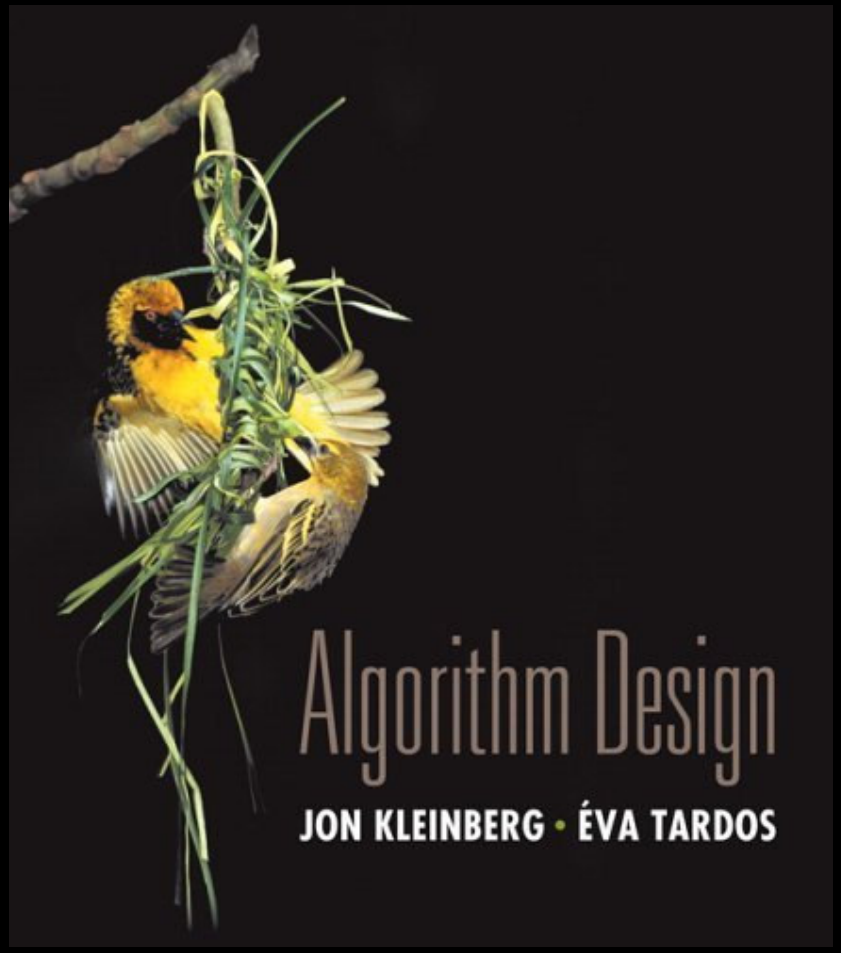


Chapter 7

Network Flow

* 7.13 Assignment Problem

Assignment Problem

Assignment problem.

- Input: weighted, complete bipartite graph $G = (L \cup R, E)$ with $|L| = |R|$.
- Goal: find a perfect matching of min weight.

	1'	2'	3'	4'	5'
1	3	8	9	15	10
2	4	10	7	16	14
3	9	13	11	19	10
4	8	13	12	20	13
5	1	7	5	11	9

Min cost perfect matching

M = { 1-2', 2-3', 3-5', 4-1', 5-4' }

cost(M) = 8 + 7 + 10 + 8 + 11 = 44

Applications

Natural applications.

- Match jobs to machines.
- Match personnel to tasks.
- Match PU students to writing seminars.

Non-obvious applications.

- Vehicle routing.
- Signal processing.
- Virtual output queueing.
- Multiple object tracking.
- Approximate string matching.
- Enhance accuracy of solving linear systems of equations.

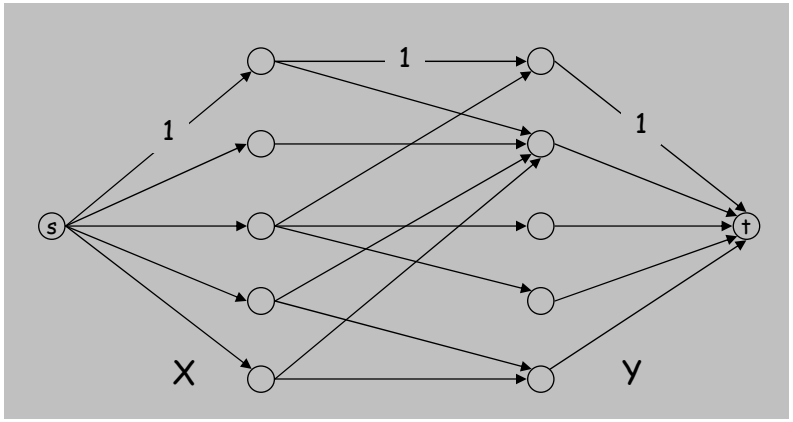
Bipartite Matching

Bipartite matching. Can solve via reduction to max flow.

Flow. During Ford-Fulkerson, all capacities and flows are 0/1. Flow corresponds to edges in a matching M.

Residual graph G_M simplifies to:

- If $(x, y) \notin M$, then (x, y) is in G_M.
- If $(x, y) \in M$, the (y, x) is in G_M.
- If x is unmatched, (s,y) is in G_M
- If y is unmatched, (y,t) is in G_M

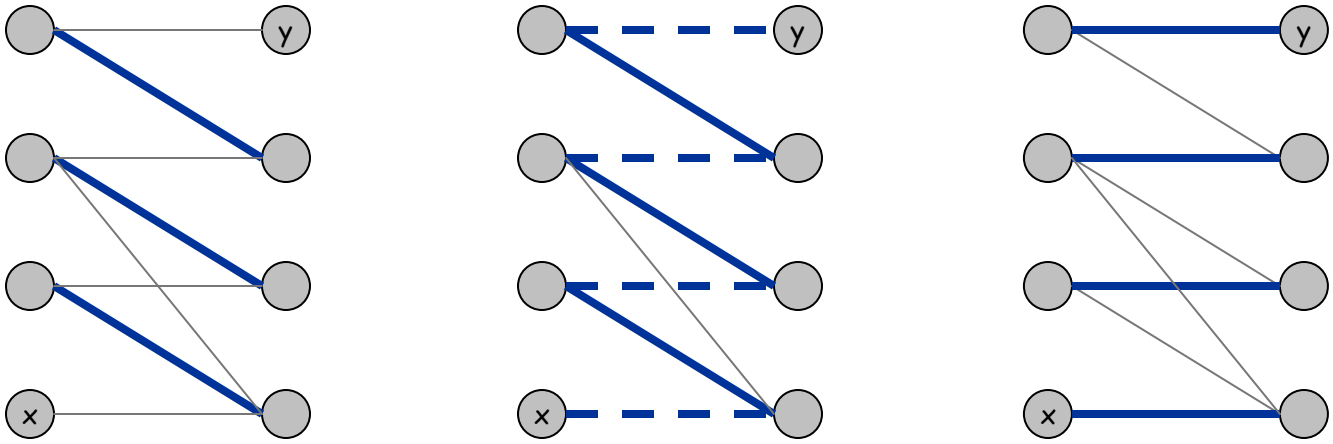


Augmenting path simplifies to:

- Edge from s to an unmatched node $x \in X$.
- Alternating sequence of unmatched and matched edges.
- Edge from unmatched node $y \in Y$ to t.

Alternating Path

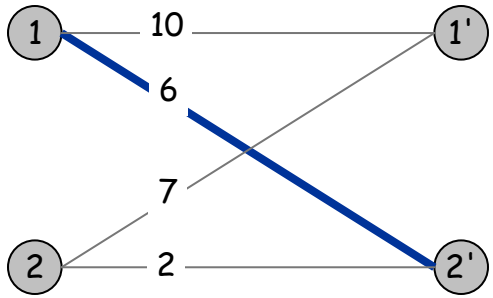
Alternating path. Alternating sequence of unmatched and matched edges, from unmatched node $x \in X$ to unmatched node $y \in Y$.



matching M | alternating path | matching M'

Assignment Problem: Successive Shortest Path Algorithm

Cost of an alternating path. Pay c(x, y) to match x-y; receive c(x, y) to unmatch x-y.



cost(2 - 1') = 7
cost(2 - 2'- 1 - 1') = 2 - 6 + 10 = 6

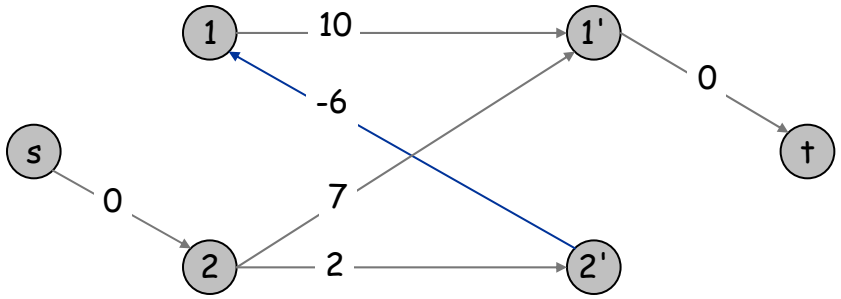
Shortest alternating path. Alternating path from any unmatched node $x \in X$ to any unmatched node $y \in Y$ with smallest cost.

Successive shortest path algorithm.

- Start with empty matching.
- Repeatedly augment along a shortest alternating path.

Finding The Shortest Alternating Path

Shortest alternating path. Corresponds to shortest s-t path in G_M.



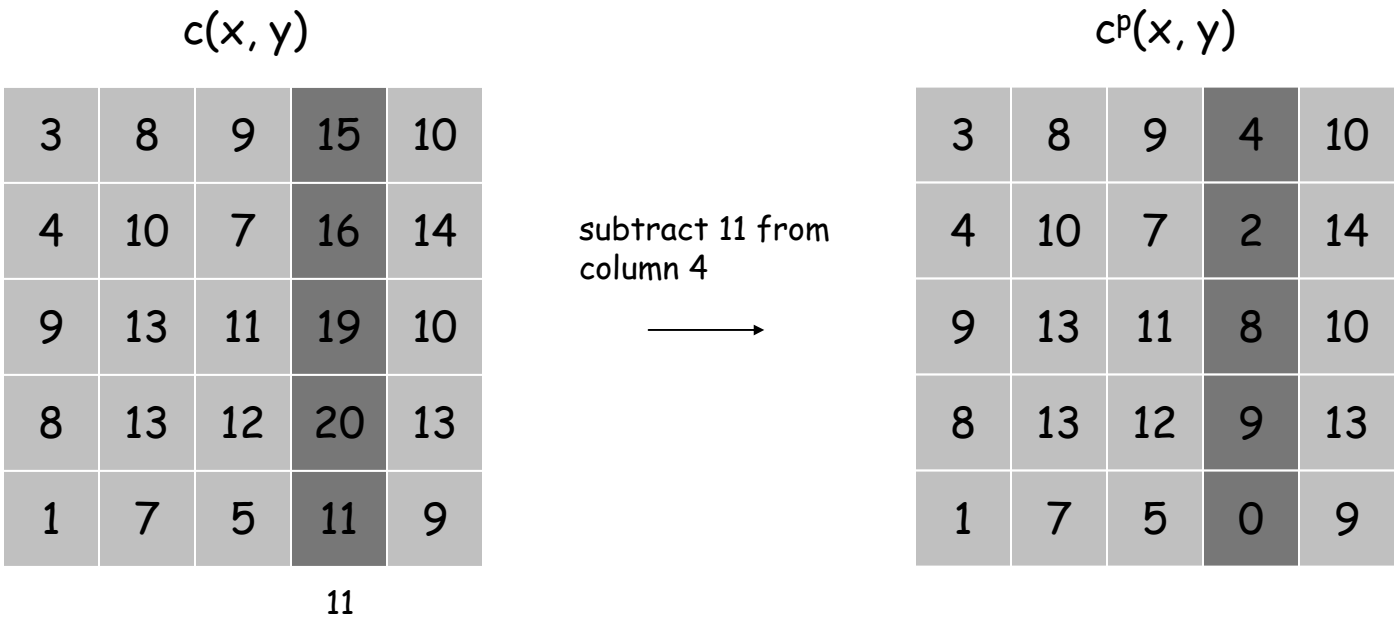
Concern. Edge costs can be negative.

Fact. If always choose shortest alternating path, then G_M contains no negative cycles. The reason will be clear at the end $\Rightarrow$ compute using Bellman-Ford.

Our plan. Use duality to avoid negative edge costs (and negative cost cycles) $\Rightarrow$ compute using Dijkstra.

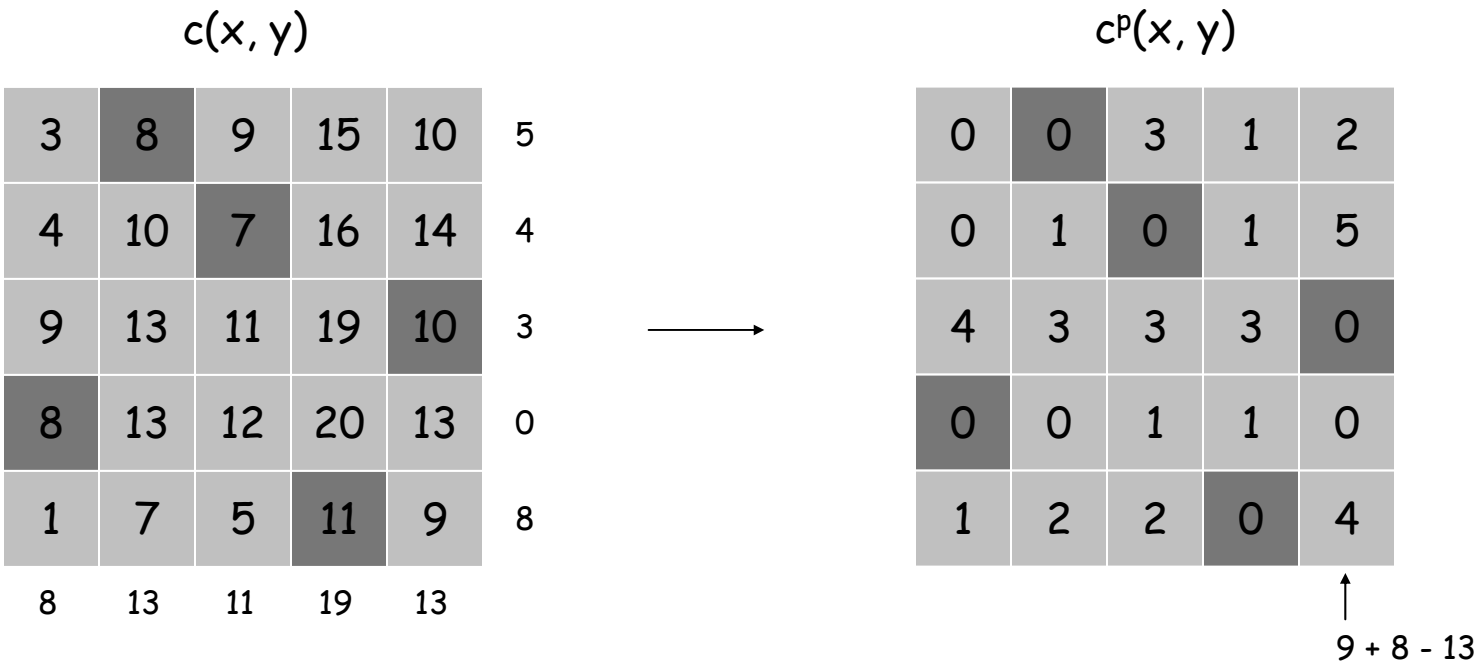
Equivalent Assignment Problem

Duality intuition. Adding (or subtracting) a constant to every entry in row x or column y does not change the min cost perfect matching(s).



Equivalent Assignment Problem

Duality intuition. Adding p(x) to row x and subtracting p(y) from column y does not change the min cost perfect matching(s).



Reduced Costs

Reduced costs. For $x \in X$, $y \in Y$, define $c^p(x, y) = p(x) + c(x, y) - p(y)$.

Observation 1. Finding a min cost perfect matching with reduced costs is equivalent to finding a min cost perfect matching with original costs.

$c(x, y)$

3	8	9	15	10	5
4	10	7	16	14	4
9	13	11	19	10	3
8	13	12	20	13	0
1	7	5	11	9	8
8	13	11	19	13	

$c^p(x, y)$

0	0	3	1	2
0	1	0	1	5
4	3	3	3	0
0	0	1	1	0
1	2	2	0	4

9 + 8 - 13

Compatible Prices

Compatible prices. For each node v, maintain prices p(v) such that:

- (i) $c^p(x, y) \geq 0$ for for all $(x, y) \notin M$.
- (ii) $c^p(x, y) = 0$ for for all $(x, y) \in M$.

Observation 2. If p are compatible prices for a perfect matching M, then M is a min cost perfect matching.

$c(x, y)$

3	8	9	15	10	5
4	10	7	16	14	4
9	13	11	19	10	3
8	13	12	20	13	0
1	7	5	11	9	8
8	13	11	19	13	

$\longrightarrow$

$c^p(x, y)$

0	0	3	1	2
0	1	0	1	5
4	3	3	3	0
0	0	1	1	0
1	2	2	0	4

$\text{cost}(M) = \Sigma_{(x, y) \in M} c(x, y) = (8+7+10+8+11) = 44$

$\text{cost}(M) = \Sigma_{y \in Y} p(y) - \Sigma_{x \in X} p(x) = (8+13+11+19+13) - (5+4+3+0+8) = 44$

Successive Shortest Path Algorithm

Successive shortest path.

```
Successive-Shortest-Path(X, Y, c) {
   M ← ϕ
   foreach x ∈ X:  p(x) ← 0
   foreach y ∈ Y:  p(y) ← min_{e into y} c(e)

   while (M is not a perfect matching) {
      Compute shortest path distances d
      P ← shortest alternating path using costs c^p
      M ← updated matching after augmenting along P
      foreach v ∈ X ∪ Y:  p(v) ← p(v) + d(v)
   }
   return M
}
```

p is compatible with M = ϕ

Maintaining Compatible Prices

Lemma 1. Let M' be matching obtained by augmenting along a min cost path with respect to c^{p+d}. Then p' = p + d is compatible with M'.

Pf.

- Case 1. (x,y) in M. By construction, the only edge that arrives at x is G_M is (y,x). This implies that d(x)=d(y) - (c(x,y)+p(x) –p(y)). Thus,

$$c^{p+d}(x, y) = c(x,y) + p(x) - p(y) + d(x) - d(y) = 0.$$

- Case 2 (x,y) in M' – M. (x,y) belongs to the shortest path in G_M. We must have d(y)=d(x) + (c(x,y)+p(x) –p(y)). Thus,

$$c^{p+d}(x, y) = c(x,y) + p(x) - p(y) + d(x) - d(y) = 0.$$

- Case 3. (x,y) is not in M U M'. In this case, (x,y) in G_M. We have that d(y)<=d(x) + (c(x,y)+p(x) –p(y)). Thus,

$$c^{p+d}(x, y) = c(x,y) + p(x) - p(y) + d(x) - d(y) >= 0.$$

Successive Shortest Path: Analysis

Invariant. The algorithm maintains a matching M and compatible prices p.

Pf. Follows from Lemmas 1 and initial choice of prices. ▪

Theorem. The algorithm returns a min cost perfect matching.

Pf. Upon termination M is a perfect matching, and p are compatible prices. Optimality follows from Observation 2. ▪

Theorem. The algorithm can be implemented in $O(n^3)$ time.

Pf.

- Each iteration increases the cardinality of M by 1 $\Rightarrow$ n iterations.
- Bottleneck operation is computing shortest path distances d. Since all costs are nonnegative, each iteration takes $O(n^2)$ time using (dense) Dijkstra. ▪

Weighted Bipartite Matching

Weighted bipartite matching. Given weighted bipartite graph, find maximum cardinality matching of minimum weight. ← m edges, n nodes

Successive shortest path algorithm. $O(mn \log n)$ time using heap-based version of Dijkstra's algorithm.

Best known bounds. $O(mn^{1/2})$ deterministic; $O(n^{2.376})$ randomized.

Planar weighted bipartite matching. $O(n^{3/2} \log^5 n)$.