Preference Elicitation for Step-Wise Explanations in Logic Puzzles

Marco Foschini[1], Marianne Defresne[2], Emilio Gamba[3], Bart Bogaerts[1,4], Tias Guns[1]

[1]KU Leuven, Dept. of Computer Science, Celestijnenlaan 200A, 3001 Heverlee, Belgium
[2]Université de Toulouse, LAAS-CNRS, Av. du Colonel Roche 7, 31400 Toulouse, France
[3]Flanders Make, Gaston Geenslaan 8, 3001 Heverlee, Belgium
[4]Vrije Universiteit Brussel, Dept. of Computer Science, Pleinlaan 2, Brussels, Belgium
marco.foschini@kuleuven.com, mdefresn@insa-toulouse.fr, emilio.gamba@flandersmake.be, bart.bogaerts@kuleuven.be, tias.guns@kuleuven.be

Abstract

Step-wise explanations can explain logic puzzles and other satisfaction problems by showing how to derive decisions step by step. Each step consists of a set of constraints that derive an assignment to one or more decision variables. However, many candidate explanation steps exist, with different sets of constraints and different decisions they derive. To identify the most comprehensible one, a user-defined objective function is required to quantify the quality of each step. However, defining a good objective function is challenging. Here, interactive preference elicitation methods from the wider machine learning community can offer a way to learn user preferences from pairwise comparisons. We investigate the feasibility of this approach for step-wise explanations and address several limitations that distinguish it from elicitation for standard combinatorial problems. First, because the explanation quality is measured using multiple sub-objectives that can vary a lot in scale, we propose two dynamic normalization techniques to rescale these features and stabilize the learning process. We also observed that many generated comparisons involve similar explanations. For this reason, we introduce MACHOP (Multi-Armed CHOice Perceptron), a novel query generation strategy that integrates non-domination constraints with upper confidence bound-based diversification. We evaluate the elicitation techniques on Sudokus and Logic-Grid puzzles using artificial users, and validate them with a real-user evaluation. In both settings, MACHOP consistently produces higher-quality explanations than the standard approach.

Code — https://github.com/ML-KULeuven/MACHOP
Extended Version — https://arxiv.org/abs/2511.10436

1 Introduction

The field of Explainable Artificial Intelligence (XAI) aims to build user trust by providing systems with explainable agency. XAI for Constraint Programming (CP) (Rossi, van Beek, and Walsh 2006) aims to explain why a model is unsatisfiable (Liffiton and Sakallah 2008), why a specific variable assignment is chosen (Bogaerts, Gamba, and Guns 2021), or why a solution is optimal (Bleukx et al. 2023). Without explanations, understanding these decisions can be difficult, especially given the complexity of CP models.

Step-wise explanations (Bogaerts, Gamba, and Guns 2021; Bleukx et al. 2026) can explain how decision variables, logically implied by the constraints, are derived. For instance, in nurse rostering, one explanation step might justify assigning a nurse to a night shift by noting that other nurses requested earlier shifts and the night shift must meet minimum staffing. Without this explanation, the user would have to go through the full problem specification to see what led to that decision.

Multiple explanations exist because variable assignments can be derived with different subsets of constraints (Gamba, Bogaerts, and Guns 2023), though they are not all equally interpretable. Defining a criterion to assess explanation quality is crucial for generating comprehensible explanations. One early approach measures explanation quality by the number of used constraints (Ignatiev et al. 2015). However, cardinality is not the only relevant factor, as longer explanations may contain constraints that are more familiar to the user. Later approaches (Bogaerts, Gamba, and Guns 2021; Gamba, Bogaerts, and Guns 2023) introduce a linear objective function to quantify human understandability, which is then optimized to identify the best explanation. Such function is problem-specific and requires defining and weighting each sub-objective, a complex and error-prone process (Mesquita-Cunha, Figueira, and Barbosa-Póvoa 2023).

Interactive optimization methods (Meignan et al. 2015) offer an interesting alternative, where users iteratively express preferences to find their preferred solution. We are especially interested in data-driven approaches that can learn over many related optimization problems at once. In particular, pairwise preference elicitation methods query the user to compare solution pairs, limiting the cognitive load. Such approaches have been extended to combinatorial settings through the Constructive Preference Elicitation (CPE) framework (Dragone, Teso, and Passerini 2017).

We investigate the feasibility of this approach for step-wise explanations and contribute the following:

- Since explanation quality is measured by multiple sub-objectives that vary in scale (e.g., number of constraints, number of facts), we examine how different normalizations impact the quality of the generated explanations.
- To generate more diverse queries, we introduce MACHOP (Multi-Armed CHOice Perceptron), a new query generation criterion for CPE based on non-domination

and Upper Confidence Bound (UCB).

- We examine the trade-off between the user's waiting time and solution quality by pre-determining the facts the solver can explain during learning.
- We validate our contributions through an experimental evaluation using simulated users and logic puzzles, which are standard benchmarks for explainable constraint programming (Bogaerts, Gamba, and Guns 2021; Gamba, Bogaerts, and Guns 2023). A real-user evaluation is then conducted on Sudoku puzzles to demonstrate that MACHOP is able to learn human preferences.

2 Related Work

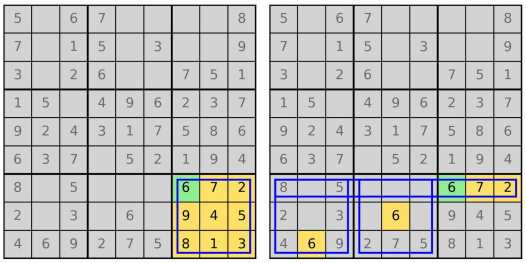
Preference elicitation approaches estimate preferences through user interaction. Pairwise comparisons of solutions are the most common queries, as they reduce cognitive load for the user (Conitzer 2007). As common in utility theory (Braziunas and Boutilier 2007), user preferences are represented by a function over features, each capturing some aspect of a solution. In combinatorial settings, features are interpreted as sub-objectives, often combined with a weighted sum (Benabbou and Lust 2019; Dragone, Teso, and Passerini 2018; Bourdache, Perny, and Spanjaard 2020). While non-linear functions can be learned (Herin, Perny, and Sokolovska 2023), they are often unsupported by solvers, which is why preference elicitation for combinatorial problems relies on linear objective functions (Benabbou and Lust 2019; Defresne, Mandi, and Guns 2025). Additionally, linear models are supported by psychological research on decision making (Dawes 2008) and remain widely used in other domains (Christiano et al. 2017; Handa et al. 2024).

Learning preferences hence correspond to estimating the weight of each objective. Approaches for multi-objective combinatorial problems include polyhedral methods (Toubia, Hauser, and Simester 2004; Benabbou and Lust 2019), which assume error-free user response, and Bayesian approaches (Bourdache, Perny, and Spanjaard 2020), which are computationally expensive. Using preference elicitation for combinatorial settings through CPE is a tractable and robust-to-noise alternative, which also enables generalization across problem instances (Defresne, Mandi, and Guns 2025). Research on CPE has focused on improving the query generation criterion, *i.e.,* the pair of solutions to show to the user, to enhance learning efficiency. Dragone, Teso, and Passerini (2018) generate two solutions, balancing quality and diversification. Defresne, Mandi, and Guns (2025) consider uncertainty over the solutions' utility, but it depends on a large pre-computed set of solutions.

3 Background

We begin by formalizing constraint satisfaction problems.

Definition 1. *A **constraint satisfaction problem** (CSP) (Rossi, van Beek, and Walsh 2006) is a triple $\langle \mathcal{X}, \mathcal{D}, \mathcal{C} \rangle$ where $\mathcal{X}$ is a set of decision variables, $\mathcal{D}$ is a set of domains D_x of allowed values for each variable $x \in \mathcal{X}$, $\mathcal{C}$ is a set of constraints over a subset of $\mathcal{X}$.*



Features	Left	Right
Facts	8	4
Block Cons.	1	2
Row Cons.	0	1
Column Cons.	0	0

Figure 1: Two Sudoku explanation steps that explain $\texttt{cell}[7,7] = 6$ (in green). Used facts $\mathcal{E}$ are in yellow, while used constraints $\mathcal{S}$ are in blue. The table provides an example of the mapping from explanations to features.

A constraint is described by an expression (*e.g.,* $x+z \neq 1$, $a \vee b$) that restricts the values that can be assigned to its variables. A (partial) assignment $\mathcal{I}$ is a (partial) mapping from variables to values within their domains; each variable-value pair is called a fact. An assignment satisfies (or falsifies) a constraint if the constraint evaluates to true (or false). A solution y is an assignment that satisfies all constraints. The set $\mathrm{Sol}(\langle \mathcal{X}, \mathcal{D}, \mathcal{C} \rangle)$ is the set of all such solutions. A set of constraints $\mathcal{S} \subseteq \mathcal{C}$ is satisfiable if there exists an assignment that satisfies all constraints in $\mathcal{S}$; otherwise it is unsatisfiable.

3.1 Explanation steps

Explainable facts logically follow from the CSP's constraints, *i.e.* all solutions that assign the same value to the variable. These can be explained with an *explanation step*:

Definition 2. *Let $\mathcal{I}$ be a partial assignment of a satisfiable CSP $\langle \mathcal{X}, \mathcal{D}, \mathcal{C} \rangle$. An **explanation step** e is a triple $(\mathcal{E}, \mathcal{S}, \mathcal{N})$, also denoted $\mathcal{E} \wedge \mathcal{S} \Rightarrow \mathcal{N}$, such that:*

- *$\mathcal{E} \subseteq \mathcal{I}$ is a set of facts $x = v$ where $x \in \mathcal{X}$ and $v \in D_x$;*
- *$\mathcal{S} \subseteq \mathcal{C}$ is a set of constraints.*
- *$\mathcal{N}$ is a set of explainable facts $x = v$, such that x is assigned the value v in all solutions of $\mathcal{E} \cup \mathcal{S}$.*

Fig. 1 visualizes two explanation steps of a Sudoku grid.

An explanation for $\mathcal{N} = \{x = v\}$ can be verified through a proof by contradiction: assume $\{x \neq v\}$, since $(\mathcal{E} \wedge \mathcal{S})$ is true, if $(\mathcal{E} \wedge \mathcal{S} \wedge \{x \neq v\})$ is unsatisfiable, we have proven $\{x = v\}$. Therefore, finding a (minimal) explanation step is closely related to finding *Minimal Unsatisfiable Subsets*.

Definition 3. *A subset $z \subseteq \mathcal{C}$ is a **Minimal Unsatisfiable Subset** (MUS) if z is unsatisfiable and all strict subsets of z are satisfiable.*

Given an MUS: $\mathcal{E} \cup \mathcal{S} \cup \{x \neq v\}$, it yields a *minimal* explanation step ($\mathcal{E} \wedge \mathcal{S} \Rightarrow \{x = v\}$). Such MUSs are the core for generating a sequence of explanation steps for a set of explainable facts $\mathcal{T}$ (Gamba, Bogaerts, and Guns 2023).

Definition 4. *Given a CSP $\langle \mathcal{X}, \mathcal{D}, \mathcal{C} \rangle$, a set of facts $\mathcal{T}$ to be explained, and a partial assignment $\mathcal{G}$, an **explanation sequence** of length n is a sequence $\langle (\mathcal{E}_i, \mathcal{S}_i, \mathcal{N}_i) \rangle_{1 \leq i \leq n}$ of explanation steps where:*

- $\mathcal{E}_i \subseteq \mathcal{G} \cup \bigcup_{1 \leq j < i} \mathcal{N}_j$ *for all* $1 \leq i \leq n$.
- $\bigcup_{1 \leq j \leq n} \mathcal{N}_j = \mathcal{T}$.
- *the sets $\mathcal{N}_i$ are pairwise disjoint.*

Each step of the sequence derives a new assignment from $\mathcal{T}$, possibly using earlier derived facts ($\mathcal{G} \cup (\bigcup_{1 \leq j < i} \mathcal{N}_j)$).

3.2 Preferred Explanations

Each explainable fact in $\mathcal{T}$ can be explained in many ways, varying in facts and constraints used. A common approach is to compute cardinality-minimal explanations (Ignatiev et al. 2015), which use the fewest facts and constraints. We will refer to these as smallest explanation steps (**SES**). Cardinality alone may not be the most relevant aspect. In Fig. 1, the explanation on the right is smaller (4 facts + 3 constraints), but the one on the left is easier to understand (8 facts + 1 constraint). To capture different quality aspects of an explanation step, we represent it as a vector of *features*.

Definition 5. *Let $\mathcal{M}(\mathcal{C}, \mathcal{I}, \mathcal{T})$ be the set of all explanation steps, given constraints $\mathcal{C}$, a partial assignment $\mathcal{I}$ and a target set of facts $\mathcal{T}$. We define $\phi : \mathcal{M}(\mathcal{C}, \mathcal{I}, \mathcal{T}) \to \mathbb{R}^p$, mapping an explanation step to a vector of real-valued features.*

These features reflect dimensions of explanation quality by counting specific subsets of constraints. In Sudoku, they could represent the number of facts or the number of row/column/block constraints involved. These features can be used as *sub-objectives* and be optimized to compute an Optimal Explanation Step (**OES**). Following the approach from Gamba, Bogaerts, and Guns (2023), preferences are expressed as a linear function f_{w^*}, where $w^* \in \mathbb{R}^p$ represents the importance of each sub-objective ϕ_i.

Definition 6. *Let $f_w(\phi(y)) = \sum_{i=1}^{p} w_i \cdot \phi_i(y)$ be a linear scalarizing function over the features of an explanation step $y \in \mathcal{M}(\mathcal{C}, \mathcal{I}, \mathcal{T})$, $w \in \mathbb{R}^p_{>0}$. An explanation step $y \in \mathcal{M}(\mathcal{C}, \mathcal{I}, \mathcal{T})$ is an **Optimal Explanation Step** (OES) if $\forall y' \in \mathcal{M}(\mathcal{C}, \mathcal{I}, \mathcal{T})$ $f_w(\phi(y)) \leq f_w(\phi(y'))$.*

Finding an OES entails identifying the optimal explanation across all unexplained facts in $\mathcal{T}$. An optimal explanation step consists of selecting both an unexplained fact (*e.g.,* the cell of a Sudoku) and its explanation. To compute an optimal explanation sequence, the concept of OES is extended:

Definition 7. *An explanation sequence $\langle (\mathcal{E}_i, \mathcal{S}_i, \mathcal{N}_i) \rangle^*_{1 \leq i \leq n}$ is optimal according to w if each triplet $(\mathcal{E}, \mathcal{S}, \mathcal{N}) \in \langle (\mathcal{E}_i, \mathcal{S}_i, \mathcal{N}_i) \rangle^*_{1 \leq i \leq n}$ is an OES according to f_w and across all remaining unexplained literals in $\mathcal{T}$.*

Since we treat these problems as multi-objective, the notion of domination becomes relevant.

Definition 8. *Given two explanation steps y^1 and y^2, $\phi(y^1)$ dominates $\phi(y^2)$, if $\phi(y^2)$ is not strictly better in any of the sub-objectives (Konak, Coit, and Smith 2006), i.e:*

$$\phi(y^1) \prec \phi(y^2) : \nexists i \in [1, \ldots, p] \; \phi_i(y^2) < \phi_i(y^1) \qquad (1)$$

Algorithm 1: Constructive Preference Elicitation framework

Input: Problems G, Initial weights w^1, No. of iterations T
Output: Learned weights w^t

1: **for** $t = 1$ to T **do**
2: $\quad \mathcal{Y} \leftarrow$ Instance Selection(G)
3: $\quad (y^1, y^2) \leftarrow$ Query Generation$(\mathcal{Y}, w^t)$
4: $\quad (y^+, y^-) \leftarrow$ Label(y^1, y^2)
5: $\quad w^{t+1} \leftarrow$ Weight Update(y^+, y^-, w^t)
6: **end for**
7: **return** w^T

3.3 Constructive Preference Elicitation

Defining weights w by hand is challenging and user-dependent. Instead, we aim to learn them through pairwise comparisons. The CPE framework (Dragone, Teso, and Passerini 2017) has been proposed for learning weights for multi-objective combinatorial problems. In this context, ϕ maps solutions $y \in \mathrm{Sol}(\langle \mathcal{X}, \mathcal{D}, \mathcal{C} \rangle)$ to a vector of real-valued sub-objectives and we assume that the preferences of the user are representable as a linear function over such sub-objectives: $f_w(\phi(y)) = \sum_{i=1}^{p} w_i \cdot \phi_i(y)$. A solution y^+ is preferred over y^-, if and only if, $f_{w^*}(\phi(y^+)) < f_{w^*}(\phi(y^-))$, where w^* are the weights of the user.

Algorithm 1 summarizes the learning process. Given a set of CSPs G, an initial estimation of weights w^1 (*e.g.,* all equal to one) and a number of iterations T, the goal is to learn weights w^T such that optimizing f_{w^T} leads to a desirable solution. At each iteration, an instance $\mathcal{Y}$ is selected (line 2), being a CSP $\langle \mathcal{X}, \mathcal{D}, \mathcal{C} \rangle$, *i.e.* with solution set $\mathrm{Sol}(\langle \mathcal{X}, \mathcal{D}, \mathcal{C} \rangle)$. Two solutions are then proposed to the user (line 3). They then choose their preferred solutions or indicate no preference (line 4). If the user expresses a preference, the weights are updated accordingly (line 5).

Query Generation. In CPE, the Choice Perceptron (Dragone, Teso, and Passerini 2018) generates a solution pair (y_1, y_2) by solving the following optimization problems:

$$\begin{aligned} y^1 &= \operatorname*{argmin}_{y \in \mathcal{Y}} \; f_{w^t}(\phi(y)) \\ y^2 &= \operatorname*{argmin}_{y \in \mathcal{Y}} \; (1 - \gamma)\, f_{w^t}(\phi(y)) - \gamma\, \alpha(y, y^1) \\ &\text{s.t.} \quad \phi(y^2) \neq \phi(y^1) \end{aligned} \qquad (2)$$

y^1 is computed by minimizing the learned objective function f_{w^t}, while y^2 is generated by minimizing f_{w^t} while maximizing a diversification metric α, namely the L1 distance:

$$\alpha_{L1}(y, y^1) = \sum_{i=1}^{p} |\phi_i(y^1) - \phi_i(y)| \qquad (3)$$

The parameter $\gamma = \frac{1}{t}$ balances explanation quality and diversification, reducing the importance of diversification as the number of iterations increases.

Weight Update. Given (y^1, y^2), the user selects their preferred solution y^+, with the alternative denoted as y^-. To update the weights w^t accordingly, the Choice Perceptron

relies on an update rule inspired by the Preference Perceptron (Shivaswamy and Joachims 2015). Assuming the minimization of sub-objectives, it is defined as:

$$w^{t+1} = w^t + \eta(\phi(y^-) - \phi(y^+)) \tag{4}$$

where η is the learning rate.

4 Extending CPE to Explanation steps

While extending CPE to explanation steps, we also propose three improvements: instance selection to reduce user wait time, enhanced diversification during query generation, and normalization of sub-objectives to stabilize learning.

4.1 Instance Selection

We consider a multi-instance setting, which involves both multiple CSPs (*e.g.* multiple sudoku start grids) and multiple individual explanation steps of that CSP (*e.g.* multiple sudoku steps). An instance $\mathcal{Y}$ in our setting will be a CSP 'state' from a CSP g, namely $\langle \mathcal{C}_g, \mathcal{I}_g, \mathcal{T}_g \rangle$, with $\mathcal{C}_g$ its constraints, $\mathcal{I}_g$ a partial assignment (facts) of it, and $\mathcal{T}_g$ the remaining explainable facts. It has a corresponding set of possible explanation steps $\mathcal{M}(\mathcal{C}_g, \mathcal{I}_g, \mathcal{T}_g)$, each including one fact to explain and a subset of facts/constraints explaining it.

All instances will be assessed with the same p sub-objectives. We assume stationary user preferences across instances, so a single weight vector must be learned across all.

For instance selection, we randomly choose a CSP and iteratively add one explainable fact each query, resulting in a new instance each time. Once all explainable facts are added, we pick a new random CSP. We consider two options for determining *which* explainable fact to add after each iteration:

Online fact selection. In online fact selection, we start with the initial CSP. Query generation of Algorithm 1 computes two explanation steps from all possible explanations. We store the fact with the best utility based on w^{t+1}, *i.e.,* the updated weights after the user expresses their preference (the fact from y^1 is selected if $f_{w^{t+1}}\phi(y^1) < f_{w^{t+1}}\phi(y^2)$). The next returned instance will be $\mathcal{M}(\mathcal{C}_g, \mathcal{I}_g, \mathcal{T}_g)$ where all stored facts for CSP g are removed from $\mathcal{T}_g$ and added to $\mathcal{I}_g$.

Offline fact selection. The online fact selection is computationally expensive, because the query generation on line 3 optimizes over all unexplained facts. The longer this process takes, the longer the user has to wait for the next query.

We propose two simple offline alternatives, where a sequence of facts is precomputed. Each time instance selection is called and $\mathcal{M}(\mathcal{C}_g, \mathcal{I}_g, \mathcal{T}_g)$ is returned, $\mathcal{T}_g$ will consist of one predetermined fact, so the query generation will only need to search over the candidate explanations of this fact. The two alternatives are:

- **Random**: a sequence of explainable facts is randomly selected upfront.
- **SES**: a sequence of facts is generated such that they are associated with the smallest explanation step.

4.2 Query Generation

The default query generation is defined in Eq. 2. We use the OCUS algorithm (Gamba, Bogaerts, and Guns 2023) to solve the optimal (constrained) explanation generation problems, which support linear objective functions and side constraints. To improve query generation for our setting, we propose two changes concerning how y^2 is computed:

Non-domination criteria. In the Choice Perceptron, y^2 is computed by jointly optimizing f_{w^t} and α (Eq. 2). The quickly decreasing γ parameter controls the trade-off. Therefore, after a few iterations, y^2 often corresponds to the second-best explanation under f_{w^t}. We empirically observed that y^2 is often dominated by y^1, making the preference label trivial to infer. Assuming the optimization direction (minimize or maximize) for each objective is known, we can ensure non-domination between explanations with a disjunctive constraint (Sylva and Crema 2004):

$$\bigvee_{i=1}^{p} \phi_i(y^2) < \phi_i(y^1) \tag{5}$$

ensuring one of the p objectives improves compared to y^1.

Weighting schemes. The function α is used to diversify y^2 from y^1. We propose to further enhance this diversification by weighing the different components of α differently:

$$\alpha_u(m, y^1) = u \cdot \alpha(m, y^1) \tag{6}$$

This strategy uses the current estimate of the preference weights as a guide: $u = w^t$. This weighs components based on estimated importance. It favors a solution differing in the most important objectives identified so far. However, it may over-focus on objectives marked as important early on. To address this, we draw inspiration from the Upper Confidence Bound (UCB), used in multi-armed bandits (Auer, Cesa-Bianchi, and Fischer 2002). UCB focuses on high-reward actions (sub-objectives in our case) as well as those less explored. We extend it to preference elicitation by introducing MACHOP (Multi-Armed CHOice Perceptron), prioritizing the diversification of objectives that are either estimated to be important or insufficiently explored. Given Q, the set of all proposed queries, MACHOP is defined as:

$$u_i = q_i + 2\sqrt{\frac{\log(|Q|)}{N_i}} \tag{7}$$

$$q_i = \frac{\sum_{\forall (y^+, y^-) \in Q} \phi_i(y^+) < \phi_i(y^-)}{N_i} \tag{8}$$

$$N_i = \sum_{\forall (y^+, y^-) \in Q} \phi_i(y^+) \neq \phi_i(y^-) \tag{9}$$

Eq. 7 is the UCB formulation. In multi-armed bandits, N_i is how many times the i^{th} arm has been pulled, while q_i is the average reward. We reinterpret N_i as the count of solution pairs where the sub-objectives differ (Eq. 9), indicating how many times we explore that trade-off; q_i is the average number of pairs in which the solution picked has that sub-objective improved (Eq. 8). A higher value of q_i implies that the i^{th} objective is important, as explanations with improvements in that objective are more frequently selected. When $N_i = 0$, $u_i = \infty$, prompting exploration of that objective.

4.3 Weight Update

The weight update (Eq. 4) is sensitive to the scale of the sub-objectives, as such ϕ can benefit from being **normalized** to be on the same scale.

Approximate Nadir Point Normalisation. A standard strategy is to normalize each sub-objective by its lower (f^{lb}) and upper (f^{ub}) bound (Özlen and Azizoglu 2009; Defresne, Mandi, and Guns 2025). Computing the lower bound of each sub-objective is straightforward and involves minimizing just that objective. Determining the upper bound, or nadir point, is more complex and approximations are used in practice (Özlen and Azizoglu 2009). In our case, the sub-objectives are based on the facts and constraints, and we want to find out the maximum value for each sub-objective. For this, for each considered problem $g \in G$, we consider the set $\mathcal{I}^*$ of partial assignments, where each $\mathcal{I}_j \in \mathcal{I}^*$ assigns values to all variables except one. The unassigned variable defines the corresponding target set $\mathcal{T}_j$. These partial assignments provide access to the maximum amount of facts; for each partial assignment $\mathcal{I}_j \in \mathcal{I}^*$, we then compute the explanation step m that maximizes the objective ϕ_i. The highest value among these is the approximate nadir point for that sub-objective.

$$f_i^{ub} = \max_{\mathcal{I}_j \in \mathcal{I}^*} \max_{m \in \mathcal{M}(\mathcal{C}, \mathcal{I}_j, \mathcal{T}_j)} \phi_i(m) \tag{10}$$

This approach overestimates the upper bound, which may result in low-scale normalized $\phi_i(y)$. To avoid overestimating these bounds, we investigate two ways to normalize based on the computed pairs:

Cumulative Normalization. The upper bound of each sub-objective is defined as the maximum encountered value during all training so far. Initially, f_i^{ub} is set to 1.

$$f_i^{ub} = \max(\phi_i(y^1), \phi_i(y^2), f_i^{ub}) \tag{11}$$

Local Normalization. The upper bound of each sub-objective is defined as the maximum value of ϕ_i of the most recent pair (y^1, y^2). If both are zero, f_i^{ub} is set to 1.

$$f_i^{ub} = \begin{cases} \max(\phi_i(y^1), \phi_i(y^2)) & \text{if } \phi_i(y^1) \neq \phi_i(y^2) \neq 0 \\ 1 & \text{otherwise} \end{cases} \tag{12}$$

5 Experimental Evaluation

To experimentally evaluate our method, we address the following research questions:

- **Q1** To what extent does the non-domination constraint improve the quality of the learned explanation sequence?
- **Q2** How does the normalization affect the quality of the learned explanation sequence?
- **Q3** How do the weighting schemes affect the quality of the learned explanation sequence?
- **Q4** What trade-off between runtime and explanation quality can be reached by offline fact selection?
- **Q5** How does our method perform when learning the preferences of real users?

5.1 Problems

Sudoku. We generate Sudoku puzzles with QQWing (Ostermiller 2011), and produce explanation steps by using a Boolean encoding of the problem.

Logic-Grid puzzles. A Logic-Grid puzzle, also known as Einstein puzzles, consists of sentences (*clues*) over a set of occurring entities. The goal is to determine the associations between these entities. Constraints can be classified into three categories (Bogaerts, Gamba, and Guns 2021): *clues*, *transitivity constraints* and *bijectivity constraints*. A fact is *positive* if one entity is associated with another, or *negative* otherwise. We consider the problems from Gamba, Bogaerts, and Guns (2023). An example is in the extended version.

5.2 Definition of the sub-objectives

To define explanation sub-objectives, we measure the distance from any constraint to the explained fact $x = v$. Constraints at distance one are those that involve variable x. We consider the (bipartite) variable-constraint graph where variables are linked to the constraints they appear in, allowing us to group constraints by their distance in this graph to x. For facts, *i.e.,* $y = z$, we group the facts based on the constraint-graph distance of x to y as well as on their values v and z.

Sub-objectives for Sudoku. Constraints for Sudoku are either `facts` or `alldifferent` constraints. We group the constraints into *adjacent* constraints (constraints at graph distance 1 from the fact to explain $x = v$) and other constraints (distance > 1). Facts are categorized based on their adjacency to the explained fact and whether they assign the same value as the one being explained. Constraints are split into separate subgroups for row, column and block constraints.

Sub-objectives for Logic-Grid puzzles. Following the approach used for Sudoku, we categorize constraints and facts by distance in the constraint graph. Facts (Boolean variables in this case) are further classified as either being True or False. Constraints are split into: transitivity, bijectivity and clues.

The full list of features is in the extended version.

5.3 User simulation

For simulated users, the response is modelled by an oracle based on the Bradley-Terry model (Bradley and Terry 1952) with indifference. Given a query (y^1, y^2), the probability of a user being indifferent is defined as (Guo and Sanner 2010):

$$P(\phi(y^1) \sim \phi(y^2)) = e^{-\beta |f_{w^*}(\phi(y^2)) - f_{w^*}(\phi(y^1))|} \tag{13}$$

with w^* as the oracle's true preference weights. The probability of indifference depends on the difference in explanation utility according to f_{w^*}. We set $\beta = 1$. We follow a similar approach to Christiano et al. (2017), assuming a 10% chance of mislabeling. Conceptually, real users make errors constantly, regardless of sub-objective distance.

To represent clear preferences for some sub-objectives, weights w^* are generated with an exponential distribution: each component w_i^* is 10^j, where j is chosen uniformly randomly between -2 and 2 (Mischek and Musliu 2024).

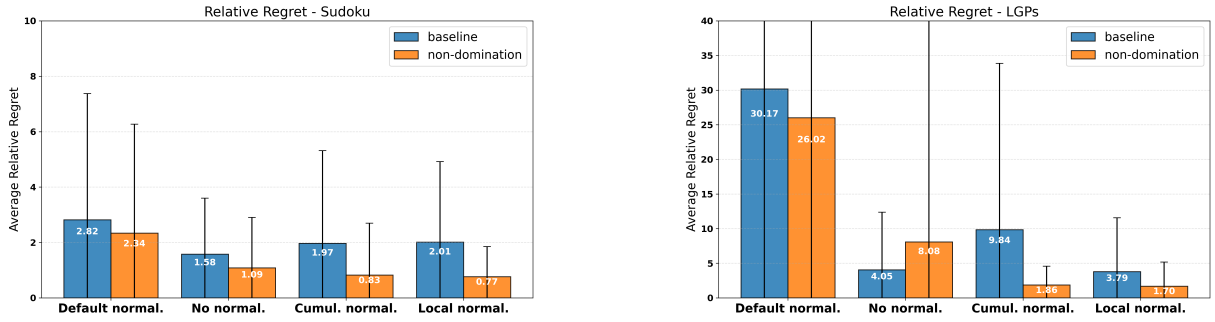


Figure 2: Average relative regret for explanation sequences.

5.4 Metric

The quality of an explanation, generated with the learned objective function f_w, is assessed with relative regret:

$$\begin{aligned} \mathrm{R}(\mathcal{C},\mathcal{I},\mathcal{T},w^*,w) &= \frac{f_{w^*}(\phi(y)) - f_{w^*}(\phi(y^*))}{f_{w^*}(\phi(y^*))} \\ y &= \operatorname*{argmin}_{m\in\mathcal{M}(\mathcal{C},\mathcal{I},\mathcal{T})} f_w(\phi(m)) \\ y^* &= \operatorname*{argmin}_{m\in\mathcal{M}(\mathcal{C},\mathcal{I},\mathcal{T})} f_{w^*}(\phi(m)) \end{aligned} \tag{14}$$

To evaluate the learned objective function in generating a desirable explanation sequence, we compare it to the explanation sequence that w^* would generate. Given a starting instance $\langle \mathcal{C},\mathcal{I},\mathcal{T}\rangle$ of a given CSP, and explanation sequence $E^* = \langle(\mathcal{E}_i^*,\mathcal{S}_i^*,\mathcal{N}_i^*)\rangle_{1\le i\le n}$ that is optimal according to w^*, we compute the average (sequential) relative regret as:

$$\begin{aligned} &\mathrm{R\_seq}(\mathcal{C},\mathcal{I},\mathcal{T},E^*,w^*,w) = \\ &\tfrac{1}{n}\sum_{i=1}^{n} \mathrm{R}\left(\mathcal{C},\mathcal{I}\bigcup_{1\le j<i}\mathcal{N}_j^*,\mathcal{T}-\bigcup_{1\le j<i}\mathcal{N}_j^*,w^*,w\right) \end{aligned} \tag{15}$$

To generate a new explanation step, we add all facts already explained by E^* ($\mathcal{I}\cup_{1\le j<i}\mathcal{N}_j^*$) as facts and generate an explanation over the unexplained ones ($\mathcal{T}-\bigcup_{1\le j<i}\mathcal{N}_j^*$).

5.5 Results

Experimental Setup. Experiments were run on systems with Intel(R) Xeon(R) Silver 4514Y and 256 GB of memory. All methods were implemented in Python, using CPMpy 0.9.24 (Guns 2019). Split OCUS (Gamba, Bogaerts, and Guns 2023) used Exact (Devriendt 2021) for the SAT calls and GurobiPy (Gurobi Optimization, LLC 2024) for the MIP calls. Each training loop consists of 100 queries. A hyperparameter search over $\eta \in [0.1, 0.5, 1, 5, 10]$ was performed by considering 4 oracles and 3 runs. Final results are obtained by aggregating over 10 oracles and 5 runs.

Q1 (non-domination criteria). We assess the impact of preventing dominated explanations using a disjunctive constraint, by comparing the quality of explanations learned with the **baseline** and its **non-domination** variant. The analysis is conducted across all normalization strategies. As shown in Fig. 2, adding the disjunctive constraints yields a significantly lower regret in 7 out of 8 setups. These results support its use, which will be applied in the subsequent experiments.

Q2 (normalization). We evaluate the considered normalization strategies. We compare the **default** normalization strategy based on (approximate) nadir points with **no normalization** and our two proposed normalizations, **cumulative** and **local**. Results in Fig. 2 show that the default normalization performs poorly for our setting, with significantly higher regret. This effect is especially pronounced in Logic-Grid puzzles, where feature value ranges are broader. Similarly, **no normalization** ignores feature scale, which negatively impacts the performance of the non-domination variant. In contrast, both proposed strategies yield the lowest regret, with **local normalization** performing best on average. We therefore adopt it in the following sections.

Q3 (weighting schemes). We compare three weighting schemes for the diversification metric: **no weights**, **learned weights** and **UCB weights**. Results are shown in Fig. 3. Not using any acquired knowledge to guide the diversification (no weights) results in the highest regret for both Sudoku and LGP. Using the learned weights is beneficial for Sudoku but not for LGPs, while UCB weights consistently reduce regret by about 40% for both problems. This suggests that guiding diversification towards both important and unexplored objectives is effective. Overall, MACHOP (non-domination with local normalisation and UCB weighting scheme) reduces the regret by 80% compared to the Choice Perceptron for both Sudoku and LGPs, as shown in Table 1.

Q4 (offline fact selection). We report MACHOP's query generation time in Table 2. When allowing **online** selection of the fact to explain, the waiting time can exceed one minute. Fixing the fact to explain is effective, as both offline sequences (**random** and **SES**) cut query generation time for Logic-Grid puzzles. For Sudoku, only SES speeds up generation time. We observed that random facts can require complex explanations that are costly to compute. However, both offline sequences experience a slight increase in regret compared to the online fact selection. When comparing with results from Fig.3, MACHOP + SES ranks second for LGPs and third for Sudoku, meaning that it offers a reasonable speed/quality trade-off.

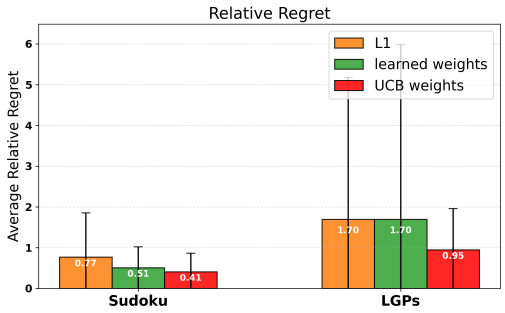


Figure 3: Relative regret for the different diversification strategies for Sudoku and LGPs.

Table 1: Relative regret for Sudoku and LGPs.

	Sudoku	**LGPs**
	Regret	Regret
Choice Perceptron	2.0 ± 2.9	3.8 ± 7.7
MACHOP	$\mathbf{0.4 \pm 0.4}$	$\mathbf{0.9 \pm 1.0}$

Table 2: Relative regret and average query generation time for MACHOP

	Sudoku		**LGPs**	
Chosen Fact	Time (s)	Regret	Time (s)	Regret
Online	35.6 ± 38.3	$\mathbf{0.4 \pm 0.4}$	49.2 ± 15.1	$\mathbf{0.9 \pm 1.0}$
Offline - Random	44.2 ± 44.7	0.7 ± 0.8	11.2 ± 1.9	2.7 ± 9.7
Offline - SES	$\mathbf{12.5 \pm 26.7}$	0.6 ± 0.4	$\mathbf{8.3 \pm 1.0}$	1.4 ± 3.2

Table 3: User study: % picked solutions from SES with MACHOP and Choice Perceptron (ChoPerc) and generation time.

	Evaluation: SES vs MACHOP			**Evaluation: SES vs ChoPerc**			**Query Generation Time**	
Query	SES (%)	MACHOP (%)	Indiff. (%)	SES (%)	ChoPerc (%)	Indiff. (%)	MACHOP (s)	ChoPerc (s)
10	**52.2 ± 19.8**	25.2 ± 16.6	22.6 ± 12.5	**63.3 ± 18.2**	16.2 ± 14.8	20.5 ± 12.0	1.4 ± 0.5	0.9 ± 0.3
30	10.8 ± 12.2	**70.7 ± 18.5**	18.5 ± 13.8	33.2 ± 17.1	**44.8 ± 21.2**	22.1 ± 14.1	2.6 ± 0.9	1.4 ± 0.6
50	13.2 ± 11.2	**72.6 ± 14.9**	14.3 ± 9.3	**42.5 ± 25.2**	38.8 ± 25.7	18.7 ± 13.0	3.0 ± 1.0	1.3 ± 0.5

Q5 (user evaluation). We validate the practical relevance of our approach through a study with 30 participants on Sudoku. Preferences were elicited interactively using both Choice Perceptron and MACHOP, with up to 50 queries. For evaluation, we use SES as a reference point to assess how well the learned objective functions of MACHOP and Choice Perceptron align with user preferences, as SES explanations are reasonable baselines in these domains (Gamba, Bogaerts, and Guns 2023).

We generate an explanation sequence using SES for a new Sudoku puzzle, comprising 56 steps. For each step of this sequence, we extract $\langle \mathcal{C}, \mathcal{I}, \mathcal{T} \rangle$ and generate explanations using each learned objective function. Users are then presented with the learned explanation and SES and asked to pick one. This evaluation was conducted after 10, 30 and 50 pairs as depicted in Table 3. Including both training and evaluation, each user labelled around 400 pairs, taking 45-90 minutes.

As Table 3 shows, when evaluating after 10 queries, SES explanations were preferred more often. After 30, both MACHOP and Choice Perceptron aligned better with users' preferences, with learned explanations being preferred more often. MACHOP achieved stronger alignment than Choice Perceptron, with its explanations preferred over SES in 70.7% of cases, compared to 44.8% of the Choice Perceptron. At 50 queries, MACHOP's performance stayed stable, while Choice Perceptron appears to overfit, likely due to excessive exploitation. For both 30 and 50 queries, no user selected explanations from the Choice Perceptron more frequently than from MACHOP. Statistical tests support this: one-sided Wilcoxon signed-rank (Wilcoxon 1992) gives $p < 10^{-3}$ at 10 queries, $p < 10^{-6}$ at 30 and 50 queries. Cliff's delta (Cliff 1993) is positive in all cases (0.37, 0.62, 0.71).

We also observe that query generation time is low, up to 10 times smaller compared to the oracle experiments. This is due to the artificial oracles' generated utility function sometimes preferring complex explanations, which are costly to compute. With the runtimes observed in our user experiments, real-time learning of preferences becomes feasible.

6 Conclusion

Generating human-understandable step-wise explanations is challenging. Existing methods rely on either cardinality-minimal explanations or problem-specific objectives. We address this by adapting the Constructive Preference Elicitation framework for step-wise explanations. However, the wide range of the sub-objectives can hinder learning, motivating new normalization strategies. Additionally, the state-of-the-art CPE method frequently suggests overly similar explanations; we propose a new query generation strategy based on non-domination and UCB. Experimental results show that our contributions lead to higher-quality explanations in both synthetic and real-user evaluations.

Future work can explore learning preferences as a non-linear utility function, which tends to be more computationally expensive to optimize. Exploring how to define such a function can further help capture aspects missed by the given sub-objectives. Perhaps learning could be further sped up by actively choosing which instance to generate a query for next. Additionally, using queries where users express no preference could accelerate learning too. Finally, the real-user evaluation paves the way for applying MACHOP to more practical and larger-scale scenarios, such as explanations for industrial problems.

Acknowledgments

This research received funding from Flemish Goverment under "Onderzoeksprogramma Artificiële Intellgentie (AI) Vlaanderen.", from Fonds Wetenschappelijk Onderzoek – Vlaanderen (projects G064925N and G070521N) and from the European Union (ERC, CertiFOX, 101122653; ERC, CHAT-Opt, 01002802). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

We also thank the thirty labellers for their time, which made this publication possible.

References

Auer, P.; Cesa-Bianchi, N.; and Fischer, P. 2002. Finite-time Analysis of the Multiarmed Bandit Problem. *Mach. Learn.*, 47(2-3): 235–256.

Benabbou, N.; and Lust, T. 2019. An Interactive Polyhedral Approach for Multi-objective Combinatorial Optimization with Incomplete Preference Information. In Amor, N. B.; Quost, B.; and Theobald, M., eds., *Scalable Uncertainty Management - 13th International Conference, SUM 2019, Compiègne, France, December 16-18, 2019, Proceedings*, volume 11940 of *Lecture Notes in Computer Science*, 221–235. Springer.

Bleukx, I.; Devriendt, J.; Gamba, E.; Bogaerts, B.; and Guns, T. 2023. Simplifying Step-Wise Explanation Sequences. In Yap, R. H. C., ed., *29th International Conference on Principles and Practice of Constraint Programming, CP 2023, August 27-31, 2023, Toronto, Canada*, volume 280 of *LIPIcs*, 11:1–11:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Bleukx, I.; Flippo, M.; Demirović, E.; Bogaerts, B.; and Guns, T. 2026. Using Certifying Constraint Solvers for Generating Step-wise Explanations. In *Proceedings of The 40th Annual AAAI Conference on Artificial Intelligence*. Accepted for publication.

Bogaerts, B.; Gamba, E.; and Guns, T. 2021. A framework for step-wise explaining how to solve constraint satisfaction problems. *Artif. Intell.*, 300: 103550.

Bourdache, N.; Perny, P.; and Spanjaard, O. 2020. Bayesian preference elicitation for multiobjective combinatorial optimization. *CoRR*, abs/2007.14778.

Bradley, R. A.; and Terry, M. E. 1952. Rank analysis of incomplete block designs: I. The method of paired comparisons. *Biometrika*, 39(3/4): 324–345.

Braziunas, D.; and Boutilier, C. 2007. Minimax regret based elicitation of generalized additive utilities. In *UAI*, volume 7, 25–32.

Christiano, P. F.; Leike, J.; Brown, T. B.; Martic, M.; Legg, S.; and Amodei, D. 2017. Deep Reinforcement Learning from Human Preferences. In Guyon, I.; von Luxburg, U.; Bengio, S.; Wallach, H. M.; Fergus, R.; Vishwanathan, S. V. N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, 4299–4307.

Cliff, N. 1993. Dominance statistics: Ordinal analyses to answer ordinal questions. *Psychological bulletin*, 114(3): 494.

Conitzer, V. 2007. Eliciting single-peaked preferences using comparison queries. In Durfee, E. H.; Yokoo, M.; Huhns, M. N.; and Shehory, O., eds., *6th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2007), Honolulu, Hawaii, USA, May 14-18, 2007*, 65. IFAAMAS.

Dawes, R. M. 2008. The robust beauty of improper linear models in decision making. In *Rationality and social responsibility*, 321–344. Psychology Press.

Defresne, M.; Mandi, J.; and Guns, T. 2025. Preference Elicitation for Multi-objective Combinatorial Optimization with Active Learning and Maximum Likelihood Estimation. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2025, August 16-22, 2025, Montreal, Canada*.

Devriendt, J. 2021. Exact Solver. https://gitlab.com/nonfiction-software/exact. Accessed: 2025-07-31.

Dragone, P.; Teso, S.; and Passerini, A. 2017. Constructive Preference Elicitation. *Frontiers Robotics AI*, 4: 71.

Dragone, P.; Teso, S.; and Passerini, A. 2018. Constructive Preference Elicitation Over Hybrid Combinatorial Spaces. In McIlraith, S. A.; and Weinberger, K. Q., eds., *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, 2943–2950. AAAI Press.

Gamba, E.; Bogaerts, B.; and Guns, T. 2023. Efficiently Explaining CSPs with Unsatisfiable Subset Optimization. *J. Artif. Intell. Res.*, 78: 709–746.

Guns, T. 2019. Increasing modeling language convenience with a universal n-dimensional array, CPpy as python-embedded example. In *Proceedings of the 18th workshop on Constraint Modelling and Reformulation at CP (Modref 2019)*, volume 19.

Guo, S.; and Sanner, S. 2010. Real-time Multiattribute Bayesian Preference Elicitation with Pairwise Comparison Queries. In Teh, Y. W.; and Titterington, D. M., eds., *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2010, Chia Laguna Resort, Sardinia, Italy, May 13-15, 2010*, volume 9 of *JMLR Proceedings*, 289–296. JMLR.org.

Gurobi Optimization, LLC. 2024. Gurobi Optimizer Reference Manual.

Handa, K.; Gal, Y.; Pavlick, E.; Goodman, N. D.; Andreas, J.; Tamkin, A.; and Li, B. Z. 2024. Bayesian Preference Elicitation with Language Models. *CoRR*, abs/2403.05534.

Herin, M.; Perny, P.; and Sokolovska, N. 2023. Learning Preference Models with Sparse Interactions of Criteria. In *IJCAI*, 3786–3794.

Ignatiev, A.; Previti, A.; Liffiton, M. H.; and Marques-Silva, J. 2015. Smallest MUS Extraction with Minimal Hitting

Set Dualization. In Pesant, G., ed., *Principles and Practice of Constraint Programming - 21st International Conference, CP 2015, Cork, Ireland, August 31 - September 4, 2015, Proceedings*, volume 9255 of *Lecture Notes in Computer Science*, 173–182. Springer.

Konak, A.; Coit, D. W.; and Smith, A. E. 2006. Multi-objective optimization using genetic algorithms: A tutorial. *Reliab. Eng. Syst. Saf.*, 91(9): 992–1007.

Liffiton, M. H.; and Sakallah, K. A. 2008. Algorithms for Computing Minimal Unsatisfiable Subsets of Constraints. *J. Autom. Reason.*, 40(1): 1–33.

Meignan, D.; Knust, S.; Frayret, J.; Pesant, G.; and Gaud, N. 2015. A Review and Taxonomy of Interactive Optimization Methods in Operations Research. *ACM Trans. Interact. Intell. Syst.*, 5(3): 17:1–17:43.

Mesquita-Cunha, M.; Figueira, J. R.; and Barbosa-Póvoa, A. P. 2023. New e- constraint methods for multi-objective integer linear programming: A Pareto front representation approach. *European Journal of Operational Research*, 306(1): 286–307.

Mischek, F.; and Musliu, N. 2024. Preference Explanation and Decision Support for Multi-Objective Real-World Test Laboratory Scheduling. In Bernardini, S.; and Muise, C., eds., *Proceedings of the Thirty-Fourth International Conference on Automated Planning and Scheduling, ICAPS 2024, Banff, Alberta, Canada, June 1-6, 2024*, 378–386. AAAI Press.

Ostermiller, S. 2011. QQwing–Sudoku Generator and Solver.

Özlen, M.; and Azizoglu, M. 2009. Multi-objective integer programming: A general approach for generating all non-dominated solutions. *Eur. J. Oper. Res.*, 199(1): 25–35.

Rossi, F.; van Beek, P.; and Walsh, T., eds. 2006. *Handbook of Constraint Programming*, volume 2 of *Foundations of Artificial Intelligence*. Elsevier.

Shivaswamy, P.; and Joachims, T. 2015. Coactive Learning. *J. Artif. Intell. Res.*, 53: 1–40.

Sylva, J.; and Crema, A. 2004. A method for finding the set of non-dominated vectors for multiple objective integer linear programs. *Eur. J. Oper. Res.*, 158(1): 46–55.

Toubia, O.; Hauser, J. R.; and Simester, D. I. 2004. Polyhedral methods for adaptive choice-based conjoint analysis. *Journal of Marketing Research*, 41(1): 116–131.

Wilcoxon, F. 1992. Individual comparisons by ranking methods. In *Breakthroughs in statistics: Methodology and distribution*, 196–202. Springer.