A Better Algorithm for Societal Tradeoffs

Hanrui Zhang
Computer Science Department
Duke University
Durham, NC 27705
hrzhang@cs.duke.edu

Yu Cheng
Computer Science Department
Duke University
Durham, NC 27705
yucheng@cs.duke.edu

Vincent Conitzer
Computer Science Department
Duke University
Durham, NC 27705
conitzer@cs.duke.edu

Abstract

In the societal tradeoffs problem, each agent perceives certain quantitative tradeoffs between pairs of activities, and the goal is to aggregate these tradeoffs across agents. This is a problem in social choice; specifically, it is a type of quantitative judgment aggregation problem. A natural rule for this problem was axiomatized by Conitzer et al. [AAAI 2016]; they also provided several algorithms for computing the outcomes of this rule. In this paper, we present a significantly improved algorithm and evaluate it experimentally. Our algorithm is based on a tight connection to minimum-cost flow that we exhibit. We also show that our algorithm cannot be improved without breakthroughs on min-cost flow.

1 Introduction

In social choice, we take as input the preferences or opinions of multiple agents and seek to aggregate them. This is a key problem in multiagent systems, which has given rise to the *computational* social choice research community (Brandt et al. 2015). The paradigmatic setting in social choice is that of voting, where each agent *ranks* a set of alternatives. However, a ranking of alternatives is not always a natural representation of the preferences or opinions that must be aggregated. For example, in a *multi-issue* setting (Lang and Xia 2015), there are exponentially many alternatives, making it infeasible to rank them all. In *judgment aggregation* (in its standard form) (Endriss 2015), agents provide true-or-false judgments on various propositions, and we must aggregate these into judgments that are logically consistent. AI researchers regularly deal with questions of how to represent the world, so it is not surprising that these problems are receiving special attention from the AI/MAS community. And indeed, entirely new problems in social choice are arising from work done in this community.

One such problem is the *societal tradeoffs problem* (Conitzer, Brill, and Freeman 2015). In it, each agent quantitatively compares pairs of activities. For example, one agent may judge that using one liter of gasoline is as bad as generating 2 bags of landfill trash; another may judge that that number should be 3 instead of 2. One possible area of application for this framework is in automated moral decision making, where an AI system has to choose a course of action based on data that it has about human judgments in similar scenarios (see, e.g., Conitzer et al. (2017)). This problem has already been considered in the specific contexts of self-driving vehicles making life-or-death decisions (Noothigattu et al. 2018) and kidney exchange algorithms prioritizing certain patients over others (Freedman et al. 2018). To fit this into the framework considered in this paper, it may be that one human subject appears to value (say) the life of a 20-year old at 1.5 times the life of a 50-year old, and another appears to use a ratio of 1.1. We then need to find an aggregate ratio.[1]

When only two activities are being compared, using the median judgment as our aggregate satisfies many nice properties. Things become more complex when more than two activities are compared. As observed by Conitzer, Brill, and Freeman (2015), taking the median on each pair separately can result in *inconsistent* tradeoffs. For example, the resulting aggregate tradeoffs may be that (one unit of) gasoline is 2 times as bad as trash, trash is 4 times as bad as deforestation, and gasoline is 2 times as bad as deforestation; but $2 \cdot 4 = 8 \neq 2$ (Figure 1).

Conitzer et al. [2016] take consistency as a hard constraint and specify some axioms that a rule should satisfy. Somewhat unusually[2] for a social choice context, instead of leading to impossibility, the axioms uniquely pin down a single rule. This rule involves first taking the logarithm of each tradeoff value. Hence, the consistency constraint changes from a multiplicative one ($a \cdot b = c$) to an additive one ($\log a + \log b = \log c$), and the problem can be thought of as follows. Each activity is to be assigned a number (its *quality*), and the agents express for pairs of activities what they consider to be the ideal *difference* in quality between these two activities. Then, aggregate qualities are chosen so that the sum of the distances between the aggregate differences and the ideal differences expressed by the voters is minimized.

Conitzer et al. [2016] show that this problem can be for-

[1] In these contexts, the word "activities" is not very appropriate to refer to the different types of people, but we will stick with this word for the sake of consistency with prior literature.

[2] Of course, there are other results where a rule is uniquely pinned down by axioms—for example, the Kemeny-Young rule was uniquely axiomatized by Young and Levenglick (1978).

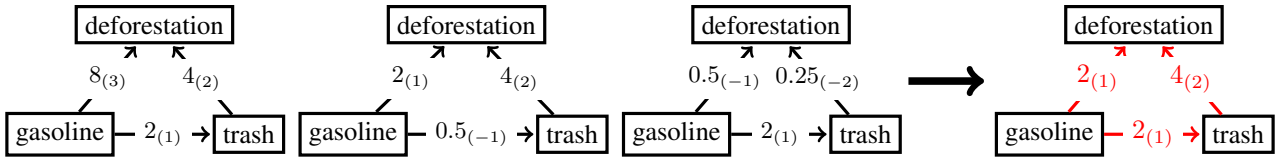

Figure 1: An example where taking pairwise medians yields an inconsistent outcome. The three figures on the left are the votes, and the right figure shows the (inconsistent) aggregate outcome. The numbers on edges show the multiplicative tradeoff factors perceived by different agents, and the numbers in parentheses are obtained by taking the logarithm with base 2, representing the additive tradeoff values—to be interpreted as ideal differences in quality. In the remainder of the paper we will focus only on the numbers obtained after taking the logarithm, which are easier to work with; these can then be exponentiated to return to the original interpretation.

mulated as a linear program, and also give a hill-climbing heuristic that does not necessarily find an optimal solution. In this paper, we show that the computational problem is in fact closely related to minimum-cost circulation. Based on this insight, we provide a significantly improved (and, barring breakthroughs on min-cost circulation, best possible) algorithm for the problem and evaluate it experimentally.

2 Preliminaries

We first present the societal tradeoffs problem, which is the problem of determining the aggregate tradeoffs according to the rule defined by Conitzer et al. [2016]. Without loss of generality and for presentational simplicity, we focus on the additive model (i.e., we assume that logarithms have already been taken). Also, we assume that each agent (voter) only votes on one pair of activities; this is also w.l.o.g., because a voter voting on multiple pairs can equivalently be split up into multiple voters voting on one edge each.

Definition 1 (Societal Tradeoffs)**.** Given a set V of n activities and a set E of m voters, each voter submits a vote of the form (a_i, b_i, c_i, w_i). Voter i believes the difference between activities $a_i, b_i \in V$ should be $c_i \geq 0$, and each voter is associated with a weight $w_i \geq 0$. The Societal Tradeoffs Problem asks us to compute potentials $x \in \mathbb{R}^V$, such that the total loss $L(V, E, x)$ is minimized:

$$L(V, E, x) = \sum_{(a_i, b_i, c_i, w_i) \in E} w_i |x_{a_i} - x_{b_i} - c_i|.$$

Throughout this paper, we will use n for the number of vertices (activities) and m for the number of edges (votes).

3 The Algorithm

We identify close connections between Societal Tradeoffs and classic combinatorial problems. Based on these, we give an algorithm for Societal Tradeoffs that is best possible given what is currently known about these classic problems.

3.1 Formulating the Problem as a Linear Program

Conitzer et al. (2016) showed how to formulate Societal Tradeoffs as a linear program (LP). In this subsection, we present an equivalent but slightly different linear program that makes it easier to take the dual.

Recall that the problem asks to set x_v for every $v \in V$ to minimize $L(V, E, x)$. For any positive integer t, we denote the set $\{1, 2, \ldots, t\}$ by $[t]$. We can then formulate the problem as the following LP, where ℓ_i^+ being positive means that we deviate from the value c_i in one direction, and ℓ_i^- being positive means we deviate in the other direction.

Linear Program 1.

$$\begin{array}{lll} \textit{minimize} & \sum_{i \in [m]} w_i (\ell_i^+ + \ell_i^-) & \\ \textit{subject to} & \ell_i^+ \geq x_{a_i} - x_{b_i} - c_i & \forall i \in [m] \\ & \ell_i^- \geq c_i + x_{b_i} - x_{a_i} & \forall i \in [m] \\ & \ell_i^+ \geq 0,\ \ell_i^- \geq 0 & \forall i \in [m] \end{array}$$

While we can solve this LP directly in polynomial time, for larger instances with thousands of activities, we want an algorithm faster than generic LP solvers.

3.2 Equivalence to Max-Cost Circulation

Consider the dual of Linear Program 1:

Linear Program 2.

$$\begin{array}{lll} \textit{maximize} & \sum_{i \in [m]} c_i (f_i^- - f_i^+) & \\ \textit{subject to} & 0 \leq f_i^+ \leq w_i,\ 0 \leq f_i^- \leq w_i & \forall i \in [m] \\ & \sum_{i:a_i=v} (f_i^- - f_i^+) = \sum_{i:b_i=v} (f_i^- - f_i^+) & \forall v \in V \end{array}$$

Observe that for any i, f_i^+ and f_i^- only appear in the form $f_i^- - f_i^+$. Let $f_i = f_i^- - f_i^+$. We obtain the following LP:

Linear Program 3.

$$\begin{array}{lll} \textit{maximize} & \sum_{i \in [m]} c_i f_i & \\ \textit{subject to} & \sum_{i:a_i=v} f_i = \sum_{i:b_i=v} f_i & \forall v \in V \\ & -w_i \leq f_i \leq w_i & \forall i \in [m] \end{array}$$

This LP has the following combinatorial interpretation. Consider each activity to be a vertex in a directed graph, and each vote (a_i, b_i, c_i, w_i) to be an edge from a_i to b_i with cost c_i and capacity w_i. We can interpret the f_i variables as flows, so that the objective of the LP corresponds to the total cost of the flows, and the two linear constraints reflect the conservation and capacity constraints of circulations. (A *circulation* is a flow such that the flow into each vertex is equal to the flow out of it.) Societal Tradeoffs is therefore closely related (via linear programming duality) to the Undirected Max-Cost Circulation problem. See Figure 2 for a visualization of the duality.

Definition 2 (Undirected Max-Cost Circulation)**.** Given vertices $V = [n]$ and edges $E = \{(a_i, b_i, c_i, w_i)\}_{i=1}^m$, the Undirected Max-Cost Circulation problem seeks a flow $f \in \mathbb{R}^m$ that satisfies

(1) Flow conservation: $\sum_{i:a_i=v} f_i = \sum_{i:b_i=v} f_i, \forall v$, and

(2) Capacity constraints: for any $i \in [m]$, $|f_i| \le w_i$,

such that the total cost of the flow, $\sum_{i\in[m]} c_i f_i$, is maximized.

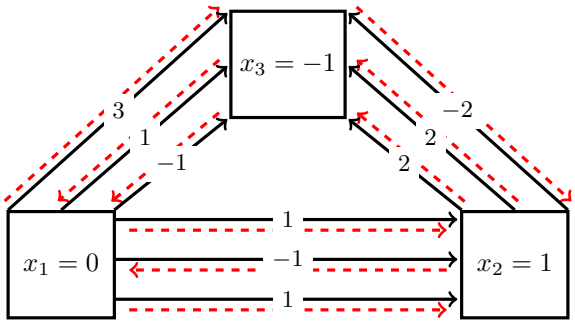

Figure 2: Illustration of the primal and dual LPs based on activities and votes in Figure 1. All votes/edges have unit weight/capacity. The cost of the edges (the c_i) are the values submitted by the three voters in Figure 1 (after taking $\log_2$). Optimal potentials (the x_v) are associated with the activities; the differences between these values necessarily constitute a solution that is consistent, unlike the one at the right of Figure 1. Specifically, the bottom-edge comparison between gasoline and trash becomes -1 instead of the 1 in parentheses at the right of Figure 1 (and the other aggregate differences are the same as in that figure). The flows on the edges from the dual LP (the f_i) are always 1 and their directions are indicated by the red/dashed arrows; note that the flows form a circulation. These flows are obtained by solving the dual LP. The total cost of these flows is 12, which is equal to the loss of the primal solution, which proves both to be optimal by weak LP duality.

Strong duality immediately gives the following theorem.

Theorem 1. *Fix $V = [n]$, $E = \{(a_i, b_i, c_i, w_i)\}_{i=1}^m$. The following problems have the same optimal objective value:*

(1) Societal Tradeoffs with activities V and votes E.

(2) Max-Cost Circulation with vertices V and edges E.

3.3 Translation between Primal/Dual Solutions

Theorem 1 implies that, in order to find the *value* of an optimal solution to the Societal Tradeoffs problem, it suffices to solve Max-Cost Circulation. Nevertheless, we are generally not so much interested in this value, but rather in the solution itself. While some LP algorithms solve for primal and dual solutions simultaneously, our goal is to avoid the use of general LP algorithms. We next show that there are more efficient ways to translate optimal solutions between Societal Tradeoffs (primal) and Max-Cost Circulation (dual).

Algorithm 1: Translation from an Undirected Max-Cost Circulation optimum to a Societal Tradeoffs optimum.

Input : A circulation $\{f_i\}$ that maximizes the total cost on graph (V, E).
Output: Potentials x that minimize the loss of Societal Tradeoffs with activities V and votes E.
Let $S = (V, \emptyset)$ be an empty system of difference constraints with variables $x \in \mathbb{R}^V$.
for $i \in [m]$ **do**
  if $f_i > -w_i$ **then**
    Add constraint $x_{a_i} - x_{b_i} \le c_i$ into S.
  if $f_i < w_i$ **then**
    Add constraint $x_{a_i} - x_{b_i} \ge c_i$ into S.
Solve S by running Single-Source Shortest Path from an arbitrary vertex in V, and return the distance labels x.

From dual optima to primal optima. Algorithm 1 computes an optimal primal solution by solving a system of difference constraints. The difference constraints come from the complementary slackness conditions of LP 2—for dual constraints that are not tight (which correspond to unsaturated edges), we require the corresponding primal variables (i.e., ℓ_i^+ or ℓ_i^- for edge i) to be 0.

Theorem 2. *Given an optimal solution to Undirected Max-Cost Circulation, Algorithm 1 runs in time*

$$O(m + n + T_{\text{SSSP}}(n, m, W)),$$

and returns an optimal solution of the corresponding Societal Tradeoffs instance. $T_{\text{SSSP}}(n, m, W)$ is the time required to solve Single-Source Shortest Path with negative weights (SSSP) on a graph with n vertices, m edges, and maximum absolute distance W.

Proof. The time complexity follows from Algorithm 1 and fact that systems of difference constraints can be efficiently solved using SSSP oracles (Pratt 1977; Aspvall and Shiloach 1979). We focus on the correctness of the algorithm.

First observe that S in Algorithm 1 is feasible, because no negative cycle exists in the graph representation of S. Otherwise, flowing backward along the cycle increases the total cost of the circulation, contradicting the optimality of f.

Next we show that the loss incurred by x is exactly the cost of f. The optimality of x then follows from weak duality. For $i \in [m]$ where $|f_i| < w_i$, the corresponding loss $w_i|x_{a_i} - x_{b_i} - c_i|$ in the primal LP is zero, because Algorithm 1 added both constraints $x_{a_i} - x_{b_i} \le c_i$ and $x_{a_i} - x_{b_i} \ge c_i$. Therefore, if we consider votes $E' = \{(a_i, b_i, c_i, w_i')\}$ where $w_i' = |f_i|$, the total loss of x on E and E' are the same.

We then decompose f into sum of flows around directed cycles. For a cycle $C = (u_1^C, \ldots, u_{l(C)}^C)$ of length $l(C)$ with flow $f^C > 0$, let d_i^C be the cost of edge (u_i^C, u_{i+1}^C) where $u_{l(C)+1}^C = u_1^C$. The cost of f_C units of flow along C is

Algorithm 2: Translation from a Societal Tradeoffs optimum to an Undirected Max-Cost Circulation optimum.

Input : Potentials x that minimize the ℓ_1-loss of Societal Tradeoffs with activities V and votes E.

Output: A circulation $\{f_i\}$ that maximizes the total cost on graph (V, E).

Let $f_i \leftarrow 0$ for all $i \in [m]$. **for** $i \in [m]$ **do**
  if $x_{a_i} - x_{b_i} - c_i < 0$ **then**
    $f_i \leftarrow w_i$.
  if $x_{a_i} - x_{b_i} - c_i > 0$ **then**
    $f_i \leftarrow -w_i$.

Let $V_0 \leftarrow V \cup \{s, t\}$, $(E_0, E_s, E_t) \leftarrow (\emptyset, \emptyset, \emptyset)$.
for $v \in V$ **do**
  $d_v = \sum_{i:b_i=v} f_i - \sum_{i:a_i=v} f_i$.
  if $d_v > 0$ **then**
    $E_s \leftarrow E_s \cup \{(s, v, d_v)\}$
  if $d_v < 0$ **then**
    $E_t \leftarrow E_t \cup \{(v, t, -d_v)\}$
for $i \in [m]$ **do**
  if $x_{a_i} - x_{b_i} - c_i = 0$ **then**
    $E_0 \leftarrow E_0 \cup \{(a_i, b_i, w_i), (b_i, a_i, w_i)\}$.

Compute a maximum s-t flow f' on $G_0 = (V_0, E_0 \cup E_s \cup E_t)$.
Return $f + f'|_{E_0}$.

$f^C \sum_i d_i^C$. Note that we consider the edges in the direction of the flow, and we always have

$$x_{u_i^C} - x_{u_{i+1}^C} - d_i^C = \pm(x_a - x_b - c).$$

Moreover, the fact that $f^C > 0$ means there are constraints of form $x_{u_i^C} - x_{u_{i+1}^C} \geq d_i^C$ in S. Therefore,

$$\begin{aligned}
\text{cost}(f) &= \sum_C f^C \sum_{i \in [l(C)]} d_i^C \\
&= \sum_C f^C \sum_{i \in [l(C)]} \left(d_i^C - x_{u_i^C} + x_{u_{i+1}^C} \right) \\
&= \sum_C f^C \sum_i \left| x_{u_i^C} - x_{u_{i+1}^C} - d_i^C \right| \\
&= \sum_{e=(a,b,c,w') \in E'} \sum_{C: e \in C} f^C |x_a - x_b - c| \\
&= \sum_{(a,b,c,w') \in E'} w' |x_a - x_b - c| \\
&= L(V, E', x) = L(V, E, x). \qquad \square
\end{aligned}$$

From primal optima to dual optima. For completeness, we show that we can also efficiently translate primal (Societal Tradeoffs) optima to dual (Max-Cost Circulation) optima. Algorithm 2 first constructs a partial flow based on votes/edges with non-zero primal costs, and then tries to balance the surplus at each activity/vertex using only votes/edges with 0 primal costs.

Theorem 3. *Given any optimum of a Societal Tradeoffs instance, Algorithm 2 runs in*

$$O(m + n + T_{\text{MaxFlow}}(n, m, U))$$

time and returns an optimum of the corresponding Undirected Max-Cost Circulation instance. $T_{\text{MaxFlow}}(n, m, U)$ *is the time complexity of Undirected s-t Max-Flow on a graph with n vertices, m edges, and maximal capacity U.*

Proof. The time complexity follows from the description of Algorithm 2. We focus on the correctness of the algorithm.

We first show that Algorithm 2 does output a circulation. In the first step, we assign a positive saturating flow $f_i = w_i$ to an edge (a_i, b_i, c_i, w_i) if $x_{a_i} - x_{b_i} - c_i < 0$, and a negative saturating flow $f_i = -w_i$ if $x_{a_i} - x_{b_i} - c_i > 0$. The surplus of each vertex v is

$$d_v = \sum_{i:b_i=v} f_i - \sum_{i:a_i=v} f_i.$$

The algorithm then tries to route the remaining surpluses on the edges

$$E_0 = \{(a_i, b_i, c_i, w_i) \mid x_{a_i} - x_{b_i} - c_i = 0\},$$

which have not been assigned any flow. This attempt succeeds if the max flow f' on graph G_0 saturates every edge from s (and every edge to t). We now show this always happens when x is optimal.

Suppose the flow computed is not saturating. That is, there exists a cut C whose size is smaller than $\sum_{v \in V: d_v > 0} d_v$. Let $S, T \subset V$ be the vertices on the s-side and t-side of C. Let $\Delta = E_0 \cap C$ be the edges/votes between S and T with 0 primal cost. The cut C includes edges in Δ, as well as edges from S to t and s to T. From the assumption on the size of C, we know that

$$\sum_{(a_i,b_i,c_i,w_i) \in \Delta} w_i - \sum_{v \in S: d_v < 0} d_v + \sum_{v \in T: d_v > 0} d_v < \sum_{v \in V: d_v > 0} d_v.$$

Rearranging the terms gives

$$D := \sum_{v \in S} d_v - \sum_{(a_i,b_i,c_i,w_i) \in \Delta} w_i > 0.$$

Note that by the construction of f in the beginning,

$$\begin{aligned}
d_v = &\sum_{i:a_i=v, x_{a_i} - x_{b_i} - c_i > 0} w_i - \sum_{i:a_i=v, x_{a_i} - x_{b_i} - c_i < 0} w_i \\
&+ \sum_{i:b_i=v, x_{a_i} - x_{b_i} - c_i < 0} w_i - \sum_{i:b_i=v, x_{a_i} - x_{b_i} - c_i > 0} w_i.
\end{aligned}$$

We will show D is exactly the rate of improvement if we decrease x_v for all $v \in S$ simultaneously. Hence, $D > 0$ contradicts the optimality of x so f' must saturate all edges from s. To see why this is true, observe that the rate at which the loss changes consists of two parts: the change on nonzero edges across C and the change on zero edges across C. Decreasing x_v slightly for $v \in S$ may decrease the loss on

some nonzero edges, and increase the loss on other nonzero edges. The total rate of this part of the change is exactly $-\sum_{v\in S} d_v$. On the other hand, changing x_v in any way increases the loss on zero edges across C, and the total rate of this part is simply the total weight of zero edges across C, namely $\sum_{(a_i,b_i,c_i,w_i)\in\Delta} w_i$. The sum of the two parts is exactly D.

Finally, we show that the cost of the constructed circulation is equal to the loss in the Societal Tradeoffs instance. For any unsaturated edge $i \in E$ (i.e. $|f_i + f'_i| < w_i$), the corresponding primal cost is 0 (i.e., $x_{a_i} - x_{b_i} - c_i = 0$). Hence, setting $w'_i = |f_i + f'_i|$ for these edges does not affect the primal or the dual cost. Let E' be the edges after this transformation. A similar cycle-decomposition argument to that in the proof of Theorem 2 yields

$$\text{cost}(f) = L(V, E', x) = L(V, E, x).$$

The optimality follows from weak duality. □

3.4 Implications on the Time Complexity of Societal Tradeoffs

A solid connection between Societal Tradeoffs and classic combinatorial problems has been established in the foregoing sections. Based on this connection, we give a significantly improved algorithm for Societal Tradeoffs, and prove that improving our algorithm would give a faster algorithm for Min-Cost Circulation.

First, we review the state of the art for Min-Cost Circulation and Single-Source Shortest Path (SSSP) algorithms. Maximum Flow, Shortest Path, and other related graph problems are core combinatorial optimization problems that have been studied extensively (see, e.g., (Edmonds and Karp 1972; Gabow and Tarjan 1989; Madry 2013; Cohen et al. 2017) and the references therein).

In this section, we will use n for the number of vertices (activities), m for the number of edges (votes), U for the maximum capacity, and W for the maximum absolute value of the cost (for Min-Cost Circulation) or distance (for SSSP).

Lemma 1 ((Gabow and Tarjan 1989))**.** *There is an algorithm for Min-Cost Circulation and SSSP which runs in time* $O(m^{1.5}\log(nW))$.

Lemma 2 ((Cohen et al. 2017))**.** *For unit-capacity graphs, there is an algorithm for Min-Cost Circulation and SSSP which runs in time* $\widetilde{O}(m^{10/7}\log W)$.[3]

We can now state our main results. Theorem 4 follows immediately from Theorem 2 and Lemmas 1 and 2.

Theorem 4 (Better Algorithms for Societal Tradeoffs)**.** *Societal Tradeoffs can be solved in time* $O(m^{1.5}\log(nW))$*, where* W *is the maximum difference suggested by the voters. If all voters have the same weight, then Societal Tradeoffs can be solved in time* $\widetilde{O}(m^{10/7}\log W)$.

We further show (in Theorem 5) that the above algorithm cannot be significantly improved without giving faster Min-Cost Circulation algorithms. More specifically, any algorithm whose dependency on m has a smaller exponent is considered significantly faster. We first show that directed and undirected Min-Cost Circulations are essentially equivalent.

Lemma 3 (Folklore)**.** *Directed Min-Cost Circulation with* n *vertices,* m *edges and maximum absolute cost* W *can be reduced to Undirected Min-Cost Circulation with parameters* $(n+1, 3m, nW)$*. The reduction takes* $O(m+n)$ *time.*

Proof. We construct an Undirected Min-Cost instance $G' = (V \cup \{v_0\}, E')$ which preserves the solution to any Directed Min-Cost Circulation instance $G = (V, E)$. Without loss of generality, we assume each capacity w_i is an even integer. Otherwise, we first multiply all capacities by 2, which does not change the nature of the instance. For each (directed) edge $(a_i, b_i, c_i, w_i) \in E$, we add 3 (undirected) edges into E':

- $(a_i, v_0, -nW, w_i/2)$,
- $(v_0, b_i, -nW, w_i/2)$, and
- $(a_i, b_i, c_i, w_i/2)$.

Note that in any minimum cost circulation of G', all edges with cost $-nW$ must be saturated. In other words, the optimum is a combination of two parts:

(1) a circulation consisting of $w_i/2$ units of flow along directed cycles (b_i, a_i, v_0), one for each edge $(a_i, b_i, c_i, w_i) \in E$; and

(2) a Min-Cost Circulation on the residual graph. Observe that the residual graph is exactly G.

Thus, a Min-Cost Circulation of G can be recovered from a Min-Cost Circulation of G', by subtracting the first part of the flow. This can be done in $O(m+n)$ time. □

Theorem 5 is a direct corollary of Theorem 1 and Lemma 3.

Theorem 5 (Improvement Gives Faster Flow Algorithms)**.** *Let* U *denote the maximum weight of any voter (and maximum capacity), and let* W *denote the maximum difference between activities (and maximum absolute cost).*

Let T_{MCC} *denote the time it takes to compute a circulation that minimizes the total cost.*

$$\begin{aligned} &T_{\text{MCC}}(n, m, U, W) \\ \le\ &T_{\text{SocietalTradeoffs}}(n+1, 3m, U, nW) \\ +\ &T_{\text{MaxFlow}}(n+1, 3m, U) + O(m+n). \end{aligned}$$

In other words, if one can solve Societal Tradeoffs significantly faster, then either (1) there is a faster algorithm that computes a Min-Cost Circulation, or (2) Min-Cost Circulation is no harder than Max-Flow. Either result would be a breakthrough on the computation of min-cost flow.

[3] We use $\widetilde{O}(f(n))$ as a shorthand for $O(f(n)\log^{O(1)} f(n))$.

4 Limitations of Hill-climbing

In this section, we provide some insights to the hill-climbing algorithm proposed in Conitzer et al. (2016). The hill-climbing heuristic works by picking one activity at a time and setting its potential to a value that minimizes the loss given other potentials. It repeats this until no further such local improvements are possible. Conitzer et al. (2016) observed that hill-climbing can get stuck at local optima. We give a simple example where hill-climbing gets stuck. In addition, we show that even when the algorithm converges to optimality, sometimes this can take a very long time.

Hill-climbing can get stuck in a local optimum. Consider an example with 6 activities, and one complete vote (or equivalently, a vote for each pair of activities with unit weight). For $i \in \{1, 2, 3\}$, the difference between x_i and x_{i+3} should be 1, and the difference between any other pair should be 0. The optimal solution is to assign every activity the same value, say 0, with total loss 3.

Consider an initial value configuration $(1, 1, 1, 0, 0, 0)$. This is a local optimum for hill-climbing with loss 6, since changing the value of any single vertex cannot reduce the total loss. To see this, consider activity 1. At this local optimum, its differences to activities x_2, x_3, and x_4 align with the vote, so 3 out of the 5 remaining activities prefer x_1 to stay the same. Similar arguments hold for other activities.

Hill-climbing can take arbitrarily long to terminate. Consider the same example except that the voter now thinks the value of activity 1 should be ε higher than that of activity 2. Again, we start from $(1, 1, 1, 0, 0, 0)$. First we move x_2 to $1 - \varepsilon$, because this improves the comparison with x_1, x_4, and x_6 (but moving it any further would hurt the comparison with x_1). Then, we move x_3 to $1 - \varepsilon$ because this improves the comparison to x_2, x_4, and x_5. After that, we move x_1 to $1 - \varepsilon$ to improve the comparison to x_3, x_5, and x_6. Now we are back to a similar configuration to where we started, i.e., $(1 - \varepsilon, 1 - \varepsilon, 1 - \varepsilon, 0, 0, 0)$. Following this pattern, it takes $\Omega(1/\varepsilon)$ time for hill-climbing to terminate. For arbitrarily small ε, this time can be arbitrarily large.

5 Experiments

All experiments were done on a laptop computer with 8GB of memory and a 2.6 GHz Intel Core i5 CPU. Results are obtained by averaging over 10 runs with different seeds.

5.1 Experimental Setup

For empirical evaluation, we implemented our algorithm based on the network simplex algorithm from LEMON (an open-source library of graph algorithms). We evaluate our flow-based algorithm against

1. an LP solver based on the GNU Linear Programming Kit (GLPK),
2. an LP solver based on CPLEX, which is generally considered one of the best general-purpose LP solvers, and
3. a hill-climbing heuristic.

(1) and (3) were studied experimentally by Conitzer et al. (2016). We generate input using 4 different distributions:

1. Uniform. For each voter and each pair of activities (u, v), we draw a number $x \in [-1, 1]$ uniformly at random, and let the voter's tradeoff between u and v be x.
2. Spanning. For each voter, we sample a random spanning tree of the activities. For each edge (u, v) of the spanning tree, we draw the voter's tradeoff x uniformly at random from $[-1, 1]$. We then fill in the voter's tradeoff between other pairs of activities by consistency.
3. Noise. We first draw a potential for each activity p_v from $[-10, 10]$ uniformly at random. For each voter i and each pair of activities (u, v), let the voter's tradeoff between u and v be $p_u - p_v + x^i_{uv}$, where $x^i_{uv} \in [-1, 1]$ is drawn independently uniformly at random.
4. Random-graph-uniform. For each voter, we draw exactly one pair of activities (u, v) and a number x from $[-1, 1]$ uniformly at random. We then let the voter's tradeoff between u and v be x.

The first three distributions generate instances where each voter expresses a preference on every pair of activities (i.e., the voting graph is a clique). The spanning tree distribution generates consistent votes; the others generally do not.[4]

5.2 Results and Evaluation

As can be seen from Figure 3, GLPK is consistently slow, particularly on instances from the spanning distribution. CPLEX is faster than GLPK, but still significantly slower than the other two methods.

Hill-climbing works reasonably well as a heuristic algorithm. As Table 1 shows, its accuracy improves as the voting graph becomes denser. In particular, the solutions hill-climbing generates are usually indistinguishable from the optimal solution when voters give complete votes (i.e., tradeoffs between all pairs of activities).

In terms of runtime, hill-climbing beats our flow-based algorithm on the uniform and spanning distributions. However, its runtime is worse under the noise model, when there is an underlying ground truth. During our experiments, we sometimes observe instances on which hill-climbing takes significantly more time than the flow-based algorithm. In contrast, the flow-based algorithm shows remarkable robustness regardless of the distribution of the input.

In general, our flow-based algorithm is robustly fast with a strong upper bound on running time, and is guaranteed to produce an optimal solution. In contrast, the formerly studied algorithms have notable flaws: LP-based methods (i.e., GLPK and CPLEX) produce optimal solutions but almost never terminate on our large instances, and hill-climbing, at the cost of producing suboptimal solutions, is not too much faster and its runtime is extremely sensitive to the structure of the input. These factors make our flow-based algorithm preferable both in theory and in practice.[5]

[4] We consider inconsistent votes here in order be consistent with prior experimental practice on this topic (Conitzer et al. 2016).

[5] It is worth noting that when computing the outcomes of a rule

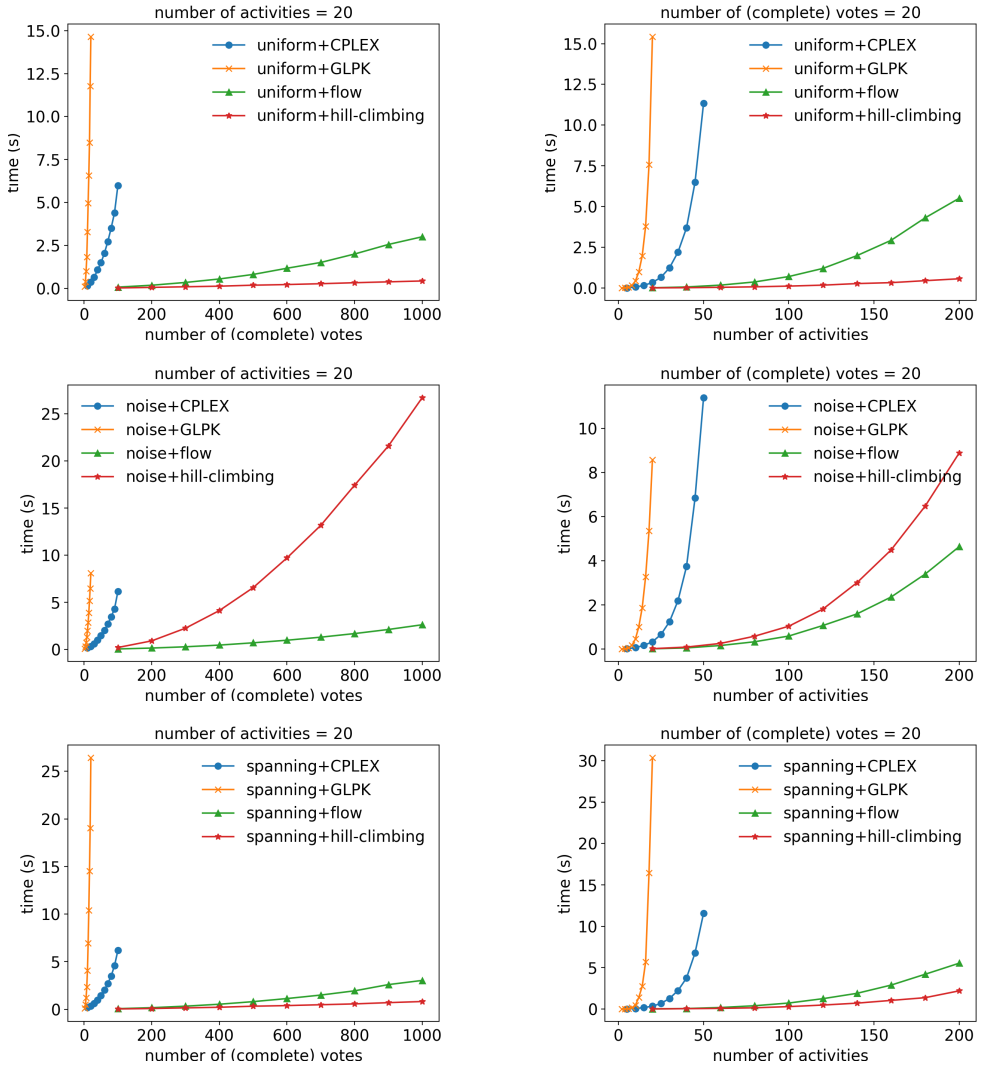

Figure 3: Runtime of the 3 algorithms on inputs from different distributions. The left 3 figures plot the growth of running times when the number of votes grows, and the right 3 figures show the running times when the number of activities grows.

Number of edges	100	200	300	400	500
Approximation ratio	2.04	1.20	1.06	1.04	1.02

Table 1: Approximation ratio of hill-climbing vs. density of the voting graph (random-graph-uniform distribution with 100 nodes).

6 Discussion

Our algorithm for societal tradeoffs scales much better than the previously known linear programming approach. Given the tight connection to min-cost circulation that we have exhibited, it appears that there is little left to be done to improve this algorithm as far as exact approaches go. Even when comparing to the previously known heuristic hill-climbing approach, our algorithm is sometimes significantly faster in experiments—and of course it comes with the benefits of exactness and provable running time guarantees.

The faster algorithm will allow us to scale to much larger sets of activities. In some contexts, this will be critical for the methodology to succeed. For example, in the autonomous vehicles and kidney exchanges examples discussed in the introduction, where "activities" corresponds to different types of people, we face a combinatorial explosion in V if many attributes are taken into account (e.g., in Freedman et al. [2018], one possible type is a young person who drinks little alcohol but has skin cancer in remission). Of course even our improved algorithm can only go so far in addressing such combinatorial explosions; at some point, we will no longer be able even to enumerate V and a different

in a social choice setting, computing only an approximate solution is likely to lose the desirable properties of the rule, and it may raise concerns about whether the outcome is legitimate.

representation scheme will be needed. What scheme is appropriate, do we need a different rule for it, and are there efficient algorithms for that rule? These are exciting questions for future research.

7 Acknowledgements

We are thankful for support from NSF under awards IIS-1814056 and IIS-1527434. Yu Cheng is also supported in part by NSF grants CCF-1527084, CCF-1535972, CCF-1637397, CCF-1704656, IIS-1447554, and NSF CAREER Award CCF-1750140. We also thank anonymous reviewers for helpful comments.

References

Aspvall, B., and Shiloach, Y. 1979. A polynomial time algorithm for solving systems of linear inequalities with two variables per inequality. In *Foundations of Computer Science, 1979., 20th Annual Symposium on*, 205–217. IEEE.

Brandt, F.; Conitzer, V.; Endriss, U.; Lang, J.; and Procaccia, A. D. 2015. *Handbook of Computational Social Choice*. Cambridge University Press.

Cohen, M. B.; Madry, A.; Sankowski, P.; and Vladu, A. 2017. Negative-weight shortest paths and unit capacity minimum cost flow in õ (m 10/7 log w) time. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, 752–771. Society for Industrial and Applied Mathematics.

Conitzer, V.; Freeman, R.; Brill, M.; and Li, Y. 2016. Rules for choosing societal tradeoffs. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Conitzer, V.; Sinnott-Armstrong, W.; Borg, J. S.; Deng, Y.; and Kramer, M. 2017. Moral decision making frameworks for artificial intelligence. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, 4831–4835. Blue Sky track.

Conitzer, V.; Brill, M.; and Freeman, R. 2015. Crowdsourcing societal tradeoffs. In *Proceedings of the Fourteenth International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, 1213–1217. Blue Sky Ideas track.

Edmonds, J., and Karp, R. M. 1972. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM (JACM)* 19(2):248–264.

Endriss, U. 2015. Judgment aggregation. In Brandt, F.; Conitzer, V.; Endriss, U.; Lang, J.; and Procaccia, A. D., eds., *Handbook of Computational Social Choice*. Cambridge University Press. chapter 17.

Freedman, R.; Borg, J. S.; Sinnott-Armstrong, W.; Dickerson, J. P.; and Conitzer, V. 2018. Adapting a kidney exchange algorithm to align with human values. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*.

Gabow, H. N., and Tarjan, R. E. 1989. Faster scaling algorithms for network problems. *SIAM Journal on Computing* 18(5):1013–1036.

Lang, J., and Xia, L. 2015. Voting in combinatorial domains. In Brandt, F.; Conitzer, V.; Endriss, U.; Lang, J.; and Procaccia, A. D., eds., *Handbook of Computational Social Choice*. Cambridge University Press. chapter 9.

Madry, A. 2013. Navigating central path with electrical flows: From flows to matchings, and back. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, 253–262. IEEE.

Noothigattu, R.; Gaikwad, S. N. S.; Awad, E.; D'Souza, S.; Rahwan, I.; Ravikumar, P.; and Procaccia, A. D. 2018. A voting-based system for ethical decision making. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*.

Pratt, V. 1977. Two easy theories whose combination is hard. Technical report, Technical report, Massachusetts Institute of Technology.

Young, H. P., and Levenglick, A. 1978. A consistent extension of Condorcet's election principle. *SIAM Journal of Applied Mathematics* 35(2):285–300.