 to ρ_1 is [22]:

$$\vec{W}(\rho_0, \rho_1) = \min_{\substack{J \in \mathbb{R}^{|\vec{E}|} \\ \mathbf{0} \preceq J}} \sum_{\vec{e} \in \vec{E}} c(\vec{e}) J(\vec{e}) \qquad \text{s.t.} \quad \vec{F}^\top J = \rho_1 - \rho_0 \, ,$$

where $J(\vec{e})$ is the non-negative mass transfer from tail to head on edge $\vec{e}$. Intuitively, $\vec{W}(\rho_0, \rho_1)$ is the minimum cost of a directed flow from ρ_0 to ρ_1. In order to extend this intuition to the undirected graphs, we need to refine the notion of incidence, and let the mass flow in either direction. Specifically, let $G = (V, E)$ be a connected undirected graph. We define the incidence matrix pertaining to G as $F(e, v) = 1$ if edge e is incident on v, and 0 otherwise. With each undirected edge $e \in E$, having cost $c(e) \in \mathbb{R}_+$, we associate two directed edges e^+ and e^-, each with cost $c(e)$, and flow variables $J^+(e), J^-(e) \geq 0$. Then, the total *undirected flow* pertaining to e is $J^+(e) + J^-(e)$. Since we incur cost for flow in either direction, we define the optimal transport cost $W(\rho_0, \rho_1)$ from ρ_0 to ρ_1 as

$$\min_{\substack{J^+, J^- \in \mathbb{R}^{|E|} \\ \mathbf{0} \preceq J^+, J^-}} \sum_{e \in E} c(e)(J^+(e) + J^-(e)) \qquad \text{s.t.} \quad F^\top(J^- - J^+) = \rho_1 - \rho_0 \, . \tag{1}$$

We call a directed edge e^+ *active* if $J^+(e) > 0$, i.e., there is some positive flow on the edge (likewise for e^-). Moreover, by extension, we call an undirected edge e active if at least one of e^+ and e^- is active. We claim that at most one of e^+ and e^- may be active for any edge e.

Theorem 1. *The optimal solution to* (1) *must have* $J^+(e) = 0$ *or* $J^-(e) = 0$ *(or both)* $\forall\, e \in E$.

The proof is provided in the supplementary material. Thus, like the directed graph setting, we either have flow in only one direction for each edge e, or no flow at all. Moreover, Theorem 1 facilitates generalizing optimal transport distance to mixed graphs $\tilde{G}(V, E, \vec{E})$, i.e., where both directed and undirected edges may be present. In particular, we adapt the formulation in (1) with minor changes: (a) we associate bidirectional variables with each edge, directed or undirected. For the undirected edges $e \in E$, we replicate the constraints from (1). For the directed edges $\vec{e}$, we follow the convention that $J^+(\vec{e})$ denotes the outward flow along $\vec{e}$ whereas $J^-(\vec{e})$ denotes the incoming flow (from head to tail), and impose the additional constraints $J^-(\vec{e}) = 0$. We will focus on undirected graphs $G = (V, E)$ since the extensions to directed and mixed graphs are immediate due to Theorem 1.

3 Graph compression

We view graph compression as the problem of minimizing the optimal transport distance from an initial distribution ρ_0 having full support on the vertices V to a target distribution ρ_1 that is supported only on a subset $S_V(\rho_1)$ of V. The compressed subgraph is obtained by restricting the original graph to vertices in $S_V(\rho_1)$ and the incident edges. The initial distribution ρ_0 encodes any prior information. For instance, it might be taken as a stationary distribution of random walk on the graph. Likewise, the cost function c encodes the preference for different edges. In particular, a high value of $c(e)$ would inhibit edge e from being active. This flexibility allows our framework to inform compression based on the specifics of different downstream applications by getting to define ρ_0 and c appropriately.

3.1 Dual characterization of the transport distance

Note that (1) defines an optimization problem over edges. However, our perspective requires quantifying $W(\rho_0, \rho_1)$ as an optimization over the vertices. Fortunately, strong duality comes to our rescue. Let $c = (c(e), e \in E)$ be the column vector obtained by stacking the costs. The dual of (1) is

$$\max_{\substack{\mathbf{0} \preceq y, \mathbf{0} \preceq z \\ -c\, \preceq\, F(y-z) \preceq c}} (y - z)^\top(\rho_1 - \rho_0), \quad \text{or equivalently,} \quad \max_{\substack{t \in \mathbb{R}^{|V|} \\ -c\, \preceq Ft \preceq c}} t^\top(\rho_1 - \rho_0)\ . \tag{2}$$

This alternative formulation of $W(\rho_0, \rho_1)$ in (2) lets us define compression solely in terms of variables over vertices. Specifically, for a budget of at most k vertices, we solve

$$\min_{\substack{\rho_1 \in \Delta(V) \\ ||\rho_1||_0 \leq k}} \max_{\substack{t \in \mathbb{R}^{|V|} \\ -c \preceq Ft \preceq c}} \underbrace{t^\top(\rho_1 - \rho_0) + \frac{\lambda}{2}||\rho_1||^2}_{\mathcal{L}_\lambda(\rho_1, t; \rho_0)}\ , \tag{3}$$

where $\lambda > 0$ is a regularization hyperparameter, and $||\rho_1||_0$ is the number of vertices with positive mass under ρ_1, i.e., the cardinality of support set $S_V(\rho_1)$. The quadratic penalty is strongly convex in ρ_1, so as we shall see shortly, would help us leverage fast algorithms for a saddle point problem. Note that a high value of λ would encourage ρ_1 toward a uniform distribution. We favor this penalty over entropy, which is not conducive to sparse solutions since entropy would forbid ρ_1 from having zero mass at any vertex in the graph.

Our next result reveals the structure of optimal solution ρ_1^* in (3). Specifically, ρ_1^* must be expressible as an affine function of ρ_0 and F. Moreover, the constraints on active edges are tight. This reaffirms our intuition that ρ_1^* is obtained from ρ_0 by transporting mass along a subset of the edges, i.e., the active edges. The remaining edges do not participate in the flow.

Theorem 2. *The optimal ρ_1^* in* (3) *is of the form* $\rho_1^* = \rho_0 + F^\top\eta$, *where* $\eta \in \mathbb{R}^{|E|}$. *Furthermore, for any active edge* $e \in E$, *we must have* $Ft^*(e) \in \{c(e), -c(e)\}$.

3.2 Constrained Boolean relaxations

The formulation (3) is non-convex due to the support constraint on ρ_1. Since recovery under ℓ_1 based methods such as Lasso often requires extensive tuning, we resort to the method of Boolean

relaxations that affords an explicit control much like the ℓ_0 penalty. However, prior literature on Boolean relaxations is limited to variables that have no additional constraints beyond sparsity. Thus, in order to deal with the simplex constraints $\rho_1 \in \Delta(V)$, we introduce constrained Boolean relaxations. Specifically, we define the characteristic function $g_V(x) = 0$ if $x \in \Delta(V)$ and ∞ otherwise, and move the non-sparsity constraints inside the objective. This lets us delegate the sparsity constraints to binary variables, which can be relaxed to $[0, 1]$. Using the definition of $\mathcal{L}_\lambda$, we can write (3) as

$$\min_{\substack{\rho_1 \in \mathbb{R}^{|V|} \\ ||\rho_1||_0 \leq k}} \quad \max_{\substack{t \in \mathbb{R}^{|V|} \\ -c \preceq Ft \preceq c}} \mathcal{L}_\lambda(\rho_1, t; \rho_0) + g_V(\rho_1) .$$

Denoting by $\odot$ the Hadamard (elementwise) product, and introducing variables $\epsilon \in \{0, 1\}^{|V|}$, we get

$$\min_{\substack{\epsilon \in \{0,1\}^{|V|} \\ ||\epsilon||_0 \leq k}} \min_{\rho_1 \in \mathbb{R}^{|V|}} \quad \max_{\substack{t \in \mathbb{R}^{|V|} \\ -c \preceq Ft \preceq c}} \mathcal{L}_\lambda(\rho_1 \odot \epsilon, t; \rho_0) + g_V(\rho_1 \odot \epsilon) .$$

Adjusting the characteristic term as a constraint, we have the following equivalent problem

$$\min_{\substack{\epsilon \in \{0,1\}^{|V|} \\ ||\epsilon||_0 \leq k}} \min_{\substack{\rho_1 \in \mathbb{R}^{|V|} \\ \rho_1 \odot \epsilon \in \Delta(V)}} \quad \max_{\substack{t \in \mathbb{R}^{|V|} \\ -c \preceq Ft \preceq c}} \mathcal{L}_\lambda(\rho_1 \odot \epsilon, t; \rho_0) . \tag{4}$$

Algorithm 1 Algorithm to compute Euclidean projection on the d-simplex Δ under a diagonal transformation.

Input: y, ϵ

Define $\mathcal{I}_> \triangleq \{j \in [d] \mid \epsilon_j > 0\}$

Define $\mathcal{I}_= \triangleq \{j \in [d] \mid \epsilon_j = 0\}$;

$y_> \triangleq \{y_j \mid j \in \mathcal{I}_>\}$; $\epsilon_> \triangleq \{\epsilon_j \mid j \in \mathcal{I}_>\}$

Sort $y_>$ into $\hat{y}_>$ and $\epsilon_>$ into $\hat{\epsilon}_>$, in non-increasing order, based on $y_j/\epsilon_j, j \in \mathcal{I}_>$. Rename indices in $(\hat{y}_>, \hat{\epsilon}_>)$ to start from 1. Let π map $j \in \mathcal{I}_>$ to $\pi(j) \in [|\hat{y}_>|]$. Thus

$$\hat{y}_1/\hat{\epsilon}_1 \geq \hat{y}_2/\hat{\epsilon}_2 \geq \ldots \geq \hat{y}_{|\mathcal{I}_>|}/\hat{\epsilon}_{|\mathcal{I}_>|}$$

$$b_j = \hat{y}_j + \hat{\epsilon}_j \frac{(1 - \sum_{i=1}^j \hat{\epsilon}_i \hat{y}_i)}{\sum_{i=1}^j \hat{\epsilon}_i^2}, \ \forall j \in [|y_>|]$$

$$\ell = \max\{j \in [|y_>|] \mid b_j > 0\}$$

$$\alpha = \frac{(1 - \sum_{i=1}^\ell \hat{\epsilon}_i \hat{y}_i)}{\sum_{i=1}^\ell \hat{\epsilon}_i^2}$$

$$x_j = \max\{\hat{y}_{\pi(j)} + \alpha \hat{\epsilon}_{\pi(j)}, 0\}, \ \forall j \in \mathcal{I}_>$$

$$x_j = y_j, \forall j \in \mathcal{I}_=$$

Algorithm 2 Mirror Prox algorithm to (approximately) find ϵ in relaxation of (9). The step-sizes at time ℓ with respect to ϵ, t, and ζ are α_ℓ, β_ℓ, and γ_ℓ respectively.

Input: ρ_0, k, λ; iterations T

Define $\tilde{\mathcal{E}}_k = \{\epsilon \in [0,1]^{|V|} \mid \epsilon^\top \mathbf{1} \leq k\}$

Define $\mathcal{T}_{F,c}$ as in (5)

Define $\psi_{\rho_0}(\epsilon, t, \zeta)$ as in (9)

Initialize $\epsilon^{(0)} = k\mathbf{1}/|V|$, $t^{(0)} = \mathbf{0}$, and $\zeta^{(0)} = 0$

for $\ell = 0, 1, \ldots, T$ **do**

    Gradient step:

$$\hat{\epsilon}^{(\ell)} = \text{Proj}_{\tilde{\mathcal{E}}_k}\left(\epsilon^{(\ell)} - \alpha_\ell \nabla_\epsilon \psi_{\rho_0}(\epsilon^{(\ell)}, t^{(\ell)}, \zeta^{(\ell)})\right)$$

$$\hat{t}^{(\ell)} = \text{Proj}_{\mathcal{T}_{F,c}}\left(t^{(\ell)} + \beta_\ell \nabla_t \psi_{\rho_0}(\epsilon^{(\ell)}, t^{(\ell)}, \zeta^{(\ell)})\right)$$

$$\hat{\zeta}^{(\ell)} = \zeta^{(\ell)} + \gamma_\ell \nabla_\zeta \psi_{\rho_0}(\epsilon^{(\ell)}, t^{(\ell)}, \zeta^{(\ell)})$$

    Extra-gradient step:

$$\epsilon^{(\ell+1)} = \text{Proj}_{\tilde{\mathcal{E}}_k}\left(\epsilon^{(\ell)} - \alpha_\ell \nabla_\epsilon \psi_{\rho_0}(\hat{\epsilon}^{(\ell)}, \hat{t}^{(\ell)}, \hat{\zeta}^{(\ell)})\right)$$

$$t^{(\ell+1)} = \text{Proj}_{\mathcal{T}_{F,c}}\left(t^{(\ell)} + \beta_\ell \nabla_t \psi_{\rho_0}(\hat{\epsilon}^{(\ell)}, \hat{t}^{(\ell)}, \hat{\zeta}^{(\ell)})\right)$$

$$\zeta^{(\ell+1)} = \zeta^{(\ell)} + \gamma_\ell \nabla_\zeta \psi_{\rho_0}(\hat{\epsilon}^{(\ell)}, \hat{t}^{(\ell)}, \hat{\zeta}^{(\ell)})$$

end for

$$\hat{\epsilon} = \sum_{\ell=1}^T \alpha_\ell \hat{\epsilon}^{(\ell)} / \sum_{\ell=1}^T \alpha_\ell$$

Our formulation in (4) requires solving a new subproblem, namely, Euclidean projection on the d-simplex Δ under a diagonal transformation. Specifically, let $D(\epsilon)$ be a diagonal matrix with the diagonal $\epsilon \in [0, 1]^d \setminus \{\mathbf{0}\}$. Then, for a given ϵ, the problem is to find the projection $x \in \mathbb{R}^d$ of a given vector $y \in \mathbb{R}^d$ such that $D(\epsilon)x = x \odot \epsilon \in \Delta$. This problem generalizes Euclidean projection on the probability simplex [41, 42, 43], which is recovered when we set ϵ to $\mathbf{1}$, i.e., an all-ones vector. Our next result shows that Algorithm 1 solves this problem exactly in $O(d \log d)$ time.

Theorem 3. *Let $\epsilon \in [0, 1]^d \setminus \{\mathbf{0}\}$ be a given vector of weights, and $y \in \mathbb{R}^d$ be a given vector of values. Algorithm 1 solves the following problem in $O(d \log d)$ time*

$$\min_{x \in \mathbb{R}^d \,:\, x \odot \epsilon \in \Delta} \quad \frac{1}{2}||x - y||^2 .$$

Theorem 3 allows us to relax the problem (4), and solve the relaxed problem efficiently since the projection steps on the other constraint sets can be solved by known methods [45, 46]. Specifically, since ϵ consists of only zeros and ones, $||\epsilon||_0 = ||\epsilon||_1 = \epsilon^\top \mathbf{1}$ and $\epsilon \odot \epsilon = \epsilon$. So, we can write (4) as

$$\min_{\epsilon \in \mathcal{E}_k} \min_{\substack{\rho_1 \in \mathbb{R}^{|V|} \\ \rho_1 \odot \epsilon \in \Delta(V)}} \max_{t \in \mathcal{T}_{F,c}} \mathcal{L}_\lambda(\rho_1 \odot \epsilon \odot \epsilon, t; \rho_0) ,$$

where we denote the constraints for ϵ and t respectively by

$$\mathcal{E}_k \triangleq \left\{\epsilon \in \{0,1\}^{|V|} \mid \epsilon^\top \mathbf{1} \le k\right\}, \quad \text{and} \quad \mathcal{T}_{F,c} \triangleq \left\{t \in \mathbb{R}^{|V|} \mid -c \preceq Ft \preceq c\right\}. \tag{5}$$

We can thus eliminate ϵ from the regularization term via a change of variable $\rho_1 \odot \epsilon \to \tilde{\rho}_1$

$$\min_{\epsilon \in \mathcal{E}_k} \; \min_{\substack{\tilde{\rho}_1 \in \mathbb{R}^{|V|} \\ \tilde{\rho}_1 \odot \epsilon \in \Delta(V)}} \; \max_{t \in \mathcal{T}_{F,c}} \; \mathcal{L}_0(\tilde{\rho}_1 \odot \epsilon, t; \rho_0) + \frac{\lambda}{2}||\tilde{\rho}_1||^2 \; . \tag{6}$$

We note that (6) is a mixed-integer program due to constraints $\mathcal{E}_k$, and thus hard to solve. Nonetheless, we can relax the hard binary constraints on the coordinates of ϵ to $[0,1]$ intervals to obtain a saddle point formulation with a strongly convex term, and solve the relaxation efficiently, e.g., via customized versions of methods such as Mirror Prox [44], Accelerated Gradient Descent [47], or Primal-Dual Hybrid Gradient [48]. An attractive property of our relaxation is that if the solution $\hat{\epsilon}$ from the relaxed problem is integral then $\hat{\epsilon}$ must be optimal for the non-relaxed hard problem (6), and so the original formulation (3). We now pin down the necessary and sufficient conditions for optimality of $\hat{\epsilon}$.

Theorem 4. *Let $S_V(\rho_1^*) = \{v \in V \mid \rho_1^*(v) > 0\}$ be the support of optimal ρ_1 in the original formulation* (3). *Let the indicator $I_{S_V^*} \in \{0,1\}^{|V|}$ be such that $I_{S_V^*}(v) = 1$ if $v \in S_V(\rho_1^*)$ and 0 otherwise. The relaxation of* (6) *is guaranteed to recover $S_V(\rho_1^*)$ if and only if there exists a tuple $(\gamma, \hat{t}, \hat{\nu}, \hat{\zeta}) \in \mathbb{R}_+ \times \mathbb{R}^{|V|} \times \mathbb{R}_+^{|V|} \times \mathbb{R}$ such that the following holds for all vertices $v \in V$,*

$$|\hat{t}(v) - \hat{\nu}(v) + \hat{\zeta}| \quad \begin{cases} > \gamma & \text{if } v \in S_V(\rho_1^*) \\ < \gamma & \text{if } v \notin S_V(\rho_1^*) \end{cases} \quad, \quad where \tag{7}$$

$$(\hat{t}, \hat{\nu}, \hat{\zeta}) \in \arg\max_{t \in \mathcal{T}_{F,c}} \max_{\nu \in \mathbb{R}_+^{|V|}} \max_{\zeta \in \mathbb{R}} -\left(\frac{1}{2\lambda}||(t - \nu + \zeta\mathbf{1}) \odot I_{S_V^*}||^2 + t^\top \rho_0 + \zeta\right). \tag{8}$$

The quantity $|\hat{t}(v) - \hat{\nu}(v) + \hat{\zeta}|$ in (7) may be viewed as the strength of a signal. In that sense, we require the vertices in support of optimal ρ_1 to have a strictly higher signal that the vertices not in the support. Such signal detection conditions appear in various contexts and often have information theoretic implications, e.g., Ising models [49]. (7) is also reminiscent of the β-min condition on regression coefficients for variable selection with Lasso in high-dimensional linear models [50].

For some applications projecting on the simplex, as required by (6), may be an expensive operation. We can invoke the minimax theorem to swap the order of $\tilde{\rho}_1$ and t, and proceed with a Lagrangian dual to eliminate $\tilde{\rho}_1$ at the expense of introducing a scalar variable. Thus, effectively, we can replace the projection on simplex by a one-dimensional search. We state this equivalent formulation below.

Theorem 5. *Problem* (6)*, and thus the original formulation* (3)*, is equivalent to*

$$\min_{\epsilon \in \mathcal{E}_k} \max_{\substack{t \in \mathcal{T}_{F,c} \\ \zeta \in \mathbb{R}}} \underbrace{-\frac{1}{2\lambda} \sum_{v:t(v) \le -\zeta} \left(\epsilon(v)(t(v) + \zeta)^2 + 2\lambda t(v)\rho_0(v)\right) - \sum_{v:t(v) > -\zeta} t(v)\rho_0(v) - \zeta}_{\psi_{\rho_0}(\epsilon, t, \zeta)} . \tag{9}$$

We present a customized Mirror Prox procedure in Algorithm 2. The projections $\mathrm{Proj}_{\mathcal{T}_{F,c}}$ and $\mathrm{Proj}_{\tilde{\mathcal{E}}_k}$ can be computed efficiently [45, 46]. We round the solution $\hat{\epsilon} \in [0,1]^{|V|}$ returned by the algorithm to have at most k vertices as the estimated support for the target distribution ρ_1 if $\hat{\epsilon}$ is not integral. The compressed graph is taken to be the subgraph spanned by these vertices.

3.3 Specifying the cost function

In our experiments, we fixed the cost of each edge, computed based on the agreement between the associated vertex labels. Here we illustrate briefly how to parameterize the cost and how the parameters could be learned. Define $\ell(i,j) = 1$ for edge (i,j) if vertices i and j have the same label, and -1 otherwise. Let the cost function be parameterized by $\theta = (\theta_s, \theta_d)$, $\theta_s > 0, \theta_d > 0$ such that

$$c_\theta(i,j) = 0.5(\theta_d(1 - \ell(i,j)) + \theta_s(1 + \ell(i,j))) .$$

Table 1: **Description of graph datasets, and comparison of accuracy on test data.** We provide the statistics on the number of graphs, number of classes, average number of nodes, and average number of edges in each dataset. The classification test accuracy (along with standard deviation) when each graph was (roughly) compressed to half is shown for each method for each training fraction in $\{0.2, \ldots, 0.8\}$. The algorithm having the best performance is indicated with bold font in each case. '-' entries indicate that the method failed to compress the dataset (e.g. due to matrix singularity).

Dataset	method	acc@0.2	acc@0.3	acc@0.4	acc@0.5	acc@0.6	acc@0.7	acc@0.8
MSRC-21C	REC	.485±.016	.543±.010	.595±.010	.625±.013	.641±.008	.696±.013	.738±.016
graphs: 209	Heavy	.408±.016	.479±.015	.516±.009	.538±.009	.557±.011	.602±.022	.653±.011
classes: 20	Affinity	.413±.021	.489±.008	.516±.011	.549±.010	.560±.016	.607±.019	.654±.021
nodes: 40.3	Alg. Dist.	.452±.036	.498±.035	.524±.021	.535±.027	.531±.029	.590±.032	.652±.044
edges: 96.6	OTC	**.548±.004**	**.605±.003**	**.639±.006**	**.679±.003**	**.696±.002**	**.742±.007**	**.778±.005**
DHFR	REC	.681±.011	.704±.014	.724±.007	.738±.009	.749±.008	.756±.011	.771±.011
graphs: 467	Heavy	.719±.010	.751±.010	.776±.012	.782±.008	.777±.009	.786±.014	.799±.013
classes: 2	Affinity	.717±.013	.733±.011	.745±.014	.761±.014	.771±.019	.767±.015	.785±.013
nodes: 42.4	Alg. Dist.	.743±.011	.761±.012	.768±.022	.786±.019	.810±.025	**.817±.033**	.809±.030
edges: 44.5	OTC	**.757±.004**	**.784±.003**	**.797±.005**	**.799±.003**	**.811±.007**	.814±.006	**.823±.004**
MSRC-9	REC	.738±.011	.782±.010	.817±.009	.818±.013	.835±.020	.833±.018	.840±.013
graphs: 221	Heavy	.648±.019	.710±.024	.766±.014	.773±.010	.786±.009	.796±.010	.813±.009
classes: 8	Affinity	.665±.015	.722±.005	.762±.010	.774±.014	.789±.026	.786±.019	.801±.017
nodes: 40.6	Alg. Dist.	.666±.048	.717±.051	.756±.029	.771±.039	.798±.032	.803±.030	.809±.046
edges: 97.9	OTC	**.784±.005**	**.808±.005**	**.826±.007**	**.846±.003**	**.839±.006**	**.842±.007**	**.854±.003**
BZR-MD	REC	.525±.011	.548±.015	**.563±.020**	.553±.021	.563±.012	.569±.012	.587±.020
graphs: 306	Heavy	.497±.000	.546±.000	.555±.000	.522±.000	.550±.000	.572±.000	.558±.000
classes: 2	Affinity	.508±.006	.534±.012	.534±.017	.532±.015	.549±.020	.567±.033	.562±.029
nodes: 21.3	Alg. Dist.	.497±.021	.546±.026	.555±.038	.522±.028	.550±.028	.572±.024	.558±.039
edges: 225.06	OTC	**.534±.000**	**.569±.000**	.547±.000	**.579±.000**	**.572±.000**	**.607±.000**	**.603±.000**
Mutagenicity	REC	.713±.006	.730±.006	.742±.005	.752±.004	.758±.005	.765±.007	.769±.007
graphs: 4337	Heavy	.718 ±.006	.738±.004	.753±.004	.763±.004	.771±.003	.779±.004	.783±.004
classes: 2	Affinity	-	-	-	-	-	-	-
nodes: 30.3	Algebraic	-	-	-	-	-	-	-
edges: 30.8	OTC	**.749±.002**	**.768±.003**	**.779±.003**	**.787±.004**	**.792±.004**	**.795±.003**	**.799±.003**

Thus, $c_\theta(i,j) \in \{\theta_s, \theta_d\}$ depending on whether i and j have the same label.

The cost parameters couldn't be driven solely by the compression criterion as this objective would lead to a trivial all-zero solution. Instead, θ must be in part driven by an external classification loss. In other words, we can learn θ by trading off compression loss against, for example, the ability to correctly classify the resulting reduced graphs. We leave this for future work.

3.4 Relation to other compression techniques

Most compression algorithms try to preserve the graph spectrum via a (multi-level) coarsening procedure: at each level they compute a matching of vertices and merge the matched vertices, e.g., Heavy Edge [2] contracts those edges (i,j) that are incident on low degree vertices. Likewise REC [1] follows a randomized greedy procedure for generating maximal matching incrementally. Let d_i be the degree of node i. Setting the cost $c(i,j) = \max(d_i, d_j)$ in our framework will incentivize flow on edges with low degree vertices, and in turn, compression of one of their end points. The vertices not in support of target distribution may then be viewed as being matched to (a subset of) adjacent vertices that they transfer flow to. Unlike other methods, our approach is flexible in terms of defining $c(i,j)$.

4 Experiments

We conducted several experiments to demonstrate the merits of our method. We start by describing the experimental setup. We fixed the value of hyperparameters in Algorithm 2 for all our experiments. Specifically, we set the regularization coefficient $\lambda = 1$, and the gradient rates $\alpha_\ell = 0.1, \beta_\ell =$

$0.1, \gamma_\ell = 0.1$ for each $\ell \in \{0, 1, \ldots, T\}$. We also let ρ_0 be the stationary distribution by setting $\rho_0(v)$ for each $v \in V$ as the ratio of $deg(v)$, i.e. the degree of v, to the sum of degrees of all the vertices. Note that the distribution thus obtained is the unique stationary distribution for connected non-bipartite graphs, and *a* stationary distribution for the bipartite graphs. Moreover, for non-bipartite graphs, it has a nice physical interpretation in that any random walk on the graph always converges to this distribution irrespective of the graph structure.

The objective of our experiments is three-fold. Since compression is often employed as a preprocessing step for further tasks, we first show that our method compares favorably, in terms of test accuracy, to the state-of-the-art compression methods on graph classification. We then demonstrate that our method performed the best in terms of the compression time. We finally show that our approach provides qualitatively meaningful compression on synthetic and real examples.

4.1 Classifying standard graph data

We used several standard graph datasets for our experiments, namely, DHFR [51], BZR-MD [52], MSRC-9, MSRC-21C [53], and Mutagenicity [54]. We focused on these datasets since they represent a wide spectrum in terms of the number of graphs, number of classes, average number of nodes per graph, and average number of edges per graph (see Table 1 for details). All these datasets have a class label for each graph, and additionally, labels for each node in every graph.

We compare the test accuracy of our algorithm, OTC (short for Optimal Transport based Compression), to several state-of-the-art methods: REC [1], Heavy edge matching (Heavy) [2], Affinity vertex proximity (Affinity) [14], and Algebraic distance (Algebraic) [12, 13]. Amongst these, the REC algorithm is a randomized method that iteratively contracts edges of the graph and thereby coarsens the graph. Since the method is randomized, REC yields a compressed graph that achieves a specified compression factor, i.e., the ratio of the number of nodes in the reduced graph to that in the original uncompressed graph, only in expectation. Therefore, in order to allow for a fair comparison, we first run REC on each graph with the compression factor set to 0.5, and then execute other baselines and our algorithm, i.e. Algorithm 2, with k set to the number of nodes in the compressed graph produced by REC. Also to mitigate the effects of randomness on the number of nodes returned by REC, we performed 5 independent runs for REC, and subsequently all other methods.

Each collection of compressed graphs was then divided into train and test sets, and used for our classification task. Since our datasets do not provide separate train and test sets, we employed the following procedure for each dataset. We partitioned each dataset into multiple train and test sets of varying sizes. Specifically, for each $p \in \{0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8\}$, we divided each dataset randomly into a train set containing a fraction p of the graphs in the dataset, and a test set containing the remaining $1 - p$ fraction. To mitigate the effects of chance, we formed 5 such independent train-test partitions for each fraction p for each dataset. We averaged the results over these multiple splits to get one reading per collection, and thus 5 readings in total for all collections, for each fraction for each method. We averaged the test accuracy across collections for each method and fraction.

We now specify the cost function c for our algorithm. As described in section 3, we can leverage the cost function to encode preference for different edges in the compressed graph. For each graph, we set $c = 0.01$ for each edge e incident on the nodes with same label, and $c = 0.02$ for each e incident on the nodes that have different label. Thus, in effect, we slightly biased the graphs to prefer retaining the edges that have separate labels at their end-points. In our experiments, we employed support vector machines (SVMs), with Weisfeiler-Leman subtree kernel [55] to quantify the similarity between graphs [53, 56, 57]. This kernel is based on the Weisfeiler-Leman test of isomorphism [58, 59], and thus naturally takes into account the labels of the nodes in the two graphs. We fixed the number of kernel iterations to 5. We also fixed $T = 25$ for our algorithm. For each method and each train-test split, we used a separate 5-fold cross-validation procedure to tune the coefficient of error term C over the set $\{0.1, 1, 10\}$ for training an independent SVM model on the training portion.

Table 1 summarizes the performance of different methods for each fraction of the data. As the numbers in bold indicate, our method generally outperformed the other methods across the datasets. We observe that on some datasets the discrepancy between the average test accuracy of two algorithms is massive for every fraction. Note that though OTC performs the best on DHFR, the performance of most methods is similar (except REC, which lags behind). In contrast, REC performs better than all methods except OTC on MSRC-9. This seems to suggest that REC performs well on graphs with
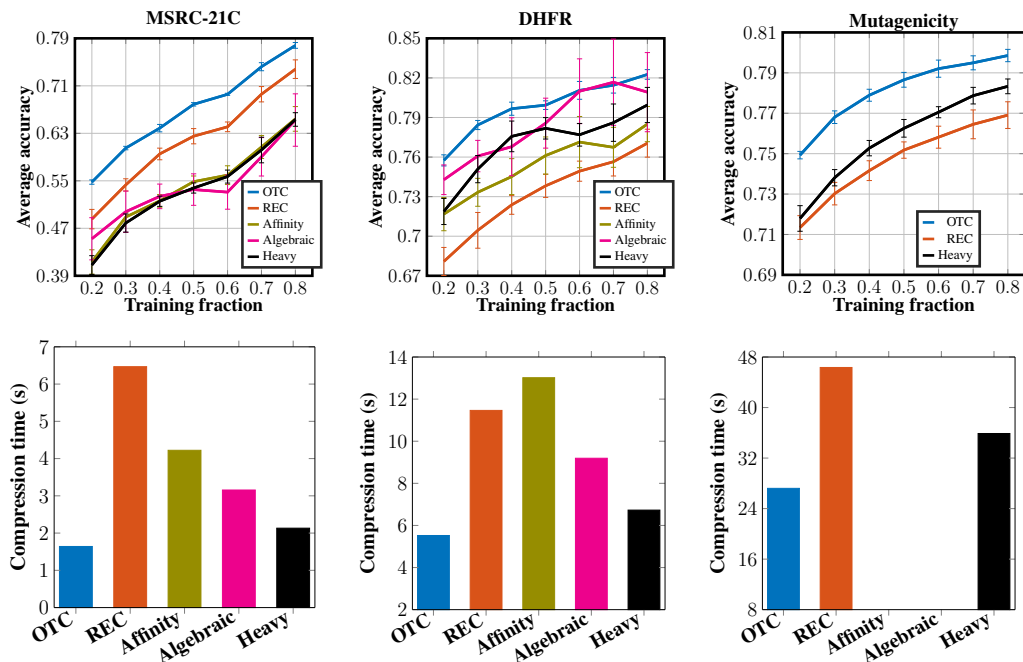
Figure 1: **Comparison on standard graph datasets**. The top row shows the average test accuracy and corresponding standard deviation for our method (OTC) and state-of-the-art baselines for different fractions of training data. The bottom row compares the corresponding compression times. Our method outperforms the other methods in terms of both accuracy and compression time.

strong connectivity, while others might be better on data with a long backbone besides these ring structures. We believe robust performance of OTC across these datasets comprising graphs with vastly different topologies underscores the promise of our approach.

Further, as Fig. 1 shows, OTC performed best in terms of compression time as well. We emphasize that the discrepancy in compression times became quite stark for larger datasets (i.e., DHFR and Mutagenicity). To provide more evidence on the scalability of our method, we also experimented with the larger Tox21AR-LBD data,[1] which consists of about 8600 graphs. Both our method and Algebraic distance performed very well in terms of classification accuracy ($\sim 97\%$) on this data. Our approach took in total about 39 seconds to compress graphs in this dataset to 90% (low compression), and about 41 seconds in total to compress to 10% (high compression). In contrast, the Algebraic distance method took about 48 seconds to compress to 90% and a significantly longer time, i.e., 3.5 minutes to compress to 10%. The other baselines failed to compress this data.

4.2 Compressing synthetic and real examples

We now describe our second set of experiments with both synthetic and real data to show that our method can be seeded with useful prior information toward preserving interesting patterns in the compressed structures. This flexibility in specifying the prior information makes our approach especially well-suited for downstream tasks, where domain knowledge is often available.

Fig. 2 demonstrates the effect of compressing a synthetic tree-structured graph. The penultimate level consists of four nodes, each of which has four child nodes as leaves. We introduce asymmetry with respect to the different nodes by specifying different combinations of $c(e)$ for edges e between the leaf nodes and their parents: there are respectively one, two, three and four heavy edges (i.e. with $c(e) = 0.5$) from the internal nodes 1, 2, 3, and 4 to their children, i.e., the leaves in their subtree.

As shown in Fig. 2, our method adapts to the hierarchical structure with a change in the amount of compression from just one node to about three-fourths of the entire graph. We also show mean-

---

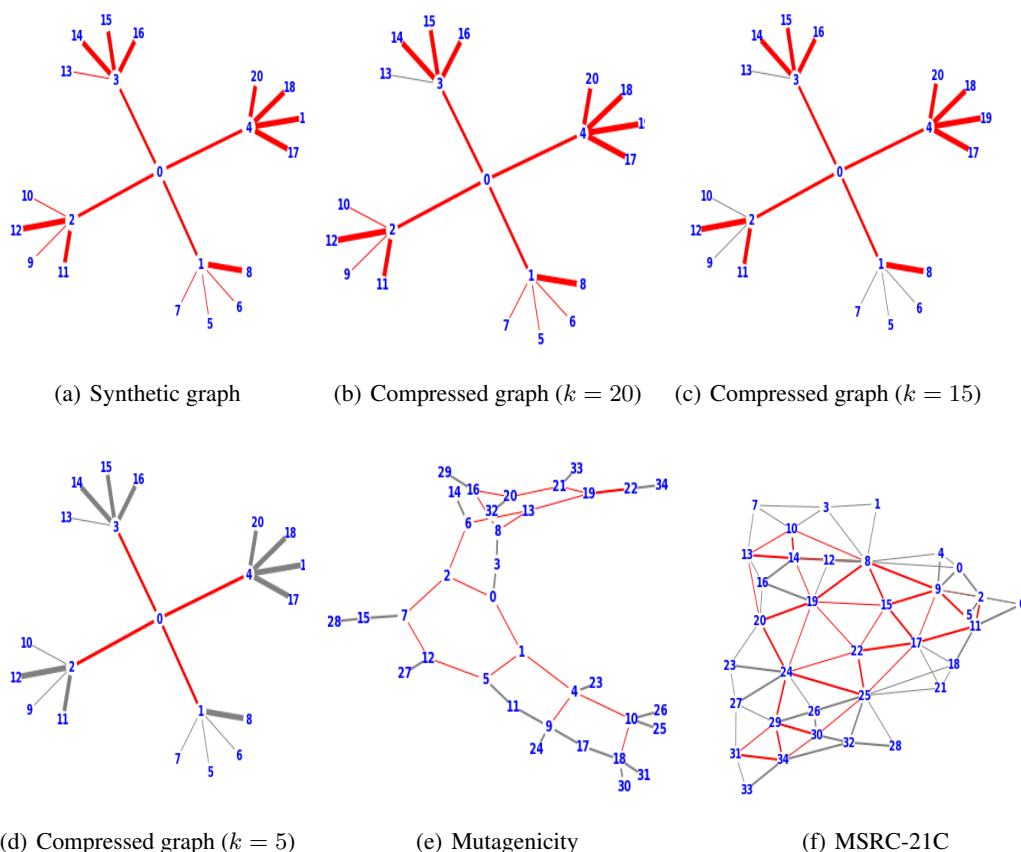[1]https://tripod.nih.gov/tox21/challenge/data.jsp

Figure 2: **Visualizing compression on synthetic and real examples.** (a) A synthetic graph structured as a 4-ary tree of depth 2. The root 0 is connected to its neighbors by edges having $c(e) = 0.3$. All the other edges have either $c(e) = 0.5$ (thickest) or $c(e) = 0.1$ (lightest). The left out portions are shown in gray. (b) Leaf node 13, which is in the same subtree as three nodes with heavy edges 14-16, is the first to go. (c) Proceeding further, 9 and 10 are left out followed by the remaining nodes (i.e. 5, 6, 7) connected by light edges. (d) When the graph is compressed to 5 vertices, only the root and its neighbors remain despite bulkier subtrees, e.g. the one with node 4 and its neighbors, that are discarded. Thus, our method yields meaningful compression on this synthetic example. (e-f) Compressed structures pertaining to some sample graphs from real datasets that have some other interesting motifs. In each case, the compressed graph consists of red edges and the incident vertices, while the discarded parts are shown in gray. All the figures here are best viewed in color.

ingful compression on some examples from real datasets. The bottom row of Fig. 2 shows two such examples, one each from Mutagenicity and MSRC-21C. For these graphs, we used the same specification for c as in section 4.1. The example from Mutagenicity contains patterns such as rings and backbone structures that are ubiquitous in molecules and proteins. Likewise, the other example is a good representative of the MSRC-21C dataset from computer vision.

Thus, our method encodes prior information, and provides fast and effective graph compression for downstream applications.

Acknowledgments

References

[1] A. Loukas and P. Vandergheynst. Spectrally approximating large graphs with smaller graphs. In *International Conference on Machine Learning (ICML)*, 2018.

[2] I. S. Dhillon, Y. Guan, and B. Kulis. Weighted graph cuts without eigenvectors a multilevel approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 29, 2007.

[3] L. Wang, Y. Xiao, B. Shao, and H. Wang. How to partition a billion-node graph. In *International Conference on Data Engineering (ICDE)*, pages 568–579, 2014.

[4] E. Ravasz and A.-L. Barabasi. Hierarchical organization in complex networks. *Physical Review E*, 67(2):026112, 2003.

[5] B. Savas and I. Dhillon. Clustered low rank approximation of graphs in information science applications. In *SIAM International Conference on Data Mining (SDM)*, 2011.

[6] R. Kondor, N. Teneva, and V. K. Garg. Multiresolution matrix factorization. In *International Conference on Machine Learning (ICML)*, pages 1620–1628, 2014.

[7] N. Teneva, P. K. Mudrakarta, and R. Kondor. Multiresolution matrix compression. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1441–1449, 2016.

[8] S. Lafon and A. B. Lee. Diffusion maps and coarse-graining: A unified framework for dimensionality reduction, graph partitioning, and data set parameterization. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 28(9):1393–1403, 2006.

[9] J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations (ICLR)*, 2014.

[10] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Neural Information Processing Systems (NIPS)*, 2016.

[11] M. Simonovsky and N. Komodakis. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[12] D. Ron, I. Safro, and A. Brandt. Relaxation-based coarsening and multiscale graph organization. *Multiscale Modeling & Simulation*, 9(1):407–423, 2011.

[13] J. Chen and I. Safro. Algebraic distance on graphs. *SIAM Journal on Scientific Computing*, 33(6):3468–3490, 2011.

[14] O. E. Livne and A. Brandt. Lean algebraic multigrid (lamg): Fast graph laplacian linear solver. *SIAM Journal on Scientific Computing*, 34, 2012.

[15] D. I. Shuman, M. J. Faraji, and P. Vandergheynst. A multiscale pyramid transform for graph signals. *IEEE Transactions on Signal Processing*, 64(8):2119–2134, 2016.

[16] F. Dörfler and F. Bullo. Kron reduction of graphs with applications to electrical networks. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 60(1):150–163, 2013.

[17] G. B. Hermsdorff and L. M. Gunderson. A unifying framework for spectrum-preserving graph sparsification and coarsening. In *arXiv:1902.09702*, 2019.

[18] C. Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

[19] G. Karakostas. Faster approximation schemes for fractional multicommodity flow problems. *ACM Transactions on Algorithms*, 4(1):13:1–13:17, 2008.

[20] M. B. Cohen, A. Madry, P. Sankowski, and A. Vladu. Negative-weight shortest paths and unit capacity minimum cost flow in Õ(m10/7 log w) time: (extended abstract). In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 752–771, 2017.

[21] J. Solomon, R. Rustamov, L. Guibas, and A. Butscher. Continuous-flow graph transportation distances. In *arXiv: 1603.06927*, 2016.

[22] M. Essid and J. Solomon. Quadratically regularized optimal transport on graphs. *SIAM Journal on Scientific Computing*, 40(4):A1961–A1986, 2018.

[23] P. Hansen, J. Kuplinsky, and D. de Werra. Mixed graph colorings. *Mathematical Methods of Operations Research*, 45(1):145–160, 1997.

[24] B. Ries. Coloring some classes of mixed graphs. *Discrete Applied Mathematics*, 155(1):1–6, 2007.

[25] R. G. Cowell, A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter. *Probabilistic Networks and Expert Systems: Exact Computational Methods for Bayesian Networks*. Springer Publishing Company, Incorporated, 1st edition, 2007.

[26] M. Beck, D. Blado, J. Crawford, T. Jean-Louis, and M. Young. On weak chromatic polynomials of mixed graphs. *Graphs and Combinatorics*, 31(1):91–98, 2015.

[27] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning (ICML)*, pages 214–223, 2017.

[28] N. Courty, R. Flamary, A. Habrard, and A. Rakotomamonjy. Joint distribution optimal transportation for domain adaptation. In *Neural Information Processing Systems (NIPS)*, 2017.

[29] I. Redko, N. Courty, R. Flamary, and D. Tuia. Optimal transport for multi-source domain adaptation under target shift. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 849–858, 2019.

[30] C. Frogner, C. Zhang, H. Mobahi, M. Araya, and T. A. Poggio. Learning with a wasserstein loss. In *Neural Information Processing Systems (NIPS)*, pages 2053–2061, 2015.

[31] D. Alvarez-Melis, T. Jaakkola, and S. Jegelka. Structured optimal transport. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 84, pages 1771–1780, 2018.

[32] T. Vayer, L. Chapel, R. Flamary, R. Tavenard, and N. Courty. Optimal transport for structured data. In *arXiv: 1805.09114*, 2018.

[33] M. Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Neural Information Processing Systems (NIPS)*, 2013.

[34] J.-D. Benamou, G. Carlier, M. Cuturi, L. Nenna, and G. Peyré. Iterative bregman projections for regularized transportation problems. *SIAM Journal on Scientific Computing*, 37:A1111–A1138, 2015.

[35] J. Solomon. Convolutional wasserstein distances : Efficient optimal transportation on geometric domains. *ACM Transactions of Graphics (TOG)*, 34(4):66:1–66:11, 2015.

[36] A. Genevay, M. Cuturi, G. Peyré, and F. Bach. Stochastic optimization for large-scale optimal transport. In *Neural Information Processing Systems (NIPS)*, pages 3440–3448, 2016.

[37] V. Seguy, B. B. Damodaran, R. Flamary, N. Courty, A. Rolet, and M. Blondel. Large-scale optimal transport and mapping estimation. In *International Conference on Learning Representations (ICLR)*, 2018.

[38] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (methodological)*, 58(1):267–288, 1996.

[39] M. Tan, I. W. Tsang, and Li Wang. Towards ultrahigh dimensional feature selection for big data. *Journal of Machine Learning Research (JMLR)*, 15(1):1371–1429, 2014.

[40] M. Pilanci, M. J. Wainwright, and L. E. Ghaoui. Sparse learning via boolean relaxations. *Mathematical Programming*, 151(1):63–87, 2015.

[41] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the ℓ_1-ball for learning in high dimensions. In *International Conference on Machine Learning (ICML)*, 2008.

[42] W. Wang and M. A. Carreira-P. Projection onto the probability simplex: An efficient algorithm with a simple proof, and an application. In *arXiv: 1309.1541*, 2013.

[43] L. Condat. Fast projection onto the simplex and the ℓ_1 ball. *Mathematical Programming*, 158(1–2):575–585, 2016.

[44] A. Nemirovski. Prox-method with rate of convergence $o(1/t)$ for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM Journal on Optimization*, 15(1):229–251, 2004.

[45] A. Beck and M. Teboulle. A fast dual proximal gradient algorithm for convex minimization and applications. *Operations Research Letters*, 42(1):1–6, 2014.

[46] V. K. Garg, O. Dekel, and L. Xiao. Learning small predictors. In *Neural Information Processing Systems (NeurIPS)*, 2018.

[47] Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1):127–152, 2005.

[48] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011.

[49] N. P. Santhanam and M. J. Wainwright. Information-theoretic limits of selecting binary graphical models in high dimensions. *IEEE Transactions on Information Theory*, 58(7):4117–4134, 2012.

[50] P. Bühlmann. Statistical significance in high-dimensional linear models. *Bernoulli*, 19(4):1212–1242, 09 2013.

[51] J. J. Sutherland, L. A. O'Brien, and D. F. Weaver. Spline-fitting with a genetic algorithm: a method for developing classification structure-activity relationships. *J. Chem. Inf. Comput. Sci.*, 43:1906–1915, 2003.

[52] N. Kriege and P. Mutzel. Subgraph matching kernels for attributed graphs. In *International Conference on Machine Learning (ICML)*, 2012.

[53] M. Neumann, R. Garnett, C. Bauckhage, and K. Kersting. Propagation kernels: efficient graph kernels from propagated information. *Machine Learning*, 102(2):209–245, 2016.

[54] J. Kazius, R. McGuire, , and R. Bursi. Derivation and validation of toxicophores for mutagenicity prediction. *Journal of Medicinal Chemistry*, 48(1):312–320, 2005.

[55] N. Shervashidze, P. Schweitzer, E. J. van Leeuwen, K. Mehlhorn, and K. M. Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research (JMLR)*, 12:2539–2561, 2011.

[56] S. V. N. Vishwanathan, N. Schraudolph, R. Kondor, and K. Borgwardt. Graph kernels. *Journal of Machine Learning Research (JMLR)*, 11:1201–1242, 2010.

[57] R. Kondor and H. Pan. The multiscale laplacian graph kernel. In *Neural Information Processing Systems (NIPS)*, pages 2990–2998, 2016.

[58] B. Y. Weisfeiler and A. A. Leman. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsia*, 9, 1968.

[59] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations (ICLR)*, 2019.

[60] M. Sion. On general minimax theorems. *Pacific Journal of Mathematics*, 8(1):171–176, 1958.

A   Supplementary Material

We provide here proofs of all the results from the main text.

Proof of Theorem 1

Proof. We will prove the result by contradiction.

Suppose there is some $\hat{e} \in E$ such that the optimal solution assigns $J^+(\hat{e}) > 0$ and $J^-(\hat{e}) > 0$. Then we note that

$$a \triangleq \min\{J^+(\hat{e}), J^-(\hat{e})\} > 0 \,.$$

Consider an alternate solution $(\tilde{J}^+, \tilde{J}^-)$ such that $\forall \, e \in E$,

$$\tilde{J}^+(e) = \begin{cases} J^+(e) - a & \text{if } e = \hat{e} \\ J^+(e) & \text{if } e \in E \setminus \{\hat{e}\} \end{cases} \,,$$

$$\tilde{J}^-(e) = \begin{cases} J^-(e) - a & \text{if } e = \hat{e} \\ J^-(e) & \text{if } e \in E \setminus \{\hat{e}\} \end{cases} \,.$$

Clearly, $(\tilde{J}^+, \tilde{J}^-)$ is feasible for (1). Moreover, it achieves a lower value of the objective than the optimal solution (J^+, J^-). Therefore, (J^+, J^-) cannot be optimal. $\square$

Proof of Theorem 2

Proof. We introduce non-negative Lagrangian vectors α and β, respectively, for the constraints $Ft \preceq c$ and $-c \preceq Ft$. We consider the terms in the objective that depend on t

$$g(t) = t^\top(\rho_1 - \rho_0) + \alpha^\top(Ft - c) - \beta^\top(Ft + c) \,.$$

The gradient $\nabla g(t)$ must vanish at optimality, so

$$\rho_1^* - \rho_0 + F^\top(\alpha - \beta) = \mathbf{0} \,.$$

The first part of the theorem follows immediately by defining $\eta = \beta - \alpha$. A closer look at (1) reveals that $\eta = J^- - J^+$ is, in fact, the net flow along the edges e^- from (1).

Now, we prove the second part. By definition, in order for an edge e to be active, at least one of e^+ and e^- must be active, i.e., we must have $J^+(e) + J^-(e) > 0$. On the other hand, Theorem 1 implies that at least one of $J^+(e)$ and $J^-(e)$ is 0 for each $e \in E$ in the optimal solution. Combining these facts, we have that for any active edge e, exactly one of e^+ and e^- is active, i.e., exactly one of the inequalities $J^+(e) > 0$ and $J^-(e) > 0$ must hold. This immediately implies, by complementary slackness, that exactly one of $\alpha(e)$ or $\beta(e)$ is 0. Thus, for any active edge e, either the lower bound or the upper bound on $Ft^*(e)$ in the constraints $-c(e) \le Ft^*(e) \le c(e)$ must become tight. Therefore, we must have $Ft^*(e) \in \{\pm c(e)\}$. $\square$

Proof of Theorem 3

Proof. Note that since at least one coordinate of ϵ is strictly greater than 0, the feasible region is non-empty, and consequently, a unique projection exists. We introduce variables $\alpha \in \mathbb{R}$ and $\beta \in \mathbb{R}_+^d$, and form the Lagrangian

$$L(x, \alpha, \beta) = \frac{1}{2}\|x - y\|^2 - \alpha((x \odot \epsilon)^\top \mathbf{1} - 1) - \beta^\top(x \odot \epsilon) \,.$$

We now write the KKT conditions for the optimal solution x. For each $j \in [d]$, we must have

$$\begin{aligned} x_j - y_j - \alpha\epsilon_j - \beta_j\epsilon_j &= 0 \\ \epsilon_j x_j &\ge 0 \\ \beta_j &\ge 0 \\ \epsilon_j x_j \beta_j &= 0 \,. \end{aligned}$$

$$\text{Additionally,} \quad \sum_{j=1}^d \epsilon_j x_j = 1 \,.$$

Clearly, for $j \in [d] \triangleq \{1, 2, \ldots, d\}, \epsilon_j = 0 \implies x_j = y_j$. Therefore, without loss of generality we assume in the rest of the proof that $\epsilon_j > 0$ for all j. Then we can immediately simplify the KKT conditions to

$$x_j - y_j - \alpha\epsilon_j - \beta_j\epsilon_j = 0 \tag{10}$$
$$x_j \geq 0 \tag{11}$$
$$\beta_j \geq 0 \tag{12}$$
$$x_j\beta_j = 0 \tag{13}$$

$$\sum_{j=1}^{d} \epsilon_j x_j = 1 \tag{14}$$

We note that $x_j > 0 \xRightarrow{(13)} \beta_j = 0 \xRightarrow{(10)} y_j + \alpha\epsilon_j > 0 \xRightarrow{\epsilon_j > 0} y_j/\epsilon_j > -\alpha$, whereas

$$x_j = 0 \xRightarrow{(10)} y_j + \alpha\epsilon_j = -\beta_j\epsilon_j \xRightarrow[\epsilon_j>0]{(12)} y_j + \alpha\epsilon_j \leq 0 \xRightarrow{\epsilon_j>0} y_j/\epsilon_j \leq -\alpha .$$

This shows that the zero coordinates x_j correspond to smaller values of y_j/ϵ_j. Thus, we can sort the indices j in non-increasing order based on the ratio y_j/ϵ_j, reorder x according to the sorted indices, and find an index $\ell \in [d]$ such that $x_j > 0$ for $j \in [\ell]$ and 0 for $\ell < j \leq d$. Without loss of generality, we therefore assume that

$$x_1 \geq x_2 \ldots \geq x_\ell > 0 = x_{\ell+1} \ldots = x_d , \text{ and}$$
$$y_1/\epsilon_1 \geq y_2/\epsilon_2 \ldots \geq y_d/\epsilon_d . \tag{15}$$

We then have from (14) that

$$1 = \sum_{j=1}^{d} \epsilon_j x_j = \sum_{j=1}^{\ell} \epsilon_j x_j = \sum_{j=1}^{\ell} \epsilon_j(y_j + \alpha\epsilon_j)$$

$$\implies \alpha = \frac{1 - \sum_{j=1}^{\ell} \epsilon_j y_j}{\sum_{j=1}^{\ell} \epsilon_j^2} . \tag{16}$$

Thus, our task essentially boils down to finding the number of positive coordinates ℓ. We now show that

$$\ell = \max \left\{ j \in [d] \,\bigg|\, y_j + \epsilon_j \frac{(1 - \sum_{i=1}^{j} \epsilon_i y_i)}{\sum_{i=1}^{j} \epsilon_i^2} > 0 \right\} .$$

First consider $j < \ell$. Then $y_j/\epsilon_j > -\alpha$ for $j \in [\ell]$. Noting that $\epsilon_j > 0$ for all j and using (16), we must have

$$y_j + \epsilon_j \frac{(1 - \sum_{i=1}^{j} \epsilon_i y_i)}{\sum_{i=1}^{j} \epsilon_i^2}$$

$$= \frac{\epsilon_j}{\sum_{i=1}^{j} \epsilon_i^2} \left(y_j \frac{\sum_{i=1}^{j} \epsilon_i^2}{\epsilon_j} + 1 - \sum_{i=1}^{j} \epsilon_i y_i \right)$$

which has the same sign as

$$y_j \frac{\sum_{i=1}^{j} \epsilon_i^2}{\epsilon_j} + 1 - \sum_{i=1}^{j} \epsilon_i y_i$$

$$= y_j \frac{\sum_{i=1}^{j} \epsilon_i^2}{\epsilon_j} + \sum_{i=j+1}^{\ell} \epsilon_i y_i + 1 - \sum_{i=1}^{\ell} \epsilon_i y_i$$

$$= y_j \frac{\sum_{i=1}^{j} \epsilon_i^2}{\epsilon_j} + \sum_{i=j+1}^{\ell} \epsilon_i y_i + \alpha \sum_{i=1}^{\ell} \epsilon_i^2$$

$$= \left(\frac{y_j}{\epsilon_j} + \alpha \right) \sum_{i=1}^{j} \epsilon_i^2 + \sum_{i=j+1}^{\ell} \epsilon_i^2 \left(\frac{y_i}{\epsilon_i} + \alpha \right)$$

$$> 0 .$$

14

Now consider $j = \ell$. Since $y_\ell/\epsilon_\ell > -\alpha$ and $\epsilon_\ell > 0$, we have $y_\ell + \alpha\epsilon_\ell > 0$. Thus

$$
\begin{aligned}
y_j + \epsilon_j \frac{(1 - \sum_{i=1}^{j} \epsilon_i y_i)}{\sum_{i=1}^{j} \epsilon_i^2} &= y_\ell + \epsilon_\ell \frac{(1 - \sum_{i=1}^{\ell} \epsilon_i y_i)}{\sum_{i=1}^{\ell} \epsilon_i^2} \\
&= y_\ell + \alpha\epsilon_\ell > 0 \,.
\end{aligned}
$$

Finally, we consider $\ell < j \le d$. We note that

$$
\begin{aligned}
&y_j + \epsilon_j \frac{(1 - \sum_{i=1}^{j} \epsilon_i y_i)}{\sum_{i=1}^{j} \epsilon_i^2} \\
&= \frac{\epsilon_j}{\sum_{i=1}^{j} \epsilon_i^2} \left(y_j \frac{\sum_{i=1}^{j} \epsilon_i^2}{\epsilon_j} + 1 - \sum_{i=1}^{j} \epsilon_i y_i \right) \,,
\end{aligned}
$$

which has the same sign as

$$
\begin{aligned}
&y_j \frac{\sum_{i=1}^{j} \epsilon_i^2}{\epsilon_j} + 1 - \sum_{i=1}^{j} \epsilon_i y_i \\
&= y_j \frac{\sum_{i=1}^{j} \epsilon_i^2}{\epsilon_j} + 1 - \sum_{i=1}^{\ell} \epsilon_i y_i - \sum_{i=\ell+1}^{j} \epsilon_i y_i \\
&= y_j \frac{\sum_{i=1}^{j} \epsilon_i^2}{\epsilon_j} + \alpha \sum_{i=1}^{\ell} \epsilon_i^2 - \sum_{i=\ell+1}^{j} \epsilon_i y_i \\
&= \left(\frac{y_j}{\epsilon_j} + \alpha \right) \sum_{i=1}^{\ell} \epsilon_i^2 + \sum_{i=\ell+1}^{j} \epsilon_i^2 \left(\frac{y_j}{\epsilon_j} - \frac{y_i}{\epsilon_i} \right) \,, \\
&\le 0 \,,
\end{aligned}
$$

by leveraging the sorted property in (15) and the fact that $y_j/\epsilon_j \le -\alpha$ for $j \in [\ell]$.

Therefore, we have shown that $y_j + \epsilon_j \frac{(1 - \sum_{i=1}^{j} \epsilon_i y_i)}{\sum_{i=1}^{j} \epsilon_i^2} > 0$ for all $j \in [\ell]$, and at most 0 for $\ell < j \le d$. Algorithm 1 implements this procedure, and that proves its correctness. The $O(d \log d)$ time complexity is due to the cost of sorting the indices $j \in [d]$ based on y_j/ϵ_j. $\square$

Proof of Theorem 4

Proof. Recall the formulation (6):

$$
\min_{\substack{\epsilon \in \mathcal{E}_k \\ \tilde{\rho}_1 \odot \epsilon \in \Delta(V)}} \max_{t \in \mathcal{T}_{F,c}} \underbrace{t^\top (\tilde{\rho}_1 \odot \epsilon - \rho_0) + \frac{\lambda}{2} ||\tilde{\rho}_1||^2}_{\phi(\epsilon, t, \tilde{\rho}_1)} \,.
$$

Making the constraints $\mathcal{E}_k$ explicit, we get

$$
\min_{\substack{\epsilon \in \{0,1\}^{|V|} \\ \epsilon^\top \mathbf{1} \le k}} \left(\min_{\substack{\tilde{\rho}_1 \in \mathbb{R}^{|V|} \\ \tilde{\rho}_1 \odot \epsilon \in \Delta(V)}} \max_{t \in \mathcal{T}_{F,c}} \phi(\epsilon, t, \tilde{\rho}_1) \right) \,. \tag{17}
$$

Note that for any fixed ϵ (a) $\{\tilde{\rho}_1 \in \mathbb{R}^{|V|} \mid (\tilde{\rho}_1 \odot \epsilon) \in \Delta(V)\}$ is convex, and $\mathcal{T}_{F,c}$ is convex and compact, (b) $\phi(\epsilon, t, \tilde{\rho}_1)$ is continuous, and (c) for every fixed t, $\phi(\epsilon, t, \cdot)$ is convex in $\tilde{\rho}_1$; while for every fixed $\tilde{\rho}_1$, $\phi(\epsilon, \cdot, \tilde{\rho}_1)$ is linear (thus concave) in t. Therefore, invoking the Sion's minimax theorem [60], we can swap the order of $\min$ and $\max$ within the parentheses in (17), and obtain

$$
\min_{\substack{\epsilon \in \{0,1\}^{|V|} \\ \epsilon^\top \mathbf{1} \le k}} \left(\max_{t \in \mathcal{T}_{F,c}} \min_{\substack{\tilde{\rho}_1 \in \mathbb{R}^{|V|} \\ \tilde{\rho}_1 \odot \epsilon \in \Delta(V)}} \phi(\epsilon, t, \tilde{\rho}_1) \right) \,. \tag{18}
$$

We introduce Lagrangian variables $\nu \in \mathbb{R}_+^{|V|}$ and $\zeta \in \mathbb{R}$, respectively, for the simplex constraints (a) $\mathbf{0} \preceq \tilde{\rho}_1 \odot \epsilon$ and (b) $(\tilde{\rho}_1 \odot \epsilon)^\top \mathbf{1} = 1$ to get

$$\Phi(\epsilon, t, \tilde{\rho}_1, \nu, \zeta) \triangleq \phi(\epsilon, t, \tilde{\rho}_1) + \zeta((\tilde{\rho}_1 \odot \epsilon)^\top \mathbf{1} - 1) - \nu^\top(\tilde{\rho}_1 \odot \epsilon) . \tag{19}$$

Applying the optimality conditions, we note for any fixed pair (ϵ, t), the corresponding optimal $\tilde{\rho}_1$ must satisfy

$$\partial_{\tilde{\rho}_1} \Phi(\epsilon, t, \tilde{\rho}_1, \nu, \zeta) = 0 ,$$

whereby

$$\tilde{\rho}_1 = - \left(\frac{(t - \nu) \odot \epsilon + \zeta\epsilon}{\lambda} \right) . \tag{20}$$

Plugging in $\tilde{\rho}_1$ from (20) into (19), we thus have the following equivalent dual formulation for (18)

$$\min_{\substack{\epsilon \in \{0,1\}^{|V|} \\ \epsilon^\top \mathbf{1} \leq k}} \max_{t \in \mathcal{T}_{F,c}} \max_{\nu \in \mathbb{R}_+^{|V|}, \zeta \in \mathbb{R}} \mathcal{M}(\epsilon, t, \nu, \zeta) ,$$

where

$$\mathcal{M}(\epsilon, t, \nu, \zeta) = \frac{-1}{2\lambda} ||(t - \nu + \zeta\mathbf{1}) \odot \epsilon||^2 - t^\top \rho_0 - \zeta . \tag{21}$$

Invoking the first order convex optimality condition for constrained optimization, the $\hat{\epsilon}$ obtained from relaxation of (6) is optimal if and only if

$$\mathbf{0} \in \left\{ \partial_\epsilon \underbrace{\max_{t \in \mathcal{T}_{F,c}} \max_{\nu \in \mathbb{R}_+^{|V|}, \zeta \in \mathbb{R}} \mathcal{M}(\epsilon, t, \nu, \zeta)}_{(A)} + \mathbb{N} \right\} , \tag{22}$$

where $\mathbb{N}$ is the normal cone of the relaxed constraints

$$\tilde{\mathcal{E}}_k = \left\{ \epsilon \in [0,1]^{|V|} \ \middle| \ \epsilon^\top \mathbf{1} \leq k \right\} .$$

Now we note that for any vector x, we can write $x \odot \epsilon = D(\epsilon)x$, where $D(\epsilon)$ is the diagonal matrix corresponding to ϵ. Also, since $\epsilon \in \{0,1\}^{|V|}$, we get $D(\epsilon)^\top D(\epsilon) = D(\epsilon)$. Thus, for any x, we have

$$||x \odot \epsilon||^2 = ||D(\epsilon)x||^2 = (D(\epsilon)x)^\top D(\epsilon)x = x^\top D(\epsilon)^\top D(\epsilon)x = x^\top D(\epsilon)x .$$

In particular, we can simplify $||(t - \nu + \zeta\mathbf{1}) \odot \epsilon||^2$ in (21) when we set x to $t - \nu + \zeta\mathbf{1}$. The theorem statement then follows immediately from (22) by representing $\mathbb{N}$ at the integral point ϵ^* and leveraging the non-negative dual parameter associated with the constraint $\epsilon^\top \mathbf{1} \leq k$. $\qquad \square$

Proof of Theorem 5

Proof. We will use the shorthand $\tilde{\rho}_{1v}$ for $\tilde{\rho}_1(v)$, and likewise for indexing t, ν, and ϵ. Using (20),

$$\tilde{\rho}_1 = - \left(\frac{(t - \nu) \odot \epsilon + \zeta\epsilon}{\lambda} \right) = - \left(\frac{t - \nu + \zeta\mathbf{1}}{\lambda} \right) \odot \epsilon .$$

Therefore,

$$\tilde{\rho}_1 \odot \epsilon = - \left(\frac{t - \nu + \zeta\mathbf{1}}{\lambda} \right) \odot \epsilon \odot \epsilon = `- \left(\frac{t - \nu + \zeta\mathbf{1}}{\lambda} \right) \odot \epsilon = \tilde{\rho}_1 , \tag{23}$$

since $\epsilon \in \{0,1\}^{|V|}$. We write one of the KKT conditions for optimality

$$\tilde{\rho}_1 \odot \epsilon \odot \nu = \tilde{\rho}_1 \odot \nu = \mathbf{0} .$$

We consider the different cases. Note that for $v \in V$, using (23), we have

$$\tilde{\rho}_{1v} > 0 \implies \nu_v = 0 \implies \tilde{\rho}_{1v} = - \frac{(t_v + \zeta)\epsilon_v}{\lambda} , \tag{24}$$

16

and

$$\tilde{\rho}_{1v} = 0 \implies (t_v - \nu_v + \zeta)\epsilon_v = 0 \implies \nu_v\epsilon_v = (t_v + \zeta)\epsilon_v . \tag{25}$$

But since $\nu_v \geq 0$, and $\epsilon_v \in \{0, 1\}$, we note that $\nu_v\epsilon_v \geq 0$. Then, by (25), we have

$$\nu_v\epsilon_v \geq 0 \implies -\frac{(t_v + \zeta)\epsilon_v}{\lambda} \leq 0 = \tilde{\rho}_{1v} . \tag{26}$$

Combining (24) and (26), we can write

$$\tilde{\rho}_{1v} = \max\left\{ -\frac{(t_v + \zeta)\epsilon_v}{\lambda}, \; 0 \right\} = \frac{\epsilon_v}{\lambda}\max\left\{ -(t_v + \zeta), \; 0 \right\} , \tag{27}$$

since $\epsilon_v \geq 0$ for all $v \in V$ and $\lambda > 0$. Therefore, we get $\tilde{\rho}_1 = \dfrac{\epsilon}{\lambda} \odot r_+$, where $r = -(t + \zeta\mathbf{1})$, and r_+ is computed by setting the negative coordinates of r to 0.

Moreover, since $\tilde{\rho}_1 \odot \nu = \mathbf{0}$, we can eliminate ν from (19) and write (17) as

$$\min_{\epsilon \in \mathcal{E}_k} \; \max_{t \in \mathcal{T}_{F,c}} \; \max_{\zeta \in \mathbb{R}} \quad t^\top(\tilde{\rho}_1 - \rho_0) + \frac{\lambda}{2}||\tilde{\rho}_1||^2 + \zeta(\tilde{\rho}_1^\top\mathbf{1} - 1)$$

Substituting for $\tilde{\rho}_1$ from (27), we obtain the following equivalent problem

$$\min_{\epsilon \in \mathcal{E}_k} \; \max_{t \in \mathcal{T}_{F,c}} \; \max_{\zeta \in \mathbb{R}} \quad -\frac{r^\top}{\lambda}(\epsilon \odot r_+) + \frac{1}{2\lambda}r_+^\top(\epsilon \odot r_+) - t^\top\rho_0 - \zeta ,$$

$$= \min_{\epsilon \in \mathcal{E}_k} \; \max_{t \in \mathcal{T}_{F,c}} \; \max_{\zeta \in \mathbb{R}} \quad \frac{-1}{2\lambda}\epsilon^\top(r_+ \odot (2r - r_+)) - t^\top\rho_0 - \zeta ,$$

which can be written as

$$\min_{\epsilon \in \mathcal{E}_k} \; \max_{t \in \mathcal{T}_{F,c}} \; \max_{\zeta \in \mathbb{R}} \quad -\frac{1}{2\lambda}\sum_{v:r_v \geq 0}\epsilon_v r_v^2 - t^\top\rho_0 - \zeta$$

$$= \quad \min_{\epsilon \in \mathcal{E}_k} \; \max_{t \in \mathcal{T}_{F,c}} \; \max_{\zeta \in \mathbb{R}} \quad -\frac{1}{2\lambda}\sum_{v:t_v \leq -\zeta}\epsilon_v r_v^2 - t^\top\rho_0 - \zeta$$

$$= \quad \min_{\epsilon \in \mathcal{E}_k} \; \max_{t \in \mathcal{T}_{F,c}} \; \max_{\zeta \in \mathbb{R}} \quad -\frac{1}{2\lambda}\sum_{v:t_v \leq -\zeta}\epsilon_v(t_v + \zeta)^2 - t^\top\rho_0 - \zeta$$

$$= \quad \min_{\epsilon \in \mathcal{E}_k} \; \max_{t \in \mathcal{T}_{F,c}} \; \max_{\zeta \in \mathbb{R}} \quad -\frac{1}{2\lambda}\sum_{v:t_v \leq -\zeta}\left(\epsilon_v(t_v + \zeta)^2 + 2\lambda t_v\rho_{0,v}\right) - \sum_{v:t_v > -\zeta}t_v\rho_{0,v} - \zeta .$$

$\square$