A Scheduling Approach to Coalitional Manipulation
A presentation on the paper authored by Lirong Xia, Vincent Conitzer, Ariel D. Procaccia

Rupam Acharyya

March 25, 2015
Outline

- Coalition Manipulation Problem
Outline

- Coalition Manipulation Problem
- Connection between coalition manipulation and scheduling
Outline

- Coalition Manipulation Problem
- Connection between coalition manipulation and scheduling
- Coalition Optimization Problem
Outline

- Coalition Manipulation Problem
- Connection between coalition manipulation and scheduling
- Coalition Optimization Problem
- Tightness
Outline

- Coalition Manipulation Problem
- Connection between coalition manipulation and scheduling
- Coalition Optimization Problem
- Tightness
- Future Research
\begin{itemize}
  \item $C$ be the set of alternatives.
\end{itemize}
• $C$ be the set of alternatives.
• $L(C)$ be the set of all linear orders (Transitive and Antisymmetric and total relation).
Notations and Definitions

- $C$ be the set of alternatives.
- $L(C)$ be the set of all linear orders (Transitive and Antisymmetric and total relation).
- $\Delta L(C)$ be the set of all convex combinations over $L(C)$. 

An indivisible vote is a linear order over $C$ i.e. an element of $L(C)$.

A divisible vote is an element of $\Delta L(C)$. 

Rupam Acharyya
A Scheduling Approach to Coalitional Manipulation
C be the set of alternatives.

$L(C)$ be the set of all linear orders (Transitive and Antisymmetric and total relation).

$\Delta L(C)$ be the set of all convex combinations over $L(C)$.

An indivisible vote is a linear order over $C$ i.e. an element of $L(C)$. 
Notations and Definitions

- $C$ be the set of alternatives.
- $L(C)$ be the set of all linear orders (Transitive and Antisymmetric and total relation).
- $\Delta L(C)$ be the set of all convex combinations over $L(C)$.
- An **indivisible** vote is a linear order over $C$ i.e., an element of $L(C)$.
- A **divisible** vote is an element of $\Delta L(C)$. 

Rupam Acharyya

A Scheduling Approach to Coalitional Manipulation
Convex combination of $n$ vectors $\vec{a}_1, \ldots, \vec{a}_n$ is defined as:

$$\sum_{i=1}^{n} \alpha_i \cdot \vec{a}_i$$

where $\alpha_i$'s are scalars and $\sum_{i=1}^{n} \alpha_i = 1$
Formal Definitions

**Definition**

The Unweighted Coalitional Manipulation (UCM) problem is defined as follows. An instance is a tuple \((r, P^{NM}, c, k)\), where \(r\) is a voting rule, \(P^{NM}\) is the non-manipulators profile, \(c\) is the alternative preferred by the manipulators, and \(k\) is the number of manipulators. We are asked whether there exists a profile \(P^M\) of indivisible votes for the manipulators such that \(c \in r(P^{NM} \cup P^M)\).

**Definition**

The Weighted Coalitional Manipulation (WCM) problem is defined as follows. An instance is a tuple \((r, P^{NM}, \vec{w}_{NM}, c, k, \vec{w}_M)\), where \(r\) is a voting rule, \(P^{NM}\) is the non-manipulators profile, \(\vec{w}_{NM}\) represents the weights of \(P^{NM}\), \(c\) is the alternative preferred by the manipulators, \(k\) is the number of manipulators, and \(\vec{w}_M = (w_1, \ldots, w_k)\) represents the weights of the manipulators. We are asked whether there exists a profile \(P^M\) of indivisible votes for the manipulators such that \(c \in r((P^{NM}, P^M), (\vec{w}_{NM}, \vec{w}_M))\).
**Definition**

The Unweighted Coalitional Manipulation (UCM) problem is defined as follows. An instance is a tuple \((r, P^{NM}, c, k)\), where \(r\) is a voting rule, \(P^{NM}\) is the non-manipulators profile, \(c\) is the alternative preferred by the manipulators, and \(k\) is the number of manipulators. We are asked whether there exists a profile \(P^M\) of indivisible votes for the manipulators such that \(c \in r(P^{NM} \cup P^M)\).

**Definition**

The Weighted Coalitional Manipulation (WCM) problem is defined as follows. An instance is a tuple \((r, P^{NM}, \vec{w}^{NM}, c, k, \vec{w}^M)\), where \(r\) is a voting rule, \(P^{NM}\) is the non-manipulators profile, \(\vec{w}^{NM}\) represents the weights of \(P^{NM}\), \(c\) is the alternative preferred by the manipulators, \(k\) is the number of manipulators, and \(\vec{w}^M = (w_1, \ldots, w_k)\) represents the weights of the manipulators. We are asked whether there exists a profile \(P^M\) of indivisible votes for the manipulators such that \(c \in r((P^{NM}, P^M), (\vec{w}^{NM}, \vec{w}^M))\).
The Unweighted Coalitional Optimization (UCO) problem is defined as follows. An instance is a tuple \((r, P^{NM}, c)\), where \(r\) is a voting rule, \(P^{NM}\) is the non-manipulators profile, and \(c\) is the alternative preferred by the manipulators. We must find the minimum \(k\) such that there exists a a profile \(P^M\) consisting of \(k\) indivisible manipulator votes that satisfies \(c \in r(P^{NM} \cup P^M)\).
Definition

The Unweighted Coalitional Optimization (UCO) problem is defined as follows. An instance is a tuple \((r, P^{NM}, c)\), where \(r\) is a voting rule, \(P^{NM}\) is the non-manipulators profile, and \(c\) is the alternative preferred by the manipulators. We must find the minimum \(k\) such that there exists a a profile \(P^M\) consisting of \(k\) indivisible manipulator votes that satisfies \(c \in r(P^{NM} \cup P^M)\)

Definition

The Coalitional Optimization for divisible votes (COd) problem is defined as follows. An instance is a tuple \((r, P^{NM}, \vec{w}^{NM}, c)\), where \(r\) is a voting rule, \(P^{NM}\) is the nonmanipulators profile, \(\vec{w}^{NM}\) represents the weights of \(P^{NM}\), and \(c\) is the alternative preferred by the manipulators. We are asked to find the minimum \(W^M\) such that there exist a divisible vote \(V^M\) for one manipulator with weight \(W^M\), such that \(c \in r((P^{NM}, \{V^M\}), (\vec{w}^{NM}, W^M))\)
Algorithms for WCM and COD

Rupam Acharyya

A Scheduling Approach to Coalitional Manipulation
Will Devise a poly-time algorithm for the WCMd by a reduction to the well known scheduling problem known as $Q|pmtn|C_{max}$. 
Will Devise a poly-time algorithm for the WCMd by a reduction to the well known scheduling problem known as $Q|pmtn|C_{\text{max}}$.

It also solves COD exactly.
**Instance of $Q|pmtn|C_{max}$:**

- $n'$ jobs $J = \{J_1, \ldots, J_{n'}\}$;
- $m'$ machines $M = \{M_1, \ldots M_{m'}\}$;
- each job $J_i$ has a workload $p_i \in \mathbb{R}^+$, and the processing speed of machine $M_i$ is $s^i \in \mathbb{R}^+$, that is, it will finish $s^i$ amount of work in one unit of time.
- **Preemptions** \(^1\) are allowed.

---

\(^1\)A preemption is an interruption of the job that is being processed on one machine (the job may be resumed later, not necessarily on the same machine)
Defining Scheduling Problem

**Instance of** $Q|pmtn|C_{max}$:

- $n'$ jobs $J = \{J_1, \ldots, J_{n'}\}$;
- $m'$ machines $M = \{M_1, \ldots M_{m'}\}$;
- each job $J_i$ has a workload $p_i \in \mathbb{R}^+$, and the processing speed of machine $M_i$ is $s^i \in \mathbb{R}^+$, that is, it will finish $s^i$ amount of work in one unit of time.
- **Preemptions** $^1$ are allowed.

**Question:** What is the minimum makespan i.e. the minimum time to complete all the jobs and what is the optimal schedule?

---

$^1$A preemption is an interruption of the job that is being processed on one machine (the job may be resumed later, not necessarily on the same machine)
After counting non-manipulators’ score let $p_i = \text{non-manipulator score of } c_i - \text{non-manipulator score of } c$. 
After counting non-manipulators’ score let $p_i = \text{non-manipulator score of } c_i - \text{non-manipulator score of } c$.

$p_i$’s can be seen as workload of machine of $M_i$.  

Connection between WCMd under positional scoring rules and $Q|pmtn|C_{\text{max}}$
After counting non-manipulators’ score let $p_i = \text{non-manipulator score of } c_i - \text{non-manipulator score of } c$.

$p_i$’s can be seen as workload of machine of $M_i$.

WLOG manipulators will rank $c$ in the top position. So a manipulator vote in which $c_j$ is ranked in the $i^{th}$ position decreases the gap between $c_j$ and $c$ by $s_1 - s_i$ (which is same as speed).
After counting non-manipulators’ score let \( p_i = \) non-manipulator score of \( c_i \) - non-manipulator score of \( c \).

\( p_i \)'s can be seen as workload of machine of \( M_i \).

WLOG manipulators will rank \( c \) in the top position. So a manipulator vote in which \( c_j \) is ranked in the \( i^{th} \) position decreases the gap between \( c_j \) and \( c \) by \( s_1 - s_i \)(which is same as speed).

So we consider \( m - 1 \) machines whose speeds are \( s_1 - s_2, \ldots, s_1 - s_m \) respectively.
So formally for a WCMd instance 
\(((s_1, \ldots s_m), P^{NM}, w^{NM}, c, k, (w_1, \ldots, w_k))\) the corresponding 
\(Q|pmtn|C_{max}\) instance will have:
Connection between WCMd under positional scoring rules and $Q|pmtn|C_{max}$

So formally for a WCMd instance $((s_1, \ldots, s_m), P^{NM}, w^{NM}, c, k, (w_1, \ldots, w_k))$ the corresponding $Q|pmtn|C_{max}$ instance will have:

- $m - 1$ jobs and $m - 1$ machines.
So formally for a WCMd instance 
((s_1, \ldots, s_m), P^{NM}, w^{NM}, c, k, (w_1, \ldots, w_k)) the corresponding 
\(Q|pmtn|C_{max}\) instance will have:

- \(m - 1\) jobs and \(m - 1\) machines.
- For any \(i \leq m - 1\) \(s^i = s_1 - s_{i+1}\).
So formally for a WCMd instance 
\(((s_1, \ldots, s_m), P^{NM}, w^{NM}, c, k, (w_1, \ldots, w_k))\) the corresponding 
\(Q|pmtn|C_{max}\) instance will have:

- \(m - 1\) jobs and \(m - 1\) machines.
- For any \(i \leq m - 1\) \(s^i = s_1 - s_{i+1}\).
- \(p_i = \max\{s^{NM}(c_i) - s^{NM}(c), 0\}\). 

Rupam Acharyya

A Scheduling Approach to Coalitional Manipulation
Let $m = 4$, $C = \{c, c_1, c_2, c_3\}$. 

The positional scoring rule is Borda (corresponds to scoring vector $(3, 2, 1, 0)$). The non manipulators have weight 1 (unweighted) and $P_{NM} = (V_{NM1}, V_{NM2}, V_{NM3}, V_{NM4})$ is their profile, defined as:

- $V_{NM1} = [c_1 \succ c_2 \succ c_3 \succ c]$
- $V_{NM2} = [c_2 \succ c_1 \succ c_3 \succ c]$
- $V_{NM3} = [c_1 \succ c_3 \succ c \succ c_2]$
- $V_{NM4} = [c_3 \succ c_2 \succ c \succ c_1]$

We have that $s(P_{NM}, c) = 2$, $s(P_{NM}, c_1) = 8$, $s(P_{NM}, c_2) = 7$, $s(P_{NM}, c_3) = 7$. 

Rupam Acharyya
A Scheduling Approach to Coalitional Manipulation
Let $m = 4$, $C = \{c, c_1, c_2, c_3\}$.

The positional scoring rule is Borda (corresponds to scoring vector $(3, 2, 1, 0)$). 

We have that $s(P_{NM}, c) = 2$, $s(P_{NM}, c_1) = 8$, $s(P_{NM}, c_2) = 7$, $s(P_{NM}, c_3) = 7$. 

Rupam Acharyya  A Scheduling Approach to Coalitional Manipulation
Let \( m = 4, C = \{ c, c_1, c_2, c_3 \} \).

The positional scoring rule is Borda (corresponds to scoring vector \((3, 2, 1, 0)\)).

The non manipulators have weight 1 (unweighted) and 
\[ P^{NM} = (V_1^{NM}, V_2^{NM}, V_3^{NM}, V_4^{NM}) \] 
is their profile, defined as:

- \( V_1^{NM} = [c_1 \succ c_2 \succ c_3 \succ c] \)
- \( V_2^{NM} = [c_2 \succ c_1 \succ c_3 \succ c] \)
- \( V_3^{NM} = [c_1 \succ c_3 \succ c \succ c_2] \)
- \( V_4^{NM} = [c_3 \succ c_2 \succ c \succ c_1] \)
Let $m = 4$, $C = \{c, c_1, c_2, c_3\}$.

The positional scoring rule is Borda (corresponds to scoring vector $(3, 2, 1, 0)$).

The non manipulators have weight 1 (unweighted) and $P^{NM} = (V_1^{NM}, V_2^{NM}, V_3^{NM}, V_4^{NM})$ is their profile, defined as:

- $V_1^{NM} = [c_1 \succ c_2 \succ c_3 \succ c]$
- $V_2^{NM} = [c_2 \succ c_1 \succ c_3 \succ c]$
- $V_3^{NM} = [c_1 \succ c_3 \succ c \succ c_2]$
- $V_4^{NM} = [c_3 \succ c_2 \succ c \succ c_1]$

We have that $s(P^{NM}, c) = 2$, $s(P^{NM}, c_1) = 8$, $s(P^{NM}, c_2) = 7$, $s(P^{NM}, c_3) = 7$. 
Therefore, we construct a $Q|\text{pmtn}|C_{\text{max}}$ instance in which there are 3 machines $M_1, M_2, M_3$ whose speeds are

- $s^1 = 1$
- $s^2 = 2$
- $s^3 = 3$

corresponding to the 2nd, 3rd and 4th position in the votes respectively,
Therefore, we construct a $Q|pmtn|C_{max}$ instance in which there are 3 machines $M_1, M_2, M_3$ whose speeds are

- $s^1 = 1$
- $s^2 = 2$
- $s^3 = 3$

corresponding to the 2nd, 3rd and 4th position in the votes respectively, and 3 jobs $J_1, J_2, J_3$, whose workloads are
Therefore, we construct a $Q|\text{pmtn}|C_{\text{max}}$ instance in which there are 3 machines $M_1, M_2, M_3$ whose speeds are

- $s^1 = 1$
- $s^2 = 2$
- $s^3 = 3$

corresponding to the 2nd, 3rd and 4th position in the votes respectively, and 3 jobs $J_1, J_2, J_3$, whose workloads are

- $p_1 = 6 = (8 - 2)$
- $p_2 = 5 = (7 - 2)$
- $p_3 = 5 = (7 - 2)$

respectively.
Figure 1: An example schedule. The machines are idle in shaded areas.
Some notations

- We say that $t$ is a preemptive break point if there is a preemption at time $t$. 

Some notations

- We say that $t$ is a preemptive break point if there is a preemption at time $t$.
- Let $B_f$ be the set of all preemptive break points of scheduling $f$. 
Example: The minimum makespan of the scheduling problem instance in previous example is \((6 + 5)/5 = 11/5\). An optimal schedule \(f\) is as follows:

- \(M_1\): For any \(0 \leq t \leq 11/5\), \(f(M_1, t) = J_3\)
- \(M_2\): For any \(0 \leq t \leq 8/5\), \(f(M_2, t) = J_2\); for any \(8/5 < t \leq 11/5\), \(f(M_2, t) = J_1\)
- \(M_3\): For any \(0 \leq t \leq 8/5\), \(f(M_3, t) = J_1\); for any \(8/5 < t \leq 11/5\), \(f(M_3, t) = J_2\)

\(t = 8/5\) is the only preemptive break point in this schedule.
Assign jobs to idle machines arbitrarily.
Assign jobs to idle machines arbitrarily.

If there are no preemption break points in \((W_{i-1}, W_i)\), we let manipulator \(i\) vote for
\[ c \succ f(M_1, W_{i-1+\epsilon}) \succ f(M_2, W_{i-1+\epsilon}) \succ \ldots \succ f(M_{m1}, W i - 1 + \epsilon), \]
where \(\epsilon > 0\) is sufficiently small.
Assign jobs to idle machines arbitrarily.

If there are no preemption break points in \((W_{i-1}, W_i)\), we let manipulator \(i\) vote for 
\[ c > f(M_1, W_{i-1} + \epsilon) > f(M_2, W_{i-1} + \epsilon) > \ldots > f(M_{m_1}, W_i - 1 + \epsilon), \]
where \(\epsilon > 0\) is sufficiently small.

If there are preemptive break points in \((W_{i_1}, W_i)\), denoted by 
\(T_a, T_{a+1}, \ldots, T_{a+b-1}\), then we let \(V_1^i, \ldots, V_{b+1}^i\) denote the orders that correspond to the schedule at times \(W_{i-1} + \epsilon, T_a + \epsilon, \ldots, T_{a+b-1} + \epsilon\), respectively. Let 
\[ \alpha_1^i = T_a W_{i-1}, \ldots, \alpha_{b+1}^i = W_i T_{a+b-1}. \]
We let manipulator \(i\) vote for 
\[ \sum_{j=1}^{b+1} \left[ \alpha_j^i / (W_i - W_{i-1}) \right] \cdot V_j^i. \]
Figure 2: Conversion of an optimal schedule to a solution for WCMd.
Suppose there are two manipulators whose weights are $w_1$ and $w_2$ as in Figure 2.

- Manipulator 1 votes $[(1/4)(c \succ c_1 \succ c_3 \succ c_2) + (1/4)(c \succ c_1 \succ c_2 \succ c_3) + (1/2)(c \succ c_2 \succ c_1 \succ c_3)];$
Suppose there are two manipulators whose weights are $w_1$ and $w_2$ as in Figure 2.

- Manipulator 1 votes $[(1/4)(c \succ c_1 \succ c_3 \succ c_2) + (1/4)(c \succ c_1 \succ c_2 \succ c_3) + (1/2)(c \succ c_2 \succ c_1 \succ c_3)]$;
- Manipulator 2 votes $(1/3)(c \succ c_2 \succ c_1 \succ c_3) + (1/3)(c \succ c_2 \succ c_3 \succ c_1) + (1/3)(c \succ c_3 \succ c_2 \succ c_1)$;
Here is the pseudocode...

```
Algorithm 1: compWCMd
1. \( \forall i \leq m - 1, s^i \leftarrow s_1 - s_{i+1} \)
2. \( \forall i \leq m - 1, p_i \leftarrow \max \{s(P^{NM}, w^{NM}, c), s(F^{NM}, w^{NM}, c), 0\} \)
3. Solve the \( Q[p_m n i, C_{max}] \) instance (for example, using the algorithm in [13]). Let \( w \) and \( f \) denote the minimum makespan and an extended optimal schedule; let \( T_1, \ldots, T_k \) denote the preemptive break points.
4. if \( w > W_k \) then
5. \( \text{return } \text{false.} \)
6. end
7. Let \( f' : M \times [0, W_k] \rightarrow J \) be such that \( f'(M_1, t), \ldots, f'(M_{m-1}, t) \) = \( \{J_1, \ldots, J_{m-1}\} \), and for any \( M \in M, t \in [0, W_k], \) we have that if \( f(M, t) \in J \), then \( f'(M, t) = f(M, t) \).
8. for \( i = 1 \) to \( k \) do
9. \( V_i = [c > f'(M_1, W_{i-1} + \epsilon) > \ldots > f'(M_{m-1}, W_{i-1} + \epsilon)] \)
10. \( j \leftarrow 2 \)
11. for each preemptive break point \( T \in (W_{i-1}, W_i) \) (in order) do
12. \( V_j = [c > f'(M_1, T + \epsilon) > \ldots > f'(M_{m-1}, T + \epsilon)] \)
13. \( j \leftarrow j + 1 \)
14. end
15. For any \( j \), let \( \alpha^j \) be the length of the \( j \)th interval in \([W_{i-1}, W_i]\) induced by the preemptive break points.
16. Let manipulator \( i \) vote \( \sum_j [\alpha^j/(W_i - W_{i-1})] \cdot V_j \), and add this vote to \( P\).
17. end
18. return \( P \)
```

So we have the following theorem:

**Theorem**

*Algorithm 1 solves WCMd and WCO (exactly) in polynomial time.*
Can we apply Algorithm 1 for this case too??
Can we apply Algorithm 1 for this case too??

**Problem:** Algorithm 1 cannot be directly applied to WCM, because the manipulators votes constructed in Line 16 can be divisible.
Here is the solution!!

Algorithm 2: compWCMi

This algorithm is the same as Algorithm 1, except for the following two lines:

3. Use the algorithm in [13] to solve the scheduling problem

16. Let manipulator $i$ vote for $V_{j^*}^i$, where for any $j \neq j^*$, $\alpha_{j^*}^i \geq \alpha_j^i$; and for any $j \neq j^*$, we add a new manipulator whose weight is $\alpha_j^i$, and let her vote $V_j^i$

---

Let the coalitional manipulation problem instance be the same as in Example 1. Suppose we have two manipulators whose weights are both 1; then, because the minimum makespan is $11/5 > 2$ (as observed in earlier), there is no solution to the WCMd and WCM problem instances. The solution to the $COd$ problem instance is $11/5$. 
Now suppose we have two manipulators, whose weights are \( w_1 = 1 \) and \( w_2 = \frac{6}{5} \), respectively. Let \( f \) be the optimal schedule defined in earlier. A solution to the \( WCMD \) problem instance is obtained as follows. Manipulator 1 votes \([c > c_3 > c_2 > c_1]\), and manipulator 2 votes \([(1/2)(c > c_3 > c_2 > c_1) + (1/2)(c > c_3 > c_1 > c_2)]\). For \( WCM \), the vote of manipulator 1 is the same, the vote of manipulator 2 is \([c > c_3 > c_2 > c_1]\), and there is one additional manipulator, whose weight is \( \frac{3}{5} \) and whose vote is \([c > c_3 > c_1 > c_2]\).
Suppose there are two manipulators whose weights are illustrated in Figure 2.

- The vote of manipulator 1 is $c > c_2 > c_1 > c_3$, and we introduce two new manipulators with weight $w_1/4$ whose votes are $c > c_1 > c_3 > c_2$ and $c > c_1 > c_2 > c_3$;
Suppose there are two manipulators whose weights are illustrated in Figure 2.

- The vote of manipulator 1 is \( c > c_2 > c_1 > c_3 \), and we introduce two new manipulators with weight \( w_1/4 \) whose votes are \( c > c_1 > c_3 > c_2 \) and \( c > c_1 > c_2 > c_3 \);
- The vote of manipulator 2 is \( c > c_2 > c_1 > c_3 \), and we introduce two new manipulators with weight \( w_2/3 \) whose votes are \( c > c_2 > c_3 > c_1 \) and \( c > c_3 > c_2 > c_1 \).
Suppose there are two manipulators whose weights are illustrated in Figure 2.

- The vote of manipulator 1 is $c \succ c_2 \succ c_1 \succ c_3$, and we introduce two new manipulators with weight $w_1/4$ whose votes are $c \succ c_1 \succ c_3 \succ c_2$ and $c \succ c_1 \succ c_2 \succ c_3$;
- The vote of manipulator 2 is $c \succ c_2 \succ c_1 \succ c_3$, and we introduce two new manipulators with weight $w_2/3$ whose votes are $c \succ c_2 \succ c_3 \succ c_1$ and $c \succ c_3 \succ c_2 \succ c_1$.

Since $|B_f|$, the number of break points is 4, there are in total four additional manipulators.
As for any $j \neq j^*$ $\alpha_j^i \leq (W_i - W_{i-1})/2 \leq W/2$ we have the following lemma

**Lemma**

If $w \geq W_k$, then there is no successful manipulation for WCMd (nor for WCMi); otherwise, Algorithm 2 returns a manipulation with at most $|B_f|$ additional manipulators, each with weight at most $W/2$. 
$|B_f|$ depends on the algorithm.
$|B_f|$ depends on the algorithm. But we have the following lemma:

**Lemma**

The number of preemptive break points in the solution of the algorithm of Gonzalez and Sahni\(^a\) is at most $m'1$. Furthermore, this bound is tight.

Hence we have the theorem...

**Theorem**

*Algorithm 2 runs in polynomial time and*

- *if the algorithm returns false, then there is no successful manipulation;*

- *otherwise, the algorithm returns a successful manipulation with a set of at most \( m - 2 \) additional manipulators, each with weight at most \( W/2 \).*
UCMd can be solved using the same algorithm (assuming weight = 1).
UCMd can be solved using the same algorithm (assuming weight = 1).

Interesting Case: Indivisible one.
UCMd can be solved using the same algorithm (assuming weight = 1).

Interesting Case: Indivisible one.

Tweak Algorithm 2.
Here is the algorithm...

**Algorithm 3: compWCMi**

This algorithm is the same as Algorithm 1, except for the following two lines:


16. Let manipulator \( i \) vote for \( V_1^i \); for any \( j > 1 \), we add a new manipulator who votes for \( V_j^i \).
Hence we have the following corollary...

**Corollary**

For UCMi, if Algorithm 3 returns false, then there is no successful manipulation; otherwise, Algorithm 3 returns a successful manipulation with at most $m^2$ additional manipulators.
Algorithm 3 is an approximation algorithm for UCO.

Lemma

Let $k \in 1, \ldots, m$. UCMi/UCOi under $k$-approval is in P.
Algorithm 3 is an approximation algorithm for UCO.

But there are some positional scoring voting rules under which UCM/UCO can be solved exactly.
Algorithm 3 is an approximation algorithm for UCO.

But there are some positional scoring voting rules under which UCM/UCO can be solved exactly.

**Lemma**

Let $k \in 1, \ldots, m_1$. $UCM_i/UCO_i$ under $k$-approval is in $P$. 
On the tightness of the results

**Theorem**

For any \( m \geq 3 \), there exists a UCO instance such that the (additive) gap between the optimal solution to UCOd and the optimal solution to UCOi is \( m - 2 \).
Proof of the theorem

For any \( m \geq 3 \) let

- the scoring vector is \((m(m1)(m2)1, \ldots, m(m1)(m2)1, m(m1)(m2)2, 0)\)
- \( V = [c_1 \succ \ldots c_{m-1} \succ c] \)
- Let \( \pi \) be the cyclic permutation on \( C\{c\} \).
- Let \( V_i \) be the linear order over \( C \) in which \( c \) is ranked in position \( (m1) \) defined by \( \pi^i(c_1) \succ \pi^i(c_{m-1}) \)
- Let \( P = (V, V_1, \ldots, V_{m1}) \) and \( P^{NM} = P \cup \pi(P) \cup \ldots \cup \pi^{m-2}(P) \)

It can be verified that for \( V' = [c, c_1, \ldots, c_{m-1}] \) the divisible vote

\[
\frac{1}{m-1}(V', \pi(V'), \pi^2(V'), \ldots, \pi^{m-2}(V'))
\]

is sufficient to make \( c \) win.
We next prove that the solution to UCO is $m1$. Clearly the profile $(V', \pi(V'), \pi^2(V'), \ldots, \pi^{m-1}(V'))$ is a successful manipulation. Hence, it remains to show that the solution is at least $m - 1$. For the sake of contradiction we assume that the solution is $m - 2$, and $P^M$ is the corresponding successful manipulation. Therefore, there must exist $i \leq m - 1$ such that $c_i$ is not ranked at the bottom of any of the votes of $P^M$. Therefore,

$$s(P^M, c) - s(P^M, c_i) \leq m - 2 \leq (m - 1)^2 - 1$$

(1)

which means that $s(P^{NM} \cup P^M, c) - s(P^{NM} \cup P^M, c_i) < 0$ This contradicts the assumption that $P^M$ is a successful manipulation.
Let $A$ be an approximation algorithm based on WCO. For any $m \geq 3$, there exists a UCO instance such that the gap between the optimal solution to $UCOi$ and the output of $A$ is $m^2$. 
Proof of theorem

For any \( m \geq 3 \), we construct an instance such that the solution to the UCO problem is 1, but at least \( m \) linear orders appear in any optimal solution to the COd problem (so the gap is \( m \)). We let the scoring vector be \((m + 2, 1, 0, \ldots, 0)\). Let

\[
V = [c \succ c_1 \succ \ldots \succ c_{m-1}]
\]

and

\[
V' = [c_{m-1} \succ c_1 \succ c \succ c_2 \succ \ldots \succ c_{m-2}]
\]

Furthermore, let \( \pi : c_1 \to c_2 \to \ldots \to c_{m-1} \to c \) and \( \pi^* : c \to c_1 \to \ldots \to c_{m-1} \to c \). We define preference profiles by letting

\[
P = \left( V', V, \pi^*(V), (\pi^*)^2(V), \ldots (\pi^*)^{m-2}(V) \right)
\]

and

\[
P^{NM} = P \cup \pi(P) \cup \ldots \cup \pi^{m-2}(P)
\]
We have that \( s(P, c) = m + 2 \), \( s(P, c_1) = m + 4 \), and for any \( 2 \leq i \leq m - 1 \), \( s(P, c_i) = m + 3 \). Therefore, 
\[ s(P^{NM}, c) = (m + 2)(m - 1) \] 
and for any \( 2 \leq i \leq m - 1 \),
\[ s(P^{NM}, c_i) = (m + 3)(m - 1) + 1. \] Therefore, for any \( i \leq m - 1 \),
\[ s(P^{NM}, c_i) - s(P^{NM}, c) = m. \] It follows that one manipulator suffices to make \( c \) the winner (by voting \( c \succ c_1 \succ ... \succ c_{m-1} \)).

On the other hand, the minimum weight for COd is \((m - 1)/m\) for example,

\[
V^M = \frac{m-1}{m} \left( \frac{1}{m-1} V + \frac{1}{m-1} \pi(V) + ... + \frac{1}{m-1} \pi^{m-2}(V) \right) \quad (6)
\]

In any manipulators vote corresponding to the minimum total weight, every alternative except \( c \) must appear in the second position for a fraction of the vote. Therefore, any algorithm based on COd must output at least \( m - 1 \) linear orders.
Theorem

UCM under a specific voting rule is strongly NP-complete, even when the number of manipulators is two.
Future Research

**Question 1:** Is there a polynomial-time algorithm that gives an additive approximation of less than $m^2$ to UCO under all positional scoring rules?
Question 2: What additive approximation to UCO does the Greedy algorithm give for positional scoring rules?
Question 3: Is UCM under Borda NP-complete?
THANK YOU!