組合せ最適化エンジンWCSPの仕組みと実装

野々部宏司（法政大学）

数理システムユーザーコンファレンス2015（2015年11月20日）

最新の数理計画法パッケージ
ニューメリカル オプティマイザー

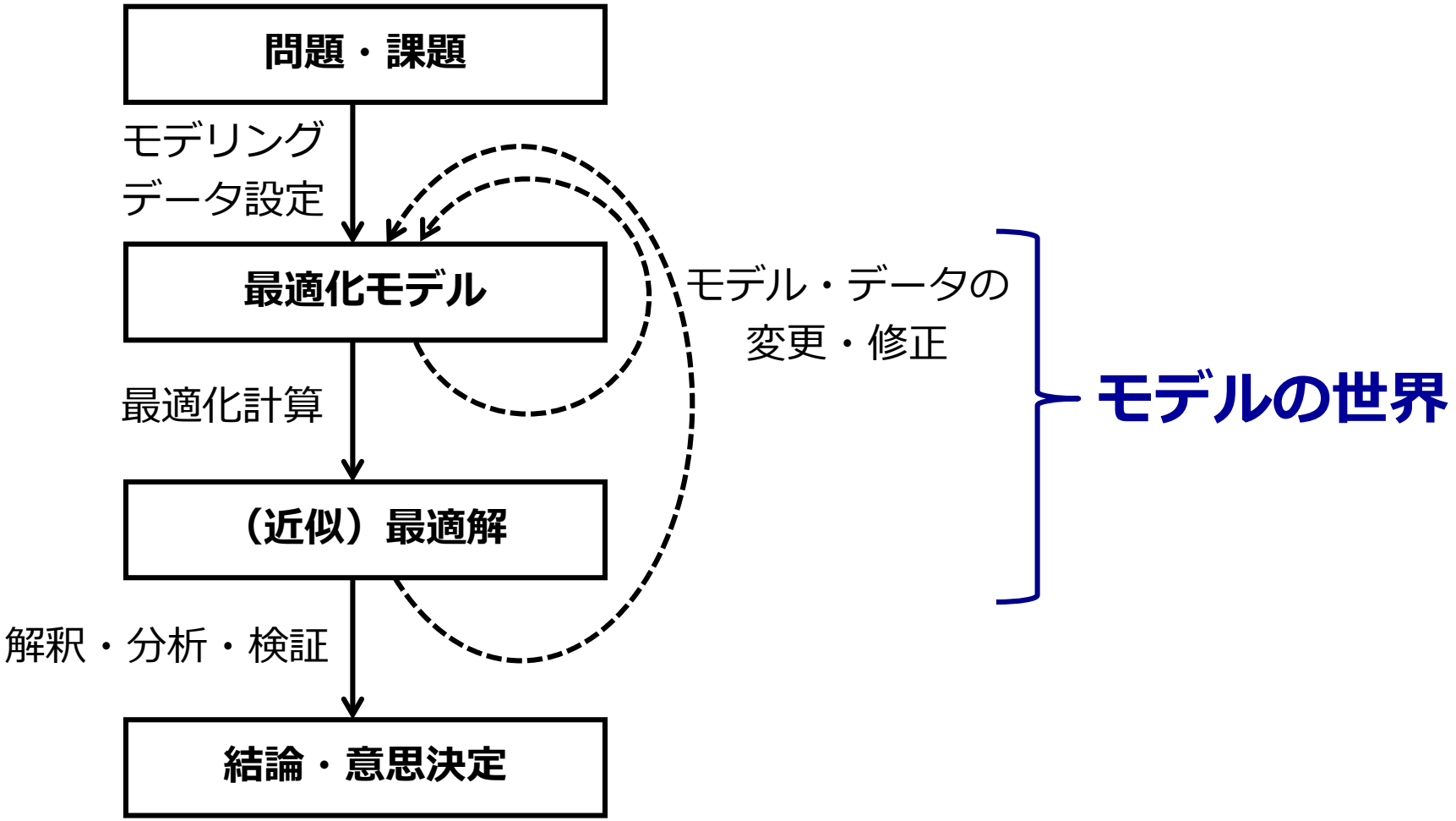
Numerical Optimizer

※2013 年 11 月 リリースの V16 より
NUOPT から改名致しました

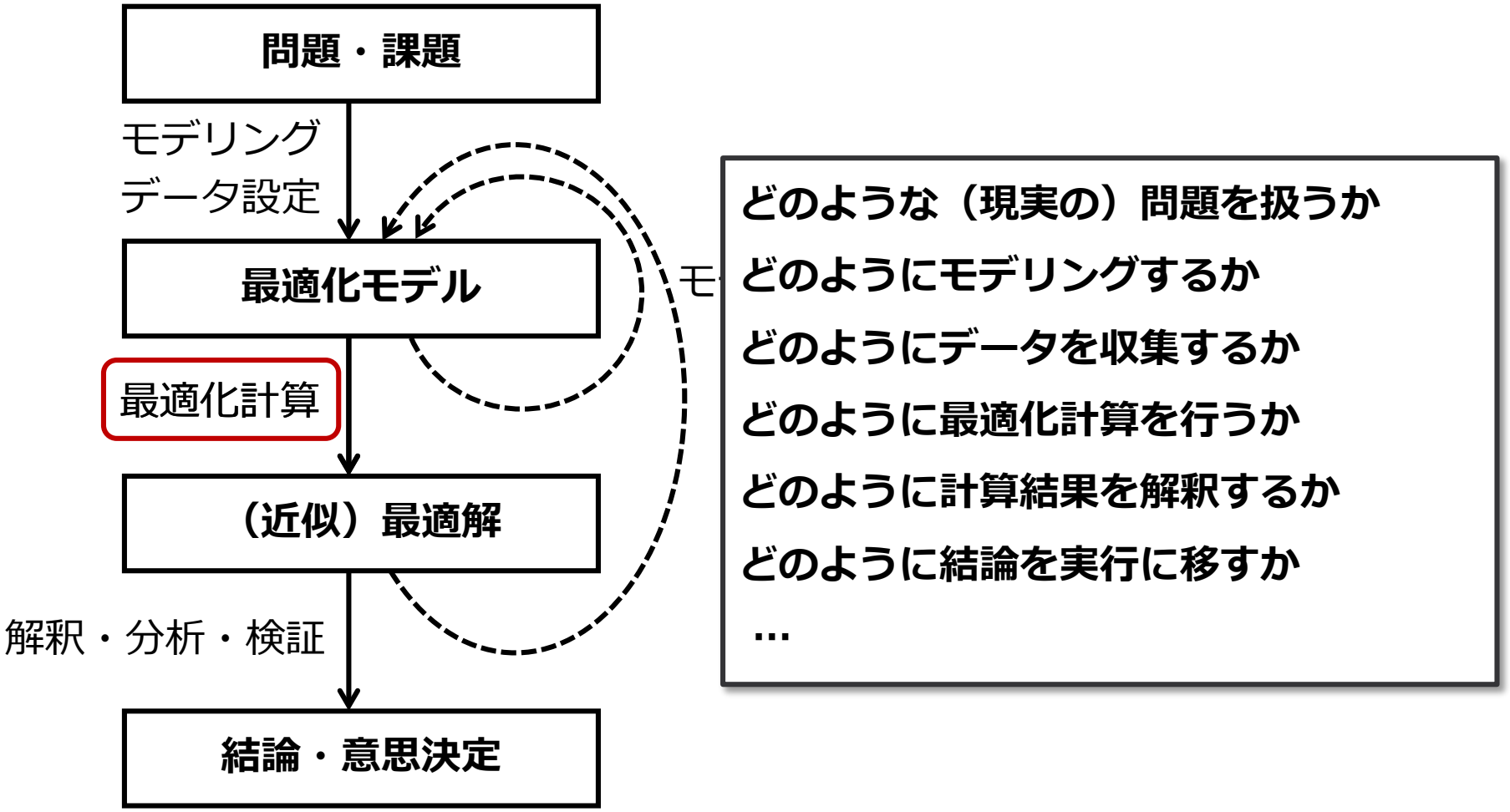
概要

- **研究の背景・目的・方法**
- **WCSPソルバー**
  - 重み付き制約充足問題
    （Weighted Constraint Satisfaction Problem）
  - アルゴリズムの概要
- **適用事例**
  - 国際コンペティション

最適化アプローチによる問題解決
（最適化ソリューション）



最適化アプローチによる問題解決
（最適化ソリューション）



最適化計算：問題解決・意思決定過程における１つのステップ

最適化問題

種々の制約のもと，
ある与えられた関数を最小化／最大化する問題の総称

$$\begin{aligned} &\text{minimize} && f(\boldsymbol{x}) \\ &\text{subject to} && \boldsymbol{x} \in F \end{aligned}$$

- $f(x)$：目的関数
- F：実行可能領域

- 解が組合せ的構造をもつ問題（**組合せ最適化問題**）を想定
  - **実社会において広く現れる問題構造**

最適化の方法

（１）既存のソルバーを利用

- **混合整数計画**（Mixed Integer Programming, MIP）

  線形等式・不等式制約のもとで，１次関数を最小化／最大化する問題
  変数は実数もしくは整数

  MIPソルバー

  商用・非商用を含め多くのソフトウェアが存在

  - **モデリング言語**が提供されていることが多く，**問題記述が容易**
  - **定式化の工夫**が必要になることも多い
  - **実用的に解ける問題の規模が限定的**であることも多い
    - 現実に解くべき組合せ最適化問題の多くは NP困難
    - 厳密な最適解を効率よく計算することはおそらく不可能：P≠NP予想

最適化の方法

（２）ソルバーを個別に開発

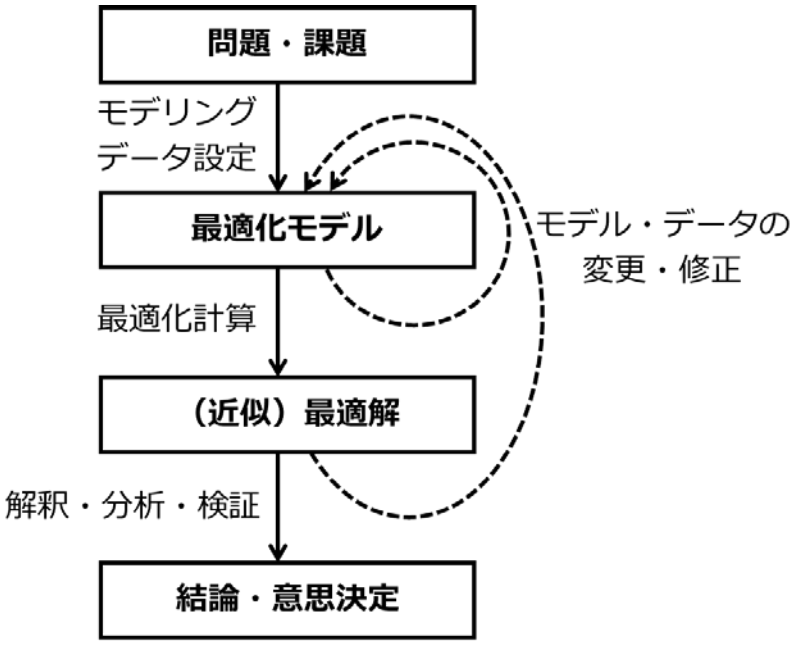
- **方針１**：厳密な最適解を計算することを目指す
- **方針２**：厳密性にこだわらず，近似最適解で構わないと考える
  - 一定時間内に実用上許容可能な解が計算できればよい

最適化の方法

（１）既存のソルバーを利用

（２）ソルバーを個別に開発

- **選定のポイント**
  - 最適化計算の実現にかけられる**手間・時間・費用**
  - **計算時間と解の質のトレードオフ**
    - 実務的観点からの要求・要望
    - 問題の難しさ（**計算の複雑さ**）
  - **解法の特徴・性質**



研究目的・方法

いろいろな問題に適用可能な汎用ソルバーを開発し，
実行可能なソフトウェアとして提供

- 対象を**組合せ最適化問題**（静的・確定的）に限定
- **実務利用**を念頭
  - **良質の解を実用的な計算時間**で出力することを目指す
    （理論的な最適性は保証しない）
  - アルゴリズム設計の枠組みや指針・テンプレートの提供ではなく，
    実装済みのアルゴリズム（ソルバー）を提供

- **方法**
  - **標準問題によるアプローチ**
  - **メタヒューリスティクス**

方法：標準問題によるアプローチ

いくつかの標準問題をソルバーとともに用意

1．解きたい問題のタイプに応じて標準問題を選択

2．標準問題の形に記述してソルバーを適用

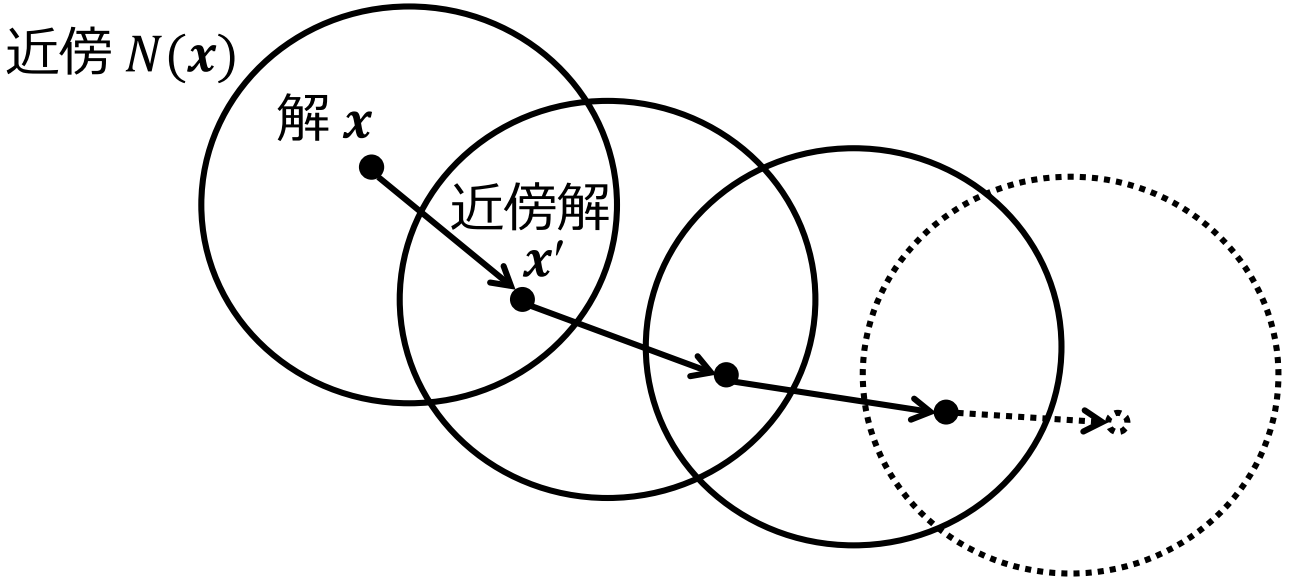
- **標準問題の例**
  - 混合整数計画問題
    （Mixed Integer Programming, MIP）
  - **重み付き制約充足問題**
    （Weighted Constraint Satisfaction Problem , WCSP）
  - 資源制約付きプロジェクトスケジューリング問題
    （Resource-Constrained Project Scheduling Problem, RCPSP）
  - …

［参考文献］ 茨木俊秀：「問題解決エンジン」への道，応用数理，Vol.14, pp.67-70 (2004).

方法：メタヒューリスティクス

計算困難な最適化問題に対する実用的アプローチのひとつ

- 様々な手法の総称
  多くは**局所探索法（ローカルサーチ）**の拡張と解釈できる
- アルゴリズムの**「枠組み」**に過ぎない
  ➔ **設計・実装を適切に行うことが必要**



概要

- 研究の背景・目的・方法
- **WCSPソルバー**
  - 重み付き制約充足問題
    （Weighted Constraint Satisfaction Problem）
  - アルゴリズムの概要
- 適用事例
  - 国際コンペティション

制約充足問題（CSP）

- n個の**変数** X_i $(i = 1, 2, \ldots, n)$
- X_i がとり得る**値の集合** D_i $(i = 1, 2, \ldots, n)$ ―**有限離散集合を仮定**
- m個の**制約** C_k $(k = 1, 2, \ldots, m)$

すべての制約を満たすように
各変数 X_i に値 $j \in D_i$ を１つずつ割当てることが目的

0-1変数 $x_{ij} = \begin{cases} 1, & X_i = j \\ 0, & X_i \neq j \end{cases} (i = 1, 2, \ldots, n,\ j \in D_i)$

解（割当て） $\boldsymbol{x} = (x_{ij} \mid i = 1, 2, \ldots, n,\ j \in D_i)$

$$\text{s.t.} \sum_{j \in D_i} x_{ij} = 1,\ i = 1,2, \ldots, n$$

例： $n = 4$, $D_1 = D_2 = D_3 = D_4 = \{1, 2, 3\}$

$\boldsymbol{x} = (0,1,0|1,0,0|0,1,0|0,0,1)$ $\leftarrow (X_1, X_2, X_3, X_4) = (2,1,2,3)$

- **すべての制約を満たす解 x が存在するとは限らない**
  存在するとしても，見つけることは計算困難

重み付きCSP（WCSP）

それぞれの制約 C_k について

- 違反度に応じた**ペナルティ** $p_k(\boldsymbol{x})$（制約を満たすとき 0）
- 重要度に応じた**重み** $w_k \geq 0$

ペナルティの加重和 $p(\boldsymbol{x}) = \sum_k w_k\, p_k(\boldsymbol{x})$ を最小化

- 絶対制約を満たしたうえで，考慮制約の違反度を最小化
- **目的関数 $f(\boldsymbol{x})$ を最小化する場合**
  ペナルティ関数 $p_k(\boldsymbol{x}) := f(\boldsymbol{x}) - LB$ の考慮制約（LB：$f(\boldsymbol{x})$ の下界値）

いろいろなタイプの割当て問題を定式化可能

- MIPなどと比べ，**より簡潔**に，**より自然な形**で制約を記述可能
  - 人為変数（本来の決定変数ではない変数）の導入を回避
    → 効率的な最適化計算

代表的な制約・ペナルティ関数

- **0-1変数 x_{ij} に関する線形または２次等式・不等式**
  - 「$g(\boldsymbol{x}) \leq b$」の形の不等式制約に対しては

    ペナルティ $:= \max\{0, g(x) - b\}$

（参考）

２次関数 $\sum_{i_1 j_1 i_2 j_2} \sum \sum \sum \sum a_{i_1 j_1 i_2 j_2} x_{i_1 j_1} x_{i_2 j_2}$ を目的関数として最小化する場合

各項に対して，

重み $w_k := a_{i_1 j_1 i_2 j_2}$ の線形考慮制約「$x_{i_1 j_1} + x_{i_2 j_2} \leq 1$」

を追加することでも記述可能

代表的な制約・ペナルティ関数

- **0-1変数 x_{ij} に関する線形または２次等式・不等式**
  - 「$g(\boldsymbol{x}) \leq b$」の形の不等式制約に対しては

    ペナルティ $:= \max\{0, g(x) - b\}$

- **all-different 制約**

  「与えられた変数集合 V に含まれる変数は
  すべて異なる値をとらなくてはならない」

  - ペナルティ $:= |V| - (V$ に含まれる変数がとっている値の種類数$)$

「制約を定義」＝「ペナルティ関数を定義」

- 個々のペナルティ関数を**単独で最小化することは容易**であると仮定

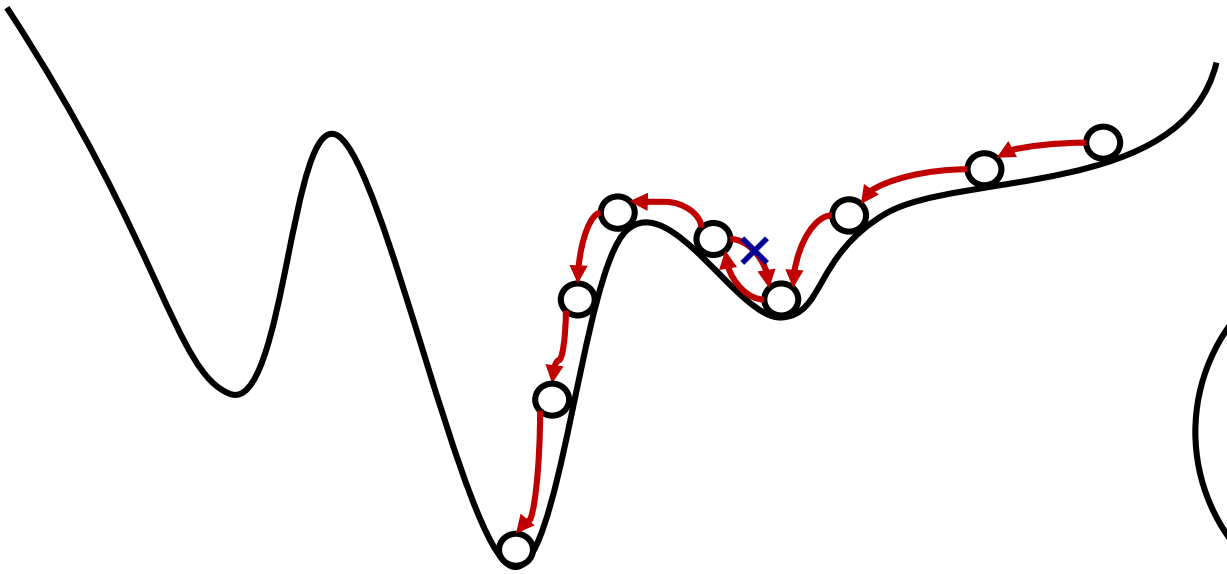
例：ナーススケジューリング問題

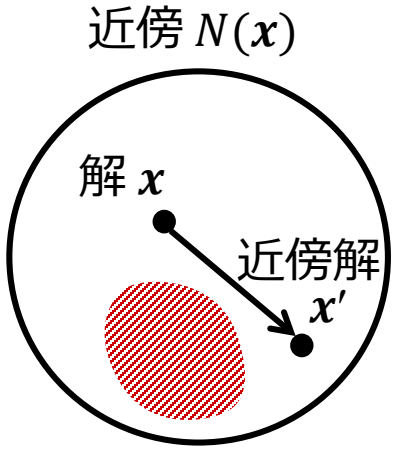
- **各ナースの毎日のシフトを決定**
- **制約**
  - **シフト拘束制約**
    - 適正人数のナースを確保
  - **ナース拘束制約**
    - ナースの労働負荷や勤務希望を考慮
- **WCSPへの定式化**
  - 変数：（ナース，日にち）の各組に対応
  - 領域： シフトの集合

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
ナース1	準	夜	夜	休	休	休	日	日	日	日	休	日	日	日	休	休	日	夜	夜	休	休	日	日	準	準	準	休	休	日	日
ナース2	夜	休	休	日	準	準	準	休	休	日	日	日	会	休	日	日	休	休	日	準	休	日	日	日	日	夜	夜	休	準	夜
ナース3	日	休	休	日	日	日	休	夜	夜	休	休	休	日	準	準	夜	夜	休	休	日	日	準	準	夜	夜	休	休	日	日	日
ナース4	休	日	日	準	休	日	日	準	準	夜	夜	休	休	日	日	準	準	準	休	会	日	休	日	日	休	休	日	夜	夜	休
ナース5	休	日	日	夜	夜	休	休	日	日	準	準	夜	夜	休	休	日	日	日	準	夜	夜	休	休	休	日	日	準	準	休	準
ナース6	日	準	準	休	日	夜	夜	休	休	休	日	準	準	夜	夜	休	休	日	日	日	準	夜	夜	休	休	日	日	日	休	休
ナース7	日	休	夜	夜	休	休	日	日	準	準	休	日	日	休	休	日	日	準	準	休	日	日	日	日	夜	夜	休	休	日	日
ナース8	夜	夜	休	準	準	休	日	日	日	会	夜	夜	休	休	日	日	準	休	休	日	日	休	日	準	休	休	日	準	休	日
ナース9	休	日	日	日	休	休	日	日	日	準	休	日	準	準	準	休	日	日	準	夜	夜	休	休	日	日	休	休	休	休	日
ナース10	会	休	日	日	休	準	準	休	日	日	日	準	休	日	日	会	休	日	日	日	休	準	準	休	日	準	夜	夜	休	休
ナース11	準	休	日	日	日	日	夜	夜	休	休	日	日	日	日	休	会	日	休	休	休	休	日	日	日	準	休	日	日	準	準
ナース12	休	日	日	日	夜	夜	休	休	休	日	日	休	日	日	日	会	休	休	休	日	準	休	日	日	日	日	準	休	日	日
ナース13	日	準	準	休	日	日	会	準	休	休	休	日	準	休	日	日	休	日	夜	夜	休	休	日	日	休	休	日	日	夜	夜
ナース14	準	休	日	準	準	休	日	日	日	会	休	休	準	準	休	日	夜	夜	休	休	日	日	日	日	準	休	休	日	夜	夜
ナース15	日	休	休	日	夜	夜	休	休	休	日	準	準	夜	夜	休	休	日	日	日	準	夜	夜	休	休	日	日	日	準	準	準
ナース16	日	日	夜	夜	休	休	休	日	準	準	休	日	会	休	日	準	準	休	休	日	会	日	夜	夜	休	準	準	休	日	日
ナース17	休	準	準	休	日	準	夜	夜	休	休	日	日	会	日	夜	夜	休	準	準	休	日	準	休	日	夜	夜	休	休	日	日
ナース18	休	日	日	日	休	日	準	準	夜	夜	休	休	日	日	準	休	日	日	日	会	準	休	準	準	休	日	夜	夜	休	休
ナース19	日	日	日	準	休	日	会	休	休	日	日	休	日	準	休	会	日	日	日	日	休	日	準	休	休	休	日	日	休	日
ナース20	日	準	休	休	日	日	日	日	夜	夜	休	休	日	日	日	日	休	休	休	日	日	休	日	日	準	休	準	準	休	会
ナース21	休	日	準	休	休	日	日	休	日	日	休	日	夜	夜	休	休	日	準	休	準	準	休	日	日	日	日	休	日	日	会
ナース22	会	休	日	日	準	休	日	日	休	準	準	準	休	日	日	日	夜	夜	休	休	休	日	日	準	休	日	日	休	休	会
ナース23	休	休	日	日	休	準	準	準	休	休	日	日	日	日	夜	夜	休	休	日	日	日	準	休	休	日	準	休	日	日	会
ナース24	準	休	休	日	日	日	休	休	日	日	日	日	準	休	準	準	休	日	準	休	日	日	夜	夜	休	休	日	日	日	日
ナース25	夜	夜	休	休	日	日	日	日	準	休	夜	夜	休	休	日	日	準	休	夜	夜	休	休	日	日	日	休	日	日	準	準

WCSPソルバーの概要

- **基本的な枠組みはタブーサーチ（局所探索法の拡張）**
  - 近傍内の最良解に移動（改悪解への移動もあり得る）
  - **近傍解のすべてを移動先の候補とはしない**
    ＝ 近傍内に移動禁止の（「タブー」な）領域がある





［参考文献］ K. Nonobe and T. Ibaraki: An improved tabu search method for the weighted constraint satisfaction problem, *INFOR* 39, pp.131–151 (2001).

WCSPソルバーの概要

- **基本的な枠組みはタブーサーチ（局所探索法の拡張）**
  - 近傍内の最良解に移動（改悪解への移動もあり得る）
  - **近傍解のすべてを移動先の候補とはしない**
    ＝ 近傍内に移動禁止の（「タブー」な）領域がある

- **基本要素**

探索空間	解（割当て）集合全体
近 傍	**ある１つの変数の値を変更**することで得られる解の集合
タブーリスト	解移動において**値を変更した変数**をしばらくの間保持 リスト内の変数の値は変更不可
初期解	ランダム割当てを欲張り法で改善

［参考文献］ K. Nonobe and T. Ibaraki: An improved tabu search method for the weighted constraint satisfaction problem, *INFOR* 39, pp.131–151 (2001).

WCSPソルバーの概要：特長・特徴

近傍解評価のための
補助メモリ利用

探索の高速化

近傍探索領域の限定

効果的な探索

プログラム・パラメータの
動的調整

利便性／探索の高性能化

ユーザ定義制約
追加の仕組み

拡張性

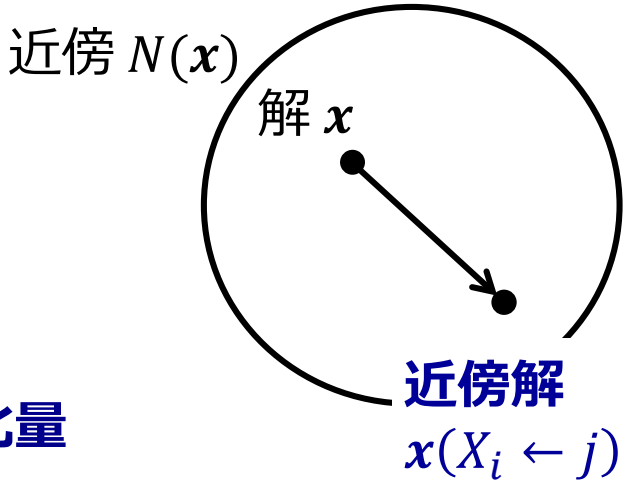
近傍解評価のための補助メモリ利用

$\boldsymbol{x}$：**現在の解**

$\boldsymbol{x}(X_i \leftarrow j)$：$X_i$の値を$j$に変更することで得られる近傍解

例：　$\boldsymbol{x} = (0,1,0|1,0,0|0,1,0|0,0,1)$

$\boldsymbol{x}(X_3 \leftarrow 3) = (0,1,0|1,0,0|0,0,1|0,0,1)$



- **近傍解の評価**
  - $X_i, j\ (\in D_i)$ の組それぞれに対して，
    X_iの値を j に変更したときの**ペナルティ変化量**

    $$\Delta p(\boldsymbol{x}, i, j) := p\big(\boldsymbol{x}(X_i \leftarrow j)\big) - p(\boldsymbol{x})$$

    を事前に計算・保持

    → 定数時間で近傍解を１つ評価することが可能
  - **解の移動ごとに情報の更新が必要**

ペナルティ変化量の保持

- 例：$\boldsymbol{x} = (0,1,0|1,0,0|0,1,0|0,0,1)$

	X_1			X_2			X_3			X_4		
	1	2	3	1	2	3	1	2	3	1	2	3
$\Delta p(\boldsymbol{x},\cdot,\cdot)$	4	0	3	0	−2	0	3	0	5	1	2	0

ペナルティ変化量 $\Delta p(\boldsymbol{x}, i, j)$

$$= p\big(\boldsymbol{x}(X_i \leftarrow j)\big) - p(\boldsymbol{x})$$

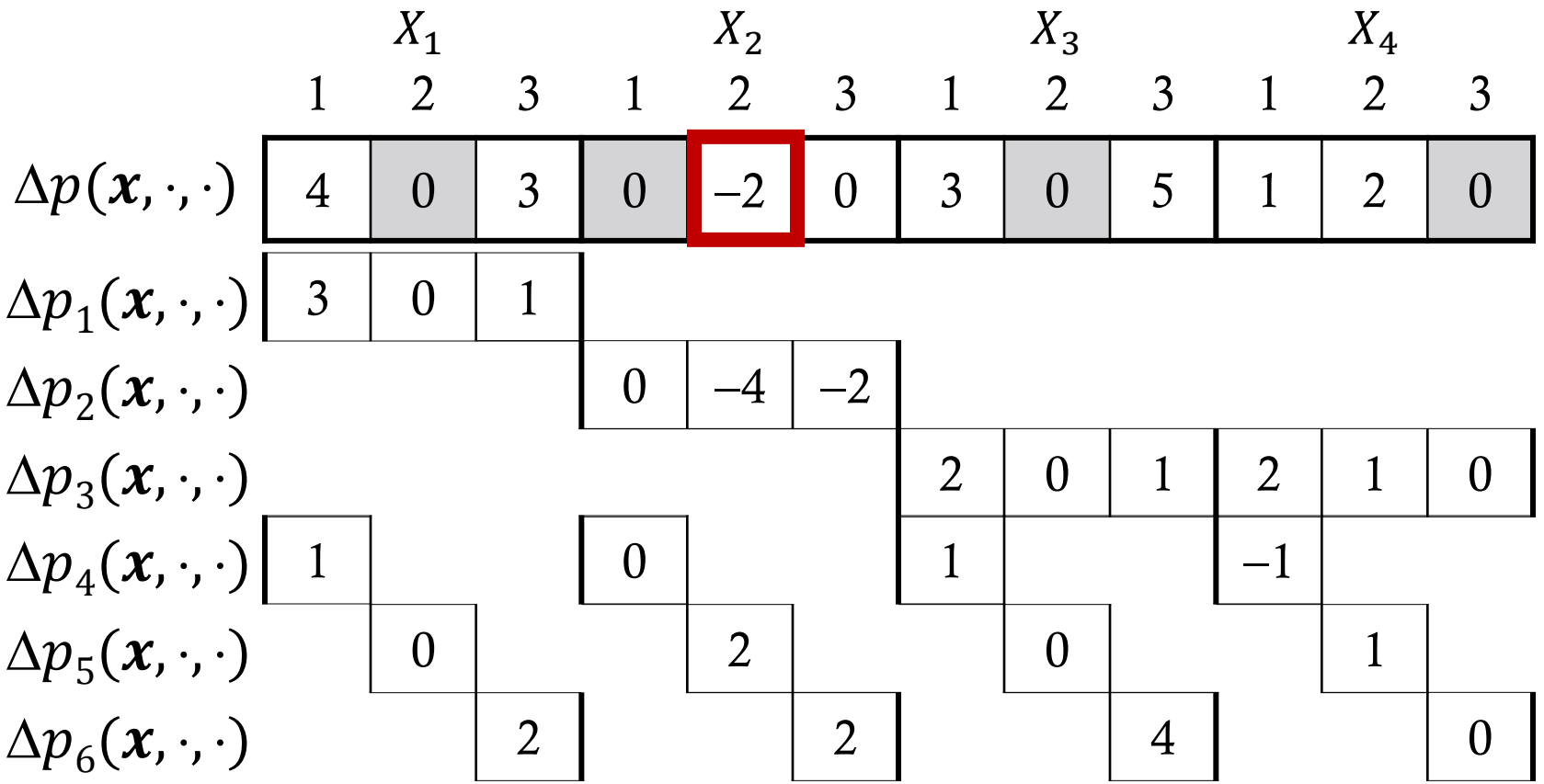
$$= \sum_k w_k p_k\big(\boldsymbol{x}(X_i \leftarrow j)\big) - \sum_k w_k p_k(\boldsymbol{x})$$

$$= \sum_k w_k \{p_k\big(\boldsymbol{x}(X_i \leftarrow j)\big) - p_k(\boldsymbol{x})\}$$

$$= \sum_k w_k \Delta p_k(\boldsymbol{x}, i, j)$$

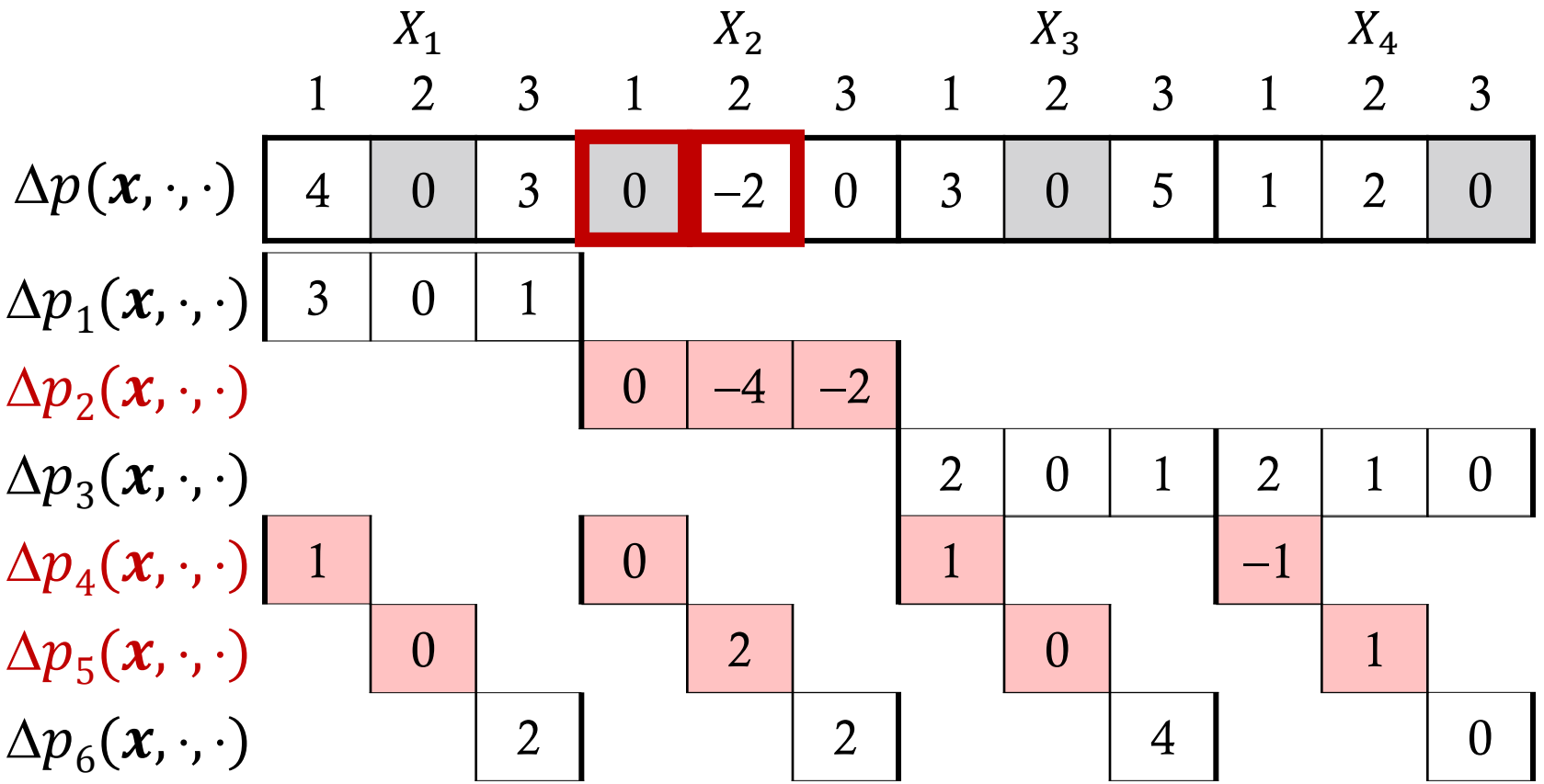
ペナルティ変化量の保持

- 例：$\boldsymbol{x} = (0,1,0|1,0,0|0,1,0|0,0,1)$
  $\rightarrow (0,1,0|\mathbf{0}, \mathbf{1}, 0|0,1,0|0,0,1)$



ペナルティ変化量の更新

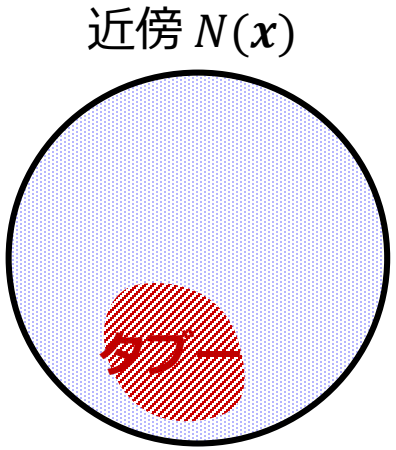
- 例：$\boldsymbol{x} = (0,1,0|1,0,0|0,1,0|0,0,1)$
  $\rightarrow (0,1,0|\mathbf{0}, \mathbf{1}, 0|0,1,0|0,0,1)$

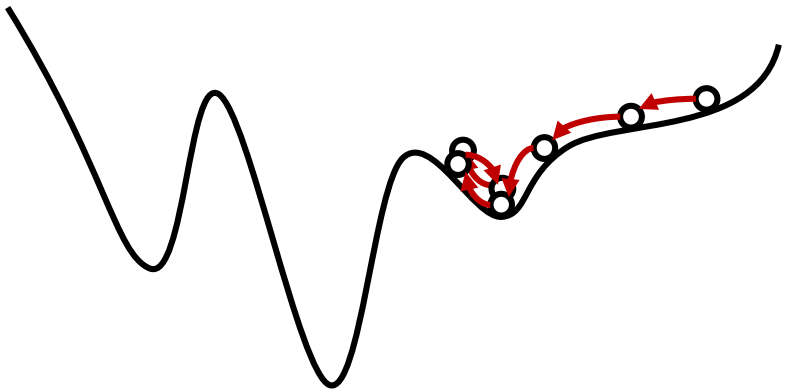


近傍探索領域の限定

- **すべての（タブーでない）近傍解をチェックすることは非効果的**
  - 「多くの計算時間を要する」ことだけが問題ではない
  - タブーサーチは「（タブーでない）最良の近傍解への移動」が基本
    - **ペナルティにほとんど影響を与えない「軽微な」修正ばかりが繰り返され，「山を越える」改善が起こりにくい**

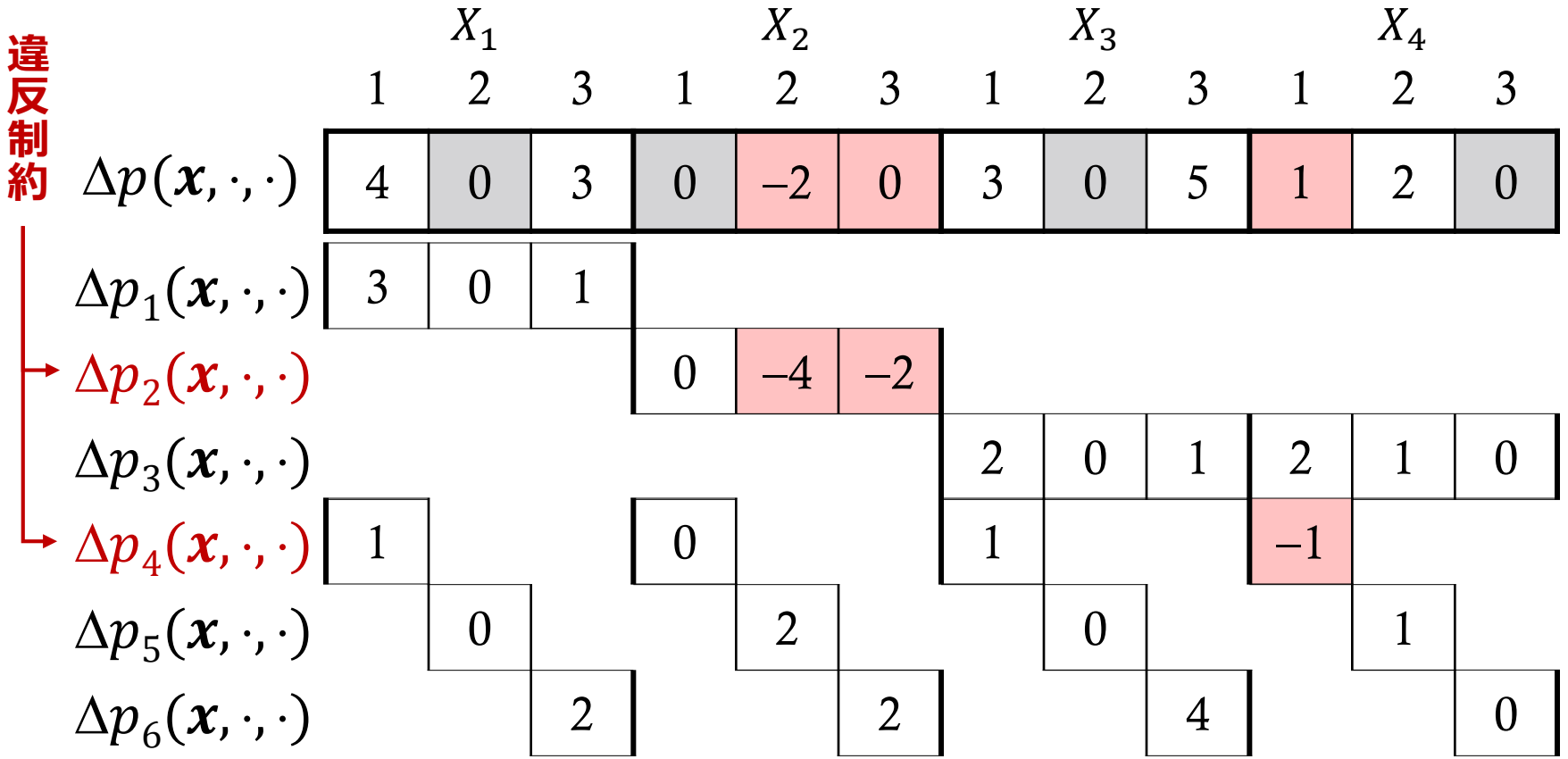
現在の解が満たしていない制約の少なくとも１つについてペナルティ値を減少させるような解移動のみを候補とする





近傍探索領域の限定

- 例：$\boldsymbol{x} = (0,1,0|1,0,0|0,1,0|0,0,1)$



パラメータの自動調整 (1)

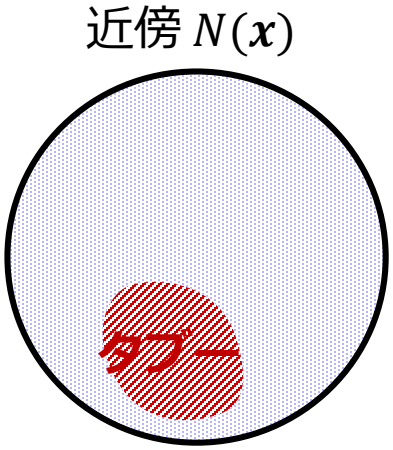
- **タブー期間（tabu tenure）**

  タブーリストにより，変数の値の変更が禁止される期間

  - **タブー期間が短い場合**
    - 緻密な探索（集中化）
    - 偏った探索

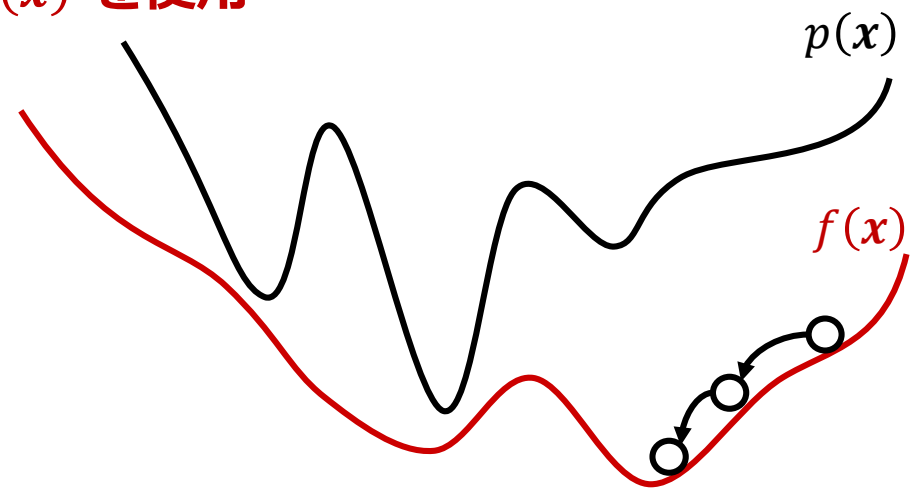
    → 探索履歴から判断し，タブー期間増
  - **タブー期間が長い場合**
    - 解の循環防止，探索の多様化
    - 過剰な禁止

    → 特別選択規則（aspiration criteria）により判断し，タブー期間減



パラメータの自動調整 (2)

- **ペナルティ重み**
  - **ペナルティ関数 $p(x) = \sum_k w_k p_k(x)$ を**
    そのまま近傍解の評価関数として用いる探索は非効果的
    - 近傍解の良し悪しが重みの大きな制約に強く依存
      その他の制約は軽視される傾向
      ➔ 重みの大きな制約を満たす解に探索が制限
  - **近傍解の評価には $f(x) = \sum_k v_k p_k(x)$ を使用**
    - $0 < v_k \leq w_k$
    - v_k **の値を自動調整**



パラメータの自動調整 (2)

近傍解の評価には $f(\boldsymbol{x}) = \sum_k v_k\, p_k(\boldsymbol{x})$ を使用

- **v_k の値の自動調整**

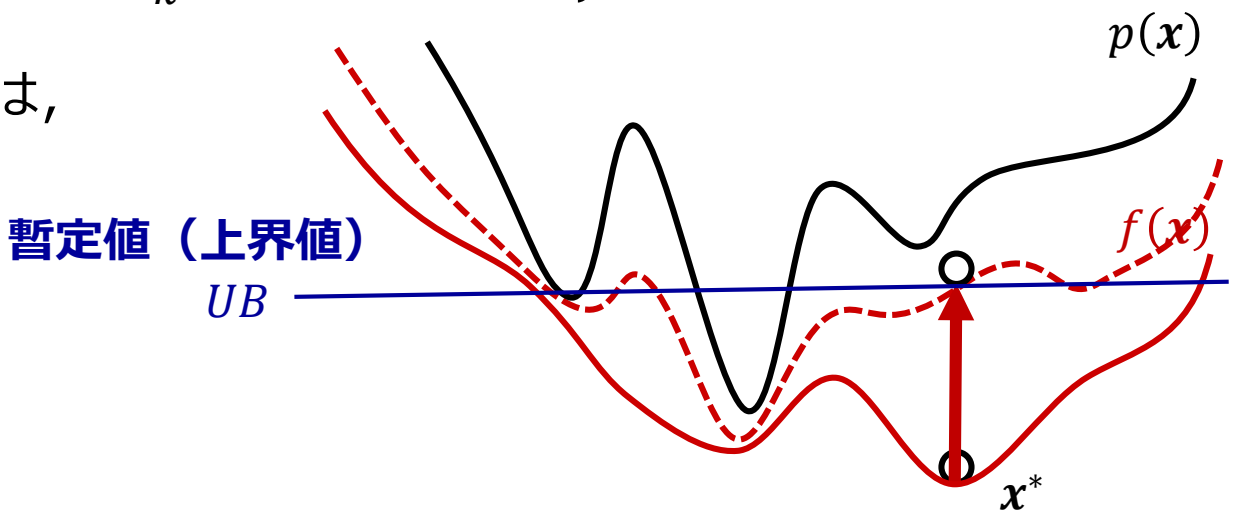
  ラグランジュ緩和法における劣勾配法のアイデアを利用

  $\boldsymbol{x}^*$：前回の v_k 調整後に得られた，「$f(\boldsymbol{x})$ 最小化」の意味での最良解

  - $p_k(\boldsymbol{x}^*) > 0$ である制約に対しては，v_k を増加

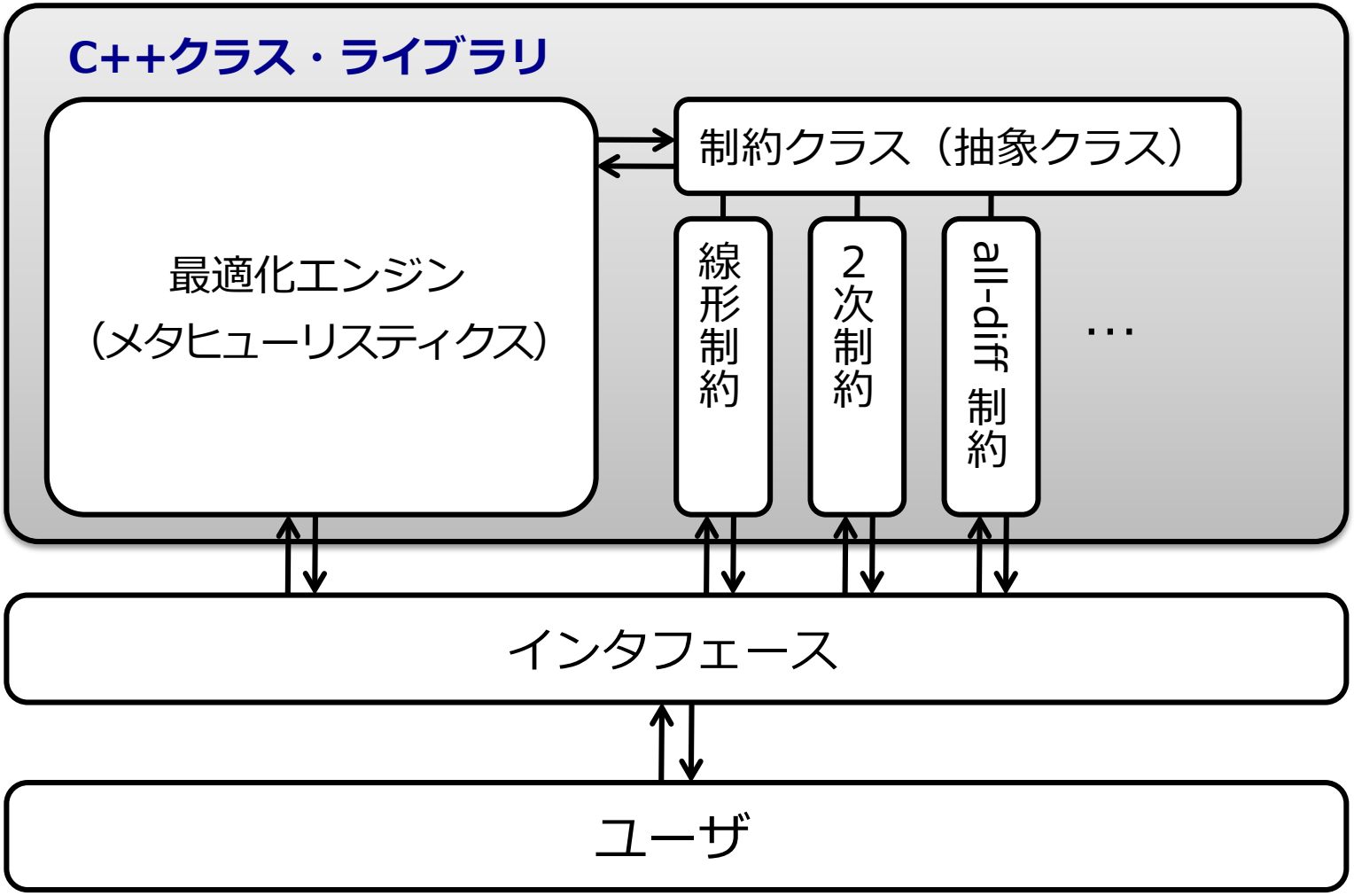
$$v_k := \min\left\{w_k, v_k + \frac{UB - f(\boldsymbol{x}*)}{\sum_k (p_k(\boldsymbol{x}^*))^2}\, p_k(\boldsymbol{x}^*)\right\}$$

  - $f(\boldsymbol{x}^*) \geq UB$ の場合は，
    すべての v_k を減少



ソルバーの概要

最適化ソルバー



ユーザ定義制約の追加

「制約を定義」＝「ペナルティ関数を定義」

実装上は，以下の関数を用意

- 与えられた解 $\boldsymbol{x}$ に対して，**ペナルティ $p_k(\boldsymbol{x})$ を計算し，**
  ペナルティ変化量 $\Delta p_k(\boldsymbol{x}, i, j)$ を列挙（非零のみ）

- **制約に関与している**（＝ペナルティ計算時に値を参照する必要のある）
  0-1変数を列挙

概要

- 研究の背景・目的・方法
- WCSPソルバー
  - 重み付き制約充足問題
    （Weighted Constraint Satisfaction Problem）
  - アルゴリズムの概要
- **適用事例**
  - 国際コンペティション

適用事例

- **Numerical Optimizer（旧NUOPT） の一機能として多くの問題に適用**
- **国際コンペティション**
  - International Timetabling Competition 2007 (ITC2007)
  - International Nurse Rostering Competition 2010 (INRC2010)
- …

適用例：時間割作成

ITC2007 (International Timetabling Competition)

- **トラック１（試験時間割）** ３位
  - 試験数：200～1000
  - ピリオド数：20～80, 教室数：1～50
  - 学生数：5000～16000
- **トラック２（履修登録に基づいた授業時間割）** ２位
  - 授業数： 200～400
  - ピリオド数：45, 教室数：5～20
  - 学生数：300～1000
- **トラック３（カリキュラムに基づいた授業時間割）** ３位
  - 教科数： 150～450
  - ピリオド数：25～45, 教室数：5～20

計算時間：約５～10分

［参考文献］茨木俊秀，熱田光紀，野々部宏司：汎用ソルバによる時間割作成―国際コンペティションITC2007に参加して―，スケジューリング・シンポジウム2008講演論文集，pp.173－176 (2008).

適用例：ナーススケジューリング

INRC2010 (International Nurse Rostering Competition)

- **３つのトラック：sprint, medium, long**
  - ナース数：10 (sprint), 30～31 (medium), 49～50 (long)
  - 日数：28
  - シフト数：4～5 (sprint), 5～6 (medium), 6 (long)
  - 計算時間：10秒 (sprint), 10分 (medium), 10時間 (long)
  - 制約数：　約1800～3600 (sprint)
約3300～11,500 (medium)
約7200～21,500 (long)

［参考文献］野々部宏司：メタヒューリスティクスを用いた制約最適化ソルバーのナーススケジューリング問題への適用，電気学会研究会資料，システム研究会，ST-10-6，pp.27－31 (2010).

INRC2010：問題概要

- **シフト拘束制約：絶対制約**
  - 各日，各シフトについて，指定された人数のナースを**過不足なく確保**
- **ナース拘束制約：考慮制約（各制約に重み）**

  制約の有無，上下限値，週末の定義はナースによって異なる
  - 各シフトについて，スケジュール期間中の**割当て回数**に上下限
  - **連続勤務日数**／**連続休暇日数**に上下限
  - **完全休暇ではない週末の連続数**に上限
  - **同じ週末**に含まれる日には**同じシフト**
  - **夜勤終了後，2日間の休暇日**
  - **望ましくないシフトの並び**を禁止（夜勤の翌日に日勤など）
  - いくつかの日に，**割当て指定シフト**／**割当て禁止シフト**
- **考慮制約違反に対するペナルティの加重和を最小化**

INRC2010：コンペティション結果

https://www.kuleuven-kulak.be/nrpcompetition/competitor-ranking

まとめ

- **実務利用を念頭に置いた汎用ソルバーの開発**
  - WCSPに対するメタヒューリスティックアルゴリズム
- **多数の適用事例**