【ICDE 2016 & WWW2016 勉強会】

Research Session 8A-2: Interactive Data Exploration with Smart Drill-Down

Manas Joglekar, Hector Garcia-Molina (Stanford University)
Aditya Parameswaran (UIUC)

担当 : 趙　セイ（名大）

nagoya-u dblab

Some figures are copied from ICDE 2016 proceedings.

概要

- 動機：伝統なDrill-down は非効果な場合がある
  - 結果が多くて，分析者を困惑させる
  - 一回で一列しかdrill-downできない
- 目標：伝統なdrill-down操作の補足として，“面白い“部分にズームインすることを可能にする
- 例 (Table I): 売り上げが一番高い商品を探す
  - 属性 Store でdrill-down
    - 伝統なdrill-down：
      (X,*,*,C,1)形式の全てのタプルを示す
    - Smart drill-down (Table II)：
      k 個の注目すべきかつ面白いタプル
      （ルール）を返す（k：ユーザ指定）

TABLE I: Initial Summary

Store	Product	Region	Count	Weight
⋆	⋆	⋆	6000	0

TABLE II: Result After First Smart Drill Down

Store	Product	Region	Count	Weight
⋆	⋆	⋆	6000	0
▷ Target	bicycles	⋆	200	2
▷ ⋆	comforters	MA-3	600	2
▷ Walmart	⋆	⋆	1000	1

スコア

- 範囲（Coverage）: $MCount(r, R)$
  - ルールrがタプルtをカバー: rの全てのnon-*の値はtの値とマッチ
    （例：ルール (a, b, *)はタプル (a,b,c)をカバー）
  - $MCount(r, R)$: Marginal count
    - 結果リストRではrの先にあるルールにカバーされていない$Count(r)$
      - $Count(r)$：ルールrにカバーされているタプルの総数
- 質 (Goodness): $W(r)$
  - ルールが具体であるほど重みが高い
    - Non-* 値の数により決まる
  - Size weight function, bits weight functionなど

問題定義 | smart drill-down

■ **問題定義**：テーブルT, 単調な重み関数W, 数値kが与えられて，
k個のルールを含めるリストRを見つける
なお，次のスコアを最大化する：

$$\text{Score}(R) = \sum_{r \in R} \underbrace{MCount(r, R)}_{\text{coverage of } r \text{ in } \mathcal{D}} \times \underbrace{W(r)}_{\text{weight of } r} \qquad (1)$$

[Rule drill down]: click on rule r'
- 全ての$r \in R$はr'のsuper-rules
  - r' のSuper-rules: *の数はr'より小さくて，non-*の値は同じ
    例：ルール（a,b,*）はルール (a,*,*)のsuper-ruleとなる

[Star drill down] : click on a * (ルールr'の c 行目)
- 全ての$r \in R$はr'のsuper-rules，かつc行目では * 値はない

Smart Drill Down Algorithms | 貪欲的な近似アルゴリズム

- 基本的なアイデア：k回に繰り返して，
  「最適な限界ルール」を結果リストに入れる

- 効率化：A-prioriアルゴリズムに基づき，パラメーターm_wを用いて上限値を計算し，枝刈りを行う
  - m_w：最適解の最大な重み
    - m_wによって，近似アルゴリズムの精度を保証
    - m_wが小さいほど，実行時間が短い
  - 上限値：$m_w \times MCount(r, R)$
    - 例：ルール$r_1(*, b, *)$のカウントは100
      →r_1のsuper rule r_2 $(*, b, c)$のスコアの上限値は$100m_w$

Smart Drill Down Algorithms|サンプリング手法

テーブルが大きい場合，計算量が多くなる

- 基本的なアイデア
  - サンプル s, 各タプルを選ぶ確率 p
    →ルール$r \in s$カウント：$Count(r) = Count(r)'/p$
    - $Count(r)'$: サンプルsで計算されたカウント
- (Section III-E1) Approximate DP solution
  - 最適に異なるサンプルにメモリを割り当てる
- (Section III-E2) SampleHandler
  - サンプルを生成し，インメモリでサンプルセットを維持
  - パラメーター $minSS$: 最小なサンプルサイズ
    - 大きい$minSS$ほど，カウントの精度が高くて，処理時間も長い