LINEAR OPTIMIZATION

LECTURE 18

COMBINATORIAL OPTIMIZATION

An optimization problem where we have to pick the optimum from a finite set of options

Eg:
- shortest path in a graph
- minimum spanning tree
- minimum cost matching
- minimum weight vertex cover

⋮

- A combinatorial algorithm is a procedure that searches through the finitely many solutions to get to the optimum. Typically there are exponentially many possibilities.

  └ The goal is to use LP theory to design efficient combinatorial algorithms

At times, the designed algorithms give exact answers, and in other cases, we get approximate answers.

- We will make use of a primal-dual pair to arrive at the optimum. So these algorithms will be called primal-dual algorithms.

- We next give a generic template to design a primal-dual algorithm.

PRIMAL - DUAL ALGORITHMS : a generic template

Optimization problem

↓

Frame it as an ILP

↓

Generate an LP from the ILP, and write its dual

We now have a primal - dual pair. Let us say they are in the following form (this is not necessary, we choose this for illustration)

Primal

$$\min \; c^T x$$

$$\text{subj. to} \quad Ax \leq b, \quad x \geq 0$$

Dual

$$\max \; b^T y$$

$$\text{subj. to} \quad A^T y \leq c, \quad y \geq 0$$

↳ How do we find the optimum combinatorially?

PRIMAL - DUAL ALGORITHMS : a generic template (contd.)

Initialization: Find a feasible soln. y_0 of the dual. Typically for the problems under consideration, $c \geq 0$ and hence $y = 0$ will be feasible.

Iterative step: After the i^{th} step, say we have a feasible soln. y_i.

Let $(A^T)'$ denote the rows of dual that are tight at y_i:

$$(A^T)' y_i = c'$$

c' is c restricted to rows of $(A^T)'$.

- If possible, find a $\bar{y}$ s.t.

adding any multiple of $\bar{y}$ to y keeps $(A^T)'(y + \epsilon\bar{y}) \leq 0$ →

$$(A^T)' \bar{y} \leq 0 \quad \text{and} \quad b^T \bar{y} > 0$$

cost at $\bar{y} > 0$

- and find an ϵ so that:

$$A^T (y_i + \epsilon \bar{y}) \leq c$$

$(A^{T'})(y_i + \epsilon\bar{y})$
$= (A^T)' y_i + \epsilon (A^T)' \bar{y}$
$= c' + \leq 0$

Set: $y_{i+1} = y_i + \epsilon \bar{y}$

Termination: When the iterative step cannot be performed anymore.

Optimality: From the final y that is obtained at the end of the algorithm, generate a primal solution x and use either complementary slackness or strong duality to show that x and y are optima of primal & dual resp.

Optimal soln. for ILP?

The primal-dual algorithm terminates with an optimal solution for the LP.

- Typically we obtain an integral soln. to the primal.
- We then need to analyze whether this coincides with the optimum of the initial ILP or it gives some approximation.

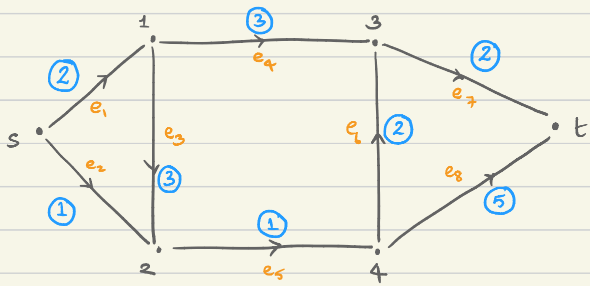
Primal-dual algorithm for the shortest paths problem:

Problem: Given a directed graph $G = (V, E)$ with non-negative weights on edges $c : E \rightarrow c_e$, and 2 vertices 's' and 't'.

Find the value of the $\underbrace{\text{shortest path}}_{\text{path with minimum weight}}$ from s to t.

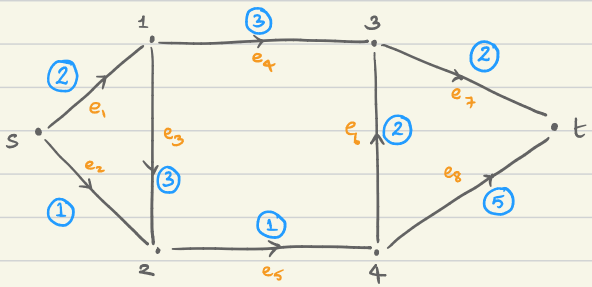
Assume s has only outgoing edges and t has only incoming edges.

Example:



Shortest path: $e_2 \; e_5 \; e_6 \; e_7$ Value: 6

Part 1: Representing the graph



Incidence matrix:

	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8
s	+1	+1	0	0	0	0	0	0
1	−1	0	+1	+1	0	0	0	0
2	0	−1	−1	0	+1	0	0	0
3	0	0	0	−1	0	−1	+1	0
4	0	0	0	0	−1	+1	0	+1
t	0	0	0	0	0	0	−1	−1

- Edges are columns
- Vertices are rows
- Cell ij is +1 if $i \xrightarrow{j}$ (Edge j is outgoing from i)

  is −1 if $\xrightarrow{j} i$ (Edge j is incoming to i)

  0 otherwise

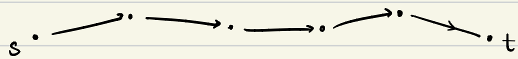
Part 2: Writing the ILP for SP.

A path from s to t is a subset of edges

$$s \xrightarrow{e_{i_1}} \xrightarrow{e_{i_2}} \longrightarrow \cdots \xrightarrow{e_{i_k}} t$$

Variables: Variables of our ILP are of the form x_e for each edge $e \in E$.

Constraints: - Constraints should be designed so that each feasible solution gives a path from s to t.



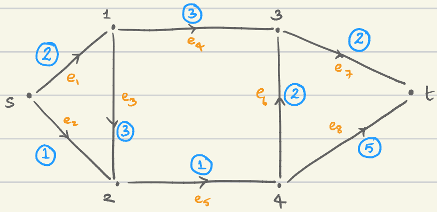
A subset of edges P is a path from s to t iff:

- there is exactly one outgoing edge from s in P
- for every vertex $v \notin \{s,t\}$ in the path, the number of incoming edges to v in P equals no. of outgoing edges from v in P
- there is exactly one incoming edge to t in P

- Assume the given graph has 'n' edges and 'm' vertices. Let A be the incidence matrix.

Above constraints can be represented as:

$$A \begin{bmatrix} x_{e_1} \\ x_{e_2} \\ \vdots \\ x_{e_n} \end{bmatrix} = \begin{array}{c} \begin{bmatrix} +1 \\ 0 \\ 0 \\ 0 \\ \vdots \\ -1 \end{bmatrix} \end{array} \begin{array}{l} \to \text{row corresponding to } s \\ \\ \\ \\ \\ \to \text{row corresponding to } t \end{array}$$



	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8
s	+1	+1	0	0	0	0	0	0
1	-1	0	+1	+1	0	0	0	0
2	0	-1	-1	0	+1	0	0	0
3	0	0	0	-1	0	-1	+1	0
4	0	0	0	0	-1	+1	0	+1
t	0	0	0	0	0	0	-1	-1

A

s: $x_{e_1} + x_{e_2} = 1$

1: $x_{e_4} + x_{e_3} - x_{e_1} = 0$

2: $x_{e_5} - x_{e_2} - x_{e_3} = 0$

3: $x_{e_7} - x_{e_4} - x_{e_6} = 0$

4: $x_{e_6} + x_{e_8} - x_{e_5} = 0$

t: $-x_{e_7} - x_{e_8} = -1$

$$\rightarrow \quad A \begin{bmatrix} x_{e_1} \\ x_{e_2} \\ \vdots \\ x_{e_8} \end{bmatrix} = \begin{bmatrix} +1 \\ 0 \\ 0 \\ \vdots \\ 0 \\ -1 \end{bmatrix}$$

row of s

row of t

Objective: Objective is to minimize cost: minimize $\sum_{e \in E} c_e x_e$

Final ILP:

$$\text{minimize} \quad \sum_{e \in E} c_e x_e$$

$$\text{subject to:} \quad A x = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ -1 \end{bmatrix}$$

$$x_e \in \{0, 1\} \quad \forall e \in E$$

Final ILP:

$$\text{Minimize} \sum_{e \in E} c_e x_e$$

$$\text{subject to:} \quad Ax = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ -1 \end{bmatrix}$$

$$x_e \in \{0,1\} \quad \forall e$$

From ILP to LP:

- We will relax the integrality constraint. This gives:

$$0 \leq x_e \leq 1$$

- Now, we remove the $x_e \leq 1$ constraint as well

It can be proved that for the final LP that we obtain, there is an optimum that assigns either 0 or 1 to each x_e. We will not worry about this now and wait for the final answer of the primal-dual algorithm. At the moment, we have LP:

$$\text{minimize} \sum_{e \in E} c_e x_e$$

$$\text{subject to:} \quad Ax = \begin{bmatrix} +1 \\ 0 \\ 0 \\ \vdots \\ 0 \\ -1 \end{bmatrix}$$

$$x \geq 0$$

Moreover: LP-optimum $\leq$ ILP-optimum.

Part 3: Write the dual

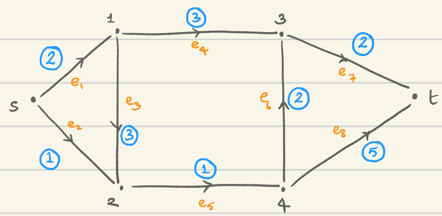
Exercise: Write the dual of the LP:

$$\min \sum_{e \in E} c_e x_e$$

$$\text{subject to} \quad Ax = \begin{bmatrix} +1 \\ 0 \\ \vdots \\ -1 \end{bmatrix}$$

$$x \geq 0$$

obtained from our running example:



		e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8
y_s	s	+1	+1	0	0	0	0	0	0
y_1	1	−1	0	+1	+1	0	0	0	0
y_2	2	0	−1	−1	0	+1	0	0	0
	3	0	0	0	−1	0	−1	+1	0
y_4	4	0	0	0	0	−1	+1	0	+1
y_t	t	0	0	0	0	0	0	−1	−1

$\hookrightarrow A$

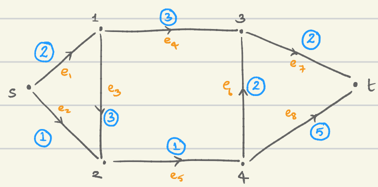
Exercise: Write the dual of the LP:

$$\min \sum_{e \in E} c_e x_e$$

$$\text{subject to} \quad A x = \begin{bmatrix} +1 \\ 0 \\ \vdots \\ -1 \end{bmatrix}$$

$$x \geq 0$$

obtained from our running example:



		e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8
y_s ×	s	+1	+1	0	0	0	0	0	0
y_1 ×	1	−1	0	+1	+1	0	0	0	0
y_2 ×	2	0	−1	−1	0	+1	0	0	0
y_3 ×	3	0	0	0	−1	0	−1	+1	0
y_4 ×	4	0	0	0	0	−1	+1	0	+1
y_t ×	t	0	0	0	0	0	0	−1	−1

$\hookrightarrow A$

$$\begin{bmatrix} x_{e_1} \\ x_{e_2} \\ x_{e_3} \\ \vdots \\ x_{e_8} \end{bmatrix} = \begin{bmatrix} +1 \\ 0 \\ 0 \\ 0 \\ 0 \\ -1 \end{bmatrix}$$

Dual: $\quad \max \quad y_s - y_t$

subj. to:

$$y_s - y_1 \leq 2 \rightarrow (c_{e_1})$$

$$y_s - y_2 \leq 1$$

$$y_1 - y_2 \leq 3$$

$$\vdots$$

$$y_4 - y_t \leq 5$$

Primal:

$$\min \sum_{e \in E} c_e x_e$$

$$Ax = \begin{bmatrix} +1 \\ 0 \\ \vdots \\ 0 \\ -1 \end{bmatrix}$$

$$x \geq 0$$

Dual:

$$\max \; y_s - y_t$$

$$y_u - y_v \leq c_{uv} \quad \text{for all edges } u \xrightarrow{c_{uv}} v$$

Part 4: The iterative step in the primal-dual algorithm

At the start of the iterative step, there is a feasible soln. y of the dual.

Let $J = \{ e : u \xrightarrow{c_{uv}} v \mid y_u - y_v = c_{uv} \}$

These are the dual constraints that are tight for y.

What we require?

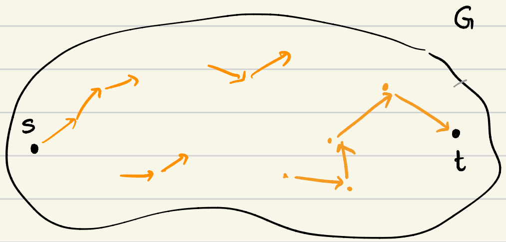
A dual feasible soln. $\bar{y}$ s.t.

1) $\bar{y}_u - \bar{y}_v \leq 0 \quad \forall$ edges $e : u \xrightarrow{c_{uv}} v \in J$

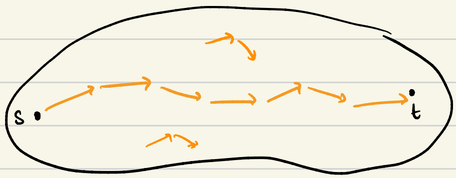
and 2) $\bar{y}_s - \bar{y}_t > 0$

↳ design a combinatorial algorithm to find this

Consider the graph with the edges from J marked in orange.



Case 1: Suppose there is a path from s to t consisting of orange edges (ie., edges from J)



Claim: There is no $\bar{y}$ satisfying the required conditions.

Proof: Suppose the path of J edges from s to t is:

$$s \longrightarrow u_1 \longrightarrow u_2 \longrightarrow \cdots \longrightarrow u_k \longrightarrow t$$

Condition (1) requires

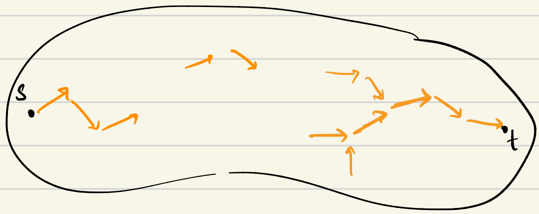
$$y_s - y_{u_1} \leq 0$$

$$y_{u_i} - y_{u_{i+1}} \leq 0 \quad \forall\, 1 \leq i \leq k-1$$

$$y_k - y_t \leq 0$$

$$\Rightarrow \quad y_s - y_t \leq 0.$$ Hence Condition (2) cannot be satisfied.

Case 2: There is no path from s to t via J edges.



Let $V_t = \{ v \in V \mid$ there is a path from v to t via J edges $\}$

Here is a $\bar{y}$ that satisfies (1) and (2)

$$\bar{y}(v) = \begin{cases} 0 & \text{if } v \in V_t \\ 1 & \text{otherwise} \end{cases}$$

condition (1)

$\bar{y}_u - \bar{y}_v \leq 0$ for $uv \in J$

Condition (1) requires $\bar{y}_u - \bar{y}_v \leq 0 \quad \forall\, u \to v \in J$

Now for each $u \to v$ in J (orange edges)

- either $u \in V_t$ and $v \in V_t$ — $\bar{y}_u = 0$, $\bar{y}_v = 0$, $\bar{y}_u - \bar{y}_v \leq 0$
- or $u \notin V_t$ and $v \notin V_t$ — $\bar{y}_u = 1$, $\bar{y}_v = 1$, $\bar{y}_u - \bar{y}_v \leq 0$
- or $u \in V_t$ and $v \notin V_t$ $\quad \bar{y}_u = 0$, $\bar{y}_v = 1$

$\bar{y}_u - \bar{y}_v \leq 0$

Consider an arbitrary edge $e : u \longrightarrow v$

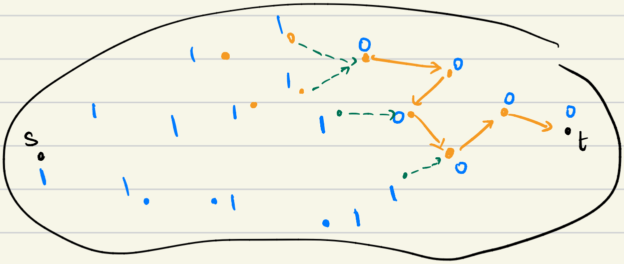
- $(u \in V_t$ and $v \in V_t)$ or $(u \notin V_t$ and $v \notin V_t$

  $\hookrightarrow \bar{y}_u - \bar{y}_v = 0$

- $u \in V_t$ and $v \notin V_t$: $\bar{y}_u - \bar{y}_v = -1 \leq 0$

- $u \notin V_t$ and $v \in V_t$: $\bar{y}_u - \bar{y}_v = 1$

Pictorially, $\bar{y}$ looks like this :



Green dotted edges are "border" edges that lead into a vertex of V_t.

The next task is to find ϵ s.t :

$$y + \epsilon \bar{y} \text{ is feasible for dual}$$

i.e., $$y_u - y_v + \epsilon(\bar{y}_u - \bar{y}_v) \leq c_{uv}$$

This condition is non-trivial only when $u \notin V_t$ and $v \in V_t$.

- Such edges $u \to v$ are not in J, hence $y_u - y_v < c_{uv}$

By construction: $\bar{y}_u - \bar{y}_v = 1$

Pick $$\epsilon = \min_{\substack{u \notin V_t \\ v \in V_t}} \left[c_{uv} - (y_u - y_v) \right]$$

Primal - dual algorithm :

Initialization: Set all y_u to 0

Iterative step: - Consider the graph $G' = (V, J)$ of tight edges.

- In this graph find all vertices that can reach t
  ↳ call this V_t
- Increase the value of $y_v \notin V_t$.
  until some edge becomes tight.
- stop when s can reach t in G'.

Notice that at each iterative step, one new edge becomes tight. Hence the algorithm terminates.

Proof of optimality:

Let y be the final solution. Let J be the tight edges.

- We know there is a path π from s to t via J edges

Consider a primal soln. x defined as:

$$x_e = 1 \quad \text{if } e \text{ is in the path } \pi$$
$$= 0 \quad \text{otherwise}$$

Clearly this soln. is feasible since it is a path and it satisfies all constraints of primal LP.

Moreover:

$$\text{Cost at } x = \sum_{(u,v)\in\pi} c_{uv}$$

$$= \sum_{(u,v)\in\pi} y_u - y_v \quad \text{as} \quad y_u - y_v = c_{uv}$$

$$= y_s - y_t$$

$$= \text{Cost at } y.$$

$\therefore$ We have a primal soln. x and a dual soln. y

st. Cost (primal soln.) = Cost (dual soln.)

By duality theorem, x should be primal optimum and y is dual optimum.

Final argument:

Recall that our primal is the LP relaxation of the original ILP.

$$\text{so} \quad \text{optimum (LP)} \leq \text{opt. (ILP)}$$

However our LP optimum is a feasible soln. of ILP. This shows that LP optimum = ILP optimum.

Hence the soln. we get is indeed the shortest path.

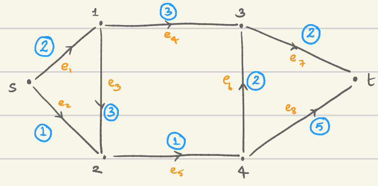
Recall the final primal-dual algorithm:

Initialization: Set all y_u to 0

Iterative Step:
- Consider the graph $G' = (V, J)$ of tight edges.
- In this graph find all vertices that can reach t
- Increase the weight of all the other edges until some edge becomes tight.
- Stop when s can reach t in G'

↳ This mimicks Dijkstra's algorithm.

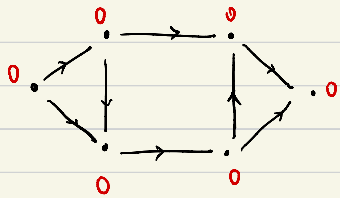
Example:



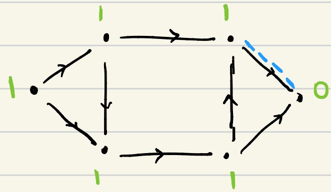
$y_1 - y_3 = 0$

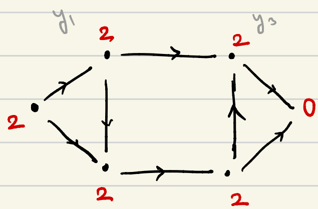
$y_1 - y_3 \leq 3$

$y_4 - y_3 \leq 2$

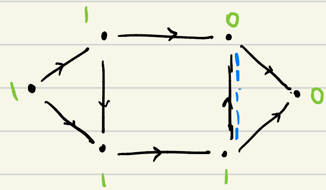


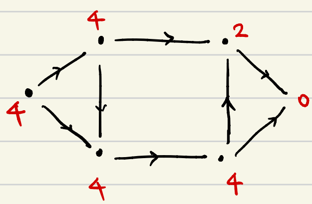
$\xrightarrow{}$ $J = \{\ \}$



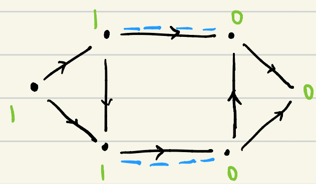


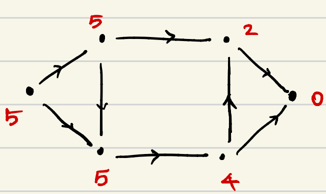
$\xrightarrow{}$ $J = \{e_7\}$



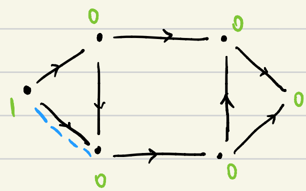


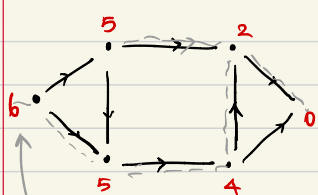
$\xrightarrow{}$ $J = \{e_6, e_7\}$



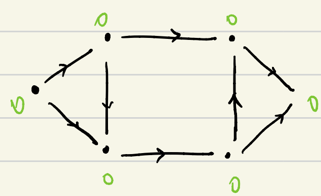


$\xrightarrow{}$ $J = \{e_4, e_5, e_6, e_7\}$





$\xrightarrow{}$ $J = \{e_2, e_4, e_5, e_6, e_7\}$



weight of shortest path