Assignment 2 - CS703 Optimisation and Computing

HOANG Thi Linh

April 14, 2025

Question 1

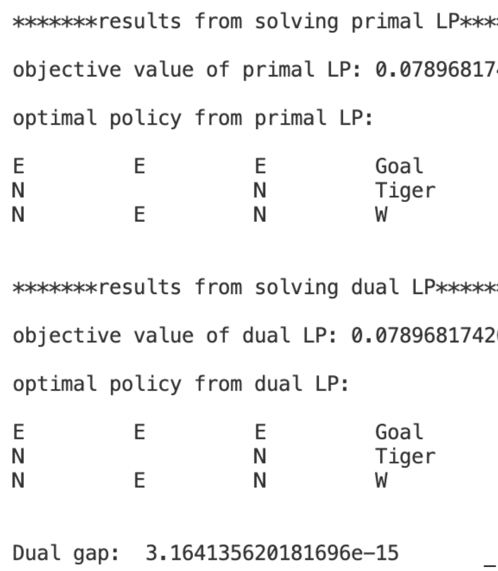
(a) Implement dual solver for LP MDP

```python
def solve_MDP_LP_dual(states, actions, trans_probs, reward, b_0, gamma):
    # TODO:create Variables
    V = cp.Variable()
    x = cp.Variable((len(states), len(actions)), nonneg=True) # Variables x(s,a)
    r_sa = np.zeros((len(states), len(actions)))
    for s in states:
        for a in actions:
            r_sa[s, a] = trans_probs[s, a, :] @ reward[s, a, :]

    # objective function = maximize x(s,a)*reward(s,a)
    objective = cp.Maximize(cp.sum(cp.multiply(x, r_sa)))
    constraints = []
    # Constraints
    # flow constraint sum(x(s',a')) = b_0(s') + gamma * sum(x(s,a))
    for sprime in states:
        lhs = cp.sum(x[sprime, :])  # sum_{a'} x(s',a')
        # sum_{s,a} P(s'|s,a)* x(s,a):
        flow_in = cp.sum(cp.multiply(trans_probs[:, :, sprime], x))
        rhs = b_0[sprime] + gamma * flow_in
        constraints.append(lhs == rhs)

    # TODO:solve the dual LP problem
    prob = cp.Problem(objective, constraints)
    prob.solve(solver=cp.GLPK, verbose=False)

    # TODO: extract deterministic policy pi(s)
    x_opt = x.value
    pi = [None] * len(states)
    pi = [np.argmax(x_opt[s, :]) for s in states]

    return prob.value, V.value, pi
```

Figure 1: Screenshot of implementation of solve_MDP_LP_dual

(b)

The optimal objective value of the dual LP is **0.0789681742618707**, which is equal to the primal LP value **0.078968174261873864** up to numerical precision. The small difference (dual gap $\approx 3.16 \times 10^{-16}$) confirms that strong duality holds, as expected for MDPs.



```
*******results from solving primal LP*******

objective value of primal LP: 0.07896817426187386

optimal policy from primal LP:

E        E        E        Goal
N                 N        Tiger
N        E        N        W

*******results from solving dual LP*******

objective value of dual LP: 0.0789681742618707

optimal policy from dual LP:

E        E        E        Goal
N                 N        Tiger
N        E        N        W


Dual gap:  3.164135620181696e-15
```

Figure 2: Screenshot of running of solve_MDP_LP_dual

(c) The derivation of dual of the MDP Dual LP

We consider the MDL Dual LP in this question:

$$\begin{aligned}
&\underset{x}{\text{maximize}} \quad \sum_{s,a} x(s,a)\, R(s,a) \\
&\text{subject to} \quad \sum_{a'} x(s',a') = b_0(s') \;+\; \gamma \sum_{s,a} P(s' \mid s,a)\, x(s,a) \quad \forall s' \\
&\qquad\qquad\quad\; x(s,a) \geq 0 \quad \forall s,a
\end{aligned} \tag{1}$$

Introduce a dual variable $V(s) \in \mathbb{R}$ for each equality constraint at state s, the Lagrangian function associated with (1) is:

$$\begin{aligned}
L(x,V) &= \sum_{s,a} x(s,a)\, R(s,a) \;+\; \sum_{s'} V(s') \Big[-\sum_{a} x(s',a) \;+\; b_0(s') \;+\; \gamma \sum_{s,a} P(s' \mid s,a)\, x(s,a) \Big] \\
&= \sum_{s'} V(s')\, b_0(s') \;+\; \sum_{s,a} x(s,a) \Big[R(s,a) \;-\; V(s) \;+\; \gamma \sum_{s'} P(s' \mid s,a)\, V(s') \Big]
\end{aligned}$$

To ensure $\sup\limits_{x \geq 0} L < +\infty$ each coefficient of $x(s,a)$ must be ≤ 0:

$$R(s,a) \;-\; V(s) \;+\; \gamma \sum_{s'} P(s' \mid s,a)\, V(s') \leq 0 \Leftrightarrow V(s) \geq R(s,a) \;+\; \gamma \sum_{s'} P(s' \mid s,a)\, V(s') \quad \forall s,a$$

then the dual of problem (1) is the same as the MDP primal LP as:

$$\begin{aligned}
&\underset{V}{\text{minimize}} \quad \sum_{V} V(s)\, b_0(s) \\
&\text{subject to} \quad V(s) \geq R(s,a) \;+\; \gamma \sum_{s'} P(s' \mid s,a)\, V(s') \quad \forall s,a
\end{aligned} \tag{2}$$

Question 2

(a) Formulating optimization problem.

We consider a profit maximization problem (what is the best selling price P to maximize the profit) for a newly designed fashion bag. The setup involves the following parameters:

- A fixed cost of \$700,000 is required for manufacturing setup and marketing.
- Each bag has a production cost of \$110.
- Market demand follows the linear relation: Customer Demand = 70000 - P, with P is the selling price per unit.
- Due to capacity constraints, production is limited to a maximum of 30,000 bags.

Let n denote the number of bags produced and sold. Then, maximizing the total profit as this problem:

$$\begin{aligned} \underset{n,\,P}{\text{maximize}} \quad & nP - 110n - 700000 \\ \text{subject to} \quad & n \leq 70000 - P \\ & n \leq 30000 \\ & n \geq 0 \\ & P \geq 0 \end{aligned} \tag{3}$$

which is equivalent to:

$$\begin{aligned} \underset{n,\,P}{\text{minimize}} \quad & f(n,P) = -nP + 110n + 700000 \\ \text{subject to} \quad & n + P - 70000 \leq 0 \\ & n - 30000 \leq 0 \\ & -n \leq 0 \\ & -P \leq 0 \end{aligned} \tag{4}$$

(b) Solving problem

We will then solve the problem from (4) by introducing the Lagrangian multipliers $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ for constraints and each $\lambda_i \geq 0$. The Lagrangian function associated of this problem is:

$$L(n,P,\lambda) = -nP + 110n + 700000 + \lambda_1(n + P - 70000) + \lambda_2(n - 30000) - \lambda_3 n - \lambda_4 P \tag{5}$$

The KKT conditions:

- Stationarity:
$$\frac{\partial L}{\partial n} = -P + 110 + \lambda_1 + \lambda_2 - \lambda_3 = 0, \frac{\partial L}{\partial P} = -n + \lambda_1 - \lambda_4 = 0.$$
- Complementary Slackness:
$$\lambda_1(n + P - 70000) = 0,$$
$$\lambda_2(n - 30000) = 0,$$
$$\lambda_3 n = 0,$$
$$\lambda_4 P = 0.$$
- Primal feasibility:
$$n + P \leq 70000, n \leq 30000, n \geq 0, P \geq 0.$$
- Feasibility:
$$\lambda_1 \geq 0, \lambda_2 \geq 0, \lambda_3 \geq 0, \lambda_4 \geq 0.$$

Solving the system of KKT conditions we have $n^* = 30000, P^* = 40000, \lambda_1 = 30000, \lambda_2 = 9890, \lambda_3 = \lambda_4 = 0$.

Question 3

(a) Problem Formulation

Variables. Let

$$x_{ij}^k = \begin{cases} 1, & \text{if agent } k \in \{1,2\} \text{ traverses edge } (i,j) \in E, \\ 0, & \text{otherwise.} \end{cases}$$

Objective. Minimize the total traversal cost: $\min\limits_x \sum\limits_{k=1}^{2} \sum\limits_{(i,j)\in E} c_{ij}\, x_{ij}^k$.

Constraints.

- Source departure: $\sum\limits_{j:(s_k,j)\in E} x_{s_k j}^k = 1, \quad k = 1,2.$
- Sink arrival: $\sum\limits_{i:(i,d_k)\in E} x_{i d_k}^k = 1, \quad k = 1,2.$
- Flow conservation:
$$\sum_{i:(i,v)\in E} x_{iv}^k \; - \sum_{j:(v,j)\in E} x_{vj}^k = 0, \quad k = 1,2,\; v \neq s_k, d_k.$$
- Coverage: $\sum\limits_{k=1}^{2} \sum\limits_{i:(i,v)\in E} x_{iv}^k \;\geq 1, \quad \forall\, v \in V.$
- Binary: $x_{ij}^k \in \{0,1\}, \quad k = 1,2,\; (i,j) \in E.$

The optimization problem is

$$\begin{aligned}
\min_x \quad & \sum_{k=1}^{2} \sum_{(i,j)\in E} c_{ij}\, x_{ij}^k \\
\text{s.t.} \quad & \sum_{j:(s_k,j)\in E} x_{s_k j}^k = 1, && k = 1,2, \\
& \sum_{i:(i,d_k)\in E} x_{i d_k}^k = 1, && k = 1,2, \\
& \sum_{i:(i,v)\in E} x_{iv}^k \; - \sum_{j:(v,j)\in E} x_{vj}^k = 0, && k = 1,2,\; v \neq s_k, d_k, \\
& \sum_{k=1}^{2} \sum_{i:(i,v)\in E} x_{iv}^k \;\geq\; 1, && \forall\, v \in V, \\
& x_{ij}^k \in \{0,1\}, && k = 1,2,\; (i,j) \in E.
\end{aligned}$$

Computational tractability. This is a mixed-integer program with binary routing and global coverage constraints and is NP-hard, so it is not solvable in polynomial time in the worst case. Therefore, the problem is not tractable for large graphs.

(b) Dual Decomposition.

Relax the coverage constraints $\sum_k \sum_{i:(i,v)\in E} x_{iv}^k \geq 1$ with multipliers $\lambda_v \geq 0$. The Lagrangian separates by agent:

$$
\begin{aligned}
L(x,\lambda) &= \sum_{k=1}^{2} \sum_{(i,j)\in E} c_{ij}\, x_{ij}^k \;+\; \sum_{v\in V} \lambda_v \Big(1 - \sum_{k=1}^{2} \sum_{i:(i,v)\in E} x_{iv}^k\Big) \\
&= \sum_{v\in V} \lambda_v \;+\; \sum_{k=1}^{2} \underbrace{\sum_{(i,j)\in E} (c_{ij} - \lambda_j) x_{ij}^k}_{\Phi_k(x^k;\lambda)} .
\end{aligned}
$$

Hence the dual is

$$
\max_{\lambda\geq 0} \Big[\sum_{v\in V} \lambda_v \;+\; \sum_{k=1}^{2} \min_{x^k\in\mathcal{X}_k} \Phi_k(x^k;\lambda) \Big],
$$

where each subproblem for agent k is

$$
\min_{x^k\in\mathcal{X}_k} \sum_{(i,j)\in E} \big(c_{ij} - \lambda_j\big)\, x_{ij}^k,
$$

and $\mathcal{X}_k$ encodes that agent's flow-conservation and integrality.

(c) Tractability of Subproblems.

Each subproblem is a *shortest-path* or *min-cost-flow* problem with nonnegative modified costs $c_{ij} - \lambda_j$. Such problems can be solved in polynomial time , so each subproblem is tractable.

Question 4

Consider the problem

$$
\begin{aligned}
&\min_{x_1,x_2,x_3>0} \quad \max\Big\{ \tfrac{x_1}{x_2},\; \tfrac{\sqrt{x_3}}{x_2} \Big\} \\
&\text{subject to} \quad x_1^2 + \frac{2\,x_2}{x_3} \leq \sqrt{x_2} \\
&\qquad\qquad\quad \tfrac{x_1}{x_2} \geq x_3^2
\end{aligned}
\tag{6}
$$

(a) Show that the above problem can convert into a convex optimization problem.

Following [1], the standard form of a geometric programming is:

$$
\begin{aligned}
\text{minimize} \quad & f_0(x) \\
\text{subject to} \quad & f_i(x) \leq 1, \quad i = 1, ..., m \\
& g_i(x) = 1, \quad i = 1, ..., p
\end{aligned}
$$

where f_i is posynominal functions, g_i are mononials, and x_i are the optimization variables. There is an implicit constraint that the variables are positive.
Introduce $t > 0$ such that $\frac{x_1}{x_2} \leq t$, $\frac{\sqrt{x_3}}{x_2} \leq t$, we can convert the problem (6) to a geometric programming:

$$
\begin{aligned}
\text{minimize} \quad & t \\
\text{subject to} \quad & x_1\, x_2^{-1}\, t^{-1} \leq 1, \\
& x_3^{1/2}\, x_2^{-1}\, t^{-1} \leq 1, \\
& x_1^2\, x_2^{-1/2} + 2\, x_2^{1/2}\, x_3^{-1} \leq 1, \\
& x_3^2\, x_2\, x_1^{-1} \leq 1.
\end{aligned}
\tag{7}
$$

Set $y_i = \log x_i$ $(i = 1, 2, 3)$ and $u = \log t$, the problem (7) is equivalent to the convex optimization:

$$
\begin{array}{ll}
\underset{y,u}{\text{minimize}} & u \\
\text{subject to} & y_1 - y_2 - u \leq 0, \\
& \dfrac{1}{2} y_3 - y_2 - u \leq 0, \\
& \log\Big(\exp(2y_1 - \frac{1}{2}y_2) + 2\exp(\frac{1}{2}y_2 - y_3)\Big) \leq 0, \\
& 2y_3 + y_2 - y_1 \leq 0.
\end{array}
\tag{8}
$$

(b) Solve the resulting convex problem using cvxpy

The implementation of GP problem as the screenshot in Fig 3. Solving the problem we get the solution in the (y, u)–space and the corresponding original (x, t) values are:

$$
\begin{aligned}
y_1^* &= -1.3785, & x_1^* &= e^{y_1^*} \approx 0.2520, \\
y_2^* &= -1.5290, & x_2^* &= e^{y_2^*} \approx 0.2168, \\
y_3^* &= \ \ 0.0753, & x_3^* &= e^{y_3^*} \approx 1.0782, \\
u^* &= \ \ 1.5666, & t^* &= e^{u^*} \approx 4.7900.
\end{aligned}
$$

question4.py

```python
import cvxpy as cp
import numpy as np

# Variables
y1 = cp.Variable(name="y1")
y2 = cp.Variable(name="y2")
y3 = cp.Variable(name="y3")
u  = cp.Variable(name="u")

# Constraints
constraints = [
    y1 - y2 - u <= 0,  # y1 - y2 - u <= 0
    0.5*y3 - y2 - u <= 0,  # (1/2) y3 - y2 - u <= 0
    # log(exp(2y1-0.5y2) + 2 exp(0.5y2 - y3)) <= 0
    cp.log_sum_exp(cp.hstack([
        2*y1 - 0.5*y2,
        cp.log(2) + 0.5*y2 - y3
    ])) <= 0,
    2*y3 + y2 - y1 <= 0  # 2y3 + y2 - y1 <= 0
]

obj = cp.Minimize(u) # objective

# Solve
prob = cp.Problem(obj, constraints)
prob.solve()

# Convert y back to x via x_i = exp(y_i)
x1_val, x2_val, x3_val = np.exp(y1.value), np.exp(y2.value), np.exp(y3.value)

print(f"Optimal y1 = {y1.value:.4f}, y2 = {y2.value:.4f}, y3 = {y3.value:.4f}, u = {u.value:.4f}")
print(f"Corresponding x1 = exp(y1) = {x1_val:.4f}")
print(f"Corresponding x2 = exp(y2) = {x2_val:.4f}")
print(f"Corresponding x3 = exp(y3) = {x3_val:.4f}")
```

Figure 3: Screenshot of the implementation using cvxpy for the problem (8)

Question 5

On the 4×4 grid we single out three neighbouring nodes i–j–ℓ with i immediately left of j and ℓ immediately above j.

(a) Random-variable domains

For every location $v \in \{i, j, \ell\}$ define a variable

$$X_v \in \{T, D, L, R\},$$

where $T =$ top, $D =$ down, $L =$ left, $R =$ right. Thus X_v records the *single* edge that agent v chooses to scan.

(b) Pairwise potential functions

A reward is obtained only when *both* endpoints of a target edge scan *toward* each other; otherwise the reward is 0.

Edge (j, i) **(horizontal target).** Reward $t_{ji} > 0$ is earned only when j scans *left* (L) and i scans *right* (R). Hence

$$\theta_{j,i}(X_j, X_i) = \begin{cases} t_{ji}, & (X_j, X_i) = (L, R), \\ 0, & \text{otherwise.} \end{cases}$$

and potential entries in table form is:

X_j	X_i	$\theta_{j,i}(X_j, X_i)$
T	T	0
T	D	0
T	L	0
T	R	0
D	T	0
D	D	0
D	L	0
D	R	0
L	T	0
L	D	0
L	L	0
L	R	t_{ji}
R	T	0
R	D	0
R	L	0
R	R	0

Similarly, **Edge** (j, ℓ) **(vertical target).** Reward $t_{j\ell} > 0$ is earned only when j scans *top* (T) and ℓ scans *down* (D):

$$\theta_{j,\ell}(X_j, X_\ell) = \begin{cases} t_{j\ell}, & (X_j, X_\ell) = (T, D), \\ 0, & \text{otherwise.} \end{cases}$$

with table form is:

X_j	X_ℓ	$\theta_{j,\ell}(X_j, X_\ell)$
T	T	0
T	D	$t_{j\ell}$
T	L	0
T	R	0
D	T	0
D	D	0
D	L	0
D	R	0
L	T	0
L	D	0
L	L	0
L	R	0
R	T	0
R	D	0
R	L	0
R	R	0

These potentials ensure that a reward is collected *iff* both agents involved in a target edge coordinate their scan directions toward that edge; any other combination yields zero.

References

[1] Stephen Boyd et al. "A tutorial on geometric programming". In: *Optimization and engineering* 8 (2007), pp. 67–127.