Integer Programming and Branch and Bound

Sommer Gentry

November 24th, 2003

Adapted from slides
by Eric Feron and Brian Williams, 16.410, 2002.

Integer Programming (IP)

- What is it?
- Expressing decisions with IP
  - Exclusion between choices
  - Exclusion between constraints
- Solutions through branch and bound
  - Characteristics
  - Solving Binary IPs
  - Solving Mixed IPs and LPs

Integer Programs

LP: Maximize $3x_1 + 4x_2$

Subject to:

$x_1 \leq 4$

$2x_2 \leq 12$

$3x_1 + 2x_2 \leq 18$

$x_1, x_2 \geq 0$

IP: Maximize $3x_1 + 4x_2$

Subject to:

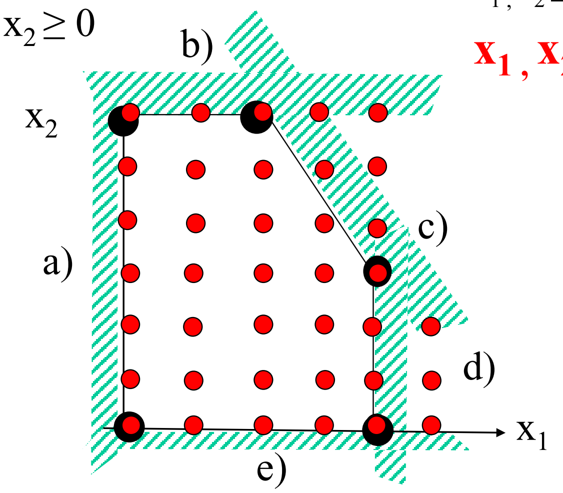
$x_1 \leq 4$

$2x_2 \leq 12$

$3x_1 + 2x_2 \leq 18$

$x_1, x_2 \geq 0$

x_1, x_2 integers



Integer Programming

Integer programs are LPs where some variables are integers

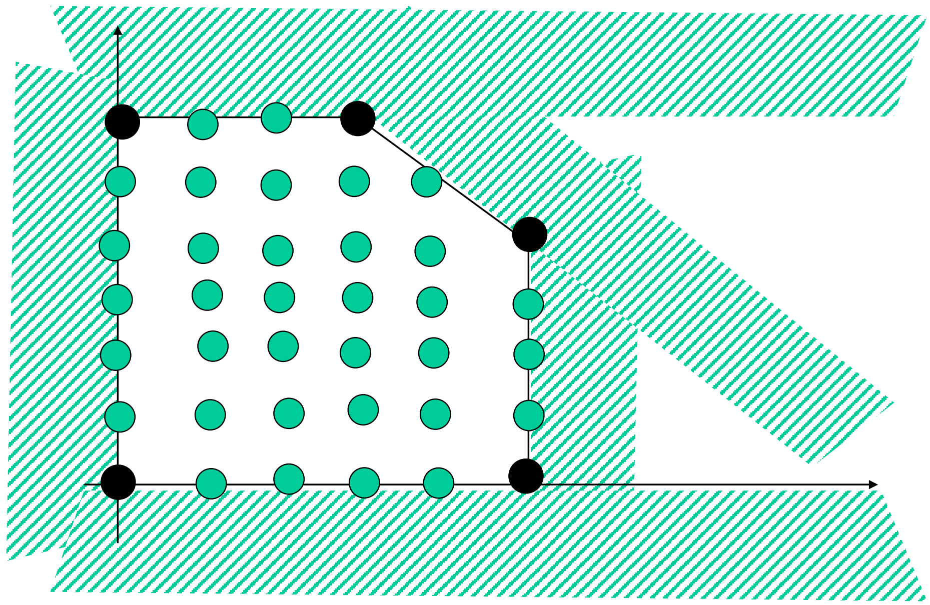
<u>Why Integer programming?</u>

1. Some variables are not real-valued:
   - Boeing only sells complete planes, not fractions.

2. Fractional LP solutions poorly approximate integer solutions:
   - For Boeing Aircraft Co., producing 4 versus 4.5 airplanes results in radically different profits.

<u>Often a mix is desired</u> of integer and non-integer variables (MILP).

Graphical representation of IP



Integer Programming (IP)

- What is it?
- Making decisions with IP
  - Exclusion between choices
  - Exclusion between constraints
- Solutions through branch and bound
  - Characteristics
  - Solving Binary IPs
  - Solving Mixed IPs and LPs

Integer Programming for Decision Making

"Yes or no" decisions encoded with binary variables:

x_j → 1 if decision is yes

x_j → 0 if decision is no.

Binary Integer Programming (BIP):

• Binary variables + linear constraints.

Binary Integer Programming Example: Cal Aircraft Manufacturing Company

Problem:

1. Cal wants to expand:
   - Build new factory in Los Angeles, San Francisco, or both.
   - Build new warehouse (only one).
   - Warehouse must be built close to city of a new factory.
2. Available capital: $10,000,000
3. Cal wants to maximize "total net present value" (profitability vs. time value of money)

		NPV	Price
1	Build a factory in L.A.?	$9m	$6m
2	Build a factory in S.F.?	$5m	$3m
3	Build a warehouse in L.A.?	$6m	$5m
4	Build a warehouse in S.F.?	$4m	$2m

Binary Integer Programming Example: Cal Aircraft Manufacturing Company

What decisions are to be made?

1. Build factory in LA
2. Build factory in SFO
3. Build warehouse in LA
4. Build warehouse in SFO

Introduce 4 binary variables x_i = 1 if decision i is yes; 0 if decision i is no

Binary Integer Programming Example: Cal Aircraft Manufacturing Company

1. Cal wants to expand **(TBD)**
2. Available capital: $10,000,000
3. Cal wants to maximize "total net present value" (profitability vs. time value of money)

		NPV	Price
1	Build a factory in L.A.?	$9m	$6m
2	Build a factory in S.F.?	$5m	$3m
3	Build a warehouse in L.A.?	$6m	$5m
4	Build a warehouse in S.F.?	$4m	$2m

What is the objective?

- Maximize NPV:

$$Z = 9x_1 + 5x_2 + 6x_3 + 4x_4$$

Remaining constraints?

- Don't go beyond means:

$$6x_1 + 3x_2 + 5x_3 + 2x_4 \leq 10$$

Binary Integer Programming Example: Cal Aircraft Manufacturing Company

LA factory(x_1), SFO factory(x_2), LA warehouse(x_3),SFO warehouse (x_4)

- Build new factory in Los Angeles, San Francisco, or both.
- Build new warehouse (only one).
- Warehouse must be built close to city of a new factory.

What are the constraints between decisions?

1. Only one warehouse:

   x_3 exclusive-or x_3

2. Warehouse in LA only if Factory is in LA:

   x_3 implies x_1

3. Warehouse in SFO only if Factory is in SFO:

   x_4 implies x_2

Encoding Choices: An Example

- Mutually exclusive choices
  - Example: at most 2 decisions in a group can be yes:

  LP Encoding:

  $x_1 + \ldots + x_k \leq 2.$

Binary Integer Programming Example:
Cal Aircraft Manufacturing Company

LA factory(x1), SFO factory(x2), LA warehouse(x3),SFO warehouse (x4)

- Build new factory in Los Angeles, San Francisco, or both.
- Build new warehouse (only one).
- Warehouse must be built close to city of a new factory.

What are the constraints between decisions?

1. Only one warehouse:

   x_3 exclusive-or x_4

   $x_3 + x_4 \leq 1$

2. Warehouse in LA only if Factory is in LA:

   x_3 implies x_1

   $x_3 - x_1 \leq 0$

3. Warehouse in SFO only if Factory is in SFO:

   x_4 implies x_2

   $x_4 - x_2 \leq 0$

Binary Integer Programming Example:
Cal Aircraft Manufacturing Company

Complete binary integer program:

Maximize $Z = 9x_1 + 5x_2 + 6x_3 + 4x_4$

Subject to: $6x_1 + 3x_2 + 5x_3 + 2x_4 \leq 10$

$x_3 + x_4 \leq 1$

$x_3 - x_1 \leq 0$

$x_4 - x_2 \leq 0$

$x_j \leq 1$

$x_j = \{0,1\}, j=1,2,3,4$

$x_j \geq 0$

Integer Programming (IP)

- What is it?
- Making decisions with IP
  - Exclusion between choices
  - Exclusion between constraints
- Solutions through branch and bound
  - Characteristics
  - Solving Binary IPs
  - Solving Mixed IPs and LPs

Cooperative Path Planning MILP Encoding: Constraints

- Min J_T — Receding Horizon Fuel Cost Fn
- $s_{ij} \leq w_{ij}$, etc. — State Space Constraints
- $\mathbf{s}_i+1 = \mathbf{A}\mathbf{s}_i + \mathbf{B}\mathbf{u}_i$ — State Evolution Equation
-

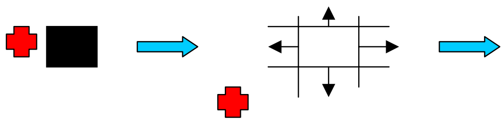
Obstacle Avoidance

Collision Avoidance

Obstacles

How do we encode?

- Each obstacle-vehicle pair represents a disjunctive constraint:



Red Vehicle is above obstacle OR
Red Vehicle is below obstacle OR
Red Vehicle is left of obstacle OR
Red Vehicle is right of obstacle

- Each disjunct is an inequality
  - if xR, yR are co-ordinates of red vehicle, inequalities above might be:
  - xR < 3
  - yR > 4, etc.

- Constraints are not limited to rectangular obstacles or obstacles oriented a particular way
  - (inequalities might include both co-ordinates)

Encoding Exclusion Constraints

- Mutually exclusive constraints
  - Example:

    Either $[\ 3x_1 + 2x_2 \leq 18\ (x_1, x_2 \text{ real})\ ]$

    Or $[\ x + 4x \leq 16].$
  - LP Encoding:
    - Use Big M to turn-off constraint:

      Either:

      $3x_1 + 2x_2 \leq 18$

      and $x_1 + 4x_2 \leq 16 + M$ (and M is very BIG)

      Or:

      $3x_1 + 2x_2 \leq 18 + M$

      and $x_1 + 6x_2 \leq 16$
    - Use binary y to decide which constraint to turn off:

      $3x1 + 2x2 \leq 18 + y\,M$

      $x1 + 2x2 \leq 16 + (1-y)M$

      $y \in \{0,1\}$

Cooperative Path Planning MILP Encoding: Constraints

- Min J_T — Receding Horizon Fuel Cost Fn
- $s_{ij} \leq w_{ij}$, etc. — State Space Constraints
- $\mathbf{s}_i+1 = \mathbf{A}\mathbf{s}_i + \mathbf{B}\mathbf{u}_i$ — State Evolution Equation
- Obstacle Avoidance:
  $x_i \leq x_{min} + My_{i1}$
  $-x_i \leq -x_{max} + My_{i2}$
  $y_i \leq y_{min} + My_{i3}$
  $-y_i \leq -y_{max} + My_{i4}$
  $\Sigma\, y_{ik} \leq 3$ — At least one enabled
- Similar constraints for Collision Avoidance (for all pairs of vehicles)

Encoding Exclusion Constraints

- <u>K out of N constraints hold:</u>

  $f_1(x_1, x_2, \ldots x_n) \leq d_1$ OR

  :

  $f_N(x_1, x_2, \ldots, x_n) \leq d_N$

  where f_i are linear expressions

- <u>At most K of N hold:</u>

- LP Encoding:
  - Introduce y_i to deselect each constraint:
  - Constrain K of y_i to select constraints:

$$\sum_{i=1}^{N} y_i = N - K \qquad\qquad \sum_{i=1}^{N} y_i \leq N - K$$

  - Use Big M to turn-off constraint:

$$f1(x_1, \text{---}, x_n) \leq d_1 + My_1$$
$$\vdots$$
$$f_N(x_1, \text{---}, x_n) \leq d_N + My_N$$

Encoding Finite Domain Functions

- <u>Function has one out of n possible values</u>:

  $a_1x_1 + \ldots a_n x_n = [d_1 \text{ or } d_2 \ \ldots \text{ or } d_p]$

  - <u>LP Encoding:</u>

    $y_i \in \{0,1\} \quad i=1,2,\ldots p$

    $(\Sigma y_i) = 1$

    $a_1x_1 + \ldots a_n x_n = (\Sigma_\iota \ d_i y_i)$

<u>Encoding Constraints</u>

- <u>Fixed – charge problem:</u>

$f_i(x_j) = \ | \ k_j + c_j x_j \text{ if } x_j > 0$

$\qquad\quad | \ 0 \text{ if } x_j = 0$

Minimizing costs:

Minimizing $z = f_1(x_1) + \text{---} + f_n(x_n)$

Yes or no decisions: should each of the activities be undertaken?

<u>Introduce auxiliary variables</u>:

$y_1, \ldots, y_n = 0,1$

$y = 1$ if $x > 0$

0 if $x = 0$

$$Z = \sum_{i=1}^{n} c_i x_i + k_i y_i$$

Which can be written as a linear constraint using big M:

$x \leq yM$

Integer Programming (IP)

- What is it?
- Making decisions with IP
  - – Exclusion between choices
  - – Exclusion between constraints
- Solutions through Branch and Bound
  - – IP Characteristics
  - – Solving Binary IPs
  - – Solving Mixed IPs and LPs

Naïve approach: rounding

- A simple way to solve an IP might be to round the non-integer solution to the nearest feasible integer. In practice, try it!
- In theory, this idea makes no sense at all. Consider

$$\text{Maximize } Z = 21x_1 + 11x_2$$
$$\text{Subject to } 7x_1 + 4x_2 \le 13$$
$$x_1 \ge 0, x_2 \ge 0,\ x_1, x_2 \text{ integer}$$

- Optimal solution x* is (13/7,0)

Naïve approach: rounding

- The rounded solution is x*=(2,0). But this is not even feasible!
- By enumeration, feasible x are (0,0) (0,1) (0,2) (0,3) (1,1) (1,0)
- The optimal integer solution x* is (0,3) [value = 33] which is nowhere near (13/7,0).
- In general, just finding a nearby feasible point is computationally challenging

Solving Integer Programs: Characteristics

- Fewer feasible solutions than LPs.
- Worst-case exponential in # of variables.
- Commercial software:
  - Cplex

Solving Integer Programs

- Branch and Bound
  - Binary Integer Programs
  - Integer Programs
  - Mixed Integer (Real) Programs
- Cutting Planes

Branch and Bound

Problem: Optimize f(x) subject to $A(x) \geq 0$, $x \in D$

B & B - an instance of Divide & Conquer:

I. Bound D's solution and compare to alternatives.

1) Bound solution to D quickly.
   - Perform quick check by relaxing hard part of problem.
   - ➔ Relax integer constraints. Relaxation is LP.
2) Use bound to "fathom" D if possible.
   - **If** relaxed solution is integer,
     Then keep soln if best found to date ("incumbent"), delete D_i
   - **If** relaxed solution is worse than incumbent, **Then** delete D_i.
   - **If** no feasible solution, **Then** delete D_i.

II. Otherwise Branch to smaller subproblems

1) Partition D into subproblems $D_1 \ldots D_n$
2) Apply B&B to all subproblems.

B&B for Binary Integer Programs (BIPs)

Problem: Optimize f(x) st A(x) ≥ 0, $x_i \in \{0,1\}$, $x \in D$

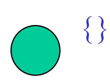
Domain D_i encoding:

- partial assignment to x,
  - {x1 = 1, x2 = 0, …}

Branch Step:

1. Find variable x_j that is unassigned in D_i
2. Create two subproblems by splitting D_i:
   - $D_{i1} \equiv D_i \cup \{x_j = 1\}$
   - $D_{i0} \equiv D_i \cup \{x_j = 0\}$

Example: B&B for BIPs



Solve:

Max $Z = 9x_1 + 5x_2 + 6x_3 + 4x_4$

Subject to:

- $6x_1 + 3x_2 + 5x_3 + 2x_4 \leq 10$
- $x_3 + x_4 \leq 1$
- $-x_1 + x_3 \leq 0$
- $-x_2 + x_4 \leq 0$
- $x_i \leq 1$, $x_i \geq 0$, x_i **integer**

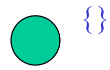
- Initialize

Queue: {}

Incumbent: none

Best cost Z*: - inf

Example: B&B for BIPs



Solve:

Max $Z = 9x_1 + 5x_2 + 6x_3 + 4x_4$

Subject to:

- $6x_1 + 3x_2 + 5x_3 + 2x_4 \leq 10$
- $x_3 + x_4 \leq 1$
- $-x_1 + x_3 \leq 0$
- $-x_2 + x_4 \leq 0$
- $x_i \leq 1, x_i \geq 0$, x_i **integer**

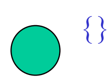
Queue: {}

Incumbent: none

Best cost Z*: - inf

- Dequeue {}

Example: B&B for BIPs



Solve:

Max $Z = 9x_1 + 5x_2 + 6x_3 + 4x_4$

Subject to:

- $6x_1 + 3x_2 + 5x_3 + 2x_4 \leq 10$
- $x_3 + x_4 \leq 1$
- $-x_1 + x_3 \leq 0$
- $-x_2 + x_4 \leq 0$
- ~~$x_i \leq 1, x_i \geq 0$, x_i **integer**~~

$Z = 16.5, x = <0.8333,1,0,1>$

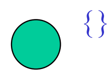
Queue:

Incumbent: none

Best cost Z*: - inf

- Bound {}
  - Constrain x_i by {}
  - Relax to LP
  - Solve

Example: B&B for BIPs



Solve:

Max $Z = 9x_1 + 5x_2 + 6x_3 + 4x_4$

Subject to:

- $6x_1 + 3x_2 + 5x_3 + 2x_4 \le 10$
- $x_3 + x_4 \le 1$
- $-x_1 + x_3 \le 0$
- $-x_2 + x_4 \le 0$
- ~~$x_i \le 1, x_i \ge 0, x_i$ **integer**~~

Z = 16.5, x = <0.8333,1,0,1>

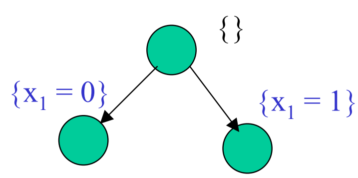
Queue:

Incumbent: none

Best cost Z*: - inf

- Try to fathom:
  - infeasible?
  - worse than incumbent?
  - integer solution?

Example: B&B for BIPs



Solve:

Max $Z = 9x_1 + 5x_2 + 6x_3 + 4x_4$

Subject to:

- $6x_1 + 3x_2 + 5x_3 + 2x_4 \le 10$
- $x_3 + x_4 \le 1$
- $-x_1 + x_3 \le 0$
- $-x_2 + x_4 \le 0$
- ~~$x_i \le 1, x_i \ge 0, x_i$ **integer**~~

Z = 16.5, x = <0.8333,1,0,1>

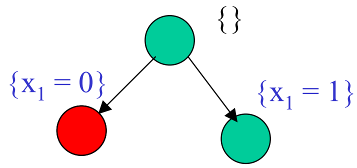
Queue: $\{x_1 = 0\}\{x_1 = 1\}$

Incumbent: none

Best cost Z*: - inf

- Branch:
  - select unassigned x_i
    - pick non-integer
  - Split on x_i

Example: B&B for BIPs



Solve:

Max $Z = 9x_1 + 5x_2 + 6x_3 + 4x_4$

Subject to:

- $6x_1 + 3x_2 + 5x_3 + 2x_4 \leq 10$
- $x_3 + x_4 \leq 1$
- $-x_1 + x_3 \leq 0$
- $-x_2 + x_4 \leq 0$
- ~~$x_i \leq 1, x_i \geq 0$, x_i **integer**~~

Z = **x =**

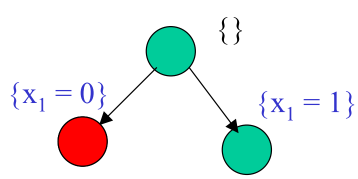
Queue: ~~$\{x_1 = 0\}$~~ $\{x_1 = 1\}$

Incumbent: none

Best cost Z*: - inf

- dequeue:
  - depth first or
  - best first

Example: B&B for BIPs



Solve:

Max $Z = 9x_1 + 5x_2 + 6x_3 + 4x_4$

Subject to:

- $6x_1 + 3x_2 + 5x_3 + 2x_4 \leq 10$
- $x_3 + x_4 \leq 1$
- $-x_1 + x_3 \leq 0$
- $-x_2 + x_4 \leq 0$
- ~~$x_i \leq 1, x_i \geq 0$, x_i **integer**~~

Z = **x =**

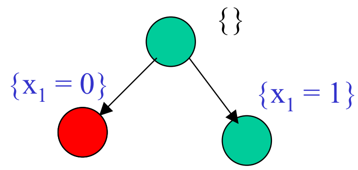
Queue: $\{x_1 = 1\}$

Incumbent: none

Best cost Z*: - inf

- Bound $\{x_1 = 0\}$
  - constrain x by $\{x_1 = 0\}$

Example: B&B for BIPs



Solve:

Max $Z = 9\ 0 + 5x_2 + 6x_3 + 4x_4$

Subject to:

- $6\ 0 + 3x_2 + 5x_3 + 2x_4 \leq 10$
- $x_3 + x_4 \leq 1$
- $-0 + x_3 \leq 0$
- $-x_2 + x_4 \leq 0$
- ~~$x_i \leq 1$, $x_i \geq 0$, x_i **integer**~~

Z = **x =**

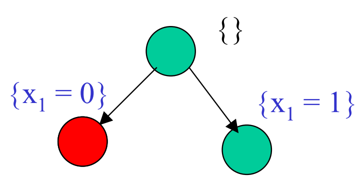
- Bound $\{x_1 = 0\}$
  - constrain x by $\{x_1 = 0\}$

Queue: $\{x_1 = 1\}$

Incumbent: none

Best cost Z*: - inf

Example: B&B for BIPs



Solve:

Max $Z = 5x_2 + 6x_3 + 4x_4$

Subject to:

$3x_2 + 5x_3 + 2x_4 \leq 10$

- $x_3 + x_4 \leq 1$

$+ x_3 \leq 0$

- $-x_2 + x_4 \leq 0$
- ~~$x_i \leq 1$, $x_i \geq 0$, x_i **integer**~~

Z = 9, x = <0,1,0,1>

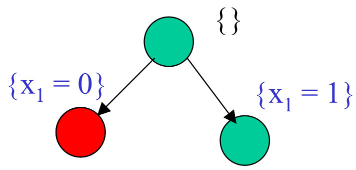
- Bound $\{x_1 = 0\}$
  - constrain x by $\{x_1 = 0\}$
  - relax to LP
  - solve

Queue: $\{x_1 = 1\}$

Incumbent: none

Best cost Z*: - inf

Example: B&B for BIPs



Solve:

Max Z = $5x_2 + 6x_3 + 4x_4$

Subject to:

$3x_2 + 5x_3 + 2x_4 \le 10$

- $x_3 + x_4 \le 1$

$+ x_3 \le 0$

- $-x_2 + x_4 \le 0$
- ~~$x_i \le 1, x_i \ge 0, x_i$ **integer**~~

Z = 9, x = <0,1,0,1>

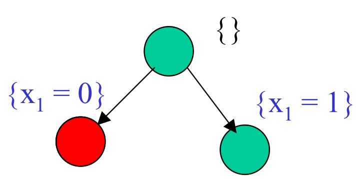
Queue: $\{x_1 = 1\}$

Incumbent: none

Best cost Z*: - inf

- Try to fathom:
  - infeasible?
  - worse than incumbent?
  - integer solution?

Example: B&B for BIPs



Solve:

Max Z = $5x_2 + 6x_3 + 4x_4$

Subject to:

$3x_2 + 5x_3 + 2x_4 \le 10$

- $x_3 + x_4 \le 1$

$+ x_3 \le 0$

- $-x_2 + x_4 \le 0$
- ~~$x_i \le 1, x_i \ge 0, x_i$ **integer**~~

Z = 9, x = <0,1,0,1>

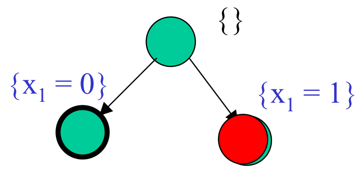
Queue: $\{x_1 = 1\}$

Incumbent: **x = <0,1,0,1>**

Best cost Z*: 9

- Try to fathom:
  - infeasible?
  - worse than incumbent?
  - integer solution?

Example: B&B for BIPs



Solve:

Max $Z = 9x_1 + 5x_2 + 6x_3 + 4x_4$

Subject to:

- $6x_1 + 3x_2 + 5x_3 + 2x_4 \leq 10$
- $x_3 + x_4 \leq 1$
- $-x_1 + x_3 \leq 0$
- $-x_2 + x_4 \leq 0$
- ~~$x_i \leq 1, x_i \geq 0, x_i$ **integer**~~

Z = **x =**

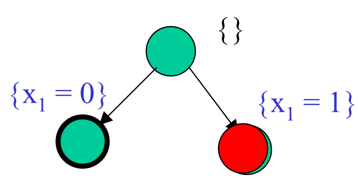
- Dequeue and bound

Queue: ~~{x_1 = 1}~~

Incumbent: **x = <0,1,0,1>**

Best cost Z*: 9

Example: B&B for BIPs



Solve:

Max $Z = 9 \quad + 5x_2 + 6x_3 + 4x_4$

Subject to:

- $6 \quad + 3x_2 + 5x_3 + 2x_4 \leq 10$
- $x_3 + x_4 \leq 1$
- $-1 \quad + x_3 \leq 0$
- $-x_2 + x_4 \leq 0$
- ~~$x_i \leq 1, x_i \geq 0, x_i$ **integer**~~

Z = 16,~~2~~, x = <1,.8,0,.8>

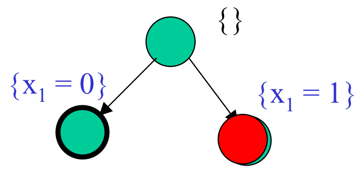
- Dequeue and bound

Queue:

Incumbent: **x = <0,1,0,1>**

Best cost Z*: 9

Example: B&B for BIPs



Solve:

Max $Z = 9 \quad + 5x_2 + 6x_3 + 4x_4$

Subject to:

- $6 \quad + 3x_2 + 5x_3 + 2x_4 \leq 10$
- $x_3 + x_4 \leq 1$
- $-1 + x_3 \leq 0$
- $-x_2 + x_4 \leq 0$
- ~~$x_i \leq 1, x_i \geq 0$, x_i **integer**~~

Z = 16.~~2~~, x = <1,.8,0,.8>

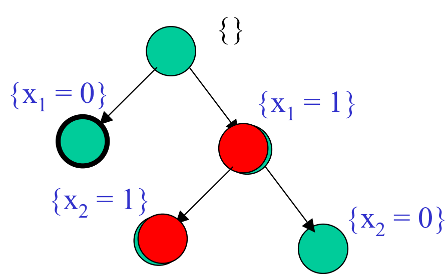
- Try to fathom:
  - infeasible?
  - worse than incumbent?
  - integer solution?

Queue:

Incumbent: **x = <0,1,0,1>**

Best cost Z*: 9

Example: B&B for BIPs



Solve:

Max $Z = 9 \quad + 5x_2 + 6x_3 + 4x_4$

Subject to:

- $6 \quad + 3x_2 + 5x_3 + 2x_4 \leq 10$
- $x_3 + x_4 \leq 1$
- $-1 + x_3 \leq 0$
- $-x_2 + x_4 \leq 0$
- ~~$x_i \leq 1, x_i \geq 0$, x_i **integer**~~

Z = 16.~~2~~, x = <1,.8,0,.8>

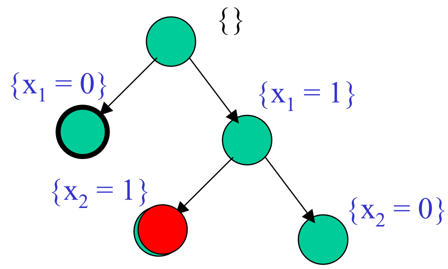
- Branch
- Dequeue

Queue: ~~$\{x_2 = 1\}$~~ $\{x_2 = 0\}$

Incumbent: **x = <0,1,0,1>**

Best cost Z*: 9

Example: B&B for BIPs



Solve:

Max $Z = 9 \quad + 5x_2 + 6x_3 + 4x_4$

Subject to:

- $6 \quad + 3x_2 + 5x_3 + 2x_4 \le 10$
- $x_3 + x_4 \le 1$
- $-1 + x_3 \le 0$
- $-x_2 + x_4 \le 0$
- ~~$x_i \le 1, x_i \ge 0, x_i$ **integer**~~

Z = **x =**

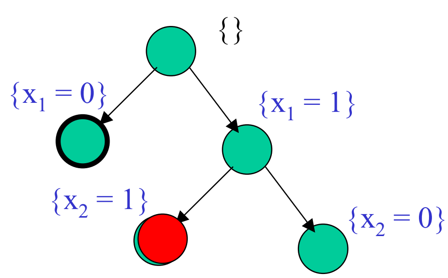
- Bound

Queue: $\{x_2 = 0\}$

Incumbent: **x = <0,1,0,1>**

Best cost Z*: 9

Example: B&B for BIPs



Solve:

Max $Z = 9 \quad + 5 \quad + 6x_3 + 4x_4$

Subject to:

- $6 \quad + 3 \quad + 5x_3 + 2x_4 \le 10$
- $x_3 + x_4 \le 1$
- $-1 + x_3 \le 0$
- $-1 + x_4 \le 0$
- ~~$x_i \le 1, x_i \ge 0, x_i$ **integer**~~

Z = 16, x = <1,1,0,.5>

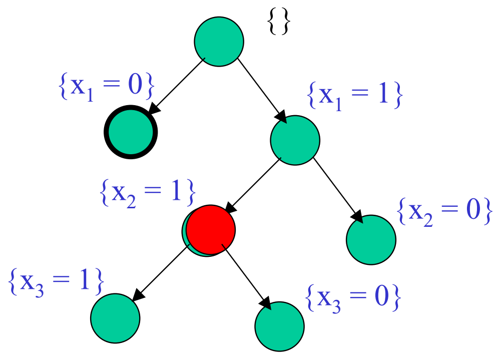
- Try to fathom:
  - infeasible?
  - worse than incumbent?
  - integer solution?

Queue: $\{x_2 = 0\}$

Incumbent: **x = <0,1,0,1>**

Best cost Z*: 9

Example: B&B for BIPs



Solve:

Max $Z = 9 \quad + 5 \quad + 6x_3 + 4x_4$

Subject to:

- $6 \quad + 3 \quad + 5x_3 + 2x_4 \leq 10$
- $x_3 + x_4 \leq 1$
- $-1 + x_3 \leq 0$
- $-1 + x_4 \leq 0$
- ~~$x_i \leq 1, x_i \geq 0, x_i$ **integer**~~

Z = 16, x = <1,1,0,.5>

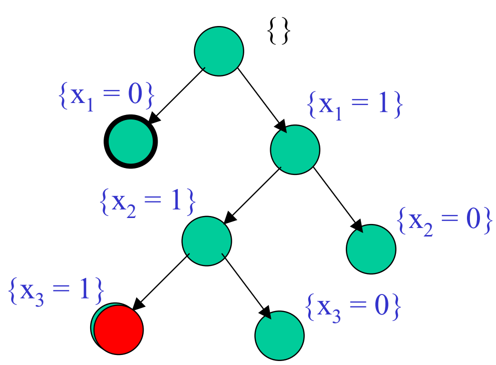
- Branch

Queue: $\{x_3 = 1\}$ $\{x_3 = 0\}$ $\{x_2 = 0\}$

Incumbent: **x = <0,1,0,1>**

Best cost Z*: 9

Example: B&B for BIPs



Solve:

Max $Z = 9 \quad + 5 \quad + 6x_3 + 4x_4$

Subject to:

- $6 \quad + 3 \quad + 5x_3 + 2x_4 \leq 10$
- $x_3 + x_4 \leq 1$
- $-1 + x_3 \leq 0$
- $-1 + x_4 \leq 0$
- ~~$x_i \leq 1, x_i \geq 0, x_i$ **integer**~~

Z =   x =

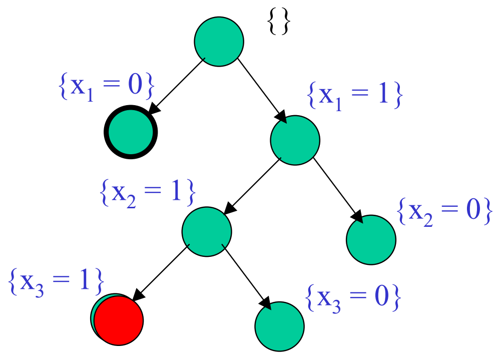
- Bound

Queue: ~~$\{x_3 = 1\}$~~ $\{x_3 = 0\}$ $\{x_2 = 0\}$

Incumbent: **x = <0,1,0,1>**

Best cost Z*: 9

Example: B&B for BIPs



Solve:

Max $Z = 9 \quad + 5 \quad + 6x_3 + 4x_4$

Subject to:

- $6 \quad + 3 \quad + 5x_3 + 2x_4 \le 10$
- $x_3 \; + x_4 \le 1$
- $-1 \; + x_3 \le 0$
- $-1 \; + x_4 \le 0$
- ~~$x_i \le 1, x_i \ge 0, x_i$ **integer**~~

Z = **x =**

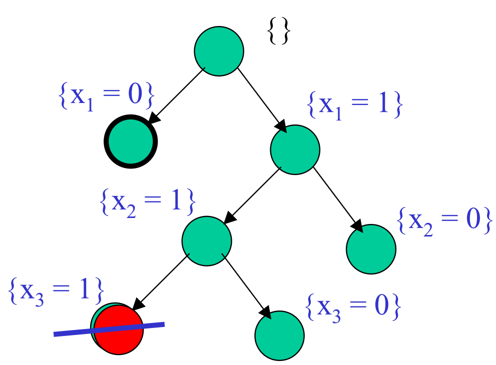
- Bound

Queue: $\{x_3 = 0\}\{x_2 = 0\}$

Incumbent: **x = <0,1,0,1>**

Best cost Z*: 9

Example: B&B for BIPs



Solve:

Max $Z = 9 \quad + 5 \quad + 6 \quad + 4x_4$

Subject to:

- $6 \quad + 3 \quad + 5 \quad + 2x_4 \le 10$
- $1 \; + x_4 \le 1$
- $-1 \; + 1 \; \le 0$
- $-1 \; + x_4 \le 0$
- ~~$x_i \le 1, x_i \ge 0, x_i$ **integer**~~

Z = **x =**

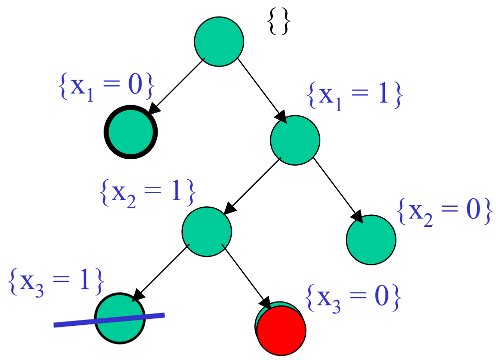
- Try to fathom:
  - infeasible?

Queue: ~~$\{x_3 = 0\}$~~$\{x_2 = 0\}$

Incumbent: **x = <0,1,0,1>**

Best cost Z*: 9

Example: B&B for BIPs



Solve:

Max $Z = 9 \quad + 5 \quad + 6x_3 + 4x_4$

Subject to:

- $6 \quad + 3 \quad + 5x_3 + 2x_4 \leq 10$
- $x_3 + x_4 \leq 1$
- $-1 + x_3 \leq 0$
- $-1 + x_4 \leq 0$
- ~~$x_i \leq 1, x_i \geq 0, x_i$ **integer**~~

Z = x =

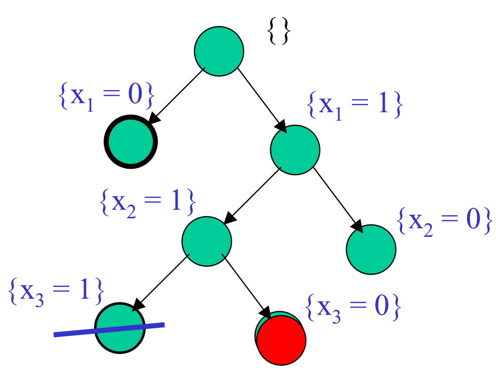
- Dequeue & bound

Queue: $\{x_2 = 0\}$

Incumbent: **x = <0,1,0,1>**

Best cost Z*: 9

Example: B&B for BIPs



Solve:

Max $Z = 9 \quad + 5 \quad + \quad + 4x_4$

Subject to:

- $6 \quad + 3 \quad + \quad + 2x_4 \leq 10$
- $+ x_4 \leq 1$
- $-1 \quad \leq 0$
- $-1 + x_4 \leq 0$
- ~~$x_i \leq 1, x_i \geq 0, x_i$ **integer**~~

Z = 16, x = <1,1,0,.5>

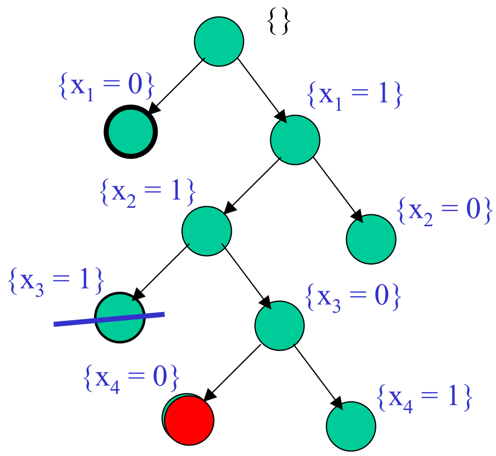
- Try to fathom:
  - infeasible?
  - worse than incumbent?
  - integer solution?

Queue: $\{x_2 = 0\}$

Incumbent: **x = <0,1,0,1>**

Best cost Z*: 9

Example: B&B for BIPs



Solve:

Max Z = 9 + 5

Subject to:

- 6 + 3 ≤ 10
- ≤ 1
- -1 ≤ 0
- -1 ≤ 0
- ~~$x_i \le 1$, $x_i \ge 0$, x_i **integer**~~

Z = 14, x = <1,1,0,0>

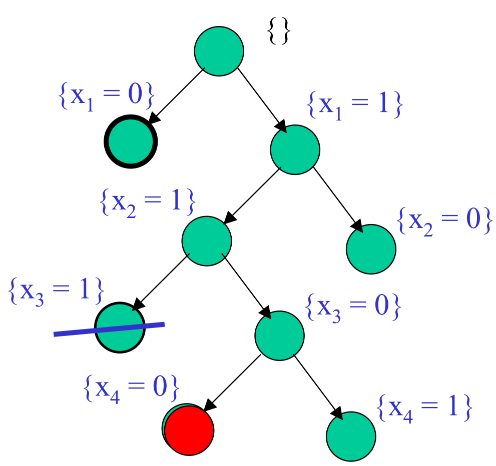
Queue: ~~$\{x_4 = 0\}$~~ $\{x_4 = 1\}$ $\{x_2 = 0\}$

Incumbent: **x = <0,1,0,1>**

Best cost Z*: 9

- Try to fathom:
  - infeasible?
  - worse than incumbent?
  - integer solution?

Example: B&B for BIPs



Solve:

Max Z = 9 + 5

Subject to:

- 6 + 3 ≤ 10
- ≤ 1
- -1 ≤ 0
- -1 ≤ 0
- ~~$x_i \le 1$, $x_i \ge 0$, x_i **integer**~~

Z = 14, x = <1,1,0,0>

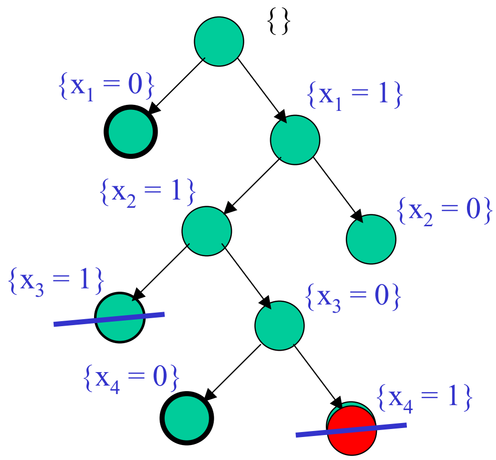
Queue: $\{x_4 = 1\}$ $\{x_2 = 0\}$

Incumbent: **x = <1,1,0,0>**

Best cost Z*: 14

- Try to fathom:
  - infeasible?
  - worse than incumbent?
  - integer solution?

Example: B&B for BIPs



Solve:

Max Z = 9 + 5 + 4

Subject to:

- 6 + 3 + 2 ≤ 10
- + 1 ≤ 1
- -1 ≤ 0
- -1 + 1 ≤ 0
- ~~$x_i \leq 1$, $x_i \geq 0$, x_i **integer**~~

Z = 18, x = <1,1,0,1>

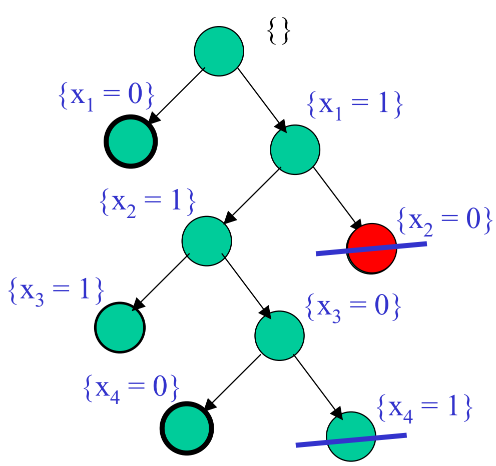
Queue: ~~$\{x_4 = 1\}$~~ $\{x_2 = 0\}$

Incumbent: **x = <1,1,0,0>**

Best cost Z*: 14

- Try to fathom:
  - infeasible?
  - worse than incumbent?
  - integer solution?

Example: B&B for BIPs



Solve:

Max Z = 9 $+ 6x_3 + 4x_4$

Subject to:

- 6 $+ 5x_3 + 2x_4 \leq 10$
- $x_3 + x_4 \leq 1$
- $-1_1 + x_3 \leq 0$
- $-1 + x_4 \leq 0$
- ~~$x_i \leq 1$, $x_i \geq 0$, x_i **integer**~~

Z = 13.~~8~~, x = <1,0,.8,0>

Queue: ~~$\{x_2 = 0\}$~~

Incumbent: **x = <1,1,0,0>**

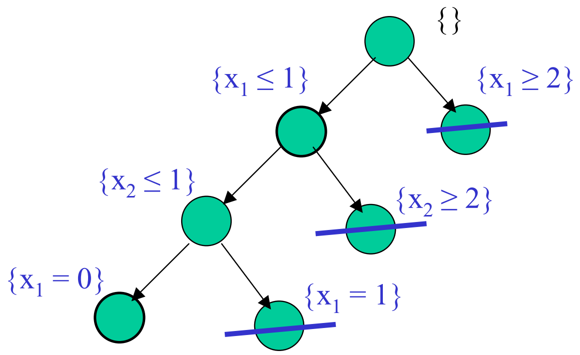
Best cost Z*: 14

- Try to fathom:
  - infeasible?
  - worse than incumbent?
  - integer solution?

Integer Programming (IP)

- What is it?
- Making decisions with IP
  - – Exclusion between choices
  - – Exclusion between constraints
- Solutions through branch and bound
  - – Characteristics
  - – Solving Binary IPs
  - – Solving Mixed IPs and LPs

Example: B&B for MIPs



Max $Z = 4x_1 - 2x_2 + 7x_3 - x_4$

Subject to:

- $x_1 + 5x_3 \leq 10$
- $x_1 + x_2 - x_3 \leq 1$
- $6x_1 + 5x_2 \leq 0$
- $-x_1 + 2x_3 - 2x_4 \leq 3$
- $x_i \geq 0$, x_i **integer** x_1, x_2, x_3,

Z = 14.25, x = <1.25,1.5,1.75,0>

Z = 14.2, x = <1,1.2,1.8,0>

Z = 14 1/6, x = <5/6,1,11/6,0>

Z = 13.5, x = <0,0,2,.5>

Infeasible, x = <1,≤1,?,?>

Z = 12 1/6, x = <5/6,2,11/6,0>

Infeasible, x = < ≥2,?,?,?>

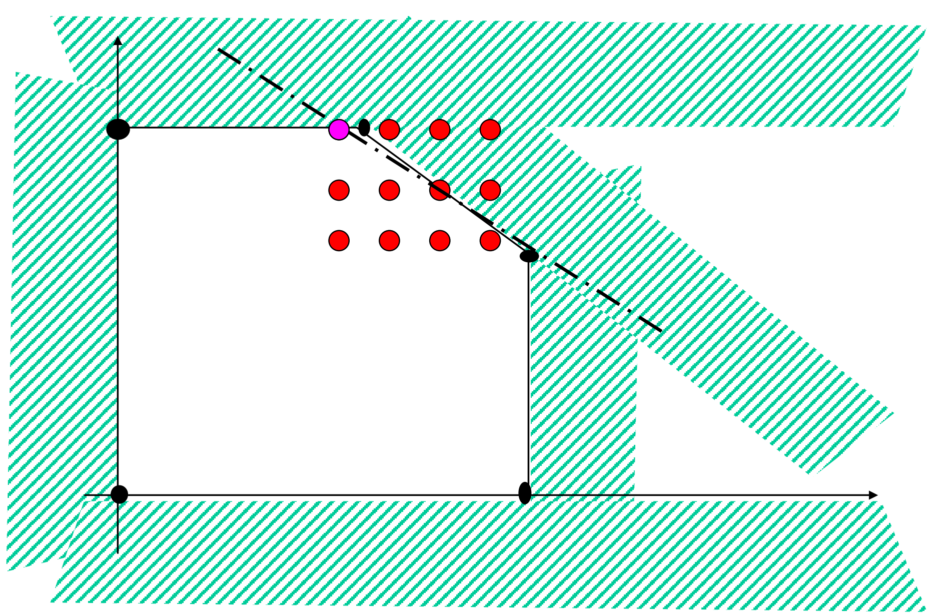
Incumbent: **x = <0,0,2,.5>**

Best cost Z*: 13.5

Cutting planes methods

- Cutting planes are an alternative to Branch and Bound search, though the methods do have similarities
- Gomory invented the first finitely terminating cutting plane method in 1958.

Cutting planes cut off non-integer solutions



Cutting planes algorithm

- Solve the linear programming relaxation of the integer program.
- If x*, the optimal solution, is integer, then x* is optimal for the integer program.
- If not, add a linear inequality constraint (cutting plane) that all integer solutions to the original problem satisfy but that is violated by x*. Repeat all steps.

Gomory cutting planes

- Given a problem with all variables integer constrained, solve the relaxed LP and examine the final simplex tableau. Take one equality with a fractional right-hand-side b_i. This is an equality that forces an integer constrained variable to be fractional.

$$x_i + \underbrace{a_{ij} x_j + \cdots + a_{ik} x_k}_{\text{Non-basic variables and coefficients}} = b_i$$

Gomory cutting planes

- Since at this tableau all of x_j through x_k are 0, and in all feasible solutions they are ≥ 0, it must be that

$$x_i + \lfloor a_{ij} \rfloor x_j + \cdots + \lfloor a_{ik} \rfloor x_k \leq x_i + a_{ij} x_j + \cdots + a_{ik} x_k = b_i$$

- So add the cutting plane constraint that x* does not satisfy but all integer solns do:

$$x_i + \lfloor a_{ij} \rfloor x_j + \cdots + \lfloor a_{ik} \rfloor x_k \leq \lfloor b_i \rfloor$$

Example: Gomory cutting plane

$$\begin{aligned} &\text{Minimize } Z = x_1 - 2x_2 \\ &\text{Subject to } -4x_1 + 6x_2 \leq 9 \\ &\qquad x_1 + x_2 \leq 4 \\ &\qquad x_1 \geq 0, x_2 \geq 0,\ x_1, x_2 \text{ integer} \end{aligned}$$

$$\begin{aligned} &\text{Minimize } Z = x_1 - 2x_2 \\ &\text{Subject to } -4x_1 + 6x_2 + x_3 \qquad = 9 \\ &\qquad x_1 + x_2 \qquad + x_4 = 4 \\ &\qquad x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, x_4 \geq 0, \\ &\qquad x_1, x_2, x_3, x_4 \text{ integer} \end{aligned}$$

Example: Gomory continued

- Solve the LP relaxation to find x*=(3/2,5/2)
- One of the equations in the final simplex tableau is:

$$x_2 + \frac{1}{10}x_3 + \frac{1}{10}x_4 \le 5/2$$

- Taking the floor of all non-basic variable coefficients and the right-hand-side, get $x_2 \le 2$ and add this constraint to the LP

Example: Gomory continued

- Solve

$$\begin{aligned}
&\text{Minimize } Z = x_1 - 2x_2 \\
&\text{Subject to } -4x_1 + 6x_2 + x_3 = 9 \\
&\qquad x_1 + x_2 + x_4 = 4 \\
&\qquad x_2 + x_5 = 2 \\
&\qquad x_1 \ge 0, x_2 \ge 0, x_3 \ge 0, x_4 \ge 0
\end{aligned}$$

- And find x*=(3/4,2). One of the equations in the optimal tableau is

$$x_1 - \frac{1}{4}x_3 + \frac{3}{2}x_5 \le \frac{3}{4}$$

Example: Gomory continued

- Add the new Gomory cut: $x_1 - x_3 + x_5 \le 0$
- Notice that in terms of the original variables, the new cut is: $-3x_1 + 5x_2 \le 7$
- Add this constraint and solve the relaxed LP

$$
\begin{aligned}
&\text{Minimize } Z = x_1 - 2x_2 \\
&\text{Subject to } -4x_1 + 6x_2 + x_3 = 9 \\
&\qquad x_1 + x_2 + x_4 = 4 \\
&\qquad x_2 + x_5 = 2 \\
&\qquad -3x_1 + 5x_2 = 7 \\
&\qquad x_1 \ge 0, x_2 \ge 0, x_3 \ge 0, x_4 \ge 0, x_5 \ge 0
\end{aligned}
$$

Example: Gomory continued

- The new x*=(1,2), which is integer. Why is this x* definitely the optimal solution to the original integer program?
  - x*=(1,2) is integer so it is feasible for the integer program
  - No better integer-valued x existed at the start, because none of the Gomory cuts can eliminate any integer solution, regardless of its value.

Problems with cutting planes

- Each cutting plane usually eliminates a very small part of the feasible region.
- Although finite, no cutting plane method is polynomial-time.
- The more one knows about the specific structure of the problem, the better one can do with cutting planes. Specialized cutting plane algorithms sometimes work remarkably well, and CPLEX allows you to turn specific types on and off.