Algorithmic Methods for Mathematical Models
COURSE PROJECT

Facultat d'Informàtica de Barcelona
Master in Innovation and Research in Informatics

- Can Beydogan - can.beydogan@estudiantat.upc.edu
- Sefa Böyükdikmen - sefa.boyukdikmen@estudiantat.upc.edu

- Fall 24
- Barcelona, December 10, 2024

Contents

Problem Statement : Inputs & Outputs

- Inputs
  - D is the number of departments in the faculty.
  - n[1..D] is the array of size D, department p has exactly a number n[p] of participants in the commission.
  - N is the total number of faculty members.
  - d[1..N] the department of i will be denoted by d[i].
  - m[1..N][1..N] is a N x N matrix where each element m[i][j] is the compatibility between members i and j. The diagonal of the compatibility matrix consists of 1's.
- Outputs
  - The solution (if one exists) with optimal objective value.
  - The selection of faculty members, represented by the array x[1..N],

$$x[i] = \begin{cases} 1 & \text{indicates that faculty member } i \text{ is selected for the commission,} \\ 0 & \text{indicates that faculty member } i \text{ is not selected.} \end{cases}$$

Integer Linear Programming Model

- Objective Function

$$\text{maximize}\left(\frac{1}{\text{numberOfPairs}}\sum_{i=1}^{N}\sum_{j=i+1}^{N} m[i][j]\cdot x[i]\cdot x[j]\right)$$

- Constraints
  - Department Participation

$$\sum_{\substack{i=1\\ d[i]=p}}^{N} x[i] = n[p], \quad \forall p \in \{1,\ldots,D\}, \quad i \in \mathbb{Z}$$

  - Zero Compatibility

$$x[i] + x[j] \leq 1, \quad \forall i,j \in \{1,\ldots,N\} \text{ such that } m[i][j] = 0$$

  - Conflict Mediation

$$x[i] + x[j] \leq 1 + \sum_{\substack{k=1\\ m[i][k]>0.85,\, m[k][j]>0.85}}^{N} x[k], \quad i,j,k \in \{1,\ldots,N\}$$

Heuristic algorithms
Greedy Constructive Algorithm

Algorithm 3 Greedy Construction Algorithm

Input: D, n, N, d, m
Output: partial_solution

partial_solution $\leftarrow \emptyset$
dep_participantN$[d] \leftarrow 0, \forall d \in \{1, \ldots, D\}$
compatibilities $\leftarrow$ sort$(N, \sum m[i][j]$, DESC$)$
for member in compatibilities **do**
    department $\leftarrow d$[member]
    if dep_participantN[department] $<$ n[department - 1] **then**
        if $\forall$other $\in$ partial_solution, m[member][other] > 0 and $\forall$other $\in$ partial_solution, (m[member][other] ≥ 0.15 or $\exists k \in$ partial_solution, m[member][k] $> 0.85 \wedge m[k]$[other] > 0.85) **then**
            partial_solution $\leftarrow$ partial_solution $\cup$ {member}
            dep_participantN[department] $\leftarrow$ dep_participantN[department]+1
    if $|$ partial_solution $| = \sum n$ **then**
        break
if $|$ partial_solution $| < \sum n$ **then return** INFEASIBLE
return partial_solution

Heuristic algorithms
Greedy Constructive and Local Search Procedure

Algorithm 4 Local Search Algorithm

Input: D, n, N, d, m, solution

Output: best_solution, best_objective

```
best_solution ← solution
best_objective ← calculate_objective(m, best_solution)
is_improved ← True
while is_improved do
    is_improved ← False
    for each i in best_solution do
        current_member ← best_solution[i]
        current_department ← d[current_member]
        for each j in {0, ..., N − 1} do
            if j in best_solution or d[j] ≠ current_department then
                continue
            new_solution ← best_solution
            new_solution[i] ← j
            if is_feasible(D, n, N, d, m, new_solution) then
                new_objective ← calculate_objective(m, new_solution)
                if new_objective > best_objective then
                    best_solution ← new_solution
                    best_objective ← new_objective
                    is_improved ← True
                    break
return best_solution, best_objective
```

Heuristic algorithms
GRASP

Algorithm 5 GRASP Main Function

Input: $D, n, N, d, m,$ iterations, α
Output: best_solution, best_objective

$best\_solution \leftarrow$ None
$best\_objective \leftarrow -\infty$
for $k = 1$ to iterations **do**
    $initial\_solution \leftarrow$ GreedyConstructionGRASP($D, n, N, d, m, 0$ **if** $i = 1$ **else** $alpha$))
    if $initial\_solution =$ INFEASIBLE **then**
        continue
    $(local\_best\_sol, local\_best\_obj) \leftarrow$ LocalSearch($D, n, N, d, m, initial\_sol$)
    if $local\_best\_obj > best\_objective$ **then**
        $best\_sol \leftarrow local\_best\_sol$
        $best\_obj \leftarrow local\_best\_obj$
return $(best\_solution, best\_objective)$

Algorithm 6 Greedy Construction with GRASP

Input: D, n, N, d, m, α
Output: partial_solution or **None**

partial_solution $\leftarrow \emptyset$
dep_participantN$[d] \leftarrow 0, \forall d \in \{1, \ldots, D\}$
compatibilities$[i] \leftarrow \sum m[i], \forall i \in \{1, \ldots, N\}$
candidates $\leftarrow$ sort(N, compatibilities$[i]$, DESC)
while | partial_solution $| <$ n **and** candidates $\neq \emptyset$ **do**
    max_compatibility $\leftarrow$ compatibilities[candidates[0]]
    min_compatibility $\leftarrow$ compatibilities[candidates[-1]]
    RCL $\leftarrow$ {member $\in$ candidates | compatibilities[member] $\geq$ max_compatibility $- \alpha \cdot$ (max_compatibility $-$ min_compatibility)}
    selected_member $\leftarrow$ random.choice(RCL)
    dept $\leftarrow d$[selected_member]
    if dep_participantN[dept] $< n$[dept $- 1$] **then**
        if $\forall$other $\in$ partial_solution, m[member][other] > 0 and $\forall$other $\in$ partial_solution, (m[member][other] ≥ 0.15 or $\exists k \in$ partial_solution, m[member][k] $> 0.85 \wedge m[k]$[other] > 0.85) **then**
            partial_solution $\leftarrow$ partial_solution $\cup$ {member}
            dep_participantN[dept] $\leftarrow$ dep_participantN[dept] $+ 1$
    candidates $\leftarrow$ candidates $-$ {selected_member}
if | partial_solution $| < \sum n$ **then**
    return INFEASIBLE
else
    return partial_solution

Tuning the alpha parameter

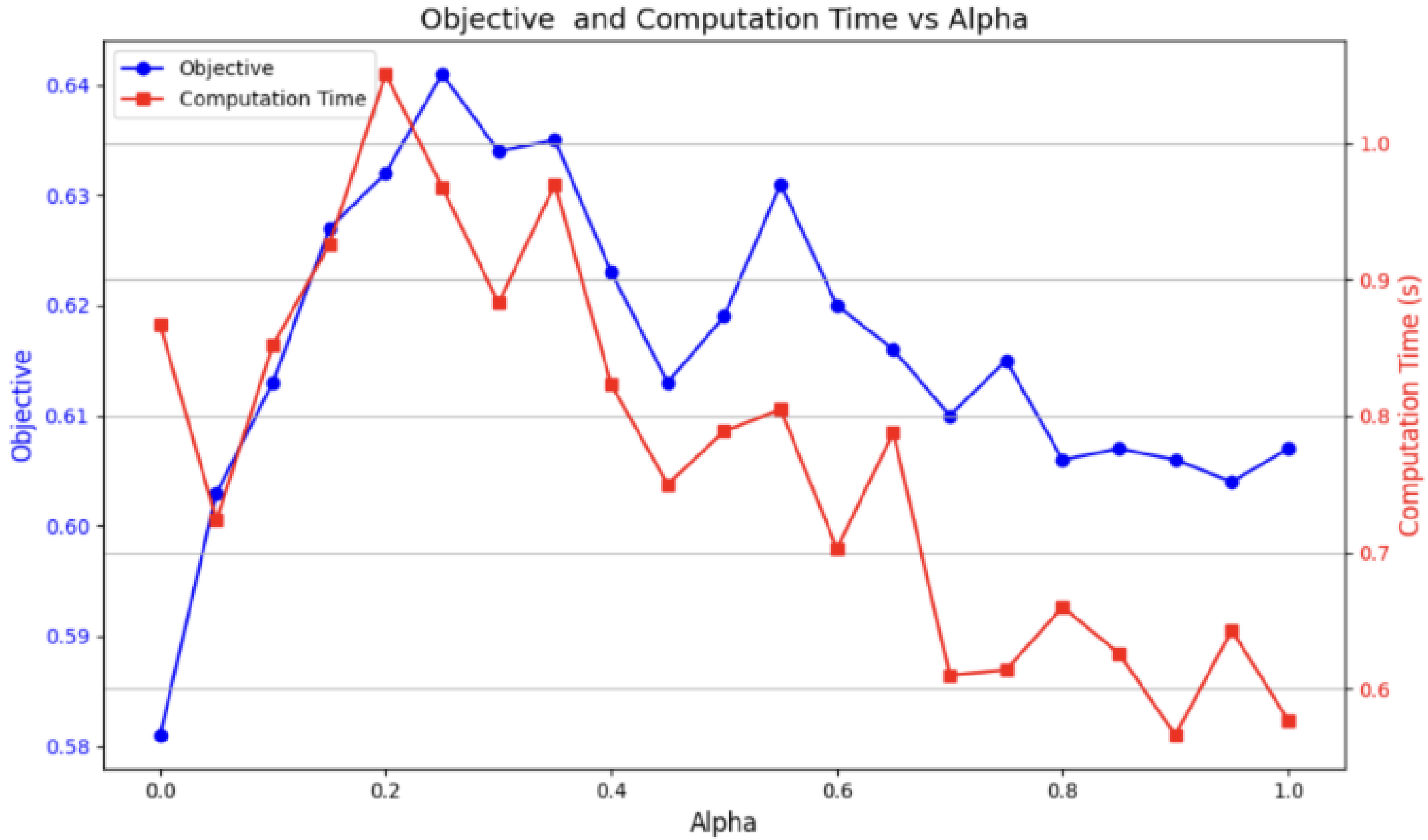

Figure 1: Objectibe and Computation Time vs Alpha

Performance