Part 4
Integer Programming

Read Chapters 11 and 12 in textbook

A capital budgeting problem

- We want to invest \$19'000
- Four investment opportunities which cannot be split (take it or leave it)
  1. Investment of \$6'700 and net present value of \$8'000
  2. Investment of \$10'000 and net present value of \$11'000
  3. Investment of \$5'500 and net present value of \$6'000
  4. Investment of \$3'400 and net present value of \$4'000
- Since investments cannot be split up, we cannot model this with continuous variables as in linear programming

An integer program

$$
\begin{aligned}
&\max 8x_1 + 11x_2 + 6x_3 + 4x_4 \\
&\text{subject to} \\
&6.7x_1 + 10x_2 + 5.5x_3 + 3.4x_4 \leqslant 19 \\
&x_i \in \{0,1\}
\end{aligned}
$$

Solving the integer program

Encode problem in lp-format (or mps format):

```
Maximize
obj: 8 x1 + 11 x2 + 6 x3 + 4 x4
Subject to
c1: 6.7 x1 + 10 x2 + 5.5 x3 + 3.4 x4 <= 19
Binary
x1 x2 x3 x4
End
```

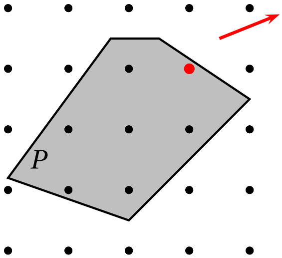
An open source solver for mixed integer programming is for example symphony (see `http://www.coin-or.org/`)
Optimal solution: $x_1 = 0$, $x_2 = x_3 = x_4 = 1$

Definition of integer programming

Mixed integer program (MIP)

$$\max c^T x$$
$$Ax \leqslant b$$
$$x_i \in \mathbb{Z} \text{ for } i = 1, \ldots, p.$$

Here $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$ and $c \in \mathbb{Q}^n$. If $p = n$ (all variables have to be integral), then we speak about pure integer program.
$x \in \mathbb{R}^n$ is integer feasible, if x satisfies all linear constraints and the constraints $x_i \in \mathbb{Z}$ for $i = 1, \ldots, p$.



Solving MIPs

LP-relaxation

Ignoring constraints $x_i \in \mathbb{Z}$ for $i = 1, \ldots, p$ yields linear program, called the LP-relaxation. The value of LP-relaxation is upper bound on the optimum value of the MIP.

Example of branch and bound

Consider pure IP

```
Maximize
obj: x1 + x2
Subject to
c1: -x1 + x2 <= 2
c2: 8 x1 + 2 x2 <= 19
Bounds
x1 x2 >= 0
Integer
x1 x2
End
```

Example of branch and bound

LP-relaxation

```
Maximize
obj: x1 + x2
Subject to
c1: -x1 + x2 <= 2
c2: 8 x1 + 2 x2 <= 19
Bounds
x1 x2 >= 0
End
```

Solution of LP-relaxation

- $x_1 = 1.5$, $x_2 = 3.5$
- Value: $x = 5$

Example of branch and bound

LP-relaxation

```
Maximize
obj: x1 + x2
Subject to
c1: -x1 + x2 <= 2
c2: 8 x1 + 2 x2 <= 19
Bounds
x1 x2 >= 0
End
```

Solution of LP-relaxation

- $x_1 = 1.5$, $x_2 = 3.5$
- Value: $z = 5$

Example of branch and bound

Create two sub-problems:

Left sub-problem

```
Maximize
obj: x1 + x2
Subject to
c1: -x1 + x2 <= 2
c2: 8 x1 + 2 x2 <= 19
Bounds
x1 x2 >= 0
x1 <= 1
End
```

Solution of left subproblem

- $x_1 = 1$, $x_2 = 3$ (integral feasible)
- Value: $z = 4$

Example of branch and bound

Right sub-problem

```
Maximize
obj: x1 + x2
Subject to
c1: -x1 + x2 <= 2
c2: 8 x1 + 2 x2 <= 19
Bounds
x1 x2 >= 0
x1 >= 2
End
```

Solution of right subproblem

- $x_1 = 2$, $x_2 = 1.5$ (integral infeasible)
- Value: $z = 3.5$

Optimal solution

- Each integer feasible solution of right sub-problem has value bounded by 3.5.
- Since value of integer feasible solution $x_1 = 1$, $x_2 = 3$ is 4, we can prune the right sub-problem
- Since integer feasible solution $x_1 = 1$, $x_2 = 3$ is also optimal solution of left sub-problem, each integer feasible solution of left-subproblem has value at most 4.
- Thus $x_1 = 1$ and $x_2 = 3$ is optimum solution to integer program.

Branch and Bound

L is list of linear programs, z_L is global lower bound on value of MIP, x^* is integer feasible solution of *MIP*

Branch & Bound

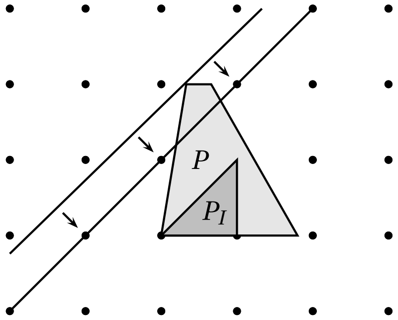
1. (Initialize) $L = \{\text{LP-Relaxation of MILP}\}$, $z_L = -\infty$, $x^* = \emptyset$
2. (Terminate?) If $L = \emptyset$, then x^* is optimal
3. (Select node) Choose and delete problem N_i from L
4. (Bound) Solve N_i. If N_i is infeasible, then goto 2), else let x^i be its optimal solution and z_i be its objective value.
5. (Prune) If $z_i \leqslant z_L$, then go to 2).
   If x^i is not integer feasible, then go to step 6)
   If x^i is integer feasible, then set $z_L = z_i$ and $x^* = x^i$. Go to step 2)
6. (branch) From N_i construct linear programs $N_i^1, \ldots, N_i^k$ with smaller feasible region whose union contains all integer feasible solutions of N_i. Add $N_i^1, \ldots, N_i^k$ to L and go to step 2).

Branching

- Let x^i be solution to linear program N_i
- Let x^i_j be one of the non-integral components of x^i for $j \in \{1, \dots, p\}$
- Each integer feasible solution satisfies $x_j \leqslant \lfloor x^i_j \rfloor$ or $x_j \geqslant \lceil x^i_j \rceil$.
- One way to branch is to create sub-problems $N^-_{ij} := \{N_i, x_j \leqslant \lfloor x^i_j \rfloor\}$ and $N^+_{ij} := \{N_i, x_j \geqslant \lceil x^i_j \rceil\}$
- Strong branching creates those sub-problems whose sum of values is as small as possible (tightening the upper bound)

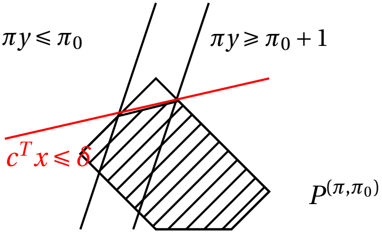
Cutting planes

- Suppose we have pure integer program $\max\{c^T x \colon Ax \leqslant b,\, x \in \mathbb{Z}^n\}$
- Set $P = \{x \in \mathbb{R}^n \colon Ax \leqslant b\}$ is called polyhedron
- Integer hull is convex hull of $P \cap \mathbb{Z}^n$.
- If $c^T x \leqslant \delta$, $c \in \mathbb{Z}^n$ is valid for P, then $c^T x \leqslant \lfloor \delta \rfloor$ valid for integer hull P_I of P.
- Cutting plane $c^T x \leqslant \lfloor \delta \rfloor$ strengthens LP-relaxation



Cutting planes for mixed integer programs

- $\max\{c^T x \colon Ax \leqslant b, x_i \in \mathbb{Z} \text{ for } i = 1,\ldots,p\}$ MIP, denote vector first p variables $x_1,\ldots,x_p$ by y
- Split: Tuple (π,π_0), $\pi \in \mathbb{Z}^p$, $\pi_0 \in \mathbb{Z}$
- Each integer feasible solution x in polyhedron satisfies $\pi^T y \leqslant \pi_0$ or $\pi^T y \geqslant \pi_0 + 1$
- $P = \{x \in \mathbb{R}^n \colon Ax \leqslant b\}$ polyhedron: $P^{(\pi,\pi_0)} = \operatorname{conv}\big(P \cap (\pi y \leqslant \pi_0), P \cap (\pi y \geqslant \pi_0 + 1)\big)$.
- Split cut is inequality $c^T x \leqslant \delta$ such that there exists a split (π,π_0) such that $c^T x \leqslant \delta$ is valid for $P^{(\pi,\pi_0)}$



In practice

Mixed integer linear programs are solved with a combination of branch & bound and cutting planes

PART 4.1
APPLICATIONS OF MIXED INTEGER PROGRAMMING

Combinatorial auctions

Problem description

- Auctioneer sells items $M = \{1, \ldots, m\}$
- Bid is a pair $B_j = (S_j, p_j)$, where $S_j \subseteq M$ and p_j is a price
- Auctioneer has received n bids $B_1, \ldots, B_n$
- Question: How should auctioneer determine winners and losers in order to maximize his revenue?

Example

- Four items $M = \{1, 2, 3, 4, \}$
- Bids: $B_1 = (\{1\}, 6)$, $B_2 = (\{2\}, 3)$, $B_3 = (\{3, 4\}, 12)$, $B_4 = (\{1, 3\}, 12)$, $B_5 = (\{2, 4\}, 8)$, $B_6 = (\{1, 3, 4\}, 16)$

Integer program

```
Maximize
obj: 6 x1 + 3 x2 + 12 x3 + 12 x4 + 8 x5 + 16 x6
Subject to
c1: x1 + x4 + x6 <= 1
c2: x2 + x5 <= 1
c3: x3 + x4 + x6 <= 1
c4: x3 + x5 + x6 <= 1
Binary
x1 x2 x3 x4 x5 x6
End
```

Several indistinguishable items

- u_i: Number of items of type i
- Bid is tuple: $B_j = (\lambda_1^j, \dots, \lambda_m^j, p_j)$

Integer program

$$\max \sum_{i=1}^{n} p_j x_j$$
$$\sum_j \lambda_i^j x_j \leqslant u_i \text{ for } i = 1, \dots, m$$
$$x_j \in \{0,1\},\ j = 1, \dots, n.$$

The lockbox problem

- National firm in US receives checks from all over the country
- Delay from obligation of customer (check postmarked) to clearing (check arrives)
- Money should be available as soon as possible
- Idea: Open offices all over country to receive checks and to minimize delay

Example

- Receive payments from four regions: West, Midwest, East, South
- Average daily value from each region is: $600 K, $240 K, $720 K, $ 360 K respectively
- Operating Lockbox costs $90 K per year

Clearing times:

From	L.A.	Pittsburgh	Boston	Houston
West	2	4	6	6
Midwest	4	2	5	5
East	6	5	2	5
South	7	5	6	3

Example cont.

- Average of $3'600K = 6 \times 600K$ is in process any given day considering West sending to Boston
- Assuming 5% interest rate per year, this corresponds to a loss of interest of 180 K per year

Complete table of lost interest in $K:

From	L.A.	Pittsburgh	Boston	Houston
West	60	120	180	180
Midwest	48	24	60	60
East	216	180	72	180
South	126	90	108	54

Example cont.

Integer programming formulation

- $y_j \in \{0, 1\}$ indicates whether lockbox j is open or not
- $x_{ij} = 1$ if region i sends checks to lockbox j
- Objective is to minimize total yearly loss

$$\min 60x_{11} + 120x_{12} + 180x_{13} + 180x_{14} + 48x_{12} \ldots + 90y_1 + \ldots + 90y_4$$

- Each region is assigned to exactly one lockbox

$$\sum_j x_{ij} = 1 \text{ for all } i$$

- Regions can only send to open lockboxes:

$$\sum_i x_{ij} \leqslant 4y_j \text{ for all } j$$

Complete IP

```
Minimize
obj:  60 X11 + 120 X12 + 180 X13 + 180 X14
+ 48 X21 + 24 X22 + 60 X23 + 60 X24
+ 216 X31 + 180 X32 + 72 X33 + 180 X34
+ 126 X41 + 90 X42 + 108 X43 + 54 X44
+ 90 Y1 + 90 Y2 + 90 Y3 + 90 Y4
Subject to
c1: X11 + X12 + X13 + X14 = 1
c2: X21 + X22 + X23 + X24 = 1
c3: X31 + X32 + X33 + X34 = 1
c4: X41 + X42 + X43 + X44 = 1
c5: X11 + X21 + X31 + X41 - 4 Y1 <= 0
c6: X12 + X22 + X32 + X42 - 4 Y2 <= 0
c7: X13 + X23 + X33 + X43 - 4 Y3 <= 0
c8: X14 + X24 + X34 + X44 - 4 Y4 <= 0
Binary
X11 X12 X13 X14 X21 X22 X23 X24 X31
```

Constructing an index fund

- Portfolio should reflect large index (like S&P 500)
- However, not all stocks should be bought (transaction costs)
- Suppose a measure of similarity is available: $0 \leqslant \rho_{ij} \leqslant 1$ for $i \neq j$, $\rho_{ii} = 1$.
- Variable x_{ij} models i being represented by j

IP model

$$
\begin{aligned}
& \max \textstyle\sum_{ij} \rho_{ij} x_{ij} \\
& \textstyle\sum_{j=1}^{n} y_j = q \\
& \textstyle\sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, \ldots, n \\
& x_{ij} \leqslant y_j, \quad i, j = 1, \ldots, n \\
& x_{ij}, y_j \in \{0, 1\}, \quad i, j = 1, \ldots, n
\end{aligned}
$$

Constructing an index fund cont.

- q stocks are selected
- Denote by V_i market value of stock i
- Weight of stock j

$$w_j = \sum_{i=1}^{n} V_i x_{ij}$$

- Fraction to be invested in j is proportional to stocks weight

$$\frac{w_j}{\sum_i w_i}$$