El entrenador de un equipo de básquet tiene que decidir la alineación para el próximo partido, a partir de la plantilla que consta de 7 jugadores. Cada jugador de la plantilla puede jugar en una o varias de las posiciones: alero (A), base (B), o pivot (P). Adicionalmente, se ha evaluado el rendimiento de cada jugador de la plantilla en una escala de 1-3 (1: flojo; 2: correcto; 3: excelente), para cada una de las siguientes habilidades: manejo del balón, tiro, rebote y capacidad defensiva. La Tabla presenta las posiciones en las que cada jugador puede jugar, así como sus puntuaciones para cada habilidad.

| Jugador | Posibles posiciones | Manejo balón (Ma) | Tiro (Ti) | Rebote (Re) | Defensa (De) |
|---|---|---|---|---|---|
| 1 | P | 3 | 3 | 1 | 3 |
| 2 | B | 2 | 1 | 3 | 2 |
| 3 | A, P | 2 | 3 | 2 | 2 |
| 4 | A, B | 1 | 3 | 3 | 1 |
| 5 | A, P | 1 | 3 | 1 | 2 |
| 6 | A, B | 3 | 1 | 2 | 3 |
| 7 | A, P | 3 | 2 | 2 | 1 |

Parte A. La alineación para el próximo partido debe de tener 5 jugadores. El entrenador desea encontrar la alineación que maximice la capacidad defensiva de los jugadores alineados.

1. (3.5 p) Formula un modelo de programación lineal entera para encontrar la alineación óptima. Implementa el modelo con OPTMODEL y obtén la solución.

**Parámetros:**

$m$=5: número de jugadores que se deben alinear

$I$={1,…, m} conjunto de índices para los jugadores.

$P$={$A$, $B$, $P$} conjunto de posiciones posibles.

$H$={$Ma$, $Ti$, $Re$, $De$} conjunto de habilidades

Para cada $i \in I$ :

$P_i \subseteq P$ : conjunto de posiciones posibles para el jugador $i$.

Para cada $h \in H$:

$v_{ih}$: valoración del jugador $i \in I$ para la habilidad $h$.

**Variables de decisión:**

$x_i \in \{0,1\}$, $i \in I$. $x_i = 1 \Leftrightarrow$ el jugador $i \in I$ está en la alineación.

**Formulación:**

Función objetivo: $Max \sum_{i \in I} v_{iDe} x_i$

Restricciones: $\sum_{i \in I} x_i = m$ (la alineación tiene $m$ jugadores)

Dominio: $x_i \in \{0,1\}$, $i \in I$.

PARTE 1

The OPTMODEL Procedure

| Solution Summary | |
|---|---|
| **Solver** | MILP |
| **Algorithm** | Branch and Cut |
| **Objective Function** | OF |
| **Solution Status** | Optimal |
| **Objective Value** | 12 |
| **Iterations** | 5 |
| **Best Bound** | 12 |
| **Nodes** | 1 |
| | |
| **Relative Gap** | 0 |
| **Absolute Gap** | 0 |
| **Primal Infeasibility** | 0 |
| **Bound Infeasibility** | 0 |
| **Integer Infeasibility** | 0 |

| [1] | X |
|---|---|
| **1** | 1 |
| **2** | 1 |
| **3** | 1 |
| **4** | 0 |
| **5** | 1 |
| **6** | 1 |
| **7** | 0 |

Los jugadores 4 y 7 no se alinean (porque son los que tienen menor valoración: 1).

2. (2.5 p) El entrenador nos ha comunicado que como mínimo 4 jugadores deben ser capaces de jugar como *alero*; como mínimo 2 jugadores como *pivot* y al menos un jugador como *base*. Extiende la formulación anterior para tener en cuenta estas consideraciones, y resuelve también con OPTMODEL.

**Más parámetros:**

Para cada posición $j \in P$ :

$n_j$ : número mínimo de jugadores que deben alinearse que pueden jugar en posición $j$.

($n_A$=4; $n_P$=2; $n_B$=1)

**Restricciones:**

Para cada $j \in P$ :

$$\sum_{i \in I:\, j \in P_i} x_i \geq n_j$$

PARTE 2

The OPTMODEL Procedure

| Solution Summary | |
|---|---|
| **Solver** | MILP |
| **Algorithm** | Branch and Cut |
| **Objective Function** | OF |
| **Solution Status** | Optimal |
| **Objective Value** | 11 |
| **Iterations** | 6 |
| **Best Bound** | 11 |
| **Nodes** | 1 |
| **Relative Gap** | 0 |
| **Absolute Gap** | 0 |
| **Primal Infeasibility** | 0 |
| **Bound Infeasibility** | 0 |
| **Integer Infeasibility** | 0 |

| [1] | X |
|---|---|
| **1** | 1 |
| **2** | 0 |
| **3** | 1 |
| **4** | 1 |
| **5** | 1 |
| **6** | 1 |
| **7** | 0 |

Ahora los que no entran en la alineación son el 2 y el 7.

3. (2p) Comprueba si la solución óptima del modelo anterior cumple las condiciones que se presentan a continuación. En caso contrario, indica en cada caso la restricción (o restricciones) que se deberían añadir para que se cumplan (no es necesario que las implementes en OPTMODEL):

   - Si el jugador 1 está en la alineación, entonces los jugadores 4 y 5 también deben ambos estar.

     $x_1 \leq x_4$

     $x_1 \leq x_5$

     O también: $2\,x_1 \leq x_4 + x_5$ [Se cumple]

   - Si el jugador 2 está en la alineación, entonces el jugador 6 no puede estar en la alineación.

     $x_2 + x_6 \leq 1$ [Se cumple]

   - Si el jugador 2 está alineado el 7 está en el banquillo, y viceversa.

     $x_2 + x_7 = 1$ [No se cumple]

Parte B. Debido a los malos resultados obtenidos en los últimos tiempos por el equipo, se ha contratado a un nuevo entrenador. El nuevo entrenador desea diseñar la alineación de manera que cada jugador esté asignado a una única posición. En concreto quiere que la alineación conste exactamente de 2 *aleros,* 2 *pivots* y un *base*. Por otro lado, mantiene el mismo criterio de maximizar la capacidad defensiva de la alineación, aunque ya no considera necesario tener en cuenta las consideraciones del apartado A3.

4. (2p) Formula un nuevo modelo de programación lineal entera para encontrar la alineación óptima (no es necesario implementarlo con OPTMODEL).

   Nuevos Parámetros:

   Para cada posición $j \in P$:
   $r_j$ : número (exacto) de jugadores que deben alinearse en la posición $j$.
   ($r_A$=2; $n_P$=2; $n_B$=1)

**Variables de decisión:**

$x_{ij} \in \{0,1\}$, $i \in I$, $j \in P$.

$x_{ij} = 1 \Leftrightarrow$ el jugador $i \in I$ está en la alineación en la posición $j \in P$.

**Formulación:**

Función objetivo: $Max \sum_{i \in I} v_{iDe} \sum_{j \in P} x_{ij}$

Restricciones: $\sum_{i \in I} \sum_{j \in P} x_{ij} = m$ (la alineación tiene en total $m$ jugadores)

$$\sum_{i \in I: j \in P_i} x_{ij} = r_j \quad \text{para cada } j \in P \quad \text{(la alineación tiene } r_j \text{ jugadores en la posición } j\text{)}$$

$$\sum_{j \in P} x_{ij} \leq 1, \text{ para cada } i \in I \quad \text{(cada jugador se asigna como mucho a una posición)}$$

Dominio: $x_{ij} \in \{0,1\}, i \in I, j \in P_i$

**Formulación alternativa:** Se puede evitar la restricción de que la alineación tenga *m* jugadores redefiniendo las variables de decisión, para restringirlas a las posiciones en las que cada jugador puede jugar:

$x_{ij} \in \{0,1\}, i \in I, j \in P_i$.

$x_{ij} = 1 \Leftrightarrow$ el jugador $i \in I$ está en la alineación en la posición $j \in P_i$.

**Formulación:**

Función objetivo: $Max \sum_{i \in I} v_{iDe} \sum_{j \in P_i} x_{ij}$

Restricciones: $\sum_{i \in I: j \in P_i} x_{ij} = r_j$ para cada $j \in P$ (la alineación tiene $r_j$ jugadores en la posición *j*)

$$\sum_{j \in P} x_{ij} \leq 1, \text{ para cada } i \in I \quad \text{(cada jugador se asigna como mucho a una posición)}$$

Dominio: $x_{ij} \in \{0,1\}, i \in I, j \in P_i$

**Código OPTMODEL:**

```
LIBNAME control3 ".";

data control3.posic;
input   Pos $ ;
datalines;
      A
      B
      P
;

data control3.habil;
input   H $ ;
datalines;
      manejo
      tiro
      rebote
      defensa
```

```
;

data control3.jugadores;
input       j   pp_1 $ pp_2 $ pun_manejo pun_tiro pun_rebote pun_defensa ;
datalines;
    1 P     .     3     3     1     3
      2     B     .     2     1     3     2
      3     A     P     2     3     2     2
      4     A     B     1     3     3     1
      5     A     P     1     3     1     2
      6     A     B     3     1     2     3
      7     A     P     3     2     2     1
;

proc optmodel;

set <str> Pos;
set <str> H;
set <number> Jug;
string posible {Jug, 1..2};
number pun {Jug, H};

read data control3.posic into Pos=[Pos];

read data control3.habil into H=[H];

read data control3.jugadores into
      Jug=[j]
      {i in 1..2} <posible[j, i]=col("pp_"||i)>
      {k in H} <pun[j, k]=col("pun_"||k)>
;
number min_pos{Pos}=[4 1 2];
var X {Jug} binary;

maximize OF = sum {j in Jug} pun[j, "defensa"] * X[j];

con equipo: sum {j in Jug} X[j] = 5 ;

/* bloque de preguntas 2 */

con perfil {p in Pos}: sum {j in Jug : or {k in 1..2} posible[j,k] EQ p} X[j]
>= min_pos[p];

expand;

solve with MILP ;

print X;

run;
```