ERATO セミナ 2010 - No. 19

ハッシュを用いた高速なグラフカーネルと ZDD

比戸将平
IBM 東京基礎研究所

2010/9/27

概要

大規模グラフにおける機械学習を目的とした、スケーラブルなグラフカーネルを紹介する。その鍵は、ラベルをビット列で表現し、近接ノード集合のラベル分布を論理演算でハッシュ値とする近傍ハッシュである。2 つのグラフのノード集合間で近傍ハッシュ値を比較することで、グラフ間のカーネルを構成する。計算量は高々エッジ数に線形である。実験において、近傍ハッシュカーネルは数千ノードのグラフに適用可能であることが示され、ベンチマークにおいて既存カーネルよりも良い性能を示した。また、ZDD を用いたカーネルの高速化に関してもアイデアを述べる。

ERATO Seminar @ Minato Discrete Structure Manipulation Project
Hokkaido University, 09/27/2010

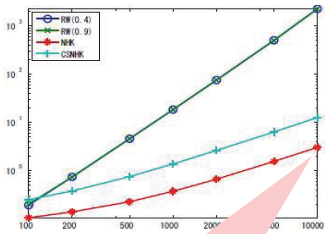
ハッシュを用いた高速なグラフカーネルとZDD

比戸 将平（IBM）
鹿島 久嗣（東大）

概要

目的：1000ノードのグラフに適用可能なカーネル

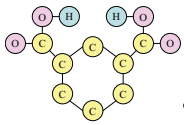
- ポイント
  - –ノードラベルのビット表現
  - –近傍ノード集合への論理演算
  - –*近傍ハッシュによるカーネル*
- 提案カーネルの成果
  - –ベンチマークで良い精度
  - –エッジ数に線形の計算量

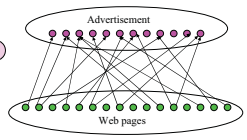


1万ノードのグラフでわずか5秒

背景：グラフデータからの学習

- グラフデータ解析の重要性
  - –何らかの存在同士の関係→グラフ
  - –ベクトルやシーケンス、木構造よりも表現力が高い
  - –ここでは離散的なラベル（付加情報）に限定
- グラフ型データの例
  - –WWWリンク構造
  - –ソーシャルネットワーク
  - –バイオインフォマティクス
  - –化学物質データベース
  - –電子回路





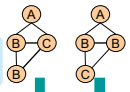
- データからの機械学習における類似度計算の重要性
  - –「データが似ている⇒性質も似ている」という大前提
  - –例：「化学物質AとA'が似ている」&「Aは毒性」⇒「A'も毒性」と推定
  - –グラフ解析の基礎：2つグラフの類似度 or 2つのノードの類似度

グラフカーネルの利点

- サポートベクトルマシン(SVM) 等、カーネル法の隆盛
  - –グラフデータにおける学習にもカーネルが使えるはず
- グラフ専用カーネルを使う理由
  - –有限長ベクトルへの変換が不要

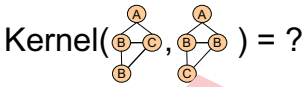
通常のカーネルを使う場合

構造の情報が失われる変換



$\mathrm{Kernel}(\ x_i\ ,\ x_k\ ) = ?$

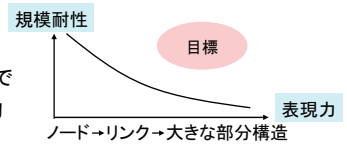
グラフカーネル



カーネル関数の値さえ計算できれば良い

代表的な既存グラフカーネルとその課題

- ランダムウォークカーネル [Kashima et al., 2003]
  - –ランダムウォークにより得られるパス列からカーネル計算
  - –高速化によりノード数の3乗オーダー[Vishwanasan et al., 2007]
- 最短パスカーネル [Borgwardt et al., 2005]
  - –各ノード間の最短パスを求めて計算
  - –ノード数の4乗オーダー
- 表現力と規模耐性のトレードオフ
  - –応用は最大数百ノードのグラフまで
  - –大きなグラフへの適用は非現実的

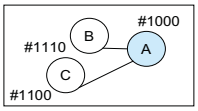


目的：数千ノードグラフ用の高速なカーネルの構築

ビットラベルの導入

- 対象としているグラフ形式：離散ラベル付き無向グラフ
  - –離散ラベルの種類数は一般にそれほど大きくない
- ビットラベル：各離散ラベルをDビット列に変換
  - –ラベルの種類数 << 2^DとなるDを選ぶ
  - –ビット列にする利点
    - 論理演算が適用可能
    - N個のラベルのソートは基数ソート（radix sort）でO(DN)に線形オーダー

4ビットのビットラベル化例



ビットラベル用論理演算：XORとROT

- XOR：排他的論理和
  - –2つのビットが等しければ0
  - –2つのビットが異なれば1
  - –ビットラベルに対してもビット毎に適用
  - –交換律：結果は適用順序に非依存

XORの性質

(a)	$s \oplus s = s_{zero}$
(b)	$s \oplus s_{zero} = s$
(c)	$s \oplus s' = s' \oplus s$
(d)	$(s \oplus s') \oplus s'' = s \oplus (s' \oplus s'')$
(e)	$s' \oplus s' \oplus s = s$

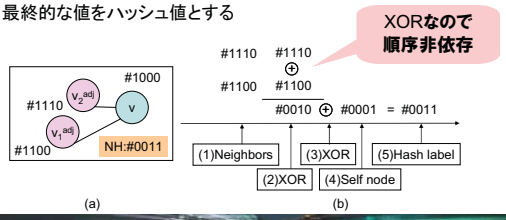
- ROT_k：kビット回転
  - –kビット左に移動(bitshift)
  - –上位にはみ出したkビットを右に回す

(a)(b)XORの計算例
(c)(d)ROTの計算例

(a)	$\#0101 \oplus \#1000 = \#1101$
(b)	$\#1101 \oplus \#1100 = \#0001$
(c)	$ROT_1(\#0001) = \#0010$
(d)	$ROT_2(\#0110) = \#1001$

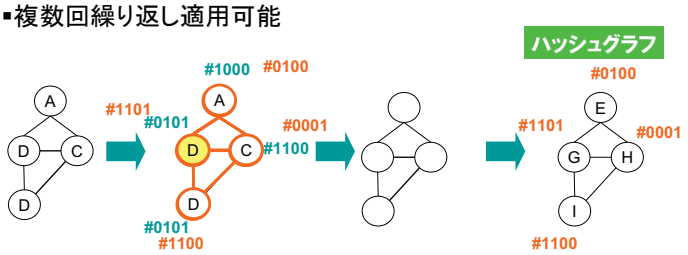
近傍ハッシュの計算方法

- 近傍ハッシュ(Neighborhood Hash = NH)
  - – 例：図(a)のノードvの近傍ハッシュを求める
    1. 近傍ノードのビットラベル集合を取得
    2. それらのXOR値を計算
    3. 自ノードともXOR値を計算
    4. ただし自ノードラベルはROT_1後の値
    5. 最終的な値をハッシュ値とする

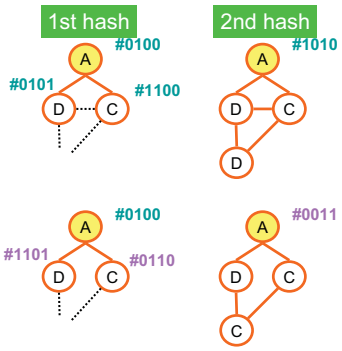


グラフにおける近傍ハッシュの例

- ノード毎に独立して計算可能
- ノードラベルの更新（＝周辺情報の付加）に相当
- ハッシュグラフは同じ構造・異なるノードラベル
- 複数回繰り返し適用可能



ハッシュの繰り返し



- ラベルAを持つ2つのノードが同じハッシュ値となるケース
  - –2回ハッシュを繰り返した後：
    - 近傍ノードの近傍ノード集合も同一の場合
  - –R回ハッシュを繰り返した後：
    - 近傍ノードの近傍ノードの近傍ノードの近傍ノードの…
  - –その割合を類似度と定義
- 長さ2以上のパスを考慮する必要無し
  - –ハッシュ繰り返しによりラベル伝播

[ハッシュラベル一致の計算方法

元のグラフ		
ビット列表現（レベル1）	A #1000 B C #0101 B #1100 #0101	A #1000 B B #0101 C #1100
1回ハッシュ計算後（レベル2）	E #0100 G H #1101 I #1100 #0001	F #1001 G G #1101 J #1100 #1011
2回ハッシュ計算後		
:		

- ハッシュラベル
- ラベル一致比較メソッド (Compare_Labels)
  1. ハッシュをソート
  2. 一致ラベルをカウント(c)
  3. 一致度を計算

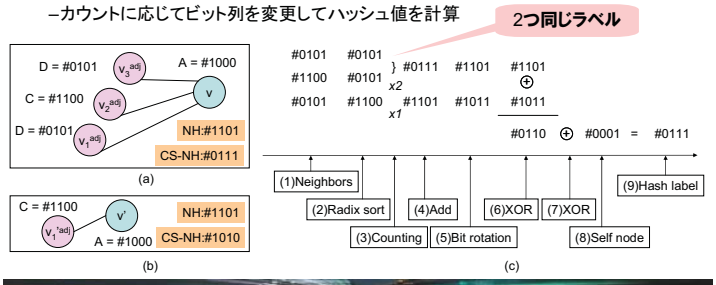
$$\frac{c}{n_a + n_b - c}$$

(n_a, n_b :各グラフのノード数)

同様にマッチング無しに計算可能

より高度な近傍ハッシュの計算方法

- 通常のNHではハッシュ衝突の可能性：同じビットラベルが2つ以上存在する場合
- 解決策：カウント考慮型近傍ハッシュ(Count Sensitive NH = CSNH)
  - –隣接ラベルをソートして同じラベルの数をカウント
  - –カウントに応じてビット列を変更してハッシュ値を計算



カーネル行列計算アルゴリズム

Algorithm 1 *Calculate_Similarity_Matrix*
Require: カーネル行列を計算するグラフ集合
1: $\Gamma = \{G_1^0, \ldots, G_h^0\}$: グラフ集合
2: $h = |\Gamma|$: グラフ数
3: R: 近傍ハッシュの最大適用数
Ensure: K 半正定カーネル行列
4: **for** $r = 1$ to R **do**
5: $K^r \leftarrow I$ {:h x h 単位行列}
6: **for** $i = 1$ to h **do**
7: $G_i^r \leftarrow \mathrm{NH}(G_i^{r-1})$ {:単純かカウント考慮型}
8: $G_i^r \leftarrow \{Radix\_Sort(V_i, \ell^r(\cdot)), E, \ell^r(\cdot)\}$
9: **end for**
10: remove G^{r-1}
11: **for** each pair $(G_i^r, G_j^r) \in \Gamma \times \Gamma$ **do**
12: $K_{ij}^r = K_{ij}^r \leftarrow Compare\_Labels(G_a, G_b)$
13: **end for**
14: **end for**
15: $K \leftarrow \frac{1}{R}\sum_{r=1}^{R} K^r$ {:繰り返しに対する正規化}
16: **return** K {:カーネル行列}

- アルゴリズム概要
  - 部分カーネル行列計算(R回繰り返し)
    1. 各グラフに近傍ハッシュを適用
    2. 各グラフのハッシュラベルをソート
    3. 各グラフペアのハッシュ一致度を部分カーネル行列の要素とする
  - 全部分カーネル行列を足して正規化
  - 最終的なカーネル行列を得る
- バリエーション
  - 近傍ハッシュカーネル(NHK)
    - 各グラフにNHを適用
  - カウント考慮型カーネル(CSNHK)
    - 各グラフにCSNHを適用

近傍ハッシュカーネルの計算量&領域量

- 近傍ハッシュの計算:O(nDd)
  - –XOR:O(Dd)
  - –ROT:O(Dd)
  - –CSNHKのソート:RadixソートでO(Dd)
- 2つのグラフのカーネルの計算:O(RnDd)
  - –近傍ハッシュの計算:各グラフに対してR回
  - –一致度の計算:ソートとカウントでO(nD)*R回

R:ハッシュ回数
n:ノード数
D:ビット長さ
d:次数

計算量:最大ノード数*平均次数(=エッジ数)に線形オーダー

- 近傍ハッシュ後のグラフのメモリ量は元のグラフと同じ

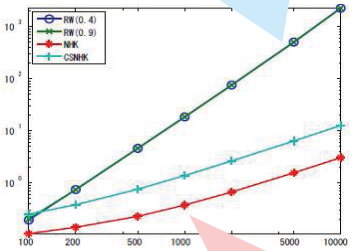
実験

- 実装アルゴリズム(Matlab)
  - –近傍ハッシュカーネル (NHK)
    - ハッシュ繰り返し数パラメータR = { 1,2,3,4,5 }
  - –カウント考慮型近傍ハッシュカーネル(CSNHK)
    - ハッシュ繰り返し数パラメータR = { 1,2,3,4,5 }
  - –ランダムウォークカーネル(RW)
    - 停止確率パラメータp = {0.4, 0.9}
- 3種類の実験
  - –人工グラフによるスケーラビリティ実験
  - –精度評価のためのベンチマーク実験
  - –新しい応用として大きなたんぱく質構造から外れ値検出

人工グラフデータにおける規模耐性実験結果

- 人工データ
  - –平均次数:5に固定
  - –ノード数:10,000まで変化
- 計測対象
  - –2グラフのカーネル計算時間
- ランダムウォークカーネル
  - –二乗オーダーで動作
  - –パラメータは無関係
- 近傍ハッシュカーネル
  - –線形オーダーで動作
  - –CSNHKは定数倍遅い

ランダムウォークカーネル:グラフサイズに二乗

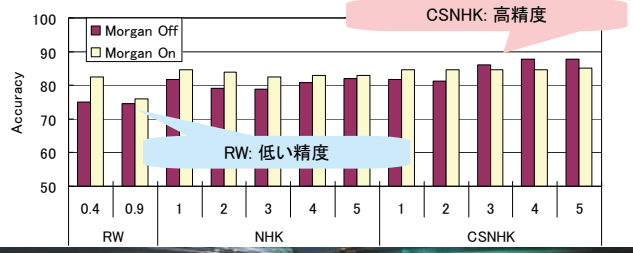


近傍ハッシュカーネル:サイズに線形

MUTAG実験

Data	#Graph	#Node	RW	NHK	CSNHK
MUTAG	188	28	145.2	38.9	41.3

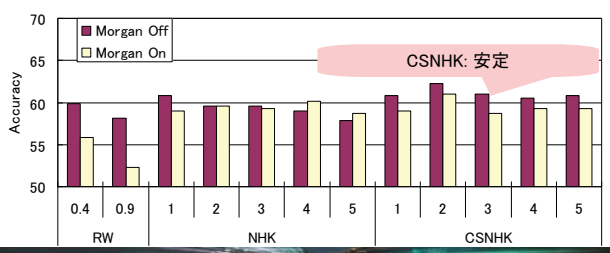
- 化学物質の毒性の有無を判別する問題
- 分類器: C-SVM in LIBSVM
- 評価法: 10-fold cross validationにおける平均精度



PTC実験

Data	#Graph	#Node	RW	NHK	CSNHK
PTC	344	109	896.3	133.7	137.6

- 化学物質の毒性の有無を判別する問題
- 分類器: C-SVM in LIBSVM
- 評価法: 10-fold cross validationにおける平均精度

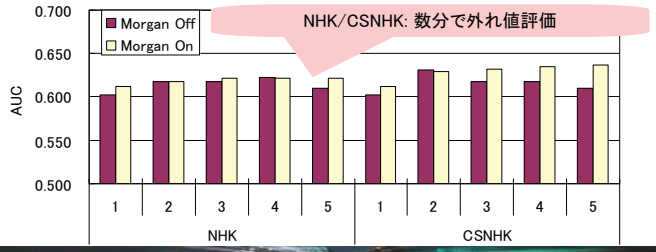


GOLD実験

Data	#Graph	#Node	RW	NHK	CSNHK
GOLD	69	4797	>1day	104.0	155.1

RandomWalk: 計算に1日以上

- たんぱく質構造における外れ値検出
- 検出法: One-class SVM in LIBSVM
- AUC: Common metric in detection problems



ここまでのまとめ

- 初めて数千ノードのグラフに対しても適用可能かつ構造情報を考慮できるグラフカーネルを提案した
- 離散ノードラベルを固定長ビット列で表現することでXORなど論理演算で効率的に隣接ノード情報を集約できる
- 近傍ハッシュを繰り返し適用することで周辺の構造を考慮したカーネル値を計算することができる
- 計算量はノード数＊平均次数に線形で抑えられる
- ハッシュ衝突確率の詳細な解析は今後の課題である

まとめ

- ZDDをはじめとする離散構造処理系はグラフ・ネットワークデータにおける機械学習においても大きな可能性があります