DOI:10.13196/j.cims.2020.07.012

移动边缘计算环境下面向安全和能耗感知的服务工作流调度方法

李万清,刘 辉,李忠金+,袁友伟
(杭州电子科技大学 计算机学院,浙江 杭州 310018)

摘 要:在移动边缘计算(MEC)环境中,用户将应用任务迁移至 MEC 端执行可以有效降低延时并减少能耗。然而,MEC 环境面临潜在的恶意攻击,这些攻击可能导致隐私数据的丢失或泄露。基于此,提出了面向安全和能耗感知的服务工作流调度方法(SEA),该算法能在满足移动应用的风险率和截止时间限制条件下,最小化移动设备的能耗。SEA 是基于粒子群优化算法,在编码中考虑了任务的调度位置、机密性服务和完整性服务。此外,还构建了新的安全模型,分别包括数据量、多核 CPU、计算频率与安全开销之间的关系。最后,通过仿真实验证明了所提算法的可行性与有效性。

关键词:移动边缘计算;工作流调度;安全模型;粒子群优化算法
中图分类号:TP311 **文献标识码**:A

Security and energy aware scheduling for service workflow in mobile edge computing

LI Wanqing, LIU Hui, LI Zhongjin+, YUAN Youwei
(College of Computer and Technology, Hangzhou Dianzi University, Hangzhou 310018, China)

Abstract: In the Mobile Edge Computing (MEC) environment, users migrate application tasks to MEC for execution to effectively reduce delay and reduce energy consumption. However, the MEC still faces data security problems, and the potential malicious attacks can result in the loss or disclosure of private data. On this basis, a Security and Energy Aware (SEA) scheduling for service workflow was proposed in MEC environment, which could minimize the energy consumption of mobile devices under the risk rate and deadline constraints of mobile applications. Based on Particle Swarm Optimization (PSO) algorithm, SEA algorithm considered the scheduling position, confidentiality service and integrity service of the tasks. In addition, a new security model was constructed, which included the relationship among data volume, multi-core CPU, computing frequency and security overhead. Simulation experiments demonstrated the feasibility and effectiveness of the proposed algorithm.

Keywords: mobile edge computing; workflow scheduling; security model; particle swarm optimization algorithm

0 引言

随着通信、网络和智能产品的发展,移动便携式的用户设备(User Equipment,UE)越来越受欢迎。新型移动应用如人脸识别、自然语言处理、增强现实等不断涌出,引起了人们的广泛关注。这些移动应用的执行需要较高的计算资源,并消耗较大的电力能源。然而,移动设备由于物理尺寸的限制,通常只具有有限的计算能力和电量。因此,如何在资源受限的移动设备上高效地运行新型移动应用是当前移动网络环境下的一个挑战[1]。

移动边缘计算(Mobile Edge Computing, MEC)

收稿日期:2018-08-21;**修订日期**:2018-11-09。Received 21 Aug. 2018;accepted 09 Nov. 2018.
基金项目:浙江省公益性资助项目(2016C33170);国家自然科学基金资助项目(61802095)。**Foundation items**: Project supported by the Science and Technology Common Wealth Foundation of Zhejiang Province, China(No. 2016C33170), and the National Natural Science Foundation, China(No. 61802095).

的出现为该问题的解决提供了新的平台和机遇。移动边缘计算通过与内容提供商和应用开发商深度合作,在靠近移动用户侧就近提供内容存储计算及分发服务,使应用、服务和内容部署在高度分布的环境中,以更好地满足低延时的需要[2-4]。因此,在 MEC 环境中,执行计算和存储的服务器均部署在网络边缘,UE 可以通过将移动应用的一部分任务卸载到边缘服务器上执行的方式,来提高移动应用的服务质量并减少 UE 的能源消耗。然而,不是任何一个任务卸载方案都会提高服务质量,不合理的任务卸载会造成大量的数据传输和更长的任务执行时间,这不仅导致应用的执行延时,还会增加更多的电池消耗[5-6]。

一个移动应用一般包括多个任务,任务之间存在先序关系和数据依赖关系。与并行任务相比,MEC 环境下的工作流应用调度问题更具有复杂性和挑战性,如任务的执行顺序以及执行位置都会对移动应用的完成时间和能耗产生影响。因此,如何将移动工作流应用中的多个任务合理地分配到移动设备和边缘服务器上执行,在满足工作流应用的约束条件限制下减少移动设备的能耗是移动边缘计算环境下的一个重要研究问题[7-8]。

MEC 能够有效减少 UE 的能耗和应用的延时,数据安全性也是其重点关注的问题之一。在 MEC 环境中,移动应用的数据往往携带隐私信息,由于用户设备和边缘服务器频繁进行数据交互,一些数据在交互过程中有可能丢失或者被恶意篡改,会给用户造成重大的损失。因此,在 MEC 环境中运行移动应用时,需要部署一系列的安全服务以保证数据的安全性[9]。

基于以上分析,本文主要研究移动边缘环境下的工作流调度问题,提出了面向安全和能耗感知(Security and Energy Aware, SEA)的调度算法,该算法能够实现在满足工作流截止时间和风险率约束的条件下 UE 能耗最小化。SEA 算法在粒子群优化(Particle Swarm Optimization, PSO)的基础上融合了工作流调度问题,在编码过程中考虑了任务的调度位置、机密性服务和完整性服务。针对异构服务器对安全服务计算时间影响,构建了新的安全开销模型,包括数据量与安全开销的关系、多核 CPU 与安全开销的关系和计算频率与安全开销的关系,使得该模型可以根据 MEC 的参数,对任务的安全开销进行定量描述。

本文的主要贡献如下:①从移动用户的安全性出发,提出一种面向安全和能耗感知的工作流调度方法;②对数据量、CPU 核数和计算频率与机密性服务和完整性服务之间的关系进行建模,提出了安全开销的定量描述方法;③提出一种基于 PSO 的工作流调度编码策略,解决了多维多约束优化问题。

1 相关工作

近年来,移动网络及应用得到了飞速发展,许多学者对应用的任务调度进行了大量研究。Zhu 等[10]在遗传算法的基础上提出了多目标优化(Evolutionary Multi-objective Optimization, EMO)算法,对工作流的执行时间和成本进行优化,解决了基础架构即服务(Infrastructure as a Service, IaaS)平台上的工作流调度问题;Jia 等[11]研究了移动边缘计算环境中多用户通信和计算资源分配问题,提出 3 种不同的卸载策略,即局部压缩卸载、移动云压缩卸载和部分压缩卸载,显著减少了应用的延迟;曹斌等[12]提出基于粒子群算法的最优化调度搜索方法,利用关键路径进行粒子初始化和搜索阶段的筛选处理,解决了在时间约束下最小化费用问题;周业茂等[13]考虑了用户的不断移动,提出了基于延时传输机制的多目标工作流调度算法,解决了网络变化情况下工作流调度问题。

Xie 等[14]提出针对具有期限和安全约束的并行应用程序的任务分配算法,以及针对并行任务的安全感知和异构感知资源分配的算法,解决了集群中具有安全性约束的实时并行作业资源分配问题;Zeng 等[15]指出云环境的工作流调度策略不仅要考虑 CPU 执行时间,还需考虑内存、存储容量等其他资源,提出一种安全和费用感知工作流调度策略(Security-Aware and Budget-Aware, SABA),该策略在市场可用的云服务提供商之间选择经济任务分配方案,为客户提供更短的完工时间以及安全服务;Li 等[16]考虑到在科学工作流中任务通常是异构的,提出一种安全和成本感知调度(Security and Cost Aware Scheduling, SCAS)算法,使任务在执行过程中,在满足截止时间和风险约束的同时,最小化工作流执行成本;Chen 等[8]提出工作流调度策略不仅需要满足中间数据的安全性需求,还要考虑加密中间数据对后续工作流任务开始的性能影响,提出一种具有选择性任务重复调度算法,有选择地重复执行前驱任务,有效地利用任务松弛时间优化执行时间、

货币成本和资源效率。

Liu 等[17]通过研究移动边缘环境下如何最小化执行延迟,提出了基于马尔可夫的一维搜索算法,该算法根据应用程序缓冲区排队状态、UE 和 MEC 服务器上可用的计算资源以及 UE 和 MEC 服务器之前网络特性,找出最优的卸载决策策略;Mao 等[18]通过研究带有能源收集装置的绿色 MEC 系统,提出了基于 Lyapunov 的动态计算卸载算法,该算法决定了卸载决策、移动设备执行的 CPU 周期频率和计算卸载的传输功率,达到执行成本(执行时延和任务失败)最优。

当前在移动边缘环境下进行工作流安全调度的研究很少,本文在上述研究的基础上提出的 SEA 算法,实现了在满足安全需求和工作流的截止时间的情况下,最小化移动设备的能耗。

2　模型及问题描述

下面介绍 MEC 环境下任务调度框架、服务工作流模型、安全服务模型、任务执行分析和工作流风险分析,提出具有安全需求的工作流调度问题。

2.1　任务调度框架

如图 1 所示,MEC 系统由用户设备、无线网络和若干个部署了边缘服务器的基站组成[19]。假设一个用户设备执行某个应用,该应用被建模成一个一般工作流。工作流中的任务有可能被卸载到演进型基站(evolved Node B,eNB)上或在 UE 上执行。通常 eNB 管理着不同类型的服务器,每台服务器拥有不同的计算资源,如 CPU 内核数量、CPU 频率大小和内存大小,因此每台服务器拥有不同的计算能力。$S=\{s_0,s_1,s_2,\cdots,s_m\}$ 表示服务器的集合,其中 s_0 为移动设备本身,m 表示边缘服务器的总数。此外,假设不同 eNB 信号不能相互干扰,eNB 之间拥有相同的通信带宽。

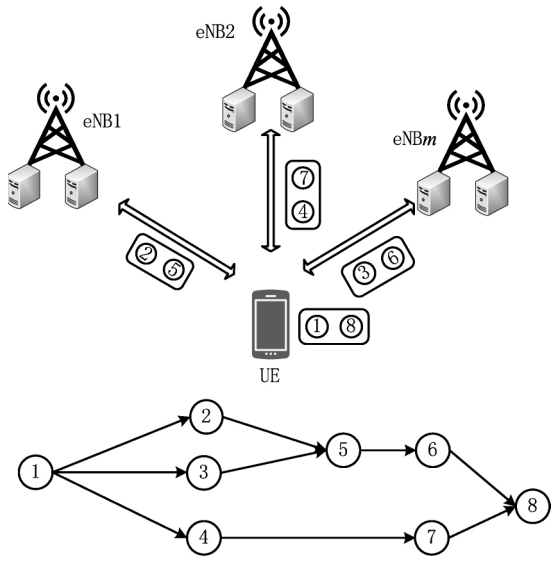


图1　MEC工作流调度环境

2.2　服务工作流模型

本文在经典的有向无环图(Directed Acyclic Graph, DAG)基础上进行扩展,重新定义了具有安全性的工作流 $W=\{T,E,\theta,T_d\}$,如图 1 所示。其中:$T=\{t_1,t_2,\cdots,t_n\}$ 表示 n 个任务的集合,E 为任务之间的有向弧或边的集合,表示任务之间的依赖关系,例如边 $e(i,j)$ 表示任务 t_i 必须在任务 t_j 开始前完成执行的约束。这种情况下,任务 t_i 被认为是任务 t_j 的前驱任务,任务 t_j 被认为是任务 t_i 的后继任务,式(1)表示任务 t_i 的所有前驱任务:

$$pred(t_i)=\{t_j \mid (t_j,t_i)\in D\}。\tag{1}$$

在工作流中,一个任务通常可以有一个或者多个输入,一旦输入完成,将触发任务执行。任务 t_i 的输入总量为 I_i,I_i 等于任务 t_i 的前驱任务传输给任务 t_i 数据总量。假设任务的输出已知,当任务 t_i 完成时,输出 O_i。在工作流中没有前驱任务节点的任务称为 t_{entry},表示工作流的开始任务。此外,没有后继任务节点的任务称为 t_{exit},表示工作流的结束任务。通常一个工作流有且仅有一个 t_{entry} 和 t_{exit}。如图 1 所示,工作流中 t_1 和 t_8 分别表示工作流的 t_{entry} 和 t_{exit}。工作流中 θ 表示安全约束,它的值是由用户根据数据的敏感程度指定的,$0\leqslant\theta\leqslant1.0$,安全约束最低为 0,最高为 1.0。此外,每个工作流都有一个截止时间 T_d,它被定义为工作流执行的时间约束。

2.3　安全服务模型

根据任务调度模型考虑单个 UE 将任务卸载到多个 eNB 的情况。这些 eNB 通常是异构的,它们支持不同的覆盖范围,具备不同的计算能力。一般而言,eNB 中的服务器也是异构,即每个服务器拥有不同的 CPU、计算性能和存储资源。

目前,在移动边缘计算环境下用户隐私数据面临安全威胁[20],但是许多现有移动边缘计算环境并没有采用任何安全机制来对付安全威胁,因此有必要部署安全服务来保护用户的数据安全[21]。欺骗和篡改是移动边缘环境中两种常见的攻击,如恶意用户利用非法程序获得 MEC 服务器部分控制权并篡改用户的数据[22]。因此,考虑实现两种安全服务(机密性服务和完整性服务)[23]来综合保障数据的

安全性。通过对应用程序(可执行文件)和数据进行加密来支持机密性,从而使信息和资源无法提供或者泄露给未经授权的人员或者应用;数据的完整性服务确保数据不被授权篡改或在篡改后能够被迅速发现。使用安全服务保护用户隐私数据是以性能为代价的,即安全服务会带来一定的时间开销,不同的安全服务算法(方法)会产生不同的时间开销[14,23]。

文献[9,14,23]都构建了安全开销模型,可以合理度量安全服务引起的安全开销,但是这些文献只考虑在固定的处理器、固定计算频率下数据量和安全服务之间的关系。在异构 MEC 环境下,eNB 是异构的,它们的服务器也是异构的,这些服务器的处理器拥有不同的 CPU 核数和不同计算频率。为了精准衡量安全服务的时间开销,还需要研究多核 CPU、不同计算频率与安全开销的关系。例如 Barnes 等[24]考虑 CPU 多核体系研究了加密算法安全开销,证明多核吞吐量远远大于单核串行执行。机密性服务的算法通常包括 IDEA(international data encryption algorithm)、DES(data encryption standard)、AES (advanced encryption standard)、Blowfish 和 RC4 (rivest cipher 4),如表 1 所示;完整性服务的算法包括 TIGER、RipeMD160(integrity primitives evaluation message digest 160)、SHA-1 (secure hash algorithm 1)、RipeMD128(integrity primitives evaluation message digest 128)和 MD5 (message digest 5),如表 2 所示。数据机密性和完整性算法分别记为 $CS=\{cs(1,c,f),cs(2,c,f),\cdots,cs(N(CS),c,f)\}$,$IS=\{is(1,c,f),is(2,c,f),\cdots,ci(N(IS),c,f)\}$,$N(j),j\in\{CI,IS\}$表示第 j 个安全服务可选算法数量。其中:$N(CS)$表示机密性加密算法总数,c 表示运行安全服务使用的 CPU 核数,f 表示 CPU 频率(单位:GHz)。例如,$ci(1,4,2.2)$表示在 4 核、频率为 2.2 GHz 的 MEC 服务器运行机密性服务第 1 个算法。运行算法的服务器为戴尔 R530 2.2 GHz,CPU 核数为 8 核。下面分别研究数据量、多核 CPU、不同计算频率和安全开销之间的关系,建立相应的数学模型。

表 1 单核机密性服务加密算法

符号	加密算法	安全等级 sl_i^l	执行速度 $es(ci(i,c,f))$/(MB/s)
$cs(1,1,2.2)$	IDEA	1.0	11.76
$cs(2,1,2.2)$	DES	0.85	13.83
$cs(3,1,2.2)$	AES	0.53	22.03
$cs(4,1,2.2)$	Blowfish	0.56	20.87
$cs(5,1,2.2)$	RC4	0.32	37.17

表 2 单核完整性服务哈希算法

符号	哈希算法	安全等级 sl_i^l	执行速度 $es(is(i,c,f))$/(MB/s)
$is(1,1,2.2)$	TIGER	1.0	75.76
$is(2,1,2.2)$	RipeMD160	0.75	101.01
$is(3,1,2.2)$	SHA-1	0.69	109.89
$is(4,1,2.2)$	RipeMD128	0.63	119.05
$is(5,1,2.2)$	MD5	0.44	172.41

2.3.1 安全等级建模

在对算法的安全等级建模时,采用单核对固定大小数据运行安全服务算法,其中服务器的计算频率为 2.2 GHz,加密数据的大小为 100 MB。根据加密算法的执行速度(如表 1 中执行速度)为每个算法分配一个安全等级,安全等级范围为 0.32~1.0。例如,将 1.0 分配给最强但计算速度最慢的加密算法 IDEA(如表 1),其余加密算法的安全等级的计算方式同文献[14,23],具体表示为:

$$sl_i^l=\frac{es(cs(1,1,2.2))}{es(cs(i,1,2.2))}1\leqslant i\leqslant 5,l=cs。\quad(2)$$

式中:$co(cs(i,1,2.2))$为第 i 个加密算法的安全开销;sl_i^l 表示安全服务 l 的第 i 个加密算法的安全等级,$l\in\{cs,is\}$。

同理,根据哈希算法的执行速度(如表 2 中执行速度),为每个算法分配安全等级,安全等级范围为 0.44~1.0。将 1.0 分配给最强但是执行速度最慢的哈希算法 TIGER(如表 2),其余哈希算法的安全等级可以由式(3)计算得到:

$$sl_i^l=\frac{es(is(1,1,2.2))}{es(is(i,1,2.2))}1\leqslant i\leqslant 5,l=is。\quad(3)$$

2.3.2 数据量与安全开销的关系

在考虑数据量与安全开销的关系时,采用单核、固定频率对不同大小的数据运行安全服务算法,其中服务器的频率为 2.2 GHz,实验效果如图 2 所示。在对不同大小的数据运行加密算法时,其安全开销和数据量成正比,具体关系如式(4)所示:

$$co(cs(i,c,f),d)=\frac{d}{es(cs(i,c,f))}。\quad(4)$$

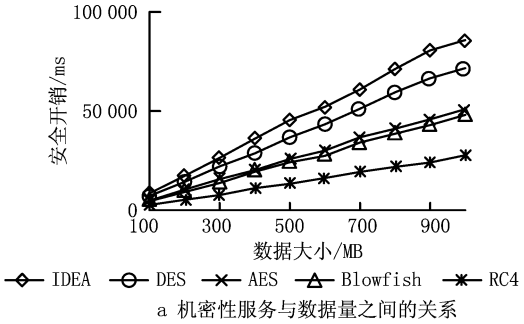
式中 d 为需要执行加密数据大小,当 $d=100$ 时,$co(cs(i,c,f),100)$表示对 100 MB 数据执行加密的安全开销。如图 2a 所示为对不同大小数据执行机密性服务,数据量和安全开销的关系。

同理,在对不同大小的数据运行哈希算法时,运行哈希算法的安全开销和数据量成正比,具体关系

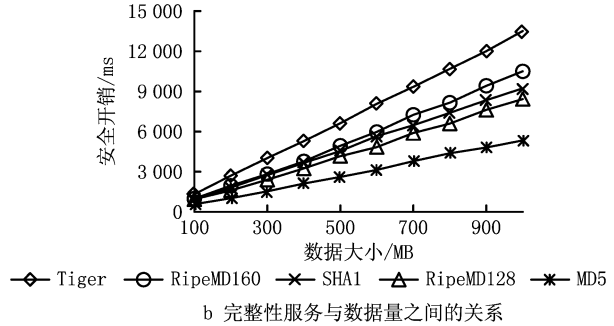
如式(5)所示:

$$co(is(i,c,f),d)=\frac{d}{es(is(i,c,f))}。\quad(5)$$

当 $d=100$ 时,$co(is(i,c,f),100)$表示对 100 MB 数据执行完整性服务的安全开销。如图 2b 所示为对不同大小数据执行完整性服务时,数据量和安全开销的关系。



a 机密性服务与数据量之间的关系



b 完整性服务与数据量之间的关系

图2　不同数据大小单核安全开销

2.3.3　多核 CPU 与安全开销的关系

考虑到 CPU 的核数与安全开销的关系,本文对固定大小安全数据实现安全服务,在固定计算频率,利用不同核数运行安全服务,得到不同核数的安全开销。选择 100 MB 数据作为实验数据,在计算频率为 2.2 GHz 的服务器上运行安全服务算法。首先,将原始数据 D 切分成 N 份数据集,其中 N 对应算法使用的核数;然后利用多进程对 N 份数据集实现安全服务,实验效果如图 3 所示,随着 CPU 核数的增加,安全开销呈下降趋势。这是由于待加密数据被分成了若干份,实际上每个核只对每个小块数据进行加密。

在保护相同大小数据时,多核实现机密性服务的安全开销和核数成反比,具体关系如式(6)所示:

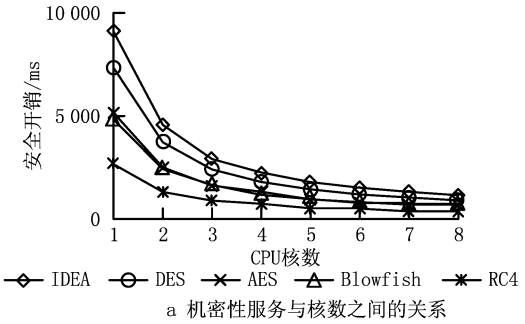
$$co(cs(i,c,f))=\frac{co(cs(i,1,f),d)}{c}。\quad(6)$$

式中 $co(is(i,1,f))$表示单核运行加密算法的安全开销。如图 3a 所示为对 $d=100$ MB 数据实现机密性服务时,多核服务器运行加密算法的安全开销和采用的核数关系。

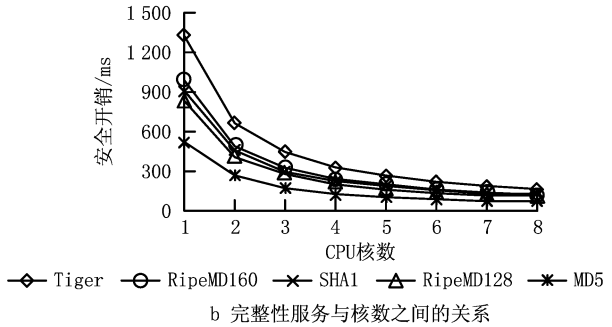
同理,可以得出多核实现完整性服务的安全开销和核数成反比,如式(7)所示:

$$co(is(i,c,f))=\frac{co(is(i,1,f),d)}{c}。\quad(7)$$

如图 3b 所示为对 100 MB 数据实现完整性服务时,多核服务器运行哈希算法的安全开销和核数关系。



a 机密性服务与核数之间的关系



b 完整性服务与核数之间的关系

图3　不同核数安全开销

2.3.4　计算频率与安全开销的关系

考虑到 CPU 的计算频率与安全开销的关系,采用具有不同计算频率单核对固定大小安全数据实现安全服务。实验效果如图 4 所示,随着计算频率升高,算法的安全开销呈下降趋势。在保护相同大小数据时,服务器实现机密性服务的安全开销和计算频率成反比,具体关系如式(8)所示:

$$co(cs(i,f,1)=\frac{co(cs(i,1,F),d)\times F}{f}。\quad(8)$$

式中 F 为 CPU 的最大计算频率。图 4a 所示为在对 100 MB 数据实现机密性服务时,服务器运行加密算法的安全开销和计算频率关系。同理,得出了完整性服务的安全开销和计算频率关系,如式(9)所示:

$$co(is(i,f,1)=\frac{co(is(i,1,F),d)\times F}{f}。\quad(9)$$

如图 4b 所示为对 100 MB 数据实现完整性服务时,服务器运行完整性的开销和安全数据的大小

关系。

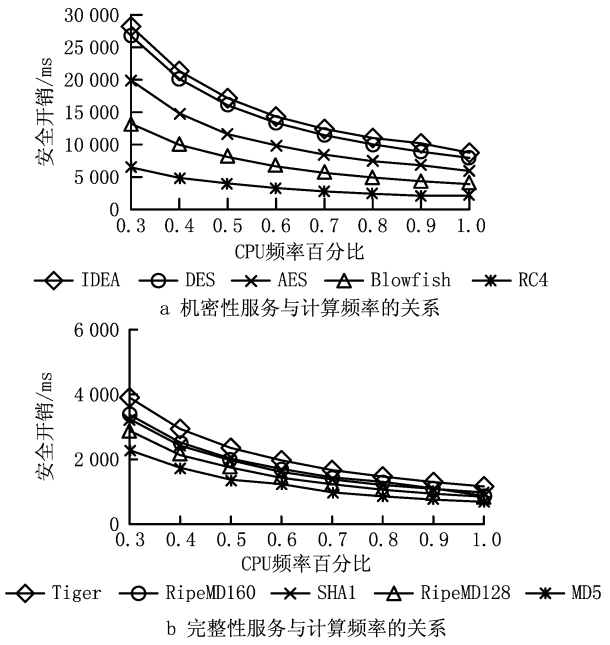


图4 不同CPU频率安全开销

基于以上分析，可以得到实现安全服务的安全开销和数据大小、CPU 核数、计算频率的关系。由于算法的安全等级是算法的固有属性，跟 CPU 的核数和计算频率无关，本文在单核上以最大频率运行算法的执行速度作为衡量算法的安全等级。对于给定一个服务器，已知它的 CPU 核数和计算频率，通过式(10)和式(11)分别计算出机密性服务和完整性服务的开销：

$$co(cs(i,f,c),D)=\frac{2.2}{f}\times\frac{1}{c}\times\frac{D}{es(cs(i,1,2.2)}, \tag{10}$$

$$co(is(i,f,c),D)=\frac{2.2}{f}\times\frac{1}{c}\times\frac{D}{es(is(i,1,2.2)}。 \tag{11}$$

其中：D 为需要保护的数据量；$cs(1,1,2.2)$表示单核以 2.2 GHz 运行第 i 个加密算法的安全开销；$is(1,1,2.2)$表示单核以 2.2 GHz 运行第 i 个哈希算法的安全开销。

2.4 任务执行分析

假设执行任务 t_i 的服务器记为 $s(t_i)$，则任务 t_1 分配给服务器 s_1 执行，记为 $s(t_1)=s_1$。图 1 中工作流的部分任务执行过程如图 5 所示，任务 t_1 在 UE 上执行，t_1 执行完成后对输出数据实现机密性服务(图 5 中 E)和完整性服务(图 5 中 H)。任务 t_1 将输出数据传输给任务 t_2，任务 t_2 接收数据后，对数据进行完整性验证(图 5 中 IV)和解密(图 5 中 DE)。由于任务 t_2 和任务 t_5 在同一个服务器执行，它们之间不需要执行安全服务，即任务 t_5 可直接使用任务 t_2 的输出数据。任务接收输入数据的时间可以表示为

$$TT(t_i)=\frac{I_i}{B}。 \tag{12}$$

式中 B 为服务器之间的一个带宽。如果两个任务在同一个服务器上执行，它们的传输时间为 0，因此图 5 中 t_2 和 t_5 的传输时间为 0。

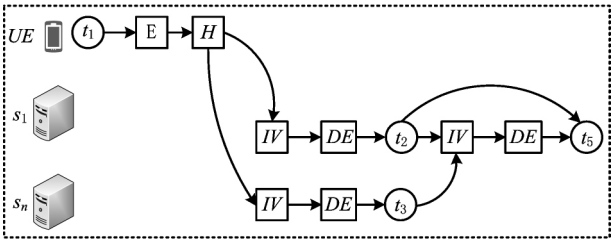


图5 具有安全服务的任务执行过程

当任务接收到输入数据，首先需要对数据进行完整性验证，时间记为 $IV(t_i)$，然后通过解密算法对数据解密，解密时间记为 $DE(t_i)$。当输入完成后，任务 t_i 开始在服务器 $s(t_i)$上执行，任务执行时间

$$ET(t_i,s(t_i))=\frac{w(t_i)}{p_s(t_i)}。 \tag{13}$$

式中：$w(t_i)$为任务 t_i 工作量；$p_s(t_i)$为服务器 $s(t_i)$计算能力。此外，由于完整性服务可以确保无人修改或者篡改数据未被发现，因此在执行任务时被用于处理数据更改的威胁。最后，为了避免对任务 t_i 的输出数据进行非法授权的拦截，对输出数据进行加密以应对窥探攻击，确保未经授权的人员无法获取数据。因此，任务 t_i 在 $s(t_i)$处理时间

$$PT(t_i,s(t_i))=TT(t_i)+IV(t_i)+DE(t_i)+ET(t_i,s(t_i))+H(t_i)+E(t_i)。 \tag{14}$$

2.5 工作流风险分析

在 MEC 环境中，工作流应用程序执行过程并非无风险，需要对安全服务进行定量评测得出整个工作流的风险率。在该风险分析模型中，假设风险概率是安全等级的函数，任何固定时间间隔的风险率分布满足 Poisson 概率分布。因此，任意第 k 个安全服务的风险概率可以用指数分布表示[25-26]为：

$$P(t_i,sl_i^l)=1-\exp(-\lambda^l(1-sl_i^l))l\in(cs,is)。 \tag{15}$$

在不同的服务器中，风险系数 λ 通常不同，例如不同服务器在单位时间间隔内遭遇 3 次更改攻击和

2 次欺骗攻击,负指数表示故障概率,当 $1-sl_i^l$ 增大时,故障率也会增大。考虑所有的安全服务,可以得到对于某个数据的风险概率。对于给定工作流 W,工作流的风险概率 $P(W)$计算如下:

$$P(t_i)=1-\prod_{l\in\{cs,is\}}(1-P(t_i,sl_i^l)),\tag{16}$$

$$P(W)=1-\prod_{t_i\in T}(1-P(t_i))。\tag{17}$$

2.6　问题描述

本文主要在移动边缘环境下实现一种调度算法,该算法在用户设备能耗最小的同时满足工作流截止期限和风险概率约束。根据资源集合、所有任务的安全级别、任务资源映射、用户设备的能耗、总执行时间、工作流风险和卸载决策定义调度算法为 $schedule=(M,SL,E,TET,P(T),x)$。其中:$M$ 表示一个映射,由 $m(t_i,s(t_i))=(t_i,s(t_i),ST(t_i),FT(t_i))$组成,映射元组的解释如下:任务 t_i 在 $s(t_i)$上执行,在 $ST(t_i)$时刻开始执行,并在 $FT(t_i)$时刻完成;$SL=\{sl_i\mid i=0,1,\cdots,n-1\}$表示每个任务的安全等级;工作流的风险率 $P(T)$通过式(17)计算得到;$x=\{x_0,x_1,\cdots,x_n\}$是整个工作流的卸载决策集合,$x_i=1$ 表示任务 t_i 在用户设备上执行,$x_i=0$ 表示任务 t_i 不在用户设备上执行。移动设备的能耗和工作流的总执行时间计算如下:

$$E=\sum_{i=0}^{n-1}x_i(FT(t_i)-ST(t_i))\times E_0,\tag{18}$$

$$TET=\max(ET(t_i)\mid t_i\in T)。\tag{19}$$

此外,$(FT(t_i)-ST(t_i))\times E_0$ 是任务 t_i 在移动设备执行的能耗,E_0 为移动设备单位时间的能耗。根据之前的定义,问题可描述为:找出一个调度算法使得 E 最小,并且 TET 不超过截止期限,$P(W)$满足工作流的风险率约束,具体如下:

$$\min E;\tag{20}$$

$$\text{s.t.}\quad TET\leqslant T_d,\tag{21}$$

$$P(W)\leqslant\theta。\tag{22}$$

3　算法实现

工作流调度问题是一个 NP 难问题,因此本文提出一种基于 PSO 能耗和 SEA 算法得到问题的最优解解集。在粒子群算法中,许多粒子被放置在某个问题的搜索空间中,以一定速度在搜索空间探索[27]。每个粒子由 3 个 D 维矢量组成,这 3 个矢量分别为当前位置 x_i、历史最佳位置 p_i 和速度 v_i。当前位置 x_i 表示优化问题的候选解,如果当前位置好于历史最佳位置,将现有位置保存在 p_i 向量中。在群体中所有粒子经历过的最好位置的索引号用向量 p_g 表示,它的适应度记为 $gbest$。此外,个体目前最佳适应度的值保存到参数 $pbest_i$ 中。在粒子群优化过程中,算法将继续迭代,直到满足停止条件:通常是达到最大迭代次数或者一个预定义的适应度值。在每次迭代中,粒子的速度和位置基于式(23)和式(24)更新:

$$\boldsymbol{v}_i\leftarrow w\times\boldsymbol{v}_i+\varphi_1\times rand_1(\boldsymbol{p}_i-\boldsymbol{x}_i)+\varphi_2\times rand_2(\boldsymbol{p}_g-\boldsymbol{x}_i),\tag{23}$$

$$\boldsymbol{x}_i\leftarrow\boldsymbol{x}_i+\boldsymbol{v}_i。\tag{24}$$

其中:加速常数 φ_1 和 φ_2 表示每个粒子推向 p_i 和 p_g 位置的统计加速项的权重大小,这两个加速度常数被认为是一个单一的加速度常数 $\varphi=\varphi_1+\varphi_2>4$,一般情况下 φ_1 和 φ_2 不一定非要相等[28]。$rand_1$ 和 $rand_2$ 是在[0,1]范围内均匀分布的随机数。惯性权重 w 使粒子保持运动的惯性,使其有扩展搜索空间的趋势,有能力探索新的区域。此外种群的大小通常在 20～50 之间,粒子的维度和它们允许移动的范围,这些值完全取决于问题的性质以及它如何模拟以适应 PSO,粒子的维度和范围将在 3.1 节中描述。SEA 算法的伪代码如算法 1 所示,下面将详细介绍它们的相关实现。

算法 1　基于 PSO 的 SEA 算法。

```
BEGIN
01. N←粒子的大小,D←粒子的维度;
02. gbest←0,pg←0;
03. for i= 0 to N//遍历所有的粒子
04. 初始化 xi 和 vi;
05.     Let pi=xi,pbesti=fitness(xi)
06. end for
07. while iterate//最大的迭代次数
08.   for i=1 to N
09. 根据工作流的调度算法和条件限制计算每个粒子的适应度;
10. 更新 pbesti,pi;
11. end for
12. 选择到目前为止所有粒子的最佳适应度值的粒子,将其索引赋给变量 g;
13.     for i=1 to N
14. 根据式(23)式(24)更新 vi,xi;
15. 保持粒子在搜索空间内,以防它超出其边界
16. end for
17. Iterate++//迭代次数加 1
18. end while
END
```

3.1 编码

要定义问题的编码，需要了解粒子的表示方式和维度。SEA 算法的目标是在满足期限和风险率约束的同时最小化移动设备的能耗，为每个任务选择适当的服务器和安全等级。为了求解式(20)，设计了以下编码策略：

$$\min E = \Gamma(s(t_0), sl_0^{cs}, sl_0^{is}, \cdots, s(t_i), sl_i^{cs}, sl_i^{is}, \cdots, s(t_{n-1}), sl_{n-1}^{cs}, sl_{n-1}^{is}, \cdots,)。 \quad (25)$$

$$\text{s.t.} \quad s(t_i) \in S, i = 0,1,\cdots,m-1; \quad (26)$$

$$sl_i^l \in SL^l, l \in \{is, cs\}, i = 0,1,\cdots,n-1。 \quad (27)$$

其中 $\Gamma(\cdot)$ 的能耗函数与执行任务的服务器、任务安全等级有关。此外，$SL^l, l \in \{is, cs\}$ 是一组安全服务的等级，其中包括 0，如机密性服务包括 6 个等级(如表 1)，$SL^l = \{1.0, 0.85, 0.53, 0.56, 0.32, 0\}$。

粒子的个数等于函数 $\Gamma(\cdot)$ 中的参数数量，即 $D = 3 \times n$。如图 1 所示，工作流有 8 个任务，每 3 个粒子属于一个任务，因此粒子个数是 24。粒子的值可以在相应的搜索空间中变化，每个任务的第一个粒子的值表示该任务运行的服务器；第二个粒子的值是任务的机密性的安全等级；第三个粒子的值是任务的完整性的安全等级，如任务 t_1 编码为(0，0.36，0.3)，表示任务 t_1 在服务器 s_0 上执行，任务的机密性和完整性的安全等级分别为 0.36 和 0.3。

3.2 工作流调度

工作流调度的伪代码如算法 2 所示。初始化服务器 S、安全等级 SL 和映射 M 为空，能耗 E、执行时间 TET 和风险率 $P(T)$ 为 0，然后根据粒子的位置构建调度。至此，开始遍历位置中每个坐标并更新 S, SL 和 M，首先确定任务、服务器和安全等级对应的粒子位置和对应值，这是通过使用前面描述的编码策略得到的，即粒子的索引 $3i, 3i+1, 3i+2$ 对应任务 t_i，它们的值 $pos[3i], pos[3i+1], pos[3i+2]$ 分别对应执行任务的服务器、机密性安全等级和完整性安全等级；然后计算任务的开始时间 $ST(t_i)$ 的值。任务开始时间 $ST(t_i)$ 主要有两种场景：①任务没有前驱任务节点，则立即执行；②任务有一个或者多个前驱任务节点，等待所有前驱任务节点完成才开始执行。

如果服务器被使用，则 $FT(t_i)$ 的值根据任务开始时间和整个任务处理时间得出。任务的处理时间 $PT(t_i, s(t_i))$ 由数据接收时间、输入数据解密时间、执行时间和安全开销组成。$FT(t_i)$ 是 $ST(t_i)$ 和 $PT(t_i)$ 总和。

算法 2 工作流调度算法。

```
Begin
01. R=M=SL=∅,E=TET=P(W)=0;
02. if T≠∅
03. for t_i∈T do
04. if pred(t_i)=∅ then
05. ST((t_i)=0;
06. else
07. ST(t_i)=max{FT((t_j)|(t_j∈pred((t_i)}
08. end if
09. if s(t_i)=s(t_{i-1})//  任务 t_i 和任务 t_{i-1}在同一台服务器上执行
10. I_{i-1,i}=0//  任务 t_{i-1}传输到任务 t_{i-1}数据量为 0
11. 根据式(14)计算执行时间 PT(t_i,s(t_i))
12. FT(t_i)=ST(t_i)+PT(t_i,s(t_i))
13.  if s(t_i)=s_0
14. x_i=1
15. end if
16. 根据式(14)计算执行时间 PT(t_i,s(t_i))
17. FT(t_i)=ST(t_i)+PT(t_i,s(t_i))
18. end if
19. 集合 M 添加 m(t_i,s(t_i),ST(t_i),FT(t_i)),SL 添加 sl_i
20. end for
21. 根据式(17)计算 P(W)
22. 根据式(18)计算 E
23. 根据式(19)计算 TET
24. 记录 schedule=(M,SL,E,TET,P(W),x)
END
```

3.3 约束处理

针对约束条件，采用文献[29]中的 3 种可行解方法优先性准则：①根据适应度函数，在两个可行解之间寻找最优解；②可行解比不可行解更好；③在两个不可行解之间，优先考虑违约之和较小的解。本文将约束条件式(21)和(22)转化为约束数的总和，具体计算如下：

$$f(\boldsymbol{x}_i) = \max(0, TET - T_D) + \max(0, P(T) - \theta)。 \quad (28)$$

3.4 粒子越界和算法复杂度分析

速度和位置可能导致粒子超出可行区域的边界，因此必须修改 PSO 算法以使粒子在约束内搜索最优解。这样做的方式可能会对算法的性能产生很大的影响，因为它会影响粒子在搜索空间中移动的方式，当最优决策变量值位于或接近边界时，这一点尤为重要。因此，当一个决策变量超出其边界时需要做两件事：①取其相应边界的值(要么是下边界，要么是上边界)；②其速度乘以 −1，以便在相反的方向上搜索[30]。

SEA 算法在每次 PSO 迭代中通过更新粒子的位置和速度获得新的适应度。种群的数量 N 和粒子的大小 D 决定了更新粒子位置和速度所需的计算次数,根据粒子定义可知粒子的大小 $D=3\times n$。粒子通过遍历所有的任务计算适应度,则适应度函数的时间复杂度为 $O(n)$。基于以上分析,SEA 的每次迭代的总体复杂度为 $O(3\cdot n^2\cdot N)$。

4　实验

首先通过仿真实验对 SEA 算法的性能进行评估,主要从能耗、安全服务和任务数量 3 个方面考虑。在仿真实验中,采用随机生成的方式产生工作流,其中任务的工作量大小在[1,10]随机产生;每个任务的输出数据大小服从[1,10]均匀分布(单位:MB);随机选择 3 个服务器。用户设备的计算能力、计算功率、数据的发送功率、数据的接收功率和带宽以及 MEC 服务器配置如表 3 所示。利用面部表情识别流程作为实例验证了算法的可用性,实验使用的移动设备为荣耀 8(4 核 2.3 GHz),服务器选择三台联想服务器 RD450(8 核 2.1 GHz)。

表 3　用户设备和 MEC 服务器性能参数

移动设备	计算能力/GHz	2.36
	计算功率/W	0.5
	发送功率/W	0.1
	接收功率/W	0.05
	带宽/(Gb/s)	2
MEC 服务器 s_1	计算能力/GHz	2.3
	CPU 核数	4
MEC 服务器 s_2	计算能力/GHz	3.1
	CPU 核数	8
MEC 服务器 s_3	计算能力/GHz	2.2
	CPU 核数	16

4.1　能耗对比

在本节中仿真下面的算法:

(1)LOCAL:该算法中工作流中任务都在用户设备执行。

(2)Max_Level:该算法将每个任务的所有安全等级设置为 1.0,因此工作流的风险率始终为 0。

(3)Min_Level:该算法中所有任务都不实现安全服务,即工作流的风险率始终为 1.0。

(4)SEA:本文所提算法是在满足截止时间和风险概率约束的情况下最小化移动设备的能耗。

在这组实验中,通过改变风险率大小,研究风险系数对 4 种算法性能的影响。风险率从 0.1 变化到 1.0,步长为 0.1。任务数量分别取 10、50 和 100;种群数量和迭代次数取值 50 和 1 000。4 种算法的能耗的仿真结果如图 6 所示。总的来说,LOCAL 算法的能耗最高,Min_Level 算法的性能最好,SEA 算法和 Max_Level 算法能耗适中,前者优于后者。Max_Level 和 Min_Level 算法的曲线表明能耗与风险率无关,因为两种算法都使用了固定的安全服务。对于 SEA 和 LOCAL 算法来说,开始时能耗下降很快,当风险率等于或大于 0.5 时,能耗下降缓慢。这是因为风险概率函数是指数函数,当工作流的风险率较低时,每个任务都要求较低的风险率,这就需要较高的安全服务,从而导致能耗较高。由于 LOCAL 算法在用户设备执行,能耗比其他算法要高,SEA 算法的能耗在 Max_Level 算法和 Min_Level 算法之间变化。因此,用户可以实现工作流执行的能耗安全权衡,以保证工作流的执行,保护自己的隐私。

4.2　安全服务对比

本节考虑 SEA 算法的两个安全服务对安全工作流的性能影响。缩写 integ_only 和 confi_only 分别表示完整性服务和机密性服务,integ_only 表示

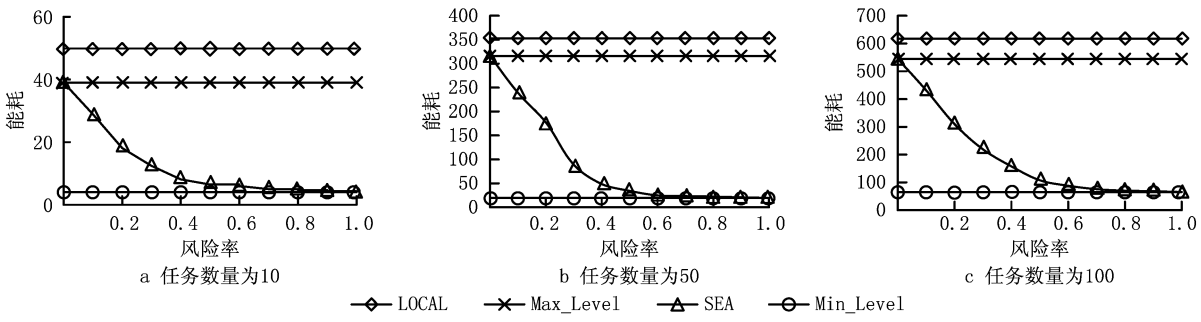


图6　4种算法能

应用只实现了任务的完整性服务；confi_only 表示应用只实现了任务的安全性服务。如图 7 所示为风险率对这两种安全服务的能耗影响。对于完整性服务和机密性服务，它们的安全开销取决于要保护的数据的大小，因此其能耗随着风险率的升高而降低。此外，由于机密性服务比完整性服务执行慢，在相同的风险率下，confi_only 的能耗高于 integ_only。

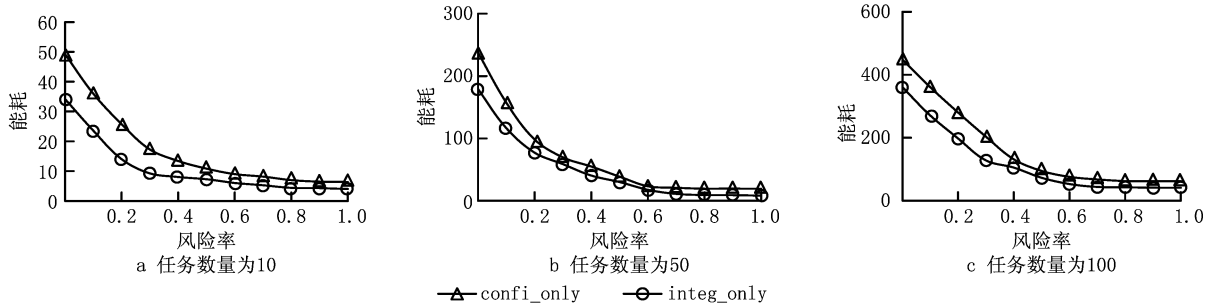


图7 不同风险率下integ_only，onfi_only算法的能耗

4.3 安全系数影响

如 2.5 节所述，风险率与风险系数高度相关。例如，如果第 l 个安全 6 服务的风险系数 $\lambda^l=0$，则其风险概率为 0，因为它不会遭受相应的攻击，所以任务不需要实现第 l 个安全服务。本节将重点讨论 SEA 算法的风险系数的性能影响，实验结果如图 8 所示，工作流的风险系数从 0.3～3.0。从图 8 可以看出，integ_only 较优于 confi_only，原因已经在 4.2 节中解释了，这里不再赘述；从图 8 中还可以看出，当风险系数高于 2.0 时，两种算法的能耗曲线变得平缓，这是因为风险数据在式(15)小范围内变化时，风险率会发生显著变化，能耗亦会以相同的速度变化。总之，不同的风险系数对能耗产生不同的影响。

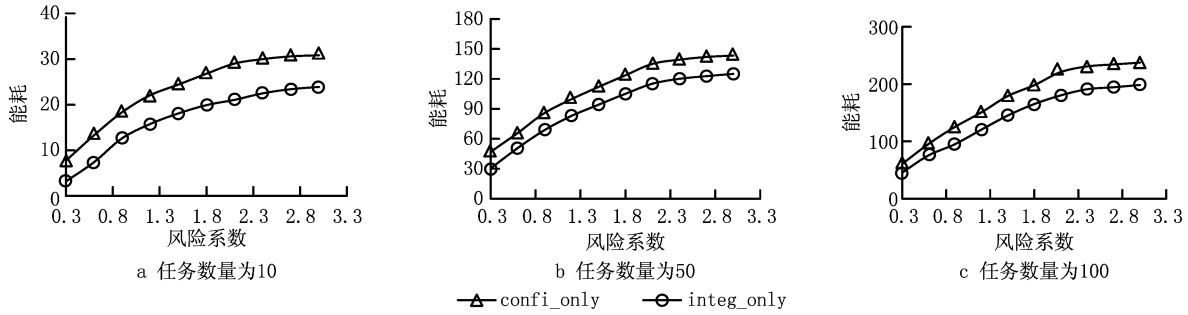


图8 不同风险系数的能耗

4.4 服务器数量的影响

本节考虑 MEC 服务器数量变化对 SEA 算法的影响，工作流的数量分别取 10、50 和 100，3 个工作流记为 SEA_10、SEA_50 和 SEA_100，风险率和风险系数分别为 0.3 和 1.5，其中 MEC 服务器的数量变化范围为[0,9]。实验结果如图 9 所示，服务器数量越多，移动设备的能耗越低，当服务器的数量达到 5 时，能耗曲线变得平缓，这是因为服务器受工作流的结构限制，当服务器的数量大于工作流的并发量时导致部分服务器空闲。因此在进行调度算法时，需要选择适量的服务器。

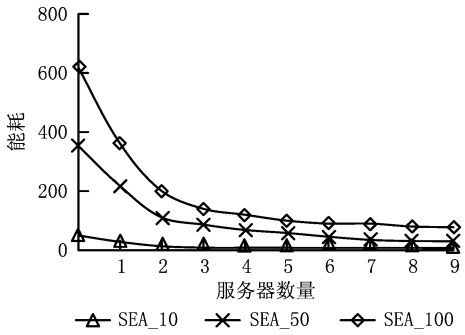


图9 不同服务器数量的能耗

4.5 实例验证

本节采用面部表情识别流程作为实例验证所提算法的可用性。算法的能耗是指手机运行面部表情识别流程消耗的电量，单位：mAh。面部表情服务流程由 8 个任务组成：输入视频、人脸检测、脸部特征检测、脸部标记、脸部特征提取、面部动作单元识

别、表情识别、输出类别[31]。这 8 个任务可以组织成一个带有数据依赖的工作流,工作流中每个任务可以在本地执行或者卸载到服务器上执行。其中任务 1 和任务 8 分别是工作流开始任务和结束任务,这两个任务必须在本地执行。实现了 4.1 节的能耗对比、4.2 节的安全服务对比和 4.3 节的风险系数的实验,实验结果证明了 SEA 算法的可用性,实验效果如图 10 所示。

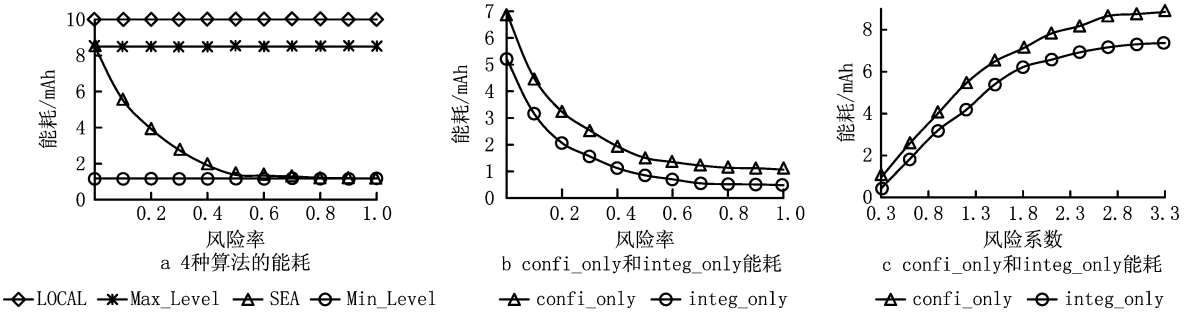


图10　面部表情识别流程的能耗

5　结束语

本文提出一种在 MEC 环境中面向安全和能耗感知的工作流调度算法。该算法在 PSO 算法的基础上融合了工作调度问题,目标是在满足工作流的截止时间和风险率两个约束条件下,最小化移动设备的能耗。此外,还提出一种新的工作流编码策略,解决了多维约束优化问题。通过仿真实验和实例对算法进行评估,验证了算法的有效性和实用性。

本文还有需要改进的地方,如在调度过程中均假设任务的执行都是成功情况,没有考虑任务失败的情况。因此,调度过程中的任务容错问题将是未来的研究工作之一。

参考文献:

[1] CHEN Xu, JIAO Lei, LI Wenzhong, et al. Efficientmulti-user computation offloading for mobile-edge cloud computing[J]. IEEE/ACM Transactions on Networking, 2016, 24(5): 2795-2808.

[2] HU Y C, PATEL M, SABELLA D, et al. Mobile edge computing—A key technology towards 5G[EB/OL]. [2018-11-01]. http://cslabcms.nju.edu.cn/problem_solving/images/4/4b/Mobile_Edge_Computing_-_A_Key_Technology_towards_5G.pdf.

[3] MACH P, BECVAR Z. Mobileedge computing: a survey on architecture and computation offloading[J]. IEEE Communications Surveys & Tutorials, 2017, 19(3): 1628-1656.

[4] ABBAS N, ZHANG Y, TAHERKORDI A, et al. Mobileedge computing: A survey[J]. IEEE Internet of Things Journal, 2018, 5(1): 450-465.

[5] LI Zhongjin, GE Jidong, LI Chuanyi, et al. Energy cost minimization with job security guarantee in Internet data center [J]. Future Generation Computer Systems, 2016, 73(C): 63-78.

[6] LIU C F, BENNIS M, POOR H V. Latency andreliability-aware task offloading and resource allocation for mobile edge computin[EB/OL]. (2017-10-02)[2018-11-01]. https://arxiv.org/pdf/1710.00590.pdf.

[7] JIANG W, JIANG K, ZHANG X, et al. Energyoptimization of security-critical real-time applications with guaranteed security protection[J]. Journal of Systems Architecture, 2015, 61(7): 282-292.

[8] LI Z, YU J, HU H, et al. Fault-tolerant scheduling for scientific workflow with task replication method in cloud[EB/OL]. [2018-11-01]. https://www.scitepress.org/Link.aspx?doi=10.5220/0006687300950104.

[9] CHEN Huangke, ZHU Xiaomin, QIU Dishan, et al. Scheduling forworkflows with security-sensitive intermediate data by selective tasks duplication in clouds[J]. IEEE Transactions on Parallel & Distributed Systems, 2017, 28(9): 2674-2688.

[10] ZHU Zhaomeng, ZHANG Gongxuan, LI Miqing, et al. Evolutionary multi-objective workflow scheduling in cloud[J]. IEEE Transactions on Parallel & Distributed Systems, 2016, 27(5): 1344-1357.

[11] JIA M, CAO J, YANG L. Heuristic offloading of concurrent tasks for computation-intensive applications in mobile cloud computing[C]//Proceedings of Computer Communications Workshops. Washington, D.C., USA: IEEE, 2014: 352-357.

[12] CAO Bin, WANG Xiaotong, XIONG Lirong, et al. Searching method for partical swarm optimization of cloud workflow scheduling with time constraint[J]. Computer IntegratedManufacturing Systems, 2016, 22(2): 372-380(in Chinese). [曹斌,王小统,熊丽荣,等. 时间约束云工作流调度的粒子群搜索方法[J]. 计算机集成制造系统, 2016, 22(2): 372-380.]

[13] ZHU Yemao, LI Zhongjin, GE Jidong, et al. Multi-objective workflow scheduling algorithm based on delay transmission mechanism in mobile cloud computing environment[J]. Jour-

nal of Software,2018,29(11):3306-3328.[周业茂,李忠金,葛季栋等.移动云计算环境下基于延时传输机制的多目标工作流调度方法[J].软件学报,2018,29(11):3306-3328.]

[14] XIE Tao, QIN Xiao. Security-aware resource allocation for real-time parallel jobs on homogeneous and heterogeneous clusters[J]. IEEE Transactions on Parallel & Distributed Systems,2007,19(5):682-697.

[15] ZENG L, VEERAVALLI B, LI X. SABA:A security-aware and budget-aware workflow scheduling strategy in clouds[J]. Journal of Parallel & Distributed Computing, 2015, 75: 141-151.

[16] LI Zhongjin, GE Jidong, HU Hongji, et al. A security and cost aware scheduling algorithm for heterogeneous tasks of scientific workflow in clouds[J]. Future Generation Computer Systems,2016,65:140-152.

[17] LIU J, MAO Y, ZHANG J, et al. Delay-optimal computation task scheduling for mobile-edge computing systems[C]// Proceedings of IEEE International Symposium on Information Theory. Washington,D. C. ,USA:IEEE,2016:1451-1455.

[18] MAO Y, ZHANG J, LETAIEF K B. Dynamic computation offloading for mobile-edge computing with energy harvesting devices[J]. IEEE Journal on Selected Areas in Communications,2016,34(12):3590-3605.

[19] LYU Xinchen, NI Wei, TIAN Hui, et al. Optimalschedule of mobile edge computing for internet of things using partial information[J]. IEEE Journal on Selected Areas in Communications,2017,35(11):2606-2615.

[20] MAO Yuyi, YOU Changsheng, ZHANG Jun, et al. Asurvey on mobile edge computing:the communication perspective [J]. IEEE Communications Surveys & Tutorials, 2017, 19 (4):2322-2358.

[21] ZHANG Jiale, CHEN Bing, ZHAO Yanchao, et al. Datasecurity and privacy-preserving in edge computing paradigm: survey and open issues [J]. IEEE Access, 2018, 6: 18209-18237.

[22] ROMAN R, LOPEZ J, MAMBO M. Mobile edge computing, fog et al. :A survey and analysis of security threats and challenges[J]. Future Generation Computer Systems,2016, 78:680-698.

[23] XIE T, QIN X. Schedulingsecurity-critical real-time applications on clusters[J]. IEEE Transactions on Computers, 2006,55(7):864-879.

[24] BARNES A, FERNANDO R, METTANANDA K, et al. Improving the throughput of the AES algorithm with multicore processors[EB/OL]. [2018-11-01]. https://arxiv.org/ftp/arxiv/papers/1403/1403.7295.pdf.

[25] XIE Tao, QIN Xiao. Performance evaluation of a new scheduling algorithm for distributed systems with security heterogeneity[J]. Journal of Parallel and Distributed Computing, 2007,67(10):1067-1081.

[26] TANG X, LI K, ZENG Z, et al. A novel security-driven scheduling algorithm for precedence-constrained tasks in heterogeneous distributed systems[J]. IEEE Transactions on Computers,2011,60(7):1017-1029.

[27] ZHANG Xiaodong,LI Xiaoping, WANG Qian,et al. Hybrid particle swarm optimization algorithm for cost minimization in service-workflows with due dates[J]. Journal on Communications,2008,29(8):87-94(in Chinese).[张晓东,李小平,王茜,等.服务工作流的混合粒子群调度算法[J].通信学报,2008,29(08):87-93.]

[28] PANDEY S, WU L, GURU S M, et al. Aparticle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments [C]//Proceedings IEEE International Conference on Advanced Information Networking and Applications. Washington, D. C. , USA: IEEE, 2010:400-407.

[29] DEB K. An efficient constraint handling method for genetic algorithms[J]. Computer Methods in Applied Mechanics & Engineering,2000,186(2):311-338.

[30] COELLO C A C, PULIDO G T, LECHUGA M S. Handling multiple objectives with particle swarm optimization [J]. IEEE Transactions on Evolutionary Computation, 2004, 8 (3):256-279.

[31] DENG S, HUANG L, TAHERI J, et al. Computation offloading for service workflow in mobile cloud computing[J]. IEEE Transactions on Parallel & Distributed Systems,2015, 26(12):3317-3329.

作者简介:

李万清(1979－),男,辽宁抚顺人,副教授,博士,研究方向:数据分析、深度学习、工作流调度,E-mail:liwanqing@hdu.edu.cn;

刘　辉(1993－),男,江苏泰州人,硕士研究生,研究方向:移动边缘计算、分布式计算;

＋**李忠金**(1986－),男,江苏句容人,讲师,博士,研究方向:云计算、工作流调度,通信作者,E-mail:lizhongjin@hdu.edu.cn;

袁友伟(1966－),男,湖北潜江人,教授,博士,研究方向:云计算、工作流调度。