IJCAI-19 Paper #2613 (Supplemental Material)
Election Control with Voters' Uncertainty: Hardness and Approximation Results

1 Omitted proofs

In what follows we give full proofs of some theorems presented in the paper, where some technical details are omitted due to space constraints.

Theorem 1. *An α-approximation to the election control problem in PLTR gives an $\alpha\beta$-approximation to the Densest k-Subgraph problem, for a positive constant $\beta < 1$.*

Proof. Given an undirected graph $G = (V, E)$ and an integer k, Densest k-Subgraph (DkS) is the problem of finding the subgraph induced by a subset of V of size k with the highest number of edges given that k is fixed.

The reduction works as follows: Consider the PLTR problem on G, where each undirected edge $\{u, v\}$ is replaced with two directed edges (u, v) and (v, u). Let us consider m candidates and let us assume that all nodes initially have null probability of voting for all the candidates but one, different from $c_\star$, that we denote as $\hat{c}$. Formally we have that, $\pi_v(\hat{c}) = 1$ and $\pi_v(c_i) = \pi_v(c_\star) = 0$ for each $c_i \neq \hat{c}$ and for each $v \in V$. Assign to each edge $(u, v) \in E$ a weight $b_{uv} = \frac{1}{n^\gamma}$, for any fixed constant $\gamma \geqslant 4$ and $n = |V|$.

We show the reduction considering the problem of maximizing the score, because in the instance considered in the reduction the MoV is exactly equal to twice the score. In fact, the score of $\hat{c}$ after PLTR starting from any initial set S is

$$
\begin{aligned}
F(\hat{c}, S) &= \sum_{v\in V} \tilde{\pi}_v(\hat{c}) = \sum_{v\in V\setminus A} \pi_v(\hat{c}) + \sum_{v\in A} \tilde{\pi}_v(\hat{c}) \\
&= |V| - |A| + \sum_{v\in A} \frac{1}{1 + \sum_{u\in A\cap N_v^i} \frac{1}{n^\gamma}} \\
&= |V| - \sum_{v\in A} \left(1 - \frac{1}{1 + \sum_{u\in A\cap N_v^i} \frac{1}{n^\gamma}}\right) \\
&= |V| - \sum_{v\in A} \left(\frac{\sum_{u\in A\cap N_v^i} \frac{1}{n^\gamma}}{1 + \sum_{u\in A\cap N_v^i} \frac{1}{n^\gamma}}\right) = |V| - F(c_\star, S),
\end{aligned}
$$

because $(\sum_{u\in A\cap N_v^i} \frac{1}{n^\gamma})/(1 + \sum_{u\in A\cap N_v^i} \frac{1}{n^\gamma}) = \tilde{\pi}_v(c_\star)$ and $\pi_v(c_\star) = 0$ for each $v \in V$. Thus, according to the definition of MoV in Equation (6), we have that

$$\mathrm{MoV}(S) = |V| - (|V| - F(c_\star, S) - F(c_\star, S)) = 2F(c_\star, S).$$

To compute the expected final score of the target candidate we average its score in all live live-edge graph in $\mathcal{G}$, according to Formula (3). In our reduction, the empty live-edge graph $G'_\emptyset = (V, \emptyset)$ is sampled *with high probability*, i.e., with probability at least $1 - \frac{1}{n^{\gamma-2}}$:

$$\mathbf{P}\left(G'_\emptyset\right) = \prod_{v\in V} \left(1 - \sum_{u\in N_v^i} b_{uv}\right) = \prod_{v\in V} \left(1 - \frac{|N_v^i|}{n^\gamma}\right) \geqslant \prod_{v\in V} \left(1 - \frac{1}{n^{\gamma-1}}\right) = \left(1 - \frac{1}{n^{\gamma-1}}\right)^n$$

$$\overset{(a)}{=} \sum_{i=0}^{n} \binom{n}{i} (1)^{n-i} \left(\frac{-1}{n^{\gamma-1}} \right)^i = \sum_{i=0}^{n} \binom{n}{i} \frac{(-1)^i}{n^{i(\gamma-1)}}$$

$$\overset{(b)}{\geqslant} \binom{n}{0} - \binom{n}{1} \frac{1}{n^{\gamma-1}} + \sum_{i=2}^{\lfloor n/2 \rfloor} \left(\binom{n}{i} \frac{1}{n^{2i(\gamma-1)}} - \binom{n}{i+1} \frac{1}{n^{(2i+1)(\gamma-1)}} \right)$$

$$\overset{(c)}{\geqslant} 1 - \frac{1}{n^{\gamma-2}}$$

where in (a) we used the binomial expansion, (b) is due to last negative term in the lhs that does not appear in the rhs when n is even, and (c) is due to

$$\binom{n}{i} \frac{1}{n^{2i(\gamma-1)}} \geqslant \binom{n}{i+1} \frac{1}{n^{(2i+1)(\gamma-1)}},$$

for any $\gamma \geqslant 2$. Since $\mathbf{P}\left(G'_\emptyset\right) \leqslant 1$, then $\mathbf{P}\left(G'_\emptyset\right) = \Theta(1)$. Moreover, $\sum_{G' \neq G'_\emptyset} \mathbf{P}\left(G'\right) = \mathcal{O}\left(\frac{1}{n^{\gamma-2}}\right)$

The score obtained by $c_\star$ in a live-edge graph G' starting from any initial set of seed nodes S is

$$F_{G'}(c_\star, S) = \sum_{v \in R_{G'}(S)} \frac{\pi_v(c_\star) + \sum_{u \in R_{G'}(S) \cap N_v^i} \frac{1}{n^\gamma}}{1 + \sum_{u \in R_{G'}(S) \cap N_v^i} \frac{1}{n^\gamma}} = \Theta\left(\frac{1}{n^\gamma} \sum_{v \in R_{G'}(S)} |R_{G'}(S) \cap N_v^i| \right),$$

since $1 \leqslant 1 + \sum_{u \in R_{G'}(S) \cap N_v^i} \frac{1}{n^\gamma} \leqslant 2$ for each $v \in R_{G'}(S)$. Note that $\sum_{v \in R_{G'}(S)} |R_{G'}(S) \cap N_v^i|$ is equal to the number of edges of the subgraph induced by the set $R_{G'}(S)$ of nodes reachable from S in G', which is not greater than n^2, and thus $F_{G'}(c_\star, S) = \mathcal{O}\left(\frac{1}{n^{\gamma-2}}\right)$.

Note that in the empty live edge graph $G'_\emptyset$ the set $R_{G'_\emptyset}(S)$ at the end of LTM is equal to S, since the graph has no edges. Thus

$$F_{G'_\emptyset}(c_\star, S) = \frac{1}{n^\gamma} \cdot \sum_{v \in S} \frac{|S \cap N_v^i|}{1 + \sum_{u \in S \cap N_v^i} \frac{1}{n^\gamma}}$$

and since the denominator is, again, bounded by two constants we have that

$$F_{G'_\emptyset}(c_\star, S) = \Theta\left(\frac{\sum_{v \in S} |S \cap N_v^i|}{n^\gamma} \right) = \Theta\left(\frac{\mathrm{SOL}_{DkS}(S)}{n^\gamma} \right),$$

where $\mathrm{SOL}_{DkS}(S) := \sum_{v \in S} |S \cap N_v^i|$ is the number of edges of the subgraph induced by S, i.e., the value of the objective function of DkS for solution S.

Thus, the expected final score of the target candidate is

$$F(c_\star, S) = \sum_{G' \in \mathcal{G}} F_{G'}(c_\star, S) \cdot \mathbf{P}(G') = F_{G'_\emptyset}(c_\star, S) \cdot \mathbf{P}(G'_\emptyset) + \sum_{G' \neq G'_\emptyset} F_{G'}(c_\star, S) \cdot \mathbf{P}(G').$$

Since $F_{G'}(c_\star, S)$ and $\sum_{G' \neq G'_\emptyset} \mathbf{P}\left(G'\right)$ are in $\mathcal{O}\left(\frac{1}{n^{\gamma-2}}\right)$, then

$$\sum_{G' \neq G'_\emptyset} F_{G'}(c_\star, S) \cdot \mathbf{P}(G') = \mathcal{O}\left(\frac{1}{n^{\gamma-2}}\right) \sum_{G' \neq G'_\emptyset} \mathbf{P}(G') = \mathcal{O}\left(\frac{1}{n^{2(\gamma-2)}}\right) = \mathcal{O}\left(\frac{\mathrm{SOL}_{DkS}(S)}{n^\gamma}\right),$$

for any $\gamma \geqslant 4$. Thus

$$F(c_\star, S) = \Theta\left(\frac{\mathrm{SOL}_{DkS}(S)}{n^\gamma}\right) \cdot \Theta(1) + \mathcal{O}\left(\frac{\mathrm{SOL}_{DkS}(S)}{n^\gamma}\right)$$

which means that $F(c_\star, S) = \Theta\left(\frac{\mathrm{SOL}_{DkS}(S)}{n^\gamma}\right)$. We apply the Bachmann-Landau definition of Θ notation: There exist three positive constants n_0, β_1, and β_2 such that, for all $n > n_0$,

$$\beta_1 \frac{\mathrm{SOL}_{DkS}(S)}{n^\gamma} \leqslant F(c_\star, S) \leqslant \beta_2 \frac{\mathrm{SOL}_{DkS}(S)}{n^\gamma}.$$

Note that, in this case, the constants n_0, β_1, and β_2 do not depend on the specific instance.

Since the previous bounds hold for any set S we also have that $\beta_1 \frac{\text{OPT}_{DkS}}{n^\gamma} \leqslant \text{OPT} \leqslant \beta_2 \frac{\text{OPT}_{DkS}}{n^\gamma}$, where OPT is the value of an optimal solution for PLTR and OPT_{DkS} is the value of an optimal solution for DkS.

Suppose that there exists an α-approximation algorithm for PLTR, i.e., an algorithm that finds a set S such that the value of its solution is $\text{MoV}(S) = 2F(c_\star, S) \geqslant \alpha \cdot \text{OPT}$. Then,

$$\frac{\alpha}{2} \cdot \beta_1 \frac{\text{OPT}_{DkS}}{n^\gamma} \leqslant \frac{\alpha}{2} \cdot \text{OPT} \leqslant F(c_\star, S) \leqslant \beta_2 \frac{\text{SOL}_{DkS}(S)}{n^\gamma}.$$

Thus $\text{SOL}_{DkS}(S) \geqslant \frac{\alpha}{2} \frac{\beta_1}{\beta_2} \text{OPT}_{DkS}$, i.e., the solution is an $\alpha\beta$-approximation to DkS, with $\beta := \frac{\beta_1}{2\beta_2}$. □

Theorem 2. *Election control in R-PLTR is NP-hard.*

Proof. We prove the hardness by reduction from Influence Maximization under LTM, which is known to be NP-hard [KKT15].

Consider an instance $\mathcal{I}_{\text{LTM}} = (G, B)$ of Influence Maximization under LTM. $\mathcal{I}_{\text{LTM}}$ is defined by a weighted graph $G = (V, E, w)$ with weight function $w : E \to [0, 1]$ and by a budget B. Let $\mathcal{I}_{\text{R-PLTR}} := (G', B)$ be the instance that corresponds to $\mathcal{I}_{\text{LTM}}$ on R-PLTR, defined by the same budget B and by a graph $G' = (V', E', w')$ that can be built as follows:

1. Duplicate each vertex in the graph, i.e., we define the new set of nodes as $V' := V \cup \{v_{|V|+1}, \ldots, v_{2|V|}\}$.
2. Add an edge between each vertex $v \in V$ to its copy in V', i.e., we define the new set of edges as $E' := E \cup \{(v_1, v_{|V|+1}), \ldots, (v_{|V|}, v_{2|V|})\}$.
3. Keep the same weight for each edge in E and we set the weights of all new edges to 1, i.e., $w'(e) = w(e)$ for each $e \in E$ and $w'(e) = 1$ for each $e \in E' \setminus E$. Note that the constraint on incoming weights required by LTM is not violated by w'.
4. Consider m candidates $c_\star, c_1, \ldots, c_{m-1}$. For each $v \in V$ we set $\pi_v(c_\star) = 1$ and $\pi_v(c_i) = 0$ for any other candidate $i \in \{1, \ldots, m-1\}$. For each $v \in V' \setminus V$ we set $\pi_v(c_\star) = 0$, $\pi_v(c_1) = 1$ and $\pi_v(c_i) = 0$ for any other candidate $i \in \{2, \ldots, m-1\}$.

Let S be the initial set of seed nodes of size B that maximizes $\mathcal{I}_{\text{LTM}}$ and let A be the set of active nodes at the end of the process. The value of the MoV obtained by S in $\mathcal{I}_{\text{R-PLTR}}$ is $\text{MoV}(S) = |V| - |V \setminus A|$. Indeed, each node $v \in V$ in G' has $\tilde{\pi}_v(c_\star) = \pi_v(c_\star) = 1$, because the probability of voting for the target candidate remains the same after the normalization. Moreover, each node $v_i \in V \cap A$ influences its duplicate $v_{|V|+i}$ with probability 1 and therefore $\tilde{\pi}_{v_{|V|+i}}(c_\star) = (\pi_{v_{|V|+i}}(c_\star) + 1)/2 = \frac{1}{2}$. Therefore, $F(c_\star, \emptyset) = F(c_1, \emptyset) = |V|$, $F(c_\star, S) = |V| + \frac{1}{2}|A|$, and $F(c_1, S) = |V \setminus A| + \frac{1}{2}|A|$.

Let S be the initial set of seed nodes of size B that achieves the maximum in $\mathcal{I}_{\text{R-PLTR}}$. Without loss of generality, we can assume that $S \subseteq V$, since we can replace any seed node $v_{|V|+i}$ in $V' \setminus V$ with its corresponding node v_i in V without decreasing the objective function. If A is the set of active nodes at the end of the process, then by using similar arguments as before, we can prove that $\text{MoV}(S) = |V| - |V \setminus A|$. Let us assume that S does not maximize $\mathcal{I}_{\text{LTM}}$, then, S would also not maximize $\mathcal{I}_{\text{R-PLTR}}$, which is a contradiction since S is an optimal solution for $\mathcal{I}_{\text{R-PLTR}}$.

We can prove the NP-hardness for the case of maximizing the score by using the same arguments. In fact, notice that maximizing the score of $c_\star$, i.e., $F(c_\star, S) = |V| + \frac{1}{2}|A|$, is exactly equivalent to maximize the cardinality of the active nodes in LTM. □

2 Simulations

We simulate our model on two real-world social networks[1] on which political campaigning messages could spread:

- *polbooks*: an undirected network with 105 nodes and 882 edges where nodes are political books and edges represent co-purchasing behavior; nodes are labeled as "liberal," "conservative," or "neutral."
- *polblogs*: a directed network with 1,224 nodes and 19,025 edges where nodes are web blogs about US politics and edges hyperlinks connecting them; nodes are labeled as "liberal" or "conservative."

The number of candidates in our simulations is based on the ground truth of the datasets; as mentioned earlier, *polbooks* has three clusters and *polblogs* has two clusters based on different US political parties. We set the probability of each node v to vote for, say, a "liberal" candidate proportionally to the number of neighbors labeled as "liberal," i.e., we set $\pi_v(c) = \frac{|N_v \cap B|}{|N_v|}$ where c is the "liberal" candidate, B is the set of nodes labeled as "liberal," and N_v is the set of neighbors of v. For each node v we sampled the "non-incoming influence weight" $\bar{b}_v$ uniformly at random in $[0, 1]$ and assigned the remaining influence weight uniformly among its incoming neighbors, i.e., we assigned to each edge (u, v) a weight $b_{uv} = \frac{1-\bar{b}_v}{|N_v^i|}$.

In our simulations we run our Greedy algorithm for the election control problem in R-PLTR. Then, we measure the score and the *Margin of Victory* (MoV) of each candidate using as starting seed nodes the ones found by the algorithm both in PLTR and in R-PLTR. We run the simulation considering each different candidate as the target one to cover multiple scenarios, considering as budget values the ones in $\{0, 1, 5, 10\}$. Then, as baseline to compare, we also considered as seed nodes the most influential ones, i.e., the nodes selected by GREEDY to solve the standard Influence Maximization problem.

For the implementation, we used .Net framework 4.6.2 and C# programming language. We have implemented five different classes for managing the graph, the LTM process, the PLTR process, and a GUI. We execute the simulations on a system with the following specifications: CPU Intel Core i7-6700HQ 2.6 GHz, with 4×32 KB 8-way L1 (data and inst) cache, and 4×256 KB 4-way L2 cache, and 6 MB 12-way L3 cache, RAM 16G DDR4. Each simulation has a running time of approximately 40 seconds for *poolbooks* and 140 minutes for *polblogs*.

The results relative to the scores are shown in Figures 1 and 2. As expected, the effect of our algorithm in R-PLTR is amplified w.r.t. PLTR, since it affects a greater number of voters. Taking as example the "liberal" candidate in *polbooks*, we need a budget $B = 5$ to make it overtake the "conservative" candidate in PLTR, while a budget $B = 1$ is enough in R-PLTR (see Figure 1, left column); in *polblogs*, instead, we are not able to make the "liberal" candidate win in PLTR with budget $B = 10$, but it is enough a budget $B = 5$ to make it overtake the "conservative" candidate in R-PLTR.

The results relative to MoV are presented in Figure 3. We can note that, as a general trend, candidates with lower probability of winning, are the most affected by the influence generated by the seed nodes selected by our algorithm both in PLTR and R-PLTR. The "neutral" and "liberal" candidates, respectively last and second last voted, have the higher MoV in *polbooks* (see Figure 3, on the left), while the "liberal" candidate, which was losing the elections, has the higher MoV in *polblogs* (see Figure 3, on the right).

Finally, in Figure 4 we present the difference between the MoV calculated by our algorithm and the MoV calculated using the greedy algorithm for the Influence Maximization problem. The simulations show that our algorithm outperforms the standard Greedy algorithm, as expected. The only scenario in which our algorithm performs worse is that in which we influence, with low budget, the already winning candidate (see Figure 4, on the left, red lines). The reason why our algorithm works better than a simple Greedy is that it looks for seeds that will influence "critical" voters, i.e., voters on which the influence will have more impact on the global score of the candidates, while the simple Greedy algorithm just looks for influential voters, independently from their initial opinion.

[1]The datasets are taken from `http://networkrepository.com/`

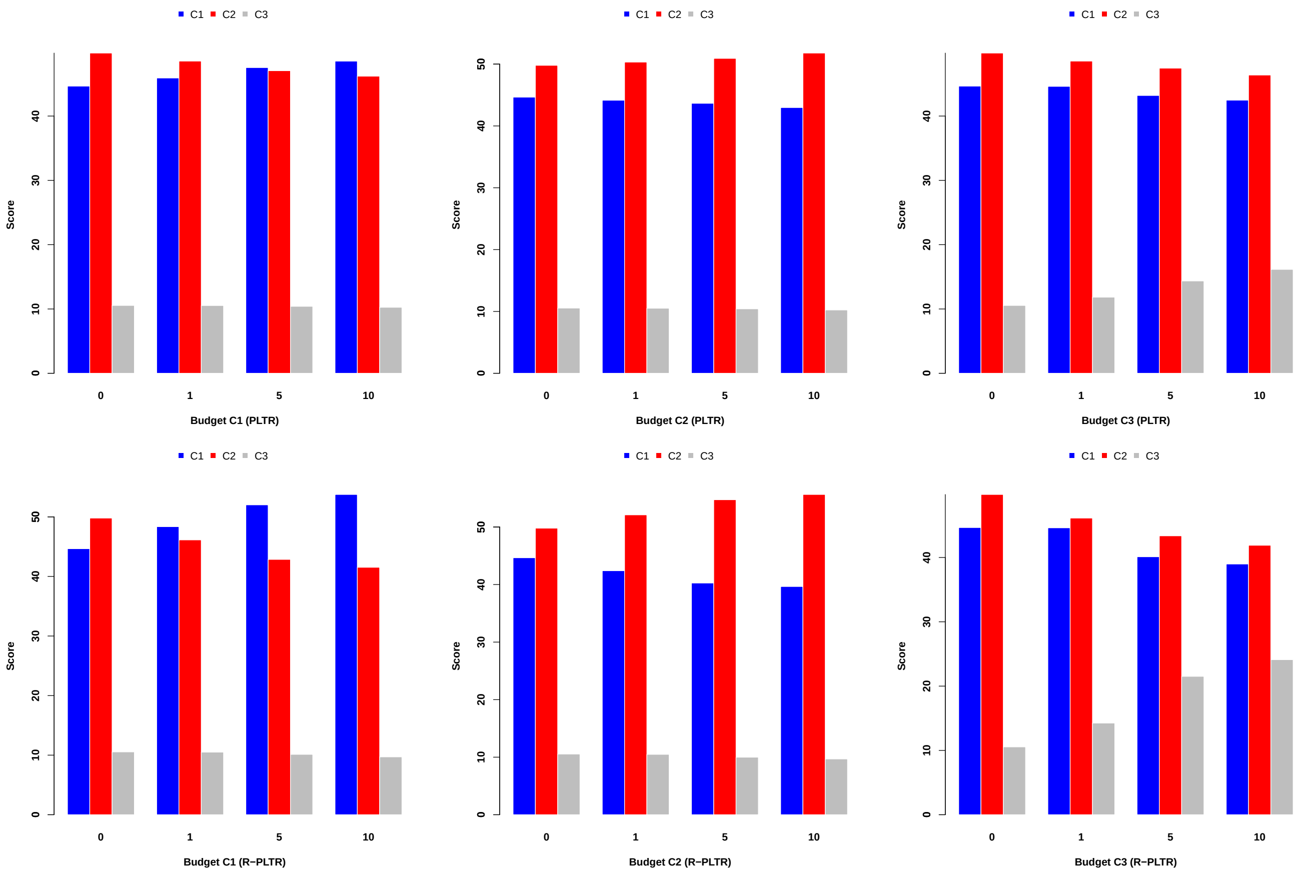

Figure 1: Candidates' scores in *polbooks* in PLTR (plots on the top) and R-PLTR (plots on the bottom), considering as target candidate the "liberal" (left), the "conservative" (middle), and the "neutral" (right).

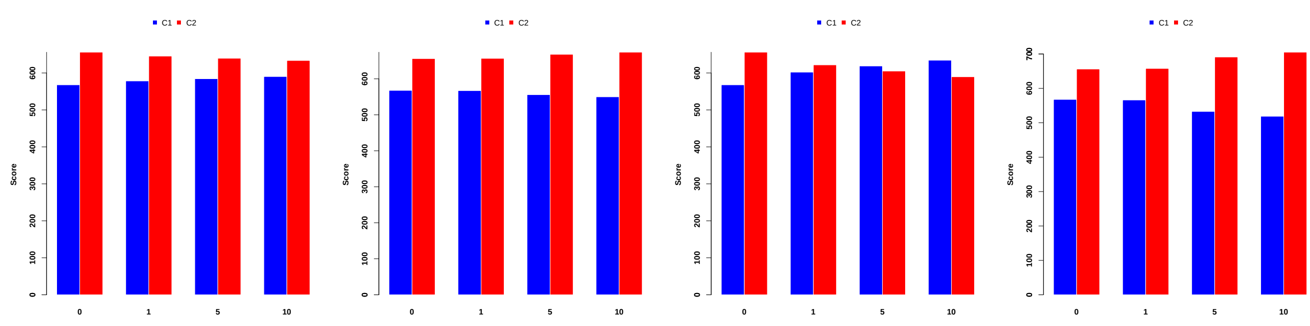

Figure 2: Candidates' scores in *polblogs* in PLTR (plots on the left) and R-PLTR (plots on the right), considering as target candidate the "liberal" (left) and the "conservative" (right).

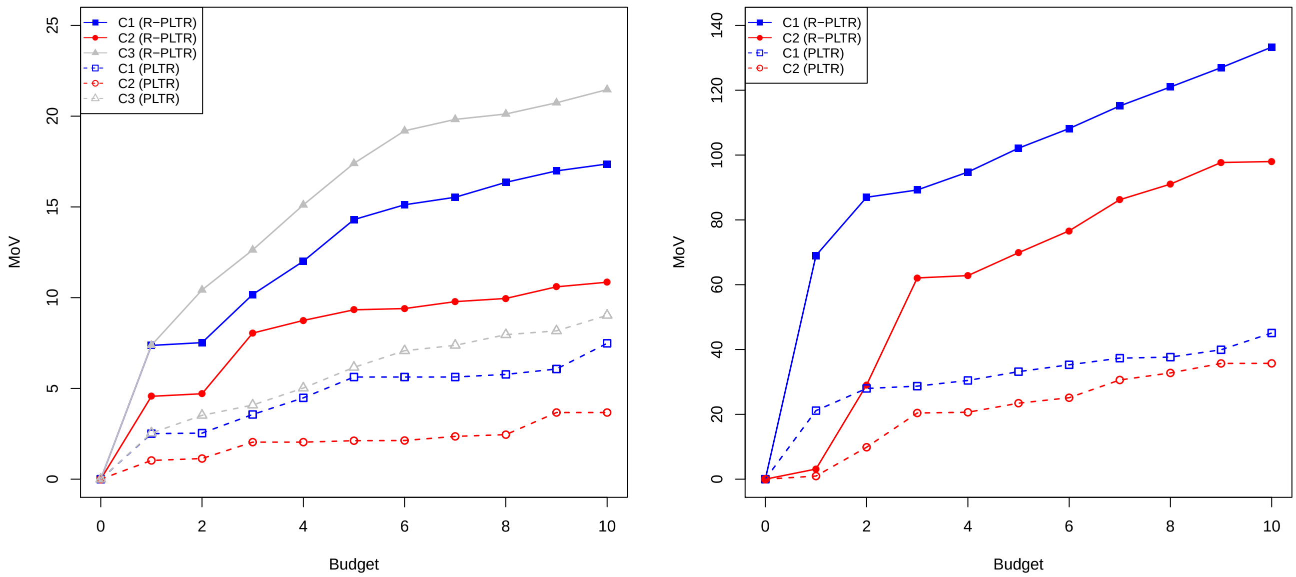

Figure 3: The MoV calculated using the presented algorithm for *polbooks* (left) *polblogs* (right), both in PLTR (dashed line) and R-PLTR (solid line), considering as target candidate the “liberal” (blue line), the “conservative” (red line), and the “neutral” (grey line).

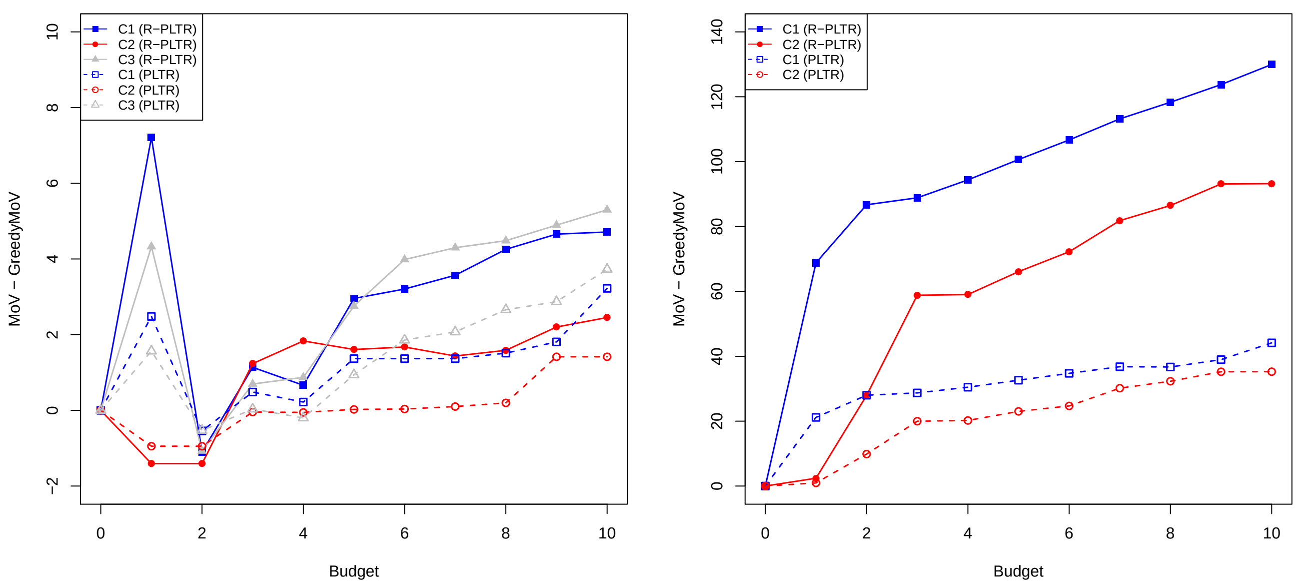

Figure 4: Difference between MoV obtained using our greedy algorithm and MoV obtained using the standard greedy algorithm for Influence Maximization problem. Values greater than 0 are when our algorithm performs better than the simple Greedy for Influence Maximization.

References

[KKT15] David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. *Theory of Computing*, 11(4):105–147, 2015.