Improving Policy-Constrained Kidney Exchange via Pre-Screening

Anonymous Author(s)
Affiliation
Address
email

Abstract

In barter exchanges, participants swap goods with one another without exchanging money; these exchanges are often facilitated by a central clearinghouse, with the goal of maximizing the aggregate quality (or number) of swaps. Barter exchanges are subject to many forms of uncertainty–in participant preferences, the feasibility and quality of various swaps, and so on. Our work is motivated by kidney exchange, a real-world barter market in which patients in need of a kidney transplant swap their willing living donors, in order to find a better match. Modern exchanges include 2- and 3-way swaps, making the kidney exchange *clearing problem* NP-hard. Planned transplants often *fail* for a variety of reasons–if the donor organ is rejected by the recipient's medical team, or if the donor and recipient are found to be medically incompatible. Due to 2- and 3-way swaps, failed transplants can "cascade" through an exchange; one US-based exchange estimated that about 85% of planned transplants failed in 2019. Many optimization-based approaches have been designed to avoid these failures; however most exchanges cannot implement these methods, due to legal and policy constraints. Instead, we consider a setting where exchanges can *query* the preferences of certain donors and recipients–asking whether they would accept a particular transplant. We characterize this as a two-stage decision problem, in which the exchange program (a) queries a small number of transplants before committing to a matching, and (b) constructs a matching according to fixed policy. We show that selecting these edges is a challenging combinatorial problem, which is non-monotonic and non-submodular, in addition to being NP-hard. We propose both a greedy heuristic and a Monte Carlo tree search, which outperforms previous approaches, using experiments on both synthetic data and real kidney exchange data from the United Network for Organ Sharing.

1 Introduction

We consider a multi-stage decision problem in which a decision-maker uses a fixed *policy* to solve a hard (stochastic) problem. Before using the policy, the decision-maker can first *measure* some of the uncertain problem parameters–in a sense, guiding the policy toward a better solution. Our primary motivation is kidney exchange, a process where patients in need of a kidney transplant swap their (willing) living donors, in order to find a better match. Many government-run kidney exchanges match patients and donors using a *matching algorithm* that follows strict policy guidelines [9]; this matching algorithm is often written into law or policy, and is not easily modified. Modern kidney exchanges use both cyclical swaps and chain-like structures (initiated by an unpaired altruistic donor) [25], and identifying the max-size or max-weight set of transplants is both NP- and APX-hard [1, 7].

In kidney exchange–as in many resource allocation settings–information used by the decision-maker is subject to various forms of uncertainty. Here we are primarily concerned with uncertainty in the *feasibility* of potential transplants: if a donor is matched with a potential recipient, will the transplant actually occur? Planned transplants may *fail* for a variety of reasons: for example, medical

Submitted to 34th Conference on Neural Information Processing Systems (NeurIPS 2020). Do not distribute.

testing may reveal that the donor and recipient are incompatible (a *positive crossmatch*); the recipient or their medical team may reject a donor organ in order to wait for a better match; or the donor may decide to donate elsewhere before the exchange is planned. Failed transplants are especially troublesome in kidney exchange, due to the cycle and chain structures used: for example, suppose that a cyclical swap is planned between three patient/donor pairs; if any one of the planned transplants fails, then none of the other transplants in that cycle can occur. Unfortunately, it is quite common for planned transplants to fail. For example, the United Network for Organ Sharing (UNOS[1]) estimates that in FY2019, about 85% of their planned kidney transplants failed [18].

Various matching algorithms have been proposed that aim to mitigate transplant failures (for example, using stochastic optimization [15, 3], robust optimization [22], or conditional value at risk [6]). However, implementing these strategies would require modifying fielded matching algorithms–which in many cases would require changing law or policy. One way to avoid failures without modifying the matching algorithm is to *pre-screen* potential transplants [18, 10, 11], by communicating with the recipients' medical team and possibly using additional medical tests. Pre-screening transplants is costly, as it requires scarce time and resources. Furthermore, there are often many thousand potential transplants in any given exchange; selecting which transplants to screen is not easy.

In this paper we investigate methods for selecting a limited number of transplants to pre-screen, in order to "guide" the fixed matching algorithm to a better outcome. We formalize this as a multistage stochastic optimization problem, and we consider both an *offline* setting (where screenings are selected all at once), and an *online* setting (where screenings are selected sequentially).

Related Work. While kidney exchange is known to be a hard packing problem, several algorithms exist that are scalable in practice, and are used by fielded exchanges [14, 3, 20]. Prior work has addressed potential transplant failures; our model is inspired by Dickerson et al. [15]. Pre-screening potential transplants has also been addressed in prior work ([11, 23], and § 5.1 of [13]), and our model is similar to stochastic matching and stochastic k-set packing [5]. However there are substantial differences between these models and ours: (a) many prior approaches assume that a large number of transplants may be pre-screened [11, 23]–on the order of one for each patient in the exchange; we assume far fewer screenings are possible; (b) prior work often assumes a *query-commit* setting–where successfully pre-screened transplants *must* be matched. Instead we assume that non-screened transplants may also be matched–which more-accurately represents the way that modern exchanges operate; (c) most prior work assumes that transplants that pass pre-screening are guaranteed to result in a transplant. In reality, transplants often fail after pre-screening, a fact reflected in our model.

One of our approaches is based on *Monte Carlo Tree Search* (MCTS), which allows efficient exploration of intractably large decision trees. While MCTS is primarily associated with Markov decision processes and game-playing [12], it has been used successfully for combinatorial optimization [16]. We use a version of MCTS, Upper Confidence Bounds for Trees (UCT), which balances exploration and exploitation by treating each tree node as a multi-armed bandit problem [4, 17].

Our Contributions

1. (§ 2) We formalize the *policy-constrained edge query problem*: where a decision-maker (such as a kidney exchange program) selects a set of potential edges (potential transplants) to pre-screen, prior to constructing a final packing (a set of transplants) using a fixed algorithm. This model generalizes existing models in the literature, as edge failure probabilities depend on whether or not the edge is pre-screened. Further, we allow for context-specific constraints, such as those imposed by public policy or the particular hospital or exchange.

2. (§ 3) We prove that when the decision-maker uses a max-weight packing policy (the most common choice among fielded exchanges), the edge query problem is both non-monotonic and non-submodular in the set of queried edges. Despite these worst-case findings we show that this problem is nearly monotonic for real and synthetic data, and simple algorithms perform quite well. On the other hand, when the decision-maker uses a *failure-aware* (stochastic) packing policy, the edge query problem becomes monotonic under mild assumptions.

3. (§ 4) We conduct numerical experiments on both simulated and real exchange data from the United Network for Organ Sharing (UNOS). We demonstrate that our methods substantially outperform prior approaches and a randomized baseline.

[1]UNOS is the organization tasked with overseeing organ transplantation in the US: `https://unos.org/`.

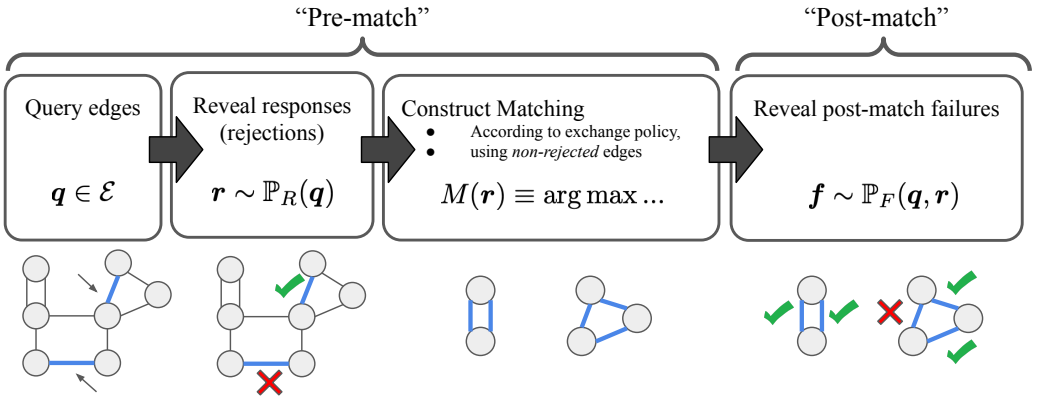


Figure 1: Single-stage edge selection: First, edges are selected to be queried, and responses revealed. Then, a final matching is constructed according to the exchange's matching policy. Finally, the post-match edge failures are revealed.

2 The Policy-Constrained Edge Query Problem

Kidney exchanges are represented by a graph $G = (E, V)$ where vertices V represent (incompatible) patient-donor pairs, and non-directed donors (NDDs) who are willing to donate without receiving a kidney in return. Directed edges $e \in E$ between vertices represent potential transplants from the donor of one vertex to the patient of another. Edge weights represent the "utility" of an edge, and are typically set by exchange policy. Solutions to a kidney exchange problem (henceforth, *matchings*) consist of both directed *cycles* on G containing only patient-donor pairs, and directed *chains* beginning with an NDD and passing through one or more pairs; see Appendix A for an example exchange graph. Each vertex may participate in only one edge in a matching–as each vertex can donate and receive at most one kidney.

Vectors are denoted in bold, and are indexed by either cycles or edges: $\boldsymbol{y}_e$ indicates the element of $\boldsymbol{y}$ corresponding to edge e, and $\boldsymbol{x}_c$ is the element of $\boldsymbol{x}$ corresponding to cycle c. Our notation uses a *cycle-chain* representation for matchings[2]: let $\mathcal{C}$ represent cycles and chains in G, where each cycle and chain corresponds to a list of edges; as is standard in modern exchanges, we assume that cycles and chains are limited in length. Matchings are expressed as a binary vector $\boldsymbol{x} \in \{0,1\}^{|\mathcal{C}|}$, where $\boldsymbol{x}_c = 1$ if cycle/chain c is in the matching, and 0 otherwise. Let $\boldsymbol{w}_c$ be the weight of cycle/chain c (the sum of c's edge weights). Let $\mathcal{M}$ denote the set of *legal matchings*–that is, the set of vertex-disjoint cycles and chains on G, with chains up to length L and cycles up to length C. Cycle length cap C and chain length cap L are set by the each exchange, typically $C = 3$ and $L = 4$. These length limits serve two purposes: (a) longer cycles and chains are risky, in that they are likely to be impacted by edge failure, and (b) policy often requires that all transplants in a cycle or chain are completed *simultaneously*, and most transplant centers can only accommodate a handful of simultaneous transplants. The total weight of a matching is simply the summed weights of all its constituent cycles and chains: $\sum_{c \in \mathcal{C}} \boldsymbol{x}_c \boldsymbol{w}_c$. We denote *sets* of edges using binary vectors, where $\boldsymbol{q} \in \{0,1\}^{|E|}$ represents the set of all edges with $\boldsymbol{q}_e = 1$.

In the remainder of this paper we refer to pre-screening a transplant as *querying an edge*, in order to be consistent with the literature.

Selecting Edge Queries. Our setting consists of two phases (see Figure 1): during *pre-match*, the decision-maker selects edges to query, and each queried edge is either accepted or rejected; then the decision-maker constructs a matching using a fixed policy. During *post-match*, each match edge either fails (no transplant) or succeeds (the transplant proceeds). We consider two version of the pre-match phase: in the *single-stage* version, the decision-maker selects all queries before observing edge responses (accept/reject); in the *multi-stage* version, one edge is selected at a time and responses are observed immediately.

Unlike most prior work, edges in our model may fail during both the pre- and post-match phase. For example, suppose the decision-maker queries an edge from a 60-year-old non-directed donor, to

[2]Our experiments use the position-indexed formulation, which is more compact and equivalent [14].

a 35-year-old recipient; if the recipient or their medical team rejects the elderly donor and decides to wait for a younger donor, this is a pre-match rejection. Instead suppose the edge is not queried, and it is included in the final matching; if medical screening reveals that the patient and donor are incompatible, this is a post-match failure. We refer to pre-match failures as *rejections* and post-match failures as *failures*; however we make no assumption about their cause. We represent potential failures and rejections using binary random variables: $\boldsymbol{r} \in \{0,1\}^{|E|}$ denotes pre-match rejections, where $\boldsymbol{r}_e = 1$ if e is queried and rejected, and 0 otherwise ($\boldsymbol{r}_e = 0$ for all non-queried edges). Similarly $\boldsymbol{f} \in \{0,1\}^{|E|}$ denotes post-match failures, where $\boldsymbol{f}_e = 1$ if edge e fails post-match, and 0 otherwise. We assume that the distribution of rejections $\boldsymbol{r} \sim \mathbb{P}_R(\boldsymbol{q})$ is known, and depends on $\boldsymbol{q}$; we assume the distribution of failures $\boldsymbol{f} \sim \mathbb{P}_F(\boldsymbol{q}, \boldsymbol{r})$ is known, and depends on both $\boldsymbol{q}$ and $\boldsymbol{r}$.

Rejections and failures impact the matching through the *weight* of each cycle and chain. If any cycle edge fails, then *no* transplants in the cycle can proceed; if a chain edge fails, than all edges *following* it cannot proceed.[3] Suppose we observe failures $\boldsymbol{f}$; the *final matching weight* of c is

$$F(c, \boldsymbol{y}) \equiv \begin{cases} \sum_{e \in c} w_e & \text{if } \sum_{e \in c} \boldsymbol{y}_e = 0 \\ 0 & \text{if } c \text{ is a cycle and } \sum_{e \in c} \boldsymbol{y}_e > 0 \\ \sum_{e \in c'} w_e & \text{if } c \text{ is a chain, where } c' \text{ includes all edges up to the first failed edge.} \end{cases}$$

Thus the *post-match expected weight* of matching $\boldsymbol{x}$, due to both rejections $\boldsymbol{r}$ and failures $\boldsymbol{f}$, is

$$W(\boldsymbol{x}; \boldsymbol{q}, \boldsymbol{r}) \equiv \mathop{\mathbb{E}}_{\boldsymbol{f} \sim \mathbb{P}_F(\boldsymbol{q}, \boldsymbol{r})} \left[\sum_{c \in \mathcal{C}} \boldsymbol{x}_c \, F(c, \boldsymbol{r} + \boldsymbol{f}) \right].$$

Matching Policy In this paper we assume that the final matching is constructed using a fixed matching policy, which uses only *non-rejected* edges; we denote this policy by $M(\boldsymbol{r})$. We focus primarily on the *max-weight* policy $M^{\texttt{MAX}}(\cdot)$, which is used by most fielded exchanges, and the *failure-aware* policy $M^{\texttt{FA}}(\cdot)$, which maximizes the expected post-match weight [15]:

$$M^{\texttt{MAX}}(\boldsymbol{r}) \in \mathop{\arg\max}_{\boldsymbol{x} \in \mathcal{M}} \sum_{c \in \mathcal{C}} \boldsymbol{x}_c \, F(c, \boldsymbol{r}) \,, \qquad M^{\texttt{FA}}(\boldsymbol{r}) \in \mathop{\arg\max}_{\boldsymbol{x} \in \mathcal{M}(\boldsymbol{r})} \mathop{\mathbb{E}}_{\boldsymbol{f} \sim \mathbb{P}_F(\boldsymbol{q}, \boldsymbol{r})} \left[\sum_{c \in \mathcal{C}} \boldsymbol{x}_c \, F(c, \boldsymbol{r} + \boldsymbol{f}) \right].$$

Evaluating this policy requires solving a kidney exchange clearing problem, which is NP-hard [1]. However, state-of-the-art method can solve realistic kidney exchange clearing problems in fractions of a second (e.g., our experiments use the PICEF method of Dickerson et al. [14]); thus, throughout this paper we treat this policy as a low- or no-cost oracle.

Next we formalize the *edge selection problem*–the main focus of this paper. We denote by $\mathcal{E}$ the set of "legal" edge subsets, subject to exchange-specific constraints; we assume that $\mathcal{E}$ is a matroid with ground set E. For example, the decision-maker may limit the number of queries issued to any one medical team (vertex in G) or transplant center (group of vertices). We aim to select an edge set $\boldsymbol{q} \in \mathcal{E}$ which maximizes the *expected weight* of the final matching. These edges are selected using only the distribution of future rejections and failures; we take a *stochastic optimization* approach, maximizing the expected outcome over this uncertainty.

Single-Stage Setting. The single-stage policy-constrained edge selection problem (henceforth, the *edge selection problem*) is expressed as

$$\max_{\boldsymbol{q} \in \mathcal{E}} V^S(\boldsymbol{q}) \,, \qquad \text{with} \quad V^S(\boldsymbol{q}) \equiv \mathop{\mathbb{E}}_{\boldsymbol{r} \sim \mathbb{P}_R(\boldsymbol{q})} \left[\, W\left(M(\boldsymbol{r}); \boldsymbol{q}, \boldsymbol{r}\right) \, \right], \tag{1}$$

where, $M(\boldsymbol{r})$ denotes the matching policy after observing rejections $\boldsymbol{r}$, and $W(\boldsymbol{x}; \boldsymbol{q}, \boldsymbol{r})$ denotes the post-match expected weight of matching $\boldsymbol{x}$. Exact evaluation of $V^S(\boldsymbol{q})$ is often intractable, as the support of $\mathbb{P}_R(\boldsymbol{q})$ grows exponentially in $|\boldsymbol{q}|$. In experiments we approximate $V^S(\boldsymbol{q})$ using sampling, and these approximations converge for a moderate number of samples (see Appendix B).

Multistage Setting. In the multi-stage setting, edge rejections are observed immediately after each edge is queried. The multi-stage problem is expressed as

$$\max_{\boldsymbol{q}^1 \in \mathcal{E}_1} \mathop{\mathbb{E}}_{\boldsymbol{r}^1 \sim \mathbb{P}_R(\boldsymbol{q}^1)} \left[\max_{\boldsymbol{q}^2 \in \mathcal{E}_1} \mathop{\mathbb{E}}_{\boldsymbol{r}^2 \sim \mathbb{P}_R(\boldsymbol{q}^2)} \left[\dots \max_{\boldsymbol{q}^K \in \mathcal{E}_1} \mathop{\mathbb{E}}_{\boldsymbol{r}^K \sim \mathbb{P}_R(\boldsymbol{q}^K)} \left[\, W\left(M(\boldsymbol{r}); \boldsymbol{q}, \boldsymbol{r}\right) \, \right] \right] \dots \right], \tag{2}$$

[3] This assumes that chains can be *partially* executed: for example, suppose that the 4^{th} edge in a 10-edge chain fails; the first three edges can still be matched, and the post-failure chain weight sums only these three edges. Not all fielded exchanges use this policy: some exchanges cancel the entire chain if one of its edges fails.

where $\boldsymbol{q} \equiv \sum_{i=1}^{K} \boldsymbol{q}^i$ denotes all queried edges, $\boldsymbol{r} \equiv \sum_{i=1}^{K} \boldsymbol{r}^i$ denotes all rejections, and $\mathcal{E}_1 \subseteq \mathcal{E}$ be denotes the legal edge subsets containing only one edge. First, we observe that Problems 1 and 2 require evaluating a matching policy $M(\boldsymbol{r})$. In the case of kidney exchange, evaluating both the max-weight policy $M^{\texttt{MAX}}(\cdot)$ and the failure-aware policy $M^{\texttt{FA}}(\cdot)$ require solving NP-hard problems; thus Problems 1 and 2 are at least NP-hard as well.

However, regardless how difficult the matching policy is, the question remains whether *edge selection* is is hard. We observe that while these problems are difficult in principle, experiments (§ 4) show that they are easy in practice. Proofs of the following propositions can be found in Appendix D.

Proposition 2.1. *With matching policy $M^{\texttt{MAX}}(\cdot)$, the objective of Problem 1 is non-monotonic in the number of queried edges, even with independent edge distributions.*

In other words, querying additional edges can sometimes lead to a *worse* outcome. This is somewhat counter-intuitive; one might think that providing additional information to the matching policy would strictly improve the outcome. This is a worst-case result–and in fact our experiments demonstrate that querying edges almost always leads to a better final matching weight.

Proposition 2.2. *With matching policy $M^{\texttt{MAX}}(\cdot)$, the objective of Problem 1 is non-submodular in the set of queried edges.*

In other words, certain edges are *complementary* to each other–and querying complementary edges simultaneously can yield a greater improvement than querying them separately. Taken together, these propositions indicate that single-stage edge selection with matching policy $M^{\texttt{MAX}}(\cdot)$ is a challenging combinatorial optimization problem. On the other hand, using the failure-aware matching policy $M^{\texttt{FA}}(\cdot)$ allows us to avoid some of these issues under mild assumptions.

Assumption 2.3. *Let $\boldsymbol{q}, \boldsymbol{r} \in \{0,1\}^{|E|}$ denote initial edge queries and responses. Let $\boldsymbol{q}'$ be* additional *edges, such that $\boldsymbol{q} + \boldsymbol{q}' \in \{0,1\}^{|E|}$ denotes an augmented edge set; let $\boldsymbol{r}' \in \{0,1\}^{|E|}$ denote the responses to edges $\boldsymbol{q}'$ only. We assume that for any such $\boldsymbol{q}$, $\boldsymbol{r}$, and $\boldsymbol{q}'$,*

$$\mathbb{E}\left[\boldsymbol{r} + \boldsymbol{f} \mid \boldsymbol{q}, \boldsymbol{r}\right] \geq \mathbb{E}\left[\boldsymbol{r} + \boldsymbol{r}' + \boldsymbol{f} \mid \boldsymbol{q} + \boldsymbol{q}', \boldsymbol{r}\right] \ .$$

Intuitively, Assumption 2.3 excludes distributions where queries arbitrarily increase edge failure or rejection. For example, Assumption 2.3 disallows the following distribution: suppose all edges are independent; all queried edges are accepted ($P(\boldsymbol{r}_e = 1 \mid \boldsymbol{q}) = 0$ for all $\boldsymbol{q}$), all accepted edges have failure probability 0.5 ($P(\boldsymbol{f}_e = 1 \mid \boldsymbol{q}_e = 1, \boldsymbol{r}_e = 0) = 0.5$), and all non-queried edges have failure probability 0.1 ($P(\boldsymbol{f}_e = 1 \mid \boldsymbol{q}_e = \boldsymbol{r}_e = 0) = 0.1$). In this case, if an edge is not queried, then it has overall rejection or failure probability 0.1 (i.e., $\mathbb{E}[\boldsymbol{r}_e + \boldsymbol{f}_e \mid \boldsymbol{q}, \boldsymbol{r}] = 0.1$ with $\boldsymbol{q}_e = 0$); if this edge is queried, then it has rejection or failure probability 0.5 (i.e., $\mathbb{E}[\boldsymbol{r}_e + \boldsymbol{r}'_e + \boldsymbol{f}_e \mid \boldsymbol{q} + \boldsymbol{q}', \boldsymbol{r}] = 0.5$ with $\boldsymbol{q}'_e = 1$).

We also assume that edge failures are *independent*.

Definition 2.4 (Edge Independence)**.** *Two edges $e, e' \in E$ are* independent *if (a) their rejection distributions are conditionally independent, given whether or not they were queried:*

$$\boldsymbol{r}_e \perp\!\!\!\perp \boldsymbol{r}_{e'} \mid \boldsymbol{q}_e \quad \textit{and} \quad \boldsymbol{r}_e \perp\!\!\!\perp \boldsymbol{r}_{e'} \mid \boldsymbol{q}_{e'}$$

and (b) their failure distributions are conditionally independent, given whether or not they were queried and rejected:

$$\boldsymbol{f}_e \perp\!\!\!\perp \boldsymbol{f}_{e'} \mid \boldsymbol{q}_e, \boldsymbol{r}_e \quad \textit{and} \quad \boldsymbol{f}_e \perp\!\!\!\perp \boldsymbol{f}_{e'} \mid \boldsymbol{q}_{e'}, \boldsymbol{r}_{e'} \ .$$

Proposition 2.5. *If edges are independent and Assumption 2.3 holds, then with a failure-aware matching policy the objective of Problem 1 is monotonic in the set of queried edges.*

While Propositions 2.1 and 2.2 state that single-stage edge selection is challenging in the worst case, our computational results suggest that these problems are often easier on realistic exchanges.

2.1 Using the Max-Weight Matching Policy as a Baseline

It might seem as though our edge pre-screening procedure is simply compensating for a flawed matching policy ($M^{\texttt{MAX}}(\cdot)$). If matching policy $M^{\texttt{MAX}}(\cdot)$ performs poorly in practice, then why not use $M^{\texttt{FA}}(\cdot)$? Indeed, $M^{\texttt{FA}}(\cdot)$ can directly account for edge failure, and Proposition 2.5 states that the edge selection problem is in fact "easy" when using this policy. However this assumes that the edge failure distribution is accurately known. The failure-aware matching policy $M^{\texttt{FA}}(\cdot)$ is very sensitive to the specified edge failure distribution, and if the assumed distribution is *incorrect* then this policy can perform very poorly. This is in fact a reason that $M^{\texttt{FA}}(\cdot)$ is not used in practice: edge failure

distributions are not accurately known, and exchange programs are hesitant—with good reason—to guide their matching policy with a noisy estimation of edge failure.

In our edge pre-screening setting, the assumed edge distribution may also be noisy, however this does not directly affect the matching algorithm. In the worst case, an incorrect edge failure distribution will lead us to pre-screen edges that do not provide useful information to the exchange (this is the status quo). This is in stark contrast to $M^{\texttt{FA}}(\cdot)$: in the worst case, a bad estimation of the edge failure distribution will cause $M^{\texttt{FA}}(\cdot)$ to match risky cycles and chains, potentially decreasing the number of transplants.

2.2 A Note on Match Run Frequency

In the real world kidney exchange is a *dynamic* process: patients are constantly entering and leaving the pool, and participants are matched every few weeks. One way to deal with edge failure uncertainty is to match *more frequently*. For example if we match patients and donors once every month, then each failed edge adds one month to the waiting time for every patient who relied on the failed transplant. If we match participants every few hours, this increase in waiting time is far less severe.

However there are good reasons to match *infrequently*. First, exchanges benefit with the addition of more patients and donors—new edges "thicken" the compatibility graph, enabling more cycles and chains. Second, each transplant center participating in exchange needs a transplant coordinator to manage individual patients and donors. Many small hospitals do not have a full-time staff for organ exchange, and they cannot deal with frequent match offers. For these reasons, most exchanges match patients and donors very infrequently: the UK national kidney exchange matches patients and donors every quarter[4] (once every three months); the Canadian national exchange matches participants once every four months[5]; UNOS currently matches patients once every week. [6]

3 Solving the Policy-Constrained Edge Query Problem

First we propose an exhaustive tree search which returns an optimal solution to Problem 1 given enough time. Building on this, we propose a Monte Carlo Tree Search algorithm and a simple greedy algorithm. Our multi-stage approaches are very similar to these, and can be found in Appendix E.

Our optimal exhaustive search uses a *search tree* where each tree node corresponds to an edge subset in $\boldsymbol{q} \in \mathcal{E}$. The children of node $\boldsymbol{q}$ correspond to any $\boldsymbol{q}' \in \mathcal{E}$ which are equivalent to the parent $\boldsymbol{q}$, but include one additional edge: $C(\boldsymbol{q}) \equiv \{(\boldsymbol{q} + \boldsymbol{q}') \; \forall \boldsymbol{q}' \in \mathcal{E} : |\boldsymbol{q}'| = 1 \;\mid\; (\boldsymbol{q} + \boldsymbol{q}') \in \mathcal{E}\}$. We say that edge sets (or tree nodes) containing L edges are on the L^{th} *level* of the tree. We refer to nodes with no children as *leaf nodes*. Unlike other tree search settings, the optimal solution to Problem 1 may be at *any* node of the tree, not only leaf nodes; this is a consequence of non-monotonicity (see Proposition 2.1). The tree defined by root node $\boldsymbol{q} = \boldsymbol{0}$ and child function $C(\boldsymbol{q})$ contains all legal edge subsets in $\mathcal{E}$, when $\mathcal{E}$ is a matroid. Thus, *any* exhaustive tree search algorithm (such as depth-first search) will identify an optimal solution, given enough time and memory.

Of course exhaustive search is only tractable if $\mathcal{E}$ is small. Consider the class of *budgeted* edge sets $\mathcal{E}(\Gamma)$ used in our experiments: $\mathcal{E}(\Gamma) \equiv \{\boldsymbol{q} \in \{0,1\}^{|E|} \;\mid\; |\boldsymbol{q}| \leq \Gamma\}$ (edge sets containing at most Γ edges). The number of edge sets in $\mathcal{E}(\Gamma)$ grows roughly exponentially in Γ and $|E|$, and is impossible to enumerate even for small graphs. Suppose a graph has 50 edges and we have an edge budget of five: there are over two million edge sets in $\mathcal{E}(5)$. Even small exchange graphs can have thousands of edges, and thus $\mathcal{E}(\Gamma)$ cannot be enumerated. Therefore, we propose search-based approach.

Monte Carlo Tree Search for Edge Selection (`MCTS`): We propose a tree-search algorithm for single-stage edge selection, `MCTS`, based on Monte Carlo Tree Search (MCTS), with the Upper Confidence for Trees (UCT) algorithm [17]. Our approach keeps track of a *value* (the objective value of Problem 1) and a UCB value estimate for each node, and these values are updated during sampling. The formula used to estimate a node's UCB value is

$$\frac{\frac{U}{N} - V^{min}}{V^{max} - V^{min}} + \sqrt{N^P/N}$$

[4] `https://nhsbtdbe.blob.core.windows.net/umbraco-assets-corp/24443/pol274-4.pdf`

[5] `https://profedu.blood.ca/en/organs-and-tissues/programs-and-services/kidney-paired-donation-kpd-program`

[6] `https://unos.org/wp-content/uploads/unos/KPD_emanual_how-matching-works.pdf`

where U is the "UCB value estimate" calculated by `MCTS`, N is the number of visits to the node, N^P is the number of visits to the node's parent, and V^{max} and V^{min} are the largest and smallest node values encountered during search.

When the set of tree nodes is too large to enumerate UCT can use a huge amount of memory–by storing values for each visited node. To limit both memory use and runtime, we incrementally search the tree from a temporary root node. Beginning from the root (the empty edge set), we use UCB sampling on the next L levels of nodes–where L is a small fixed integer. After a fixed time limit, sampling stops and we set the *new* root node to the current root's best child according to its UCB estimate–using the method of [17]. This process repeats until we reach the final level of the search tree. Algorithm 1 gives a pseudocode description of `MCTS`, which uses Algorithm 2 as a submethod. While often successful, MCTS requires extensive training and parameter tuning. As a simpler alternative, we propose a greedy algorithm.

Single-Stage Greedy Algorithm: `Greedy`. Like `MCTS`, our greedy algorithm (`Greedy`) begins with the empty edge set as the root node, and iteratively searches deeper levels of the tree. However unlike `MCTS`, `Greedy` simply selects the child node with the greatest objective value in Problem 1–that is, *greedily* improving the objective value; see Appendix E for a pseudocode description.

ALGORITHM 1: `MCTS`: Tree Search for Single-Stage Edge Selection

(input) K: maximum size of any legal edge set
(input) T: time limit per level
(input) L: number of look-ahead levels

$\boldsymbol{q}^R \leftarrow \mathbf{0}$ root node (no edges)
$\boldsymbol{q}^* \leftarrow \mathbf{0}$ the best visited node
$V^* \leftarrow$ objective value of $\boldsymbol{q}^*$
for $N = 1, \ldots, K$ **do**
  $M \leftarrow \min\{N + L, K\}$
  $Q \leftarrow$ all nodes in levels N to M
  $U[\boldsymbol{q}] \leftarrow 0 \ \forall \boldsymbol{q} \in Q$ UCB value estimate
  $V[\boldsymbol{q}] \leftarrow 0 \ \forall \boldsymbol{q} \in Q$ objective value
  $N[\boldsymbol{q}] \leftarrow 0 \ \forall \boldsymbol{q} \in Q$ number of visits
  while *less than time T has passed* **do**
    `Sample`($\boldsymbol{q}^R$, M)
  $\boldsymbol{q}^R \leftarrow \arg\max_{\boldsymbol{q} \in C(\boldsymbol{q}^R)} U[\boldsymbol{q}]$
  Delete $U[\cdot]$, $V[\cdot]$, and $N[\cdot]$
return $\boldsymbol{q}^*$

ALGORITHM 2: `Sample`: Sampling function used by `MCTS`

(input) $\boldsymbol{q}$, M

$N[\boldsymbol{q}] \leftarrow N[\boldsymbol{q}] + 1$
$V[\boldsymbol{q}] \leftarrow$ objective of edge set $\boldsymbol{q}$ in Problem 1
if $V[\boldsymbol{q}] > V^*$ **then**
  $\boldsymbol{q}^* \leftarrow \boldsymbol{q}$, $V^* \leftarrow V[\boldsymbol{q}]$
if $\boldsymbol{q}$ *has no children* **then**
  return $V[\boldsymbol{q}]$
if $\boldsymbol{q}$ *has children* **then**
  if $|\boldsymbol{q}| < M$ **then**
    $\boldsymbol{q}' \leftarrow \arg\max_{\boldsymbol{q} \in C(\boldsymbol{q}^R)} U[\boldsymbol{q}] + \text{UCB}[\boldsymbol{q}]$
    $U[\boldsymbol{q}] \leftarrow U[\boldsymbol{q}] +$ `Sample`($\boldsymbol{q}'$, M)
  else
    $\boldsymbol{q}' \leftarrow$ a random descendent of $\boldsymbol{q}$ at any level
    $V' \leftarrow$ objective value of $\boldsymbol{q}'$ in Problem 1
    if $V' > V^*$ **then**
      $\boldsymbol{q}^* \leftarrow \boldsymbol{q}'$, $V^* \leftarrow V'$
    $U[\boldsymbol{q}] \leftarrow U[\boldsymbol{q}] + V'$

Runtime. Our methods rely on an "oracle" to solve the NP-hard kidney exchange matching problem; while state-of-the-art methods solve real-sized instances of these problems in fractions of a second, there is no guaranteed bound for absolute runtime. Instead, we can report the *number of calls* to this oracle for each method as a measure of complexity. Both benchmark methods (max-weight matching and failure-aware [15]) as well as `IIAB` [11] use exactly one oracle call; i.e., they are $O(1)$. Both `Greedy` and `MCTS` use a fixed number of samples (M) to evaluate the objective of an edge set. `Greedy` evaluates the objective of an edge set exactly Γ times; thus, `Greedy` is $O(M \cdot \Gamma)$. Finally, `MCTS` can in theory visit all potential edge sets of size at most Γ (i.e., an exhaustive search), which is $O(M \cdot \sum_{\gamma=1}^{\Gamma} \binom{|E|}{\gamma})$. Since this version of `MCTS` is intractable in both runtime and memory, Algorithm 1 imposes reasonable limits on our implementation.

4 Computational Experiments

We conduct a series of computational experiments using both synthetic data, and real kidney exchange data from UNOS; all code for these experiments is available online.[7] In these experiments, "legal" edge sets are the budgeted edge sets defined as $\mathcal{E}(\Gamma) \equiv \{\boldsymbol{q} \in \{0,1\}^{|E|} \mid |\boldsymbol{q}| \leq \Gamma\}$. In Sections 4.2 and 4.3 we present results in the single- and multi-stage edge selection settings, respectively. We use both real data and synthetic data for our experiments.

[7] Link removed for review.

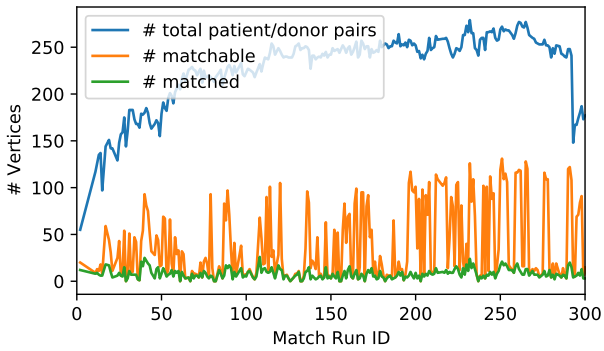


Figure 2: Number of patient donor pairs in each exchange, the number of matchable vertices (who can participate in a legal cycle or chain), and the number of vertices matched by $M^{\texttt{MAX}}(\cdot)$ in simulation.

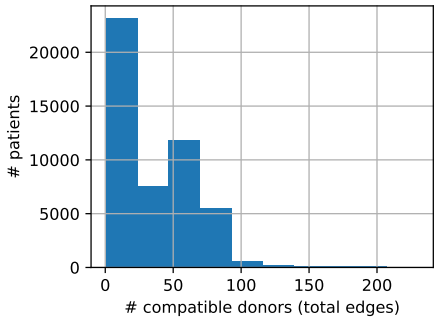


Figure 3: Histogram of the number of *compatible* donors (edges) for each recipient, over all UNOS graphs.

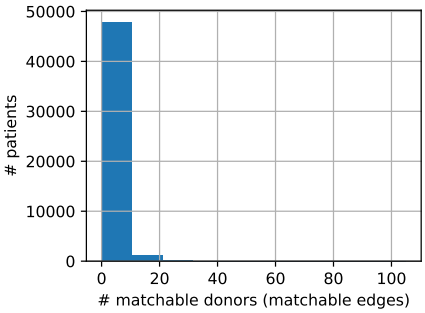


Figure 4: Histogram of the number of *matchable* edges for each recipient, over all UNOS graphs.

4.1 Data

Real Data. We use exchange graphs from the United Network for Organ Sharing (UNOS), representing UNOS match runs between 2010 and 2019. Some of these exchange graphs only have the trivial matching (no cycles or chains), or they have only one non-trivial matching. We ignore these graphs because the matching policy is a "constant" function (to return the one feasible matching) and edge queries cannot change the outcome. Removing these, we are left with 324 UNOS exchange graphs.

These exchange graphs are relatively small and sparse: most graphs have fewer than 250 vertices, and fewer of half of these can be matched via a legal cycle or chain (with cycle cap $C = 3$ and chain cap $L = 4$); we refer to these vertices as *matchable*. The number of NDDs (who can initiate chains) is extremely small: most exchanges have zero or 1 NDD. Figure 2 shows the number of vertices, the number of matchable vertices, and the number matched by $M^{\texttt{MAX}}(\cdot)$ (in simulation). These graphs are also sparse: Figure 3 shows the number of edges (compatible donors) for each patient over all UNOS graphs. The median number of edges per patients is 29, and 14% have only one edge. Furthermore, while many patients have multiple *compatible* edges, very few of these edges are *matchable* (see Figure 4): 72% of all patients cannot be matched via a legal cycle or chain; of the *matchable* patients (with one or more matchable donor), the median number of matchable donors is 3.

Synthetic Data. We generate random kidney exchange graphs based on directed Erdős-Rényi graphs defined using parameters N and p: let V be a fixed set of N vertices; for each pair of vertices (V_1, V_2) there is an edge from V_1 to V_2 with probability p, and an edge from V_2 to V_1 with probability p (independent of the edge from V_1 to V_2). Any vertices with no incoming edges are considered NDDs. In simulations we generate graphs which are smaller and more-sparse than the average UNOS graph ($N \leq 100$ and $p = 0.01$). These are meant to represent a toy-model of kidney exchange, rather than a realistic imitation of exchange graphs.

Table 1: Left: Optimality gap for Greedy, over 100 random graphs with $p = 0.01$ and various N, with edge budget $\Gamma = 3$; bottom row shows the maximum value of %OPT over all graphs. Right: Single-stage results on UNOS graphs using the variable IIAB edge budget (top rows), and the failure-aware method (bottom row). Columns P_X indicates the X^{th} percentile of $\Delta^{\texttt{MAX}}$ over all UNOS graphs.

	Num. Graphs (out of 100)		
%OPT	$N = 50$	$N = 75$	$N = 100$
$[0, 0.1]$	93	93	90
$(0.1, 1]$	5	4	9
$(1, 2]$	1	3	1
$(2, 100]$	1	0	0
Max %OPT	2.8	1.5	1.0

	Simple edge dist.			*KPD* edge dist.		
Method	P_{10}	P_{50}	P_{90}	P_{10}	P_{50}	P_{90}
MCTS	0.40	0.67	1.11	0.05	0.45	3.44
Greedy	0.47	0.64	1.00	0.02	0.47	3.44
Random	0.00	0.10	0.46	−0.11	0.00	0.63
IIAB	0.21	0.45	0.89	−0.27	0.12	2.24
Fail-Aware	0.00	0.09	0.23	$-0.27^{\dagger}$	$0.00^{\dagger}$	$2.17^{\dagger}$

In these experiments edge rejections and failures are independently distributed for each edge e; let P_R be the rejection probability, P_Q is the post-match success probability if e is queried/accepted, and P_N is the success probability if e is not queried. To simulate edge rejections and failures we use two synthetic edge distributions: *Simple* and *KPD*. In the *Simple* distribution, $P_R = 0.5$, $P_Q = 1$, and $P_N = 0.5$ for all edges. The *KPD* distribution is inspired by the fielded exchange setting from which we draw our real underlying compatibility graphs. According to UNOS, about 34% of all edges are rejected by a donor or recipient pre-match [18]; we draw P_R uniformly from $U(0.25, 0.43)$ for each edge. Edges ending in highly-sensitized patients (who are often less healthy and more likely to be incompatible) are considered high-risk; for these edges we draw P_Q from $U(0.2, 0.5)$ and P_N from $U(0.0, 0.2)$. For other edges we draw P_Q from $U(0.9, 1.0)$ and P_N from $U(0.8, 0.9)$.

4.2 Single-Stage Edge Selection Experiments

In this section we compare against the baseline of a max-weight matching *without* edge queries (using policy $M^{\texttt{MAX}}(\cdot)$). Many fielded kidney exchanges use a variant of this matching policy, so by comparing against this baseline we are illustrating the impact of edge queries on the state-of-the-art matching policies used in many real exchanges. Let V_X be the objective[8] of Problem 1 achieved by method X, we calculate $\Delta^{\texttt{MAX}}$ (the relative difference from baseline) as $\Delta^{\texttt{MAX}} \equiv (V_X - V^S(\mathbf{0}))/V^S(\mathbf{0})$. A value of $\Delta^{\texttt{MAX}} = 0$ means that method X did not improve over the baseline, a value of $\Delta^{\texttt{MAX}} = 1$ means that X achieved an objective 100% greater than the baseline, and so on. Furthermore a value of $\Delta^{\texttt{MAX}} > 0$ means that method X *increases* the objective by querying edges, while $\Delta^{\texttt{MAX}} < 0$ means that method X decreases the objective by querying edges.

Result: Greedy is essentially Optimal with small random graphs. First we investigate the *difficulty* of edge selection. Using random graphs, we compare Greedy to the *optimal* solution to Problem 1, found by exhaustive search (OPT). We generate three sets of 100 random graphs with $N = 50$, 75, and 100 vertices, and each with $p = 0.01$. For all graphs we run both OPT and Greedy with edge budget 3; we calculate the *optimality gap* of Greedy as $\%\texttt{OPT} \equiv 100 \times (V_{\texttt{OPT}} - V_{\texttt{Greedy}})/V_{\texttt{OPT}}$, where V_X denotes the objective achieved by method X. ($V_{\texttt{OPT}} > 0$ in all graphs used in these experiments.) If $\%\texttt{OPT} = 0$ then Greedy returns an optimal solution, and $\%\texttt{OPT} > 0$ means that Greedy is not optimal. Table 1 (left) shows the number of random graphs binned by %OPT, as well as the maximum %OPT over all graphs. For each N, Greedy returns an optimal solution for at least 90 of the 100 graphs; the *maximum* %OPT over all graphs is 2.8.

In other words, Greedy always returns an *optimal* or nearly-optimal set of edges to query for small random graphs. This is somewhat unexpected, since the edge selection problem is both non-monotone and non-submodular (see Section 2).

Result: Greedy is essentially monotonic with UNOS graphs. We test Greedy on real UNOS graphs, using maximum budget 100. Figure 5a shows the median $\Delta^{\texttt{MAX}}$ over all UNOS graphs, with shading between the 10^{th} and 90^{th} percentiles. Larger edge budgets almost never decrease the objective achieved by Greedy, and Greedy *never* produces a worse outcome than the baseline.

[8] All objective values are estimated using up to 1000 sampled rejection scenarios (see Appendix B), as it is intractable to evaluate the exact objective of large edge sets.

[†] We use an approximation of Fail-Aware for the *KPD* dist.; *true* Fail-Aware should always have $\Delta^{\texttt{MAX}} > 0$.

Thus–in our setting–single-stage edge selection is effectively monotonic in our setting, and Greedy is an effective method.

Result: MCTS and Greedy are nearly equivalent with UNOS graphs. We compare all methods on UNOS graphs, using smaller, more-realistic edge budgets from 1 to 10. For MCTS we use a 1-hour time limit per edge (Γ hours total). Figures 5b and 5d compare $\Delta^{\texttt{MAX}}$ for MCTS, Greedy, and random edge selection, for the *Simple* and *KPD* edge distributions, respectively. We draw two conclusions from these results: (1) MCTS and Greedy produce almost identical results, further suggesting that Greedy is nearly optimal in our setting; (2) in our setting, edge selection is *effectively* monotonic, as $\Delta^{\texttt{MAX}}$ almost never decreases. However Figure 5d gives an example of non-monotonicity for both Greedy and Random: in some cases, querying edges can lead to a *worse* outcome than querying no edges.

Result: Both MCTS and Greedy outperform benchmarks from the literature. We also compare against two state-of-the-art approaches: the edge selection approach of [11] (IIAB), which uses a *variable* edge budget that depends on the graph structure; and and the failure-aware matching policy of [15] (Fail-Aware[9]), which does not query edges To our knowledge, IIAB is the only edge selection method in the literature. We compare against the Fail-Aware method because it is a state-of-the-art kidney exchange matching policy which aims to maximize the expected matching weight, under a similar edge failure model to ours; we compare against this approach to further illustrate the utility of querying edges.

Table 1 (right) shows a comparison of all edge-selection methods–each using the variable edge budget of IIAB; the bottom row shows results for Fail-Aware. Both MCTS and Greedy achieve greater $\Delta^{\texttt{MAX}}$ (in distribution) than both benchmark methods. This is expected in both cases: IIAB uses a heuristic to select edges to query, which does not consider the final matching weight—the objective of our edge selection problem; on the other hand, both MCTS and Greedy are designed to maximize this objective. We do not expect Fail-Aware to out-perform any edge selection methods, since Fail-Aware does not have access to information revealed after edge queries.

It is notable that Greedy performs better than MCTS (in distribution). This likely means that MCTS is *under-trained*—that the time and memory limits used in our implementation are too restrictive; alternatively, this indicates that Greedy is simply very effective in our setting.

4.3 Multi-Stage Edge Selection Experiments on UNOS Graphs

We run initial multi-stage edge selection experiments on all UNOS graphs with the *Simple* edge distribution. For each graph we test our multi-stage variants of MCTS and Greedy, and compare with a baseline of random edge selection; as before, MCTS uses a 1-hour training time per level. It is substantially harder to evaluate the multi-stage objective, as each edge edge-selection method changes depending on rejections observed in prior stages. Similarly, the MCTS search tree is orders of magnitude larger in the multi-stage setting: each node in tree corresponds to both an edge set *and* a rejection scenario (see Appendix E).

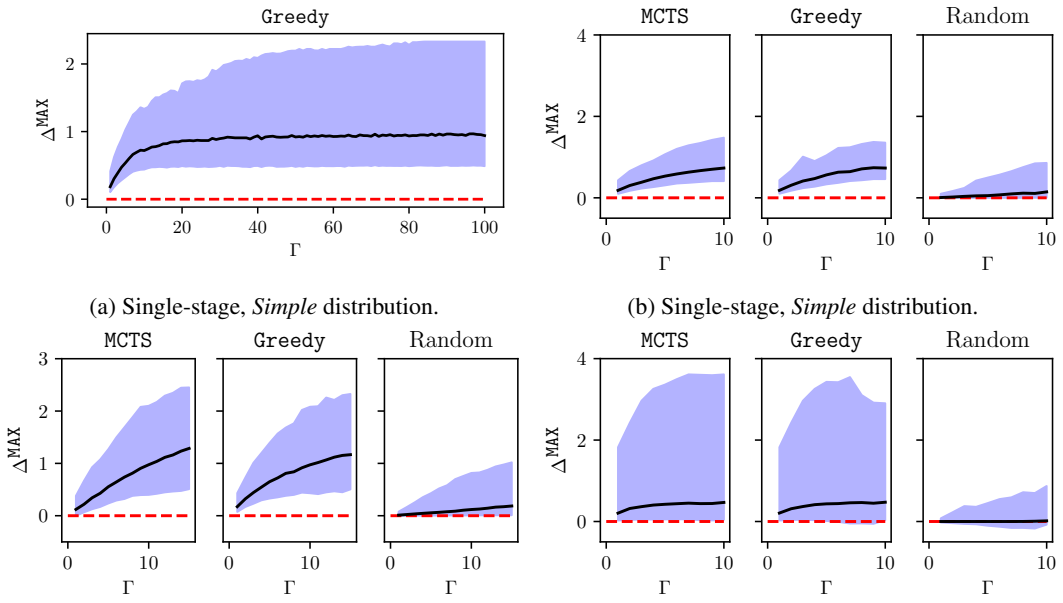
In these initial experiments we evaluate each method on 10 edge rejections *realizations* (only a small subset). We estimate $\Delta^{\texttt{MAX}}$ for each method and each graph by averaging the final matching weight over all realizations. Figure 5c shows the results of these experiments.

These initial multi-stage results are quite similar to our single-stage results. However it is notable that the objective value in the multi-stage setting is somewhat higher than in the single-stage setting–even using the simple method Greedy. Further, this suggests that more can be gained by developing a more sophisticated multi-stage edge selection policy. We leave this for future work.

5 Conclusions and Future Research Directions

Many planned kidney exchange transplants *fail* for a variety of reasons; these failures greatly reduce the number of transplants that an exchange can facilitate, and increase the waiting time for many patients in need of a kidney. Avoiding transplant failures is a challenge, as exchanges are often constrained by policy and law in how they match patients and donors. We consider a setting where exchanges can *pre-screen* certain transplants, while still matching patients and donors using a fixed policy. We formalize a multi-stage optimization problem based on realistic assumptions about how transplants fail, and how exchanges match patients and donors; we emphasize that these important assumptions are not included in prior work. While this problem is challenging in theory, we show

[9]For the *KPD* distribution we use an approximation of Fail-Aware, which assumes a uniform edge failure probability.



(a) Single-stage, *Simple* distribution.

(b) Single-stage, *Simple* distribution.

(c) Multi-stage, *Simple* distribution.

(d) Single-stage, *KPD* distribution.

Figure 5: Results for UNOS graphs. Right: edge budget up to 10 for the *Simple* distribution (top) and the *KPD* distribution (bottom). Top-left: `Greedy` with edge budget up to 100, for the simple distribution. Bottom-left: multi-stage methods using the *Simple* distribution. In all plots, a solid line indicates median $\Delta^{\texttt{MAX}}$ over all UNOS graphs, and shading is between the 10^{th} and 90^{th} percentiles; a dotted line indicates the baseline.

that it is much easier in practice–with computational experiments using both synthetic data and real data from the United Network for Organ Sharing. In experiments, we find that pre-screening even a small number of potential transplants (around 10) significantly increases the overall quality of the final match–by more than 100% of the original match weight.

Our initial study of the pre-screening problem suggests several areas for future work. First we assume that the distribution of transplant failures is known, when in reality only rough approximations of these distributions are available. Second, we assume that exchange participants (donors, recipients, hospitals) are not strategic. In reality, strategic behavior plays a substantial role in real exchanges [2]; we expect that participants might behave strategically when responding to pre-screening requests. Third, our model does not account for equitable treatment of different patients [21]. For example, it may be the case that pre-screening a transplant decreases the likelihood of the transplant being matched. That might disproportionately impact highly-sensitized patients, which are both sicker and more difficult to match than other patients.

Broader Impact

This work lives within the broader context of kidney exchange research. For clarity, we separate our broader impacts into two sections: first we discuss the impact of kidney exchange in general; then we discuss our work in particular, within the context of kidney exchange research and practice.

Impacts of Kidney Exchange Patients with end-stage renal disease have only two options: receive a transplant, or undergo dialysis once every few days, for the rest of their lives. In many countries (including the US), these patients register for a deceased donor waiting list–and it can be months or years before they receive a transplant. Many of these patients have a friend or relative willing to donate a kidney, however many patients are incompatible with their corresponding donor. Kidney exchange allows patients to "swap" their incompatible donor, in order to find a *higher-quality* match, *more quickly* than a waiting list. Transplants allow patients a higher quality of life, and cost far less, than lifelong dialysis. About 10% of kidney transplants in the US are facilitated by an exchange.

Finding the "most efficient" matching of kidney donors to patients is a (computationally) hard problem, which cannot be solved by hand in most cases. For this reason many fielded exchanges use algorithms to quickly find an efficient matching of patients and donors. Many researchers study kidney exchange from an algorithmic perspective, often with the goal of improving the number or quality of transplants facilitated by exchanges. Indeed, this is the purpose of our paper.

Impacts of Our Work In this paper we investigate the impact of pre-screening certain potential transplants (edge) in an exchange, prior to constructing the final patient-donor matching. To our knowledge, some modern fielded exchanges pre-screen potential transplants in an ad-hoc manner; meaning they do not consider the impacts of pre-screening on the final matching. We propose methods to estimate the importance of pre-screening each edge, as measured by the change in the overall number and quality of matched transplants.[10] Importantly, our methods do not require a change in matching policy; instead, they indicate to policymakers which potential transplants are important to pre-screen, and which are not. The impacts of our contributions are summarized below:

Some potential transplants cannot be matched, because they cannot participate in a "legal" cyclical or chain-like swap (according to the exchange matching policy). Accordingly, there is no "value" gained by pre-screening these transplants; our methods will identify these potential transplants, and will recommend that they not be pre-screened. Pre-screening requires doctors to spend valuable time reviewing potential donors; removing these unmatchable transplants from pre-screening will allow doctors to focus only on transplants that are relevant to the current exchange pool.

Some transplants are more important to pre-screen than others, and our methods help identify which are most important for the final matching. We estimate the value pre-screening of each transplant by *simulating* the exchange matching policy in the case that the pre-screened edge is pre-accepted, and in the case that it is pre-refused.

To estimate the value of pre-screening each transplant, we need to know (a) the likelihood that each transplant is pre-accepted and pre-refused, and (b) the likelihood that each planned transplant fails for any reason, after being matched. These likelihoods are used as input to our methods, and they can influence the estimated value of pre-screening different transplants. Importantly, it may not be desirable to calculate these likelihoods for each potential transplant (e.g., using data from the past). For example if a patient is especially sick, we may estimate that any potential transplant involving this patient is very likely to fail prior to transplantation (e.g., because the patient is to ill to undergo an operation). In this case, our methods may estimate that all potential transplants involving this patient have very low "value", and therefore recommend that these transplants should not be pre-screened. One way to avoid this issue is to use the same likelihood estimates for all transplants.

To estimate the impact of our methods (and how they depend on the assumed likelihoods, see above), we recommend using extensive modeling of different pre-screening scenarios before deploying our methods in a fielded exchange. This is important for several reasons: first, exchange programs cannot always *require* that doctors pre-screen potential transplants prior to matching. Since we cannot be sure which transplants will be pre-screened and which will not, simulations should be run to evaluate each possible scenario. Second, theoretical analysis shows that pre-screening transplants can—in the worst case—negatively impact the final outcome. While this worst-case

[10] Quality and quantity of transplants is measured by transplant weight, a numerical representation of transplant quality (e.g., see UNOS/OPTN Policy 13 regarding KPD prioritization points `https://optn.transplant.hrsa.gov/media/1200/optn_policies.pdf`).

outcome is possible, our computational experiments show that it is very unlikely; this can be addressed further with mode experiments tailored to a particular exchange program.

References

[1] D. Abraham, A. Blum, and T. Sandholm. Clearing algorithms for barter exchange markets: Enabling nationwide kidney exchanges. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, pages 295–304, 2007.

[2] N. Agarwal, I. Ashlagi, E. Azevedo, C. R. Featherstone, and Ö. Karaduman. Market failure in kidney exchange. *American Economic Review*, 109(11):4026–70, 2019.

[3] R. Anderson, I. Ashlagi, D. Gamarnik, and A. E. Roth. Finding long chains in kidney exchange using the traveling salesman problem. *Proceedings of the National Academy of Sciences*, 112 (3):663–668, 2015.

[4] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.

[5] N. Bansal, A. Gupta, J. Li, J. Mestre, V. Nagarajan, and A. Rudra. When LP is the cure for your matching woes: Improved bounds for stochastic matchings. *Algorithmica*, 63(4):733–762, 2012.

[6] H. Bidkhori, J. P. Dickerson, K. Ren, and D. C. McElfresh. Kidney exchange with inhomogeneous edge existence uncertainty. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI), forthcoming*, 2020.

[7] P. Biró, D. F. Manlove, and R. Rizzi. Maximum weight cycle packing in directed graphs, with application to kidney exchange programs. *Discrete Mathematics, Algorithms and Applications*, 1(04):499–517, 2009.

[8] P. Biró, B. Haase-Kromwijk, T. Andersson, E. I. Ásgeirsson, T. Baltesová, I. Boletis, C. Bolotinha, G. Bond, G. Böhmig, L. Burnapp, et al. Building kidney exchange programmes in europe—an overview of exchange practice and activities. *Transplantation*, 103(7):1514, 2019.

[9] P. Biró, J. van de Klundert, D. Manlove, W. Pettersson, T. Andersson, L. Burnapp, P. Chromy, P. Delgado, P. Dworczak, B. Haase, et al. Modelling and optimisation in european kidney exchange programmes. *European Journal of Operational Research*, 2019.

[10] A. Blum, A. Gupta, A. D. Procaccia, and A. Sharma. Harnessing the power of two crossmatches. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, pages 123–140, 2013.

[11] A. Blum, J. P. Dickerson, N. Haghtalab, A. D. Procaccia, T. Sandholm, and A. Sharma. Ignorance is almost bliss: Near-optimal stochastic matching with few queries. *Operations Research*, 2020. Earlier version appeared in the ACM Conference on Economics and Computation (EC), 2015.

[12] B. Bouzy. Associating shallow and selective global tree search with Monte Carlo for $9\times$ 9 Go. In *International Conference on Computers and Games*, pages 67–80. Springer, 2004.

[13] J. P. Dickerson. A unified approach to dynamic matching and barter exchange. Technical report, Doctoral Dissertation, Carnegie Mellon University, Pittsburgh, PA, 2016.

[14] J. P. Dickerson, D. Manlove, B. Plaut, T. Sandholm, and J. Trimble. Position-indexed formulations for kidney exchange. In *Proceedings of the ACM Conference on Economics and Computation (EC)*, 2016.

[15] J. P. Dickerson, A. D. Procaccia, and T. Sandholm. Failure-aware kidney exchange. *Management Science*, 65(4):1768–1791, 2019. Earlier version appeared in the ACM Conference on Economics and Computation (EC), 2013.

[16] B. Kartal, E. Nunes, J. Godoy, and M. Gini. Monte Carlo tree search with branch and bound for multi-robot task allocation. In *The IJCAI-16 workshop on autonomous mobile service robots*, volume 33, 2016.

[17] L. Kocsis and C. Szepesvári. Bandit based Monte-Carlo planning. In *European conference on machine learning*, pages 282–293. Springer, 2006.

[18] R. Leishman. Challenges in match offer acceptance in the optn kidney paired donation pilot program. In *INFORMS Annual Meeting*, 2019. Presentation in Session TB94 - Kidney Allocation & Exchange.

[19] Z. Li, N. Gupta, S. Das, and J. P. Dickerson. Equilibrium behavior in competing dynamic matching markets. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 389–395, 2018.

[20] D. Manlove and G. O'Malley. Paired and altruistic kidney donation in the UK: Algorithms and experimentation. *ACM Journal of Experimental Algorithmics*, 19(1), 2015.

[21] D. C. McElfresh and J. P. Dickerson. Balancing lexicographic fairness and a utilitarian objective with application to kidney exchange. *AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

[22] D. C. McElfresh, H. Bidkhori, and J. P. Dickerson. Scalable robust kidney exchange. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1077–1084, 2019.

[23] M. Molinaro and R. Ravi. Kidney exchanges and the query-commit problem. Manuscript, 2013.

[24] F. T. Rapaport. The case for a living emotionally related international kidney donor exchange registry. *Transplantation Proceedings*, 18:5–9, 1986.

[25] M. Rees, J. Kopke, R. Pelletier, D. Segev, M. Rutter, A. Fabrega, J. Rogers, O. Pankewycz, J. Hiller, A. Roth, T. Sandholm, U. Ünver, and R. Montgomery. A nonsimultaneous, extended, altruistic-donor chain. *New England Journal of Medicine*, 360(11):1096–1101, 2009.

[26] A. Roth, T. Sönmez, and U. Ünver. Kidney exchange. *Quarterly Journal of Economics*, 119(2): 457–488, 2004.

[27] A. Roth, T. Sönmez, and U. Ünver. A kidney exchange clearinghouse in New England. *American Economic Review*, 95(2):376–380, 2005.

[28] A. Roth, T. Sönmez, and U. Ünver. Pairwise kidney exchange. *Journal of Economic Theory*, 125(2):151–188, 2005.

[29] UNOS. United Network for Organ Sharing (UNOS). http://www.unos.org/.

A Kidney Exchange and Edge Failures

Brief history. Rapaport [24] proposed the initial idea for kidney exchange, while the first organized kidney exchange, the New England Paired Kidney Exchange (NEPKE), started in 2003–04 [26, 27, 28]. NEPKE has since ceased to operate; at the point of cessation, its pool of patients and donors was merged into the United Network for Organ Sharing (UNOS) exchange in late 2010. That exchange now contains over 60% of transplant centers in the US, and performs matching runs via a purely algorithmic approach (as we discuss in Sections 1 and 2, and in much greater depth by UNOS [29], which is mandated to transparently and publicly reveal its matching process).

There are also two large private kidney exchanges in the US, the National Kidney Registry (NKR) and the Alliance for Paired Donation (APD). They typically only work with large transplant centers. NKR makes their matching decisions manually and greatly prefers matching incrementally through chains. APD makes their decisions through a combination of algorithmic and manual decision making. There are also several smaller private kidney exchanges in the US. They typically only involve one or a couple of transplant centers. These include an exchange at Johns Hopkins University, a single-center exchange at the Methodist Specialty and Transplant Hospital in San Antonio, and a single-center exchange at Barnes-Jewish Hospital affiliated with the Washington University in St. Louis. Largely, these exchanges also make their matching decisions via a combined algorithmic and manual process. These exchanges compete in a variety of ways (e.g., by allowing patient-donor pairs to register in multiple exchange programs); this competition can lead to loss in efficiency [2] as well as sub-optimal changes to individual exchanges' matching polices [19].

There are now established kidney exchanges in the UK [20], Italy, Germany, Netherlands, Canada, England, Portugal, Israel, and many other countries. European countries are also explicitly exploring connecting their individual exchanges together in various ways [8].

Edge failures. The dilemma of edge failures is illustrated in the example exchange graph shown in Figure 6. This exchange consists of a 3-chain (dashed edges) and two 2-cycles (solid edges). Suppose the decision-maker queries edge e_A: if e_A is accepted, then the chain from the NDD (n) through pairs (d_1, p_1), (d_2, p_2), and (d_3, p_3), i.e., the dashed edges, can be included in the matching. However if e_A is queried and rejected, then the NDD cannot initiate the chain, and only the cycles may be matched. In our model, if e_A is not queried then it may still be matched.

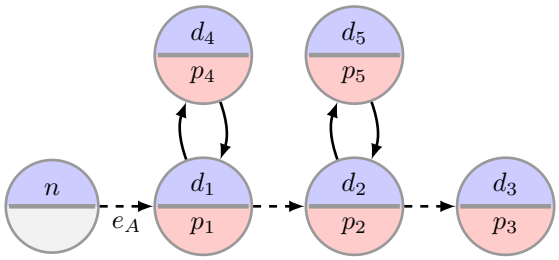


Figure 6: Sample exchange graph with a 3-chain (dashed edges) and two 2-cycles (solid edges). The NDD is denoted by n, and each patient (and associated donor) is denoted by p_i (d_i). If edge e_1 is not queried, or queried and *accepted*, then the chain may be included in the final matching. However if edge e_A is queried and *rejected*, then only the 2-cycles may be included in the final matching.

B Estimating The Objective of Problem 1

The objective of the single-stage edge selection problem requires evaluating all rejection scenarios $\boldsymbol{r} \sim \mathbb{P}_R(\boldsymbol{q})$, and the support of this distribution grows exponentially in the number of edges $|\boldsymbol{q}|$. In computational experiments, to estimate the objective of Problem 1, we sample up to 1000 scenarios from $\mathbb{P}_R(\boldsymbol{q})$. More explicitly: we *exactly* evaluate the objective of edge sets with fewer than 10 edges; for larger edge sets, we sample the objective using 1000 draws from $\mathbb{P}_R(\boldsymbol{q})$.

Using bootstrapping experiments we demonstrate that our sampling approach is sufficient to accurately estimate the true objective, even for large edge sets. For 152 UNOS graphs, we computed edge sets by running `Greedy` with edge budgets ranging from 1 to 100. For each edge set, we then sample a subset of $N \in \{10, 30, 50, 100, 1000\}$ rejection scenarios, with replacement, from the set of all sampled edge outcomes. For each edge set and choice of N we repeat 200 times and calculate the sample mean for each replication. We then compute the standard deviations of these bootstrap

Edge budgets	$N = 10$	$N = 30$	$N = 50$	$N = 100$	$N = 1000$
1-10	0.10	0.06	0.04	0.03	0.01
11-20	0.12	0.07	0.05	0.04	0.01
21-30	0.13	0.08	0.06	0.04	0.01
31-40	0.14	0.08	0.06	0.04	0.01
41-50	0.14	0.08	0.06	0.04	0.01
51-60	0.15	0.08	0.07	0.05	0.01
61-70	0.15	0.09	0.07	0.05	0.02
71-80	0.16	0.09	0.07	0.05	0.02
81-90	0.17	0.10	0.08	0.05	0.02
91-100	0.18	0.10	0.08	0.06	0.02

Table 2: Median normalized standard deviation of the bootstrap mean, over 200 bootstrap samples for each sample size N, binned by edge budget.

Table 3: Single-stage results on random graphs with the *Simple* edge distribution, using the variable IIAB edge budget (top rows), and the failure-aware method (bottom row). Columns P_X indicates the X^{th} percentile of $\Delta^{\texttt{MAX}}$ over all 30 random graphs, for graphs with $N = 50$, 75, and 100 vertices.

	$N = 50$			$N = 75$			$N = 100$		
Method	P_{10}	P_{50}	P_{90}	P_{10}	P_{50}	P_{90}	P_{10}	P_{50}	P_{90}
`MCTS`	0.22	0.30	0.38	0.11	0.33	0.46	0.23	0.33	0.38
`Greedy`	0.21	0.30	0.38	0.12	0.32	0.48	0.27	0.39	0.43
Random	0.12	0.19	0.23	0.10	0.19	0.28	0.12	0.19	0.23
`IIAB`	0.07	0.24	0.34	0.11	0.22	0.41	0.07	0.24	0.34
Fail-Aware	0.00	0.02	0.10	0.00	0.06	0.18	0.00	0.02	0.10

sample means to estimate the variance due to sampling. For each N, we calculate the mean sample standard deviation, normalized by the sample mean. Table 2 shows the median normalized standard deviation for all experiments under each N, with edge budgets aggregated into 10 bins. We find that with $N = 1000$ samples, the standard deviation was on average only about 2% of the overall mean value, even for large edge budgets.

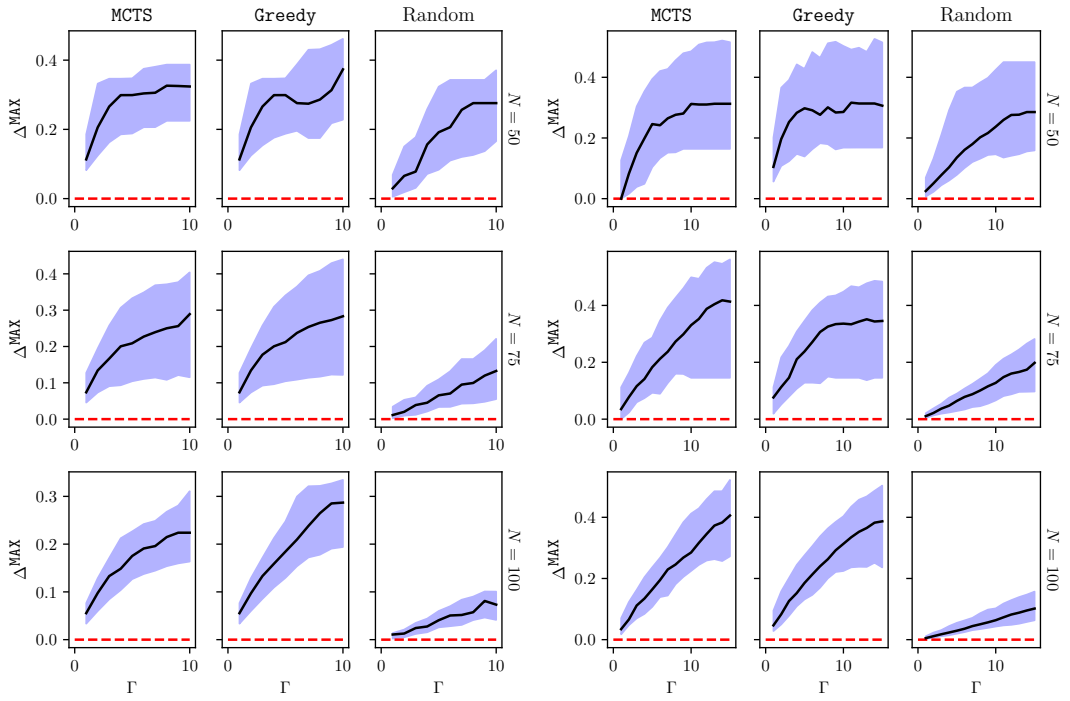
C Additional Computational Results

First we show results for both single-stage and multi-stage edge selection on random graphs (see § 4 for a description of these graphs). For $N = 50$, 75, and 100, we generate 30 random graphs with N vertices and $p = 0.01$. For each graph we run single-stage experiments with $\Gamma = 1, \ldots, 10$ and multi-stage experiments with $\Gamma = 1, \ldots, 15$. Unlike experiments on UNOS graphs we use a time limit of 20 minutes per edge; all other parameters are the same. Figure 7a and 7b show single-stage and multi-stage results for all random graphs, respectively. Table 3 shows comparisons to `IIAB` and Fail-Aware for random graphs with $N = 50$, 75, and 100.

As with UNOS graphs, results for `MCTS` and `Greedy` are quite similar, and both methods achieve larger $\Delta^{\texttt{MAX}}$ than Random, `IIAB`, and Fail-Aware. We make two observations: (1) `Greedy` appears to achieve larger $\Delta^{\texttt{MAX}}$ than `MCTS` in the single-stage setting, likely because of insufficient training time for `MCTS`; (2) in the multi-stage setting, `MCTS` performs *at least* as well as `Greedy`, and often better. Observation (2) is consistent with our experiments on UNOS graphs, and is somewhat surprising given that `MCTS` used less training time in these experiments. This suggests that `MCTS` may substantially improve over `Greedy` in the multi-stage setting; we leave further investigation to future work.

D Proofs for Section 2

In the proofs of Proposition 2.1 and Proposition 2.2 we consider a setting where all edges' pre-match rejections and post-match failures are i.i.d., where $P_R = 0.5$ is the pre-match rejection probability, $P_Q = 1.0$ is the post-match success probability if the edge is queried-and-accepted,



(a) Single-stage. (b) Multi-stage.

Figure 7: Results for 30 random graphs with edge probability $p = 0.01$ and $N = 50$ vertices (top row), $N = 75$ (middle row), and $N = 100$ (bottom row). All experiments use the *Simple* edge distribution. In all plots, a solid line indicates median $\Delta^{\texttt{MAX}}$ over all 30 random graphs, and shading is between the 10^{th} and 90^{th} percentiles; a dotted line indicates the baseline.

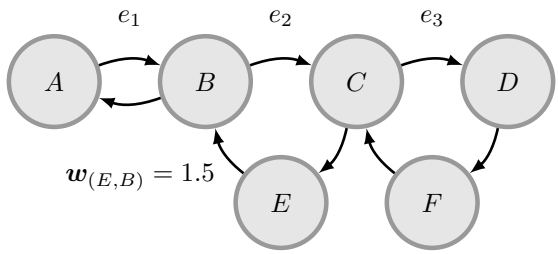


Figure 8: Exchange graph for Propositions 2.1 and 2.2. All edges have weight 1 except for edge (E, B), which has weight 1.5.

and $P_N = 0.5$ is the success probability if e is not queried. That is, queried edges have rejection probability 0.5, accepted edges have zero failure probability, and non-queried edges have failure probability 0.5.

D.1 Proof of Proposition 2.1

(Proof by counterexample.) We provide an example where querying a single edge results in a *lower* objective value in Problem 1 (i.e., final expected matching weight) than querying no edges–when using the max-weight matching policy $M^{\texttt{MAX}}(\cdot)$.

Consider the exchange graph in Figure 8; edge (E, B) has weight 1.5, while all other edges have weight 1. First we consider the objective due to querying no edges, $V^S(\mathbf{0})$. In this case, no edges can be rejected pre-match, the max-weight matching includes cycle (C, D, F) (expected weight $3 \times (1/2)^3 = 3/8$) and cycle (A, B) (expected weight $2 \times (1/2)^2 = 1/2$), with total expected matching weight $7/8$. That is, $V^S(\mathbf{0}) = 7/8$.

Next consider the objective due to querying only edge $e_3 = (C, D)$, and let $\boldsymbol{q}'$ denote edge set $\{e_3\}$. With probability $1/2$, e_3 is rejected and cycle (B, C, E) is the max-weight matching – with

expected weight $3.5/8$. With probability $1/2$, e_3 is accepted and the max-weight matching includes cycles (A, B) (with expected weight $1/2$) and (C, D, F) (with expected weight $3/4$); this matching has total expected weight $5/4$. Thus, $V^S(\boldsymbol{q}) = 27/32 < 7/8 = V^S(\mathbf{0})$, which concludes the proof.

D.2 Proof of Proposition 2.2

(Proof by counterexample.) We provide an example where the objective value in Problem 1 (i.e., final expected matching weight) is non-submodular–when using the max-weight matching policy $M^{\texttt{MAX}}(\cdot)$. We use the same rejection and failure distribution as in the proof of Proposition 2.1.

Consider the exchange graph in Figure 8; edge (E, B) has weight 1.5, while all other edges have weight 1. With some abuse of notation, we will denote by $V^S(\{e_a, \ldots, e_N\})$ the objective of Problem 1 due to edge set $\{e_a, \ldots, e_N\}$. Our counterexample for submodulartiy is that, for this graph,

$$V^S(X \cup \{e_1, e_2\}) + V^S(X) > V^S(X \cup \{e_1\}) + V^S(X \cup \{e_2\}),$$

with set $X \equiv \{e_3\}$. That is, the objective increase due to of querying *both* edges e_1 and e_3 is greater than the combined increase due to querying both edges separately. Next we explicitly calculate each of the above terms.

$V^S(X) = V^S(\{e_3\})$**.** There are two cases to consider:

- e_3 is accepted, with probability $1/2$. The max-weight matching is cycles (A, B) and (C, D, F), with expected weight $(1/2 + 3/4)$,
- e_3 is rejected, with probability $1/2$. The max-weight matching is cycle (B, C, E), with expected weight $3.5/8$.

Thus, $V^S(X) = (1/2)(1/2 + 3/4) + (1/2)(3.5/8) = 27/32$.

$V^S(X \cup \{e_1\}) = V^S(\{e_1, e_3\})$**.** There are four cases to consider:

- e_1 and e_3 are accepted, with probability $1/4$. The max-weight matching is cycles (A, B) and (C, D, F), with expected weight $(1 + 3/8)$,
- e_1 is rejected and e_3 is accepted, with probability $1/4$. The max-weight matching is cycle (B, C, E), with expected weight $3.5/8$.
- e_1 is accepted and e_3 is rejected, with probability $1/4$. The max-weight matching is cycle (B, C, E), with expected weight $3.5/8$.
- e_1 and e_3 are rejected, with probability $1/4$. The max-weight matching is cycle (B, C, E), with expected weight $3.5/8$.

Thus the objective is $V^S(X \cup \{e_3\}) = (1/4)(1 + 3/8) + (3/4)(3.5/8) = 43/64$.

$V^S(X \cup \{e_2\}) = V^S(\{e_2, e_3\})$**.** There are three cases to consider

- e_3 is accepted, with probability $1/2$. The max-weight matching is cycles (A, B) and (C, D, F), with expected weight $(1/2 + 3/4)$,
- e_3 is rejected and e_3 is accepted, with probability $1/4$. The max-weight matching is cycle (B, C, E), with expected weight $3.5/4$,
- e_3 and e_2 are rejected, with probability $1/4$. The max-weight matching is cycle (A, B), with expected weight $1/2$.

Thus the objective is $V^S(X \cup \{e_2\}) = (1/2)(1/2 + 3/4) + (1/4)(3.5/4) + (1/4)(1/2) = 31/32$.

$V^S(X \cup \{e_1, e_2\}) = V^S(\{e_1, e_2, e_3\})$**.** There are four cases to consider:

- e_1 and e_3 are accepted, with probability $1/4$. The max-weight matching is cycles (A, B) and (C, D, F), with expected weight $(1 + 3/4)$,
- e_1 is accepted and e_2 is rejected, with probability $1/4$ (the response from e_3 is irrelevant). The max-weight matching is (A, B) and (C, D, F), with expected weight $1 + 3/8$.
- e_1 is rejected and e_2 is accepted (the response from e_3 is irrelevant), with probability $1/4$. The max-weight matching is cycle (B, C, E), with expected weight $3.5/4$.
- e_1 and e_2 are rejected (the response from e_3 is irrelevant), with probability $1/4$. The max-weight matching is cycle (C, D, F), with expected weight $3/8$.

Thus the objective is $V^S(X \cup \{e_1, e_2\}) = (1/4)(1 + 3/4) + (1/4)(1 + 3/8) + (1/4)(3.5/4) + (1/4)(3/8) = 35/32$.

Finally, we have:

$$\begin{aligned} V^S(X \cup \{e_1, e_2\}) + V^S(X) &= 35/32 + 27/32 \\ &= 1.9375 \end{aligned}$$

and

$$\begin{aligned} V^S(X \cup \{e_1\}) + V^S(X \cup \{e_2\}) &= 43/64 + 31/32 \\ &= 1.640625 \end{aligned}$$

Therefore, $V^S(X \cup \{e_1, e_2\}) + V^S(X) > V^S(X \cup \{e_1\}) + V^S(X \cup \{e_2\})$, which concludes the proof.

D.3 Proof of Proposition 2.5

For the proof of Proposition 2.5 we make one assumption about the distribution of edge rejections and failures: querying *additional* edges cannot increase the overall probability of rejection or failure for any edge.

First we prove a handful of useful results.

Lemma D.1. *If all edges are independent and Assumption 2.3 holds, then additional edge queries cannot decrease expected post-match cycle and chain weights. Formally,*

$$\mathbb{E}\left[F(c, \boldsymbol{r} + \boldsymbol{f}) \mid \boldsymbol{q}, \boldsymbol{r}\right] \leq \mathbb{E}\left[F(c, \boldsymbol{r} + \boldsymbol{r}' + \boldsymbol{f}) \mid \boldsymbol{q} + \boldsymbol{q}', \boldsymbol{r}\right]$$

for any $\boldsymbol{q}, \boldsymbol{q}' \in \{0,1\}^{|E|}$ such that $\boldsymbol{q} + \boldsymbol{q}' \in \{0,1\}^{|E|}$, for any $\boldsymbol{r} \in \{0,1\}^{|E|}$, and for all $c \in \mathcal{C}$.

Proof. We address cycles and chains separately.

Cycles. Conditional on fixed $\boldsymbol{q}$ and $\boldsymbol{r}$, the expected weight of cycle $c = (e_1, \ldots, e_L)$ is expressed as

$$\begin{aligned} \mathbb{E}\left[F(c, \boldsymbol{r} + \boldsymbol{f}) \mid \boldsymbol{q}, \boldsymbol{r}\right] &= \left(\sum_{e \in c} w_e\right) \mathbb{E}\left[\prod_{e \in c} (1 - \boldsymbol{r}_e - \boldsymbol{f}_e) \mid \boldsymbol{q}, \boldsymbol{r}\right] \\ &= \left(\sum_{e \in c} w_e\right) \prod_{e \in c} \left(1 - \mathbb{E}\left[\boldsymbol{r}_e + \boldsymbol{f}_e \mid \boldsymbol{q}, \boldsymbol{r}\right]\right) \end{aligned}$$

where the second step is due to the fact that all $\boldsymbol{f}_e$ are independent. Similarly, for fixed $\boldsymbol{q}'$,

$$\mathbb{E}\left[F(c, \boldsymbol{r} + \boldsymbol{r}' + \boldsymbol{f}) \mid \boldsymbol{q} + \boldsymbol{q}', \boldsymbol{r}\right] = \left(\sum_{e \in c} w_e\right) \prod_{e \in c} \left(1 - \mathbb{E}\left[\boldsymbol{r}_e + \boldsymbol{r}'_e + \boldsymbol{f}_e \mid \boldsymbol{q} + \boldsymbol{q}', \boldsymbol{r}\right]\right) \ .$$

Due to Assumption 2.3, the following inequality holds for all edges $e \in E$

$$\mathbb{E}\left[\boldsymbol{r}_e + \boldsymbol{f}_e \mid \boldsymbol{q}, \boldsymbol{r}\right] \geq \mathbb{E}\left[\boldsymbol{r}_e + \boldsymbol{r}'_e + \boldsymbol{f}_e \mid \boldsymbol{q} + \boldsymbol{q}', \boldsymbol{r}\right] \ ,$$

and it follows that

$$\mathbb{E}\left[F(c, \boldsymbol{r} + \boldsymbol{f}) \mid \boldsymbol{q}, \boldsymbol{r}\right] \leq \mathbb{E}\left[F(c, \boldsymbol{r} + \boldsymbol{r}' + \boldsymbol{f}) \mid \boldsymbol{q} + \boldsymbol{q}', \boldsymbol{r}\right].$$

Chains. Similarly, the expected weight of chain $c = (e_1, \ldots, e_L)$ is expressed as

$$\begin{aligned} \mathbb{E}\left[F(c, \boldsymbol{r} + \boldsymbol{f}) \mid \boldsymbol{q}, \boldsymbol{r}\right] &= \sum_{k=1}^{L} \left(\sum_{j=1}^{k} w_j\right) \mathbb{E}\left[\prod_{j=1}^{k} (1 - \boldsymbol{r}_{e_j} - \boldsymbol{f}_{e_j}) \mid \boldsymbol{q}, \boldsymbol{r}\right] \\ &= \sum_{k=1}^{L} \left(\sum_{j=1}^{k} w_j\right) \prod_{j=1}^{k} \left(1 - \mathbb{E}\left[\boldsymbol{r}_{e_j} + \boldsymbol{f}_{e_j} \mid \boldsymbol{q}, \boldsymbol{r}\right]\right) \ , \end{aligned}$$

where the second step is due to the fact that $\boldsymbol{f}_e$ are independent. Similarly,

$$\mathbb{E}\left[F(c, \boldsymbol{r} + \boldsymbol{r}' + \boldsymbol{f}) \mid \boldsymbol{q} + \boldsymbol{q}', \boldsymbol{r}\right] = \sum_{k=1}^{L} \left(\sum_{j=1}^{k} w_j\right) \prod_{j=1}^{k} \left(1 - \mathbb{E}\left[\boldsymbol{r}_{e_j} + \boldsymbol{r}'_{e_j} + \boldsymbol{f}_{e_j} \mid \boldsymbol{q} + \boldsymbol{q}', \boldsymbol{r}\right]\right) \ .$$

as before, due to Assumption 2.3 it follows that

$$\mathbb{E}\left[F(c, \boldsymbol{r}+\boldsymbol{f}) \mid \boldsymbol{q}, \boldsymbol{r}\right] \leq \mathbb{E}\left[F(c, \boldsymbol{r}+\boldsymbol{r}'+\boldsymbol{f}) \mid \boldsymbol{q}+\boldsymbol{q}', \boldsymbol{r}\right].$$

□

Lemma D.2. *With a failure-aware matching policy, and if all edges are independent, adding a single edge to any edge query set weakly improves the objective of Problem 1. Formally, for any $\boldsymbol{q}, \boldsymbol{q}' \in \{0,1\}^{|E|}$ with $\boldsymbol{q}+\boldsymbol{q}' \in \{0,1\}^{|E|}$ and $|\boldsymbol{q}'|=1$, and $M(\boldsymbol{r}) \equiv M^{\texttt{FA}}(\boldsymbol{r})$,*

$$V^S(\boldsymbol{q}) \leq V^S(\boldsymbol{q}+\boldsymbol{q}')$$

Proof. The objective of Problem 1 for edge set $\boldsymbol{q}$ is expressed as

$$\begin{aligned}
V^S(\boldsymbol{q}) &= \mathop{\mathbb{E}}_{\boldsymbol{r}|\boldsymbol{q}}\left[\mathop{\mathbb{E}}_{\boldsymbol{f}|\boldsymbol{q},\boldsymbol{r}}\left[\sum_{c\in\mathcal{C}} M_c^{\texttt{FA}}(\boldsymbol{r})F(c,\boldsymbol{r}+\boldsymbol{f})\right]\right] \\
&= \sum_{\boldsymbol{r}\in\{0,1\}^{|\boldsymbol{q}|}} P_{\boldsymbol{q}}(\boldsymbol{r}) \mathop{\mathbb{E}}_{\boldsymbol{f}|\boldsymbol{q},\boldsymbol{r}}\left[\sum_{c\in\mathcal{C}} M_c^{\texttt{FA}}(\boldsymbol{r})F(c,\boldsymbol{r}+\boldsymbol{f})\right] \\
&= \sum_{\boldsymbol{r}\in\{0,1\}^{|\boldsymbol{q}|}} P_{\boldsymbol{q}}(\boldsymbol{r}) \sum_{c\in\mathcal{C}} M_c^{\texttt{FA}}(\boldsymbol{r}) \mathop{\mathbb{E}}_{\boldsymbol{f}|\boldsymbol{q},\boldsymbol{r}}\left[F(c,\boldsymbol{r}+\boldsymbol{f})\right]
\end{aligned}$$

For edge set $\boldsymbol{q}+\boldsymbol{q}'$ we partition response variables into $\boldsymbol{r}, \boldsymbol{r}' \in \{0,1\}^{|E|}$, where $\boldsymbol{r}_e$ is the response variable for all edges $e \in \boldsymbol{q}$, and $\boldsymbol{r}_e = 0$ for all other edges (including the edge in $\boldsymbol{q}'$). Similarly, $\boldsymbol{r}'_e$ is the response variable for edge $\boldsymbol{q}'$, and $\boldsymbol{r}'_e = 0$ for all other edges. The objective of $\boldsymbol{q}+\boldsymbol{q}'$ is expressed as

$$\begin{aligned}
V^S(\boldsymbol{q}+\boldsymbol{q}') &= \mathop{\mathbb{E}}_{\boldsymbol{r},\boldsymbol{r}'|\boldsymbol{q}+\boldsymbol{q}'}\left[\mathop{\mathbb{E}}_{\boldsymbol{f}|\boldsymbol{q}+\boldsymbol{q}',\boldsymbol{r}+\boldsymbol{r}'}\left[\sum_{c\in\mathcal{C}} M_c^{\texttt{FA}}(\boldsymbol{r}+\boldsymbol{r}')F(c,\boldsymbol{r}+\boldsymbol{r}'+\boldsymbol{f})\right]\right] \\
&= \sum_{\boldsymbol{r}\in\{0,1\}^{|\boldsymbol{q}|}} P_{\boldsymbol{q}+\boldsymbol{q}'}(\boldsymbol{r}) \mathop{\mathbb{E}}_{\boldsymbol{r}'|\boldsymbol{q}+\boldsymbol{q}'}\left[\mathop{\mathbb{E}}_{\boldsymbol{f}|\boldsymbol{q}+\boldsymbol{q}',\boldsymbol{r}+\boldsymbol{r}'}\left[\sum_{c\in\mathcal{C}} M_c^{\texttt{FA}}(\boldsymbol{r}+\boldsymbol{r}')^\top F(c,\boldsymbol{r}+\boldsymbol{r}'+\boldsymbol{f})\right]\right] \\
&= \sum_{\boldsymbol{r}\in\{0,1\}^{|\boldsymbol{q}|}} P_{\boldsymbol{q}}(\boldsymbol{r}) \mathop{\mathbb{E}}_{\boldsymbol{r}'|\boldsymbol{q}+\boldsymbol{q}'}\left[\mathop{\mathbb{E}}_{\boldsymbol{f}|\boldsymbol{q}+\boldsymbol{q}',\boldsymbol{r}+\boldsymbol{r}'}\left[\sum_{c\in\mathcal{C}} M_c^{\texttt{FA}}(\boldsymbol{r}+\boldsymbol{r}')F(c,\boldsymbol{r}+\boldsymbol{r}'+\boldsymbol{f})\right]\right],
\end{aligned}$$

where in the final line we replace $P_{\boldsymbol{q}+\boldsymbol{q}'}(\boldsymbol{r})$ with $P_{\boldsymbol{q}}(\boldsymbol{r})$, because each $\boldsymbol{r}_e$ is conditionally independent, given $\boldsymbol{q}_e$.

Next, by definition

$$\mathop{\mathbb{E}}_{\boldsymbol{f}|\boldsymbol{q}+\boldsymbol{q}',\boldsymbol{r}+\boldsymbol{r}'}\left[\sum_{c\in\mathcal{C}} M_c^{\texttt{FA}}(\boldsymbol{r}+\boldsymbol{r}')F(c,\boldsymbol{r}+\boldsymbol{r}'+\boldsymbol{f})\right] \geq \mathop{\mathbb{E}}_{\boldsymbol{f}|\boldsymbol{q}+\boldsymbol{q}',\boldsymbol{r}+\boldsymbol{r}'}\left[\sum_{c\in\mathcal{C}} \boldsymbol{x}_c F(c,\boldsymbol{r}+\boldsymbol{r}'+\boldsymbol{f})\right] \quad \forall \boldsymbol{x}\in\mathcal{M}.$$

That is, M^{FA} is guaranteed to maximize this expectation, and thus

$$\begin{aligned}
V^S(\boldsymbol{q}+\boldsymbol{q}') &\geq \sum_{\boldsymbol{r}\in\{0,1\}^{|\boldsymbol{q}|}} P_{\boldsymbol{q}}(\boldsymbol{r}) \mathop{\mathbb{E}}_{\boldsymbol{r}'|\boldsymbol{q}+\boldsymbol{q}'}\left[\mathop{\mathbb{E}}_{\boldsymbol{f}|\boldsymbol{q}+\boldsymbol{q}',\boldsymbol{r}+\boldsymbol{r}'}\left[\sum_{c\in\mathcal{C}} M_c^{\texttt{FA}}(\boldsymbol{r})F(c,\boldsymbol{r}+\boldsymbol{r}'+\boldsymbol{f})\right]\right] \qquad \text{(B)} \\
&= \sum_{\boldsymbol{r}\in\{0,1\}^{|\boldsymbol{q}|}} P_{\boldsymbol{q}}(\boldsymbol{r}) \sum_{c\in\mathcal{C}} M_v^{\texttt{FA}}(\boldsymbol{r}) \mathop{\mathbb{E}}_{\boldsymbol{r}'|\boldsymbol{q}+\boldsymbol{q}'}\left[\mathop{\mathbb{E}}_{\boldsymbol{f}|\boldsymbol{q}+\boldsymbol{q}',\boldsymbol{r}+\boldsymbol{r}'}\left[F(c,\boldsymbol{r}+\boldsymbol{r}'+\boldsymbol{f})\right]\right] \qquad \text{(C)}
\end{aligned}$$

Finally, combining (B) and (C) with Lemma D.1, the following inequality holds

$$V^S(\boldsymbol{q}) \leq V^S(\boldsymbol{q}+\boldsymbol{q}').$$

□

Using the above lemmas, the proof of Proposition 2.5 is straightforward:

Proposition 2.5 *If edges are independent, and Assumption 2.3 holds, then with a failure-aware matching policy the objective of Problem 1 is monotonic in the set of queried edges.*

Proof. Let $\boldsymbol{q}', \boldsymbol{q}'' \in \mathcal{E}$ be two edge sets such that $\boldsymbol{q}' \subseteq \boldsymbol{q}''$. It remains to show that, with matching policy $M(\boldsymbol{r}) \equiv M^{\texttt{FA}}(\boldsymbol{r})$,

$$V^S(\boldsymbol{q}'') \leq V^S(\boldsymbol{q}').$$

First note that because $\mathcal{E}$ is a matroid, there is a sequence of edges $(\boldsymbol{q}^{e_1}, \ldots, \boldsymbol{q}^{e_L})$ (with each $|\boldsymbol{q}^{e_i}| = 1$) such that $\boldsymbol{q}'' + \boldsymbol{q}^{e_1} + \cdots + \boldsymbol{q}^{e_L} = \boldsymbol{q}'$. Due to Lemma D.2, the following sequence of inequalities hold:

$$\begin{aligned} V(\boldsymbol{q}'') &\leq V(\boldsymbol{q}'' + \boldsymbol{q}^{e_1}) \\ &\leq V(\boldsymbol{q}'' + \boldsymbol{q}^{e_1} + \boldsymbol{q}^{e_2}) \\ &\cdots \\ &\leq V(\boldsymbol{q}'' + \boldsymbol{q}^{e_1} + \cdots + \boldsymbol{q}^{e_L}) \\ &= V(\boldsymbol{q}') \end{aligned}$$

which concludes the proof. □

E Algorithm Descriptions

Here we describe more explicitly the algorithms for `Greedy` and `MCTS`, for both the single-stage and multi-stage settings.

E.1 UCB Value Estimates for `MCTS`

Both the single- and multi-stage versions of `MCTS` use the method of [17] to select the next child node to explore. The formula used to estimate a node's UCB value is

$$\frac{\frac{U}{N} - V^{min}}{V^{max} - V^{min}} + \sqrt{N^P/N}$$

where U is the "UCB value estimate" calculated by `MCTS`, N is the number of visits to the node, N^P is the number of visits to the node's parent, and V^{max} and V^{min} are the largest and smallest *node values* encountered during search. In single-stage `MCTS`, all nodes have both a *node value* (the objective value of Problem 1) and a UCB value estimate; as described below, in multi-stage `MCTS` only query nodes have a UCB value estimate, and only leaf nodes have a *node value* (expected matched weight, after observing responses from all queried edges).

E.2 Greedy Single-Stage Edge Selection

Algorithm 3 gives a pseudocode description of `Greedy` for the single-stage setting.

ALGORITHM 3: `Greedy`: Greedy Search Heuristic for Single-Stage Edge Selection

(input) $\mathcal{E}$: legal edge sets

$\boldsymbol{q}^R \leftarrow \mathbf{0}$ the root node (no edges)
$V^* \leftarrow$ objective value of $\boldsymbol{q}^R$ Problem 1
while $\boldsymbol{q}^R$ *has children* **do**
  $\boldsymbol{q}' \leftarrow$ child node of $\boldsymbol{q}^R$ with maximal objective value in Problem 1
  $\boldsymbol{q}^R \leftarrow \boldsymbol{q}'$
return $\boldsymbol{q}^R$

E.3 Multi-Stage Edge Selection

In the following sections we describe multi-stage versions of `MCTS` and `Greedy`. Unlike in the single-stage setting, these algorithms take as input a set of previously-queried edges $\boldsymbol{q} \in \{0,1\}^{|E|}$ and a corresponding set of observed rejections $\boldsymbol{r} \in \{0,1\}^{|E|}$; they output the *next* edge to query.

Multi-Stage `MCTS`**.** The multi-stage search tree is somewhat more complicated than in the single-stage setting, as each node in the search tree corresponds to both a set of queried edges and a set of observed rejections. For this purpose we use two types of nodes: *outcome* nodes, and *query* nodes. Outcome nodes consist of previously-queried edges $\boldsymbol{q}$ and previously-observed rejections $\boldsymbol{r}$, and are represented by tuple $(\boldsymbol{q}, \boldsymbol{r})$. (The root of the search tree corresponds to *no* queries or observed rejections, $(\mathbf{0}, \mathbf{0})$.) The children of an outcome node are *query* nodes, represented by the next edge to query from the parent (outcome), represented by tuple $(\boldsymbol{q}, \boldsymbol{r}, e)$. Each outcome node has one child for every edge that has not yet been queried:

$$C^O(\boldsymbol{q}, \boldsymbol{r}) \equiv \{(\boldsymbol{q}, \boldsymbol{r}, e) \mid \forall e \in E : \boldsymbol{q} + \boldsymbol{u}^e \in \mathcal{E}\}$$

where $\boldsymbol{u}^e$ is the unit vector for element e ($\boldsymbol{u}^e_i = 0$ for all $i \neq e$, and $\boldsymbol{u}^e_e = 1$). Each query node has exactly two children: one where the queried edge is accepted, and one where the queried edge is rejected,

$$C^Q(\boldsymbol{q}, \boldsymbol{r}, e) \equiv \{(\boldsymbol{q} + \boldsymbol{u}^e, \boldsymbol{r}), (\boldsymbol{q} + \boldsymbol{u}^e, \boldsymbol{r} + \boldsymbol{u}^e)\}$$

As before, the *level* of a node refers to the number of queried edges: $|\boldsymbol{q}|$ for outcome nodes, and $|\boldsymbol{q}| + 1$ for query nodes.

As before we refer to nodes with no children as leaf nodes; note that only outcome nodes are leaf nodes. Unlike the single-stage version of `MCTS`, in the multi-stage setting we only consider the value of leaf nodes[11]. The value of a leaf (outcome) is

$$V^O(\boldsymbol{q}, \boldsymbol{r}) \equiv W(M(\boldsymbol{r}); \boldsymbol{q}, \boldsymbol{r}),$$

where as before $M(\boldsymbol{r})$ denotes the matching policy, and $W(\boldsymbol{x}; \boldsymbol{q}, \boldsymbol{r})$ denotes the expected matching weight of $\boldsymbol{x}$, subject to $\boldsymbol{q}$ and $\boldsymbol{r}$. The value of leaf outcome nodes is used to by `QSample` and `OSample` to guide multi-stage `MCTS`.

[11]This decision was made in part because initial results indicate that edge selection is essentially monotonic.

Algorithm 4 describes the multi-stage version of `MCTS`, taking previously-queried edges and observed responses as input. This algorithm initializes the value estimate $U[\cdot]$ and number of visits $N[\cdot]$ for query nodes in the next L levels–these quantities are used in the UCB calculation.

ALGORITHM 4: Multi-Stage `MCTS`

(input) $\mathcal{E}$: legal edge sets
(input) K: maximum size of any legal edge set
(input) T: time limit
(input) L: number of look-ahead levels
(input) $\boldsymbol{q}^R$: previously-queried edges
(input) $\boldsymbol{r}^R$: previously-observed rejections

$M \leftarrow \min\{N + L, K\}$
$Q \leftarrow$ all query nodes which are descendants of $(\boldsymbol{q}^R, \boldsymbol{r}^R)$, up to level M
$U[(\boldsymbol{q}, \boldsymbol{r}, e)] \leftarrow 0 \ \forall (\boldsymbol{q}, \boldsymbol{r}, e) \in Q$ UCB value estimate
$N[(\boldsymbol{q}, \boldsymbol{r}, e)] \leftarrow 0 \ \forall \boldsymbol{q} \in Q$ number of visits
while *less than time* T *has passed* **do**
    `QSample`($\boldsymbol{q}^R$, $\boldsymbol{r}^R$ M)
$(\boldsymbol{q}^R, \boldsymbol{r}^R, e^*) \leftarrow$ child node of $(\boldsymbol{q}^R, \boldsymbol{r}^R)$ with the greatest UCB estimate
return e^*

ALGORITHM 5: `QSample`: Function for sampling query nodes in multi-stage `MCTS`

(input) $(\boldsymbol{q}, \boldsymbol{r})$: outcome node
(input) M: maximum level to sample from

if $(\boldsymbol{q}, \boldsymbol{r})$ *has no children* **then**
    return $V^O(\boldsymbol{q}, \boldsymbol{r})$ (return the value of this outcome node)
if $(\boldsymbol{q}, \boldsymbol{r})$ *has children* **then**
    if $|\boldsymbol{q}| < M - 1$ **then**
        $(\boldsymbol{q}, \boldsymbol{r}, e') \leftarrow$ child node of $(\boldsymbol{q}, \boldsymbol{r})$ with the greatest UCB estimate
        `OSample`($\boldsymbol{q}$, $\boldsymbol{r}$, e)
    else
        $(\boldsymbol{q}', \boldsymbol{r}') \leftarrow$ random leaf node, descendant from $(\boldsymbol{q}, \boldsymbol{r})$
        return $V^O(\boldsymbol{q}', \boldsymbol{r}')$

ALGORITHM 6: `OSample`: Function for sampling outcome nodes in multi-stage `MCTS`

(input) $(\boldsymbol{q}, \boldsymbol{r}, e)$: query node

$N[(\boldsymbol{q}, \boldsymbol{r}, e)] \leftarrow N[(\boldsymbol{q}, \boldsymbol{r}, e)] + 1$
$\boldsymbol{q}' \leftarrow \boldsymbol{q} + \boldsymbol{u}^e$ (new query vector with edge e added)
$Z \leftarrow$ randomly sample a response to edge e (0 if accept, 1 if reject)
$\boldsymbol{r}' \leftarrow \boldsymbol{r} + Z\boldsymbol{u}^e$ (updated rejection vector)
$U[(\boldsymbol{q}, \boldsymbol{r}, e)] \leftarrow U[(\boldsymbol{q}, \boldsymbol{r}, e)] +$ `QSample`$(\boldsymbol{q}', \boldsymbol{r}')$

Algorithm 5 (`QSample`) samples query nodes from an outcome node, while Algorithm 6 (`OSample`) samples outcome nodes from a query node (and updates the query node's UCB value estimate).

Multi-Stage `Greedy`. Algorithm 7 gives a pseudocode description of the multi-stage version of `Greedy`. This search heuristic returns the next edge to query with the highest expected final matching weight, *ignoring all future queries*. In other words, this approach treats every edge as the *last* edge; one might call this heuristic "myopic" as well as greedy.

ALGORITHM 7: `Greedy` Heuristic for Multi-Stage Edge Selection

(input) $\mathcal{E}$: legal edge sets
(input) $\boldsymbol{q}$: previously-queried edges
(input) $\boldsymbol{r}$: previously-observed rejections

$e^* \leftarrow \emptyset$ $V^* \leftarrow 0$
for *all* $\boldsymbol{q}'$ *in* $\boldsymbol{q}$*'s children* **do**
  $e' \leftarrow$ the new edge queried in child node $\boldsymbol{q}'$
  $\boldsymbol{r}^A \leftarrow \boldsymbol{r}$
  $\boldsymbol{r}^R \leftarrow \boldsymbol{r}$
  $\boldsymbol{r}^A_{e'} \leftarrow 0$ (response scenario where e' is accepted, and $\boldsymbol{r}_{e'} = 0$)
  $\boldsymbol{r}^R_{e'} \leftarrow 1$ (response scenario where e' is rejected, and $\boldsymbol{r}_{e'} = 1$)
  $p^A \leftarrow$ probability that e is accepted, conditional on previous responses
  $p^R \leftarrow$ probability that e is rejected, conditional on previous responses
  $V' \leftarrow p^A \cdot W(M(\boldsymbol{r}^A); \boldsymbol{q}', \boldsymbol{r}^A) + p^R W(M(\boldsymbol{r}^R); \boldsymbol{q}', \boldsymbol{q}^R)$ (value of querying edge e')
  if $V' > V^*$ **then**
    $\boldsymbol{e}^* \leftarrow e'$
    $V^* \leftarrow V'$
return $\boldsymbol{e}^*$