MAT3007 - Assignment 9

Yohandi

December 15, 2023

Problem 1: Eight Queens Problem

Let x_{ij} be a binary variable that equals to 1 if a queen is placed on the chessboard at row i and column j, and 0 otherwise. Each constraint is addressed as follows:

- No two queens on the same row.
  $\sum_{j=1}^{8} x_{ij} = 1, \forall i \in \{1, \ldots, 8\}$
- No two queens on the same column.
  $\sum_{i=1}^{8} x_{ij} = 1, \forall j \in \{1, \ldots, 8\}$
- No two queens on the same diagonal lines.
  For the $i - j$ diagonals: $\sum_{k=-(8-1)}^{(8-1)} \sum_{(i,j)\in \mathrm{R}\times\mathrm{C}, i-j=k} x_{ij} \le 1$
  For the $i + j$ diagonals: $\sum_{k=2}^{(2\cdot 8-2)} \sum_{(i,j)\in \mathrm{R}\times\mathrm{C}, i+j=k} x_{ij} \le 1$

The objective function for this problem is not needed; we simply want to find one of the configurations that satisfy all the constraints.

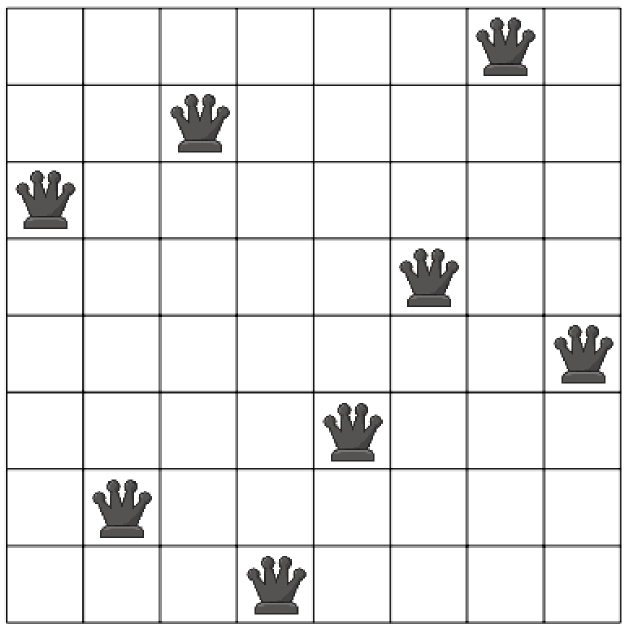
Python code:

```python
import cvxpy as cp

n = 8
x = cp.Variable((n,n), boolean=True)

constraints = [cp.sum(x, axis=0) == 1, cp.sum(x, axis=1) == 1]
for k in range(-n+2, n):
    constraints.append(cp.sum([x[i,j] for i in range(n) for j in range(n)
      if i-j == k]) <= 1)
for k in range(3, n*2-1):
    constraints.append(cp.sum([x[i,j] for i in range(n) for j in range(n)
      if i+j == k]) <= 1)

objective = cp.Maximize(1) # Any objective function doesn't matter

cp.Problem(objective, constraints).solve()
print(x.value)
```

Console output:

```
[[0. 0. 0. 0. 0. 0. 1. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 1.]
 [0. 0. 0. 0. 1. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0. 0.]]
```

The solution to the problem is visualized as follows:



Problem 2: Branch-and-Bound Method

```
from scipy.optimize import linprog
import numpy as np

c = [-17, -12]
A = [[3, 4], [-2, 1]]
b = [25, -2]

x_bounds, y_bounds = (0, None), (0, None)
result = linprog(c, A_ub=A, b_ub=b, bounds=[x_bounds, y_bounds], method='highs')

print("Objective value:", -result.fun, "with pair (x, y) =", result.x[0], result.x[1])
```

```
Objective value: 141.66666666666669 with pair (x, y) = 8.333333333333334 0.0
```

The optimal solution to the current problem is $(8.\bar{3}, 0)$ with objective value $141.\bar{6}$. As x is not an integer, we perform a branch on this variable.

We create two subproblems: $x \leq 8$ and $x \geq 9$.

```python
result_sub1 = linprog(c, A_ub=A + [[1, 0]], b_ub=b + [8], bounds=[x_bounds, y_bounds], method='highs')
max_value_sub1 = -result_sub1.fun if result_sub1.success else None
solution_sub1 = result_sub1.x if result_sub1.success else None

print("x <= 8:")
if max_value_sub1 != None:
    print("Objective value:", max_value_sub1, "with pair (x, y) =", solution_sub1[0], solution_sub1[1])
else:
    print("[infeasible]")

result_sub2 = linprog(c, A_ub=A + [[-1, 0]], b_ub=b + [-9], bounds=[x_bounds, y_bounds], method='highs')
max_value_sub2 = -result_sub2.fun if result_sub2.success else None
solution_sub2 = result_sub2.x if result_sub2.success else None

print("x >= 9:")
if max_value_sub2 != None:
    print("Objective value:", max_value_sub2, "with pair (x, y) =", solution_sub2[0], solution_sub2[1])
else:
    print("[infeasible]")
```

```
x <= 8:
Objective value: 139.0 with pair (x, y) = 8.0 0.25
x >= 9:
[infeasible]
```

As $x \geq 9$ is infeasible and y is not an integer, we branch into $y \leq 0$ and $y \geq 1$ along with $x \leq 8$.

```python
result_sub1_y1 = linprog(c, A_ub=A + [[1, 0], [0, 1]], b_ub=b + [8, 0], bounds=[x_bounds, y_bounds], method='highs')
max_value_sub1_y1 = -result_sub1_y1.fun if result_sub1_y1.success else None
solution_sub1_y1 = result_sub1_y1.x if result_sub1_y1.success else None

print("x <= 8 and y <= 0:")
if max_value_sub1_y1 != None:
    print("Objective value:", max_value_sub1_y1, "with pair (x, y) =", solution_sub1_y1[0], solution_sub1_y1[1])
else:
    print("[infeasible]")

result_sub1_y2 = linprog(c, A_ub=A + [[1, 0], [0, -1]], b_ub=b + [8, -1], bounds=[x_bounds, y_bounds], method='highs')
max_value_sub1_y2 = -result_sub1_y2.fun if result_sub1_y2.success else None
solution_sub1_y2 = result_sub1_y2.x if result_sub1_y2.success else None

```

```python
print("x <= 8 and y >= 1:")
if max_value_sub1_y1 != None:
    print("Objective value:", max_value_sub1_y2, "with pair (x, y) =",
    solution_sub1_y2[0], solution_sub1_y2[1])
else:
    print("[infeasible]")
```

```
x <= 8 and y <= 0:
Objective value: 136.0 with pair (x, y) = 8.0 -0.0
x <= 8 and y >= 1:
Objective value: 131.0 with pair (x, y) = 7.0 1.0
```

Clearly, from the result, we notice that the pair $(x, y) = (8, 0)$ gives a better objective result than the pair $(x, y) = (7, 1)$. Final result is obtained with the pair $(x, y) = (8, 0)$ and objective value 136.