

# Abstract

- When scheduling healthcare providers, it is frequently not possible to satisfy every scheduling request. Multi-criteria objective functions provide one method for overcoming this challenge, but can result in undesirable schedules. We discuss an alternative method for resolving conflicting scheduling requests for a resident scheduling problem at a major teaching hospital. Our method, Request Selection Via Cuts (RSVC), involves iteratively solving a sequence of optimization problems in order to identify all maximally feasible and minimally infeasible sets of time off requests. Although we use our method for scheduling medical residents, it is applicable to any problem that involves soft constraints (i.e., constraints that are preferred but not required to be satisfied).

# Scheduling Medical Residents With Conflicting Requests for Time Off

Brian Lemay

University of Michigan

POMS: May 8<sup>th</sup>, 2016

Advisor: Prof Amy Cohn

# Outline

- Motivation and Background
- Goal and Method
- Definitions
- Relevant Literature
- Algorithm Explanation
- Computational Testing
- Case Study Example
- Conclusions/Future Work

# Motivation



# Motivation



3 points



1 point



# Goal and Method

**Goal:** Identify the exhaustive collection of *maximally feasible* and *minimally infeasible* sets of requests which can in turn be used by the scheduler to select their preferred solution

**Method:** We develop an algorithm, which we call **Request Selection Via Cuts (RSVC)**, to identify these sets by solving a sequence of optimization problems

# Definitions

**Feasible Request Set** – A set of time-off requests such that a schedule exists that satisfies every “hard” constraint and every request in the set

**Infeasible Request Set** – A set of time-off requests such that no schedule exists that satisfies every “hard” constraint and every request in the set






# Definitions

**Maximally Feasible Set (MFS)** – A feasible request set is maximally feasible if there is no single request that can be added to the set such that the resulting set is feasible

**Minimally Infeasible Set (MIS)** – An infeasible request set is minimally infeasible if there is no single request that can be removed from the set such that the remaining set is infeasible





















# An Example

- $r_1$ : Gene wants to attend his sister's wedding ()
- $r_2$ : Billy want to go to a Michigan football game ()
- $r_3$ : Marina wants to attend a Red Sox playoff game ()
- $r_4$ : Brian wants to watch his wife's Ironman race ()
- $r_5$ : Amy wants to run the Chicago Marathon ()

Simply maximizing the number of granted requests could result in granting , , and 

# An Example

It may be more useful to identify all minimally infeasible sets since they indicate incompatible sets of requests: {, **M**}, {, , }, {, , }, {, , }, {**M**, , , }, and {**M**, , , }

Alternatively, it may be more useful to identify all maximally feasible sets: {, }, {**M**, , }, {**M**, , , }, and {, , , }

# Some Relevant Literature

- Analyzing infeasible linear, mixed-integer, and integer linear programs: **J.N.M. van Loon, Oliver Guieu, John Chinneck, Erik Dravnieks, and Nilotpal Chakravarti**
- Maximum feasible and minimum infeasible subsystems: **Edoardo Amaldi, Marc Pfetsch, Leslie Trotter Jr., Edoardo Amaldi, and Viggo Kann**
- Finding minimal “conflict sets”: **Benjamin Han, Shie-Jue Lee, Maria Garcia de la Banda, Peter Stuckey, Jeremy Wazny**
- Using optimization to find maximal feasible sets: **Amy Cohn and Cynthia Barnhart**
- Finding minimal unsatisfiable and maximal satisfiable sets: **James Bailey and Peter Stuckey**

# Finding All Maximally Feasible Sets

1. Maximize the number of requests satisfied such that a feasible schedule exists.
2. Let  $R^F$  be the set of satisfied requests in the optimal solution found ( $R^F$  is a maximally feasible set).
3. Add a cut to the optimization problem that requires satisfying at least one request that is not in  $R^F$  (i.e. one request from the set  $\bar{R}^F$ ).
  - This cut eliminates exactly those solutions that only satisfy the requests in  $R^F$  or a subset of the requests in  $R^F$ .
4. Return to Step 1.

# Finding All Minimally Infeasible Sets

**Key Point:** An infeasible request set, by definition, is not a subset of any feasible request set.

Thus, any infeasible request set must include at least one request from the complement of each maximally feasible set.

# Finding All Minimally Infeasible Sets

**Given all maximally feasible sets:**

1. Find a set of requests that is minimal in size and includes at least one request from the complement of each maximally feasible set.
  - Note: it is not necessary to consider the feasible region of the resident scheduling problem.
2. Let  $R^I$  be the set of satisfied requests in the optimal solution found ( $R^I$  is a minimally infeasible set).
3. Add a cut to the optimization problem that eliminates all solutions that satisfy every request in  $R^I$ .
  - This cut eliminates exactly those solutions that only satisfy the requests in  $R^I$  or a superset of the requests in  $R^I$ .
4. Return to Step 1.



# Finding MFSs and MISs Simultaneously

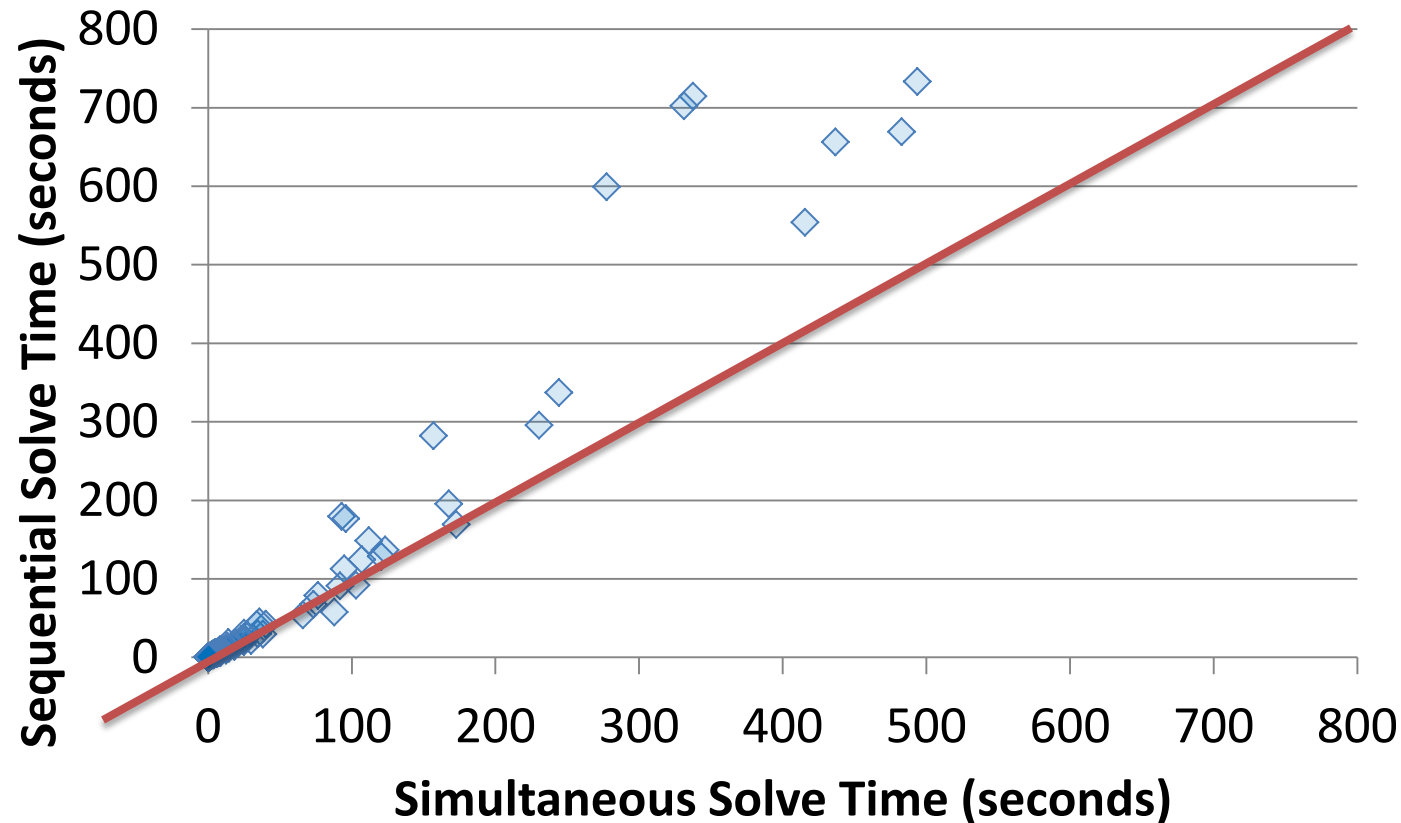
1. Attempt to find a minimally sized set of requests such that:
  - The set is not a subset of any known MFS (i.e., a known feasible set)
  - The set is not a superset of any known MIS (i.e., a known infeasible set)
2. If feasible, let  $R^*$  be set of satisfied requests in the optimal solution found.
3. Attempt to find a maximally feasible request set such that:
  - All requests in  $R^*$  are satisfied
4. If feasible, a new maximally feasible set has be identified. Otherwise,  $R^*$  is a new minimally infeasible set.
5. Return to Step 1

# RSVC Summary

- “Sequential” method:
  - Iteratively finds every maximally feasible set
  - Then, iteratively finds every minimally infeasible set
  - One optimization problem solved for each identified set
- “Simultaneous” method:
  - Iteratively finds maximally feasible and minimally infeasible sets in no particular order
  - Two optimization problems solved for each identified set

# Some Computational Results

## Solve Time Comparison



**Simultaneous is generally faster despite needing to solve twice as many optimization problems**

# Case Study Example

	Maximally Feasible Sets:															
Name-Reason	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Singla-My birthday	D															
Schellpfeffer-Fellowship interview		D														
Queen-Camping			D	D			D			D						
Dubin-Anaesthesia			D		D				D					D		
Madison-Camping				D		D						D				
Neher-A Full Thursday off					D			D								
Royer-Spouse's birthday						D							D	D	D	
Chunzer-Conference							D	D							D	D
Dubin-Nephew Birthday									D							
Martel-Chicago Trip										D	D		D			
Martel-Holiday											D					
Madison-Dog's birthday												D				
Chunzer-Camping																D

**\*This Resident Scheduling Problem included 200 Total Requests.**

# Case Study Example

	Minimally Infeasible Sets:															
Name-Reason	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Singla-My birthday	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Schellpfeffer-Fellowship interview	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Dubin-Anaesthesia	1	1	1	1	1	1				1		1				
Chunzer-Conference	1	1	1	1			1		1		1				1	
Martel-Chicago Trip	1	1			1	1	1	1	1							1
Madison-Camping	1		1		1		1	1		1	1		1			
Queen-Camping		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Royer-Spouse's birthday		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Madison-Dog's birthday		1		1		1			1			1		1	1	1
Martel-Holiday			1	1						1	1	1	1	1	1	
Neher-A Full Thursday off					1	1	1	1	1	1	1	1	1	1	1	1
Chunzer-Camping					1	1		1		1		1	1	1		1
Dubin-Nephew Birthday							1	1	1		1		1	1	1	1

**\*This Resident Scheduling Problem included 200 Total Requests.**

# Conclusions and Future Work

## **Conclusions:**

- Developed algorithm that identifies all maximally feasible and minimally infeasible sets of requests
- Algorithm eliminates the need to define a weighted objective function
- Algorithm can be extended to other soft constraints

## **Future Work:**

- Additional computational testing
- Identify opportunities for performance improvements
- Case studies involving real-world schedulers
- Visualization tool



# Thank You!

**Contact info:**

Brian Lemay: [blemay@umich.edu](mailto:blemay@umich.edu)

# Formulations (sequential):

$$\begin{aligned} (\text{FP}_i) \quad & \text{maximize} && \sum_{r \in R} x_r \\ & \text{subject to} && \mathbf{x} \in \mathbf{X}^H \\ & && \sum_{r \in R \setminus R_k^F} x_r \geq 1 \quad \forall k = 0, \dots, i-1 \end{aligned}$$

$$\begin{aligned} (\text{IP}_j) \quad & \text{minimize} && \sum_{r \in R} x_r \\ & \text{subject to} && \sum_{r \in R \setminus F} x_r \geq 1 \quad \forall F \in \mathbb{F} \\ & && \sum_{r \in R_k^I} x_r \leq |R_k^I| - 1 \quad \forall k = 0, \dots, j-1 \end{aligned}$$

# Formulations (simultaneous):

$$\begin{array}{ll} \text{(MIN SET)} & \text{minimize} \quad \sum_{r \in R} x_r \\ & \text{subject to} \quad \sum_{r \in R \setminus F} x_r \geq 1 \quad \forall F \in \mathbb{F} \\ & \quad \quad \quad \sum_{r \in I} x_r \leq |I| - 1 \quad \forall I \in \mathbb{I} \end{array}$$

$$\begin{array}{ll} \text{(MAX SET)} & \text{maximize} \quad \sum_{r \in R} x_r \\ & \text{subject to} \quad \mathbf{x} \in \mathbf{X}^H \\ & \quad \quad \quad x_r = 1 \quad \forall r \in R^* \end{array}$$