EVA: A Symbolic Approach to Accelerating Exploratory Video Analytics with Materialized Views

Jicong Gao

2022.10.21

背景与目标

- 视频分析存在重复计算。数据库中谓词计算成本较高，因为它们包含有使用深度学习的用户自定义函数（UDF）。
- 本文提出了一种视频数据库管理系统（VDBMS）EVA，自动实现和重复使用耗费较高 UDF 的结果，从而加速数据分析。

特点

- 注重于重复使用 UDF 的结果，而不是重复使用子计划；
- 采用一种符号的方法，分析谓词及查询之间的重复部分；
- 在 UDF 消耗成本评估中引入重复使用参数，在关键查询优化决策（如谓词排序、模型选择）中使用更新后的成本函数。

总览图

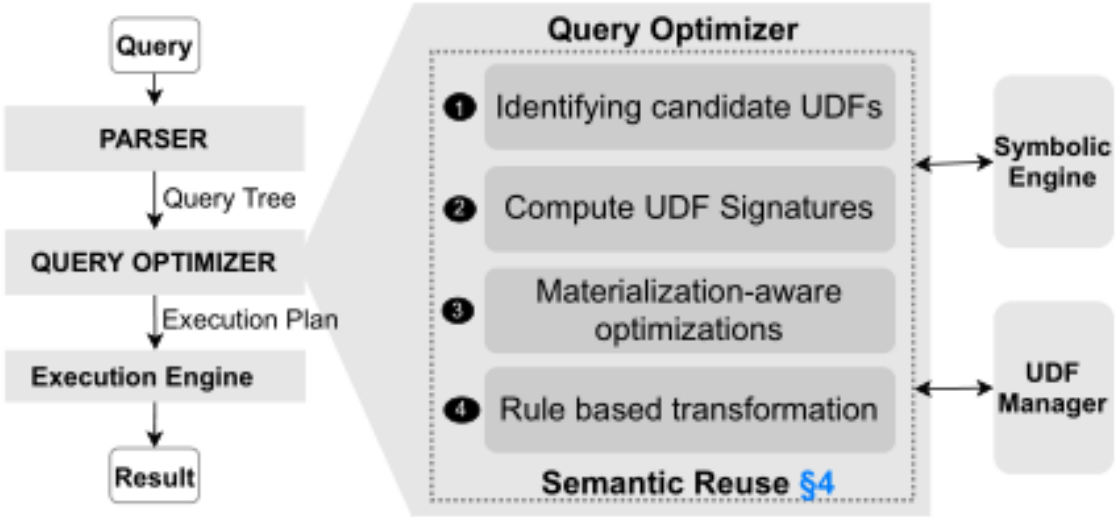

Figure 1: Overview of EVA– It takes in a client query, parses it and generates an optimized plan which is executed to generate results. We modify the OPTIMIZER to incorporate the reuse algorithm that accelerates query processing by leveraging the results of earlier UDF invocations.

图 1: 系统总览图

语义重用算法

如第 7 页图所示，语义重用算法包括以下四个步骤：

1. 识别候选 UDF：识别有价值的 UDF 结果并将其物化（存储），使用评估成本来过滤掉花销较小的 UDF。
2. 计算 UDF 签名：签名区分了不同的 UDF，它由两部分组成：UDF 名称及评估 UDF 所需要的的资源；具有相同签名的 UDF 将重用结果，由 UDF 管理器（Manager）维护。
3. 支持物化（Materialization-aware）优化：给定候选 UDF 及其历史调用（UDF 管理器提供），借助符号引擎（SymbolicEngine），在 (1) 求解多个 UDF 的顺序 (2) 选择合适的深度学习模型（也叫作物理 UDF）两个方向上进行优化。
4. 对候选 UDF 进行合法转换：利用物化结果，进行恰当转换进行加速。

识别重用机会

为了识别具有相同签名 UDF 调用 X 和 Y 之间的重叠计算，EVA 计算三个派生谓词：

- 谓词交集。后一个 UDF 调用可以重用前一个的结果。
- 谓词差集。后一个 UDF 调用必须进行计算，加入评估。
- 谓词并集。在计算结束后，并集的计算结果可供重用。

重用会影响 UDF 评估

UDF 调用成本更新为由谓词的交集、差集共同决定，从而考虑到了物化结果重用的消耗优化。优化器在谓词重新排序中使用该更新后的成本函数，生成支持物化的排序函数，从而确定恰当的顺序。同样在模型选择任务中，优化器将其转化为加权集覆盖问题。

重写查询

两个转换规则进行优化：

- 基于 UDF 的谓词转换规则：拆分包含多个 UDF 调用的操作符。
- 支持物化的转化规则：用物化结果替代消耗较多的 UDF 计算。

两者都支持 UDF 结果的有效重用，无论查询位置如何。

符号谓词分析

Algorithm 1: Symbolic Predicate Analysis

```
Input  : predicate: input predicate
Output : simplifiedPredicate: simplified predicate
1  Procedure ReducePredicate(predicate)
2      ❶ DNFPredicate ← DNF (predicate)
3      foreach conjunctive in DNFPredicate do
4          ❷ ReduceConjunctive (conjunctive)
5      repeat
6          c1, c2 ← PopTwoConjunctives (DNFPredicate)
7          ❸ c1, c2 ← ReduceUnionConjunctives (c1, c2)
8          PushConjunctives (DNFPredicate, c1, c2)
9      until TimeOut or NoChange
10     return DNFPredicate
11 Procedure ReduceUnionConjunctives (c1, c2)
12     foreach dimension in c1 ∨ c2 do
13         fc1 ← FilterDimension (c1, dimension)
14         fc2 ← FilterDimension (c2, dimension)
15         if fc1 ⊃ fc2 or fc1 ⊂ fc2 then
16             return ReduceUnionSingleDimension (c1, c2, dimension)
17     return c1, c2
```

图 2: 符号谓词分析

支持物化的优化

用以前查询的物化结果可以替代调用 UDF 的较高消耗，根据视图可用性来调整 UDF 调用成本。

- 重定义 UDF 调用成本。
- 求解基于 UDF 的谓词的最佳顺序。

逻辑 UDF 重用

Algorithm 2: Logical UDF Reuse – The algorithm to rewrite the logical UDF with the corresponding physical UDFs that minimizes the execution cost.

Input : sig: UDF signature, $\mathcal{C}$: set of UDF constraints, q: assocaited predicate for UDF
Output: $\mathcal{D}$: optimal set of equivalent physical UDFs

Procedure `OptimalPhysicalUDFs(sig, C, q)`
  $X \leftarrow$ `PhysicalUDFs(sig, C)` ▷ Phy UDFs that satisfy constraints
  $y \leftarrow arg\,min_{x \in X}\ c_x$ ▷ Min cost UDF
  repeat
    for $x \in X$ **do**
      Compute $W(x,q) = \frac{C(m_x)}{s_{p \cap_x}\,|m_x|}$ ▷ Cost per uncovered tuple
    $x^* \leftarrow argmin_{x \in X} W(x,q)$ ▷ UDF with min W(x, q)
    if $W(x^*,q) < c_y$ **then**
      $\mathcal{D} \leftarrow \mathcal{D} \cup \{(x^*, p_{\cap_{x^*}})\}$ ▷ Select mat view of x^*
      $q \leftarrow \mathrm{DIFF}(\tilde{p}_{x^*}, q)$
    else
      $\mathcal{D} \leftarrow \mathcal{D} \cup \{(y,q)\}$ ▷ Select cheapest UDF
      $q \leftarrow \emptyset$
  until $q \neq \emptyset$
  return $\mathcal{D}$

图 3: 逻辑 UDF 重用算法

基于规则的查询修改

两个转换规则进行优化：

- 基于 UDF 的谓词转换规则：拆分包含多个 UDF 调用的操作符。
- 支持物化的转化规则：用物化结果替代消耗较多的 UDF 计算。

待改进方向

- 联接谓词的符号分析。
- 链式函数调用与模糊匹配。

Thanks!