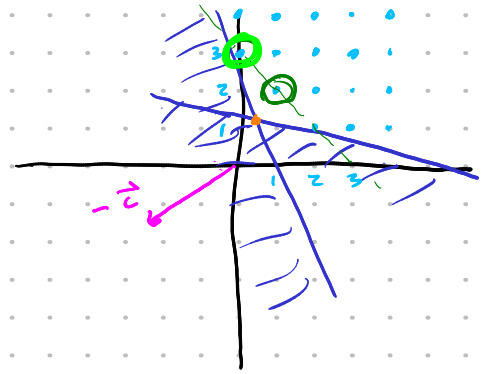


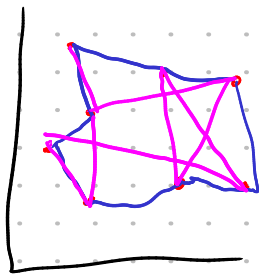
Discrete Optimization

Integer Linear Program (ILP)

$$\begin{aligned} & \underset{\vec{x}}{\text{minimize}} \quad \vec{c}^T \vec{x} \\ & \text{subject to} \quad A\vec{x} \leq \vec{b} \\ & \quad \quad \quad \vec{x} \geq 0 \\ & \quad \quad \quad \vec{x} \in \mathbb{Z}^n \end{aligned}$$



Travelling Salesman Problem (TSP)



Find shortest "tour" (path passing through all points)

Formulate as ILP

$$\begin{aligned} & \underset{\vec{x}}{\text{minimize}} \quad \sum_{i,j} d_{ij} x_{ij} \\ & \text{subject to} \quad x_{ij} \in \{0,1\} \end{aligned}$$

$$x_{ij} \in \{0,1\} \\ = \begin{cases} 1 & \text{if path contains } i \rightarrow j \\ 0 & \text{otherwise} \end{cases}$$

$$\begin{aligned} \sum_j x_{ij} &= 1 \quad \forall i \quad \text{"exit"} & \sum_j x_{ij} &= n \\ \sum_i x_{ij} &= 1 \quad \forall j \quad \text{"entry"} & \forall i \end{aligned}$$

$$u_i \in \{2, \dots, n\}$$

$$\forall i \quad x_{ii} = 0$$

$$u_i - u_j + 1 \leq (n-1)(1 - x_{ij}) \quad \forall i, j$$

Miller-Tucker-Zemlin

$$\text{want if } \begin{aligned} x_{ij} &= 1 \\ u_j - u_i &= 1 \end{aligned}$$

$$\text{if } x_{ij} = 1$$

$$u_i - u_j + 1 \leq 0$$

$$u_i - u_j \leq -1$$

$$u_j - u_i \geq 1$$

when applied to all i, j forces

$$u_j - u_i = 1$$

$$\text{if } x_{ij} = 0$$

$$u_i - u_j + 1 \leq n-1$$

worst case

$$u_i = n, u_j = 2$$

does not constrain u

MILP "Mixed Integer LP"

$$\min_{\vec{x}} \vec{c}^T \vec{x}$$

$$A\vec{x} \leq \vec{b} \quad \leftarrow \quad A\vec{x} = \vec{b}$$

$$\vec{x} \geq 0$$

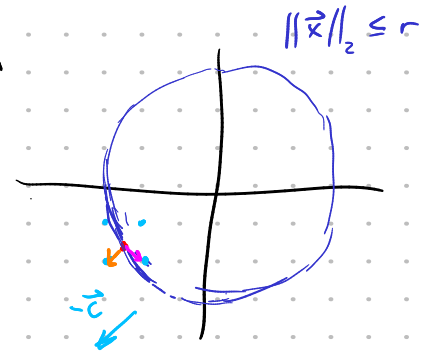
$$\vec{x}_D \in \mathbb{Z}^n$$

Rounding

relax $\vec{x}_D \in \mathbb{Z}^n$ constraint to $\vec{x}_D \in \mathbb{R}^n$

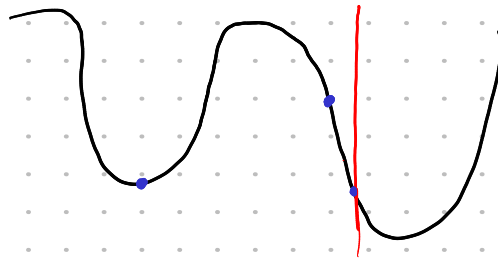
solve

round to nearest integer \vec{x}_D



Problems

- 1) result might be infeasible
- 2) nearest feasible integral solution might be much worse than optimal



Sometimes there are guarantees on how close the solution is

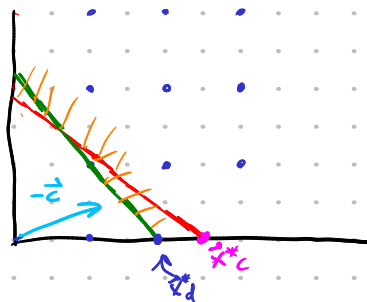
If $A \in \mathbb{Z}^{n \times m}$

$$\|x_d^* - x_c^*\| \leq n \times \max \text{ absolute value of determinants of submatrices of } A$$

\nwarrow relaxed solution
 \nearrow optimal discrete solution

S.O.T.A. "branch + cut"

Cutting Plane



Introduce new constraint that

- 1) excludes x_c^*
- 2) includes all other discrete solutions

Partition \vec{x}_c^* into (B, V)

$$\vec{x}_B^* \quad \vec{x}_V^* = 0$$

\nwarrow all of the non-integral components

For each $b \in B$ where x_b^* is non-integral introduce a new constraint

$$\underbrace{x_b^* - \lfloor x_b^* \rfloor}_{\geq 0} - \underbrace{\sum_{v \in V} (\bar{A}_{bv} - \lfloor \bar{A}_{bv} \rfloor)}_{=0} x_v \leq 0 \quad \bar{A}_{bv} = A_B^{-1} A_v$$

$$x_b + \sum_{v \in V} (\lfloor \bar{A}_{bv} \rfloor - \bar{A}_{bv}) x_v = \lfloor x_b^* \rfloor - x_b^*$$

$x_b \in \mathbb{N}$

Consider the integer program:

$$\begin{aligned} & \underset{x}{\text{minimize}} && 2x_1 + x_2 + 3x_3 \\ & \text{subject to} && \begin{bmatrix} 0.5 & -0.5 & 1.0 \\ 2.0 & 0.5 & -1.5 \end{bmatrix} x = \begin{bmatrix} 2.5 \\ -1.5 \end{bmatrix} \\ & && x \geq 0 \quad x \in \mathbb{Z}^3 \end{aligned}$$

The relaxed solution is $x^* \approx [0.818, 0, 2.091]$, yielding:

$$A_B = \begin{bmatrix} 0.5 & 1 \\ 2 & -1.5 \end{bmatrix} \quad A_v = \begin{bmatrix} -0.5 \\ 0.5 \end{bmatrix} \quad \bar{A} = \begin{bmatrix} -0.091 \\ -0.455 \end{bmatrix}$$

From equation (19.7), the constraint for x_1 with slack variable x_4 is:

$$\begin{aligned} x_4 + (\lfloor -0.091 \rfloor - (-0.091))x_2 &= \lfloor 0.818 \rfloor - 0.818 \\ x_4 - 0.909x_2 &= -0.818 \end{aligned}$$

The constraint for x_3 with slack variable x_5 is:

$$\begin{aligned} x_5 + (\lfloor -0.455 \rfloor - (-0.455))x_2 &= \lfloor 2.091 \rfloor - 2.091 \\ x_5 - 0.545x_2 &= -0.091 \end{aligned}$$

The modified integer program has:

$$A = \begin{bmatrix} 0.5 & -0.5 & 1 & 0 & 0 \\ 2 & 0.5 & -1.5 & 0 & 0 \\ 0 & -0.909 & 0 & 1 & 0 \\ 0 & -0.545 & 0 & 0 & 1 \end{bmatrix} \quad b = \begin{bmatrix} 2.5 \\ -1.5 \\ -0.818 \\ -0.091 \end{bmatrix} \quad c = \begin{bmatrix} 2 \\ 1 \\ 3 \\ 0 \\ 0 \end{bmatrix}$$

Solving the modified LP, we get $x^* \approx [0.9, 0.9, 2.5, 0.0, 0.4]$. Since this point is not integral, we repeat the procedure with constraints:

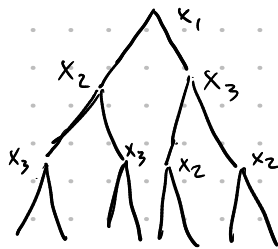
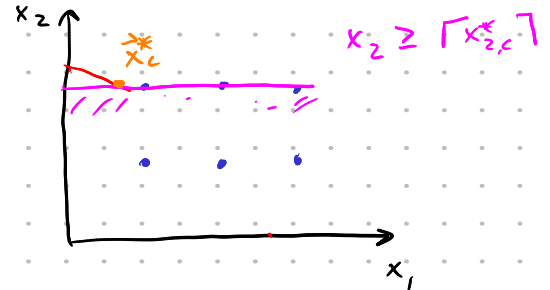
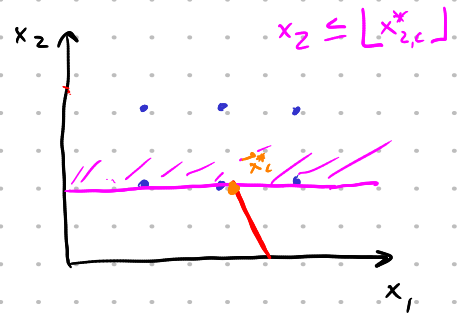
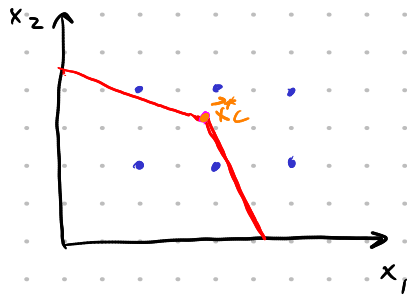
$$\begin{aligned} x_6 - 0.9x_4 &= -0.9 & x_7 - 0.9x_4 &= -0.9 \\ x_8 - 0.5x_4 &= -0.5 & x_9 - 0.4x_4 &= -0.4 \end{aligned}$$

and solve a third LP to obtain: $x^* = [1, 2, 3, 1, 1, 0, 0, 0, 0]$ with a final solution of $x_i^* = [1, 2, 3]$.

Branch and Bound

add constraints

$$x_i \leq \lfloor x_{i,c}^* \rfloor \quad x_i \geq \lceil x_{i,c}^* \rceil$$



or

Priority Queue
ranked by lower
bound

Which i to branch on next

Common heuristic $\vec{x}_c^* = [0.1, 0.4, 0.9]$

```
function minimize_lp_and_y(LP)
    try
        x = minimize_lp(LP)
        return (x, x.LP.c)
    catch
        return (fill(NaN, length(LP.c)), Inf)
    end
end

function branch_and_bound(MIP)
    LP = relax(MIP)
    x, y = minimize_lp_and_y(LP)
    n = length(x)
    x_best, y_best, Q = deepcopy(x), Inf, PriorityQueue()
    enqueue!(Q, (LP, x, y), y)
    while !isempty(Q)
        LP, x, y = dequeue!(Q)
        if any(isnan.(x)) || all(isint(x[i]) for i in MIP.D)
            if y < y_best
                x_best, y_best = x[1:n], y
            end
        else
            i = argmax([abs(x[i] - round(x[i])) for i in MIP.D])
            # x_i ≤ floor(x_i)
            A, b, c = LP.A, LP.b, LP.c
            A2 = [A zeros(size(A,1));
                  [j==i for j in 1:size(A,2)]' -1]
            b2, c2 = vcat(b, floor(x[i])), vcat(c, 0)
            LP2 = LinearProgram(A2, b2, c2)
            x2, y2 = minimize_lp_and_y(LP2)
            if y2 ≤ y_best
                enqueue!(Q, (LP2, x2, y2), y2)
            end
            # x_i ≥ ceil(x_i)
            A2 = [A zeros(size(A,1));
                  [j==i for j in 1:size(A,2)]' -1]
            b2, c2 = vcat(b, ceil(x[i]), vcat(c, 0)
            LP2 = LinearProgram(A2, b2, c2)
            x2, y2 = minimize_lp_and_y(LP2)
            if y2 ≤ y_best
                enqueue!(Q, (LP2, x2, y2), y2)
            end
        end
    end
    return x_best
end
```

Shortest Path

travel from node a to b in minimum distance

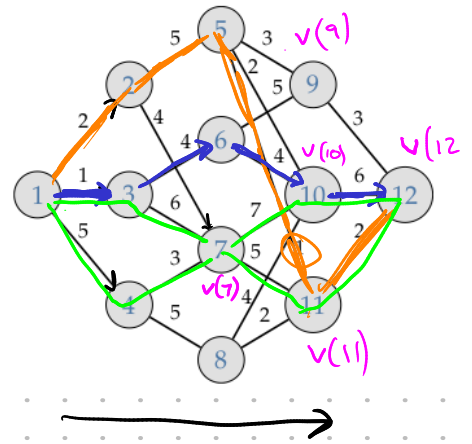
$$\text{minimize } \sum_{i,j} c_{ij} x_{ij}$$

$$\text{subject to } x_{ij} \in \{0, 1\}$$

$$\sum_{j \in \text{children}(i)} x_{ij} = 1 \quad \text{"start"}$$

$$\sum_{i \in \text{parents}(n)} x_{in} = 1 \quad \text{"end"}$$

$$\sum_{i \in \text{parents}(k)} x_{ik} - \sum_{j \in \text{children}(k)} x_{kj} = 0$$



$$\text{minimize}_{x_1, \dots, x_n} (c(s_1, x_1) + c(s_2, x_2) + \dots + c(s_n, x_n))$$

$$s_{k+1} = + (s_k, x_k)$$

$$c(s_k, x_k)$$

Greedy Algorithm (not always optimal)

$$\text{minimize}_{x_i} c(s_i, x_i)$$

Bellman's Principle of Optimality

"Every sub-path of an optimal path is optimal"

$$v(s_i) = \text{minimize}_{x_i, \dots, x_n} (c(s_i, x_i) + \underbrace{c(s_{i+1}, x_{i+1}), \dots, c(s_n, x_n)}_{v(s_{i+1})})$$

$$= \text{minimize}_{x_i} (c(s_i, x_i) + v(s_{i+1}))$$

$$= \text{minimize}_{x_i} (c(s_i, x_i) + v(+ (s_i, x_i)))$$

