J. ホロムコヴィッチ,

計算困難問題に対するアルゴリズム理論

Springer, (訳: 和田幸一, 増澤利光, 元木光雄)

第3章 決定性アプローチ(後半)

2014.06.21

理論輪読会 #2-3

M2 若林由弥

発表内容

3.5 指数時間の最悪計算量の低減
- 基本概念
- 3SAT問題への適用

3.6 局所探索
- 近傍
- 局所探索スキーム
- Kernighan-Linの深さ可変探索アルゴリズム

3.7 線形計画法への緩和
-線形計画法
-ポリトープ
-シンプレックスアルゴリズム
-LP双対性
-プライマルデュアル法

1 指数時間の最悪計算量の低減とは

基本概念：「指数部」の計算量を改善すれば，困難問題が解けるようになる

計算量	$n = 10$	$n = 50$	$n = 100$	$n = 300$
2^n	1024	(16桁)	(31桁)	(91桁)
$2^{2^{\frac{n}{2}}}$	32	$\sim 33 \cdot 10^6$	(16桁)	(46桁)
$(1.2)^n$	7	9100	$\sim 29 \cdot 10^6$	(24桁)
$10 \cdot 2^{\sqrt{n}}$	89	1350	10240	$\sim 1.64 \cdot 10^6$
$n^2 \cdot 2^{\sqrt{n}}$	894	336000	$\sim 10.24 \cdot 10^6$	$\sim 14.8 \cdot 10^9$

□ 指数部

現状，現実的な指数時間アルゴリズムを設計するための
一般的な戦略を与えるような概念は存在しない

||

「どの困難問題が現実的な指数アルゴリズムを持てるか」
を説明するような理論が存在しない

2 決定問題 3SAT への適用

充足可能性問題(CNF-SAT問題)

CNF論理式 ψ が与えられたとき，ψ は充足可能になるか?

・CNF論理式(和積標準形, Conjunctive Normal Form)とは

論理式ψ が，
・$\psi = \psi_1 \wedge \psi_2 \wedge \psi_3 \wedge \cdots \wedge \psi_n$ の形で表現でき，
かつ，
・各 ψ_i (節)がリテラル（原始式又はその否定）の論理和である
とき，この形をCNF論理式(和積標準形)という.

・ψが充足可能とは

ψ を 1 にさせるような ψ の変数への 0/1 割り当てが存在するとき，
ψは充足可能である.

このうち，節の大きさが高々3であるSAT問題を3SAT問題と呼ぶ.

例)

$$\psi = x_1 \wedge (x_1 \vee \overline{x_2}) \wedge (x_1 \vee x_2 \vee \overline{x_3}) \wedge (x_2 \vee x_3 \vee \overline{x_4}) \wedge (x_3 \vee x_4 \vee \overline{x_5}) \wedge (x_3 \vee x_4 \vee x_5)$$

は，$x_1 = x_2 = x_3 = x_4 = x_5 = 1$のとき，充足可能になる

3 決定問題 3SAT への適用

決定問題 $(3SAT, \sum logic)$について考える

= 3CNF式Fが充足可能であるかを決定する

分割統治法を利用して，$O(|F| \cdot 2^n)^{16}$ の計算量を$O(|F| \cdot 1.84^n)$に改善可能

lをFに現れるリテラルとする．
この時，$F(l=1)$と$F(l=0)$を以下の手順で求める．

$F(l=1)$：

1. リテラル l を含む全ての節を F から取り除く
2. F の節がリテラル $\bar{l}$ を含み，さらに $\bar{l}$ と異なるリテラルを少なくとも1つ含むなら，
   $\bar{l}$ をその節から取り除く．
3. Fの節がリテラル $\bar{l}$ のみから成り立っているならば，$F(l=1)$ は式0となる．

$F(l=0)$：

1. リテラル $\bar{l}$ を含む全ての節を F から取り除く
2. F の節がリテラル lを含み，さらに lと異なるリテラルを少なくとも1つ含むなら，
   l をその節から取り除く．
3. Fの節がリテラル l のみから成り立っているならば，$F(l=0)$ は式0となる．

4 決定問題 3SAT への適用

例) $F = (x_1 \vee \overline{x_2} \vee x_4) \wedge (\overline{x_2}) \wedge (\overline{x_2} \vee \overline{x_3} \vee x_5) \wedge (x_1 \vee \overline{x_5}) \wedge (\overline{x_1} \vee x_2 \vee x_3)$ に対して，

$$F(x_1 = 1) = (\overline{x_2}) \wedge (\overline{x_2} \vee \overline{x_3} \vee x_5) \wedge (x_2 \vee x_3)$$
$$F(\overline{x_2} = 1) = (x_1 \vee \overline{x_5}) \wedge (\overline{x_1} \vee x_3)$$
$$F(\overline{x_3} = 0) = (x_1 \vee \overline{x_2} \vee x_4) \wedge (\overline{x_2}) \wedge (\overline{x_2} \vee x_5) \wedge (x_1 \vee \overline{x_5})$$

一般に，リテラル $l_1, l_2, \dots, l_c, h_1, \dots, h_d$ に対して，

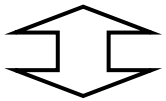
$$F(l_1 = 1, l_2 = 1, \dots, l_c = 1, h_1 = 0, h_2 = 0, \dots, h_d = 0)$$

は，F から

$$F(l_1 = 1), F(l_1 = 1)(l_2 = 1), \dots$$

というように順番に適用して得られる．

$F(l_1 = 1, l_2 = 1, \dots, l_c = 1, h_1 = 0, h_2 = 0, \dots, h_d = 0)$が充足可能



$l_1 = 1, l_2 = 1, \dots, l_c = 1, h_1 = 0, h_2 = 0, \dots, h_d = 0$ かつ F を満たす割り当てが存在する

5 決定問題 3SAT への適用

全ての正整数 n, r について，以下のように定義する

$$3\mathrm{CNF}(n, r) = \{\Phi | \Phi \text{は高々 } n \text{ 変数の3CNF式であり, } \Phi \text{ は高々 } r \text{ 節を含む}\}$$

ある正整数 n, r に対して，$F \in 3\mathrm{CNF}(n, r)$ とし，
F の1つの節が$(l_1 \vee l_2 \vee l_3)$ であるとすると，以下が成り立つ

F が充足する $\Longleftrightarrow$ $F(l_1 = 1), F(l_1 = 0, l_2 = 1), F(l_1 = 0, l_2 = 0, l_3 = 1)$ の少なくとも1つが充足する

$F(l_1 = 1) \in 3\mathrm{CNF}(n - 1, r - 1)$

$F(l_1 = 0, l_2 = 1) \in 3\mathrm{CNF}(n - 2, r - 1)$

$F(l_1 = 0, l_2 = 1, l_3 = 1) \in 3\mathrm{CNF}(n - 3, r - 1)$

より，$3\mathrm{CNF}(n, r)$の式Fが充足可能かどうかは，

$$3\mathrm{CNF}(n - 1, r - 1), 3\mathrm{CNF}(n - 2, r - 1), 3\mathrm{CNF}(n - 3, r - 1)$$

における充足可能性問題に帰着される

6 決定問題 3SAT への適用

以上より，3SATに対しては次の再帰的なアルゴリズムが得られる

input : 3CNFの式F

Step1

if ある $m, k \in N - \{0\}$ に対して $F \in \mathrm{3CNF}(3, k)$ または $F \in \mathrm{3CNF}(m, 2)$
then $F \in \mathrm{3SAT}$ かどうかを F の変数への全ての割当をテストし決定
if $F \in \mathrm{3SAT}$
output(1)
else
output(0)

Step2

HをFの中で最も短い節の1つとする.

$D\&C$: 分割統治法

if $H = (l)$
then output $D\&C\_3SAT\big(F(l = 1)\big)$
if $H = (l_1 \vee l_2)$
then output $D\&C\_3SAT\big(F(l_1 = 1)\big) \vee D\&C\_3SAT\big(F(l_1 = 0, l_2 = 1)\big)$
if $H = (l_1 \vee l_2 \vee l_3)$
then output $D\&C\_3SAT\big(F(l_1 = 1)\big) \vee D\&C\_3SAT\big(F(l_1 = 0, l_2 = 1)\big)$
$\vee D\&C\_3SAT\big(F(l_1 = 0, l_2 = 0, l_3 = 1)\big)$

7 局所探索：近傍と近傍グラフ

近傍

実行可能解を$\mathrm{M}(I)$とし，集合内の各実行可能解の近傍を以下のように定義する

$U = (\Sigma_I, \Sigma_O, L, L_I, \mathrm{M}, cost, goal)$ を最適化問題とする．
各$x \in L_I$に対して，$\mathrm{M}(x)$の近傍は，以下の条件が成り立つような任意の写像

$$f_x : \mathrm{M}(x) \to Pot(\mathrm{M}(x))$$

である．

(i) 各 $\alpha \in \mathrm{M}(x)$ に対して，$\alpha \in f_x(\alpha)$
(ii) ある $\alpha \in \mathrm{M}(x)$ に対して，$\beta \in f_x(\alpha)$ ならば $\alpha \in f_x(\beta)$
(iii) 全ての $\alpha, \beta \in \mathrm{M}(x)$ に対して，正整数kと$\gamma_1, \gamma_2, \ldots, \gamma_k \in \mathrm{M}(x)$が存在して，
各$i = 1, \ldots, k-1$について$\gamma_1 \in f_x(\alpha), \gamma_{i+1} \in f_x(\gamma_i)$，かつ$\beta \in f_x(\gamma_k)$

このとき，集合$f_x(\alpha)$を$\mathrm{M}(x)$における実行可能解αの近傍という
$\alpha, \beta \in \mathrm{M}(x)$に対して，$\alpha \in f_x(\beta)$が成り立つとき，$\alpha$と$\beta$は隣接しているという

近傍グラフ

$$G_{\mathrm{M}(x), f_x} = \left(M(x), \left\{\{\alpha, \beta\} \mid \alpha \in f_x(\beta), \alpha \neq \beta, \alpha, \beta \in \mathrm{M}(x)\right\}\right)$$

を近傍f_xに関する$\mathrm{M}(x)$の近傍グラフと呼ぶ

8 局所変換と局所最適解

一般的には，$\mathrm{M}(x)$の近傍を定義したい場合，関数や関係は定式化せず，
$\mathrm{M}(x)$上の局所変換を用いる

||

実行可能解αからαの仕様を局所的に変更して別の実行可能解βに変換する

例)
- 最小全域木に対する局所探索アルゴリズムにおける，2辺の交換
- 巡回セールス問題における，2-Exchangeや3-Exchange(後述)

以上より，局所最適解は以下のように定義される

最適化問題 $U = (\Sigma_I, \Sigma_O, L, L_I, \mathrm{M}, cost, goal)$ について，
各$x \in L_I$に対して，関数f_xを$\mathrm{M}(\mathrm{x})$の近傍とすると，実行可能解$\alpha \in \mathrm{M}(x)$は

$$cost(\alpha) = goal\{cost(\beta) \mid \beta \in f_x(\alpha)\}$$

を満たすとき，f_xに関してUの入力インスタンスxに対する局所最適解であるといい，xに対する全ての局所最適解の集合を$LocOPT_U(x, f_x)$で表す

9 局所探索スキーム

任意の$x \in L_I$に対する近傍$Neigh_x$によって決定されるM(x)上の構造が存在する時，局所探索の一般的なスキームは以下のように表される．

$LSS(Neigh)$: 近傍$Neigh$に関する局所探索スキーム

入力 : 最適化問題Uの入力インスタンス

STEP1 : ある実行可能解$\alpha \in$ M(x)を見つける

STEP2 : **while** $\alpha \notin LocOPT_U(x, Neigh_x)$ **do**

・Uが最小化問題なら$cost(\beta) < cost(\alpha)$

・Uが最大化問題なら$cost(\beta) > cost(\alpha)$

となる$\beta \in Neigh_x(\alpha)$を見つける．

$\alpha := \beta$

end

出力 : output(α)

局所探索アルゴリズムが上手く動作するかは，
近似をどのように選ぶかに依存している

近傍が小さい : 近傍探索の効率が良くなるが，局所最適に陥る
近傍が大きい : whileループの1回の計算量が大きくなりすぎる

10 局所探索スキーム

$LSS(Neigh)$に対して，局所探索をうまく行うことに
影響を及ぼしうる2つのパラメータ

1. 初期解の選び方

・ランダムに初期解を選択する方法 ←―― 多スタート局所探索
・他のアルゴリズムを用いた前処理計算を行う方法

があるが，選び方の一般的な理論は存在しない．

2. 近傍解の選び方

実行可能解αの近傍$Neigh_x(\alpha)$の中で，
・最初に現れたコストを改善する解を選択する
・最もコストを改善する解を選択する

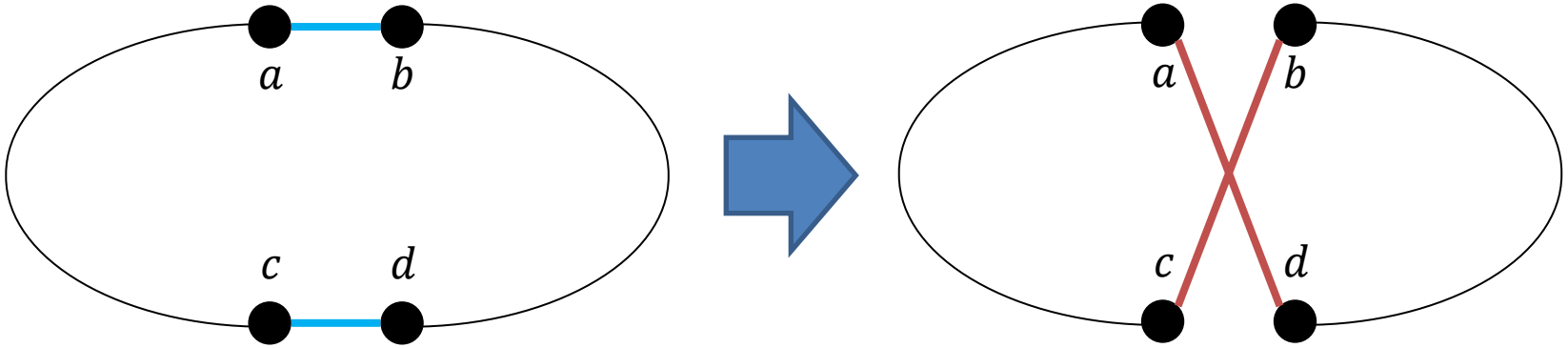
前者はwhileループ1回の計算量は減るが，ループの回数が多くなる
後者はwhileループの回数は減るが，ループ1回の計算量が増える
また，同じ初期解でも違う結果をもたらしうる．

11 近傍の例

巡回セールス問題(TSP)における2-Exchangeと3-Exchange

2-Exchange

ハミルトン閉路αから，$|\{a,b,c,d\}| = 4$となる2つの辺$\{a,b\}$と$\{c,d\}$を取り除き，2つの辺$\{a,d\}$と$\{b,c\}$をαに付け加える



TSPの任意の入力インスタンス(K_n, c) $(n \in N - \{0\})$，および任意の$\alpha \in \mathrm{M}(K_n, c)$に対して，近傍2-Exchange($\alpha$)のサイズは$\frac{n\cdot(n-3)}{2}$であり，計算量は$\Omega(n^2)$のオーダーになる．

3-Exchangeも同様に3つの辺の交換によって定義できるが，
1つの辺を元に戻すパターンもあることに注意する

12 深さ可変探索アルゴリズム

$Neigh$を局所変換によってできる最適化問題Uの近傍とする.
各正整数kとUの入力インスタンスIに対する任意の$\alpha \in \mathrm{M}(I)$に対して,

$$Neigh_I^k(\alpha) = \{\beta \in \mathrm{M}(I) | distance_{Neigh_I}(\alpha, \beta) \leq k\}$$

をαから局所変換を高々k回適用して得られる実行可能解の集合とする.

この時，各$\alpha \in \mathrm{M}(I)$に対して

$$Neigh_I^m(\alpha) = \mathrm{M}(I)$$

となるようなmが存在する(通常は，mは仕様の長さnにほぼ等しい)

以降で述べる貪欲戦略により,
$Neigh_I^n(\alpha)$から解βを効率よく求めることができる

任意の最小化問題Uと，Uの任意のインスタンスIに対する
全ての実行可能解$\alpha, \beta \in \mathrm{M}(I)$に対して,

$$gain(\alpha, \beta) = cost(\alpha) - cost(\beta)$$

を定義する

13 深さ可変探索アルゴリズム

$Neigh$の局所探索に陥ることを回避するために，以下のような
局所変換を高々n回行う

1. 実行可能解が$\alpha = (p_1, p_2, \ldots, p_n)$から始まるならば，得られる実行可能解
$\gamma = (q_1, q_2, \ldots, q_n)$は全ての$i = 1, \ldots, n$に対して$q_i \neq p_i$が成り立つ

2. $\alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_m$ $(\alpha = \alpha_0, \alpha_m = \gamma)$を得られる実行可能解の列とすると，

(a) $gain(\alpha_i, \alpha_{i+1}) = \max\{gain(\alpha_i, \delta) | \delta \in Neigh(\alpha_i)\}$ 貪欲戦略

(b) α_iからα_{i+1}を得るステップで，初期実行解αのパラメータp_jがq_jに
変更されるとき，それ以降q_jは変更されない

2の処理において，たとえ数回 $gain$ が負になる方向に進んでも，
最終的には多くの正しい方向へのステップによって回復できる

14 深さ可変探索アルゴリズム

入力 : 最適化問題Uの入力インスタンスI

STEP1 : 実行可能解$\alpha = (p_1, p_2, \ldots, p_n) \in \mathrm{M}(I)$を生成する

STEP2: IMPROVEMENT := TRUE;
EXCHANGE := $\{1,2, \ldots, n\}$; ← 変更可能な解の位置を表すベクトル
$J := 0$; $\alpha_J := \alpha$;
while IMPROVEMENT = TRUE **do begin**
while EXCHANGE $\neq \phi$ **do begin**
$J := J + 1$;
$\alpha_J := Neigh(\alpha_{J-1})$からの解.
EXCHANGE := EXCHANGE - {α_Jとα_{J-1}で変更したパラメータの位置}
end;
各$i = 1,2, \ldots, J$に対して，$gain(\alpha, \alpha_i)$を計算する
$gain(\alpha, \alpha_l) = \max\{gain(\alpha, \alpha_i) | i \in \{1,2, \ldots, J\}\}$
if $gain(\alpha, \alpha_l) > 0$ **then begin**
$\alpha := \alpha_l$
EXCHANGE := $\{1,2, \ldots, n\}$
end
else IMPROVEMENT := FALSE
end

$gain(\alpha_{J-1}, \alpha_J) = \max\{gain(\alpha_{J-1}, \delta) | \delta \in Neigh(\alpha_{J-1})\}$

STEP3: **output**(α)

15 線形計画法への緩和

基本的な概念：
- 線形計画法(LP)は多項式時間で解ける
- 0/1線形計画法(0/1-LP)や整数計画法(IP)はNP困難

0/1-LPやIPの問題インスタンスをLPのインスタンスとして解けるように条件を緩和する

線形計画法への緩和

以下の3STEPから構成される

(1) 帰着

最適化問題Uの与えられたインスタンスxを
0/1-LPまたはIPの入力インスタンス$I(x)$として表現する

(2) 緩和

$I(x)$をLPの入力インスタンスと考えて，線形計画法の
アルゴリズムによって$I(x)$に対する最適解αを計算する

(3) 元々の問題を解く

αを利用して，元々の問題の最適解を計算する．
または，元々の問題に対する精度の高い実行可能解を見つける

16 線形計画法の標準形

LPの(等式)標準形

実数上の任意の入力インスタンス

$$A = [a_{ji}]_{j=1,\dots,m,i=1,\dots,n}$$
$$b = (b_1, \dots, b_m), \quad c = (c_1, \dots, c_n)$$

に対して，制約

$$A \cdot X = b \leftrightarrow \sum_{i=1}^{n} a_{ji} x_i = b_j \ (j = 1, \dots, m) \qquad x_i \geq 0 \ (i = 1, \dots, n)$$

の下で

$$X \cdot c^T = \sum_{i=1}^{n} c_i x_i$$

を最小化する問題

LPにはいくつかの形があるが，全てこの標準形で表現することが可能

17 線形計画法の標準形

LPの不等式標準形 → LPの等式標準形

LPの不等式標準形　入力インスタンスA, b, cに対して

$$A \cdot X \leq b \leftrightarrow \sum_{i=1}^{n} a_{ji} x_i \leq b_j \ (j = 1, \dots, m) \quad \text{かつ} \quad x_i \geq 0 \ (i = 1, \dots, n)$$

の制約の下で

$$X \cdot c^T = \sum_{i=1}^{n} c_i x_i$$

を最小化する問題

新しい変数s_j(余裕変数)を導入する

$$\sum_{i=1}^{n} a_{ji} x_i + s_j = b_j \quad \text{かつ} \quad s_j \geq 0$$

ここで，

$$B = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} & 1 & 0 & \cdots & 0 \\ a_{21} & a_{22} & \vdots & a_{2n} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & 0 & 0 & \cdots & 1 \end{pmatrix} \qquad d = (c_1, \dots, c_n, 0, \dots 0)$$

とおけば，入力インスタンスB, b, dの等式標準形になる

18 線形計画法の標準形

具体例：

ナップサック問題

$\omega_1, \omega_2, \ldots, \omega_n, c_1, c_2, \ldots, c_n, b$ をインスタンスとする

n個のブール変数：$x_1, x_2, \ldots, x_n$

※x_i=1はi番目の荷物がナップサックに詰められることを表す

この時，問題は

$\sum_{i=1}^{n} \omega_i x_i \leq b$　かつ　$x_i \in \{0,1\}\ (i = 1, \ldots, n)$

の制約の下で，

$\sum_{i=1}^{n} c_i \cdot x$

を最大化すること，すなわち

$\sum_{i=1}^{n} (-c_i) \cdot x$

を最小化することである

$x_i \in \{0,1\}$を$x_i \geq 0$に緩和すれば
LPの不等式標準形が得られる

19 半空間

nを正整数とする．次元$n-1$のR^nのアフィン部分空間は超平面と呼ばれる

逆に，全てのaは0とはならないようなある$a_1, a_2, \ldots, a_n, b$が存在し，

$$a_1x_1 + a_2x_2 + \cdots + a_nx_n = b$$

を満たす全ての$X = (x_1, \ldots, x_n)^T \in R^n$の集合が，$R^n$における超平面である

また，集合

$$HS_{\geq}(a_1, \ldots, a_n, b) = \{X = (x_1, \ldots, x_n)^T \in R^n | \sum_{i=1}^{n} a_i x_i \geq b\}$$

$$HS_{\leq}(a_1, \ldots, a_n, b) = \{X = (x_1, \ldots, x_n)^T \in R^n | \sum_{i=1}^{n} a_i x_i \leq b\}$$

を半空間と呼ぶ

R^nにおける任意の半空間はR^nと同じ次元nを持ち，凸集合である．
よって，

$$\{X \in R^n | A \cdot X \leq b\} = \cap_{j=1}^{m} HS_{\leq}(a_{j1}, \ldots, a_{jn}, b_j)$$

は凸集合であり，$Polytope(AX \leq b)$と表す

20 ポリトープ

nを正整数とする.
R^n における有限個の半空間の共通部分は R^n の(凸)ポリトープである.

与えられた制約

$$\sum_{i=1}^{n} a_{ji}x_i \leq b_j \ (j = 1, \ldots, m) \qquad x_i \geq 0 (i = 1, \ldots, n)$$

に対して

$$Polytope(AX \leq b, X \geq 0_{n\times 1}) = \{x \in (R^{\geq 0})^n | A \cdot X \leq b\}$$

例) R^nにおける次のような制約を考える

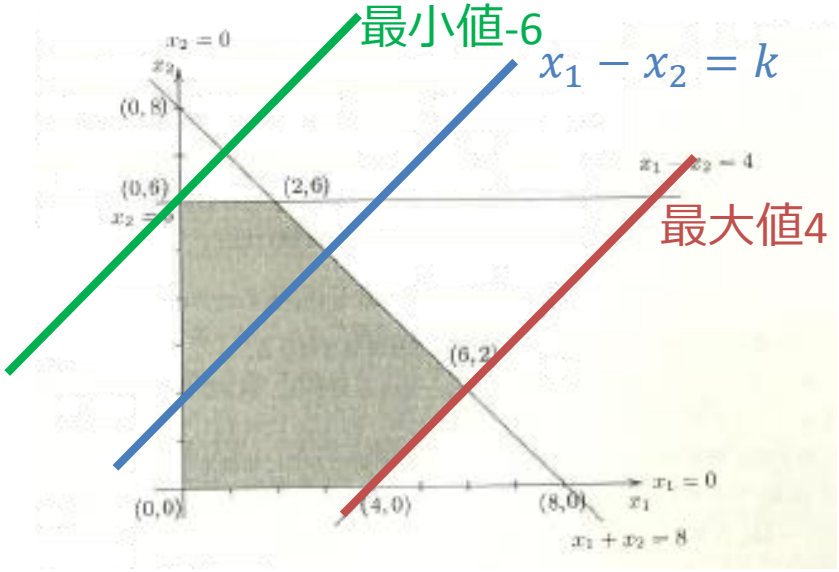
$$x_1 + x_2 \leq 8$$

$$x_2 \leq 6$$

$$x_1 - x_2 \leq 4$$

$$x_1 \geq 0$$

$$x_2 \geq 0$$



これらの制約に対して$x_1 - x_2$を最小化，最大化することを考える

21 ポリトープ

線形計画問題
＝与えられたポリトープと与えられた超平面に対して超平面をポリトープに沿って移動したとき，ポリトープと超平面の最後の共通点となるポリトープの表面上の点を見つける問題

dとnを正整数とする.
AをR^nにおける次元dの凸ポリトープとする.
HをR^nの超平面とし，HSをHによって定義される半空間とする

$A \cap HS \subseteq H$ならば，$A \cap HS$はAの面と呼ばれ，
Hは$A \cap HS$を定義する支持超平面という

この表現($A \cap HS \subseteq H$)によって
面，辺，頂点を表すことができる

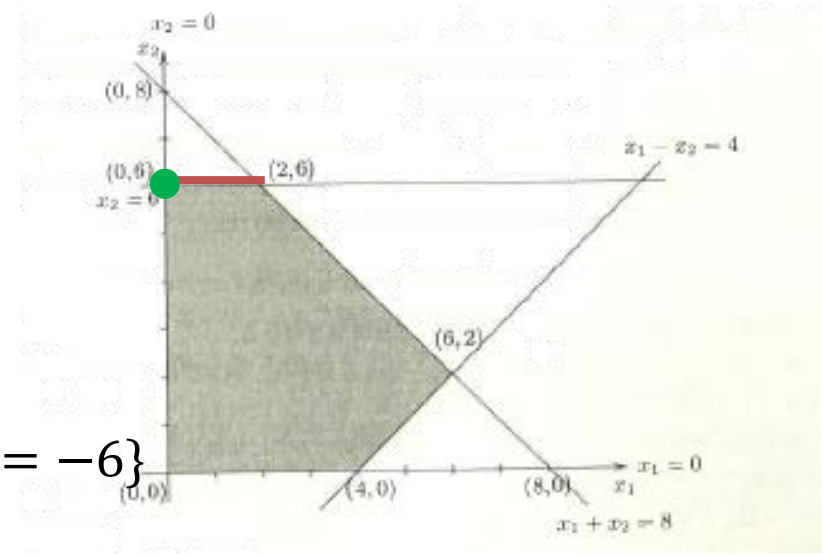
例)

$A \cap HS_{\geq}(0,1,6) \subseteq \{(x_1, x_2)^T \in R^2 | x_2 = 6\}$

→(0,6)と(2,6)を結ぶ辺

$A \cap HS_{\leq}(1,-1,-6) \subseteq \{(x_1, x_2)^T \in R^2 | x_1 - x_2 = -6\}$

→頂点(0,6)



22 シンプレックスアルゴリズム

局所探索アルゴリズムの1つ.
近傍$Neigh$はポリトープ上の辺で結ばれる頂点によって定義される.

入力 : 不等式標準形におけるLPの入力インスタンス

STEP1 : $Polytope(AX \leq b, X \geq 0)$の頂点$X$を見つける

STEP2 : **while** Xが$Neigh$に関して局所最適解でない **do**
$c^T \cdot X > c^T \cdot Y$ならば$X$を頂点$Y \in Neigh_{(A,b,c)}(X)$で置き換える

STEP3 : **output**(X)

シンプレックスアルゴリズムが局所最適解に到達するまでの
最悪時間計算量は，指数時間である
(頂点数が制約の数のサイズの指数になるため)

実際に最悪時間計算量かかることは稀で，
平均時間計算量は多項式時間アルゴリズムより優れているため，
好んで用いられることが多い

23 LP双対性

与えられた最適化問題Uの双対問題$Dual(U)$を探索する

Uが最小化問題の時，$Dual(U)$は以下の性質を持たなければならない

1. $Dual(U)$はUの各インスタンスIを$Dual(U)$のインスタンス$Dual(I)$に変換して得られる「最大化」問題である.

2. Uの各インスタンスIに対して，Uの任意の実行可能解のコストは$Dual(U)$より小さくはない.

全ての$\alpha \in \mathrm{M}(I)$と全ての$\beta \in \mathrm{M}(Dual(I))$に対して

$$cost(\alpha) \geq cost(\beta)$$

2. Uの任意のインスタンスIに対して，

$$Opt_U(I) = Opt_{Dual(U)}(Dual(I))$$

$$Opt_U(I) = Opt_{Dual(U)}(Dual(I))$$

24 プライマルデュアルスキーム

LP双対性に基づいた計算法

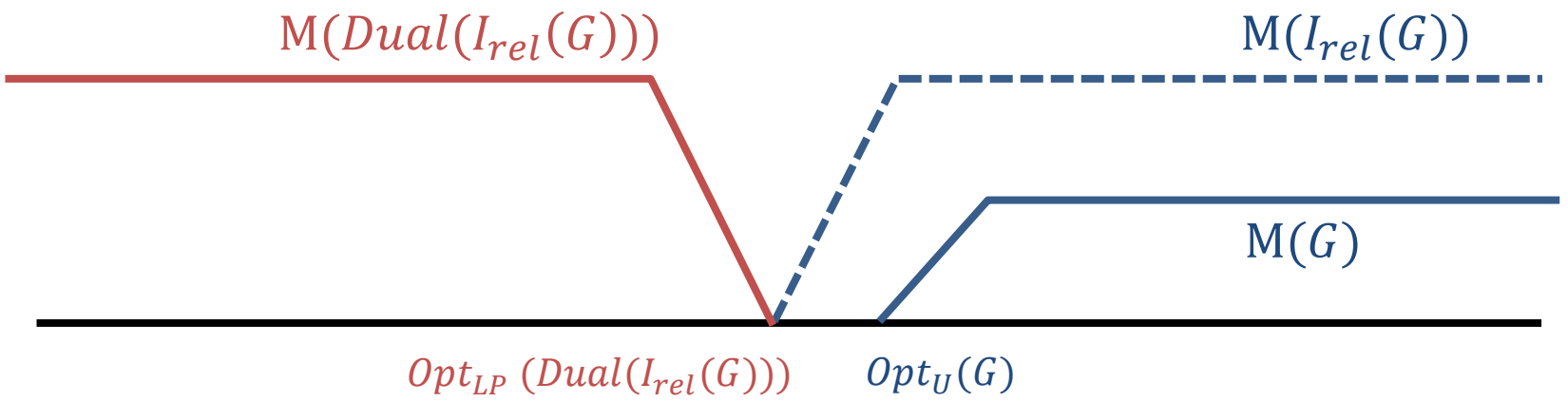
入力 : 最適化問題Uの入力インスタンスG

STEP1 : GをIPのインスタンスI(G)として表現し，
$I(G)$をLPのインスタンス$I_{rel}(G)$に緩和する

STEP2 : LPのインスタンスとして$Dual(I_{rel}(G))$を構成

STEP3: LPのインスタンスとして$Dual(I_{rel}(G))$を解く
同時に,IP のインスタンスとして$I(G)$を解く

出力 : $I(G)$に対する実行可能解αと$Opt_{LP}(Dual(I_{ref}(G)))$



25 まとめ

・NP困難な問題に対するアプローチとして

  ・指数時間の最悪計算量の低減

  ・局所探索

  ・線形計画法への緩和

の3つのアプローチを紹介した.

・いずれのアプローチも計算量を抑えることを目的としており,
計算量と精度のトレードオフについて十分に理解する必要がある.