Eliminating Inconsistencies among CP-Theory Qualitative Preferences

Erik Rauer
Iowa State University
Ames, IA, United States
etrauer@iastate.edu

Samik Basu
Iowa State University
Ames, IA, United States
sbasu@iastate.edu

Vasant Honavar
Pennsylvania State University
University Park, PA, United States
vuh14@psu.edu

Jia Tao
Lafayette College
Easton, PA, United States
taoj@lafayette.edu

ABSTRACT

Inconsistency in preference reasoning arises when a set of preferences implies that an outcome is preferred over itself. In multi-agent settings, conflicting preferences of the agents lead to inconsistencies in their collective preferences. We examine the problem of establishing consistency by selectively discarding a subset of input preferences, where preferences are expressed qualitatively in CP-theory language. Specifically, we explore two variants (1) identifying a minimal set of preferences to discard in order to eliminate inconsistencies, and (2) finding a set of preferences whose removal minimally alters the induced dominance so as to eliminate the inconsistencies. We show that both minimization problems are NP-complete. We propose an iterative Integer Linear Programming (ILP)-based approach to their solution. Finally, we present experimental results that demonstrate the feasibility of our solution. We observe that optimizing one objective in isolation often compromises the other. We explore sequential strategies that prioritize one objective followed by the optimization of the other, and propose an empirically balanced approach that achieves improved overall outcomes.

KEYWORDS

Qualitative Preferences, CP-theory, Preference Consistency

ACM Reference Format:
Erik Rauer, Samik Basu, Vasant Honavar, and Jia Tao. 2026. Eliminating Inconsistencies among CP-Theory Qualitative Preferences. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 9 pages. https://doi.org/10.65109/WOCS8295

1 INTRODUCTION

Many real-world decision-making scenarios, for example, public policy, product design, healthcare delivery, and environmental regulation, require aggregating the preferences of multiple agents over possible outcomes. Of particular interest are qualitative preferences, described using languages such as CP-net [5], TCP-net [6], and CP-theory [23], which allow each agent to express preferences over outcomes in terms of preferences over properties of outcomes, preferences that hold only under certain conditions, or relative importance of preferences [21]. The semantics of such preferences induce a dominance relation over the possible outcomes described by these properties, resulting in an *induced preference graph* over the outcomes, where the edges of the graph denote the dominance relation [20].

Problem. When the preferences of multiple agents are aggregated, it is inevitable that the preferences of some agents are inconsistent with those of other agents. For example, in deliberations on where to build a new airport, business travelers may prefer to locate the airport close to the city center for ease of access, whereas residents of the city may prefer to locate it far away because of noise. To complicate matters, these preferences may be conditional on other factors, e.g. availability of affordable public transportation, cost of land, economic benefits, etc.

A set of preferences is said to be consistent if the induced preference graph (IPG) is acyclic (which ensures that no outcome dominates itself). The problem of checking the consistency of an IPG is PSPACE-Complete [10]. Recent work has focused on identifying sufficient conditions to efficiently check the consistency of qualitative preferences [17]. Existing approaches for identifying the most preferred, that is, *non-dominated* set of outcomes relative to a given set of preferences, assume that the underlying preferences are consistent [22].

Since consistent preferences induce an acyclic IPG, while inconsistent preferences result in cycles, one might consider using Minimum Feedback Arc Set (MFAS) strategies to address inconsistency. However, simply removing edges from the IPG to eliminate cycles, without considering the underlying preferences that generated those edges, does not guarantee that the resulting acyclic IPG corresponds to any valid subset of the original preferences. This is because a single edge in the IPG can be induced by multiple preference statements, and a single preference can induce multiple edges. Therefore, resolving inconsistencies requires identifying and removing specific preference statements, which will, in turn, remove the associated edges from the IPG. In other words, it is desirable to identify a minimal set of preferences that agents must compromise on to resolve any inconsistencies.

Existing work on aggregating the preferences of multiple agents falls into three broad categories: voting-based mechanisms that

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/WOCS8295

assume preferences are expressed directly over outcomes, rather than over properties of outcomes [7, 18, 19]; voting schemes for preference aggregation under various restrictions of the expressive power of the language used to express the preferences [15, 16]; optimization strategies that minimize different notions of disagreement between agents with respect to their preferences based on pairwise distance between them (more precisely, their CP-net representations) [1, 2]. None of these works addresses the problem of identifying a *minimum* subset of preferences whose removal would make the aggregated preferences consistent.

Contributions. We investigate the problem of eliminating inconsistencies in the aggregated preferences of multiple agents, e.g. by minimally perturbing the preferences. Specifically, we explore two variants of this problem:

(1) Identifying a minimal set of preferences to discard in order to eliminate inconsistencies, and
(2) Finding a set of preferences whose removal minimally alters the IPG so as to eliminate the inconsistencies (cycles in the IPG).

The difficulty of these problems stems from the fact that a single preference statement can induce dominance relationships between multiple pairs of outcomes and, therefore, may participate in multiple cycles within the IPG. Furthermore, a given dominance relationship between a pair of outcomes can be induced by multiple preference statements.

We analyze the computational complexity of these problems and show that they are NP-complete. To solve the problems, we propose an iterative approach based on Integer Linear Programming (ILP).

We present experimental results demonstrating the feasibility of our approach in addressing the key objectives: minimizing preference removal to eliminate inconsistency, and minimizing the impact on the IPG. Our evaluation is conducted on a large dataset of CP-net and CP-theory preferences, synthesized using the technique described in [3].

Our observations indicate that solutions focused solely on minimizing preference removal to resolve inconsistencies do not necessarily minimize impact on the IPG. Conversely, solutions that prioritize minimizing IPG impact may require removing more preferences than necessary. To address this trade-off, we explore strategies that sequentially tackle the two objectives by framing them as a lexicographic optimization problem [27]–first solving one minimization problem, then the other.

Empirical results show that resolving inconsistencies by simply removing the minimal set of preferences strikes an effective balance between the two goals and that introducing a second sequential objective does not significantly reduce the impact of the solution on the IPG.

2 BACKGROUND

CP-Theory [23] offers an expressive formalism to represent qualitative preferences over attributes of alternatives. For instance, one can express a CP-theory preference

$$X = a \land Y = b : Z = z \succ Z = z' \quad [W]$$

In the above, $X = a \land Y = b$ indicates the condition under which the preference $Z = z \succ Z = z'$ holds, while the variable set $[W]$ captures the set of variables whose valuations do not impact the given preference over Z's valuations.

Such a preference induces a dominance relation over outcomes as follows. Consider outcomes o and o' such that the valuation of Z in o is z, the valuation of Z in o' is z', the valuations of all variables other than Z and W are identical and in particular, the valuations of X and Y are a and b, respectively. In that case, as per the semantics of CP-theory preferences, o dominates o' and is denoted by $o \succ o'$. We say that the above preference statement induced the dominance. We will denote each outcome using the set of variable valuations. For instance, $\{a, b, z, w\}$ is an outcome where the valuations of X, Y, Z, W are a, b, z, w, respectively.

A set of CP-theory preferences induces a (transitive) dominance relation over the outcomes, which, in turn, results in an *induced preference graph* (IPG), where the vertices of the graph are the outcomes and the directed edges from one outcome to another capture the dominance relation.

Statement ID	Preference Statement		
p_1		$Y = y_1 \succ Y = y_2$	[X]
p_2	$X = x_1$:	$Y = y_1 \succ Y = y_2$	[ ]
p_3	$Y = y_2$:	$X = x_1 \succ X = x_2$	[ ]
p_4		$X = x_2 \succ X = x_1$	[ ]
p_5		$Y = y_2 \succ Y = y_1$	[ ]

Table 1: A set P of preference statements

Example 1. *Table 1 shows the preference statements over two binary variables $X = \{x_1, x_2\}$ and $Y = \{y_1, y_2\}$. The statements induce a dominance relation captured in Figure 1. Each edge is annotated with the preferences that induce the dominance captured by the edge. The figure does not illustrate the transitive closure relation of dominance.*

Remark 1. *It is worth noting that a preference statement can induce dominance between different pairs of outcomes, and multiple preferences can induce the same dominance. For instance, if we assume that the variables that describe each outcome as X, Y, Z, W (and all are binary variables), then the preference statement*

$$X = a \land Y = b : Z = z \succ Z = z' \quad [W]$$

can induce dominance over the following pair of outcomes

$$\{a, b, z, w\} \succ \{a, b, z', w\}$$
$$\{a, b, z, w\} \succ \{a, b, z', w'\}$$
$$\{a, b, z, w'\} \succ \{a, b, z', w'\}$$
$$\{a, b, z, w'\} \succ \{a, b, z', w\}$$

3 ENSURING CONSISTENCY IN IPG

As noted in the Introduction, a set of preferences is inconsistent if there exists a cycle in the corresponding induced preference graph. This is because any outcome in such a cycle is related to itself by the dominance relation. Our objective is to resolve such inconsistencies.

It is immediate that inconsistency resolution amounts to removing some edges (dominance relationship) in IPG such that the result is acyclic. Based on Remark 1, in the process of removing one edge

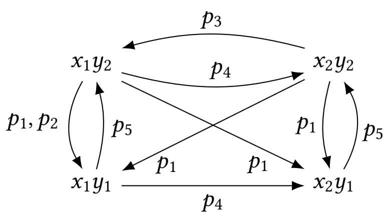

Figure 1: SL-IPG corresponding with preference statements of Table 1

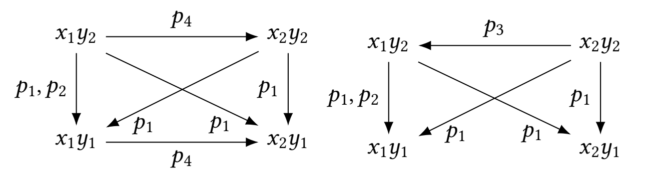

Figure 2: Acyclic SL-IPG after discarding $\{p_3, p_5\}$ and $\{p_4, p_5\}$ from the set of preferences in Table 1

(say, e), we may end up removing multiple edges as the preference(s) inducing e may be responsible for inducing other edges as well. Furthermore, removing one edge may necessitate discarding multiple preferences that label that edge. To account for the impact of edge removals in achieving an acyclic induced preference graph, we augment its definition as follows.

Definition 1. *A Set-labeled IPG, SL-IPG, is a tuple $G = (V, E, A, L)$ where V is the set of outcomes (vertices), $E \subseteq V \times V$ is an ordered edge relation describing the dominance of one outcome over another as per the semantics of the preference statements, A is the set of preferences and L, $L\colon E \to \mathcal{P}(A)$ maps each edge to a set of preferences that induces the edge relation.*

Given a set of preferences, the problem of ensuring consistency amounts to discarding certain preferences such that some edges induced by these preferences are removed from the SL-IPG, resulting in an acyclic induced preference graph. We formulate this problem as an optimization problem where the optimization objective is defined in terms of a set of discarded preferences. In general, one can consider a function that quantifies the *cost* of discarding a given set of preferences and the optimization objective is to minimize such cost while ensuring that the resulting SL-IPG is acyclic. Formally,

Definition 2. *Given an SL-IPG, $G = (V, E, A, L)$ and a cost function $c : \mathcal{P}(A) \to \mathbb{Z}^+$, the general minimum cost inconsistency resolution problem, MCIR$_G$, is to identify $OPT = \texttt{argmin}_{\mathcal{X} \subseteq A}\, c(\mathcal{X})$ such that $E' = E \setminus \{e \mid L(e) \subseteq \mathcal{X}\}$ and $G' = (V, E', A, L)$ is acyclic.*

In this paper, we consider two definitions of the cost function over preferences that are typical in our context: (a) the number of preferences, and (b) the number of edges that are being induced by the preferences. The corresponding cost functions result in two problem instances.

Problem 1. *Given an SL-IPG, $G = (V, E, A, L)$, the minimal preference removal problem, MP$_G$, is the MCIR$_G$ problem using cost function*

$$c_1(\mathcal{X}) = |\mathcal{X}|$$

and the minimal dominance removal problem, MD$_G$, is the MCIR$_G$ problem using the cost function

$$c_2(\mathcal{X}) = |\{e \in E \mid L(e) \subseteq \mathcal{X}\}|$$

In the above, MP$_G$ corresponds to the case where the objective is to minimize the number of preferences being discarded to ensure consistency, while MD$_G$ corresponds to the case where the preferences are discarded such that the number of edges being removed from SL-IPG is minimized.

We will denote the problems as MP and MD without the subscript G for the SL-IPG if the G is immediate from the context of the discussion.

Example 2. *Returning to the preferences of Example 1, it is clear that the corresponding SL-IPG G in Figure 1 contains 8 cycles and is thus inconsistent. There are two optimal solutions to the MP$_G$ problem: $\{p_3, p_5\}$ and $\{p_4, p_5\}$ whose corresponding SL-IPG can be seen in Figure 2. However, since p_4 induces more edges than p_3 does, the only optimal solution to the MD$_G$ problem is $\{p_3, p_5\}$.*

It is important to note that identifying a minimal set of edges to remove–such as in the minimal feedback arc set problem–can lead to solutions that eliminate edges like $(x_2y_1 \to x_2y_2)$, $(x_1y_2 \to x_1y_2)$, and $(x_1y_2 \to x_1y_1)$. While this does result in an acyclic graph, it may contain edges that are no longer induced from any subset of the original preferences. For instance, the preference p_4 becomes problematic: it must be excluded because the edge $(x_1y_2 \to x_2y_2)$ is removed, yet it must also be retained since other edges it induces, such as $(x_1y_1 \to x_2y_1)$, remain in the graph. Another similar problematic preference is p_1.

We note that there exist several sufficient conditions which imply the consistency of a set of CP-Theory preferences [17, 24]. Of these sufficient conditions, the fully acyclic and cardinality-based conditionally acyclic conditions require the acyclicity of a dependency graph whose vertices are the variables that the preferences are on. By labeling the edges of such a dependency graph with the preferences that create them, it becomes a SL-IPG, just on variables instead of outcomes. Thus using any method for solving the MP or MD problems on these graphs would still give a consistent solution. While doing so would have a better runtime than using the SL-IPG, such a solution most likely will not be optimal due to the acyclicity only being a sufficient condition.

4 MP AND MD ARE NP-COMPLETE

4.1 The Two Problems are NP-Hard

It is evident that the MP problem is closely related to the minimum feedback arc set (MFAS) problem [26]. The objective of the MFAS problem is to remove a minimal number of edges from a graph to make it acyclic. The MFAS problem is in the class of NP-Complete problems [13] and can be reduced to the MP problem in polynomial time as follows. For each edge of the input graph $G = (V, E)$ for the MFAS problem, we associate a unique label (in $O(|E|)$ time); the resultant graph $G^T = (V, E, A, L)$ is of type SL-IPG, where $|A| = |E|$ and the labeling L function induces a one-to-one mapping between elements in E and in A. Therefore, a solution to the MP$_{G^T}$ problem is equivalent to a solution to the problem MFAS for G.

A similar reduction applies for the MD problem. Here, since each edge has a unique label that is a single preference, we have that $|\{e \in E \mid L(e) \subseteq \mathcal{X}\}| = |\mathcal{X}|$. Thus, a solution to the MD_{G^T} problem is equivalent to the MFAS for G.

Therefore, MFAS is polytime reducible to both MP and MD, and the problems are at least as hard as the problem MFAS. Hence, the MP and MD problems are in the NP-Hard class.

4.2 The Two Problems are in NP

Consider the decision problem corresponding to the problem MP_G: *Does there exist a set of preferences of size k whose removal eliminates edges from the graph G such that the resulting graph is acyclic?*

Let W be the set of k preferences provided as a solution to the decision problem. One can validate whether W is indeed a solution for the problem in polynomial time. The strategy involves depth-first exploration of the given graph such that any edge e, where $L(e) \subseteq W$ is not used in the exploration. If the exploration terminates without identifying any back edges, then solution W is validated; otherwise, it is not. The runtime for depth-first exploration is $O(|V| + |E|)$. Hence, the given solution can be verified in polynomial time and the MP_G problem is in the NP class.

Now consider the decision problem corresponding to the problem MD_G: *Does there exist a set of preferences whose removal from the graph G results in the elimination of at most k edges, such that the resulting graph is acyclic?*

Let W be the set of preferences provided as a solution to the decision problem. First, one can iterate through all edges of G and count the number of edges e such that $L(e) \subseteq W$. If this number is k or less, continue using the DFS process as described above. Otherwise, this is not a valid solution. Checking that k or less edges are removed can be done in $O(|E|)$ time, so validation can be done in $O(|E| + |V|)$ time. Hence, the given solution can be verified in polynomial time and the MD_G problem is in the NP class.

Based on the above observations, both problems are in the NP-Complete class.

5 EXACT METHOD FOR MP AND MD

We present a solution methodology for MP and MD based on Integer Linear Programming (ILP) formulation for the problems. The formulation amounts to expressing the relationship between cycles, edges in the cycles and the preferences labeling the edges in the SL-IPG as a set of linear equations and computing a satisfiable assignment to these equations taking into consideration an optimization objective expressed over linear combination of some specific variables. We utilize the technique presented in [4] to develop a iterative strategy where in each iteration successively larger problem instances (specified by the number of cycles) are considered to compute the solution to the problem. The advantage of this iterative strategy is that the solution to the problem may not require consideration of all possible cycles in the SL-IPG.

5.1 ILP Formulation for MP_G

Given an SL-IPG G, the MP_G problem can be encoded as an ILP problem as follows. For each preference $k \in A$, let t_k be a binary variable that takes the value 1 if the preference k is removed in the solution, and 0 otherwise. Similarly, for each edge $j \in E$ of G, let s_j be a binary variable that takes the value 1 if the edge j is removed, and 0 otherwise. Finally, for each cycle $c_1, \ldots, c_\ell$, the constant a_{ij} is 1 if edge j is in cycle c_i, and 0 otherwise. Then the ILP formulation for MP_G is:

$$\min \sum_{k \in A} t_k \quad \text{s.t.}$$

$$\begin{aligned}
&(1) \quad \sum_{j=1}^{|E|} a_{ij} s_j \geq 1 && \forall i = 1, \ldots, \ell \\
&(2) \quad s_j \leq t_k && \forall j \in E,\ \forall k \in L(j) \\
&(3) \quad s_j, t_k \in \{0, 1\} && \forall j \in E,\ \forall k \in A
\end{aligned}$$

The objective function is to minimize the number of preferences which are removed, and thus the total number of variables t_k whose values are 1. For each cycle, at least one edge in the cycle should be removed, hence the constraint-set (1). The second set of constraints (2) captures the fact that $s_j = 1$ implies that all preferences $k \in L(j)$ must be removed (i.e., $t_k = 1$). Conversely, when $t_k = 1$ for all $k \in L(j)$, the valuation of s_j can still be 0, it does not have to be 1.

In total there are $|A| + |E|$ variables with $O(|A||E| + \ell)$ constraints, where ℓ is the number of cycles.

5.2 ILP Formulation for MD_G

Given an SL-IPG G, the MD_G problem can be encoded as an ILP problem as follows. Let variables s_j and t_k, as well as constants a_{ij} to be defined as described for the MP_G above along with the constraints (1), (2) and (3). The objective of the problem is to remove all inconsistencies by removing preferences while minimizing the number of edges impacted due to the removed preferences. Hence, it is necessary to include additional constraints such that $s_j = 1$ if and only if $t_k = 1$ for all $k \in L(j)$. In other words, $s_j = \min_{k \in L(j)}\{t_k\}$. This is achieved by introducing two new constraints and a set of binary variables $r_{j,k}$ as follows.

$$\begin{aligned}
&(4) \quad \sum_{k \in L(j)} r_{j,k} = 1 && \forall j \in E \\
&(5) \quad r_{j,k} - 1 \leq s_j - t_k \leq 1 - r_{j,k} && \forall j \in E,\ \forall k \in L(j) \\
&(6) \quad r_{j,k} \in \{0, 1\} && \forall j \in E,\ \forall k \in L(j)
\end{aligned}$$

According to the constraint-set (4), there is exactly one $r_{j,k}$ to be equal to 1 among all k. The constraint-set (5) ensures that when $r_{j,k} = 1$, the valuation of s_j is equal to t_k. These constraints coupled with the constraints (2) and (3) above ensures the following. If for all $k \in L(j)$, $t_k = 1$, then the corresponding s_j is also 1. This is because $r_{j,k} = 1$ for some k by the constraint-set (4), and in this case, $s_j = t_k$ due to the constraints (5). On the other hand, if there is some $k \in L(j)$, $t_k = 0$, then by constraints (2), s_j is equal to 0 as well.

Proceeding further, the ILP formulation for MD_G is:

$$\min \sum_{j \in E} s_j \quad \text{s.t. (1), (2), (3), (4), (5) and (6)}$$

As it has been the case with MP_G ILP, there are $|A| + |E|$ variables with $O(|A||E| + \ell)$ constraints, where ℓ is the number of cycles.

Example 3. *Consider the subgraph G' of the SL-IPG from Figure 1 consisting of only the vertices $\{x_1y_1, x_1y_2\}$ (and the corresponding*

Algorithm 1 Algorithm for Finding Exact Solution to the MP_G or MD_G Problem

1: **procedure** E-ALGO(SL-IPG $G = (V, E, A, L)$)
2: $P \leftarrow \emptyset, C \leftarrow \emptyset$
3: DFS explore G to identify cycles Cyc
4: **while** $Cyc \neq \emptyset$ **do**
5: $C \leftarrow C \cup Cyc$
6: $P \leftarrow \text{ILP}(C)$ ▷ ILP problem solution using C
7: $G' = (V, E', A, L)$ s.t. $E' = E \setminus \{e \mid L(e) \subseteq P\}$
▷ G' is G where edges e with $L(e) \subseteq P$ removed
8: DFS explore G' to identify cycles Cyc
9: **end while**
10: **return** P
11: **end procedure**

edges between them). The ILP formulation of the $MD_{G'}$ for this subgraph is:

$$
\begin{aligned}
\min \quad & s_1 + s_2 \quad s.t. \\
& s_1 + s_2 \geq 1 \\
& s_1 \leq t_1 \quad s_1 \leq t_2 \quad s_2 \leq t_5 \\
& r_{1,1} + r_{1,2} = 1 \quad r_{2,5} = 1 \\
& r_{1,1} - 1 \leq s_1 - t_1 \leq 1 - r_{1,1} \\
& r_{1,2} - 1 \leq s_1 - t_2 \leq 1 - r_{1,2} \\
& r_{2,5} - 1 \leq s_2 - t_5 \leq 1 - r_{2,5} \\
& r_{1,1}, r_{1,2}, r_{2,5}, s_1, s_2, t_1, t_2, t_5 \in \{0, 1\}
\end{aligned}
$$

Where s_1 corresponds with the edge from x_1y_2 to x_1y_1, s_2 with the opposite, and t_1, t_2, t_5 with preferences p_1, p_2, p_5 respectively.

5.3 Iterative Exact Method

Observe that the ILP-based exact method formulation requires identifying all cycles in the SL-IPG G–and the number of cycles in a graph can be exponential in the size of the graph. We present an iterative ILP-based algorithm following the MFAS algorithm discussed in [4]. The primary feature of the algorithm is its *lazy* identification of cycles–detecting them only as needed during the computation. The advantage of this lazy approach is that it may lead to a solution of the ILP problem by only considering a set of "representative" cycles, which is relatively small compared to the total number of cycles in the graph.

Algorithm 1 presents the proposed iterative algorithm. The DFS exploration of the input graph G is performed to identify some cycles *Cyc* (Line 3). In the first iteration of the while-loop (Line 4), the ILP problem with these cycles is solved (Lines 5, 6). The solution identifies some set of preferences P to discard. A new graph G' is generated based on the edges that are removed due to discarding preferences in P. The DFS exploration is conducted on this new graph (Lines 7, 8). In the subsequent iteration, if new cycles are identified in the updated graph, then these cycles are added to already found cycles (Line 5) and the process is repeated; otherwise, in the absence of any new cycles, the set P is returned as the solution.

Note that at worst only one cycle is added in each iteration. This means that there might be an exponential number of calls to the ILP solver. Additionally, the actual number of such calls depends on the order of the DFS, since some starting points might find more cycles than others.

THEOREM 1. *E-ALGO(G) outputs the solution to MP_G or MD_G depending on the ILP encoding.*

Proof. We first show that the ILP formulation of the MP_G problem is correct. Clearly the objective function matches that of the MP_G problem. The constraints must thus enforce two properties: at least one edge must be removed from every cycle, and if an edge is removed, then all preferences labeling that edge must also be removed. Let c_i be a cycle in G, then there exists a constraint in (1) corresponding to c_i. Since all a_{ij} are constants, the only way for this constraint to be satisfied is for some edge variable s_j whose corresponding a_{ij} is 1 to also be 1. However, by definition a_{ij} is 1 if and only if the corresponding edge j is in c_i, so there must be at least one edge $j \in c_i$ whose corresponding variable $s_j = 1$ and is thus removed. The second property is enforced by the constraints (2), which make it so that an edge variable s_j cannot be 1 unless all corresponding preference variables in $L(j)$ are also 1.

To establish the correctness of the ILP formulation for the MD_G problem, the constraints must also ensure two properties: at least one edge is removed from every cycle, and an edge is removed if and only if all preferences labeling that edge have been removed. As was shown for the MP_G ILP formulation, the constraints of (1) and (2) ensure the first such property and one direction of the second. Thus all that is required is the other direction of the second property: if all preferences labeling an edge are removed, that edge is removed. Let $e \in E$ be an edge and suppose that for all $k \in L(e)$, k is removed, i.e. $t_k = 1$. Then there exists exactly one $k \in L(e)$ such that $r_{e,k} = 1$. For this specific k, constraints (4) make it so that $s_e = t_k$.

The correctness of E-ALGO follows from [4]. We present an outline of the proof for the sake of completeness. Suppose that E-ALGO(G) halts on iteration i, let C_i be the set of cycles for this iteration, P_i be the optimal solution to ILP(C_i), and T_i be the corresponding set of preferences that gives P_i. Also let P be the optimal solution to the problem, i.e. P is the solution to ILP($\mathbb{C}$) where $\mathbb{C}$ is the set of all cycles in G. Since the algorithm terminated on iteration i, the removal of T_i makes G acyclic, hence the corresponding solution P_i is a feasible solution to removing all cycles $\mathbb{C}$. Thus $|P| \leq |P_i|$. However, since $C_i \subseteq \mathbb{C}$, ILP($\mathbb{C}$) formulation includes all constraints of ILP(C_i) along with additional constraints corresponding to $\mathbb{C} \setminus C_i$. As the objective function is minimization, additional constraints can only increase the size of the optimal solution. Hence, $|P_i| \leq |P|$, making P_i the optimal solution. □

EXAMPLE 4. *Consider again the preferences of Example 1 and the corresponding SL-IPG of Figure 1. Suppose that the DFS in the first iteration of Algorithm 1 started at the outcome x_1y_2 and that the DFS exploration finds the following cycles:*

Cycle	*Labels*
(x_1y_2, x_2y_1, x_2y_2)	$\{p_1\}, \{p_5\}, \{p_3\}$
(x_2y_1, x_2y_2)	$\{p_1\}, \{p_5\}$
$(x_1y_2, x_2y_1, x_2y_2, x_1y_1)$	$\{p_1\}, \{p_5\}$
$(x_2y_1, x_2y_2, x_1, y_1)$	$\{p_5\}, \{p_1\}, \{p_4\}$

Every cycle here has an edge labeled by $\{p_5\}$, making $\{p_5\}$ the optimal solution to the partial ILP formulation. However, this does not remove the cycle (x_1y_2, x_2y_2), so it is not a complete solution and another iteration of Algorithm 1 is needed.

On the other hand, suppose the DFS exploration started at outcome x_2y_2, and it found the following cycles:

Cycle	*Labels*
(x_2y_2, x_1y_2)	$\{p_3\}, \{p_4\}$
(x_2y_2, x_1y_2, x_2y_1)	$\{p_3\}, \{p_1\}, \{p_5\}$
(x_1y_2, x_1y_1)	$\{p_1, p_2\}, \{p_5\}$

As the first cycle is not labeled by $\{p_5\}$, so the partial solution will give either $\{p_3, p_5\}$ or $\{p_4, p_5\}$ (depending on whether MP_G or MD_G is being solved and the solver used). In either case, it will be one of the optimal solutions given in Example 2, hence this is a "representative" set of cycles and the algorithm will terminate after a single iteration. Observe that, the number of cycles identified in a DFS exploration in each iteration is not a driving factor for the number of iteration necessary to compute the optimal solution.

6 EXPERIMENTAL EVALUATION

6.1 Data Set Generation

We utilize the technique proposed by Allen et al. [3] for random generation of CP-net statements to create a pair of data-sets for our experiments. The first data-set consists of the generated CP-net statements. Since the generated CP-nets are acyclic and consistent, we generate three CP-nets at a time, whose statements are combined, which may result in a (potentially) inconsistent CP-net. This thus mimics combining three separate agent's preferences. We generate CP-nets in this way by varying the number of variables (from 3 to 9). All variables are binary, otherwise the default parameters for the tool of Allen et al. [3] are used.

The preferences of each CP-net are modified in order to create the second data-set. Each generated CP-net preference statement (say p) has the form:

$$\varphi \,:\, Z = z \succ Z = \bar{z}$$

To each p, we randomly associate zero or more variables as relatively less important, ensuring that these additional variables do not overlap with those appearing in φ or Z. Each statement has a 50% chance of staying the same and all eligible variables have the same probability of being relatively less important. The resulting dataset comprises a collection of CP-theory statements.

6.2 Research Queries

As part of our experimental evaluation, we investigate the following research questions:

RQ-1 How many preferences must be removed to achieve consistency, under different problem formulations (MP_G and MD_G)?

RQ-2 What is the minimal impact on the preference graph in terms of edge removal when minimal number of preferences are removed as per the solution to MP_G? What is the minimal number of preferences that correspond to least number of edges necessary to be removed for eliminating inconsistencies as per the solution to MD_G?

RQ-3 How do the runtimes of the different problem formulations compare, and how many iterations are required to compute each solution?

All experiments are conducted on an Intel Core i7-1065G7 CPU at 1.30 GHz running with Python 3.9.10 calling the CBC mixed integer programming solver [9]. Table 2 presents the results of our experiments[1].

6.3 Findings

6.3.1 RQ-1. As expected, the number of preferences and edges removed is minimal for MP_G and MD_G, respectively, owing to the nature of the minimization objective.

In some cases (typically for preference sets with small number of variables), the number of edges impacted by the solution for MP_G and MD_G are identical, however the solution to MD_G removes more preferences than the solution to MP_G. Even when the number of edges removed are different, they are relatively close together. The MP_G solutions remove on average 1.06 times as many edges as the corresponding MD_G solutions for CP-nets, and 1.3 times as many edges for CP-theories. However, the number of preferences removed differs a little more, with the MD_G solutions containing on average 1.61 times as many preferences as the MP_G for CP-nets, and 1.56 times as many preferences for CP-theories.

6.3.2 RQ-2. The observations from RQ-1 are not surprising, as they align with the underlying minimization objectives: minimizing preference removals versus minimizing edge removals induced by those preferences. While solving one problem it may be desirable to also minimize the other objective as much as possible, without compromising on the primary objective. For example, when solving the MP_G problem one may want to minimize the number of edges being removed as much as possible, while keeping the number of preferences removed minimal, or vice versa for the MD_G problem.

However, due to the nature of Algorithm 1 these "secondary objectives" are not guaranteed to be minimized. In fact, the secondary objective can take a range of possible values as it depends on both the set of representative cycles being considered and which solution the ILP solver chose. We can find both the minimum and maximum values of this range through lexicographic optimization [27] by first solving the primary objective as normal, then using the given solution as a constraint when solving the secondary objective.

For example, finding the minimal number of edges impacted by removing the minimal number of preferences can be realized by first solving MP_G to determine the minimal number n of preferences to remove, and then adding the hard constraint that at most n preferences can be removed in the ILP formulation of MD_G. The largest possible number of edges removed can also be found using a similar strategy, except that the objective function of the MD_G ILP formulation is switched to maximization instead of minimization. Finding the smallest and largest number of preferences removed while removing the minimal number of edges can be done in a similar manner, except that MD_G is solved first and its solution is used to modify the ILP formulation of MP_G. The min and max of these ranges can be seen in the last two columns of the "MP_G Stats" and "MD_G Stats" sections of Table 2. This method for finding the

[1] See https://github.com/ErikRauer/EliminatingInconsistenciesQualitativePreferences for all code and data

CP-Nets												
IPG Stats			MP_G Stats					MD_G Stats				
# Vars	**# Prefs**	**# Edges IPG**	**Time (s)**	**Prefs Removed**	**Edges Removed**	**Max Possible Edges Removed**	**Min Possible Edges Removed**	**Time (s)**	**Edges Removed**	**Prefs Removed**	**Max Possible Prefs Removed**	**Min Possible Prefs Removed**
3	21	21	0.01	8	9	9	9	0.02	9	10	17	8
3	20	20	<0.01	3	8	8	8	0.02	8	10	16	3
3	18	23	<0.01	6	11	11	11	0.01	11	11	15	6
4	42	51	0.04	13	19	20	19	0.05	19	14	37	13
4	42	57	0.02	12	25	25	25	0.07	25	18	38	12
4	43	57	0.03	13	26	26	25	0.03	25	23	37	13
5	83	140	0.11	22	63	64	60	0.11	60	26	76	22
5	91	130	0.10	26	52	52	50	0.16	50	30	80	26
5	89	143	0.13	22	68	68	67	0.24	63	46	81	28
6	184	333	0.46	50	148	158	148	0.61	141	78	168	59
6	182	331	0.40	49	145	149	142	1.01	139	71	173	52
6	189	345	0.57	56	155	155	153	0.72	153	75	171	56
7	278	779	2.53	95	356	363	353	3.08	331	122	251	101
7	280	778	2.27	90	358	365	352	2.76	330	126	268	102
7	281	780	2.34	85	352	354	352	1.98	332	149	332	107
8	372	1786	9.49	139	785	814	779	9.89	762	189	287	152
8	377	1816	13.89	125	876	881	871	10.86	792	218	343	158
8	373	1841	14.38	126	887	892	874	10.39	817	201	339	155
9	468	4024	66.00	158	1870	1882	1798	80.62	1720	256	428	202
9	476	4018	68.53	174	1852	1878	1836	65.35	1714	296	432	219
9	472	4048	86.25	173	1966	2000	1966	72.90	1744	283	438	204
CP-Theories												
3	21	33	0.01	8	13	17	13	0.02	13	8	16	8
3	20	30	0.02	3	12	12	12	0.01	12	9	16	3
3	18	34	0.02	7	18	18	18	0.03	14	9	15	9
4	42	105	0.06	13	22	58	31	0.10	31	17	34	13
4	42	75	0.03	12	35	35	27	0.08	25	16	31	16
4	43	90	0.06	13	50	50	45	0.20	36	19	31	18
5	83	222	0.23	22	106	132	106	0.44	89	41	61	41
5	91	230	0.21	26	90	114	88	0.31	70	36	67	33
5	89	242	0.26	22	135	135	134	0.77	101	38	73	33
6	184	637	1.61	53	318	399	314	2.16	227	90	140	84
6	182	598	1.19	49	307	315	307	1.75	224	69	134	68
6	189	529	0.97	57	249	266	238	2.20	217	76	142	68
7	278	1368	41.22	106	697	726	666	447.58	579	137	174	134
7	280	1709	17.61	100	950	973	926	9.63	593	171	235	171
7	281	1849	38.33	98	1182	1202	1165	175.66	644	154	223	152

Table 2: Experimental Results

min of the ranges can also be used as a more balanced heuristic that simultaneously considers both objectives.

Minimizing edges removed by minimal set of preferences as per MP_G. We first consider the balanced approach: minimizing the number of edges removed by a minimal set of preferences comparing the third and the last columns of the MP_G Stats. Experimentally, this combined approach does not yield a significant reduction in the number of edges impacted compared to using MP_G alone. In fact, the possible range for the number of edges removed by MP_G is quite small, with the maximum number being less than 1.07 times the minimum for CP-nets and, with one exception, less than 1.31 times the minimum for CP-theories. The results in terms of the number of edges removed computed in this combined approach are closer to the optimal solution from MD_G, particularly for CP-nets where the number of edges removed is up to 13% more than that of the MD_G solution. However, this advantage is not observed when the input preferences follow CP-theory, where the number of edges removed are up to 81% greater than that of the MD_G solution.

These observations indicate that the MP_G solutions themselves tend to balance both objectives: the number of preferences removed will always be minimal and the number of edges removed is fairly close to minimal (especially for CP-nets). Additionally, the small range of possible values ensure only a small gain in the number of edges removed when a secondary objective is introduced.

Minimizing preferences needed to remove minimal set of edges as per MD_G. We now consider minimizing the number of preferences that still remove a minimal set of edges. In this case, we observe (see last column in Table 2) a significant reduction in the number of preferences removed, particularly when the input preferences are CP-nets. This is likely due to the large possible range in the number of removed preferences that remove the minimum number of edges (difference between last two columns of MD_G Stats). In our observation the maximum values are generally 2-3 times larger than the minimum for CP-nets and 1.25-2.25 times larger for CP-theories. These large ranges indicate that simply solving MD_G on its own might remove significantly more preferences than necessary and allow the secondary objective to provide a substantial improvement. Interestingly, despite the large possible range of preference set sizes for CP-theories, our observed values tended to be very close to the lower end of the range, indicating that the maximum values of the range might just be due to outlier solutions. However, the number of edges affected by the solution to the combined approach (MD_G followed by MP_G) remains far from the minimal number of preference removals achieved by solving MP_G alone. Here, for CP-nets up to 30% more preferences are removed than in the MP_G solutions, while there might be as many as 87% more preferences removed for CP-theories.

6.3.3 RQ-3. To evaluate the execution time overhead of the ILP-based solutions for MP_G and MD_G, we also implemented an ILP formulation of the Minimal Feedback Arc Set (MFAS) problem. As noted in Example 2, MFAS is not suitable for enforcing consistency in preference graphs, since it disregards the underlying preferences that induce the removed edges. Nevertheless, MFAS serves as a useful baseline for estimating the computational cost associated with enforcing acyclicity in graphs.

We observe that MD_G is typically faster than computing solution to MFAS. For instance, for variable size 9 and CP-net preference set inducing a preference graph of size ~ 400 nodes and $\sim 4,000$ edges, ILP formulation of MFAS takes more than $10,000$ seconds, while the solution to MD_G is computed in at most 90 seconds. There are two possible explanations for this observation. The ILP solution for MD_G relies on computing the solution by considering preferences, that are likely to impact multiple edges. Furthermore, there are inconsistencies in preference graph due to direct contradictions between preferences. Each such contradiction would induce several identical/isomorphic cycles in the SL-IPG. Due to their similarity/isomorphism, resolving one of these cycles by removing preferences will resolve all the other cycles as well. Therefore, in order to remove multiple isomorphic cycles, only one of them needs to be considered in the MD_G and MP_G ILP formulations, while MFAS strategy would need to consider most if not all of these cycles, leading to more iterations and in turn a longer execution time.

7 RELATED WORK

A common source of inconsistency in reasoning with qualitative preferences arises in the context of multi-agent decision making process. Even when each agent expresses a consistent set of preferences, the combination of the preferences from all may not be consistent. This problem is often referred to as the preference aggregation problem in the existing literature and has been addressed in primarily two different ways depending on the objective.

One line of work focuses on answering dominance queries–i.e., determining the preference relationship between two outcomes. In this setting, solutions are often based on voting strategies [12, 15, 18, 25]. Voting-based semantics such as Pareto, majority, and max have been proposed to evaluate whether one outcome is preferred over another. Notably, this approach does not require the aggregated preferences to be consistent.

An alternative approach, proposed in [1, 2], seeks to construct a consensus CP-net from a set of input CP-nets. This method relies on the concept of swap disagreements, which are pairs of outcomes o, o' that differ in one variable value and for which the dominance relation in the consensus CP-net differs from that of the input CP-net being considered. The goal is to minimize the total number of such disagreements across all input CP-nets. While this objective might appear aligned with the one addressed in this paper, a key difference between the solutions stems from the fact that the resulting consensus CP-net is not guaranteed to be consistent. For example, consider the following four preferences: $x_1: y_1 \succ y_2$, $y_1: x_2 \succ x_1$, $x_2: y_2 \succ y_1$, and $y_2: x_1 \succ x_2$. Suppose there are four CP-nets, each one agrees with only three of the four preferences. In this case, the four preferences will form the consensus CP-net, however, there exists an outcome (x_1, y_1) which is preferred to itself in the consensus CP-net. In contrast, our problem focuses on minimally eliminating preferences (characterized as MP_G) or edges minimally in terms of preference removal (characterized as MD_G) so that the resultant preference set is consistent. To the best of our knowledge, our work is the first to address this problem.

8 CONCLUSION

We have tackled the problem of establishing consistency by selectively discarding some subset of qualitative preferences expressed in the CP-theory language. We have solved two variants of this problem: (1) identifying a minimal set of preferences to discard in order to eliminate inconsistencies, and (2) finding a set of preferences whose removal minimally alters the induced dominance so as to eliminate the inconsistencies. We have shown that these problems are NP-complete. We have proposed an ILP-based approach to their solution. The results of our experiments establish the feasibility of our approach and indicate that minimal sets of preferences whose removal eliminates inconsistency strikes an adequate balance between the two objectives, particularly when the preferences are CP-nets. Additionally, the experiments suggest that the impact of the removal of any such minimal set of preferences on the induced dominance does not vary much, making it so that the introduction of a sequential objective does not majorly reduce the impact, regardless of which solution set is considered. We plan to explore and apply other strategies from MFAS solutions to address our problem (such as cutting plane algorithm [11], heuristics, and approximation algorithms [8, 14]). Work in progress aims to explore techniques for achieving consistency by minimally altering preferences rather than removing them. Also of interest are techniques for ensuring that consistency is achieved in a manner that is fair to all the agents.

ACKNOWLEDGMENTS

This work is supported in part by the National Science Foundation through grants IIS 2225823 and IIS 2225824.

REFERENCES

[1] Abu Mohammad Hammad Ali, Howard J. Hamilton, Elizabeth Rayner, Boting Yang, and Sandra Zilles. 2021. Aggregating Preferences Represented by Conditional Preference Networks. In *Algorithmic Decision Theory*, Dimitris Fotakis and David Ríos Insua (Eds.). Springer International Publishing, Cham, 3–18. https://doi.org/10.1007/978-3-030-87756-9_1

[2] Abu Mohammad Hammad Ali, Boting Yang, and Sandra Zilles. 2024. Approximation algorithms for preference aggregation using CP-nets. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence (AAAI'24/IAAI'24/EAAI'24, Vol. 38)*. AAAI Press, 10433–10441. https://doi.org/10.1609/aaai.v38i9.28911

[3] Thomas E. Allen, Judy Goldsmith, Hayden Elizabeth Justice, Nicholas Mattei, and Kayla Raines. 2017. Uniform Random Generation and Dominance Testing for CP-Nets. *Journal of Artificial Intelligence Research* 59 (2017), 771–813.

[4] Ali Baharev, Hermann Schichl, Arnold Neumaier, and Tobias Achterberg. 2021. An Exact Method for the Minimum Feedback Arc Set Problem. *ACM J. Exp. Algorithmics* 26, Article 1.4 (April 2021), 28 pages. https://doi.org/10.1145/3446429

[5] C. Boutilier, R. I. Brafman, C. Domshlak, H. H. Hoos, and D. Poole. 2004. CP-nets: A Tool for Representing and Reasoning with Conditional Ceteris Paribus Preference Statements. *Journal of Artificial Intelligence Research* 21 (2004), 135–191.

[6] Ronen I. Brafman and Carmel Domshlak. 2002. Tcp-nets for preference-based product configuration. In *In ECAI02 Workshop on Configuration*.

[7] Cristina Cornelio, Judy Goldsmith, Umberto Grandi, Nicholas Mattei, Francesca Rossi, and K. Brent Venable. 2021. Reasoning with PCP-Nets. *Journal of Artificial Intelligence Research* 72 (2021), 1103–1161.

[8] Guy Even, J. Naor, and B. Schieber. 1998. Approximating Minimum Feedback Sets and Multicuts in Directed Graphs. *Algorithmica* 20 (Feb. 1998), 151–174. https://doi.org/10.1007/PL00009191

[9] John Forrest and Robin Lougee-Heimer. 2005. CBC user guide. In *Emerging theory, methods, and applications*. INFORMS, 257–277.

[10] J. Goldsmith, J. Lang, M. Truszczynski, and N. Wilson. 2005. The computational complexity of dominance and consistency in CP-nets. In *Proceedings of International Joint Conference on Artificial Intelligence*. 144–149.

[11] Martin Grötschel, Michael Jünger, and Gerhard Reinelt. 2022. Comments on "An Exact Method for the Minimum Feedback Arc Set Problem". *ACM J. Exp. Algorithmics* 27, Article 1.3 (July 2022), 4 pages. https://doi.org/10.1145/3545001

[12] Adrian Haret, Arianna Novaro, and Umberto Grandi. 2018. Preference Aggregation with Incomplete CP-Nets. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Sixteenth International Conference (KR 2018)*. AAAI Press, Palo Alto, California, Tempe, United States, 308–317. https://hal.science/hal-03888672

[13] Richard M. Karp. 1972. *Reducibility among Combinatorial Problems*. Springer US, Boston, MA, 85–103. https://doi.org/10.1007/978-1-4684-2001-2_9

[14] Robert Kudelić. 2022. *Papers and Algorithms*. Springer International Publishing, Cham, 19–120. https://doi.org/10.1007/978-3-031-10515-9_3

[15] Jérôme Lang and Lirong Xia. 2009. Sequential composition of voting rules in multi-issue domains. *Mathematical Social Sciences* 57, 3 (2009), 304–324. https://doi.org/10.1016/j.mathsocsci.2008.12.010 Special Issue: Voting Theory and Preference Modeling.

[16] Minyi Li, Quoc Bao Vo, and Ryszard Kowalczyk. 2011. Majority-rule-based preference aggregation on multi-attribute domains with CP-nets. In *The 10th International Conference on Autonomous Agents and Multiagent Systems - Volume 2* (Taipei, Taiwan) *(AAMAS '11)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 659–666.

[17] Erik Rauer, Samik Basu, and Vasant Honavar. 2025. Checking Consistency of CP-Theory Preferences in Polynomial Time. *Proceedings of the AAAI Conference on Artificial Intelligence* 39, 14 (Apr. 2025), 15126–15133. https://doi.org/10.1609/aaai.v39i14.33659

[18] F. Rossi, K. B. Venable, and T. Walsh. 2004. mCP Nets: Representing and Reasoning with Preferences of Multiple Agents. In *Proceedings of the 19th National Conference on Artificial Intelligence (AAAI'04)*. AAAI Press, 729–734.

[19] Francesca Rossi, Kristen Brent Venable, and Toby Walsh. 2011. A Short Introduction to Preferences Between Artificial Intelligence and Social Choice. *Synthesis Lectures on Artificial Intelligence and Machine Learning* (2011), 102.

[20] Ganesh Ram Santhanam, Samik Basu, and Vasant Honavar. 2010. Dominance Testing via Model Checking. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*. AAAI Press, 357–362.

[21] Ganesh Ram Santhanam, Samik Basu, and Vasant Honavar. 2016. *Representing and reasoning with qualitative preferences: Tools and applications*. Morgan & Claypool Publishers.

[22] Ganesh Ram Santhanam, Samik Basu, and Vasant Honavar. 2016. Representing and Reasoning with Qualitative Preferences: Tools and Applications. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 10, 1 (2016), 1–154. https://doi.org/10.2200/S00689ED1V01Y201512AIM031

[23] Nic Wilson. 2004. Extending CP-nets with stronger conditional preference statements. In *Proceedings of the 19th National Conference on Artifical Intelligence* (San Jose, California) *(AAAI'04)*. AAAI Press, 735–741.

[24] Nic Wilson. 2011. Computational techniques for a simple theory of conditional preferences. *Artificial Intelligence* 175, 7 (2011), 1053–1091. Representing, Processing, and Learning Preferences: Theoretical and Practical Challenges.

[25] Lirong Xia, Vincent Conitzer, and Jérôme Lang. 2008. Voting on multiattribute domains with cyclic preferential dependencies. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 1* (Chicago, Illinois) *(AAAI'08)*. AAAI Press, 202–207.

[26] D. Younger. 1963. Minimum Feedback Arc Sets for a Directed Graph. *IEEE Transactions on Circuit Theory* 10, 2 (1963), 238–245. https://doi.org/10.1109/TCT.1963.1082116

[27] A. V. Zykina. 2004. A Lexicographic Optimization Algorithm. *Automation and Remote Control* 65, 3 (01 Mar 2004), 363–368. https://doi.org/10.1023/B:AURC.0000019366.84601.8e