一般化上界制約付き集合多重被覆問題に対する発見的解法

A heuristic algorithm for the set multicover problem with generalized upper bound constraints

with **荒川正尚 (富士通)，柳浦睦憲 (名古屋大学)**

梅谷 俊治

大阪大学 大学院情報科学研究科 情報数理学専攻
科学技術振興機構 さきがけ PRESTO

2012 **年** 11 **月** 30 **日**
離散構造と最適化：展開と連携
(九州大学マス・フォア・インダストリ研究所)

本講演の概要

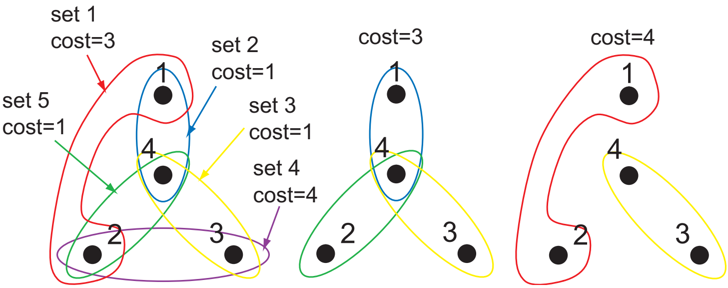
- 集合被覆問題
  - 定式化と応用例
  - 緩和問題とその性質
  - 価格法を用いた問題の縮小
- 一般化上界制約付き集合多重被覆問題
  - 定式化と緩和問題
  - 提案アルゴリズム
  - 数値実験

集合被覆問題

入力: m 個の要素 $i \in M = \{1, \ldots, m\}$．$n$ 個の集合 $S_j \subseteq M$ とそのコスト $c_j > 0$ $(j \in N = \{1, \ldots, n\})$．

制約: 全要素 $i \in M$ を含む集合 $S_j (j \in N)$ の組合せ $X \subseteq N$ を選ぶ．

目的: 選んだ集合 $S_j (j \in X)$ のコスト c_j の総和を最小化する．



0-1 整数計画問題による定式化

- 行 i を要素 $i \in M$ に，列 j を集合 $S_j (j \in N)$ に対応させることで，0-1 整数計画問題に定式化できる．
- $a_{ij} = 1$ ならば『列 j は行 i を被覆 (カバー)』すると呼ぶ．

$$
\begin{array}{lll}
\min & z(x) = \displaystyle\sum_{j \in N} c_j x_j & \\
\text{s.t.} & \displaystyle\sum_{j \in N} a_{ij} x_j \geq 1 & (i \in M) \\
& x_j \in \{0, 1\} & (j \in N)
\end{array}
$$

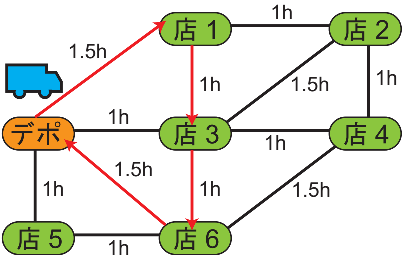
$$
a_{ij} = \begin{cases} 1 & i \in S_j \\ 0 & i \notin S_j \end{cases}
$$

$$
x_j = \begin{cases} 1 & j \in X \\ 0 & j \notin X \end{cases}
$$

乗務員スケジューリング，配送計画，施設配置，
データの論理的解析など，多くの現実的な問題を定式化できる

応用例：配送計画

- m カ所の店舗に商品を配送するのに必要なトラックの台数を最小にするような配送ルートは？
- 1 台のトラックで時間内に配送できるルートを全て列挙すると集合被覆問題として定式化できる．



6 時間以内で回れるルートを全て列挙

$$\boldsymbol{c} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

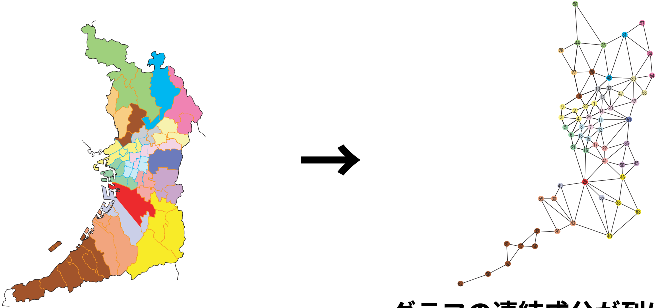
$$A = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

各ルートが列に対応

勤務スケジュール作成問題や時間割問題も定式化できる

応用例：選挙区割り

- m 個の市区郡を一票の格差が最小になるように n 個の選挙区に分割するには？
- 市区郡の連結な部分集合を全て列挙すると集合分割問題として定式化できる．



グラフの連結成分が列に対応

施設配置問題，ネットワーク設計問題なども定式化できる

応用例：判別問題における特徴選択

- 正例と負例のデータ集合 $T, F \subseteq \{0,1\}^n$ が与えられたとき，$a \in T$ に 1 を $b \in F$ に 0 を返す関数 $f : \{0,1\}^n \to \{0,1\}$ を求める.
- 関数 f を求めるのに必要な変数集合 (特徴集合) で，位数が最小のものを求める問題は集合被覆問題に定式化できる．
  ($a \in T$ と $b \in F$ を区別するには $a_j \neq b_j$ となる変数 x_j が少なくとも 1 つ必要)

		x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
	$a_1 =$	0	1	0	1	0	1	1	0
T	$a_2 =$	1	1	0	1	1	0	0	1
	$a_3 =$	0	1	1	0	1	0	0	1
	$b_1 =$	1	0	1	0	1	0	1	0
F	$b_2 =$	0	0	0	1	1	1	0	0
	$b_3 =$	1	1	0	1	0	1	0	1
	$b_4 =$	0	0	1	0	1	0	1	0

LP 緩和問題

- **各変数** x_j **の整数条件を** $0 \le x_j \le 1$ **に緩和する．**
- $\tilde{c}_j(u) = c_j - \sum_{i \in M} a_{ij} u_i$ **は被約費用と呼ばれる．**
- **主** LP **緩和問題の最適値** $z_{\mathrm{LP}}(\bar{x})$ = **双対** LP **緩和問題の最適値** $z_{\mathrm{DLP}}(\bar{u})$.

主 LP **緩和問題**:

$$\begin{array}{lll} \min & z_{\mathrm{LP}}(x) = \sum_{j \in N} c_j x_j & \\ \text{s.t.} & \sum_{j \in N} a_{ij} x_j \ge 1 & (i \in M) \\ & x_j \ge 0 & (j \in N) \end{array}$$

双対 LP **緩和問題**:

$$\begin{array}{lll} \max & z_{\mathrm{DLP}}(u) = \sum_{i \in M} u_i & \\ \text{s.t.} & \sum_{i \in M} a_{ij} u_i \le c_j & (j \in N) \\ & u_i \ge 0 & (i \in M) \end{array}$$

ラグランジュ緩和問題

- いくつかの制約式を取り除く代わりに取り除いた制約式の違反度に**ラグランジュ乗数** u をかけて目的関数に繰り込む方法．
- $\tilde{c}_j(u) = c_j - \sum_{i \in M} a_{ij} u_i$ は**被約費用**と呼ばれる．

$$
\begin{array}{lll}
\min & z_{\mathrm{LR}}(u) & = \displaystyle\sum_{j \in N} c_j x_j + \sum_{i \in M} u_i \left(1 - \sum_{j \in N} a_{ij} x_j \right) \\
 & & = \displaystyle\sum_{j \in N} \tilde{c}_j(u) x_j + \sum_{i \in M} u_i \\
\text{s.t.} & x_j \in \{0, 1\} & \quad (j \in N)
\end{array}
$$

ラグランジュ緩和問題 (続き)

- **被約費用 $\tilde{c}_j(u)$ の正負によって，ラグランジュ緩和問題の最適解 $\tilde{x}(u)$ を決定できる．**

$$\tilde{x}_j(u) = \begin{cases} 1 & \tilde{c}_j(u) < 0 \\ \{0,1\} & \tilde{c}_j(u) = 0 \\ 0 & \tilde{c}_j(u) > 0 \end{cases}$$

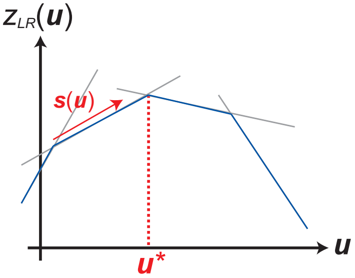
- **最良の下界値を与えるラグランジュ乗数 u を求める問題はラグランジュ双対問題と呼ばれる．**

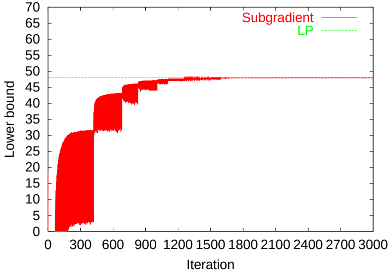
$$\begin{array}{lll} \max & z_{\mathrm{LR}}(u) = \displaystyle\min_{x \in \{0,1\}^n} \sum_{j \in N} c_j x_j + \sum_{i \in M} u_i \left(1 - \sum_{j \in N} a_{ij} x_j \right) & \\ \text{s.t.} & u_i \geq 0 \qquad (i \in M) & \end{array}$$

ラグランジュ双対問題に対する劣勾配法

- 区分線形な凸関数なので**劣勾配法**を適用する．
- 劣勾配 $s(u) = (s_1(u), s_2(u), \ldots, s_m(u))$ の方向に沿ってラグランジュ乗数 u を更新する．

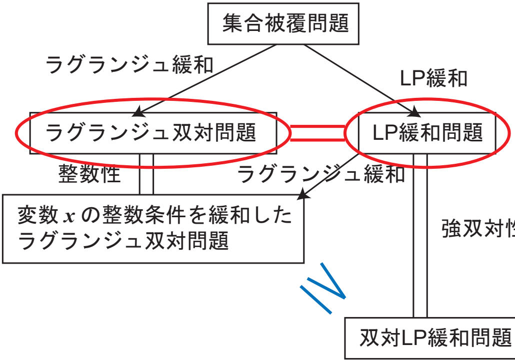
$$s_i(u) = 1 - \sum_{j \in N} a_{ij} \tilde{x}_j(u) \quad (i \in M)$$





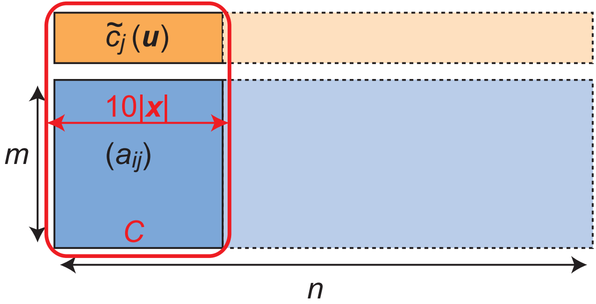
LP 緩和とラグランジュ緩和の関係

- LP 緩和問題の最適値 ＝ ラグランジュ双対問題の最適値.
- 双対 LP 緩和問題の最適解 ⇒ ラグランジュ双対問題の最適解.
- 単体法：LP 緩和問題の最適解 (実数解).
- 劣勾配法：ラグランジュ双対問題の近似解 (整数解).



問題の縮小

- 変数の数 n が非常に多い事例が頻繁に現れる．
  例：乗務員スケジューリングの事例では制約式 4284 本に対して変数 1,092,610 個．
- $x_j = 1$ となる変数は高々制約式の本数となる．
- 有望な変数のみを用いた制限された集合被覆問題を解く．
- コスト係数 c_j だけでは有望な変数が特定できない．



価格法

- LP 緩和問題ではその最適値と双対問題の最適値が一致する．
- **主問題に変数の追加 ＝ 双対問題に制約式を追加．**
- 部分問題 $C \subset N$ の LP 緩和問題を解いた後に，残りの変数の**被約費用** $\tilde{c}_j(\bar{u})(j \in N \setminus C)$ を計算して，$\tilde{c}_j(\bar{u}) < 0$ となる変数を部分問題 C に追加する．

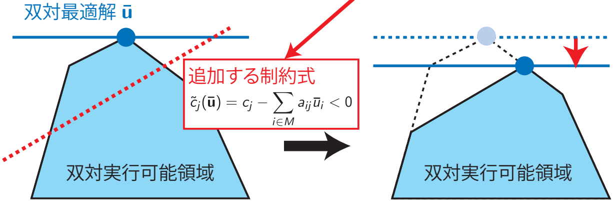
主LP緩和問題：

$$
\begin{array}{lll}
\min & \sum_{j \in C} c_j x_j & \\
\text{s.t.} & \sum_{j \in C} a_{ij} x_j \geq 1 & (i \in M) \\
& x_j \geq 0 & (j \in C)
\end{array}
$$

$\longleftrightarrow$

双対LP緩和問題：

$$
\begin{array}{lll}
\max & \sum_{i \in M} u_i & \\
\text{s.t.} & \boxed{\sum_{i \in M} a_{ij} u_i \leq c_j} & (j \in C) \\
& u_i \geq 0 & (i \in M)
\end{array}
$$



下界値の計算と変数固定

下界値の計算

- 原問題の下界値 $z_{\mathrm{DLP}}(u)$ は以下の式で計算できる．

$$z_{\mathrm{DLP}}(u) = z^{c}_{\mathrm{DLP}}(u) + \sum_{j \in N \setminus C} \min\{\tilde{c}_j(u), 0\}$$

変数固定

- LP 緩和問題の解が $\bar{x}_j = 0$ の場合，仮に変数の値を $x_j = 1$ に固定すると，下界値は $\tilde{c}_j(\bar{u})$ だけ増加する．
- $\bar{x}_j = 0$ かつ $z_{\mathrm{DLP}}(\bar{u}) + \tilde{c}_j(\bar{u}) > z_{\mathrm{UB}}$ ならば，$x_j \leftarrow 0$ として列 j を削除できる．

上界と下界のギャップが比較的小さければ，
価格法で変数を絞り込むことで探索が効率化できる

一般化上界制約付き集合多重被覆問題

- **一般化上界制約：変数の添字集合 N のある分割 $G_1, \ldots, G_k$ について，各分割 $G_h (h \in K = \{1, \ldots, k\})$ の中から値を 1 にできる変数 $x_j (j \in G_h)$ の数を d_h 個以下に制限する．**
- **多重被覆制約：各要素 $i \in M$ を b_i 回以上被覆する．**

$$
\begin{array}{lll}
\min & \displaystyle\sum_{j \in N} c_j x_j & \\
\text{s.t.} & \displaystyle\sum_{j \in N} a_{ij} x_j \geq b_i & (i \in M) \\
& \displaystyle\sum_{j \in G_h} x_j \leq d_h & (h \in K) \\
& x_j \in \{0, 1\} & (j \in N)
\end{array}
$$

一般化上界制約，多重被覆制約によって応用範囲が大幅に拡大

ペナルティ関数の導入

- **実行可能解がない問題例があり，かつ実行可能解の存在判定問題は** NP **完全である．**
- **各要素** $i \in M$ **被覆回数** b_i **の不足分** y_i **に重み** $w_i (> 0)$ **を掛けた値をペナルティとして目的関数に加える．**
- x **が決まると最適な** y **は** $y_i = \max\{b_i - \sum_{j \in N} a_{ij} x_j, 0\}$ **となる．**

$$
\begin{array}{lll}
\min & z(x) = \displaystyle\sum_{j \in N} c_j x_j + \sum_{i \in M} w_i y_i & \\
\text{s.t.} & \displaystyle\sum_{j \in N} a_{ij} x_j + y_i \geq b_i & (i \in M) \\
& \displaystyle\sum_{j \in G_h} x_j \leq d_h & (h \in K) \\
& x_j \in \{0, 1\} & (j \in N) \\
& y_i \in \{0, 1, \ldots, b_i\} & (i \in M)
\end{array}
$$

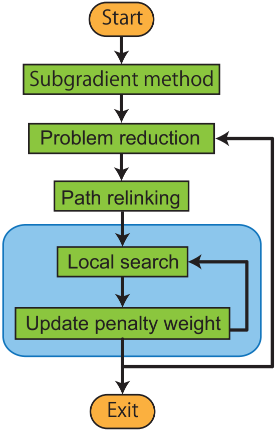
ラグランジュ緩和問題

- **緩和した制約式の違反度にラグランジュ乗数** $u \in \mathbb{R}^m_+$ **を掛けたものを目的関数に加える．** $\tilde{c}_j(u) = c_j - \sum_{i \in M} a_{ij} u_i$ **は被約費用と呼ばれる．**
- **各分割** $G_h (h \in K)$ **にて** $\tilde{c}_j(u) < 0$ **となる変数が** d_h **個以下なら全て** $\tilde{x}_j(u) = 1$ **とする．** d_h **個より多いなら** $\tilde{c}_j(u)$ **の小さい順に** d_h **個を選んで** $\tilde{x}_j(u) = 1$ **とする．**

$$
\begin{aligned}
\min \quad z_{\mathrm{LR}}(u) &= \sum_{j \in N} c_j x_j + \sum_{i \in M} w_i y_i + \sum_{i \in M} u_i \left(b_i - \sum_{j \in N} a_{ij} x_j - y_i \right) \\
&= \sum_{j \in N} \tilde{c}_j(u) x_j + \sum_{i \in M} y_i \left(w_i - u_i \right) + \sum_{i \in M} b_i u_i \\
\text{s.t.} \quad & \sum_{j \in G_h} x_j \le d_h \qquad (h \in K) \\
& x_j \in \{0, 1\} \qquad (j \in N)
\end{aligned}
$$

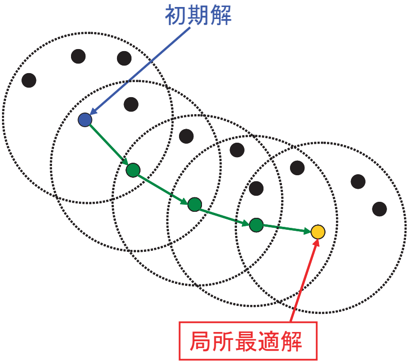
提案アルゴリズムの概要

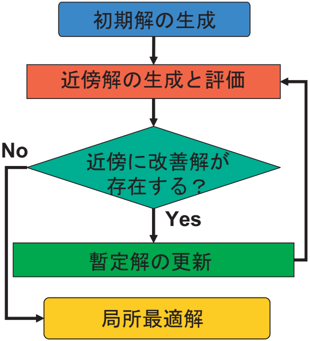
問題の縮小: (修正版) 価格法.
初期解の生成: パス再結合法.
解の探索: 局所探索法とペナルティ重み $\hat{w}_i (\leq w_i)$ の適応的な調整.



局所探索法

- 適当な初期解 x から始めて，現在の解 x の近傍 $N(x)$ 内に改善解 x' があれば移動する (近傍探索).
- 近傍内に改善解が存在しなくなるまで近傍探索を繰返す.





近傍と評価関数

近傍

追加近傍: 集合 S_j を追加 $(x_j = 0 \to 1)$.
削除近傍: 集合 S_j を削除 $(x_j = 1 \to 0)$.
交換近傍: 集合 S_{j_1} と S_{j_2} を交換 $(x_{j_1} = 1 \to 0, x_{j_2} = 0 \to 1)$.

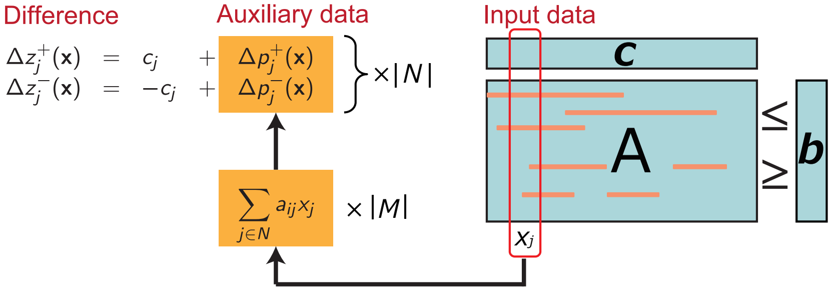
評価関数

- 実行不可能領域も探索できるように，適応的に調整を加えたペナルティ重み $\hat{w}_i (\leq w_i)$ で置き換えた評価関数 $\hat{z}(x)$ を用いる．
- 同時に元の評価関数 $z(x)$ でも評価して最良の実行可能解 x^* は逃さずに記憶する．

$$\hat{z}(x) = \sum_{j \in N} c_j x_j + \sum_{i \in M} \hat{w}_i \cdot \max\{b_i - \sum_{j \in N} a_{ij} x_j, 0\}$$

局所探索法の効率化

- 解の評価回数 $>>$ 解の更新回数なので，解 1 個当たりの評価時間を短縮する．
- 補助記憶 $\Delta p_j^+(x), \Delta p_j^-(x), \sum_{j \in N} a_{ij} x_j$ を用いて，評価関数の変化量 $\Delta \hat{z}(x)$ を $O(1)$ 時間で評価する．
- 疎な係数行列であれば補助記憶も効率良く更新できる．



交換近傍操作の効率化

- 交換近傍 $(x_{j_1} = 1 \to 0, x_{j_2} = 0 \to 1)$ による評価関数の変化量．

$$\Delta \hat{z}_{j_1,j_2}(x) = \Delta \hat{z}^-_{j_1}(x) + \Delta \hat{z}^+_{j_2}(x) - \sum_{i \in M_{\mathrm{E}}(x) \cap S_{j_1} \cap S_{j_2}} \hat{w}_i \text{※}$$

- 追加近傍，削除近傍に関して局所最適ならば，交換近傍で $\Delta \hat{z}_{j_1,j_2}(x)$ となるのは以下の場合のみ．
  (i) j_1, j_2 が共に $\sum_{j \in G_h} x_j = d_h$ を満たす分割 G_h に属する．
  (ii) $M_{\mathrm{E}}(x) \cap S_{j_1} \cap S_{j_2} \neq \emptyset$

- 特に前者の場合は各分割 G_h で

$$\min_{j \in G_h} \Delta \hat{z}^-_j(x) + \min_{j \in G_h} \Delta \hat{z}^+_j(x) < 0$$

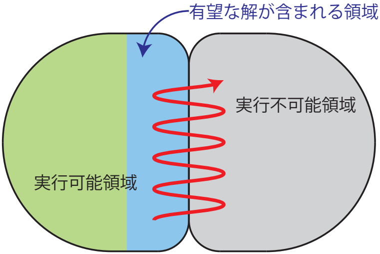
をチェックするだけで改善解が高速に見つけられる．

※ $M_{\mathrm{E}}(x) = \{i \in M \mid \sum_{j \in N} a_{ij} x_j = b_i\}$

ペナルティ重みの適応的な調整

- **実行可能領域と実行不可能領域の境界周辺を集中的に探索するために，ペナルティ重み $\hat{w}_i (\leq w_i)$ を適応的に更新する．**
- **$\hat{z}(x) \geq z(x^*)$ ならば $\hat{w}_i (i \in M)$ を一様に減少する．そうでなければ，以下の式に従って $\hat{w}_i$ を増加する．**

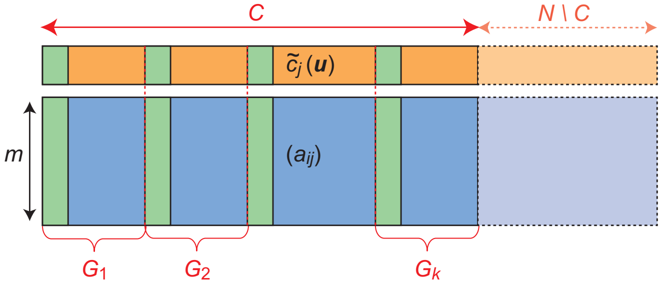
$$\hat{w}_i \leftarrow \min\left\{ \hat{w}_i \left(1 + \delta \frac{\rho_i(x)^{※}}{\max_{i \in M} \rho_i(x)} \right), w_i \right\} \quad (i \in M)$$



※ $\rho_i(x) = \max\{b_i - \sum_{j \in N} a_{ij} x_j, 0\}$

価格法の問題点

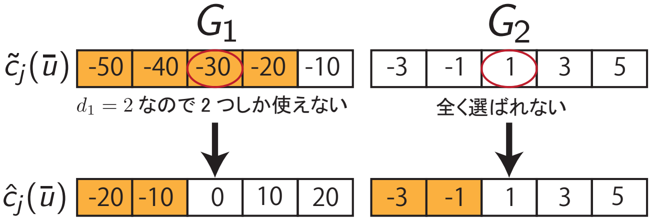
- 原問題と緩和問題の最適値のギャップが大きく緩和問題の情報のみでは良い部分問題 $C(\subset N)$ が構成できない．
- 被約費用 $\tilde{c}_j(\bar{u})$ の小さい変数で部分問題 C を構成しても，一般化上界制約のため同時に $x_j = 1$ とできない．



部分問題の大半の変数が使えないため解精度が著しく悪化する

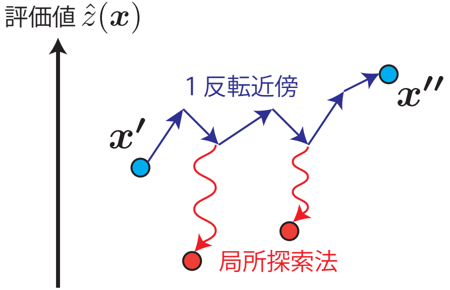
修正被約費用を用いた価格法

- 分割 G_h によっては $\tilde{c}_j(\bar{u})$ が非常に小さくても $x_j = 1$ とならない変数が多数生じる．
- 各分割 G_h で，$d_h + 1$ 番目に小さい被約費用 $\tilde{c}_j(\bar{u})$ の値が 0 となるように正規化した修正被約費用を用いて部分問題 C を構成する．



パス再結合法

- 複数個の解の結合により新たな解を生成することで探索の多様化を実現する散布探索法の 1 つのバリエーション．
- x' から x'' へ 1 反転近傍で移動する経路の途中の解を初期解として局所探索法を適用する．



遺伝アルゴリズムと似ているが，良い解の構造を壊すことなく探索の多様化を実現できる

ベンチマーク問題例

- OR-Library の集合被覆問題のベンチマーク問題例※(ランダム生成) を利用した．
- CPLEX12.3 と提案アルゴリズムの (i) 通常の被約費用，(ii) 修正被約費用を用いたバリエーションを比較した．
- Intel Xeon E5420(4 core) 2.5GHz, 4GB memory を搭載した PC で 1 スレッド実行した．

Instance	Rows	Columns	Density	Cost	CPLEX12.3	proposed
G.1–G.5	1000	10,000	2.0%	[1,100]	3600s	600s
H.1–H.5	1000	10,000	5.0%	[1,100]	3600s	600s
I.1–I.5	1000	50,000	1.0%	[1,100]	3600s	600s
J.1–J.5	1000	100,000	1.0%	[1,100]	3600s	600s
K.1–K.5	2000	100,000	0.5%	[1,100]	7200s	1200s
L.1–L.5	2000	200,000	0.5%	[1,100]	7200s	1200s
M.1–M.5	5000	500,000	0.25%	[1,100]	18,000s	3000s
N.1–N.5	5000	1,000,000	0.25%	[1,100]	18,000s	3000s

※クラス I–N は新しく生成した．

ベンチマーク問題例 (続き)

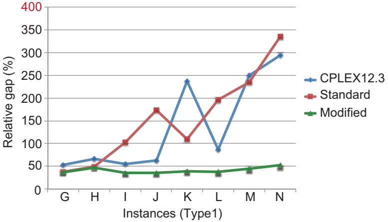
- クラス G–N に対して $d_h/|G_h|$ の異なる問題例を作成した．b_i は $[1, 5]$ のランダムな整数値に設定した．
- 解精度は以下の式に基づく相対誤差 (%) で評価した．

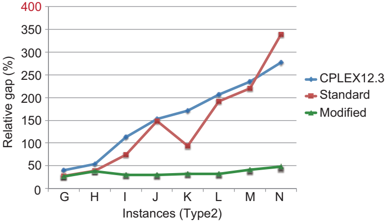
$$\mathrm{gap}(x) = \frac{z(x) - z_{\mathrm{LP}}(\bar{x})}{z_{\mathrm{LP}}(\bar{x})} \times 100.0$$

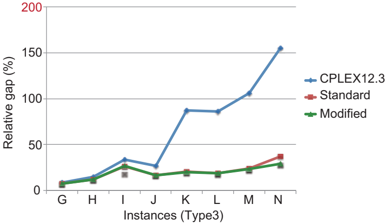
Instance	Rows	Columns	Type1	Type2	Type3	Type4
G.1–G.5	1000	10,000	1/10	10/100	5/10	50/100
H.1–H.5	1000	10,000	1/10	10/100	5/10	50/100
I.1–I.5	1000	50,000	1/50	10/500	5/50	50/500
J.1–J.5	1000	100,000	1/50	10/500	5/50	50/500
K.1–K.5	2000	100,000	1/50	10/500	5/50	50/500
L.1–L.5	2000	200,000	1/50	10/500	5/50	50/500
M.1–M.5	5000	500,000	1/50	10/500	5/50	50/500
N.1–N.5	5000	1,000,000	1/100	10/1000	5/100	50/1000

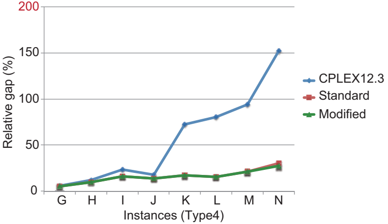
数値実験の結果

- 一般化上界制約の厳しい問題例では (ii) 修正被約費用を用いた提案アルゴリズムの性能が良い．









まとめと今後の課題

- 集合被覆問題とその応用
  - 定式化と応用例
  - 緩和問題とその性質
  - 価格法を用いた問題の縮小
- 一般化上界制約と多重制約による集合被覆問題の拡張
  - 補助記憶を用いた効率的な局所探索法
  - ペナルティ重みの適応的な調整
  - 修正被約費用を用いた価格法
  - パス再結合法
  - 数値実験

参考文献

- S. Umetani, M. Arakawa and M. Yagiura: A heuristic algorithm for the set multicover problem with generalized upper bound constraints, submitted.
- S. Umetani and M. Yagiura: Relaxation heuristics for the set covering problem, *Journal of Operations Research Society of Japan*, **50**(2007), 350-375. `http://www.orsj.or.jp/~archive/pdf/e_mag/50-4-350-375.pdf`
- Y. Yagiura, M. Kishida and T. Ibaraki: A 3-flip neighborhood local search for the set covering problem, *European Journal of Operational Research*, **172**(2006), 472-499.