Test 3

Aditya Sanjay Mhaske

Question 1

1. Show that if x is an optimal assignment, then it is compatible with any stable payoff (u, v)

To show that if x is an optimal assignment, then it is compatible with any stable payoff (u, v), we need to prove that x cannot be blocked by any pair (u', v') that is stable.

Assume that x is an optimal assignment and that (u', v') is a stable payoff that blocks x. This means that there exists a resource i such that $u'(i) < u(i)$, and that there exists a task j such that $v'(j) < v(j)$, and such that $x(i) = j$. Let $c(i,j)$ be the cost of assigning resource i to task j.

Since x is optimal, we have that:

$\sum c(i, x(i)) \leq \sum c(i, j)$ for all j,

where the summations are over all i.

Since (u', v') is stable, we have that:

$u'(i) + v'(j) \geq c(i, j)$ for all i, j,
and
$u(i) + v(j) \geq c(i, j)$ for all i, j.

Combining these two inequalities, we get:

$u'(i) - u(i) + v'(j) - v(j) \geq 0.$

But since $x(i) = j$, we have:

$c(i, x(i)) = c(i, j),$

which implies that:

$u'(i) - u(i) + v'(x(i)) - v(x(i)) \geq 0.$

Since (u', v') is a stable payoff, we have:

$u'(i) + v'(x(i)) \geq c(i, x(i)),$

which implies that:

$u'(i) - u(i) + v'(x(i)) - v(x(i)) \geq 0$.

This contradicts our assumption that (u', v') blocks x, and therefore x is compatible with any stable payoff.

2. If ((u, v), x) is a stable outcome, then x is an optimal assignment.

To show that if ((u, v), x) is a stable outcome, then x is an optimal assignment, we need to prove that x minimizes the total cost (or maximizes the total valuation) among all possible assignments.

Assume that ((u, v), x) is a stable outcome. This means that x is a one-to-one assignment of n resources to n tasks, and that (u, v) is a pair of payoffs such that:

- $u(i) + v(j) \leq c(i, j)$ for all i, j, where c(i, j) is the cost (or valuation) of assigning resource i to task j;
- $u(i) + v(j) = c(i, j)$ for all i, j such that $x(i) = j$ (i.e., for all assigned resources and tasks);
- $u(i) + v(j) > c(i, j)$ for all i, j such that $x(i) \neq j$ (i.e., for all unassigned resources and tasks).

We want to show that x minimizes the total cost (or maximizes the total valuation) among all possible assignments.

Suppose, for the sake of contradiction, that there exists another assignment y such that the total cost (or valuation) of y is less than the total cost (or valuation) of x. Let T be the set of resources that are assigned differently in x and y. Since x and y are both one-to-one assignments, we have $|T| \leq 2$.

Now, consider the following pair of payoffs (u', v'), defined as follows:

- $u'(i) = u(i)$ for all $i \notin T$;
- $u'(i) = u(i) + \varepsilon$ for all $i \in T$, where ε is a small positive number;
- $v'(j) = v(j)$ for all $j \notin T$;
- $v'(j) = v(j) - \varepsilon$ for all $j \in T$.

Note that (u', v') satisfies the first condition of stability, since $u'(i) + v'(j) = u(i) + v(j) - \varepsilon \leq c(i, j)$ for all i, j. Moreover, since ε is small, (u', v') satisfies the second and third conditions of stability as well, since:

- $u'(i) + v'(j) = u(i) + v(j) - \varepsilon = c(i, j)$ for all i, j such that $x(i) = j$ (i.e., for all assigned resources and tasks);
- $u'(i) + v'(j) = u(i) + v(j) - \varepsilon < c(i, j)$ for all i, j such that $y(i) = j$ (i.e., for all assigned resources and tasks where x and y differ);
- $u'(i) + v'(j) > c(i, j)$ for all i, j such that $i \notin T$ and $j \notin T$ (i.e., for all unassigned resources and tasks).

Therefore, ((u', v'), y) is a blocking pair for ((u, v), x), contradicting the assumption that ((u, v), x) is a stable outcome.

Thus, we conclude that x is an optimal assignment since it cannot be improved by any other assignment.

3. Let ((u, v), x) and ((u′, v′), x′) be stable outcomes of the assignment game (P, Q, α). Show
that if x′
i j = 1 and u′
i > ui implies v′j < v j.

To prove that if ((u, v), x) and ((u', v'), x') are stable outcomes of the assignment game (P, Q, α), and x' is not optimal, then there exists a blocking pair ((u'', v''), y), where y is an optimal assignment and u''i > ui for some i, j, such that x''i = j and v''j < vj.

Suppose, for the sake of contradiction, that ((u', v'), x') is a stable outcome and that x' is not optimal. This means that there exists an optimal assignment y such that the total cost (or valuation) of y is less than the total cost (or valuation) of x'. Let T be the set of resources that are assigned differently in x' and y. Since x' and y are both one-to-one assignments, we have |T| ≤ 2.

Without loss of generality, assume that x'i = j for some i, j. Let us consider the following payoffs:

- u''i = ui + ε, where ε is a small positive number;
- u''k = u'k for all k ≠ i;
- v''j = v'j - ε;
- v''l = v'l for all l ≠ j.

Note that (u'', v'') satisfies the first condition of stability, since u''i + v''j = ui + v'j - ε ≤ c(i, j) for all i, j. Moreover, (u'', v'') satisfies the second and third conditions of stability as well, since:

- u''i + v''j = ui + v'j - ε < c(i, j) for all i, j such that y(i) = j (i.e., for all assigned resources and tasks where x' and y differ);
- u''i + v''j = ui + v'j - ε = c(i, j) for all i, j such that x'(i) = j (i.e., for all assigned resources and tasks in x');
- u''i + v''j > c(i, j) for all i, j such that x(i) ≠ j and y(i) ≠ j (i.e., for all unassigned resources and tasks).

Therefore, ((u'', v''), y) is a blocking pair for ((u', v'), x'), contradicting the assumption that ((u', v'), x') is a stable outcome.

Since x' is not optimal, there exists an optimal assignment y such that the total cost (or valuation) of y is less than the total cost (or valuation) of x'. Furthermore, we have shown that there exists a blocking pair ((u'', v''), y), where y is an optimal assignment and u''i > ui for some i, j, such that x''i = j and v''j < vj. This completes the proof.

Question 2

1. For a given price vector p, define the demand set Di(p) for bidder i.

The demand set Di(p) for bidder i with respect to a given price vector p is defined as the set of objects that bidder i would choose to be assigned to if the price of every object is given by p. More formally, we can define Di(p) as follows:

$Di(p) = \{j \in Q \mid \alpha ij \geq \max\{k \in Q \mid pk \geq cj, k \neq j\}$ or i is assigned to the null object O}

In other words, the demand set for bidder i consists of all objects j such that the value of object j to bidder i is at least as high as the highest price among all objects that bidder i prefers to object j, or bidder i is assigned to the null object O.

Intuitively, the demand set Di(p) captures the set of objects that bidder i would like to be assigned to at the given prices. If the prices of the objects in Di(p) are lowered, bidder i would demand all objects in Di(p) and if the prices are increased, bidder i would demand fewer objects in Di(p).

2. A price vector p is called quasi-competitive if there is a matching μ from P to Q such that μ(i) = j then j is in Di(p), and if i is unmatched under μ then O is in Di(p). In other words, at a quasi-competitive prices p each buyer can be assigned to an object in his or her demand set. In this case, μ is said to be compatible with p. The pair (p,μ) is a competitive equilibrium if p is quasi-competitive, μ is compatible with p, and pj =cj for all $j \in \mu(P)$. In this case we denote (p,μ) is a competitive equilibrium and p is called an equilibrium price vector.
Show that if (p,μ) is a competitive equilibrium then the corresponding payoffs are stable.

To show that the payoffs in a competitive equilibrium are stable, we need to show that there is no profitable deviation for any bidder or object. In other words, we need to show that if ((u,v),μ) is a payoff vector and matching such that (u,v) is feasible and (u,v) is not blocked by any bidder or object, then ((u,v),μ) is the unique outcome of the game.

Assume that (p,μ) is a competitive equilibrium, and let ((u,v),μ) be a payoff vector and matching such that (u,v) is feasible and (u,v) is not blocked by any bidder or object. We need to show that ((u,v),μ) is the unique outcome of the game.

Since (p,μ) is a competitive equilibrium, μ is compatible with p, which means that every bidder can be assigned to an object in their demand set. Therefore, for any bidder i, we have $u_i = \max\{j \in Di(p)\} \alpha_{ij}$, and for any object j, we have $v_j = \min\{i \in P, j \in Di(p)\} u_i$.

Now, suppose for contradiction that there exists a bidder i and an object j such that $u_i + v_j < \alpha_{ij}$. This means that bidder i and object j have an incentive to form a blocking pair, which contradicts our assumption that (u,v) is not blocked by any bidder or object. Therefore, we must have $u_i + v_j \geq \alpha_{ij}$ for all i and j.

Next, suppose that there exists a bidder i and an object j such that μ(i) = j, but j is not in Di(p). This means that bidder i has an incentive to deviate and choose a different object in their demand set, which contradicts the assumption that (p,μ) is a competitive equilibrium. Therefore, we must have μ(i) ∈ Di(p) for all i.

Finally, suppose that there exists an object j ∈ μ(P) such that p_j > c_j. This means that the object j is overpriced, and any bidder who is assigned to j has an incentive to deviate and choose a different object in their demand set. This contradicts the assumption that (p,μ) is a competitive equilibrium, which requires that p_j = c_j for all j ∈ μ(P). Therefore, we must have p_j = c_j for all j ∈ μ(P).

Since all possible profitable deviations are ruled out, we conclude that ((u,v),μ) is the unique outcome of the game, which means that the payoffs in a competitive equilibrium are stable.

3. Describe an algorithm to compute an equilibrium price vector.

To compute an equilibrium price vector, we can use the following algorithm:

1. Set the price vector p to be equal to the reservation prices c for all objects, i.e., p_j = c_j for all j.
2. While p is not quasi-competitive, do the following:
    1. a. For each bidder i, compute their demand set Di(p), which consists of all objects j such that α_ij ≥ max{k ≠ j} α_ik - p_k.
    2. b. If there exists a bidder i such that Di(p) is empty, set p_i to be the maximum value of {α_ij - max{k} α_ik} + c_j over all objects j, and go back to step 2a.
    3. c. If there exists a bidder i such that there is more than one object in their demand set, randomly assign them to one of the objects in their demand set.
3. Once p is quasi-competitive, set the price of all matched objects to their corresponding bidder's value, i.e., p_j = α_μ(j) for all j ∈ μ(P).
4. Output (p,μ) as the competitive equilibrium.

This algorithm works by starting with the reservation prices and iteratively adjusting the prices to make the demand sets quasi-competitive. In each iteration, we first compute the demand sets for each bidder based on the current prices. If any bidder has an empty demand set, we increase the price of their null object until their demand set becomes non-empty. If any bidder has more than one object in their demand set, we randomly assign them to one of the objects.

Once the demand sets become quasi-competitive, we set the prices of the matched objects to their corresponding bidder's value, which guarantees that the resulting price vector is an equilibrium.

4. In your previous answer: does the algorithm generate an equilibrium price vector that maximizes total happiness?

No, the algorithm I described does not guarantee that the resulting equilibrium price vector maximizes total happiness. The algorithm aims to find a competitive equilibrium, which ensures that the demand sets are quasi-competitive and each bidder can be assigned to an object in their demand set. However, there may be multiple competitive equilibria that lead to different total happiness levels.

To maximize total happiness, we would need to modify the algorithm to take into account the bidders' preferences and find the equilibrium price vector that maximizes the sum of the bidders' values. One way to do this is to use a market-clearing algorithm that adjusts the prices until the total demand for all objects equals the total supply, while also ensuring that each bidder is assigned to their most preferred object in their demand set. This algorithm would require additional information about the bidders' preferences and their valuation functions.