Preference Elicitation and Interview Minimization in Stable Matchings

Joanna Drummond and **Craig Boutilier**
Department of Computer Science
University of Toronto
{jdrummond,cebly}@cs.toronto.edu

Abstract

While stable matching problems are widely studied, little work has investigated schemes for effectively eliciting agent preferences using either preference (e.g., comparison) queries or interviews (to form such comparisons); and no work has addressed how to combine both. We develop a new model for representing and assessing agent preferences that accommodates both forms of information and (heuristically) minimizes the number of queries and interviews required to determine a stable matching. Our *Refine-then-Interview (RtI)* scheme uses coarse preference queries to refine knowledge of agent preferences and relies on interviews only to assess comparisons of relatively "close" options. Empirical results show that RtI compares favorably to a recent pure interview minimization algorithm, and that the number of interviews it requires is generally independent of the size of the market.

Introduction

Due to its interesting structure and real-world importance, *stable marriage/matching problems (SMPs)* have generated considerable interest since Gale and Shapley's (1962) seminal paper. In its most basic form, participants from each of two sides of a matching market (e.g., men and women, hospitals and residents) express preferences—in the form of a ranking—over those on the other side. The aim is to find a bipartite matching (e.g., one-to-one in marriage, many-to-one in resident matching) that is game-theoretically *stable*, i.e., robust to deviation by any couple/pair. The problem has wide applicability, including school choice and matching in residency (or other labor) markets (Niederle, Roth, and Sonmez 2008). Since Gale and Shapley's (1962) *deferred acceptance algorithm (DFA)*, numerous algorithmic developments have extended the utility of this model.

One impediment to the use of DFA (and its extensions) is the requirement that agents specify their *complete preferences*, in the form of a total order (or preorder) for their counterparts on the other side of the market. This creates two difficulties. First, ranking all options imposes a significant cognitive burden on agents, requiring a large number of (say) pairwise comparisons. Many of these may in fact be unnecessary to the discovery of a stable matching. *Preference elicitation* schemes, including DFA, that incrementally elicit only the preference information deemed to be necessary to construct a stable matching, can help alleviate this burden. In the sequel, we draw heavily on the partial preference representation and minimax-regret-based elicitation scheme devised in our previous work (Drummond and Boutilier 2013) (we refer to this model and method as DB).

A second difficulty is that, even with efficient elicitation schemes, agents may be unable to rank two options without engaging in additional *information gathering* activities. In marriage markets, this might take the form of dating, while in labor markets, this takes the form of *interviews*. Indeed, the National Residency Matching Program (NRMP) has served as a centralized clearing house since 1952 to determine stable matchings of hospitals and residents in the US (NMRP 2013). However, preference assessment is very costly—residents spend $1,000 to $5,000 to interview with an average of 11 programs each (Anderson et al. 2000). Since residents only rank the programs with whom they interview, rankings submitted to NRMP will be strongly influenced their choice of interviews.

Given their costly nature, *interview minimization* is critical. Rastegari et al. (2013) address this problem, showing that interview minimization is computationally intractable, but developing a feasible method that is tractable if certain (rather restrictive) preference information is provided *a priori*; we refer to this model as RCIL in the sequel. Two weaknesses of this approach are: the limited circumstances in which it works; and lack of assessment of the elicitation requirements needed to provide the "prior" preferences.

In this work, we provide a unified model that allows for the assessment of preferences using *both direct queries and interviews*. Specifically, we assume that direct comparisons can be made by agents if two options are sufficiently distinct (i.e., well-spaced) in their *true (but partially latent)* underlying ranking; but that such a comparison requires interviews of both options if they are close in their ranking. Our method uses *minimax regret* to determine robust solutions to matching problems given partial information, and generates (a heuristically minimal number of) queries and interviews that are guaranteed to reach a stable matching. In doing so, we: (i) extend the preference representations of both DB and RCIL; (ii) extend the minimax-regret based approach of DB to accommodate interviews; and (iii) provide a tractable (polytime) scheme for generating interviews (and

queries) that deals with more general partial preferences that RCIL. Experimental results demonstrate the effectiveness of our scheme.

Background

We review *stable matching problems (SMPs)*, focusing on one-to-one matchings (e.g., marriages) for ease of exposition. We also discuss recent work on SMPs with partial preferences, preference elicitation, and interview minimization, explicating key concepts upon which our work is based.

Stable Matching Problems. We describe one-to-one SMPs using stable marriage terminology. Assume a set of men M and women W, each of size n. Each person q provides a strict total ordering $\succ_q$ over the set $R(q)$, where $R(q)$ is the "opposite side" of the market (e.g., if $q \in W$, then $R(q) = M$) and $a \succ_q b$ means q prefers a to b. Let $\succ$ be a *preference profile* consisting of a ranking for each $q \in M \cup W$. A *matching* $\mu : M \cup W \rightarrow M \cup W$ requires $\mu(w) \in M, \forall w \in W, \mu(m) \in W, \forall m \in M$, and $\mu(w) = m$ iff $\mu(m) = w$. A *blocking pair* for μ is a pair (m, w) s.t. $w \succ_m \mu(m)$ and $m \succ_w \mu(w)$ (i.e., m, w prefer to be with each other than their partners in μ). We say μ is *stable* if it admits no blocking pairs.

The *deferred acceptance algorithm (DFA)* is a polynomial time algorithm for SMPs (Gale and Shapley 1962). Briefly, the (female-proposing) algorithm proceeds as follows: initially, everyone is unmatched. During each round, all unmatched women propose to the man highest in their ranking to whom they have yet to propose. A man receiving multiple proposals (including any tentative partner) accepts his most preferred proposal (tentative partner). This continues until all women are tentatively matched, and this final tentative match μ is returned. Despite its simplicity, DFA always returns a stable matching, and has many important properties (e.g., it is proposer optimal and strategy-proof). DFA can be applied to many-to-one problems with a variety of real-world applications (Roth 1984; Abdulkadiroglu et al. 2005) and extends to handle richer forms of preferences (e.g., indifference, acceptability cutoffs).

Matching and Preference Elicitation. As discussed above, the burden of providing a complete ranking of all potential partners can be considerable in large-scale matching markets. Recent work has considered eliciting *partial information* about agent preferences, just enough to ensure a stable match is found. In principle, DFA can be used as an elicitation scheme, but rarely done so in practice. Biró and Norman (2012) propose a stochastic matching technique that can be interpreted as an elicitation scheme. However, it scales poorly and requires far more rounds than the DB scheme we adopt below (Drummond and Boutilier 2013).

Drummond and Boutilier (2013) propose a method for computing robust matchings, using the notion of *minimax regret*, that are approximately stable given partial information about agent preferences in the form of pairwise comparisons. They use this to drive an elicitation scheme that asks "relevant" queries of agents to help determine stable matchings with relatively little preference information—typically $\log_2(n)$ queries per agent on average, much less than elicited by DFA. We describe this model (hereafter dubbed DB) in detail, since we draw on it later.

A *partial preference ranking* P_q for agent q is a consistent set of pairwise comparisons (i.e., partial order) over $R(q)$. We say P_q are *partitioned preferences (PP)* if it partitions $R(q)$ into subsets or *blocks* $G_1, \ldots, G_k$ s.t. $g_i \succ_q g_j$ for any $g_i \in G_i, g_j \in G_j, i < j$, and all $g_i \in G_i$ are uncompared. Let $C(P_q)$ be the set of all consistent completions of P_q. Such partial preferences reflect *incompleteness of the knowledge of the matching mechanism*, hence the set of completions $C(P_q)$ reflect the possible realizations of q's preferences from the perspective of the mechanism. In the DB model, one assumes all agents q have a complete *underlying* ranking $\succ_q$; and that q can compare any two alternatives r and r' (i.e., determine whether $r \succ_q r'$ or $r' \succ_q r$) without gathering additional information. Of course, certain comparisons may be more computationally or cognitively demanding than others; indeed, DB also consider such cognitive costs, as we do below.

If the mechanism must select a matching given a profile $\mathbf{P}$ of *partial* preferences, stability may not be guaranteed. DB use max regret to define the potential *degree of instability* of a matching, assuming a worst-case completion of the profile, and propose *minimax regret* as a robustness criterion for matching. Let $s(r, \succ) = n - rank(r, \succ)$ be the *(inverse Borda) score* of r in ranking $\succ$. DB define:

$$Regret(q, r', r, \succ_q) = s_q(r', \succ_q) - s_q(r, \succ_q) \quad (1)$$

$$PMR(q, r', r, P_q) = max_{\succ_q \in C(P_q)} s_q(r', \succ_q) - s_q(r, \succ_q) \quad (2)$$

$$Inst(m, w, \mu, \succ_m, \succ_w) = \quad (3)$$
$$\min\{PMR(m, w, \mu(m), \succ_m), PMR(w, m, \mu(w), \succ_w)\}$$

$$Inst(\mu, \succ) = \max_{(m,w)} Inst(m, w, \mu, \succ_m, \succ_w) \quad (4)$$

$$MR(\mu, \mathbf{P}) = \max_{\succ \in C(\mathbf{P})} Inst(\mu, \succ) \quad (5)$$

$$MMR(\mathbf{P}) = \min_{\mu} MR(\mu, \mathbf{P}); \quad \mu^*_{\mathbf{P}} \in \operatorname{argmin}_{\mu} MR(\mu, \mathbf{P}). \quad (6)$$

The *maximum regret* $MR(\mu, \mathbf{P})$ of matching μ given partial profile $\mathbf{P}$ is the maximum incentive any blocking pair has to defect from μ, under any realization of preferences consistent with the partial profile $\mathbf{P}$. The minimax-optimal matching μ^* minimizes this max regret. If $MMR(\mathbf{P}) = 0$, then μ^* is in fact stable, despite the incompleteness in preferences. While computing μ^* is NP-complete, DB devise polytime methods, known as *PPGS*, that offer excellent results in practice.

DB also describe an elicitation scheme that uses the MMR-solution to guide the querying process. Agent partial preferences are always of the partitioned form, and at each stage, one or more agents is asked to refine one of its blocks by dividing it into a "top half" (more preferred) and "bottom half" (less preferred). They show this scheme to be quite effective, but again it assumes that agents can answer queries without additional information gathering.

Interview Minimization. Rastegari et al.(2013) (hereafter RCIL) investigate interview policies that minimize total interview costs for SMPs from a theoretical perspective. Much like the DB elicitation method, their aim is to find an interview policy that, beginning with a partial profile, will

provide enough preference information to ensure a stable matching can be computed. RCIL define an *interview* to be an information gathering step taken by two agents (say m, w), one on each side of the market. After the interview, m (resp. w) is able to rank all options they have interviewed with, including w (resp. m). In other words, if (say) q has interviewed with both r and r', then q can definitively answer whether $r \succ_q r'$ or $r' \succ_q r$. This is the form of interview we assume in this work as well.

RCIL show the general problem of interview minimization is NP-hard, even with the PP model, and propose an MDP formulation to solve the problem (though one that is quite impractical). They also consider the restricted case when the partitioned preferences of agents on one side of the market have *identical structure* (i.e., identical blocks), and provide a poly-time algorithm *Lazy Gale-Shapley (LGS)* to compute an interview-minimizing policy. The method provides a proposer-optimal stable matching. However LGS restricts the true underlying preferences, requires small blocks (i.e., considerable prior information) to ensure small number of interviews, and does not analyze the method or costs needed to assess the "prior" preferences.

Combining Elicitation and Interviews

In many settings, the practical assessment of preferences in matching markets requires both direct preference elicitation—asking agents to compare or rank options about which they have adequate information—and interviews—allowing agents to determine the properties of specific options needed to make such comparisons. We outline a framework and elicitation scheme for doing just this. Our scheme is a direct extension of the DB elicitation method, allowing agents to express *uncertainty* in the response to certain queries *prior to interviewing with specific options*. It can also be viewed somewhat loosely as an extension of the RCIL/LGS scheme, since it does not rely on restrictive preference assumptions of LGS; and it provides a means for acquiring the prior preferences needed by the LGS scheme to compute the minimal set of interviews.

We first describe our preference query model and assumptions about agents' ability to answer queries without interviews, as well as the representation of preferences. We then describe a method for approximating minimax regret given a partial profile. Finally, we describe a combined elicitation-interview protocol, *Refine-then-Interview (RtI)*, that in some sense unifies the DB and RCIL models.

Overlapping Partitioned Preferences

The DB model for representing partial preferences (and computing minimax regret) supports arbitrary pairwise comparisons. However, their elicitation scheme is based on *halving queries*, which ensures that each partial preference is in fact partitioned. Let agent q have true (latent) preference $\succ_q$ over n options, and let the mechanism's knowledge of $\succ_q$ be a PP $G_1, \ldots, G_k$. A halving query asks q to split one of the blocks G_j into a more preferred half G_j^+ and a less preferred half G_j^-, leading to a refined PP $G_1, \ldots, G_{j-1}, G_j^+, G_j^-, \ldots, G_k$. This scheme assumes that q is able to answer arbitrary pairwise comparisons to refine any block of the partition.

While many pairwise comparisons are relatively straightforward, others cannot be assessed without engaging in an interview. To model this, let $w \leq n$ be a *window size*. We assume q is able to state whether option a is preferred to b if $|s(a, \succ_q) - s(b, \succ_q)| \geq w$ in the q's latent ranking $\succ_q$; otherwise we assume that interviewing with both options is needed to distinguish them. For example, if the attributes of a and b are sufficiently distinct (e.g., East Coast vs. West Coast hospital), then q's preference may be obvious; but if they are similar, then q may need interviews to rank one over the other. Having rank-distance determine the difficulty of comparison (or odds of choosing incorrectly) is commonly assumed in econometrics and psychometrics as well (e.g., as in the Luce-Shepard choice model (Luce 1959; Shepard 1959)).

In our query model below, we ask halving queries as in DB. However, we assume that when splitting a block G_k of size $g > w$, q is able to determine the $(g-w)/2$ options she *knows* to lie in the top half G_k^+ of G, and the $(g-w)/2$ options in the bottom half G_k^- but the remaining w options form an uncertain *middle tier*, $G_k^?$, all elements of which could lie somewhere in the top or bottom half of G_k. We assume that distinguishing any two of these options from each other—and from the $w/2$ least-ranked elements of G_k^+ and the $w/2$ top-ranked elements of G_k^-—requires interviews. In particular, prior to interviews, q believes the elements of $G_k^?$ lie in any of the middle $2w$ positions of G_k.

We fix the window size w, and assume that the top and bottom partitions have the same size, i.e., $|G_k^-| = |G_k^+| = (g-w)/2$, for all agents and all queries. This is merely for ease of exposition. Neither of these assumptions impact the algorithms below, and a model where the window size and precise location of the resulting split varies across agents—and for a given individual, across queries—can be addressed using the same techniques.

Our representation, the *overlapping partitioned preferences (OPP) model*, maintains a set of blocks, as in the partitioned case, but also the set of *eligible positions*, $p(G_k) = (t(G_k), b(G_k))$, that elements of any block G_k can occupy in the true ranking $\succ_q$ (where $t(G_k), b(G_k)$ are the top and bottom such positions). A response to a halving query for G_k (with mid-point $m(G_k)$) thus refines that block into three blocks with the following sizes and allowable positions:

$$|G_k^+| = (|G_k| - w)/2, \quad p(G_k^+) = [t(G_k), m(G_k) - 1]$$

$$|G_k^?| = w,\ p(G_k^?) = \begin{cases} [m(G_k) - w, m(G_k) + w], \text{if } |G_k| \geq 2w \\ [t(G_k), b(G_k)], \text{otherwise} \end{cases}$$

$$|G_k^-| = (|G_k| - w)/2, \quad p(G_k^-) = [m(G_k), b(G_k)]$$

We define $t(a) = t(G_k)$ and $b(a) = b(G_k)$ for all $a \in G_k$, i.e., denoting the top and bottom positions an option a can take (given the block it occupies).

These constraints imply that no block of size less than $w + 2$ can be halved. To refine agent preferences further, the mechanism proposes a bidirectional *interview*, which allows agents to determine their relative preference for "close" options. For any agent q on one side of the market, let $I(q)$

be the set of options (from the other side) with who she has interviewed. We assume that q is able to totally order all elements of $I(q)$, so that $\succ_{I(q)} \subseteq \succ_q$. Furthermore, we require $p \in I(q)$ iff $q \in I(p)$. When OPP is complemented by the interview ordering constraints $\succ_{I(q)}$, we call this *overlapping partitioned preferences with interviews (OPPI)*.

Approximating Minimax Regret. As in the DB approach, we use *minimax regret (MMR)* to compute robust matchings given a partial preference profile, where our profiles take the OPPI form. We will use these solutions to drive the querying and interviewing process in the next section. Of course, since DB show computing MMR is NP-hard even for PP, and since OPP and OPPI include PP as a special case, the problem remains NP-hard in our model. Following DB, we adopt the following strategy: given an partial profile $\mathbf{P}$ in OPP or OPPI form, we assign each agent q some complete ranking $\succ_q$ consistent with their partial preference P_q. (We discuss completion functions below.) We then run the GS algorithm on this completion to determine a stable matching μ. The max regret $MR(\mu, \mathbf{P})$ of μ provides an upper bound on $MMR(\mathbf{P})$ (repeating with multiple completions can tighten the bound). DB show this *PPGS* method is tractable by proving that pairwise max regret $PMR(q, r', \mu(q), P_q)$, see Eq. 2, is computable in polytime. These PMR terms can then be used to compute $MR(\mu, \mathbf{P})$ using Eq. 5 over the $O(n^2)$ potential blocking pairs.

Unfortunately, computing $PMR(q, r', r, P_q)$ is somewhat more straightforward in the PP model than in OPP or OPPI. Thus our primary goal is to show that PMR can be computed efficiently in both models. We begin with OPP. For our purposes, we need accurate calculation only when $PMR \geq 0$ (if $PMR < 0$ we "cut it off" at 0). Using a variant of Hall's Theorem (see Demange, Gale, and Sotomayer 1986), we show, given OPP G_q, that $PMR(q, r', r, G_q) = \max(b(r) - t(r'), 0)$. Intuitively, we recast PMR computation as an allocation problem, where the options "bid" for a rank position, bidding on all eligible positions given their assigned bounds. Each option must win exactly one spot. The two options used to compute PMR are set by the adversary, and thus are pre-assigned their spots, which is represented by a transformation function that removes the pre-assigned alternatives and now-unavailable spots. We then show that there are no over-demanded sets after the transformation. A full proof can be found in an extended version of the paper.[1]

PMR, hence $MR(\mu, \mathbf{P})$, can also be computed in polynomial time for OPPI, requiring that we account for the interview ranking constraints. We calculate PMR for an agent q with OPPI profile G_q and interview set $I(q)$ via a subtractive counting argument, where we calculate the maximal possible distance between two options using positional information, and then reduce this distance to account for interview-related information. For clarity, we present only the core of the algorithm, deferring analysis of special cases to the extended version of the paper.

We calculate PMR by computing *segments* S_q, a complementary set to G_q; while each block in G_q is a disjoint set of alternatives (with overlapping positions), each segment in S_q is a disjoint set of positions (with overlapping alternatives). Alg. 1 shows how to calculate these segments. (For ease of exposition, we assume $|G_k| \geq w,\ \forall G_k \in G_q$; see extended version of the paper describes details for other cases).

Algorithm 1 Constructing Segments ($|G_k| \geq w,\ \forall k$)

Require: $G_q, I(q)$
1: **for** $G_k \in G_q$
2: **if** k is odd
3: mid $= \sum_{i=0}^{k-1} |G_i| + |G_k|/2$
4: $t(S_{G_k^-}), b(S_{G_k^-}) =$ mid $- w,$ mid $- 1$
5: $t(S_{G_k^+}), b(S_{G_k^+}) =$ mid, mid $+ w - 1$
6: **else**
7: prev_mid $= \sum_{i=0}^{k-2} |G_i| + |G_{k-1}|/2$
8: next_mid $= \sum_{i=0}^{k} |G_i| + |G_{k+1}|/2$
9: $t(S_{G_k^-}), b(S_{G_k^-}) =$ prev_mid, prev_mid $+ w - 1$
10: $t(S_{G_k^+}), b(S_{G_k^+}) =$ next_mid $- w,$ next_mid $- 1$
11: $S_{G_k^-} =$ generate_seg$(G_k, t(S_{G_k^-}), b(S_{G_k^-}))$
12: $S_{G_k^+} =$ generate_seg$(G_k, t(S_{G_k^+}), b(S_{G_k^+}))$
13: $S_q[G_k] = S_{G_k^-}, S_{G_k^+}$
14: **def** generate_seg$(G_k,$ top, bottom)
15: //Generates segment s
16: $t(s) =$ top, $b(s) =$ bottom
17: s.dom $= \{G_x$ s.t. $t(G_x) \leq b(s)$ and $b(G_x) \geq t(s)\}$
18: s.boundaries $= \{(\max(t(s), t(G_x)), \min(b(s), b(G_x))$ s.t. $G_x \in$ dom$\}$
19: s.required $=$ populate_required(s)
20: s.ordered_req $= s$.required, s.t. it's consistent with $I(q)$
21: **def** populate_required(s)
22: push_up $= G_{s.\text{dom}[0]}$
23: push_down $= G_{s.\text{dom}[1]}$
24: s.required$[dom[-1]] = |\{x \in I(q) \cap G_{s.\text{dom}[-1]}$ s.t. $\exists y \in$ push_up s.t. $x \succ_{I(q)} y\}|$
25: s.required$[dom[0]] = |\{x \in I(q) \cap G_{s.\text{dom}[0]}$ s.t. $\exists y \in$ push_down s.t. $y \succ_{I(q)} x\}|$

Segments are generated by first identifying the extremal, overlapping portions of each block, and then characterizing them. Each block has two extremal positional segments—an upper and lower segment (corresponding to its upper and lower bounds). When $|G_k| \geq w,\ \forall G_k \in G_q$, each segment will be exactly of size w (when $|G_k| < w$, then $|s| > w$ for one of its segments s). We identify these extremal positions using the partial preference structure imposed by our elicitation scheme (Lines 2–13), though these upper and lower positional bounds could be given as input. In our setting, the uncertain blocks ($G_j^?$ when halving block G_j) partition the extremal segments. (All uncertain blocks have an odd index; Lines 2–5.) All other blocks have their extremal positions defined by the odd blocks as well (Lines 7–10).

Once the positions are determined (or given) for a segment, then all characteristic information is generated. The *domain* of the segment is every block G_x s.t. G_x overlaps the segment's bounds (Line 17). Each block in the domain has *boundaries* associated with it that are the subset of positions an element in G_x is allowed to take in the segment—some G_x may have tighter bounds than the segment itself (Line 18). Most importantly, some constraints in $I(q)$ *require* that certain elements from some domains must be placed in this

[1] See www.cs.toronto.edu/~cebly/papers.html.

Algorithm 2 Calculating PMR for OPPI ($|G_k| \geq w,\ \forall k$)

Require: Agent q's OPPI profile G_q, with interviews $I(q)$, segments S_q, reversed input q^{-1}, and alternatives a, a'

1: **if** $|I(q)| < 2$
2: $PMR(q, a', a, G_q) = \max(b(a) - t(a'), 0)$
3: **else**
4: **if** $a \succ_{I(q)} a'$ or $b(a) < t(a')$
5: $PMR(q, a', a, G_q) = 0$
6: **else if** $k < j$
7: Edge case (see extended version)
8: **else**
9: $r(a') = n - 1 -$ find_lower_bound(a', q^{-1})
10: $r(a) =$ find_lower_bound(a, q)
11: $PMR(q, a', a, G_q) = \max(r(a) - r(a'), 0)$
12: **def** find_lower_bound(a, q)
13: **if** $a \notin I_q$
14: seg_up, seg_down $= S_q[G_k]$
15: **if** placing a in seg_down is legal
16: **return** $b(a)$
17: **else** //place in the bottom of upper segment
18: **return** b(seg_up)
19: **else** //$a \in I(q)$
20: **if** a required to be in seg_up
21: lowest_a $= b$(seg_up) $- |\{x \in I(q)$ s.t. $a \succ_{I(q)} x$ and $p(x) \geq b$(seg_up)$\}|$
22: holes = # spaces not already required via $I(q)$ between b(seg_up), b(seg_down)
23: **else**
24: $s =$ seg_down
25: lowest_a $= b(a) - |\{x \in I(q)$ s.t. $a \succ_{I(q)} x$ and $x \in$ seg_down.req$\}|$
26: holes = 0
27: free = $|G_{\text{seg\_down.dom}[-1]} - I(q)|$
28: mand = $|$seg_down.req.dom$[-1]|$
29: **return** lowest_a - $\max(0, w/2 -$ (holes + mand + free)

segment. This is what allows for us to quickly count the interview constraints when calculating PMR. Lines 21–25 describe the method for finding these required elements. Intuitively, given some x in the less desirable block, and y in the more desirable block, if $x \succ_{I(q)} y$, both x and y *must* be in this segment.

We now examine these extremal segments to calculate $PMR(q, a', a, G_q)$ given chosen option a and adversarial selection a', as shown in Alg. 2 (again, see the extended paper for the full algorithm). Note that, if $|I(q)| < 2$, calculating PMR for OPPI reduces to calculating PMR on OPP preferences (Lines 1–2). Otherwise, we check if $PMR(q, a', a, G_q)$ is forced to be zero either given available positional information (a' *must* be placed below a) or interview information ($a \succ_{I(q)} a'$) (Lines 4–5). If not, the algorithm consists of two main cases using the segments: a is in a more desirable block than a', and the maximal positions of a and a' must be calculated simultaneously; or, the maximal position of a and a' can be calculated independently (this is a simple, but tedious, argument by cases to ensure that no options are double-counted). We note that calculating the best position a' can take is equivalent to calculating the worst position a' can take for an agent q^{-1} with "reversed" preferences (Lines 9–11). Thus, the remainder of the PMR algorithm involves calculating this worst position, which uses similar intuitions to those used to calculate a and a''s positions simultaneously (some edge cases must be considered to ensure that no double-counting occurs).

Computing the maximum (worst) position a can take (Lines 12–29) consists of two main cases: $a \notin I(q)$ (i.e., a not interviewed) and $a \in I(q)$ (i.e., a interviewed). If $a \notin I(q)$, there are fewer constraints on a's placement, though we still need to guarantee that a *valid* ranking exists given a's assigned placement. We want to place a as low as possible in its bottom extremal segment. However, if $I(q)$ requires too many options to be placed in this bottom segment, we place a at the bottom of its top segment (Line 18). Otherwise, there is room in the bottom segment, and with no constraints on a's placement imposed by $I(q)$, we place a at its maximal position (Line 16).

When $a \in I(q)$, we first compute the number of options that must lie between a and $b(a)$ dictated by $I(q)$. We then count to check that placing a there still results in a valid ranking; we subtract positions from a's maximal position (i.e., move a higher in the ranking) until a valid ranking is obtained. Since $a \in I(q)$, either a is required to be in the upper extremal segment given interview information $I(q)$ or this is not implied by $I(q)$. The former occurs when there is some option x whose positional constraints require it to be in the upper segment, and $a \succ_{I(q)} x$. In either case, we compute the lowest position a is allowed to take given $I(q)$, a's *maximal* position (Lines 21 and 25). As discussed above in segment generation, $w/2$ options from the least-desirable block whose positional information allows its members to be placed in the bottom segment *must* be placed in that segment (otherwise a valid ranking does not exist). The remainder of the algorithm verifies that enough of the options from this least-desirable block can be placed in the lower segment. Since we need to ensure that there is space in the lowest segment for the $w/2$ options from the least-desirable block, we may need to move a up in the ranking by as many as $w/2$ positions from its maximal position. However, there are different ways in which we could place these $w/2$ options that do not affect a's placement. Counting these ways, as we describe below, allows the accurate computation of the lowest position that a can take while guaranteeing a valid ranking is constructed.

If $a \in I(q)$, the number of positions between a's placement and $b(a)$ may not form a "dense set" (i.e., $I(q)$ may not require that all of these positions be filled; Line 22). Such "holes" allow us to place more of the $w/2$ required elements in the bottom segment without changing a's position. Furthermore, options in the least desirable block that do not have their order fixed by $I(q)$ (i.e., they have not been interviewed) can be placed above a, again not changing a's position (Line 27). All options that are required to be in the segment have already been counted (via the number of "holes" and the lowest legal position of a), and thus we must not double count them (Line 28). If all of these options account for the $w/2$ required positions, the lowest possible position computed for a is valid and is returned. Otherwise, we subtract from this the extra positions required to place the $w/2$ options in the lowest segment, ensuring a valid ranking exists (Line 28). As we show in the extended version of the

paper, this results in a method that allows $PMR(q, r', r, G_q)$ to be computed in $O(n^3)$ time.

Finally, we require an efficiently computable completion function in order to use PPGS to approximate MMR. Given OPPI profile $\mathbf{P}$, we assume a *reference ranking* σ_M (resp., σ_W) over each side of the market.[2] Our completion function places, for any q, all interviewed options in $I(q)$ in the highest allowable position that results in a valid ranking, and fills remaining positions with uninterviewed options within the relevant blocks (ties are broken using σ). More formally: (i) Sort all blocks according to σ. (ii) For each position p, if no interviewed options can be placed at p, place the next uninterviewed option at p. Otherwise, count the number of uninterviewed options r that must lie before the next block boundary $b(G)$. If $r + 1 < b(G) - p$, place an interviewed option at p. Otherwise, place an uninterviewed option at p. This completion takes polynomial time.

Elicitation Scheme

We now define our *Refine-then-Interview (RtI)* elicitation scheme. Given a current OPPI profile $\mathbf{P}$, RtI computes an (approximately) regret-minimizing matching μ. This solution is used to guide the choice of queries or interviews at each round with the aim or reducing MMR as much as possible. The scheme focuses on *Regret Inducing (RI)* agents in μ, those whose max regret (or instability) dictates the max regret $MR(\mu)$ of the matching. By gathering additional information, through elicitation or interviews, about their preferences or those of their blocking partners, we will reduce the regret of μ, and ideally MMR as well.

At any point we can ask an agent to either halve one of their blocks (which we take to be less costly) or engage in one or more interviews (which we take to be more costly). Of course, halving only works to the point where a block has been refined to size w; after that, interviews are required to make further ranking distinctions. Thus, RtI, see Alg. 3, prioritizes queries over interviews. Let $RI(\mu)$ denoting the set of all regret-inducing individuals in μ. For any $r \in RI(\mu)$, let $BP(r)$ be the set of r's potential blocking partners. Intuitively, RtI uses halving queries to determine roughly where in their preference ranking each agent will be matched. Once their preferences have been refined in the "relevant" regions of their rankings to the extent possible without interviews, RtI proposes interviews for the few options with whom they could still form blocking pairs.

Since halving queries split a block into three smaller blocks, we expect between $\log_3(n)$ and $\log_2(n)$ queries per agent. However, we expect the number of interviews to be *independent* of n (the size of the market) and depend only on w (the degree to which agents can make comparisons without interviews), in fact, to be at most approximately $2w$. This is due to the fact that halving will reduce the "relevant" blocks in any agent q's preference (i.e., blocks containing viable partners) to size no greater than w; and at most two additional blocks (one of which must be of size less than w) can contribute options to such a block.

[2]When using Mallows models (see below), we use the reference ranking defining the Mallows model (i.e., the ranking with maximum prior probability). Otherwise, we use an arbitrary ranking.

Algorithm 3 Refine-then-Interview Elicitation Scheme

Require: OPPI profile $\mathbf{G}$, threshold τ

loop

Compute (approximate) $\mu = \mu^*(\mathbf{G})$, compute $MR(\mu, \mathbf{G})$.
if $MR(\mu, \mathbf{G}) \leq \tau$, done.
for each $q \in RI(\mu)$ s.t. q not queried this round
  $Q = \{g_k \in G_r$ s.t. $|g_k| > w + 1 \wedge \exists b \in BP(r) \cup \{\mu(r)\}$ s.t. $b \in g_k\}$
  Query (halve) every block in Q
  if $|Q| = 0$ (i.e., no blocks were halved)
    for $b \in BP(r)$
      if b not queried this round:
        $Q = \{g_k \in G_r$ s.t. $|g_k| > w + 1 \wedge (r \in g_k \vee \mu(b) \in g_k)\}$
        Query (halve) every block in Q
if no blocks were halved this round
  for $r \in RI$
    if r not interviewed with this round
      r interviews with $BP(r) \cup \{\mu(r)\}$
      if r did not interview ($BP(r) \cup \{\mu(r)\} \subseteq I(r)$)
        for $b \in BP(r)$
          b interviews with $\mu(b)$ ($r \in I(b)$)

Empirical Evaluation

We evaluate RtI on a variety of randomly generated matching problems, using several different probabilistic models as well as preferences derived from real-world ratings data. All results are reported over 20 random matching instances.

Comparison to LGS. RtI and LGS solve slightly different problems, since LGS requires very specific (more restrictive) prior preferences, but at the same time finds proposer-optimal matchings (while RtI finds some stable matching). Nevertheless, we show that with the same prior information, RtI generates almost as few interviews as LGS, and that it can more effectively assess "prior" preferences.

Using markets of size $n = 124, 252$, and a window size of $w = 4$, we first compare the two using partitioned preferences of the form needed by LGS: women's (employers') preferences are drawn from a Mallows ϕ-model (Mallows 1957; Marden 1995) with dispersion ϕ and reference ranking σ, and assign w options to each of n/w blocks. Men (applicants) must have identical blocks, so we first partition all options into n/w blocks of size w, then create each man's true preference *within each block* by drawing a (smaller) ranking from a Mallows distribution (same ϕ and projected σ). Results for varying dispersion values are shown in Table 1. We see that LGS and RtI generate similar numbers of interviews, with RtI averaging less than one interview per person more than LGS, despite its broader applicability (we note of course that LGS is providing female (employer) optimal matchings and RtI is not). RtI also requires far fewer rounds of interviews, meaning that many interviews can be run in *parallel*; i.e., individual participants can work through their interviews without for additional input from the matching mechanism (e.g., waiting for *other* participants to complete interviews). LGS by contrast must run a large number of interviews sequentially (using prior interview results before setting interviews at the current round).

n	ϕ	RtI interviews	LGS interviews	RtI rounds	LGS rounds
124	0.2	3.93 (0.09)	3.66 (0.07)	7.4 (1.3)	154.3 (6.0)
124	0.6	3.19 (0.18)	2.42 (0.09)	12.2 (2.9)	163.9 (3.4)
124	1.0	3.08 (0.16)	2.37 (0.06)	13.8 (2.7)	130.0 (1.8)
252	0.2	3.92 (0.05)	3.64 (0.05)	8.8 (0.9)	314.6 (0.1)
252	0.6	3.24 (0.16)	2.41 (0.09)	15.1 (3.0)	336.6 (7.8)
252	1.0	3.03 (0.11)	2.31 (0.04)	17.2 (2.1)	258.5 (3.2)

Table 1: RtI vs. LGS (identical input): interviews/person (std.).

We also compare RtI to LGS by analyzing the "total information" requirements of the algorithms. To do so, we provide LGS the strict blocks it needs (as above), but provide RtI with no prior preference information. We compare the number/cost of RtI's queries with the queries/cost needed to produce LGS's prior blocks, as well as the number of interviews. Table 2 shows that RtI performs well w.r.t. LGS. While RtI generates roughly twice as many interviews as LGS, the strict block boundaries provided to LGS require pairwise comparisons that in the RtI model cannot be assessed without interviews (hence, this comparison is misleading). Despite the fact that RtI starts with no preference information (in contrast to LGS's blocks), RtI takes significantly fewer rounds of combined elicitation/interviews than LGS needs for only interviews, except when $\phi = 1$ (uniformly distributed preference rankings).

Finally, we compare the (non-interview) elicitation requirements of both. The number of halving queries used by RtI, as well as their *cognitive cost* is show in the table. Here we measure the cognitive cost of a pairwise comparison using a Luce-Shepard model, see (Drummond and Boutilier 2013). Given preferences $\succ_q$, temperature $\gamma \geq 0$, and threshold τ, the cost for q to compare r with r' is:

$$c(r, r') = e^{\gamma(n - \min(|s_q(r,\succ_q) - s_q(r',\succ_q)|,\tau))} \quad (7)$$

We set $\gamma = 0.5$, $\tau = 5$, and normalize reported cognitive cost by $e^{(\gamma n)}$ (as in DB). To assess the cost of creating LGS blocks, we assume that agents can engage in partial Quicksort to create blocks and have access to "perfect" pivots at block boundaries. Again, this assumes agents can accurately compare two "close" options without interviews (something not allowed in RtI). RtI's cost is computed similarly, but unanswerable comparisons have cost 0. Eliciting LGS's prior preferences, when $n = 124$ (resp., 252), requires 586 (resp., 1445) comparisons, with a cognitive cost of 107.58 (resp., 241.61). Table 2 shows that the cost of RtI's queries is roughly 16% of that of LGS, and RtI needs about 55% fewer comparisons than LGS. Thus, while LGS requires fewer interviews—though we emphasize that LGS obtains information *without* interviews that force interviews in our model—it needs significantly more preference information overall.

Evaluation of RtI. We now evaluate RtI using random matching problems with preferences generated using several different probabilistic preference models, varying both the market size n and window size w. The models are the same as those used by DB: the Mallows ϕ-model (Mallows 1957); a riffled independence model (Huang and Guestrin 2009) derived by riffling two Mallows models; and preferences derived from MovieLens ratings data.[3] In all instances, RtI begins with no preference information.[4]

n	ϕ	RtI interviews	LGS interviews	RtI queries	RtI cog cost	RtI rounds	LGS rounds
124	0.2	8.81 (0.26)	3.66 (0.07)	4.63 (0.17)	17.8 (0.3)	92.3 (11.1)	154.3 (6.0)
124	0.6	7.57 (0.30)	2.42 (0.09)	4.88 (0.19)	18.1 (0.4)	125.9 (12.9)	163.9 (4.1)
124	1.0	4.94 (0.15)	2.37 (0.06)	6.35 (0.07)	21.1 (0.1)	199.1 (22.9)	123.0 (1.8)
252	0.2	8.95 (0.23)	3.64 (0.05)	5.85 (0.22)	38.8 (0.6)	135.3 (9.0)	314.6 (7.1)
252	0.6	7.60 (0.26)	2.41 (0.09)	6.17 (0.3)	39.7 (0.8)	213.4 (22.4)	304.2 (7.8)
252	1.0	4.78 (0.16)	2.31 (0.04)	8.69 (0.1)	48.3 (0.4)	488.4 (48.0)	258.5 (3.2)

Table 2: RtI (with queries) vs. LGS: interviews, etc./person (std.).

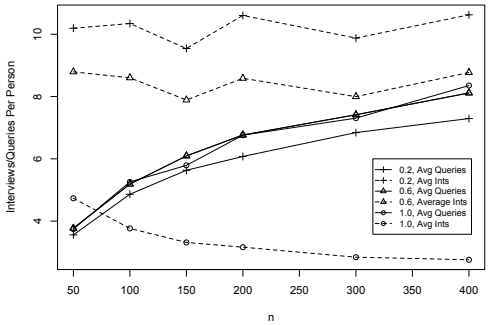


Figure 1: RtI: Avg. queries and interviews/person; varying n, ϕ.

We first test RtI on a Mallows distributions, varying the degree of preference correlation (or dispersion ϕ). Fig. 1 describes results for $w = 4$. As n increases, the number of interviews (dashed line) remains virtually constant. The exception is when $\phi = 1.0$ (i.e., no preference correlation, or impartial culture), where the number of interviews required *decreases*, which occurs due to more heterogeneous preferences. With $\phi = 0.2$, RtI generates about $2.5w$ interviews/person, consistent with the conjecture that it will require between w and $2w$ interviews. With highly correlated preferences, as expected, more interviews are required (this also occurs with LGS). The average number of halving queries per person (solid lines) increases logarithmically w.r.t. market size, and is unaffected by degree of preference correlation. This mirrors results in the DB query model. Table 3 shows results as we vary w ($n = 300$). As expected, the number of interviews increases with w (and peaks at about $2w$–$2.5w$ interviews/person when preferences are highly correlated), while the number of queries is roughly constant. We note that $w = 4, 6$ results in a number of interviews similar to the 11 that residents average in the NRMP (Anderson et al. 2000).

Since approximately stable matches may be sufficient for many real world problems, we show the anytime performance of RtI in Figs. 2a and 2b, plotting reduction in the max regret (MR) of the induced matching as rounds (either queries or interviews) progress ($n = 300$ and $\phi = 0.2, 1.0$). Each point represents MR after a round of RtI, vs. the number of queries/interviews to that point. We see that no

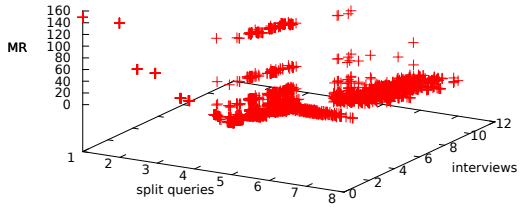
[3] See http://www.grouplens.org/node/73, the 100K data set.

[4] Error bars are omitted (too small to be seen).

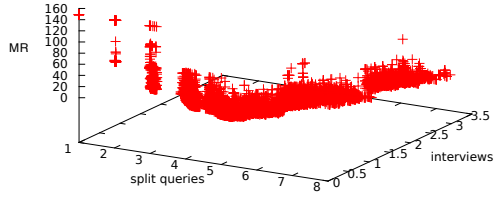
w	Interviews			Split Queries		
	$\phi = 0.2$	0.6	1.0	0.2	0.6	1.0
4	9.88	8.00	2.84	6.84	7.42	7.31
6	15.09	13.02	4.23	6.09	6.48	7.10
8	17.82	15.45	5.13	6.27	6.59	7.08

Table 3: RtI performance varying w; $n = 300$.

costly interviews are generated until MR has been significantly reduced via halving queries. Once MR is sufficiently small, interviews are used drive MR of the matching to 0 (i.e., true stability), though occasionally, new queries are required after some initial interviews (e.g., when the estimated matches are changed significantly after some interviews). These trends are more obvious in Fig. 2b, where few interviews are requested until 5 to 6 split queries per person have been asked, and after interviews begin, relatively few split queries are generated.



(a) $\phi = 0.2$



(b) $\phi = 1.0$

Figure 2: Anytime performance for RtI; $n = 300$.

Results on the riffle model, Fig. 3, exhibit the same trends as above. Comparing Mallows results with $\phi = 0.2$ to riffle results that merge two $\phi = 0.2$ models, we see fewer interviews are required in the riffle case; small perturbations in correlated preferences, combined with two "types" of preferences (as we would expect in real-world data) induces enough heterogeneity to reduce the number of interviews.

Finally, we apply RtI to the MovieLens preference model, with $n = 300$, where agent preferences are determined using an affinity score based on how similarly they rank movies (see (Drummond and Boutilier 2013) for details). With more

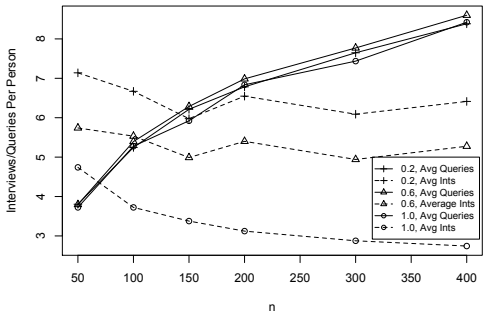


Figure 3: RtI (riffle model): Avg. queries and interviews/person; varying n, ϕ.

correlated affinities, RtI averages 4.95 interviews and 6.76 queries per person, while with less correlated affinities, it averages 2.34 interviews and 6.65 queries per person. These results, on a model based on real-world ratings data, are consistent with the observations above.

Conclusions and Future Work

We have developed a new elicitation scheme, which uses both queries and interviews, to assess agent preferences in stable matching problems. When compared with the interview-minimizing LGS algorithm (which works on restricted preference structures), RtI requires a similar number of interviews when given identical input, but generally requires significantly less overall preference information. Our comparison has focused on cases with relatively low preference uncertainty on the part of agents (i.e., small w). We hypothesize this level of uncertainty is realistic in markets of the size studied here, since it generates a number of interviews comparable to those seen in practice (Anderson et al. 2000). Of course, for markets with greater participant uncertainty, RtI would require more interviews, as would LGS or any other interview-minimizing scheme. RtI also scales well: the number of interviews increases with the agent uncertainty (w), not with market size, and appears to require about $2w$ interviews/person on all probabilistic models tested. The number of (less expensive) halving queries increases logarithmically with market size.

Many interesting questions remain. In many domains, agents naturally express their preferences using option attributes (not ranked lists), requiring an extension of our elicitation method to multi-attribute preferences (e.g., (Pilotto et al. 2009)). We also hope to analyze other probabilistic preference models, and account for other objectives (e.g., social-welfare-maximizing stable matchings) and constraints (e.g., matching with couples).

Acknowledgments. We acknowledge the support of NSERC. Drummond was supported by OGS and a Microsoft Research Graduate Women's Scholarship. Thanks to Ettore Damiano for helpful discussions and the reviewers for their suggestions.

References

Abdulkadiroglu, A.; Pathak, P.; Roth, A. E.; and Sönmez, T. 2005. The Boston public school match. *American Economic Review* 95(2):368–371.

Anderson, K. D.; Dorough, M. M.; Stein, C. R.; Optenberg, S. A.; Thompson, I. M.; et al. 2000. The urology residency matching program in practice. *The Journal of urology* 163(6):1878–1887.

Biró, P., and Norman, G. 2012. Analysis of stochastic matching markets. *International Journal of Game Theory* 1–20.

Demange, G.; Gale, D.; and Sotomayer, M. 1986. Multi-item auctions. *Journal of Political Economy* 94:863–872.

Drummond, J., and Boutilier, C. 2013. Elicitation and approximately stable matching with partial preferences. to appear.

Gale, D., and Shapley, L. S. 1962. College admissions and the stability of marriage. *American Mathematical Monthly* 69(1):9–15.

Huang, J., and Guestrin, C. 2009. Riffled independence for ranked data. In *Advances in Neural Information Processing Systems 21*, 799–807.

Luce, R. D. 1959. *Individual Choice Behavior: A Theoretical Analysis*. Wiley.

Mallows, C. L. 1957. Non-null ranking models. *Biometrika* 44:114–130.

Marden, J. I. 1995. *Analyzing and Modeling Rank Data*. London: Chapman and Hall.

Niederle, M.; Roth, A. E.; and Sonmez, T. 2008. Matching and market design. In Durlauf, S. N., and Blume, L. E., eds., *The New Palgrave Dictionary of Economics (2nd Ed.)*, volume 5. Cambridge: Palgrave Macmillan. 436–445.

National Resident Matching Program. 2013. National resident matching program, results and data: 2013 main residency match. NMRP, Washington, DC.

Pilotto, E.; Rossi, F.; Venable, K. B.; and Walsh, T. 2009. Compact preference representation in stable marriage problems. In *Algorithmic Decision Theory*. Springer. 390–401.

Rastegari, B.; Condon, A.; Immorlica, N.; and Leyton-Brown, K. 2013. Two-sided matching with partial information. In *Proceedings of the Fourteenth ACM Conference on Electronic Commerce*, 733–750. ACM.

Roth, A. E. 1984. The evolution of the labor market for medical interns and residents: A case study in game theory. *Journal of Political Economy* 92(6):991–1016.

Shepard, R. N. 1959. Stimulus and response generalization: A stochastic model relating generalization to distance in psychological space. *Psychometrika* 22(4):325–345.