
ICDE 2010 勉強会

Session 11: Top-k Queries

TASM: Top-k Approximate Subtree Matching

目的

- データ木 T から問合せ木 Q に近い部分木を検索
- 編集距離でランキングし top-k を見つけたい

既存手法と問題点

- 動的計画法で T 全体と Q の距離を求めれば全部分木の距離も求まる
- T 全体と計算するのは無駄。特に T 全体がメモリに入らない

提案手法

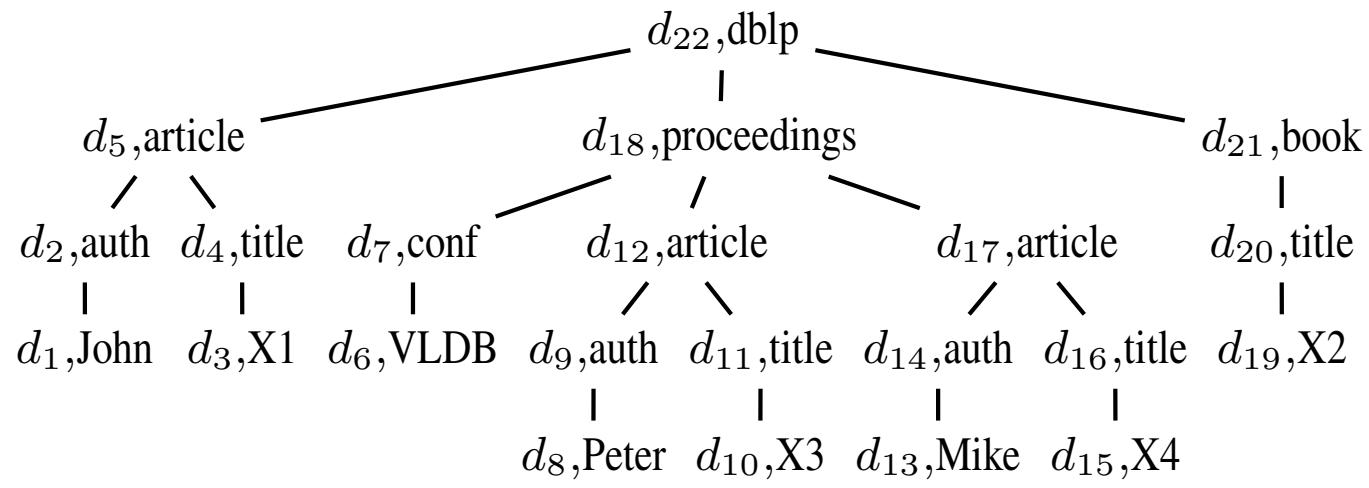
- 既知の上位解から「サイズ τ 以上の部分木は見なくてよい」という τ を求める
- 大きさが「ぎりぎり τ 以下」の部分木を効率よく見つける
- それらに対して通常の手法で編集距離を求める

TASM: Top-k Approximate Subtree Matching

大きさが「ぎりぎり τ 以下」の部分木を効率よく見つける

- 各ノードの (ラベル, 部分木サイズ) の対を **postorder** に並べる
 1. 容量 τ のリングバッファに読み込んでいく
 2. 最初のノードが葉ならバッファ中の祖先で一番大きいものを出力 (まだバッファ外にある, より上位の祖先は必ずサイズが τ 以上)
最初のノードが葉でないなら捨てる
 3. バッファが空いた分を埋める
- 上の step 2 のために, バッファ中の葉ノードとその最上位祖先との関係を表すデータ構造をメンテナンス

TASM: Top-k Approximate Subtree Matching



	d_1	d_2	d_3	d_4	d_5	d_6	
	John,1	auth,2	X1,1	title,2	article,5	VLDB,1	return D_5
	$\uparrow e = 0$	$\uparrow s = 1$					

	d_7	d_8	d_9	d_{10}	d_{11}	d_5	d_6	
	conf,2	Peter,1	auth,2	X3,1	title,2	article,5	VLDB,1	return D_7
						$\uparrow e = 5$	$\uparrow s = 6$	

	d_7	d_8	d_9	d_{10}	d_{11}	d_{12}	d_{13}	
	conf,2	Peter,1	auth,2	X3,1	title,2	article,5	Mike,1	return D_{12}
	$\uparrow e = 0$	$\uparrow s = 1$						

	d_{14}	d_{15}	d_{16}	d_{17}	d_{18}	d_{12}	d_{13}	
	auth,2	X4,1	title,2	article,5	proc.,13	article,5	Mike,1	return D_{17}
						$\uparrow e = 5$	$\uparrow s = 6$	

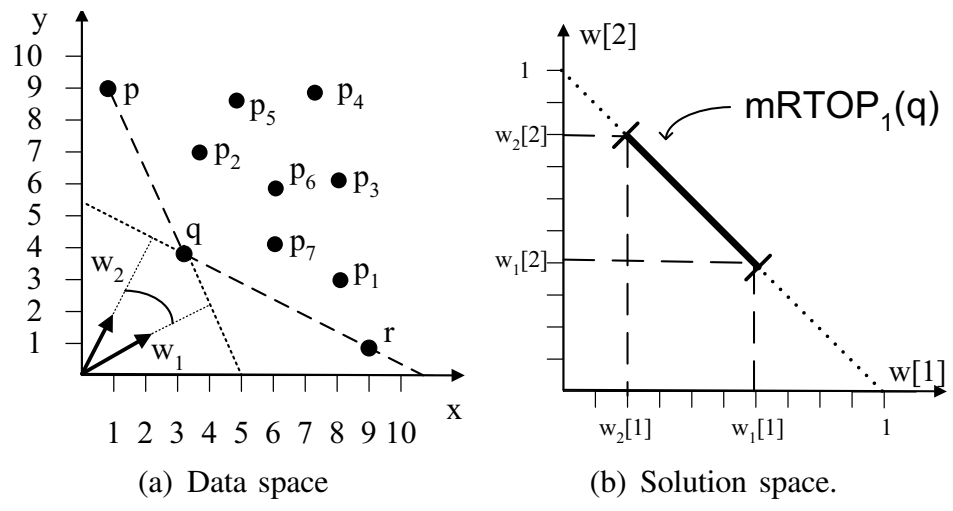
	d_{21}	d_{22}	d_{16}	d_{17}	d_{18}	d_{19}	d_{20}	
	book,3	dblp,22	title,2	article,5	proc.,13	X2,1	title,2	skip d_{18}
			$\uparrow e = 2$		$\uparrow s = 4$			

Reverse Top-k Queries

- 複数の属性 (例: 価格, 性能) の多次元データに対して, 嗜好ベクトル (属性への重み付け) を与えて, top-k を検索するとする.
(★ 嗜好ベクトルは向きにのみ意味がある)
- データ (例: 自社製品) を与えられて, それがどんな嗜好ベクトルに対しては top-k に入るかを検索したい.
 - **monochromatic**: 条件を満たすベクトルの向きの範囲を答える
 - **bichromatic**: 嗜好DBから条件を満たすユーザを抜き出す

Reverse Top-k Queries

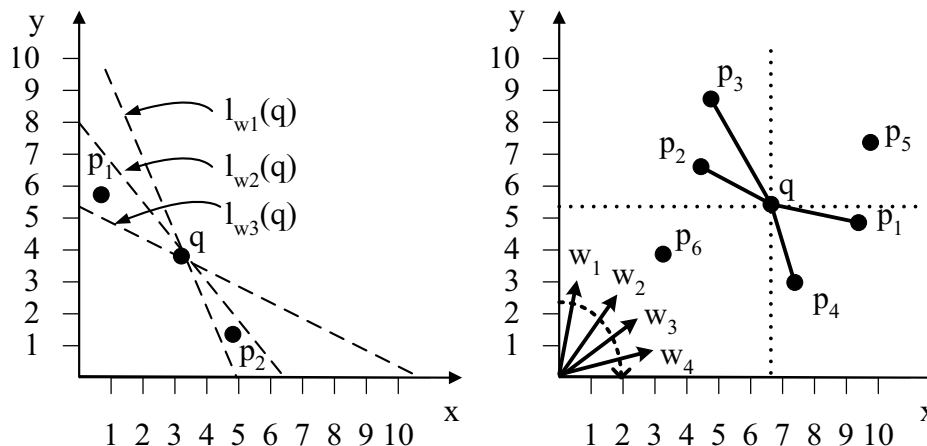
Monochromatic Reverse Top-k Query



- w_1 と w_2 の間の嗜好ベクトルに対しては, q は top-1
- q を質問とする逆 top-1 質問の解は $\{[w_1, w_2]\}$

Reverse Top-k Queries

Monochromatic Reverse Top-k Query



- $k \geq 2$ では解は複数の範囲になりうる。
- q を質問とする逆 top-2 質問の解は $\{[., w_1], [w_3, .]\}$
- q を原点として第一(三)象限の点は常に q より下(上)位
- 第二, 第四象限の点と q を結んだ直線に垂直な嗜好ベクトルについて top-k を求めればよい。

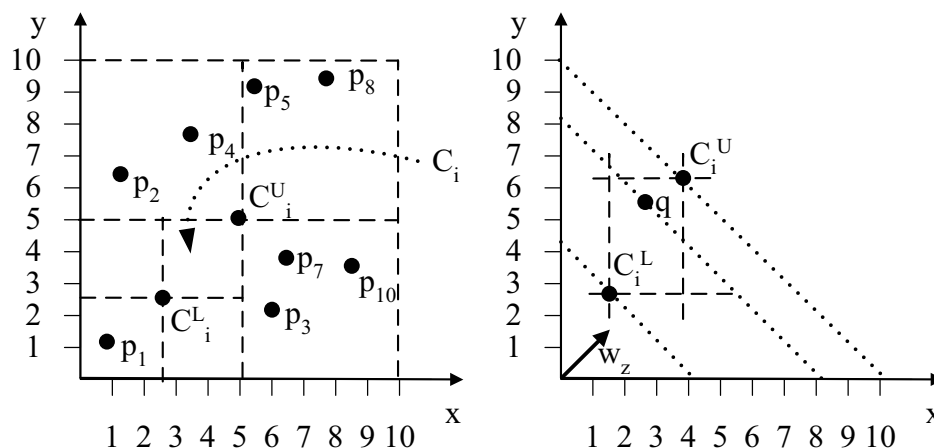
Reverse Top-k Queries

Bichromatic Reverse Top-k Query

- 単純に嗜好ベクトルDB中の全ベクトルについて top-k を求めると大変
- DB中の嗜好ベクトルを近い順にソート (greedy にやる)
- 一つ前の嗜好ベクトルの top-k の解を覚えておき, それらが, 次の嗜好ベクトルに対して, q より上位になるか調べる.
 - 上位なら, q は, top-k に入らない.
 - つまり, その嗜好ベクトルは, q の逆 top-k に入らない.

Reverse Top-k Queries

Bichromatic Reverse Top-k Query のキャッシュによる効率化



- データ空間をグリッドで切る
- 各グリッドの左下と右上の点の逆 top-k を計算しておく
- q が与えられたら, q が入るグリッドを見る
 - 左下の点の逆 top-k の解でないものは, 絶対に q の解にならない
 - 右上の点の逆 top-k の解は, 絶対に q 解になる
 - それ以外の嗜好ベクトルについてのみ調べればよい

Top-K Aggregation Queries over Large Networks

ネットワーク上の h -hop 内の集約演算

ノード v の値: $f(v) \rightarrow [0, 1]$

v の h -近傍: $S_h(v)$

h -近傍内の集約: $F(v) = v$ の h -近傍内の $f(u)$ の和 (or 平均)

$F(v)$ が top-k に入る k 個のノードを求める。

手法 1

- ノード v 毎に $|S_h(v)|$ を記録しておく
- エッジ (v, u) 毎に $|S_h(v) \setminus S_h(u)|$ を記録しておく
- $F(v) \leq F(u) + |S_h(v) \setminus S_h(u)|$
- $F(v) \leq |S_h(v)| - 1 + f(v)$
- これらの上限から, top-k に入り得ない v については計算しない



Top-K Aggregation Queries over Large Networks

手法2

- $f(v)$ の大きい順に, そのノードの $f(v)$ を逆向きに分配
- 各ノードには分配された値 $f(u_1), f(u_2), \dots, f(u_l)$ が貯まる
- ある程度分配したら, 以下の上限を使って top-k に入り得ない v を求める

$$F(v) \leq f(u_1) + \dots + f(u_l) + (N(v) - l) * f(u_l) + f(v)$$

- それ以外の v について通常の手法で集約値を求める

TopCells: Keyword-Based Search of Top-k Aggregated ...

Brand	Model	CPU	OS	Customer Review
Acer	AOA110	1.6GHz	Linux	d ₁ : light weight of only 2.2 lb, fun and powerful features
Acer	AOA110	1.8GHz	XP	d ₂ : weight just over 0.9 kg, powerful Intel Processor
Asus	EEE PC	1.8GHz	XP	d ₃ : ...in pearl white style, images are sharp, disk is large

目的

- 質問「light, powerful」に対して、(Acer, AOA110, *, *) 等のような集約セルで、 n 個以上のデータを含み、かつ、集約文書のスコアが top-k となるセルを求める。

手法

- 基底セル(*がないもの)のスコアからボトムアップに集約していく
- スコアの高いものから集約していく
- 集約セルのスコアは集約されたスコア中の最低値と最大値の間
- 現時点のどのセルの途中結果よりも高いスコアを持つものは top-k