OPTIMIZED MULTI-CRITERIA DECISION ANALYSIS THROUGH MEDIAN RANKING AND THE ANALYTICAL HIERARCHY PROCESS

Department of Computer Science and Engineering,
University of Ioannina, Greece, October 2023

Presented by: **Nefeli Eleftheria Sextou**

Advisor: **Konstantinos Parsopoulos**

INTRODUCTION

INTRODUCTION

“Median Ranking Problem” <=> “Consensus Ranking Problem” <=> “Social Choice Problem” <=> “Kemeny Aggregation Problem”

<u>Preference Ranking:</u> represented by a ranking vector ex: <1,3,4,2> -> idx = 1 , rank = 1 ; idx = 2 , rank = 3 ...

- Describes an individual’s preferences over a set of alternative choices
- *Full/Complete* Rankings, *Tied/Weak* Rankings, *Incomplete* Rankings, *Partial* Rankings

m *judges* provide m rankings for ***n*** *items* → ***m*** ranking *vectors* of *size* ***n***

<u>Median Ranking:</u> the ranking that best represents the rankings given by a set of m judges

“ the most agreeable ranking ”

❖ Trying to find it, is a very old problem, found in a wide array of fields (computer science, social sciences, biology, operations research and more). Some application examples include:
  - Decision making within important governmental departments (Canadian Dept.Of Defense, Emond & Mason)
  - Aggregating genomics research result lists
  - Combatting spam in search engines (main idea: human evaluators -> Search Engine -> web page rankings)

❖ Different Algorithmic Solutions Under Various Frameworks: QUICK,FAST, Branch And Bound and **<u>DECoR (Kemeny Axiomatic Framework)</u>**

INTRODUCTION

Proposal :

- The use of another metaheuristic under **Kemeny's axiomatic framework** , as presented in the paper introducing DECoR (A differential evolution algorithm for finding the median ranking under the Kemeny axiomatic approach, by A. D'Ambrosio, G. Mazzeo , C. Iorio, R. Siciliano)
  - i.e. the application of **a modified version of Particle Swarm Optimization** equipped with a **restarting strategy** and its comparison with a proposed variant of DECoR that exploits the same restarting strategy

- The rank aggregation problem is closely tied to **decision making procedures**
  - it is also proposed that the input rankings are the result of an *indirect criteria-based methodology* that employs the **Analytic Hierarchy Process**.

- The benchmarking problem is inspired by the financial industry

PROBLEM DESCRIPTION

KEMENY'S AXIOMATIC FRAMEWORK

Two very significant contributions to the way the median ranking problem is perceived today were made by:

- *Kendall in 1938 :* He proposed a correlation coefficient , ***Kendall's Tau (τ),*** which provides a measure of comparison between two different rankings, regardless of whether the objective order for the ranked items has been given in the form of one of the two rankings

- *Kemeny And Snell in 1962:* They proposed a distance based axiomatic approach. They introduced a distance metric, the ***Kemeny Distance,*** and proposed a set of **axioms**.

Kemeny Distance:

$$d(A,B) = \frac{1}{2}\sum_{i,j=1}^{n} |a_{ij} - b_{ij}|$$

a_{ij} and b_{ij} represent elements of the *score matrices* A and B

Kendall's original formulation:

- If the object at index ***i*** is ranked **ahead** of the object at index ***j***, then $\boldsymbol{a_{ij} = 1}$
- If the object at index ***i*** is ranked **behind** the object at index ***j***, then $\boldsymbol{a_{ij} = -1}$
- If the objects are **tied**, then $\boldsymbol{a_{ij} = 0}$

Example 2.1 : If the rank vector $\langle 3 \quad 1 \quad 2 \quad 4 \rangle$ represents the ranking of preference of the objects O_1,O_2,O_3 and O_4 which are considered to correspond to a fixed position at indexes 1,2,3 and 4 respectively; the score matrix for this ranking would be:

$$\begin{bmatrix} 0 & -1 & -1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & -1 & 0 & 1 \\ -1 & -1 & -1 & 0 \end{bmatrix}$$

KEMENY'S AXIOMATIC FRAMEWORK

The Kemeny and Snell Axioms:

Axiom 1: *the distance measure must be a metric, i.e the following three conditions must hold:*

- $d(A, B) \geqslant 0$ *with the equality holding if and only if* A *is the same ranking as* B
- $d(A, B) = d(B, A)$
- $d(A, B) + d(B, C) \geqslant d(A, C)$ *with the equality holding if and only if* B *is a ranking between rankings* A *and* C

Axiom 2: *If* A' *is the result of a permutation of* A *and* B' *the result of the same permutation of* B, *then* $d(A', B')$ *must be equal to* $d(A, B)$.

Axiom 3: *If* A *and* B *agree for all elements apart from a subset of* κ *elements, which is a subset of both rankings, then the distance* $d(A, B)$ *may be calculated as if the only objects that need to be ranked is that subset of elements for which there is a disagreement.*

Axiom 4: *The minimum positive value of the distance measure is 1.*

SEARCH SPACE

The problem may now be defined as:

"The Kemeny Aggregation Problem" : a consensus ranking minimizes the sum of the Kemeny distance between itself and all other rankings, each of which has been provided by a different judge out of a set of judges

Search Space Definition:

- When dealing with **complete rankings**, the **Kemeny distance is equal to the Kendall distance**
- This allows the search space to be defined as a **generalized permutation polytope**, also known as a **permutohedron**, which is a convex hull of finite points in **n-dimensional space**
- This holds because the Kemeny distance is a naturally defined distance in the permutohedron under the condition that ties are allowed
- **A median ranking is an optimum within this space**

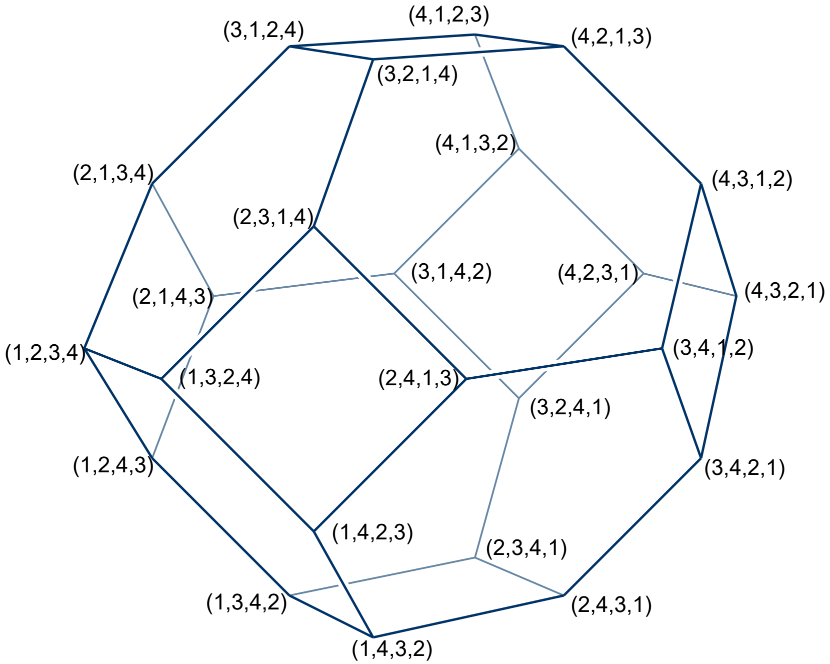
Search Space Size:

- If the universe is that of **complete** rankings, then the search space size is: $Z^n = n!$
- If the universe is that of **weak** rankings, then the search space size is: $S^n = \Sigma_{r=0}^{n} r! \left\{ {n \atop r} \right\}$ $\quad Z^n \subset S^n$

n : the number of ranked objects

$\left\{ {n \atop r} \right\}$: is the Stirling number of a second kind, which indicates the number of ways a set of ***n*** objects may be partitioned in ***r*** non-empty subsets

SEARCH SPACE



- The search space grows significantly for the case of S^n and the complexity of the problem is high and fully dependent on the number of ranked objects $\boldsymbol{n}$.
- If **ties are allowed**, then it is definite that the search space is $\boldsymbol{S^n}$.
- If **ties** are **not allowed**, and there is **no** explicit **specification**, **it is not definite that the search space is Z^n**, and it must considered to be $\boldsymbol{S^n}$.
- The problem is NP-hard

Ali and Meila benchmarked a total of 104 different algorithms and their combinations on the problem, and concluded that it is intractable, while **it is impossible to avoid trading-off solution quality for computational time and vice versa** (A. Ali and M. Meilă, “Experiments with Kemeny ranking: What works when?”)

Image Source: https://en.wikipedia.org/wiki/Permutohedron#/media/File:Permutohedron.svg

CORRELATION COEFFICIENT τ_x

- τ_b is a variation of Kendall's tau intended to deal with ties: $$\tau_b(A,B) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} b_{ij}}{\sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}^2 \sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij}^2}}$$
- Emond and Mason argued that τ_b has the following two problems:

  1. The resulting τ_b for the all-ties ranking with any other ranking containing ties, produces division by zero. This cannot be bypassed using a convention of considering that division equal to zero, since it would result in the all-ties ranking having zero correlation with all other rankings as well as itself

  2. The second issue they highlighted was the fact that there is at least one case for which τ_b ceases to be a metric since it violates Axiom 1 of the Kemeny-Snell Axioms. More precisely, they provided a counter-example for which τ_b violates the triangle inequality.

To overcome these issues, they proposed their own correlation coefficient , τ_x : $$\tau_x(A,B) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} b_{ij}}{n(n-1)}$$

It extends Kendall's Tau and **follows a different score matrix formulation rule**:

- If the object at index ***i*** is ranked ahead of the object at index ***j*** or they are tied, then $a_{ij} = 1$
- otherwise $a_{ij} = -1$

Furthermore, they proved that ***Kemeny Distance* is equivalent to τ_x** : $$\tau_x(A,B) = 1 - 2\frac{d(A,B)}{n(n-1)}$$

THE MEDIAN RANKING PROBLEM AND THE COST FUNCTION

Kemeny And Snell: a median ranking is one that minimizes the sum of the distances (or the mean of distances) between itself and all other *m* rankings.

"Kemeny Aggregation Problem"

Let $A_1, .. A_m$ be the set of rankings provided by m judges.
The median ranking is:

$$\hat{Y} = \arg\min_{Y \in S^n} \sum_{k=1}^{m} d(A_k, Y)$$

Emond And Mason: a median ranking is one that maximizes the sum of correlation coefficients (or the mean of the correlation coefficients) between itself and all other *m* rankings

Let $A_1, .. A_m$ be the set of rankings provided by m judges.
The median ranking is:

$$\begin{aligned}\hat{Y} &= \arg\max_{Y \in S^n} \frac{\Sigma_{k=1}^{m} w_k (\Sigma_{i,j=1}^{n} a_{ij}^{(k)} y_{ij})}{n(n-1)\Sigma_{k=1}^{m} wk} \\ &= \arg\max_{Y \in S^n} \sum_{i,j=1}^{n} c_{ij} y_{ij} \\ &= \arg\max_{Y \in S^n} \tau_x(C, Y)\end{aligned}$$

It takes advantage of the **τ_x – *Kemeny Distance* equivalence**

THE MEDIAN RANKING PROBLEM AND THE COST FUNCTION

w_k is the weight of the k-th ranking that quantifies its importance compared to the rest

C is the **combined input matrix** , each element is defined as $c_{ij} = \sum_{k=1}^{m} w_k a_{ij}^{(k)}$ Score matrix of **k-**th ranking out of ***m***

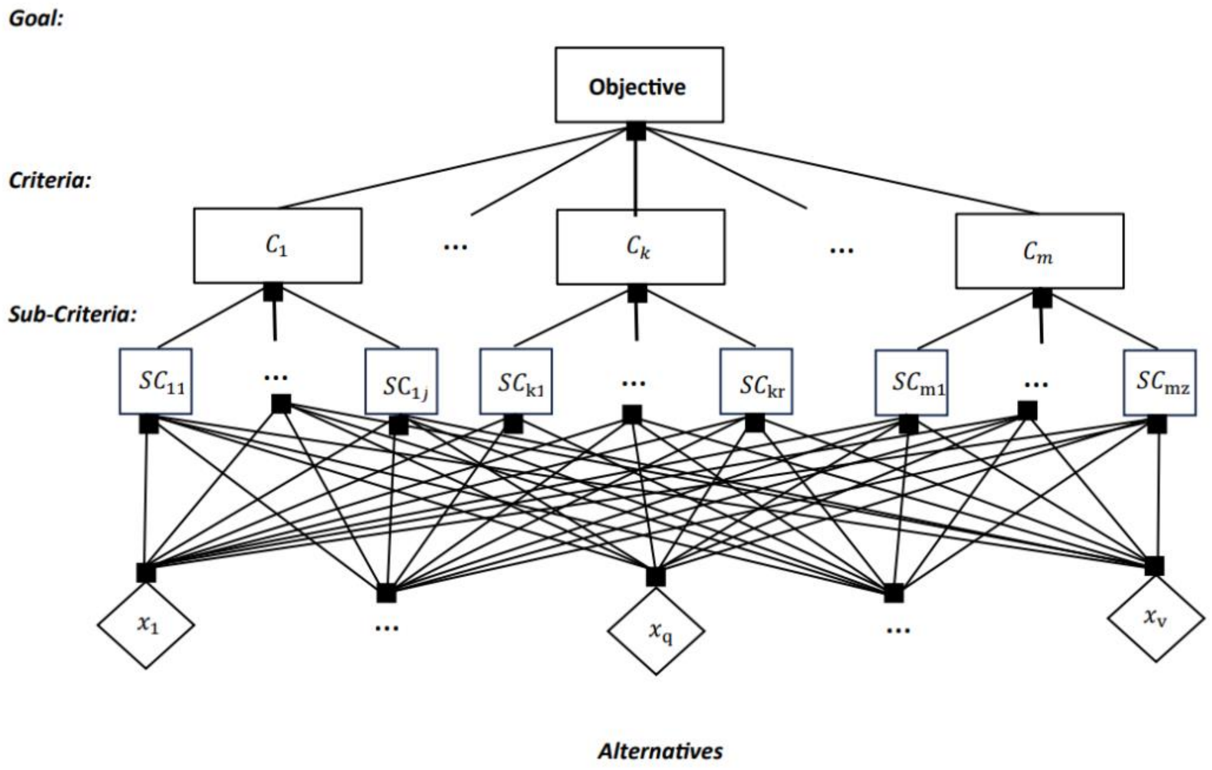
In order to apply DE on the median ranking problem, considering Emond and Mason's expression of the problem and their proof of the *Kemeny Distance* equivalent , the following cost function was introduced:

$$\boxed{cost(S) = \sum_{k=1}^{m} w_k \frac{n(n-1)}{2}(1 - \tau_x(C, S))}$$

Both PSO and DECoR try to minimize this cost function.

THE ANALYTICAL HIERARCHY PROCESS (AHP)

Analytic Hierarchy Process (AHP) : is a problem-solving framework developed by T.L Saaty, which supports breaking up a larger problem into smaller parts in an organized and structured way while also incorporating judgments derived by humans on the basis of their knowledge, experience, intuition and goals. It is a multiobjective and multi-criteria decision making methodology that employs the concept of hierarchy to simplify and structure a problem, and a pairwise comparison scheme to derive a scale of preference for the available alternative.



THE ANALYTICAL HIERARCHY PROCESS (AHP)

Alternatives and criteria are compared in a pairwise fashion, this is represented through Pairwise Comparison Matrices which have the following form:

$$A = (a_{ij})_{n\times n} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} = \begin{bmatrix} \frac{w_1}{w_1} & \frac{w_1}{w2} & \dots & \frac{w_1}{w_n} \\ \frac{w_2}{w_1} & \frac{w_2}{w_2} & \dots & \frac{w_2}{w_n} \\ \dots & \dots & \dots & \dots \\ \frac{w_n}{w_1} & \frac{w_n}{w_2} & \dots & \frac{w_n}{w_n} \end{bmatrix} = \begin{bmatrix} 1 & a_{12} & \dots & a_{1n} \\ \frac{1}{a_{12}} & 1 & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ \frac{1}{a_{1n}} & \frac{1}{a_{2n}} & \dots & 1 \end{bmatrix}$$

$$a_{ij} > 0,\ \forall i,j \qquad a_{ij} \approx \frac{w_i}{w_j},\ \forall i,j \qquad a_{ij} = \frac{1}{a_{ji}}\ \forall i,j$$

- If the set of criteria is $C = \{c_1, \dots, c_m\}$, then **for each criterion** there should exist **a pairwise comparison matrix** $A^{(cm)}$. This signifies that the evaluator is comparing alternatives, considering each criterion independently.
- **Another pairwise comparison** will be necessary to compare the **different criteria amongst each other**, with regard to their **importance to the objective** stated at the top of the hierarchy
- Each pairwise comparison matrix is used to extract a preference weight vector called a **priority vector**
- According to the Eigenvector method, the priority vector is the **principal eigenvector** of a pairwise comparison matrix

THE ANALYTICAL HIERARCHY PROCESS (AHP)

To obtain the principal eigenvector , we must consider:

$$A^{(c_k)}w = \begin{bmatrix} \frac{w_1}{w_1} & \frac{w_1}{w2} & \cdots & \frac{w_1}{w_n} \\ \frac{w_2}{w_1} & \frac{w_2}{w_2} & \cdots & \frac{w_2}{w_n} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{w_n}{w_1} & \frac{w_n}{w_2} & \cdots & \frac{w_n}{w_n} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \cdots \\ w_n \end{bmatrix} = \begin{bmatrix} nw_1 \\ nw_2 \\ \cdots \\ nw_n \end{bmatrix} = nw$$

- It is known that if n is an eigenvalue of $A^{(ck)}$, the other eigenvalue is 0.
- Given the multiplicity of A is $n(n-1)$, it is derived that $\boldsymbol{n}$ is in fact **the largest eigenvalue of $\boldsymbol{A^{(ck)}}$**.
- According to Saaty, this can be extended to be the maximum eigenvalue of the matrix $\boldsymbol{\lambda_{max}}$
- **$\boldsymbol{w}$ is this eigenvalue's corresponding eigenvector** and it may be obtained by solving the linear system:

$$\begin{cases} A^{(c_k)}w = \lambda_{\max} w \\ w^\top (1,\ldots,1)^\top = (1,\ldots,1)^\top \end{cases}$$

The priority vectors are synthesized to create the final preference scale vector using the *weighted arithmetic mean.*

Example 2.2 : For a goal G , criteria $\{C_1, C_2, C_3\}$ and alternatives $\{x_1, x_2, x_3, x_4\}$, the pairwise comparison matrices are the 4×4 matrices : A^1,A^2,A^3. From each of these matrices, the priority vectors extracted using the eigenvector method are w_1, w_2 and w_3. The pairwise comparison matrix that results from the comparison of the criteria amongst each other with regard to the goal G is A^c, and its corresponding priority vector is w^c. The final vector is the result of the linear combination:

$$w^{final} = w^c_{11} w_1 + w^c_{21} w_2 + w^c_{31} w_3$$

THE ANALYTICAL HIERARCHY PROCESS (AHP)

- In real decision-making environments, only an approximation of the weight ratios are given.
- The judgments are relative and not exact, often construed in a verbal manner.
- This is why Saaty had also proposed a scale with verbal descriptions that would facilitate the comparison of a small number of alternatives
- The pairwise comparison matrices must also be ***consistent.***
  - $A^{(ck)}$ is consistent **if and only if** the condition $a_{ik} = a_{ij}a_{jk} \; \forall \, i, j, k$ holds
  - This protects the decision maker from giving contradicting judgments.
  - Saaty proposed a **Consistency Index CI** that utilizes the fact that if a matrix is consistent then its maximum eigenvalue is equal to n.

$$CI = \frac{\lambda_{max} - n}{n - 1}$$

- $CI > 0.1$ *must hold*

Integer Value	Verbal Description
1	Equal preference regarding the two alternatives
2	Intermediate option between 1 and 3
3	Moderate preference of one alternative over the other
4	Intermediate option between 3 and 5
5	Strong preference of one alternative over the other
6	Intermediate option between 5 and 7
7	Very Strong preference of one alternative over the other
8	Intermediate option between 7 and 8
9	Extreme preference of one alternative over the other

EMPLOYED METAHEURISTICS

REQUIRED MODIFICATIONS FOR CONSENSUS RANKING

Both Differential Evolution (DE) and Particle Swarm Optimization (PSO) were developed for **continuous problems**.
The consensus ranking problem is a permutation-based problem in discrete space.

Two modifications are required to apply DE and PSO on this problem:

1. The first is the use of the **cost function** as the objective function
2. The second adaptation is **discretization**. This is necessary because the **mutation and crossover** operations in DE (similarly, the **position and velocity updates** in PSO) **may result in out-of-bound, repeated or non-integer values**. There are two approaches to achieve that:
   - **a.** **Closest Integer :** the closest integer value is retained for each real-valued component of a candidate solution vector
   - **b.** **Hierarchical Approach:** the values of the solution receive the ordering rank that corresponds to them within the solution

The Hierarchical Approach is preferred here, because it neither results in out-of-bound values, nor does it require any other repair operation. ***" sortAndRank" -> produces full rankings***

Example 3.1: If $w^{final} = (0.456, 0.567, 0.116, 0.113)^\top$, the resulting ranking vector (assuming a tie breaking ranking procedure took place and that the best alternative receives the value of 1) is $r = (2, 1, 3, 4)^\top$ and the best alternative is x_2.

DIFFERENTIAL EVOLUTION FOR CONSENSUS RANKING

- DE was developed by Storn And Price (1995)

- It belongs to the extensive class of Evolutionary Algorithms, which are inspired by evolutionary processes observed in nature. Just like its counterparts, it relies on the concept of environmental pressure triggering natural selection inspired mechanisms, resulting in populations of higher fitness levels.

- DECoR is a variant of DE specifically adapted for the consensus ranking problem developed by A. D'Ambrosio, G. Mazzeo , C. Iorio, R. Siciliano (2017) [DECoR = Differential Evolution For Consensus Ranking]

Initialization: The initial population is a set of P random candidate solutions which must take the form of full rankings. This is achieved through the discretization routine *sortAndRank.*

Mutation: For each individual in the population $x_{i,G}$, where $i = \{0, 1, \dots, P\}$, a new vector $v_{i,G+1}$ is generated in accordance to one of several known mutation strategies.

The chosen strategy for DECoR is : $/DE/rand/1/bin: \; v_{i,G+1} = x_{r_1,G} + F(x_{r_2,G} - x_{r_3,G})$

$$r_1 \neq r_2 \neq r_3 \neq i \quad r_1, r_2, r_3 \in [0, P] \quad F > 0 \quad F \in (0, 2]$$

DIFFERENTIAL EVOLUTION FOR CONSENSUS RANKING

- The chosen strategy has been proven to work best for most problems, including the median ranking problem.
- r_1, r_2 and r_3 represent random individual indices picked from the current population
- F is an amplification factor (step size) and is one of the main parameters of the algorithm and requires proper tuning

Mutation Strategy	Formula
$DE/rand/1/bin\ or\ /ex$	$v_{i,G+1} = x_{r_1,G} + F(x_{r_2,G} - x_{r_3,G})$
$DE/best/1/bin\ or\ /ex$	$v_{i,G+1} = x_{best,G} + F(x_{r_2,G} - x_{r_3,G})$
$DE/rand/2/bin\ or\ /ex$	$v_{i,G+1} = x_{r_1,G} + F(x_{r_2,G} + x_{r_3,G} - x_{r_4,G} - x_{r_5,G})$
$DE/rand-to-best/2/bin\ or\ /ex$	$v_{i,G+1} = x_{best,G} + F(x_{r_1,G} + x_{r_2,G} - x_{r_3,G} - x_{r_4,G})$
$DE/x/1/bin\ or\ /ex$	$v_{i,G+1} = x_{r_1,G} + F_1(x_{r_2,G} - x_{r_3,G}) + F_2(x_{best,G} - x_{r_1,G})$

Crossover: can be **exponential (/ex)** or **binomial (/bin)**. The crossover strategy for DECoR is **/bin**.

$$u_{ij,G+1} = \begin{cases} v_{ij,G} & if\ r > CR \quad r:\ a\,random\,number,\ r \in [0,1] \\ x_{ij,G} & otherwise \end{cases}$$

$CR \in [0,1]$: *Crossover Ratio*

- *CR* controls the strength of a mutation
- It requires tuning

Discretization : is applied on the offspring resulting from crossover operation.

Selection : the child candidate solution $v_{i,G+1}$ is compared to the original candidate solution on the basis of their cost function values. Since the cost function needs to be **minimized**, the individual **retained** is the one achieving the **smallest cost function value**.

DIFFERENTIAL EVOLUTION FOR CONSENSUS RANKING

Stopping Criterion: a limit is enforced on the total number of generations the population is allowed to evolve for. So, the stop criterion is $\boxed{while\ gen \leq\ Budget}$

Restarting Strategy: if the number of **non-improving** iterations reaches a **percentage of** the overall ***Budget***, then the population is re-initialized and the percentage is doubled

- Up to 4 restarts may occur

- The first restart takes place if the number of no-improvements becomes equal to 5 percent of the overall *Budget*.

- Then, the no-improvements counter is set to zero so that the next restart will take place if the number of no-improvements now reaches a number equal to 10 percent of the Budget, and so on until the final restart, which may take the 40 percent mark.

- If these restarts happened within the least temporal distance between them, in direct succession of each other, for a setting of 40 percent in the condition, at least 75 percent of the budget would have been spent in a no-improvements condition.

The purpose of this adaptive restarting strategy is to alleviate stagnation and getting trapped around local minima, as well as the opportunity to reinvigorate the search.

DIFFERENTIAL EVOLUTION FOR CONSENSUS RANKING

Algorithm 3.1 DECoR Variant

```
Require: P,F,CR,Budget
pop = initilizePopulation(P)
prevBestData[2] = getBestValueAndIndex(pop)
globalBestData[2] = prevBestData[2]
gen = 1; noImprovement = 0; restartCounter = 0; lastHit = 0
restartIntervals = {0.05, 0.1, 0.2, 0.4, 0.8}
gen ≤ Budget
if noImprovement = restartIntervals[restartCounter] · Budget then
    pop = initilizePopulation(P)
    prevBestData[2] = getBestValueAndIndex(pop)
    noImprovement = 0
    restartCounter ++
end if
for i  in  [0, P − 1] do
    evo = mutation(pop[i], F)
    evo = crossover(evo, pop[i], CR)
    rankedEvo = sortAndRank(evo)
    if cost(rankedEvo) < cost(pop[i]) then
        pop[i] = evo
    end if
    currBestData[2] = getBestValueAndIndex(pop)
end for
globalBestData[2] = getGlobalBestDataAndIndex(currBestData[2], globalBestData[2])
if prevBestData[0] = currBestData[0] then
    noImprovement ++
else
    noImprovement = 0
end if
prevBestData[2] = currBestData[2]
gen ++
```

PARTICLE SWARM OPTIMIZATION FOR CONSENSUS RANKING

- PSO was developed by J. Kennedy (1995)

- It belongs to the class of swarm intelligence algorithms. So far it counts numerous variants and a vast number of applications. Its operation is based on the coordinated movement of a swarm of candidate solutions towards more prospective areas of the search space by taking advantage of both individual and collective information.

- The PSO variant presented here exploits the same adaptations as DECoR with the purpose of examining its successful application on the consensus ranking problem. The stopping criterion and restarting strategy are also the same.

Swarm Of Particles : a set S, of N candidate solutions. Each candidate solution x_i is called a **particle**. It is initialized randomly.

$$S = \{x_1, x_2, \dots, x_N\} \qquad x_i = (x_{i1}, x_{i2}, \dots, x_{in})^\top$$

Velocities: each particle x_i, has a velocity v_i, which it used to update its position.

$$v_i = (v_{i1}, v_{i2}, \dots, v_{in})^\top$$

PARTICLE SWARM OPTIMIZATION FOR CONSENSUS RANKING

Neighborhood: Each particle also has a set of other particles with which it exchanges information called its neighborhood, and has a size *q*.

$$NB_i = (x_{z_1}, x_{z_2}, ..., x_{z_q})^\top \qquad \{z_1, z_2, ..., z_q\} \subseteq \{1, 2, .., N\}$$

Fully Connected Topology (gbest): the neighborhood includes all other particles in the swarm $NB_i = S$

Ring Topology (lbest): $NB_i = \{x_{i-r}, x_{i-r+1}, ..., x_{i-1}, x_i, x_{i+1}, ..., x_{i+r-1}, x_{i+r}\}$

- ***r*** is the neighborhood radius, and it defines the neighborhood size
- **It requires a tuning phase**

The neighborhood is used to locate the neighbor with the smallest objective function evaluation: $p_{g_i} = \underset{j \, such \, that \, x_j \in NB_i}{\arg\min} f(p_j)$

g_i is its index

Best Positions: As each particle is updated across iterations, it retains the best position it has reached as: $p_i = (p_{i1}, p_{i2}, ..., p_{in})^\top$

The best position is the one where the smallest objective value has been achieved:

$$p_i^{G+1} = \begin{cases} x_i & if \ f(x_i^{G+1}) < f(p_i^G) \\ p_i & otherwise \end{cases}$$

$i = 1, ..., N \qquad G$ is the generation counter

PARTICLE SWARM OPTIMIZATION FOR CONSENSUS RANKING

Velocity And Position Update Equations:

$$v_{ij}^{(G+1)} = \chi[v_{ij}^{(G)} + r_1 c_1 (p_{ij}^{(G)} - x_{ij}^{(G)}) + r_2 c_2 (p_{g_i j}^{(G)} - x_{ij}^{(G)})]$$

$$x_{ij}^{(G+1)} = x_{ij}^{(G)} + v_{ij}^{(G+1)}$$

G $(or\ g)$: generation counter
g_i : index of the best neighboring particle
r_1, r_2: random numbers, uniformly distributed in [0,1]
c_1 : cognitive parameter
c_2 : the social parameter
χ : constriction coefficient.

$$i = 1, 2, ..., N\ ,\ j = 1, 2, ..., n \qquad c_1 = c_2 = 2.05 \text{ and } \chi = 0.729$$

Velocity And Position Clamping: Velocity updates may result in swarm explosion while position updates may result in values outside the defined search space. In these scenarios, clamping must be applied before the updates values are used for any purpose.

If the search space is:

$$A = [l_1, u_1] \times [l_2, u_2] \times ... \times [l_n, u_n]$$

$$i, j = 1, 2, ..., n$$

$$x_{ij}^{G+1} = \begin{cases} l_i & if \quad x_{ij}^{G+1} < l_i \\ u_i & if \quad x_{ij}^{G+1} > u_i \\ x_{ij}^{G+1} & otherwise \end{cases}$$

If the search space is considered to be: $A = [0,1]^n$

$$x_{ij}^{g+1} = \begin{cases} 0 & if \quad x_{ij}^{g+1} < 0 \\ 1 & if \quad x_{ij}^{g+1} > 1 \\ x_{ij}^{g+1} & otherwise \end{cases}$$

Maximum absolute value per velocity component: $v_{max} = \dfrac{u_i - l_i}{k} = \alpha \cdot (u_i - l_i)$

-> defines the velocity boundary

Common initial (uninformed) choice: **a=0.5**

- Affects swarm convergence speed
- **Requires tuning**

$$A = [l_1, u_1] \times [l_2, u_2] \times ... \times [l_n, u_n]$$

$$v_{ij}^{g+1} = \begin{cases} -v_{max} & if \quad v_{ij}^{g+1} < -v_{max} \\ v_{max} & if \quad v_{ij}^{g+1} > v_{max} \\ v_{ij}^{g+1} & otherwise \end{cases}$$

$$A = [0,1]^n$$

$$v_{ij}^{g+1} = \begin{cases} -\alpha & if \quad v_{ij}^{g+1} < -\alpha \\ \alpha & if \quad v_{ij}^{g+1} > \alpha \\ v_{ij}^{g+1} & otherwise \end{cases}$$

PARTICLE SWARM OPTIMIZATION FOR CONSENSUS RANKING

Algorithm 3.2 PSO for Consensus Ranking: Part 1/2

Require: $S, \chi, c_1, c_2, \alpha, r, Budget$

$swarm = initilizeSwarm(S)$

$prevBestData[2] = getBestValueAndIndex(swarm)$

$globalBestData[2] = prevBestData[2]$

$bestPostitions = swarm$

$velocities = initVelocities(\alpha)$

$gen = 1; noImprovement = 0; restartCounter = 0$

$restartIntervals = \{0.05, 0.1, 0.2, 0.4, 0.8\}$

while $gen \leq Budget$ **do**

    if $noImprovement = restartIntervals[restartCounter] \cdot Budget$ **then**

        $swarm = initilizePopulation(S)$

        $prevBestData[2] = getBestValueAndIndex(pop)$

        $bestPostitions = swarm$

        $velocities = initVelocities(\alpha)$

        $noImprovement = 0$

        $restartCounter{+}{+}$

    end if

    for $i \quad in \quad [0, S-1]$ **do**

        $g_i = getNBbestIndex(r, swarm[i], swarm)$

        for $ij \quad in \quad [0, n-1]$ **do**

            $v_{ij}^{gen+1} = updateVelocity(velocities, swarm, bestPositions, i, j, g_i, \chi, c_1, c_2)$

            $v_{ij}^{gen+1} = velocityClamping(\alpha)$

            $x_{ij}^{gen+1} = updatePosition(swarm[i][j], v_{ij}^{gen+1})$

            $x_{ij}^{gen+1} = positionClamping([0, 1])$

            $velocities[i][j] = v_{ij}^{gen+1}$

            $swarm[i][j] = x_{ij}^{gen+1}$

        end for

PSO For Consensus Ranking: Part 2/2

        $rankedEvo = sortAndRank(swarm[i])$

        if $cost(swarm[i]) < cost(bestPositions[i])$ **then**

            $bestPositions[i] = swarm[i]$

        end if

        $currBestData[2] = getBestValueAndIndex(pop)$

    end for

    $globalBestData[2] = getGlobalBestDataAndIndex(currBestData[2], globalBestData[2])$

    if $prevBestData[0] = currBestData[0]$ **then**

        $noImprovement{+}{+}$

    else

        $noImprovement = 0$

    end if

    $prevBestData[2] = currBestData[2]$

    $gen{+}{+}$

end while

EXPERIMENTAL METHODOLOGY

BENCHMARK PROBLEM AND DATASETS

The benchmark problem is inspired by the financial industry.

- A **brokerage firm** is a financial institution that facilitates buying and selling financial assets and instruments on the behalf of clients.
  - **Class 1 (C1): Boutique trading firms** are the smallest ones and provide services for a very particular part of the market.
  - **Class 2 (C2): Discount brokerage firms** are slightly larger and offer cost effective services aimed at self-directed investors.
  - **Class 3(C3): Full-service firms** are much larger and offer many different services to their clients, including research and consultation, usually at higher fees. They also offer online services.

- The investors may offer their preference on the firm's traders according to evaluation criteria related to their financial interests. The **traders** are the employees responsible of executing those trades.
- Collecting those rankings and finding the one that best represents the consensus of clients (investors) would provide valuable insight for a firm.
- In the context of median ranking, the **clients are the judges** and the **traders are the ranked items**
- As the number of traders can get very large, their direct ranking from the investors may be impossible.
- For this reason, the investors only rank their preference criteria, and AHP is then used to provide the complete set of rankings.

BENCHMARK PROBLEM AND DATASETS

Rankings Generation: the number of traders **n** and the number of clients **m** is defined according to

10 scenarios per class

$$n = \mathrm{round}(N + N_s * \mathrm{randn}), \quad m = \mathrm{round}(M + M_s * \mathrm{randn})$$

Parameter	C1	C2	C3
N	10	100	1000
Ns	2	20	250
M	50	1500	50000
Ms	10	250	5000

- N and M are prescribed mean values relevant to the corresponding class
- N_s and M_s are the corresponding standard deviations
- **randn** is a random scalar sampled from the standard normal distribution

Each trader is evaluated on the basis of three performance criteria:

Profitability: the trader's ability to generate profits from trading activities and higher values correspond to a better performance

Return Of Investment (ROI): a measure of the profit or loss generated by a trader and is relative to the capital invested. Again, higher values correspond to better performance.

Maximum Drawdown (MDD): is a measure of risk that is used by traders to evaluate how their strategies perform. It is the maximum percentage loss of their trading account value to their lowest point. Here, smaller values are considered better. It is convenient to use the value of *100 − MDD* instead of MDD. In this case, a higher value would signal better performance, since it now represents how much of the initial value of an investment is retained instead of lost.

A trader's performance in accordance with a criterion is: **bad**, **average** or **good**

Criterion	**Bad** $bound_1$	**Average** $bound_2$	**Good** $bound_3$
PROFITABILITY	10	20	40
ROI	5	20	30
100 - MDD	50	80	90

$$\begin{cases} \mathrm{bad} = bound_1 * \mathrm{randn} \\ \mathrm{average} = bound_1 + \mathrm{randn} * (bound_2 - bound_1) \\ \mathrm{good} = bound_2 + \mathrm{randn} * (bound_3 - bound_2) \end{cases}$$

BENCHMARK PROBLEM AND DATASETS

- A pairwise n × n comparison matrix is created for each criterion.
- The pairwise comparison values are set only for above or below the diagonal, since according to the pairwise comparison matrix definition, if a component a_{ij} receives the value x, then a_{ji} receives the inverse value $\frac{1}{x}$. The value has an *equal probability* of belonging to the bad, average or good range
- The priority vectors are then extracted for each criterion's pairwise comparison matrix.
- It is important to note that this process is done once, and it can be considered as the firm's evaluation of the traders.
- Each client has their own preference over the criteria. Thus, for each client a pairwise comparison matrix describing his preferences is created.
- The priority vector extracted from this matrix is linearly combined with the priority vectors that describe the traders' performance across each of the three criteria to get the final preference vector
- This final vector is translated into a ranking following the hierarchical approach

BENCHMARK PROBLEM AND DATASETS

Investor Weights:

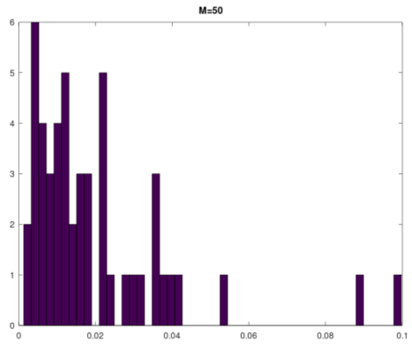
- It is considered that each investor has his own importance for the firm.

- This is quantified through a weight value, which models the firm’s consideration that a client’s significance is relative to his loyalty and amount of invested capital through the firm

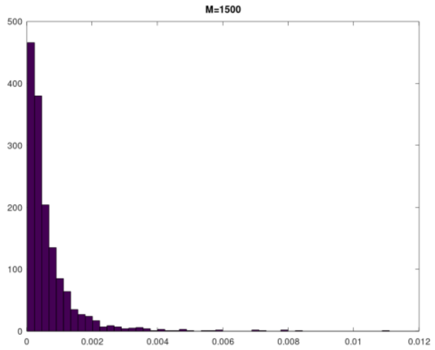
- A normalized log-normal distribution (positively skewed) is used to produce the weights, with the intention of mirroring the phenomenon where a small portion of clients invests large amounts of money, and therefore, they are considered more important compared to the rest

- Pareto’s Principle tends to hold in business contexts, not always in a strict mathematical sense

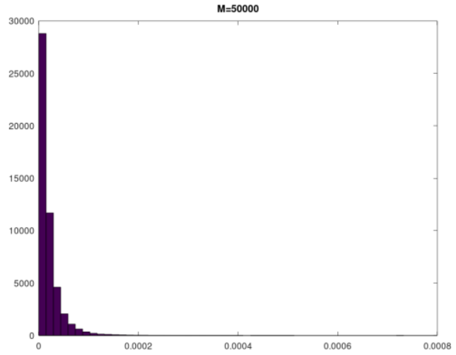
- ~20% of the inputs (clients, investments, etc) would produce ~80% of the outputs (revenue, profit, etc)



(a) $m = M = 50$



(b) $m = M = 1500$



(c) $m = M = 50000$

- *Mean* = log(2000)
- *Standard Deviation* = 1
- *m* = 50, 1500, 50000

METHODOLOGY

Overall Goals:

1. to examine how the DECoR and PSO variants perform on the median ranking problem, both individually, and relative to each other
2. probe the ability of these methods to provide insightful information within decision making contexts akin to the one described in the simulation problem

In order to ascertain a result for the first goal, **two phases of experimentation and analysis need to take place**

Output files
- Median ranking file → Experiment ID, Median Ranking
- **Statistical data file** → Experiment ID, **"Output variables"**

Output variables that are examined :

- ***CostVal*: minimum cost function value found**
- ***LastHit***: holds the value of function evaluations that were spent to find the output minimum
- ***Time***: seconds elapsed for a complete run of the algorithm
- ***Restarts***: how many restarts were triggered during the run

Other output variables produced by the implementation:

- ***experimental ID***: denotes which round of execution for a parameter combination provided that result
- ***seed***: is provided for reproducibility purposes

****CostVal is the primary performance indicator***

METHODOLOGY

Stage 1: Scoping Out Promising Parameterizations

- The first stage of parameter tuning aims to narrow down the parameter search space for each metaheuristic, and choose the four best performing combinations on the basis of the median cost value, which is considered as the primary performance indicator

- DECoR: 18 different parametrizations were tested on the C1 and C2 datasets , for 25 execution rounds each .

- PSO: 8 different parameterizations were tested on the C1 and C2 datasets for 25 execution rounds each

- The four best parametrizations are found through an emperical search rather than statistical testing since there is inadequate data produced and collected during the tuning

Parameter	Values		
P	100	200	-
$Budget$	50000	25000	-
F	0.25	0.5	0.75
CR	0.2	0.5	0.8

Parameter	Values	
S	100	200
$Budget$	50000	25000
r	0	1
α	0.25	0.5

METHODOLOGY

Stage 2: Parameter Tuning

- The metaheuristics are applied for all dataset classes, for all four parameterizations
- This process is relatively fast for the classes C1 and C2, but significantly time consuming for C3
- Each dataset combination is run 25 times
- The results from this thorough tuning phase are statistically analyzed via the production of essential statistical information like the **mean**, **median**, **standard deviation**, **minimum**, and **maximum** values for each important output variable
- This is complemented by **boxplots** that compare the parameterizations amongst each other
- **Statistical testing** is also performed to assess the significance of the observed differences.

METHODOLOGY

Stage 2: Parameter Tuning

*The **Paired Wilcoxon Signed Rank test*** is a non-parametric statistical test that evaluates if there exists a statistically significant difference between two sets of related (paired) data.

- It is used to assess whether or not the parameterizations perform differently.

- If the received p-value is less than the significance level of 0.05, then the null hypothesis is rejected and it may be concluded that those parameterizations are different

Similarly…

The ***One-Tailed Kolmogorov-Smirnov (KS) test*** is a non-parametric test that can be used to determine whether two independent samples are drawn from the same distribution.

METHODOLOGY

The winning parameterization of the modified DECoR is compared with the best parameterization of the PSO variation:

- This is achieved using the one-tailed KS test to ascertain whether or not one performs significantly better than the other.
- Further comparison is conducted by analyzing boxplots, as well as standard statistical data which provide more details and insight (mean, median, standard deviation, minimum, maximum)
- Plots describing their overall performance and value magnitude across all dimensions, and for all output variables, are also used to attain a more complete perspective

METHODOLOGY

Implementation Details:

- All the methods were coded in the C programming language, as single threaded implementations, and utilizing an ANSI C compiler
- The datasets were produced using MATLAB scripts, and the statistical testing along with any data handling and cleaning were achieved using Python Jupyter Notebook scripts and using the Pandas, Numpy, Matplotlib and SciPy libraries
- Experiments were conducted using machines with 8 Intel (R) Core (TM) i7-2600 3.4Ghz CPUs (with each core supporting up to 2 threads) and 8GB of RAM

EXPERIMENTAL RESULTS

STAGE 1 RESULTS: MOST PROMISING PARAMETERIZATIONS

Param. ID	F	CR	*Budget*	*P*
D1	0.25	0.5	50000	100
D2	0.5	0.5	50000	100
D3	0, 75	0.5	50000	100
D4	0.75	0.8	50000	100

DECoR

Param. ID	r	α	*Budget*	*S*
P1	1	0.25	25000	200
P2	1	0.5	25000	200
P3	1	0.25	50000	100
P4	1	0.5	50000	100

PSO

STAGE 2: PARAMETER TUNING (DECOR)

Paired Wilcoxon Signed Rank Test Results:

p-value table

H_0 rejection table

CostVal	D1	D2	D3	D4
D1	-	0.765198	0.390533	0.025094
D2	0.765198	-	0.313463	0.025094
D3	0.390533	0.313463	-	0.025094
D4	0.025094	0.025094	0.025094	-

CostVal	D1	D2	D3	D4
D1	0	0	0	1
D2	0	0	0	1
D3	0	0	0	1
D4	1	1	1	0

LastHit	D1	D2	D3	D4
D1	-	0.080327	0.935399	0.164184
D2	0.080327	-	0.063556	0.745655
D3	0.935399	0.063556	-	0.009932
D4	0.170598	0.745655	0.009932	-

LastHit	D1	D2	D3	D4
D1	0	0	0	0
D2	0	0	0	0
D3	0	0	0	1
D4	0	0	1	0

Time	D1	D3	D3	D4
D1	-	0.855272	0.220647	0.280087
D2	0.855272	-	0.745655	0.027741
D3	0.220647	0.745655	-	0.002367
D4	0.280087	0.027741	0.002367	-

Time	D1	D2	D3	D4
D1	0	0	0	0
D2	0	0	0	1
D3	0	0	0	1
D4	0	1	1	0

Restarts	D1	D2	D3	D4
D1	-	0.317311	-	0.317311
D2	0.317311	-	0.317311	0.563703
D3	-	0.317311	-	0.317311
D4	0.317311	0.563703	0.317311	-

Restarts	D1	D2	D3	D4
D1	0	0	0	0
D2	0	0	0	0
D3	0	0	0	0
D4	0	0	0	0

One-Tailed KS Test Scores:

Variable	D1	D2	D3	D4
Cost Val	9	11	11	33
Last Hit	7	6	6	11
Time	0	0	1	62
Restarts	0	0	0	1

Initial thoughts:

- D4 seems to be the best
- D2 and D3 perform similarly

STAGE 2: PARAMETER TUNING RESULTS (DECOR)

***CostVal* Boxplot And Statistical Table Observations**:

- The winner amongst D4, D2 and D3 is not clear
- D4's performance on C3 biases the KS test score towards a higher value (30 points)
- D4's performance on C:2 N:129 M:1082 gives it another 3 points to reach the total score of 33
- The boxes are significantly shorter for these cases, indicating a narrow range of values around the median
- **D2 and D3 outperform D4 with differences in the range of decimals for C1 and C2**
- **D4 on the other hand outperforms the rest with differences in the hundreds and thousands for C3(dimension higher than 324)**

***LastHit* Boxplot And Statistical Table Observations**:

- No consistent pattern is observed
- **The value increases significantly towards the total limit imposed by the budget (5*10^6) for classes C2 and C3**
- Many outliers are apparent in several cases for C3
- Big difference in terms of range for classes C1 and C2, which is much larger in comparison to C3 (and small corresponding standard deviation)
- This variable is likely governed by random fluctuations, and thus it appears to be an inferior indicator for precise performance evaluation

STAGE 2: PARAMETER TUNING RESULTS (DECOR)

***Time* Boxplot And Statistical Table Observations**:

- There exist outliers with no observable pattern
- Most of the plots for C1 and C2 are positively skewed with medians indicating the majority of values are concentrated towards lower values
- The opposite is apparent for most cases of C3
- Lack of consistency in terms of box length and by result distribution range
- **Agree with KS test scores but is subject to machine architecture type and implementation quality (inadequate performance indicator)**

***Restarts* Statistical Table Observations**:

- The four parameterizations perform similarly across the majority of the datasets
- Match KS test scores
- **the number of restarts falls, as the problem dimension increases**
- not considered a main decision factor

STAGE 2: PARAMETER TUNING RESULTS (DECOR)

Conclusion:

- The main goal is to choose the parameterization that achieves the best minimization result
- The main performance indicator is *CostVal*
- It is definite that D4 has statistically different performance in comparison to the rest (overall)
- The one-tailed KS Test and boxplots provide more details but also lead to a dilemma between choosing D2 and D3 , and D4 -> 2 options
  - **Option 1:** choose D2 and D3 for C1 and C2 datasets, and D4 for C3 datasets
  - **Option 2:** choose D4 because it performs well in accordance with the other variables and consider a trade-off in accuracy for C1 and C2 datasets (since the differences are relatively small)

Option 2 is preferred because it is of interest to make the best overall choice with the data and parameters at hand

STAGE 2: PARAMETER TUNING (PSO)

Paired Wilcoxon Signed Rank Test Results:

Cost Val	P1	P2	P3	P4
P1	-	0.000089	0.067355	0.000089
P2	0.000089	-	0.000089	0.232226
P3	0.067355	0.000089	-	0.000089
P4	0.000089	0.232226	0.000089	-

CostVal	P1	P2	P3	P4
P1	0	1	0	1
P2	1	0	1	0
P3	0	1	0	1
P4	1	0	1	0

LastHit	P1	P2	P3	P4
P1	-	0.244946	0.034537	0.012048
P2	0.244946	-	0.007111	0.006640
P3	0.034537	0.006640	-	0.452164
P4	0.012048	0.006640	0.452164	-

LastHit	P1	P2	P3	P4
P1	0	0	1	1
P2	0	0	1	1
P3	1	1	0	0
P4	1	1	0	0

Time	P1	P2	P3	P4
P1	-	0.776569	0.715133	0.556113
P2	0.776569	-	0.129353	0.427955
P3	0.715133	0.129353	-	0.280087
P4	0.556113	0.427955	0.280087	-

Time	P1	P2	P3	P4
P1	0	0	0	0
P2	0	0	0	0
P3	0	0	0	0
P4	0	0	0	0

Restarts	P1	P2	P3	P4
P1	-	0.317311	0.004678	0.025347
P2	0.317311	-	0.002700	0.014306
P3	0.004678	0.002700	-	0.083265
P4	0.025347	0.014306	0.083265	-

Restarts	P1	P2	P3	P4
P1	0	0	1	1
P2	0	0	1	1
P3	1	1	0	0
P4	1	1	0	0

One-Tailed KS Test Scores:

Variable	P1	P2	P3	P4
Cost Val	39	6	35	1
Last Hit	12	3	31	21
Time	37	34	0	0
Restarts	1	0	21	18

Similarities:

CostVal : P1, **P3**
LastHit, *Restarts* : **P3**, P4

STAGE 2: PARAMETER TUNING RESULTS (PSO)

***CostVal* Boxplot And Statistical Table Observations**:

- The assumption that P1 and P3 perform similarly on this variable is confirmed
- P1 performs slightly better but their differences are in the range of decimals and ones , as confirmed by the corresponding statistical tables
- P1 also has a narrower range of values around its median for many cases
- Additionally, outliers become more prevalent in cases within C3 where the problem dimension is high

***LastHit* Boxplot and Statistical Table Observations**:

- Agree with Wilcoxon and KS test results: P3 and P4 behave alike and P3 performs better overall
- Outliers appear for datasets with a higher dimension and there is no apparent pattern that can be commented upon regarding the range and distribution of values
- **The value increases significantly towards the total limit imposed by the budget for C3 datasets**

STAGE 2: PARAMETER TUNING RESULTS (PSO)

***Time* Boxplot And Statistical Table Observations:**

- There exist outliers with no observable pattern
- Most of the plots are positively skewed with medians indicating the majority of values are concentrated toward lower values
- There is no overall pattern evident regarding the distributions
- The median values across all dataset classes explain the high KS test score for P1 and P2
- Is subject to machine architecture and implementation and therefore not a deciding factor

***Restarts* Statistical Table Observations:**

- P3 and P4 perform best, but only narrowly, since they resulted in the same median values for most datasets, with small standard deviation
- **The number of restarts falls as the problem dimension increases**

STAGE 2: PARAMETER TUNING RESULTS (PSO)

Conclusion:

- The main goal is to choose the parameterization that achieves the best minimization result
- The main performance indicator is *CostVal*
- If the choice was made on that alone, the obvious choice would be P1
- However, P1 performs poorly with regard to *LastHit* and *Restarts*
- **P3** is considered a better option because there is strong evidence supporting it results in good values for those as well

COMPARISON OF THE DISTINGUISHED DECOR AND PSO

Param. ID	**F**	**CR**	$Budget$	P
D4	0.75	0.8	50000	100
Param. ID	r	α	$Budget$	S
P3	1	0.25	50000	100

KS Test Results:

Variable	**D4**	**P3**
Cost Val	20	10
Last Hit	8	19
Time	10	13
Restarts	0	20

It is noteworthy that for both methods, the chosen parameter combinations favor **smaller *population/swarm size*** and a **higher *Budget***

According to the KS test:

- ***D4*** outperforms ***P3*** in terms of the ***CostVal*** output variable
- ***P3*** outperforms ***D4*** with regard to the ***LastHit, Time,*** and ***Restarts variable***
- ***P3's*** win in terms of ***Time*** is narrow

COMPARISON OF THE DISTINGUISHED DECOR AND PSO

***CostVal* Boxplot And Statistical Table Observations**:

- KS test scores are confirmed
- D4 achieves not only smaller values but also more consistent and robust results around the median
- Outliers are more prevalent for both as the dimension increases
- This is further supported by median and standard deviation values in the corresponding statistical tables:
  - The medians are very close to each other, but D4 achieves a smaller one compared to P3, by a difference in the ones range
  - The standard deviation is small for C1 and C2 datasets and increases as the dimension and the outliers increase significantly within the C3 dataset range

***LastHit* Boxplot and Statistical Table Observations**:

- Agree with KS test results: the medians of P3 are smaller than D4 and for most cases P3 also achieves a tight range around them
- Outliers are more frequent on datasets corresponding to higher problem dimensions
- According to the statistical tables, that there are strong differences for the medians, depending on the dataset
- **It must be noted that the dimension for which LastHit starts being very close to the limit of 5*10^6 imposed by the Budget and P/S parameters, is 182**
- This is very interesting given it has been shown in that DECoR can perform well **up to 200 objects**.
- However, it must be noted that the DECoR variant presented here also has a restarting strategy, which was not included in the original formulation

COMPARISON OF THE DISTINGUISHED DECOR AND PSO

***Time* Boxplot And Statistical Table Observations**:

- KS test scores are confirmed, the algorithms behave similarly
- No evident patters
- Fluctuations do exist both in the median and the standard deviation, again without clear correlations
- This result is expected, given that differences illustrated here are most likely the product of machine architecture and implementation related consequences

***Restarts* Statistical Table Observations**:

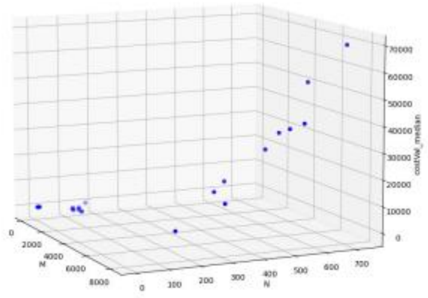
- KS test scores are confirmed
- **P3 achieves more restarts than D4 across the board**
- In most cases, the standard deviation is quite smaller than the mean and median value. This is an indicator that the obtained results are robust

COMPARISON OF THE DISTINGUISHED DECOR AND PSO

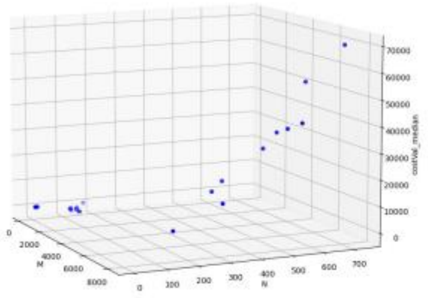
A Highlight Regarding *Restarts* and *LastHit* :

- As the dimension increases, the *Restarts* decrease and the *LastHit* value increases almost to the limit defined by the Budget and P/S parameters.
- This means that for smaller dimensions, all the restart opportunities have been exploited, which also implies that the corresponding method stagnated
- Yet, more restarts provide the ability for better exploration of the search space.
- For C2 datasets, P3 achieved much smaller *LastHit* values and more *Restarts*, while the cost values for the same cases, are very close to those of D4, which is using up all the *Budget* and fewer restarts.
- This could mean that D4 retained different solutions for more generations and didn’t stay on the detecting points long enough to stagnate.
- P3 found the value a bit earlier, and despite the extra restart, was unable to locate a better solution before its budget ran out.
- P3 also does not reach its fourth restart for this class (C2)

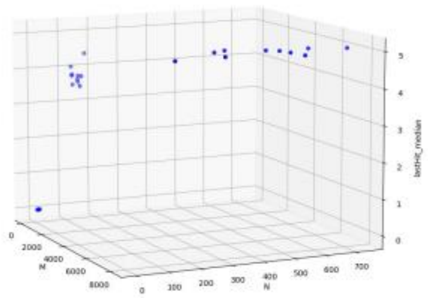
COMPARISON OF THE DISTINGUISHED DECOR AND PSO



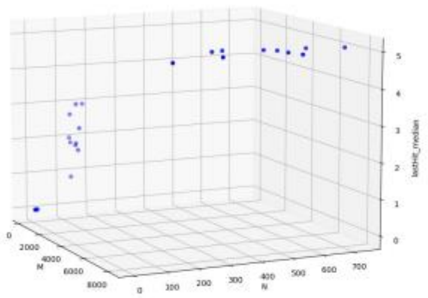
(a) *CostVal* Medians (DECoR)



(b) *CostVal* Medians (PSO)

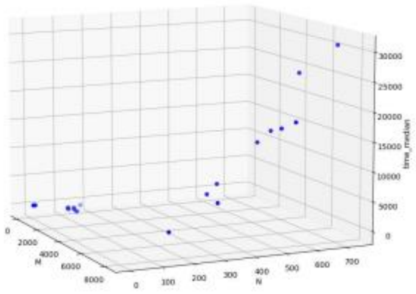


(c) *LastHit* Medians (DECoR)

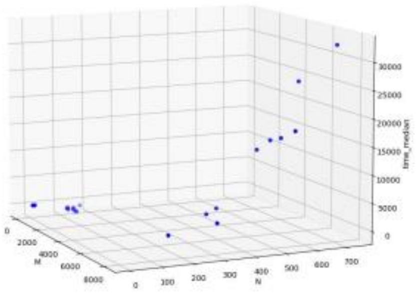


(d) *LastHit* Medians (PSO)

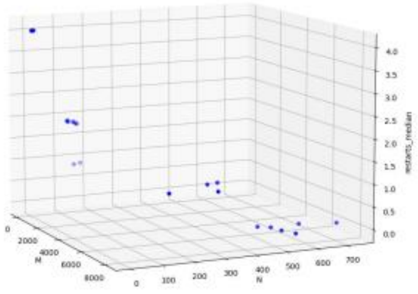
COMPARISON OF THE DISTINGUISHED DECOR AND PSO



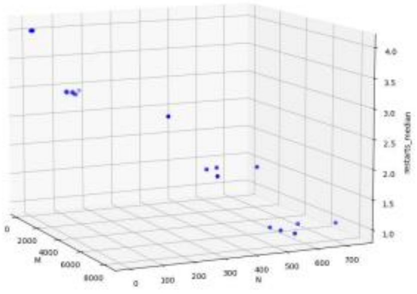
(a) *Time* Medians (DECoR)



(b) *Time* Medians (PSO)



(c) *Restarts* Medians (DECoR)



(d) *Restarts* Medians (PSO)

CONCLUSION

CONCLUSION

A small recap:

- Two population-based metaheuristics were adapted and applied on the consensus ranking problem within Kemeny's axiomatic framework

- The methods were applied on a test suite of datasets of increasing dimension, simulating a multi-criteria decision analysis problem inspired by the financial industry. Each class of datasets represents a different size of brokerage firm, and the traders of a firm are ranked combining their performance evaluations and the criteria preferences of the firm's clients using the AHP

- After an extensive preprocessing phase for parameter tuning, the distinguished variants were compared.

CONCLUSION

Conclusions regarding the comparison of the two metaheuristics:

- ❖ The specialized DECoR variant outperforms the PSO variant in terms of cost value.
- ❖ The PSO variant appears to find similar results for up to around 100 objects, although quite faster, as it achieved with a significantly smaller last hit.
- ❖ DECoR has been proven to be accurate despite the fact it cannot guarantee the detection of all possible solutions. By extension, considering that PSO actually provides very similar results, it may be assumed that the modified **PSO for the median ranking problem may share DECoR's properties and can be considered as a viable alternative**
- ❖ This is supported **only** for the case of **complete rankings**, due to the nature of the simulation datasets
- ❖ There is enough evidence encourage to further research on the application of PSO on partial and weak rankings

CONCLUSION

Conclusions regarding the practical implications of the study:

- ❖ It is reasonable to accept that for up to 100 objects, applying any of the two metaheuristics would provide a median ranking(s) that would be very useful in a real world decision-making environment

- ❖ It merits underlining, that the form of the studied simulation problem is sufficiently generic and may be employed as a problem-solving framework that combines the AHP, the median ranking problem and two state-of-the-art population metaheuristics

- ❖ Furthermore, metaheuristics do not guarantee the discovery of global optima for intractable problems, but they are often able to provide a good enough solution in reasonable time that may be critical for the operation of a firm

THANK YOU