AN ALGORITHM FOR DISTRIBUTING COALITIONAL VALUE CALCULATIONS AMONG COOPERATING AGENTS
Introduction

- Problem:
  - Computing coalitional values exponentially complex

- Solution:
  - Algorithm Distributing Work
Considerations to take:

- No bottleneck;
- Communication minimized;
- Redundancy minimized;
- Balanced work;
- Memory minimized.
## Previous Works

<table>
<thead>
<tr>
<th>Authors</th>
<th>Characteristic</th>
<th>Complexity</th>
<th>Overlapping coalitions?</th>
<th>Distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sandholm et al.</td>
<td>Anytime algorithm</td>
<td>Exponential</td>
<td>No</td>
<td>Expected amount</td>
</tr>
<tr>
<td>Dang and Jennings</td>
<td>Same results with smaller space search</td>
<td>Exponential</td>
<td>No</td>
<td>None</td>
</tr>
<tr>
<td>Shehory and Kraus</td>
<td>Limited coalition size</td>
<td>Polynomial</td>
<td>Yes</td>
<td>Negotiation</td>
</tr>
</tbody>
</table>
DCVC Algorithm – Basic Version

Each agent $a_i$ does:
- Sort the agents based on an UID
- For every permitted coalition size:
  - Calculate the size of your share: $N_{s,i} = \text{floor}(N_s / n)$;
  - Calculate the index of the last coalition in your share: $index_{s,i} = i \times N_{s,i}$;
  - Calculate the values of the coalitions.
## DCVC Algorithm – Basic Version

<table>
<thead>
<tr>
<th>$L_1$</th>
<th>$L_2$</th>
<th>$L_3$</th>
<th>$L_4$</th>
<th>$L_5$</th>
<th>$L_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>5, 6</td>
<td>4, 5, 6</td>
<td>3, 4, 5, 6</td>
<td>2, 3, 4, 5, 6</td>
<td>1, 2, 3, 4, 5, 6</td>
</tr>
<tr>
<td>5</td>
<td>4, 6</td>
<td>3, 5, 6</td>
<td>2, 4, 5, 6</td>
<td>1, 3, 4, 5, 6</td>
<td>1, 2, 3, 4, 5, 6</td>
</tr>
<tr>
<td>4</td>
<td>4, 5</td>
<td>3, 4, 6</td>
<td>2, 3, 5, 6</td>
<td>1, 2, 4, 5, 6</td>
<td>1, 2, 3, 4, 5, 6</td>
</tr>
<tr>
<td>3</td>
<td>3, 6</td>
<td>3, 4, 5</td>
<td>2, 3, 4, 6</td>
<td>1, 2, 3, 5, 6</td>
<td>1, 2, 3, 4, 5, 6</td>
</tr>
<tr>
<td>2</td>
<td>3, 5</td>
<td>2, 5, 6</td>
<td>2, 3, 4, 5</td>
<td>1, 2, 3, 4, 6</td>
<td>1, 2, 3, 4, 5, 6</td>
</tr>
<tr>
<td>1</td>
<td>3, 4</td>
<td>2, 4, 6</td>
<td>1, 4, 5, 6</td>
<td>1, 2, 3, 4, 5</td>
<td>1, 2, 3, 4, 5, 6</td>
</tr>
<tr>
<td>2, 6</td>
<td>2, 4, 5</td>
<td>1, 3, 5, 6</td>
<td>1, 3, 4, 6</td>
<td>1, 2, 3, 4, 5, 6</td>
<td></td>
</tr>
<tr>
<td>2, 5</td>
<td>2, 3, 5</td>
<td>1, 2, 5, 6</td>
<td>1, 2, 3, 4, 5, 6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2, 4</td>
<td>2, 3, 4</td>
<td>1, 2, 4, 6</td>
<td>1, 2, 3, 4, 5, 6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2, 3</td>
<td>1, 5, 6</td>
<td>1, 2, 4, 5</td>
<td>1, 2, 3, 4, 5, 6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1, 6</td>
<td>1, 4, 6</td>
<td>1, 2, 3, 6</td>
<td>1, 2, 3, 4, 5, 6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1, 5</td>
<td>1, 4, 5</td>
<td>1, 2, 3, 5</td>
<td>1, 2, 3, 4, 5, 6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1, 4</td>
<td>1, 3, 6</td>
<td>1, 2, 3, 4, 5, 6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1, 3</td>
<td>1, 3, 5</td>
<td>1, 2, 3, 4, 5, 6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1, 2</td>
<td>1, 3, 4</td>
<td>1, 2, 3, 4, 5, 6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1, 2, 6</td>
<td>1, 2, 5</td>
<td>1, 2, 3, 4, 5, 6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1, 2, 5</td>
<td>1, 2, 4</td>
<td>1, 2, 3, 4, 5, 6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1, 2, 4</td>
<td>1, 2, 3</td>
<td>1, 2, 3, 4, 5, 6</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

João Silva – CG
DCVC Algorithm – Basic Version

- Agent only knows the index of last coalition
- To know the last coalition:
  - Build Pascal table
  - Find first value such that \( \text{Pascal}[s,x] \geq \text{index} \)
  - First agent in the coalition is \( (n - s + 1) - x + 1 \)
  - For next agent repeat with \( s = s-1 \),
    \[ \text{index} = \text{index} - \text{Pascal}[s,x-1] \]
  - If \( \text{Pascal}[s,x] = \text{index} \) rest of agents are calculated by adding 1 to the previous agent
DCVC Algorithm – Basic Version

- **Index\(_{4,5}\) = 10**
  - *Pascal\([4,3]\) = 15*
  - *Agent \#1 = (6-4+1)-3+1 = 1*
  - *Next Index = 10-5 = 5*

- *Pascal\([3,3]\) = 10*
  - *Agent \#2 = (6-3+1)-3+1 = 2*
  - *Next Index = 5-4 = 1*

- *Pascal\([2,1]\) = 1*
  - *Agent \#3 = (6-2+1)-1+1 = 5*
  - *Since Pascal\([2,1]\) = Index:*
  - *Agent \#4 = Agent 3 + 1 = 6*

- *Last Coalition = \{1,2,5,6\}*
To know previous coalition:

- Check values $c_{i,s}$, $c_{i,s-1}$, ... where $c_{i,x}$ is the agent in position $x$ of coalition $i$.

- Find a value such that $c_{i,x} < c_{1,x}$, then:
  - $c_{i-1,k} = c_{i,k}$ : $1 \leq k < x$
  - $c_{i-1,k} = c_{i,k} + 1$ : $k = x$
  - $c_{i-1,k} = c_{i-1,k-1} + 1$ : $x < k \leq s$
DCVC Algorithm – Basic Version

- **Coalition** = \{1,2,5,6\}
  - \(c_{i,2} < c_{1,2}\) (2 < 4)
  - Position 1 : \(c_{i-1,k} = c_{i,k} = 1\)
  - Position 2 : \(c_{i-1,k} = c_{i,k} + 1 = 3\)
  - Position 3 : \(c_{i-1,k} = c_{i-1,k-1} + 1 = 4\)
  - Position 4 : \(c_{i-1,k} = c_{i-1,k-1} + 1 = 5\)

- **Previous Coalition** = \{1,3,4,5\}
For leftover coalitions:

- In the beginning each agent sets $\alpha = 1$;
- Calculate the number of leftover coalitions: 
  $$N' = N_s - (n \times N_{s,i})$$
- If $N' \neq 0$, $N'$ agents starting with $a_\alpha$ calculate one extra value and $\alpha$ is increased by:
  - $N': \alpha + N' < n$;
  - $N' - n : otherwise$. 
DCVC Algorithm – Basic Version

\[ \begin{array}{c}
\alpha = 1 \\
a_1 \{ 6 \\
a_2 \{ 5 \\
a_3 \{ 4 \\
a_4 \{ 3 \\
a_5 \{ 2 \\
a_6 \{ 1 \\
\end{array} \quad \begin{array}{c}
\alpha = 1 \\
a_1 \{ 5, 6 \\
a_2 \{ 4, 5 \\
a_3 \{ 3, 5, 6 \\
a_4 \{ 2, 6, 3 \\
a_5 \{ 2, 4 \\
a_6 \{ 1, 6 \\
\end{array} \quad \begin{array}{c}
\alpha = 4 \\
ap_1 \{ 4, 5, 6 \\
ap_2 \{ 3, 4, 5 \\
ap_3 \{ 2, 4, 5 \\
ap_4 \{ 2, 3, 4 \\
ap_5 \{ 1, 2, 3, 4 \\
ap_6 \{ 1, 2, 3, 5 \\
\end{array} \quad \begin{array}{c}
\alpha = 6 \\
ap_1 \{ 3, 4, 5, 6 \\
ap_2 \{ 2, 3, 5, 6 \\
ap_3 \{ 2, 3, 4, 5 \\
ap_4 \{ 1, 3, 4, 5 \\
ap_5 \{ 1, 3, 4, 6 \\
ap_6 \{ 1, 2, 3, 4, 5 \\
\end{array} \quad \begin{array}{c}
\alpha = 3 \\
ap_1 \{ 2, 3, 4, 5, 6 \\
ap_2 \{ 1, 3, 4, 5, 6 \\
ap_3 \{ 1, 2, 4, 5, 6 \\
ap_4 \{ 1, 2, 3, 4, 6 \\
ap_5 \{ 1, 2, 3, 4, 5 \\
ap_6 \{ 1, 2, 3, 4, 5 \\
\end{array} \quad \begin{array}{c}
\alpha = 3 \\
a_1 \{ 1, 2, 3, 4, 5, 6 \\
a_2 \{ 1, 3, 4, 5, 6 \\
a_3 \{ 1, 2, 4, 5, 6 \\
a_4 \{ 1, 2, 3, 4, 6 \\
a_5 \{ 1, 2, 3, 4, 5 \\
a_6 \{ 1, 2, 3, 4, 5 \\
\end{array} \end{array} \]
DCVC Algorithm – Modifying Assigned Coalitions

- Agent do different numbers of operations:
DCVC Algorithm – Modifying Assigned Coalitions

João Silva – CG
Dealing with Unavailable Agents

- Certain cases agents can’t join a certain coalition
  - Coalitions can’t overlap;
  - Resources needed for a coalition.

- Two ways of recalculating the values
  - Search through a set $P$ of potential coalitions;
  - Repeat the entire process.
Dealing with Unavailable Agents – Search through $P$

- Each agent contains $P_i$ (set of coalitions in its share) but not $P^*$ (set of possible coalitions).
  - Agents need to look at $A^*$ (agents that can form coalitions) and go through $P$

- Each agent contains $P_i$ and $P^*$.
  - Agents can simply go through $P^*$, but needs more memory
Dealing with Unavailable Agents - Computational Complexity
Performance Evaluation

- When compared with Shehory and Kraus algorithm (for the case of 25 agents):
  - Distribution: 0.02% of the time
  - Communication: from 1146989648 bytes to 0
  - Redundancy: from 383229848 redundant values to 0
  - Memory: 0.000006% of the memory
Questions?