



混合云环境中改进的工作负载分配方案

刘 颖¹, 甘 泉², 王 宇³

(1. 宜春学院数学与计算机科学学院 宜春 336000;

2. 平顶山学院计算机科学与技术学院 平顶山 467002; 3. 武汉大学计算机学院 武汉 336000)

摘 要:研究了混合云环境下计算任务在公有云和私有云中的高效安全分布机制;然后,针对使用公共云引入的不同层级约束,将最优风险感知工作负载分布问题描述为工作负载响应时间最小化问题,提出一种可以实现检索工作量最优分布的动态规划算法,可在已知检索工作负载和公共云使用边界(资源分配成本和敏感性数据泄露)的条件下,搜索出最优计算和数据划分方案;最后,在各种参数配置下,基于标准的 TPC-H 基本数据集,对提出的算法进行了全面评估。实验结果表明,新方案通过利用混合云组件,且不违反先前确定的任何公共云使用约束的情况下,显著提升了系统性能。

关键词:混合云;工作负载;响应时间;检索;动态规划

doi: 10.11959/j.issn.1000-0801.2015024

Improved Workload Distribution Scheme in Hybrid Cloud Setting

Liu Ying¹, Gan Quan², Wang Yu³

(1. School of Mathematics and Computer Science, Yichun University, Yichun 336000, China;

2. College of Computer Science and Technology, Pingdingshan University, Pingdingshan 467002, China;

3. Computer School of Wuhan University, Wuhan 336000, China)

Abstract: An efficient and secure mechanism to partition computations across public and private machines in a hybrid cloud setting was explored. Aiming at the restrictions in different levels for public cloud usage, the optimal risk-aware workload distribution problem was formalized as a mechanism for workload response time minimization. A dynamic programming approach that results in an optimal distribution of the query workload was proposed, which that searches for an optimal computation and data partitioning scheme given a query workload and public cloud usage boundaries (resource allocation cost and sensitive data disclosure). Finally, the performance of the dynamic programming approach was evaluated. Experimental results show that the proposed scheme can lead to a major performance gain by exploiting both the hybrid cloud components without violating any pre-determined public cloud usage constraints.

Key words: hybrid cloud, workload, response time, query, dynamic programming



1 引言

云计算^[1,2]的兴起使终端用户可以使用在线支付模式访问强大的计算基础设施、平台和服务,引发了计算机领域的一场变革。混合云是云计算领域出现的一种新趋势。在混合云模式下,终端用户的内部功能/资源与云服务无缝集成,在数据泄露控制和云资源最小化方面具有优势。但是,如果要对这些优势加以利用,面临的首要问题是如何将数据和计算任务在公用基础设施和私用基础设施间进行分配。从敏感数据泄露、计算性能和资源分配成本方面看,不同的选择有不同的效果。一种极端情况是将整个数据和工作负载外包给公共云。虽然实施起来非常简单,但是这种解决方案的云服务资源分配成本最高(存储和计算),且数据泄露的风险很大。此外,外包策略的性能不是最优的,因为如果当前没有处于使用状态,则会浪费大量资源。一种替代性策略就是在公有云和私有云上复制数据,并在两者间分配工作负载。在私有云上计算简单的检索,复杂的检索可以在公有基础设施上进行。上述策略利用了本地资源,因此可以降低所需云服务的成本。然而,此时的资源分配成本和暴露于公共云下的敏感性数据量达到最大。

为此,多篇文献对混合云条件下的安全数据处理进行了研究。参考文献[3]的Sedic从单个MapReduce作业角度对问题展开了研究;参考文献[4]的关系云使用基于图的分割策略将数据和计算任务分配到公共云端和私有云端。基

于多个加密层对数据分区进行加密并存储于服务器上,因此在检索时可能需要多轮解密及公共云和私有云之间的 τ 通信。

参考文献[5]和参考文献[6]在数据库外包背景下,对检索计算任务在可靠和不可靠服务器间的分配问题展开了研究。这些方法的重点是对各个检索进行分配,以确定可以直接运行于不可靠服务器上加密表示的最大分区。参考文献[7]对数据和检索在多个服务器上的分布进行了研究。在其模型中,私有服务器和公有服务器均假设为不可靠服务器,不允许保密政策中指定的所有属性在任何时间暴露于任何服务器下。参考文献[8]针对内存攻击条件下的单机架构研究了基于风险的方法。但是没有在云环境下进行研究。鉴于此,本文正式定义混合云计算(及相关数据)分配问题,并提出一种数据处理任务分配方法,实现各种性能、数据泄露风险和经济成本目标。在各种参数配置下,基于标准的TPC-H基本数据集,全面评估了本文方法的有效性。

2 系统架构

图1给出了混合云系统架构,系统主要由两部分组成:统计采集层,负责数据集和检索作业的统计采集任务;数据和检索管理层,在已知检索集合条件下确定数据和工作负载的划分方案。

用户开始时提交一组关系 $R=\{R_1, R_2, \dots, R_n\}$ 、一个检索工作量(工作负载) $Q=\{q_1, q_2, \dots, q_n\}$ 以及一组资源分配和敏感性数据泄露约束 C 。其中, R 表示将划分于混合云上的

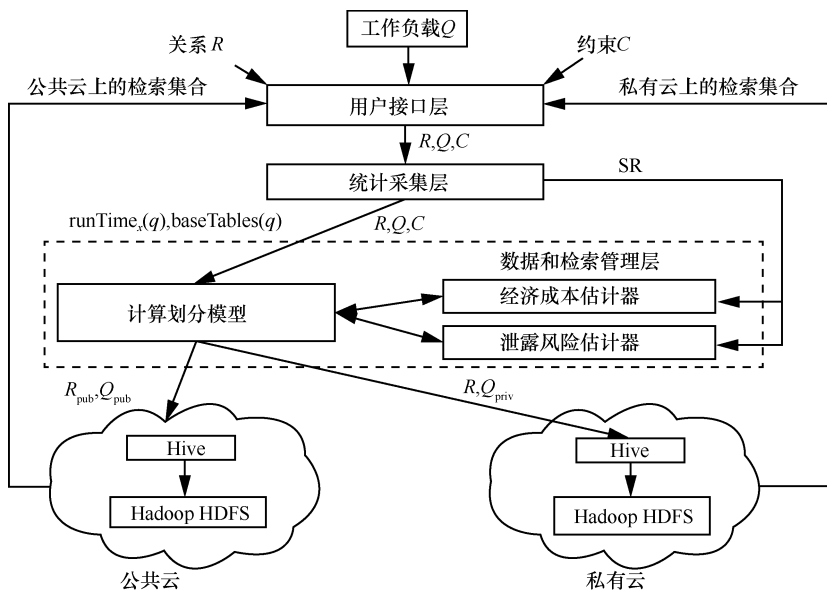


图1 混合云系统架构

数据集, Q 是混合云环境下, 定义于数据集 R 上的检索工作量。系统开始时通过统计采集模型, 对 R 和 Q 执行统计采集任务。该模块估计出回答 Q 中每个检索所需要的数据项最小集合及基本关系 I/O 大小(或者说是运行时间)。此外, 创建统计量作为等宽柱状图, 并发送给估计器模块。计算划分模块接收到 R 、 Q 、 C 、 I/O 尺寸估计值以及回答 Q 中每个检索需要的数据项最小集合, 然后系统求解计算划分问题(CPP)。在求解 CPP 时, 使用经济成本估计器来估计处理公共云检索及存储中公共端数据划分时的经济成本, 而泄露风险估计器用于计算候选解决方案的敏感度。在求解 CPP 时, 该层会生成两个输出: $R_{pub} = \{R_1^m, R_2^m, \dots, R_m^m\}$, 其中 $R_i^m \subseteq R_i$, 表示 R 的公共云部分; $Q_{pub} = \{q_1^m, q_2^m, \dots, q_n^m\}$, 表示将在公共云上执行的检索集合。

私有云存储整个数据集 R , 而公共云只存储公共端数据划分 R_{pub} 。利用合适的表示技术, 将 R_{pub} 和 R 中的非敏感性数据和敏感性数据分别存储于公共云和私有云上。本文使用基于 Hadoop HDFS^[9] 的基本架构来部署存储模式。

当系统根据 CPP 问题的解存储了相关数据后, 系统可以向检索处理提供支持。具体方法是把公共端和私有端检索分配给相应的 Hive 检索引擎。使用基于 Hive 的基础架构来处理检索, 其中 Hive 是基于 Hadoop 的数据仓库, 支持用户定义文件存储于底层 HDFS 时的文件结构。

3 问题描述

本节定义了混合云环境下的计算划分问题。问题的目的是实现已知检索工作量运行时间最小化, 且受到两个独立约束的限制。第一个约束是通过限制公共云资源使用导致的经济成本, 限制运行于公共云之上的总处理量; 第二个约束是关于泄露风险, 即用户愿意存储于公共端上的敏感数据量。这一问题的解就是在公共端和私有端间划分数据和计算任务。

3.1 标记法

在定义 CPP 问题前, 先给出将在本文中使用的标记法, 以便阐述 CPP 问题及 CPP 问题的解。

$sens(R')$: 存储于 R' 中的敏感单元的估计数量, 且 $R' \subseteq R$ 。

$baseTables(q)$: 为了回答检索 $q \in Q$ 而需要的数据项最小估计集合。 $baseTables(q_i) \subseteq R, 1 \leq i \leq n$ 。

$freq(q)$: 检索 q 的访问频率。

$runT_x(q)$: 位于 x 处(公共端或私有端)检索 $q \in Q$ 的估

计运行时间。

$ORunT(Q', Q'')$: 已知 Q'' 中检索运行于公共云上时, 在混合云环境下, Q'' 中检索的总体运行时间。请注意, $Q'' \subseteq Q'$, 否则为未定义。即:

$$ORunT(Q', Q'') = \max \left\{ \begin{array}{l} \sum_{q \in Q''} freq(q) \times runT_{pub}(q) \\ \sum_{q \in Q' - Q''} freq(q) \times runT_{priv}(q) \end{array} \right. \quad (1)$$

通过对类似于 SQL 的检索运行语法分析程序, 为每个检索构建 $baseTables$ 。本文语法分析程序首先解析已知的检索字符串, 并确定该检索内的表(关系)。然后, 对每个相关表 T , 搜索检索字符串, 以确定其相关属性和谓词。

3.2 CPP 问题的定义

已知检索工作量及其相关基本表, 可以将 CPP 问题建模为一个优化问题, 且该优化问题的目标是在混合云上分割检索工作量, 使工作量 Q 的总体运行时间(或性能)最小化。CPP 问题被建模为一个优化问题, 目的是寻找检索工作量的一个子集 $Q_{pub} \subseteq Q$ 及 R 的子集 $R_{pub} \subseteq R$ 。

$\min ORunT(Q, Q_{pub})$

$$s.t. \begin{cases} store(R_{pub}) + \sum_{q \in Q_{pub}} freq(q) \times proc(q) \leq PRA_CONST \\ sens(R_{pub}) \leq DISC_CONST \\ \forall q \in Q_{pub}, baseTables(q) \subseteq R_{pub} \end{cases} \quad (2)$$

其中, PRA_CONST 表示最大可容许公共资源分配成本, $DISC_CONST$ 表示可以泄露于公共云上的敏感性数据最大量。通过综合使用检索工作量划分模型和约束(公共端经济成本和敏感性数据泄露), 可以在同一框架下解决多种现实场景。这些场景包括: 鉴于法律法规等因素, 用户非常反对将敏感性数据存储于公共云上; 用户希望提升性能, 愿意担负一定风险将敏感性数据存储于公共端。

3.3 成本指标

上述 CPP 问题的定义涉及 3 个指标: 检索运行时间(即性能)、经济成本和泄露风险。下面将详细讨论这 3 个指标。

(1) 检索运行时间

准确计算检索 q 在公共端或私有端上的运行时间取决于多种因素。除了每台机器的处理器速度和可用内存等底层架构技术规格外, 选用的检索运行方案也是影响检索响应时间的重要因素。在估计阶段, 本文使用检索方案的



I/O 尺寸替代运行时间。混合云环境下检索策略 τ 的运行时间可估计为:

$$runTime_x(\tau) = \frac{\sum_{\forall \rho \in \tau} inpSize(\rho) + outSize(\rho)}{\omega_x} \quad (3)$$

其中, $inpSize(\rho)$ 和 $outSize(\rho)$ 表示运算符 ρ 的估计输入和输出尺寸。权重 ω_x 表示在位置 x (公共或私有) 处, 单位时间内可执行多少个 I/O 操作。它也描述了公共集群和私有集群间的计算资源差异。第 5 节给出了 ω_x 的计算方法。利用统计量 SR 来计算策略 τ 不同环节的 I/O 尺寸。然而, Hive 没有为关系保存属性级别统计量。因此, 本文部署了统计采集模块, 对数据集中的每个关系展开分析, 针对关系的非敏感性和敏感性部分, 利用单独的直方图保存关系及其属性统计量。然后, 结合柱状图及相关属性估计检索的 I/O 尺寸。

(2) 经济成本

一般情况下, 所有云供应商支持竞争性出价模型, 并提供不同服务水平的数据存储和处理服务协议 (SLA)。通过对承包给公共端的数据和处理任务进行约束, 可以控制经济成本。由于公共云服务往往受到运营花费的约束, 因此使用这一指标作为本文问题的约束。将公共端计算导致的总体经济成本定义为如下两项之和。

- $store(R_{pub})$: 公共云部分的存储经济成本 R_{pub} 。
- $proc(q)$: 公共端检索 q 的处理经济成本。此外, CPP 问题可保证存储公共端部分和执行公共端检索的总体经济成本 (即 Q_{pub} 和 R_{pub}) 受到最大公共资源分配约束 PRA_CONST 的限制。

(3) 敏感数据泄露风险

敏感性数据泄露对处理敏感性数据的机构来说具有重要意义, 如果丢失敏感性数据, 则该云服务机构将面临合规罚款及诉讼费用。本文将总体泄露风险估计为公共云条件下数据分区内的敏感单元的数量, 即 $sens(R_{pub})$ 或 $sens\left(\bigcup_{q \in Q_{pub}} baseTables(q)\right)$ 。此外, 泄露风险还取决于用于存储敏感性数据的数据表示法, 在本文中泄露风险是固定的。同时, 泄露成本的边界也设置为用户定义的数值 $DISC_CONST$ 。本文将焦点放在数据表示法确定条件下计算任务的最优划分。

4 CPP 问题的解

如上文所述, CPP 问题的目的是确定可以转换到公共

云上的已知数据集和检索工作量的子集。实现这一目的的具体方法就是在多种约束及混合云条件下使检索工作量的总处理时间最小化。由于可以认为 R_{pub} 与 $\bigcup_{q \in Q_{pub}} baseTables(q)$ 等价, 因此 CPP 经过简化只需确定 Q_{pub} 即可。换句话说, 其余使总体性能最小化的任意解 R' 应该是 $\bigcup_{q \in Q_{pub}} baseTables(q)$ 的超集, 而解 $\bigcup_{q \in Q_{pub}} baseTables(q)$ 的敏感数据泄露和经济成本最低。于是, CPP 问题转化为寻找检索工作量的子集, 在满足已知约束的同时, 使工作量执行时间最小化。

为了表示 CPP 问题及其输入和约束, 本文使用如下标记法: $CPP(Q, PRA_CONST, SENS_CONST)$ 。此外, 假设检索工作量 Q 和约束 PRA_CONST 均事前已知。CPP 求解算法如下。

输入: $Q, PRA_CONST, SENS_CONST$

输出: Q_{pub}

初始化 $pub\ Q$

for $i=1 \rightarrow Q.size\ do$

$procCost \leftarrow proc(q_i)$

$totCost \leftarrow procCost + store(baseTables(q_i))$

$disc \leftarrow sens(baseTables(q_i))$

for $j=0 \rightarrow PRA_CONST\ do$

for $k=0 \rightarrow DISC_CONST\ do$

if $i=1\ then$

if $checkConstr(\{q_i\}, j, k)$

AND $ORunT(Q^1, Q^1) < ORunT(Q^1, \emptyset)$

$pub\ Q[i][j][k] \leftarrow \{q_i\}$

else

$pub\ Q[i][j][k] \leftarrow \emptyset$

end if

else

$pubCaseOT \leftarrow \infty$

$(j', k') \leftarrow (NaN, NaN)$

if $chenkConstr(\{q_i\}, j, k)\ then$

for all $j - totCost \leq iC \leq j - procCost\ do$

for all $k - disc \leq iD \leq k\ do$

$tmpSet \leftarrow pub\ Q[i][j][iD] \cup q_i$

if $checkConstr(tmpSet, iC, iD)\ and$

$ORunT(Q^i, tmpSet) < pubCaseOT\ then$

$pubCaseOT \leftarrow ORunT(Q^i, tmpSet)$

```

(j',k') ← (iC, iD)
    end if
  end for
end for
end if
privCaseOT ← ORunT(Qi, pubQ[i-1][j][k])
if privCaseOT ≤ pubCaseOT then
  pubQ[i][j][k] ← pubQ[i-1][j][k]
else
  pubQ[i][j][k] ← pubQ[i-1][j'] [k'] ∪ {qi}
end if
end if
end for
end for
end for
返回 pubQ[Q.size-1][PRA_CONST][DISC_CONST]

```

4.1 CPP 问题的动态规划方法

已知检索工作量量子集指数数值, 本文使用动态规划方法(CPP 求解算法)来寻找最优解, 该算法可以输出一个检索集合 Q_{pub} 作为 $CPP(Q, RA_CONST, DISC_CONST)$ 的解。CPP 求解算法使用一种数据结构 $pubQ$, 并频繁调用函数 $checkConstr$ 。 $pubQ[i][j][k] \subseteq Q^i$, 是保存 $CPP(Q^i, j, k)$ 问题的检索解集合, 其中 $Q^i = \{q_1, q_2, \dots, q_i\}$ 。已知最大可接受经济成本和最大泄露风险分别等于 j 和 k , 该数据结构存储前 i 个检索中被选择在公共云上处理的检索, 以实现前 i 个检索的总体响应时间最小化。当 Q^i 中的检索运行于公共端上时, $checkConstr(Q^i, j', k')$ 返回经济成本边界 j' 和泄露风险边界 k' 是否被突破。同时, 该方法检查 $store\left(\bigcup_{q \in Q} baseTables(q)\right) + \sum_{q \in Q'} (freq(q) \times proc(q)) \leq j'$ 及 $sens\left(\bigcup_{q \in Q'} baseTables(q)\right) \leq k'$ 是否成立。

4.2 算法分析

在对算法进行分析前, 首先需要确定把 q_3 分配给公共端是否会违反相关约束。如果把 q_3 转移到公共端, 则总体经济成本和总体泄露成本分别为 25 美元和 20 美元(假设 $1 \leq i \leq 3$, 则 $freq(i)=1$)。如果 $j < 25$ 或 $k < 20$, 则将 q_3 作为公共端检索的任意解是不可行的, 于是 $CPP(Q^3, j, k) = CPP(Q^2, j, k)$ 。请注意, 因为任意检索在私有端上运行不会违反任一约束, 所以这一情况不需要进行可行性分析。

$CPP(Q^i, PRA_CONST, SENS_CONST)$ 可以统一表示为 $CPP(Q^i, j, k)$ 。这一广义问题的解是 Q^i 的子集, 所以 Q^i 中最后一个检索 q_i 可能有两种结果, 即检索 q_i 要么是 $CPP(Q^i, j, k)$ 的解, 要么不是。因此, 以下两种情况需要仔细分析。

情况 1 如果 q_3 运行于公共端, 则不止 1 个 CPP 子问题需要求解。

这是因为如果 q_3 运行于公共端, 将会使总体经济成本增加 15~25 美元。泄露风险数值为 0~20 个敏感单元。部分或整个 $baseTables(q_3)$ 可能已经包括于某个 $CPP(Q^2, j', k')$ 问题的解 Q_3 中。此时, 除了 $\bigcup_{q \in Q_3} baseTables(q)$ 外再存储 $baseTables(q_3)$ 不会导致上文中 CPP 求解算法所给出的经济成本和泄露风险。所以, 需要求解问题 $CPP(Q^2, j', k')$, 其中 $j-25 \leq j' \leq j-15, k-20 \leq k' \leq k$, 以获得 $CPP(Q^3, j, k)$ 的最优解。然而, 必须对基于 q_3 与 $CPP(Q^2, j', k')$ 问题的解的并集形成的所有候选检索集合进行测试, 以保证其不会违反相关约束, 并保证它的解在性能方面是情况 1 下所有解中的最优解。如果它没有提供最优解, 则它与情况 2 解共同构成候选解。

情况 2 如果 q_3 运行于私有云上, 则 $CPP(Q^3, i, j) = CPP(Q^2, i, j)$ 。

在计算完两种情况下的最优候选解后, 本文算法将比较两个解的总体预期运行时间, 并且选择最低值作为 $CPP(Q^3, j, k)$ 问题的解。上述算法需要确定已知检索工作量 Q 和随机数据划分条件下的各种成本(即泄露成本、经济成本和检索运行成本)。本文模型中的泄露风险只取决于公共端部分 R_{pub} , 可由已知的检索工作量间接确定。然而, 检索运行时间和经济成本取决于检索工作量, 二者均可估计为各个检索的成本之和。

5 实验结果

首先给出本文实验配置, 然后给出 CPP 问题动态规划求解算法的实验结果。

实验配置: 实验基于两个集群: 一个位于 UTD, 一个位于 UCI。第一个集群包括 14 个节点, 作为私有云节点使用; 第二个集群包括 38 个节点, 作为公共云节点使用。私有云节点包括奔腾 IV 处理器 290~320 GB 磁盘空间、4 GB 内存、公共云节点 AMD 双核心处理器、631 GB 磁盘空间、8 GB 内存。两个集群基于 Hadoop v0.20.2 和



Hive v0.7.1 进行配置。

统计采集:统计采集模块分析了 100 GB TPC-H 数据集^[10],为 TPC-H 关系的每个属性生成等宽柱状图。在 Hive 中使用整数、双精度和字符串数据类型来表示 TPC-H 数据。此外,还创建“date(日期)”数据类型以表示来自 TPC-H 模式的各种日期。柱状图使用的分区数量取决于数据类型;该数量对已知类型数据是固定的:对整数和双精度类型,分区数量为 $b(max-min)$,其中 max 和 min 表示 TPC-H 要求的最大和最小域值;对字符串类型,创建 95 个分区,覆盖字母(a~z 和 A~Z)、数字(0~9)及所有特殊字符(!、@、#等)。

检索工作量:本文实验使用 TPC-H 基准且比例因子为 100,使用的检索工作量有 40 个检索,包括变更过的 TPC-H 检索 Q^1 、 Q^3 、 Q^6 和 Q^{11} 。本文对各个检索均没有执行分组和聚合操作,因为在 Hive 上为这些运算符估计总体 I/O 尺寸的复杂度太高。另外,假设工作量中各个检索的概率是相同的。随机变更每个检索的谓词,以改变被访问的数据范围,满足 TPC-H 的要求。

权重 ω_x 的估计:权重 ω_x 为在公共云和私有云上每秒执行的 I/O 操作的数量。通过运行上面含有 300 GB 数据集的所有 22 个 TPC-H 检索,估计了本文混合云架构的权重 ω_{pub} 和 ω_{priv} 。然后, $\omega_{pub}(\omega_{priv})$ 为公共端(私有端)检索所要求的 I/O 操作与公共端(私有端)运行所有检索所要求的总时间之比。于是, ω_{pub} 和 ω_{priv} 估计为 4230663005 byte/s (≈ 40 MB/s)和 8786423.69 byte/s (≈ 8 MB/s)。当 ω_{pub} 数值较高时,表明本文公共云的 I/O 吞吐量大于私有云。

资源分配成本的计算:基于 Amazon Web Services 的单位价格计算资源分配成本。具体来说,利用 Amazon S3 定价策略来确定存储(0.140 美元/GB + PUT)和通信(0.120 美元/GB + get)成本,其中 put 和 get 操作的价格分别为每 1 000 个请求 0.01 美元和每 10 000 个请求 0.01 美元。同时,利用 Amazon EC2 和 EMR 定价策略来计算处理成本,即 0.085 美元/h+0.015 美元/h=0.1 美元/h。最后,根据先前指定数值,将整个数据集和检索工作量计算转移到公共端,估计了公共云资源分配总成本 PUB_MAX_COST 约为 25000 美元。在实验中将 PRA_CONST 定义为 PUB_MAX_COST 的一部分(如占比为 25%、50%、75%和 100%)。

使用如下策略来定义敏感数据泄露风险。

(1)为已知实验中使用的数据集定义总体敏感度。该

敏感度假设 customer 表的 c_name 、 c_phone 和 $c_address$ 属性的所有单元始终为敏感数据,而 lineitem 表中元组的部分内容为敏感数据(lineitem 表 1%、5%或 10%内容为敏感数据)。这提供了可在各种实验中使用的多种总体敏感度。

(2)定义 $DISC_CONST$ 为已知数据集总体敏感度的一部分,即 $DISC_CONST$ 范围为先前定义的总体敏感度的 0~100%。例如,如果 lineitem 表的敏感度定义为 10%,则确定 $DISC_CONST$ 的范围为 0(10%敏感元组均不应被暴露)到 100%(所有 10%敏感元组可以被暴露)。

实验结果:在所有实验中,首先计算所有计算任务在私有云上执行时,检索工作量的运行时间,并使用这一情况作为基准线来确定先前提出的 CPP 问题动态规划算法的性能。在实验中改变了所有参数,具体包括如下几种参数。

(1)资源分配成本,资源分配成本为先前确定的 PRA_CONST ,数值为 25%~100%。

(2)敏感度,定义了 4 种敏感度:零敏感度(整个数据集不含敏感性数据)、1%敏感度、5%敏感度、10%敏感度(lineitem 表中 1%、5%和 10%的元组为敏感性数据)。

(3)敏感性数据泄露水平,定义如下几种敏感性数据泄露水平:0(无敏感性数据被泄露)、10%、25%、40%、50%、75%和 100%(所有被定义的敏感性数据均被泄露)。

然后,计算了这 3 种参数不同组合时检索工作量的总体性能,具体结果如图 2 所示。从图 2 中可以看出,当用户愿意承担一定风险进而将大量敏感性数据存储于公共端上时,总体运行速度可以得到显著提升(超过 50%)。此外,即使用户愿意承担一定风险进而将大量敏感性数据存储于公共端上,公共端资源的经济成本也非常低,50%、75%、100%的资源分配成本表明,即使公共端的资源花费成本上升,这些情形的总体性能基本相同,这说明只需要 PRA_CONST 50%左右的预算就足以将节约性能提升 50%。

图 2 还表明,当用户对资源分配的投入更多时,可以进一步实现性能的总体提升(甚至高于 50%),原因是当为公共端分配更多资源时,可以利用混合云的并行操作提升性能。最后,从图 2 可以看出,当风险($\approx 40\%$)和资源分配成本($\approx 50\%$)较低时,显著提高了检索性能($\approx 50\%$)。

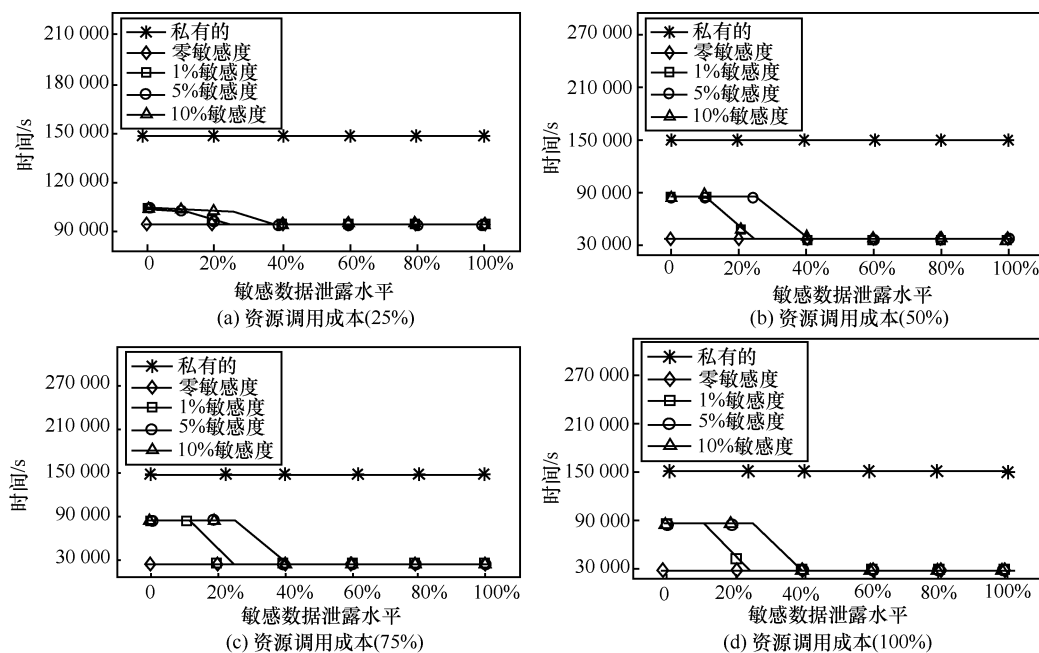


图2 CPP问题动态规划算法的性能

另外需要指出的是,在本文提出的混合云架构下(如图1所示),系统根据用户提供的检索工作量及一组资源分配和敏感性数据泄露约束,通过统计采集模型来估计每个检索所需的数据项最小集合,然后根据CPP求解算法求解CPP问题。当系统根据CPP问题的解存储了相关数据后,系统即向检索处理提供支持。

由于该方案是在考虑了经济成本的前提下,根据用户需求来进行统计采集、计算划分,减少了数据存储量,从而有效地降低了云服务的资源分配成本,也避免了在公有云和私有云上复制数据所带来的资源和存储空间的浪费。

6 结束语

随着云计算的出现,希望平衡数据安全风险同时降低公共云服务使用花费的用户可以使用混合云。但是在使用混合云之前,一个关键问题是当数据中存在敏感性信息时,需要在私有云和服务提供商之间高效地分配计算任务。本文将此问题定义为风险感知性能优化问题,并提出一种可以实现检索工作量最优分布的动态规划算法。下一步工作的重点是研究降低公共云使用导致的经济成本;另外,将本文方案进行扩展后,利用检索内并行操作来运行已知的检索任务。

参考文献

- 邓维, 刘方明, 金海等. 云计算数据中心的新能源应用: 研究现状与趋势. 计算机学报, 2013, 36(3): 582~596
Deng W, Liu F M, Jin H, *et al.* Leveraging renewable energy in cloud computing datacenters: state of the art and future research. Chinese Journal of Computers, 2013, 36(3): 582~596
- Abouzeid A, Bajda-Pawlikowski K, Abadi D, *et al.* HadoopDB: an architectural hybrid of MapReduce and DBMS technologies for analytical workloads. Proceedings of the VLDB Endowment, 2009, 2(1): 922~933
- Zhang K, Zhou X, Chen Y, *et al.* Sedic: privacy-aware data intensive computing on hybrid clouds. Proceedings of the 18th ACM Conference on Computer and Communications Security. ACM, Chicago, USA, 2011: 515~526
- Wang S, Agrawal D, El Abbadi A. A comprehensive framework for secure query processing on relational data in the cloud. Proceedings of Secure Data Management, Seattle, WA, USA, 2011: 52~69
- Di Vimercati S D C, Foresti S. Privacy of Outsourced Data. Privacy and Identity Management for Life. Berlin Heidelberg: Springer, 2010: 174~187
- Hacıgümüş H, Iyer B, Li C, *et al.* Executing SQL over encrypted data in the database-service-provider model. Proceedings of the ACM SIGMOD International Conference on Management of Data, New York, USA, 2002: 216~227

(2015024-8 接排在 2015007-5 后)