新数理計画法システム LocalSolver による
大規模組合せ最適化問題への適用

01606110　MSI 株式会社　＊宮崎　知明　MIYAZAKI Tomoaki
MSI 株式会社　石村　猛　ISHIMURA　Takeshi

１．はじめに

大規模組合せ最適化問題は、既存の数理計画法システム（MIP）や制約論理システム（CP）では、解探索で組合せ爆発が起こり、実用時間内に解くことは出来なかったのが現状である。

新数理計画法システム LocalSolver は、解探索に組合せ爆発が起こらないよう、局所探索法をベースとしたシステムであり、１０００万以上の０－１整数変数を扱うことができる。

Localsolver は大規模組合せ最適化問題を汎用的かつコンパクトに定式化できるモデル記述言語（LSP 言語）を備えている。本稿では、LocalSolver のモデル記述言語である LSP 言語によるモデリングの考え方を紹介するとともに、大規模組合せ最適化問題への適用事例を示す。

２．LSP 言語による定式化

LocalSolverは意志決定変数(bool)として定義した0-1変数の組合せを超高速に試行することで、大規模組合せ最適化問題を実用的に解くことが基本の考え方である。bool変数の数がたとえ1000万変数を超えても実用的な意味で解を求めることができる。bool変数で定義した意志決定変数の組合せだけを変化させ解を探索していくため、LocalSolver用に定式化するためには、以下の点に注意する必要がある。

ナップサック問題のように条件に合う品物を選ぶのが目的であれば、品物を選ぶか選ばないかをbool変数として定義すれば良い。ある品物が選ばれた場合、bool変数を使って品物の重さ及び価値を直接計算できるため、bool変数の組合で制約条件、目的関数を評価することができる。

逆にLocalSolverでは、意志決定変数に制約または目的関数に関与する量（生産量等）を意味付けする必要がある。生産量の候補ごとにbool変数を定義する等、複数のbool変数の組合せで可変の生産量を関連づける必要がある。

2.1　LSP モデリングの考え方

以下の手順でモデリングを行う。

１）意志決定変数(bool 変数)を定義する。

２）選ばれた bool 変数の組み合わせで、制約、目的関数が必要とする数量（生産量等）が決まるよう、bool 変数を定義する。

３）bool 変数を使い、制約条件、目的関数を定義する。

この時、MIP のように、線形制約にこだわる必要はなく、制約条件、目的関数ともに、非線形表現が可能である。

2.2　LSP での bool 変数の定義

以下に典型的な bool 変数の定義イメージを示す。

-ナップサック問題：Xp　（ｐ：品物）
-ルート選択問題：　Xr　（ｒ：ルート）
-裁断計画問題：　Xp,q（p:裁断パターン、q:パターンの使用回数）
-人員配置問題：　Xp,t,j　（p：人員、ｔ：時間、ｊ：ジョブ）
-車両投入計画：　Xc,p　（ｃ：車両、ｐ：ポジション（順番））
-SCM：　Xt,i,j,k,p　（t:期、i:工場、j:ライン、k：倉庫、p：製品）
-スケジューリング：Xt,i,j,p　（t：時間等、i：工場、j：ライン、p：製品等）

2.3　LSP モデル

LSP モデルは、以下の要素から構成される。

- 意志決定変数：bool()
- 副生変数：任意の変数であり、プログラミングをわかりやすくすることができる。
  変数の定義には、<－　を使用する。
- 制約：constraint（予約語）で、制約条件を定義する。
  Constraint の条件が実行可能性の判定で使用される。
- 目的関数：minimize（予約語）または maximaize（予約語）で目的関数を定義する。目的関数は複数定義可能であるが、定義された順番に最適化を行う。目的計画法として利用可能である。

2.4　LSP 言語の特徴

LSP 言語は、モデリング及びモデルのチューニングを行うフェーズで試行錯誤を行うのに最適な環境を提供する。

LSP 言語は、最新の関数型プログラミング言語であり、型推論を備えている。Java や C 言語と異なり、コンパイラが自動的にデータの種類（型等）を推定するため、データの型等を指定する必要がない。その結果、プログラムの記述は Ruby など軽量言語のように簡潔である。

LSP言語の特徴は以下：

－迅速に開発できる（開発生産性が良い。従来に比べて、1/5から1/2の開発量）

－バグを抑えやすい（コンパイラが型の間違い等を自動的にチェックする）

－アプリケーションの性能を向上させやすい

－簡潔かつシンプルなモデリング言語（できるだけ省略できるよう設計）

※大規模問題でも制約条件及びデータがそろっていれば、１日でモデリングと実行が可能である。

－作成(修正)←→実行が同時にできる（一つはエディタ、もう一つは DOS コマンドプロンプトの二つのウインドウを操作しながら開発が可能である。

－目的計画法のように目的を段階的に設定することができるため、モデルの開発及び解の検証を段階的に行うことができる。

３．事例（裁断計画）

5000㎜の幅をもつフィルム等のロールから、色々な幅を持つ複数製品のロールを要求本数にできるだけ近くなるよう、裁断パターンとパターンの使用回数を求める。

目的関数（複合目的関数）は以下。

１．パターン数最小

２．要求数量とのギャップ最小

３．ロス部分(5000㎜からの差)。

有効幅		3600㎜ ～ 4600㎜		
製品	製品幅	要求本数	最小本数	最大本数
1	600	10	8	12
2	740	15	12	18
3	920	20	16	24
4	1060	5	4	6
5	1200	10	8	12

定式化として、裁断パターンと裁断パターンの使用回数を Bool 変数として定義する。定義した bool 変数で、目的関数、制約条件（製品数量）を定義する。

以下に MIP プログラムとの実行結果比較をしめす。

実行結果（１） 通常モデル

製品種類数：１０、総製品数量：１５３
裁断パターン候補数：１２８９８
（意志決定変数： ７５３０２）

LocalSolver 3.1： 11秒で 4パターン、要求差 = 3
市販汎用ソフト ：400秒すぎても 9パターン、要求差= 10

実行結果（２） 大規模モデル

製品種類数：１８、総製品数量：５０４
裁断パターン候補数：７０９２１
（意志決定変数： ３９９３７２）

LocalSolver 3.1：31秒で 8パターン、要求差=17
市販汎用ソフト ：4060秒で 10パターン、要求差＝22

４．おわりに

３０年前には殆ど実現出来なかった大規模組合せ最適化問題に対して、実践的なアプローチが実現できる時代になったと考える。「実学に役立つ OR」として、人間と機械の調和を実現して日本の産業界の再生の一助となれば幸いである。

参考文献

1) T. Benoist, B. Estellon, F. Gardi, R. Megel, K. Nouioua (2011).
・ 「LocalSolver 1.x: a black-box local-search solver for 0-1 programming」、*4OR, A Quarterly Journal of Operations Research* 9(3), pp. 299-316. Springer.

2) MSI 株式会社
「http://msi-jp.com/localsolver/」ホームページ

参考文献

3) T. Benoist, B. Estellon, F. Gardi, R. Megel, K. Nouioua (2011).
・ 「LocalSolver 1.x: a black-box local-search solver for 0-1 programming」、*4OR, A Quarterly Journal of Operations Research* 9(3), pp. 299-316. Springer.

4) MSI 株式会社
「http://msi-jp.com/localsolver/」ホームページ