University of British Columbia

Math 441

A Constraint Satisfaction Approach to Optimizing Sport Schedules

Authors: Siddharth Nand, Eros Rojas, Jashan Brar, Yovindu Don

Supervisor: Dr. Patrick Walls

April 12, 2024

Contents

Abstract

Crafting sports schedules is a complex task, requiring consideration of various factors for fairness and logistical feasibility. This paper explores using constraint satisfaction techniques, drawn from artificial intelligence, to optimize sports schedules. Our methodology involves outlining the constraint satisfaction framework and employing the FIFA World Cup as a case study. We aim to develop an optimal and equitable schedule for major sporting events, considering factors such as match distribution, team rest periods, and revenue maximization. Additionally, we offer a predictive glimpse into what the schedule for the 2026 FIFA World Cup could resemble, based on our approach. This speculative projection aims to spark discussions for the planning and execution of one of the world's most anticipated sporting events.

1 Introduction

Crafting a sports schedules is a complex process that involves satisfying numerous factors to ensure fairness, competitiveness and logistical feasibility. Whether it's organizing a league's regular season or orchestrating a major tournament the goal is to create schedules that meet the needs of teams, venues, broadcasters and fans. In recent years, computational methods have become invaluable for optimizing these schedules by offering innovative solutions to intricate scheduling challenges.

1.1 Overview

This paper delves into the use of constraint satisfaction techniques for optimizing sports schedules. Constraint satisfaction provides a robust framework for modeling and solving scheduling problems. By encoding scheduling constraints as logical relationships, this approach enables the creation of schedules that meet multiple criteria while respecting specified constraints.

Our methodology is in two key phases: firstly, we outline the constraint satisfaction framework for sports scheduling, outlining the formulation of constraints and the algorithmic methods used for optimization. Throughout our paper, we continuously use the FIFA World Cup as an example to illustrate the application of our approach within the context of a major international sporting event.

1.2 Research Question

Our study aims to address the question: How can we develop an optimal and equitable schedule of major sporting events, such as the FIFA World Cup, considering factors such as geographical distribution of matches, sufficient rest periods for teams whilst maximizing revenue?

1.3 Objectives and Significance

The main goal of this research is to devise a scheduling framework that tackles the specific demands of organizing major sporting events. This involves ensuring fairness in playing conditions across all participating teams, optimizing match distribution among host cities and maximizing revenue for the venues. The significance of this study lies in its potential to contribute to the successful execution of large-scale sporting events serving as a blueprint for future tournaments and offering valuable insights into the planning and optimization of such events **?**.

2 Background

2.1 Optimization

Optimization is a fundamental concept in mathematics and computer science, concerned with finding the best solution to a given problem from a set of possible solutions. It plays a crucial role in various fields such as engineering, economics, operations research, and machine learning.

2.2 Convexity

Convexity lies at the heart of optimization, shaping the behavior of functions and sets in profound ways. Its mathematical elegance and practical implications make it a cornerstone in optimization theory and practice. Understanding convexity unveils fundamental principles that drive efficient solutions in a wide range of real-world applications.

Convex Optimization

Convex optimization refers to the class of mathematical optimization problems where the objective function and the feasible set are both convex. Convexity is a fundamental property in optimization, characterized by the property that any line segment connecting two points on the graph of the function lies above or on the graph itself. Mathematically, a function $f(x)$ defined on a set C is convex if, for all $x_1, x_2 \in C$ and for all λ in the interval $[0, 1]$, the following inequality holds: **?**

$$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda)f(x_2)$$

Convex optimization problems have desirable properties, including the guarantee of global optimality for local optima, the existence of efficient algorithms for finding optimal solutions, **?** and robustness to perturbations **?**. Common examples of convex optimization problems include linear programming, quadratic programming, and convex semi-definite programming. **?**

Non-Convex Optimization

Non-convex optimization, in contrast, encompasses problems where either the objective function, the feasible set, or both are non-convex. Non-convex optimization problems are significantly more challenging than their convex counterparts due to the presence of local optima, saddle points, and multiple disconnected feasible regions. Unlike convex optimization problems, non-convex problems do not enjoy the same guarantees of global optimality for local optima, and finding global optima can be computationally intensive or even NP-hard in some cases. **?**

2.3 Scheduling Theory

Scheduling theory is a crucial segment of operational research. Initially it was intended for applications in computer science and manufacturing but has now extended into fields such as agriculture and transport. This is due to its decision-making pro-

cesses which involves the allocation of limited resources over time **?**. This discipline addresses the challenges of optimally arranging and timetabling tasks with goals such as minimizing completion times or maximizing resource utilization **? ?**.

In sports scheduling, the scheduling frameworks are adapted to account for the geographical distribution of venues, the need for rest days between matches for teams and the sequential manner in tournament phases. Therefore by extending operations research frameworks to sports scheduling, it can generate high-quality match schedules that meet logistical needs as well as the engagement of fans.

2.4 Factors Influencing Scheduling in Sports Events

Geographical Considerations

An important factor when scheduling international sports competition is Geography. As there are significant logistical challenges such as the distances between venues which require meticulous planning **?**. Travel logistics need to take into account: time zones and local climatic conditions, this is essential to minimize athlete fatigue and optimize performance. The implications also extend beyond well-being of athletes as it also needs to account for fans who might travel. So the aim is to minimize travel distances and set fair match start times to ensure equity, optimize performance and an enjoyable fan participation.

Rest and Recovery

Athlete health is a big pillar in sports scheduling where the allocation of adequate rest periods between matches is critical. Research indicates that sufficient rest is integral for injury prevention and maintaining optimal performance **?**. In scheduling terms, this translates into mathematical constraints that maintain a minimum rest period between consecutive games for each team. Models must effectively balance rest periods with the logistical aspects of the tournament.

Popularity and Viewership

The attractiveness and expected audience are crucial factors in sports scheduling for both fans and revenue. High-profile matches involving teams such as Argentina, Brazil, France and England to name a few are scheduled during peak hours to maximize viewership and advertising income. Integrating economic and viewership models into the scheduling process enables the prediction of revenue streams and audience sizes which vary with different match timings and pairings **?**.

2.5 Discrete Optimization

Discrete optimization involves finding the optimal solution from a finite set of choices and includes problem types like integer programming where solutions must be integer and combinatorial optimization which deals with finding the most efficient ordering of elements in a set **?**.

Key terms in discrete optimization include the objective function which is the formula that needs to be maximized or minimized, constraints which are the limitations or requirements that the solution must satisfy and decision variables which represent the choices available that can affect the outcome of the objective function. Integer programming and combinatorial optimization problems are essential in fields like logistics or finance ast they are characterized by their complex and discrete decision spaces. They are often NP-hard in nature indicating that finding optimal solutions for them is computationally difficult and likely requires exponential time.

Discrete optimization principles are relevant to scheduling problems especially ones aiming to allocate resources or events optimally under constraints. Round-robin tournaments in which each team plays every other team once such as the Group Stage have significant need for discrete optimization **?**. A simplified implementation of discrete optimization can be represented by the following:

Decision Variables:

$$X_{ijk} = \begin{cases} 1 & \text{if team } i \text{ plays against team } j \text{ in time slot } k, \\ 0 & \text{otherwise.} \end{cases}$$

Objective Function:

$$\min \sum_{i,j,k} d_{ij} X_{ijk}$$

where d_{ij} denotes the distance between the locations of teams i and j.

Constraints are added to ensure each team plays the required number of matches, no team plays more than once per time slot and other logistical or fairness conditions are met.

2.6 Constraint Satisfaction Problems (CSP)

Constraint Satisfaction Problems (CSP) serve as a crucial framework within combinatorial optimization. They are aimed at identifying solutions that satisfy multiple constraints simultaneously. A CSP involves assigning values to a set of variables, represented as $X = \{x_1, x_2, \ldots, x_n\}$, from predetermined domains $D = \{D_1, D_2, \ldots, D_n\}$, such that all imposed constraints $C = \{c_1, c_2, \ldots, c_m\}$ are met **?**. Variables represent decision elements, domains represent possible values for each variable and constraints represent restrictions on the variables

Techniques for solving CSPs typically involve algorithms such as backtracking. It involves systematically exploring variable assignments and backtracking when a constraint is violated **?**. Constraint propagation is a another key technique which reduces search space by eliminating values that cannot participate in a solution based on current partial assignments **?**. While some instances are solvable in polynomial time, many fall into the NP-Hard category or even NP-complete.

In sports scheduling CSPs offer a structured method to develop feasible schedules

that comply with constraints **?**. Matches are modeled as variables while potential dates and venues form the domains and logistical and fairness conditions represent the constraints. The application of CSPs enables a systematic approach to explore all possible scheduling configurations to satisfy constraints.

2.7 Solving CSPs

There are three primary methods used to solve Constraint Satisfaction Problems (CSPs): backtracking, constraint propagation, and local search techniques. This section delves deeper into these methods to provide a clearer understanding of how CSPs are tackled.

Backtracking Approach

Backtracking involves attempting to build a solution gradually, one component at a time and eliminating those components if they aren't possible. It is similar to depth-first search with the addition of pruning where the algorithm discards any partial answer and goes back to attempt other options if it is ever found that a partial solution cannot potentially lead to a complete solution **?**. Backtracking in the context of CSPs involves choosing a variable then giving it a value that is consistent with the current partial solution and moving on to the next variable in an iterative manner through the search space. Until all variables are allocated or there are no more values to assign, the procedure continues if no conflicts are found.

In the backtracking framework for scheduling, one can define decision variables similar to those in optimization models. For instance, let X_{ij} denote whether team i plays against team j in a particular round. The backtracking process can be represented by a decision tree, where each node represents a decision stage, and branches represent possible choices. The algorithm progresses by traversing this tree, making assignments to variables X_{ij}, and checking at each step whether the current partial solution violates any scheduling constraints. If a constraint is violated, the algorithm backtracks, undoing the last assignment and trying a different path in the decision tree. For example, it can be simplified as follows:

1. Select an unassigned pair of teams.

2. Assign them to the next available time slot.

3. Check if this leads to a conflict with existing assignments.

4. If a conflict is detected, remove the last assignment and try a different pair.

5. If no conflict is detected, proceed to assign the next pair of teams.

6. Continue until all teams are assigned without conflict or no more assignments are possible.

Constraint Propagation

Constraint Propagation is an inferential method used in solving CSPs to simplify constraints and reduce the search space. It involves reducing the domain of variables by enforcing consistency techniques and works by repeatedly applying local consistency methods until no further reduction is possible. A common form is arc consistency, which requires that for every variable x_i with a domain D_i and every variable x_j with a domain D_j for which there exists a constraint c_{ij}, every value in D_i has some corresponding value in D_j that satisfies c_{ij} **?**.

It begins before the constraint satisfaction problem even begins and can eliminate scheduling times that would violate constraints like venue availability or team preferences which reduces the search space effectively. To do this, take the domain D_i representing possible match times for team i, and D_j for team j where constraint propagation effectively removes times from D_i that conflict with already scheduled games in D_j according to the constraints C. It doesn't only streamline the problem before search algorithms like backtracking are used but also runs alongside to continuously reduce the problem space and expedite the solution. This is critical as many scheduling problems have a large space of potential solutions and are expensive computationally.

Local Search

Local Search is a heuristic method for problem-solving that iteratively explores the solution space of optimization problems by moving from one solution to another.

These movements are made within the neighborhood of the current solution with the aim of finding improved solutions based on constraints. Basically start with an initial solution and iteratively move to a neighbor solution if the neighbor is considered better **?**. It consists of algorithms such as Hill Climbing, Simulated Annealing, and Tabu Search where each avoids local minima and explores the solution space in their own different ways **?**.

Local Search methods are able to handle the multi-modal nature of sports scheduling problems where multiple solutions can exist. It can begin with an initial schedule S and iteratively generate neighboring schedules S' through small adjustments like swapping the time slots of two matches. If the adjustment leads to a schedule with improved evaluation $f(S') > f(S)$, the algorithm moves to the new schedule **?**.

3 Literature Review

Little research has been published with respect to this particular domain, largely due to the fact that sport-scheduling is only a small subset of applied constraint satisfaction. Nonetheless, there are two relevant papers that seem to touch on our problem: Scheduling of Sport Tournaments using Constraint Programming (Spanne, 2020), and Sports Scheduling: Problems and Applications, (Ribeiro, 2011). Although these papers are insightful in their own ways, we chose to use these simply as references as our problem is different enough to be not directly applicable. **??**

3.1 Spanne, 2020

Spanne touches on a slightly different style of sports scheduling compared to what we are focused on. Since most sports scheduling events have a time span of several weeks, Spanne attempts constrain the schedule within the realm of several days. Although this is largely unrealistic for real-world events, the goal of his thesis was to derive a method that is able to be fully adaptable for any scheduling problem whilst also being scalable. Previously to his paper, the best known algorithm for this standardized scheduling problem can handle approximately 50 teams, and 1000 matches. However, given there exist larger real world events, he was unsatisfied with the current techniques and sought to find a more efficient solution. **?**

Spanne derived a three-stage algorithm where match pairings, game scheduling, and the determination of stadiums are each their own independent optimization prob-

lems. More specifically, this algorithm is constructed as follows:

1. Generate game templates:

    (a) In this stage, the teams are enumerated into pairwise matches, where they are further separated into groups and rounds. In the context of Spanne's analysis, a round is a set of matches that takes place in the same day at the same arena. This is done to simplify the problem by a bit, as rounds are forced to be independent from one another.

2. Assign timeslots:

    (a) In this stage, every round that was created in the previous stage is assigned to an arena. Since our rounds are independent, one CSP can be optimized per round. It is important to notice that Spanne does not have a location constraint, unlike our analysis which does.

3. Assign arenas:

    (a) In this stage, every round is now assigned a date and start time. Although this is historically the most difficult part of scheduling problems, due to the work done in the previous two stages, Spanne makes this very simple. An independent CSP is given to each arena and day, and individual times are then scheduled.

This modular design allows for the user to modify intermediate results, or to switch out any specific component for something that may work better. Additionally, since each component is independent, they are able to run in parallel, which can drastically reduce computation time **?**. In conclusion, Spanne considered his algorithm to be a success as it satisfied all of the constraints that he sought to fix.

3.2 Ribeiro, 2011

Riberiro's paper is a case study which compares and contrasts the problem formulations that are utilized across different sports leagues. For example, when scheduling a game like Football there are specific considerations that must be taken in to account. These considerations and constraints are likely different for a game like Cricket, therefore in most situations a slightly different scheduling optimization is required **?**. Unfortunately, Riberio does not go in depth into any method in specific, and instead summarizes a wide variety of different methods across many disciplines. Since our problem is quite specific in nature, Riberio's work serves no other purpose other than simply being useful background information.

4 Design Process

Following our literature review, our design process began by identifying the type of optimization problem we faced. It became evident that our challenge was a discrete optimization problem. We then delved into considering suitable data structures and algorithms to develop a custom solver for optimizing sports schedules. However, recognizing the complexity involved in building a custom solution, we opted to explore existing tools. Initially, we attempted to utilize the popular CVXPY package. Regrettably, CVXPY's design for convex optimization posed significant challenges for our non-convex scheduling problem, **?** leading us to seek alternative solutions.

4.1 Problem Classification

Our first step in problem classification involved determining which class of optimization problems sports scheduling belonged to. We discerned between discrete and continuous optimization, ultimately identifying it as a discrete optimization problem due to the discrete decision variables involved, such as teams, stadiums, and days. Subsequently, we categorized the discrete problem further into three main types: Integer Programming (IP), Combinatorial Optimization, and Constraint Programming (CP).

- Integer Programming (IP): While IP initially seemed like a natural fit, we noted its potential limitations in modeling complex logical constraints. Expressing rules like "if a team plays on day i, they need D days of rest" could pose challenges with linear inequalities.

- Combinatorial Optimization: This initially seemed like a good option for our problem. However, combinatorial optimization faces challenges in effectively addressing sports scheduling problems due to the intricate interplay between various constraints, such as venue availability, team preferences, and rest periods. Combinatorial optimization algorithms may struggle to simultaneously optimize these diverse constraints while also achieving the desired scheduling objectives, leading to sub-optimal solutions or a combinatorial explosion.

- Constraint Programming (CP): CP stands out in solving sports scheduling problems by adeptly handling the constraints involved. For instance, when scheduling a multi-day soccer tournament, constraint programming can easily accommodate constraints like venue availability, team preferences, and required rest periods between games. By intelligently modeling and enforcing these constraints, CP algorithms can efficiently generate feasible schedules that meet logistical needs and ensure fair play, making it the optimal choice for sports scheduling tasks.

4.2 Solving from Scratch

After determining that our problem could be addressed using constraint programming, we embarked on developing our custom solver. Initially, we explored two approaches: constructing a tree structure and utilizing a 3D matrix to represent scheduling possibilities. With the tree structure, we aimed to encompass all feasible solutions within our problem domain, optimizing our objective function through tree traversal to

identify the most favorable solution. Alternatively, the 3D matrix approach provided a structured framework for evaluating potential solutions, allowing us to navigate through the matrix to assess scheduling options.

Tree Structure

Recognizing the suitability of constraint programming for our task, we chose Python to craft our solver. Our approach centered on utilizing a tree structure to encompass all potential solutions within the problem domain. Each node in the tree represented a specific assignment to a decision variable, with leaf nodes indicating complete assignments and each level corresponding to the assignment of a specific variable. However, as we progressed with coding our custom solver, we encountered challenges. The sheer volume of code required increased the risk of errors, raising concerns about the program's reliability. Additionally, implementing efficient traversal through the tree while optimizing the objective function proved to be a challenging task. These challenges underscored the complexities inherent in developing a solver from scratch, prompting us to consider exploring other structures.

3-D Matrix

At its core, we have 3 decision variables: matches, days and stadiums. Why not use a 3D matrix to represent this? Each dimension of the matrix corresponded to days, matches, and stadiums, respectively. This approach aimed to provide a structured framework for evaluating potential solutions. However, traversing the 3D matrix in a

non-brute force way was very difficult; there doesn't seem to exist traversal algorithms on matrices. The sheer size of the matrix, compounded by the intricate constraints of sports scheduling, made navigating through it a difficult task. Each cell in the matrix required careful consideration of various factors, including team preferences, venue availability, and rest periods. These challenges underscored the complexity of developing a custom solver and highlighted the need for alternative approaches to efficiently address sports scheduling problems.

4.3 CVXPY

We considered utilizing CVXPY, a Python-based modeling tool for convex optimization, to solve our scheduling problem. However, upon closer examination, we realized that while the objective function is continuous, CVXPY would still not be suitable for our problem due to the non-convex nature of the discrete decision variables and potentially other non-linear constraints. These binary variables represent match occurrences between teams on specific days and stadiums, creating discontinuities in the constraints. Additionally, other constraints in the problem may introduce non-linearity, further contributing to its non-convexity. As a result, CVXPY, designed specifically for convex optimization **?**, would not be appropriate for solving our scheduling problem. We recognized the need for alternative optimization approaches capable of handling non-convex problems, prompting us to explore solvers and optimization techniques for non-convex problems.

4.4 Non-Convex Optimization Solvers

After realizing our problem is most likely a non-convex optimization problem, we looked into non-convex solvers. We found that Google Or-Tools (`ortools`) is the most popular packages for constraint programming, so we went with that.

5 Mathematical Formulation

Below we develop a mathematical definition of a constraint satisfaction sports schedule optimizer.

5.1 Variables

Lets define the following variables:

- d_i: Day i of the tournament schedule, where $i \in \{0, 1, ..., D\}$ and D represents the total number of days.

- s_j: Stadium j within the hosting regions, where $j \in \{0, 1, ..., S\}$ and S represents the total number of stadiums.

- t_k: Unique team k, where $k \in \{0, 1, ..., T\}$ and T represents the total number of teams.

- $X(d_i, s_j, t_k, t_l)$: Binary variable indicating whether team t_k plays against team t_l on day d_i at stadium s_j. It takes values in $\{0, 1\}$.

5.2 Constraints

We enforce the following constraints:

1. **Each team plays each other α times:**

$$\sum_{j=0}^{T} X(d_i, s_j, t_k, t) + X(d_i, s_j, t, t_l) = \alpha \quad \text{for all } t$$

For each team t, the constraint sums the occurrences of matches between team t_k and all other teams t_l across all days d_i and stadiums s_j, along with the reverse matches (between t_l and t_k). The total count of these matches should equal α, indicating that each team plays against each other α times throughout the tournament.

2. **Each team plays β times per day:**

$$\sum_{j=0}^{S} X(d_i, s_j, t_k, t) = \beta \quad \text{for all } d_i, t$$

For each team t, the constraint sums the occurrences of matches involving team t_k across all stadiums s_j on the given day d_i. The total count of these matches should equal β, indicating that each team plays β times on that particular day.

3. **Each stadium holds δ matches throughout the whole tournament:**

$$\sum_{k=0}^{T} \sum_{l=0}^{T} X(d_i, s, t_k, t_l) = \delta \quad \text{for all } s$$

For each stadium s, the constraint sums the occurrences of matches between any pair of teams t_k and t_l across all days d_i in that stadium. The total count of these matches for each stadium should equal δ, indicating that each stadium hosts exactly δ matches over the entire tournament.

4. **Each teams plays in a stadium ϵ times:**

$$\sum_{k=0}^{S}\sum_{l=0}^{T} X(d_i, s, t_k, t) = \epsilon \quad \text{for all } s, t$$

For each team t, the constraint sums the occurrences of matches where team t plays in stadium s across all days d_i. The total count of these matches for each team in each stadium should equal ϵ, indicating that each team plays in ϵ different stadiums over the course of the tournament.

5. ϕ **days of rest per team between games:**

$$\sum_{d=i+1}^{i+\phi} X(d, s, t', t_k) + \sum_{d=i+1}^{i+\phi} X(d, s, t', t_l) \leq 1$$
$$\sum_{d=i+1}^{i+\phi} X(d, s, t_k, t') + \sum_{d=i+1}^{i+\phi} X(d, s, t_l, t') \leq 1$$

for each d_i, s_j, t_k, t_l such that $X(d_i, s_j, t_k, t_l) = 1$

for all s and for all $t' \neq t_k, t_l$

This constraint is expressed in two parts:

If a match between team t_k and t_l occurs on day d_i at stadium s, then for each subsequent day d_{i+1} to $d_{i+\phi}$, the sum of matches involving any other team t' against team t_k or t_l in the same stadium s should be at most 1.

Similarly, for each subsequent day d_{i+1} to $d_{i+\phi}$, the sum of matches involving team t_k or t_l against any other team t' in the same stadium s should be at most 1.

These inequalities ensure that there is no more than one match involving team t_k or t_l in stadium s for each day between d_i and $d_{i+\phi}$, providing the necessary rest period between games for the teams involved.

5.3 Objective Function

The objective is to maximize the revenue generated from the games. We define the objective function as follows:

$$\text{Maximize} \sum_{i=0}^{D} \sum_{j=0}^{S} \sum_{k=0}^{T} \sum_{l=0}^{T} X(d_i, s_j, t_k, t_l) \cdot r(d_i, s_j, t_k, t_l)$$

Here, $r(d_i, s_j, t_k, t_l)$ represents the estimated revenue generated by team t_k playing

against team t_l at stadium s_j on day d_i. It is defined as:

$$r(d_i, s_j, t_k, t_l) = \text{capacity}(s_j) \cdot \frac{\text{rank}(t_k) + \text{rank}(t_l)}{2}$$

where capacity(s_j) is the stadium capacity of s_j and rank(t_i) is the rank of team i. The revenue function increases with both the stadium's capacity and the combined rankings of the two teams.

6 Google OR-Tools

The introduction of modern day optimization techniques has allowed for massive breakthroughs within the field of computational optimization. Namely, problems such as the the Travelling Salesman Problem (TSP) are able to be solved up to an exact solution, given enough computational power. Within the domain of constraint satisfaction problems, most of them are traditionally solved linearly through a simple LP or IP framework, however, recent breakthroughs have allowed for more complex and efficient algorithms that can drastically reduce the required runtime, one of which is Google OR-Tools.

6.1 What is Google OR-Tools

Google OR-Tools is an open-source software package that offers efficient implementations of solvers for various types of problems. These problems span the areas of LP linear programming, mixed integer programming, constraint programming (CP), vehicle routing (VRP), and many more. In all of these use-cases, all of the given problems suffer from having an extremely large solution space (set of possible solution), and thus narrowing down the search space is crucial in order to find an optimal (or approximately optimal) solution. Google OR-Tools helps mitigate this issue by offering extremely quick algorithms that help narrow down the search space, without needing to perform a brute force approach **?**.

6.2 Constraint Programming with Google OR

For the purpose of this paper, we are primarily interested in the family of CP solvers. Since our scheduler operates under the assumption of a set of specific constraints, we are focused on finding a set of feasible solutions that adhere to all of the given rules. To do so, Google OR offers two distinct algorithms (each with their own pitfalls): a CP-SAT solver, or a regular CP-solver. For this project, we decided to use the CP-SAT solver for reasons that will be discussed later.

On a high level, the CP-SAT solver is centered around the Lazy Clause Generation approach, which utilizes features from both SAT solving techniques (NP-complete) and finite domain (FD) propagation. The problem is approximated into an SAT-friendly description such that the SAT solver is able to utilize boolean representations of the constraint variables, which are generated by FD propagation. The fine details of this algorithm are omitted due to its extreme complexity, however if interested, the full paper for Lazy Clause Generation can be found here with its corresponding implementation here.**?**

6.3 Limitations of CP-SAT

While CP-SAT is one of the best state-of-the-art methods for CP problems, it has a unique set of limitations when compared to more traditional and straight forward algorithms.

1. Unlike standard LP solvers which can take continuous input variables, CP-SAT can only take discrete values. This drastically restricts the space of problems which CP-SAT can solve. Although there exist workarounds and plug-ins, they are not nearly as efficient as standard LP solvers which handle continuous values.

2. For classic IP problems, CP-SAT generally is slower than existing IP solvers. This is because of the approximations that CP-SAT makes in order to be a general CP solver.

3. Similarly, when given SAT-formulas, CP-SAT is not as fast as dedicated SAT solvers.

Although CP-SAT is bounded by the above limitations, its performance still remains commendable as no known alternate solvers are able to solve all limitations listed above. **?**

7 Pseudo-code

In this section, we outline the general structure of what a Google OR-Tools solver for our mathematical formulation looks like. The implementation details for all of the constraints as well as the revenue function have been omitted since they have already been defined within the Mathematical Formulation section, and they can be found here. The formulation of the 3 days of rest constraint will be shown in the set-up section. The source code takes advantage of Google OR-Tools, which is why on a high level the implementation of the solver is extremely simplistic.

Algorithm 1 FIFA Scheduling Solver

$M \leftarrow Constant$ ▷ Number of matches
$S \leftarrow Constant$ ▷ Number of stadiums
$D \leftarrow Constant$ ▷ Number of days

$m \leftarrow$ cp_model.CpModel() ▷ Initialize CP sovler
matches[(m,d,s)] ▷ Define variables for all M,D,S
m.Add(C1) ▷ Constraint #1 (matches scheduled once, sum of m over all S,D = 1)
m.Add(C2) ▷ Constraint #2 (one match per stadium per day, sum over all m $\leq$ 1)
m.Add(C5) ▷ Constraint #3 (ϕ days of rest between games per team)

revenue $\leftarrow f((d_i, s_j, t_k, t_l))$ ▷ Function of our assignment

m.Maximize(revenue) ▷ maximizing revenue objective function

8 Optimizing the FIFA World Cup

The FIFA World Cup is a football event that has become iconic at a global level. It was initially conducted to bring countries together and encourage people to put aside their differences. The early tournaments were modest in size with a few competing nations. But over time, it has become a global phenomenon becoming the world's most watched event with almost 1.5 billion people watching the last World Cup in 2022 **?**.

It has fostered a sense of global community. There were significant milestone events that have made it so iconic such as England's 1966 victory at home or the dramatic 1970 tournament in Mexico. It has become much more than just a football game as everything from music to culture has revolved around the event. The host selection itself has become its own event as the selected nation can proudly display the best its country has to offer for the world to see. As such, getting selected to host the World Cup is a triumph on its own but along with the economic boost and national pride comes a logistical nightmare to plan and execute **?**.

8.1 2026 FIFA World Cup

The 2026 FIFA World Cup is expected to mark a significant milestone in the history of the tournament. It will not only be expanding from 32 to 48 teams but instead of a single host, it will be co-hosted by the three nations of Canada, Mexico and the United States **?**. It is a part of FIFA's ongoing efforts to further globalize football and leave an everlasting impact on the American market. But with that decision comes

logistical challenges that have never been faced at this magnitude. From coordinating across multiple time zones to widespread travel between games spanning the entire North American continent. Although football is far from the most popular sport in North America, they hope that would drive a further spark of interest so 2026 is a large investment for the future of sport but also the countries **?**.

The tournament is expected to leverage advanced infrastructure and technology to deliver a sustainable and viewer-friendly experience. It will aim to set new standards for how the World Cup can promote unity and innovation **?**.

8.2 The Group Stage

There are a few qualifying rounds that are organized independently by different continental organizations. For the FIFA World Cup itself, the group stage is where the tournament begins and sets the stage for the teams that will move on to the knockout phases. The expansion to 48 teams has introduced a new format with 12 groups of four teams each, where they play against every other team in their respective group in a round-robin format **?**. Points are awarded for each match: three for a win, one for a draw, and none for a loss. The teams in each group are ranked based on the total points accumulated and the top two teams from each group advance to the knockout stages. This format is designed to ensure that each team has the opportunity to play multiple matches and that advancement is based on performance across these games. The group stage not only determines which teams move forward but also sets the tone

for the rest of the tournament **?**.

8.3 Set-up

The group stage teams are determined by random draw from the qualifying teams. The draw is made such that the qualifying teams are ranked in order according to their world ranking, and then each group gets one team from each "level" of rank. For example, if there are 16 teams to choose from, each group would get one team from the ranks 1-4, 5-8, 9-12, and 13-16. This ensures that each group has good and bad teams relatively speaking. So first we created groups in a similar way trying to ensure one team from each relative ranking. We focused specifically on scheduling only the central region matches.

Group 1: Argentina, Brazil, Denmark, Peru

Group 2: France, Croatia, Mexico, Poland

Group 3: Egypt, Belgium, Spain, USA

Group 4: Portugal, England, Morocco, Uruguay

We then created a matrix of all the match-ups for each group.

Knowing that the team formulation had been tried and was difficult to work with for the 3 days of rest constraint, we simplified the problem to work with just the matches

as each group only had 6 possible matches to schedule.

Knowing this, we created a matrix in which each row i contains the matches that cannot be scheduled close to match i. We called this the 'no adjacent matches' matrix.

Then we simply created a matrix M which defines the schedule laid out by FIFA. The rows represent the stadiums and the columns represent the days. Lastly, before running our optimization function we needed to store the stadium capacities in an array to use in our objective function.

We then ran the function for each group and after each iteration We updated the matrix M to get rid of the days which had already been scheduled. This was repeated until all 4 groups had been scheduled. Of course the scheduled matrix only contained the match numbers, so now all that was left was to use the previously generated list of matchups and apply them to the matrix.

8.4 Results

Figure 1 is the final that was generated. Each color represents a group.

	June 11th	June 12th	June 13th	June 14th	June 15th	June 16th	June 17th	June 18th	June 19th	June 20th	June 21st	June 22nd	June 23rd	June 24th	June 25th	June 26th	June 27th
Guadalajara	POR-URU							ENG-URU					POR-MOR			MOR-URU	
Mexico City	DEN-PER						FRA-MEX							FRA-CRO			
Monterrey				POR-ENG						ENG-MOR				EGY-USA			
Houston				EGY-SPA			EGY-BEL			SPA-USA			BEL-USA			BEL-SPA	
Dallas				BRA-PER			ARG-DEN					ARG-PER			ARG-BRA		BRA-DEN
Kansas City						MEX-POL				CRO-POL					FRA-POL		CRO-MEX

Figure 1: Final Schedule

9 Discussion & Alternative Formulations

As can be seen in the example solution that was generated, unfortunately the 3 days of rest constraint did not work as intended. It is unclear to us what the issue could be. Further attempts and improvements could be done by formulating the problem in a different manner entirely. Particularly this problem could be formulated as an **assignment** problem rather than a scheduling one.

- Groups of workers (in this case teams) get assigned tasks (in this case stadiums)
- Assigning a team to a stadium has a cost which we try to minimize.
- Assign each team a starting stadium with cost 0. Each following stadium assignment has the travel distance associated as the cost.
- Minimize the total travel distance for each team.
- Ensure that a team does not stay at one stadium by adding a constraint that a team cannot play in the same stadium consecutively
- Another way to do this is by adding a constraint that each stadium must be played in at least once by every team (this could severely reduce the feasibility depending on the set schedule)

Another way this can be formulated is a graph colouring problem because of the 3 days of rest constraint.**?**

10 Conclusion

In conclusion, this paper has showcased the effectiveness of using constraint satisfaction methods to optimize sports schedules, particularly focusing on the FIFA World Cup as an example. Through the application of constraint satisfaction techniques, we've established a framework for designing schedules that consider factors like revenue, fairness, and logistical practicality.

The insights gained from this study aren't limited to the FIFA World Cup alone; they extend to other major sporting events as well. These findings offer valuable strategies for event organizers to create schedules that are both efficient and fair, benefiting teams, venues, broadcasters, and fans alike.

Looking forward, this research opens doors for further exploration and refinement of scheduling methods. By continuously improving our understanding and application of optimization techniques, we can enhance the planning and execution of future sports events. Ultimately, this progress holds the promise of delivering tournaments that are more enjoyable, equitable, and smoothly organized for everyone involved.