

Primal Dual Approximation Algorithms

Or, Partially Covering topics in Partial Covering

Sambuddha Roy¹

¹IBM Research – India

Outline

- 1 Motivation
- 2 The Primal Dual Schema
- 3 Vertex Cover
- 4 Prize Collecting Vertex Cover
- 5 Partial Vertex Cover
- 6 Lagrangian Relaxations
- 7 Ending Comments

The Basic Paradigm

- Many real-life problems can be modeled as Integer Linear Programs (IP). Since IPs are NP-hard to solve, they are often *relaxed* to a linear program (shortened as LP).
- Modus operandi: solve the linear program in polynomial time, and extract useful information about an integer optimum solution.
- Since the solution to the LP may not be integral, we may need to *massage* the LP fractional solution to an integral one. The hope is that this is a *good* integral solution.
- However, for certain problems, we do not need to even solve the LP to get good (reasonable approximation factor) solutions to our problem.

The Basic Paradigm

- Many real-life problems can be modeled as Integer Linear Programs (IP). Since IPs are NP-hard to solve, they are often *relaxed* to a linear program (shortened as LP).
- Modus operandi: solve the linear program in polynomial time, and extract useful information about an integer optimum solution.
- Since the solution to the LP may not be integral, we may need to *massage* the LP fractional solution to an integral one. The hope is that this is a *good* integral solution.
- However, for certain problems, we do not need to even solve the LP to get good (reasonable approximation factor) solutions to our problem.

The Basic Paradigm

- Many real-life problems can be modeled as Integer Linear Programs (IP). Since IPs are NP-hard to solve, they are often *relaxed* to a linear program (shortened as LP).
- Modus operandi: solve the linear program in polynomial time, and extract useful information about an integer optimum solution.
- Since the solution to the LP may not be integral, we may need to *massage* the LP fractional solution to an integral one. The hope is that this is a *good* integral solution.
- However, for certain problems, we do not need to even solve the LP to get good (reasonable approximation factor) solutions to our problem.

The Basic Paradigm

- Many real-life problems can be modeled as Integer Linear Programs (IP). Since IPs are NP-hard to solve, they are often *relaxed* to a linear program (shortened as LP).
- Modus operandi: solve the linear program in polynomial time, and extract useful information about an integer optimum solution.
- Since the solution to the LP may not be integral, we may need to *massage* the LP fractional solution to an integral one. The hope is that this is a *good* integral solution.
- However, for certain problems, we do not need to even solve the LP to get good (reasonable approximation factor) solutions to our problem.

Primal Dual Algorithms

Over the years, the primal dual paradigm has been used for:

- Approximation Algorithms (bicriteria too)
- Semidefinite Programs
- Online Algorithms
- ... etc.

Primal Dual Algorithms

Over the years, the primal dual paradigm has been used for:

- Approximation Algorithms (bicriteria too)
- Semidefinite Programs
- Online Algorithms
- ... etc.

Primal Dual Algorithms

Over the years, the primal dual paradigm has been used for:

- Approximation Algorithms (bicriteria too)
- Semidefinite Programs
- Online Algorithms
- ... etc.

Primal Dual Algorithms

Over the years, the primal dual paradigm has been used for:

- Approximation Algorithms (bicriteria too)
- Semidefinite Programs
- Online Algorithms
- ... etc.

- George Dantzig started linear programming, and his ideas contain the first germs of primal dual algorithms. The Hungarian method was an application of the paradigm.
- Edmonds (1965) gave the first (sophisticated) application of the paradigm in his work on maximum weight matchings in arbitrary graphs.
- Bar-Yehuda and Even (1981) first enunciated the paradigm in their work on the weighted Vertex Cover problem.

History

- Agrawal, Klein and Ravi (1991) developed sophisticated primal dual algorithms for Steiner Network problems.
- Their paradigm was utilized by Goemans, Williamson in 1994 to prove a seminal 2-factor approximation algorithm for the Steiner Forest problem and its prize collecting variant.
- Jain, Vazirani (1999,2001) showed the powerful use of Lagrangian Relaxations in addition to Primal Dual to give constant factor algorithms for Facility Location, k -Median.
- Garg (1996) gave a 3-factor approximation algorithm for the k -MST problem. Later, Chudak, Roughgarden, Williamson (2004) showed that this is an application of the Lagrangian Relaxation technique.
- Arora, Karakostas (2000) gave a $(2 + \epsilon)$ -factor algorithm for the same problem; Garg (2005) gave a 2-factor approximation.

- For Prize Collecting Vertex Cover, Hochbaum (2002) showed a 2-factor approximation. Later, Bar-Yehuda and others gave local ratio algorithms for the same.
- For Partial Vertex Cover, Gandhi, Khuller and Srinivasan gave the first 2-factor approximation algorithms. Later, Mestre (2005) improved the running time of the algorithm.
- and lot more papers...

Primal, Dual and Weak Duality

Consider a LP in n variables x_1, \dots, x_n , and with m constraints.

Primal

$$\begin{array}{ll}\min & \sum_i c_i x_i \\ \text{s.t.} & a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n \geq b_1 \\ & a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n \geq b_2 \\ & \dots\dots\dots \\ & a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n \geq b_m \\ \text{for all } i & x_i \geq 0\end{array}.$$

The dual is an effort to construct the best lower bound for the primal objective. The dual program will have a variable corresponding to every constraint in the primal; a variable in the primal manifests itself as a dual constraint.

Primal, Dual and Weak Duality

Variables \leftrightarrow Constraints.

Let z_1, \dots, z_m denote the dual variables. The dual is as follows:

Dual

$$\begin{array}{ll}\max & \sum_j b_j z_j \\ \text{s.t.} & a_{11}z_1 + a_{21}z_2 + \dots + a_{m1}z_m \leq c_1 \\ & a_{12}z_1 + a_{22}z_2 + \dots + a_{m2}z_m \leq c_2 \\ & \dots\dots\dots \\ & a_{1n}z_1 + a_{2n}z_2 + \dots + a_{mn}z_m \leq c_n \\ \text{for all } j & z_j \geq 0\end{array}$$

Primal, Dual and Weak Duality

Weak Duality formalizes the notion of “best” lower bound for the (minimization) primal.

Weak Duality

For every feasible solution \vec{x} to the primal and \vec{z} to the dual programs, the following holds:

$$\sum_i c_i x_i \geq \sum_j b_j z_j$$

Proof :

$$\sum_i c_i x_i \geq \sum_i \left(\sum_j a_{ji} z_j \right) x_i = \sum_{i,j} a_{ji} x_i z_j = \sum_j \left(\sum_i a_{ji} x_i \right) z_j \geq \sum_j b_j z_j$$

Conditions for Optimality: Complementary Slackness

The (max) dual objective provides a lower bound for the (min) primal objective. The two are equal iff

Primal Complementary Slackness

$$x_i > 0 \implies \sum_j a_{ji} z_j = c_i$$

and

Dual Complementary Slackness

$$z_j > 0 \implies \sum_i a_{ji} x_i = b_j$$

Relaxed Complementary Slackness

- The idea behind the primal dual paradigm arises from the fact that we may *relax* the primal and dual slackness conditions in order to get approximation algorithms (instead of optimal algorithms).
- Suppose we have a set of feasible solutions (\vec{x}, \vec{z}) for which *relaxed* dual slackness conditions hold, as follows:

Dual Complementary Slackness

$$z_j > 0 \implies \sum_i a_{ji} x_i \leq r \cdot b_j$$

for some factor $r \geq 1$.

Relaxed Complementary Slackness

- Suppose also, that the primal feasible solution \vec{x} is *integral*. Consider the following calculation (almost the same as that for Weak Duality).

$$\sum_i c_i x_i = \sum_i \left(\sum_j a_{ji} z_j \right) x_i = \sum_{i,j} a_{ji} x_i z_j = \sum_j \left(\sum_i a_{ji} x_i \right) z_j \leq r \cdot \sum_j b_j z_j$$

- Thus, by Weak Duality, the feasible solution \vec{x} is a r -factor approximation to the optimal, OPT. Why? Weak Duality implies $\sum_j b_j z_j \leq \sum_i c_i x'_i$ for any feasible \vec{x}' , in particular for the optimal (integral) solution OPT. Thus

$$\sum_i c_i x_i \leq r \cdot \sum_j b_j z_j \leq r \cdot \text{OPT}$$

- We could also relax the primal slackness conditions, with a similar calculation.

LP for Vertex Cover

The IP/LP for Vertex Cover is as follows. We are given a graph $G = (V, E)$.

Primal

$$\begin{array}{ll}\min & \sum_{i \in V} c_i x_i \\ \text{s.t.} & x_i + x_j \geq 1 \quad \text{for all } (i, j) = e \in E \\ & x_i \geq 0 \quad \text{for all } i \in V.\end{array}$$

The dual is as follows:

Dual

$$\begin{array}{ll}\max & \sum_{e \in E} z_e \\ \text{s.t.} & z(\delta(i)) \leq c_i \quad \text{for all } i \in V \\ & z_e \geq 0 \quad \text{for all } e \in E.\end{array}$$

LP for Vertex Cover

The IP/LP for Vertex Cover is as follows. We are given a graph $G = (V, E)$.

Primal

$$\begin{array}{ll}\min & \sum_{i \in V} c_i x_i \\ \text{s.t.} & x_i + x_j \geq 1 \quad \text{for all } (i, j) = e \in E \\ & x_i \geq 0 \quad \text{for all } i \in V.\end{array}$$

The dual is as follows:

Dual

$$\begin{array}{ll}\max & \sum_{e \in E} z_e \\ \text{s.t.} & z(\delta(i)) \leq c_i \quad \text{for all } i \in V \\ & z_e \geq 0 \quad \text{for all } e \in E.\end{array}$$

LP for Vertex Cover

The IP/LP for Vertex Cover is as follows. We are given a graph $G = (V, E)$.

Primal

$$\begin{array}{ll}\min & \sum_{i \in V} c_i x_i \\ \text{s.t.} & x_i + x_j \geq 1 \quad \text{for all } (i, j) = e \in E \\ & x_i \geq 0 \quad \text{for all } i \in V.\end{array}$$

The dual is as follows:

Dual

$$\begin{array}{ll}\max & \sum_{e \in E} z_e \\ \text{s.t.} & z(\delta(i)) \leq c_i \quad \text{for all } i \in V \\ & z_e \geq 0 \quad \text{for all } e \in E.\end{array}$$

- Of course, we see that in the Vertex Cover LP above, for every edge $e \in E$, we are guaranteed that one of the two endpoints x_i or x_j is $\geq 1/2$.
- Thus, we may just *round up* the x_i 's that are $\geq 1/2$. Easy to check that
 - 1 The resulting (integer) solution is indeed a valid vertex cover.
 - 2 The cost of the solution is at most *twice* that of the LP value.
- However, our goal (in this talk) is to use the primal dual paradigm.

Primal Dual for Vertex Cover

Primal

$$\begin{array}{ll}\min & \sum_{i \in V} c_i x_i \\ \text{s.t.} & x_i + x_j \geq 1 \quad \text{for all } (i, j) = e \in E \\ & x_i \geq 0 \quad \text{for all } i \in V.\end{array}$$

Dual

$$\begin{array}{ll}\max & \sum_{e \in E} z_e \\ \text{s.t.} & z(\delta(i)) \leq c_i \quad \text{for all } i \in V \\ & z_e \geq 0 \quad \text{for all } e \in E.\end{array}$$

- Start with the *integer infeasible* primal solution $x = 0$, and the *feasible* dual solution $z = 0$.
Repeat while some constraint in primal is unsatisfied:
- Increase all (unfrozen) variables z_e until some dual constraint becomes tight (say, for vertex i).
- Set $x_i = 1$. *Freeze* all the variables z_e such that e is incident to i .

Primal Dual for Vertex Cover

Primal

$$\begin{array}{ll}\min & \sum_{i \in V} c_i x_i \\ \text{s.t.} & x_i + x_j \geq 1 \quad \text{for all } (i, j) = e \in E \\ & x_i \geq 0 \quad \text{for all } i \in V.\end{array}$$

Dual

$$\begin{array}{ll}\max & \sum_{e \in E} z_e \\ \text{s.t.} & z(\delta(i)) \leq c_i \quad \text{for all } i \in V \\ & z_e \geq 0 \quad \text{for all } e \in E.\end{array}$$

- Start with the *integer infeasible* primal solution $x = 0$, and the *feasible* dual solution $z = 0$.

Repeat while some constraint in primal is unsatisfied:

- Increase all (unfrozen) variables z_e until some dual constraint becomes tight (say, for vertex i).
- Set $x_i = 1$. *Freeze* all the variables z_e such that e is incident to i .

Primal Dual for Vertex Cover

Primal

$$\begin{array}{ll}\min & \sum_{i \in V} c_i x_i \\ \text{s.t.} & x_i + x_j \geq 1 \quad \text{for all } (i, j) = e \in E \\ & x_i \geq 0 \quad \text{for all } i \in V.\end{array}$$

Dual

$$\begin{array}{ll}\max & \sum_{e \in E} z_e \\ \text{s.t.} & z(\delta(i)) \leq c_i \quad \text{for all } i \in V \\ & z_e \geq 0 \quad \text{for all } e \in E.\end{array}$$

- Start with the *integer infeasible* primal solution $x = 0$, and the *feasible* dual solution $z = 0$.

Repeat while some constraint in primal is unsatisfied:

- Increase all (unfrozen) variables z_e until some dual constraint becomes tight (say, for vertex i).
- Set $x_i = 1$. *Freeze* all the variables z_e such that e is incident to i .

Primal Dual for Vertex Cover

Primal

$$\begin{array}{ll}\min & \sum_{i \in V} c_i x_i \\ \text{s.t.} & x_i + x_j \geq 1 \quad \text{for all } (i, j) = e \in E \\ & x_i \geq 0 \quad \text{for all } i \in V.\end{array}$$

Dual

$$\begin{array}{ll}\max & \sum_{e \in E} z_e \\ \text{s.t.} & z(\delta(i)) \leq c_i \quad \text{for all } i \in V \\ & z_e \geq 0 \quad \text{for all } e \in E.\end{array}$$

- Start with the *integer infeasible* primal solution $x = 0$, and the *feasible* dual solution $z = 0$. **Repeat** while some constraint in primal is unsatisfied:
- Increase all (unfrozen) variables z_e until some dual constraint becomes tight (say, for vertex i).
- Set $x_i = 1$. *Freeze* all the variables z_e such that e is incident to i .

Primal Dual for Vertex Cover

Primal

$$\begin{array}{ll}\min & \sum_{i \in V} c_i x_i \\ \text{s.t.} & x_i + x_j \geq 1 \quad \text{for all } (i, j) = e \in E \\ & x_i \geq 0 \quad \text{for all } i \in V.\end{array}$$

Dual

$$\begin{array}{ll}\max & \sum_{e \in E} z_e \\ \text{s.t.} & z(\delta(i)) \leq c_i \quad \text{for all } i \in V \\ & z_e \geq 0 \quad \text{for all } e \in E.\end{array}$$

- Start with the *integer infeasible* primal solution $x = 0$, and the *feasible* dual solution $z = 0$.
Repeat while some constraint in primal is unsatisfied:
- Increase all (unfrozen) variables z_e until some dual constraint becomes tight (say, for vertex i).
- Set $x_i = 1$. *Freeze* all the variables z_e such that e is incident to i .

In the end...

- When the above process stops, we have increased the variables z_e suitably.
- Some vertices i were chosen (indicated by $x_i = 1$).
- This set of vertices is our output Vertex Cover.

- The set S of vertices output is indeed a Vertex Cover. Why? Because we would have continued if some primal constraint were still unsatisfied.
- Cost of the solution? This is where we arrive at the *relaxed complementary slackness conditions*.

Relaxed Complementary Slackness

Two conditions hold:

- The primal complementary slackness conditions:

$$x_i > 0 \implies z(\delta(i)) = c_i$$

- The dual (relaxed) slackness conditions:

$$z_e > 0 \implies x_i + x_j \leq 2$$

The two conditions above imply that the above algorithm is a 2-factor approximation.

Further Comments

- In the above algorithm, we increased the (active) dual variables *simultaneously*.
- What we are essentially trying is to get the highest (the best) *lower bound* that we can get for the primal minimization objective.
- We may also consider increasing the dual variables *one-by-one*. This would be a different algorithm (under the same primal dual paradigm). In the case of Vertex Cover, this also works to give a 2-factor approximation.

Prize Collecting Vertex Cover (PCVC)

- In this version of the problem, the set of vertices chosen does not need to cover every edge. However, there is a *penalty* p_e for an edge e left uncovered.
- The objective is to minimize the (cumulative) cost of the “vertex cover” chosen and the total penalty incurred (for leaving edges uncovered).
- The IP formulation? We will keep an indicator variable y_e that is 1 iff the edge e is left *uncovered*.

PCVC: The LP relaxation

The primal LP is as follows:

Primal

$$\begin{array}{ll}\min & \sum_{i \in V} c_i x_i + \sum_{e \in E} p_e y_e \\ \text{s.t.} & x_i + x_j + y_e \geq 1 \quad \text{for all } (i, j) = e \in E \\ & x_i, y_e \geq 0 \quad \text{for all } i \in V, e \in E.\end{array}$$

Dual

$$\begin{array}{ll}\max & \sum_{e \in E} z_e \\ \text{s.t.} & z(\delta(i)) \leq c_i \quad \text{for all } i \in V \\ & z_e \leq p_e \quad \text{for all } e \in E \\ & z_e \geq 0 \quad \text{for all } e \in E.\end{array}$$

PCVC: The LP relaxation

The primal LP is as follows:

Primal

$$\begin{array}{ll}\min & \sum_{i \in V} c_i x_i + \sum_{e \in E} p_e y_e \\ \text{s.t.} & x_i + x_j + y_e \geq 1 \quad \text{for all } (i, j) = e \in E \\ & x_i, y_e \geq 0 \quad \text{for all } i \in V, e \in E.\end{array}$$

Dual

$$\begin{array}{ll}\max & \sum_{e \in E} z_e \\ \text{s.t.} & z(\delta(i)) \leq c_i \quad \text{for all } i \in V \\ & z_e \leq p_e \quad \text{for all } e \in E \\ & z_e \geq 0 \quad \text{for all } e \in E.\end{array}$$

- What does rounding give? Each primal constraint now consists of 3 variables, so some variable has to have value $\geq 1/3$. This straightaway gives a 3-factor approximation.
- On to primal dual.

Rehash of earlier primal dual

- Start with the *integer infeasible* primal solution $\vec{x} = \vec{0}$, $\vec{y} = \vec{0}$, and the *feasible* dual solution $\vec{z} = \vec{0}$.

Repeat while some primal constraint is unsatisfied:

- Increase all (unfrozen) variables z_e until some dual constraint becomes tight.
- However, here, we have two types of dual constraints, one of which could have become tight
 - Either the constraint is of the type " $z(\delta(i)) \leq c_i$ ". In this case, we set $x_i = 1$, and for all $e \in E$ incident to i , we set $y_e = 0$.
 - Or the constraint is of the type " $z_e \leq p_e$ ". In this case, we set $y_e = 1$ and set $x_i = x_j = 0$ where $e = (i, j)$.
- Note that the algorithm (as stated so far) is a trivial extension of the primal dual algorithm that we presented for Vertex Cover.

Rehash of earlier primal dual

- Start with the *integer infeasible* primal solution $\vec{x} = \vec{0}$, $\vec{y} = \vec{0}$, and the *feasible* dual solution $\vec{z} = \vec{0}$.

Repeat while some primal constraint is unsatisfied:

- Increase all (unfrozen) variables z_e until some dual constraint becomes tight.
- However, here, we have two types of dual constraints, one of which could have become tight
 - Either the constraint is of the type " $z(\delta(i)) \leq c_i$ ". In this case, we set $x_i = 1$, and for all $e \in E$ incident to i , we set $y_e = 0$.
 - Or the constraint is of the type " $z_e \leq p_e$ ". In this case, we set $y_e = 1$ and set $x_i = x_j = 0$ where $e = (i, j)$.
- Note that the algorithm (as stated so far) is a trivial extension of the primal dual algorithm that we presented for Vertex Cover.

Rehash of earlier primal dual

- Start with the *integer infeasible* primal solution $\vec{x} = \vec{0}$, $\vec{y} = \vec{0}$, and the *feasible* dual solution $\vec{z} = \vec{0}$.

Repeat while some primal constraint is unsatisfied:

- Increase all (unfrozen) variables z_e until some dual constraint becomes tight.
- However, here, we have two types of dual constraints, one of which could have become tight
 - Either the constraint is of the type " $z(\delta(i)) \leq c_i$ ". In this case, we set $x_i = 1$, and for all $e \in E$ incident to i , we set $y_e = 0$.
 - Or the constraint is of the type " $z_e \leq p_e$ ". In this case, we set $y_e = 1$ and set $x_i = x_j = 0$ where $e = (i, j)$.
- Note that the algorithm (as stated so far) is a trivial extension of the primal dual algorithm that we presented for Vertex Cover.

Rehash of earlier primal dual

- Start with the *integer infeasible* primal solution $\vec{x} = \vec{0}, \vec{y} = \vec{0}$, and the *feasible* dual solution $\vec{z} = \vec{0}$.

Repeat while some primal constraint is unsatisfied:

- Increase all (unfrozen) variables z_e until some dual constraint becomes tight.
- However, here, we have two types of dual constraints, one of which could have become tight
 - Either the constraint is of the type " $z(\delta(i)) \leq c_i$ ". In this case, we set $x_i = 1$, and for all $e \in E$ incident to i , we set $y_e = 0$.
 - Or the constraint is of the type " $z_e \leq p_e$ ". In this case, we set $y_e = 1$ and set $x_i = x_j = 0$ where $e = (i, j)$.
- Note that the algorithm (as stated so far) is a trivial extension of the primal dual algorithm that we presented for Vertex Cover.

Rehash of earlier primal dual

- Start with the *integer infeasible* primal solution $\vec{x} = \vec{0}$, $\vec{y} = \vec{0}$, and the *feasible* dual solution $\vec{z} = \vec{0}$.

Repeat while some primal constraint is unsatisfied:

- Increase all (unfrozen) variables z_e until some dual constraint becomes tight.
- However, here, we have two types of dual constraints, one of which could have become tight
 - Either the constraint is of the type " $z(\delta(i)) \leq c_i$ ". In this case, we set $x_i = 1$, and for all $e \in E$ incident to i , we set $y_e = 0$.
 - Or the constraint is of the type " $z_e \leq p_e$ ". In this case, we set $y_e = 1$ and set $x_i = x_j = 0$ where $e = (i, j)$.
- Note that the algorithm (as stated so far) is a trivial extension of the primal dual algorithm that we presented for Vertex Cover.

Issue in the algorithm for PCVC

- However the following issue arises: At the end of the step above, it may happen that for an edge e , we set $y_e = 1$ first and then later, because of some constraint of the type " $z(\delta(i)) \leq c_i$ ", we end up setting $x_i = 1$ for i an endpoint of y_e . As such, this would therefore give us a 3-factor approximation.
- **Delete Step** Thus, we add the following *last* step: at the end of the previous step, if $x_i = 1$ for an endpoint i of an edge e such that $y_e = 1$, then reset $y_e = 0$.
- Now we are ready to do the analysis of the algorithm.

We will again check the relaxed complementary slackness conditions:

- Primal Complementary Slackness: If $x_i > 0$, then $z(\delta(i)) = c_i$. If $y_e > 0$, then $z_e = p_e$.
This is relatively clear from the algorithm..
- Dual Complementary Slackness: If $z_e > 0$, then the corresponding primal constraint $x_i + x_j + y_e \leq 2$. Why?
 - 1 Either the constraint " $z(\delta(i)) \leq c_i$ " was made tight. In this case, $x_i + x_j + y_e \leq 2$, since y_e was set equal to 0 for every e incident on i .
 - 2 Or the constraint " $z_e \leq p_e$ " was made tight. In this case, observe that in the **Delete Step** of the algorithm, either y_e was reset to 0 (because some endpoint i of e was picked: $x_i = 1$). In this case, $x_i + x_j + y_e \leq 2$. Or it was the case that y_e retained its value of 1; thus, $x_i + x_j + y_e = 1$.
- Thus, what we have proven is ...

- For the solution \vec{x}, \vec{y} generated by the above algorithm, the following holds:

$$\left(\sum_{i \in V} c_i x_i + \sum_{e \in E} p_e y_e\right) \leq 2 \times \left(\sum_e z_e\right)$$

- But, by Weak Duality, $\sum_e z_e \leq \text{OPT}$. Thus, the solution generated is upper bounded by 2OPT .
- In fact more holds true:

$$\left(\sum_{i \in V} c_i x_i + 2 \sum_{e \in E} p_e y_e\right) \leq 2 \times \left(\sum_e z_e\right)$$

- Why? Note that for the outcome generated by the algorithm, $x_i + x_j + 2y_e \leq 2$.

PVC: Problem Definition

- We will see that the stronger statement made above is precisely what is used for *partial cover* problems. This will take us into the domain of Lagrangian relaxations. We will not go there, right yet.
- The *Partial Vertex Cover* (PVC) problem is the following: We are given a graph $G = (V, E)$ and a number k . A **partial vertex cover** is a set of vertices chosen such that the vertices cover *at least* k of the edges in the graph.
- We start by formulating the problem as an integer (linear) program.

Primal

$$\begin{array}{ll}\min & \sum_{i \in V} c_i x_i \\ \text{s.t.} & x_i + x_j + y_e \geq 1 \quad \text{for all } (i, j) = e \in E \\ & \sum_{e \in E} y_e \leq s \\ & x_i, y_e \geq 0 \quad \text{for all } i \in V, e \in E.\end{array}$$

Here, $s = (m - k)$ is the maximum number of edges that can be left *uncovered* by a valid (partial) vertex cover.

PVC: a rewrite

For understanding the integrality gap of the relaxation, it might help to consider the re-writing of the above LP.

Primal

$$\begin{aligned} \min \quad & \sum_{i \in V} c_i x_i \\ \text{s.t.} \quad & x_i + x_j \geq y'_e \quad \text{for all } (i, j) = e \in E \\ & \sum_{e \in E} y'_e \geq k \\ & y'_e \leq 1 \quad \text{for all } e \in E \\ & x_i, y'_e \geq 0 \quad \text{for all } i \in V, e \in E. \end{aligned}$$

Here, we have just made the substitution $y'_e = (1 - y_e)$. This makes the LP look more “homogeneous”.

Integrality Gap

- The LPs in the previous slide(s) have *bad* integrality gap.
- Consider the “star” graph $K_{1,m}$, and let $s = (m - 1)$ (i.e. $k=1$: one has to cover just one edge). Let each vertex have cost $c_i = 1$.
- An integer solution has to pick some vertex, and has a cost of 1.
- A fractional LP solution can pick the central vertex with $x_i = 1/m$ (every other vertex is given $x_j = 0$), and for every edge, e , set $y'_e = 1/m$ (i.e. $y_e = (m - 1)/m$). Thus, the cost of this solution is $1/m$.
- Thus, there is a gap of $\Omega(m)$ between the optimal LP and the optimal IP solution.

Integrality Gap

- The LPs in the previous slide(s) have *bad* integrality gap.
- Consider the “star” graph $K_{1,m}$, and let $s = (m - 1)$ (i.e. $k=1$: one has to cover just one edge). Let each vertex have cost $c_i = 1$.
- An integer solution has to pick some vertex, and has a cost of 1.
- A fractional LP solution can pick the central vertex with $x_i = 1/m$ (every other vertex is given $x_j = 0$), and for every edge, e , set $y'_e = 1/m$ (i.e. $y_e = (m - 1)/m$). Thus, the cost of this solution is $1/m$.
- Thus, there is a gap of $\Omega(m)$ between the optimal LP and the optimal IP solution.

Integrality Gap

- The LPs in the previous slide(s) have *bad* integrality gap.
- Consider the “star” graph $K_{1,m}$, and let $s = (m - 1)$ (i.e. $k=1$: one has to cover just one edge). Let each vertex have cost $c_i = 1$.
- An integer solution has to pick some vertex, and has a cost of 1.
- A fractional LP solution can pick the central vertex with $x_i = 1/m$ (every other vertex is given $x_j = 0$), and for every edge, e , set $y'_e = 1/m$ (i.e. $y_e = (m - 1)/m$). Thus, the cost of this solution is $1/m$.
- Thus, there is a gap of $\Omega(m)$ between the optimal LP and the optimal IP solution.

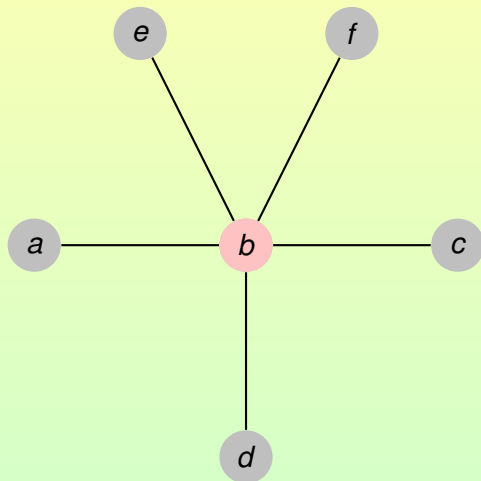
Integrality Gap

- The LPs in the previous slide(s) have *bad* integrality gap.
- Consider the “star” graph $K_{1,m}$, and let $s = (m - 1)$ (i.e. $k=1$: one has to cover just one edge). Let each vertex have cost $c_i = 1$.
- An integer solution has to pick some vertex, and has a cost of 1.
- A fractional LP solution can pick the central vertex with $x_i = 1/m$ (every other vertex is given $x_j = 0$), and for every edge, e , set $y'_e = 1/m$ (i.e. $y_e = (m - 1)/m$). Thus, the cost of this solution is $1/m$.
- Thus, there is a gap of $\Omega(m)$ between the optimal LP and the optimal IP solution.

Integrality Gap

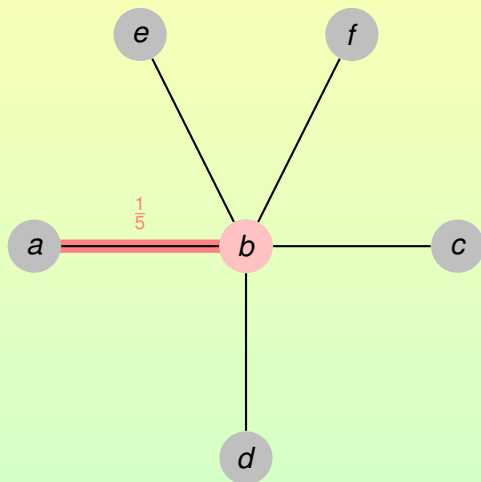
- The LPs in the previous slide(s) have *bad* integrality gap.
- Consider the “star” graph $K_{1,m}$, and let $s = (m - 1)$ (i.e. $k=1$: one has to cover just one edge). Let each vertex have cost $c_i = 1$.
- An integer solution has to pick some vertex, and has a cost of 1.
- A fractional LP solution can pick the central vertex with $x_i = 1/m$ (every other vertex is given $x_j = 0$), and for every edge, e , set $y'_e = 1/m$ (i.e. $y_e = (m - 1)/m$). Thus, the cost of this solution is $1/m$.
- Thus, there is a gap of $\Omega(m)$ between the optimal LP and the optimal IP solution.

LP solution



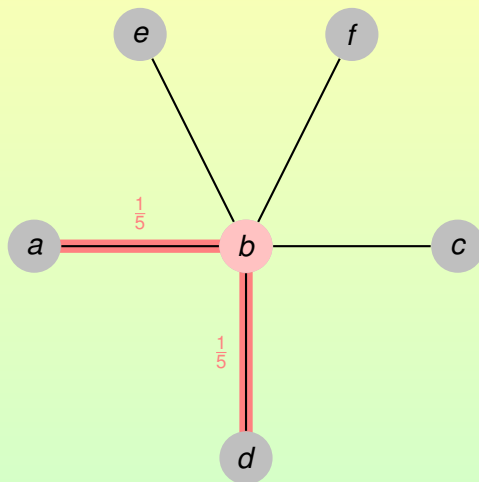
Cost of LP solution: $\frac{1}{5}$.

LP solution



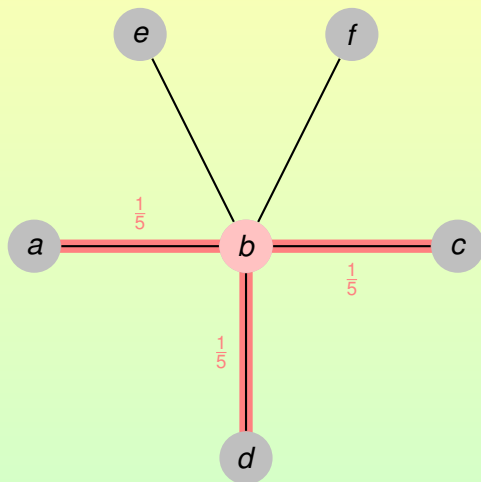
Cost of LP solution: $\frac{1}{5}$.

LP solution



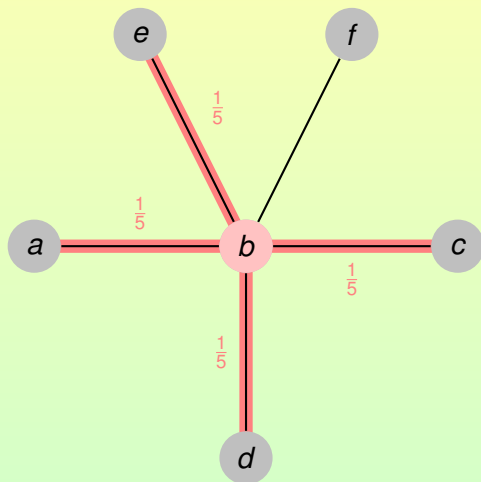
Cost of LP solution: $\frac{1}{5}$.

LP solution



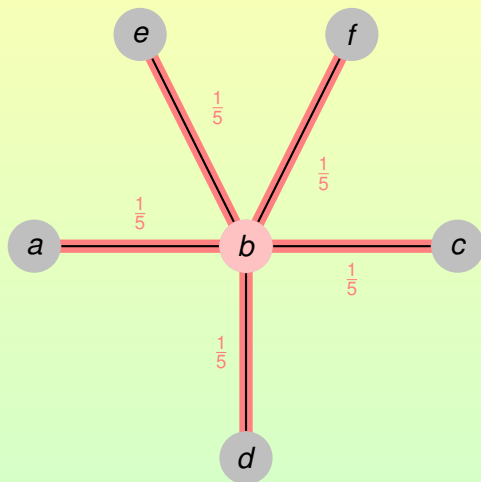
Cost of LP solution: $\frac{1}{5}$.

LP solution



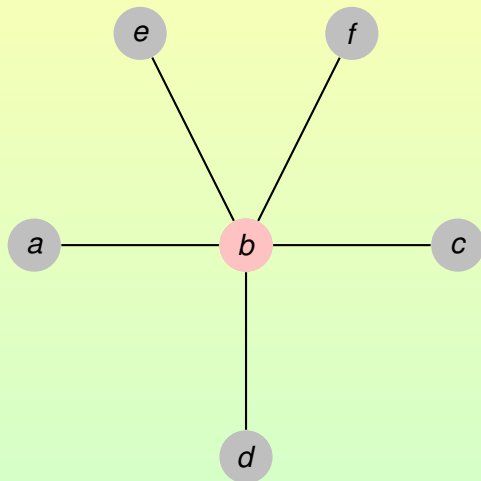
Cost of LP solution: $\frac{1}{5}$.

LP solution



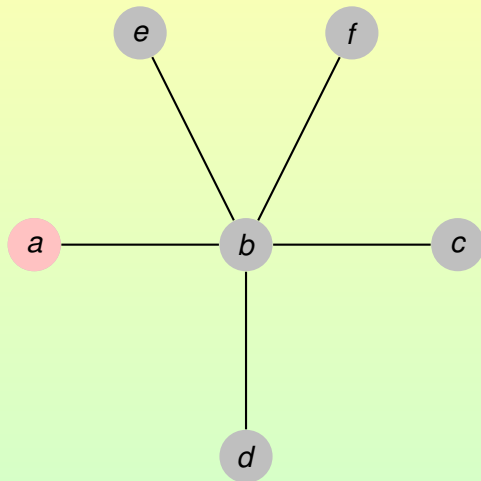
Cost of LP solution: $\frac{1}{5}$.

IP solution



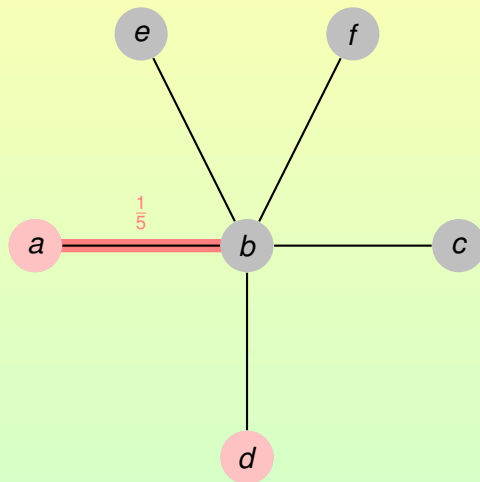
Cost of IP solution: 1.

Note however...



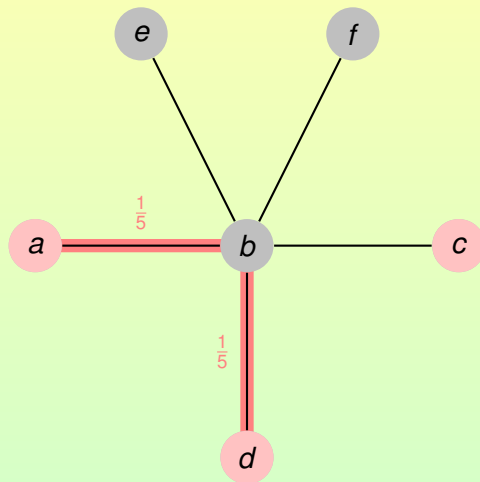
If we had chosen the non-central vertices, then cost of LP solution equals the cost of IP solution.

Note however...



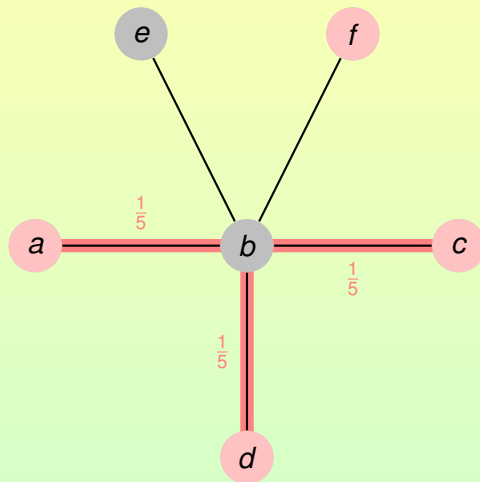
If we had chosen the non-central vertices, then cost of LP solution equals the cost of IP solution.

Note however...



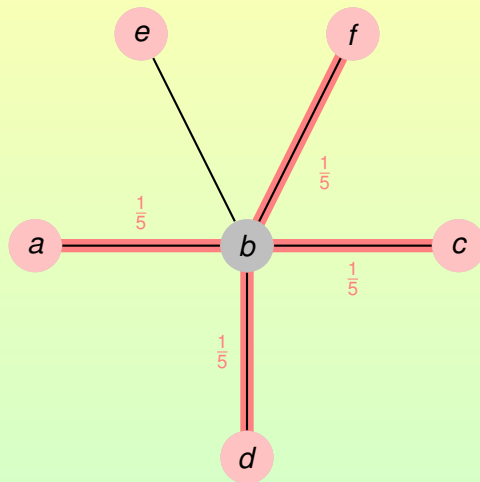
If we had chosen the non-central vertices, then cost of LP solution equals the cost of IP solution.

Note however...



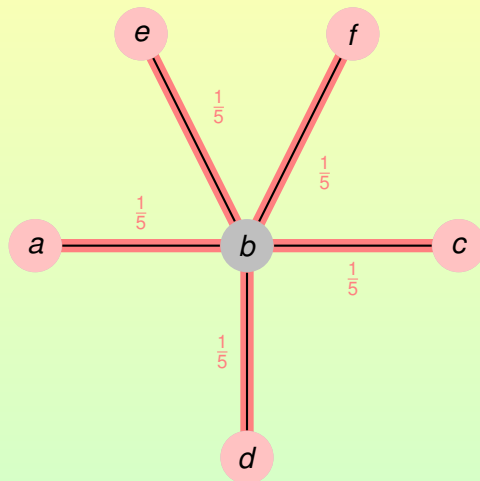
If we had chosen the non-central vertices, then cost of LP solution equals the cost of IP solution.

Note however...



If we had chosen the non-central vertices, then cost of LP solution equals the cost of IP solution.

Note however...



If we had chosen the non-central vertices, then cost of LP solution equals the cost of IP solution.

- Thus, the LP can serve as a better lower bound, provided we give it some *guidance*.
- What kind of guidance can we give it? Is the guidance "stay away from high degree vertices"?
- Well, we are trying to cover edges, so this should not be the guidance.
- In order to uncover the guidance, let us play with the costs of the vertices. After all, they are what figure in the objective function.

Guide the LP: Playing with the costs

- Consider (again) the graph $K_{1,m}$.
- Suppose, the central vertex had *high cost* 20 (say), while the other vertices retain their costs of 1.
- The IP solution still has cost 1. The LP solution has cost $20/m$ which can be very small (as m grows).
- So the guidance can be: reject the high cost vertex, i.e. $x_i = 0$ for the high cost vertex i .
- But of course, we have to “iterate” this guidance – i.e. the relevant *troublesome* high cost vertex may not be the highest cost, but may be slightly *hidden*.

Thus, the motto is:

Guidance

Order the vertices by non-decreasing costs. Let this order be $\{v_1, v_2, \dots, v_n\}$. Imagine *disallowing* the graph beyond vertex v_i (i.e. *guess* that OPT does not have any vertex of cost higher than v_i). Solve the modified LP.

- Since we do not know the highest cost vertex in OPT, we will have to *guess* it. Thus in effect we will have to solve n LPs.
- Note that the earlier integrality gap example vanishes now.
- So, are we in a better state now?

Algo

- Order the vertices by non-decreasing costs. Let the order be $\{v_1, v_2, \dots, v_n\}$.
- Let h be the highest cost *guessed* vertex in OPT.
- Modify costs: For $i > h$, set $c'_i = \infty$; for $i < h$, set $c'_i = c_i$.
- Since we assume that vertex v_h is in the desired optimum, update k (the number of edges to be covered) as follows:
 $k' = k - \deg(v_h)$. Remove vertex h from consideration.
- Run **usual** Vertex Cover (primal dual) algorithm on the residual graph until k' edges are covered. Let the cover output be S .
- Output $C = S \cup \{h\}$ as the generated Vertex Cover.

Primal

$$\begin{aligned} \min \quad & \sum_{i \in V'} c_i x_i \\ \text{s.t.} \quad & x_i + x_j + y_e \geq 1 \quad \text{for all } (i, j) = e \in E' \\ & \sum_{e \in E'} y_e \leq s \\ & x_i, y_e \geq 0 \quad \text{for all } i \in V', e \in E'. \end{aligned}$$

Dual

$$\begin{aligned} \max \quad & \sum_{e \in E'} z_e - s\lambda \\ \text{s.t.} \quad & z(\delta(i)) \leq c'_i \quad \text{for all } i \in V' \\ & z_e \leq \lambda \\ & z_e, \lambda \geq 0 \quad \text{for all } e \in E'. \end{aligned}$$

The **usual** Vertex Cover run on G'

- **Initialize Primal:** Set all x_i 's to 0, all y_e 's to 1. All the primal constraints are satisfied but for the cardinality constraint.
- **Initialize Dual:** Set all $z_e = 0$, set $\lambda = 0$.
- While all the constraints in the primal are not satisfied:
 - ① (Simultaneously) increase *active* dual variables z_e such that some dual constraint (corresponding to vertex $i \in V'$) becomes tight.
 - ② Set $x_i = 1$, set $y_e = 0$ for all e incident on vertex i .
 - ③ Check if $\sum_e y_e \leq s$. If YES, STOP. If NO, continue.
- Set $\lambda = \max_e z_e$.

Analysis

- Suppose OPT contains v_h as the vertex of highest cost. Consider the residual graph G' on vertex set $V' = V \setminus \{v_h\}$, with modified costs c' . Let E' denote the residual set of edges in G' .
- The costs c' are modified as follows: For a vertex v_t if $c_t > c_h$, then $c'_t = \infty$; else $c'_t = c_t$.
- Modify k as follows: k' (the coverage requirement on G') $= k - \deg(v_h)$. Note that the *non-coverage* requirement stays the same, s .
- Note that we *do not* throw away vertices of higher cost. This is because they still may have edges incident on them that may be useful for our solution.
- Let OPT' denote the optimum on the graph G' . Thus $\text{OPT} = \text{OPT}' + c_h$.

Primal

$$\begin{aligned} \min \quad & \sum_{i \in V'} c_i x_i \\ \text{s.t.} \quad & x_i + x_j + y_e \geq 1 \quad \text{for all } (i, j) = e \in E' \\ & \sum_{e \in E'} y_e \leq s \\ & x_i, y_e \geq 0 \quad \text{for all } i \in V', e \in E'. \end{aligned}$$

Dual

$$\begin{aligned} \max \quad & \sum_{e \in E'} z_e - s\lambda \\ \text{s.t.} \quad & z(\delta(i)) \leq c'_i \quad \text{for all } i \in V' \\ & z_e \leq \lambda \\ & z_e, \lambda \geq 0 \quad \text{for all } e \in E'. \end{aligned}$$

Primal

$$\begin{aligned} \min \quad & \sum_{i \in V'} c_i x_i \\ \text{s.t.} \quad & x_i + x_j + y_e \geq 1 \quad \text{for all } (i, j) = e \in E' \\ & \sum_{e \in E'} y_e \leq s \\ & x_i, y_e \geq 0 \quad \text{for all } i \in V', e \in E'. \end{aligned}$$

Dual

$$\begin{aligned} \max \quad & \sum_{e \in E'} z_e - s\lambda \\ \text{s.t.} \quad & z(\delta(i)) \leq c'_i \quad \text{for all } i \in V' \\ & z_e \leq \lambda \\ & z_e, \lambda \geq 0 \quad \text{for all } e \in E'. \end{aligned}$$

Analysis

- In the run of the primal dual on the modified graph G' , let v_l be the **last** vertex chosen. Let S' denote the other vertices chosen by the primal dual algorithm (the final vertex cover output is $C = S' \cup \{v_l, v_h\}$).
- We will prove

Bound on $c(S')$

$$c(S') \leq 2 \cdot \left(\sum_{e \in E'} z_e - s\lambda \right) \leq 2 \cdot \text{OPT}'$$

- This proves the result (noting that $c_l \leq c_h$):

Bound on output vertex cover

$$c(C) = c(S') + c_l + c_h \leq 2 \cdot \text{OPT}' + c_l + c_h \leq 2 \cdot (\text{OPT}' + c_h) = 2 \cdot \text{OPT}$$

Analysis

- In the run of the primal dual on the modified graph G' , let v_l be the **last** vertex chosen. Let S' denote the other vertices chosen by the primal dual algorithm (the final vertex cover output is $C = S' \cup \{v_l, v_h\}$).
- We will prove

Bound on $c(S')$

$$c(S') \leq 2 \cdot \left(\sum_{e \in E'} z_e - s\lambda \right) \leq 2 \cdot \text{OPT}'$$

- This proves the result (noting that $c_l \leq c_h$):

Bound on output vertex cover

$$c(C) = c(S') + c_l + c_h \leq 2 \cdot \text{OPT}' + c_l + c_h \leq 2 \cdot (\text{OPT}' + c_h) = 2 \cdot \text{OPT}$$

Analysis – bound on $c(S')$

- Why did we have to choose the last vertex v_l ?
- Simply because, till that point $\sum_e y_e > s$ still (not enough coverage).
- But while choosing the last vertex v_l , we do not change the dual variables z_e any more! Thus, the final λ is already set at this point.
- Let us consider the situation just before choosing the last vertex v_l .

Analysis – bound on $c(S')$

- We will simply check the dual complementary slackness conditions at the point *right before* the choice of the last vertex v_l . The primal slackness conditions hold (tightly).

Dual Slackness

$$z_e > 0 \implies x_i + x_j + y_e \leq 2$$

$$\lambda > 0 \implies \sum_{e \in E'} y_e > s$$

- But this precisely proves the bound on $c(S')$ (cf. slide on Relaxed Complementary Slackness).

Analysis – bound on $c(S')$

- We will simply check the dual complementary slackness conditions at the point *right before* the choice of the last vertex v_l . The primal slackness conditions hold (tightly).

Dual Slackness

$$z_e > 0 \implies x_i + x_j + y_e \leq 2$$

$$\lambda > 0 \implies \sum_{e \in E'} y_e > s$$

- But this precisely proves the bound on $c(S')$ (cf. slide on Relaxed Complementary Slackness).

Analysis – bound on $c(S')$

- We will simply check the dual complementary slackness conditions at the point *right before* the choice of the last vertex v_l . The primal slackness conditions hold (tightly).

Dual Slackness

$$z_e > 0 \implies x_i + x_j + y_e \leq 2$$

$$\lambda > 0 \implies \sum_{e \in E'} y_e > s$$

- But this precisely proves the bound on $c(S')$ (cf. slide on Relaxed Complementary Slackness).

Analysis – bound on $c(S')$

- We will simply check the dual complementary slackness conditions at the point *right before* the choice of the last vertex v_l . The primal slackness conditions hold (tightly).

Dual Slackness

$$z_e > 0 \implies x_i + x_j + y_e \leq 2$$

$$\lambda > 0 \implies \sum_{e \in E'} y_e > s$$

- But this precisely proves the bound on $c(S')$ (cf. slide on Relaxed Complementary Slackness).

- Essentially, we compared the cost of an integer **infeasible** primal solution with the feasible dual solution. Later, some of the y_e s were turned to 0, lowering the cost even further.
- Why did we have to stop right before the last vertex v_l chosen by the algorithm? Could we not have enforced some complementary slackness conditions then?
- It may be that the last vertex chosen causes lots of edges to be picked up, so that $\sum_e y_e$ falls down **much below** s . Thus, we might not be able to get any sensible complementary slackness there. For instance, after the choice of the last vertex, it may be that $\sum_e y_e = 0$!
- Thus, the artifice of pinning down the algorithm at the “heaviest” (guessed) vertex is essentially to hedge such problems created by the *last* vertex chosen by the algorithm.

Back to Lagrangian Relaxations

- Jain, Vazirani (2001) show how to use a prize collecting coverage algorithm to construct a partial covering algorithm, using Lagrangian Relaxations.

PCVC with $p_e = \alpha$ for all e

$$\begin{array}{ll}\min & \sum_{i \in V} c_i x_i + \sum_{e \in E} \alpha y_e \\ \text{s.t.} & x_i + x_j + y_e \geq 1 \quad \text{for all } (i, j) = e \in E \\ & x_i, y_e \geq 0 \quad \text{for all } i \in V, e \in E.\end{array}$$

PVC

$$\begin{array}{ll}\min & \sum_{i \in V} c_i x_i \\ \text{s.t.} & x_i + x_j + y_e \geq 1 \quad \text{for all } (i, j) = e \in E \\ & \sum_{e \in E} y_e \leq s.\end{array}$$

From PCVC to PVC

- Formulate a Lagrangian Relaxation of the PVC LP, taking the constraint $\sum_e y_e \leq s$ into the objective function with a Lagrange multiplier α .

Lagrangian Relaxation of PVC

$$\begin{array}{ll} \min & \sum_{i \in V} c_i x_i + \alpha \left(\sum_e y_e - s \right) \\ \text{s.t.} & x_i + x_j + y_e \geq 1 \quad \text{for all } (i, j) = e \in E \\ & x_i, y_e \geq 0 \quad \text{for all } i \in V, e \in E. \end{array}$$

The objective function “looks” like that of PCVC (with an extra αs factor).

- This is a *lower bound* on PVC. Why? For PVC, $\sum_e y_e \leq s$, thus, the additional factor $\alpha(\sum_e y_e - s)$ is non-positive.

From PCVC to PVC

- Suppose we solve the PCVC problem

Lagrangian Relaxation of PVC: LPVC

$$\begin{array}{ll} \min & \sum_{i \in V} c_i x_i + \alpha \sum_e y_e \\ \text{s.t.} & x_i + x_j + y_e \geq 1 \quad \text{for all } (i, j) = e \in E \\ & x_i, y_e \geq 0 \quad \text{for all } i \in V, e \in E. \end{array}$$

via the primal dual algorithm, and the outcome happens to satisfy (for some α) $\sum_e y_e = s$.

- **Claim:** The solution (partial) vertex cover generated is a 2-factor approximation to the PVC problem.

From PCVC to PVC

- Let OPT' denote the optimum integral value of LPVC, and OPT denote that of PVC. Observe that

$$\text{OPT}' \leq \text{OPT} + \alpha s$$

- This is because: any feasible solution (\vec{x}, \vec{y}) to PVC is also feasible for LPVC. The value of the **optimum** solution to PVC, looked upon as a feasible solution to LPVC is $\text{OPT} + \alpha \sum_e y_e \leq \text{OPT} + \alpha s$. The optimum solution to LPVC can only be lower than this (recall that LPVC is a minimization problem).
- Now, note that for the LPVC run (for some α) we have that :

$$\sum_{i \in V} c_i x_i + 2\alpha \sum_e y_e \leq 2\text{OPT}_1$$

Thus, $\sum_{i \in V} c_i x_i \leq 2(\text{OPT}_1 - \alpha s) \leq 2\text{OPT}$.

- So, in the LPVC run, if it turns out that $\sum_e y_e = s$, then we are done.

From PCVC to PVC

- However, that may not happen. For different values of α , we may end up getting $\sum_e y_e < s$ or $\sum_e y_e > s$, but never equal.
- In this case, run the LPVC primal dual algorithm for various values of α (in increasing order) and get two values α_1 and α_2 (sufficiently close) such that for α_1 , the sum $\sum_e y_e = s_1 < s$, while for α_2 , the $\sum_e y_e = s_2 > s$.
- There are two candidate vertex covers such that one covers too many edges, and one covers too few edges.
- The idea then is to *combine* the two vertex covers to get a suitable vertex cover that covers at least k edges, at a small cost.

Combining the two (partial) vertex covers

LPVC

$$\begin{array}{ll}\min & \sum_{i \in V} c_i x_i + \sum_{e \in E} \alpha y_e \\ \text{s.t.} & x_i + x_j + y_e \geq 1 \quad \text{for all } (i, j) = e \in E \\ & x_i, y_e \geq 0 \quad \text{for all } i \in V, e \in E.\end{array}$$

PVC

$$\begin{array}{ll}\min & \sum_{i \in V} c_i x_i \\ \text{s.t.} & x_i + x_j + y_e \geq 1 \quad \text{for all } (i, j) = e \in E \\ & \sum_{e \in E} y_e \leq s.\end{array}$$



Combining the two (partial) vertex covers

- Since the two values of α are sufficiently close, for brevity, we will assume them to be equal, i.e. $\alpha_1 = \alpha_2 = \alpha$. Let OPT_i ($i = 1, 2$) denote the optimum value of the LPVC run with $\alpha = \alpha_i$. Let OPT denote the optimum value of PVC.
- Consider the two solutions C_1 and C_2 . These have the following properties.
 - 1 $c(C_1) + 2\alpha s_1 \leq 2\text{OPT}_1$. Thus,
 $c(C_1) \leq 2\text{OPT}_1 - 2\alpha s_1 \leq 2\text{OPT} + 2\alpha(s - s_1)$
 - 2 $c(C_2) + 2\alpha s_2 \leq 2\text{OPT}_2$. Thus,
 $c(C_2) \leq 2\text{OPT}_2 - 2\alpha s_2 \leq 2\text{OPT} + 2\alpha(s - s_2)$
 - 3 $s_1 < s < s_2$ (thus, for instance, $c(C_2) < 2\text{OPT}$).

Combining the two (partial) vertex covers

- Since s is the non-coverage requirement, the solution C_1 (that leaves $s_1 < s$ edges uncovered) is feasible, but may be much costlier than $2OPT$; while C_2 is infeasible, but is actually upper bounded by $2OPT$.
- We will consider a *convex* combination of the two solutions C_1 and C_2 .

Convex Combination of C_1 and C_2

- Let $\beta = \frac{s-s_1}{s_2-s_1}$. Thus, $(1 - \beta) = \frac{s_2-s}{s_2-s_1}$.
- Thus it follows that

$$(1 - \beta)c(C_1) + \beta c(C_2) \leq 2\text{OPT}$$

- If $\beta < 1/2$ (i.e. $(1 - \beta) \geq 1/2$, then C_1 is the major component in this convex combination. and $c(C_1) \leq 4\text{OPT}$; since C_1 is actually feasible, we can output this as the solution.
- Suppose then that $\beta > 1/2$. In this case, we have to *augment* C_2 with (some part of C_1) to get a feasible solution of cost $\leq 5\text{OPT}$.

Convex Combination of C_1 and C_2

- Why did we not face the earlier issue of *high cost* vertices?
- Well, that figures in the last mentioned *augmentation* of C_2 by some part of C_1 . We will have to assume here that every vertex involved has cost $\leq \text{OPT}/2$ (or something to that effect).
- Optimizing the parameters in the above gives a $8/3$ -factor approximation for PVC.

- Konemann, Parekh and Segev (2006) showed that given a **suitable** primal dual r -factor algorithm for the prize collecting variant of a covering problem as a *blackbox*, one can design a $4r/3$ -factor algorithm for the partial covering variant. (This is what gave the $8/3$ above).
- Mestre (2008) showed that this extra factor of $4/3$ is inherent, whenever one uses such a primal dual algorithm as a blackbox.
- Thus, for instance, for the k -MST problem, Garg (1996) showed a 3-factor approximation, while the underlying prize collecting Steiner tree problem has a 2-factor approximation. It required breaking the blackbox and dealing with intricate details of the primal dual algorithm for the prize collecting Steiner tree problem in order to bring the factor down to 2 (Garg, 2001).



- In this talk, we did not mention the **Local Ratio** paradigm for designing approximation algorithms. The local ratio paradigm was first described by Bar-Yehuda and Even in 1982. Algorithms for various problems were designed using the local ratio paradigm (for instance, for Feedback Vertex Set, Job Interval Selection Problem, etc.).
- The paradigm essentially reduces a *weighted* problem to the *unweighted* version. Thus, it involves changing the weights (in case of Vertex Cover, the weights c_i) of an instance, to derive a new (simpler) instance.
- It was somewhat mysterious that problems that had algorithms in the primal dual schema turned out to have local ratio algorithms and vice versa.
- In 2005, Bar-Yehuda and Rawitz proved the formal equivalence of the two techniques.

- While Primal Dual algorithms work with an underlying LP and its dual, they do not require to actually solve the LP. Thus, most of such algorithms are rather fast, and are combinatorial in nature.
- Comparison with other paradigms: Iterative Rounding vs. Primal Dual. The two techniques showed up a sharp contrast for the MBDST problem: primal dual only gave a $O(\log n)$ additive approximation, whereas iterative rounding gave the optimal $+1$ approximation (Singh, Lau).
- On the other hand, for problems like Job Interval Selection Problem, iterative rounding does not give much.

Thanks

Thanks!

For Further Reading I

-  A. Author.
Handbook of Everything.
Some Press, 1990.
-  S. Someone.
On this and that.
Journal of This and That, 2(1):50–100, 2000.