计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.190900054

边缘计算环境下服务质量可信的任务迁移节点选择

王 妍[1,2] 韩 笑[1] 曾 辉[1] 刘荆欣[1] 夏长清[2,3,4]
1 辽宁大学信息学院 沈阳 110036
2 中国科学院沈阳自动化研究所机器人学国家重点实验室 沈阳 110016
3 中国科学院沈阳自动化研究所网络化控制系统重点实验室 沈阳 110016
4 中国科学院机器人与智能制造创新研究院 沈阳 110169
(wang_yan@lnu.edu.cn)

摘 要 随着物联网、大数据和5G网络的快速发展及应用,传统的云计算模式已无法高效处理网络边缘设备所产生的海量计算任务,边缘计算应运而生。边缘计算环境下,计算任务将被迁移到接近数据源的计算设备上执行,这为拓展终端节点资源以及缓解云中心负载提供了新的解决方案。现有的任务迁移决策均是在任务迁移节点确定的前提下制定的,并未考虑存在多个任务迁移节点可选的情景,而边缘计算下任务迁移节点的选择直接影响着任务迁移的服务质量,因此文中构建了服务质量可信模型,分别从时间可信、行为可信、资源可信3个维度对任务迁移节点进行评价。为了解决任务迁移节点数量巨大带来的选择效率低的问题,采用基于聚类编码的skyline查询算法对任务迁移节点进行筛选,并利用灰色关联分析法进行任务迁移节点的最终选择。实验结果表明,所提基于服务质量可信的任务迁移节点选择策略的任务迁移成功率平均提高了36%,任务完成吞吐量平均提高了18%。

关键词:边缘计算;任务迁移;迁移节点选择;skyline查询;灰色关联分析法
中图法分类号 TP302

Task Migration Node Selection with Reliable Service Quality in Edge Computing Environment

WANG Yan[1,2], HAN Xiao[1], ZENG Hui[1], LIU Jing-xin[1] and XIA Chang-qing[2,3,4]
1 College of Information, Liaoning University, Shenyang 110036, China
2 State Key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China
3 Key Laboratory of Networked Control System, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China
4 Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences, Shenyang 110169, China

Abstract With the rapid development and wide application of the Internet of things, big data and 5G network, the traditional cloud computing mode has been unable to efficiently handle the massive computing tasks generated by network edge devices, so edge computing came into being. Computing tasks in edge computing environments will be migrated to computing devices close to data sources for execution, providing new solutions for expanding terminal node resources and alleviating cloud center load. The existing task migration decisions are made on the premise that the task migration node is determined, without considering the situation that multiple task migration nodes are available. The selection of the task migration node in edge computing directly affects the service quality of task migration, so, in this paper, a service quality trust model is constructed to evaluate the task migration nodes from three dimensions: time trust, behavior trust and resource trust. In order to avoid the problem of low selection efficiency caused by the large number of task migration nodes, a skyline query algorithm based on cluster coding is adopted to screen the task migration nodes, and grey relative analysis is used for the final selection of task migration nodes. The experimental results show that the proposed task migration node selection strategy based on reliable service quality can increase the success rate of task migration by 36% and the throughput of task completion by 18% on average.

Keywords Edge computing, Task migration, Migration node selection, Skyline query, Grey relative analysis method

到稿日期:2019-09-07 返修日期:2020-01-06 本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:国家重点研发计划(2019YFB1406002);国家自然科学基金(61903356);机器人学国家重点实验室开放基金(2019-O22);辽宁省自然科学基金计划重点项目(20180520029)
This work was supported by the National Key R&D Program of China (2019YFB1406002), National Natural Science Foundation of China (61903356), State Key Laboratory of Robotics (2019-O22) and National Natural Science Foundation of Liaoning province (20180520029).
通信作者:夏长清(xiachangqing@sia.cn)

近年来，随着万物互联时代的到来和无线网络的普及，网络边缘设备的数量和数据规模都快速增长[1]，这导致传统集中式处理模型下将任务交给云中心处理方式的不足日益突出。传统的云计算模式呈现出了 3 方面不足：1）海量边缘设备加入物联网中加重了云中心的计算负载，导致其计算能力不足；2）终端设备产生的海量计算任务全部传输到云中心，增加了传输带宽的负载量，增加了网络延迟；3）个人隐私数据的保护[2]与安全性问题尤为突出。边缘计算的出现为解决上述问题提供了新的视角和切入点。边缘计算是在网络边缘执行计算的一种新型计算模型[3]，其将计算存储能力下移至网络边缘，就近提供服务[4]。边缘计算涉及 3 个层次的对象：唯一远程资源丰富的云中心、一定数量资源相对丰富的边缘服务器、数量巨大但资源有限的终端节点。计算任务可以迁移到任意一层设备中进行处理。现有的任务迁移策略均是在任务迁移节点已经确定的前提下制定的，并未考虑到多任务迁移节点可选的情景。为了寻求更高的服务质量，本文构建了服务质量可信模型，分别以时间可信、行为可信、资源可信为切入点对任务迁移节点进行评价，并采用基于聚类编码的轻量型 skyline 查询算法对任务迁移节点进行筛选，最后利用灰色关联分析法求得最优的任务迁移节点。

1　相关工作

物联网的快速发展导致传统云计算模式的弊端逐渐显露，边缘计算逐渐走进人们的视野。目前，许多学者对边缘计算的架构以及安全问题展开了深入研究，如文献[5-7]分别对边缘计算的系统架构进行了设计，旨在与现有的云计算架构无缝融合；文献[8-10]对边缘计算下的安全问题进行了研究，旨在使边缘计算在安全的环境中运行。边缘计算环境可以将本地计算任务迁移到其他设备上执行，因此任务迁移问题也逐渐成为了学者们研究的热点。

目前，任务迁移领域的主要研究内容是任务迁移决策。Jia 等[11]以能耗为优化目标，并利用拉格朗日松弛算法 LARAC 判断任务是否迁移。此迁移策略能极大地节约能耗；但仅考虑了能耗因素，并未顾及时间要求。Wang 等[12]利用李雅普诺夫优化理论，以能耗为优化目标决定任务是本地执行还是迁移执行。李雅普诺夫优化理论可以最优化迁移决策；但算法复杂度高，且执行时间长，不适用于边缘计算下需要实时决策的场景。Mao 等[13]提出了另一种基于 Lyapunov 优化的动态任务迁移算法，旨在缩短应用程序的执行时延。此迁移策略同样仅考虑了时间因素，而未参考其他因素。文献[14]提出的 MRA 选择策略把迁移距离和资源作为决定因素进行迁移位置决策。此迁移策略虽然考虑了多维因素，且算法复杂度相对较小；但没有考虑到边缘计算环境下存在的节点不可信问题。综上，现有的任务迁移策略存在 3 方面问题：决策依据过于单一，算法复杂度高，没有考虑边缘计算环境下节点不可信的问题。

边缘计算下可选迁移节点的数量影响着选择的效率，因此本文采用 skyline 查询算法对迁移节点进行初步筛选。但是，现有的 skyline 查询算法并不能与边缘计算很好地融合。例如，文献[15]提出的 BNL 算法只是在 skyline 集较小的情况下效率极高，并不适用于边缘计算下迁移节点数量巨大的场景；文献[16]提出的基于排序的 skyline 算法缩短了算法的执行时间，但算法复杂度高，不适用于节点资源有限的场景；文献[17]提出的基于海量不完备数据集的 skyline 偏好查询策略对不完备数据集的 skyline 查询的效率极高，并且算法的时间复杂度较小，但本文涉及的任务迁移节点的各项参数的缺失率很低，因此该 skyline 查询算法并不适用于本文的任务迁移节点的筛选工作。

2　问题定义

边缘计算下的任务迁移模型如图 1 所示。边缘计算下的任务迁移包括上行迁移和下行迁移：上行迁移指终端节点因自身资源不足将计算任务迁移到云中心、边缘服务器或其他终端节点；下行迁移指负载过大的云中心将部分计算任务迁移到边缘服务器或终端节点。

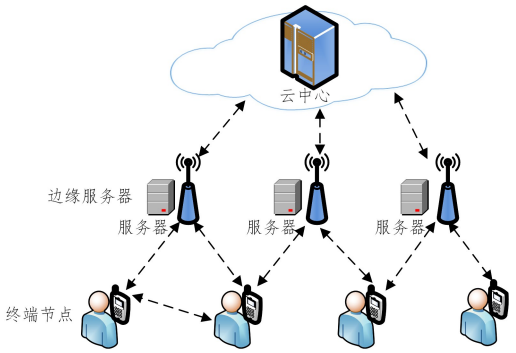

图 1　边缘计算下的任务迁移模型

Fig. 1　Task migration model in edge computing

对于边缘计算下存在多个可选的迁移节点的情况，需要解决以下几方面问题。

（1）边缘计算下存在大量实时性任务，如自动驾驶等，如何判断所选的任务迁移节点能否及时完成该计算任务？

（2）边缘计算下的任务迁移处于不可信的环境，那么如何确定任务迁移节点的可信性？

（3）任务迁移节点的资源宝贵，如何合理利用资源以免造成资源浪费？

（4）边缘计算下任务迁移源节点的资源有限，如何选择轻量型任务迁移节点选择方法？

基于上述问题，亟需构建一个迁移节点的评价模型对迁移节点进行综合评价，并选择合适的策略来基于评价指标选择出最优的任务迁移节点。

3　服务质量可信模型的构建

针对现有迁移决策的决策依据单一以及边缘计算下任务迁移存在的问题，本文构建了迁移节点服务质量可信模型，分别从时间可信、行为可信、资源可信 3 个角度对迁移节点进行评价。

3.1　时间可信模型

由于边缘计算下存在大量实时性计算任务，为了判断所选迁移节点能否处理此类任务，本文提出时间可信模型。该模型通过计算本地处理的时间与任务迁移处理的时间差值来

确定迁移节点的时间可信度。

任务本地处理的时间 T_{loc} 包括等待时间 T_{wait} 和执行时间 T_{exe}^{loc}。本地设备资源有限，不能同时处理多个任务，因此当有多个任务产生时会出现排队等待的情况，排队的顺序与本地调度策略有关。假设用 W_{wt} 表示先于当前任务执行的 CPU 周期数（W_{wt} 的值可为 0），W_{task} 表示当前计算任务所需 CPU 周期数，C_{loc} 表示本地设备的计算能力，则任务在本地处理的时间如式(1)所示：

$$\begin{cases} T_{wait}=\dfrac{W_{wt}}{C_{loc}} \\ T_{exe}^{loc}=\dfrac{T_{task}}{C_{loc}} \\ T_{loc}=T_{wait}+T_{exe}^{loc} \end{cases} \tag{1}$$

任务迁移处理的时间包括传输时间和任务执行时间。由于迁移节点有足够的资源处理迁移来的任务，因此不用考虑等待时间，但是需要考虑任务的传输时间。假设分别用 W_{task}，C_i，L_{task} 和 bw_{worst} 表示任务所需 CPU 的周期数、迁移节点 i 的计算能力、任务的大小以及网络带宽，为了避免网络波动对传输时间的预估造成影响，网络带宽的取值为 worst case 下的数值并用 bw_{worst} 表示，则任务迁移到节点 i 处理的时间如式(2)所示：

$$\begin{cases} T_{trans}=\dfrac{L_{task}}{bw_{worst}} \\ T_{exe}^{off}=\dfrac{W_{task}}{C_i} \\ T_i=T_{exe}^{off}+T_{trans} \end{cases} \tag{2}$$

基于式(1)和式(2)可以求出任务在本地处理的时间和任务迁移处理的时间。迁移节点 i 的时间可信度 T_i^{trust} 可以通过两者的差值求得，如式(3)所示：

$$T_i^{trust}=T_{loc}-T_i \tag{3}$$

从式(3)可以看出，任务迁移到节点 i 处理的时间相对于本地处理的时间越短，则节点 i 的时间可信度就越高。

3.2 行为可信模型

因为边缘计算下的任务迁移处于不可信的环境，所以需要考虑迁移节点的行为可信问题。行为可信度高的迁移节点能够高效地完成迁移来的任务，而行为可信度低的迁移节点可能会存在故意作恶的情况，因此本文提出行为可信模型以从主观可信和推荐可信两个维度对迁移节点进行综合考量。

3.2.1 主观可信

主观可信是源节点 s 依据与迁移节点 i 的历史交互记录，计算出迁移节点 i 的主观可信度。主观可信度的判定分为两个阶段：行为检测和主观可信度的计算。

(1)行为检测

源节点 s 根据迁移节点 i 为自己处理事务的结果建立一系列行为检测方法和参数统计规则，为主观可信度的计算提供数据依据。设 $\alpha\langle i,s\rangle$ 和 $\beta\langle i,s\rangle$ 分别表示迁移节点 i 处理来自源节点 s 的任务的正确数量和错误数量，当任务计算正确时 $\alpha\langle i,s\rangle$ 增加，否则 $\beta\langle i,s\rangle$ 增加。$\rho\langle i,s\rangle$ 和 $\upsilon\langle i,s\rangle$ 表示迁移节点 i 为源节点 s 转发任务时的成功数量和失败数量，当转发任务成功时 $\rho\langle i,s\rangle$ 增加，否则 $\upsilon\langle i,s\rangle$ 增加。

(2)主观可信度的计算

源节点 s 根据行为检测阶段收集到的两种行为的证据信息计算迁移节点 i 的主观可信度 $T_{\langle i,s\rangle}^{d}$。我们将主观可信度分为任务计算可信度 $T_{\langle i,s\rangle}^{c}$ 和任务转发可信度 $T_{\langle i,s\rangle}^{t}$ 两部分，相关计算公式如下：

$$\begin{cases} S_{\langle i,s\rangle}^{c}=\dfrac{\alpha_{\langle i,s\rangle}}{\alpha_{\langle i,s\rangle}+\beta_{\langle i,s\rangle}} \\ F_{\langle i,s\rangle}^{c}=\dfrac{\beta_{\langle i,s\rangle}}{\alpha_{\langle i,s\rangle}+\beta_{\langle i,s\rangle}} \\ U_{\langle i,s\rangle}=\dfrac{\alpha_{\langle i,s\rangle}}{(\alpha_{\langle i,s\rangle}+\beta_{\langle i,s\rangle})^2+(\alpha_{\langle i,s\rangle}+\beta_{\langle i,s\rangle}+1)} \end{cases} \tag{4}$$

$S_{\langle i,s\rangle}^{c}$，$F_{\langle i,s\rangle}^{c}$，$U_{\langle i,s\rangle}$ 分别表示迁移节点 i 处理源节点 s 的计算任务的正确率、错误率以及容错因子。为了避免节点在处理计算任务时因外力因素造成处理任务失败的影响，引入了容错因子 $U_{\langle i,s\rangle}$ 的概念。$U_{\langle i,s\rangle}$ 表示节点交互中由于非恶意因素导致交互失败的概率。本文依据 D-S 证据理论[18]以及在实际应用中的特定场景给出了容错因子的量化表示。

任务计算可信度 $T_{\langle i,s\rangle}^{c}$ 的计算公式如式(5)所示：

$$T_{\langle i,s\rangle}^{c}=S_{\langle i,s\rangle}^{c}-F_{\langle i,s\rangle}^{c}(1-U_{\langle i,s\rangle}) \tag{5}$$

$$\begin{cases} S_{\langle i,s\rangle}^{t}=\dfrac{\rho\langle i,s\rangle}{\rho\langle i,s\rangle+\upsilon\langle i,s\rangle} \\ F_{\langle i,s\rangle}^{t}=\dfrac{\upsilon\langle i,s\rangle}{\rho\langle i,s\rangle+\upsilon\langle i,s\rangle} \\ V_{\langle i,s\rangle}=\dfrac{\rho\langle i,s\rangle}{(\rho_{\langle i,s\rangle}+\upsilon_{\langle i,s\rangle})^2+(\rho_{\langle i,s\rangle}+\upsilon_{\langle i,s\rangle}+1)} \end{cases} \tag{6}$$

$S_{\langle i,s\rangle}^{t}$，$F_{\langle i,s\rangle}^{t}$，$V_{\langle i,s\rangle}$ 分别表示任务迁移节点 i 为源节点 s 转发任务的成功率、失败率以及容错因子，则任务转发可信度 $T_{\langle i,s\rangle}^{t}$ 如式(7)所示：

$$T_{\langle i,s\rangle}^{t}=S_{\langle i,s\rangle}^{t}-F_{\langle i,s\rangle}^{t}(1-V_{\langle i,s\rangle}^{t}) \tag{7}$$

根据任务计算可信度和任务转发可信度，可以计算出源节点 s 对迁移节点 i 的主观可信度 $T_{\langle i,s\rangle}^{d}$：

$$T_{\langle i,s\rangle}^{d}=T_{\langle i,s\rangle}^{c}\times T_{\langle i,s\rangle}^{t} \tag{8}$$

3.2.2 推荐可信

推荐可信由源节点 s 根据推荐节点 j（$1\leqslant j\leqslant n$）反馈的 i 的可信度 $T_{\langle i,j\rangle}^{r}$ 进行计算，$T_{\langle i,j\rangle}^{r}$ 的值为节点 j 对节点 i 的主观可信度。为了避免推荐节点的恶意推荐和推荐节点的联合欺骗，引入推荐权重 w_j 来表示节点 j 推荐的可信度 $T_{\langle i,j\rangle}^{r}$ 的可信性。推荐权重的计算式如式(9)所示：

$$\begin{cases} w_j=\lambda_1 T_{\langle j,s\rangle}+\lambda_2 D_j+\lambda_3 D'_j \\ D_j=1-\dfrac{|T_{\langle i,j\rangle}^{r}-\sum_{j=1}^{n}T_{\langle i,j\rangle}^{r}\div n|}{2} \\ D_j'=1-\dfrac{|T_{\langle i,j\rangle}^{r}-T_{\langle i,s\rangle}^{d}|}{2} \end{cases} \tag{9}$$

其中，$T_{\langle j,s\rangle}$ 表示源节点 s 对推荐节点 j 的信任度，D_j 表示节点 j 推荐的可信度与所有节点推荐的可信度的均值的贴近度，D_j' 表示节点 j 推荐的可信度与源节点 s 对节点 i 的主观可信度的贴近度；λ_1，λ_2，λ_3 为系数，且 $\lambda_1+\lambda_2+\lambda_3=1$。结合多个推荐节点反馈的 i 的可信度可以得到迁移节点 i 的推荐可信度 T_i^{r}：

$$T_i^{r}=\frac{\sum_{j=1}^{n}w_j * T_{\langle i,j\rangle}^{r}}{\sum_{j=1}^{n}w_j} \tag{10}$$

为了使行为可信度的计算更为准确,不仅要考虑当前主观可信和推荐可信的值,还要将源节点 s 上一次对节点 i 的行为可信度的值考虑进来(假设源节点 s 有记录行为可信度值的记录表)。迁移节点 i 的行为可信度如式(11)所示:

$$B_{\langle i,s\rangle}^{trust}=\zeta^{\Delta t}T_{\langle i,s\rangle}^{k-1}+(1-\zeta^{\Delta t})\lfloor \omega T_{\langle i,s\rangle}^{d}+(1-\omega)T_i^r \rfloor (0<\zeta<1,0<\omega<1) \tag{11}$$

其中,$T_{\langle i,s\rangle}^{k-1}$ 表示上一次节点 i 的行为可信度的值,如果当前对迁移节点 i 的行为可信度的评价为首次,则 $T_{\langle i,s\rangle}^{k-1}$ 为 0;$\zeta^{\Delta t}$ 表示上一次行为可信度的值的权重;ω 控制着主观可信度和推荐可信度的权重,可以根据实际情况进行动态调整。

3.3　资源可信模型

对迁移节点资源的合理利用有利于任务迁移活动的良性运行。在选择任务迁移节点时,如果不考虑资源的合理利用,则不可避免地会造成如下几种后果:1)频繁使用资源丰富的迁移节点,导致资源需求较大的任务无处迁移;2)迁移节点的剩余资源碎片化且资源碎片很难被利用,造成资源浪费;3)资源余量失衡,例如节点的计算资源几乎耗尽而存储资源剩余较多,迁移节点的某一项资源不足便不能接受迁移任务,这会造成资源的浪费。为了避免此类事件的发生,发起任务的源节点都要遵循合理利用资源的规则,某个源节点如果打破了规则,则会受到惩罚并影响下一次的任务迁移活动。基于此规则,源节点进行迁移节点选择时应尽量合理利用资源。本文设计了资源可信模型,该模型分为资源均衡模型和资源供需匹配模型。

3.3.1　资源均衡模型

定义 1(资源均衡度 RBD)　迁移节点上资源的均衡状况。RDB 越小,表示此迁移节点对资源的使用越均衡。迁移节点 i 在接受迁移任务前的资源均衡度如式(12)所示:

$$RBD_i=\left|\frac{R_i^c}{R_i^c+R_i^m}-\xi\right| \tag{12}$$

其中,R_i^c 和 R_i^m 分别表示该节点的计算、存储资源的剩余数量;ξ 为调控因子,通常情况下取值为 0.5。

迁移节点 i 在接受迁移任务后的资源均衡度如式(13)所示:

$$RBD_i^{\text{after}}=\left|\frac{R_i^{c,\text{after}}}{R_i^{c,\text{after}}+R_i^{m,\text{after}}}-\xi\right| \tag{13}$$

定义 2(资源均衡度变化差 RBDV)　任务迁移前后迁移节点的资源均衡度的差值。RBDV 为负数时,表明任务迁移到此节点后使该节点资源的使用更加均衡。RBDV 的计算公式如式(14)所示:

$$RBDV_i=RBD_i^{\text{after}}-RBD_i \tag{14}$$

3.3.2　资源供需匹配模型

资源均衡模型可以保证迁移节点的资源被均衡使用,但不能解决资源碎片化以及频繁使用资源丰富的节点导致资源需求较大的任务无处可迁的问题。为解决此问题,本文提出了资源供需匹配模型。资源供需匹配度是任务对资源的需求量与迁移节点 i 的资源剩余量的相近度,供需匹配度越小则表明资源利用越充分。资源供需匹配度的计算公式如式(15)所示:

$$RMD_i=\sqrt{\sum_{k\in\{c,m\}}(R_t^k-R_i^k)^2} \tag{15}$$

其中,R_t^k 表示任务对资源的需求量,R_i^k 表示节点 i 的资源剩余量。

结合资源均衡变化差 RBDV 与供需匹配度 RMD,我们可以得到资源可信度 R^{trust}:

$$R_i^{\text{trust}}=\frac{1}{\sqrt{\sum_{k\in\{c,m\}}e^{RBDV_i}(R_t^k-R_i^k)^2}} \tag{16}$$

4　任务迁移节点的选择

4.1　基于聚类编码 skyline 的节点筛选方法

4.1.1　迁移节点的筛选策略

迁移节点数量巨大会降低迁移节点选择的效率,因此本文采用 skyline 查询算法对迁移节点进行初步筛选。由于传统的 skyline 查询算法的复杂度高且查询效率低,并不适用于边缘计算下迁移节点的筛选工作,因此本文对传统 skyline 查询算法进行改进,提出了基于聚类编码的 skyline 查询算法 CBS。CBS 通过聚类操作减少了大量不必要的支配比较,因此可以提高筛选效率。

定义 3(聚类编码)　给定一个元组,如果其某一维度上的值不小于此维度上的均值,则该维度上的编码为 1,否则编码为 0。

定义 4(支配关系)　元组 p 支配元组 q,当且仅当 p 在所有维度上的值都不比 q 差,且在至少一个维度上 p 比 q 优。本文采用值大为优的支配策略。

迁移节点的筛选工作分为两个阶段。

阶段 1:根据元组的聚类编码将集合中的元组分为 8 个聚类集合,聚类编码集合为{111,110,101,011,100,010,001,000}。聚类编码为 000 的集合中的元组不可能成为 skyline 点,因此直接将该集合中的元组删除;对余下的 7 个聚类集合进行集合内 skyline 点的查找。

阶段 2:经过阶段 1 的处理,每个聚类集合中剩下本集合内的 skyline 点,然后将这些聚类集合按聚类码包含 1 的个数分为 3 个大的集合,分别为 U_3={Cluster 111},U_2={Cluster 110,Cluster 101,Cluster 011},U_1={Cluster 100,Cluster 010,Cluster 001}。U_3,U_2,U_1 集合内的元组不存在支配关系,因此只需在集合间进行 skyline 查询。U_2 和 U_1 两个集合中的元组不能支配 U_3 中的元组。U_2 中的元组只可能被 U_3 中的元组支配,即 $\forall p_i\in U_3$,如果 $\exists p_j\in U_2$ 且 p_i 支配 p_j,那么将 p_j 移出 U_2。U_3 和 U_2 中的元组都有可能支配 U_1 中的元组,即 $\forall p_i'\in U_3$,$\forall p_j'\in U_2$,$\exists p_k\in U_1$,p_i' 支配 p_k 或者 p_j' 支配 p_k,则将 p_k 移出 U_1,最后将所有剩余的元组放入 U' 中。

4.1.2　迁移节点的筛选算法及分析

迁移节点的筛选算法如算法 1 所示。

算法 1　基于聚类编码 skyline 的节点筛选算法

输入:服务质量可信模型处理后的 n 个迁移节点元组构成的集合 U;

输出:筛选后的迁移节点集合 U′

```
1. for(each tuple p∈U)
2. if (p[i]>ave_i)//p[i]表示 p 在第 i 维上的值
3.     p[i].code=1;
4. else
5. p[i].code=0;
```

```
6. for(each tuple p∈U)
7. 将元组 p 按编码放入对应的聚类集合 V, V∈{U111, U110, U101, U011, U100, U010, U001}
8. for(each tuple p,q∈V)
9. Int flag=dominate(p,q);//判断 p 和 q 的支配关系
10.   if flag==0//q 支配 p
11. delete p;
12. 将 U111 中的元组添加到 U3 中,将 U110, U101, U011 中的元组添加到 U2 中,将 U100, U010, U001 中的元组添加到 U1 中
13. for(each tuple p∈U3, each tuple q∈U2)
14. Int flag=dominate(q,p);
15. if flag==0//p 支配 q
16. delete q;
17. for(each tuple p∈U3, each tuple q∈U1)
18. Int flag=dominate(q,p);
19. if flag==0//p 支配 q
20. delete q;
21. for(each tuple p∈U2, each tuple q∈U1)
22. Int flag=dominate(q,p);
23. if flag==0//p 支配 q
24. delete q;
25. 把 U3, U2, U1 中的元组添加到 U'中
26. Return U';
```

根据算法 1 的描述,算法 1 先对集合 U 中的元组进行编码和聚类操作,时间复杂度为 $O(n)$;然后进行集合内 skyline 查询以及集合间 skyline 查询,时间复杂度均为 $O(n)$。因此,基于聚类编码 skyline 的节点筛选算法的时间复杂度为 $O(n)$。

4.1.3 迁移节点筛选算法的示例

假设有 10 个迁移节点,根据时间可信模型、行为可信模型、资源可信模型中的计算公式求得的结果如表 1 所列。

表 1 迁移节点示例

Table 1 Sample of migration nodes

tuple	T^{trust}	B^{trust}	R^{trust}
p_1	5	3	7
p_2	9	8	2
p_3	8	4	6
p_4	6	5	4
p_5	7	3	4
p_6	2	8	5
p_7	9	3	3
p_8	10	2	5
p_9	7	5	6
p_{10}	3	5	5

表 1 中的 10 个元组先经过算法 1 进行聚类编码以及聚类操作。例如,经过聚类编码操作将 p_1 编码为(0,0,1),然后通过聚类操作将 p_1 加入集合 U_{001}。所有的元组经过上述操作后的结果如表 2 所列。

表 2 聚类结果

Table 2 Clustering results

U_{111}	U_{110}	U_{101}	U_{011}	U_{100}	U_{010}	U_{001}	U_{000}
p_9	p_2	p_3, p_8	p_6, p_{10}	p_5, p_7	p_4	p_1	—

表 2 中的元组根据算法 1 中的支配比较算法进行集合内 skyline 查询,再进行集合间的 skyline 查询,最后得到全部元组的 skyline 点集合 $U'=\{p_3, p_7, p_8, p_9\}$。通过算法示例可以看出,原有的 10 个元组经过算法 1 处理之后变成了集合 U'中的 4 个元组。由此可见,基于聚类编码 skyline 的节点筛选算法可以有效地对迁移节点进行筛选。

4.2 基于灰色关联分析的最优节点选择

4.2.1 最优迁移节点的求解策略

对于 U'中存在多个 skyline 点的情况,即可筛选出多个候选迁移节点,我们采用灰色关联分析法求解最优迁移节点。

步骤 1 将集合 U'中的元组构成数列矩阵

$$X=\begin{bmatrix}X_1\\X_2\\\vdots\\X_n\end{bmatrix}=\begin{bmatrix}x_{11}&x_{12}&x_{13}\\x_{21}&x_{22}&x_{23}\\\vdots&\vdots&\vdots\\x_{n1}&x_{n2}&x_{n3}\end{bmatrix}$$

其中,$X_i=(x_{i1}, x_{i2}, x_{i3})$表示集合中的第 i 个元组。

步骤 2 确定参考数据行

参考数据行是一个理想的比较标准,因为在 skyline 点查询时采用大值支配小值的原则,所以理想的数据行应该为每个维度的最大值。参考数据行为 $X_0=(x_{01}, x_{02}, x_{03})$,其中 x_{01}, x_{02}, x_{03}分别为各自维度的最大值。然后,重新建立矩阵。

$$X=\begin{bmatrix}X_0\\X_1\\X_2\\\vdots\\X_n\end{bmatrix}=\begin{bmatrix}x_{01}&x_{02}&x_{03}\\x_{11}&x_{12}&x_{13}\\x_{21}&x_{22}&x_{23}\\\vdots&\vdots&\vdots\\x_{n1}&x_{n2}&x_{n3}\end{bmatrix}$$

步骤 3 数据归一化

$$Y=\begin{bmatrix}Y_0\\Y_1\\Y_2\\\vdots\\Y_n\end{bmatrix}=\begin{bmatrix}y_{01}&y_{02}&y_{03}\\y_{11}&y_{12}&y_{13}\\y_{21}&y_{22}&y_{23}\\\vdots&\vdots&\vdots\\y_{n1}&y_{n2}&y_{n3}\end{bmatrix}$$

$$y_{ij}=\frac{x_{ij}}{\max\{x_{1j}, x_{2j}, \cdots, x_{nj}\}}, 0\leqslant i\leqslant n, 1\leqslant j\leqslant 3 \tag{17}$$

迁移节点的评价指标具有不同的量纲单位,为了消除量纲不同的影响,本文采用数据归一化处理方式。

步骤 4 计算关联系数及关联度

关联系数如式(18)所示:

$$r_{ij}=\frac{\min\limits_i\min\limits_j|y_{ij}-y_{0j}|+\rho\max\limits_i\max\limits_j|y_{ij}-y_{0j}|}{|y_{ij}-y_{0j}|+\rho\max\limits_i\max\limits_j|y_{ij}-y_{0j}|} \tag{18}$$

其中,ρ 是分辨系数,在[0,1]内取值,通常取值为 0.5。

关联度的求解如式(19)所示:

$$r_{0i}=\frac{1}{3}\sum_{j=1}^{3}w_j r_{ij} \tag{19}$$

其中,w_j表示在迁移节点选择中各个指标所占的权重,可以根据当前任务的属性进行调整。

最后,将 U'中的元组按关联度降序排序,关联度最大的元组代表的节点为最终的任务迁移节点。

4.2.2 最优迁移节点的求解算法及分析

最优迁移节点的求解算法如算法 2 所示。

算法 2　最优迁移节点求解算法

输入：经过筛选后的迁移节点元组集合 U′

输出：最终的任务迁移节点

```
1. for(each tuple p∈U′)
2. maxi=max(p[i]),i=1,2,3//确定参考数据行
3. for(each tuple p∈U′)
4. p[i]=p[i]/maxi;//归一化
5. for(each tuple p∈U′)
6. 计算元组 p 的关联度 pr;
7. for(each tuple p∈U′)
8. if(p.pr>max)//p.pr 表示 p 的关联度
9. max=p.pr;
10.  return p′;//p′为关联度最大的元组
```

根据算法 2 的描述，首先求出参考数据行每一维度的值，并基于参考数据行对元组的每个维度值进行归一化处理，时间复杂度均为 $O(n)$；接着依据相关公式求出每个元组 p 的关联度并对所有关联度进行降序排序，返回最大值，时间复杂度也均为 $O(n)$。因此，基于灰色关联分析最优节点选择算法的时间复杂度为 $O(n)$。

4.2.3　最优迁移节点的求解算法示例

所有的迁移节点经过算法 1 筛选得到备选迁移节点集合 $U'=\{p_3,p_7,p_8,p_9\}$。U'中的每个元组根据式(18)得到各个维度的关联系数，结果如表 3 所列。

表 3　关联系数

Table 3　Correlation coefficients

元组	第 1 维关联系数	第 2 维关联系数	第 3 维关联系数
p_3	0.428	1	0.391
p_7	0.333	0.631	1
p_8	1	0.263	0.243
p_9	0.593	0.222	0.377

将集合 $U'=\{p_3,p_7,p_8,p_9\}$中元组的每一维度的关联系数作为参数代入式(19)，计算出每个元组的关联度，元组关联度如表 4 所列。

表 4　元组关联度

Table 4　Correlation degree of tuple

tuple	correlation degree
p_3	0.606
p_7	0.655
p_8	0.502
p_9	0.459

最后，将表 4 中元组的关联度降序排序为 0.655＞0.606＞0.502＞0.459，即元组 p_7 的关联度最大，因此选择元组 p_7 代表的节点为最终的任务迁移节点。从上述示例可以看出，最优迁移节点的求解算法的确可以选择出一个最优的迁移节点。

5　实验及分析

为了在一个模拟的边缘计算环境下测试本文提出的任务迁移节点选择策略的有效性，我们使用 CORE[19] 运行实验。CORE 中的虚拟节点包括各种交换机、路由器、边缘服务器、终端设备[20]。

实验主要从任务完成吞吐量和迁移成功率等方面对本文提出的任务迁移节点选择策略与文献[15]提出的 MRA 选择策略以及随机选取策略进行对比。任务完成吞吐量的实验结果如图 2 所示。

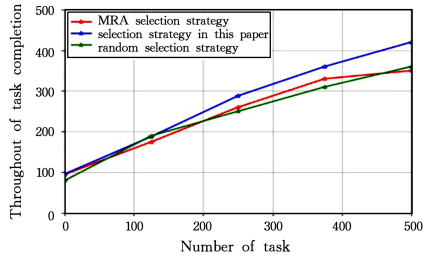

图 2　任务完成吞吐量

Fig. 2　Task completion throughput

从图 2 可以看出，当迁移的任务数量较少时，3 种选择策略的任务完成吞吐量几乎一致，这是因为当迁移的任务数量较少时，有足够多的迁移节点接受任务的迁移；随着迁移任务数量的不断增加，本文提出的选择策略的任务完成吞吐量比其他两种选择策略大，这是因为本文的任务迁移节点选择策略中提出的服务质量可信模型可以合理地利用各个迁移节点的资源。

如图 3 所示，随着边缘计算环境下恶意节点数量的增加，相比其他两种选择策略，本文提出的选择策略能更准确地识别出恶意节点，极大限度地避免了将任务迁移到恶意节点上，进而提高了任务迁移的成功率。

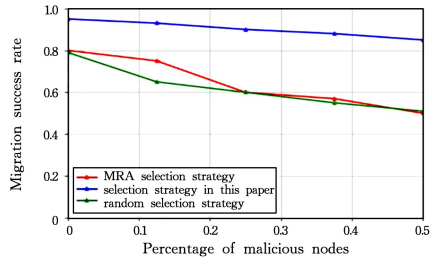

图 3　迁移成功率

Fig. 3　Migration success rate

本文还对迁移节点的筛选效率进行了测试，实验环境是单台 PC 机，PC 机的配置为 Intel Pentium CPU G3220 3.00 GHz，4 GB 内存。编程环境采用的是 Eclipse，编程语言为 Java。将本文提出的基于聚类编码 skyline 查询算法 CBS 与现有的 CRSQ 和 BBRS 算法进行筛选效率的比较，实验结果如图 4 所示。

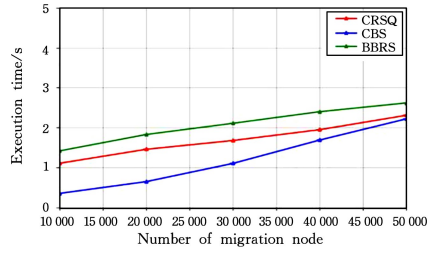

图 4　执行时间的比较

Fig. 4　Comparison of execution time

实验结果表明，本文提出的基于聚类编码 skyline 查询算法 CBS 在迁移节点数量为 50 000 以下的筛选效率明显优于

其他两种算法;虽然数量达到 50 000 之后其相对于其他两种算法的优势不是很明显,但实际应用中超过 50 000 个节点的情况极少。本文算法是轻量型的,在 50 000 个节点以内效果最好,因此 CBS 在实际应用中相对于其他两种算法更具有优势。

结束语 本文从边缘计算下任务迁移的特点出发,对任务迁移节点的选择策略进行研究,提出了一种边缘计算环境下基于服务质量可信模型的任务迁移节点选择方法。本文方法可以在边缘计算的非可信环境下,充分合理地利用迁移节点资源,实现计算任务的实时处理。实验结果表明,本文提出的基于服务质量可信的任务迁移节点选择方法提高了任务迁移成功率和任务完成吞吐量,保证了任务迁移的高质量运行。本文所研究的任务迁移节点选择策略是基于所有的迁移节点无私奉献自身资源的前提下提出的,对于边缘计算下任务迁移所涉及的利益问题还没有进行深入的研究,这将是未来的研究方向。

参 考 文 献

[1] SHI W S, ZHANG X Z, WANG Y F, et al. Edge Coputing: State-of-the-Art and Future Directions[J]. Journal of Computer Reaserch and Development, 2019, 56(1): 69-89.

[2] WANG B, LI B, HUI L. Oruta: privacy-preserving public auditing for shared data in the cloud[J]. IEEE Transactions on Cloud Computing, 2014, 2(1): 43-56.

[3] SHI W S, SUN H, CAO J, et al. Edge Coputing: An Emerging Computing Model for the Internet of Everything Era[J]. Journal of Computer Reaserch and Development, 2017, 54(5): 907-924.

[4] LV H Z, CHEN D, FAN B, et al. Standardization Progress and Case Analysis of Edge Computing[J]. Journal of Computer Reaserch and Development, 2018, 55(3): 487-511.

[5] SATYANARAYANAN M, ZHUO C, HA K, et al. Cloudlets: at the leading edge of mobile-cloud convergence[C]// International Conference on Mobile Computing. 2014: 1-9.

[6] GOSAIN A, BERMAN M, BRINN M, et al. Enabling Campus Edge Computing Using GENI Racks and Mobile Resources [C]// 2016 IEEE/ACM Symposium on Edge Computing (SEC). ACM, 2016: 41-50.

[7] NASTIC S, TRUONG H L, DUSTDAR S. A Middleware Infrastructure for Utility-Based Provisioning of IoT Cloud Systems [C]// Edge Computing. IEEE, 2016: 28-40.

[8] ESPOSITO C, CASTIGLIONE A, POP F, et al. Challenges of Connecting Edge and Cloud Computing: A Security and Forensic Perspective[J]. IEEE Cloud Computing, 2017, 4(2): 13-17.

[9] YANG K, JIA X, REN K, et al. DAC-MACS: Effective Data Access Control for Multiauthority Cloud Storage Systems[J]. IEEE Transactions on Information Forensics & Security, 2013, 8(11): 1790-1801.

[10] ROMAN R, LOPEZ J, MAMBO M. Mobile edge computing, Fog et al. A survey and analysis of security threats and challenges [J]. arXiv: 1602.00484, 2016.

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

[12] HUANG D, WANG P, NIYATO D. A Dynamic Offloading Algorithm for Mobile Computing[J]. IEEE Transactions on Wireless Communications, 2012, 11(6): 1991-1995.

[13] MAO Y, ZHANG J, LETAIEF K B. Dynamic Computation Offloading for Mobile-Edge Computing with Energy Harvesting Devices[J]. arXiv: 1605.05488, 2016.

[14] DENG X N, GUSN P Y, WAN Z W, et al. Integrated Trust Based Resource Cooperation in Edge Computing[J]. Journal of Computer Reaserch and Development, 2018, 55(3): 449-477.

[15] BÖRZSÖNYI S, KOSSMANN D, STOCKER K. The Skyline Operator[C]// International Conference on Data Engineering. 2002.

[16] LI Y Y, LI Z Y, DONG M X, et al. Efficient subspace skyline query based on user preference using MapReduce[J]. Ad Hoc Networks, 2015, 35: 105-115.

[17] YAN W, ZHAN S, WANG J, et al. Skyline Preference Query Based on Massive and Incomplete Dataset[J]. IEEE Access, 2017, 5(99): 3183-3192.

[18] LI N, DAS S K. A trust-based framework for data forwarding in opportunistic networks[J]. Ad Hoc Networks, 2013, 11(4): 1497-1509.

[19] AHRENHOLZ J. Comparision of CORE network emulation platforms[C]// Proc of MILCOM 2010. Piscataway, NJ: IEEE, 2010: 166-171.

[20] FIGUEROA M, UTTECHT K, ROSENBERG J. A SOUND approach to security in mobile and cloud-oriented environments [C]// IEEE International Symposium on Technologies for Homeland Security. 2015: 147-156.

WANG Yan, born in 1978, Ph.D, associate professor, is a member of China Computer Federation. Her main research interests include IIoT data processing, task scheduling and big data technology, etc.

XIA Chang-qing, born in 1985, Ph.D, research assistant, is a member of China Computer Federation. His main research interests include industry network and task scheduling in Edge Computing, etc.