□ Exam 3

- Linear Programming
  - must max or min something given restraints
    - ex max profit given limits on materials
    - objective function is max/min of linear function of vars
    - all constraints are linear functions of variables
- Standard Form
  - n variables $x_1, \dots, x_n$
  - objective function: $\max\ c_1x_1 + c_2x_2 + \dots + c_nx_n$
  - constraints ⇒ s.t.: $a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n \leq b_1$

$$\vdots$$

$$a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n \leq b_m$$

$$x_1, \dots, x_n \geq \emptyset$$

- Linear Algebra Form
  - variables $x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$ objective function $c = \begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix}$

constraints $b = \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix}$ constraint matrix A (m×n size)

- ex: $\max\ x_1 + 6x_2 + 10x_3$

st

$$x_1 \leq 300$$
$$x_2 \leq 200$$
$$x_1 + 3x_2 + 2x_3 \leq 1000$$
$$x_2 + 3x_3 \leq 500$$
$$x_1, x_2, x_3 \geq 0$$

$c = \begin{bmatrix} 1 \\ 6 \\ 10 \end{bmatrix}$ $b = \begin{bmatrix} 300 \\ 200 \\ 1000 \\ 500 \end{bmatrix}$ $A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 3 & 2 \\ 0 & 1 & 3 \end{bmatrix}$

- Converting to standard
  - $\min c^T x \Leftrightarrow \max -c^T x$
  - $a_1x_1 + \ldots + a_nx_n \geq b \Leftrightarrow -a_1x_1 - \ldots - a_nx_n \leq -b$
  - $a_1x_1 + \ldots + a_nx_n = b \Leftrightarrow a_1x_1 + \ldots + a_nx_n \leq b$ <u>and</u> $\geq b$
  - unconstrained $x \Rightarrow$ add x^+, x^- ; $x^+ \geq 0$; $x^- \geq 0$; $x = x^+ - x^-$
  - NO STRICT INEQUALITIES
- Simplex Algorithm $\Rightarrow$ worst case exponential time
  - Start at $x=0$
  - Look for neighboring vertex with higher objective value
    - if exists, move to that vertex and repeat
    - if not, output x
- feasible region is convex (linear proves this)
  - infeasible $\Rightarrow$ feasible region is empty
  - unbounded $\Rightarrow$ optimal is arbitrarily large
  - feasibility depends ONLY on constraints
  - bounds depend also on objective
  - check for infeasible
    - add new var z with no constraint and max z
    - if $z \geq 0$, original LP is feasible
- Duality
  - make LP in standard form, dual is min
  - m constraint values become objective function c
  - n objectives become constraint values b, all go $\leq \rightarrow \geq$
  - constraints A are transposed
  - ex:

$$\begin{array}{rcl} \max\ x_1 + 6x_2 + 10x_3 & & \min\ 300y_1 + 200y_2 + 1000y_3 + 500y_4 \\ x_1 \leq 300 & & y_1 + y_3 \geq 1 \\ x_2 \leq 200 & \Rightarrow & y_2 + 3y_3 + y_4 \geq 6 \\ x_1 + 3x_2 + 2x_3 \leq 1000 & & 2y_3 + 3y_4 \geq 10 \\ x_2 + 3x_3 \leq 500 & & y_1, y_2, y_3, y_4 \geq 0 \\ x_1, x_2, x_3 \geq 0 & & \end{array}$$

- dual of dual is original LP

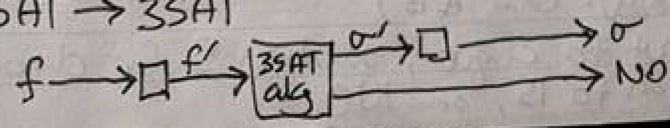
- Weak Duality Theorem
  - feasible x for primal ⇔ feasible y for dual
  - $c^T x \leq b^T y$
  - if exists feasible x and y where $c^T x = b^T y$, x, y are optimal
    - This always exists if primal and dual are feasible and bounded
- ~~Strong Duality~~ Weak Duality II
  - unbounded primal ⇒ infeasible dual
  - unbounded dual ⇒ infeasible primal
  - infeasible primal ⇒ unbounded or infeasible dual
  - infeasible dual ⇒ unbounded or infeasible primal
- Strong Duality
  - Primal feasible, bounded iff Dual feasible, bounded
  - primal has optimal x^* iff Dual has optimal y^*
  - $c^T x^* = b^T y^*$
    - size of max flow = capacity of min cut

- NP
  - NP ⇒ Class of all search problems
    - solutions can be verified in polynomial time
  - P ⇒ class of search problems solvable in poly-time
  - $P \subset NP$? $P = NP$? we don't know
  - SAT
    - input: boolean formula in CNF w/ n vars and m clauses
    - output: satisfying assignment if exists, NO otherwise
    - verify in $O(mn)$
    - SAT ∈ NP: $O(n)$ to check each clause, $O(mn)$ total
- K-Colorings
  - input: undir $G = (V, E)$ & int $K > 0$
  - output: each $v \in V$ assigned a color $\{1, \ldots, K\}$ so that adjacent verts get different colors, or NO
  - K-Colorings ∈ NP: $O(m)$ to check for $(v,w) \in E$ $color(v) \neq color(w$

- $MST \in P$
  - can find solution in poly-time
- Knapsack
  - Knapsack $\notin NP$
    - cannot verify value is max without solving
    - Knapsack $\notin P$, so cannot be done in poly-time
  - add goal g, want a solution $\geq g$
  - Knapsack Search $\in NP$: given input (w, v, B, g) and solution S
    - check:

$$\left.\begin{array}{l}\sum_{i \in S} w_i \leq B \\ \sum_{i \in S} v_i \geq g\end{array}\right\} O(n)$$

- NP- Complete
  - can all be reduced to each other
  - if any NP-complete problem can be solved in poly-time ALL NP-complete problems can be solved in poly-time
- SAT is NP-complete
  1) prove $SAT \in NP$
  2) reduce any OTHER NP-complete problem $\rightarrow$ SAT
- Proving NP-completeness
  - given new prob A, find similar known NP-comp. B
  - prove $A \in NP$
  - reduce $B \rightarrow A$
- 3SAT
  - input: bool formula f in CNF, w/ n vars & m clauses each clause has ≤ 3 literals
  - $3SAT \in NP$: given input f and solution for each var $(x_1 \rightarrow x_n)$ for each clause $c \in f$: check in $O(1)$ that ≥ 1 literal in c is satisfied $\Rightarrow O(m)$ total
- SAT $\rightarrow$ 3SAT



- SAT → 3SAT
  - $f \to f'$:
    for $c \in f$:
      if $|c| \leq 3$ add c to f'
      if $|c| > 3$ create $k-3$ new vars and replace c by $k-2$ clauses:
      $c = (a_1 \vee a_2 \vee \ldots \vee a_k)$
      $c' = (a_1 \vee a_2 \vee y_1) \wedge (\overline{y}_1 \vee a_3 \vee y_2) \wedge (\overline{y_2} \vee a_4 \vee y_3) \wedge \ldots \wedge (\overline{y_{k-4}} \vee a_{k-2} \vee y_{k-3}) \wedge (\overline{y_{k-3}} \vee a_{k-1} \vee a_k)$
- Independent Set
  - for undir $G=(V,E)$, $S \subset V$ is IS if for all $x,y \in S$, $(x,y) \notin E$
    - ie no 2 verts are connected
  - Max IS ∉ NP for same reason as Knapsack, add goal $\underline{g}$
  - IS is NP-complete
    - IS ∈ NP: given G, g, and solution S
      - $O(n^2)$ to check $x,y \in S$ $(x,y) \notin E$
      - $O(n)$ to check $|S| \geq g$
    - 3SAT → IS
      - for each $c \in f$: add a vertex for each literal and connect
      - add edge between all x_i and all $\overline{x_i}$ for $x_i \in f$
  - Max IS is at least as hard as everything in NP
    - this is NP-hard
- Clique
  - for $G=(V,E)$ and goal g, $S \subset V$ is clique if for all $x,y \in S$, $(x,y) \in E$
  - opposite of IS
  - Clique ∈ NP: given input (G, g) and solution S
    - $O(n^2)$ to check $x,y \in S$ $(x,y) \in E$
    - $O(n)$ to check $|S| \geq g$
  - IS → Clique
    - for $G=(V,E)$ input for IS, create $\overline{G}=(V,\overline{E})$ where $\overline{E} = \{(x,y) : (x,y) \notin E)$
    - $\overline{G}$ and g is input to clique, solution S for clique is also solution to IS, or NO

- Vertex Cover
  - for $G=(V,E)$ and budget b, $S \subset V$ is VC if for every $(x,y) \in E$ $x \in S$ and/or $y \in S$
  - "Covers every edge" with one or both vertices
  - want VC of size $|S| \leq b$ or NO
  - $VC \in NP$: Given (G,b) and S
    - $O(n+m)$ to check for $(x,y) \in E$, ≥ 1 of x,y are in S
    - $O(n)$ to check $|S| \leq b$
  - $IS \to VC$:
    - for input to IS G and g, let $b = n-g$
    - run VC on G, b
    - G has VC of size $\leq n-g \Leftrightarrow G$ has IS of size $\geq g$
    - given S for VC, $\overline{S}$ is solution for IS
- Subset Sum
  - given positive ints $\{a_1, \dots, a_n\}$ & t, find $S \subset \{1, \dots, n\}$ where $\sum_{i \in S} a_i = t$ or NO
  - $SubSum \in NP$: given inputs $a_1, \dots, a_n, t$ and solution S
    - $O(n \log t)$ to check $\sum_{i \in S} a_i = t$
  - $3SAT \to SubSum$
    - input to SubSum: $2n + 2m + 1$ ints
    - $v_1, v_1', v_2, v_2', \dots, v_n, v_n'; s_1, s_1', \dots, s_m, s_m'; t$
      - all are $\leq n+m$ digits long and base 10
      - $t \approx 10^{n+m}$
        - this shows why $O(nt)$ is not poly-time
    - v_i corresponds to x_i: $v_i \in S \Leftrightarrow x_i = T$
    - v_i' corresponds to $\overline{x_i}$: $v_i' \in S \Leftrightarrow x_i = F$
    - v_i OR v_i' is in S
    - Digit $n+j$ corresponds to clause C_j
      - if $x_i \in C_j$ put 1 in digit $n+j$ for v_i
      - if $\overline{x_i} \in C_j$ put 1 in digit $n+j$ for v_i'
    - put 3 in digit $n+j$ of t
    - put 1 in digit $n+j$ of s_j and s_j' as buffers
    - put 0 in digit $n+j$ for all others

- Halting Problem
  - undecidable: computationally impossible
  - given procedures Harmful(J) and Terminator (J,J)

```
Harmful(J)
 (1) if TERMINATOR(J,J):
          GOTO (1)
      else: return()

Terminator(J,J)  // runs prg J on input J
     return TRUE if J(J) terminates
     return FALSE if J(J) never terminates
```

- if J(J) terminates, Harmful(J) never terminates
- if J(J) never terminates, Harmful(J) terminates

- Exam 3 quick sheet
- Simplex Algorithm
  - worst case exponential
  - finds optimal objective val by traversing neighboring vertices
- Strong and Weak Duality

Primal/Dual	Infeasible	Optimal	Unbounded
Infeasible	✓	X	✓
Optimal	X	✓	X
Unbounded	✓	X	X

- NP- Complete Problems
  - SAT: Given input f, find satisfying assignment
  - K-Colorings: Given G and k, find S where for $(v,w) \in E$ $color(v) \neq color(w)$
  - Knapsack: Given (w, v, B, g), find solution w/ value $\geq g$
  - 3SAT: Given input f with $|c \in f| \leq 3$, find satisfying assignment
  - IS: Given G and g, find S where all $x, y \in S$, $(x,y) \notin E$ and $|S| \geq g$
  - Clique: Given (G, g), find S where all $x, y \in S$, $(x,y) \in E$ and $|S| \geq g$
  - VC: Given (G, b), find S where all $(x,y) \in E$: $x \in S$ and/or $y \in S$
  - SubSum: Given pos. ints $\{a_1, \ldots, a_n\}$ & t find $S \in \{1, \ldots, n\}$ where $\sum_{i \in S} a_i = t$