日本オペレーションズ・リサーチ学会
2004年秋季研究発表会

計算グラフと分枝限定法を利用した大域的最適化—2

（株）数理システム　＊逸見　宣博　HENMI Nobuhiro
01701240　（株）数理システム　　山下　浩　　YAMASHITA Hiroshi

1. はじめに

本発表では，問題：

$$\text{最小化} \quad f(x), \quad x \in X_0 \subset \mathbf{R}^n \qquad (G_0)$$

の大域的最適解を数値的に求めるためのアルゴリズムとその実装について述べる．問題の形式を特に限定しないで大域的最適解を得るアルゴリズムの構築が目的である．アルゴリズムの基本的アイディアについては [1,2] で発表されているが，ここでも簡単に解説する．本方法は

(i) 元の問題を計算グラフによって表現して凸緩和問題を生成する．

(ii) 上記の凸緩和問題を空間的に分割することによって部分問題を生成して，分枝限定法のアルゴリズムを適用する．

という二つの基本的ブロックから成っている．

問題 (G_0) の緩和問題を

$$\text{最小化} \quad f_0(x), \quad x \in \overline{X}_0 \subset \mathbf{R}^n \qquad (C_0)$$

とする．凸緩和問題は次の性質を持っている．

(i) (C_0) が許容解を持たないとき，(G_0) も許容解を持たない．

(ii) (C_0) の最適値を f_0^* としたとき，(G_0) の最適値 f^* とは $f_0^* \leq f^*$ の関係がある．

問題の制約領域 X_0 が部分領域 $X_1, X_2, ...$ に分割されたとき，それらに対応する問題：

$$\text{最小化} \quad f(x), \quad x \in X_k \subset \mathbf{R}^n \qquad (G_k)$$

を子問題という．子問題 (G_k) の緩和問題を

$$\text{最小化} \quad f_k(x), \quad x \in \overline{X}_k \subset \mathbf{R}^n \qquad (C_k)$$

と定義する．

2. 分枝限定法

[1] のアルゴリズムの記述に一部誤りがあったので以下で再掲する．

[アルゴリズム]

ステップ 0. $\overline{z} = +\infty, \mathcal{P} = \{(G_0)\}, \epsilon > 0.$

ステップ 1.

● $\mathcal{P} \neq \emptyset$ ならばステップ 2 へ．

● $\mathcal{P} = \emptyset$ ならば停止．($\overline{z} = +\infty$ ならば問題 (G_0) には許容解が存在しない．$\overline{z} < +\infty$ ならば $\overline{x}$ が最適解，$\overline{z}$ が最適値となる．)

ステップ 2. （部分問題選択）$\mathcal{P}$ の中から問題を一つ選び，それを (G_k) とする．$\mathcal{P} = \mathcal{P} \setminus \{(G_k)\}$.

ステップ 3. 緩和問題 (C_k) を定義し，それを解いて最適解 $\widehat{x}_k$，最適値 $\widehat{f}_k$ を求める．

● （分枝停止）許容解がないか，$\widehat{f}_k \geq \overline{z} - \epsilon$ ならばステップ 1 へもどる．

● $\widehat{x}_k \in X_k$ ならばステップ 4 へ．

● $\widehat{x}_k \notin X_k$ ならばステップ 5 へ．

ステップ 4.

● （分枝停止）$f_0(\widehat{x}_k) \geq \overline{z} - \epsilon$ ならばステップ 1 へもどる．

● （上界値更新）$f_0(\widehat{x}_k) < \overline{z}$ ならば $\overline{z} = f_0(\widehat{x}_k), \overline{x} = \widehat{x}_k$ とする．

ステップ 5. （分枝）(G_k) の子問題を作り，$\mathcal{P}$ に加える．ステップ 1 へもどる． □

上界値として部分問題の局所最適界が利用できる．

3. 緩和問題

問題を

$$(1) \quad \begin{array}{ll} \text{最小化} & f(x), \quad x \in \mathbf{R}^n, \\ \text{条　件} & g_{iL} \leq g_i(x) \leq g_{iU}, i = 1, ..., m \\ & x_L \leq x \leq x_U \end{array}$$

と書く．有界な領域を扱うために，変数に対する（有界な）上下限が常に存在することを前提とする．本発表では凸緩和として線形計画緩和を採用する．したがって以下では，連続な非線形関数である目的関数 $f(x)$ と制約条件 $g_{iL} \leq g_i(x) \leq g_{iU}, i = 1, ..., m$ の線形計画緩和について考える．

問題が計算グラフの形に表現されているということは，すべての計算過程が 2 項演算 (例：$x \times y$)，単項演算 (例：x^2)，関数評価（例：$\sin(x)$）の素過程の連なりとして表現されていることを意味する．以下では，そのような計算グラフを利用した緩和問題の生成について述べる．$x_1, ..., x_n$ の任意の関数 f の計算法が計算グラフで与えられているとする．計算グラフの各ノード（1 項あるいは 2 項演算の結果）に中間変数 $x_{n+1}, ..., x_{n+p}$ を

設定する．関数 f の計算は

$$
\begin{aligned}
w &= a \times (x \times y) && \text{(積)} \\
w &= a \times (x \div y) && \text{(商)} \\
w &= a \times (x + y) && \text{(和)} \\
w &= a \times (x - y) && \text{(差)} \\
w &= func(x), func(x,y) && \text{(関数)}
\end{aligned}
$$

などの演算の連鎖となる．ここで，w は中間変数，x と y は $x_1, ..., x_n$ のどれか，あるいは中間変数である．a は定数である．グラフのトップは上記の w が関数 f となった表式である．そして，すべての中間変数を最適化問題の変数として扱い，上記のような中間変数の定義式を等式条件として扱う．新たに定義された最適化の変数全体とその空間を $z, \mathbf{R}^N$ とすると，問題 (1) は

$$
\begin{array}{ll}
\text{最小化} & z_1, \quad z \in \mathbf{R}^N, \\
\text{条　件} & z_i = h_i(z), i = 1, ..., M \\
 & z_L \leq z \leq z_U
\end{array} \tag{2}
$$

と書くことが出来る．関数 h_i は上記の五つのタイプのどれかである．したがって，緩和問題の生成は制約条件 $z_i = h_i$ の線形緩和となる．ここでは，具体例は省く．

4. 実装

以下に実装に関する事柄の要点を幾つか述べる．

(1) 関数　モデリング言語 SIMPLE で使用可能な関数の中で以下のものが現在使用可能になっている．すなわち，以下の関数を含む最適化問題に対して自動的に線形凸緩和が得られる．実用上重要な関数はほぼ網羅されていると考えている．

log, exp, sqrt, cbrt, pow, sin, cos, tan, cot,
sinh, cosh, tanh, coth, ifelse, min, max,
fabs, ceil, floor

(2) 分枝変数の選択　子問題 (G_k) の緩和問題 (C_k) を解いて，最適解 $\hat{z}_k$ が得られたとする．このとき，問題 (2) の最適値の最小値を与える問題を分割する．変数 $z_i, i = 1, ..., N$ の中から以下のような基準を基にして分枝変数を選択する．

(a) 適当なタイミングで，変数の領域を 2 分する．元の領域巾に比べて現在の領域巾の減少度合いが少ないものを選択．

(b) 「制約条件の侵犯」に最も寄与していると思われる変数：たとえば，$w = xy$ という制約式の緩和条件の場合は，$\|\hat{w} - \hat{x}\hat{y}\|$ という量を変数 w, x, y の与える侵犯量とする．すべての緩和制約条件の侵犯量を変数毎に足し合わせてスコア化する．

(3) 領域の境界近くでの数値的困難　本方法は，与えられた領域の中で確実に大域的最適解を得るという目的のために対象領域を有限なものに制限している．その際に，問題の中に現れる表式が領域の境界近くで非常に大きな値を取ることがある．(このような領域に近似反復解が近づくことは，通常の局所的最適化法では起こらない.) なるべくタイトな線形緩和を求めるために部分領域の端点での接線の情報を利用することが多いが，そうすると接線の傾きが非常に大きな値を持ち，得られる線形計画緩和問題が悪条件になり，意味のある解を得ることが困難になる．このような状況では，タイトな緩和を諦めて矩形領域で近似することが良い方策となる．

(4) 要素的式の発散の問題　上記の問題と類似であるが，問題の中に現れる各種表式の中にたとえば $1/x$ や $\tan x$ などのように有界な点でも無限大の値を取ることがある．そのような点の周辺の領域を自動的に排除するような手続きが必要になる．

5. 数値実験

現在，実際の応用に基づいた問題，大域的最適化の各種テスト問題 ([3])，Hock and Schittkowski 全問題などをテストの対象にして実験を行っている．詳細は当日報告するが，以下に幾つかの結果を示す．使用計算機は Pentium IV 1.5GHz, 1GB メモリ．OS は Redhat Linux 7．分枝限定法の停止条件は，「緩和問題の最適解の（元の問題に対する）制約侵犯が 10^{-8} 以下になるまで」としている．

・Reactor Network Design （6 変数，5 制約，双 1 次，平方根など）

LP	最適値	単体法	時間（秒）
28 × 36	-0.3746104606	18	0.02

・Separation Network Synthesis （23 変数，16 制約，双 1 次など）

LP	最適値	単体法	時間（秒）
65 × 84	1.257359459	36	0.07

・HS81（5 変数，3 制約，指数関数，5 次多項式など）

LP	最適値	単体法	時間（秒）
56 × 73	0.05394983602	11092	33.14

・HS104（8 変数，6 制約，実数べき乗，双 1 次など）

LP	最適値	単体法	時間（秒）
77 × 105	3.951163342	39228	135.25

参考文献

[1] 山下浩，逸見信博，「計算グラフと分枝限定法を利用した大域的最適化」日本オペレーションズリサーチ学会 2003 年秋季研究発表会アブストラクト集，284-285.

[2] 逸見信博，山下浩，「分枝限定法による大域的最適化の実装」，「最適化：モデリングとアルゴリズム」シンポジウム発表，2004.

[3] Adjiman, Androulakis, and Floudas, "A Global Optimization Method, αBB, for General Twice-Differentiable Constrained NLPs - II. Implementation and Computational Results." Comp. Chem. Eng., 22, 1159, 1998.