Assignment and the Hungarian Method

Choose one worker for each job at minimum total cost.

> We will use the dual constraints to identify promising worker–job pairs, improve a partial assignment, and certify the final answer.

Study sequence	What you should be able to do
Model	Write the assignment LP and its dual
Hungarian method	Find zero reduced-cost pairs, adjust the dual, and complete a matching
Network simplex next	Connect one-unit assignments to general minimum-cost flow

The auction example is optional appendix reading. It is not needed for network simplex.

Four technicians, four jobs

A company must complete four jobs. Each technician can do exactly one job, and each job needs exactly one technician.

The table gives the cost c_{ij} of assigning worker i to job j. All 16 assignments are allowed.

> Choose four entries: one in every row and one in every column. Minimize their sum.

Original costs

Worker	Job 1	Job 2	Job 3	Job 4
W1	10	19	8	15
W2	10	18	7	17
W3	13	16	9	14
W4	12	19	8	18

From a whole assignment to an LP

Let x_{ij} be the fraction of job j assigned to worker i. For n workers and n jobs, a whole assignment uses 1 for a chosen pair and 0 otherwise.

$$\begin{aligned} \min \quad & \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \\ \text{s.t.} \quad & \sum_{j} x_{ij} = 1 \quad (\forall i), \\ & \sum_{i} x_{ij} = 1 \quad (\forall j), \\ & x_{ij} \geq 0. \end{aligned}$$

This continuous LP permits fractions, but an optimal whole assignment exists. The Hungarian method will construct one; the appendix explains why fractions cannot improve on it.

Keep the assignment LP and its dual together

Primal: assignment fractions

$$\begin{aligned} \min \quad & \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \\ \text{s.t.} \quad & \sum_{j} x_{ij} = 1 \quad (\forall i), \\ & \sum_{i} x_{ij} = 1 \quad (\forall j), \\ & x_{ij} \geq 0. \end{aligned}$$

Dual: one variable per equality

$$\begin{aligned} \max \quad & \sum_{i} u_i + \sum_{j} v_j \\ \text{s.t.} \quad & u_i + v_j \leq c_{ij} \quad (\forall i, j), \\ & u_i,\ v_j \text{ unrestricted}. \end{aligned}$$

u_i belongs to worker i's row; v_j belongs to job j's column. The coefficient of x_{ij} in the two equalities is 1, giving $u_i + v_j \leq c_{ij}$.

Choose an initial feasible dual solution

$$\max \sum_i u_i + \sum_j v_j \qquad \text{s.t. } u_i + v_j \le c_{ij} \quad (\text{every } i, j).$$

Original costs: worker i, job j

Worker	Job 1	Job 2	Job 3	Job 4
W1	10	19	8	15
W2	10	18	7	17
W3	13	16	9	14
W4	12	19	8	18

$$u = (8, 7, 9, 8), \quad v = (0, 0, 0, 0).$$

Choose all job dual variables $v_j = 0$ as a convenient starting choice. Then $u_i \le c_{ij}$ for every job j.

For worker 1, all four inequalities must hold:

$$u_1 \le 10, \quad u_1 \le 19, \quad u_1 \le 8, \quad u_1 \le 15.$$

To maximize the sum, choose each u_i as large as allowed: its **row minimum**. Thus $u_1 = 8$.

> These dual values are feasible. We have not established optimality.

Compute the reduced costs from u and v

Use $u = (8, 7, 9, 8)$, $v = (0, 0, 0, 0)$. Define the **reduced cost** of worker–job pair (i, j) by $R_{ij} = c_{ij} - u_i - v_j$.

Original costs

Worker	Job 1	Job 2	Job 3	Job 4
W1	10	19	8	15
W2	10	18	7	17
W3	13	16	9	14
W4	12	19	8	18

Subtract each row's minimum

Worker	Job 1	Job 2	Job 3	Job 4
W1	2	11	**0**	7
W2	3	11	**0**	10
W3	4	7	**0**	5
W4	4	11	**0**	10

$$\text{Worker 1:} \quad (10, 19, 8, 15) - 8 = (2, 11, 0, 7).$$

> $R_{ij} \geq 0$ means $u_i + v_j \leq c_{ij}$. A zero means an **active dual constraint**; the original assignment cost is still c_{ij}.

Why look for an assignment using only zeros?

Each complete assignment uses every worker and every job once. Substitute $c_{ij} = u_i + v_j + R_{ij}$ into its total cost:

$$\sum_{i,j} c_{ij}x_{ij} = \sum_i u_i + \sum_j v_j + \sum_{i,j} R_{ij}x_{ij}.$$

If all selected reduced costs are zero, assignment cost equals the feasible dual value. Weak duality proves both solutions optimal.

Initial reduced costs

Worker	Job 1	Job 2	Job 3	Job 4
W1	2	11	**0**	7
W2	3	11	**0**	10
W3	4	7	**0**	5
W4	4	11	**0**	10

Here $u = (8, 7, 9, 8)$, $v = (0, 0, 0, 0)$, so every assignment costs at least $8 + 7 + 9 + 8 = 32$.

All four zeros are in job 3. Only one worker can receive that job. No complete assignment attains 32, so we must continue.

Improve the dual solution by increasing v

Keep $u = (8, 7, 9, 8)$ fixed. The constraints require $v_j \le c_{ij} - u_i$ for every worker i. Choose the largest permitted value in each column.

For job 1, choose $v_1 = \min\{2, 3, 4, 4\} = 2$. All column minima give $v = (2, 7, 0, 5)$.

Before: subtract worker values

Worker	Job 1	Job 2	Job 3	Job 4
W1	2	11	**0**	7
W2	3	11	**0**	10
W3	4	7	**0**	5
W4	4	11	**0**	10

After: also subtract job values

Worker	Job 1	Job 2	Job 3	Job 4
W1	**0**	4	**0**	2
W2	1	4	**0**	5
W3	2	**0**	**0**	**0**
W4	2	4	**0**	5

For example, $R_{14} = 15 - 8 - 5 = 2$. All reduced costs remain nonnegative, and the dual lower bound increases from 32 to $32 + 2 + 7 + 0 + 5 = 46$.

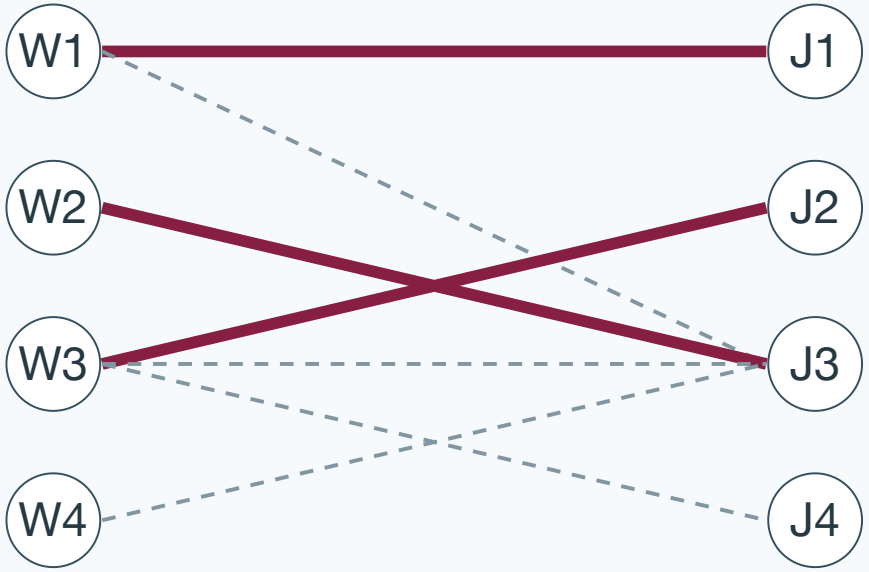
> Costs stay fixed. Next, choose one zero entry in every row and every column.

A matching cannot reuse a worker or a job

A **matching** is a set of pairs with no worker or job repeated. It is **perfect** when all workers and all jobs are matched. A **maximum matching** has the largest possible number of pairs.

In the zero reduced-cost graph, draw a line only when the corresponding dual constraint is active.

> Several workers may want the same zero entry’s column. A zero in each row is therefore not enough: we must find distinct jobs.



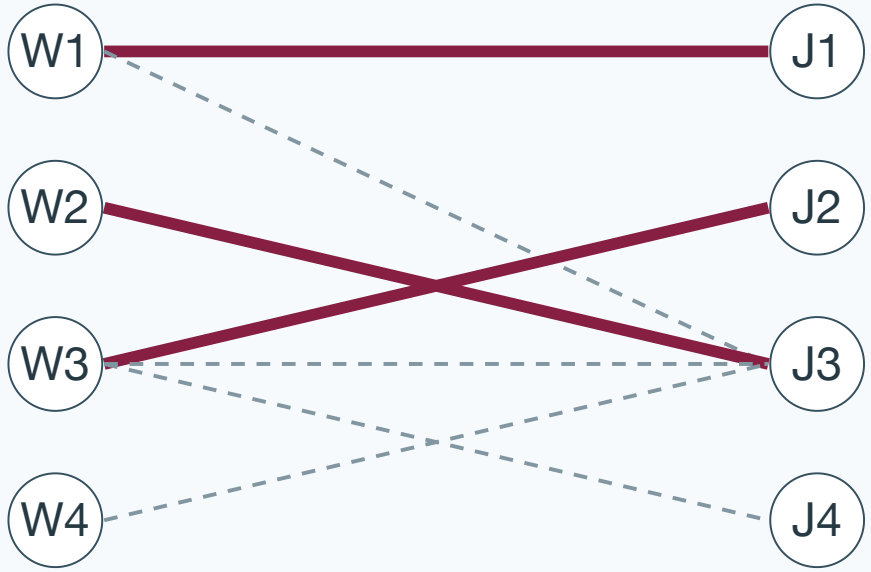
Every line has reduced cost zero. Thick maroon lines are chosen pairs; dashed lines are other available pairs.

Three pairs work, but worker 4 is unmatched

Reduced costs: original cost - worker dual variable - job dual variable

Worker	Job 1	Job 2	Job 3	Job 4
W1	**0 ★**	4	**0**	2
W2	1	4	**0 ★**	5
W3	2	**0 ★**	**0**	**0**
W4	2	4	**0**	5

★ marks W1–J1, W2–J3, W3–J2. These three pairs use different rows and columns.



Every line has reduced cost zero. Thick maroon lines are chosen pairs; dashed lines are other available pairs.

W2 and W4 both have only J3 available at reduced cost zero. They cannot both be matched. Thus no perfect zero matching exists yet.

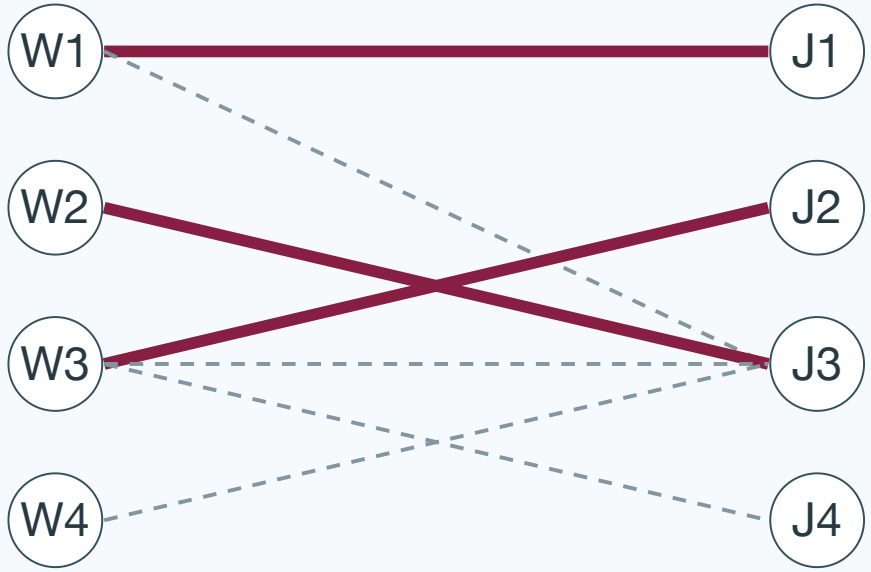
Search for a way to rearrange the chosen pairs

Start at unmatched worker W4. Follow an unused zero edge to a job.

If the job is assigned, follow its chosen pair backward to its worker and continue. An **alternating path** switches between unused and chosen edges.

> **Unmatched job reached:** exchange the pairs along the path; the matching grows by one.
>
> **No unmatched job reachable:** after exploring all reachable zero edges, adjust the dual variables.

Here W4 → J3 → W2 stops: W2 has no alternative zero edge. Reassigning now would leave W2 unmatched.



Every line has reduced cost zero. Thick maroon lines are chosen pairs; dashed lines are other available pairs.

Reached workers and covered rows are different

Reached by W4 → J3 → W2: workers W2, W4 and job J3.

Cover means draw a line through an entire row or column of the reduced-cost table.

Cover the **other workers' rows: W1, W3**, and the **reached job's column: J3**. W2 and W4 have zeros only in J3, so these lines cover every zero.

> The uncovered entries connect reached workers to unreached jobs. Decrease those entries to create another zero, while preserving the chosen pairs and keeping all reduced costs nonnegative.

Shaded: covered worker rows W1, W3 and job column J3

Worker	Job 1	Job 2	Job 3	Job 4
W1	0	4	0	2
W2	1	4	0	5
W3	2	0	0	0
W4	2	4	0	5

Create a new zero without breaking dual feasibility

Reduced cost: $R_{ij} = c_{ij} - u_i - v_j$. Original costs stay fixed.

Increase u_2, u_4 by θ to decrease reduced costs in the reached workers' rows. Decrease v_3 equally to preserve their zeros in reached column J3.

Uncovered entries	J1	J2	J4
W2	1	4	5
W4	2	4	5

$$R'_{23} = 7 - (7 + \theta) - (0 - \theta) = 0.$$

$$\theta = \min\{1, 4, 5, 2, 4, 5\} = 1.$$

The chosen pair W2–J3 stays zero.

$$R'_{21} = 10 - (7 + \theta) - 2 = 1 - \theta.$$

> These entries decrease by 1. All other entries stay unchanged or increase, so every reduced cost remains nonnegative.

At $\theta = 1$, W2–J1 becomes a new zero.

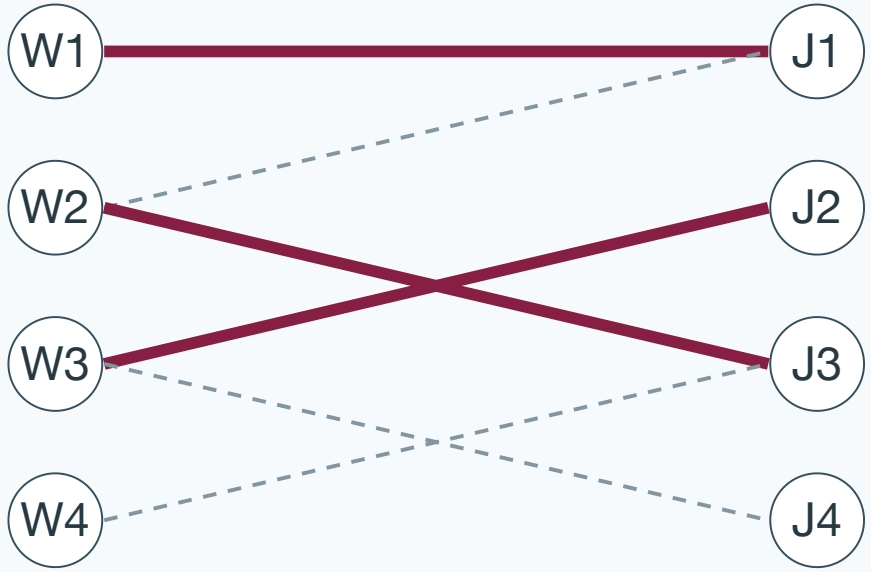
After adjustment 1: still three pairs

Current reduced costs

Worker	Job 1	Job 2	Job 3	Job 4
W1	**0 ★**	4	1	2
W2	**0**	3	**0 ★**	4
W3	2	**0 ★**	1	**0**
W4	1	3	**0**	4

$$u = (8, 8, 9, 9), \quad v = (2, 7, -1, 5),$$
$$\sum_i u_i + \sum_j v_j = 47.$$

New zero: W2–J1. Search: W4 → J3 → W2 → J1 → W1. W1 has no alternative zero edge.



Every line has reduced cost zero. Thick maroon lines are chosen pairs; dashed lines are other available pairs.

No unmatched job is reachable. Adjust the dual variables again.

Your turn: find the second adjustment

Current reduced costs

Worker	Job 1	Job 2	Job 3	Job 4
W1	**0**	4	1	2
W2	**0**	3	**0**	4
W3	2	**0**	1	**0**
W4	1	3	**0**	4

Current $u = (8, 8, 9, 9)$, $v = (2, 7, -1, 5)$.

Work for about 5 minutes.

1. The reached workers are W1, W2, W4; reached jobs are J1, J3. Which other workers' rows and reached jobs' columns cover all zeros?
2. Find the smallest uncovered entry.
3. Update u and v. Which new zero appears?

Adjustment 2 creates the missing route to job 4

Reached workers: W1, W2, W4. Reached jobs: J1, J3. Cover the other worker's row W3 and reached columns J1, J3.

Uncovered row	J2	J4
W1	4	2
W2	3	4
W4	3	4

The smallest uncovered entry is $\theta = 2$. Increase u_1, u_2, u_4 by 2; decrease v_1, v_3 by 2.

$$u = (10, 10, 9, 11), \qquad v = (0, 7, -3, 5).$$

> W1–J4 becomes zero. The dual value increases from 47 to 49. Negative dual variables are allowed because the primal assignment constraints are equalities.

Use the new alternating path to assign worker 4

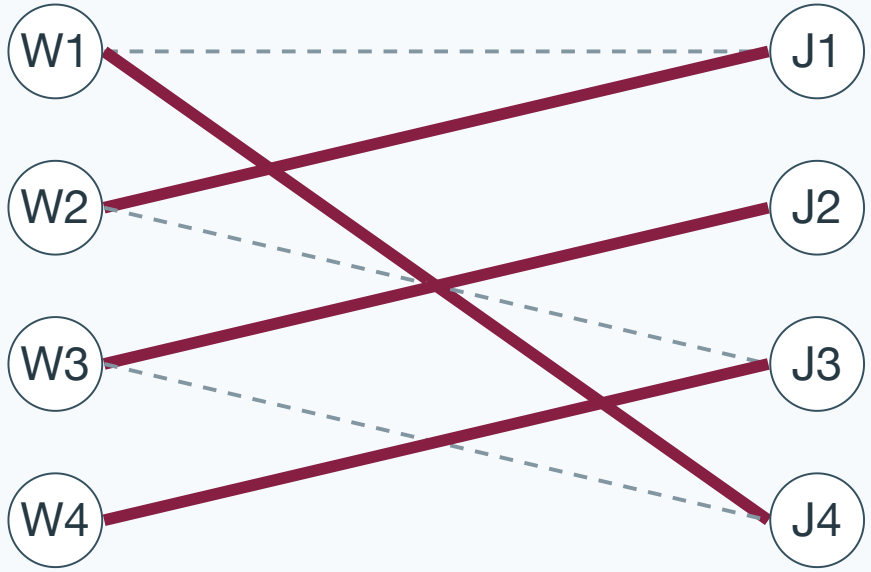
Final reduced costs

Worker	Job 1	Job 2	Job 3	Job 4
W1	**0**	2	1	**0**
W2	**0**	1	**0**	2
W3	4	**0**	3	**0**
W4	1	1	**0**	2

$$W4 \to J3 \to W2 \to J1 \to W1 \to J4.$$

Before reassignment, J4 is unmatched. Stop searching and exchange the pairs.

Remove W2–J3 and W1–J1. Add W4–J3, W2–J1, W1–J4. Keep W3–J2.



Every line has reduced cost zero. Thick maroon lines are chosen pairs; dashed lines are other available pairs.

After: every worker has a different job at zero reduced cost. The assignment is optimal.

Original costs and the dual agree: 49

Original costs; ★ marks the final assignment

Worker	Job 1	Job 2	Job 3	Job 4
W1	10	19	8	15 ★
W2	10 ★	18	7	17
W3	13	16 ★	9	14
W4	12	19	8 ★	18

Primal assignment cost

$$15 + 10 + 16 + 8 = 49.$$

Dual lower bound

$$(10 + 10 + 9 + 11) + (0 + 7 - 3 + 5) = 49.$$

Every reduced cost is nonnegative, so the dual is feasible. Equal values prove optimality.

The Hungarian adjustment in general

Maintain $R_{ij} = c_{ij} - u_i - v_j \geq 0$ and a matching on zero entries. Run alternating searches from unmatched workers.

> If an unmatched job is reached, exchange the pairs along the path to grow the matching. Otherwise, explore all reachable zero edges before adjusting.

Let S be the reached workers and T the reached jobs. Cover worker rows **outside** S and job columns **in** T.

$$\theta = \min_{i \in S,\ j \notin T} R_{ij} > 0.$$

Increase u_i by θ for $i \in S$; decrease v_j by θ for $j \in T$. Leave the other dual variables unchanged.

Continue searching. Stop the algorithm when every worker and job is matched on zeros. A complete cost matrix is assumed here.

Why the four kinds of entries change differently

Entry location	Change in u + v	Change in reduced cost
Reached worker; unreached job	$+\theta$	$-\theta$: creates a new zero
Reached worker; reached job	0	0
Unreached worker; unreached job	0	0
Unreached worker; reached job	$-\theta$	$+\theta$

A chosen pair has either both endpoints reached or both unreached, so its reduced cost stays zero. Some other old zeros can become positive; they are not all preserved.

Appendix: feasibility, matching preservation, progress, and optimality.

Practice: one more complete assignment

A new cost matrix

Worker	Job 1	Job 2	Job 3	Job 4
W1	82	83	69	92
W2	77	37	49	92
W3	11	69	5	86
W4	8	9	98	23

Work for about 5 minutes.

Start from $u = (69, 37, 5, 8)$, $v = (0, 0, 0, 15)$.

1. Compute the reduced table.
2. Find a maximum zero matching and a cover.
3. Make one adjustment and give an optimal assignment.

Practice: compute reduced costs and search from W3

Use $u = (69, 37, 5, 8)$, $v = (0, 0, 0, 15)$. Compute $R_{ij} = c_{ij} - u_i - v_j$.

Original costs

Worker	Job 1	Job 2	Job 3	Job 4
W1	82	83	69	92
W2	77	37	49	92
W3	11	69	5	86
W4	8	9	98	23

Reduced costs; ★ marks chosen pairs

Worker	Job 1	Job 2	Job 3	Job 4
W1	13	14	**0 ★**	8
W2	40	**0 ★**	12	40
W3	6	64	**0**	66
W4	**0**	1	90	**0 ★**

$$R_{14} = 92 - 69 - 15 = 8.$$

$$R_{31} = 11 - 5 - 0 = 6.$$

Choose W1–J3, W2–J2, W4–J4. **W3 and J1 are unmatched.**

Search W3 → J3 → W1. W1 has no alternative zero edge, so no unmatched job is reachable. Adjust the dual variables.

Practice: cover the zeros and choose θ = 6

Shaded: covered rows W2, W4 and column J3

Worker	Job 1	Job 2	Job 3	Job 4
W1	13	14	0 ★	8
W2	40	0 ★	12	40
W3	6	64	0	66
W4	0	1	90	0 ★

Reached: workers W1, W3 and job J3.

Cover the other workers' rows W2, W4 and the reached job's column J3. Uncovered entries connect W1 or W3 to J1, J2 or J4.

Uncovered row	J1	J2	J4
W1	13	14	8
W3	6	64	66

$$\theta = \min\{13, 14, 8, 6, 64, 66\} = 6.$$

Increase the reached workers' dual variables and decrease the reached job's dual variable:

$$\begin{aligned} u_1' &= 69 + 6 = 75, \\ u_3' &= 5 + 6 = 11, \\ v_3' &= 0 - 6 = -6. \end{aligned}$$

All other dual variables stay unchanged.

Practice answer: complete the assignment and verify 140

Updated reduced costs; ★ chosen pairs

Worker	Job 1	Job 2	Job 3	Job 4
W1	7	8	**0 ★**	2
W2	40	**0 ★**	18	40
W3	**0 ★**	58	**0**	60
W4	**0**	1	96	**0 ★**

$$u = (75, 37, 11, 8), \quad v = (0, 0, -6, 15).$$

New zero: $R'_{31} = 11 - 11 - 0 = 0$.

W3 → J1 reaches an unmatched job. Add this pair and keep the other three. The assignment is complete.

All reduced costs are nonnegative, so the dual remains feasible.

Original assignment cost equals the feasible dual value, proving optimality:

$$69 + 37 + 11 + 23 = 140,$$
$$(75 + 37 + 11 + 8) + (0 + 0 - 6 + 15) = 140.$$

Appendix: the updated table and path in one place.

From assignments to minimum-cost flow

Hungarian method: what to retain	Reason
Maintain a feasible dual	All reduced costs stay nonnegative.
Search and adjust	Rearrange zero pairs along an alternating path; if the search fails, adjust the dual to create a new zero.
Finish with a complete zero matching	Its original cost equals the feasible dual value, proving optimality.

Next: network simplex. An assignment sends one unit from each worker to one job. A general minimum-cost flow problem allows other supply and demand quantities and intermediate locations.

We will use a spanning tree to represent a simplex basis and carry out pivots on the network.

Appendix: proofs, extra practice, and optional auction bidding.

Appendix: proofs and additional worked details

Why Hungarian adjustments are valid

Why the procedure reaches a perfect matching

Why fractions do not lower the optimum

Practice: the complete adjusted table

Optional: auction bidding, complete example and guarantees

Unequal counts and maximization

Dual feasibility and chosen pairs survive an adjustment

After a failed alternating search, there is no zero edge from S to a job outside T. Every entry used in the minimum θ is therefore strictly positive.

Those entries decrease by θ and remain nonnegative. Entries from workers outside S to jobs in T increase. The other two entry types are unchanged. Hence the updated dual is feasible.

If a matched job is reached, its matched worker is reached by the backward step. A reached matched worker was reached through its own matched job. Thus a chosen pair has both endpoints reached or neither.

> Every matched reduced cost stays zero. The smallest uncovered entry becomes zero, extending the possible alternating search.

Why a perfect matching is eventually reached

For a fixed matching, start alternating searches from all unmatched workers. If no unmatched job is reached, each reached job is paired with one reached worker; S also contains the unmatched roots. Thus $|S| > |T|$.

An adjustment leaves every existing alternating-search edge zero and adds at least one edge from S to outside T. Continue the same search. It reaches a new job or an unmatched job.

With finitely many jobs, an augmenting path is eventually found. Exchanging its edges grows the matching by one while keeping every chosen reduced cost zero.

> After at most n augmentations, a perfect zero matching and a feasible dual have equal values. Efficient implementations maintain these searches and minima to achieve $O(n^3)$ time.

Why the zero cover is minimum

Let a zero matching have maximum size k. Search from all unmatched workers. No unmatched job is reachable, or an alternating path would enlarge the matching.

The unreached worker rows together with reached job columns cover every zero. Otherwise a zero edge from a reached worker would lead to an unreached job.

$$|\text{cover}| = n - |S| + |T| = k.$$

Each line can cover at most one edge of any matching because matching edges have distinct endpoints. Therefore any zero cover has at least k lines, and this cover is minimum.

> This proves the matching–cover equality for this bipartite graph and explains how to construct the cover.

Why an optimal whole assignment exists

Let X be any feasible fractional assignment. In its graph of positive entries, take a set S of workers and their neighboring jobs N(S).

$$|S| = \sum_{i \in S, j} x_{ij} \leq \sum_{j \in N(S), i} x_{ij} = |N(S)|.$$

A maximum matching missing a worker would have a failed alternating search with |S| > |N(S)|, contradicting this inequality. Hence the positive entries contain a perfect matching. Let P be its assignment matrix: 1 for each chosen pair and 0 otherwise.

Let α be the smallest selected entry. Subtract αP. If α < 1, divide the remainder by 1 - α to obtain another feasible fractional assignment, with fewer positive entries.

> Repeating gives X as a convex combination of whole assignments. At least one has cost no greater than X. Thus fractions cannot improve the minimum.

Practice: all updated reduced costs

After θ = 6

Worker	Job 1	Job 2	Job 3	Job 4
W1	7	8	**0**	2
W2	40	**0**	18	40
W3	**0**	58	**0**	60
W4	**0**	1	96	**0**

$$u = (75, 37, 11, 8),\ v = (0, 0, -6, 15).$$

Before adjustment, choose W1–J3, W2–J2, W4–J4; W3 is unmatched. The alternating search reaches W3, J3, W1.

The new W3–J1 zero directly reaches an unmatched job. Add it; the other three chosen pairs stay.

> Cost = 69 + 37 + 11 + 23 = 140. Dual sum = 131 + 9 = 140.

Optional auction example: two workers want J1

A new assignment problem: two technicians, W1 and W2, must do two jobs, J1 and J2. Each worker gets one job; each job gets one worker.

The table gives the benefit a_{ij} of assigning worker i to job j, measured in points. **Maximize the total benefit:** larger numbers are better.

Original benefit a_{ij}	J1	J2
W1	8	6
W2	7	4

> **Both workers prefer J1.** Sending W1 to J2 loses 8 – 6 = 2 points; sending W2 to J2 loses 7 – 4 = 3 points.

The **auction algorithm** resolves competition by raising an artificial price for a contested job. Repeated comparisons can lead an unassigned worker to choose another job.

Bid 1: define the job prices and calculate 2.25

An **artificial job price** q_j is subtracted from benefit when comparing jobs: $a_{ij} - q_j$. Raising q_j makes job j less attractive. Prices are algorithm numbers, not payments; original benefits stay fixed.

Original benefit a_{ij}	J1	J2
W1	8	6
W2	7	4

Both workers start unassigned; $q = (q_1, q_2) = (0, 0)$. Choose a positive extra increment $\varepsilon = 0.25$; it is an algorithm setting.

W1 bids first. Its comparisons are J1: 8 – 0 = 8 and J2: 6 – 0 = 6. Choose J1.

Raising its price by 8 – 6 = 2 would give equal comparisons. Add 0.25 so the price strictly increases even when comparisons tie.

$$q_1^{\text{new}} = 0 + (8 - 6) + 0.25 = 2.25.$$

After: W1 gets J1; W2 is unassigned. W1 compares $8 - 2.25 = 5.75$ for J1 and $6 - 0 = 6$ for J2. The method permits its assigned job to be worse by $\varepsilon = 0.25$.

Bid 2: W2 takes J1 and W1 becomes unassigned

Before this bid: W1 has J1; W2 is unassigned. Prices are $q = (q_1, q_2) = (2.25, 0)$. Only an unassigned worker bids.

Original benefit a_{ij}	J1	J2
W1	8	6
W2	7	4

W2 compares	Calculation
J1	$7 - 2.25 = 4.75$
J2	$4 - 0 = 4$

J1 is best before the price update. Increase its current price by the comparison difference plus ε = 0.25.

$$q_1^{\text{new}} = 2.25 + (4.75 - 4) + 0.25 = 3.25.$$

After: W2 gets J1; W1 becomes unassigned. The prices are now (3.25, 0). W1 must compare the jobs again using these new prices.

Bid 3: W1 takes J2 and the assignment is complete

Before this bid: W2 has J1; W1 is unassigned. Prices are $q = (q_1, q_2) = (3.25, 0)$.

Original benefit a_{ij}	J1	J2
W1	8	6
W2	7	4

W1 compares	Calculation
J1	$8 - 3.25 = 4.75$
J2	$6 - 0 = 6$

J2 is now best. Raise its price by the comparison difference plus ε = 0.25.

$$q_2^{\text{new}} = 0 + (6 - 4.75) + 0.25 = 1.50.$$

After: W1–J2 and W2–J1. Everyone is assigned, so stop.

Original total benefit: 6 + 7 = 13. The other complete assignment gives 8 + 4 = 12. The final prices (3.25, 1.50) are not subtracted from this objective.

The bidding rule and what the final assignment guarantees

Repeat until everyone is assigned	Operation
Choose	An unassigned worker compares benefit minus current price for every job.
Raise	Choose a best job. Raise its price by (best comparison – second-best comparison) + ε.
Assign	Give the job to this worker. Its previous worker, if any, becomes unassigned.

Choose $\varepsilon > 0$. Each assigned worker's comparison value is no more than ε below its largest benefit-minus-price comparison. A smaller tolerance can require more bids.

> With n workers and n jobs, the final total benefit is at most $n\varepsilon$ below the maximum. Here $n\varepsilon = 2(0.25) = 0.5 < 1$. Since complete assignments have integer benefits, the final assignment is exactly optimal.

Assume every pair is allowed and benefits are finite; the rule uses at least two jobs. Appendix: the formal condition, optimality bound and why it is preserved.

Auction: derive the nε guarantee

Let $\sigma(i)$ be worker i's assigned job and $U_i = \max_j(a_{ij} - q_j)$ its best comparison value. The formal ε-**complementary slackness** condition is

$$a_{i,\sigma(i)} - q_{\sigma(i)} \geq U_i - \varepsilon.$$

Because $U_i + q_j \geq a_{ij}$ for every pair, $D = \sum_i U_i + \sum_j q_j$ is a dual upper bound on OPT, the maximum total benefit. Summing the condition for a complete assignment gives

$$\sum_i a_{i,\sigma(i)} \geq D - n\varepsilon \geq \mathrm{OPT} - n\varepsilon.$$

Why the condition is preserved: a bid leaves the chosen comparison at $W - \varepsilon$, where W was the best comparison among the other jobs. It is within ε of the best alternative.

For a worker who stays assigned, its own job price stays fixed and other price increases can only lower alternative comparisons. A worker whose job price changes is displaced and bids again.

Auction: why bidding stops for a complete finite matrix

Each bid raises a price by at least ε. Once a job is assigned, it stays assigned, though its owner may change. A job that has never been assigned still has its initial price 0.

Suppose bidding continued forever. The number of assigned jobs can increase at most n times. Eventually it would remain fixed below n, with some unassigned job at price 0.

Let A be the largest absolute benefit. Every later bid is for an already assigned job. Its second-best comparison value W is at least -A, because an unassigned price-0 job remains available.

The newly assigned job price is a_{ij} - W + ε ≤ 2A + ε. Finitely many prices cannot increase by at least ε infinitely often while staying bounded.

> This contradiction proves termination. Assumptions: finite benefits, all pairs allowed, and ε > 0.

Unequal counts, prohibited pairs, and maximum benefit

Variation	Modeling change
More jobs than workers	Add dummy workers with explicitly specified costs
More workers than jobs	Add dummy jobs representing non-assignment
A prohibited worker–job pair	Remove that variable and its graph edge
Maximize total benefit a_{ij}	Minimize c_{ij} = M - a_{ij}, for a common M

A complete assignment always uses n entries, so its transformed cost equals nM minus its original benefit.

> For prohibited pairs, a perfect matching may not exist. A failed search with no uncovered permitted entry proves infeasibility; do not take a minimum over an empty set.

Additional practice: a zero matching after reductions

New original costs

Worker	Job 1	Job 2	Job 3	Job 4
W1	9	2	7	8
W2	6	4	3	7
W3	5	8	1	8
W4	7	6	9	4

Use $u = (2, 3, 1, 4)$, $v = (3, 0, 0, 0)$.

The zero matching is W1–J2, W2–J1, W3–J3, W4–J4.

$$2 + 6 + 1 + 4 = 13 = \sum_i u_i + \sum_j v_j.$$

No adjustment is needed: a feasible dual and a complete zero matching already certify the answer.