Gurobi Seminar 4
Vertex Packing

Demetrios Papazaharias

UB INFORMS Student Chapter
Department of Industrial & Systems Engineering
University at Buffalo, SUNY

November 4, 2019

Welcome!

- Welcome to the fourth installment of our Gurobi series!
- In this workshop we will cover the vertex packing problem
- At the end of this workshop you will know how to implement user cuts in Gurobi
- Materials for this workshop can be found here:

  `github.com/Dpapazaharias1/UB-INFORMS-Gurobi-Seminar`

Topics to Cover

1. A Brief Introduction to Integer Programming

2. Vertex Packing

3. User Cuts in Gurobi

Contents

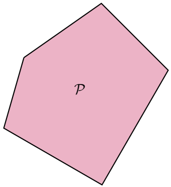
Integer Programming

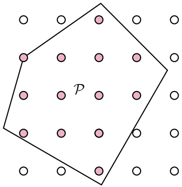
- So far we have covered linear programs where $\mathbf{x} \in \mathbb{R}^n_+$.
- We will now consider the problems where some or all of the variables are restricted to be integers
- Restricting $\mathbf{x} \in \mathbb{Z}^n_+$ or $\mathbf{x} \in \{0, 1\}^n$ makes solving the program considerably harder
- In fact, finding a feasible solution to a 0-1 integer program is one of Karp's 21 NP-Complete problems

Why is IP hard?

$$\begin{aligned} \min \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{b} \quad \text{(LP)} \\ & \mathbf{x} \geq 0 \end{aligned}$$

$$\begin{aligned} \min \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{b} \quad \text{(IP)} \\ & \mathbf{x} \in \mathbb{Z}^n_+ \end{aligned}$$





- (LP) - linear objective, convex feasible region: local search finds global optimum
- Not all extreme points of $\mathcal{P}$ are feasible in (IP)
- Except for very special cases, Simplex cannot solve (IP)

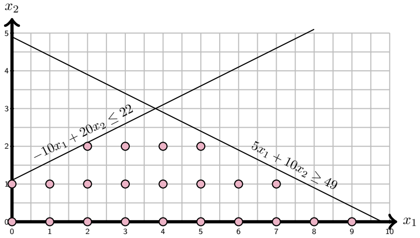
How to solve IP?

- **Branch and Bound**: Partitions the feasible region into subdivisions on non-integer $\mathbf{x}$-values
- **Cutting Planes**: Works with a single LP, refining the quality of the relaxation by adding constraints until an integer solution is found.

Branch-and-Bound

Consider the IP,

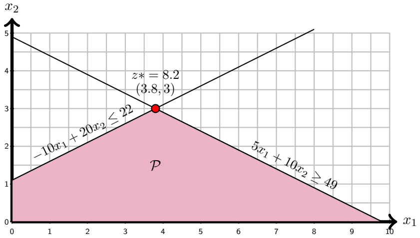
$$\begin{aligned} \max \quad & -x_1 + 4x_2 \\ \text{s.t.} \quad & -10x_1 + 20x_2 \leq 22 \\ & 5x_1 + 10x_2 \leq 49 \\ & x_1, x_2 \in \mathbb{Z}_+ \end{aligned}$$



Branch-and-Bound

Consider the IP,

$$\begin{aligned} \max \quad & -x_1 + 4x_2 \\ \text{s.t.} \quad & -10x_1 + 20x_2 \leq 22 \\ & 5x_1 + 10x_2 \leq 49 \\ & x_1, x_2 \geq 0 \end{aligned}$$

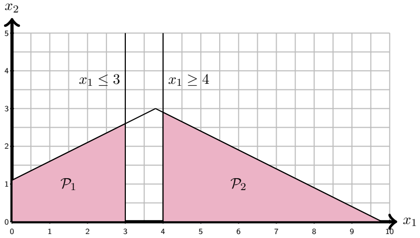


- Solve the LP Relaxation

Branch-and-Bound

Consider the IP,

$$
\begin{aligned}
\max \quad & -x_1 + 4x_2 \\
\text{s.t.} \quad & -10x_1 + 20x_2 \leq 22 \\
& 5x_1 + 10x_2 \leq 49 \\
& x_1, x_2 \geq 0
\end{aligned}
$$

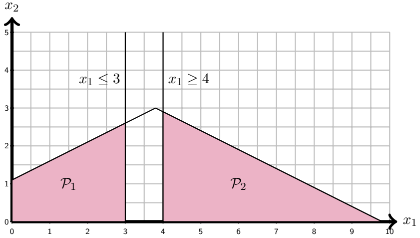


- Solve the LP Relaxation
- Since x_1 is fractional we partition $\mathcal{P}$ into separate subregions $\mathcal{P}_1, \mathcal{P}_2$ by adding $x_1 \leq 3$ and $x_1 \geq 4$.

Branch-and-Bound

Consider the IP,

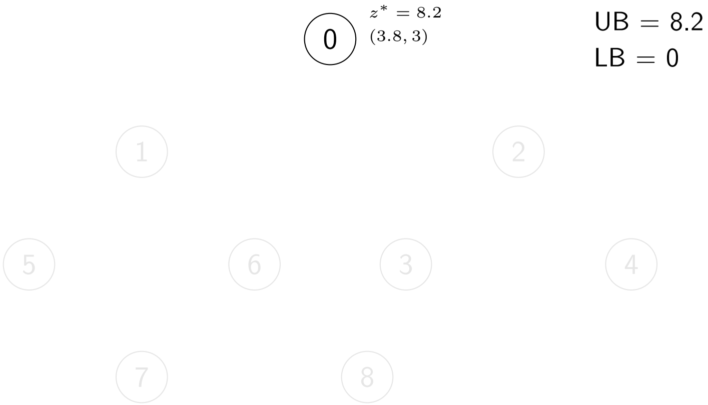
$$\begin{aligned} \max \quad & -x_1 + 4x_2 \\ \text{s.t.} \quad & -10x_1 + 20x_2 \leq 22 \\ & 5x_1 + 10x_2 \leq 49 \\ & x_1, x_2 \geq 0 \end{aligned}$$



- Solve the LP Relaxation
- Since x_1 is fractional we partition $\mathcal{P}$ into separate subregions $\mathcal{P}_1, \mathcal{P}_2$ by adding $x_1 \leq 3$ and $x_1 \geq 4$.
- Solve the LP over $\mathcal{P}_1$ and $\mathcal{P}_2$, continuing this procedure and keeping track of the best integer solution

Branch and Bound

We can represent the Branch and Bound procedure in a tree

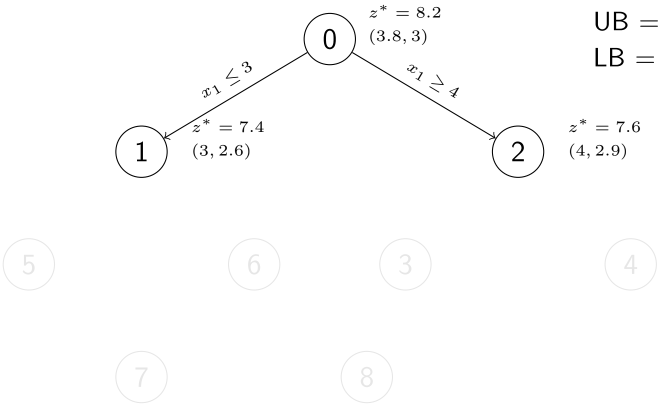


Branch and Bound

We can represent the Branch and Bound procedure in a tree

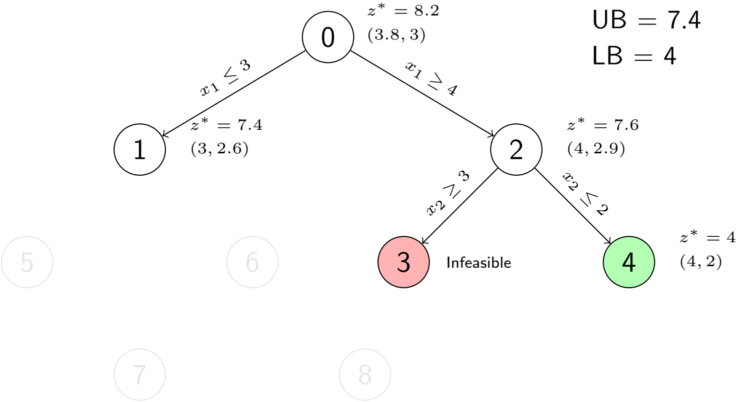
UB = 7.6

LB = 0



Branch and Bound

We can represent the Branch and Bound procedure in a tree

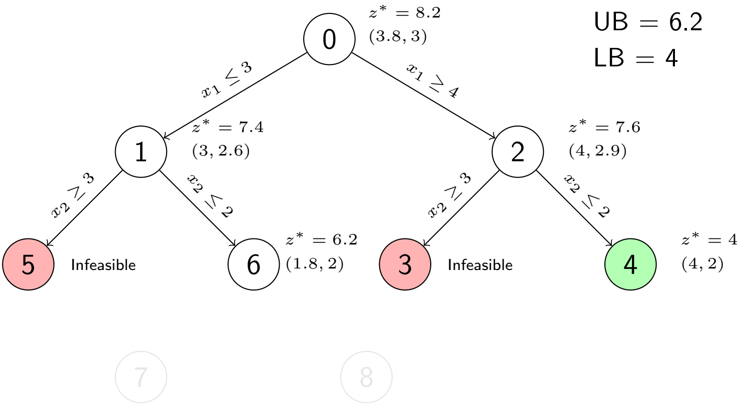


UB = 7.4

LB = 4

Branch and Bound

We can represent the Branch and Bound procedure in a tree

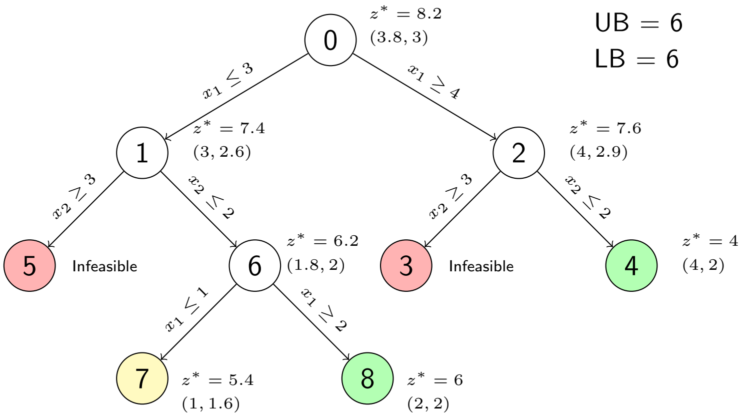


UB $= 6.2$

LB $= 4$

Branch and Bound

We can represent the Branch and Bound procedure in a tree



Cutting Planes

$$\begin{aligned} \max \quad & -x_1 + 4x_2 \\ \text{s.t.} \quad & -10x_1 + 20x_2 \leq 22 \\ & 5x_1 + 10x_2 \leq 49 \\ & x_1, x_2 \in \mathbb{Z}_+ \end{aligned}$$

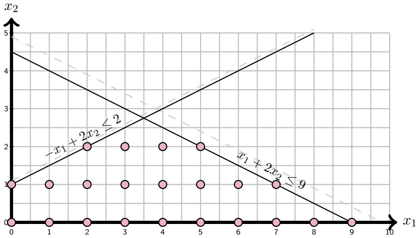
- Consider the first constraint. We can divide the inequality by 10 without changing the feasible region

$$-10x_1 + 20x_2 \leq 22 \rightarrow -x_1 + 2x_2 \leq 2.2$$

- Since $x_1, x_2 \in \mathbb{Z}_+$ with integer coefficients the largest feasible value $-x_1 + 2x_2$ can take is 2
- Thus, an improved version of this inequality is $-x_1 + 2x_2 \leq 2$
- Similarly for the second constraint $x_1 + 2x_2 \leq 9$

Cutting Planes

- Consider our feasible region:



- We can improve our formulation even further. Consider the sum of our two constraints

$$4x_2 \leq 11$$

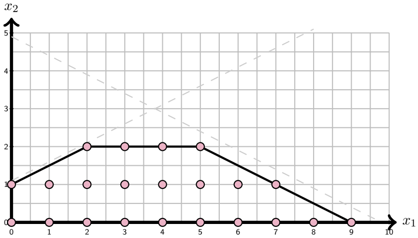
- Using the same procedure we can obtain the inequality:

$$x_2 \leq 2$$

- Every extreme point of the new polytope is an integer solution

Cutting Planes

- Consider our feasible region:



- We can improve our formulation even further. Consider the sum of our two constraints

$$4x_2 \leq 11$$

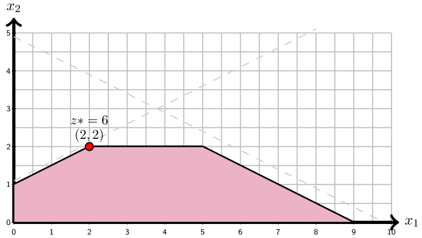
- Using the same procedure we can obtain the inequality:

$$x_2 \leq 2$$

- Every extreme point of the new polytope is an integer solution

Cutting Planes

- Consider our feasible region:



- We can improve our formulation even further. Consider the sum of our two constraints

$$4x_2 \leq 11$$

- Using the same procedure we can obtain the inequality:

$$x_2 \leq 2$$

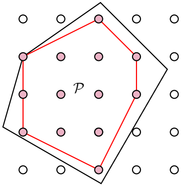
- Every extreme point of the new polytope is an integer solution

Branch and Cut

- In this workshop we will implement our own branch and cut scheme to solve vertex packing
- Simply put, **Branch and Cut** is an adaptation of branch and bound where cutting planes are added at each node to improve the LP relaxation
- We will first introduce some key definitions related to cutting planes

Terminology: Convex Hull

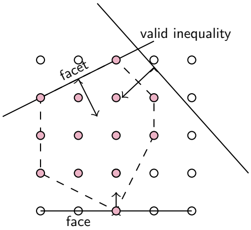
- The **convex hull** is the smallest polytope enclosing all feasible points



- The extreme points of conv($\mathcal{P}$) are all integer solutions
- Given the description (constraints) of the conv($\mathcal{P}$), we can use the simplex method to solve (IP)

Valid Inequalities, Faces and Facets

- Consider the set of integer points

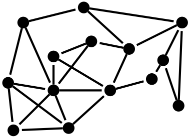


- A **valid inequality** is any constraint that does not eliminate a feasible integer solution.
- A valid inequality such that at least one feasible point satisfies the constraint at equality is called a **face** of $\mathcal{P}$.
- A face F of $\mathcal{P}$ is called a **facet** if $\dim(F) = \dim(\mathcal{P}) - 1$

Contents

Terminology: Graph

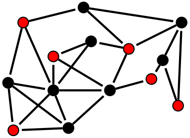
- A **Graph** denoted $G = (V, E)$, where V is a finite set of vertices and E is a finite set of edges.
- Each edge is an unordered associated between pairs of vertices
- Edges in in graphs, as opposed to digraphs, are undirected



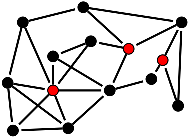
Problem Description

- Input is an undirected graph $G = (V, E)$
- A vertex packing or independent set, is a subset $S \subseteq V$ such that no two vertices are connected by an edge
- A maximal vertex packing I is a vertex packing such that when adding any other $i \in V \setminus I$ to I, it is no longer a vertex packing
- A maximum vertex packing is a vertex packing of largest possible size for G

Example



Maximum Vertex Packing



Maximal Vertex Packing

Formulation

$$x_i = \begin{cases} 1 & \text{if vertex } i \text{ is in the packing} \\ 0 & \text{otherwise} \end{cases}$$

$$\max \quad \sum_{i \in V} x_i \tag{1}$$

$$\text{s.t.} \quad x_i + x_j \leq 1 \quad \{i, j\} \in E \tag{2}$$

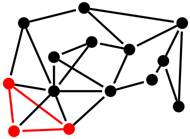
$$x_i \in \{0, 1\} \quad i \in V \tag{3}$$

Constraints (2)-(3) provide a valid formulation for the vertex packing problem, however it is not the convex hull.

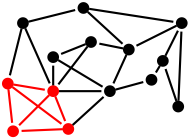
Cliques

- A **clique** $C \subseteq V$, is a subset of vertices such that for all $i, j \in C$, $\{i, j\} \in E$

- A clique C is maximal if by adding any $i \in V \setminus C$ to the clique, then C is no longer a clique



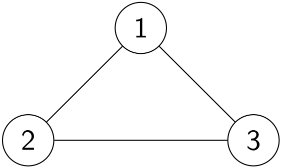
Clique, not maximal



Maximal Clique

Motivation

- Consider a maximal clique of size 3 and associated constraint set in the VP formulation



$$\begin{aligned} x_1 + x_2 &\leq 1 \\ x_1 + x_3 &\leq 1 \quad (VP) \\ x_2 + x_3 &\leq 1 \end{aligned}$$

- Every solution that satisfies $x_1 + x_2 + x_3 \leq 1$ satisfies (VP)

- Consider $\mathbf{x} = (0.5, 0.5, 0.5)$. Does not violate any constraint in (VP), however violates the maximal clique cut.

- Thus, $x_1 + x_2 + x_3 \leq 1$ is a stronger constraint than (VP)

Clique Cuts

- For a maximal clique C, the constraint:

$$\sum_{i \in C} x_i \leq 1$$

is facet defining for vertex packing

- We cannot add these inequalities right away, as there may be an exponential number of maximal cliques

- We will begin solving the IP with none of the maximal clique constraints and add them to improve each LP relaxation

Separation Algorithm

A **separation algorithm** is a procedure that separates a solution from the feasible region.

- Since we cannot add all clique inequalities from the beginning, we will create a procedure to generate these cuts
- Based on the current LP relaxation solution, we will find a clique inequality that is violated by the solution and add it to the formulation
- For VP, we will use maximum weighted clique problem to find the most violated constraints (optimal separation)
- Reminder: Clique inequalities are not essential in producing a valid formulation. They simply strengthen the upper bound

Maximum Weighted Clique

- Given an LP solution $\bar{\mathbf{x}}$, we find the most violated constraint
- We assign $\bar{x}_i$ as the "weight" of vertex i and solve MWC

$\bar{x}_i$ – the value of x_i in the LP relaxation

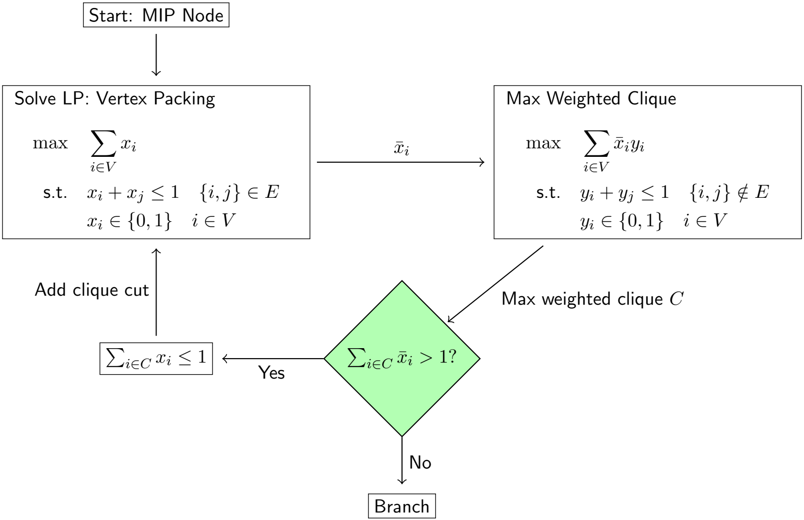
$$y_i = \begin{cases} 1 & \text{if } i \text{ is in the maximum weighted clique } C \\ 0 & \text{otherwise} \end{cases}$$

$$\max \quad \sum_{i \in V} \bar{x}_i y_i \tag{4}$$

$$\text{s.t.} \quad y_i + y_j \leq 1 \quad \{i, j\} \notin E \tag{5}$$

$$y_i \in \{0, 1\} \quad i \in V \tag{6}$$

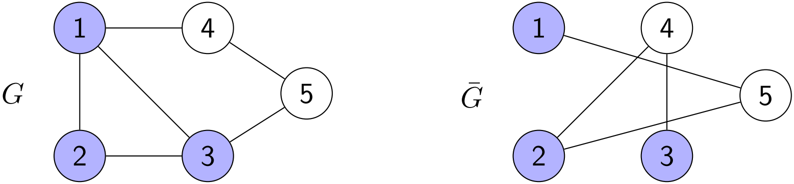
Procedure



Start: MIP Node
Solve LP: Vertex Packing
$$\max \sum_{i \in V} x_i$$
$$\text{s.t.} \quad x_i + x_j \leq 1 \quad \{i,j\} \in E$$
$$x_i \in \{0,1\} \quad i \in V$$
$\bar{x}_i$
Max Weighted Clique
$$\max \sum_{i \in V} \bar{x}_i y_i$$
$$\text{s.t.} \quad y_i + y_j \leq 1 \quad \{i,j\} \notin E$$
$$y_i \in \{0,1\} \quad i \in V$$
Max weighted clique C
$\sum_{i \in C} \bar{x}_i > 1?$
Yes
$\sum_{i \in C} x_i \leq 1$
Add clique cut
No
Branch

Caveat

- Max clique is just as hard as vertex packing
- Consider $\bar{G}$, the complement graph of G
- Vertices i and j are adjacent in $\bar{G}$ if and only if $\{i, j\} \notin E$



- A maximum clique in G is a maximum vertex packing in $\bar{G}$

Contents

Callbacks

A **callback** is a user function that is called periodically by the optimizer in order to allow the user to query or modify the state of the optimization.

- Pass a function that takes two arguments (`model` and `where`) to Model.optimize and your function will be called during optimization
- The argument `where` indicates from which point in the optimization process the callback is used
- MIP callbacks can be used to retrieve solutions from a current node and add any violating constraints to the model

User Cuts

- User cuts are ordinary constraints that are used to improve the continuous relaxation, but do not rule out any feasible integer solution.

- The essential Gurobi attributes for user cuts:

Attribute	User Cuts
parameters	Model.Params.PreCrush
where	GRB.Callback.MIPNODE
current solution	Model.cbGetNodeRel()
add constraint	Model.cbCut()

- More information about callback codes can be found here `https://www.gurobi.com/documentation/8.1/refman/py_callbacks.html`

Passing Data to Callbacks

Since callback functions can have only two arguments, we need a way to pass data to our callback functions.

- We can do this through a Model object
- For example, if your program contains the statement `model._data = 1` before the optimization begins, your callback function can query the value of `model._data`
- **IMPORTANT: The name of the user data field must begin with an underscore!**