2000年度日本オペレーションズ・リサーチ学会
秋季研究発表会

多資源一般化割当問題に対する大規模近傍探索法の適用について

京都大学　*岩崎 慎司　IWASAKI Shinji
01704164　京都大学　柳浦 睦憲　YAGIURA Mutsunori
01001374　京都大学　茨木 俊秀　IBARAKI Toshihide

1　まえがき

本研究では, 代表的なNP困難問題の1つである多資源一般化割当問題 (multi-resource generalized assignment problem) に対し, 連鎖シフト近傍を用いた大規模近傍探索法 (very large-scale neighbourhood search) を提案する. 大規模な近傍を用いれば解の質は向上するが, 単純に近傍を広げただけでは近傍内の探索に時間がかかってしまう. そのため, 近傍内の改善解を発見する過程を改善解探索グラフと呼ばれるグラフで表現し, そのグラフの探索によって, 効率的に近傍を探索する工夫を加えている. 計算実験は, 1資源の代表的なベンチマーク問題と, これらに基づいてランダムに生成した多資源での問題例に対して行い, 連鎖シフト近傍を用いた場合, 単純な近傍に基づく多スタート局所探索法やタブー探索法より良い解が得られる傾向が確認できた.

2　問題の定式化

多資源一般化割当問題は, エージェント集合 $I=\{1,...,m\}$ と, それらに割当てる仕事集合 $J=\{1,...,n\}$ が与えられたとき, 資源集合 $K=\{1,...,s\}$ に関する制約条件の下で, 各仕事をちょうど一つのエージェントに割当てたときに, 割当コストの総和を最小にする問題である. この問題では, 各 $i\in I$, $j\in J$, および $k\in K$ に対し,

c_{ij}: エージェント i が仕事 j を処理するのにかかるコスト
a_{ijk}: エージェント i が仕事 j を処理するのに使用する資源 k の量 ($a_{ijk}\geq 0$ と仮定)
b_{ik}: エージェント i における資源 k の使用可能量 ($b_{ik}>0$ と仮定)

が与えられる. また, 割当を写像 $\sigma: J\to I$ ($\sigma(j)=i$ は, 仕事 j をエージェント i に割当てることを意味する) で表す. また, 解 σ において, エージェント i に割当てられる仕事の集合を

$$J_i(\sigma)=\{j\in J\,|\,\sigma(j)=i\},\ i\in I$$

と記す. これらを用いて問題は以下のように定式化される:

$$\text{minimize}\quad cost(\sigma)=\sum_{j\in J}c_{\sigma(j),j}$$

$$\text{subject to}\quad \sum_{j\in J_i(\sigma)}a_{ijk}\leq b_{ik},\ \forall i\in I,\ \forall k\in K.$$

3　局所探索法

局所探索法は, 適当な初期解から始め, 現在の解 σ の近傍 $N(\sigma)$ 内に σ より良い解があればそれに置き換えるという操作を, 近傍内に改善解がなくなるまで反復する方法である. 本研究では, 探索する解空間を任意の割当 $\sigma: J\to I$ とし, 探索解の評価の際に, 実行可能解からの逸脱の度合いをペナルティ関数として表現して目的関数に組み込む, ペナルティ関数法を用いる. エージェント i に仕事集合 S ($\subseteq J$) を割当てたときの資源 k に関するペナルティを

$$p_{ik}(S)=\max\left\{0,\left(\sum_{j\in S}a_{ijk}\right)-b_{ik}\right\},\ i\in I,\ k\in K$$

とする. このとき, 解の評価関数は

$$pcost(\sigma)=cost(\sigma)+\sum_{i\in I}\sum_{k\in K}\alpha_{ik}p_{ik}(J_i(\sigma))$$

と定義される. なお, α_{ik} は, 正の値をとるパラメータで, 探索の状況に応じて適応的に制御する.

近傍 $N(\sigma)$ には, シフト近傍, 交換近傍, および連鎖シフト近傍の3つを用いる. このうち連鎖シフト近傍は, 定義がやや複雑なので, 3.1節で説明する. シフト近傍 N_{shift} と交換近傍 N_{swap} は, それぞれ

$$N_{shift}(\sigma)=\{\sigma'\,|\,\sigma'\text{は}\sigma\text{の一つの仕事の割当先を変更することで得られる}\}$$

$$N_{swap}(\sigma)=\{\sigma'\,|\,\sigma'\text{は}\sigma\text{の二つの仕事の割当先を互いに交換することで得られる}\}$$

と定義される.

3.1　連鎖シフト近傍

連鎖シフト近傍 $N_{chain}(\sigma)$ は, 任意の l ($l=2,3,\ldots,n$) に対し, l 個の仕事 $j_1,j_2,\ldots,j_l$ を

$$\sigma'(j_{r+1}) := \sigma(j_r),\quad \forall r=1,2,\ldots,l-1$$
$$\sigma'(j_1) := \sigma(j_l)$$

とシフトすることにより得られる解 σ' の集合である. すなわち, ある一つの仕事 j_r を現在属しているエージェント $\sigma(j_r)$ から別のエージェントにシフトすることにより, $\sigma(j_r)$ の利用可能な資源の量が増えるので, $\sigma(j_r)$ へ次の仕事 j_{r+1} を $\sigma'(j_{r+1}):=\sigma(j_r)$ とシフトするという操作を, 連鎖的に繰り返すのである. 複数の仕事を同時に交換するので, 連鎖シフト近傍のサイズはきわめて大き

く, 近傍のサイズ $|N_{chain}(\sigma)|$ はエージェント数に関して指数的に増加する. そのため, 単純に全ての可能性を列挙して調べるのではなく, 近傍の大きさを制限するなどの工夫を加える必要がある. 本研究では, 次に述べる改善解探索グラフを利用することにより, 近傍探索の効率化を実現している.

3.2 改善解探索グラフ

改善解探索グラフ $G(\sigma)$ は, 解 σ に応じて定義され, その節点集合は仕事集合 J である. また, 枝集合は $\{(j,j') \in J \times J \mid \sigma(j) \neq \sigma(j')\}$ である. すなわち異なるエージェントに属する仕事の対に対応する2節点間に有向枝が存在する. 各枝 (j,j') には重み

$$w_{jj'} = pcost_i(J_i(\sigma) \cup \{j'\} \backslash \{j\}) - pcost_i(J_i(\sigma))$$

(ただし $i = \sigma(j)$) が付される. ただし

$$pcost_i(S) = \sum_{j \in S} c_{ij} + \sum_{k \in K} \alpha_{ik} p_{ik}(S), \quad i \in I, S \subseteq J$$

である。これは, 仕事 j が現在の解 σ において属しているエージェント i より, j を取り除き, 別の仕事 j' を割当てたときの, i における評価関数の増加量である.

ここで仕事 $j_1, j_2, \ldots, j_l$ に対する連鎖シフトと, グラフ $G(\sigma)$ のサイクル $j_1 \to j_2 \to \cdots \to j_l \to j_1$ を考える. すべての $r \neq r' \in \{1, \ldots, l\}$ に対して $\sigma(j_r) \neq \sigma(j_{r'})$ が成り立つならば, 連鎖シフトによる評価関数 $pcost$ の変化量は

$$w_{j_l j_1} + \sum_{r=1}^{l-1} w_{j_r j_{r+1}}$$

と計算できる. よって, 改善解探索グラフにおいて, エージェントの重複がないようなサイクルの中で重みの合計が負となるものを見つけ出すことができれば, 連鎖シフト近傍内の改善解を発見できる. この様なサイクルを探索する問題は, 厳密には NP 困難であるが, エージェントの重複に関する条件を取り去った問題は, 動的計画法を用いて効果的に解くことができる. そこで, 本研究では, 動的計画法に基づいてグラフ上の重み負のサイクルを探索し, 重み負のサイクルが発見されるたびに, 対応する連鎖シフトによる $pcost$ の正確な変化量を再計算するという方法を用いる. サイクル上にエージェントの重複がある場合でも, 枝の重みの合計は $pcost$ の変化量のよい近似になっている場合が多いと考えられ, 効果的な探索が期待できる.

4 動的計画法に基づく連鎖シフト近傍の探索

$f^*(j_1, j, l)\ (j_1, j \in J, l = 1, 2, \ldots, m-1)$ を, j_1 を始点とし, j を終点とする長さ l のパスの最小重みとする. なお, パスの長さは枝数, パスの重みはパス上の枝の重みの合計である.

各 $j_1 \in J$ に対し, $f^*(j_1, j, l)$ は漸化式

$$f^*(j_1, j, 1) = w_{j_1 j}, \qquad j \in J$$

$$f^*(j_1, j, l) = \min_{j' \in J}\{f^*(j_1, j', l-1) + w_{j'j}\},$$

$$j \in J,\ l = 2, 3, \ldots, m-1$$

によって計算できる. $f^*(j_1, j, l) + w_{jj_1}$ は, j_1 を始点, j を終点とする長さ l のパスの中で重みが最小のものに, 枝 (j, j_1) を加えたサイクルの重みを表すので, 改善解探索グラフ上の重み負のサイクルを見つけるには, この値が負になる (j_1, j, l) の組を探索すれば良い. なお, この探索は一回の探索時間にかかる時間が大きいため (1) 動的計画法の漸化式の計算の高速化, (2) 改善解探索グラフの枝を強制的に減らす, などの工夫を行っている.

5 アルゴリズムの概要

提案手法では, シフト近傍, 交換近傍, 連鎖シフト近傍の3つの近傍による局所探索を順次行い, いずれの近傍内にも改善解が見つからなくなると, 評価関数の増加を許して強制的にシフトを行う. その際, 探索の重複を避けるためにタブー探索法を利用している. また, ペナルティー重み α_{ik} の適応的制御も行っている. これらの工夫は, (1 資源) 一般化割当問題において非常に有効であることが分かっており [2], 多資源の場合にも同様の効果が期待できる.

6 計算結果および結論

実験は, ワークステーション Sun Ultra 2 Model 2300 (300 MHz, 1 GB memory) 上で C 言語を用いて行った. 問題例は, 一般化割当問題の代表的なベンチマーク問題に基づき多資源の場合の問題例の生成方法 [1] を参考にランダムに作成した (http://www-or.amp.i.kyoto-u.ac.jp/~yagiura/mrgap/ より入手可能) .

表1に, 提案手法 (TS-CS), TS-CS においてい連鎖シフト近傍を用いないタブー探索法 (TS-noCS), およびシフト近傍と交換近傍を用いたランダム多スタート局所探索法 (MLS) による最良解を示す。全てのアルゴリズムに対し, 制限時間を 600 秒とした. 表より, 連鎖シフト近傍を用いた手法は, MLS や TS-noCS のように単純な近傍に基づく手法と比べて, より良い解を得ることが観測された.

表1. 3つのアルゴリズムによる最良解の比較
(*印は3つの内の最良であることを示す)

m	n	s	TS-CS	TS-noCS	MLS
5	200	1	*12751	12756	12895
		2	*12766	12772	12923
		4	*12775	12778	12896
		8	*12805	12809	12994
10	200	1	*12471	12490	12737
		2	*12477	12518	12741
		4	*12496	12552	12810
		8	*12571	12592	12822
20	200	1	*12312	12391	12977
		2	*12332	12384	12883
		4	*12396	12488	12969
		8	*12485	12650	12974

参考文献

[1] B. Gavish and H. Pirkul, "Algorithms for the Multi-Resource Generalized Assignment Problem," *Management Science*, 37 (1991) 695-713.

[2] M. Yagiura, T. Ibaraki and F. Glover, "An Ejection Chain Approach for the Generalized Assignment Problem," Technical Report #99013, Department of Applied Mathematics and Physics, Graduate School of Informatics, Kyoto University, (1999) (available at http://www-or.amp.i.kyoto-u.ac.jp/~yagiura/).