A Mixed-Integer Programming Approach to Generating Sports Schedules

Joe Keefer, Siyang Chen

Question: How are sports events scheduled?

Example: Major League Baseball Scheduling

- Scheduling sports events is an interesting and difficult problem.

Example: Major League Baseball Scheduling

- Scheduling sports events is an interesting and difficult problem.
- There may be a large number of constraints and specifications.

Example: Major League Baseball Scheduling

- Scheduling sports events is an interesting and difficult problem.
- There may be a large number of constraints and specifications.
- For example:

Example: Major League Baseball Scheduling

- Scheduling sports events is an interesting and difficult problem.
- There may be a large number of constraints and specifications.
- For example:
  - Each team plays 162 games: 81 home and 81 away. [3]

Example: Major League Baseball Scheduling

- Scheduling sports events is an interesting and difficult problem.
- There may be a large number of constraints and specifications.
- For example:
  - Each team plays 162 games: 81 home and 81 away. [3]
  - The length of the regular season must be between 178 and 183 days. [3]

Example: Major League Baseball Scheduling

- Scheduling sports events is an interesting and difficult problem.
- There may be a large number of constraints and specifications.
- For example:
  - Each team plays 162 games: 81 home and 81 away. [3]
  - The length of the regular season must be between 178 and 183 days. [3]
  - Teams cannot play every day for more than 20 days consecutively. [5]

Example: Major League Baseball Scheduling

- Scheduling sports events is an interesting and difficult problem.
- There may be a large number of constraints and specifications.
- For example:
  - Each team plays 162 games: 81 home and 81 away. [3]
  - The length of the regular season must be between 178 and 183 days. [3]
  - Teams cannot play every day for more than 20 days consecutively. [5]
  - Teams cannot play on the west coast on one day and on the east coast on the following day, or vice versa. [5]

Example: Major League Baseball Scheduling

There may be other strange requests as well. For example:

Example: Major League Baseball Scheduling

There may be other strange requests as well. For example:

- "...every time the Pope comes to America, he goes to a baseball stadium." [4]

Example: Major League Baseball Scheduling

There may be other strange requests as well. For example:

- "...every time the Pope comes to America, he goes to a baseball stadium." [4]
- "The Boston Red Sox must play a home game on Patriot's Day, the third Monday in April." [4]

Example: Major League Baseball Scheduling

There may be other strange requests as well. For example:

- "...every time the Pope comes to America, he goes to a baseball stadium." [4]
- "The Boston Red Sox must play a home game on Patriot's Day, the third Monday in April." [4]
- The Minnesota Twins play away on the first day of hunting season. [8]

Example: Major League Baseball Scheduling

There may be other strange requests as well. For example:

- "...every time the Pope comes to America, he goes to a baseball stadium." [4]
- "The Boston Red Sox must play a home game on Patriot's Day, the third Monday in April." [4]
- The Minnesota Twins play away on the first day of hunting season. [8]
- "...northern teams must begin the season on the road in warm weather cities." [4]

Intramural Sports Scheduling

- Intramural games have less restrictions, but scheduling is tricky nonetheless.

Intramural Sports Scheduling

- Intramural games have less restrictions, but scheduling is tricky nonetheless.
- There are paid services and software for scheduling intramural sports:

Intramural Sports Scheduling

- Intramural games have less restrictions, but scheduling is tricky nonetheless.
- There are paid services and software for scheduling intramural sports:
  - `http://www.allprosoftware.com/` [2]

Intramural Sports Scheduling

- Intramural games have less restrictions, but scheduling is tricky nonetheless.
- There are paid services and software for scheduling intramural sports:
  - `http://www.allprosoftware.com/` [2]
  - Duke currently pays to use `http://www.imleagues.com` [1].

Considerations in Intramural Sports Scheduling

- Maximise use of available facilities

Considerations in Intramural Sports Scheduling

- Maximise use of available facilities
- Schedule must fit teams' preferences

Considerations in Intramural Sports Scheduling

- Maximise use of available facilities
- Schedule must fit teams' preferences
- Sufficient resting period between games (e.g. at most one game per day)

Considerations in Intramural Sports Scheduling

- Maximise use of available facilities
- Schedule must fit teams' preferences
- Sufficient resting period between games (e.g. at most one game per day)
- Each team plays at least a certain number of games during the season

Our Model: Introduction

To create a way to schedule intramural sports, we make the following assumptions:

Our Model: Introduction

To create a way to schedule intramural sports, we make the following assumptions:

- We schedule games for only one sport.

Our Model: Introduction

To create a way to schedule intramural sports, we make the following assumptions:

- We schedule games for only one sport.
- A certain number of teams are assigned to some available game slots.

Our Model: Introduction

To create a way to schedule intramural sports, we make the following assumptions:

- We schedule games for only one sport.
- A certain number of teams are assigned to some available game slots.
- Exactly two teams can participate in a game.

Our Model: Introduction

To create a way to schedule intramural sports, we make the following assumptions:

- We schedule games for only one sport.
- A certain number of teams are assigned to some available game slots.
- Exactly two teams can participate in a game.
- Each team has preferences for some game slots over others.

Our Model: Introduction

To create a way to schedule intramural sports, we make the following assumptions:

- We schedule games for only one sport.
- A certain number of teams are assigned to some available game slots.
- Exactly two teams can participate in a game.
- Each team has preferences for some game slots over others.
- Each team can play every other team.

Our Model: Introduction

- Many possibly related objectives, sharing common parameters (team preferences, game start times, etc. . .)

Our Model: Introduction

- Many possibly related objectives, sharing common parameters (team preferences, game start times, etc. . .)
- Idea:

Our Model: Introduction

- Many possibly related objectives, sharing common parameters (team preferences, game start times, etc. . .)
- Idea:
  - Use multiple MIPs to solve easy subproblems of general problem (e.g. max use of facilities, team preferences, etc. . .)

Our Model: Introduction

- Many possibly related objectives, sharing common parameters (team preferences, game start times, etc. . .)
- Idea:
  - Use multiple MIPs to solve easy subproblems of general problem (e.g. max use of facilities, team preferences, etc. . .)
  - MIPs have common input parameters

Our Model: Introduction

- Many possibly related objectives, sharing common parameters (team preferences, game start times, etc. . .)
- Idea:
  - Use multiple MIPs to solve easy subproblems of general problem (e.g. max use of facilities, team preferences, etc. . .)
  - MIPs have common input parameters
  - 'Combine' MIPs by taking linear combination of objective functions from each subproblem

Our Model: Formal Description

These parameters and variables are common across all subproblems:

Our Model: Formal Description

These parameters and variables are common across all subproblems:

- **sets**:
  - *teams*
  - *games*

Our Model: Formal Description

These parameters and variables are common across all subproblems:

- **sets**:
  - *teams*
  - *games*
- **parameters:**
  - $time[g \in games]$:= the starting time of game g (assumed to be unique)
  - $preferences[g \in games, t \in teams]$:= between 0 and 1, how much team t wants to play in game g

Our Model: Formal Description

These parameters and variables are common across all subproblems:

- **sets**:
  - *teams*
  - *games*
- **parameters:**
  - $time[g \in games]$:= the starting time of game g (assumed to be unique)
  - $preferences[g \in games, t \in teams]$:= between 0 and 1, how much team t wants to play in game g
- **variables:**
  - $plays[g \in games, t \in teams]$:= binary, denoting whether or not team t plays in game g

Subproblem: Facility Usage

We want to use the given facilities as much as possible:

Subproblem: Facility Usage

We want to use the given facilities as much as possible:

- **constraints:**
  - $\forall g \in games : \sum_{t \in teams} plays[g, t] = 2$

Subproblem: Utility

Maximise teams' preferences in the games they play:

Subproblem: Utility

Maximise teams' preferences in the games they play:

- **max:** $\sum_{g \in games} \sum_{t \in teams} plays[g, t] \cdot preferences[g, t]$

Subproblem: Minimum Games Played

Maximise the minimum number of games that any team plays:

Subproblem: Minimum Games Played

Maximise the minimum number of games that any team plays:

- **max:** *min_games*

Subproblem: Minimum Games Played

Maximise the minimum number of games that any team plays:

- **max:** *min_games*
- **constraints:**
  - $\forall t \in teams : \sum_{g \in games} plays[g, t] \geq min\_games$

Subproblem: Minimum Gap between Games

Maximise the minimum gap between any two games played by one team:

Subproblem: Minimum Gap between Games

Maximise the minimum gap between any two games played by one team:

- **max:** *min_gap*

Subproblem: Minimum Gap between Games

Maximise the minimum gap between any two games played by one team:

- **max:** min_gap
- **constraints:**
  - $\forall t \in teams, \forall g_1, g_2 \in games, g_1 \neq g_2$:
    if $plays[g_1, t], plays[g_2, t]$:
    $|time[g_1] - time[g_2]| \geq min\_gap$

Subproblem: Minimum Gap between Games

Maximise the minimum gap between any two games played by one team:

- **max:** min_gap
- **constraints:**
  - $\forall t \in teams, \forall g_1, g_2 \in games, g_1 \neq g_2$:
    if $plays[g_1, t], plays[g_2, t]$:
    $|time[g_1] - time[g_2]| \geq min\_gap$

Problem: How do we structure the constraint in terms of a mixed-integer program?

Subproblem: Minimum Gap between Games

Solution: Rewrite constraint using logical disjunctions:

Subproblem: Minimum Gap between Games

Solution: Rewrite constraint using logical disjunctions:

$$\forall t \in \mathit{teams}, \forall g_1, g_2 \in \mathit{games}, g_1 \neq g_2 :$$
$$\text{if } \mathit{plays}[g_1, t], \mathit{plays}[g_2, t] :$$
$$|\mathit{time}[g_1] - \mathit{time}[g_2]| \geq \mathit{min\_gap}$$

Subproblem: Minimum Gap between Games

Solution: Rewrite constraint using logical disjunctions:

$$\forall t \in \mathit{teams}, \forall g_1, g_2 \in \mathit{games}, g_1 \neq g_2 :$$
$$\text{if } \mathit{plays}[g_1, t], \mathit{plays}[g_2, t] :$$
$$|\mathit{time}[g_1] - \mathit{time}[g_2]| \geq \mathit{min\_gap}$$

$$\Updownarrow$$

Subproblem: Minimum Gap between Games

Solution: Rewrite constraint using logical disjunctions:

$$\forall t \in \mathit{teams}, \forall g_1, g_2 \in \mathit{games}, g_1 \neq g_2 :$$
$$\text{if } \mathit{plays}[g_1, t], \mathit{plays}[g_2, t] :$$
$$|\mathit{time}[g_1] - \mathit{time}[g_2]| \geq \mathit{min\_gap}$$

$$\Updownarrow$$

$$\forall t \in \mathit{teams}, \forall g_1, g_2 \in \mathit{games} :$$
$$g_1 = g_2$$
$$\text{OR}$$
$$\mathit{plays}[g_1, t] + \mathit{plays}[g_2, t] \leq 1$$
$$\text{OR}$$
$$|\mathit{time}[g_1] - \mathit{time}[g_2]| \geq \mathit{min\_gap}$$

Subproblem: Minimum Gap between Games

Rewriting the constraint to work in `mathprog` language [7, 6]:

Subproblem: Minimum Gap between Games

Rewriting the constraint to work in `mathprog` language [7, 6]:

$$
\begin{gathered}
bit\_1, bit\_2, bit\_3, bit\_4 \text{ binary} \\
time[g_1] - time[g_2] \leq MAX\_GAP \cdot bit\_1 \\
time[g_2] - time[g_1] \leq MAX\_GAP \cdot bit\_1 \\
plays[g_1, t] + plays[g_2, t] \leq 1 + bit\_2 \\
time[g_1] - time[g_2] \geq min\_gap - MAX\_GAP \cdot bit\_3 \\
time[g_2] - time[g_1] \geq min\_gap - MAX\_GAP \cdot bit\_4 \\
bit\_1 + bit\_2 + bit\_3 + bit\_4 \leq 3
\end{gathered}
$$

Subproblem: Minimum Gap between Games

Rewriting the constraint to work in `mathprog` language [7, 6]:

$$
\begin{gathered}
bit\_1, bit\_2, bit\_3, bit\_4 \text{ binary} \\
time[g_1] - time[g_2] \leq MAX\_GAP \cdot bit\_1 \\
time[g_2] - time[g_1] \leq MAX\_GAP \cdot bit\_1 \\
plays[g_1, t] + plays[g_2, t] \leq 1 + bit\_2 \\
time[g_1] - time[g_2] \geq min\_gap - MAX\_GAP \cdot bit\_3 \\
time[g_2] - time[g_1] \geq min\_gap - MAX\_GAP \cdot bit\_4 \\
bit\_1 + bit\_2 + bit\_3 + bit\_4 \leq 3
\end{gathered}
$$

We use a trick to turn multiple OR statements into linear inequalities with boolean variables. MAX_GAP is picked to be sufficiently large enough such that whenever bit_i is true, then inequality i can be violated by MAX_GAP. The last statement forces at least one inequality to hold.

Combined Objective Function

Finally, we can combine the separate objective functions from our subproblems:

$$\textbf{max: } \beta_1 \cdot \textit{min\_games} + \beta_2 \cdot \textit{utility} + \beta_3 \cdot \textit{min\_gap}$$

The β_i parameters are used to weight particular subproblems and objectives.

Data

Time in seconds taken to calculate the following data sets:

$\beta_3 > 0$		$\lvert games \rvert$				
		10	12	13	15	48
$\lvert teams \rvert$	4	26.867	29.134	> 200	> 210	
	6	> 390	203.875			
	15					> 9001.9

$\beta_3 = 0$		$\lvert games \rvert$		
		12	15	16
$\lvert teams \rvert$	4	0.452	> 160	
	6	0.348	0.228	> 100
	8	0.868	> 210	
	10	> 3800		

The use of $>$ denotes that the program was forcibly terminated at this time because it had taken too long to run.

Comments about Scalability

- β_3 is the coefficient for the minimum gap constraint; we noticed that this subproblem required substantially more time to compute, due to its complexity. As a result we decided to test the program with $\beta_3 = 0$ also.

Comments about Scalability

- β_3 is the coefficient for the minimum gap constraint; we noticed that this subproblem required substantially more time to compute, due to its complexity. As a result we decided to test the program with $\beta_3 = 0$ also.
- The program overall is very slow: a regular intramural league at Duke University has anywhere between 7 and 15 teams playing around 50 games; however, our program became extremely slow with more than 10 games or 8 teams.

Comments about Scalability

- β_3 is the coefficient for the minimum gap constraint; we noticed that this subproblem required substantially more time to compute, due to its complexity. As a result we decided to test the program with $\beta_3 = 0$ also.
- The program overall is very slow: a regular intramural league at Duke University has anywhere between 7 and 15 teams playing around 50 games; however, our program became extremely slow with more than 10 games or 8 teams.
- For example, the program with 15 teams and 48 games produced 42192 integer variables, all of which were binary.

Comments about Scalability

- β_3 is the coefficient for the minimum gap constraint; we noticed that this subproblem required substantially more time to compute, due to its complexity. As a result we decided to test the program with $\beta_3 = 0$ also.
- The program overall is very slow: a regular intramural league at Duke University has anywhere between 7 and 15 teams playing around 50 games; however, our program became extremely slow with more than 10 games or 8 teams.
- For example, the program with 15 teams and 48 games produced 42192 integer variables, all of which were binary.
- By Moore's Law, we estimate that by the year 2044 there will be enough processing power to schedule the freshman men's soccer intramural league at Duke University.

Further Issues: Robustness

- In practice, every season a large number of games are forfeited because one of more assigned teams cannot participate at the given time.

Further Issues: Robustness

- In practice, every season a large number of games are forfeited because one of more assigned teams cannot participate at the given time.
- Given a cancellation, can we find a way to swap the participating teams with another game while still maximising the objective?

Further Issues: Robustness

- In practice, every season a large number of games are forfeited because one of more assigned teams cannot participate at the given time.
- Given a cancellation, can we find a way to swap the participating teams with another game while still maximising the objective?
- How can we mitigate this by creating a flexible or robust schedule? (I.e., can we create a schedule such that for any set of k cancellations, we can reschedule games in a way that maintains a high objective value?)

Further Issues: Robustness

- In practice, every season a large number of games are forfeited because one of more assigned teams cannot participate at the given time.
- Given a cancellation, can we find a way to swap the participating teams with another game while still maximising the objective?
- How can we mitigate this by creating a flexible or robust schedule? (I.e., can we create a schedule such that for any set of k cancellations, we can reschedule games in a way that maintains a high objective value?)
- A simpler problem: given a fixed schedule, how do we find the worst possible set of k cancelled games?

Further Issues: Reporting Preferences

- Is it always optimal for teams to report preferences truthfully?

Further Issues: Reporting Preferences

- Is it always optimal for teams to report preferences truthfully?
- How do we design a mechanism so that teams always report truthfully?

Sources I


:: imleagues ::.
http://www.imleagues.com/, 2010.


All pro software: League scheduler.
http://www.allprosoftware.com/, 2010.


Major league baseball schedule.
http://en.wikipedia.org/wiki/Major_League_Baseball_schedule, 2010.


David P. Anderson and Gilles Fedak.
New yorkers & co.; major league scheduling: The high, hard one.
The New York Times, April 1999.


The Plain Dealer Bill Lubinger.
Dissatisfaction guaranteed: Major league baseball can't please anyone with scheduling.
May 2009.

Sources II


J. Bisschop.
AIMMS-optimization modeling.
Lulu. com, 2006.


L. Foulds, B. Toklu, and J. Wilson.
Modelling either-or relations in integer programming.
Annals of Operations Research, 166:203–222, 2009.
10.1007/s10479-008-0409-z.


Ken Mcaloon, Carol Tretkoff, and Gerhard Wetzel.
Sports league scheduling.
In *In Proceedings of the 3th Ilog International Users Meeting*, 1997.