Robust Bidirectional Poly-Matching

Ween Jiann Lee, *Graduate Student Member, IEEE*, Maksim Tkachenko, and Hady W. Lauw, *Senior Member, IEEE*

***Abstract*—A fundamental problem in many scenarios is to match entities across two data sources. It is frequently presumed in prior work that entities to be matched are of comparable granularity. In this work, we address one-to-many or poly-matching in the scenario where entities have varying granularity. A distinctive feature of our problem is its bidirectional nature, where the 'one' or the 'many' could come from either source arbitrarily. Moreover, to deal with diverse entity representations that give rise to noisy similarity values, we incorporate novel notions of receptivity and reclusivity into a robust matching objective. As the optimal solution to the resulting formulation is proven computationally intractable, we propose more scalable yet still performant heuristics. Experiments on multiple real-life datasets showcase the effectiveness and outperformance of our proposed algorithms over baselines.**

***Index Terms*—Entity resolution, matching, one-to-many.**

I. Introduction

Entity matching needs to take into account the varying granularity of entities. By 'entity', we refer to a data record within a collection. This work is concerned with matching entities from two collections that are respectively duplicate-free.[1] Conceptually, what constitutes an entity may differ in granularity across data sources. Consider a product with color variants. One e-commerce site may consider a single record for the product, while another may have multiple records for each color variant individually. Hence, an entity from the former (coarser granularity) may match multiple entities from the latter (finer granularity). For other examples, a chapter in a textbook may span multiple chapters in another; different organizations define job roles by carving out scopes of responsibility differently.

In such scenarios, using *Bipartite matching* with one-to-one constraint is inappropriate. For multi-granular entities, *poly-matching* (Poly) allows an entity of coarser granularity (denoted *host*) from one source to match multiple entities of finer granularity (denoted *clients*) from the other source. Prior works [1] often designate one source for hosts and the other for clients, which is an overly restrictive requirement.

The authors are with the Singapore Management University, Singapore 188065 (e-mail: lwj5@hotmail.com; maksim.tkatchenko@gmail.com; hadywlauw@smu.edu.sg).

Digital Object Identifier 10.1109/TKDE.2023.3266480

[1]If necessary, each respective data source could go through a deduplication process as a precursor to this problem.

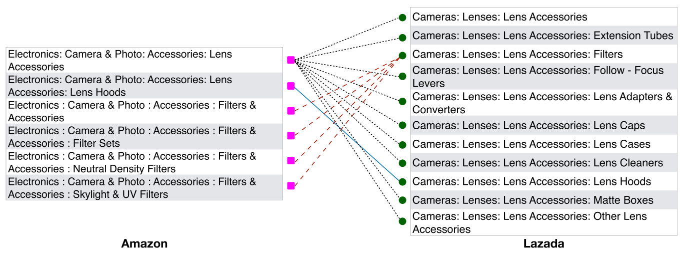

Electronics: Camera & Photo: Accessories: Lens Accessories
Electronics: Camera & Photo: Accessories: Lens Accessories: Lens Hoods
Electronics : Camera & Photo : Accessories : Filters & Accessories
Electronics : Camera & Photo : Accessories : Filters & Accessories : Filter Sets
Electronics : Camera & Photo : Accessories : Filters & Accessories : Neutral Density Filters
Electronics : Camera & Photo : Accessories : Filters & Accessories : Skylight & UV Filters
Amazon
Cameras: Lenses: Lens Accessories
Cameras: Lenses: Lens Accessories: Extension Tubes
Cameras: Lenses: Lens Accessories: Filters
Cameras: Lenses: Lens Accessories: Follow - Focus Levers
Cameras: Lenses: Lens Accessories: Lens Adapters & Converters
Cameras: Lenses: Lens Accessories: Lens Caps
Cameras: Lenses: Lens Accessories: Lens Cases
Cameras: Lenses: Lens Accessories: Lens Cleaners
Cameras: Lenses: Lens Accessories: Lens Hoods
Cameras: Lenses: Lens Accessories: Matte Boxes
Cameras: Lenses: Lens Accessories: Other Lens Accessories
Lazada

Fig. 1. A bidirectional poly-matching between the lens accessories categories in Amazon and Lazada.

A more general formulation is *bidirectional poly-matching* (BiPoly), where host or client alike could come from any source. Consider an example matching of entities (product categories) from two e-commerce taxonomies in Fig. 1. Amazon's `Lens Hoods` *is equivalent to* Lazada's `Lens Hoods` (prefixes to category names are omitted for brevity). In turn, Amazon's `Lens Accessories` *consists of* Lazada's `Lens Caps` and `Lens Cases`, whereas Amazon's `Filter Sets` and `UV Filters` are *parts of* Lazada's `Filters`.

We assume as input similarity weights between any pair of entities across sources. In practice, similarity weights could be noisy as they are often derived from entity descriptions (e.g., product titles). Instead of naively maximizing the sum of weights, we propose a more robust objective that further incorporates two types of rewards. *Reclusivity* encourages entities to refrain from participating in less rewarding matching. *Receptivity* induces more connected components of smaller sizes while discouraging large clusters.

Contributions and Organization. In summary, the main contributions of this paper are as follow:

- *Bidirectional poly-matching constraint* (Section II-A): We introduce the problem of BiPoly-matching for multi-granular entities. To our best awareness, the bidirectionality is novel, inducing a more general formulation than previously known poly-matchings.
- *Robust objective* (Section II-B): We propose a robust optimization objective by incorporating receptivity and reclusivity to adjust for the number of false positives and negatives arising from noisy similarity.
- *Optimal solution* (Section III): We express an optimal formulation for robust BiPoly using Integer Linear Programming (ILP), which subsumes bipartite and poly-matchings as special cases. We further prove that this formulation is computationally intractable.

- *Approximation algorithms* (Section IV): We develop two greedy algorithms, one has a known approximation bound while the other is more efficient.
- *Experiments* (Sections V and VI): We conduct experiments to demonstrate the effectiveness of our algorithms against baselines, and study their scalability.

We cover the related literature in Section VII and discuss the key findings as well as future work in Section VIII.

II. Problem Formulation

We are given two sets of entities, the left set $L = \{l_1, l_2, \dots, l_m\}$ and the right set $R = \{r_1, r_2, \dots, r_n\}$, that contain m and n records respectively. Informally, our objective is to find all pairs of matching multi-granular entities, $\rho \subseteq L \times R$, such that for every pair $\langle l, r\rangle \in \rho$ we can say that either l *is a part of* r, l *consists of* r, or l *is equivalent to* r.

We identify an entity matching by similarity function $h : L \times R \to [0, 1]$. For any $r \in R$ and $l \in L$, $h(r, l)$ is defined as $h(l, r)$. $h(l, r)$ ranges from 0 (l and r are not related) to 1 (highly related). Ideally, $\langle l, r\rangle \in \rho$ if and only if $h(l, r) \approx 1$. In practice, we deal with similarity functions – assumed specified as input – that often produce high similarity scores for entities that ought not to be included in ρ (i.e., false positives) or low similarity scores for entities that ought to be included in ρ (i.e., false negatives). Therefore we introduce constraints and a robust objective function.

A. Matching Constraints

Let $G(L, R)$ be a bipartite graph over the left L and right R entity sets, $L \cap R = \emptyset$. All pairs of entities between two parts are connected by an edge: $\langle l, r\rangle \in E(G)$. Each edge is weighted by the similarity score $h(l, r)$. Any $\rho \subseteq E(G)$ is a matching. We call a graph $G_\rho(L, R)$ with edges from ρ a matching-induced subgraph or simply matching subgraph. For any $l \in L$, let $\rho_l = \{\langle u, r\rangle \in \rho | u = l\}$. Analogously, for any $r \in R$, let $\rho_r = \{\langle l, u\rangle \in \rho | u = r\}$. ρ_u is essentially the set of all the edges connected with u. Since $L \cap R = \emptyset$, ρ_u is unambiguously defined over any $u \in L \cup R$.

To capture the relations of *is equivalent to* between L and R entities, it is apt to employ one-to-one constraint [2], [3].

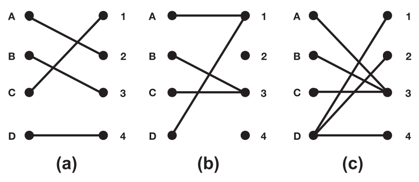
Definition II.1 (Bipartite Matching). $\rho \subseteq E(G)$ is a bipartite (one-to-one) matching in $G(L, R)$ if and only if $\forall u \in L \cup R, |\rho_u| \leq 1$.

This is the most restrictive constraint. Fig. 2 illustrates a toy example involving $L = \{A, B, C, D\}$ and $R = \{1, 2, 3, 4\}$. The adjacency matrix specifies the similarity between any pair of entities. Based on these scores, the maximum weight bipartite matching is $\rho = \{\langle A, 2\rangle, \langle B, 3\rangle, \langle C, 1\rangle, \langle D, 4\rangle\}$.

As bipartite matching is inappropriate for multi-granular relations of *consists of* or *is part of*, we introduce *poly-matching* to relax the one-to-one constraint [3], [4], [5].

Definition II.2 (Poly-Matching). $\rho \subseteq E(G)$ is a poly-matching in $G(L, R)$ if and only if $\forall l \in L, |\rho_l| \leq 1$. When the constraint is imposed on the L, we say the left L is matched into the right R.

h	1	2	3	4
A	0.9	0.9	0.8	0
B	0.5	0	0.7	0.3
C	0.4	0.2	0.5	0.4
D	0.8	0.5	0	0.8



	(a)	(b)	(c)
Bipartite	✓	✗	✗
Poly	✓	✓	✗
BiPoly	✓	✓	✓

Fig. 2. Results based on max-weighted Bipartite, Poly (one-to-many) and the BiPoly constrained matching.

Following this, entities on the right can be matched to multiple entities on the left, but entities on the left can be matched to at most one entity on the right (what constitutes 'left' or 'right' is specific to the domain at hand). In Fig. 2, poly-matching allows A and D on the left to match the same entity on the right 1. Ditto, we have $\rho_3 = \{\langle B, 3\rangle, \langle C, 3\rangle\}$.

In addition to *is equivalent to*, Poly captures either *consists of* or *is part of*, but not both. Our proposed *bidirectional poly-matching* allows the matching to work in both directions.

Definition II.3 (Bidirectional Poly-Matching). $\rho \subseteq E(G)$ is BiPoly in $G(L, R)$ if and only if $\forall \langle l, r\rangle \in \rho, |\rho_l| = 1 \vee |\rho_r| = 1$.

Fig. 1 depicts exactly this kind of matching. BiPoly is also the least restrictive matching as its solution space entails that of Bipartite and Poly as shown in Fig. 2.

B. Objective Function

Having identified possible matching space $\mathfrak{P}$, we seek to find $\rho \in \mathfrak{P}$, which maximizes a particular objective function.

Max-Weight. The classical objective is to maximize the total similarity score, known as the maximum weight objective.

$$\rho = arg \max_{\rho' \in \mathfrak{P}} \sum_{\langle l, r\rangle \in \rho'} h(l, r) \tag{1}$$

In practice, the similarity measure $h(l, r)$ (e.g., cosine similarity) is imperfect. In most cases, the entity representations used to measure that similarity (e.g., product title) are highly noisy. For instance, entities of a domain may contain a generic word (e.g., *camera*) which results in non-zero similarity. Applying a max-weight objective on noisy similarity may result in false positives from 'over-estimated' similarity or false negatives from 'under-estimated' similarity.

Robust. To address this, we develop a robust version of the constraint objective. It introduces incentives that can be adjusted independently to reduce the number of false positives and negatives cooperatively. The rewards are cast upon the connected components in $\rho \in \mathfrak{P}$. Any matching-induced subgraph G_ρ consists of one or more connected components. We distinguish between two types of connected components. A component is *reclusive* if it contains exactly one entity u, i.e., $|\rho_u| = 0$. Otherwise, it is *receptive*, i.e., for each entity u in that component, we have $|\rho_u| \geq 1$.

TABLE I
SUMMARY OF NOTATIONS

Notation	Description
$h(l,r) \in [0,1]$	similarity score between l and r (input)
$v_u \in \{1,0\}$	variable whether u is a host/client
$z_u \in \{1,0\}$	variable whether u is a receptive/reclusive
$v_u = 1, z_u = 1$	indicating u is a receptive host
$v_u = 1, z_u = 0$	indicating u is a reclusive host
$v_u = 0, z_u = 0$	indicating u is a client
$x_{l\to r} \in \{0,1\}$	whether host r accepts request from client l
$x_{r\to l} \in \{0,1\}$	whether host l accepts request from client r
$\omega_u \in [-1,1]$	reclusivity reward offered to u (parameter)
$\eta_u \in [-1,1]$	receptivity reward offered to u (parameter)

We define the robust objective with the reclusivity incentive function $\mathcal{F}_\Omega$ and receptivity incentive function $\mathcal{F}_H$:

$$\rho = arg\max_{\rho' \in \mathfrak{P}} \sum_{\langle l,r\rangle \in \rho'} h(l,r) + \mathcal{F}_H\left(C(G_\rho)\right) + \mathcal{F}_\Omega\left(\bar{C}(G_\rho)\right), \tag{2}$$

where $C(\cdot)$ and $\bar{C}(\cdot)$ are the sets of receptive and reclusive components in a given graph respectively.

In some low similarity cases, we may discourage matching as they could be incidental. In other cases, we may encourage matching if similarity scores are systematically low. $\mathcal{F}_\Omega$ provides an incentive to each reclusive component. This reward can be positive, which encourages entities to stay reclusive unless they can form a match that captures a similarity score higher than the incentive. Alternatively, this reward can be negative, which discourages reclusivity.

Among receptive components, there is a trade-off between a smaller number of connected components of larger cardinalities, or a larger number of components of smaller cardinalities. If $\mathcal{F}_H$ offers positive rewards, the outcome tends towards more receptive components, which contain fewer entities. The converse applies to negative rewards.

Together, $\mathcal{F}_\Omega$ and $\mathcal{F}_H$ lend greater robustness to noisy similarity by steering the matching away from pure max-weight. In Section III, we define these functions concretely. In Section V-E, we discuss in detail the effects of reclusivity and receptivity in the experimental settings.

III. Linear Programming Models

To articulate a concrete definition of robust BiPoly (other formulations represent special cases), we propose a binary linear program that lends itself to an optimal solution.

A. Formulation

A matching-induced subgraph has multiple connected components. For each component, we designate one node as the *host* and the rest as *clients*, indicated by the binary variable: $v_u = 1$ if u is the host and $v_u = 0$ if u is a client. A reclusive component has one host and no client. In a receptive component of one edge, either end-point can be the host. In a larger receptive component, the host is either the only left entity or the only right entity. These variables and other notations are summarized in Table I. To differentiate reclusive from a receptive host, we use binary variable z_u, i.e., $(v_u = 1, z_u = 1)$ if u is receptive and $(v_u = 1, z_u = 0)$ if u is reclusive. For any client, $(v_u = 0, z_u = 0)$.

To indicate connectivity within receptive components, for every $\langle l,r\rangle \in E(G)$, we introduce two mutually exclusive variables $x_{l\to r}$ and $x_r \to l$ that indicate whether l and r are connected $\langle l,r\rangle \in \rho$, the subscript arrow indicates the direction: from a client to the host. For example, $x_{l\to r} = 1$ means host r accepts a request from client l.

To arrive at a robust BiPoly matching, we seek a configuration that fulfills the following linear program.

$$\max \sum_{l\in L}\sum_{r\in R} h(l,r)\left(x_{l\to r} + x_{r\to l}\right)$$

$$+ \sum_{u\in L\cup R} \omega_u\left(v_u - z_u\right) + \sum_{u\in L\cup R} \eta_u z_u \text{s.t.} \tag{3}$$

$$\sum_{l\in L} x_{r\to l} + v_r = 1 \qquad \forall r\in R \tag{4}$$

$$\sum_{r\in R} x_{l\to r} + v_l = 1 \qquad \forall l\in L \tag{5}$$

$$x_{r\to l} \le v_l \qquad \forall l\in L, \forall r\in R \tag{6}$$

$$x_{l\to r} \le v_r \qquad \forall l\in L, \forall r\in R \tag{7}$$

$$\sum_{l\in L} x_{l\to r} \ge z_r \qquad \forall r\in R \tag{8}$$

$$\sum_{r\in R} x_{r\to l} \ge z_l \qquad \forall l\in L \tag{9}$$

$$x_{l\to r} \le z_r \qquad \forall l\in L, \forall r\in R \tag{10}$$

$$x_{r\to l} \le z_l \qquad \forall l\in L, \forall r\in R \tag{11}$$

$$x_{l\to r}, x_{r\to l} \in \{0,1\} \qquad \forall l\in L, \forall r\in R$$

$$v_u, z_u \in \{0,1\} \qquad \forall u\in L\cup R$$

The first line of (3) subject to the constraints reproduces the max-weight objective in (1).

The first part of the second line is a formulation of $\mathcal{F}_\Omega$ from (2). Each entity u is offered a reclusivity reward $\omega_u \in [-1,1]$ (a parameter to be specified), which is earned only if u is a reclusive host, i.e., $(v_u = 1, z_u = 0)$. If $\omega_u > 0$, this incentivizes u to stay reclusive unless it can form a match that captures a similarity score higher than the incentive. When $\omega_u < 0$, it tends to form a match, with ω_u compensating for low similarity scores. $\omega_u = 0$ is neutral.

In turn, the second part presents a concrete formulation of $\mathcal{F}_H$ in (2). We introduce a receptivity reward $\eta_u \in [-1,1]$ (a parameter), earned only if u is a receptive host. When $\eta_u > 0$, more receptive components are encouraged to be formed, which may as a by-product reduce the cardinality of other receptive components. If $\eta_u < 0$, it consumes a part of the similarity within each component, thus, discouraging any connection from being formed, which may drive entities to join other components. $\eta_u = 0$ is neutral.

While unique rewards for every node u are possible, in practice we experiment with a simpler framework where rewards are tied for each part of $G(L,R)$: $\omega_l = \omega_L$ and $\eta_l = \eta_L$, $\forall l \in L$, analogously $\omega_r = \omega_R$ and $\eta_r = \eta_R$, $\forall r \in R$.

(4) and (5) ensure that entities can either be host or client, and a client's request can only be accepted by a single host. (6) and (7) state that only hosts can accept requests. These jointly enforce the bidirectional one-to-many restriction. (8) through (11) ensure that $z_u = 1$ if and only if u is a host that has accepted one or more clients.

BiPoly-matching presents the loosest constraint setting. To recover poly-matching, we add linear constraint $x_{l\to r} = 0$ for all $\langle l, r\rangle \in E(G)$ blocking all the client requests from the left part of the graph L. To recover bipartite matching from poly-matching, we restrict the number of incoming requests for each host to at most one: $\sum_{r\in R} x_{r\to l} \leq 1$ for all $l \in L$.

B. Computational Intractability

We show that the robust BiPoly-matching problem is NP-hard by proving that the uncapacitated facility location problem (UFLP) [1] (known to be NP-hard) is reducible to our problem. In UFLP, we are given a set of customers P and a set of facilities O. Each facility $o \in O$ incurs a fixed opening cost $\epsilon_o \in \mathbb{R}^+$ and cost $k_{op} \in \mathbb{R}^+$ of serving customer $p \in P$. A solution to the problem assigns every customer to a facility. To encode this assignment, we use a set of binary variables: $\alpha_{op} = 1$ if facility $o \in O$ serves customer $p \in P$, $\alpha_{op} = 0$ otherwise. A binary variable β_o tracks opened facilities, $\beta_o = 1$ if facility $o \in O$ is open, and $\beta_o = 0$ otherwise. The objective is to find a solution that minimizes the total cost of serving and opening:

$$\min \sum_{p\in P}\sum_{o\in O} k_{op}\alpha_{op} + \sum_{o\in O}\epsilon_o\beta_o \text{ s.t.} \tag{12}$$

$$\sum_{o\in O}\alpha_{op} = 1 \qquad \forall p \in P$$

$$\alpha_{op} \leq \beta_o \qquad \forall o \in O, \forall p \in P$$

$$\beta_o, \alpha_{op} \in \{0,1\} \qquad \forall o \in O, \forall p \in P$$

We formally identify an instance of the UFLP problem by a tuple UFLP(O,P,K,E), where $K = \{k_{op} | o \in O, p \in P\}$ and $E = \{\epsilon_o | o \in O\}$.

Lemma III.1. Given UFLP(O,P,K,E), $\delta \in \mathbb{R}_{>0}$ and $\mu \in \mathbb{R}$, let $\mu + \delta K = \{\mu + \delta k_{op} | o \in O, p \in P\}$ and $\delta E = \{\delta\epsilon_o\}_{o\in O}$, then a solution that minimizes the objective of UFLP(O,P,K,E) also minimizes the objective of UFLP$(O,P,\mu+\delta K,\delta E)$ and vice versa.

Proof: Subject to the constraints of Program 12, we have:

$$\min \sum_{p\in P}\sum_{o\in O}(\mu + \delta k_{op})\alpha_{op} + \sum_{o\in O}\delta\epsilon_o\beta_o$$

$$= \min \sum_{p\in P}\sum_{o\in O}\mu\alpha_{op} + \sum_{p\in P}\sum_{o\in O}\delta k_{op}\alpha_{op} + \sum_{o\in O}\delta\epsilon_o\beta_o$$

$$= \min \sum_{p\in P}\sum_{o\in O}\delta k_{op}\alpha_{op} + \sum_{o\in O}\delta\epsilon_o\beta_o$$

$$= \min \sum_{p\in P}\sum_{o\in O} k_{op}\alpha_{op} + \sum_{o\in O}\epsilon_o\beta_o$$

□

UFLP is usually defined over non-negative serving cost. The shifted cost $\mu + \delta K$ could be negative. This alters neither the solution space nor the minimizing solution.

The following corollary helps to satisfy the constraints when reducing an instance of UFLP to a Robust BiPoly.

Corollary III.1.1. For any non-degenerate UFLP(O,P,K,E), there exists UFLP(O,P,K',E') that has the same minimizing solutions as the original UFLP. Furthermore $k'_{op} \in (-\tau, 0]$ for all $k'_{op} \in K'$ and $\epsilon'_o \in [0,\tau)$ for all $\epsilon'_o \in E'$, where $\tau^{-1} = 2|O|$.

Proof. The proof is an immediate consequence of the lemma, let $\delta = \tau(1+\max(K,E))^{-1}$ and $\mu = -\delta\max(K)$. □

We assume that any instance of UFLP has service and opening costs that satisfy this corollary. Therefore, opening costs and total serving cost per client are predictably bounded. Carefully choosing the weights for an instance of Robust BiPoly, one can always get a solver for a one-sided poly-matching as in UFLP. The following theorem details how the weight should be chosen for the reduction.

Theorem III.2. Robust BiPoly-matching is NP-hard.

Proof. Assume that the robust BiPoly-matching is solvable in polynomial time. To arrive at a contradiction, we intend to show that any UFLP(O,P,K,E) can be solved using the robust BiPoly-matching as stated in Program (3). We, thus, create the following instance of the robust BiPoly–matching:

$$L \overset{\text{def}}{=} O \quad \text{and} \quad R \overset{\text{def}}{=} P$$

$$\omega_l \overset{\text{def}}{=} 0 \quad \text{and} \quad \eta_l \overset{\text{def}}{=} -\epsilon_l \qquad \forall l \in L$$

$$\omega_r = \eta_r \overset{\text{def}}{=} -1 \qquad \forall r \in R$$

$$h(l,r) = h(r,l) \overset{\text{def}}{=} -k_{lr} \qquad \forall l \in L, \forall r \in R$$

Subject to the constraints of Program 3, we want to solve the following maximization problem:

$$\max \sum_{o\in O}\sum_{p\in P} -k_{op}\left(x_{o\to p} + x_{p\to o}\right) - \sum_{o\in O}\epsilon_o z_o - \sum_{p\in P} v_p,$$

or its corresponding minimization problem (equivalent to robust BiPoly):

$$\min \sum_{o\in O}\sum_{p\in P} k_{op}\left(x_{o\to p} + x_{p\to o}\right) + \sum_{o\in O}\epsilon_o z_o + \sum_{p\in P} v_p.$$

For every $o \in O$ and $p \in P$, let $\bar{x}_{o\to p}$, $\bar{x}_{p\to o}$, $\bar{v}_o$, $\bar{v}_p$, $\bar{z}_o$, and $\bar{z}_p$ be a minimizing solution of this problem, then we have: $\bar{v}_p = 0$, $\bar{v}_o = 1$, $\bar{z}_p = 0$, and $\bar{x}_{o\to p} = 0$. Let us show this by contradiction. Assume for some p', $\bar{v}_{p'} = 1$. The cases are: (I) p' is receptive host or (II) reclusive host, as in Fig. 3.

I. If p' is a receptive host, then it accepts clients' requests ($z_{p'} = 1$), $\exists o \in O$, $\bar{x}_{o\to p'} = 1$. Let $O_{p'} = \{o \in O | \bar{x}_{o\to p'} = 1\}$, the contribution of the connected component associated with p' to the objective function is $\Delta_b = 1 + \sum_{o\in O_{p'}} k_{op'} > 0.5$ (see Corollary III.1.1). If we set $\bar{x}_{o'\to p'} = 0$ and $\bar{v}_o = 1$ for all $o \in O'_p$, $\bar{x}_{p'\to o'} = 1$ and $\bar{z}_{o'} = 1$ for some $o' \in O$ (to satisfy the

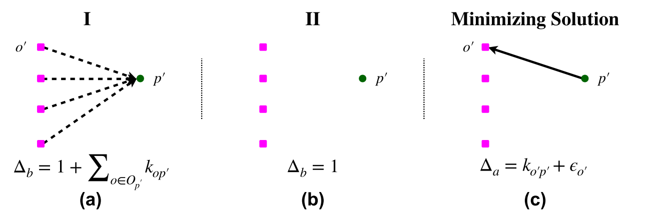

I
II
Minimizing Solution
o'
p'
$\Delta_b = 1 + \sum_{o \in O_{p'}} k_{op'}$
$\Delta_b = 1$
$\Delta_a = k_{o'p'} + \epsilon_{o'}$
(a)
(b)
(c)

Fig. 3. Illustration of the proof: (a) - receptive host I, (b) - reclusive host II, (c) - a solution with a lower objective.

constraints we also set $v_{p'} = z_{p'} = 0$), the contribution decreases to $\Delta_a = k_{o'p'} + \epsilon_{o'} < 0.5 < \Delta_b$, contradicting the assumption of this being a minimizing solution.

II. If p' is a reclusive host ($z_{p'} = 0$), then its contribution to the objective is $\Delta_b = 1$, we can improve the solution by connecting p' to some $o' \in O$ by setting $x_{p' \to o'} = 1$ (to satisfy the constraints we also set $v_{p'} = z_{p'} = 0$ and $v_{o'} = z_{o'} = 1$). The contribution of p' to the objective then decreases and becomes at most $\Delta_a = k_{o'p'} + \epsilon_{o'} < 0.5$ (see Corollary III.1.1). Since $\Delta_b > \Delta_a$, we, therefore, arrive at the contradiction.

Since $\bar{v}_p = 0$ for all $p \in P$, the other equalities follow from the constraints. If we include these equalities in the program, we get an equivalent minimization problem:

$$\min \sum_{o \in O} \sum_{p \in P} k_{op} x_{p \to o} + \sum_{o \in O} \epsilon_o z_o. \tag{13}$$

After removing the inconsequential constraints, Program 13 has the following constraints:

$$\sum_{o \in O} x_{p \to o} = 1 \qquad \forall p \in P$$

$$x_{p \to o} \leq z_o \qquad \forall o \in O, \forall p \in P,$$

Which recovers exactly UFLP formulation as in Program 12 (up to variable names). Therefore, robust BiPoly-matching can be used to solve any instance of the UFLP. Since UFLP is NP-hard [1], [6], this contradicts the premise of the proof. □

IV. Greedy Approximations

As the linear program may not be tractable for large data sizes, we present two greedy approximations.

A. Set Cover-Based Approximation

Algorithm. Let us consider the following minimization problem subject to the constraints of the robust BiPoly-matching:

$$\min \sum_{l \in L} \sum_{r \in R} (1 - h(l,r)) (x_{l \to r} + x_{r \to l}) \tag{14}$$

$$+ \sum_{u \in L \cup R} (1 - \omega_u)(v_u - z_u) + \sum_{u \in L \cup R} (1 - \eta_u) z_u$$

$$= \min \sum_{l \in L} \sum_{r \in R} (x_{l \to r} + x_{r \to l}) + \sum_{u \in L \cup R} v_u - \sum_{u \in L \cup R} \eta_u z_u$$

$$- \sum_{u \in L \cup R} \omega_u (v_u - z_u) - \sum_{l \in L} \sum_{r \in R} h(l,r)(x_{l \to r} + x_{r \to l})$$

$$= \min \sum_{l \in L} \left(v_l + \sum_{r \in R} x_{l \to r} \right) + \sum_{r \in R} \left(v_r + \sum_{l \in L} x_{r \to l} \right)$$

$$- \sum_{u \in L \cup R} \eta_u z_u - \sum_{u \in L \cup R} \omega_u (v_u - z_u)$$

$$- \sum_{l \in L} \sum_{r \in R} h(l,r)(x_{l \to r} + x_{r \to l}).$$

Substituting the variable summations with (4) and (5) shows that this minimization problem is equivalent to robust BiPoly, with the objective value shifted by $n + m$. Furthermore, it has non-negative variable weights and can be converted to an instance of weighted set cover. In weighted set cover, we aim to cover all the nodes in $L \cup R$ with the sets that yield minimal total cost. The collection of covering sets and their weights are defined as follows:

$$\mathcal{S} = \mathcal{S}_L \cup \mathcal{S}_R, \text{ where } \begin{array}{l} \mathcal{S}_L = \left\{ \{l\} \cup e \,\middle|\, l \in L, e \in 2^R \right\} \\ \mathcal{S}_R = \left\{ \{r\} \cup e \,\middle|\, r \in R, e \in 2^L \right\} \end{array} \tag{15}$$

$$w_s = \begin{cases} 1 - \omega_u & \text{if } \{u\} = s \\ 2 - \max(\eta_l, \eta_r) - h(l,r) & \text{if } \{l\} = s \cap L \text{ and } \{r\} = s \cap R \\ |s| - \eta_l - \sum_{r \in s \setminus \{l\}} h(l,r) & \text{if } s \in \mathcal{S}_L \setminus \mathcal{S}_R \text{ and } \{l\} = s \cap L \\ |s| - \eta_r - \sum_{l \in s \setminus \{r\}} h(l,r) & \text{if } s \in \mathcal{S}_R \setminus \mathcal{S}_L \text{ and } \{r\} = s \cap R \end{cases}$$

The weighted set cover is related to the connected component interpretation of the robust BiPoly outlined in Section III-A. $\mathcal{S}$ defines a set of all possible connected components in $G(L,R)$ w.r.t. BiPoly as in Definition II.3. $\mathcal{S}_L$ identifies all possible connected components between the hosts from L and the clients from R including closed hosts, $\mathcal{S}_R$ does the same for the hosts from R. Weights $W = \{w_s\}_{s \in \mathcal{S}}$ are mapped to the contributions of these components as in the robust BiPoly. The first case for w_s identifies the contribution of reclusive hosts, the second counts the minimal contribution of the interchangeable host-client pairs, and the last two identify the contribution of the components that have multiple clients. Though the solution space is larger for the weighted set cover than for robust BiPoly, one can show that at least one of the minimizing solutions of the set cover consists of disjoint sets, thus, is convertible to a minimizing robust BiPoly solution.

BiPoly Set Cover (BiPoly.S) adapts the greedy solver as in [7]. At each iteration, the procedure finds set $s \in \mathcal{S}$ minimizing its per node weight $w_s/|s|$ to cover nodes yet not assigned. To avoid enumerating all possible sets as defined in (15), for each node $u \in L \cup R$ we maintain a list of potential clients in descending order of similarity $h(u, \cdot)$. At each round of the procedure, we retrieve a minimizing set in polynomial time. This is shown in Algorithm 1.

Complexity Analysis. The complexity of Algorithm 1 can be split into two parts: sorting clients for each host, which accounts for $\mathcal{O}(nm \log nm)$ in worst-case, and selecting a minimizing set from $\mathcal{S}$. The worst case is when every viable combination of hosts and clients is evaluated, but only a singleton host would ultimately be selected in the round, and we always select a host

Algorithm 1: Greedy Algorithm for BiPoly.S.

```
Input  : an instance of Robust BiPoly: $L$, $R$, $h$,
         $\Omega = \{\omega_u\}_{u \in L \cup R}$, $H = \{\eta_u\}_{u \in L \cup R}$
Output: a bidirectional poly-matching $\rho$
Initialize: $U \leftarrow L \cup R$, $\rho \leftarrow \emptyset$
foreach $l \in L$ do $C[l] \leftarrow$ SortClients($l, R, h$)
foreach $r \in R$ do $C[r] \leftarrow$ SortClients($r, L, h$)
while $U$ is not empty do
    $A \leftarrow \emptyset$
    foreach $u \in U$ do
        $a.host \leftarrow u$
        $a.clients \leftarrow \emptyset$
        $a.w \leftarrow 1 - \omega_u$
        $b.host \leftarrow u$
        $b.clients \leftarrow \emptyset$
        $b.w \leftarrow 1 - \eta_u$
        foreach $c \in C[u] : c \in U$ do
            $b.w \leftarrow b.w \cdot (|b.clients| + 1) + 1 - h(u, c)$
            $b.clients \leftarrow b.clients \cup \{c\}$
            $b.w \leftarrow b.w / (|b.clients| + 1)$
            if $a.w \geq b.w$ then $a \leftarrow b$
        end
        $A \leftarrow A \cup \{a\}$
    end
    $s \leftarrow \arg\min_{a \in A} a.w$
    $U \leftarrow U \setminus (\{s.host\} \cup s.clients)$
    if $s.host \in L$ then $\rho \leftarrow \rho \cup \{\langle s.host, u\rangle | u \in s.clients\}$
    else $\rho \leftarrow \rho \cup \{\langle u, s.host\rangle | u \in s.clients\}$
end
return $\rho$
```

Algorithm 2: Greedy Algorithm for BiPoly.C.

```
Input  : an instance of Robust BiPoly: $L$, $R$, $h$,
         $\Omega = \{\omega_u\}_{u \in L \cup R}$, $H = \{\eta_u\}_{u \in L \cup R}$
Output: a bidirectional poly-matching $\rho$
Initialize: $\rho \leftarrow \emptyset$, $C \leftarrow$ SortPairs($L, R, h$)
foreach $u \in L \cup R$ do $T[u] \leftarrow unknown$, $P[u] \leftarrow \emptyset$
function AssignHostClient($a, b$):
    $T[a] \leftarrow host$, $T[b] \leftarrow client$
    if $a \in L$ then $\rho \leftarrow \rho \cup \{\langle a, b\rangle\}$ else $\rho \leftarrow \rho \cup \{\langle b, a\rangle\}$
foreach $\langle a, b\rangle \in C$ do
    if $T[a] = T[b] = unknown$ then
        if $h(a, b) > \omega_a - \eta_a \wedge h(a, b) > \omega_b - \eta_b$ then
            $T[a] \leftarrow pending$, $T[b] \leftarrow pending$
            $P[a] \leftarrow b$, $P[b] \leftarrow a$
        else if $h(a, b) > \omega_a - \eta_a$ then
            AssignHostClient($a, b$)
        else if $h(a, b) > \omega_b - \eta_b$ then
            AssignHostClient($b, a$)
    else if $T[a] = client \vee T[b] = client$ then
        continue
    else if $T[b] = unknown \wedge h(a, b) > \omega_b$ then
        if $T[a] = pending$ then
            AssignHostClient($a, P[b]$)
        AssignHostClient($a, b$)
    else if $T[a] = unknown \wedge h(a, b) > \omega_a$ then
        if $T[b] = pending$ then
            AssignHostClient($b, P[a]$)
        AssignHostClient($b, a$)
    end
end
$\rho \leftarrow \rho \cup$ ResolvePending($T, P, \Omega, H, C, h$)
return $\rho$
```

from the bigger part. Let $a = \min(n, m)$, $b = \max(n, m)$, and $k = b - a$, then the computations for the first k iterations can be bounded by $\mathcal{O}(a \sum_{i=1}^{k} (2b - i + 1))$ or $\mathcal{O}(ab^2)$. For the next $2a$ iterations the procedure alternates between selecting hosts from L and R, thus, the number of operations for every two consecutive rounds can be bounded (up to a constant) by $2a(n + m)$, $2(a - 1)(n + m)$, $2(a - 2)(n + m)$ and so on, which in total is bounded by $\mathcal{O}((m + n) \sum_{i=1}^{a} i)$ or $\mathcal{O}(a^2(m + n))$. The worst case complexity of BiPoly.S is $\mathcal{O}((n + m)^3)$. However, the empirical evaluation on real-world data (see Fig. 8) suggests that the running time behaves like $\mathcal{O}((n + m)^2)$.

Approximation Bound. BiPoly.S has an approximation guarantee [8]: $log(n + m)$ to an optimal solution of (14). Let $g^{\text{min}}_{\text{greedy}}$ be the objective value with the greedy solution, $g^{\text{min}}_{\text{opt}}$ be the optimal objective value, then:

$$g^{\text{min}}_{\text{greedy}} \leq g^{\text{min}}_{\text{opt}} \log (n + m).$$

As the maximization objective is shifted by $n + m$, we derive a lower bound for a greedy solution g_{greedy} of Program 3:

$$g_{\text{greedy}} \geq g_{\text{opt}} \log (n + m) + (n + m) (1 - \log (n + m)),$$

where g_{opt} is the optimal objective value for robust BiPoly.

B. CENTER-Based Approximation

Algorithm. We propose another greedy method that has a lower worst-case complexity but without a lower bound guarantee. We consider our task as a form of clustering for duplicate detection [9]. Although the clustering methods outlined in [9] do not adopt any constraints, one work, in particular, CENTER [10] can produce "one-to-many" clusters. The algorithm works by iterating over the pairs of nodes of an arbitrary graph. The pairs are sorted by distance. When node u appears in the scan for the first time, it is marked as a cluster center (host in our terms). All subsequent nodes v that appear in pair $\langle u, v\rangle$ are assigned to u (in our terms, v becomes a client of host u).

BiPoly CENTER (BiPoly.C) adapts the algorithm to work with nodes from two partitions and asserts that for any host, the clients must come from the opposing partition. Scanning pair $\langle u, v\rangle$, we mark both nodes as pending if they are unseen, and only when one of these nodes is encountered again, it is promoted to a host and the other to a client. This process produces bidirectional one-to-many disjointed sets where the host and clients are from different partitions.

To further improve the performance, BiPoly.C applies robustness by introducing the reclusitivity and receptivity rewards to deter or incentivize hosts and clients from forming. This is outlined in Algorithm 2. Enclosed within is the method `ResolvePending`, which helps to resolve any remaining *pending-pending* pairs in P by assigning *host* and *client* to the nodes in the pair that minimizes the overall cost.

Complexity Analysis. As BiPoly.C is a single pass algorithm, it only takes $\mathcal{O}(nm)$ to iterate over all the sorted node pairs. The sorting can be done in $\mathcal{O}(nm \log nm)$. The method `ResolvePending` is a linear procedure which in the worst case requires $\mathcal{O}(nm)$ iterations. The worst-case complexity of the algorithm is $\mathcal{O}(nm \log nm)$, which puts it in the same category as Bipartite greedy. This is supported by the empirical evidence on real-world data (see Fig. 8).

TABLE II
SUMMARY OF BLOCKING DATASETS

Dataset	m	n	$\lvert e^L \rvert$	$\lvert e^R \rvert$	Matches
Cross-Platform	94	173	6,913	166,307	992
Multi-Lingual	119	169	710,475	20,000	2,315

V. EXPERIMENT: BLOCKING SCENARIO

The first experimental scenario we investigate is matching products across two data sources. In a strategy called *blocking*, we first match a category from one source to another category from another source. Once the two categories are aligned, we only need to compare the products across these two categories. In this case, the multi-granular entities that BiPoly matching applies to refer to the product categories.

A. Datasets

As blocking on multi-granular entities is novel, we gather two real-world datasets with ground truths[2] as in Table II.

Cross-Platform. The first dataset concerns matching product categories across two e-commerce platforms. It has $m = 94$ categories from Amazon US (the left set L) and $n = 173$ categories from Lazada (the right set R).

Determining whether two categories match must go beyond superficial names and necessarily be informed by whether they contain the same products. We employ Mechanical Turk (MTurk) to manually identify matching *products*, and assess the matching of *categories* indirectly by how the latter facilitates the former. MTurk workers were instructed to select whether a candidate product matches the target product, where two products are considered a match only if they have the same brand, model, type, size, color, etc. Each task was assigned to three workers and the majority vote was accepted. Let e^L (resp. e^R) be the union of products under the category entities in L (resp. R). In total, 992 product matches are identified out of 15,516 pairs labeled, sampled from $|e^L| = 6,913$ products from Amazon and $|e^R| = 166,307$ from Lazada.

Multi-Lingual. The second dataset again concerns category entities, but the respective sources correspond to two regional platforms of Amazon, namely the US (the left set L) and China (the right set R). Product categorization differs vastly between both regions. Amazon uses a unique identifier (UID) that is consistent across regions, making it feasible to identify matching product pairs. In the case of Amazon China, we use machine translation with Microsoft Azure Translator to produce the English representations for similarity measurement with Amazon US vocabulary. In total, this dataset involves $m = 119$ categories from the US covering $|e^L| = 710,475$ products and $n = 169$ categories from China covering $|e^R| = 20,000$, with 2,315 matching product pairs identified by UID.

By way of example, Fig. 4 shows a visualization of the ground truth matching results for *Cross-Platform* and *Multi-Lingual*. The left nodes are magenta squares, and the right nodes are green circles. The multiple connected components, with a mixture of left nodes and right nodes as hosts, indicate the BiPoly effect. The singletons advocate for the robust objective, as the maximum weight objective would have linked up some singletons.

[2]These datasets and the ground truths are available at https://code.preferred.ai/BiPoly.

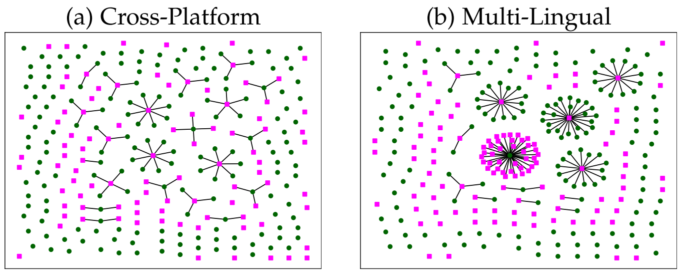


Fig. 4. Theoretical maximum coverage results for robust BiPoly for the various datasets.

The representation of a product entity is its title, while that of a category entity is a bag of words of product titles within the category. To derive the similarity weights for matching $h(l, r)$, we apply three common normalized similarity functions and select the best tradeoff (see below). These are Jaccard Coefficient, Szymkiewicz–Simpson coefficient [11], and TFIDF cosine similarity.

B. Evaluation Measures

We employ metrics commonly associated with blocking [12]. Two categories from different sources are well-aligned if they collectively contain the ground-truth product pairs.

Coverage. We define a blocking ρ's goodness in terms of *coverage*, how well it recovers ground-truth product pairs.

$$\text{product-pairs}(\rho) = \left\{ \langle a, b \rangle \,\middle|\, a \in e^l, b \in e^r, \langle l, r \rangle \in \rho \right\}$$

$$\text{coverage}(\rho) = \frac{\sum_{\langle a,b \rangle \in \text{product-pairs}(\rho)} g(a,b)}{\sum_{\langle a,b \rangle \in \text{product-pairs}(L \times R)} g(a,b)} \qquad (16)$$

Tradeoff. Without blocking, we would compare all $|e^L| \times |e^R|$ pairs of products. With category matching as a blocking strategy, we compare only products across matched categories (much fewer). We can define *reduction* due to a category matching outcome ρ as follows.

$$\text{reduction}(\rho) = 1 - \frac{\sum_{\langle l,r \rangle \in \rho} |e^l| \cdot |e^r|}{|e^L| \cdot |e^R|} \qquad (17)$$

The higher the reduction, the fewer product pairs that need to be compared, the more efficient is the product-to-product matching. We have found that most of the methods we tested achieve a reduction of over 70%.

However, higher reduction often corresponds to lower coverage. Conversely, a higher coverage could be achieved by favoring matching larger categories that might well lead to more comparisons. Hence, a more balanced metric than coverage per se is the *tradeoff*, expressed as the harmonic mean of coverage and reduction, which has been established in product-to-product

TABLE III
COMPARISON ON LINEAR PROGRAM

	Constraint	ω_L	ω_R	η_L	η_R	Robust Coverage	Robust Tradeoff	Max-Weight Coverage	Max-Weight Tradeoff
Cross-Platform	Bipartite	0.3	0.3	0.0	—	38.91	55.72	37.50	54.27
	UFLP	0.0	0.0	-0.9	—	46.17	62.51	46.06	62.44
	Poly	0.5	0.5	0.7	—	47.68	63.92	46.06	62.44
	BiPoly	0.1	0.1	0.2	0.2	**56.85**	**71.40**	**56.75**	**71.32**
	Oracle					67.44	79.05	67.44	79.05
Multi-Lingual	Bipartite	0.1	0.1	0.0	—	20.52	33.91	20.22	33.49
	UFLP	0.0	0.0	-0.8	—	39.87	56.27	35.12	51.45
	Poly	-0.8	-0.8	-1.0	—	39.87	56.27	35.12	51.45
	BiPoly	0.0	0.0	0.7	0.7	**47.00**	**57.76**	**41.04**	**55.02**
	Oracle					55.51	63.09	55.51	63.09

TABLE IV
GREEDY APPROXIMATIONS

	Constraint	ω_L	ω_R	η_L	η_R	Robust Coverage	Robust Tradeoff	Max-Weight Coverage	Max-Weight Tradeoff
Cross-Platform	Bipartite	0.2	0.2	0.0	—	36.59	53.30	36.59	53.28
	UFLP	0.0	0.0	-0.1	—	45.26	61.72	44.96	61.46
	Poly	0.1	0.1	0.1	—	45.26	61.76	44.96	61.46
	K-core	—	—	—	—	33.57	49.84	33.57	49.84
	BiPoly.C	0.2	0.0	-0.6	0.0	**56.05**	**70.82**	**53.33**	**68.60**
	BiPoly.S	0.1	0.0	-0.1	-0.1	55.75	70.73	51.51	67.14
Multi-Lingual	Bipartite	0.2	0.2	0.0	—	15.29	26.47	15.29	26.47
	UFLP	0.0	0.0	-0.2	—	32.92	48.82	31.92	47.81
	Poly	-0.2	-0.2	-0.8	—	38.27	54.35	31.92	47.81
	K-core	—	—	—	—	22.42	36.29	22.42	36.29
	BiPoly.C	-0.6	0.0	-1.0	-1.0	36.89	53.14	**32.83**	47.31
	BiPoly.S	0.2	0.2	0.0	0.2	**39.31**	**54.55**	23.24	37.58

matching literature [13].

$$\text{tradeoff}(\rho) = \frac{2 \times \text{coverage}(\rho) \times \text{reduction}(\rho)}{\text{coverage}(\rho) + \text{reduction}(\rho)} \quad (18)$$

C. Optimal Solutions Via Linear Program

We analyze the optimal solutions under various objectives.

Max-Weight Objective. We first conduct a comparison under the maximum weight objective (see Section II-B). The baselines are Bipartite and Poly, which share the same objective and differ only in constraints. Table III (rightmost columns) summarizes the results. Evidently, constraints do matter. Due to the inherent multi-granularity of entities, Bipartite with the most restrictive one-to-one constraint has the lowest coverage. Poly-matching allows one-to-many and realizes more coverage, but is still limited by the restriction that the 'one' must come from one source (the source with fewer entities[3]). Because BiPoly factors in the possibility for hosts to flexibly come from either source, it attains the highest coverage. A similar trend manifests in the Tradeoff.

For reference, *Oracle* indicates the theoretical maximum achievable coverage, given that there is some inherent noise in the data (e.g., incorrect categorization of products in the taxonomy, noisy similarity). Evidently, the BiPoly results are not too far off from the Oracle results.

Robust Objective. We now conduct a comparison under the robust objective (see Section II-B). The comparable baselines are now *robust* Bipartite and Poly-matching, which can be recovered from robust BiPoly formulation by adding a few linear constraints (see Section III-A). For completeness, we include an additional method, namely UFLP [1], which constitutes a special case of robust Poly when $\omega_L = \omega_R = 0$, $\eta \leq 0$, and flipped similarity coefficients (every edge is weighted by $1 - h(l,r)$ as opposed to $h(l,r)$). We report the results for the best selection of the robust parameters (i.e., $\omega_L, \omega_R, \eta_L$, and η_R). The parameters are optimized simultaneously via grid search with a range from -1 to 1 in incremental steps of 0.1. Since Bipartite and Poly could produce only one-sided matchings, η_R does not influence the maximizing solution and can be set arbitrarily for these problems (we indicate this fact by a dash). As in Table III (middle columns), the most restrictive method, Bipartite, is the weakest. From UFLP to Poly to BiPoly, the constraints are increasingly flexible, and recall and tradeoff improve correspondingly towards the oracular results.

We can compare like-for-like across Robust and Max-weight in Table III to see how different objectives perform, holding the constraints fixed. For instance, robust Bipartite outperforms max-weight Bipartite. The same generally holds for Poly and BiPoly as well. In particular, the robust objective makes a more significant difference on *Multi-Lingual* than on *Cross-Platform*, as the former has noisier similarities arising from the translation from Chinese to English when deriving similarity weights. The robust objective, designed to counter noises, apparently delivers on this count. BiPoly's coverage improves from 41% with max-weight to 47% with the robust objective, substantially closing the gap between BiPoly and the oracle. For subsequent discussion, we focus on the robust objective that is showing better performance, and after all the max-weight objective could always be recovered as a special case ($\omega = 0, \eta = 0$).

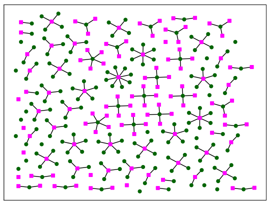
D. Greedy Approximations

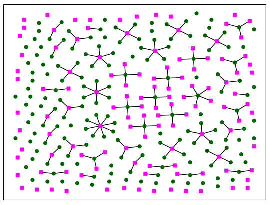
We present the greedy solutions in Table IV. Included as an additional reference is k-core, a many-to-many graph-based matching that provides a performance comparison at the same reduction level offered by BiPoly. For one, BiPoly (as represented by one of its variants C or S) generally outperforms the baselines. For another, we appreciate how closely the greedy solutions approach the optimal by comparing across Tables IV and III. Observably, they get close to the optimal on *Cross-Platform*, which we attribute to its relatively 'cleaner' and presumably more informative similarity weights. On the noisier *Multi-Lingual*, the gap is expectedly larger, yet there is still a credible outperformance over the baselines by BiPoly. In both cases, the difference in tradeoff is less than coverage, as greedy is able to match records that contain more matching entities and only misses out on matches in records with far lower similarity.

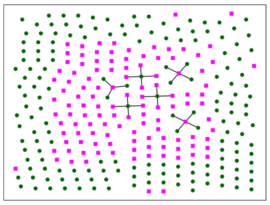
E. Sensitivity Analysis

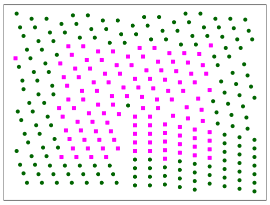
Fig. 6 tracks coverage and tradeoff as reclusivity ω and receptivity η vary respectively on *Multi-Lingual* with BiPoly.S.

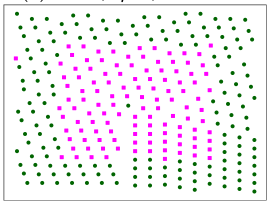
[3]We also investigated the reverse scenario where the 'one' come from the source with more entities, which performs worse in all cases.

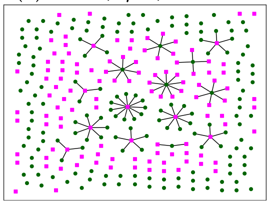

(a) ω=-1, η=0, TO=58.5

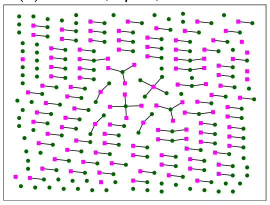

(b) ω=0.1, η=0, TO=58.5


(c) ω=0.4, η=0, TO=52.5


(d) ω=1, η=0, TO=0


(e) ω=0, η=-1, TO=0


(f) ω=0, η=-0.5, TO=63.3


(g) ω=0, η=0.5, TO=63.5

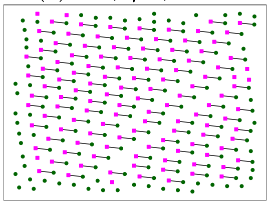
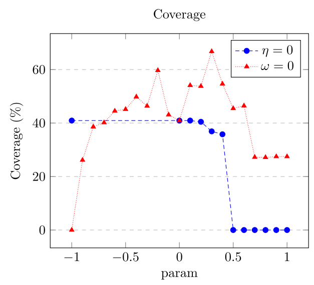
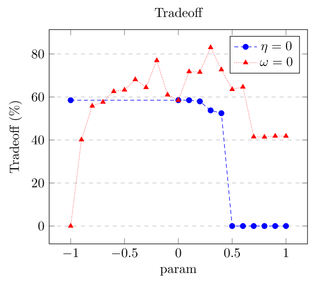
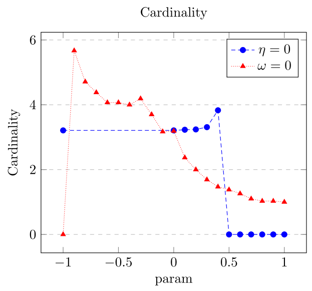

(h) ω=0, η=1, TO=41.8

Fig. 5. Greedy matching outputs while varying reclusivity ω and receptivity η for *Multi-Lingual*. The first row of figures show the output when $\eta = 0$. The second row show the output when $\omega = 0$. *tradeoff* (TO) is indicated for each subfigure.







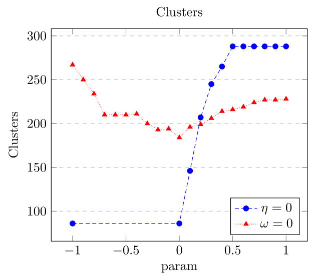


Fig. 6. Varying reclusivity ω and receptivity η for *Multi-Lingual* for BiPoly.S. *param* $= \omega$ when $\eta = 0$ and vice versa.

Varying Reclusivity ω. The first row of Fig. 5 (magenta squares are left nodes, green circles are right nodes) shows the effect of varying ω when $\eta = 0$. To understand this intuitively, we refer to the four illustrations corresponding to $\omega = \{-1, 0.1, 0.4, 1\}$ respectively on *Multi-Lingual*. In the leftmost figure ($\omega = -1$), there is a penalty for being a reclusive host, thus there are relatively few 'singletons'. As we increase ω to 1, there is now a reward for being reclusive, and thus progressively there are more singletons to the point where there are no connected components at all when $\omega = 1$.

This behavior is reflected by the red curves of Fig. 6 that track various measures as ω varies. Recall and tradeoff start out high with lots of matching, decrease as ω increases, and finally collapse when $\omega > 0.5$ as no matching takes place. Structurally, the number of clusters increases with ω with less matching, while the cardinality or the average membership size of those clusters plummets. It is worth noting that the matching result is unchanged $\omega \leq 0$.

Varying Receptivity η. The second row Fig. 5 shows the effect of varying η when $\omega = 0$. The four illustrations correspond to $\eta = \{-1, -0.5, 0.5, 1\}$. The leftmost figure shows that when $\eta = -1$, there is a severe penalty to being a receptive host, and all nodes remain single. As we increase η to -0.5, there is still some penalty, and only a few become receptive hosts, but those few attract a lot of clients. As η increases further, some of the formerly reclusive hosts are now incentivized to be receptive, attracting clients either from other former reclusive hosts or clients of other open hosts. Ultimately, the rightmost figure shows that when $\eta = 1$, we realize as many receptive hosts as possible, but each is part of a small 2-node cluster. The blue curves of Fig. 6 analyze this systematically. In terms of measures, as η goes up, recall and tradeoff initially increase as more clusters appear and eventually decrease as clusters get smaller in cardinality (average number of clients among open hosts). We start out with a large number of clusters of singletons, which reduces over time as fewer clusters of larger cardinality form. Eventually, the number of clusters increases again, some larger clusters break up into pairs.

F. Meta-Blocking

BiPoly is orthogonal and can be applied in conjunction with existing blocking techniques. We introduce matching constraints as meta-blocking techniques [14] and show that BiPoly performs well in this scenario. BiPoly acts as an edge-centric technique, pruning edges that are unlikely a match while maintaining the rest. To illustrate this, we apply keyword blocking with a sample of ten different tokens to each of the datasets followed by the

TABLE V
GREEDY APPROXIMATIONS AFTER KEYWORD BLOCKING

	Cross-Platform			Multi-Lingual		
Constraint	Coverage	Reduction	Tradeoff	Coverage	Reduction	Tradeoff
Bipartite	0.48	**0.93**	0.60	0.34	**0.92**	0.44
UFLP	0.60	0.88	0.70	0.59	0.88	0.69
Poly	0.60	0.88	0.70	0.50	0.90	0.62
BiPoly.C	0.68	0.86	**0.76**	**0.66**	0.86	**0.73**
BiPoly.S	**0.69**	0.86	**0.76**	0.62	0.88	0.72

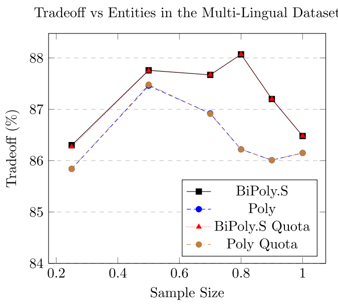


Fig. 7. The effects of quota on noisy similarity.

various matching constraints. We measured the relative difference and averaged the results across the ten tokens. The results are summarized in Table V. BiPoly continues to outperform the baselines in both coverage and tradeoff while achieving an 86% reduction over keyword blocking alone.

G. Noisy Similarity

We examine different extents of noisy similarity, and how the robust objective helps. To simulate a noisier representation of a category entity, we down-sample the number of product titles being aggregated to compute a category representation at various proportions down to a minimum of 25%. We compare the robust Poly (best unidirectional approach) with BiPoly.S. Fig. 7 shows the results for *Multi-Lingual*, averaged over ten runs for each sample size. Robust BiPoly still outperforms robust Poly. As an orthogonal means of countering noisy similarity, we set a quota for the maximum clients of a host. Applying quota (tuned to with the best results) on robust matching yields no appreciable gain on *Multi-Lingual* in most cases.

VI. EXPERIMENT: ENTITY MATCHING SCENARIO

In the second scenario, we are matching directly two distinct granularities of entities: item and category for *Product Taxonomy*, and journals and subject areas for *Journal*.

A. Datasets

We gather two real-world datasets with known ground truths[2] as in Table VI. They encompass four large datasets for scalable approximation solutions.

Product Taxonomy {CN, U.K., US}. Three distinct datasets concern regional platforms of Amazon, respectively China (CN), United Kingdom (U.K.), and United States (US). In each dataset, there are two distinct granularities: some entities are *categories*, while others are *products*. We randomly assign each root category and its descendant categories to one of two partitions. Subsequently, products are extracted with probability p from each category and transferred to the opposing partition. The objective is to match products to their original category in the opposing partition. These large datasets are intended for scalability study (see Section VI-B), whereby we adjust p while holding the number of categories fixed. m and n are the numbers of entities in the left and right partitions respectively. $|e^L|$ and $|e^R|$ are the total number of unique products in the datasets. The last column specifies the number of ground-truth matches.

TABLE VI
SUMMARY OF MATCHING DATASETS

Dataset		m	n	$\|e^L\|$	$\|e^R\|$	Matches
Product Taxonomy	CN	15,774	20,839	1,001,437	758,321	35,176
	UK	11,066	7,219	3,538,298	5,414,148	17,861
	US	65,942	126,681	1,253,119	2,489,225	186,300
Journal		1,032	1,100	5,265	5,266	2,080

Journal. Taken from Wikipedia, this contains all major *journals* and *subject areas*. These entities make up the distinct granularities that we will use to determine the effectiveness of BiPoly in a different domain. The dataset goes through the same transformation as the Product Taxonomy datasets. The statistics are shown in Table VI.

For these datasets, we define its goodness overall in terms of *recall*, *precision* and *F1*.

We make use of the same similarity functions described in Section V-A. For *Product Taxonomy*, the representation of a category is derived only from products not included as test cases to match, while for *Journal*, the representation of a subject area is derived from journals that are not included as test cases to match. For this case, we select the best F1 score (see below) out of the three methods.

B. Results and Scalability

Their greater efficiencies allow the greedy solutions to scale up to larger datasets. We study the complexity and the performance degradation by the greedy methods empirically over various sizes of the three *Product Taxonomy* datasets. The experiments ran on a single thread on a server with 2 x AMD EPYC 7502 @ 2.50 GHz and 512 GB of RAM.

Fig. 8 (left) shows the greedy performance on various scaled versions of *Product Taxonomy* (first three rows correspond to US, U.K., China). We fix the similarity measure, ω, and η for the rest of the runs after searching for the best combination for each method using the smallest-scaled version. The parameters are: $\omega = 0.4$ for bipartite, $\eta_L = -0.9$ for UFLP, $(\omega = 0.7, \eta_L = 0.9)$ for Poly and $(\omega = 0.6, \eta_= 0.9)$ for BiPoly. Fig. 8 shows that BiPoly achieves the highest F1 while Bipartite and K-core fare the worst. The k-core algorithm did not complete on the larger points of US (largest dataset) due to memory constraints. Bipartite suffers the greatest degradation (95%), and BiPoly.S has the least degradation (2.5%), between the smallest and largest datasets.

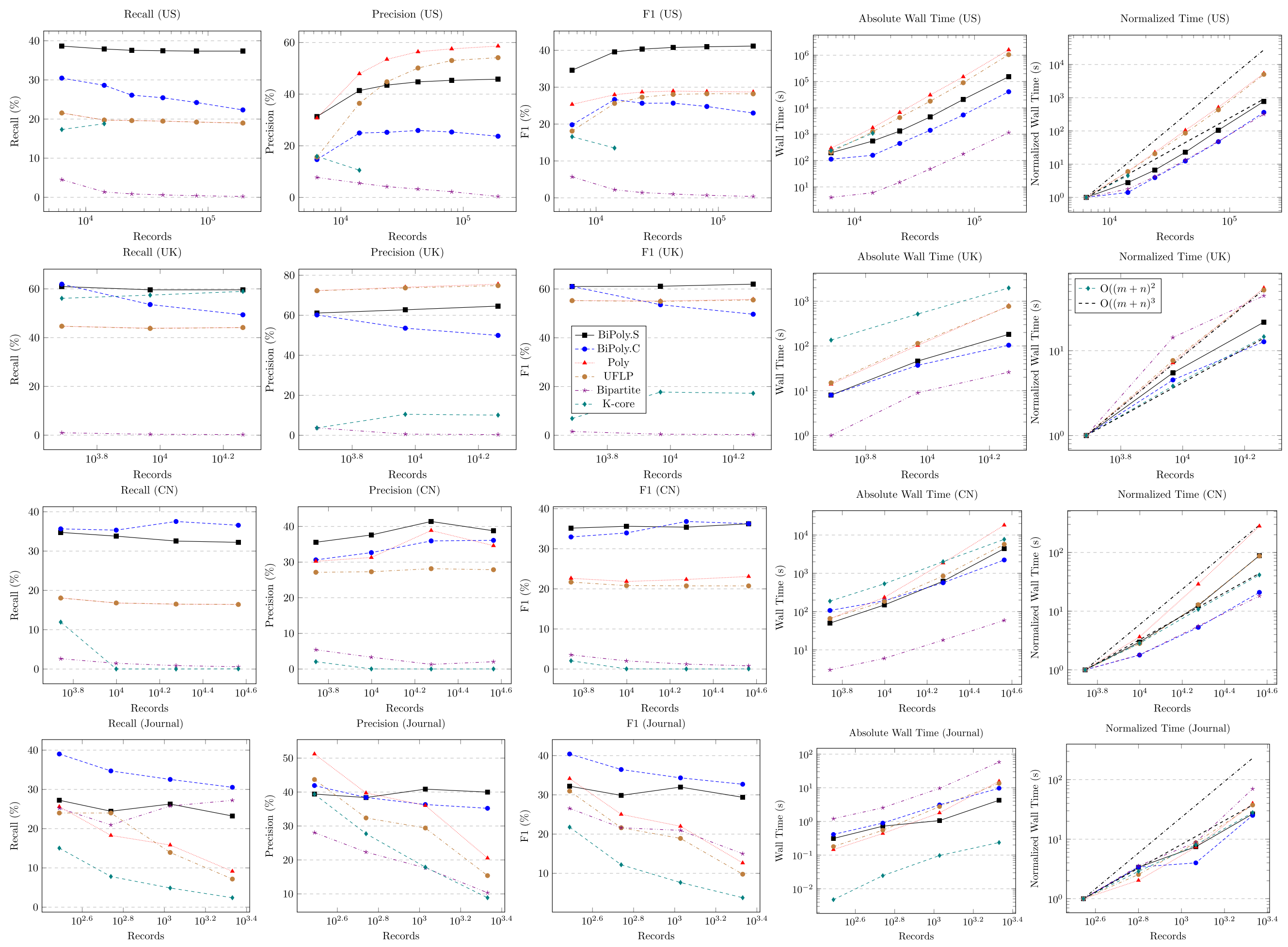


Fig. 8. The left three show the Recall, Precision and F1 of greedy approximation for robust while scaling. The right two show the effects of the number of entities on actual running times and the running time as a factor of the time taken to complete the smallest dataset per method.

We apply these algorithms to the *Journal* domain and the results are shown on the last row of Fig. 8. BiPoly still maintains the highest F1 while k-core fares the worst, which produced results similar to *Product Taxonomy* datasets.

The right figures show the running time of each method on the three datasets. By normalizing each algorithm's runtime by the smallest run, we found that BiPoly.S performs closer to $\mathcal{O}((n+m)^2)$, substantially less than the worst case of $\mathcal{O}((n+m)^3)$ mentioned in Section IV-A. The normalized runtime of BiPoly.C tracks Bipartite very closely as they are bounded by sorting which takes $\mathcal{O}(nm \log nm)$ but it is not surprising that Bipartite has the fastest runtime empirically given that it is the most restrictive. However, it is interesting that UFLP and Poly are less performant empirically than BiPoly. Both UFLP and Poly require $\mathcal{O}(mn \log n)$ to sort the clients, which differs from BiPoly as open hosts are only allowed to form on the right. This also results in a simpler inner loop compared to BiPoly reducing the number of operations for each consecutive round to nm, $n(m-1)$, $n(m-2)$, and so on. This results in a worst-case complexity of $\mathcal{O}(nm^2)$. We refer to the selection process of the best connected component. The best combination of clients for every host needs to be found to select the best set in each round. In UFLP and Poly, hosts (one) and the clients (many) can only be selected from the left and right sides respectively, resulting in the reduction of only one host each round. However, though BiPoly starts out with more potential hosts ($n+m$ instead of n), hosts (likewise clients) can be selected from either side, resulting in larger decrements of potential hosts in subsequent rounds. Thus, BiPoly ends up needing fewer iterations than UFLP or Poly.

VII. Related Work

Assignment Problems. Constraint matching stems from the assignment problem (AP), a combinatorial optimization that assigns 'tasks' to 'agents' [15], [16]. One distinction is how task and agent in AP are designated, whereas an entity could take on either a host or client role in our problem. *Cost-focused* variants of AP minimize (resp. maximize) the overall cost (resp. reward). They differ in constraints. Classical ones include one-to-one [15], [17] and (unidirectional) one-to-many [18], and the extended version, UFLP [1]. The generalized AP (GAP) [16] imposes a quota on one-to-many AP, the $\sum k$ AP only considers the top k-costs, and the k-cardinality AP [19], [20] requires only

k tasks to be completed. *Equitable-focused* variants 'balance' the costs borne by entities, by minimizing "inequality" such as the maximum task cost [21], the difference between the highest and lowest task costs [22] or between the task costs and its mean [23], or the maximum tasks assigned to an agent [24].

For entity matching, the emphasis is placed on cost-focused approaches to maximize similarity weights. Many have equivalent problems in graph theory. The classic AP is known as the weighted bipartite matching which can be solved using the Hungarian algorithm [25], the one-to-many AP is akin to max-weighted poly-matching, and GAP is also known as the b-matching problem and can be solved in $O(n^4 logn)$ time for edges with real weights [26]. In turn, the aforementioned UFLP is an extension of GAP [27]. These instantiations of AP for entity matching have been included as baselines to showcase our distinctions of bidirectionality in poly-matching and robust objective.

We discover mentions of 'robustness' in AP [28], [29], but these polysemously refer to a different concept: the uncertainty of the objective function coefficients.

Entity Matching. There are various directions in improving Entity Matching (EM) [9], [30], [31]. One is to improve the similarity estimation either by better representation of entities [32], [33], [34], or working with multiple attributes [2], [35], or employing supervised similarity learning [36], [37]. Another direction is to improve the efficiency of matching, by blocking [38], [39], [40], hashing [41], or end-to-end workflow [13], [42]. Our work pursues an orthogonal direction in applying 'global constraints' to improve the quality of matching for multi-granular entities. Few prior works have studied leveraging global constraints. While [43], [44], [45] use some constraints, their conditions myopically apply between two records, e.g., *John Doe* and *J. Doe* must live in the same zip code to match. To our best knowledge, the only work that uses global constraints is [3]. While they have applied one-to-one and one-to-many constraints on matching (included as baselines), there have not been any studies on multi-granular datasets or robustness.

This article is a significant extension and comprehensive treatment of an earlier work [46] that appeared in ICDM 2021. Here, we present a formal proof of NP-Hardness, describe an additional approximation algorithm that is more efficient, include three new and significantly larger datasets, and expand the experiments with a scalability study and in-depth sensitivity analysis of the robustness constraint.

VIII. Conclusion

We address bidirectional poly-matching of multi-granular entities. The key to its robustness is the novel notions of reclusivity and receptivity, which cooperatively help to counter the noisy similarity. An optimal solution is formulated via linear programming. We develop more efficient greedy algorithms, which analytically grow polynomially and empirically scale well. Comprehensive experiments validate our contributions and shed light on the workings of the algorithms on real-world datasets. Several directions for future work include exploring the potential benefit of supervised similarity learning, constraint relaxation and investigating tighter bounds of approximation.

References

[1] G. Cornuéjols, G. Nemhauser, and L. Wolsey, "The uncapicitated facility location problem," Cornell University Operations Research and Industrial Engineering, Tech. Rep. 605, 1983.

[2] A. Hashemi, M. B. Dowlatshahi, and H. Nezamabadi-Pour, "A bipartite matching-based feature selection for multi-label learning," *Int. J. Mach. Learn. Cybern.*, vol. 12, no. 2, pp. 459–475, 2021.

[3] J. Gemmell, B. I. Rubinstein, and A. K. Chandra, "Improving entity resolution with global constraints," 2011, *arXiv:1108.6016*.

[4] A. Khan et al., "Efficient approximation algorithms for weighted b-matching," *SIAM J. Sci. Comput.*, vol. 38, no. 5, pp. S593–S619, 2016.

[5] A. Dutta and A. Asaithambi, "One-to-many bipartite matching based coalition formation for multi-robot task allocation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 2181–2187.

[6] S. Guha and S. Khuller, "Greedy strikes back: Improved facility location algorithms," *J. Algorithms*, vol. 31, no. 1, pp. 228–248, 1999.

[7] V. Chvatal, "A greedy heuristic for the set-covering problem," *Math. Operations Res.*, vol. 4, no. 3, pp. 233–235, 1979.

[8] P. Slavık, "A tight analysis of the greedy algorithm for set cover," *J. Algorithms*, vol. 25, no. 2, pp. 237–254, 1997.

[9] O. Hassanzadeh, F. Chiang, H. C. Lee, and R. J. Miller, "Framework for evaluating clustering algorithms in duplicate detection," *Proc. VLDB Endowment*, vol. 2, no. 1, pp. 1282–1293, 2009.

[10] T. Haveliwala, A. Gionis, and P. Indyk, "Scalable techniques for clustering the web," in *Proc. WebDB Informal Proc.*, 2000, vol. 129, p. 134.

[11] M. Vijaymeena and K. Kavitha, "A survey on similarity measures in text mining," *Mach. Learn. Appl.: An Int. J.*, vol. 3, no. 2, pp. 19–28, 2016.

[12] G. Papadakis, E. Ioannou, T. Palpanas, C. Niederee, and W. Nejdl, "A blocking framework for entity resolution in highly heterogeneous information spaces," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 12, pp. 2665–2682, Dec. 2013.

[13] G. Papadakis, L. Tsekouras, E. Thanos, G. Giannakopoulos, T. Palpanas, and M. Koubarakis, "The return of jedAI: End-to-end entity resolution for structured and semi-structured data," *Proc. VLDB Endowmen*, vol. 11, no. 12, pp. 1950–1953, 2018.

[14] G. Papadakis, G. Koutrika, T. Palpanas, and W. Nejdl, "Meta-blocking: Taking entity resolutionto the next level," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 8, pp. 1946–1960, Aug. 2014.

[15] D. W. Pentico, "Assignment problems: A golden anniversary survey," *Eur. J. Oper. Res.*, vol. 176, no. 2, pp. 774–793, 2007.

[16] T. Öncan, "A survey of the generalized assignment problem and its applications," *Inf. Syst. Oper. Res.*, vol. 45, no. 3, pp. 123–141, 2007.

[17] G. T. Ross and A. A. Zoltners, "Weighted assignment models and their application," *Manage. Sci.*, vol. 25, no. 7, pp. 683–696, 1979.

[18] H. Zhu, M. Zhou, and R. Alkins, "Group role assignment via a Kuhn–Munkres algorithm-based solution," *IEEE Trans. Syst. Man Cybern. Part A: Syst. Hum.*, vol. 42, no. 3, pp. 739–750, May 2012.

[19] M. Dell'Amico and S. Martello, "The K-cardinality assignment problem," *Discrete Appl. Math.*, vol. 76, no. 1/3, pp. 103–121, 1997.

[20] A. Volgenant, "Solving the K-cardinality assignment problem by transformation," *Eur. J. Oper. Res.*, vol. 157, no. 2, pp. 322–331, 2004.

[21] R. S. Garfinkel, "An improved algorithm for the bottleneck assignment problem," *Operations Res.*, vol. 19, no. 7, pp. 1747–1751, 1971.

[22] S. Martello, W. R. Pulleyblank, P. Toth, and D. De Werra, "Balanced optimization problems," *Operations Res. Lett.*, vol. 3, no. 5, pp. 275–278, 1984.

[23] S. Gupta and A. Punnen, "Minimum deviation problems," *Operations Res. Lett.*, vol. 7, no. 4, pp. 201–204, 1988.

[24] G. J. Chang and P.-H. Ho, "The β-assignment problems," *Eur. J. Oper. Res.*, vol. 104, no. 3, pp. 593–600, 1998.

[25] H. W. Kuhn, "The Hungarian method for the assignment problem," *Nav. Res. Logistics Quart.*, vol. 2, no. 1/2, pp. 83–97, 1955.

[26] R. P. Anstee, "A polynomial algorithm for b-matchings: An alternative approach," *Inf. Process. Lett.*, vol. 24, no. 3, pp. 153–157, 1987.

[27] G. T. Ross and R. M. Soland, "Modeling facility location problems as generalized assignment problems," *Manage. Sci.*, vol. 24, no. 3, pp. 345–357, 1977.

[28] P. Kouvelis and G. Yu, *Robust Discrete Optimization and Its Applications*, vol. 14. Berlin, Germany: Springer, 2013.
[29] J. Pereira and I. Averbakh, “Exact and heuristic algorithms for the interval data robust assignment problem,” *Comput. Operations Res.*, vol. 38, no. 8, pp. 1153–1163, 2011.
[30] L. Getoor and A. Machanavajjhala, “Entity resolution: Theory, practice & open challenges,” *Proc. VLDB Endowment*, vol. 5, no. 12, pp. 2018–2019, 2012.
[31] G. Papadakis et al., “Three-dimensional entity resolution with jedAI,” *Inf. Syst.*, vol. 93, 2020, Art. no. 101565.
[32] S. Thirumuruganathan et al., “Deep learning for blocking in entity matching: A design space exploration,” *Proc. VLDB Endowment*, vol. 14, no. 11, pp. 2459–2472, 2021.
[33] R. Cappuzzo, P. Papotti, and S. Thirumuruganathan, “Creating embeddings of heterogeneous relational datasets for data integration tasks,” in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2020, pp. 1335–1349.
[34] Y. Li, J. Li, Y. Suhara, A. Doan, and W.-C. Tan, “Deep entity matching with pre-trained language models,” *Proc. VLDB Endowment*, vol. 14, no. 1, pp. 50–60, 2021.
[35] C. Fu, X. Han, J. He, and L. Sun, “Hierarchical matching network for heterogeneous entity resolution,” in *Proc. 29th Int. Joint Conf. Artif. Intell.*, 2020, pp. 3665–3671.
[36] D. Zhang, Y. Nie, S. Wu, Y. Shen, and K.-L. Tan, “Multi-context attention for entity matching,” in *Proc. Web Conf.*, 2020, pp. 2634–2640.
[37] C. Zhao and Y. He, “Auto-EM: End-to-end fuzzy entity-matching using pre-trained deep models and transfer learning,” in *Proc. World Wide Web Conf.*, 2019, pp. 2413–2424.
[38] G. Papadakis, J. Svirsky, A. Gal, and T. Palpanas, “Comparative analysis of approximate blocking techniques for entity resolution,” *Proc. VLDB Endowment*, vol. 9, no. 9, pp. 684–695, 2016.
[39] J. Fisher, P. Christen, Q. Wang, and E. Rahm, “A clustering-based framework to control block sizes for entity resolution,” in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2015, pp. 279–288.
[40] A. Gruenheid, X. L. Dong, and D. Srivastava, “Incremental record linkage,” *Proc. VLDB Endowment*, vol. 7, no. 9, pp. 697–708, 2014.
[41] L. Paulevé, H. Jégou, and L. Amsaleg, “Locality sensitive hashing: A comparison of hash function types and querying mechanisms,” *Pattern Recognit. Lett.*, vol. 31, no. 11, pp. 1348–1358, 2010.
[42] V. Christophides, V. Efthymiou, T. Palpanas, G. Papadakis, and K. Stefanidis, “An overview of end-to-end entity resolution for Big Data,” *ACM Comput. Surv.*, vol. 53, no. 6, pp. 1–42, 2020.
[43] S. E. Whang, O. Benjelloun, and H. Garcia-Molina, “Generic entity resolution with negative rules,” *VLDB J.*, vol. 18, no. 6, pp. 1261–1277, 2009.
[44] S. Chaudhuri, A. Das Sarma, V. Ganti, and R. Kaushik, “Leveraging aggregate constraints for deduplication,” in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2007, pp. 437–448.
[45] H. Liu and Y. Yang, “Bipartite edge prediction via transductive learning over product graphs,” in *Proc. 32nd Int. Conf. Int. Conf. Mach. Learn.*, 2015, pp. 1880–1888.
[46] W. J. Lee, M. Tkachenko, and H. W. Lauw, “Robust bipoly-matching for multi-granular entities,” in *Proc. IEEE Int. Conf. Data Mining*, 2021, pp. 1192–1197.

Ween Jiann Lee (Graduate Student Member, IEEE) received the bachelor’s degree in business management majoring in Operations Management and Analytics from Singapore Management University (SMU). He is currently working toward the PhD degree in computer science at SMU.

Maksim Tkachenko is a computer scientist focusing on natural language processing problems. He served as a data science and operations research advisor with FedEx Express and as a research scientist at the School of Computing and Information Systems, Singapore Management University, where earlier he earned his PhD.

Hady W. Lauw (Senior Member, IEEE) received the PhD degree from Nanyang Technological University. He is an associate professor with SMU, where he leads the Preferred.AI research group working on modeling preferences and recommender systems. Formerly, he served as a postdoctoral Researcher with Microsoft Research in Silicon Valley, as well as a scientist with A*STAR’s Institute for Infocomm Research.