



截止期限约束的实例密集型云服务流调度算法^{*}

刘 炜,李陶深,黄汝维

(广西大学计算机与电子信息学院 南宁 530004)

摘 要:针对商业云计算中存在大量实例密集型服务流的问题,提出一种新的云环境下两阶段服务流调度算法。该算法先将用户自定义的全局截止期限分配到系统中的每个实例,再将每个实例的截止期限分配到实例中的每个任务中,最后在服务流执行阶段,动态调整后续任务的截止期限,解决了任务可能存在的未能在其截止期限内完成的时间异常问题。CloudSim 仿真结果表明,与现有的算法相比,该算法能满足用户定义的截止期限,节约了执行成本,并减少了资源的竞争率,提高了调度的成功率。

关键词:云计算;服务流调度;时间异常;截止期限

doi: 10.3969/j.issn.1000-0801.2013.12.014

A Deadline-Constrained Scheduling Algorithm for Intensive-Instance Serviceflows in Cloud Environment

Liu Wei, Li Taoshen, Huang Ruwei

(School of Computer, Electronics and Information, Guangxi University, Nanning 530004, China)

Abstract: The problem that there are many intensive-instance serviceflows in business cloud computing environment was aimed to solved, so a new two-phase serviceflows scheduling algorithm was proposed. In the first phase, the global deadline to each instance was assigned, and then the deadline to the each task of instances was assigned. In the serviceflows execution phase, in order to solve the possible problem of time exception which task was not meet its sub-deadline, the algorithm adjusted deadlines of subsequent task dynamically. The experimental results in CloudSim show that the algorithm compared with others algorithm saves the execution cost with meeting the user-defined deadline, avoids the fierce competition of cheaper services and improves the success ratio of the scheduling.

Key words: cloud computing, serviceflows scheduling, time exception, deadline

1 引言

云计算作为一种计算模式,同时也是一种商业模式,它不仅要考虑用户的服务质量(quality of service, QoS)要求,而且需要满足一定的服务等级协议(service level agreement, SLA)。在“Pay-as-Use”模式下,用户根据实际需要申请服务,云计算服务提供商根据用户申请的服务进行

费用收取,所以用户会尽量将费用支出控制在一个预算的范围内。对于某些实际应用环境,用户对于执行时间没有严格的要求,因此与执行费用相比,执行时间并不是影响服务流调度的主要因素。如何在给定的时间内完成用户提交的任务,并达到执行费用的最小化是一个常见的 NP 难的问题^[1-4]。

Yang Y^[5,6]等人通过对一些电子商务系统以及电子政

^{*} 国家自然科学基金资助项目(No.60973074),广西自然科学基金资助项目(No.2013GXNSFBA019281)



务系统的研究,发现这些系统中存在一种新型的工作流,并把它命名为实例密集型工作流。实例密集型工作流应用是指在系统中同时存在大量并行实例,这些实例对于资源竞争非激烈,但每个实例本身并不复杂,通常由几个相对简单的任务构成,例如电子订票系统,每个事务都是一个简单工作流实例。云计算技术的发展为这些实例密集型商业服务流系统提供了良好的技术支持。

相对于科学计算工作流系统,实例密集型工作流的研究还比较少,参考文献[7]提出了 Deadline-MDP 算法,首先将流程任务划分为简单任务与同步任务两类,将用户指定的全局截止时间进行分解,再将工作流任务按照不同规则进行分类调度。该算法没有具体给出截止日期的分配方式,对于并行的分支设置相同的截止日期,这使得在云环境下会产生激烈的资源竞争。Liu K^[8]提出了基于云环境下的 QoS 限制的 CTC (compromised-time-cost) 调度算法并在 SwinDeW-C 平台里获得了较好的性能。CTC 算法考虑成本和执行时间因素,并采用轮转循环调度策略控制工作流应用执行成本与执行时间。参考文献[9]针对实例密集型工作流调度问题提出了基于预演算的调度算法(PCS A),最大特性是能够在调度前对任务的紧迫程度进行排序,得到每个任务的优先级序列,按照优先级序列高低,选择空闲计算机资源来执行,从而使执行结果最近最优。晏婧、吴开贵^[10]提出了适用实例密集型云工作流调度的 MCUD (minimum total cost under user-designated total deadline) 调度算法,该算法对需要处理的工作流实例分为不同类别,然后采用新的方法把用户指定的总截止期限划分为该类工作流实例各自任务的子期限。

传统的截止日期约束的工作流调度算法,大多只支持单个工作流的调度,并且用于科学计算等计算密集型工作流系统中,对于云环境下存在大量并行实例密集型工作流的问题,这些算法往往迭代次数过高,CTC 算法虽然能够使得实例密集型工作流系统有较好的性能,但是它处理的服务流实例都是流管(stream-pipe)模式,但是实际的单个服务流实例结构可能更为复杂,服务流实例会有并行分支并且往往会存在有限延迟时间异常的情况。有限延迟异常指的是工作流执行期间由于系统内部或外部产生的错误或者意外引起的工作流,整个流程不能满足其期望的时间限制^[11,12],例如由于工作流引擎的调度错误使得任务不能满足其在截止期限内完成工作。

针对这些问题,研究了 CTC 算法的不足和现有的工作

流系统未能解决工作流执行期间有限延迟异常的问题,本文提出了一种两阶段实例密集型工作流调度策略,第一阶段将用户全局的截止期限分解到每个实例上,第二阶段再将每个实例的截止期限分配给实例中的任务。服务流的执行阶段,根据任务的实际完成时间,动态更新后续任务的截止期限,若任务的截止期限小于任务的完成时间,则增加后续的截止期限,用来避免服务流调度中可能存在有限时间异常情况。

2 问题的描述

定义 1 单个服务流 $SF=(T, E)$, 其中 T 为服务流中的任务的集合 $T=\{t_1, t_2, \dots, t_n\}$; E 表示任务之间时序关系的集合 $E=(t_c, t_p)$, t_c 为子任务, t_p 为父任务, 当父任务完成后子任务才可以执行; 对于 SF 中的任务 t_i , 服务空间 S 中至少存在一个对应的候选服务资源, 记为 $s(j) \in \{s_1, s_2, \dots, s_m\}$ 。假定服务执行过程中服务质量状态不变, 那当服务与任务绑定后, 其执行时间与执行费用便确定, 执行费用为服务流中所有任务的费用之和。

定义 2 任务 t_i 在资源 s_j 预计执行时间 $ET(t_i, s_j)$, 任务预计执行花费 $EC(t_i, s_j)$, 任务 t_i 在可选的 m 个资源上平均执行时间计算方法如式(1), 每个服务流实例的平均执行时间 AET 等于每个实例关键路径的上任务的平均执行时间之和:

$$AET(t_i) = \frac{\sum_{j=1}^m ET(t_i, s_j)}{m} \quad (1)$$

服务流调度的目标是: 当前系统中所有服务流满足全局截止期限下优化执行代价, 根据本文算法, 将服务流全局的截止期限分配到每个任务中, 变为任务的局部优化问题, 得到调度目标是在满足任务时序关系和任务子截止期限的约束条件下, 选择执行费用可能少的资源进行服务, 问题的描述如下。

优化目标为:

$$\sum_{i=1}^n \min \left\{ \sum_{j=1}^m x_j^i c_j^i \right\} \quad (2)$$

约束条件为:

$$s.t. AST(t_i) \geq \max_{t_p \in parent^i t_i} \{AFT(t_p)\} \quad (3)$$

$$AST(t_i) + ET(t_i, s) \leq DT(t_i) \quad (4)$$

$$\sum_{j=1}^m x_j^i = 1, x_j^i \in (0, 1) \quad (5)$$

式(2)为工作流调度的优化目标, c_j^i 为任务 t_i 在 s_j 上

的执行成本;式(3)约束任务的时序关系,任务的实际开始时间 $AST(t_i)$ 大于或等于其父任务最大的实际完成时间 $AFT(t_p)$;式(4)约束每个任务选择服务资源必须满足截止期限的约束条件;式(5)中 x_j^i 为布尔变量,当任务 t_i 选择服务 s_j 时为 1,否则为 0。

3 截止期限约束的实例密集型云服务流调度算法

3.1 算法流程描述

假设 D 为用户定义的整个服务流的完成时间,任务 $t_{k,i}$ 为第 k 个实例的第 i 个任务,其截止期限 $TD(t_{k,i})$ 为完成该任务的最晚完成时间。则本文算法的描述如下。

输入:提交一组服务流实例,包括实例的数量 $InsNum$,用户自定义的截止期限 D 。

输出:调度结果。

步骤 1 将全局的截止期限分配到每一个实例。

步骤 2 将每个实例的截止期限分配到实例中的每个任务。

步骤 3 针对每一个就绪的任务(前驱任务执行成功),计算出任务在每一个可用资源上的执行时间和执行费用。为每一个任务选取最佳的服务资源进行服务(这里是指满足截止期限下,执行花费最小的资源;若所有资源都不能满足截止期限,则分配当前性能最优资源,此时出现有限延迟异常)。

步骤 4 执行就绪队列中的任务,计算出最后的完成时间与截止期限的差值 γ ,根据差值动态更新后续任务的截止期限(TD)。返回步骤 3。

3.2 分配截止期限到每个实例

本文算法参考 CTC 算法中分解全局的截止期限到每个实例中的方法,该方法错开了实例中任务的截止期限,相当于具有较长任务期限的任务有机会获取较短任务期限释放的便宜资源,从而减少对便宜资源的竞争。根据图1所示的实例最后期限的划分方式,使用式(6)计算各实例中任务的截止期限。

$$AET + (InsNum - 1) \times OT = D \quad (6)$$

其中, AET 是每个实例的平均执行时间, OT 为实例和实例之间可重叠的时间,系统中同时存在 $InsNum$ 个实例,用户自定义的全局截止期限为 D 。计算出 OT 后,可以得到每个实例最后一个任务的截止期限,进而分配实例中每个任务的截止期限。

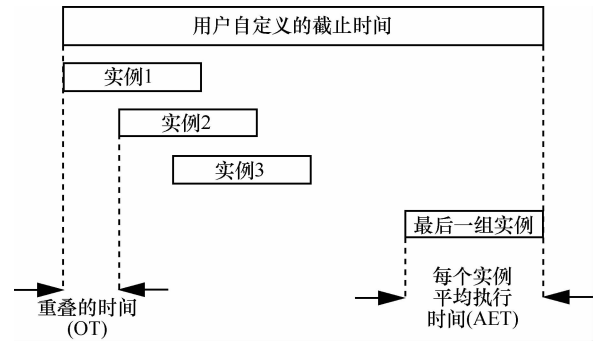


图1 实例的最后期限划分

3.3 分配每个任务的截止期限

每一个实例都是单个的服务流(SF),对于单个服务流,本文算法逆向搜索的方式,分配任务的截止期限。操作的步骤如下。

步骤 1 利用求解关键路径技术得到服务流的关键路径。

步骤 2 采用现有的性能估计技术预测出关键路径的平均执行时间,等于该路径上各个任务平均执行时间之和,关键路径的平均执行时间为该实例的平均执行时间(AET)。

步骤 3 采用逆向搜索技术从结束任务搜索服务流,根据式(7)分配截止期限,其中 k 为 $InsNum$ 个实例中的第 k 个; $t_{InsNum,c}$ 为任务 $t_{InsNum,i}$ 的下一个任务, n 为每个实例中包含的任务数。

$$\begin{cases} TD(t_{InsNum,n}) = D \\ TD(t_{InsNum,i}) = \min \left\{ TD(t_{InsNum,c}) - AET(t_{InsNum,c}) \right\} \\ TD(t_{k,i}) = TD(t_{InsNum,i}) - (InsNum - k) \times OT \end{cases} \quad (7)$$

3.4 动态调整后续任务的截止期限

服务流执行阶段,计算出任务实际完成的时间与其截止期限的差值,有助于动态调整后续任务可调度执行时间的范围,分两种情况:若 $\gamma > 0$,则表示任务在预定的截止期限内完成,调度成功;若 $\gamma < 0$,则表示任务未能在截止期限内完成,出现执行期间有限延迟异常,此时根据式(8)动态增加该路径上的后续任务的截止期限,保证整个服务流能够在截止期限内完成。

$$TD(t_{k,i}) = TD(t_{k,i}) + \gamma \quad (8)$$

其中, γ 为任务实际完成时间与截止期限的差值。

4 实验仿真及性能评价

4.1 实验数据准备

实验的硬件环境为 Intel Core 3.30 GHz CPU、4 GB 内



存、500 GB 的硬盘;软件环境为 Windows 7 操作系统、Eclipse 3.2 和 Java 1.7.0 语言;云仿真软件 CloudSim 模拟仿真现实云计算环境,实验主要包括以下两方面的模拟。

(1) 服务流实例的模拟

为了模拟普遍情况下的实例密集型服务流,实验中生成 100 个服务流,每个服务流结构如图 2 所示,用来模拟实际应用中实例密集型服务流。每个任务所包含的计算指令数如表 1 所示。

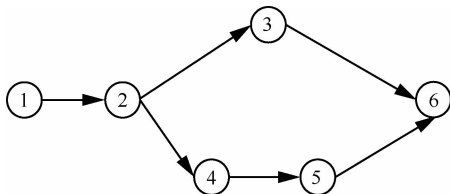


图 2 单个服务流结构

表 1 任务的计算量(单位 MI)

任务 1	任务 2	任务 3	任务 4	任务 5	任务 6
19 800	27 400	15 800	34 700	14 700	15 700

(2) 服务资源的模拟

本文采用亚马逊的云系统的定价模式——分级定价(tiered pricing),即云服务集中等级的方式提供,并且每一级提供固定的计算规格(如内存分配、CPU 类型和速度等),实验中本文模拟了 4 种类型服务资源,如表 2 所示,每一种类型服务资源有 4 个提供者,在实验模拟为不同处理能力的虚拟机。对于某一个任务而言,完成时间与执行价格成正比。

表 2 服务执行速率和执行一个任务的花费

服务类型	执行指令速度(MIPS)	花费(美元/任务)
1	250	3
2	500	6
3	1 000	12
4	2 000	24

4.2 实验结果与性能分析

实例密集型服务流作为商业云计算中的一种应用,实例的平均执行代价是考虑的一种重要因素,要求满足用户自定义的截止期限的情况下,实例的平均执行代价越小越好。CTC 算法是目前已验证的在实例密集型工作流环境下较为有效的方法。本文分析研究了 CTC 算法存在不能调度服务流中的非管流模式的任务这一问题,从而提出本文算法解决非管流模式调度问题,并且在调度过程中动态

调整任务的截止期限。本文的截止期限为实际的标准时间(1 000 s、1 500 s、2 000 s、2 500 s、3 000 s、3 500 s、4 000 s、4 500 s),仿真实验表明,本文算法执行成本略低于 CTC 算法,但与 Deadline-MDP 算法比较,可以在满足用户定义的截止期限的同时,更加优化执行代价。执行速率与执行花费呈线性增长,如图 3 所示,所以截止时间增大时,执行代价优化程度呈递减趋势。

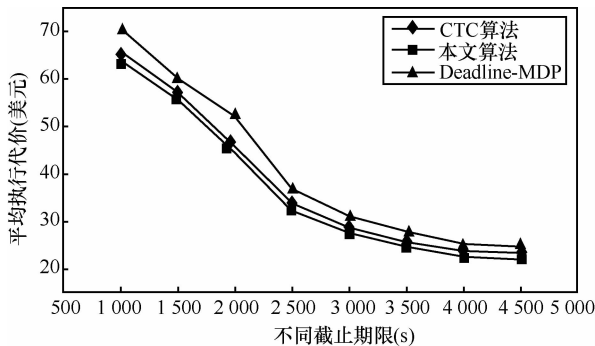


图 3 平均执行代价

由于云计算资源是异构的分布式资源,存在实时性和不稳定性,服务流调度的过程中可能会存在有限的时间异常情况,导致实例不能在截止期限内完成。实例不能满足截止期限内完成直接影响到用户对于服务提供商的满意程度,实例完成的成功率越高,服务的性能越好。假设截止期限为 1 000 s 时的 100 个实例中的第 10 个实例中的第 2 个任务 $t_{10,2}$,调度执行中因为执行错误未按照截止期限完成,出现 $\gamma=50$ s 的有限延迟异常,调度失败。根据本文算法在调度前分解截止期限 $TD(t_{10,1})=109.34$ s; $TD(t_{10,2})=138.57$ s; $TD(t_{10,3})=191.25$ s; $TD(t_{10,4})=175.57$ s; $TD(t_{10,5})=191.25$ s; $TD(t_{10,6})=208.0$ s,未出现异常时调度结果如表 3 所示。出现异常后根据式(8)动态调整后续任务的截止期限,得到后续任务的截止期限为 $TD(t_{10,3})=241.25$ s; $TD(t_{10,4})=225.57$ s; $TD(t_{10,5})=241.25$ s; $TD(t_{10,6})=258.00$ s,如表 4 所示。

通过表 3 和表 4 的实验对比,可以看出本文算法可以自动地处理调度过程中任务可能存在的任务时间异常现象,该实例可以在截止期限内完成并且优化了一定的执行代价,从而提高了调度的成功率。

在实验中为了模拟可能出现的任务不能在截止期限内完成的有限延迟异常,在调度过程中给部分任务增加了 $\gamma=100$ s 的延迟,图 4 为出现有限延迟异常情况下所有实例在其截止期限下完成的比率,可以看出本文算法相比

表 3 未出现有限时间异常的调度结果

Cloudlet ID	状态	数据中心 ID	VM ID	时间(s)	开始时间(s)	完成时间(s)	任务截止期限(s)
$t_{10,1}$	成功	2	12	79.2	0	79.2	109.34
$t_{10,2}$	成功	2	14	27.4	96.8	124.2	138.57
$t_{10,3}$	成功	2	5	31.6	157.4	189.0	191.25
$t_{10,4}$	成功	2	3	17.5	131.6	148.1	175.57
$t_{10,5}$	成功	2	6	14.7	173.2	187.9	191.25
$t_{10,6}$	成功	2	2	15.7	190.5	206.2	208.00

表 4 出现异常后本文算法的调度结果

Cloudlet ID	状态	数据中心 ID	VM ID	时间(s)	开始时间(s)	完成时间(s)	任务截止期限(s)
$t_{10,1}$	成功	2	12	79.2	0	79.2	109.34
$t_{10,2}$	成功	2	11	13.7	147.9	188.6	138.57
$t_{10,3}$	成功	2	1	31.6	204.4	236.0	241.25
$t_{10,4}$	成功	2	7	17.5	193.6	211.1	225.57
$t_{10,5}$	成功	2	10	14.7	221.1	235.8	241.25
$t_{10,6}$	成功	2	14	15.7	238.4	254.1	258.00

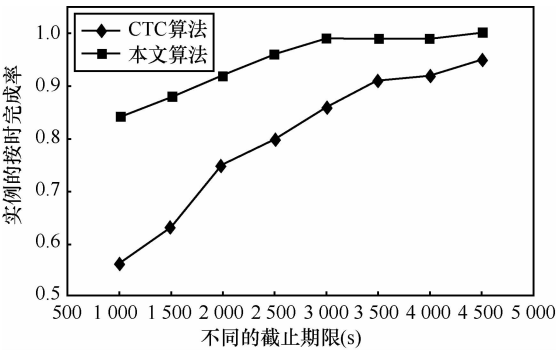


图 4 实例按时完成率

CTC 算法,动态调整后续任务的截止时间,可以提高调度的成功率,从而提高整体的服务性能。

5 结束语

本文提出了一种云计算环境两阶段实例密集型服务流调度算法,该算法先将全局的截止期限分配到每个实例,再将每个实例的截止期限分配到每个实例的任务中,最后在工作流执行阶段,动态调整后续任务的截止期限,解决了工作流调度过程中可能存在的时间异常问题。通过仿真实验,证明本文算法可以满足一定的截止期限,优化执行代价。但是本文算法只是考虑到时间费用这两个 QoS 属性,对于其他的 QoS 属性,如用户满意度、信誉度等没有加以考虑,同时也未考虑到资源动态变化的情况。这些都是下一步研究工作需要深入考虑的问题。

参考文献

1 Buyya R, Abramson D, Giddy J, *et al.* Economic models for resource management and scheduling in grid computing. *Concurrency and Computation: Practice and Experience*, 2002, 14(13-15): 1507~1542

2 Van den Bossche R, Vanmechelen K, Broeckhove J. Cost-optimal scheduling in hybrid IaaS clouds for deadline constrained workloads. *Proceedings of in Cloud Computing (CLOUD) IEEE 3rd International Conference*, Miami, America, 2010: 228~235

3 Van den Bossche R, Vanmechelen K, Broeckhove J. Cost-efficient scheduling heuristics for deadline constrained workloads on hybrid clouds. *Proceedings of in 2011 IEEE Third International Conference on in Cloud Computing Technology and Science (CloudCom)*, Athens, Greece, 2011: 320~327

4 Abrishami S, Naghibzadeh M, Epema D H J. Deadline-constrained workflow scheduling algorithms for Infrastructure as a service clouds. *Future Generation Computer Systems*, 2013, 29(1): 158~169

5 Shen J, Yang Y, Yan J. Adapting P2P based decentralised workflow system swindow-s with web service profile support. *Proceedings of the Ninth International Conference on Computer Supported Cooperative Work in Design*, Chicago, 2005: 535 ~ 540

6 Yang Y, Liu K, Chen J, *et al.* Peer-to-peer based grid workflow runtime environment of SwinDeW-G. *Proceedings of e-Science and Grid Computing*, Bangalore, 2007: 51~58

(下转第 100 页)



- 16 Firesmith D, Donald G. The method framework for engineering system architectures (MFESA). Proceedings of the Seventh International Conference on Composition-Based Software Systems, Washington DC, USA, 2008
- 17 Department of Homeland Security. The system of systems approach for interoperable communications. http://www.safecomprogram.gov/NR/rdonlyres/FD22B528-18B7-4CB1-AF49-F9626C608290/0/SOSApproachforInteroperableCommunications_02.pdf, 2012
- 18 National Center for Systems of Systems Engineering. What is systems of systems engineering. http://www.eng.odu.edu/ncsose/What_is_SOSE.shtml, 2011
- 19 Purdue University, College of Engineering. System of system (SoS). <https://engineering.purdue.edu/Engr/Research/Initiatives/SoS/>, 2012
- 20 Wikipedia. System of systems. http://en.wikipedia.org/wiki/System_of_systems, 2012
- 21 李慧, 杨乃定, 郭晓. 复杂项目系统复杂性构成研究. 软科学, 2009, 23(2): 75~79
- 22 楼俊钢, 江建慧, 靳昂. 考虑软件不同失效过程偏差的软件可靠性模型. 计算机学报, 2010, 33(7): 1264~1271
- 23 张国春, 胡晓峰. 体系对抗仿真中体系效能分析初探. 系统仿真学报, 2003, 15(12): 1698~1701
- 24 张秀伟, 何克清, 王健等. SaaS 服务超市的构建模型和服务推荐方法. 通信学报, 2011, 32(9A): 158~165
- 25 游光荣, 张英朝. 关于体系与体系工程的若干认识和思考. 军事运筹与系统工程, 2010, 24(2): 14~21

军事运筹与系统工程, 2010, 24(2): 14~21

[作者简介]



张秀伟, 男, 武汉大学博士研究生, 主要研究方向为服务计算、云计算、电子商务。



何克清, 男, 武汉大学教授、博士生导师, 主要研究方向为需求工程、ISO 互操作标准、语义互操作技术、服务计算。

王健, 男, 武汉大学讲师, 主要研究方向为需求工程、Web 服务发现、语义互操作技术。

王翀, 女, 武汉大学讲师, 主要研究方向为语义互操作技术、流程管理。

(收稿日期: 2013-07-30)

(上接第 91 页)

- 7 Yu J, Buyya R, Chen K T. Cost-based scheduling of scientific workflow applications on utility grids. Proceedings of International Conference on e-Science and Grid Computing, Melbourne, 2005: 8~147
- 8 Liu K, Jin H, Chen J, *et al.* A compromised-time-cost scheduling algorithm in SwinDeW-C for instance-intensive cost-constrained workflows on a cloud computing platform. International Journal of High Performance Computing Applications, 2010, 24 (4): 445~456
- 9 李文浩, 王海洋. 事务密集型环境下的工作流自适应调度算法. 计算机工程, 2010, 36(19): 84~86
- 10 晏婧, 吴开贵. 适用于实例密集型云工作流的调度算法. 计算机应用, 2010(11): 2864~2866
- 11 Chiu D K W, Li Q, Karlapalem K. A meta modeling approach to workflow management systems supporting exception handling. Information Systems, 1999, 24(2): 159~184
- 12 罗海滨, 范玉顺, 吴澄. 工作流技术综述. 软件学报, 2000, 11(7): 899~907

[作者简介]



刘伟, 女, 广西大学计算机与电子信息学院硕士研究生, 主要研究方向为云计算、工作流技术。



李陶深, 男, 博士, 广西大学计算机与电子信息学院教授, 主要研究方向为云计算、分布式数据库、无线 mesh 网络、网络与信息安全。

黄汝维, 女, 博士, 广西大学计算机与电子信息学院副教授, 主要研究方向为云计算、信息安全。

(收稿日期: 2013-08-20)