

Progetto di Ricerca Operativa

Università degli Studi di Parma

Ingegneria Informatica

Sabini Giada - Matricola: 310668

Organizzazione Torneo Pallavolo

Si consideri un insieme di N squadre (N numero pari) che devono fare un girone all'italiana. Ad ogni coppia i, j , $i \neq j$, di squadre è associato un livello di interesse $s_{ij} \in \{1, 2, 3\}$ della partita tra esse (1=minimo interesse, 2=medio interesse, 3=massimo interesse). Si definisca un calendario in modo tale che:

- in ogni giornata ci sia almeno una partita di massimo interesse.
- il minimo livello medio di interesse tra tutte le giornate sia massimizzato.

Formulare il modello per questo problema e provare a inserire qualche dato, risolvendo il problema corrispondente con AMPL. Si valuti anche il cambiamento della soluzione in corrispondenza di piccole variazioni dei dati.

Modello Matematico del Problema

$$\max \quad M \leq \frac{\sum_{i \in Teams} \sum_{j \in Teams} x_{i,j,g} \cdot interest_{i,j}}{num_teams/2}; \quad \forall g \in Days$$

$$x_{i,j,g} = 0; \quad \forall i \in Teams, \forall j \in Teams, \forall g \in Days \quad | \quad i = j$$

$$\sum_{g \in Days} x_{i,j,g} + x_{j,i,g} = 1; \quad \forall i \in Teams, \forall j \in Teams \quad | \quad i \neq j$$

$$\sum_{i \in Teams} \sum_{j \in Teams} x_{i,j,g} = num_teams/2; \quad \forall g \in Days$$

$$\sum_{j \in Teams} x_{i,j,g} + x_{j,i,g} = 1; \quad \forall i \in Teams, \forall g \in Days$$

$$\sum_{i \in Teams} \sum_{j \in Teams} x_{i,j,g} \geq 1; \quad \forall g \in Days \quad | \quad interest_{i,j} = 3$$

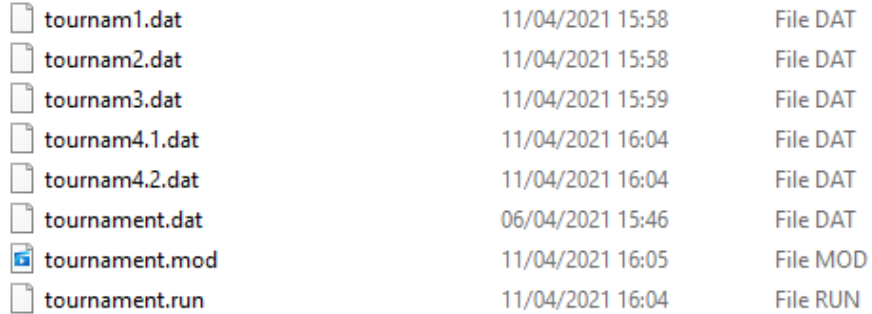
$$x_{i,j,g} \in \{0, 1\}; \quad \forall i \in Teams, \forall j \in Teams, \forall g \in Days$$

$$num_teams > 0$$

$$interest_{i,j} \in \{1, 2, 3\}; \quad \forall i \in Teams, \forall j \in Teams$$

AMPL

All'interno della cartella del progetto sono stati predisposti: *tournament.mod*, *tournament.run* e diversi file ".dat" pronti per essere eseguiti.











 tournam1.dat	11/04/2021 15:58	File DAT
 tournam2.dat	11/04/2021 15:58	File DAT
 tournam3.dat	11/04/2021 15:59	File DAT
 tournam4.1.dat	11/04/2021 16:04	File DAT
 tournam4.2.dat	11/04/2021 16:04	File DAT
 tournament.dat	06/04/2021 15:46	File DAT
 tournament.mod	11/04/2021 16:05	File MOD
 tournament.run	11/04/2021 16:04	File RUN

Figure 1: File nella cartella di progetto.

Il file con estensione ".mod" mostra come il problema è stato reso a modello, in linguaggio AMPL. Al suo interno si troveranno diverse dichiarazioni di parametri, insiemi e variabili, ma senza la loro assegnazione di valore. I valori da assegnare non si trovano infatti nel file ".mod", ma bensì nei diversi file con estensione ".dat". Ogni singolo ".dat" contiene dati tali che possa essere costruita un'istanza di rilievo.

Quest'ultimo tipo di file permette di effettuare l'assegnazione ai valori presenti in ".mod", a patto che vi sia corrispondenza di nome. Si è scelto di creare un file ".dat" per ogni esempio ritenuto interessante e, come si vedrà in seguito, sarà piuttosto semplice scegliere quale utilizzare per popolare i valori del ".mod".

All'interno della cartella vediamo infine un file con estensione ".run", tipo di file in grado di contenere una serie di istruzioni e riferimenti ad altri file, ed eseguire tali comandi in un determinato ordine. Passiamo ora ad analizzare il contenuto dei file, partendo proprio dal file ".run", descrivendolo brevemente.

Tournament.run

```
reset;
option solver gurobi;
model tournament.mod;
data tournam1.dat;
#data tournam2.dat;
#data tournam3.dat;

#data tournam4.1.dat;
#data tournam4.2.dat;

display num_teams;
display interest;

solve;

display x;
display Min_level;
```

Figure 2: Sequenza di istruzioni contenute nel file ".run".

L'istruzione "reset" è necessaria per "pulire" la memoria da dati precedenti, eventualmente ancora presenti in memoria. Nella riga seguente viene scelto il risolutore, "gurobi" in questo caso. In seguito si passa alla scelta del file ".mod" e di quello ".dat" da utilizzare. Il file ".mod" sarà sempre ovviamente lo stesso, il file ".dat" di cui si vuole fare uso può essere scelto decommentando la rispettiva riga di codice e commentando i riferimenti agli altri ".dat". Sono inoltre state inserite alcune istruzioni che permettono di stampare a console: numero di squadre, matrice dei livelli di interesse relativi ad ogni partita, lista dei giorni con le relative partite da effettuare e valore della funzione obiettivo.

Successivamente si ha l'istruzione "solve", necessaria a dare il via alla risoluzione e due istruzioni che permettono di stampare i valori delle variabili binarie e il valore della soluzione ottima del problema. Per "lanciare" un file ".run" è necessario disporre di "ampl.exe", avviarlo a terminale e digitare a riga di comando come in figura sottostante, a partire da ":".

```
ampl: include tournament.run;
```

Figure 3: Comando di esecuzione ".run", lanciato da terminale ampl.

Passiamo ora alla descrizione del modello del problema.

Tournament.mod

Partendo dal file ".mod", osserviamo una serie di "param", "set" e "var". Si tratta rispettivamente di dichiarazioni di parametri, insiemi e variabili.

```
## PARAMETRI ##
param num_teams > 0;          #numero PARI di squadre

## INSIEMI ##
set Teams;                   #insieme delle squadre
set Days := {1..num_teams-1}; #numero di giorni necessari per completare il torneo

## PARAMETRI ##
param interest{Teams,Teams}>=0; #interesse di ogni match

## VARIABILI ##
var x{Teams,Teams,Days} binary; #variabile binaria realtiva (i,j,g):
                                #1 se il match viene effettuato in una tal giornata g
                                #0 altrimenti
```

Figure 4: Dichiarazione di parametri, insiemi e variabili.

La prima dichiarazione indica il numero di squadre ("*num_teams*"), numero pari, che dovranno effettuare il girone all'italiana.

In secondo luogo vengono definiti due insiemi:

- *Teams*: insieme degli indici per identificare le squadre;
- *Days*: insieme degli indici per identificare i giorni necessari per effettuare il torneo, ovvero ("*num_teams-1*").

Appena sotto viene dichiarato un altro parametro: "*interest{Teams,Teams}*", riferito ad ogni coppia di squadra, il cui valore indica il livello di interesse della partita (1=minimo interesse, 2=medio interesse, 3=massimo interesse). Questa matrice di interessi verrà definita nel file ".dat".

Infine la dichiarazione della variabile "*x{Teams,Teams,Days}*", variabile binaria, utilizzata per indicare se una coppia di squadre effettuerà la partita in un determinato giorno.

Essendo binaria, tale variabile, può assumere solo "1" o "0" come valore.

- "1": la squadra "i" e la squadra "j" dovranno effettuare la partita nel giorno "g";
- "0": le squadre "i" e "j" non dovranno "scontrarsi" nel giorno "g".

Vengono successivamente impostati dei vincoli affinché non si verifichino situazioni di inconsistenza e affinché il requisito di avere in ogni giornata almeno una partita di massimo interesse venga soddisfatto.

Per comprendere i vincoli è necessario sapere in cosa consiste il sistema del girone all'italiana.

Il girone all'italiana (in inglese *round-robin tournament*) è uno schema di rotazioni che prevede che ogni squadra incontri lo stesso numero di volte tutte le squadre avversarie e nessuna venga eliminata; Se il numero di squadre partecipanti al girone è N , il numero totale di incontri che ogni squadra deve disputare è $N-1$; inoltre, come nel nostro caso, se il numero di squadre è pari ($N = 2n$) è possibile far giocare contemporaneamente tutte le squadre e completare $N/2$ sfide a turno.

```
## VINCOLI ##

#vincolo per evitare che una squadra sfidi se stessa
subject to no_self_matching{i in Teams, g in Days, j in Teams: j=i}: x[i,j,g] = 0;

#vincolo per evitare che una squadra sfidi due volte lo stesso avversario
subject to no_same_matching{i in Teams, j in Teams: j!=i}:
    sum{g in Days} (x[i,j,g] + x[j,i,g]) = 1;

#vincolo sul numero di match disponibili in una giornata
subject to number_of_match_in_a_day{g in Days}:
    sum{i in Teams}
    sum{j in Teams}
    x[i,j,g] = (num_teams/2);

#vincolo affinché una squadra abbia solo un avversario in ogni giornata
subject to only_one_match_in_a_day{i in Teams, g in Days}:
    sum{j in Teams} (x[i,j,g] + x[j,i,g]) = 1;

#vincolo affinché in ogni giornata ci sia almeno una partita di massimo interesse:
#(almeno un nodo in una giornata ha un arco uscente/entrante con valore 3)
subject to at_least_one{g in Days}:
    sum{i in Teams, j in Teams: interest[i,j] == 3} (x[i,j,g]) >= 1 ;
```

Figure 5: Vincoli utilizzati.

Passiamo alla descrizione dei vincoli utilizzati.

1. *no_self_matching*: tale vincolo impone che ogni squadra non possa sfidare se' stessa ($i = j$), in ogni giornata e che quindi la variabile " $x_{i,j,g}$ " risulti uguale a zero nel caso in cui $i = j$;
2. *no_same_matching*: tale vincolo impone che la squadra non sfidi due volte il suo avversario nel corso del torneo. Quindi in tutte le $N-1$ giornate del torneo la sommatoria delle variabili " $x_{i,j,g}$ ", con $i \neq j$, sarà = 1;
3. *number_of_match_in_a_day*: tale vincolo impone che in ogni giornata "g" ci sia un massimo di partite pari a " $num_teams / 2$ ";

4. *only_one_match_in_a_day*: tale vincolo impone che in ogni giornata "g", ogni squadra giochi una partita solo contro un avversario.
5. *at_least_one*: questo ultimo vincolo è lo specifico vincolo che viene imposto dal problema, ovvero che in ogni giornata "g" ci sia almeno una partita di massimo interesse.

Passiamo infine alla realizzazione della funzione obiettivo del problema.

```
## OBIETTIVO ##

#il minimo livello medio di interesse tra tutte le giornate sia massimizzato

var M;
subject to Min_lev{g in Days}: M <= (sum{i in Teams}
                                     sum{j in Teams}
                                     x[i,j,g] * interest[i,j]) / (num_teams/2);

maximize Min_level: M;
```

Figure 6: Definizione dell'obiettivo.

Da consegna, è necessario massimizzare il minimo livello medio di interesse tra tutte le giornate. Per la realizzazione della funzione obiettivo ci si è serviti di una variabile e di un ulteriore vincolo. In primo luogo è stata quindi dichiarata una variabile "M". Appena sotto, definiamo un vincolo che agisce sulla variabile dichiarata. Il vincolo è definito come "*Min_lev*" e agisce sui giorni. Il vincolo impone che tale variabile "M" sia, in ogni giornata, minore o uguale alla media dei livelli di interesse.

Appena sotto si ha l'istruzione "*maximize*", che ha lo scopo di massimizzare il valore assunto da una certa variabile, nel nostro caso si tratta proprio della variabile "M". Questa istruzione permette all'algoritmo di re-iterare tra le soluzioni, cercando quella che fornisce il minimo livello medio di interesse, tra tutti i giorni, massimo.

Esempi di Esecuzione

Vengono ora analizzati alcuni esempi di file ".dat" per il problema. Un file ".dat" contiene assegnazioni di valori. Tali valori andranno a popolare i parametri dichiarati all'interno di "*schedulazione.mod*". Ogni file ".dat" e il relativo output saranno discussi brevemente.

Tournament 1

```
## tournament_1 ##  
  
# Parametri  
  
param num_teams = 4;           #numero di squadre  
set Teams := A, B, C, D;  
  
param interest:                #livello di interesse tra ogni match  
  A   B   C   D :=  
A   0   3   1   3  
B   3   0   1   3  
C   1   1   0   1  
D   3   3   1   0;
```

Figure 7: Tournament 1.

Intuitibilmente, in questo primo semplice esempio, si sceglie di generare un problema in cui si hanno 4 squadre identificate dalle lettere A, B, C, D. Successivamente è definita una matrice 4x4, simmetrica (numero di righe = numero delle colonne = numero di squadre) per l'assegnazione dei livelli di interesse di ogni partita.

Nell'esempio: la partita della squadra A contro la squadra B avrà un livello di interesse pari a 3, quindi massimo; la partita della squadra A contro la squadra C avrà un livello di interesse pari a 1, quindi minimo; la partita della squadra A contro la squadra D avrà un livello di interesse pari a 3, quindi massimo; via via così per ogni squadra.

Verifichiamo quindi l'output prodotto:

```

ampl: include tournament.run;
num_teams = 4

interest :=
A A  0
A B  3
A C  1
A D  3
B A  3
B B  0
B C  1
B D  3
C A  1
C B  1
C C  0
C D  1
D A  3
D B  3
D C  1
D D  0
;

```

Figure 8: Output dei dati presenti nel *tournam1.dat*.

```

Gurobi 9.1.1: optimal solution; objective 2
plus 1 simplex iteration for intbasis
x [*,*,1]
:  A  B  C  D  :=
A  0  1  0  0
B  0  0  0  0
C  0  0  0  1
D  0  0  0  0

[*,*,2]
:  A  B  C  D  :=
A  0  0  0  0
B  0  0  0  0
C  0  1  0  0
D  1  0  0  0

[*,*,3]
:  A  B  C  D  :=
A  0  0  1  0
B  0  0  0  0
C  0  0  0  0
D  0  1  0  0
;

Min_level = 2

```

Figure 9: Output con soluzione ottima trovata, iterazioni eseguite e scelta delle variabili.

Da ciò che risulta dall'output (Figure 9): vengono rappresentati a schermo tante matrici quante il numero di giorni necessari per completare il torneo; Queste matrici rappresentano i valori delle variabili binarie " $x_{i,j,g}$ ", infatti, tali variabili assumono valore "1" se sarà disputata la partita tra la squadra "i" e la squadra "j", "0" altrimenti.

In questo caso specifico il risolutore ha trovato la soluzione ottima con valore della funzione obiettivo pari a 2, e tale soluzione è rappresentata in questo modo:

- Per $g=1$: " $x_{1,2,1}$ "=1, " $x_{3,4,1}$ "=1;
- Per $g=2$: " $x_{4,1,2}$ "=1, " $x_{3,2,2}$ "=1;
- Per $g=3$: " $x_{1,3,3}$ "=1, " $x_{2,4,3}$ "=1;

Nella pratica, questa scelta di variabili binarie è traducibile in:

- Giorno 1:
 - squadra A contro squadra B, con livello di interesse pari a 3;
 - squadra C contro squadra D, con livello di interesse pari a 1;
- Giorno 2:
 - squadra D contro squadra A, con livello di interesse pari a 3;
 - squadra C contro squadra B, con livello di interesse pari a 1;
- Giorno 3:
 - squadra A contro squadra C, con livello di interesse pari a 1;
 - squadra B contro squadra D, con livello di interesse pari a 3;

Come possiamo notare dalla soluzione appena trovata, ogni vincolo viene rispettato: ogni squadra non sfida mai nè se stessa nè una squadra avversaria più di una volta durante il torneo; in ogni giornata è presente un numero massimo di partite pari a $N/2$ ($4/2 = 2$); in ogni giornata è presente almeno una partita di massimo interesse ($=3$).

Infine, sempre in (Figure 9), notiamo il valore ottimo corrispondente alla soluzione ottima appena osservata; esso vale "2" e corrisponde al minimo livello medio di interesse tra tutte le giornate.

Calcolo dei vincoli su M:

$$\text{Per } g = 1 : M \leq (x_{1,2,1} * 3 + x_{3,4,1} * 1) / 2 = 2$$

$$\text{Per } g = 2 : M \leq (x_{4,1,2} * 3 + x_{3,2,2} * 1) / 2 = 2$$

$$\text{Per } g = 3 : M \leq (x_{1,3,3} * 3 + x_{2,4,3} * 1) / 2 = 2$$

Di conseguenza si avrà $M \leq 2$. Dal momento che il valore ottimo va massimizzato, otteniamo $M = 2$ per questo torneo.

Tournament 2

```
## tournament_2 ##

# Parametri

param num_teams = 6;           #numero di squadre
set Teams := A, B, C, D, E, F;

param interest:                #livello di interesse tra ogni match
    A  B  C  D  E  F:=
A  0  2  2  3  1  3
B  2  0  1  3  3  1
C  2  1  0  3  1  3
D  3  3  3  0  2  1
E  1  3  1  2  0  3
F  3  1  3  1  3  0;
```

Figure 10: Tournament 2.

In questo secondo esempio, si sceglie di generare un problema in cui si hanno 6 squadre identificate dalle lettere A, B, C, D, E, F. Successivamente e' definita una matrice 6x6 simmetrica per l'assegnazione dei livelli di interesse di ogni partita.

Verifichiamo quindi l'output prodotto:

```
ampl: include tournament.run
num_teams = 6

interest [*,*]
:  A  B  C  D  E  F  :=
A  0  2  2  3  1  3
B  2  0  1  3  3  1
C  2  1  0  3  1  3
D  3  3  3  0  2  1
E  1  3  1  2  0  3
F  3  1  3  1  3  0
;
```

Figure 11: Output dei dati presenti nel *tournam2.dat*.

Da ciò che risulta dall'output (Figure 12): come nell'esempio precedente vengono rappresentati a schermo tante matrici quante il numero di giorni necessari per completare il torneo, cioè $N - 1 = 6 - 1 = 5$.

```

Gurobi 9.1.1: optimal solution; objective 1.666666667
139 simplex iterations
1 branch-and-cut nodes
plus 2 simplex iterations for intbasis
x [*,*,1]
:  A  B  C  D  E  F  :=
A  0  0  0  0  1  0
B  0  0  0  1  0  0
C  0  0  0  0  0  1
D  0  0  0  0  0  0
E  0  0  0  0  0  0
F  0  0  0  0  0  0

[*,*,2]
:  A  B  C  D  E  F  :=
A  0  0  1  0  0  0
B  0  0  0  0  1  0
C  0  0  0  0  0  0
D  0  0  0  0  0  0
E  0  0  0  0  0  0
F  0  0  0  1  0  0

[*,*,3]
:  A  B  C  D  E  F  :=
A  0  0  0  0  0  0
B  0  0  0  0  0  1
C  0  0  0  0  1  0
D  1  0  0  0  0  0
E  0  0  0  0  0  0
F  0  0  0  0  0  0

[*,*,4]
:  A  B  C  D  E  F  :=
A  0  0  0  0  0  0
B  0  0  0  0  0  0
C  0  1  0  0  0  0
D  0  0  0  0  0  0
E  0  0  0  1  0  0
F  1  0  0  0  0  0

[*,*,5]
:  A  B  C  D  E  F  :=
A  0  0  0  0  0  0
B  1  0  0  0  0  0
C  0  0  0  0  0  0
D  0  0  1  0  0  0
E  0  0  0  0  0  0
F  0  0  0  0  1  0
;

Min_level = 1.66667

```

Figure 12: Output con soluzione ottima trovata, iterazioni eseguite e scelta delle variabili.

In quest'altro caso il risolutore ha trovato la soluzione ottima con valore della funzione obiettivo pari a 1.66667, e tale soluzione è rappresentata in questo modo:

- Per $g=1$: $x_{\{2,4,1\}}=1$, $x_{\{1,5,1\}}=1$, $x_{\{3,6,1\}}=1$;
- Per $g=2$: $x_{\{1,3,2\}}=1$, $x_{\{2,5,2\}}=1$, $x_{\{6,4,2\}}=1$;
- Per $g=3$: $x_{\{4,1,3\}}=1$, $x_{\{2,6,3\}}=1$, $x_{\{3,5,3\}}=1$;
- Per $g=4$: $x_{\{6,1,4\}}=1$, $x_{\{5,4,4\}}=1$, $x_{\{3,2,4\}}=1$;
- Per $g=5$: $x_{\{2,1,5\}}=1$, $x_{\{4,3,5\}}=1$, $x_{\{6,5,5\}}=1$;

Nella pratica, questa scelta di variabili binarie è traducibile in:

- Giorno 1:
 - squadra B contro squadra D, con livello di interesse pari a 3;
 - squadra A contro squadra E, con livello di interesse pari a 1;
 - squadra C contro squadra F, con livello di interesse pari a 3;
- Giorno 2:
 - squadra A contro squadra C, con livello di interesse pari a 2;
 - squadra B contro squadra E, con livello di interesse pari a 3;
 - squadra F contro squadra D, con livello di interesse pari a 1;
- Giorno 3:
 - squadra D contro squadra A, con livello di interesse pari a 3;
 - squadra C contro squadra E, con livello di interesse pari a 1;
 - squadra B contro squadra F, con livello di interesse pari a 1;
- Giorno 4:
 - squadra F contro squadra A, con livello di interesse pari a 3;
 - squadra C contro squadra B, con livello di interesse pari a 1;
 - squadra E contro squadra D, con livello di interesse pari a 2;
- Giorno 5:
 - squadra B contro squadra A, con livello di interesse pari a 2;
 - squadra D contro squadra C, con livello di interesse pari a 3;
 - squadra F contro squadra E, con livello di interesse pari a 3;

Come possiamo notare anche in questo esempio ogni vincolo viene rispettato: ogni squadra non sfida mai nè se stessa nè una squadra avversaria più di una volta durante il torneo; in ogni giornata è presente un numero massimo di partite pari a $N/2$ ($6/2 = 3$); in ogni giornata è presente almeno una partita di

massimo interesse.

Infine il valore ottimo corrispondente alla soluzione ottima appena osservata; esso vale 1.66667 e corrisponde al minimo livello medio di interesse tra tutte le giornate.

Infatti, se vado a calcolare i vincoli su M , ricordano la funzione obiettivo in Figure 13:

Per $g = 1$: $M \leq (x_{\{2,4,1\}} * 3 + x_{\{1,5,1\}} * 1 + x_{\{3,6,1\}} * 3) / 3 = 2,33333$

Per $g = 2$: $M \leq (x_{\{1,3,2\}} * 2 + x_{\{2,5,2\}} * 3 + x_{\{6,4,2\}} * 1) / 3 = 2$

Per $g = 3$: $M \leq (x_{\{4,1,3\}} * 3 + x_{\{2,6,3\}} * 1 + x_{\{3,5,1\}} * 1) / 3 = 1,66667$

Per $g = 4$: $M \leq (x_{\{6,1,4\}} * 3 + x_{\{5,4,4\}} * 2 + x_{\{3,2,4\}} * 1) / 3 = 2$

Per $g = 5$: $M \leq (x_{\{2,1,5\}} * 2 + x_{\{4,3,5\}} * 3 + x_{\{6,5,5\}} * 3) / 3 = 2,66667$

```
var M;
subject to Min_lev{g in Days}: M <= (sum{i in Teams}
                                     sum{j in Teams}
                                     x[i,j,g] * interest[i,j]) / (num_teams/2);
```

Figure 13: Funzione obiettivo.

Di conseguenza si avrà $M \leq 1.66667$. Dal momento che il valore ottimo va massimizzato, otteniamo $M = 1.66667$ per questo torneo.

Facciamo notare che il numero di iterazioni "*simplex iterations*" ha subito un notevole incremento, da 1 a 139 iterazioni, ed un uso dell'algoritmo *Branch&Cut*.

Tournament 3

```
## tournament_3 ##

# Parametri

param num_teams = 8;           #numero di squadre
set Teams := A, B, C, D, E, F, G, H;

param interest:               #livello di interesse tra ogni match
  A B C D E F G H:=
A 0 3 2 1 2 3 2 1
B 3 0 2 3 1 1 2 2
C 2 2 0 2 2 1 2 3
D 1 3 2 0 3 2 2 2
E 2 1 2 3 0 2 1 1
F 3 1 1 2 2 0 3 1
G 2 2 2 2 1 3 0 3
H 1 2 3 2 1 1 3 0;
```

Figure 14: Tournament 3.

In questo terzo esempio, si sceglie di generare un problema in cui si hanno 8 squadre identificate dalle lettere A, B, C, D, E, F, G, H. Successivamente è definita una matrice 8x8, simmetrica per l'assegnazione dei livelli di interesse di

ogni partita.

Verifichiamo quindi l'output prodotto:

```

ampl: include tournament.run
num_teams = 8

interest [*,*]
:   A   B   C   D   E   F   G   H   :=
A   0   3   2   1   2   3   2   1
B   3   0   2   3   1   1   2   2
C   2   2   0   2   2   1   2   3
D   1   3   2   0   3   2   2   2
E   2   1   2   3   0   2   1   1
F   3   1   1   2   2   0   3   1
G   2   2   2   2   1   3   0   3
H   1   2   3   2   1   1   3   0
;

```

Figure 15: Output dei dati presenti nel *tournam3.dat*.

```

Gurobi 9.1.1: optimal solution; objective 1.75
204 simplex iterations
plus 2 simplex iterations for intbasis
x [*,*,1]
:   A   B   C   D   E   F   G   H   :=
A   0   0   0   0   0   0   0   0
B   0   0   0   0   0   0   0   0
C   0   0   0   1   0   0   0   0
D   0   0   0   0   0   0   0   0
E   0   0   0   0   0   0   1   0
F   1   0   0   0   0   0   0   0
G   0   0   0   0   0   0   0   0
H   0   1   0   0   0   0   0   0

[*,*,2]
:   A   B   C   D   E   F   G   H   :=
A   0   0   0   0   0   0   0   0
B   0   0   0   0   0   0   1   0
C   0   0   0   0   0   0   0   0
D   0   0   0   0   0   0   0   0
E   0   0   0   1   0   0   0   0
F   0   0   1   0   0   0   0   0
G   0   0   0   0   0   0   0   0
H   1   0   0   0   0   0   0   0

```

Da ciò che risulta dall'output (Figure 16): come nell'esempio precedente vengono rappresentati a schermo tante matrici quante il numero di giorni necessari per completare il torneo, cioè $N - 1 = 8 - 1 = 7$.

```

[*,* ,3]
:  A  B  C  D  E  F  G  H  :=
A  0  0  0  0  0  0  0  0
B  0  0  0  0  1  0  0  0
C  0  0  0  0  0  0  0  0
D  0  0  0  0  0  0  0  0
E  0  0  1  0  0  0  0  0
F  0  0  0  0  0  0  0  1
G  1  0  0  0  0  0  0  0
H  0  0  0  0  0  0  0  0

[*,* ,4]
:  A  B  C  D  E  F  G  H  :=
A  0  0  0  0  0  0  0  0
B  0  0  0  0  0  0  0  0
C  0  0  0  0  0  0  0  0
D  0  0  0  0  0  0  0  0
E  1  0  0  0  0  0  0  0
F  0  1  0  0  0  0  0  0
G  0  0  0  1  0  0  0  0
H  0  0  1  0  0  0  0  0

[*,* ,5]
:  A  B  C  D  E  F  G  H  :=
A  0  0  0  0  0  0  0  0
B  0  0  0  0  0  0  0  0
C  1  0  0  0  0  0  0  0
D  0  0  0  0  0  0  0  0
E  0  1  0  0  0  0  0  0
F  0  0  0  0  0  0  0  0
G  0  0  0  0  0  1  0  0
H  0  0  0  1  0  0  0  0

[*,* ,6]
:  A  B  C  D  E  F  G  H  :=
A  0  0  0  0  0  0  0  0
B  1  0  0  0  0  0  0  0
C  0  0  0  0  0  0  1  0
D  0  0  0  0  0  0  0  0
E  0  0  0  0  0  0  0  1
F  0  0  0  1  0  0  0  0
G  0  0  0  0  0  0  0  0
H  0  0  0  0  0  0  0  0

[*,* ,7]
:  A  B  C  D  E  F  G  H  :=
A  0  0  0  0  0  0  0  0
B  0  0  0  0  0  0  0  0
C  0  1  0  0  0  0  0  0
D  1  0  0  0  0  0  0  0
E  0  0  0  0  0  0  0  0
F  0  0  0  0  1  0  0  0
G  0  0  0  0  0  0  0  0
H  0  0  0  0  0  0  1  0
;

Min_level = 1.75

```

Figure 16: Output con soluzione ottima trovata, iterazioni eseguite e scelta delle variabili.

In quest'altro caso il risolutore ha trovato la soluzione ottima con valore della funzione obiettivo pari a 1.75, e tale soluzione è rappresentata in questo modo (Figure 16):

- Per g=1: $x_{3,4,1}=1$, $x_{5,7,1}=1$, $x_{6,1,1}=1$, $x_{8,2,1}=1$;
- Per g=2: $x_{8,1,2}=1$, $x_{2,7,2}=1$, $x_{5,4,2}=1$, $x_{6,3,2}=1$;

-Per $g=3$: " $x\{7,1,3\}$ "=1, " $x\{2,4,3\}$ "=1, " $x\{5,3,3\}$ "=1, " $x\{6,8,3\}$ "=1;
-Per $g=4$: " $x\{5,1,4\}$ "=1, " $x\{6,2,4\}$ "=1, " $x\{7,4,4\}$ "=1, " $x\{8,3,4\}$ "=1;
-Per $g=5$: " $x\{3,1,5\}$ "=1, " $x\{5,2,5\}$ "=1, " $x\{7,6,5\}$ "=1, " $x\{8,4,5\}$ "=1;
-Per $g=6$: " $x\{2,1,6\}$ "=1, " $x\{3,7,6\}$ "=1, " $x\{6,4,6\}$ "=1, " $x\{5,8,6\}$ "=1;
-Per $g=7$: " $x\{4,1,7\}$ "=1, " $x\{3,2,7\}$ "=1, " $x\{6,5,7\}$ "=1, " $x\{8,7,7\}$ "=1;

Nella pratica, questa scelta di variabili binarie è traducibile in:

- Giorno 1:
 - squadra C contro squadra D, con livello di interesse pari a 2;
 - squadra E contro squadra G, con livello di interesse pari a 1;
 - squadra F contro squadra A, con livello di interesse pari a 3;
 - squadra H contro squadra B, con livello di interesse pari a 2;
- Giorno 2:
 - squadra H contro squadra A, con livello di interesse pari a 1;
 - squadra B contro squadra G, con livello di interesse pari a 2;
 - squadra F contro squadra C, con livello di interesse pari a 1;
 - squadra E contro squadra D, con livello di interesse pari a 3;
- Giorno 3:
 - squadra G contro squadra A, con livello di interesse pari a 2;
 - squadra B contro squadra D, con livello di interesse pari a 3;
 - squadra F contro squadra H, con livello di interesse pari a 1;
 - squadra E contro squadra C, con livello di interesse pari a 2;
- Giorno 4:
 - squadra E contro squadra A, con livello di interesse pari a 2;
 - squadra F contro squadra B, con livello di interesse pari a 1;
 - squadra G contro squadra D, con livello di interesse pari a 2;
 - squadra H contro squadra C, con livello di interesse pari a 3;
- Giorno 5:
 - squadra C contro squadra A, con livello di interesse pari a 2;
 - squadra E contro squadra B, con livello di interesse pari a 1;
 - squadra G contro squadra F, con livello di interesse pari a 3;
 - squadra H contro squadra D, con livello di interesse pari a 2;
- Giorno 6:

- squadra B contro squadra A, con livello di interesse pari a 3;
- squadra C contro squadra G, con livello di interesse pari a 2;
- squadra F contro squadra D, con livello di interesse pari a 2;
- squadra E contro squadra H, con livello di interesse pari a 1;

• Giorno 7:

- squadra D contro squadra A, con livello di interesse pari a 1;
- squadra F contro squadra E, con livello di interesse pari a 2;
- squadra C contro squadra B, con livello di interesse pari a 2;
- squadra H contro squadra G, con livello di interesse pari a 3;

Anche in questo esempio ogni vincolo viene rispettato: ogni squadra non sfida mai nè se stessa nè una squadra avversaria più di una volta durante il torneo; in ogni giornata è presente un numero massimo di partite pari a $N/2$ ($8/2 = 4$); in ogni giornata è presente almeno una partita di massimo interesse.

Possiamo notare infine il valore ottimo corrispondente alla soluzione ottima appena osservata; esso vale 1.75 e corrisponde al minimo livello medio di interesse tra tutte le giornate.

Infatti, se vado a calcolare i vincoli su M:

$$\text{Per } g = 1 : M \leq (x\{3, 4, 1\} * 2 + x\{5, 7, 1\} * 1 + x\{6, 1, 1\} * 3 + x\{8, 2, 1\} * 2) / 4 = 2$$

$$\text{Per } g = 2 : M \leq (x\{8, 1, 2\} * 1 + x\{2, 7, 2\} * 2 + x\{5, 4, 2\} * 3 + x\{6, 3, 2\} * 1) / 4 = 1,75$$

$$\text{Per } g = 3 : M \leq (x\{7, 1, 3\} * 2 + x\{2, 4, 3\} * 3 + x\{5, 3, 1\} * 2 + x\{6, 8, 3\} * 1) / 4 = 2$$

$$\text{Per } g = 4 : M \leq (x\{5, 1, 4\} * 1 + x\{6, 2, 4\} * 3 + x\{7, 4, 4\} * 2 + x\{8, 3, 4\} * 1) / 4 = 1,75$$

$$\text{Per } g = 5 : M \leq (x\{3, 1, 5\} * 2 + x\{5, 2, 5\} * 1 + x\{7, 6, 5\} * 3 + x\{8, 4, 5\} * 2) / 4 = 2$$

$$\text{Per } g = 6 : M \leq (x\{2, 1, 6\} * 3 + x\{3, 7, 6\} * 2 + x\{6, 4, 6\} * 2 + x\{5, 8, 6\} * 1) / 4 = 2$$

$$\text{Per } g = 7 : M \leq (x\{4, 1, 7\} * 1 + x\{3, 2, 7\} * 2 + x\{6, 5, 7\} * 2 + x\{8, 7, 7\} * 3) / 4 = 2$$

Di conseguenza si avrà $M \leq 1,75$. Dal momento che il valore ottimo va massimizzato, otteniamo $M = 1,75$ per questo torneo.

Facciamo notare che il numero di iterazioni "*simplex iterations*" ha subito un considerevole incremento, da 139 a 204 iterazioni.

Tournament 4_1

Dopo alcuni esempi di *tournament* in grado di mostrare il funzionamento, si è deciso di analizzare un esempio con una modifica dei valori nella matrice degli interessi. Si noterà successivamente che, attraverso l'utilizzo di questa scelta, il risolutore non sarà in grado di generare una soluzione che soddisfi i vincoli imposti nel problema.

```
## tournament_4_1 ##

# inserisco una matrice di interessi che NON mi garantirà la realizzazione del torneo

# Parametri

param num_teams = 6;           #numero di squadre
set Teams := A, B, C, D, E, F;

param interest:                #livello di interesse tra ogni match
    A  B  C  D  E  F:=
A  0  2  2  3  1  2
B  2  0  1  3  3  1
C  2  1  0  2  1  1
D  3  3  2  0  2  1
E  1  3  1  2  0  3
F  2  1  1  1  3  0;
```

Figure 17: Tournament 4_1.

```
Gurobi 9.1.1: infeasible or unbounded
No basis.
x [*,*,1]
:  A  B  C  D  E  F  :=
A  0  0  0  0  0  0
B  0  0  0  0  0  0
C  0  0  0  0  0  0
D  0  0  0  0  0  0
E  0  0  0  0  0  0
F  0  0  0  0  0  0

[*,*,2]
:  A  B  C  D  E  F  :=
A  0  0  0  0  0  0
B  0  0  0  0  0  0
C  0  0  0  0  0  0
D  0  0  0  0  0  0
E  0  0  0  0  0  0
F  0  0  0  0  0  0

[*,*,3]
:  A  B  C  D  E  F  :=
A  0  0  0  0  0  0
B  0  0  0  0  0  0
C  0  0  0  0  0  0
D  0  0  0  0  0  0
E  0  0  0  0  0  0
F  0  0  0  0  0  0
```

```

[*,* ,4]
:  A  B  C  D  E  F  :=
A  0  0  0  0  0  0
B  0  0  0  0  0  0
C  0  0  0  0  0  0
D  0  0  0  0  0  0
E  0  0  0  0  0  0
F  0  0  0  0  0  0

[*,* ,5]
:  A  B  C  D  E  F  :=
A  0  0  0  0  0  0
B  0  0  0  0  0  0
C  0  0  0  0  0  0
D  0  0  0  0  0  0
E  0  0  0  0  0  0
F  0  0  0  0  0  0
;

Min_level = 0

```

Figure 18: Output con soluzione ottima trovata, iterazioni eseguite e scelta delle variabili.

Nell' esempio, si sceglie di generare un problema in cui si hanno 6 squadre: A, B, C, D, E, F.

Se verifichiamo l'output (Figure 18) prodotto, notiamo che il valore della funzione obiettivo è nullo. Non si riescono infatti a generare N-1 giornate affinché in ogni giornata i vincoli imposti dal problema vengano rispettati.

Tournament 4_2

In questo ultimo esempio (Figure 19) si è scelto di partire dalla matrice degli interessi dell'esempio *tournam_3.dat*, modificando il valore della coppia AB (=BA) dal valore 3 al valore 2. Con questa leggera modifica notiamo che si ricade in un esempio in cui non si riescono a generare N-1 giornate che soddisfino i vincoli assegnati in precedenza.

```
## tournament_4_2 ##

# inserisco una matrice di interessi che NON mi garantirà la realizzazione del torneo

# Parametri

param num_teams = 8;           #numero di squadre
set Teams := A, B, C, D, E, F, G, H;

# stessa matrice del file tournam_3 ma con alcune modifiche:
# arco AB (= BA) va da 3 -> 2

param interest:                #livello di interesse tra ogni match
  A  B  C  D  E  F  G  H:=
A  0  2  2  1  2  3  2  1
B  2  0  2  3  1  1  2  2
C  2  2  0  2  2  1  2  3
D  1  3  2  0  3  2  2  2
E  2  1  2  3  0  2  1  1
F  3  1  1  2  2  0  3  1
G  2  2  2  2  1  3  0  3
H  1  2  3  2  1  1  3  0;
```

Figure 19: Tournament 4_2.

Verifichiamo l'output (Figure 20):

```
Gurobi 9.1.1: infeasible or unbounded
268 simplex iterations
No basis.
x [*,*,1]
:  A  B  C  D  E  F  G  H  :=
A  0  0  0  0  0  0  0  0
B  0  0  0  0  0  0  0  0
C  0  0  0  0  0  0  0  0
D  0  0  0  0  0  0  0  0
E  0  0  0  0  0  0  0  0
F  0  0  0  0  0  0  0  0
G  0  0  0  0  0  0  0  0
H  0  0  0  0  0  0  0  0
```

```

[*],*,2]
: A B C D E F G H :=
A 0 0 0 0 0 0 0 0
B 0 0 0 0 0 0 0 0
C 0 0 0 0 0 0 0 0
D 0 0 0 0 0 0 0 0
E 0 0 0 0 0 0 0 0
F 0 0 0 0 0 0 0 0
G 0 0 0 0 0 0 0 0
H 0 0 0 0 0 0 0 0

[*],*,3]
: A B C D E F G H :=
A 0 0 0 0 0 0 0 0
B 0 0 0 0 0 0 0 0
C 0 0 0 0 0 0 0 0
D 0 0 0 0 0 0 0 0
E 0 0 0 0 0 0 0 0
F 0 0 0 0 0 0 0 0
G 0 0 0 0 0 0 0 0
H 0 0 0 0 0 0 0 0

[*],*,4]
: A B C D E F G H :=
A 0 0 0 0 0 0 0 0
B 0 0 0 0 0 0 0 0
C 0 0 0 0 0 0 0 0
D 0 0 0 0 0 0 0 0
E 0 0 0 0 0 0 0 0
F 0 0 0 0 0 0 0 0
G 0 0 0 0 0 0 0 0
H 0 0 0 0 0 0 0 0

[*],*,5]
: A B C D E F G H :=
A 0 0 0 0 0 0 0 0
B 0 0 0 0 0 0 0 0
C 0 0 0 0 0 0 0 0
D 0 0 0 0 0 0 0 0
E 0 0 0 0 0 0 0 0
F 0 0 0 0 0 0 0 0
G 0 0 0 0 0 0 0 0
H 0 0 0 0 0 0 0 0

[*],*,6]
: A B C D E F G H :=
A 0 0 0 0 0 0 0 0
B 0 0 0 0 0 0 0 0
C 0 0 0 0 0 0 0 0
D 0 0 0 0 0 0 0 0
E 0 0 0 0 0 0 0 0
F 0 0 0 0 0 0 0 0
G 0 0 0 0 0 0 0 0
H 0 0 0 0 0 0 0 0

[*],*,7]
: A B C D E F G H :=
A 0 0 0 0 0 0 0 0
B 0 0 0 0 0 0 0 0
C 0 0 0 0 0 0 0 0
D 0 0 0 0 0 0 0 0
E 0 0 0 0 0 0 0 0
F 0 0 0 0 0 0 0 0
G 0 0 0 0 0 0 0 0
H 0 0 0 0 0 0 0 0
;

Min_level = 0

```

Figure 20: Output con soluzione ottima trovata, iterazioni eseguite e scelta delle variabili.

Conclusioni

Questa relazione descrive un piccolo progetto scritto in linguaggio AMPL il quale usa come risolutore "Gurobi". Sono state spiegate le varie scelte riguardanti parametri, variabili e vincoli utilizzati, servendosi di alcuni semplici esempi. Successivamente sono stati scelti due esempi: il primo in cui i valori della matrice dei livelli di interesse sono stati scelti in modo randomico, e questo può portare ad una soluzione nulla del problema; il secondo in cui è stato effettuato una piccola modifica ad un esempio funzionante che ha portato la non possibilità di ritrovare una soluzione valida per il problema.

Per questo problema si è deciso di risolvere un girone singolo all'italiana, in cui ogni partecipante sfida una sola volta ogni altro partecipante. Nel caso in cui ogni partecipante dovesse sfidare due volte tutti gli altri giocatori, effettuando quindi una partita di andata e una di ritorno, questo verrebbe chiamato girone multiplo all'italiana.

Inoltre nel nostro problema il numero di squadre è un numero N pari ($N=2n$). Nel caso di un numero dispari di squadre ($N=2n+1$) ad ogni squadra verrebbe alternatamente abbinato l'elemento "riposo".

Si potrebbero quindi aggiungere ulteriori vincoli, parametri, variabili, con lo scopo di implementare un girone multiplo, considerando un numero di squadre N generico. Le aggiunte possibili per arricchire l'assegnamento sono quindi molteplici.