DOI: 10.3785/j.issn.1008-973X.2013.01.007

云计算环境下改进的能效度量模型

宋 杰, 侯泓颖, 王 智, 朱志良
(东北大学 软件学院, 辽宁 沈阳 110819)

摘 要: 针对大规模计算的能效问题,提出改进的能效度量模型,通过"能源"和"效率"2 种度量来综合评价系统能效.在"能源"方面,考虑计算机、网络和附属设备的能耗;在"效率"方面,考虑 CPU、内存、磁盘以及网络的情况.提出的能效度量模型描述了改进后的能效度量的定义和数学表达,通过实验验证了该模型的合理性.基于该度量模型,评估并分析了 MapReduce 环境中 CPU 密集型、I/O 密集型和交互型计算的能效,总结了 MapReduce 环境中的能效规律.

关键词: 云计算; 能效; 度量模型; MapReduce
中图分类号: TP 301.41 **文献标志码**: A **文章编号**: 1008-973X(2013)01-0044-09

Improved energy-efficiency measurement model for cloud computing

SONG Jie, HOU Hong-ying, WANG Zhi, ZHU Zhi-liang
(*Software College, Northeastern University, Shenyang* 110819, *China*)

Abstract: An energy efficiency measurement model was proposed to address the large-scale computing problem through two metrics: "energy" and "efficiency". As for "energy", the energy consumption of computer, network and affiliated equipments was considered; as for "efficiency", that of CPU, memory, disk and network was considered. The proposed energy efficiency measurement model describes the definition and mathematical expression of the improved energy efficiency measurement, and is proved reasonable through experiments. The energy efficiency of CPU intensive, I/O intensive and interactive computing was evaluated and analyzed based on the measurement model, and the energy efficiency laws in MapReduce environment were summarized.

Key words: cloud computing; energy efficiency; measurement model; MapReduce

近些年,云计算作为一种崭新的计算模式,受到产业界和学界的广泛关注,报告[1]估计云计算明年带来的收益将达到 1500 亿美元,然而巨大经济收益的背后是刻不容缓的能耗问题.云计算这种大规模的计算模式的能效(单位能源完成的计算)问题备受关注.

本文着重研究能效的度量模型,定义能效可以通过"能源"和"效率"2 种度量来综合评价,能效是单位时间内系统的"运算量"和"耗电量"的比值.本文提出的能效度量是对现有能效度量的改进,尤其是针对文献[2]的能效度量模型.笔者等[2]提出一种能效度量及其测量和计算方式,总结了云计算系统的能效规律.本文定义"能源"和"效率"这 2 个因素,由此得出云计算系统中更具体的能效规律.本文在研究背景中详细论述传统能效度量、文献[2]的能效度量以及改进的原因,描述改进的能效度量的定义和数学表达;基于改进度量模型,以 Hadoop 框架为例,评估 MapReduce 环境中 CPU 密集型、I/O 密集

收稿日期: 2012-07-10. 浙江大学学报(工学版)网址: www.journals.zju.edu.cn/eng
基金项目: 国家自然科学基金资助项目(61202088);辽宁省自然科学基金资助项目(200102059);中央高校基本科研业务费专项资金资助项目(N110417002).
作者简介: 宋杰(1980—),男,副教授,从事云计算、海量数据计算及高能效计算的研究. E-mail: songjie@mail.neu.edu.cn

型和交互型计算的能效，总结 MapReduce 环境中的能效规律和优化办法.

1　研究基础与相关工作

国内外现有研究在这一方面持有 2 种不同观点：Xu 等[3]认为能耗和性能是 2 种不同的优化目标，它们之间存在着某种折中关系，提高性能必然导致高能耗，相反，降低能耗必然损失性能；Tsirogiannis 等[4]认为能耗和性能的优化是一致的，最节能的系统往往是性能最好的系统. 概念上，前者符合硬件设计思路，运算越快越耗电；后者符合软件设计思路，任务完成越快，计算机工作时间越少，越省电. 这 2 种截然相反的观点的产生是因为采用不同的评价方法. 前者没有考虑空闲能耗和“能量按比率的计算”(energy proportional computing)[5]，后者没有考虑计算机功率的动态性. 但无论何种观点，能效(energy efficiency)都能很好地度量性能和能耗之间的关系，使两者统一.

能效为单位能量完成的有效任务，因此，能效的度量有 2 个部分. 一是单位时间内系统消耗的能量，大部分现有研究都采用“焦耳”来衡量电能的消耗，本研究将沿用这一单位. Kumar 等[6-9]提出了自己的能耗度量，但几乎都是简单的功率模型[7-9]，或是用任务的执行时间来度量性能[6]，没有给出通用的能效度量的定义以及详实的计算和测量方法. 包括笔者等[2]的研究工作，都仅考虑云计算系统中计算机的能耗，忽略了网络、空调等附属设备的能耗. 本文更细致地划分一个云计算系统的各个组成部分以及它们的能耗特点和度量方法.

能效度量的另一个关键部分是单位时间内系统完成的有效任务，如何度量“有效任务”是一个难题. Abdelsalam 等[10]都简单地设定一个任务，用任务的完成时间来代替任务量，这种度量不适合运行复杂任务的 MapReduce 系统，也不具有一般性. 若能够定义一种原子的任务单元，则可以以计数的方式衡量任务量. 事实上，任务单元可以根据算法设计，如排序算法，本文可以定义排序 1 000 条数据的任务量作为单位 1，或者在数据库系统中定义一个事务为一个任务单元，或是一次用户请求和响应作为任务单元. 笔者考虑用 Map 任务或者 Reduce 任务的个数作为任务单元，但该方法不具有通用性，且与算法的复杂度相关. 因此，需要找到一种较为底层的度量任务量的方式.

人们通常用浮点运算或单字长定点指令作为计算执行的最小任务单元，笔者等[2]指出这种度量任务量的方式的不足，设计了一种新的任务量单位，定义主频为 1 GHz 的 CPU 在 1 s 中的运算量作为 1 任务量单位，记为 1 U. 该任务计算单位的优点是非常容易测量，可以直接采用 CPU 频率与使用率的乘积来计算，即

有效任务量＝CPU 频率×CPU 使用率.

任务单位 U 的最大缺点是仅承认 CPU 的运算为系统完成的有效任务，忽略了磁盘读写运算、网络数据传输运算，任务单元的不公平性会导致 CPU 密集型计算的能效高于 I/O 密集型计算. 有效任务量不能用单因素来衡量，因此本文采用多种与硬件相关的、细粒度的性能数据来度量有效任务量，但不是在度量硬件能效而是在度量软件系统的能效. 在软件层面上，任务的类型多种多样，但最终对应到 CPU 频率和使用率、磁盘读写和网络收发数据量等硬件层面的测量值时，具有一致性. 集成这些测量值是一个难题. 首先，任务量不能重复地度量，若任务量包括了网络数据收发则不必再次包含路由器数据处理，因为它们都代表网络 I/O 运算；其次，一些硬件系统的作用主要是为了性能优化而非完成了软件系统定义的任务，因此不能简单地将所有耗能组件的运算都作为“有效任务”. 此外，每种运算度量量纲都不同，如 CPU 频率(GHz)、磁盘读写速率(kb/s)、网络读写速率(bit/s)，若将它们集成为任务量，则如何统一它们的量纲是一个难题.

在能效度量模型的基础上，对 MapReduce 系统进行能效测试，分析能效规律. 选择 MapReduce 是因为它是云计算环境中主流的编程模型，大多 Internet服务商的核心业务均采用 MapReduce 实现. MapReduce 有多种实现版本，选择 Hadoop 是因为它是开源的、并广泛地被开发者和研究人员接受. 本文采用 CPU 密集型、I/O 密集型和交互型计算作为测试用例，因为这 3 种计算是 MapReduce 应用领域中最常见的计算种类，均能够产生 CPU、磁盘和网络的混合计算任务.

2　能效度量模型

计算系统的能效是消耗单位能量完成的任务量，因此分别定义计算系统消耗的能量和完成的任务，称为“能耗”和“任务量”.

首先考虑“能耗”因素. 云计算系统的能耗包括 3 部分：计算机、网络和空调等辅助设备. 本文仅考虑计算机能效和网络设备的能耗、空调等辅助设备

的能效,因为这些设备不直接参与任务的执行,能耗很难通过“软件方法”优化.网络设备在计算中主要完成数据传输任务,路由器、交换机以及网线都会消耗能量,应该将系统能耗考虑进去.定义云计算系统的能耗为

$$E(T) = E_c(T) + E_n(T) + E_o(T). \quad (1)$$

式中:$E_c(T)$为计算机的能耗,$E_n(T)$为各种网络设备的能耗,$E_o(T)$为空调等附属设备的能耗.使用焦耳作为单位表示上述能耗,在下节将给出这些能耗的度量方法.

考虑“任务量”因素.本文设计的任务量度量是面向应用的,而非面向硬件,因为只用这种度量可以清晰地反映软件系统特征,使任务量成为一个整体,更有利于软件层的能效优化.采用底层硬件系统的性能参数来度量任务量,目的是为了使模型更通用,与具体算法无关,但度量结果能够反映算法特征.

任意一个计算本质上包含2个部分的任务:1)执行算法的运算任务;2)I/O任务.对于前者,均由CPU完成(不考虑GPU对媒体数据的处理),因此可以采用CPU的工作状态来度量;对于后者,可以分为3种I/O操作:内存I/O、本地磁盘I/O和网络数据I/O.

从算法角度考虑,读写内存的目的是为了提高性能,内存I/O次数会影响磁盘和网络I/O次数.一方面,内存是一个媒介设备.在操作系统的多级存储中,内存是为了调和外部存储与CPU在速度上的矛盾而存在的,被当作CPU和外部存储(包括磁盘和网络)的中间媒介.所有被读取的外部存储数据都需要经过内存,与作为媒介的各级缓存一样,该部分内存的读写不应该被算入有效任务量.另一方面,操作系统利用内存作了许多性能上的优化,对内存的读写需求不来自于软件系统或算法.在Linux系统中,内存分为实际内存、共享内存、Cache和Buffer 4部分.Buffer是将被写入磁盘的数据的缓冲区,Cache是已经读取的磁盘数据的缓存区.在使用过程中,只要还有空余内存,就会被Cache和Buffer充分利用.因此,系统对内存的读写不仅仅是实际内存和共享内存的读写,甚至与实际内存和共享内存的读写相差甚远,这造成了内存的不可测量性.磁盘和网络的读写真实地反映了一个任务输入的数据量、结果数据量以及计算过程中产生并存储在外部存储上的中间数据(在MapReduce应用中后者非常大).综上所述,本文没有将内存的读写量计入有效任务量.

理论上,所有输入数据都来自于磁盘,所有输出数据都会保存在磁盘中,因此磁盘I/O将计入有效任务.减少网络I/O不会影响本地I/O,但网络I/O是分布式计算的特点,很大程度上决定了计算的性能,而且网络I/O的大小与算法相关,因此将网络I/O计入有效的任务量;但一些参与数据传输设备的I/O操作,如总线和网络设备,不会影响其他I/O运算,因此不计入任务总量.用磁盘和网卡的工作状态来度量I/O任务.设$L(T)$是T时间内的任务量,

$$L(T)=L_c(T)\oplus_1[L_d(T)\oplus_2 L_n(T)]. \quad (2)$$

式中:$L_c(T)$为CPU的运算量,$L_d(T)$为磁盘读写运算量,$L_n(T)$为网络收发运算量,符号$\oplus_1$和$\oplus_2$表示聚集函数.研究$L_c(T)$、$L_d(T)$和$L_n(T)$的量纲统一方法.$L_c(T)$可以用“执行的浮点运算量”来度量.对于任意时刻t,可得$L_c(t)$的计算公式:

$$L_c(t) = C_f(t) \times C_u(t) \times C_c \times C_{fn}. \quad (3)$$

式中:$C_f(t)$为t时刻CPU的频率,$C_u(t)$为t时刻CPU的使用率,C_c为CPU核心数,C_{fn}为每周期浮点运算次数.CPU每次进行浮点运算的对象都是2个长度为机器字长的值.式(3)可以写成

$$L_c(t) = C_f(t) \times C_u(t) \times C_c \times C_{fn} \times 2C_{mw}. \quad (4)$$

式中:C_{mw}为机器字长.$L_c(T)$的单位为

$$1\ 024\ \text{M}\times\text{cycles/second}\times 1/\text{cycles}\times\text{bit}=$$
$$1\ 024\ \text{MB}/8=2^7\ \text{MB}.$$

于是,将CPU运算量的量纲统一到了MB.

当I/O操作发生的时候,CPU为了响应此类操作,需要执行相应指令,根据式(2)可知,这部分指令会计入运算任务量;但从算法角度考虑,运算任务量应该取决于执行算法的复杂度而非读取多少数据,因此需要排除此类指令,否则I/O操作将会被2次计算任务量.设磁盘读取1 MB数据时CPU对应会执行f_d MB的指令,网卡读取1 MB数据时CPU对应会执行f_n MB的指令,式(2)转换为

$$L(T) = [L_c(T) - f_d L_d(T) - f_n L_n(T)] \oplus_1 [L_d(T) \oplus_2 L_n(T)]. \quad (5)$$

考虑网络数据的发送和接收,对于整个云计算系统,从算法角度来说,通过网络接收的数据量和发送的数据量应该是一致的(实际测量也是近似的),某节点发送的数据必定会被一个或多个其他节点接收.在任务量计算时,仅计算发送或接收的数据量.

定义1 能效(energy efficiency) 在T时间内计算系统的能效$\eta(T)$定义为完成的任务量$L(T)$与消耗的能量$E(T)$的比值:

$$\eta(T) = \frac{L(T)}{E(T)} = \frac{L_c(T)\oplus_1[L_d(T)\oplus_2 L_n(T)]}{E_c(T)+E_n(T)+E_o(T)}; E(T)\neq 0. \quad (6)$$

3　能效度量方法

3.1　计算机能耗

对于云计算系统，设存在 N 个运算节点，记为 $c_i(1 \leqslant i \leqslant N)$，运算节点 c_i 的 CPU 在 t 时刻的功率为 $p_{c_i}(t)$，则

$$E_c(T) = \sum_{i=1}^{N} \int_0^T p_{c_i}(t)\mathrm{d}t. \tag{7}$$

测量 $E_c(T)$ 的方法有很多种. 所有计算机都有额定功率，但实际功率是动态变化的，因此很难找到 $p_{c_i}(t)$ 的数学表达式，并按照式(1)计算能耗 $E(T)$. 每隔 Δt 时间对节点进行一次有功功率的采样，从 $0 \sim T$ 时刻共采样 M 次. 由定积分的定义可知，

$$E_c(T) = \sum_{i=1}^{N} \int_0^T p_{c_i}(t)\mathrm{d}t \approx \sum_{i=1}^{N} \sum_{j=1}^{M} p_{c_i}(\Delta t \cdot j)\Delta t. \tag{8}$$

当 Δt 足够小时，式(8)的值即为 $E(T)$ 的值. 其中功率测量方法见文献[2].

为了简化表述，假设某一属性在 t 时刻的度量值为 $x(t)$，则

$$x(T) = \int_0^T x(t)\mathrm{d}t \approx \sum_{j=1}^{M} x(\Delta t \cdot j)\Delta t. \tag{9}$$

因此，式(8)可以简化记为

$$E_c(T) = \sum_{i=1}^{N} p_{c_i}(T). \tag{10}$$

3.2　网络设备能耗

$E_n(T)$ 包含 2 个部分：1)路由器或交换机节点的能耗 $E_{rou}(T)$；2)数据线路的能耗 $E_{cab}(T)$. 网络设备的能耗在负载变化时功率波动很小，可以认为是静态的. $E_{rou}(T)$ 和 $E_{cab}(T)$ 与是否有数据在网络中传输关系不大，因此采用功率与时间的乘积来计算能耗.

对于 $E_{rou}(T)$，使用有功功率而非视在功率或额定功率. 在交流系统中，视在功率包括电容和电感的电路元件引起的电流. 该循环功率流返回到电源，在每个周期结束时，导致净零能量转移. 这种类型的功率没有作任何有效功，不应包含在 $E_{rou}(T)$ 中. 有功功率度量的是实际做功的非零净功率. 对于云计算系统，设存在 R 个路由器，记为 $r_j(1 \leqslant j \leqslant R)$，$r_i$ 路由器的有功功率为 p_{r_i}，则

$$E_{rou}(T) = T\sum_{j=1}^{R} p_{r_j}. \tag{11}$$

网线内的电流为高频交流电，可以按照交流电的功率来计算 $E_{cab}(T)$. 以常用的 5 类 4 对非屏蔽双绞线为例，加载在双绞线两端的交流电压 U 有 -2.5、0 和 2.5 V 3 种，最高传输频率 f 为 100 MHz，此时阻抗 z 约为 22 Ω. 电流 $I = U/z \approx 0.11$ A，电路的功率因数 $\cos \Phi = R/z \approx 0.43$(100 m 时，$R = 9.38$ Ω). 由交流电功率的计算公式可知，电路的平均功率 $P = UI\cos \Phi = 0.11 \times 2.5 \times \cos \Phi \approx 0.12$ W. 可见，$E_{cab}(T)$ 非常小，相对于一个云计算系统可以忽略，因此认为 $E_{rou}(T) \approx E_n(T)$.

3.3　附属设备能耗 $E_o(T)$

一个云计算系统的附属设备主要包括空调、不间断电源等. 以空调为例，功率特性非常复杂. 空调功率与工作环境和工作状态相关. 计算机消耗的电能大部分都转换为热量散发出来，计算机能耗降低，空调的工作时间和功率都会减小，因此计算机的节能和空调设备的节能是一致的. 考虑模型的可测量性，近似地认为附属设备的能耗与时间成线性关系：

$$E_o(T) = P_o T. \tag{12}$$

式中：P_o 为功率常数.

3.4　运算量 $L_c(T)$、$L_d(T)$ 和 $L_n(T)$

介绍运算量 $L_c(T)$、$L_d(T)$ 和 $L_n(T)$ 的量纲统一和测量方法. $L_d(T)$ 和 $L_n(T)$ 的数量级一致，可以令 $A \oplus_2 B = A + B$. 当采用式(4)将 CPU 的频率单位 GHz 转换成存储大小单位 MB 时，转换算法使 $L_c(T)$ 远大于 $L_d(T) + L_n(T)$，$\oplus_1$ 若为简单的相加，则会导致 $L_c(T)$ 在 $L(T)$ 的计算中起决定作用，$L_d(T)$ 和 $L_n(T)$ 的影响被缩小. 因此，式(2)中的聚集运算不是简单的加法能够代替的. 为了使 A 与 B 保持在同一数量级，定义 $A \oplus_1 B = \Phi A + B$，其中 Φ 为调整系数. 式(2)可以改写为

$$L(T) = \Phi L_c(T) + [L_d(T) + L_n(T)]. \tag{13}$$

对于一个算法，“运算”和“I/O”是同等重要的. 定义 $\Phi = T_{IO}/T_{CPU}$. 其中，T_{IO} 表示 I/O 最大吞吐量，包括磁盘和网络 2 部分；T_{CPU} 表示 CPU 最大吞吐量. 对于不同的计算机，由于硬件不同，Φ 值也不同，保证了 $L_c(T)$ 与 $L_d(T) + L_n(T)$ 在 $L(T)$ 中所占的比重一致.

表 1 定义了 t 时刻第 i 个节点($1 \leqslant i \leqslant N$)的若干特征属性来计算运算量. 每隔 Δt 时间对每节点进行一次表 1 中特征属性的采样，在 $0 \sim T$ 时间内共采样 M 次. 其中，CPU 频率和使用率、磁盘 I/O 读写速率及网络通讯量分别代表了 CPU、磁盘以及网卡的实时运算量. 此外，表 1 定义了一些需要测量的静态参数.

基于上述测量值，定义 $L_c(T)$、$L_d(T)$ 和 $L_n(T)$ 的计算方法如下.

表 1 用于计算运算量的每计算机节点特征属性

Tab. 1 Properties of each node to compute workload

特征属性	单位	描述
$C_u^i(t)$	%	CPU 处理用户操作时间占 CPU 总时间的百分比
$C_f^i(t)$	GHz	CPU 工作频率
$D_r^i(t)$	MB	磁盘每秒读入数据量
$D_w^i(t)$	MB	磁盘每秒写入数据量
$N_r^i(t)$	MB	网卡每秒接收的数据量
$N_w^i(t)$	MB	网卡每秒发送的数据量
C_{dru}^i	%	当用户操作仅为磁盘读时,每读取 1 MB 数据时 CPU 的平均 C_u 值
C_{drf}^i	GHz	当用户操作仅为磁盘读时,每读取 1 MB 数据时 CPU 的平均 C_f 值
C_{dwu}^i	%	当用户操作仅为磁盘写时,每写 1 MB 数据时 CPU 的平均 C_u 值
C_{dwf}^i	GHz	当用户操作仅为磁盘写时,每写 1 MB 数据时 CPU 的平均 C_f 值
C_{nru}^i	%	当用户操作仅为网络数据读时,每读取 1 MB 数据时 CPU 的平均 C_u 值
C_{nrf}^i	GHz	当用户操作仅为网络数据读时,每读取 1 MB 数据时 CPU 的平均 C_f 值
C_{nwu}^i	%	当用户操作仅为网络数据写时,每写 1 MB 数据时 CPU 的平均 C_u 值
C_{nwf}^i	GHz	当用户操作仅为网络数据写时,每写 1 MB 数据时 CPU 的平均 C_f 值

$$\left.\begin{aligned} L_c(T) &= \sum_{i=1}^{N}[C_f^i(T)C_u^i(T) - C_{drf}^i C_{dru}^i D_r^i(T) - \\ &\quad C_{dwf}^i C_{dwu}^i D_w^i(T) - C_{nrf}^i C_{nru}^i N_r^i(T) - C_{nwf}^i C_{nwu}^i N_w^i(T)], \\ L_d(T) &= \sum_{i=1}^{N}[D_r^i(T) + D_w^i(T)], \\ L_n(T) &= \frac{1}{2}\sum_{i=1}^{N}[N_r^i(T) + N_w^i(T)] = \\ &\quad \sum_{i=1}^{N} N_r^i(T) = \sum_{i=1}^{N} N_w^i(T). \end{aligned}\right\} \tag{14}$$

3.5 能效度量公式

根据式(7)~(14),可以得到一个包含 N 个节点和 R 个路由器节点的云计算系统的能效度量:

$$\begin{aligned} \eta(T) &= \frac{L(T)}{E(T)} = \frac{\Phi L_c(T) + [L_d(T) + L_n(T)]}{E_c(T) + E_n(T) + E_o(T)} = \\ &\Big\{\sum_{i=1}^{N}\Phi^i[C_f^i(T)C_u^i(T) - C_{drf}^i C_{dru}^i D_r^i(T) - \\ &C_{dwf}^i C_{dwu}^i D_w^i(T) - C_{nrf}^i C_{nru}^i N_r^i(T) - \\ &C_{nwf}^i C_{nwu}^i N_w^i(T)]C_c C_{mw} + \sum_{i=1}^{N}[D_r^i(T) + D_w^i(T)] + \\ &\sum_{i=1}^{N} N_r^i(T)\Big\} \Big/ \Big[\sum_{i=1}^{N} p_{c_i}(T) + T\sum_{j=1}^{R} p_{r_j} + P_o T\Big]. \end{aligned} \tag{15}$$

4 实验验证

为了验证该能效度量模型的合理性以及对 MapReduce 环境中不同类型计算进行分析研究,选用 12 台 PC 机构建了基于 Hadoop HDFS 和 Hadoop MapReduce 的云计算环境. 实验环境参见文献[2].

计算能效值首先需要确定每个计算机节点对应的系数 Φ 和静态参数(见表 1). 由于集群是同构的,每个节点的 Φ 和特征属性相同. 实验环境中所用的网络带宽为 100 Mbit/s,理论最大峰值传输速率为 100 MB/8 = 12.5 MB/s. 硬盘接口类型为 Serial ATA 1.0a,数据传输率理论可以达到150 MB/s. 因此,T_{IO} = 162.5 MB/s. 由式(4)计算可得, $T_{CPU} = 2.8 \times 100\% \times 4 \times 4 \times 2 \times 64 \times 2^7$ MB/s = 734 003.2 MB/s (C_f = 2.80 GHz, C_u = 100%, C_c = 4, C_{fn} = 4, C_{mw} = 64 bit). 综上可知,$\Phi = 2.2 \times 10^{-4}$. 表 1 中的静态参数可以通过实验测得. C_{dwu}^i、C_{dwf}^i、C_{dru}^i和 C_{drf}^i是与磁盘相关的静态参数,在 Linux 系统中,/dev/null 等价于一个只写文件,所有写入它的内容都会永远丢失;/dev/zero 是一个伪文件,但它实际上产生连续不断的二进制的零流,主要的用处是用来创建一个指定长度用于初始化的空文件. 可以通过从/dev/zero 读取文件并重定向到一个新文件中来确定 C_{dwu}^i和 C_{dwf}^i,从一个大文件读数据并重定向到/dev/null 来确定 C_{dru}^i和 C_{drf}^i. C_{nru}^i、C_{nrf}^i、C_{nwu}^i和 C_{nwf}^i是与网络相关的静态参数,可以通过程序测定,选取一个节点作为服务器,另一节点作为客户端. 客户端生成数据向服务器端发送,服务器端从网络中读取数据并放入到缓冲区中,若缓冲区放满则覆盖掉原来的数据. C_{nwu}^i和 C_{nwf}^i可由客户端测得,C_{nru}^i和 C_{nrf}^i可由服务器端测得.

在云环境中对不同任务量的 MRBench(交互式任务,小作业高并发地执行)、WordCount(CPU 密集型计算)和 Sort(I/O 密集型计算)进行测试,研究能效规律. Map 和 Reduce 任务的完成时间可以由输出文件获取. 为了便于讨论和分析,在后文中使用 L_c 表示 $\Phi L_c(T)$,L_d 表示 $L_d(T)$,L_n 表示 $L_n(T)$.

作为 MapReduce 编程模型下的3 种典型应用,MRBench、WordCount 和 Sort 具有不同的 CPU 和 I/O 特性. 要研究 MapReduce 编程模型下应用的能效的特点,并说明模型的合理性,需要分别对这 3 种典型应用的任务量和能效进行分析. 实验 1 中,主要讨论 MRBench、WordCount 和 Sort 3 种应用的任务量特征,对比使用文献[2]的能效度量模型(下文

简称为 Original）与使用本文提出的改进的能效度量模型（下文简称为 Improved）的度量结果.

研究不同数据量 S 下 MRBench、WordCount 和 Sort 的任务量，实验结果如图 1 所示.

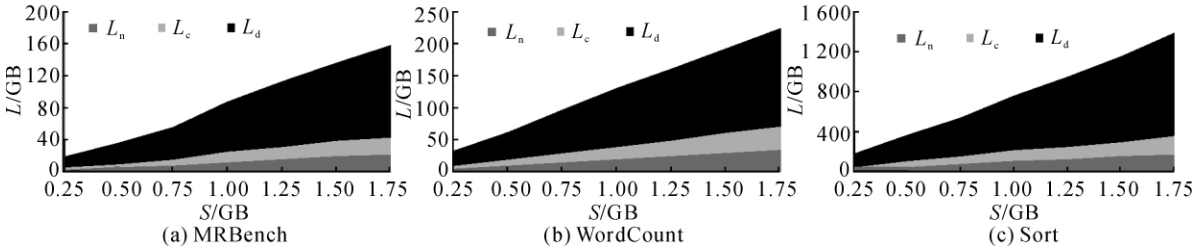

图 1　不同数据量下 MRBench、WordCount 和 Sort 的任务量

Fig. 1　Workloads of MRBench, WordCount and Sort under different amounts of data

如图 1 所示为任务量累积图. 由式(13)可知，总任务量由 L_n、L_c 和 L_d 3 部分构成. 可以看出，对于 3 种计算，L_d 均明显高于 L_n 和 L_c，随着数据量的增加，三者都增长，但 L_d 的增长速度很快，相比之下，L_n 和 L_c 的增长较缓慢.

由图 1 可以总结出 3 种类型计算任务量的基本特征如下. 1）MapReduce 应用本质上是 I/O 密集型计算，I/O 操作很多. 2）无论是交互型计算（MRBench）、CPU 密集型计算（WordCount）还是 I/O 密集型计算（Sort），L_d 与 L_n 之和都大于 L_c. 改进的能效模型使用 I/O 设备和 CPU 的最大数据吞吐量之比作为调整系数，所以任务量的差距是设备利用率的差距. 这 3 种 MapReduce 应用没有充分利用 CPU 资源，该实验结论与文献[2]的实验结果保持一致. 3）WordCount 是 CPU 密集型计算，Sort 是 I/O密集型计算，不是因为处理相同数据量的数据时，一个 CPU 运算量较大，另一个 I/O 运算量较大，而是因为 WordCount 的 I/O 运算相对于 CPU 运算少，Sort 的 I/O 运算远多于 CPU 运算.

从算法角度进一步分析图 1 可知：1）在相同数据量的前提下，Sort 的 CPU 运算量大于 WordCount 的 CPU 运算量. 因为 Sort 的 CPU 运算体现在 Map 端的 Hash 运算和 Reduce 端的排序过程，WordCount 的 CPU 运算体现在 Map 端的 Hash 运算和 Reduce 端的累加过程，排序算法比累加算法复杂. 2）在相同数据量的前提下，Sort 的 I/O 操作大于 WordCount 的 I/O 操作. 这是由于 Sort 在 Reduce 端对数据进行排序时，对数据的读操作非常多，并且 Sort 的输出与输入数据的规模相同. WordCount 在 Map 端将相同的 key 合并，在 Reduce 端只须对 value 值进行累加，输出数据只是单词和单词出现的次数，输出数据量显然少于 Sort. 如图 2 所示为不同数据量下 3 种类型计算的 Original 能效 η_{or} 和 Improved 能效 η_{im} 的对比. 可以看出，

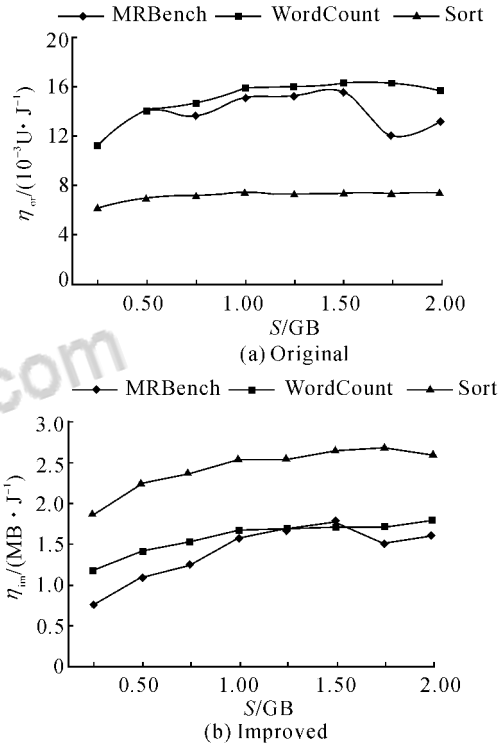

图 2　不同数据量下 MRBench、Sort 和 WordCount 的集群能效

Fig. 2　Cluster energy efficiency curves of MRBench, Sort and WordCount under different amounts of data

无论是 MRBench、WordCount 还是 Sort，随着数据量的增大，Original 能效和 Improved 能效都增加并趋于平稳，其中 Original 能效曲线比 Improved 能效曲线更平缓. 尽管 Original 能效和 Improved 能效的单位和数值不同，但 3 种计算中，Sort 的能效在 Original度量下是最低的（与文献[2]一致），而在 Improved度量下变成最高的，其他 2 种运算相差不大. 对图 2 进行分析，可以得到如下结论.

1）Sort 是一种 I/O 密集的运算，Original 能效度量只考虑 CPU 的计算量，因此 Sort 的 Original 能效低于其他运算，而 Improved 能效综合考虑了

CPU 的运算量和 I/O 运算量. 从图 1、3 可以看出，Sort 的 I/O 运算量非常高，因此 Sort 的 Improved 能效高于其他 2 种运算，这说明 Improved 能效度量更加符合任务的特点，而 Original 能效不能真实地反映单位能量下实际完成的任务量.

2) 在数据量较小的情况下，总任务量不饱满，CPU 运算量和 I/O 运算量都较小，导致 CPU 与I/O 在一定程度上空闲，因此能效较低. 随着数据量的上升，CPU 与I/O的利用率都有一定程度的提高，使得能效稍有增大，这一特点对于 Original 能效和 Improved 能效均适用.

3) 当任务量饱满时，资源利用率高低是决定 Improved 能效的关键因素，空闲的资源没有完成运算，但耗费能量. 比较了 3 种计算的集群 CPU 平均使用率 u、CPU 频率 f、平均磁盘读写速度 v_d 和平均网络传输速度 v_n，如图 3 所示. 图中，M 表示 MRBench，W 表示 WordCount，S 表示 Sort，1 代表 Sort 的 Map 阶段(55 s 前)，2 代表 Sort 的 Reduce 阶段(55 s 后)，所有值为按时间的平均值.

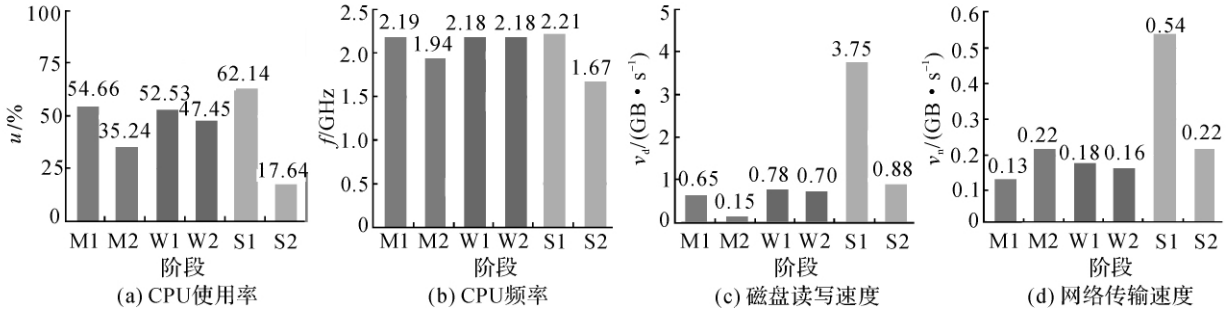

图 3 资源使用率对比

Fig. 3 Comparison of resource utilization

图 3 的数据显示了 Map 和 Reduce 2 个阶段的资源平均使用情况，由后续试验的图 6 可以看出，Sort 可以明显分为 Map 阶段和 Reduce 阶段. 2 个阶段的资源使用情况不同，若在总体时间上求平均值则难以体现出差异，因此用阶段 1 表示 Sort 的 Map 节点(55 s 前)，阶段 2 表示 Sort 的 Reduce 阶段(55 s 后). 从图 3 可以看出，在阶段 1，Sort 的 CPU 平均使用率、CPU 频率、平均磁盘读写和平均网络传输速度都高于 WordCount 和 MRBench，即资源的使用率很高，尤其是磁盘读写和网络传输；在阶段 2，虽然 Sort 的 CPU 频率和 CPU 使用率低于 MRBench 和 WordCount，平均磁盘读写和平均网络传输速度与 MRBench 和 WordCount 近似相等. 进一步分析数据可以得出，Sort 的 Improved 能效在阶段 1 明显高于 WordCount 和 MRBench，在阶段 2 与 WordCount 和 MRBench 近似相等，这一数据可以参考图 4(c)、5(c)和 6(c)得出. 因此，Sort 中占主导地位的资源是 I/O 资源，总体资源使用率比 MRBench 和 WordCount 高. 结合图 2 可知，应用的资源使用率越高，Improved 能效越高.

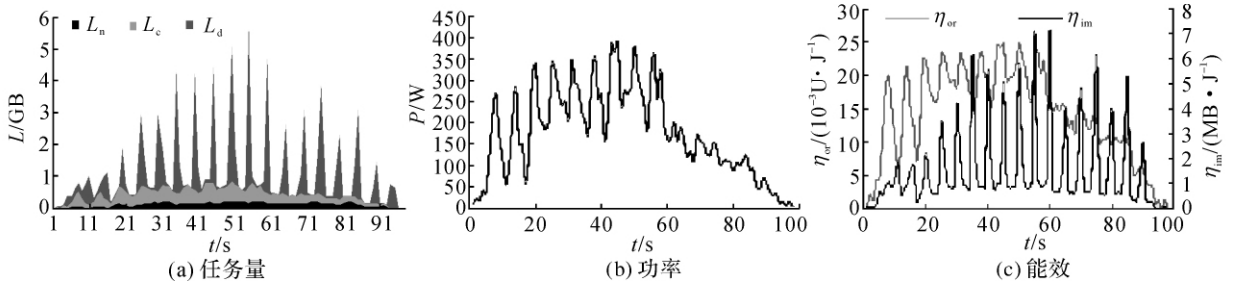

图 4 10 并发作业下每节点 1 GB 数据的 MRBench 计算的任务量、功率、能效曲线

Fig. 4 Workload, power and energy efficiency curves of MRBench with 1 GB data per node and 10 concurrent jobs

当负载饱满时，交互型计算 MRBench 向 Hadoop 集群提交多个并发的小任务，Hadoop 中常见的任务调度策略有 3 种，分别为：默认的调度器(FIFO scheduler)；基于容量的任务调度器——计算能力调度器(capacity scheduler)；多用户、多类型作业的任务调度器——公平调度器(fair scheduler). 本文采用默认的调度器，没有考虑能效和资源利用率问题. 在多个小任务并发工作的情况下存在资源的争夺和等待，每一个小任务不能充分地利用资源，因此，MRBench 的 Improved 能效不高.

Improved 度量比 Original 度量更精细地描述了能效，Improved 能够反映出 I/O 对能效的影响，而 Original 不能. 由图 4～6 可知，计算机功率（能耗）与 CPU 使用率是正相关的，与磁盘使用率的相关度很小，这一点在文献[2]中也得到证实. 随着任务量的增加，Original 能效中 CPU 使用率提高（Original能效公式的分子增大），能耗对应增加（Original能效公式的分母增大），两者相互抵消（比值略有增大），能效曲线平缓；而 Improved 的能耗与 Original的能耗（能效公式的分母）一致，但Improved 能效公式的分子部分包含 I/O 运算量. 当总任务量增加时，Improved 能效公式的分子比 Original 增加得更快，所以 Improved 能效曲线上升明显. 可见，本文提出的能效度量比文献[2]更加符合 MapReduce 应用的特点.

综上所述，影响能效的关键因素是资源的使用率，当资源空闲时，耗费能量没有完成运算，能效会降低. 导致资源使用率低的原因之一是任务量不饱满，但主要原因是资源瓶颈，即资源间等待，此外在多任务环境下资源分配和任务调度算法会影响资源使用率. 本文提出的能效度量较传统能效模型能够更加全面地定义能效，符合 MapReduce 应用的特点.

1）实时能效分析. 能效优化是一个实时的过程，因此需要研究 3 种计算的能效实时规律以及能效、任务量和功率之间的关系，比较 Original 能效度量和 Improved 能效度量的差异. 为了达到上述目的，针对 3 种计算的集群实时任务量、实时功率和实时能效进行分析.

如图 4(a)所示为 MRBench 的任务量累积图，可以看到由任务量波动产生的、较规则的波峰. 如图 4(b)所示为 MRBench 的功率 P 曲线图，功率在 60 s前较高，之后开始下降. 如图 4(c)所示为能效曲线图，由较规则的波峰构成的曲线是 Improved 曲线，总体上与任务曲线相似；另一条为 Original 曲线，总体上与功率曲线相似.

由图 4 可以得出如下结论：1）功率变化和 CPU 运算量大致保持一致，而非和总任务量一致，说明 CPU 是主要的耗能设备，是功率的主要决定因素，这一点符合计算机的硬件特点. 2）任务量的波峰波谷没有影响到功率和 Original 能效，但明显影响了 Improved 能效，使 Improved 能效呈现较规则的波峰. Original 能效的波动仅与功率的波动一致，而 Improved 能效的波动是总任务量和功率波动的叠加，最明显的是任务开始的前 11 s，任务量不大，功率却很高，此时 Original 能效为较高值，而 Improved 能效为较低值，因此，后者比前者更能够反映实际负载.

采用图 4 所示的方式分析 WordCount 的实时能效，结果如图 5 所示. 图 5(c)中 Original 能效与 Improved 能效走势相似，这是因为 CPU 的运算量和总任务量的走势相似，而功率曲线实际上和 CPU 运算量走势相似. Original 能效更多地与功率相关，Improved 能效更多地与任务相关，图 5 的结论与图 4一致. 此外，从图 5(c)不能看出 WordCount 的 Map 阶段能效和 Reduce 阶段的能效没有明显的差异，因为节点会同时执行 Map 任务和 Reduce 任务，Map 和 Reduce 2 个阶段无法明确地划分. 分析 Map 阶段执行读取数据的任务和中间数据的写入，Reduce 阶段执行统计运算，Reduce 阶段写入的数据量很小几乎可以不计，WordCount 的 Map 阶段任务量大于 Reduce 阶段，根据任务量和能效之间的关系可知，Map 阶段的能效应该大于 Reduce.

采用图 4 所示的方式分析 Sort 的实时能效，结果如图 6 所示. 从图 6(a)可以看出，前期任务量较高，在 55 s 以后降低且较平稳；图 5(b)中功率是前期较高，后来降低至平稳状态；图 5(c)中 Original 能效和 Improved 能效走势相似，都是由高到低，图 6 的结论与图 4、5 一致. 此外，由于 Sort 的计算执行

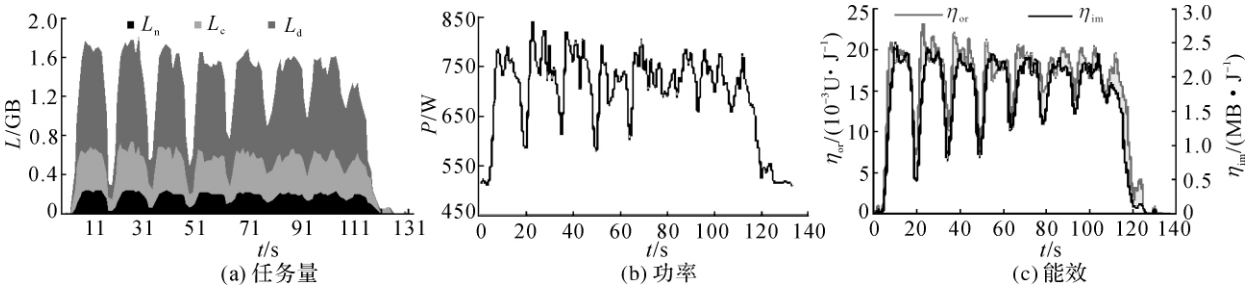

图 5　每节点 1 GB 数据的 WordCount 计算的任务量、功率、能效曲线

Fig. 5　Workload, power and energy efficiency curves of WordCount with 1 GB data per node

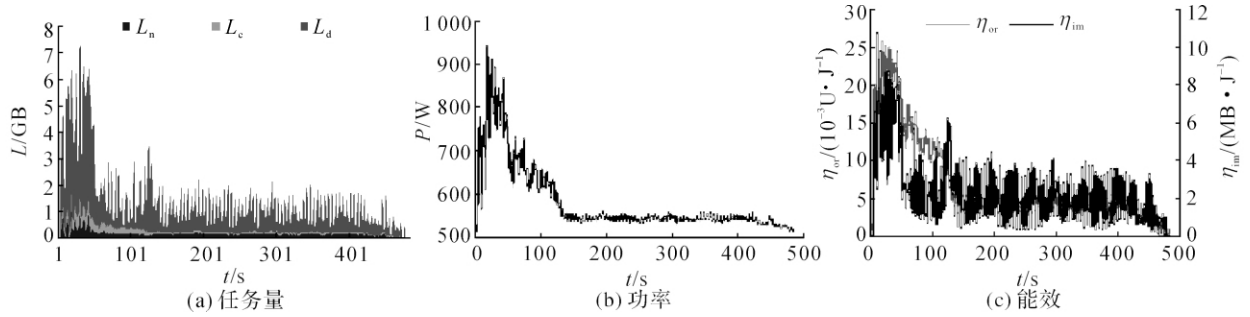

图6 每节点1 GB数据的Sort计算的任务量、功率、能效曲线

Fig. 6 Workload, power and energy efficiency curves of Sort with 1 GB data per node

时间较长,且Map阶段和Reduce阶段划分清晰,可以得出2个阶段的能效特点,Improved能效在Map阶段显著高于Reduce阶段、Map阶段的I/O操作,主要为磁盘读取,因此I/O操作比较密集. Reduce阶段的I/O也较高,但是由于传输速度受到网络带宽限制,使其延续时间较长,单位时间的I/O运算量不大. 因此,Improved能效在Map阶段是最高的,第一个Reduce启动后开始缓慢地降低.

综上所述,在一个MapReduce作业的执行过程中,本文提出的能效度量较传统模型能够更加准确地定义能效. Original能效的任务量仅与CPU运算量相关,计算机功率主要取决于CPU的工作状态,因此Original能效与功率较一致. Improved能效首先承认了计算机功率的动态性,体现了计算机功率会随CPU组件的工作状态变化,但Improved能效强调了特定功率下完成的任务量,这种与任务量相关的能效度量更符合软件层面的能效优化,可以通过调整任务量、优化任务调度来优化能效. 此外,在MapReduce作业Improved模型下,不能够简单地认为Map阶段的能效高于Reduce阶段,因为MapReduce应用的I/O运算量和CPU运算量在Map端和Reduce端的分布,会随着应用的改变而改变,但根据MapReduce的执行特点,可以很好地分析和比较2个阶段的能效,并区别对待,以实现细粒度的能效优化.

5 结 语

本文提出一种改进的云计算环境下的能效度量模型. 该度量模型不仅能够从CPU、磁盘和网络3个方面综合评价云计算集群的效率,而且可以分析计算机能耗、网络能耗和附属设备能耗. 首先,提出比现有模型更加全面的能效度量模型;其次,分析了不同类型的MapReduce应用的能效特征. 所有这些方法和实现都将对云环境下高能效计算这一研究领域作出一定的贡献,同时对能效模型和能效优化方法以及云计算环境下的应用起到了指导作用.

参考文献(References):

[1] Gartner Incorporation. Gartner says worldwide cloud services revenue will grow 21.3 percent in 2009 [EB/OL]. [2012-02-01]. http://www.gartner.com/it/page.jsp?id=920712.

[2] 宋杰, 李甜甜, 闫振兴,等. 一种云计算环境下的能效模型和度量方法[J]. 软件学报, 2012, 23(2): 200-214.
SONG Jie, LI Tian-tian, YAN Zhen-xing, et al. Energy-efficiency model and measuring approach for cloud computing [J]. **Journal of Software**, 2012, 23(2): 200-214.

[3] XU Zi-chen, TU Yi-Cheng, WANG Xiao-rui. Exploring power-performance tradeoffs in database systems [C]// **ICDE 2010**. California: IEEE, 2010: 485-496.

[4] TSIROGIANNIS D, HARIZOPOULOS S, SHAH M A. Analyzing the energy efficiency of a database server [C]// **SIGMOD 2010**. Indianapolis, Indiana, USA: ACM, 2010: 231-242.

[5] BARROSO L A, HÖLZLE U. The case for energy-proportional computing computer [J]. **IEEE Computer**, 2007, 40(12): 33-37.

[6] KUMAR K, LU Y H. Cloud computing for mobile users: can offloading computation save energy?[J]. **IEEE Computer**, 2010, 43(4): 51-56.

[7] ORGERIE A C, EVRE L L, GELAS J P. Save Watts in your grid: green strategies for energy-aware framework in large scale distributed systems [C]// **ICPADS 2008**. Melbourne: IEEE, 2008: 171-178.

[8] YOUNGE A J, VON LASZEWSKI G, WANG L, et al. Efficient resource management for cloud computing environments [C]// **International Green Computing Conference**. Chicago: IEEE, 2010: 357-364.

[9] SRIKANTAIAH S, KANSAL A, ZHAO F. Energy aware consolidation for cloud computing [C]// **Conference on Power Aware Computing and Systems**. Berkeley: USENIX Association, 2008: 10.

[10] ABDELSALAM H S, MALY K, MUKKAMALA R, et al. Analysis of energy efficiency in clouds [C]// **Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns, Computation World**. Norfolk: IEEE, 2009: 416-421.