Symmetric Interdiction for Matching Problems

Samuel Haney[1], Bruce Maggs[2], Biswaroop Maiti[3], Debmalya Panigrahi[4], Rajmohan Rajaraman[5], and Ravi Sundaram[6]

1 **Duke University, Durham, NC, USA**
shaney@cs.duke.edu
2 **Duke University, Durham, NC, USA; and Akamai Technologies, Cambridge, MA, USA**
bmm@cs.duke.edu
3 **Northeastern University, Boston, MA, USA**
biswaroop@ccs.neu.edu
4 **Duke University, Durham, NC, USA**
debmalya@cs.duke.edu
5 **Northeastern University, Boston, MA, USA**
rraj@ccs.neu.edu
6 **Northeastern University, Boston, MA, USA**
koods@ccs.neu.edu

Abstract

Motivated by denial-of-service network attacks, we introduce the symmetric interdiction model, where both the interdictor and the optimizer are subject to the same constraints of the underlying optimization problem. We give a general framework that relates optimization to symmetric interdiction for a broad class of optimization problems. We then study the symmetric matching interdiction problem – with applications in traffic engineering – in more detail. This problem can be simply stated as follows: find a matching whose removal minimizes the size of the maximum matching in the remaining graph. We show that this problem is APX-hard, and obtain a 3/2-approximation algorithm that improves on the approximation guarantee provided by the general framework.

1998 ACM Subject Classification G.2.2 [Graph Theory] Graph Algorithms, Network Problems

Keywords and phrases Approximation algorithms, matching, interdiction

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.9

1 Introduction

A recent study of malicious network traffic observed at Microsoft data centers [17] made the surprising observation that a large volume of attack traffic originated from virtual machines hosted within the data centers themselves. The machines generating these attacks may have been compromised, or they may have been rented with stolen credit cards or on a free-trial basis. While the authors of the study used heuristics to identify traffic that was obviously malicious, in general it is very difficult to distinguish legitimate traffic from malicious traffic. In particular, an attacker in possession of a "botnet" of compromised machines can launch a denial-of-service attack against a service simply by using these machines to send a large number of legitimate-looking requests to the servers that implement the service.

The following question then arises: how does a network operator decide which connection requests to admit if she cannot distinguish between legitimate and malicious requests? One natural strategy is to minimize *regret*: the number of legitimate requests that are not served

but might have been otherwise. This motivates us to define the *symmetric interdiction* model in this paper, where the goal is to select a feasible set of edges whose removal minimizes the maximum feasible set in the remaining graph. We give a general framework for converting algorithms for a broad class of optimization problems to algorithms for the corresponding symmetric interdiction problems.

We instantiate our general model in the *symmetric matching interdiction* problem (abbreviated SMI in the rest of the paper), where the goal is to select a matching whose removal minimizes the maximum matching in the remaining graph. The SMI problem models our motivating scenario. Suppose clients located in a data center issue requests to servers in the same data center, where each client and each server has the capacity to participate in a single client-server interaction. Each client provides the operator of the data center with a list of servers it would like to contact, and the operator selects a matching of clients and servers. The operator would prefer to prioritize legitimate requests, but cannot distinguish between legitimate and malicious clients. By minimizing the size of the remaining maximum matching, an optimal solution to the SMI problem bounds the number of legitimate requests that are not satisfied but might otherwise have been. For the SMI problem, we show hardness results, and give a carefully designed algorithm that improves upon the result obtained from the general framework.

Main Results. Consider a generic optimization problem Π that is specified by an input graph $G = (V, E)$, by a set $\mathcal{F}$ of subgraphs of G which constitute feasible solutions to the problem, and a maximization (resp., minimization) objective function f on graphs. An example of Π is the maximum matching problem: $\mathcal{F}$ is the set of all matchings and the function f returns the number of edges in the matching. For the optimization problem Π, we define the symmetric interdiction problem $I(\Pi)$ as follows: the goal is to produce a subgraph $H = (V, F)$ of G such that H is in $\mathcal{F}$ and minimizes (resp., maximizes) the optimum value of f achievable on the remaining graph $(V, E \setminus F)$. Thus, the symmetric matching interdiction (SMI) problem is given a graph G and seeks a matching M of G so as to minimize the maximum matching in $G \setminus M$.

Our first result is a general framework for converting optimization algorithms to symmetric interdiction algorithms for a broad class of problems. This result, described informally below, is stated formally in Theorem 3 and proved in Section 2.

▶ **Theorem 1** (Informal). *An α-approximation to a packing problem Π implies a $(1 + \alpha)$-approximation to the corresponding symmetric interdiction problem $I(\Pi)$, modulo some technical conditions.*

Next, we focus on the SMI problem. Theorem 3 implies that any maximum matching algorithm is a 2-approximation algorithm for this problem. In fact, we show that any *maximal* matching also achieves an approximation factor of 2. However, this is the limit of the general framework in the sense that there are graphs where a maximum matching has an approximation factor of exactly 2 for the SMI problem. Our main algorithmic contribution is to obtain a more careful algorithm for the SMI problem that obtains an approximation factor of 1.5. We complement this result with a proof of APX-hardness of the problem by giving an approximation lower bound of $(1 + \epsilon)$ for small but fixed positive ϵ.

▶ **Theorem 2.** *There is a polynomial-time deterministic algorithm for the symmetric matching interdiction problem with an approximation factor of* 1.5*. Moreover, the symmetric matching interdiction problem is APX-hard.*

Extensions. We consider a randomized variant of the SMI problem in Section 5. Specifically, we show that if the interdictor is allowed to use randomness that is invisible to the optimizer, then the SMI problem becomes polynomial time solvable. Another natural extension of the SMI problem that captures practical parameters arising in networking is the capacitated case, where every edge has a capacity and the input and output ports have maximum capacities on the total amount of network flow that can be routed through them. On the other hand, if the edge capacities are unsplittable and both the interdictor's and optimizer's solutions are edge subsets, then the corresponding optimization problem is a special case of the previously studied *demand matching* problem [22]. Existing results for the optimization problem, applied to our interdiction framework, gives an approximation algorithm for the interdiction problem using Theorem 1. Improving this result using a more specific algorithm, such as the one that we give for the SMI problem, is left as an open problem in this work.

Finally, our symmetric interdiction framework can be applied to other diverse combinatorial optimization problems. See Section 6 for a brief discussions on other symmetric interdiction problems.

Related Work. Interdiction variants of classical graph optimization problems have attracted considerable research interest in recent years. Typically, these problems are modeled as a two-step game between an *interdictor* and an *optimizer*. In the first step, the interdictor removes a limited number of edges from the graph, with the goal of worsening the objective of the optimizer who solves the graph optimization problem on the remaining graph in the second step. For instance, in the matching interdiction problem, the goal is to remove at most k edges (for a given k) such that the size of the maximum matching in the remaining graph is minimized [26]. One can similarly define interdiction variants for maximum flow [7, 9, 10, 25, 27, 6, 1, 5, 16], minimum spanning tree [28, 8], and many other classic graph optimization problems [4, 23, 12, 15]. The main distinction between this model and the symmetric interdiction model is that both the interdictor and the optimizer in our problem are constrained by the same feasibility conditions, whereas the interdictor was constrained by a budget on the number of edges in previous work.

The SMI problem is similar to the matching interdiction problem studied by Kamalian *et al.* [14, 13], the key difference being that the interdictor's matching is also required to be a maximum matching in their case. We show that this restriction can produce suboptimal SMI solutions; indeed, the results of [14, 13] have no implication for SMI. More broadly, interdiction problems have a long history, having been studied for military applications in the Cold War [20]. Closer to our work, they have been used to model competitive markets in economic theory. In particular, in the Stackelberg model [24], two firms compete sequentially on the quantity of output they produce of a homogeneous good. Furthermore, both players play by the same rules and therefore must operate under the same constraints. This is conceptually identical to our symmetric interdiction model and we hope that this model will be applied to other domains in the future.

2 Symmetric Interdiction: A General Framework

In this section, we give a general theorem that relates symmetric interdiction problems to their corresponding optimization problems for a broad class of optimization problems called *packing* problems. This includes many classical problems such as maximum matching, knapsack, maximum flow, etc. Formally, packing problems are those that can be encoded by the linear program (LP) given below, where all entries of the coefficient matrix $\mathbf{A}$, and that

of vectors $\mathbf{b}$ and $\mathbf{c}$ are non-negative:

$$\text{maximize } \mathbf{c}^\intercal\mathbf{x}, \text{ subject to } \mathbf{A}\mathbf{x} \leq \mathbf{b} \text{ and } \mathbf{0} \leq \mathbf{x} \leq \mathbf{1}. \tag{1}$$

Suppose $\mathbf{x}$ is a feasible solution to LP (1). Then, we define the *residual LP* of $\mathbf{x}$ as:

$$\text{maximize } \mathbf{c}^\intercal\mathbf{y}, \text{ subject to } \mathbf{A}\mathbf{y} \leq \mathbf{b} \text{ and } \mathbf{0} \leq \mathbf{y} \leq \mathbf{1} - \mathbf{x}. \tag{2}$$

The *symmetric interdiction problem* is to find a feasible solution $\mathbf{x}$ that minimizes the optimal solution to the residual LP of $\mathbf{x}$. While we only focus on packing problems in this paper, we note that one can analogously define symmetric interdiction for covering problems[1] as well. Additionally, we note that all results in this section hold when $\mathbf{x}$ and $\mathbf{y}$ are constrained to be integral.

We call LP (1) the *optimization problem*. In this section, we develop a framework to obtain approximate solutions to the interdiction problem using exact/approximate solutions to the optimization problem. Before stating the result formally (Theorem 3), we set up some basic notation. Let

$$\mathbf{x} \setminus \mathbf{x}' = \begin{pmatrix} \max(0, x_1 - x'_1) \\ \vdots \\ \max(0, x_n - x'_n) \end{pmatrix}. \tag{3}$$

Note that $\mathbf{x} \setminus \mathbf{x}'$ is feasible if $\mathbf{x}$ and $\mathbf{x}'$ are feasible.

Let $\mathbf{x}^*$ be an optimal solution to the interdiction problem, and let $\mathbf{y}^*$ be an optimal solution to the residual LP w.r.t. $\mathbf{x}^*$. Now, consider a solution $\mathbf{x}$ that is feasible for LP (1). Ideally, we would like to claim that if $\mathbf{x}$ is an approximately optimal solution for the optimization problem, then it is also an approximately optimal for the interdiction problem. Unfortunately, this may not be true in general. However, we can show this connection between optimization and interdiction if $\mathbf{x}$ satisfies the following stronger condition:

$$\mathbf{c}^\intercal(\mathbf{x}^* \setminus \mathbf{x}) \leq \alpha \cdot \mathbf{c}^\intercal(\mathbf{x} \setminus \mathbf{x}^*) \text{ for some approximation factor } \alpha \geq 1. \tag{4}$$

This condition says that after removing any overlap between the interdiction and optimization solutions, the approximation ratio must be α. For example, consider an optimal interdiction solution M^* to the maximum matching problem, and another matching M. After removing edges that appear in both M and M^*, the number of remaining edges in M must be within a factor α of the number of remaining edges in M^*. In particular, when $\mathbf{x}$ is an optimal solution to the optimization problem, condition (4) holds with $\alpha = 1$ for any maximization problem. Now, we formally state and prove the theorem that establishes the relationship between optimization and interdiction.

▶ **Theorem 3.** *Let $\mathbf{x}^*$ be an optimal solution to the interdiction problem, and let $\mathbf{y}^*$ be an optimal solution to the residual LP w.r.t. $\mathbf{x}^*$. Suppose $\mathbf{x}$ is a feasible solution satisfying condition (4), i.e., $\mathbf{c}^\intercal(\mathbf{x}^* \setminus \mathbf{x}) \leq \alpha \cdot \mathbf{c}^\intercal(\mathbf{x} \setminus \mathbf{x}^*)$. Then, $\mathbf{x}$ is a $(1+\alpha)$-approximation to the corresponding interdiction problem. That is, if $\mathbf{y}$ is an optimal solution to the residual LP of $\mathbf{x}$, then $\mathbf{c}^\intercal\mathbf{y} \leq (1+\alpha) \cdot \mathbf{c}^\intercal\mathbf{y}^*$.*

[1] Covering problems are minimization problems where the constraints are $\mathbf{A}\mathbf{x} \geq \mathbf{b}$, with the same non-negativity restrictions.

Proof. We define the intersection $\mathbf{x} \cap \mathbf{x}'$ to be

$$\mathbf{x} \cap \mathbf{x}' = \begin{pmatrix} \min(x_1, x_1') \\ \vdots \\ \min(x_n, x_n') \end{pmatrix}.$$

Observe that $\mathbf{c}^\intercal \cdot \mathbf{y} = \mathbf{c}^\intercal \cdot (\mathbf{y} \setminus (\mathbf{1} - \mathbf{x}^*)) + \mathbf{c}^T \cdot (\mathbf{y} \cap (\mathbf{1} - \mathbf{x}^*))$. We upper bound each summand of this equation. We will need to use the fact that $\mathbf{x} \setminus \mathbf{x}^*$ is a feasible solution to the residual LP of $\mathbf{x}^*$. This follows from two observations: (1) $\mathbf{x}$ satisfies the constraint $\mathbf{Ax} \leq \mathbf{b}$ which implies $\mathbf{x} \setminus \mathbf{x}^*$ does too, and (2) $\mathbf{x} \leq \mathbf{1}$ implies $\mathbf{x} \setminus \mathbf{x}^* \leq \mathbf{1} - \mathbf{x}^*$. We first upper bound the left summand:

$$\begin{aligned} \mathbf{c}^\intercal(\mathbf{y} \setminus (\mathbf{1} - \mathbf{x}^*)) &\leq \mathbf{c}^\intercal((\mathbf{1} - \mathbf{x}) \setminus (\mathbf{1} - \mathbf{x}^*)) && \text{(since } \mathbf{y} \text{ is feasible for the residual LP of } \mathbf{x}) \\ &\leq \mathbf{c}^\intercal(\mathbf{x}^* \setminus \mathbf{x}) \\ &\leq \alpha \cdot \mathbf{c}^\intercal(\mathbf{x} \setminus \mathbf{x}^*) && \text{(by assumption that } \mathbf{x} \text{ satisfies (4))} \\ &\leq \alpha \cdot \mathbf{c}^\intercal \cdot \mathbf{y}^* \\ & && \text{(since } \mathbf{x} \setminus \mathbf{x}^* \text{ is feasible for the residual LP of } \mathbf{x}^*, \text{ shown above)} \end{aligned}$$

Next we bound the right summand. Note that $\mathbf{y} \cap (\mathbf{1} - \mathbf{x}^*)$ is feasible for the residual LP of $\mathbf{x}^*$ since $\mathbf{y} \cap (\mathbf{1} - \mathbf{x}^*) \leq (\mathbf{1} - \mathbf{x}^*)$. Therefore, since $\mathbf{y}^*$ is optimal for the residual LP of $\mathbf{x}^*$, we have $\mathbf{c}^\intercal \cdot (\mathbf{y} \cap (\mathbf{1} - \mathbf{x}^*)) \leq \mathbf{c}^\intercal \cdot \mathbf{y}^*$. Putting together the bounds on the left and right summands, we get

$$\mathbf{c}^\intercal \cdot \mathbf{y} = \mathbf{c}^\intercal \cdot (\mathbf{y} \setminus (\mathbf{1} - \mathbf{x}^*)) + \mathbf{c}^\intercal \cdot (\mathbf{y} \cap (\mathbf{1} - \mathbf{x}^*)) \leq \alpha \cdot \mathbf{c}^\intercal \cdot \mathbf{y}^* + \mathbf{c}^\intercal \cdot \mathbf{y}^* = (1 + \alpha) \cdot \mathbf{c}^\intercal \cdot \mathbf{y}^*. \blacktriangleleft$$

▶ **Corollary 4.** *Any optimal solution $\hat{\mathbf{x}}$ to the optimization problem, is a 2-approximation to the corresponding symmetric interdiction problem.*

Proof. Note that $\hat{\mathbf{x}} = (\hat{\mathbf{x}} \setminus \mathbf{x}^*) + (\hat{\mathbf{x}} \cap \mathbf{x}^*)$. Similarly, $\mathbf{x}^* = (\mathbf{x}^* \setminus \hat{\mathbf{x}}) + (\hat{\mathbf{x}} \cap \mathbf{x}^*)$. Since $\hat{\mathbf{x}}$ is an optimal solution to the optimization problem, $\mathbf{c}^\intercal \mathbf{x}^* \leq \mathbf{c}^\intercal \hat{\mathbf{x}}$. Therefore, $\mathbf{c}^\intercal(\mathbf{x}^* \setminus \hat{\mathbf{x}}) \leq \mathbf{c}^\intercal(\hat{\mathbf{x}} \setminus \mathbf{x}^*)$. ◀

3 Symmetric Matching Interdiction: A 3/2 Approximation

Let $G = (V, E)$ be a graph. Then the symmetric matching interdiction (SMI) problem is to find some matching M^* such that the maximum matching in the graph $(V, E \setminus M^*)$ is minimized.

From Corollary 4, we get that any maximum matching is a 2-approximation for the SMI problem. In fact, any *maximal* matching is also a 2-approximation.

▶ **Lemma 5.** *Any maximal matching is a 2-approximation for the symmetric matching interdiction problem.*

Proof. For a graph G, let M be a maximal matching and L be the maximum matching on $G \setminus M$. Each component of $M \cup L$ is a path or a cycle of alternating edges of M and L. Any edge that appears by itself in a component of $M \cup L$ must be in M, by the maximality of M.

Let C be a component of $M \cup L$ that contains at least one edge of L. We show that for any matching M^* on C, the maximum matching on $C \setminus M^*$ has at least $|L \cap C|/2$ edges, which will complete the proof. Let j be the number of edges of C. Then, $|L| = \frac{j}{2}$ if j is even, and $|L| \leq \frac{j+1}{2}$ if j is odd. That is, $|L| \leq \lceil \frac{j}{2} \rceil$.

We will show later by a case analysis in Lemma 8 that the maximum matching on $C \setminus M^*$ has at least $\lceil \frac{j-1}{3} \rceil$ edges for any M^*. Since $\lceil \frac{j}{2} \rceil / \lceil \frac{j-1}{3} \rceil \leq 2$ for integers $j \geq 2$, the lemma follows. ◀

This is better than the 3-approximation guarantee for maximal matchings that we get from Theorem 3. In fact, the approximation factor of 2 is the best achievable, if we were to choose an arbitrary maximum or maximal matching. Consider a length-4 path. The optimal interdiction solution contains the edges at the two ends, leaving a matching of size 1. On the other hand, the first and third edges form a maximum matching, but leaves behind a matching of size 2.

But, what if we choose the *best* maximum matching instead of an arbitrary one? In the previous example, the optimal interdiction solution also turned out to be a maximum matching. Our first result in this section is to show that there always exists a maximum matching that is a 3/2-approximation to the optimal interdiction matching. In the second part of this section, we make this result constructive, i.e., give a polynomial-time algorithm for finding such a maximum matching. Before describing our result, we note that the approximation factor of 3/2 is the best we can hope for from a maximum matching, even the best one. Consider a cycle of length 6. The optimal interdiction solution contains any pair of opposite edges, leaving behind two disjoint length-2 paths containing a matching of size 2. On the other hand, any maximum matching contains 3 edges, which leaves behind 3 disjoint components forming a matching of size 3.

3.1 Approximating the SMI problem with maximum matchings

We show that the maximum matching with the largest intersection with any fixed optimal solution to the SMI problem is a 3/2 approximation to the SMI problem. In this section, M^* denotes an optimal solution to SMI, i.e., a matching that minimizes the size of the maximum matching L^* in the remaining graph $(V, E \setminus M^*)$. M denotes a maximum matching on G, and L denotes a maximum matching in the remaining graph $(V, E \setminus M)$. All matchings and connected components that we refer to in this section are defined as sets of edges; hence, set operations are only on the edges and do not affect vertices.

For any M and L, the size of a matching on $(M \cup L) \setminus M^*$ serves as a lower bound on the size of L^*, since $(M \cup L) \subseteq E$. So, our goal will be to show that the size of L is at most 3/2 times the size of a matching that we construct in $(M \cup L) \setminus M^*$. We will show this individually for every component of $M \cup L$. Let C be a component of $M \cup L$. We say M is *locally* 3/2-competitive on C with respect to M^* if $C \setminus M^*$ contains a matching of at least 2/3 times the size of $C \cap L$. If M is locally 3/2-competitive for each component, then that implies an approximation factor of 3/2 overall.

For some fixed M^*, there are only certain types of components of $M \cup L$ that may not be locally competitive. We call these components *critical*, and define their structure below. Note that $M \cup L$ is a set of vertex disjoint paths and even-length cycles, since it is composed of two matchings.

▶ **Definition 6.** We call component C *critical* w.r.t. matching M^* if all the following hold:
1. C is an even-length path,
2. the edges at the two ends of C are in M^*, and
3. $C \setminus M^*$ is a set of length-2 paths.

We will show in Lemma 8 that critical components, as defined in Definition 6, are the only ones that may not be locally competitive. From Definition 6, for a component to be critical, it must be a path with ℓ edges, where $\ell \equiv 4 \mod 6$ edges. We call these components *bad*:

▶ **Definition 7.** Let C be a component of $M \cup L$. Call C *bad* if C is a path and $|C| \equiv 4 \mod 6$, where $|C|$ denotes the number of edges in C.

Note that all critical components are bad, but not vice-versa, since criticality also depends on the structure of M^*.

We next show that M is locally 3/2-competitive on all components that are not critical. In fact, the lemma gives tighter bounds, which will be helpful in developing an algorithm later. Note that, till now, the only assumption we have made about M is that it is a maximum matching, i.e., the next lemma holds for *all* maximum matchings.

▶ **Lemma 8.** *Fix M, L, M^*, and L^*. Let C be a component of $M \cup L$. Let ℓ^* denote the size of a maximum matching on $C \setminus M^*$, and c denote the number of edges in C. (Note that ℓ^*, summed over all components C, lower bounds the size of L^*.) Then,*

1. *If C is not bad and c is odd, $\ell^* \geq \frac{c-1}{3}$.*
2. *If C is not bad and c is even, $\ell^* \geq \frac{c}{3}$.*
3. *If C is bad but not critical, $\ell^* \geq \frac{c+2}{3}$.*
4. *If C is bad and critical, $\ell^* \geq \frac{c-1}{3}$.*

Proof. We find these lower bounds on ℓ^* by constructing a matching $\widehat{L}$ on $C \setminus M^*$. Note that $C \setminus M^*$ is either an even cycle or a set of vertex-disjoint paths. In the former case, we pick every alternate edge on the cycle in $\widehat{L}$. In the latter case, for each path, we pick every alternate edge in $\widehat{L}$, including the two edges at the ends for odd length paths. Let m^* denote $|M^* \cap C|$. $\widehat{L}$ has the following properties:

1. $\widehat{L}$ contains at least $\lceil \frac{c-m^*}{2} \rceil$ edges.
2. For each component of $C \setminus M^*$, $\widehat{L}$ contains at least one edge.

Next, we show that these two properties are sufficient to prove that for each of the 4 cases in the statement of the lemma, the corresponding inequality holds.

Case (1). Note that C must be a path, since all cycles have even length in the union of two matchings. Therefore, property 2 ensures that $\widehat{L}$ has at least $m^* - 1$ edges. Along with property 1, this implies $\ell^* \geq |\widehat{L}| \geq \min(m^* - 1, \frac{c-m^*}{2})$. Optimizing over the possible values of m^* then gives us $\ell^* \geq \frac{c-1}{3}$.

Case (2). C is either a path or a cycle. We treat these cases differently.

1. When C is a cycle, property 2 implies that $\widehat{L}$ has at least m^* edges. Along with property 1, this implies $\ell^* \geq |\widehat{L}| \geq \min(m^*, \frac{c-m^*}{2})$. Optimizing over the possible values of m^* gives $\ell^* \geq \frac{c}{3}$.
2. When C is a path, property 2 implies that $\widehat{L}$ has at least $m^* - 1$ edges. Identical to case (1) above, we can now infer that $\ell^* \geq \frac{c-1}{3}$. Since ℓ^* is integral, we can claim that $\ell^* \geq \lceil \frac{c-1}{3} \rceil$. We also know that c is even and $c \not\equiv 4 \mod 6$. Together, this shows that $\lceil \frac{c-1}{3} \rceil \geq \frac{c}{3}$, which implies that $\ell^* \geq \frac{c}{3}$.

Case (3). We subdivide into two cases based on the size of M^*. Note that $c \equiv 4 \mod 6$; hence, $\frac{c+2}{3}$ is an integer.

1. Suppose $m^* \neq \frac{c+2}{3}$. If $m^* \geq \frac{c+2}{3} + 1$, then property (2) implies $\widehat{L} \geq \frac{c+2}{3}$. On the other hand, if $m^* \leq \frac{c+2}{3} - 1$, then property (1) ensures that $|\widehat{L}| \geq \lceil \frac{c+1/2}{3} \rceil = \frac{c+2}{3}$. In either case, $\ell^* \geq |\widehat{L}| \geq \frac{c+2}{3}$.
2. Suppose $m^* = \frac{c+2}{3}$. If M^* does not contain at least one end edge of path C, then $C \setminus M^*$ has m^* components, and therefore, property (2) ensures that $\widehat{L}$ has at least m^* edges. Now, consider the case where M^* contains both end edges of path C. In this case, the number of components in $C \setminus M^*$ is $m^* - 1 = \frac{c-1}{3}$. But, the total number of edges in

$C \setminus M^*$ is $c - m^* = \frac{2c-2}{3}$. Therefore, the average number of edges in each component of $C \setminus M^*$ is 2. Since C is not critical w.r.t. M^*, every component in $C \setminus M^*$ cannot have exactly 2 edges. As a consequence, there must be at least one component α in $C \setminus M^*$ that contains at least 3 edges. By property (2), $\widehat{L}$ contains at least $m^* - 1$ edges, but this matching can be augmented by picking a second edge from component α to produce a matching of size m^* in $C \setminus M^*$. Therefore, $\ell^* \geq m^* = \frac{c+2}{3}$.

Case (4). The proof is identical to the proof of case (1). ◀

We claim that the above lemma implies that M is locally 3/2-locally competitive on all non-critical components. Let ℓ denote the number of edges of L in C. In case (1), $\ell = \frac{c-1}{2}$, and in cases (2) and (3), $\ell = \frac{c}{2}$. Only case (4) is not locally 3/2-competitive, since $\ell = \frac{c}{2}$.

We now show that there is a maximum matching that has no critical components with respect to a fixed optimal M^*; this proves the existence of a 3/2-approximate maximum matching.

▶ **Lemma 9.** *A maximum matching with the largest intersection with some optimal solution M^* is a 3/2-approximation to the optimal interdiction solution, i.e., $|L| \leq \frac{3}{2}|L^*|$.*

Proof. Let M be a maximum matching with the largest intersection with M^* and let L and L^* be arbitrary maximum matchings in the respective remaining graphs. Let C be a critical component in $M \cup L$. Since $|C|$ is even, one of its end edges must be in L. Call this edge e, and let f denote its adjacent edge in C (note that f is in M). Since C is critical, we have $e \in M^*$. Then $(M \setminus \{f\}) \cup \{e\}$ is also a maximum matching. Since $e \in M^*$ and $f \notin M^*$, this contradicts the fact that we chose M as the maximum matching that maximizes $|M \cap M^*|$. ◀

3.2 A 3/2-Approximation algorithm

In this section, we make the results of the previous section constructive. If we knew M^*, we could give an algorithm that performed swaps of the kind used in the proof of Lemma 9. These swaps would each increase the size of $M \cap M^*$, and we would eventually obtain a solution with no critical components. Unfortunately, we don't know M^*. We show, however, that sometimes we can perform sets of swaps such that the overlap of M with *every* optimal solution M^* is increased. If such a set of swaps does not exist, we argue that our solution is already a 3/2-approximation.

The formal algorithm is given in Algorithm 1. We outline the steps here. We start with an arbitrary maximum matching M, and a maximum matching in $G \setminus M$. We then repeatedly perform swaps of the form given above on the set of all bad components for a total of $|E| + 1$ iterations. Finally, we output the best matching found over all these iterations. We argue that while a 3/2-approximate solution has not been obtained, each iteration of swaps increases the overlap of M with every optimal solution. Such an increase cannot happen more than $|E|$ times, and therefore a 3/2-approximate solution is found in some iteration of the algorithm.

▶ **Lemma 10.** *Let M be a maximum matching and L be a maximum matching in $G \setminus M$. Suppose there exists an optimal interdiction solution M^* such that M^* is critical on at most half the bad paths in $M \cup L$. Then, $|L| \leq \frac{3}{2}|L^*|$.*

Before proving Lemma 10, we show that this implies correctness of the algorithm. Suppose that for some iteration of the algorithm, the condition from Lemma 10 does not hold, i.e.,

Algorithm 1 A 3/2-approximation algorithm for the SMI problem.

1: $M \leftarrow$ arbitrary maximum matching in G
2: $L \leftarrow$ arbitrary maximum matching in $G \setminus M$
3: $l_{\min} \leftarrow |L|$
4: $M_{\min} \leftarrow M$
5: **for** $j = 1 \rightarrow |E| + 1$ **do**
6: **if** $|L| < l_{\min}$ **then**
7: $l_{\min} \leftarrow |L|$
8: $M_{\min} \leftarrow M$
9: **for** bad path C in $M \cup L$ **do**
10: $M \leftarrow M \setminus \{e\} \cup \{f\}$ ▷ Let e be the edge at the end of C that is in L, $f \in M$ is the adjacent edge in C.
11: $L \leftarrow$ arbitrary maximum matching in $G \setminus M$.
12: **return** $M_{\min}$

every optimal solution M^* is critical on strictly more than half the bad paths in $M \cup L$. After the for loop beginning on line 9, the size of the intersection between M and every optimal solution will have increased. This is because for every M^*, every $e \rightarrow f$ swap on a critical path increases the size of the overlap between M^* and M by 1, while every $e \rightarrow f$ swap on a non-critical bad path decreases the overlap by at most 1. This increase in overlap can happen at most $|E|$ times, so after $|E| + 1$ iterations, we must have produced a solution M with $|L| \leq \frac{3}{2}|L^*|$ as desired. We now prove Lemma 10 using Lemma 8. Although critical components have a local approximation ratio slightly worse than 3/2, non-critical bad paths offset this with a ratio better than 3/2.

Proof of Lemma 10. Let $\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3, \mathcal{C}_4$ denote the sets of components of type (1), (2), (3), and (4) respectively from Lemma 8. Let ℓ^*_C denote a maximum matching on component $C \setminus M^*$. Also, let $E(C)$ denote the edges of component C and $E(\mathcal{C}_i) = \bigcup_{C \in \mathcal{C}_i} E(C)$. Then,

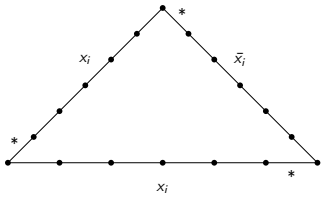
$$
\begin{aligned}
|L^*| &\geq \sum_{C \in \mathcal{C}_1 \cup \mathcal{C}_2 \cup \mathcal{C}_3 \cup \mathcal{C}_4} \ell^*_C \\
&\geq \sum_{C \in \mathcal{C}_1} \frac{|E(C)| - 1}{3} + \sum_{C \in \mathcal{C}_2} \frac{|E(C)|}{3} + \sum_{C \in \mathcal{C}_3} \frac{|E(C)| + 2}{3} + \sum_{C \in \mathcal{C}_4} \frac{|E(C)| - 1}{3} \quad \text{(from Lemma 8)} \\
&= \frac{|E(\mathcal{C}_1)| - |\mathcal{C}_1|}{3} + \frac{|E(\mathcal{C}_2)|}{3} + \frac{|E(\mathcal{C}_3)| + 2|\mathcal{C}_3|}{3} + \frac{|E(\mathcal{C}_4)| - |\mathcal{C}_4|}{3} \\
&\geq \frac{|E(\mathcal{C}_1)| - |\mathcal{C}_1|}{3} + \frac{|E(\mathcal{C}_2)|}{3} + \frac{|E(\mathcal{C}_3)| + |\mathcal{C}_3|}{3} + \frac{|E(\mathcal{C}_4)|}{3} \\
&\quad \text{(since } |\mathcal{C}_3| \geq |\mathcal{C}_4|, \text{ i.e., at most half of all bad paths are critical)} \\
&= \frac{2}{3}|L \cap \mathcal{C}_1| + \frac{2}{3}|L \cap \mathcal{C}_2| + \frac{2|L \cap \mathcal{C}_3| + |\mathcal{C}_3|}{3} + \frac{2}{3}|L \cap \mathcal{C}_4| \geq \frac{2}{3}|L|. \\
&\quad \text{(since } |L \cap C| = |C|/2 \text{ for } C \in \mathcal{C}_2 \cup \mathcal{C}_3 \cup \mathcal{C}_4 \text{ and } |L \cap C| = (|C| - 1)/2 \text{ for } C \in \mathcal{C}_1) \quad \blacktriangleleft
\end{aligned}
$$

4 Symmetric Matching Interdiction: Hardness of Approximation

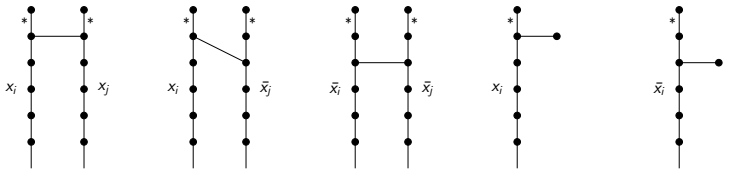
In this section, we show that the symmetric matching interdiction problem is APX-hard which rules out the possibility of a PTAS for the problem. We give an approximation-preserving reduction from a variant of MAX-SAT called 3-OCC-MAX-2-SAT that we define below.

▶ **Definition 11.** Let ϕ be a set of clauses, where each clause is a conjunction of at most 2 literals. Additionally, each variable appears in at most 3 literals in ϕ. Let k be an integer. (ϕ, k) is said to be in 3-OCC-MAX-2-SAT if there is a setting of the variables such that at least k clauses are satisfied.

3-OCC-MAX-2-SAT is known to be APX-hard [3]. To show the hardness of the SMI problem, we give an approximation preserving reduction from 3-OCC-MAX-2-SAT to the SMI problem. For the purposes of the reduction, we construct an instance graph G of the SMI problem from an instance of the 3-OCC-MAX-2-SAT problem (ϕ, k) as follows. For each variable x_i, we have a cycle in G containing $6z_i$ edges, where $z_i \leq 3$ is the number of times x_i appears as a literal in ϕ. We partition each cycle into z_i paths of length 6 each, which we call *literal paths*, such that each path is associated with one of the literals containing x_i. We order the edges of each path, denoting the first edge with '*' so that we can refer to the first, second, etc. edge on a literal path without ambiguity. The construction up until now is illustrated below:



We call all edges in such cycles *cycle edges*. Next, we add one edge to G for each clause in ϕ (we call these *clause edges*). Each clause contains either one or two literals. For a clause containing two literals, the clause edge connects the two literal paths corresponding to those literals. For a clause containing one literal, the clause edge connects that literal's path to a new vertex. Clause edges are adjacent to the second vertex on the literal path corresponding to a positive literal, and the third vertex on the literal path corresponding to a negative literal. Below, we illustrate the clauses $(x_i \vee x_j)$, $(x_i \vee \overline{x_j})$, $(\overline{x_i} \vee \overline{x_j})$, (x_i), and $(\overline{x_i})$, respectively.



This completes the construction of G. The following is our main technical lemma of the reduction.

▶ **Lemma 12.** *There is a setting of the variables that satisfies at least k clauses in ϕ if and only if there is a matching M such that in $G \setminus M$, the size of the maximum matching is at most $2\ell + m - k$, where m is the number of clauses in ϕ and ℓ is the number of literals.*

Before proving this lemma, we show that it is sufficient to prove APX-hardness of the SMI problem.

▶ **Theorem 13.** *Symmetric matching interdiction is APX-hard.*

Proof. Suppose we have an $(1+\epsilon)$-approximation to the SMI problem, i.e., a matching M in G such the maximum matching in $G \setminus M$ has size at most

$$(1+\epsilon) \cdot (2\ell + m - k) = 2\ell + m - \left[1 - \epsilon\left(\frac{2\ell + m}{k} - 1\right)\right] k.$$

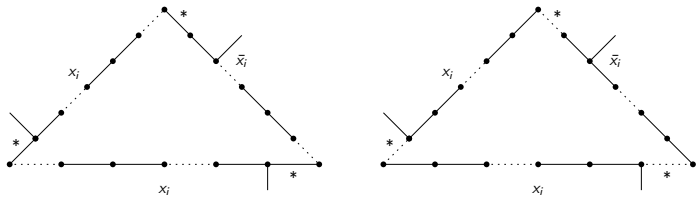
By Lemma 12, we can find a formula ϕ and an assignment $\mathbf{x}$ in the 3-OCC-MAX-2-SAT instance such that $\mathbf{x}$ satisfies at least $\left[1-\epsilon\left(\frac{2\ell+m}{k}-1\right)\right]k$ clauses of ϕ. Note that $\ell \leq 2m$ since each clause contains at most two literals; therefore, $2\ell+m \leq 5m$. If each variable is set i.i.d. to T/F with equal probability, then each clause is satisfied with probability 1/2 if it contains a single literal, and with probability 3/4 if it contains 2 literals. Therefore, the expected number of clauses satisfied by a 2-SAT formula under this random assignment is at least $m/2$. By the probabilistic method, it follows that the maximum number of satisfiable clauses $k \geq m/2$. Therefore, $m/k \leq 2$, which implies

$$1-\epsilon((2\ell+m)/k-1) \geq 1-\epsilon(5m/k-1) \geq 1-9\epsilon.$$

Therefore, this gives a $(1-9\epsilon)$-approximate solution to 3-OCC-MAX-2-SAT. ◀

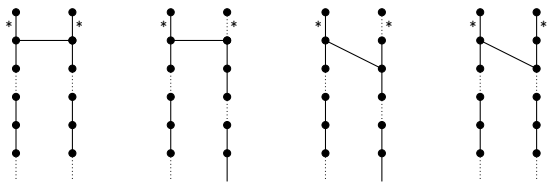
We spend the rest of the section proving Lemma 12. We first give a high level overview of the proof, and then give the technical details. We give a mapping from a setting of variables, $\mathbf{x}$ in ϕ, to a matching $M_{\mathbf{x}}$ in G. We argue that $\mathbf{x}$ satisfies k clauses of ϕ if and only if the maximum matching in $G \setminus M_{\mathbf{x}}$ contains $2\ell+m-k$ edges (Lemma 14). Then, we argue that for graph G produced by the reduction from a formula ϕ, there is a setting of variables $\mathbf{x}$ in ϕ such that $M_{\mathbf{x}}$ is the optimal solution to the SMI problem in G (Lemmas 15, 16, 17). Together, these lemmas prove Lemma 12.

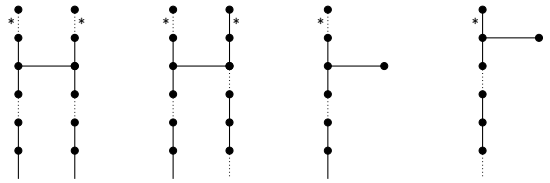
For an assignment $\mathbf{x}$ to the variables of ϕ, we construct matching $M_{\mathbf{x}}$ as follows: $M_{\mathbf{x}}$ does not contain any clause edge. $M_{\mathbf{x}}$ contains every third edge on each variable cycle. For the cycle corresponding to variable x_i, these edges are chosen in the following way: If x_i set to true, $M_{\mathbf{x}}$ contains the third and sixth edges of each literal path. We call $M_{\mathbf{x}}$ true on such a path. If x_i is set to false, $M_{\mathbf{x}}$ contains the first and fourth edges of each literal path. We call $M_{\mathbf{x}}$ false on such a path. For the cycle in G corresponding to variable x_i, we show the two possibilities for the edges in $G \setminus M_{\mathbf{x}}$ below; $M_{\mathbf{x}}$ is true on the first cycle, and false on the second.



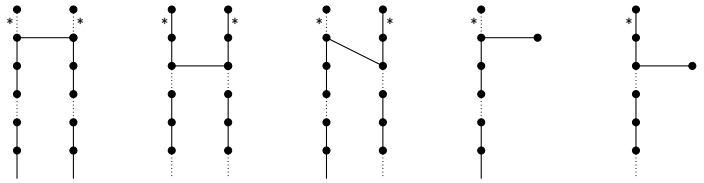
▶ **Lemma 14.** *An assignment* $\mathbf{x}$ *satisfies* k *clauses if and only if the maximum matching in* $G \setminus M_{\mathbf{x}}$ *has size* $2\ell+m-k$.

Proof. For each clause, we show that a maximum matching on the cycle and clause edges corresponding to that clause after removing $M_{\mathbf{x}}$ contains two edges for each literal in the clause, along with an additional edge if the clause is not satisfied by $\mathbf{x}$. This shows that the maximum matching on $G \setminus M_{\mathbf{x}}$ has size at most $2\ell+m-k$. Moreover, in each case there is a maximum matching that does not use the last edge on each literal path. Therefore, they can be combined into a single matching with $2\ell+m-k$ edges. To prove this, we enumerate over all types of clauses. Edges in $M_{\mathbf{x}}$ are drawn as dotted lines. The following are all possible satisfied clauses.





The following are the unsatisfied clauses:



This completes the proof. ◀

Lemma 14 implies the forward direction of Lemma 12. To show that the reduction holds in the other direction, we show that given any optimal solution M to the SMI problem, we can transform it to a matching $M_{\mathbf{x}}$ that is also optimal and corresponds to an assignment $\mathbf{x}$ of ϕ. We call such matchings that correspond to assignments in ϕ *consistent* matchings. The following are necessary and sufficient conditions for a matching to be consistent:

(a) On each variable cycle, the matching is either true or false (i.e. it contains either the third and sixth edges of each literal path, or the first and fourth edges), and

(b) the matching does not contain any clause edge.

We show that M can be transformed into a consistent matching as follows.

- If property (a) is violated, iteratively identify a cycle C on which M violates (a) and locally replace M with $M_{\mathbf{x}}$, which is defined below.
- Once only property (b) is violated, remove all remaining clause edges.

We show that neither of these steps increases the size of the maximum matching in $G \setminus M$; therefore, the eventual consistent matching is also optimal for the SMI problem.

First, we consider violations of property (a). Let assignment $\mathbf{x}$ be defined as follows: for each variable $x_i \in \mathbf{x}$, $x_i = true$ if x_i appears as at most one negative literal in ϕ and $x_i = false$ if x_i appears as at most one positive literal in ϕ. If M is not consistent, we will show that we can iteratively replace variable cycles of matching M with the corresponding variable cycles of $M_{\mathbf{x}}$.

M must violate property (a) on cycle C in one of the following two ways:

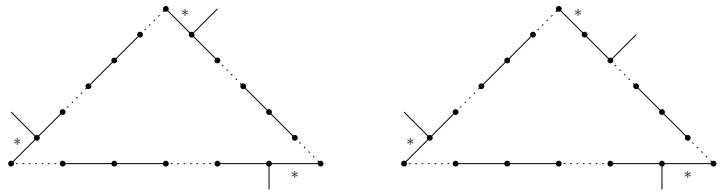
1. M does not contain every third edge of C.
2. M contains every third edge of C, but is neither true nor false on C (i.e. it contains the second and fifth edges of each literal path).

Let C_{clause} denote the set of clause edges adjacent to C. We will replace $M \cap (C \cup C_{clause})$ with $M_{\mathbf{x}} \cap C$ for violation (1), and $M \cap C$ with $M_{\mathbf{x}} \cap C$ for violation (2). We show in Lemmas 15 and 16 respectively that both these replacements result in valid matchings, and neither increases the size of the maximum matching in $G \setminus M$. Let $\xi_G(M)$ denote the size of the maximum matching in $G \setminus M$, and $\overline{C}$ denote $G \setminus (C \cup C_{clause})$.

▶ **Lemma 15.** *Consider a variable cycle C such that M does not contain every third edge of C. Then replacing M with $M' = (M \cap \overline{C}) \cup (M_{\mathbf{x}} \cap C)$ produces a matching, and does not increase the size of the maximum matching in $G \setminus M$.*

Proof. First, note that M' is a valid matching, since edges in $M \cap \overline{C}$ share no vertices with edges in $M_{\mathbf{x}} \cap C$. To complete the proof, we will show that $\xi_G(M') \leq \xi_G(M)$.

First, we claim that $\xi_C(M') \leq 3j+1 \leq \xi_C(M)$. The proof that $\xi_C(M) \geq 3j+1$ is very similar to the proof of case (3) of Lemma 8 and is not repeated here. The proof that $\xi_{C \cup C_{clause}}(M_{\mathbf{x}}) \leq 3j+1$ is by enumeration over all possible structures of $(C \cup C_{clause}) \cap M_{\mathbf{x}}$. We show two cases below. On the left, the variable x_C corresponding to the clause C appears as three true literals. On the right, it appears as two true literals and a false literal.



The maximum matching in the first graph has size $j/3 = 6$, and matching in the second has size $j/3 + 1 = 7$. It is straightforward to verify the other cases.

The rest of the proofs follows:

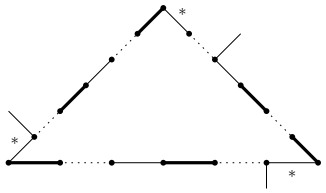
$$\begin{aligned}
\xi_G(M) &\geq \xi_{\overline{C}}(M) + \xi_C(M) && \text{(vertex sets of } \overline{C} \text{ and } C \text{ are disjoint)} \\
&\geq \xi_{\overline{C}}(M) + \xi_{C \cup C_{clause}}(M_{\mathbf{x}}) \\
&\geq \xi_G(M').
\end{aligned}$$

◀

▶ **Lemma 16.** *Consider a variable cycle C such that M contains every third edge of C, but is neither true nor false on C (i.e. M contains the second and fifth edge of each literal path). Then, replacing M with $M' = (M \cap (\overline{C} \cup C_{clause})) \cup (M_{\mathbf{x}} \cap C)$ produces a matching, and does not increase the size of the maximum matching in $G \setminus M$.*

Proof. It is not immediately clear that M' is a valid matching. To show that it is, it is sufficient to show that M does not contain any edges of C_{clause}. This follows from the fact that every edge of C_{clause} is adjacent to an edge of $M \cap C$ since M contains the second and fifth edges of each literal path of C. To complete the proof, we will show that $\xi_G(M') \leq \xi_G(M)$.

First, we claim $\xi_G(M) \geq \xi_{\overline{C} \cup C_{clause}}(M) + \xi_C(M)$. For this, it is enough to show that there is a matching on $C \setminus M$ of size $\xi_C(M)$ that leaves every vertex adjacent to a clause edge unmatched. The proof is by enumeration. We show one case below, it is straightforward to show the others. Edges of the matching on $C \setminus M$ are highlighted, and edges of M are shown as dotted lines.



We can now complete the proof:

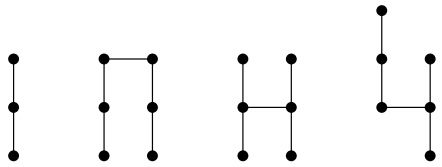
$$\begin{aligned}
\xi_G(M) &\geq \xi_{\overline{C} \cup C_{clause}}(M) + \xi_C(M) \\
&= \xi_{\overline{C} \cup C_{clause}}(M) + \xi_C(M_{\mathbf{x}}) \\
&\qquad\qquad (M \setminus C \text{ and } M_{\mathbf{x}} \setminus C \text{ are the same up to a rotation of cycle } C) \\
&\geq \xi_G(M').
\end{aligned}$$

◀

So, we can always replace M locally with $M_{\mathbf{x}}$ in a way that does not increase the size of the maximum matching in $G \setminus M$. By iteratively performing these replacements, we obtain a matching M which violates only property (b) of the consistency conditions. We now show that if matching M only violates property (b), any clause edge of M can be removed without changing the size of the maximum matching in $G \setminus M$.

▶ **Lemma 17.** *Suppose M is either true or false on all cycle edges, but M contains one or more clause edges. Then, removing the clause edges from M does not increase the size of the maximum matching in $G \setminus M$.*

Proof. $G \setminus M$ consists of a set of connected components, each of which is either a pair of cycle edges, or two pairs of cycle edges connected by a clause edge (either a path, a barbell, or a T as shown below):



Removing a clause edge from M transforms a pair of two-edge paths into a barbell in $G \setminus M$, which does not increase the size of the maximum matching. ◀

5 Randomized Symmetric Matching Interdiction

We now consider a randomized version of the symmetric matching interdiction problem. Rather than selecting matchings deterministically, the interdictor and the optimizer select *random* matchings M and L in G; the goal for the optimizer is to select M so as to minimize the maximum expected size of $L \setminus M$, the maximum taken over all choices of the random matching L. Note that unlike in the standard (deterministic) SMI model, the randomly chosen matchings M and L need not be disjoint. It is easy to see that L can be a (deterministically chosen) best response matching since the support of a randomized best response must consist only of best response matchings. Thus, formally, the randomized SMI problem is to find a probability distribution $\mathcal{M}$ over matchings that minimizes

$$\max_{\text{matching } L} \mathbb{E}[|L \setminus M|], \tag{5}$$

where M is a random matching drawn from $\mathcal{M}$. Any distribution (convex combination) over integral matchings, M, can be viewed as a fractional matching, i.e., as a point in the matching polytope [21]. Let $\bar{x} = \langle x_e \rangle$ denote a point in the matching polytope with x_e the probability (equivalently the fractional weight) of choosing edge e. The expected size of matching L in $G \setminus M$ is $\sum_{e \in L}(1 - x_e)$. Minimizing Eqn. 5 is therefore equivalent to minimizing over the matching polytope, the maximum over all matchings L, $\sum_{e \in L}(1 - x_e)$. This gives rise to the following LP, where, $E(S)$ denotes the set of edges with both endpoints in S, and $\delta(v)$ denotes the set of edges adjacent to v.

$$\begin{aligned} &\min\ y \\ &\text{s.t. } y \geq \sum_{e\in L}(1-x_e) && \forall \text{ matchings } L \in G \\ &\sum_{e\in E(S)} x_e \leq \frac{|S|-1}{2} && \forall S \subset V,\ S \text{ odd} \\ &\sum_{e\in\delta(v)} x_e \leq 1 && \forall v \in G \\ &0 \leq x_e \leq 1 && \forall e \in G \end{aligned} \tag{6}$$

The constraints on the x_e variables ensure that $\bar{x} = \langle x_e \rangle$ lies in the matching polytope [21]. This LP has exponentially many constraints (the first two sets of constraints – matching constraints and odd set constraints), so we give a separation oracle, enabling it to be solved using the ellipsoid algorithm [11]. For the matching constraints let z be the value of the maximum matching in graph G with edge weights of $(1 - x_e)$. If $y \geq z$, then the solution is feasible. Otherwise, the constraint corresponding to the matching with value z is violated. And for the odd set constraints we use the Gomory-Hu based separation oracle given by Padberg and Rao [18].

Thus, by solving the above LP, we can obtain the point, $\bar{x}$, in the matching polytope. However, we need a representation of this point as a convex combination of (or, distribution over) integral matchings in order to determine the (polynomial-time) strategy of the interdictor. Such a representation is guaranteed by the following known lemma (e.g. see [11]). For completeness, we give a proof in Appendix A.

▶ **Lemma 18.** *Let* $\mathbf{x}$ *be a fractional matching.* $\mathbf{x}$ *can be written as the convex combination of polynomially many integral matchings, and these matchings and their weights can be found in polynomial time.*

Finally, we note that there can be a gap of 2 between the optimal randomized and deterministic matchings. Consider a length 2 path. The optimal deterministic matching is either edge, and this matching has value 1 (since it leaves a matching of size one). On the other hand, the randomized matching that assigns probability 1/2 to each edge has value 1/2: Regardless of which edge is chosen to be the second matching, the expected size is 1/2.

6 Other Problems: Acyclic Subgraph Interdiction

As discussed in Section 2, our symmetric interdiction framework can be applied to a diverse set of combinatorial optimization problems. For example, consider any downward closed set system such as acyclic forests, independent vectors in a vector space, and more generally matroids; we can ask how much the interdictor can reduce some measure of the residual set system (e.g., rank) by removing a subset and its elements from the family (we can pose similar questions for families of upward closed sets). We illustrate this idea with the *symmetric acyclic subgraph interdiction problem*. The goal is to determine an acyclic subgraph T of a given graph G so as to minimize the maximum-size acyclic subgraph of $G \setminus T$. Our general framework implies a 2-approximation for this interdiction problem.

▶ **Lemma 19.** *An arbitrary spanning tree on* G *is a 2-approximation to symmetric acyclic subgraph interdiction, and this bound is tight.*

The above lemma follows directly from Corollary 4. We provide a different, more direct proof of this lemma below. This proof enables us to derive an example for which the bound is tight; i.e., there exists a graph G and a spanning tree of G that is at least a 2-approximate solution for G.

Proof of Lemma 19. We start with an alternate proof that an arbitrary spanning tree is at most a 2-approximation. Let T^* be a minimal optimal solution, and T be an arbitrary spanning tree. (If G is not connected, we argue on each component separately.) Note that for $S \subseteq G$, the size of the largest set of acyclic edges in $G \setminus S$ is $n - c$, where c is the number of components in $G \setminus S$.

Let c^* be the number of components in $G \setminus T^*$ and c be the number of components in $G \setminus T$. We consider two cases.

Case 1: $c^* \leq n/2$. Then since $c \geq 1$, $(n - c)/(n - c^*) \leq 2$.

Case 2: $c^* = n/2 + k$. The c^* components of $G \setminus T^*$ form a partition of G, where all of the edges of T^* cross the partition (by minimality of T^*). T must span the components of $G \setminus T^*$, and therefore

$$|T^* \cap T| \geq c^* - 1 = n/2 + k - 1.$$

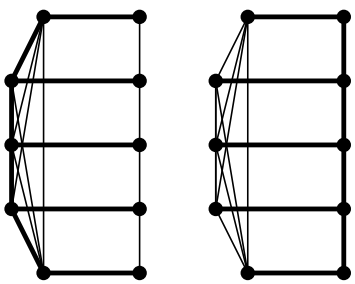
Additionally, $|T^*| \leq n - 1$, so we have

$$|T^* \setminus T| \leq n - 1 - n/2 - k + 1 = n/2 - k.$$

Starting from $G \setminus T^*$, adding back each edge of $T^* \setminus T$ can decrease the number of components by at most one. Therefore, the number of components of $(G \setminus T^*) \cup (T^* \setminus T) = G \setminus (T \cap T^*)$ is at least $(n/2 + k) - (n/2 - k) = 2k$. Therefore, the edges of T partition G into at least $2k$ components, i.e. $c \geq 2k$. Then we have

$$\frac{n - c}{n - c^*} = \frac{n - c}{n/2 - k} \leq \frac{n - 2k}{n/2 - k} = 2. \qquad \blacktriangleleft$$

Next, we show that the bound is tight. Consider a graph with n vertices, such that $n/2$ vertices form a complete graph and $n/2$ vertices form a line. Additionally, there is an edge between the ith vertex on the line, and the ith vertex in the complete graph (vertices in the complete graph have arbitrary order). The optimal spanning tree is the line and all connecting edges, which leaves behind $\frac{n}{2} + 1$ components. A spanning tree that does not contain any edges of the line leaves just 2 components. The construction for $n = 10$ is shown below, with OPT on the right, and a bad solution on the left.



7 Concluding Remarks

In this paper, we have introduced a new symmetric interdiction model, and have focused on symmetric matching interdiction, for which we establish APX-hardness and a polynomial-time achievable 1.5-approximation. The symmetric interdiction model naturally extends to other matroid problems, as illustrated by the acyclic subgraph interdiction problem. Studying symmetric interdiction versions of other combinatorial optimization problems defined on matroids in an interesting direction of future research.

References

1 Douglas S. Altner, Özlem Ergun, and Nelson A. Uhan. The maximum flow network interdiction problem: valid inequalities, integrality gaps, and approximability. *Operations Research Letters*, 38(1):33–38, 2010.

2 Imre Bárány and Roman Karasev. Notes about the carathéodory number. *Discrete & Computational Geometry*, 48(3):783–792, 2012.

3 Piotr Berman and Marek Karpinski. On some tighter inapproximability results. In *International Colloquium on Automata, Languages, and Programming*, pages 200–209. Springer, 1999.

4 Stephen R. Chestnut and Rico Zenklusen. Interdicting structured combinatorial optimization problems with {0, 1}-objectives. *Mathematics of Operations Research*, 2016.

5 Stephen R. Chestnut and Rico Zenklusen. Hardness and approximation for network flow interdiction. *Networks*, 2017.

6 Richard L. Church, Maria P. Scaparra, and Richard S. Middleton. Identifying critical infrastructure: the median and covering facility interdiction problems. *Annals of the Association of American Geographers*, 94(3):491–502, 2004.

7 Eugene Peter Durbin. An interdiction model of highway transportation. *Rand Memorandum*, 1966.

8 Greg N. Frederickson and Roberto Solis-Oba. Increasing the weight of minimum spanning trees. *Journal of Algorithms*, 33(2):244–266, 1999.

9 P. M. Ghare, Douglas C. Montgomery, and W. C. Turner. Optimal interdiction policy for a flow network. *Naval Research Logistics Quarterly*, 18(1):37–45, 1971.

10 Bruce Golden. A problem in network interdiction. *Naval Research Logistics Quarterly*, 25(4):711–713, 1978.

11 Martin Grötschel, Lászlo Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer, second corrected edition edition, 1993.

12 Alpár Jüttner. On budgeted optimization problems. *SIAM Journal on Discrete Mathematics*, 20(4):880–892, 2006.

13 Rafael R. Kamalian and Vahan V. Mkrtchyan. Two polynomial algorithms for special maximum matching constructing in trees. *arXiv preprint arXiv:0707.2295*, 2007.

14 Rafael R. Kamalian and Vahan V. Mkrtchyan. On complexity of special maximum matchings constructing. *Discrete Mathematics*, 308(10):1792–1800, 2008.

15 Leonid Khachiyan, Endre Boros, Konrad Borys, Khaled Elbassioni, Vladimir Gurvich, Gabor Rudolf, and Jihui Zhao. On short paths interdiction problems: Total and node-wise limited interdiction. *Theory of Computing Systems*, 43(2):204–233, 2008.

16 Churlzu Lim and J. Cole Smith. Algorithms for discrete and continuous multicommodity flow network interdiction problems. *IIE Transactions*, 39(1):15–26, 2007.

17 Rui Miao, Rahul Potharaju, Minlan Yu, and Navendu Jain. The Dark menace: Characterizing network-based attacks in the cloud. In *Proceedings of the 2015 ACM Internet Measurement Conference*, 2015.

18 Manfred W. Padberg and M. R. Rao. Odd minimum cut-sets and b-matchings. *Mathematics of Operations Research*, 7(1):67–80, 1982. doi:10.1287/moor.7.1.67.

19 Jörg Rambau. On a generalization of schönhardt's polyhedron. *Combinatorial and computational geometry*, 52:510–516, 2003.

20 Alexander Schrijver. On the history of the transportation and maximum flow problems. *Mathematical Programming*, 91(3):437–445, 2002.

21 Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency. Vol. A. Paths, flows, matchings. Chapters 1–38.* Algorithms and combinatorics. Springer-Verlag, 2003. URL: http://opac.inria.fr/record=b1100334.

22 F Bruce Shepherd and Adrian Vetta. The demand-matching problem. *Mathematics of Operations Research*, 32(3):563–578, 2007.

23 Adrian Vetta and Gwenaël Joret. Reducing the rank of a matroid. *Discrete Mathematics & Theoretical Computer Science*, 17, 2015.

24 Heinrich Von Stackelberg. *Market structure and equilibrium.* Springer Science & Business Media, 2010.

25 R. Kevin Wood. Deterministic network interdiction. *Mathematical and Computer Modelling*, 17(2):1–18, 1993.

26 Rico Zenklusen. Matching interdiction. *Discrete Applied Mathematics*, 158(15):1676–1690, 2010.

27 Rico Zenklusen. Network flow interdiction on planar graphs. *Discrete Applied Mathematics*, 158(13):1441–1455, 2010.

28 Rico Zenklusen. An o(1)-approximation for minimum spanning tree interdiction. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 709–728. IEEE, 2015.

A Representation as a convex combination

We need to show that the representation of fractional matching as a convex combination of integral matchings can be obtained in polynomial-time. We show something more general – namely, that given a feasible point in the polytope defined by a set of linear constraints (even an exponential number in implicit form as a separation oracle [11]) we can find a representation of the point as a convex combination of the vertices of the polytope, in polynomial-time. Our constructive algorithm is folklore, but for completeness, we describe it in its entirety.

Let PT represent the d-dimensional polytope (convex bounded polyhedron) of solutions to $LP = \{C\}$, a set of linear constraints with F_C denoting the face, of dimension at most $d-1$, generated by constraint C. Let $\hat{p} \in PT$ be the given point. The set of constraints may be given in explicit form or implicitly with a bounding ball and a separation oracle [11]. The main idea is to take the ray starting at any vertex v of PT through $\hat{p}$ to its intersection p_i with the face opposite, F_C, then recursively represent p_i as the convex combination of vertices V_C of F_C; now it is an easy matter to see that $\hat{p}$ can be represented as a convex combination of $v \cup V_C$. In fact, it is easy to see, by strengthening the inductive hypothesis, that $\hat{p}$ is the convex combination of at most $d+1$ vertices of PT. All that is left to do is to see that a vertex of a polytope, the point of intersection of a ray with a face of the polytope and the representation of the face as a set of constraints (along with bounding ball and separation oracle, in the general case) can all be computed in polynomial-time. These

operations are easy to compute when the constraints are given in explicit form and we leave it to the reader as an exercise. In the general setting of the separation oracle, a vertex can be computed using the ellipsoid algorithm [11]; the point of intersection of a ray with a face can be computed using binary search; and the polytope restricted to the face F_C can be represented by the same separation oracle augmented with the constraint that C be satisfied with equality, i.e the restricted polytope lies on the hyperplane (the bounding ball stays the same).

Note that the above proof shows that any point in a d-dimensional polytope lies in a simplex formed by at most $d+1$ vertices of the polytope. This gives an alternate proof of Caratheodory's theorem [2]. It also shows that the simplex can be chosen to contain any particular vertex of the polytope. As a small digressional note, it is worth pointing out that the above technique does not extend to showing that every polyhedron (not polytope, polyhedrons may be non-convex) can be triangulated (simpliciated); in fact, though every polyhedron in 2 dimensions (i.e. polygon) can be triangulated this is not true in 3 (or higher) dimensions [19].