Gerald Jones, Kimon Swanson, Alex Hines

Home Work 4, Problem 2: Swimmer to Swim Stroke Assignment

ISE522 Spg 22

Notebook Links/Sections:

Problem Description:

> A decision needs to be made about which of a team of swimmers will be assigned to compete in which type of swimming stroke style based on their expected times for each. There are 4 swimmers as shown in the Data Display section with the shown times in seconds for each of the four different types of stroke. The task requires that each swimmer be assigned to one of the swimming stroke competitions that will lead to the minimum cumulative time for the team.

Notes and Observations

- quick ad hoc solution would be just assign based on the lowest swim time for each stroke type

Assumptions:

Module imports and data loading

In [1]:
```
from _GUROBI_TOOLS_.GUROBI_MODEL_BUILDING_TOOLS import *
from _NOTE_BOOK_UTILS import *
notebook_name = "_HW4_Problem2.ipynb"

# define dataframe for swimmer data
swimmer_df = pd.DataFrame(
    {
        "Swimmer":["Gary Hall", "Mark Spitz", "Jim Montgomery", "Chet Jastremski"],
        "Free":[54, 51, 50, 56,],
        "Breast":[54, 57, 53, 54],
        "Fly":[51, 52, 54, 55],
        "Back":[53, 52, 56, 53],
    }
)
```

Data Display

In [2]:
```
# display data for problem
display(swimmer_df)
```

	Swimmer	Free	Breast	Fly	Back
0	Gary Hall	54	54	51	53
1	Mark Spitz	51	57	52	52
2	Jim Montgomery	50	53	54	56
3	Chet Jastremski	56	54	55	53

Model Formulation

- Paremeters and Sets
- Variables
- Equations and Constraints
- Objective

Parameters and Sets:

$\mathbf{M}$ set of team members, $m \in \mathbf{M}$

$\mathbf{S}$ set of swimming strokes, $s \in \mathbf{S}$

$T_{m,s}$ expected time for member m in stroke s,

Variables:

$X_{m,s}$ 1 if member m is assigned to stroke s, 0 otherwise

Equations and Constraints:

> ## Member assigned to single stroke Constraint

$$\sum_{s=1}^{|S|} X_{m,s} = 1, \ \forall w, s$$

> ## Single Member assigned to a stroke Constraint

$$X_{a,s} + X_{b,s} \leq 1, \ \forall a, b, s \text{ where a != b}, \ \{a, b\} \subset M, \ \ s \in S$$

Objective:

$$\min(\sum_{m=1}^{|M|} \sum_{s=1}^{|S|} X_{m,s} \cdot T_{m,s})$$

Method Definitions

In [3]:
```
# generate objective
def generate_total_swim_time(model, X, T, M, S):
    expression = 0
    for m in range(M):
        for s in range(S):
            expression += X[m,s] * T[m,s]
    return expression



# generate constraints
def member_assigment_constraint(model, X, M, S):
    # for each member
    for m in range(M):
        expression = 0
        # ensure that only one of its binary selectors is 1
        for s in range(S):
            expression += X[m, s]
        model.addConstr(expression == 1)
    return

def single_member_assignment_constraint(model, X, M, S):
    # for each stroke
    for s in range(S):
        expression = 0
        # ensure that only one member of the team can be assigned at once
        for m in range(M):
            # only one member can be assigned to the current stroke
            # by ensuring the maximum of the sum of the binary variables is 1
            for m2 in range(M):
                if m != m2:
                    model.addConstr(X[m, s] + X[m2, s] <= 1)
                    model.addConstr(X[m, s] + X[m2, s] >= 0)
    return
```

Gurobi Implementation and Solution

In [4]:
```
try:
    # instantiate model object
    m = gp.Model("G_MOD")


    ########################################################################################
    ################################### Parameters set up ##################################
    ########################################################################################
    M = len(swimmer_df)
    Tms = swimmer_df.loc[:, "Free":"Back"].values
    swim_strokes = swimmer_df.loc[:, "Free":"Back"].columns.tolist()
    S = len(swim_strokes)
    swimmers = swimmer_df.loc[:, "Swimmer"].tolist()
    print(Tms)
    print(swim_strokes)
    print(swimmers)

    ########################################################################################
    ################################### Variables set up ###################################
    ########################################################################################
    Xms = m.addVars(len(swimmers), len(swim_strokes), vtype=GRB.BINARY, name="X", lb=0, ub=1)

    ########################################################################################
    ################################### Objective set up ###################################
    ########################################################################################
    m.setObjective(generate_total_swim_time(m, Xms, Tms, M, S), GRB.MINIMIZE)

    ########################################################################################
    ################################### Constraint set up ##################################
    ########################################################################################
    member_assigment_constraint(m, Xms, M, S)
    single_member_assignment_constraint(m, Xms, M, S)
    ########################################################################################
    ################################### SOLVE:OPTIMIZE #####################################
    ########################################################################################


    m.optimize()

    ########################################################################################
    ################################### Display Results ####################################
    ########################################################################################
    displayDecisionVars(m, end_sentinel=",3")

    print("\n-------------Does it make sense?----------------------")
    print('Obj: {:.2f}'.format(m.ObjVal))


# catch some math errors
except gp.GurobiError as e:
    print('Error code ' + str(e.errno) + ': ' + str(e))

except AttributeError:
    print('Encountered an attribute error')
```

```
Restricted license - for non-production use only - expires 2023-10-25
[[54 54 51 53]
 [51 57 52 52]
 [50 53 54 56]
 [56 54 55 53]]
['Free', 'Breast', 'Fly', 'Back']
['Gary Hall', 'Mark Spitz', 'Jim Montgomery', 'Chet Jastremski']
Gurobi Optimizer version 9.5.0 build v9.5.0rc5 (win64)
Thread count: 6 physical cores, 12 logical processors, using up to 12 threads
Optimize a model with 100 rows, 16 columns and 208 nonzeros
Model fingerprint: 0x3f5585a7
Variable types: 0 continuous, 16 integer (16 binary)
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [5e+01, 6e+01]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 1e+00]
Found heuristic solution: objective 208.0000000
Presolve removed 92 rows and 0 columns
Presolve time: 0.00s
Presolved: 8 rows, 16 columns, 32 nonzeros
Variable types: 0 continuous, 16 integer (16 binary)

Root relaxation: objective 2.070000e+02, 7 iterations, 0.00 seconds (0.00 work units)

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

*    0     0               0     207.0000000  207.00000  0.00%     -    0s

Explored 1 nodes (7 simplex iterations) in 0.01 seconds (0.00 work units)
Thread count was 12 (of 12 available processors)

Solution count 2: 207 208

Optimal solution found (tolerance 1.00e-04)
Best objective 2.070000000000e+02, best bound 2.070000000000e+02, gap 0.0000%
X[0,0] -0.00000
X[0,1] -0.00000
X[0,2] 1.00000
X[0,3] -0.00000

X[1,0] 0.00000
X[1,1] -0.00000
X[1,2] 0.00000
X[1,3] 1.00000

X[2,0] 1.00000
X[2,1] -0.00000
X[2,2] -0.00000
X[2,3] -0.00000

X[3,0] -0.00000
X[3,1] 1.00000
X[3,2] -0.00000
X[3,3] 0.00000

-------------------------------------



-------------Does it make sense?----------------------
Obj: 207.00
```

Solution Discussion

> ## The solution....
>
> The problem requires the optimal assignment of team member to swimming stroke to get the minimum overall time. A logical approach would be to assign team members based on which has the lowest time for a given stroke. Looking down each column in the shown data the member with the lowest time for that column would be assigned that stroke. The assignment would go from the lowest overall chosen time to the largest. This would lead to Jim assigned to the Free stroke, Gary being assigned to the Fly stroke, Mark being assigned to the Back stroke, and since Jim is already assigned to the free stroke Chet is assigned to the Breast stroke. The solution generated by Gurobi does this exactly.
>
> ### The optimal solution generated by the implemented model suggests to:
>
> - assign **Jim Montegomery** to the **Free stroke** (50 s)
> - assign **Gary Hall** to the **Fly stroke** (51 s)
> - assign **Mark Spitz** to the **Back stroke** (52 s)
> - assign **Chet Jastremski** to the **Breast stroke** (54 s)
> - **This leads to an overall team time of 50 + 51 + 52 + 54 = 207 seconds**

In [5]:
```
# save the notebook as a pdf
to_PDF(notebook_name)
```

```
filename: _HW4_Problem2.ipynb
```