MIZUHO

Channel to Discovery

ICML2007を読む会（2007年8月20日）

“Entire Regularization Paths for Graph Data” (Koji Tsuda) の論文紹介

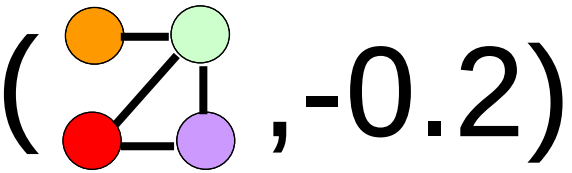
紹介者：浜田道昭
みずほ情報総研（株）

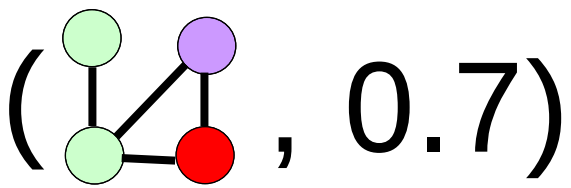
論文の概要

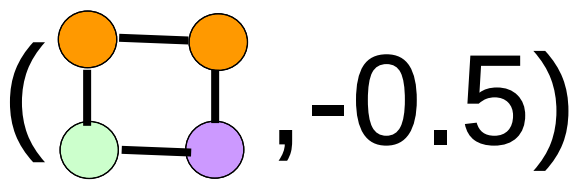
- グラフデータのL1-regularized回帰問題のregularization path（λを変化させたときの重みベクトルのパス）を完全かつ効率的にトレースするアルゴリズムを提案
  - 基本となるのはLAR-LASSOアルゴリズム
- 計算コストの大きな部分はパターンの探索問題に帰着
  - ナイーブに探索すると計算が爆発する
- 効率的な枝狩り条件の提案
- 計算機実験による効率化の検証

グラフ回帰問題

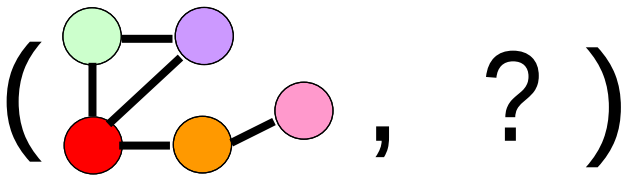
Training







Test

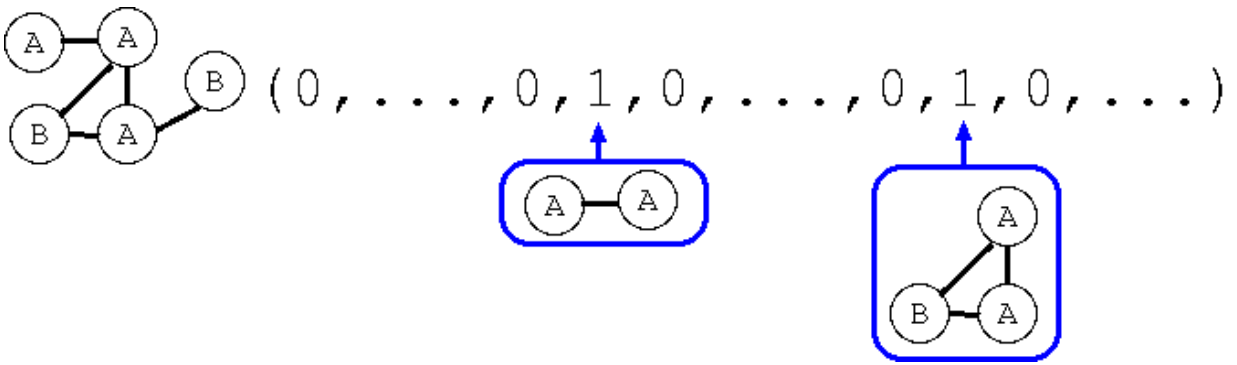


例えば化合物の毒性予測などはグラフ回帰問題と考えることができる

グラフのベクトル表現

- グラフデータ：$\mathcal{G} = \{G_i\}_{i=1}^{n}$
- 部分グラフ（パターン）の集合：T
- 各グラフG_iを以下の特徴ベクトルで表現

$$\boldsymbol{x}_i = (x_{it})_{t \in T} \quad x_{it} = I(t \subseteq G_i)$$



一般にグラフのベクトル表現は非常に高次元になる

LASSO回帰
LASSO regression

$X \in \Re^{n \times d}$ 行：データ, 列：特徴量を表す行列

$\boldsymbol{y} \in \Re^{n}$ 目的変数ベクトル

$\boldsymbol{\beta} \in \Re^{d}$ 重みベクトル

損失関数

$$\boldsymbol{\beta}(\lambda) = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} L(\boldsymbol{y}, X\boldsymbol{\beta}) + \lambda \|\boldsymbol{\beta}\|_1$$

$$L(\boldsymbol{y}, X\boldsymbol{\beta}) = \frac{1}{2}\sum_i (y_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i)^2$$

LASSO回帰
[Tibshirani 1996]

LASSO回帰はスパースな重みベクトルが得られることが知られている

βの0でない成分をActive set と呼ぶ

LASSO回帰
LASSO regression

$X \in \Re^{n \times d}$ 行：データ, 列：特徴量を表す行列

$\boldsymbol{y} \in \Re^{n}$ 目的変数ベクトル

$\boldsymbol{\beta} \in \Re^{d}$ 重みベクトル

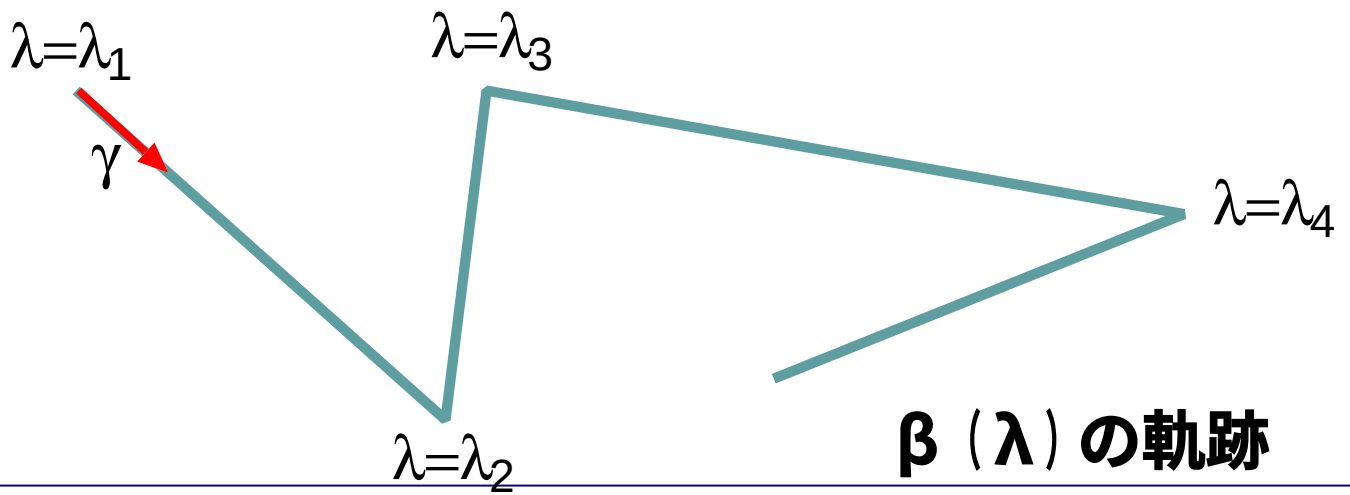
$$\boldsymbol{\beta}(\lambda) = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} L(\boldsymbol{y}, X\boldsymbol{\beta}) + \lambda \|\boldsymbol{\beta}\|_1$$

Regularization path：λを∞から0まで変化させたときのβ（λ）の軌跡

Efronらは**LASSO回帰の**Regularization path**が区分的線形**であることを示し、そのパスを完全にトレースするためのアルゴリズム（LAR-LASSO**アルゴリズム**）を提案 [Efron *et al* 2004]

Regularization Path & LAR-LASSO アルゴリズム概要

- Turning pointでは以下の2つのイベントのいずれかが行われる
  - 新しい特徴をActive setに追加
  - Active setから特徴を1つ削除



LAR-LASSOアルゴリズム
[Efron *et al* 2003]

- 初期値の設定 $\beta \leftarrow 0$,
- 初期active set Aの探索 Initial search
- 方向ベクトル γ を設定
- 全ての特徴が含まれるまで以下を繰り返す
  - d_1 = 次ステップがinclusionの場合のステップ幅 Main search
  - d_2 = 次ステップがexclusionの場合のステップ幅
  - $d = \min(d_1, d_2)$
  - パスの移動: $\beta \leftarrow \beta + d\gamma$
  - active setを更新
  - 方向ベクトル γ の更新

(もっと正確に書いた) LAR-LASSOアルゴリズム

Algorithm 1 The LAR-LASSO algorithm

1: $\boldsymbol{\beta} = 0$, $\mathcal{A} = \text{argmax}_j |\nabla L(\boldsymbol{\beta})|_j$ Initial search
2: $\boldsymbol{\gamma}_{\mathcal{A}} = -\text{sgn}(\nabla L(\boldsymbol{\beta}))_{\mathcal{A}}$, $\boldsymbol{\gamma}_{\mathcal{A}^C} = 0$.
3: **while** $\max |L(\boldsymbol{\beta})| > 0$ **do**
4: $d_1 = \min\{d > 0 : |\nabla L(\boldsymbol{\beta} + d\boldsymbol{\gamma})_j| = |\nabla L(\boldsymbol{\beta} + d\boldsymbol{\gamma})_{\mathcal{A}}|, j \notin \mathcal{A}\}$.
5: $d_2 = \min\{d > 0 : (\boldsymbol{\beta} + \boldsymbol{\gamma})_j = 0, j \in \mathcal{A}\}$. Main search
6: Find step length: $d = \min(d_1, d_2)$
7: Take step: $\boldsymbol{\beta} \leftarrow \boldsymbol{\beta} + d\boldsymbol{\gamma}$
8: If $d = d_1$ then add the variable attaining the minimum to $\mathcal{A}$.
9: If $d = d_2$ then remove the variables such that $\beta_j = 0$ from $\mathcal{A}$.
10: $C = \frac{1}{2} \sum_i \boldsymbol{x}_{\mathcal{A},i} \boldsymbol{x}_{\mathcal{A},i}^{\top}$
11: $\boldsymbol{\gamma}_{\mathcal{A}} = C^{-1} \text{sgn}(\boldsymbol{\beta}_{\mathcal{A}}), \boldsymbol{\gamma}_{\mathcal{A}^C} = 0$.
12: **end while**

Main Search＝パターン探索問題

- **問題**：Active set に**含まれない**パターン t で以下の値が最小となるパターン t を見つけなさい

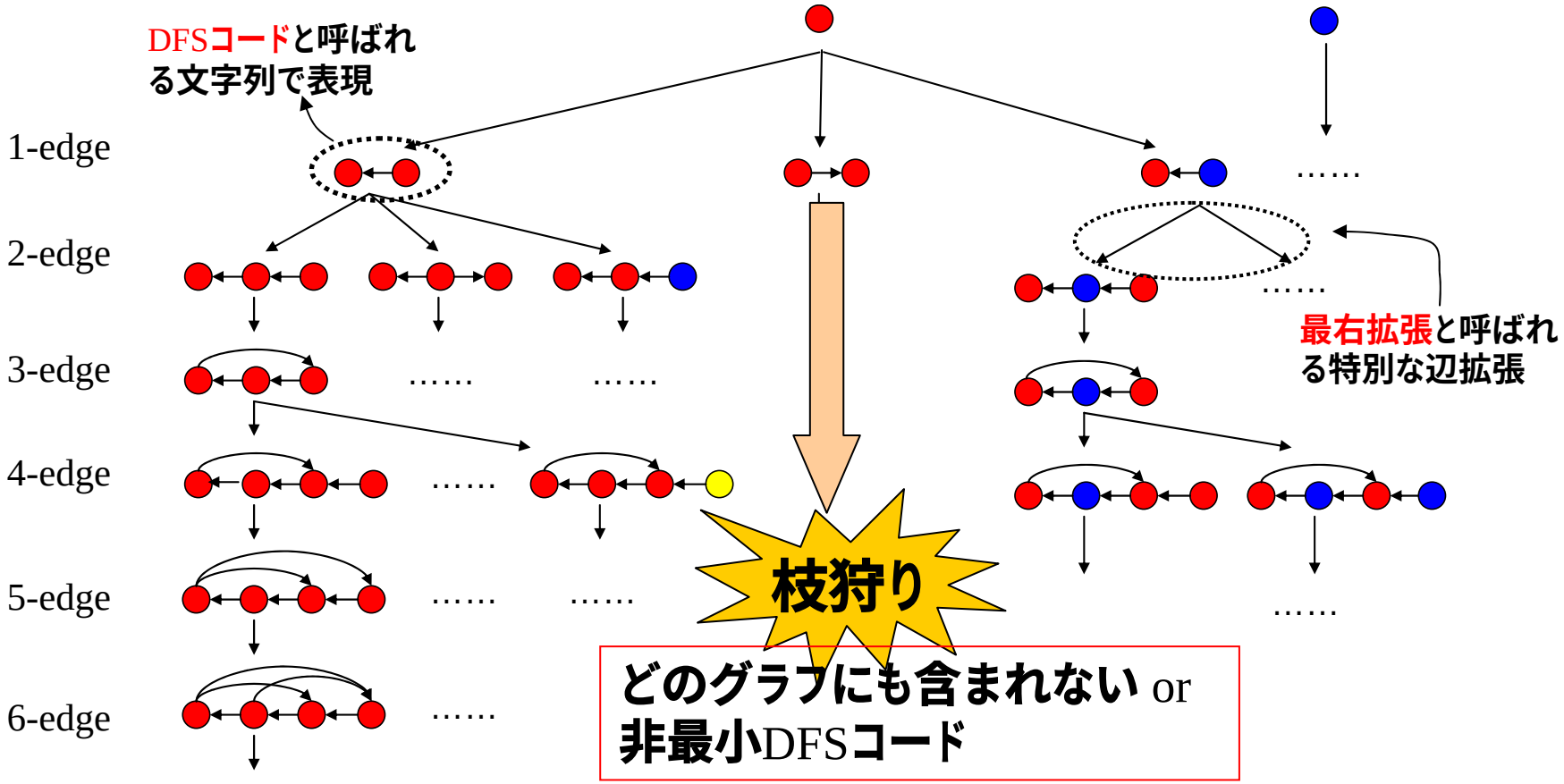
$$d_t = \min_{+} \left\{ \frac{\rho_0 - \sum_i w_i x_{it}}{\eta_0 - \sum_i v_i x_{it}}, \quad \frac{\rho_0 + \sum_i w_i x_{it}}{\eta_0 + \sum_i v_i x_{it}} \right\}.$$

w_i, v_i, ρ_0, η_0 ：**定数** (Active setに依存)

大量のパターン候補からパターンを探索しなくてはならない

基本となるパターン探索空間

DFSコード木 [Yan *et al* 2002]



パターンを重複無く完全に数え上げることが可能

提案する枝狩り条件

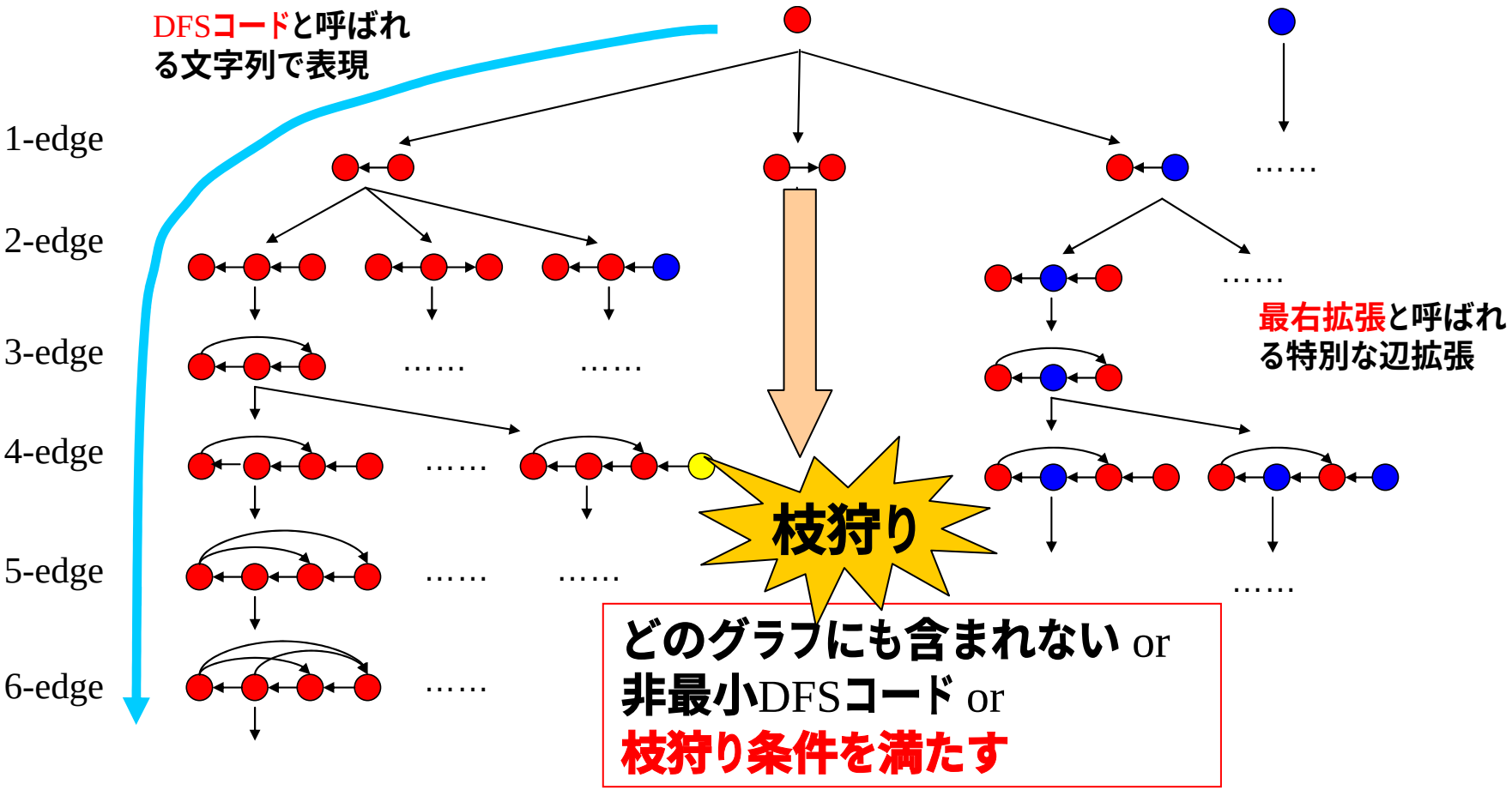
- パターン t まで探索
- d_t^*: 現在までに探索したパターンの d_t の**最小値**
- 以下の条件を満たすならば t の下流を**枝狩りする**
  (つまり t の**下流は探索しない**)

$$b_{\boldsymbol{w}} + d_t^* b_{\boldsymbol{v}} < |\rho_0| - d_t^* |\eta_0|.$$

where $\quad b_{\boldsymbol{w}} = \max \left\{ \sum_{w_i<0} |w_i| x_{it}, \sum_{w_i>0} |w_i| x_{it} \right\}$

Theorem 1：**この枝狩りによりd_tが最小のパターンを発見し損なうことはない**

探索方法のまとめ



d_t が最小となるパターンtを必ず発見できる

探索空間の再利用

- Main searchでは何回もパターン探索問題を解く必要がある
  - 探索空間を保存し再利用
  - 主にDFSコードの最小性のチェックを省く目的
    - DFSコードの最小性チェック＝グラフの同型性チェック　(よって多項式オーダーでは解けない)
- DFSコードが最小か否かをプールするだけでもよいと思う

Initial Search (もパターン探索問題)

問題：以下のg_tを最小にするパターンtを発見せよ

$$g_t = \left| \sum_{i=1} y_i x_{it} \right|$$

枝狩り条件：以下の条件を満たすパターンtの下流

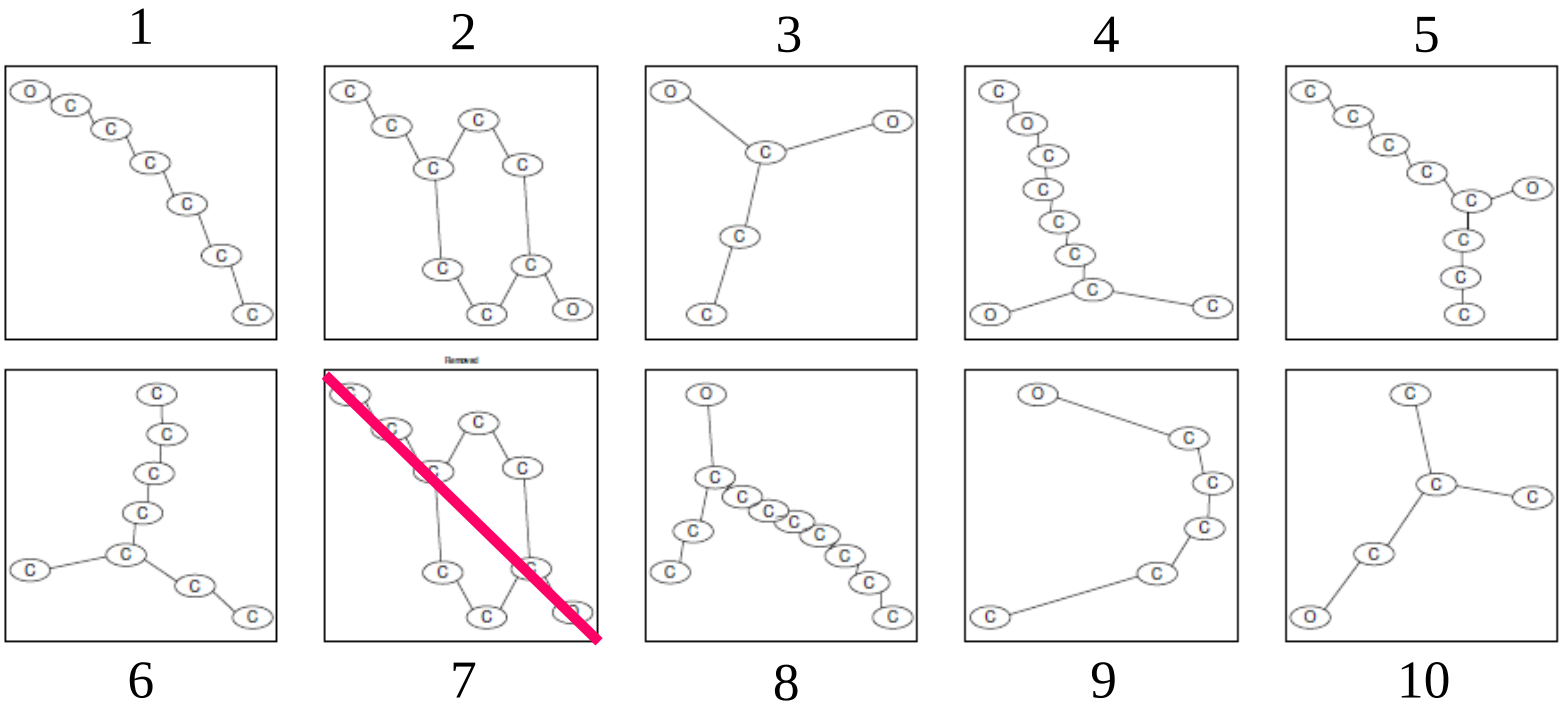
$$g_t^* > \max \left\{ \sum_{y_i<0} |y_i| x_{it}, \sum_{y_i>0} |y_i| x_{it} \right\}$$

この条件を用いて探索はmain searchと全く同様にして行える

実験 (1)

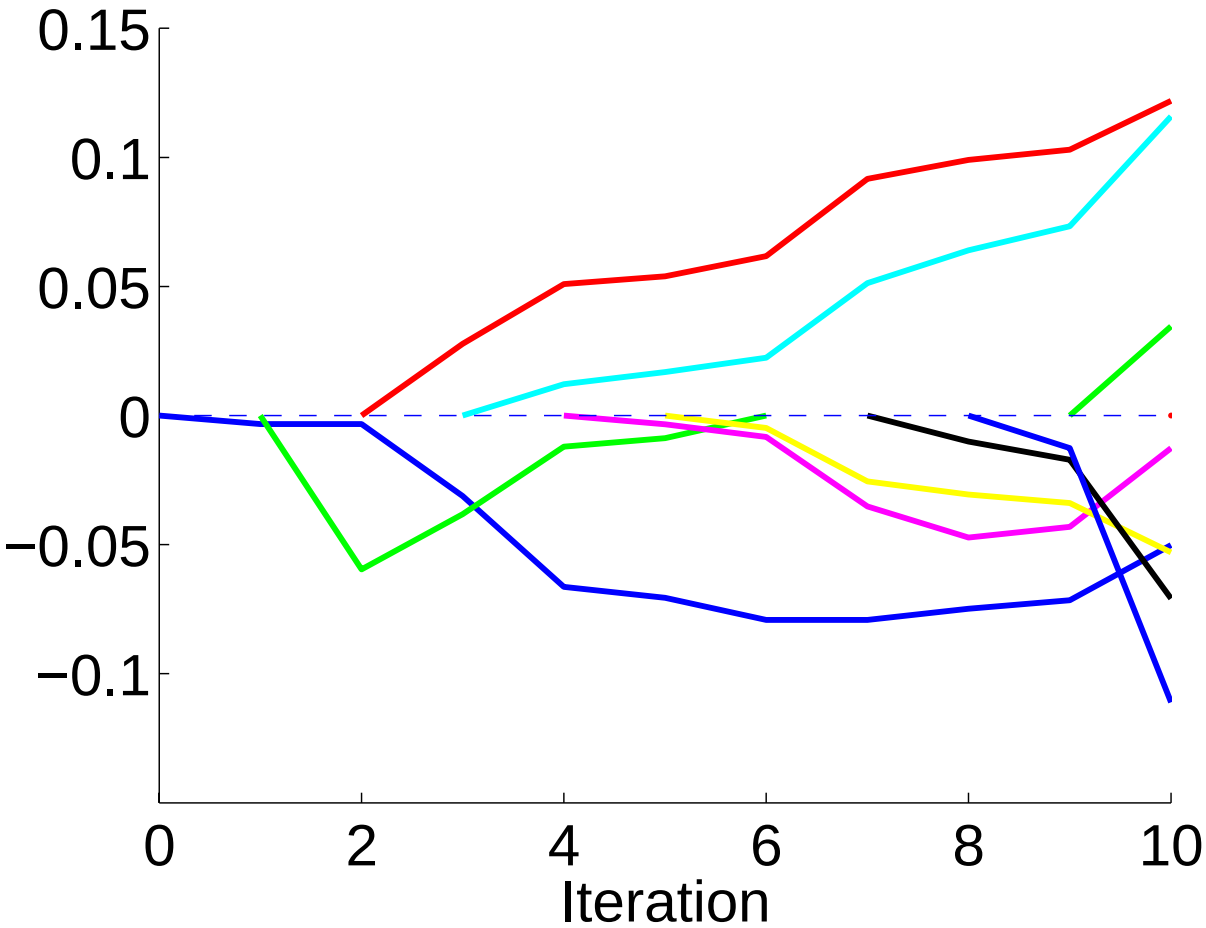
- Endocrine Disruptors Knowledge Base
  - http://edkb.fda.gov
  - 131**の環境ホルモンと考えられる化合物**
    - 131**のラベル付き無向グラフ**
  - **毒性** (toxicity) **を予測**

イベント／Events



7**番目**：exclusion event

Regularization path

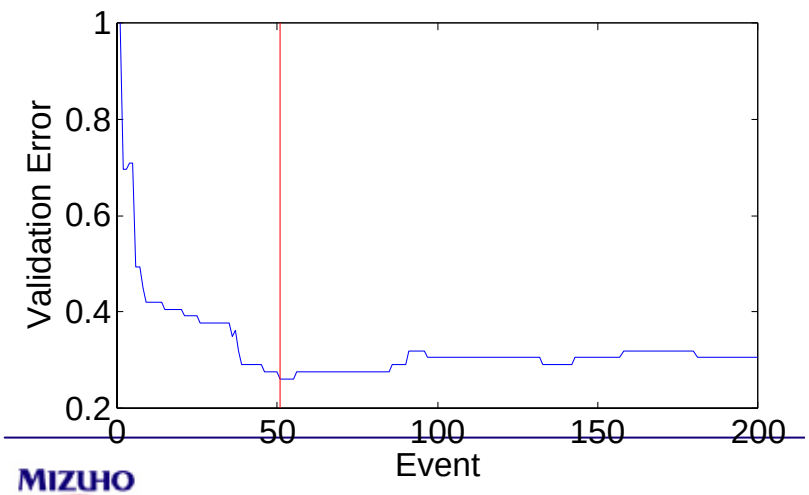


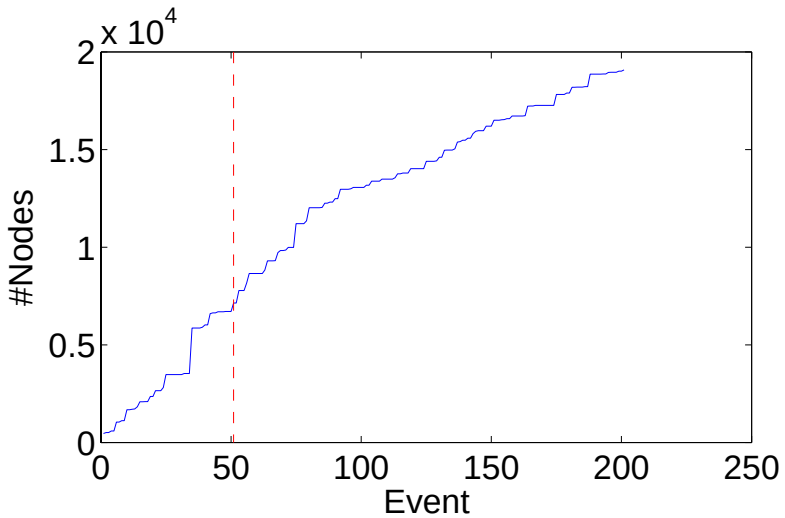
実験(2)

- 提案する枝狩り手法による効率化を評価
- CPDBデータセット [Helma *et al* 2004]
  - 687の化合物＝ラベル付き無向グラフ
  - 分類問題(遺伝子毒性がある or ない)
  - 回帰問題に変換(y=-1 or +1)
- 比較手法(Naïveな手法)
  - 提案する枝狩り条件を使用しないでLAR-LOSSOアルゴリズムを適用する
  - 比較手法と提案手法で得られる解は同一

評価方法

- 90%をトレーニング,10%をテストデータとして使用
- Validation error が最小となるイベントにおける以下の値を記録
  - それまでの計算時間
  - それまでに探索したパターンの数





結果

枝狩りを行わない方法では計算が爆発する

maxpat	Naive #nodes	Naive time	Progressive #nodes	Progressive time
5	2142	0.51	1171	0.38
6	5717	1.39	2111	0.67
7	14309	3.79	3614	1.36
8	33862	9.93	4936	1.90
9	75814	25.64	6605	2.42
10	161858	65.60	7961	2.80
11	332553	164.20	8613	3.00
12	665202	397.95	8857	3.12
13	1302273	931.61	8964	3.11
∞	N/A	N/A	9088	3.15

まとめ

- グラフデータに対するLASSO回帰のregularization pathを**完全**かつ**効率的**にトレースする方法を提案
  - 基本はLAR-LASSOアルゴリズム
  - 計算コストがかかる部分はパターン探索問題に帰着される
  - 効率的な探索を行う**枝狩り条件**を提案
- 計算機実験により枝狩りの有効性を確認
- 木、文字列データなどに容易に**拡張可能**
- 今回とは異なる損失関数への拡張はそれほど**容易ではない**
  - Regularization path が区分的線形になるためのより広い損失関数のクラスに関しては[Rosset *et al* 2007]で詳細に議論されている