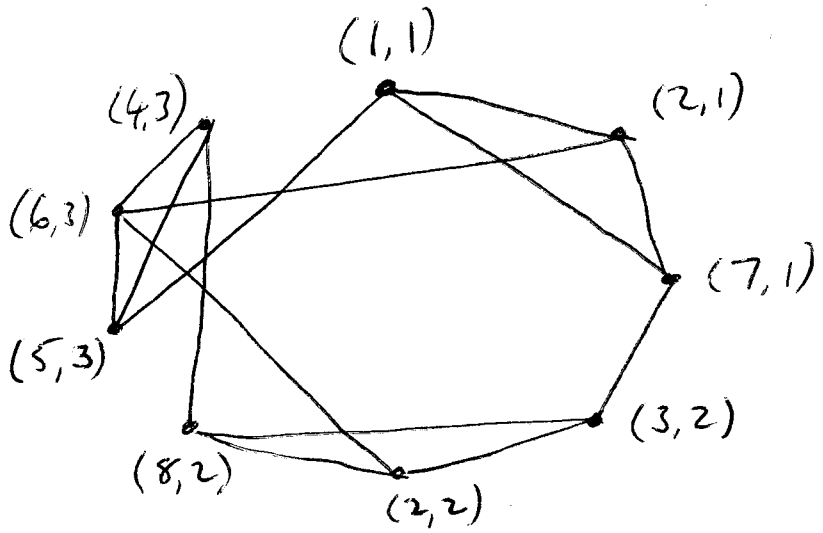
Reducing 3-SAT to Unweighted Node Packing:

C_1: $x_1 + x_2 + \bar{x}_3$

C_2: $x_2 + x_3 + \bar{x}_4$

C_3: $\bar{x}_1 + \bar{x}_2 + x_4$



Want a node packing of size ≥ 3.

(1,1), (2,1), (7,1) correspond to clique C_1

— at most one of these in a packing

(2,2), (3,2), (8,2) correspond to clique C_2,

and (5,3), (6,3), (4,3) correspond to clique C_3.

(1, i) is linked to (5, j):

taking (1, i) corresponds to setting x_1 = true,

so cannot have (5, j) simultaneously, because

that would correspond to x_1 = false.

Packing (1,1), (3,2), (6,3) shows instance is satisfiable:

x_1 = true, x_3 = true, x_2 = false, x_4 = anything.

Unweighted node packing (aka independent set)

Given a graph $G = (V, E)$, is there a $U \subseteq V$ such that $|U| \geq k$ and U is a node packing? i.e. if v_i, v_j are in U then $(v_i, v_j) \notin E$

Thm The lower bound feasibility problem for unweighted node packing is NP-complete

Proof Problem is a special case of 0-1 [illegible], so it is in NP.

To show NP-complete:

Reduce SAT to node packing:

Let $C_i = (C_i^+, C_i^-)$ be a clause, so C_i^+ is the literals in C_i which are not negations, C_i^- are those that are negations.

For C_i, we construct a clique with an vertex for each literal in C_i:

Let $V_i^+ = \{(j, i) : j \in C_i^+\}$, $V_i^- = \{(n+j, i) : j \in C_i^-\}$

$V_i = V_i^+ \cup V_i^-$

Each pair of nodes in V_i is connected by an edge, so any packing includes at most one vertex for V_i.

Node in packing $\leftrightarrow$ literal being true.

In addition, need to link cliques together, so that a variable can't have different values in

different cliques.

So $\nexists$ join (j, l) to $(i+1, k)$ for $j = 1, \ldots, n$, $k \neq l$.

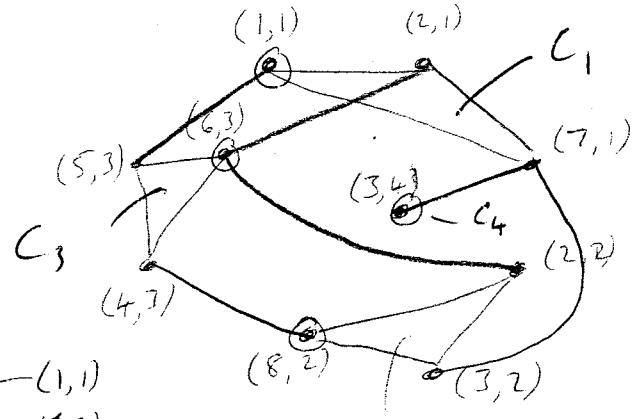
Σ

Look for a packing of size at least m.

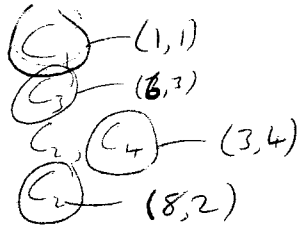
If SAT is true, there is an assignment such that at least one literal in each clause is true; taking the corresponding vertices gives a node packing.

Given a node packing of size k, have exactly one vertex from each clique; can set corresponding literal to be true. //

i	C_i^+	C_i^-
1	$\{1, 2\}$	$\{3\}$
2	$\{2, 3\}$	$\{4\}$
3	$\{4\}$	$\{1, 4\}$
4	$\{3\}$	ϕ



x_1 = true
x_2 = ~~true~~ false
x_3 = true
x_4 = ~~true~~ false



Knapsack feasibility w/ lower bounds problem:

~~Problem~~ Given a, b, c, z integers, does $\exists$ binary x with $c^T x \geq z$, $a^T x \leq b$?

~~$a, c \in \mathbb{N}^n$, $b, z \in \mathbb{N}$~~ $a, c \in \mathbb{Z}^n$, $b, z \in \mathbb{Z}$.

Thm The knapsack feasibility with lower bounds problem is NP-complete.

(No proof) (P&S, p. 374) (N&W, p. 137)

Thm $\exists$ $O(nb)$ algorithm for the knapsack problem

(No proof) (P&S, p. 287)
not but same general approach

So knapsack is somehow "close" to being polynomial – only reason it's not polynomial is because b requires only $\log b$ storage and $b = 2^{\log b}$.

The $O(nb)$ algo for knapsack is a pseudopolynomial algorithm.

Defn An algo runs in pseudopolynomial time if its running time is a polynomial function of the length of the data encoded in unary (a one-symbol alphabet) (or: poly in length of data and largest number in problem formulation

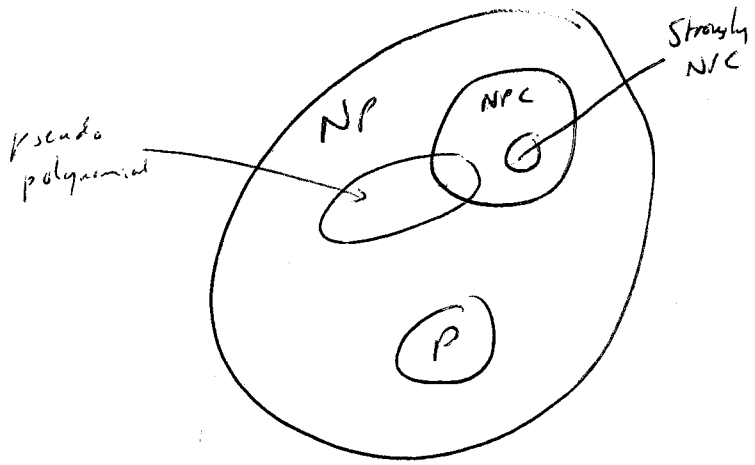
Defn Let number(D) be the size of the largest number appearing in the formulation of instance D of feasibility problem X. Let X_p denote X restricted to instances D for which $\text{number}(D) \leq p(|D|)$, where $|D|$ is length of encoding of instance D. X is strongly NP-complete if X_p is NP-complete for some polynomial $p(n)$. Eg clique, sat (node cover?).

NP, ≥ is related to feasibility problem.

What about optimization problem?

A problem is called NP-hard if there is an NP-complete problem that can be reduced to it.

So NP-hard problems are at least as difficult as NP-complete problems.



Co NP.

Co NP

Exa... NP ...?

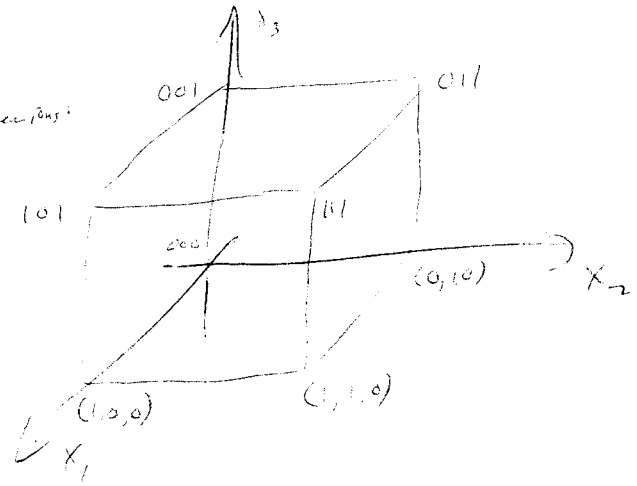
How good is the simplex algorithm?

Worst case behaviour is exponential (for all known pivot rules)

Klee-Minty cube example:

Consider d dimensional unit cube

(in 3 dimensions:



Described by

$$0 \le x_1 \le 1$$
$$0 \le x_2 \le 1$$
$$0 \le x_d \le 1$$

$2d$ inequalities, d variables, 2^d extreme points.

So number of vertices is exponential in number of constraints and variables

Consider perturbing cube slightly: $(0 < \varepsilon < \frac{1}{2})$

$$0 \le x_1 \le 1$$
$$\varepsilon x_{j-1} \le x_j \le 1 - \varepsilon x_{j-1} \quad j = 2, 3, \ldots, d$$

For $\varepsilon = 0$, this is the original cube.

In three dimensions, extreme points are

$(0, 0, 0)$
$(1, \varepsilon, \varepsilon^2)$
$(1, 1-\varepsilon, \varepsilon(1-\varepsilon))$
$(0, 1, \varepsilon)$
$(0, 1, 1-\varepsilon)$
$(1, 1-\varepsilon, 1-\varepsilon(1-\varepsilon))$
$(1, \varepsilon, 1-\varepsilon^2)$
$(0, 0, 1)$

} Each point is adjacent to the ~~two~~ ones on either side of it.

~~Those are adjacent~~

Obj. fn. max x_d

As the objective function increases as we move down the list ($0 < \varepsilon <$

So could visit all vertices while always improving objective function val

(choose one with max reduced cost to start)

To make Dantzig's rule ~~fail~~ cover every vertex to be visited,

need slightly more severe permutation of cube;

see Schrijver for details.

Expected behavior. $O(n)$ here n = smaller of # rows, # columns

Borgwardt, Haimovich.

How good is the simplex algorithm?

Can visit every bfs on the way to the optimal soln —

Klee-Minty, 1972.

$$\min \quad -\sum_{j=1}^{n} 10^{n-j} x_j$$

$$\text{s.t.} \quad \left(2 \sum_{j=1}^{i-1} 10^{i-j} x_j\right) + x_i \leq 100^{i-1} \quad i = 1, \ldots, n$$
$$x_j \geq 0$$

Simplex method takes $2^n - 1$ iterations.

See example for $n = 3$.

This example is for the largest reduced cost entering rule.

But, for all proposed rules, similar examples have been constructed.

In practice: for $\min c^T x$, $Ax = b$, $x \geq 0$, A $m \times n$,

$$\#\text{ its} \approx \frac{3m}{2}.$$

Expected behavior: (Borgwardt 1982) (Haimovich 1983)

Expected # its $\approx \min\{\frac{1}{2}n, m\}$.

Klee-Minty cube for $n=3$

0. $\min \quad -100x_1 - 10x_2 - x_3$

s.t. $x_1 + x_4 = 1$

$20x_1 + x_2 + x_5 = 100$

$200x_1 + 20x_2 + x_3 + x_6 = 10000$

$x_i \geq 0.$

1. x_1 enters, x_4 leaves:

$\min \quad -10x_2 - x_3 + 100x_4 \quad -100$

s.t. $x_1 + x_4 = 1$

$x_2 - 20x_4 + x_5 = 80$

$20x_2 + x_3 - 200x_4 + x_6 = 9800$

$x_i \geq 0$

2. x_2 enters, x_5 leaves:

$\min \quad -x_3 - 100x_4 + 10x_5 \quad -900$

s.t. $x_1 + x_4 = 1$

$x_2 - 20x_4 + x_5 = 80$

$x_3 + 200x_4 - 20x_5 + x_6 = 8200$

$x_i \geq 0$

3. x_4 enters, x_1 leaves:

$\min \quad 100x_1 - x_3 + 10x_5 \quad -1000$

s.t. $x_1 + x_4 = 1$

$20x_1 + x_2 + x_5 = 100$

$-200x_1 + x_3 - 20x_5 + x_6 = 8000$

$x_i \geq 0$

4. x_3 enters, x_6 leaves:

$$\min \quad -100x_1 \qquad -10x_5 + x_6 \qquad -9000$$

$$\text{s.t.} \quad x_1 \qquad + x_4 \qquad = 1$$

$$20x_1 + x_2 \qquad + x_5 \qquad = 100$$

$$-200x_1 \qquad + x_3 \qquad -20x_5 + x_6 = 8000$$

$$x_i \geq 0$$

5. x_1 enters, x_4 leaves:

$$\min \quad 100x_4 - 10x_5 + x_6 \qquad -9100$$

$$\text{s.t.} \quad x_1 \qquad + x_4 \qquad = 1$$

$$x_2 \qquad -20x_4 \quad (+x_5) \qquad = 80$$

$$x_3 \qquad +200x_4 \qquad -20x_5 + x_6 = 8200$$

$$x_i \geq 0$$

6. x_5 enters, x_2 leaves:

$$\min \quad 10x_2 \qquad -100x_4 \qquad + x_6 \qquad -9900$$

$$\text{s.t.} \quad x_1 \qquad + x_4 \qquad = 1$$

$$x_2 \qquad -20x_4 + x_5 \qquad = 80$$

$$20x_2 + x_3 \qquad -200x_4 \qquad + x_6 = 9800$$

$$x_i \geq 0$$

7. x_4 enters, x_1 leaves:

$$\min \quad 100x_1 + 10x_2 \qquad + x_6 \qquad -10000$$

$$\text{s.t.} \quad x_1 \qquad + x_4 \qquad = 1$$

$$20x_1 + x_2 \qquad + x_5 \qquad = 100$$

$$200x_1 + 20x_2 + x_3 \qquad + x_6 = 10000$$

$$x_i \geq 0.$$

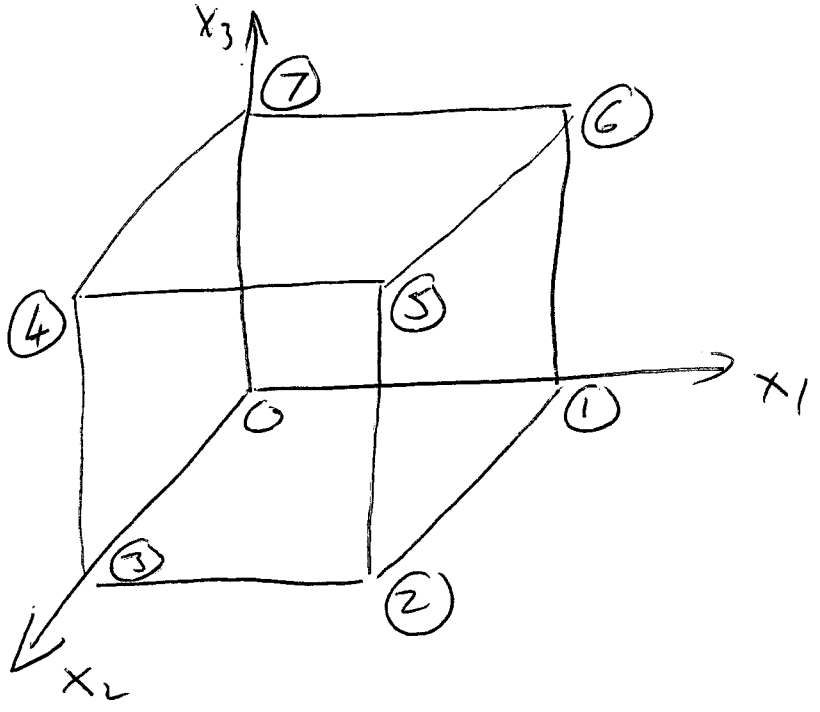
Optimal.

Notice: if x_3 had entered at Step 1 instead of x_1, then we would have solved the problem in one step.

The constraints are approximately the cube

$$0 \le x_1 \le 1$$
$$0 \le x_2 \le 100$$
$$0 \le x_3 \le 10000.$$

We start at the origin, then (approximately) visit the vertices
① (1, 0, 0)
② (1, 100, 0)
③ (0, 100, 0)
④ (0, 100, 10000)
⑤ (1, 100, 10000)
⑥ (1, 0, 10000)
⑦ (0, 0, 10000) - optimal.



Ellipsoid algorithm (Brief outline) (See NW p. 1~~4~~7, PS p 170ish)

Consider LP feasibility problem

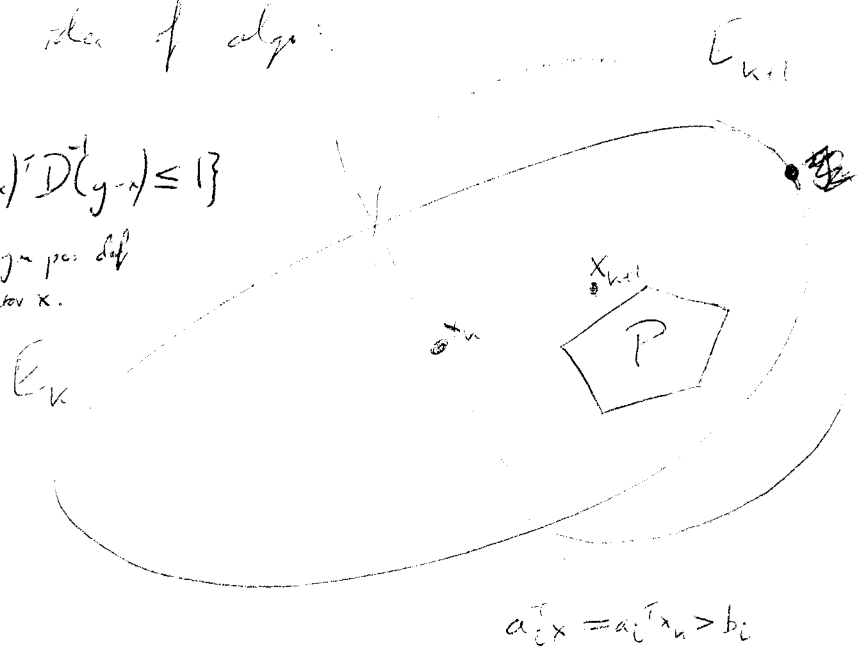
Note strict inequality. Can make $Ax \leq b$ into $Ax < b$ by perturbing RHS slightly

Does $\exists x$ satisfying $Ax \leq b$? A is $m \times n$.

Denote ith row of A by a_i (column vector). Let $P = \{x : Ax < b\}$. Assume P bounded.

Basic idea of algo:

$E = \{y : (y-x)^T D^{-1} (y-x) \leq 1\}$ for some ~~pos~~ sym pos def matrix D, vector x.



Have ellipsoid E_k containing feasible region, if it is nonempty. Check centre of ellipsoid. If it is feasible, done. Otherwise, it violates some constraint $a_i^T x < b_i$

Can then cut off half of ellipsoid and ~~replace~~ construct a new ellipsoid E_{k+1} which contains important half of current ellipsoid.

How does size of E_{k+1} relate to size of E_k ?

Thm Consider ellipsoid E, center x, ~~inequality~~

Let $H = E \cap \{y : d^T y \leq d^T x\}$ - half ellipsoid.

Then H is contained in an ellipsoid E' with the property

$$\frac{vol(E')}{vol(E)} \leq e^{-\frac{1}{2(n+1)}}.$$

(No proof)

So shrink ellipsoid at each iteration.

~~Eventually, ellipsoid center of ellipsoid~~ a sphere of radius R, volume V

Initial ellipsoid is drawn large enough so that it ~~boundary~~ contains P ~~feasible solution~~ ~~$Ax \le b$, if one exists.~~

Also, if $\exists$ sol'n to $Ax \le b$ there is a sphere of positive radius r contained in the feasible region. This sphere has volume v.

~~So, since ellipsoid contains~~ P shall contain a sphere of radius ~~if LP is feasible~~, will find a solution before vol of ellipsoid decreases below vol of sphere of radius r.

Note that if the volume of our ellipsoid is less than v, and we still haven't found a feasible point, then the LP is infeasible.

Since: ~~$v \le \text{vol}(P)$, $P \subseteq \text{ellipsoid} \Rightarrow$ vol~~

$$v > \text{vol}(\text{ellipsoid}) \ge \text{vol}(P) \quad \text{since } P \subseteq \text{ellipsoid}$$

$$\ge v \quad \text{provided } P \neq \phi.$$

$\ast\ast$

So we have a terminable criterion.

Let ~~t^*~~ $t^* = \lceil 2(n+1)\left(\log(V) - \log(v)\right) \rceil$

Then what is volume of t^* ellipsoid?

$$\text{vol}(E_{t^*}) \leq e^{-\frac{1}{2(n+1)}} \text{vol}(E_{t^*-1})$$

$$\leq \left[e^{-\frac{1}{2(n+1)}}\right]^{t^*} \text{vol}(E_0)$$

~~$\leq e$~~

$$= e^{-\frac{t^*}{2(n+1)}} \text{vol}(E_0)$$

$$\leq e^{-(\log(V)-\log(v))} V$$

$$= e^{-\log\left(\frac{V}{v}\right)} V = \frac{v}{V} \cdot V = v$$

See Prop 2.12, p. 158, N&W.

So need at most t^* iterations.

$\log(V)$ and $\log(v)$ can both be taken polynomial in length of data.

So also taken a polynomial number of iterations.

Work at each iteration:

Need to calculate new ellipsoid.

Requires calculating square roots. So can not calculate everything exactly.

So some subtle analysis is required to show that only can be implemented to only require a polynomial amount of work each iteration.

Extend to optimization.

POLYNOMIAL EQUIVALENCE OF SEPARATION & OPTIMIZATION.

Require definition of the separation problem

Given convex set C, a point x, $x \in C$? If not, find a hyperplane.

(Report MPLO (1988) 59-93.)