# Metaheuristic algorithm for timetabling problem

(Seminar III)

Camilo Rodríguez-Garzón
Juan Carlos Rivera

Universidad EAFIT
Mathematical Modeling Research Group

Medellín, Colombia

June 5, 2018

Updated: November 18, 2018

# Overview

- Motivation
- Introduction
- Hard constraints
- Soft constraints
- Methodology
- Problem formulation
- Algorithmic solution
- Computational experiments
- Conclusions

**Motivation:**

- Currently EAFIT does not have an automated system that allows to schedule university timetabling.
- Each school or university has different characteristics that make the problem particular.
- This is a NP-Hard combinatorial optimization problem **?**.
- The algorithm in developing can be applied to different topics that are related to timetabling scheduling.

# Introduction:



Figure 1.1: Timetabling problem

**Hard constraint events:**

A factible solution meets the following set of hard constraints $HC_1 - HC_6$ presented below:

- $HC_1$: Two event can not be scheduled on the same day, period and room.
- $HC_2$: Assigned rooms must meet the characteristics required by the events.
- $HC_3$: The time load defined for each event must be satisfied.
- $HC_4$: Two lessons from the same event must be consecutive when scheduled for the same day, in case it is required by the event.

**Hard constraint teachers:**

- $HC_5$: A teacher can not be scheduled to more than one lesson in a given period.
- $HC_6$: A teacher can not be scheduled to a period in which she/he is unavailable.

**Hard constraint students:**

- $HC_7$: For students with perfect curriculum, at least six events for each course must be feasible.
- $HC_8$: No student can be assigned more than one event at the same time.
- $HC_9$: The number of students attending the event must be less than or equal to the capacity to the classroom.

**Soft constraints:**

In addition to the feasibility with respect to mandatory constraints, the requirements of $SC_1 - SC_4$ listed below should be satisfied as much as possible:

- $SC_1$: Avoid teachers' idle periods.
- $SC_2$: Students should not have a single course on a day.
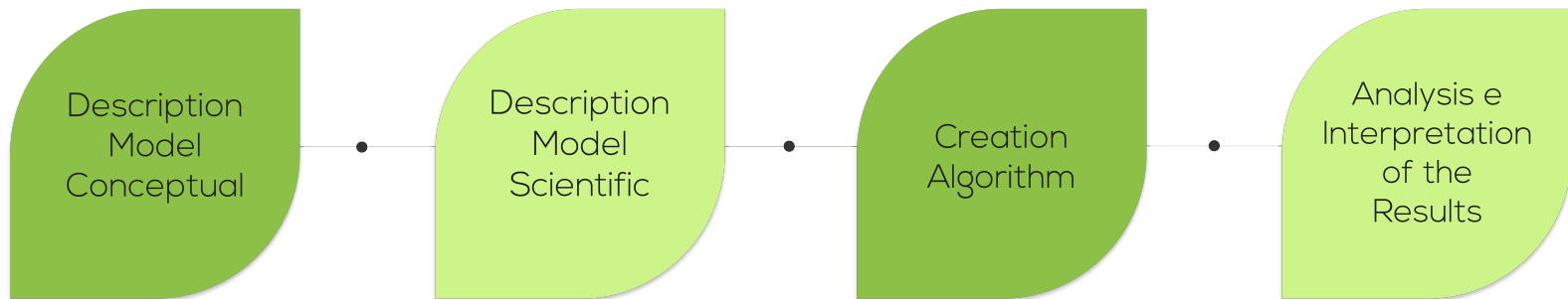- $SC_3$: Students should not have more than two consecutive courses.

# Methodology:



Description Model Conceptual

Description Model Scientific

Creation Algorithm

Analysis e Interpretation of the Results

Figure 1.2: Timetabling problem methodology

# Methodology:
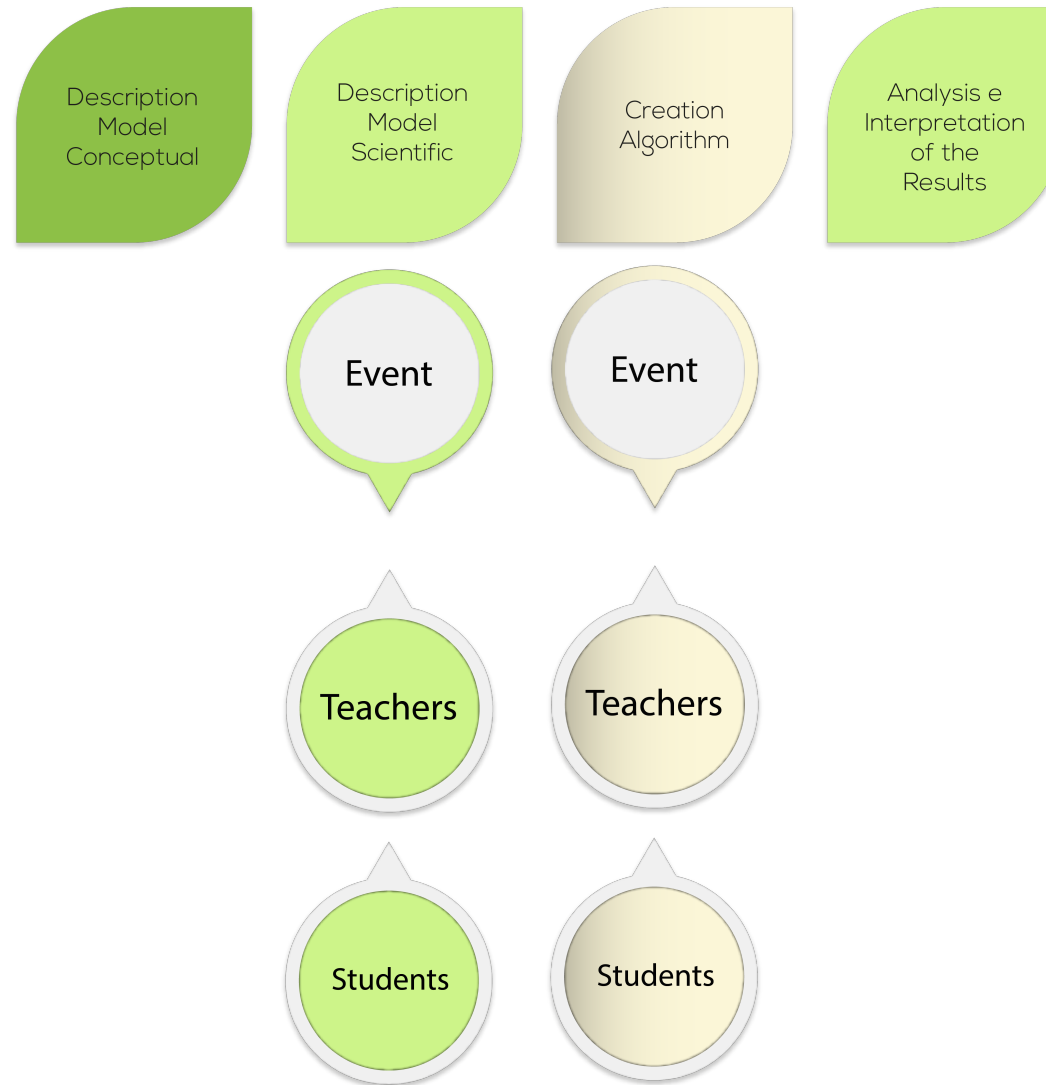


Figure 1.3: Timetabling problem methodology

# Related work:

| Techniques | | Article name |
|---|---|---|
| Graph Coloring (GC) | | An Introduction to TimeTabling **?** |
| Integer/Linear Programming (IP/LP) | | New integer linear programming approaches for course timetabling **?**, Integer programming methods for large-scale practical classroom assignment problems **?** |
| Constraint Satisfaction Programming (CSP) | | Timetable planning using the constraint-based reasoning **?** |
| Multi-population | Genetic Algorithm (GA) | A Utilization-based Genetic Algorithm for Solving the University Timetabling Problem (UGA) **?** |
| | Ant Colony Optimization (ACO) | A MAX - MIN Ant System for the University Course Timetabling Problem **?**, A MAX-MIN Ant System for the University Course Timetabling Problem **?** |
| | Artificial Bee Colony (ABC) | University course timetabling using hybridized artificial bee colony with hill climbing optimizer **?** |
| | Memetic Algorithm (MA) | Using improved memetic algorithm and local search to solve university Course Timetabling problem (UCTP) **?** |
| | Harmony Search Algorithm (HSA) | University course timetabling using a hybrid harmony search metaheuristic algorithm **?** |
| Single-population | Local Search (LS) | A fuzzy genetic algorithm with local search for university course timetabling **?**, Genetic algorithms with guided and local search strategies for university course timetabling **?** |
| | Variable Neighborhood Search (VNS) | An Investigation Of Variable Neighbourhood Search For University Course Timetabling **?** |
| | Simulated Annealing (SA) | Solving the Course Scheduling Problem Using Simulated Annealing **?**, A hybrid simulated annealing with Kempe Chain neighborhood for the university timetabling problem **?** |
| | Tabu Search (TS) | The effect of neighborhood structures on tabu search algorithm in solving course timetabling problem **?**, Design and implementation of a course scheduling system using Tabu Search **?** |
| Novel Intelligent | Hybrid Algorithms (Hybrid Metaheuristic) | A new hybrid algorithm for university course timetabling problem using events based on groupings of students **?** |
| | Fuzzy method | Fuzzy genetic heuristic for university course timetable problem **?**, A fuzzy solution based on Memetic algorithms for timetabling **?** |
| | Clustering Algorithms | Applying a novel clustering technique based on FP- tree to University timetabling problem: A case study **?** |
| Multi-Agent Systems | Review papers Multi-Agent Systems | A multi-agent system for course timetabling **?**, Implementation of class timetabling using multi agents **?** |

Table 1.1: Brief literature about University Timetabling Problem

# Problem formulation:

| Symbols | Definition |
|---|---|
| **Sets** | |
| $s \in S$ | Set of slot composed of days $D$ and periods $P$ |
| $r \in R$ | Set of rooms |
| $e \in E$ | Set of event composed of courses $C$ and groups $G$ |
| **Parameters** | |
| $w_r$ | Unit cost of using the room $r$ |
| $f_e$ | frequency with which an event $e$ is given in the week |
| $dem_e$ | Intensity in hours with which an $e$ event is imparted in a day |
| $cap_s$ | Capacity in hours the slot $s$ has |
| $com_{er}$ | $\begin{cases} 1 & \text{if the event } e \text{ can be assigned to the room } r \\ 0 & \text{Otherwise.} \end{cases}$ |
| $\phi_s$ | Subset of incompatible slots with slot $s$ |
| **Decision variables** | |
| $x_{esr}$ | $\begin{cases} 1 & \text{if the event } e \text{ is programmed to a time slot } s \text{ that is assigned to the room } r \\ 0 & \text{Otherwise.} \end{cases}$ |

Table 1.2: Notation used for modeling events at EAFIT University

# Model:

The objective (**??**) is to minimize the use of artificial salons. The IP restrictions are described below:

$$\min \sum_{e \in E} \sum_{s \in S} \sum_{r \in R} w_r \cdot x_{esr} \tag{1}$$

*s. t.*

$$\sum_{e \in E} x_{esr} \leq 1, \quad \forall\, s \in S, r \in R \tag{2}$$

$$\sum_{r \in R} \sum_{s \in S} x_{esr} = f_e, \quad \forall\, e \in E \tag{3}$$

$$x_{esr} \leq com_{er}, \quad \forall\, e \in E, s \in S, r \in R \tag{4}$$

$$\sum_{e \in E} \sum_{s' \in \phi(s)} x_{es'r} \leq 1, \quad \forall\, r \in R, s \in S \tag{5}$$

$$M \cdot (x_{esr} - 1) \leq -\frac{dem_e}{cap_s} + 1, \quad \forall\, e \in E, s \in S, r \in R \tag{6}$$

**Constraints:**

- **??** - satisfies the restriction $HC_1$.
- **??** - satisfies the restriction $HC_3$.
- **??** - satisfies the restriction $HC_2$.
- **??** - only one event can be assigned to one of the slots of the subset of incompatible slots $\phi_s$.
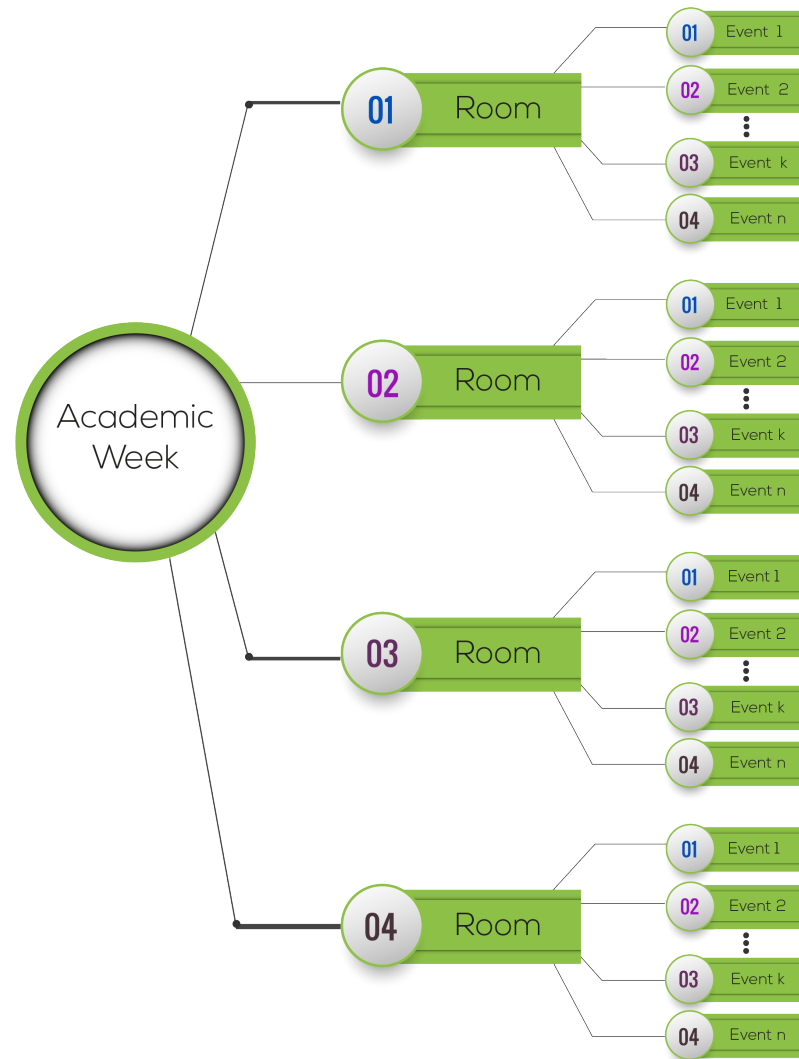- **??** - satisfies the restriction $HC_4$.

# Algorithmic solution:



Figure 1.4: Chain Responsability to Timetabling problem

## Algorithmic solution:

```
1:  procedure GRASP METAHEURÍSTIC
2:      f* ← α
3:      input ← ReadInput();
4:      if validate(input) then
5:          for i ≤ i_max do
6:              x ← GreedyRandomized();
7:              x ← LocalSearch(x);
8:              if f(x) < f* then
9:                  f* ← f(x);
10:                 x* ← x;
11:             end if
12:         end for
13:     end if
14:     WriteOutput(x*);
15: end procedure
```

**Computational experiments:**

According to (**??**) the algorithm will be tested using the following instances:

| Instance | small | medium | large |
|----------|-------|--------|-------|
| Events   | 100   | 400    | 400   |
| Room     | 5     | 5      | 10    |
| Day      | 5     | 5      | 5     |
| Slot     | 8     | 8      | 8     |

Table 1.3: Set of test instances at EAFIT University

# Computational experiments:

| Room | Type Room | Capacity | Event | Group | Max Student | Day | Start Time | End Time |
|------|-----------|----------|-------|-------|-------------|-----|------------|----------|
| 35401 | Aula Normal | 50 | METODOLOGÍA DEL APRENDIZAJE | 024 | 26 | T | 7.0 | 9.0 |
| 35401 | Aula Normal | 50 | MODELACIÓN Y SIMULACIÓN I | 001 | 30 | T | 9.0 | 12.0 |
| 35401 | Aula Normal | 50 | METODOLOGÍA DEL APRENDIZAJE | 024 | 26 | W | 14.0 | 15.0 |
| 35401 | Aula Normal | 50 | MATEMÁTICAS I | 002 | 40 | TH | 18.0 | 21.0 |
| 35401 | Aula Normal | 50 | MATEMÁTICAS 1 | 010 | 37 | F | 6.0 | 9.0 |
| 35403 | Aula Normal | 36 | MATEMÁTICAS III (ECONOMÍA) | 003 | 32 | F | 6.0 | 9.0 |
| 35501 | Aula Normal | 50 | BIOLOGÍA MOLECULAR | 101 | 30 | F | 6.0 | 9.0 |
| 35501 | Aula Normal | 50 | MATEMÁTICAS 1 | 001 | 37 | S | 12.0 | 15.0 |
| 38107 | Aula Normal | 25 | FUNDAMENTOS DE FISICOQUÍMICA | 156 | 20 | T | 15.0 | 17.0 |
| 38107 | Aula Normal | 25 | FUNDAMENTOS DE FISICOQUÍMICA | 156 | 20 | S | 9.0 | 12.0 |
| 38201 | Aula Normal | 50 | BIOÉTICA | 001 | 30 | M | 6.0 | 8.0 |
| 38201 | Aula Normal | 50 | METODOLOGÍA DEL APRENDIZAJE | 022 | 26 | W | 14.0 | 15.0 |
| 38201 | Aula Normal | 50 | ESTADÍSTICA 2 (ECONOMÍA) | 001 | 26 | W | 9.0 | 12.0 |
| 38201 | Aula Normal | 50 | BIOÉTICA | 001 | 30 | F | 11.0 | 12.0 |

Table 1.4: Timetabling Events at EAFIT University

**Conclusions:**

- The first stage of the problem was successfully modeled, which corresponds to the assignment of events.
- An algorithm was proposed for the scheduling of events at the EAFIT University.
- An algorithm was proposed for the scheduling of teacher at the EAFIT University.

(Bonus Slides)

## Solution construction flow:

```
1: procedure GREEDYRANDOMIZED
2:     teacher ← SetOutData( GetInData(event));
3:     event ← SetNext(teacher);
4:     room ← SetOutData( GetInData(event) );
5:     room ← SetNext(event);
6:     slot ← SetSlotFree( GetInData(slot) );
7:     slot ← SetOutData( GetInData(room) );
8:     slot ← SetNext(room);
9:     processor(slot);
10: end procedure
```

## Processor:

```
1: procedure PROCESSOR
2:     MakeObject(inData);
3:     if   Next() ≠ null then
4:         Next().processor()
5:     end if
6: end procedure
```