

AMMM Final Project

Xin Tong and Qiuchi Chen

Universitat Politècnica de Catalunya
Facultat d'Informàtica de Barcelona

Advisors:

Prof. Enric Rodriguez and Prof. Luis Velasco

May 27, 2025



Table of Contents

1. Problem Identification

2. ILP Model

3. Greedy Method

4. Local Search Method

5. GRASP Method

6. Experiment Result

Topological Sort (Kahn's algorithm)

Cyclic graphs don't have valid topological orderings.

ILP Model

Decision Variables

$Priority_{ij}$: Binary $\forall i, j \in Members$

$Order_n$: Integer $\forall n \in Members$

Objective Function

maximize $\sum_{i,j \in Members} Priority_{ij} \cdot m_{ij}$

Each node appears before all the nodes it points to

$Priority_{ij} = 1 \rightarrow Order_i < Order_j$

$Priority_{ij} = 0 \rightarrow Order_j < Order_i$

Constraints

$Priority_{ij} + Priority_{ji} = 1$

$\forall i, j \in Members \text{ and } m_{ij} + m_{ji} \neq 0 \quad (1)$

$Order_i + 1 \leq Order_j + (1 - Priority_{ij}) \cdot N$

$\forall i, j \in Members \text{ and } m_{ij} + m_{ji} \neq 0 \quad (2)$

Pseudo Code & Greedy Cost Function

Greedy Cost Function

$$q(<i,j>, S) = \begin{cases} -1 & \text{if } G(S, <i,j>) \text{ is not a directed acyclic graph} \\ Bids_{ij} & \text{if } G(S, <i,j>) \text{ is a directed acyclic graph} \end{cases}$$

Algorithm Greedy Method

```
Solution ← φ
Candidates ← {<i, j> | i, j ∈ Members and Bidsij + Bidsji ≠ 0}
while Candidates ≠ φ do
    Evaluate q(<i', j'>, Solution) ∀ <i', j'> ∈ Candidates
    <i, j> ← argmax{q(<i', j'>, Solution) | <i', j'> ∈ Candidates}
    Candidates ← Candidates \ {<i, j>, <j, i>}
    Solution ← Solution ∪ {<i, j>}
return Solution
```

Greedy Procedure (data=project.4.dat)

Local Search Method

Target Edges

$$\{<i,j> | <j,i> \in \text{Solution} \text{ and } \text{Bids}_{ij} > \text{Bids}_{ji}\}$$

Neighborhood

Solutions obtained by swapping target edges not included in the solution with their flipped counterparts that are included in the solution.

Knockon Flipped Edges Cost Function

$$q(< i', j' >, \text{edges}^{\text{flip}}) =$$

$$\begin{cases} -1 & \text{if } \text{Bids}_{i'j'} - \text{Bids}_{j'i'} + \sum_{<i'',j''> \in \text{edges}^{\text{flip}}} \text{Bids}_{i''j''} - \text{Bids}_{j''i''} \geq \text{Bids}_{ij} - \text{Bids}_{ji} \\ & \text{or } \text{edges}^{\text{residual}} \mid \text{edges}^{\text{flip}} = \text{edges}^{\text{residual}} \mid \text{edges}^{\text{flip}} \cup \{<i', j'>\} \\ \sum_{k' \in \{i', j'\}} \text{indegree}(k') + \text{outdegree}(k') & \text{if } \text{Bids}_{i'j'} - \text{Bids}_{j'i'} + \sum_{<i'',j''> \in \text{edges}^{\text{flip}}} \text{Bids}_{i''j''} - \text{Bids}_{j''i''} < \text{Bids}_{ij} - \text{Bids}_{ji} \\ & \text{and } \text{edges}^{\text{residual}} \mid \text{edges}^{\text{flip}} > \text{edges}^{\text{residual}} \mid \text{edges}^{\text{flip}} \cup \{<i', j'>\} \end{cases}$$

Knockon Flipped Edges Selection Process

Pseudo Code

Algorithm Local Search Method

```

 $solution^{current} \leftarrow doGreedyConstructionPhase()$ 
improved  $\leftarrow$  True
while improved = True do
    improved  $\leftarrow$  False
    for each  $< i, j > \in \{< i, j > | < j, i > \in solution^{current} \text{ and } bids_{ij} > bids_{ji}\}$  do
        edgesresidual  $\leftarrow doTopologicalSort()$ 
        edgescandidate, edgesflip  $\leftarrow edges^{residual}, \phi$ 
        while edgesresidual  $\neq \phi$  do
            Evaluate  $q(< i', j' >, edges^{flip}) \quad \forall < i', j' > \in edges^{candidate}$ 
             $< i'', j'' > \leftarrow argmax\{q(< i', j' >, edges^{flip}) | < i', j' > \in edges^{candidate}\}$ 
            edgesresidual  $\leftarrow doTopologicalSort()$ 
            edgescandidate  $\leftarrow edges^{residual}$ 
            edgesflip  $\leftarrow edges^{flip} \cup \{< i'', j'' >\}$ 
        solutionflip  $\leftarrow solution^{current}$ 
        Swap  $< j', i' >$  with  $< i', j' >$  in solutionflip  $\quad \forall < i', j' > \in edges^{flip}$ 
        if  $G(solution^{flip})$  is a DAG then
            improved  $\leftarrow$  True
            solutioncurrent  $\leftarrow solution^{flip}$ 
    return solutioncurrent

```

Local Search Procedure (data=project.4.dat)

Pseudo Code & RCL

Algorithm GRASP Constructive Phase

```

solution  $\leftarrow \phi$ 
edgescandidate  $\leftarrow \{<i, j> | i, j \in Members \text{ and } Bids_{ij} + Bids_{ji} \neq 0\}$ 
while edgescandidate  $\neq \phi$  do
    Evaluate  $q(<i', j'>, solution) \forall <i', j'> \in edges^{candidate}$ 
    edgescandidate  $\leftarrow \{<i', j'> \in edges^{candidate} | q(<i', j'>, solution) > 0\}$ 
     $q^{min} \leftarrow \min\{q(<i', j'>, solution) | <i', j'> \in edges^{candidate}\}$ 
     $q^{max} \leftarrow \max\{q(<i', j'>, solution) | <i', j'> \in edges^{candidate}\}$  ▷ RCL Procedure
    RCLmax  $\leftarrow \{<i', j'> \in edges^{candidate} | q(<i', j'>, solution) \geq q^{max} - \alpha(q^{max} - q^{min})\}$ 
     $< i, j > \leftarrow \text{select } < i', j'> \in \text{RCL at random}$ 
    edgescandidate  $\leftarrow edges^{candidate} \setminus \{<i, j>, <j, i>\}$ 
    solution  $\leftarrow solution \cup \{<i, j>\}$ 
return solution

```

Algorithm GRASP Procedure

```

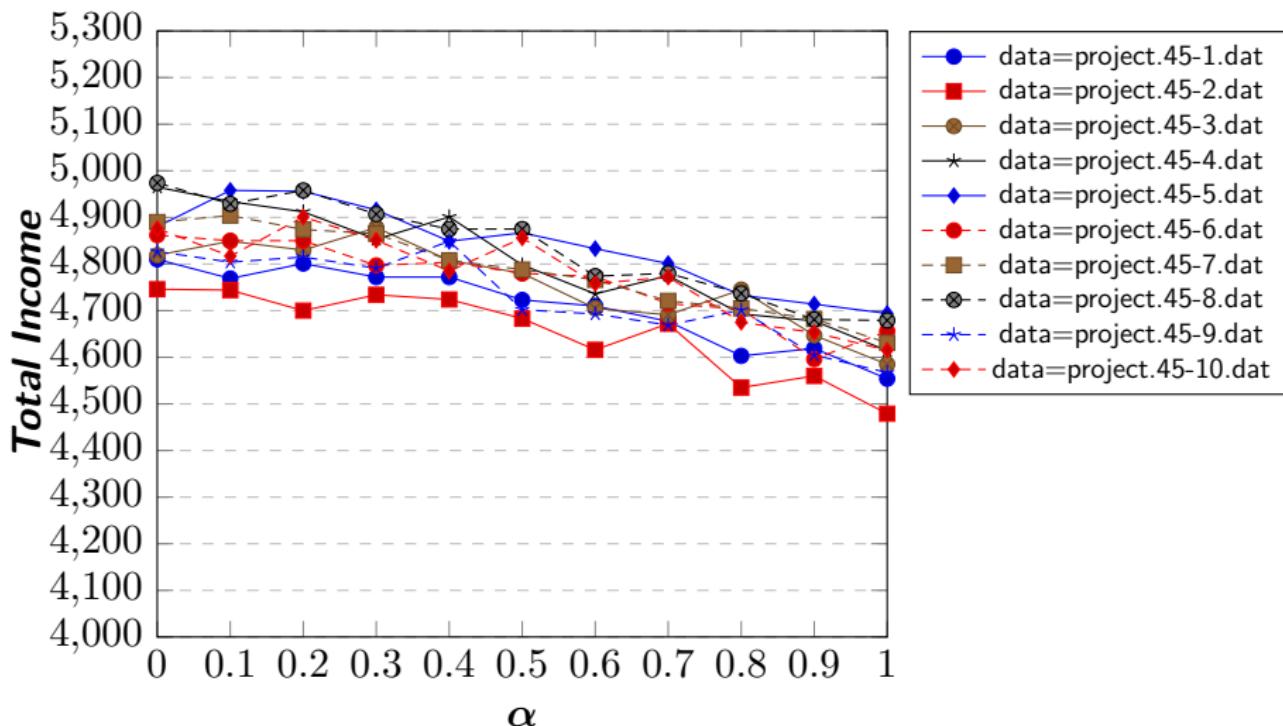
objectbest, solutionbest  $\leftarrow 0, \phi$ 
for retry = 1 to MaxIterations do
    obejective, solution  $\leftarrow 0, \phi$ 
    solution  $\leftarrow doConstructivePhase()$ 
    solution  $\leftarrow doLocalSearchPhase(solution)$ 
    if obejective  $>$  objectbest then
        objectbest  $\leftarrow obejective$ 
        solutionbest  $\leftarrow solution$ 
return solutionbest

```

GRASP Constructive Phase (data=project.4.dat, $\alpha = 0.2$)

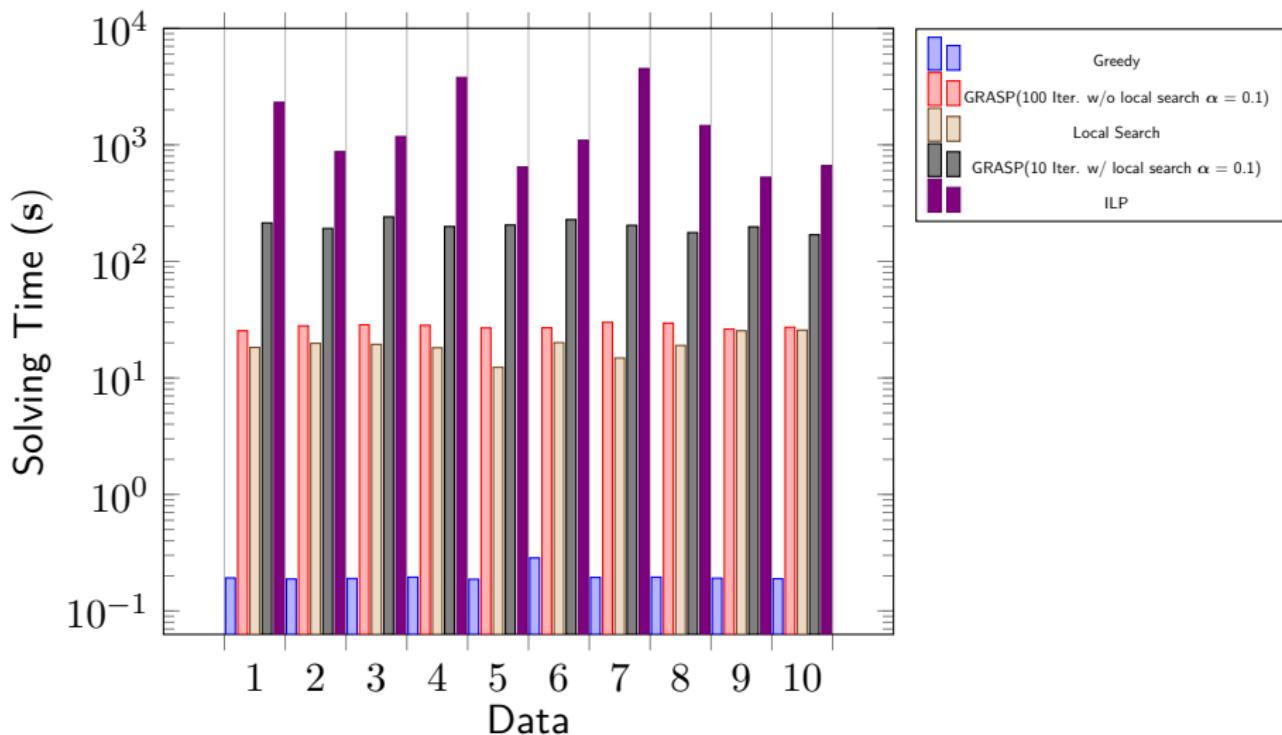
α Tuning for GRASP

Quality of Solutions (100 iter. w/o local search phase)

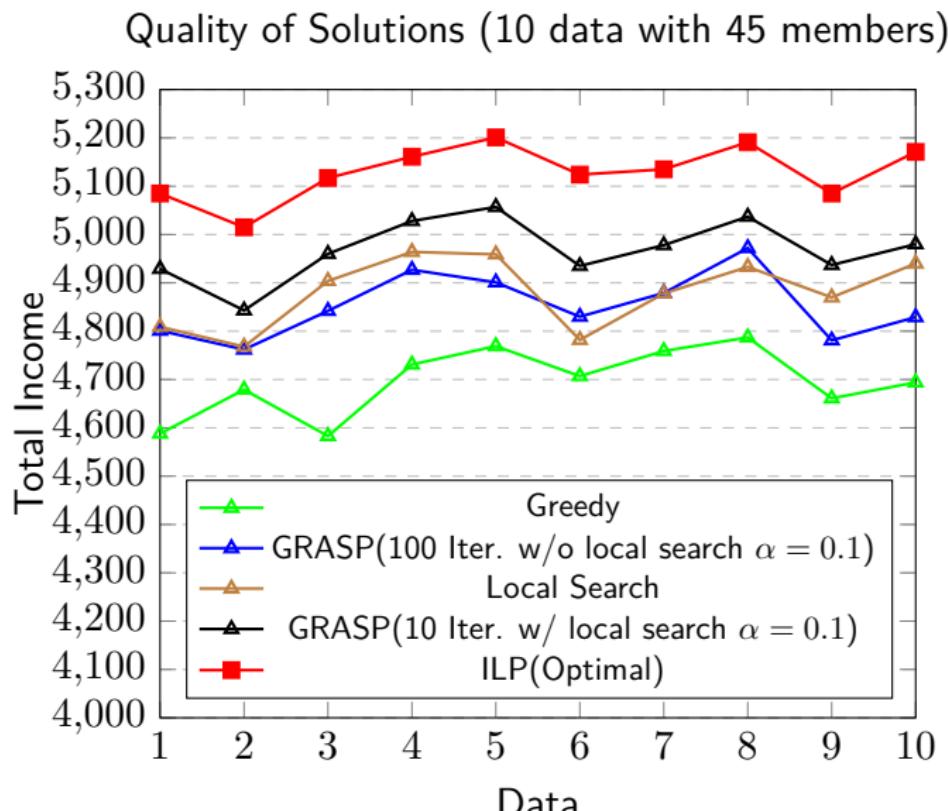


Solving Time

Solving Time (10 data with 45 members)



Solution Quality



Thanks