AN ALGORITHM FOR DISTRIBUTING COALITIONAL VALUE CALCULATIONS AMONG COOPERATING AGENTS

João Silva

Cooperative Games

Introduction

- Problem:
  - Computing coalitional values exponentially complex

- Solution:
  - Algorithm Distributing Work

Introduction

- Considerations to take:
  - No bottleneck;
  - Communication minimized;
  - Redundancy minimized;
  - Balanced work;
  - Memory minimized.

Previous Works

Authors	Characteristic	Complexity	Overlapping coalitions?	Distribution
Sandholm et al.	Anytime algorithm	Exponential	No	Expected amount
Dang and Jennings	Same results with smaller space search	Exponential	No	None
Shehory and Kraus	Limited coalition size	Polynomial	Yes	Negotiation

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: $N_{s,i} = floor(N_s / n)$;
    - 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

L_1	L_2	L_3	L_4	L_5	L_6
6	5, 6	4, 5, 6	3, 4, 5, 6	2, 3, 4, 5, 6	1, 2. 3, 4, 5, 6
5	4, 6	3, 5, 6	2, 4, 5, 6	1, 3, 4, 5, 6	
4	4, 5	3, 4, 6	2, 3, 5, 6	1. 2, 4, 5, 6	
3	3, 6	3, 4, 5	2, 3, 4, 6	1, 2, 3, 5, 6	
2	3, 5	2, 5, 6	2, 3, 4, 5	1, 2, 3, 4, 6	
1	3, 4	2, 4, 6	1, 4, 5, 6	1, 2, 3, 4, 5	
	2, 6	2, 4, 5	1, 3, 5, 6		
	2, 5	2, 3, 6	1, 3, 4, 6		
	2, 4	2, 3, 5	1, 3, 4, 5		
	2, 3	2, 3, 4	1, 2, 5, 6		
	1, 6	1, 5, 6	1, 2, 4, 6		
	1, 5	1, 4, 6	1, 2, 4, 5		
	1, 4	1, 4, 5	1, 2, 3, 6		
	1, 3	1, 3, 6	1, 2, 3, 5		
	1, 2	1, 3, 5	1, 2, 3, 4		
		1, 3, 4			
		1, 2, 6			
		1, 2, 5			
		1, 2, 4			
		1, 2, 3			

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 *Pascal[s,x]* $\geq$ *index*
  - First agent in the coalition is $(n - s + 1) - x + 1$
  - For next agent repeat with $s = s-1$, $index = index - Pascal[s,x-1]$
  - If *Pascal[s,x] = index* rest of agents are calculated by adding 1 to the previous agent

1	2	3	4	5
1	3	6	10	15
1	4	10	20	35
1	5	15	35	70
1	6	21	56	126

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}

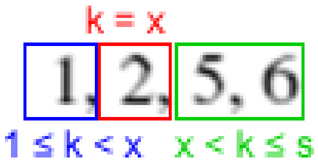
1	2	3	4	5
1	3	6	10	15
1	4	10	20	35
1	5	15	35	70
1	6	21	56	126

DCVC Algorithm – Basic Version

- 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 : $c_{i-1,k} = c_{i-1,k-1} + 1 = 5$



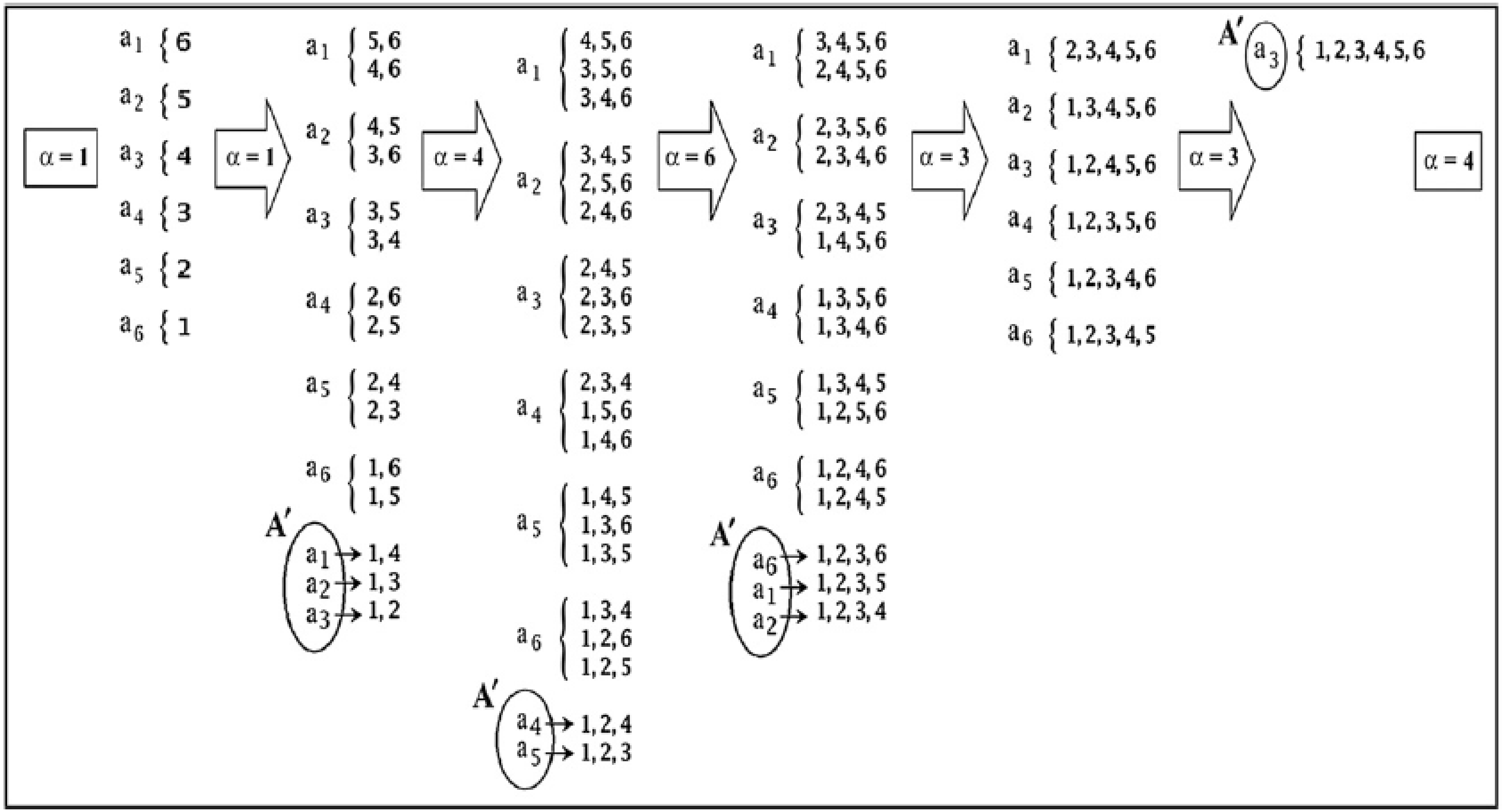
  - *Previous Coalition = {1,3,4,5}*

DCVC Algorithm – Basic Version

- For leftover coalitions:
  - In the beginning each agent sets α = 1;
  - Calculate the number of leftover coalitions:

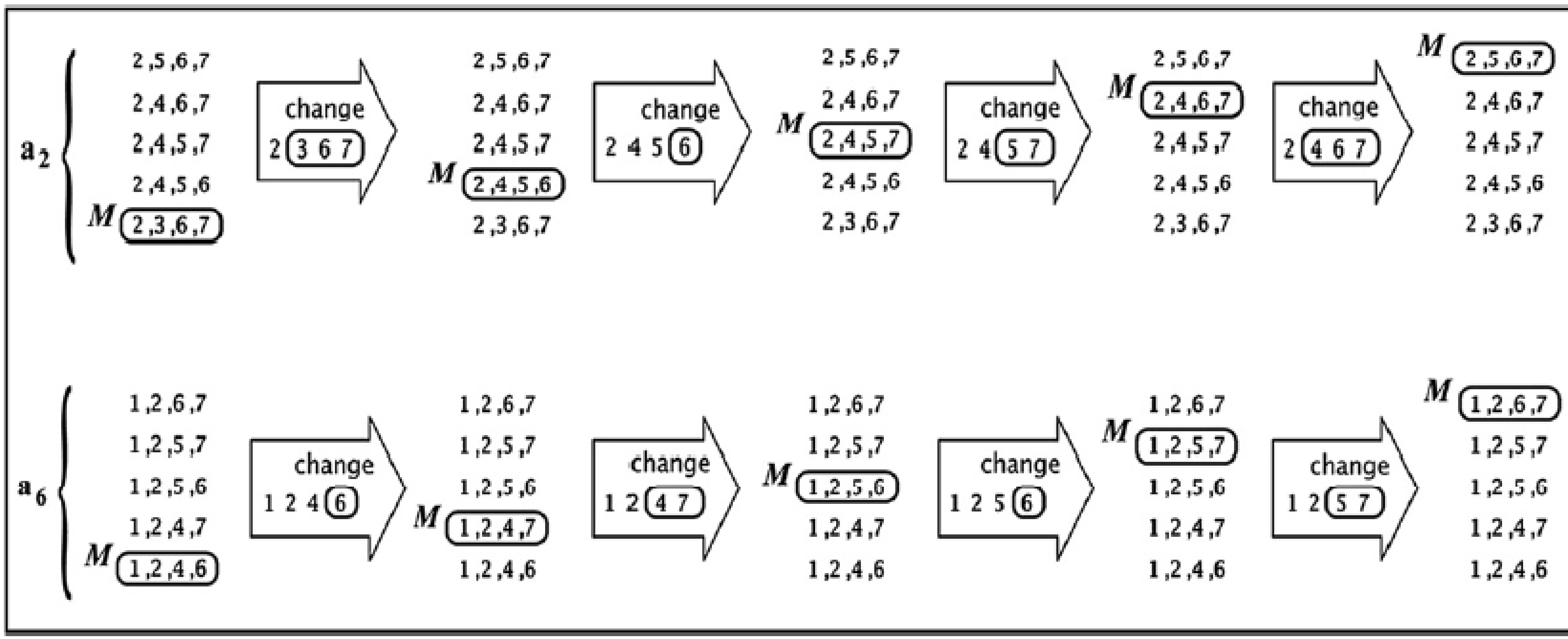
    $N' = N_s - (n \times N_{s,i})$
  - If $N' \neq 0$, N' agents starting with a_α calculate one extra value and α is increasead by:
    - N' : $\alpha + N' < n$;
    - $N' - n$: *otherwise.*

DCVC Algorithm – Basic Version

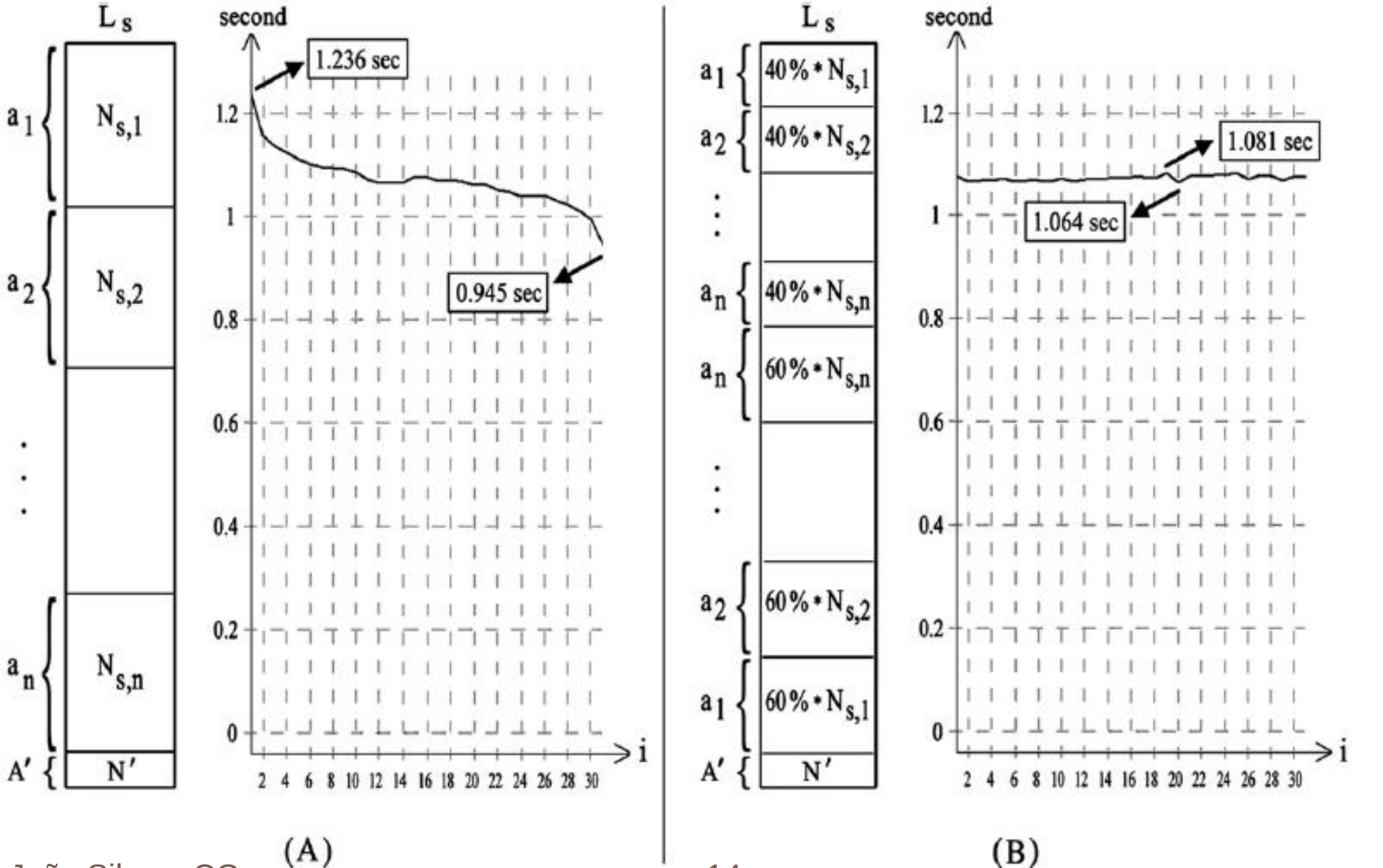


DCVC Algorithm – Modifying Assigned Coalitions

- Agent do different numbers of operations:



DCVC Algorithm – Modifying Assigned Coalitions



Dealing with Unavailable Agents

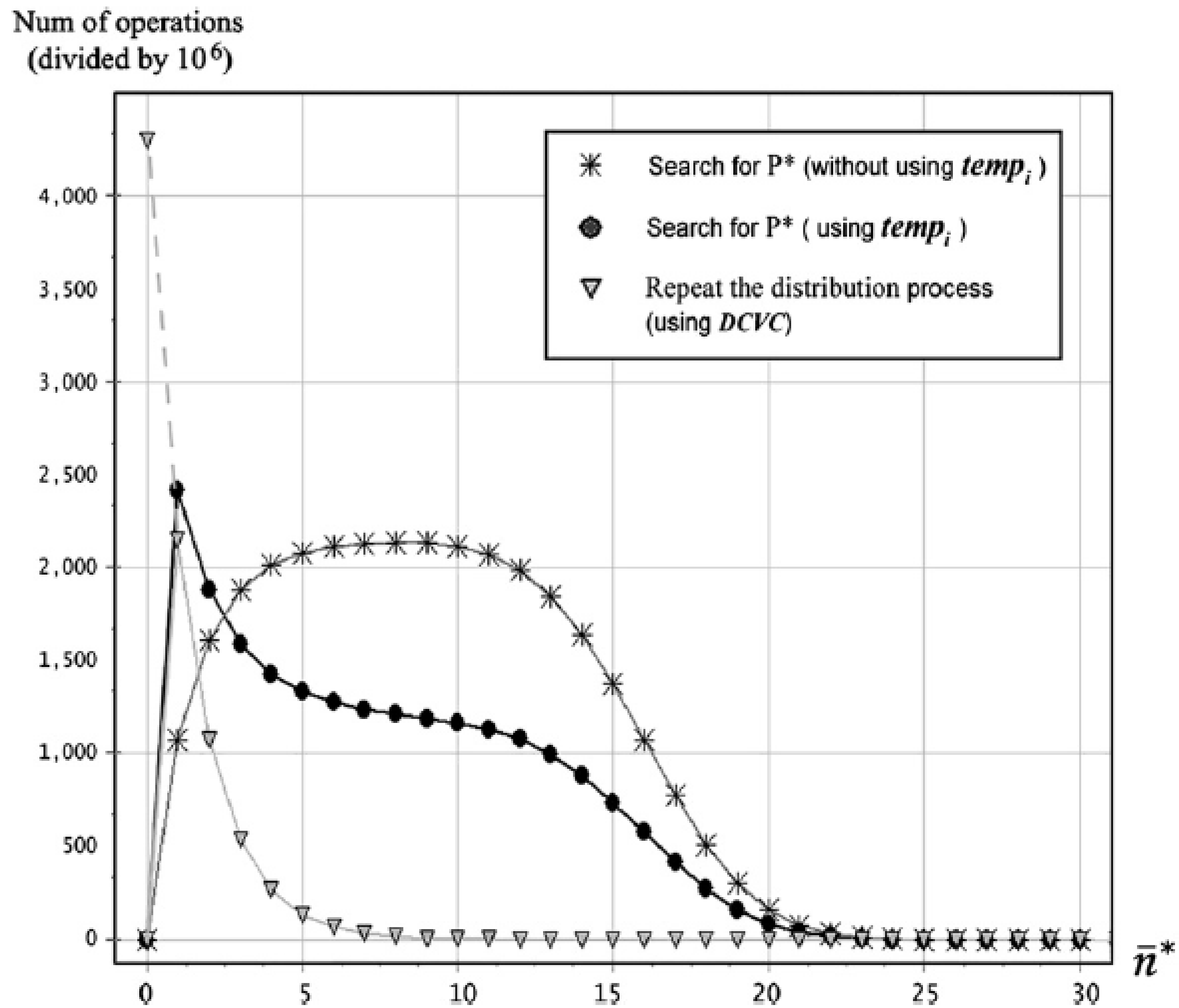
- Certain cases agents can’t join a certain coalition
  - Coalitions can’t overlap;
  - Resources needed for a coalition.

- 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?