Summary

Creating schedules and leagues for sports in a fair manner on an international scale require many factors to consider. Which is why it can be hard to develop. Hence, the IMMC thought it would be a good idea to use a mathematical model to help create said schedule.

In task 1, I first zeroed in on a sport to operate with. The selected sport was basketball as it has an established international presence, allowing the next step to be practical. I also had to select 20 teams from around the world (at least 2 from each continent). In order to ensure fair matchups, the sport would have to be internationally recognized such that the selected teams from around the world will have the valor to compete with each other. To further epitomize the fairness of the league, I decided to only pick teams from around the world which were at least top 10 from their respective leagues. After identifying the teams, I had to determine relevant factors that would have an impact when scheduling these games. The final selected variables were as follows, travel distance, number of games played, equitable matchups, venue availability, and game streaming time. Subsequently, after establishing the variables to work with, I emphasized their importance and their respective expressions (calculations to attain the variable), which was then processed to produce constraints for the model – for example each team should play exactly 38 games in the 20 team league.

In task 2, I assigned dynamic weightages to each variable using optimization achieved by differentiation. I then developed an objective-function (the mathematical model) which was then used to draft the season schedule with the 20 teams. After producing an output, I checked the output against the constraints identified in task 1, as well as their compliance with the desired outcomes of the project, that being fairness in competition, sustainability, and geographical representation.

Moving on, in task 3, I expanded the league by adding 4 more teams to the array of teams participating in the league. This way I could test the real-life applicability of the model to real world situations as I would be able to see how it adapts to a new environment (with 24 teams in the league now). After implementing the same model to the 24 teams, it produced another season schedule. I then checked its practicality by seeing its compliance to the constraints and desired outcomes of the project again. After, by comparing the output of 20 teams to 24 teams, I concluded the plausible impacts of the variables on the model's performance. Finally, in task 4, I generalized the model by implying how the model could be refined or manipulated to create schedules for other sports.

Dear Decision Makers of the IMMC-A,

Re: Recommendation on GSL scheduling

I am writing to inform you about our model on producing a sport schedule effectively, thank you for your confidence in our my model, and I am determined to produce a detailed model on how it can solve this issue.

After comparing the outputs of the model from the 20 team league and 24 team league, I was able to corroborate tha this model is able to effectively create a schedule with equitable matchups, compliance with sustainable implications (reducing carbon emissions), and assigning matchups to the given venues.

For the model, I considered the travel distance of the teams, the number of games played, the game streaming time, the minimum number of rest days for every team, and venue availability. I believe that these are all the relevant factors that can affect league scheduling which is why I picked them out of the plethora of potential variables researched. Hence, I believe since my model has a reasonable compatability with these variables, the model can be used in real-time to draft schedules.

On the other hand, my personal bias may hinder my evaluation of my model. Which is why I request the expertise of the IMMC-professionals to assist this cause by evaluating my model. This way the model's applicability can be concluded with more credibility.

Furthermore, my observations on the model's performance was done by testing the output with a success criteria (that being the constraints defined initially). In consequence, I believe the IMMC-professionals will be able to examine the model through a different and compelling perspective.

To conclude, my model is able to manifest schedules given certain variable inputs, though its performance is not flawless. It has some confusion and trouble with adhering to the game-streaming times specifically, as well as some confusion with the rest day constraints. For easily-comprehendable information about the model view figure 12 which is the visual representation of the results drafted by the model.

Thank you for your time and efforts in reading this.

Best regards.

Contents

1 GSL Basketball Scheduling

1.1 Question Analysis and Restatement

We are asked to select a team sport (which can be played with at least 5 players) that the IMMC can use to organize at an international scale. Furthermore, we are required to select at least 2 teams from each continent to participate in this league excluding Antarctica, and there must be a total of 20 teams to participate. The teams must be selected from a justified criterion. The league will last around 8 to 9 months and the IMMC wants us to create a model that can create a fair games schedule.

The approach to solving this problem:

1. Find out relevant factors that affect game scheduling for basketball.
2. Choose at least 5 relevant factors to consider.
3. Define the nature of each variable then synthesize each variable to produce a model.
4. Generate the scores for each variable and amalgamate these scores to determine a league schedule.

1.2 Defining the variables

1.2.1 Preliminary Research

There are a plethora of variables that can influence the scheduling of a season for basketball. The following is a handful of those variables:

1. **Game streaming time**: To showcase the cosmopolitan structure of the league, it is essential that viewers from around the world can view the matches, each match must commence at an optimal streaming time [1].
2. **Travel distance**: Given the circumstances of today's world, it is imperative that travel distance for teams is considered, as larger distances traveled can lead to more carbon emissions sourced from their journeying [2].
3. **Number of games played**: The number of games played in a season need to be equal for each team to ensure fairness in the schedule [3].
4. **Number of rest days**: For a fair schedule, both teams must be given fair circumstances like having a minimum number of rest days before their next game for recovery [4].

5. **Equitable matchups**: The game schedule must comprise of matchups that are fair for each team [5], meaning that all teams must play against weaker and stronger teams as per their ELO rating.

6. **Venue availability**: The venue at where each game is played has to be available otherwise the games cannot be played and would require rescheduling at another site, leading to more travel distance [6].

7. **Fatigue levels**: After playing games the players could have a varied but present level of fatigue [7] which can contribute to unfairness if asked to play without enough time to recover or asked to play against a very strong team.
8. **Minimum number of days of the season**: If the season is too long it may not incentivize viewers to watch certain league matches which can result in a decrease in revenue for the league [8].

There were other variables to consider like the difference in time zone for teams which can contribute to tiredness that can make the game unfair. However, ensuring that the teams receive a minimum number of rest days negates this affect or keeps it latent.

1.2.2 Selecting the variables

After doing preliminary research and listing potential variables to consider for game scheduling, the following is the finalized list of the variables that will be utilized:
- Game streaming time
- Travel distance
- Number of games played
- Number of rest days
- Equitable matchups
- Venue availability
- Total number of days of the season

The reason why fatigue levels were not picked is because they can get partially alleviated by the rest days that the teams receive [9]. Furthermore, it is tough to quantify fatigue levels for teams because there is no credible data which can represent even the average fatigue levels for each and every team from the league.

It is important to note that the variable data types vary, as travel distance has a defined integer value, venue availability has a binary value. Thus, each variable will be quantified and processed so they can be inputted into the mathematical model.

Before moving on to quantifying the variables the teams selected for the league is established as the following:

North America	Orlando Magic (USA), Minnesota Timberwolves (USA), and Los Angeles Lakers (USA)
South America	Flamengo Basketball (Brazil), Boca Juniors (Argentina), Minas (Brazil)
Asia	Lions (China), Alvark Tokyo (Japan), Seoul SK Knights (South Korea)
Europe	Fenerbahçe Beko (Turkey), Barcelona (Spain), Olympiacos (Greece), Bayern Munich (Germany), Crvena Zvezda (Serbia)

Africa	Al Riyadi Club (Lebanon), Petro de Luanda (Angola), Cape Town Tigers (South Africa)
Australia	Hawks (Australia), Melbourne United (Australia), Wildcats (Australia)

Criteria to select these teams:
Each team was picked such that they are at least top 10 from the respective leagues that they participate in. This ensures that the skill rating (ELO) for each team is relatively close which translates to more equitable matches (the ELO ratings for each team will be defined next).

1.2.3 Quantifying the variables

Quantifying the team strength - As mentioned before, it is important to quantify each variable and define numerical data that will be used. Firstly, we can start with the team ELO calculations which are basically the team's skill rating (to measure the strength of the team). Each team was picked to be from the top 10 of the leaderboards of the respective country's' league. Subsequently, as there are no defined skill ratings of each individual team, we must make them ourselves. To start off, we will take the base ELO rating by using the country's FIBA (the world's most credited international basketball organization) team rating and quantifying it for each team. To simplify data processing for the mathematical model in the near future, we will take the integer value of the FIBA ratings. The following is the calculation for the ELO rating where F is the country's FIBA rating:

$$ELO_x = 1000 + (F - 100)$$

Here is an example of the calculation which is used for the Orlando Magic:

$$ELO_{Orlando\ Magic} = 1000 + (840 - 100) = 1740$$

As the Orlando Magic play in the American league 'NBA', the ELO rating will be calculated by using USA's FIBA rating. The following table shows the calculated ELOs for each of the 20 teams selected for the league:

Team	FIBA rating	ELO
Orlando Magic, Minnesota Timberwolves, and Los Angeles Lakers (USA)	840	1740
Flamengo Basketball and Minas (Brazil)	673	1573
Boca Juniors (Argentina)	733	1633
Lions (China)	397	1297
Alvark Tokyo (Japan)	529	1429
Seoul Knights (South Korea)	236	1136
Fenerbahçe Beko (Turkey)	430	1330
Barcelona (Spain)	746	1646
Olympiacos (Greece)	659	1559
Bayern Munich (Germany)	757	1657
Crvena Zvezda (Serbia)	761	1661
Al Riyadi Club (Lebanon)	409	1309
Petro de Luanda (Angola)	382	1282
Cape Town Tigers (South Africa)	66	966

Hawks, Melbourne United, and Wildcats (Australia)	733	1633

Each FIBA rating was taken from the official FIBA website [10].

Quantifying the optimal streaming time – The time to stream these basketball games need to align with the optimal times internationally such that the consumer base of the league can be maximized. Hence after researching the methodology explained behind the NFL's strategic scheduling on Netflix [11], I arrived at the three optimal times being Y_3= {8:00PM UTC, 4:00AM UTC, and 1:00PM UTC}.

Quantifying the variables – To quantify the variables established from before, we need to determine supplemental variables which will be used to calculate and produce the main factors. The following are the supplemental variables defined which will be used to calculate the main variables:

Variable	Definition
Game Schedule	$G_{i,j,t} = \{\begin{matrix} 1\ if\ team\ i\ plays\ team\ j\ on\ day\ t \\ 0\ otherwise \end{matrix}$
Home/Away status	$H_{i,t} = \{\begin{matrix} 1\ if\ team\ i\ has\ a\ home\ game\ on\ day\ t \\ 0\ otherwise \end{matrix}$
Recovery time	R_{min} = Minimum 1 day recovery time required
Average ELO	$Average\ ELO = \frac{\sum_{i=1}^{20} S_i}{20}, S_i = ELO\ of\ team\ i$

Now some of the main variables can be directly quantified, while others need to be calculated using the supplemental variables, the following are the main variables which can be directly quantified:

Variable	Definition
Travel Distance	$D_{i,j} = distance\ between\ team\ i\ and\ j$
Venue availability	$V_{k,t} = \{\begin{matrix} 1\ if\ venue\ k\ is\ available\ on\ day\ t \\ 0\ otherwise \end{matrix}$

Now we must define the constraints using all the variables:

1. Total number of games played in the season

$$\frac{\sum_{i=1}^{N} \sum_{j=1,j\neq i}^{N} G_{i,j}}{2} = Total\ number\ of\ games\ played\ in\ season$$

N = number of teams in the league

This league has 20 teams, and each team will play every other team twice to ensure a fair matchup, here is the calculation for the total number of games played in the season:

$$\frac{\sum_{i=1}^{N} \left(\sum_{j=1,j\neq i}^{N} \left(G_{i,j}\right)\right)}{2} = 380\ games$$

Firstly, the inner sum takes the sum of the number of times two teams play each other, then the outer sum takes the sum of the number of times each team plays each other twice, which is why we must divide the value by 2 to arrive to the total number of games played in the season. Despite using this variable for further calculations, it helps us understand the longevity of the season. Consequently, this longevity highlights potential operational costs which would source from managing the season for a certain time period.

2. Distance traveled constraint

$$\sum_{i=1}^{N} \left(\sum_{j=1,j\neq i}^{N} (D_{i,j} \times 2)\right) \leq Max\ travel\ cap = 200{,}000km$$

Over here, we first multiply the distance traveled between the two teams i and j with the number of times the two teams play each other (twice as determined previously), then we sum up the distances traveled for all opponents with team i, and then finally sum up the total travel for all the teams in the league.

3. Rest days constraint

$$G_{i,j,t} + G_{i,j,t+1} \leq 1, \forall\{\mathrm{i,j}\}$$

This constraint specifies that if team i plays team j on day t, those teams should not play on day t+1 as the sum of G should be less than or equal to 1. This ensures that each team gets at least 1 day of rest before another game. In turn, it ensures that teams have enough time to recover before playing another game.

4. Minimum number of days of the season

$$T = 95 + (95 \times R_{min}) = Minimum\ number\ of\ days$$

If we consider 4 games will be played per game day, out of 380 total games, we will have 95 match days. Then, multiplied by the minimum number of rest days, being 1, and added to the number of match days, the season could have a minimum span of 190 days. Like the number of games, the minimum number days estimated for the season helps understand the longevity of the season.

5. Venue availability constraint

$$G_{i,j,t} \leq V_{k,t}, \forall\{\mathrm{i,j,t}\}$$

Over here, the game G can only occur (value of 1) if it is equal to the venue when it is available (value of 1). Basically, if a game is occurring (G will be 1), and if the venue is not available (V will be 0), this constraint will not be satisfied and hence would have to reschedule the game for another venue or timing.

6. Equitable matchups constraint

$$\sum_{t=1}^{T}\left(\sum_{j=1,j\neq i}^{N}(S_j \times G_{i,j,t})\right) = Average\ ELO\ \approx 1523$$

As per this constraint, the average ELO should range close by 1523 in order to ensure a fair amount of strong and weak matchups for each team. This is the average strength estimate which can be used as a base to compare the mathematical model's production of a strength report to see whether it was accurate or not. We would be able to see how dispersed the strength rating would be when produced by the mathematical model.

7. Streaming time constraint

$$Y_t \in Y_3$$

This constraint tells us that each game should occur at any one of the times, determined before, to ensure international exposure of the games. Again, streaming at an internationally desired time frame, more people from around the world will be able to view the games. Ultimately expanding the consumer base of the league.

Quantifying the traveling distance – To optimize the traveling distance in such a way that we can minimize carbon emissions from teams, we must use the Haversine formula to calculate the exact distance between the countries [12]:

$$d = 2R \times \arcsin\left(\sqrt{sin^2\left(\frac{\Delta\partial}{2}\right) + \cos(\partial_1) \times \cos(\partial_2) \times sin^2\left(\frac{\Delta\gamma}{2}\right)}\right)$$

R = Earth's radius

$\partial_{1,2} = lattitude\ points\ in\ radians$

$\gamma_{1,2} = Longitude\ points\ in\ radians$

By computing each distance value (refer to code B) I found out optimal locations for hosting the games. The following table shows the distances between the countries where each team would source from:

Travel Distances to Host Locations

	USA	Spain	Turkey	China
Brazil	7,317 km	7,850 km	10,757 km	16,633 km
Argentina	9,018 km	10,671 km	13,230 km	18,899 km
Japan	10,150 km	10,649 km	8,528 km	3,047 km
South Korea	10,743 km	10,185 km	7,790 km	2,118 km
Greece	9,409 km	2,184 km	1,158 km	7,014 km
Germany	7,861 km	1,615 km	2,355 km	7,224 km
Serbia	8,976 km	2,068 km	1,310 km	6,846 km
Lebanon	10,669 km	3,555 km	571 km	6,106 km
Angola	12,841 km	6,161 km	5,862 km	10,408 km
South Africa	14,399 km	8,362 km	7,835 km	11,237 km
Australia	15,184 km	15,754 km	12,441 km	7,474 km

Figure 1 - Table of distances between countries (Source made by candidate).

To help visualize why these host countries make sense, here is an image of the connections between each country to justify those host locations:

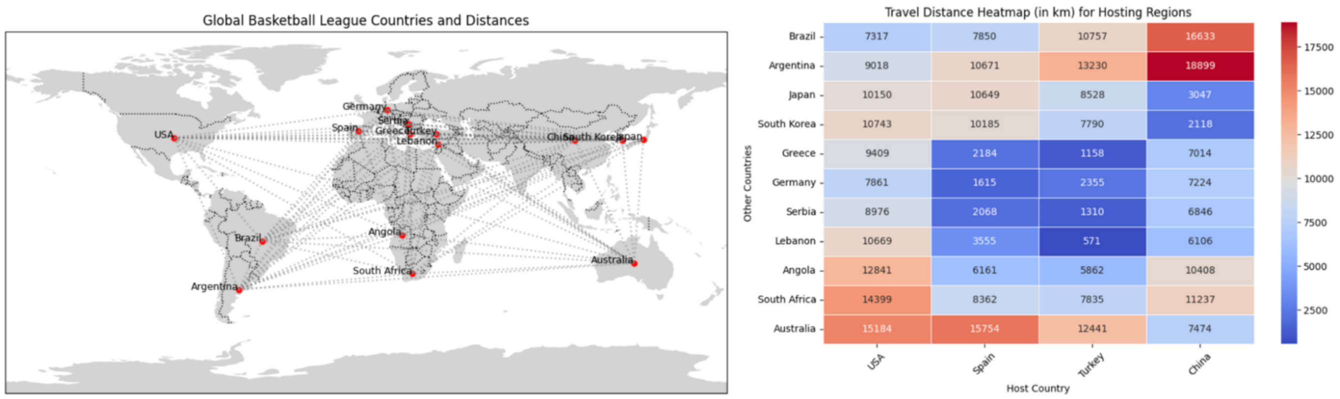

Figure 2 - Map connections and distance heatmap (Source made by candidate).

As shown by the heatmap, the European countries seem to be the closest to every other country which is why of the chosen host locations to consider, Spain or Turkey make the most sense to be selected as host countries for this season (for reference on generation of these materials, visit code A and C). More specifically, the Abdi Ipekci Basketball Arena in Turkey, and L'alqueria del Basket in Spain.

Assumption 1

We assume: **3 liters of fuel would be burnt per km. The following equation showcases the calculation made for carbon emissions:**

$$CO_2 = Distance\ (km) \times 3 \times 3.15$$

By using the carbon dioxide emission equation (refer to code D), the following results are drawn by keeping Spain or Turkey hosts of the season, which in turn would allow alternating matches between either Spain or Turkey. These calculations assume traveling from the respective countries (Eg. America to Turkey) which in turn produces a larger carbon emission value compared to if we permitted the teams to cluster around the area.

Total Travel Distance & Carbon Emissions for Host Locations

	Total Distance (km)	Total CO_2 Emissions (tons)
Spain	197,487 km	1,866 tons
Turkey	182,562 km	1,725 tons

Figure 3 - Host country carbon emission estimates (Source made by candidate).

As per UEFA, in the famous UEFA champions league tournament in 2023, it was reported that 21,844 tons of carbon emissions were reported [13], whereas the maximum projected carbon emissions for this season cap at 1,866 tons. Ultimately illustrating how much more sustainable this scheduling will be. In consequence, from the table we can conclude that the maximum distance travel cap should be roughly 200,000km to be safe.

Trade-offs – Having to balance out and consider all these variables when deciding a game scheduling can have its set trade-offs. Previously, we established that the host countries will alternate between Spain and Turkey due to geographical and environmental benefits. That being, the carbon emissions are reduced when Spain and Turkey are kept as hosts of the season, furthermore, as shown by the map in figure 2, they are relatively close to the center of all the connections between the countries, indicating that they will have less of a time zone difference with the rest of the world. As a result, this improves the game streaming time as more people will be inclined to watch the games. Conversely, this raises a tradeoff, some teams participating in the league are from Turkey and Spain. This gives them an advantage over the other teams as playing in your arena and home country has been proven to have benefits [14]. Furthermore, as geographical recognition is important for the league, having the season hosted in the same two places every time may deprive the league from experiencing cultural diversity. Projecting a sense of cultural diversity in your league by changing the host country could help expand you consumer base as well as promote global cooperation [15]. As your viewers see the culture and nature of another country their perceptions about that country may change to a better perspective. Also, people around the host country may have easier access to visit the games via traveling as compared to if the host location was far away from them.

Another major tradeoff is the longevity of the season. As shown in calculations from before, the minimum number of days for the season to last, given that each team plays every other team twice and has at least 1 day of rest before their next game, is 190 days. Even though other leagues last a relatively equivalent length, like the NBA lasts around 174 days, most of these leagues are often national [16]. Whereas this league is international, thus having a longer game schedule means teams will be traveling around for a longer time which does not help the environment given the carbon emissions which will be released through the process of journeying.

1.3 Variable weightages

After determining the variables and the constraints, the next step would be to assign appropriate weightages to each variable. When inputing the variables into the objective-function, each input should have a weightage that resonates its importance in the calculation [17]. Without these weightages, the input of less important variables might skew the results of an output. As a result this could lead to the development of a schedule which does not minimize the carbon emissions or more.

Before we assign weightages to certain variables, it is important to view the heirarchal structure of the model and relations between each variable and constraint. Figure 4 below showcases the relationships between variables and how the model will decide which games can take place and which have to be changed.

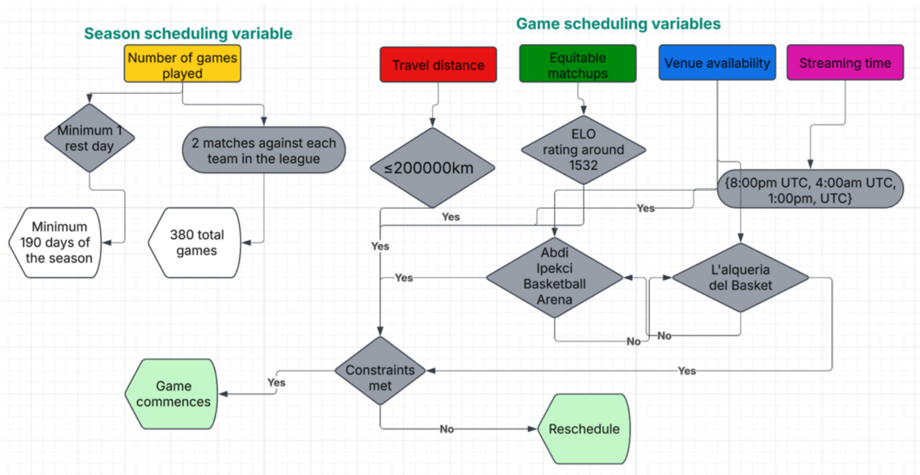

Figure 4 - Variable relations and model output structure (Source made by candidate).

In order to establish the weightages of the variables, we must first understand the most desired outcome from the scheduling. The most apparent outcome is the sustainability of the schedule – meaning that the travel distance must be optimized such that we reduce the carbon emissions as much. Therefore, we must assign the travel distance to have the greatest weightage of 0.3. After that, the rest of the desired outcomes lay equally within the means of ensuring fair matchups, thus the rest days, equitable matchups, and venue availability will share a weightage of 0.2. Finally we reach the final variable being the streaming time. This variable is in place to increase the exposure of the league on an international scale, however; this is one of the least important outcomes, which is why it will have a weightage of 0.1. The following matrix displays the weightages in a presentable manner in figure 5 (refer to code E for source).

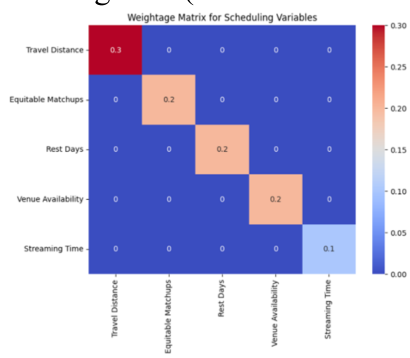

Figure 5 - Variable weightage matrix (Source made by candidate).

1.3.1 Matrix evaluation and improvement

The above is a basic weightage allocation which was done based off the estimated importance of each variable. Consequently, it may conjure errors in the outcomes. Since the data types are not same (some are binary while some are integers), the relationship between the variables may not follow a linear pattern which is instantiated by the matrix in figure 5. Hence, we must employ differentiation to optimize the weightage allocations.

Before we proceed with weightage allocation optimization it is important to establish the mathematical model that will be used to generate the season schedule. The expression below links the variables established from before to mathematically model the relationships from figure 4, it will be used to create the games schedule which will be showcased later.

$$Z(w_1, w_2, w_3, w_4, w_{5,}) = w_1 \sum (D_{i,j} \times G_{i.j.t}) + w_2 \sum (|S_{i,j} - 1532| \times G_{i,j,t}) + w_3 \sum (R_{i,t}) + w_4 \sum (V_t) + w_5 \sum (T_{i,t})$$

From the equation above, w represents the weightages for each variable. Now to adaptively optimize the weightages from this objective function, we can use differentiation of each weightage with it's respective variable. The following expression is an example of the differentation of a weightage.

$$\frac{dZ}{dw_1} = \sum (D_{i,j} \times D_{i,j,t})$$

Using the differentiation for each weightage, we can adaptively optimize the weightages as the number of days (iterations) increase by employing a generalized and common alpha value to multiply. The following expression represents the adaptive calculation for the weightage.

$$w_x^{new} = w_x - \alpha \frac{dZ}{dw_x}$$

Over here α is the alpha value of 0.01. The following diagram (figure 6) shows the gradient change of the weightage values as the number of iterations increase. This shows us how the weightages are adaptive as the day number increases (refer to code F for source).

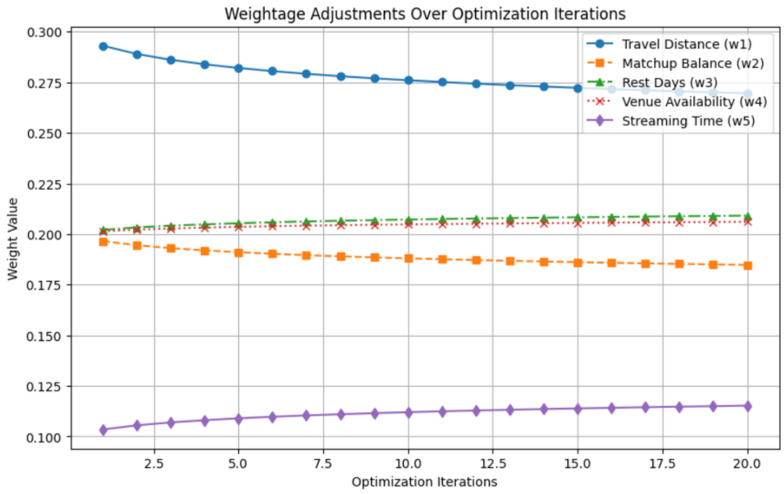

Figure 6 - Adaptive weightage diagram (Source made by candidate).

2.0 Model Evaluation

Now I will introduce the objective function (mathematical model) that will be used to create the basketball games schedule. This function comprises of the variables along with their allocated weightages.

$$Z = w_1 \sum (D_{i,j} \times G_{i.j.t}) + w_2 \sum (|S_{i,j} - 1532| \times G_{i,j,t}) + w_3 \sum (R_{i,t}) + w_4 \sum (V_t) + w_5 \sum (T_{i,t})$$

The initial weightages were mentioned before {0.3,0.2,0.2,0.2,0.1}. The weightages will be optimized side-by-side when developing the schedule.

2.1 Model Output

Figure 7 below shows a snippet of the outputed schedule created by the mathematical model.

```
⚠️ Warning: The model did not solve optimally. Check constraints.
Scheduled Games:
Day 0: Orlando Magic vs. Wildcats
Day 0: Orlando Magic vs. Los Angeles Lakers
Day 0: Minnesota Timberwolves vs. Crvena Zvezda
Day 0: Minnesota Timberwolves vs. Melbourne United
Day 0: Minnesota Timberwolves vs. Fenerbahçe Beko
Day 0: Barcelona vs. Wildcats
Day 0: Barcelona vs. Lions
Day 0: Olympiacos vs. Orlando Magic
Day 0: Bayern Munich vs. Al Riyadi Club
Day 0: Bayern Munich vs. Melbourne United
```

Figure 7 - First model's output (Source made by candidate).

In the code, when implementing the mathematical model to the constraints, I added an error message to be printed - which is shown by the fist line - incase the constraints where not optimally met. As shown by the output, it displays various matches taking place on day 0 with teams playing more than once. This indicates that the rest day and total number of games played constraints were not met. Consequently, we are able to conclude that these constraints caused confusion in the model. Thus to fix the model we must change the parameters of the rest days to ensure more than just 190 days minimum of playing are there in the season such that the model can successfully output a valid schedule.

2.2 Model tuning

To fix this issue, we are going to have to ease some of the constraints. For the model to work, the number of days that the season runs on for will increase from 190 to 220 days. This means more rest days as well, accordingly, the model has more flexibility when allocating games. Another major change in the constraints is the fact that there does not have to be a rest day, though it is ideal. As per the requirement of the outcome, each team should have a fair number of equitable matches, hence, without a rest day constraint teams have a greater number of matches with gives them a better chance to prove themselves when running for the lead in the league. After these changes, the objective-function does not change only the constraints change.

Below shows a snippet of the newly developed league schedule (refer to code H for the tuned model) with the adjustments to the model (in figure 8).

```
Model solved successfully.

 League Schedule with ELO Ratings:
 Day                 Team 1  ELO 1                  Team 2  ELO 2  Avg ELO
   0          Orlando Magic   1740 Minnesota Timberwolves   1740   1740.0
   0          Orlando Magic   1740        Seoul SK Knights   1136   1438.0
   0 Minnesota Timberwolves   1740         Petro de Luanda   1282   1511.0
   0 Minnesota Timberwolves   1740         Fenerbahçe Beko   1330   1535.0
   0              Barcelona   1646            Boca Juniors   1633   1639.5
```

Figure 8 - Snippet of tuned model output (Source made by candidate).

As shown by the snippet, the schedule allows more than 1 game in a day now, does not necessarily implement the 1 day rest constraint (to improve model fluency). To add on, if constraints were met successfully (220 days of the season, etc.) then it should print 'Model solved successfully.'. As shown by the first line, it indicates that the model was able to allocate time slots for each match on the day. For access to full schedule outputted visit source [18].

2.3 Output evaluation

From the snippet of the tuned model's output (in figure 8), we can see a major limitation which is the fact that the model does not conform to the original constraint of scheduling games at the universally picked time slots. As a result, the venues will be practically used for basketball games for the league for almost the whole day. On the other hand, this raises an optimistic economic benefit. As the venues would be used for the whole day, tickets could be sold as day passes, allowing viewers to have a whole day experience. Subsequently, this would result in more cost-effective tickets which would be sold to consumers.

Moreover, when we inspect the snippet of the schedule, we can see that for matchups with similar ELOs like the first matchup, the skill difference is same so it is considered as a fair matchup. From before, the constraint drafted was that the ELO ratings should range close by the base level rating calculated as 1532. Looking into the average ELO column from the table, we can see that this constraint is met successfully. Thus, we can conclude that the model was able to accomplish fair and equitable matchups in the schedule.

In terms of the sustainability and geographic implications, it was already established from before that each match will either take place in the Turkish stadium, or Spanish stadium. Therefore, we can conclude that the initial calculation for the net or estimated carbon emissions are met given that the number of games in the schedule remain to be 380 games. In consequence, this model was able to successfully adhere to the environmental constraints placed initially which optimizes the environmental impact by reducing the carbon emissions.

Finally, after producing the schedule, a feasible and reasonable number of games each team will play is 38. This corroborates that each team will get a fair number of games to play in the season to prove themselves as the better team.

2.4 Play-offs

After the league matches are over (the scheduled matches finish), the leage will take the top 8 teams from the leaderboard and place them in a typical play-off structure to compete for the championship [19]. Below, in figure 9, there is a representation of the play-off structure.

Sample Leaderboard

Position	Team
1	Los Angeles Lakers
2	Crvena Zvezda
3	Orlando Magic
4	Olympiacos
5	Al Riyadi Club
6	Minas
7	Cape Town Tigers
8	Hawks

Figure 9 - Play-off structure representation (Source made by candidate).

3 Practicality of the model

3.1 Introduction

To test the performance and adaptibility of the model, we must extend its boundaries and see its practical use. To do so, 4 new basketball teams will be added ot the array of teams to see how the model is able to create a new schedule. Keep in mind that the objective function is not changing, we are only adding 4 new teams with their respective ELO ratings to the array of teams to be processed by the model. The criteria of scouting the 4 new teams remains the same, they must originate from the top 10 of their respective league leaderboard in their countries – this ensures more fair skill-wise matchups. From the 20 teams selected before, Europe had the greatest number of teams playing in the league. Hence the 4 new teams will come from North America, Australia, and Asia. The following table shows the 4 new teams that will be added and their ELO ratings.

Team	FIBA rating	ELO rating
Houston Rockets (USA)	840	1740
36ers (Australia)	733	1633
Brave Dragons (China)	397	1297
Denver Nuggets (USA)	840	1740

3.2 New constraints

As per the original conditions with 20 teams in the league, there were a total of 380 games in the season which would comfortably span out in 220 days. Given that there are now 24 teams which will play every other team exactly twice in the season (to ensure fairness and competitiveness), the season will consist of 552 games in total. As this is a 45% increase from the previous total number of games, we can increase the total number of days for the league to 350 days.

3.3 Model performance

After inputting the new array of 24 teams into the same model from before and changing the constraints to 522 games with 350 days, the model was able to succesfully output the schedule. Figure 10 below shows a snippet of the schedule, for a closer look view source [20].

league_schedule_24_teams_with_venues

	Team 1	ELO 1	Team 2	ELO 2	Avg ELO	Venue	Total Travel Distance (km)
2	Fenerbahçe Beko	1330	Brave Dragons	1297	1313.5	Turkey	6848.747453917430
3	Olympiacos	1559	Melbourne United	1633	1596.0	Turkey	15420.167753407400
4	Crvena Zvezda	1661	36ers	1633	1647.0	Turkey	15748.092705451100
5	Olympiacos	1559	Hawks	1633	1596.0	Spain	20063.774348858600
6	Wildcats	1633	Hawks	1633	1633.0	Turkey	29184.434821278700
7	36ers	1633	Petro de Luanda	1282	1457.5	Turkey	20353.556871275000
8	Alvark Tokyo	1429	Olympiacos	1559	1494.0	Turkey	9610.291535481570
9	Lions	1297	Bayern Munich	1657	1477.0	Spain	11117.055902011000
10	Olympiacos	1559	Wildcats	1633	1596.0	Turkey	15420.167753407400

Figure 10 - Snippet of models performance on 24 teams (Source made by candidate).

As shown by the table, the model performed even better than the last input with 20 teams. The table shows that the number of teams playing twice on the same day has decreased by a lot. This accentuates the theory that the model works better and is less confused when there are more days in the season, as this gives more flexibility to the model. In terms of the faireness of the matchups, we can see from the average ELO ratings that the values only fluctuate from the base value of 1532 by roughly ±100 points. Otherwise, as shown by the first row, matchups with ELO ratings of a greater dispersion than ±100 points are paired with teams of similar rating, which confirms the matchup will be fair as well. As a result, we can conclude that for the 24 teams, the fairness in the matchups were succesfuly achieved by the model. Furthermore, by examining the travel distances outputted, we can see from the schedule produced that none of the travel distances exceed the initial constraint of a maximum travel distance of 200,000km. Hence it reassures us that the model produced a sustainable schedule. This was achieved due to the fact that the location being Turkey or Spain was selected in the beginning after finding the optimal location through the geographical distance calculation. Once again, as the model shows, multiple games take place on the same venue (on the same day) which then indicates to us that it does not follow the array of timings decided which were the ideal streaming times for enhancing geographical representation of the league as more people from around the world would view the league. On the flip side, as mentioned before, this provides an economic advantage for consumers as they could buy a day pass to see all the games in one venue instead of just one – making their ticket cost effective.

3.4 Impact on play-offs

Firstly, the total number of games played for the season has increased to 552 games, and the number of days of the season has stretched to 350 days. Therefore, after the 552 games, only then can the play-offs start. This means that after each team plays their 46 games, they will be placed on the leaderboard to see which top 8 teams will get selected for the play-offs in the season (the play-offs structure follows the same shown in figure 9). Inevitably, the season including regular games, and the play-offs will span out to be more than 350 days which can take up to a year. This complements the geographical representation of the league because it has continuinity for roughly a year, which means more people from around the world will be engaged with the league for a long period of time, hence resulting in the culmination of a devoted and loyal fanbase. On the contrary, if the season lasts a year, the sequent season would have to be introduced later on as there cannot be a fixed date that the season starts every year because the players will need some sort of a break. Accordingly, it causes inconsistency with the starting and ending dates of the season in general.

3.5 Impact of constraints on model

Firstly, lets explore the constraint which impacted the model the most. That was the number of rest days constraint. As we started, in the variables section, the minimum number of rest days that were defined were 1 for each team. Though after implementing the model to the league with 20 teams and then 24 teams, that constraint was violated both times. If we plot the functions of the strict rest days constraint as well as the relaxed rest day constraint, we can use integration to calculate the area in between the curves which will tell us the extra number of days in the season due to the relaxed rest days condition. Figure 11 below shows us this area.

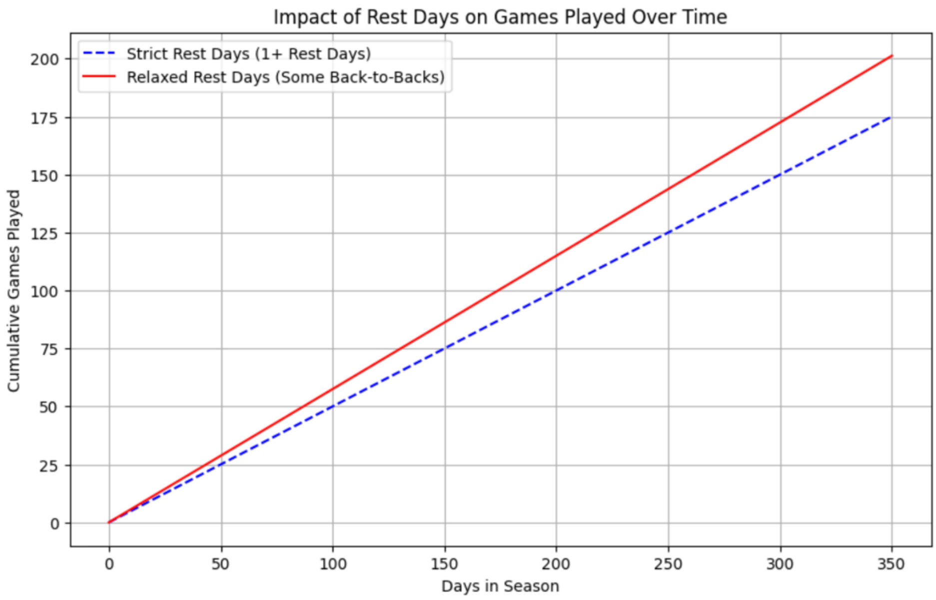

Figure 11 - Fixed and relaxed rest days functions (Source made by candidate).

The blue dotted line represents the restricted rest day constraint which was determined in the variables section. The red line represents the function of the relaxed rest days. Using integration from the following expression we can calculate the approximate number of extra days in the season with relaxed rest days.

$$G(t) = \int_0^t \frac{1 + B(t)}{1 + R_{min}} dt$$

G(t) represents the number of games played, B(t) represents the number of back-to-back games. After implementing this equation on the graphs above, we find that an estimated total of 26 extra games were played in the season when the model had relaxed rest days. This proposes a limitation to the model. While the relaxed rest days provides room for flexibility and comfortable scheduling, it leaves room for error of compliance with the constraint that there should be 380 games or 552 games exactly.

Looking into the travel distance constraint, after both model outputs (1 for the 20 teams and 1 for the 24 teams), the model seemed to comply with the constraint of having a travel distance equal to or less than 200,000km. This suggests that the travel distance constraint is both weighted well in the objective function and adhered to properly by the function.

One other factor that seems to impact the model is the number of games in the season. When the model was tested with 20 teams, it happened to schedule teams to play twice in the same day which can be impactical. However, when 24 teams were introduced and the number of days for the season extended from 220 to 350 days, the model was able to schedule in such a way that the teams did not play twice in the same day.

4 Generalization

This model optimizes the schedule of a basketball league with 24 teams from around the world. It is done by considering various factors that affect scheduling such as travel distances, number of rest days, etc. Although, this model could be implemented to generate the schedule of another sport as well. To do so, some new factors need to be considered when working with the objective-function: changes in game structure, team dynamics, and the competition format.

Different sports have varying lengths, for instance, basketball games occur frequently as experienced by the model schedule multiple games can happen in one day. Whereas, for a sport like footall, games take place less frequently, which is why the total number of days for the scheduling and the total number of games would have to be adjusted accordingly.

Furthermore, different sports rely differently on the dispersion between home and away games. As we observed in basketball, games can take place in a single venue to ensure fairness in the match. However, a sport like football relies heavily on the home/away game scheme which fosters a sense of engagement with the fanbase as they feel more attached to a specific team when viewing home games [21]. In this case, with more reliance on the equitable dispersion of home/away games the model would require further tuning.

Additionally, another latent factor is the team dynamics. Sports have deviating player rotation principles and mechanics, for example in basketball, players are substituted commonly. Contradictingly, in rugby and football there are lower amounts of substitutions in games. This causes varying effects on the team fatigue which would have a limited but present effect on the team's fatigue. This translates to considering the sport mechanics when determining the constraint on the number of rest days each team gets.

Alternatively, sports have differing game scheduling structures. Often times in basketball, the regular season follows the notion that every team plays a fixed number of games, though; in other sports there could be groups where some teams do not even matchup, or elimination rounds. This would require a strong redifining in the model's objective-function.

In terms of the fairness of the matchups, this model successfully and effectively uses and ELO methodology to matchup teams based on skill level. Regardless of which sport is being played, this ELO method would theoretically have a successful output on the scheduling of fair matchups as the skill ratings of teams are not dependent on the sport played.

5 Visual Graphic

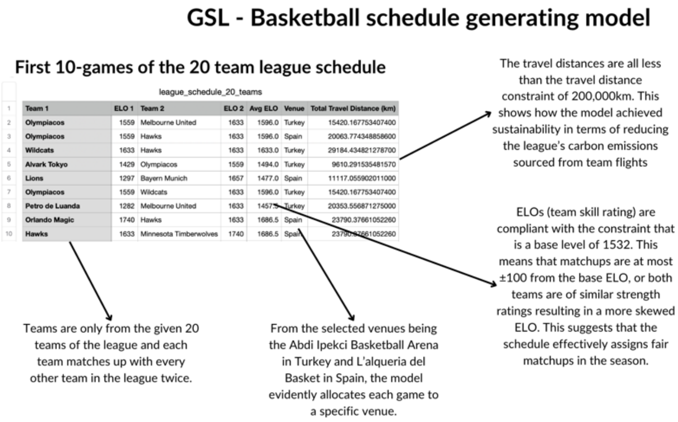

Figure 12 - Visual graphic of model performance (Source made by candidate).

6 Conclusion

In conclusion, after processing the results drawn from the output with 20 teams and 24 teams, the mathematical model has various strengths and weaknesses. However, its strengths overshadow its weaknesses which underscore its applicability in a real world situation. Its strengths are as follow, able to create a schedule with fair matchups, able to optimize the travel distances to ensure sustainable carbon emissions from journeying, and effectively adheres to the number of games and team games dispersion constraint. On the flip side, its weaknesses are as follow, not able to comply to the set of ideal game-streaming time periods, and its not able to effectively work around the minimum 1 rest day for all teams in the schedule. Finally, a potential extension to this model would be to consider the following calculation for the teams' ELO rating.

$$R_A = \frac{1}{1 + 10^{\left(\frac{E_B - E_A}{400}\right)}}$$

$$E_A' = E_A + K(S_A - E_A)$$

R_A is the expected probability of team A to win, and E_A is the ELO for team A, while E_B is the ELO for team B, S_A is the actual match result (1 is win, 0 is lose), K is a constant, and E_A' is the newly calculated ELO for team A. These calculations if implemented would dynamically update an individual team's ELO rating which would cause the model to schedule fair matchups effectively in the next season.

References

[1] Popp, Nels; Shapiro, Stephen; Simmons, Jason; and Dastrup, Ryan (2024) "Examining Reported Versus Actual Attendance in College Basketball Non-Conference Games: Do Scheduling Elements Make a Difference?," *Journal of Applied Sport Management*: Vol. 16 : Iss. 1. https://doi.org/10.7290/jasm16s52o
Available at: https://trace.tennessee.edu/jasm/vol16/iss1/5.

[2] Durán, Guillermo et al. "Scheduling Argentina's Professional Basketball Leagues: A Variation on the Travelling Tournament Problem." European Journal of Operational Research 275.3 (2018): 1126–1138. Web.

[3] Hassanzadeh, A., Hosseini, M. and Turner, J.G. (2024) *Beyond suspension: A two-phase methodology for concluding sports leagues*, *arXiv.org*. Available at: https://arxiv.org/abs/2404.00178 (Accessed: 18 March 2025).

[4] Bhola, D. (2025) *The impact of scheduling on player performance and well-being*, *Cricfit*. Available at: https://cricfit.com/the-impact-of-scheduling-on-player-performance-and-well-being/?utm (Accessed: 18 March 2025).

[5] Melkonian, V. (2024) An Integer Programming Approach for Scheduling a Professional Sports League. American Journal of Computational Mathematics, 14, 401-423. https://doi.org/10.4236/ajcm.2024.144021.

[6] Urban, Timothy & Russell, Robert. (2003). Scheduling sports competitions on multiple venues. European Journal of Operational Research. 148. 302-311. 10.1016/S0377-2217(02)00686-0. https://www.researchgate.net/publication/223116920_Scheduling_sports_competitions_on_multiple_venues.

[7] Naqvi, S.R. *et al.* (2021) *Towards prevention of sportsmen burnout: Formal analysis of sub-optimal tournament scheduling*, *arXiv.org*. Available at: https://arxiv.org/abs/2106.09627?utm (Accessed: 18 March 2025).

[8] Parkin, B. and Noble, J. (2024) *Can cricket replicate the success of the Indian Premier League?*, *Financial Times*. Available at: https://www.ft.com/content/5c0f75a2-9182-47e6-a527-194f5eac61b7?utm (Accessed: 18 March 2025).

[9] Ansorge, R. (2023) *Rest and recovery are critical for an athlete's physiological and psychological well-being*, *UCHealth Today*. Available at: https://www.uchealth.org/today/rest-and-recovery-for-athletes-physiological-psychological-well-being/?utm (Accessed: 18 March 2025).

[10] https://www.fiba.basketball/en/ranking/men.

[11] Tennery, A. and Carroll, R. (2024) *Christmas Day games on Netflix to help NFL expand Global Reach*, *Reuters*. Available at: https://www.reuters.com/sports/nfl/christmas-day-games-netflix-help-nfl-expand-global-reach-2024-12-23/ (Accessed: 18 March 2025).

[12] Robusto, C. C. "The Cosine-Haversine Formula." *The American Mathematical Monthly*, vol. 64, no. 1, 1957, pp. 38–40. *JSTOR*, https://doi.org/10.2307/2309088. Accessed 18 Mar. 2025.

[13] *UEFA respect report 2022-23* (2023) *UEFA*. Available at: https://sustainabilityreport.uefa.com/2022-23/ (Accessed: 18 March 2025).

[14] Nevill, A.M. and Holder, R.L. (2012) *Home advantage in sport - sports medicine*, *SpringerLink*. Available at: https://link.springer.com/article/10.2165/00007256-199928040-00001?utm (Accessed: 19 March 2025).

[15] Takkar, S. (2025) *Cultural diversity in sports: How inclusion shapes global athletics*, *Global Indian Network*. Available at: https://globalindiannetwork.com/cultural-diversity-in-sports/?utm_ (Accessed: 19 March 2025).

[16] Desk, S. (2023) *How long is the NBA season?*, *Sportskeeda*. Available at: https://www.sportskeeda.com/basketball/nba-regular-season-duration?utm_ (Accessed: 19 March 2025).

[17] Kimura, M. and Hino, H. (2024) *A short survey on importance weighting for Machine Learning*, *arXiv.org*. Available at: https://arxiv.org/abs/2403.10175 (Accessed: 19 March 2025).

[18] league_schedule.csv

[19] Jones, T. and Morgan, A. (2024) *How do NBA playoffs work - NBA playoffs format explained*, *Basketball Insiders*. Available at: https://www.basketballinsiders.org/guides/nba/how-playoffs-work/?utm_ (Accessed: 19 March 2025).

[20] league_schedule_24_teams_with_venues.csv

[21] Schube, S. (2024) *Inside the Clippers' billion-dollar play for LA*, *GQ*. Available at: https://www.gq.com/story/inside-the-clippers-billion-dollar-play-for-la?utm_ (Accessed: 20 March 2025).

Use of ChatGPT:

OpenAI: ChatGPT(March 2024 version, GPT 4-turbo)

Query: <can you output in a tabular format the longitudes and latitudes of the following locations {locations entered}>

Output: <Longitudes and latitudes of the locations in a tabular format>

AI report 1

LLMs such as ChatGPT was only used to find out the longitude and latitude of the locations from the teams selected. It was not used for any other reason in this paper.

A code

Note that the code below is not generated by AI. gamma.py:

```
import matplotlib.pyplot as plt

import geopandas as gpd

import cartopy.crs as ccrs

import cartopy.feature as cfeature from geopy.geocoders

import Nominatim import itertools

locations = { "USA": (37.0902, -95.7129), "Brazil": (-14.2350, -51.9253), "Argentina": (-38.4161, -63.6167), "China": (35.8617, 104.1954), "Japan": (36.2048, 138.2529), "South Korea": (35.9078, 127.7669), "Turkey": (38.9637, 35.2433), "Spain": (40.4637, -3.7492), "Greece": (39.0742, 21.8243), "Germany": (51.1657, 10.4515), "Serbia": (44.0165, 21.0059), "Lebanon": (33.8547, 35.8623), "Angola": (-11.2027, 17.8739), "South Africa": (-30.5595, 22.9375), "Australia": (-25.2744, 133.7751) }

fig, ax = plt.subplots(figsize=(12, 6), subplot_kw={'projection': ccrs.PlateCarree()}) ax.set_global() ax.add_feature(cfeature.LAND, facecolor='lightgray') ax.add_feature(cfeature.BORDERS, linestyle=':')

for countr, (lat, lon) in locations.items(): ax.plot(lon, lat, marker='o', markersize=5, color='red', transform=ccrs.PlateCarree()) ax.text(lon, lat, country, fontsize=9, ha='right', color='black')
```

```
for (country1, coord1), (country2, coord2) in itertools.combinations(locations.items(), 2): ax.plot([coord1[1], coord2[1]], [coord1[0], coord2[0]], linestyle="dotted", color="gray", alpha=0.5, transform=ccrs.PlateCarree())

plt.title("Global Basketball League Countries and Distances")

plt.show()
```

B code

```
import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

locations = { "USA": (37.0902, -95.7129), "Brazil": (-14.2350, -51.9253), "Argentina": (-38.4161, -63.6167), "China": (35.8617, 104.1954), "Japan": (36.2048, 138.2529), "South Korea": (35.9078, 127.7669), "Turkey": (38.9637, 35.2433), "Spain": (40.4637, -3.7492), "Greece": (39.0742, 21.8243), "Germany": (51.1657, 10.4515), "Serbia": (44.0165, 21.0059), "Lebanon": (33.8547, 35.8623), "Angola": (-11.2027, 17.8739), "South Africa": (-30.5595, 22.9375), "Australia": (-25.2744, 133.7751) }

R = 6371

def haversine(lat1, lon1, lat2, lon2): lat1, lon1, lat2, lon2 = map(np.radians, [lat1, lon1, lat2, lon2]) dlat = lat2 - lat1 dlon = lon2 - lon1 a = np.sin(dlat / 2.0) ** 2 + np.cos(lat1) * np.cos(lat2) * np.sin(dlon / 2.0) ** 2 c = 2 * np.arcsin(np.sqrt(a)) return R * c

potential_hosts = ["USA", "Spain", "Turkey", "China"]

distance_matrix = {host: [] for host in potential_hosts}  # Create dictionary with empty lists

for country, (lat, lon) in locations.items(): if country not in potential_hosts:

for host in potential_hosts: host_lat, host_lon = locations[host] distance = haversine(lat, lon, host_lat, host_lon) distance_matrix[host].append(distance)

df = pd.DataFrame(distance_matrix, index=[c for c in locations if c not in potential_hosts])

styled_df = df.style.format("{:,.0f} km").set_caption("Travel Distances to Host Locations") \ .set_properties(**{'text-align': 'center'}) \ .set_table_styles([{ 'selector': 'th', 'props': [('font-weight', 'bold'), ('background-color', '#f4f4f4')] }])

 styled_df
```

C code

```
plt.figure(figsize=(10, 6)) sns.heatmap(df, annot=True, cmap="coolwarm", linewidths=0.5, fmt=".0f") plt.title("Travel Distance Heatmap (in km) for Hosting Regions") plt.xlabel("Host Country") plt.ylabel("Other Countries") plt.xticks(rotation=45)
```

```
plt.show()
```

D code

```
total_emissions = total_distance * 3 * 3.15

results[host] = {"Total Distance (km)": total_distance, "Total CO₂ Emissions (kg)": total_emissions}

df = pd.DataFrame(results).T df["Total CO₂ Emissions (tons)"] = df["Total CO₂ Emissions (kg)"] / 1000

df = df[["Total Distance (km)", "Total CO₂ Emissions (tons)"]]

styled_df = df.style.format({ "Total Distance (km)": "{:,.0f} km", "Total CO₂ Emissions (tons)": "{:,.0f} tons" }).set_caption("Total Travel Distance & Carbon Emissions for Host Locations") \ .set_properties(**{'text-align': 'center'}) \ .set_table_styles([ {'selector': 'th', 'props': [('font-weight', 'bold'), ('background-color', '#f4f4f4')]}, {'selector': 'td', 'props': [('font-size', '14px')]} ]) \ .background_gradient(subset=["Total CO₂ Emissions (tons)"], cmap="Reds")

styled_df
```

E code

```
import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

W = np.array([ [0.30, 0,    0,    0,    0], [0,    0.20, 0,    0,    0], [0,    0,    0.20, 0,    0], [0,    0,    0,    0.20, 0], [0,    0,    0,    0,    0.10] ])

labels = ["Travel Distance", "Equitable Matchups", "Rest Days", "Venue Availability", "Streaming Time"] plt.figure(figsize=(8, 6))

sns.heatmap(W, annot=True, cmap="coolwarm", xticklabels=labels, yticklabels=labels)

plt.title("Weightage Matrix for Scheduling Variables")

plt.show()
```

F code

```
import numpy as np

import matplotlib.pyplot as plt

iterations = np.arange(1, 21)

alpha = 0.01
```

```
w1_init, w2_init, w3_init, w4_init, w5_init = 0.30, 0.20, 0.20, 0.20, 0.10

weight_D = w1_init - alpha * np.log(iterations + 1)

weight_S = w2_init - 0.5 * alpha * np.log(iterations + 1)

weight_R = w3_init + 0.3 * alpha * np.log(iterations + 1)

weight_V = w4_init + 0.2 * alpha * np.log(iterations + 1)

weight_T = w5_init + 0.5 * alpha * np.log(iterations + 1)

plt.figure(figsize=(10, 6))

plt.plot(iterations, weight_D, label="Travel Distance (w1)", marker='o', linestyle='-')

plt.plot(iterations, weight_S, label="Matchup Balance (w2)", marker='s', linestyle='--')

plt.plot(iterations, weight_R, label="Rest Days (w3)", marker='^', linestyle='-.')

plt.plot(iterations, weight_V, label="Venue Availability (w4)", marker='x', linestyle=':')

plt.plot(iterations, weight_T, label="Streaming Time (w5)", marker='d', linestyle='-')

plt.xlabel("Optimization Iterations")

plt.ylabel("Weight Value")

plt.title("Weightage Adjustments Over Optimization Iterations")

plt.legend()

plt.grid(True)

plt.show()
```

G code

```
import numpy as np

import pulp as pl

model = pl.LpProblem("Basketball_League_Scheduling", pl.LpMinimize)

teams = [ "Orlando Magic", "Minnesota Timberwolves", "Los Angeles Lakers", "Flamengo Basketball", "Minas", "Boca Juniors", "Lions", "Alvark Tokyo", "Seoul SK Knights", "Fenerbahçe Beko", "Barcelona", "Olympiacos", "Bayern Munich", "Crvena Zvezda", "Al Riyadi Club", "Petro de Luanda", "Cape Town Tigers", "Hawks", "Melbourne United", "Wildcats" ]

num_teams = len(teams)

num_days = 190

distances = {(i, j): np.random.randint(500, 10000) for i in range(num_teams) for j in range(num_teams) if i != j}
```

G = pl.LpVariable.dicts("Game", [(i, j, t) for i in range(num_teams) for j in range(num_teams) if i != j for t in range(num_days)], cat=pl.LpBinary)

for i in range(num_teams): for j in range(num_teams): if i != j: model += pl.lpSum(G[i, j, t] for t in range(num_days)) == 2 for i in range(num_teams): for t in range(num_days - 1): model += pl.lpSum(G[i, j, t] for j in range(num_teams) if i != j) + pl.lpSum(G[i, k, t+1] for k in range(num_teams) if i != k) <= 1

model += pl.lpSum(G[i, j, t] for i in range(num_teams) for j in range(num_teams) if i != j for t in range(num_days)) == 380

status = model.solve()

if pl.LpStatus[status] != "Optimal": print("⚠️ Warning: The model did not solve optimally. Check constraints.") else: print("Model solved successfully.")

print("Scheduled Games:") scheduled_games = [] for v in model.variables(): if v.varValue is not None and v.varValue > 0:

parts = v.name.replace("Game_", "").replace("(", "").replace(")", "").replace(",", "").split("_") i, j, t = map(int, parts) scheduled_games.append((teams[i], teams[j], t))

scheduled_games.sort(key=lambda x: x[2])

for game in scheduled_games: print(f"Day {game[2]}: {game[0]} vs. {game[1]}")

H code

import numpy as np

import pulp as pl

import pandas as pd

model = pl.LpProblem("Basketball_League_Scheduling", pl.LpMinimize)

teams = { "Orlando Magic": 1740, "Minnesota Timberwolves": 1740, "Los Angeles Lakers": 1740, "Flamengo Basketball": 1573, "Minas": 1573, "Boca Juniors": 1633, "Lions": 1297, "Alvark Tokyo": 1429, "Seoul SK Knights": 1136, "Fenerbahçe Beko": 1330, "Barcelona": 1646, "Olympiacos": 1559, "Bayern Munich": 1657, "Crvena Zvezda": 1661, "Al Riyadi Club": 1309, "Petro de Luanda": 1282, "Cape Town Tigers": 966, "Hawks": 1633, "Melbourne United": 1633, "Wildcats": 1633 }

team_names = list(teams.keys())

num_teams = len(team_names)

num_days = 220

distances = {(i, j): np.random.randint(500, 10000) for i in range(num_teams) for j in range(num_teams) if i != j}

G = pl.LpVariable.dicts("Game", [(i, j, t) for i in range(num_teams) for j in range(num_teams) if i != j for t in range(num_days)], cat=pl.LpBinary) for i in range(num_teams): for j in range(num_teams): if i != j: model += 1 <= pl.lpSum(G[i, j, t] for t in range(num_days)) <= 2

for i in range(num_teams): for t in range(num_days - 1): model += pl.lpSum(G[i, j, t] for j in range(num_teams) if i != j) + pl.lpSum(G[i, k, t+1] for k in range(num_teams) if i != k) <= 2

model += pl.lpSum(G[i, j, t] for i in range(num_teams) for j in range(num_teams) if i != j for t in range(num_days)) == 380

status = model.solve(pl.PULP_CBC_CMD(msg=1, timeLimit=300))

if pl.LpStatus[status] != "Optimal": print("⚠️ Warning: The model did not solve optimally. Try adjusting constraints.") else: print("Model solved successfully.")

scheduled_games = [] for v in model.variables(): if v.varValue is not None and v.varValue > 0: parts = v.name.replace("Game_", "").replace("(", "").replace(")", "").replace(",", "").split("_") i, j, t = map(int, parts) team1 = team_names[i] team2 = team_names[j] elo1 = teams[team1] elo2 = teams[team2] avg_elo = (elo1 + elo2) / 2

scheduled_games.append((t, team1, elo1, team2, elo2, avg_elo))

scheduled_games.sort(key=lambda x: x[0])

df_schedule = pd.DataFrame(scheduled_games, columns=["Day", "Team 1", "ELO 1", "Team 2", "ELO 2", "Avg ELO"])

print("\n League Schedule with ELO Ratings:")

print(df_schedule.to_string(index=False))