

SQL 能耗建模及优化研究

国冰磊¹ 于 炯¹ 廖 彬² 杨德先¹

(新疆大学软件学院 乌鲁木齐 830008)¹ (新疆财经大学统计与信息学院 乌鲁木齐 830012)²

摘 要 IT 系统能耗的节节攀升,使得设计新一代 DBMS 时必须考虑其能耗效率问题。由于 SQL 语句的执行过程大约消耗 70%~90% 的数据库资源,因此对 SQL 进行能耗建模及优化对提高数据库的能源使用效率具有重要的意义。在对 SQL 查询处理机制进行研究的基础上,构建了 SQL 能耗模型,并对一系列查询优化原则进行了实验,以表明不同优化原则对性能提升及能耗减少的有效性。实验及能耗数据分析表明:CPU 利用率是影响系统功耗的最关键因素,SQL 能耗优化方法可忽略内存优化且应该均衡考虑性能优化及功耗优化两方面,提出的 SQL 能耗模型及节能优化方法具有较强的应用价值。

关键词 绿色计算, SQL 能耗优化, SQL 能耗建模, 查询处理

中图法分类号 TP315 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2015.10.041

Research on SQL Energy Consumption Modeling and Optimization

GUO Bing-lei¹ YU Jiong¹ LIAO Bin² YANG De-xian¹

(School of Software, Xinjiang University, Urumqi 830008, China)¹

(College of Statistics and Information, Xinjiang University of Finance and Economics, Urumqi 830012, China)²

Abstract The increasing energy consumption of IT system makes us take energy efficiency into consideration when designing a new generation of DBMS. Because SQL queries consumes almost 70%~90% of the database resources, the energy efficiency of database can be improved by optimizing query and energy consumption modeling. After an in-depth study on optimization of query processing mechanism, an energy consumption model for SQL was proposed and many experiments were designed on a series of query optimization methods to show their effectiveness of performance improvement and energy reduction. Experiments and energy consumption data analysis prove that CPU utilization is the most critical factor that affects power consumption, SQL energy consumption optimization can ignore the memory optimization and should balance two aspects: performance optimization and power consumption optimization, and the model and the proposed methods have good application value.

Keywords Green computing, SQL energy consumption optimization, SQL energy consumption modeling, Query processing

1 引言

数据中心数据^[1]的爆炸性增长让我们不得不关注数据库系统设计过程中的能源成本。文献[2-4]指出美国一年用于供应服务器的电力成本为 27~140 亿美元。此外,据纽约时报^[20]报道,全球数据中心一年的总用电量约为 3000 亿瓦特,相当于 30 座核电厂的产电量,而巨大的能耗中却只有 6%~12% 的能耗被用于处理相应用户的请求。对一个普通数据中心来说,服务器和冷却系统消耗的电力大概占总拥有成本(Total Cost of Ownership, TCO)的 20%,相当于总维护成本的 1/3^[5]。这让电力成本在一个普通 IT 部门中成为仅次于劳动费用的第二大成本。此外,文献[2-4]认为在未来几年内

电力消耗还会持续增长。与此同时,系统的高能耗将导致温室气体过量排放并引发环境问题。文献[6]指出,IT 领域的二氧化碳排放量占全球的 2%,到 2020 年这一比例将翻一番。因为数据中心无法控制能源的使用,供电和冷却成本已经开始超过硬件成本。此外,这对数据中心的密度、可扩展性和相关环境的设计都产生了负面影响。最后,增加的能耗导致的环境问题促使政府在全球各地开始监管企业的 IT 能力,并且这一问题仅靠节能的硬件和操作系统是无法解决的。所以无论是从降低数据中心的运营成本,还是从降低能耗、保护环境的角度出发,研究数据库节能技术都具有现实意义与应用前景。

为了提高数据中心的能源使用效率,学术界与工业界分

到稿日期:2014-10-09 返修日期:2014-12-08 本文受国家自然科学基金项目(61262088, 60863003, 61063042, 61363004),新疆维吾尔自治区自然科学基金项目(2011211A011)资助。

国冰磊(1991-),女,硕士生,CCF 学生会会员,主要研究方向为云计算、绿色计算, E-mail: xinggbli@163.com; **于 炯**(1964-),男,博士,教授,博士生导师,主要研究方向为云计算与大数据; **廖 彬**(1986-),男,博士,讲师,主要研究方向为数据库理论与技术、绿色计算、数据挖掘; **杨德先**(1991-),男,硕士生,主要研究方向为云计算、绿色计算。

别从硬件、操作系统、虚拟机、分布存储系统^[7-9]等层次去提高数据中心的能源使用效率。而本文关注 DBMS 中的查询优化模块,从查询处理的方式以及查询优化(能耗优化)的本质考虑数据库系统的节能问题,提出基于 SQL 的能耗优化方法。本文的主要贡献在于在对数据库优化机制分析的基础上,构建了 SQL 能耗模型,总结了一系列节能的查询优化机制,并通过大量的实验及能耗数据分析,验证了能耗优化方法的有效性;其次,在验证原则节能潜力的同时,通过对比拥有相同查询结果不同执行计划的 SQL 语句在优化前后的性能、CPU 利用率、内存利用率的变化趋势,研究这些变化对 SQL 语句执行功耗或能耗的影响,得出了其对能耗优化有指导意义的结论,为未来设计与实现基于资源利用率累加的节能数据库奠定了基础。

本节介绍相关背景;第2节对相关工作进行了介绍;第3节提出了 SQL 能耗模型并进行优化案例设计与分析;第4节结合理论与实验对本文提出的 SQL 能耗模型、SQL 能耗优化方法进行验证,并对其进行分析,得出相关结论;最后对全文进行了总结。

2 相关研究

本节将数据库系统中的能耗优化问题分为基于硬件的节能与基于软件的节能两个方面进行讨论。在硬件层面上,主要通过以高能效的设备(SSD)替换现有的低能效设备,来达到节能的目的。图灵奖获得者 Gray 先生就曾预测:“就像磁盘会取代磁带一样,闪存将会取代磁盘”^[10]。然而,现有的数据库系统大都基于磁盘存储进行设计和优化,闪存与磁盘不同的物理特性,简单地把磁盘替换为 SSD 不能充分发挥闪存的特性^[11-14]。基于闪存的数据库是短期内的研究热点^[10-14],文献[15]基于此提出了闪存数据库系统框架,总结了缓冲区、索引、查询和事务等数据库关键技术。虽然基于闪存的固态硬盘比磁盘具有更大的性能优势,但是短期内 SSD 不会完全取代磁盘成为主流存储介质,此外大量的硬件替换也会给数据中心带来成本过高的问题。为此,构建混合存储系统是当前的研究热点^[10,15]。软件节能主要是通过研究影响数据库能耗的因素,为数据库工作负载估算能耗,选择节能的查询计划从而达到节能的目的。现阶段的软件节能研究不需重新构建硬件体系,但是随着 SSD 这样的高能效硬件逐渐取代旧一代的硬件设备,软硬件结合必然是未来的研究热点。

当前软件节能研究主要集中在构建功率感知的节能查询优化器。构建这样的优化器需要运用两个模型,一个是功率估算模型或能耗估算模型,用于为数据库工作负载估算能耗;另一个是成本评价模型,用于选择满足系统性能和能耗要求的查询计划。其中功率估算模型又分为以下两种类型:为数据库运算符构建的估算模型和为数据库查询计划构建的估算模型。为此,文献[14]用两类数据库和存储管理器做实验,改变影响因素(跨度从不同的查询计划到压缩算法,从物理布局到 CPU 频率和操作系统调度)后发现最节能的数据库配置通常也是性能最好的,通过优化数据库配置能达到节能的目的。文献[16]利用传统数据库中查询优化领域的成熟技术与控制论模型对数据库进行优化,讨论了利用功耗模型来精确测量

查询计划的能源成本及通过成本评估模型选择查询计划,参数估计带遗忘因子递归最小二乘法(RLS)实现了一个在线的模型估算方法,达到了为数据库选择节能的查询计划的目的。基于此,文献[17]提出了一个节能框架,这个框架在为 DBMS 节能的同时确保性能是可接受的。后期,Zichen Xu^[18]等人又提出了更高级的框架版本(Power-Energy-Time, PET),PET 为一组通用的关系运算符构建了功率模型,并通过一个查询评估引擎按照性能和能源消耗来评估一个查询计划的好坏,从而达到节能的目的。文献[19]使用 multiple-linear(多元线性)回归和因子实验设计,依靠现成的统计数据为数据库工作负载实现了准确的峰值功耗估算模型,并通过在执行前预测工作负载的能源成本,避免高能耗负载来达到节能的目的。

为了提高数据库的能源使用效率,现有的研究大都通过提出成本模型的方法来估算数据库工作负载的能源成本。但不幸的是,以上这些方法有时会产生大量的估计误差,增加了对数据库查询进行调度的可能性;对于不符合特定关系的工作负载,还需要在原来的模型上进行扩展,增加了建模的复杂性并局限了查询优化器设计的广泛性。文献[14]的工作表明,当前的节能查询优化器只是通过减少资源使用这一个优点来减少能源消耗。这种策略虽然可能对单站点查询工作负载适用,但是不适用于处理虚拟化多用户数据库系统。

文献[14]与本文的工作最为相关,通过观察硬件组件利用率对数据库能耗的影响得出相关结论,重点考察了 CPU 与磁盘的使用。本文与已有工作的不同之处在于侧重研究性能、CPU 利用率、内存利用率对数据库功耗或能耗的影响,通过实验和手工调优的解决方案总结了一系列立竿见影的查询优化机制,验证了这些机制的节能潜力,并得出了相关结论。因为本文提出的数据库能耗优化方法可以直接应用到真实的生产环境中,能够避免高能耗的查询方法,为数据库能耗优化方面的研究人员提供了一套解决问题的思路及方法,也为数据库管理员提供了能够量化的 SQL 能耗优化方案,所以本文的研究具有较好的实用性。

3 SQL 能耗模型及优化案例设计

3.1 SQL 能耗模型及优化分析

SQL 语句是操作数据库的标准接口,所有上层应用程序对数据库的操作最终都会转化为 SQL 语句对数据库的操作。而 SQL 语句的执行效率和产生的能耗将直接影响数据库的性能及能量消耗。所以,保证性能的同时减少 SQL 语句的执行能量消耗,是当前功耗感知数据库的研究重点。SQL 语句的执行过程大约消耗 70%~90% 的数据库资源,所以对 SQL 进行能耗建模及优化对提高数据库能源使用效率具有重要的意义。SQL 执行过程中消耗的资源包括 CPU、磁盘、内存、网络等,SQL 语句的执行能耗由使用相关资源产生的能耗累加而成。

其中,CPU 的能耗模型受到处理器的活动状态、指令执行情况、高速 cache 使用情况、当前执行频率及制造工艺等因素的影响。设 P_{cpu} 表示 CPU 利用率为 u_{cpu} 时的功率,根据 Kansal 等人在文献[21]中提出的能耗模型, P_{cpu} 与 u_{cpu} 关系可表示如下:

$$P_{cpu} = a_{cup} \cdot u_{cpu} + \lambda_{cpu} \quad (1)$$

式中, a_{cup} 与 λ_{cpu} 为 CPU 能耗模型的常数, 不同的 CPU 之间存在着差异, 其值可通过大量的训练获得。

已有大量工作针对内存的能耗模型进行了研究, 发现影响内存能耗的主要因素是内存读写数据的吞吐量^[22]。记录内存最后一层 Cache (Last Level Cache) 的缺失次数, 是一种轻量级的内存吞吐量评估方法。根据吞吐量指标, 设 $E_{mem}(T)$ 表示内存存在一条 SQL 语句执行的 T 时间区间内的能耗, N 表示最后一层 Cache 缺失次数, 内存模型表达如下^[21]:

$$E_{mem}(T) = a_{mem} \cdot N + \lambda_{mem} \quad (2)$$

其中, a_{mem} 与 λ_{mem} 为内存能耗模型的常数。

磁盘能耗与读写数据量密切相关, 设 $E_{disk}(T)$ 表示一条 SQL 执行过程中 T 时间区间内磁盘的能耗, r 与 w 分别表示在 T 时间区间内磁盘的读与写数据量, 磁盘能耗模型可表达如下^[22]:

$$E_{disk}(T) = a_r \cdot r + a_w \cdot w + \lambda_{disk} \quad (3)$$

其中, a_r 与 a_w 分别表示磁盘读与写参数, λ_{disk} 表示磁盘的静态能耗, 3 个参数的值可通过大量的能耗数据分析与训练获得。

与磁盘能耗模型类似, 网卡的能耗与发送、接收到的数据量成正比。设 $E_{net}(T)$ 表示网卡在 T 时间区间内 (一条 SQL 的执行时间) 的能耗, s 与 v 分别表示在 T 时间区间内网卡发送与接收到的数据量, 网卡能耗模型可表达如下:

$$E_{net}(T) = a_s \cdot s + a_v \cdot v + \lambda_{net} \quad (4)$$

其中, a_s 与 a_v 分别表示网卡在发送、接收数据时的能耗参数, λ_{net} 为网卡的静态能耗参数。

设任意一条 SQL 语句执行成功后的能耗为 E_{SQL} , 由于一条 SQL 执行过程中主要消耗了 CPU、磁盘、内存及网络资源, 因此在一条 SQL 语句执行完毕的时间周期 $T = [t_s, t_e]$ 内, 其能耗可由式(5)得到:

$$\begin{aligned} E_{SQL} &= E_{cpu} + E_{mem} + E_{disk} + E_{net} \\ &= \int_{t_s}^{t_e} (a_{cup} \cdot u_{cpu} + \lambda_{cpu}) + \int_{t_s}^{t_e} (a_{mem} \cdot N + \lambda_{mem}) + \\ &\quad \int_{t_s}^{t_e} (a_r \cdot r + a_w \cdot w + \lambda_{disk}) + \int_{t_s}^{t_e} (a_s \cdot s + a_v \cdot v + \lambda_{net}) \end{aligned} \quad (5)$$

由式(5)可知, 可通过减少 SQL 执行时间及优化 SQL 执行过程中的资源利用率两方面达到优化 SQL 能耗的目的。首先, 本文研究 CPU、磁盘、内存等组件对 SQL 语句执行能耗的影响, 从而能够衡量 SQL 语句的质量及其节能潜力, 为设计与实现基于资源利用率累加的节能数据库做铺垫。其次, 本文通过大量的实验总结了一系列具有节能潜力的 SQL 优化原则。

3.2 优化案例设计

关系型数据库具有理解容易、使用方便、维护简单的特点, 并利用二维表模型存储了几乎整个世界的结构化数据, 使得关系型数据库在现代技术与商业中起着非常重要的作用。由于 Oracle 是目前关系型数据库中最流行的产品, 因此本文选择 Oracle 10g 数据库作为实验平台。

SQL 的优化可分为以下 5 个步骤:

- 204 •

(1) 找出顶级 (高负荷) SQL。

(2) 验证查询优化器给出的执行计划是否合理。

(3) 检查执行计划中的统计信息。

(4) 分析相关表的记录数、索引使用情况。

(5) 进行优化措施: 改写 SQL、使用 Hint、调整索引等。

从而达到最佳的执行计划。

实验主要关注步骤(5)。本文设计了多组实验并选取其中有代表性的 20 组实验, 这些优化规则包括避免索引列上使用计算、避免索引列上进行数据类型转换、SQL 语句区分大小写等 (表 2 与表 3 分别为实验表的表结构)。每组实验里模拟了两句 SQL 语句 (执行结果完全一样, 为等价的 SQL 语句)。其目的是使有相同查询结果的 SQL 语句产生不同的执行计划, 在节能的前提下观察两种执行计划的性能、CPU 利用率、内存利用率的变化, 对 3.1 节中提出的能耗模型进行验证。实验把一组 SQL 语句分为优化之前和优化之后, 通过测量每组实验的功率以及所用时间, 将两者相乘得到消耗掉的总能量, 同时测试 Oracle 进程实时使用实验机的 CPU 利用率和内存利用率情况等。其中功率与能耗之间的关系如式(6)所示:

$$E = pt \quad (6)$$

其中, E 为消耗的总能量 (单位为焦耳 J), p 为功率 (单位为瓦特 W) (在此处采用平均功率计算), t 为时间 (单位为秒 s)。

因为 Oracle 有着一套优良的优化机制, 为了保证测试结果的准确性, 在每次测量时都清空了它的 Cache 以及 SHARE_POOL; 而且为了避免误差, 每一组实验都要连续测量 100 次取平均值, 此功能通过 PL/SQL 存储过程来实现。任意 SQL 语句进行测量之前, 需要注意 Oracle 内部提供的执行计划, 以免造成较大的实验误差。

实验中, 20 个优化规则测试环境为表 EMP 及表 DEPT, 两个表的数据都在同一个表空间中。EMP 表及 DEPT 表结构分别如表 1 及表 2 所列, 其中 EMP 表数据量为 355 万, DEPT 表数据量为 15 万, 笛卡尔集操作数据量为 5325 万。

表 1 EMP 表结构

字段名	数据类型	大小	是否允许为空	备注
Emp_no	Number		否	主键
First_name	Varchar2	20	否	
Last_name	Varchar2	20	否	
Gender_sex	Varchar2	5	否	
Job_todo	Varchar2	20	否	
Manager	Number		是	
Salary_month	Number		否	
Hire_date	Date		否	
Birth_state	Varchar2	20	是	
Dept_no	Number		否	

表 2 DEPT 表结构

字段名	数据类型	大小	是否允许为空	备注
Dept_id	Number		否	主键
D_name	Varchar2	20	否	
Location	Varchar2	20	否	

4 实验设计及评价

4.1 实验环境及参数配置

实验采用北电电力监测仪 (USB 智能版), 数据采样频率

设置为 1 次/秒,能耗数据(包括瞬时功率、电流值、电压值等)可通过 USB 接口实时地传输到能耗数据监测机上,实现能耗数据的收集。实验总体环境描述如表 3 所列。

表 3 总体实验环境描述

项目	描述
操作系统	Windows XP
数据库管理系统	Oracle 10g
能耗数据测量	北电电力监测仪(USB 智能版),标准为 GB/T17215-2003,功率误差值±0.01~0.1W,采样频率为 1.5s~3s 之间,单位为 kWh
能耗数据采集	电力监测仪用电监测系统 V1.0.1
能耗相关单位	功率:瓦特(W),能耗:焦耳(J),时间:秒(s)
数据采样频率	1 秒采集数据 1 次
实验机 CPU	DualCore Intel Pentium D 925,3000 MHz (15 x 200)
实验机内存	1527 MB (DDR2 SDRAM)
实验机硬盘	ST3250824A (250 GB,7200 RPM,Ultra-ATA/100)

实验采用两台主机进行测试,一台作为实验机,安装 Oracle 10g 和应用程序性能监测软件;另一台用于能耗数据采集,安装测电软件。为了保证实验结果的准确性,避免能耗数据采集干扰,通过双机通信的方式进行实验。实验测试所用数据使用 Oracle 数据生成器自动模拟,表 EMP 及表 DEPT 结构见表 1 及表 2。

4.2 实验步骤及注意事项

实验的 5 个步骤如下:

- (1)部署实验设备,准备实验环境(安装 DBMS、应用程序性能实时监测软件、功率监测软件、Office 软件等)。
- (2)测试之前,校对实验机的时间,清空数据收集软件内之前的数据,把待测试 SQL 语句按情况放入 PL/SQL 代码段。
- (3)开始测试,先运行测试软件,再执行 SQL 语句,不定期检查有无异常。

(4)测量完毕,收集数据;计算并记录下一次运行时间,从测电软件导出并记录测试本句的实时功率,计算并记录平均功率,导出并记录 Oracle 进程的 CPU 和内存使用情况,计算并记录其平均值。

- (5)重复步骤(2),开始测量下一条 SQL 语句。

实验过程中需要注意以下 4 点:

(1)为了避免误差,实验时每一条 SQL 要在清空缓存的情况下连续测试 100 次,而 Oracle 没有提供这样的机制,所以采用 PLSQL 实现。而 PLSQL 虽然是面向对象的,但是普通的 SQL 语句不能无缝地嵌套在 PLSQL 里,所以采用 PLSQL+动态 SQL 的测试方式。

(2)PLSQL 存储过程是面向对象的,即把 SQL 的执行结果当作一个有数据类型的对象保存在相应的容器中。而实验中 SQL 的执行结果的数据类型并不唯一,所以把 SQL 放入 PLSQL 代码段中时要注意 SQL 的执行结果是什么。如:结果为一个数字就放入数字类型;若结果是一张表,就要放在一个表对象里。

(3)因为数据收集是在不同的实验设备上完成的,而原始数据又全都为实时数据,所以所有实验机器的时间一定要校对到分秒不差。

(4)为了保证 SQL 语句被完全测试,要先运行监测软件,再运行 SQL 执行,测试完毕收集数据时,以 SQL 的执行时间

为主,即在监测软件的收集结果中去掉开头和结束两段无用的数据。

将实验中实现的 PLSQL 存储过程的代码模板进行整理,如表 4 所列。

表 4 PLSQL 存储过程代码

存储类型	对应代码
结果为数字的模板	<pre>declare i number:=0; counts number; begin loop i:=i+1; execute immediate 'alter system flush BUFFER_CACHE'; execute immediate 'alter system flush shared_pool'; select count(*) into counts from testsys.employee_table where birth_state is null; exit when i=10; end loop; end;</pre>
结果为表的模板	<pre>declare i number:=0; type t_table_type is table of testsys.employee_table%rowtype; t_table t_table_type; begin loop i:=i+1; execute immediate 'alter system flush BUFFER_CACHE'; execute immediate 'alter system flush shared_pool'; execute immediate '待测 SQL 语句'; bulk collect into t_table; exit when i=100; end loop; end;</pre>
结果为字符串的模板	<pre>declare i number:=0; str_job varchar2(50); str_ave varchar2(50); begin loop i:=i+1; execute immediate 'alter system flush BUFFER_CACHE'; execute immediate 'alter system flush shared_pool'; select job_todo,avg(salary_month) into str_job,str_ave from testsys.employee_table group by job_todo having job_todo='SUPERVISOR'; exit when i=100; end loop; end;</pre>

4.3 实验结果

本节将实验测得的原始数据整理后总结为实验结果,如表 5 所列。根据实验结果,整理出各组实验优化效果汇总表,如表 6 所列。

根据表 6 的优化效果,按性能优化的程度,实验结果可分为以下 3 组:

- (1)优化效果明显的,实验编号为 1,2,3,4,5,9,10,11,12,13,优化规则总结为避免索引失效或者减少排序。
- (2)优化效果一般的,实验编号为 6,7,8,15,优化规则总结为一般替代。
- (3)优化没有效果的,实验编号为 14,16,17,18,19,20,优化规则总结为失效或过时。其中实验 19,20 的规则是在 RBO(Rule Based Optimization)优化模式下有效的,而 10g 以后已经废除 RBO 模式,故没有测试。

表5 实验结果汇总

编号	优化前时间 (s)	优化后时间 (s)	优化前平均 CPU使用 (%)	优化后平均 CPU使用 (%)	优化前平均 内存使用 (%)	优化后平均 内存使用 (%)	优化前平均 功率 (W)	优化后平均 功率 (W)	优化前 能耗 (J)	优化后 能耗 (J)
1	12.8	0.93	7.24	14.97	32.34	32.32	92.42	94.19	1183	87.6
2	14.89	7.45	16.44	5	30.95	31.2	96.56	91.8	1438	683.9
3	13.6	12.32	13.83	4.14	31.39	31.46	95.65	91.68	1301	1130
4	14.34	1.05	13.3	14.06	31.48	31.55	95.12	94.29	1364	99
5	13.26	1.93	11.75	8.52	30.94	6.47	94.46	91.95	1253	177.5
6	21.48	20.82	32.73	39.23	68.94	70.47	102.9	107.9	2210	2247
7	16.23	15.48	19.03	17.66	39.72	40	94.95	95.04	1541	1471
8	13.53	13	10.71	7.47	29.53	31.26	96.43	92.81	1305	1207
9	21.91	15.39	29.62	37.94	53.97	53.93	99.82	102.8	2187	1583
10	5.36	1.05	12.42	15.39	32.62	30.42	94.68	93.68	507.5	98.36
11	15.7	1	18.64	15.04	32.33	32.33	96.64	93.5	1517	93.5
12	5.84	1.2	38.04	12.33	30.74	30.71	102.3	92.8	597.3	111.4
13	15.5	13.09	19.38	8.23	30.96	30.86	96	92.19	1488	1207
14	12.99	13.51	10.36	10.17	31.92	32.83	92.73	93.51	1205	1263
15	3.58	3.48	21.84	21.78	30.41	30.43	97.31	97.09	348.4	337.9
16	14.27	14.27	14.66	15.29	35.93	37.02	94.74	94.75	1352	1352
17	14.28	14.53	17.22	17.02	41.36	39.52	97.18	95.58	1388	1389
18	13.22	13.23	12.05	12.01	34.6	34.58	93.76	93.55	1240	1238

表6 各组实验优化效果汇总

编号	净时间优化 (单位:s)	优化率 (%)	净能耗优化 (单位:J)	优化率 (%)
1	11.87	92.73%	1095.4	92.6%
2	7.44	49.97%	754.1	52.44%
3	1.28	9.41%	171	13.14%
4	13.29	92.68%	1265	92.74%
5	11.33	85.44%	1075.5	85.83%
6	0.66	3.07%	-37	-1.67%
7	0.75	4.62%	70	4.54%
8	0.53	3.92%	98	7.51%
9	6.52	29.76%	604	27.61%
10	4.31	80.41%	409.14	80.62%
11	14.7	93.63%	1423.5	93.84%
12	4.64	79.45%	485.9	81.35%
13	2.41	15.55%	281	18.88%
14	-0.52	4%	-58	-4.81%
15	0.1	2.79%	10.5	3.01%
16	0	0	0	0
17	-0.25	-1.75%	-1	-0.07%
18	-0.01	-0.08%	2	0.16%

4.4 实验结果分析

观察图1(a),优化后的SQL语句执行能耗基本处于优化前的SQL语句执行能耗之下,证明了能耗优化原则的有效性,优化率最高可达到92.6%,具体的优化数据见表6。

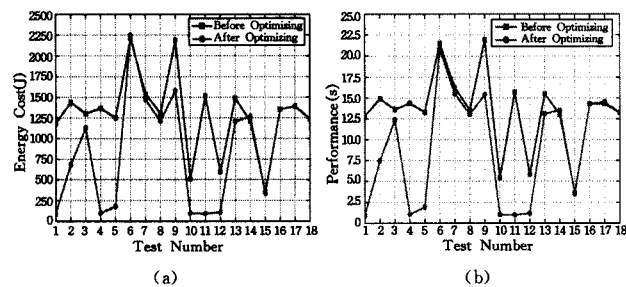


图1 能耗优化前后与性能优化前后的对比

整体上对比图1,发现优化前后的能耗与时间基本一致,这是由于性能改进是决定SQL能耗的关键因素,性能改进最显著的实验组为1、2、4、5、9、10、11、12,同时它们也是能耗改进最显著的。与此相比,实验组3和13的能耗改进也很显著,这两组SQL能耗改进高达18.88%,但是性能改进却不

到3%。产生这一现象的原因是资源的利用率得到显著优化,最终达到节能的目的,这也印证SQL能耗模型即式(5)。基于此,推测时间与能耗成正比关系,即最节能的查询拥有最佳的性能改进,符合传统数据库的优化策略(即数据库优化的本质就是最大限度地缩短查询时间)。但结合式(6)发现,若性能与能耗是正比关系,则功率应该是个常数,观察表5发现功率的跨度值为16.22(最低91.68W,最高107.9W),不是一个常数。对比表5中优化前后的性能及功率,发现产生这一现象的原因是性能的波动幅度(最高14.7s)总体上比功率波动幅度(最高5W)大,这就造成性能的波动对能耗的影响偏大,而不是其在总能耗中占的比例偏大,功率始终是影响能耗的最关键因素。综上证明了3.1节提出的SQL能耗模型公式,即可以通过减少SQL的执行时间和资源利用率达到节能的目的,同时启发本文对性能波动幅度大的SQL要针对其性能进行优化,对功率波动幅度大的SQL要针对其功率进行优化。另外,还发现了一个有趣的现象,即序号6的实验组出现性能得到改进,但是能耗反而升高的现象,分析数据后发现原因是:性能的波动幅度小于功率的波动幅度(性能对能耗的影响偏小),虽然性能得到了改进(改进为0.66s),但是CPU、内存等资源利用率升高导致了功耗的升高,最终导致总能耗的增多(增值为37J)。这种牺牲功率获得性能改进的方法是不值得提倡的,因为它最终导致了能耗的增加,表明传统数据库的优化理念(数据库优化的本质就是减少SQL的执行时间)不适用于SQL能耗优化,印证了SQL能耗模型,即节能的查询要同时考虑性能改进和减少资源使用率。这一现象也印证了文献[18]的观点:节能的查询不一定有最短的处理时间(最佳的性能)。由此得出结论:节能的查询通常有较好的性能,但不一定有最短的处理时间,能耗优化要均衡考虑性能和功率两方面,对数据库的能耗优化要有针对性,一味地追求性能而忽略功率终将导致能源的浪费。

基于以上结论,为了进一步得到硬件组件利用率对SQL执行能耗的影响(这也是要模拟相同执行结果不同执行计划的SQL语句的本质原因),本文对实验数据进行整理与分析,得到图2。

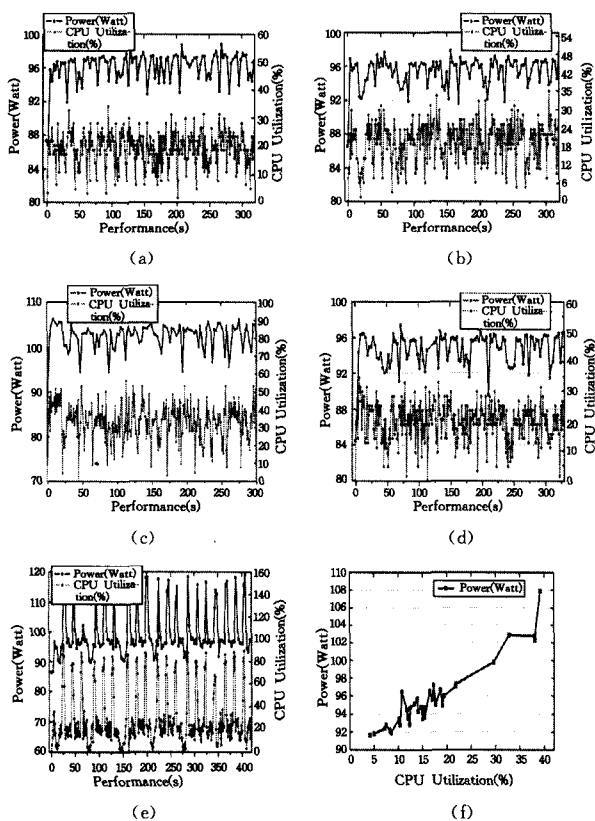


图2 功率与CPU利用率的关系

由图2可以看出,被执行SQL语句的实时功率与Oracle进程的CPU利用率之间的联系紧密,当CPU利用率上升时,功率上升;当CPU利用率下降时,功率下降;CPU利用率的变化趋势与功率的变化趋势基本一致,尤其是图(e)。同时,观察表6发现,序号为1的实验组在优化后性能得到大幅改进,CPU利用率大幅上升;序号为2的实验组在优化后性能大幅改进,CPU利用率大幅下降;对比其他实验组发现,性能与CPU利用率之间没有明确关系。由图2中的平均值汇总表(图(f))发现:功率与CPU利用率之间有强烈的正相关性,由SQL能耗模型可得CPU利用率不是影响功率使用的唯一因素,纯粹基于CPU利用率的简单模型不能准确预测功耗。由此得出结论:CPU利用率是影响功率使用的最关键因素。下面继续探究内存利用率对SQL执行功耗的影响。

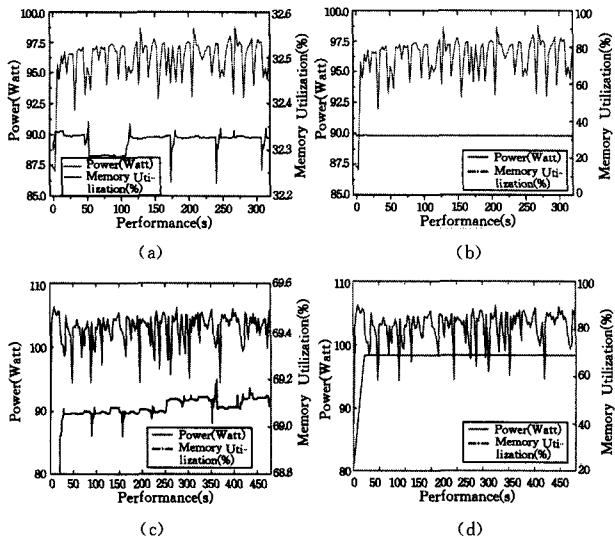


图3 瞬时内存利用率与功率的关系

由图3(a)、(c)可以看出在细小的波动范围内,内存利用率和功率之间没有明确的关系,但是当把波动范围扩大到正常值时内存的变动就可以忽略不计了,如图3(b)、(d)。

结合图4,可以看出优化前后Oracle进程对内存的占用基本没有变,再结合数据分析:即使一个SQL语句的执行能耗在优化前后变动很大(最高优化率为93.84%,平均优化率为34.97%),内存的使用也没有明显变化。结合表5,内存的使用在优化前后整体基本没有变化,即内存产生的功耗是一个基本不变的值,亦即内存的波动对SQL语句的执行总功耗无影响。基于此得出结论:通过优化内存利用率带来的功率优化升值空间不大,内存利用率对功率的影响可以忽略不计,在对数据库优化时可以基本忽略内存优化。考虑到SSD是一种低能耗高效率的存储设备,预测这些节能的优化原则在SSD上会有更好的优化效果,同时联系第2节关于软硬件优化的总结,下一步预期在存储设备为SSD的数据库上更进一步探究这些组件如何影响SQL的执行能耗,软硬件结合的优化是未来数据库节能研究的趋势。

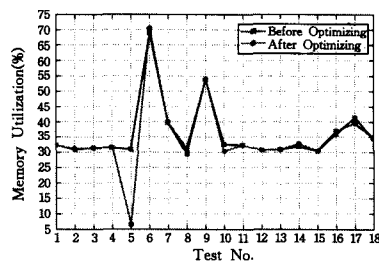


图4 优化前后平均内存使用对比

结束语 现有的数据库优化集中在最小化响应时间,优化的重点是通过最小化SQL查询执行时间达到数据库应用程序性能最大化的目的。但是这种传统的优化方式却没有考虑到数据库的能耗优化问题。本文在建立SQL能耗模型的基础上,关注影响SQL执行能耗的因素,并在单站点数据库服务器上使用一组广泛的查询进行实验,总结了一系列节能的查询优化原则。经过实验对这些查询优化原则的节能效果进行了量化,结合实验数据从理论上论证了能耗模型的有效性。通过实验及能耗数据分析表明,CPU利用率是影响系统功耗的最关键因素,SQL能耗优化方法可忽略内存优化且应该均衡考虑性能优化及功耗优化两方面。本文的研究成果为将来研究能耗感知的数据库查询优化系统打下了基础。

下一步工作重点:以开源数据库系统为基础,进一步研究如何将本文提出的SQL能耗模型应用在实际关系数据库中,并开发以能耗为优化目标的SQL自动优化框架;同时,结合本文提出的SQL能耗模型,研究语句级的SQL能耗预测及实时测量方法。

参考文献

- [1] Katz R H. Tech titans building boom [J]. IEEE Spectrum, 2009,46(2):40-54
- [2] Koomey J G. Estimating total power consumption by servers in the US and the world[EB/OL]. http://ccsl.iccip.net/koomey_long.pdf
- [3] IDC. Solutions for Data Centers' Thermal Challenges[EB/OL]. http://www.blade.org/docs/wp/idc_cool_bluewhitepaper.pdf

(下转第231页)

- Proceedings of the IEEE Congress on Evolutionary Computation (CEC 1998). Piscataway, NJ, 1998: 69-73
- [3] Tizhoosh H R. Opposition-based learning: A new scheme for machine intelligence [C]// Proceedings of the IEEE International Conference on Computational Intelligence for Modelling, Control and Automation, 2005 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce. 2005: 695-701
 - [4] Wang Hui, Li H, Liu Y, et al. Opposition-based particle swarm algorithm with Cauchy mutation[C]// Proceedings of the IEEE Congress on Evolutionary Computation, 2007. Tokyo, 2007: 356-360
 - [5] Wang Hui, Wu Zhi-jian, Rahnamayan S, et al. Enhancing particle swarm optimization using generalized opposition-based learning [J]. Information Sciences, 2011, 181(20): 4699-4714
 - [6] 周新宇, 吴志健, 王晖, 等. 一种精英反向学习的粒子群优化算法 [J]. 电子学报, 2013, 41(8): 1647-1652
Zhou X Y, Wu Z J, Wang H, et al. Elite Opposition-Based Particle Swarm Optimization [J]. Acta Electronica Sinica, 2013, 41(8): 1647-1652
 - [7] Wang Hui, Wu Z J, Liu Y, et al. Space transformation search: a new evolutionary technique [C]// Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation, 2009. New York, USA, 2009: 537-544
 - [8] 龚纯, 王正林. 精通 MATLAB 最优化计算 [M]. 电子工业出版社, 2012: 283-285
Gong Chun, Wang Zheng-lin. Proficient optimization calculation in MATLAB [M]. Electronic Industry Press, 2012: 283-285
 - [9] 汪慎文, 丁立新, 谢承旺, 等. 应用精英反向学习策略的混合差分演化算法 [J]. 武汉大学学报(理学版), 2013, 59(2): 111-116
Wang Shen-wen, Ding Li-xin, Xie Cheng-wang, et al. A hybrid differential evolution with elite opposition-based learning [J]. J. Wuhan Univ. (Nat. Sci. Ed.), 2013, 59(2): 111-116
 - [10] van den Bergh F. An Analysis of Particle Swarm Optimizers [D]. South Africa: Department of Computer Science, University of Pretoria, 2002
 - [11] Tang Ke, Li Xiao-dong, Suganthan P N, et al. Benchmark Functions for the CEC' 2010 Special Session and Competition on Large-Scale Global Optimization [R]. Hefei: Nature Inspired Computation and Applications Laboratory, USTC, 2009
-
- (上接第 207 页)
- [4] Institute for Energy Efficiency. Greenscale [EB/OL]. <http://www.iee.ucsb.edu/greenscale>
 - [5] Rasmussen N. Determining Total Cost of Ownership for Data Center and Network Room Infrastructure [EB/OL]. http://apcpartnercentral.com/assets/2012/07/CMRP-5T9PQG_R4_EN.pdf
 - [6] Global action plan. An inefficient truth [EB/OL]. [2011-02-12]. <http://globalactionplan.org.uk>
 - [7] 廖彬, 于炯, 张陶, 等. 基于分布式文件系统 HDFS 的节能算法研究 [J]. 计算机学报, 2013, 36(5): 1047-1064
Liao Bin, Yu Jiong, Zhang Tao, et al. Energy-Efficient Algorithms for Distributed File System HDFS [J]. Chinese Journal of Computers, 2013, 36(5): 1047-1064
 - [8] 廖彬, 于炯, 孙华, 等. 基于存储结构重配置的分布式存储系统节能算法 [J]. 计算机研究与发展, 2013, 50(1): 3-18
Liao Bin, Yu Jiong, Sun Hua, et al. Energy-Efficient Algorithms for Distributed Storage System Based on Data Storage Structure Reconfiguration [J]. Journal of Computer Research and Development, 2013, 50(1): 3-18
 - [9] 廖彬, 于炯, 张陶, 等. 一种适应节能的云存储系统元数据动态建模与管理方法 [J]. 小型微型计算机系统, 2013, 34(10): 2407-2412
Liao Bin, Yu Jiong, Zhang Tao, et al. Novel Energy-efficient Metadata Dynamic Modeling and Management Approach for Cloud Storage System [J]. Journal of Chinese Computer Systems, 2013, 34(10): 2407-2412
 - [10] Gray J. Tape is dead, disk is Tape, flash is disk, RAM locality is king [EB/OL]. http://www.signallake.com/innovation/Flash_is_Good.pdf
 - [11] Schall D, Hudlet V, Härder T. Enhancing energy efficiency of database applications using SSDs [C]// Proceedings of the Third C* Conference on Computer Science and Software Engineering. ACM, 2010: 1-9
 - [12] Lee S W, Moon B. Design of flash-based DBMS: an in-page logging approach [C]// Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data. ACM, 2007: 55-66
 - [13] Lee S W, Moon B, Park C, et al. A case for flash memory ssd in enterprise database applications [C]// Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data. ACM, 2008: 1075-1086
 - [14] Tsirogiannis D, Harizopoulos S, Shah M A. Analyzing the energy efficiency of a database server [C]// Proc. of SIGMOD '10 Indianapolis, IN, USA, 2010: 231-242
 - [15] 王江涛, 赖文豫, 孟小峰. 闪存数据库: 现状、技术与展望 [J]. 计算机学报, 2013, 36(8): 1549-1567
Wang Jiang-tao, Lai Wen-yu, Meng Xiao-feng. Flash-Based Database: Studies, Techniques and Forecasts [J]. Chinese Journal of Computers, 2013, 36(8): 1549-1567
 - [16] Xu Z. Building a power-aware database management system [C]// Proceedings of the Fourth SIGMOD PhD Workshop on Innovative Database Research. ACM, 2010: 1-6
 - [17] Xu Z, Tu Y C, Wang X. Exploring power-performance tradeoffs in database systems [C]// 2010 IEEE 26th International Conference on Data Engineering (ICDE). IEEE, 2010: 485-496
 - [18] Xu Z, Tu Y C, Wang X. PET: reducing database energy cost via query optimization [J]. Proceedings of the VLDB Endowment, 2012, 5(12): 1954-1957
 - [19] Rodriguez-Martinez M, valdivia H, Seguel J, et al. Estimating Power/Energy consumption in Database Servers [J]. Procedia Computer Science, 2011, 6: 112-117
 - [20] Times N Y. Power, Pollution and the Internet [EB/OL]. [2013-5-20]. <http://www.nytimes.com/2012/09/23/technology/data-centers-waste-vast-amounts-of-energy-belying-industry-image.html>
 - [21] Kansal A, Zhao F, Liu J, et al. Virtual machine power metering and provisioning [C]// Proceedings of the 1st ACM Symposium on Cloud Computing. Indianapolis, USA, 2010: 39-50
 - [22] Bao Y, Chen M, Ruan Y, et al. HMTT: A platform independent full-system memory trace monitoring system [J]. ACM SIGMETRICS Performance Evaluation Review, 2008, 36(1): 229-240