多云环境的虚拟应用网络部署决策方法研究

朱华旻 吴礼发 赵 鹏
(解放军理工大学指挥信息系统学院 南京 210007)

摘 要 多云环境下,用户能够基于虚拟应用技术与基础设施虚拟技术自由组合基础设施资源以部署虚拟应用网络,并且能够快速构建具有一定业务功能的分布式应用系统。鉴于现有多云部署决策方法,以及在部署描述及处理用户多目标需求方面的不足,首先给出了基于开放虚拟格式文档的虚拟应用网络部署描述方法;其次研究定义了基础设施资源组合的常见服务质量指标及多目标优化模型,并使用第二代非支配排序遗传算法(NSGA-II)和多目标粒子群优化(MOPSO)算法求解模型;最后给出了一种基于模糊决策的最终满意解确定方法。多次实验统计表明,两种算法均能在合理时间内实现较好收敛,NSGA-II适合2~3个目标的场景,而MOPSO能够用于更多目标场景,且均有更好表现;模糊决策所得最终解能够最佳匹配用户的目标偏好。

关键词 多云,虚拟应用网络,基础设施服务,多目标优化,模糊决策
中图法分类号 TP393.09 **文献标识码** A

Research on Deployment Decision Method of Virtual Application Network in Multi-cloud Environment

ZHU Hua-min WU Li-fa ZHAO Peng
(Institute of Command Information System, PLA University of Science and Technology, Nanjing 210007, China)

Abstract In multi-cloud environment, users can freely combine infrastructure resources to deploy virtual application network based on virtual application technology and infrastructure virtualization technology, and thus quickly build a distributed application system with certain business functions. Regarding the shortcomings of the existing multi-cloud deployment decision-making methods in deployment descriptions and addressing user multi-objective requirements, this paper first presented a deployment description method for the virtual application network based on the open virtual format document. Secondly, the common service quality metrics of the infrastructure resource combination were studied, and a multi-objective optimization model of the combination was subsequently defined. Then, the second generation non-dominated sorting genetic algorithm (NSGA-II) and multi-objective particle swarm optimization (MOPSO) algorithm were used to solve the model. Finally, a fuzzy decision-making method was designed to select the final satisfactory solution from the obtained results. The statistics of multiple experiments show that both algorithms can achieve a good convergence within a reasonable time, NSGA-II is only suitable for scenes with 2 to 3 objectives, and MOPSO can be used for scenes with more objectives and performs better than NSGA-II. Moreover, the fuzzy decision-making result can match the user's objective preference best.

Keywords Multi-cloud, Virtual application network, Infrastructure as a service, Multi-objective optimization, Fuzzy decision-making

1 引言

传统的云计算模型下,用户通常使用单个提供商的资源,然而,提供商技术方案的异构使用户在云间自由选择、迁移时面临障碍;同时,单个提供商的服务体系很可能不再满足日趋多样化的用户需求。近年来,学术界和工业界出现了多种互联云模型[1-4],其中多云(Multi-Cloud)模型无需提供商对原技术方案及运营方式等进行任何改变,可以独立于提供商的方式,从外部广泛、自由地组合云资源,成为了一种认可度较高、具有重要推广价值的互联云模型[1-2]。

虚拟应用(Virtual Appliance)技术可将操作系统及预创建、预配置的应用程序打包成虚拟镜像,在虚拟机上快速部署运行。基础设施虚拟技术与虚拟应用技术的结合,为多云分布式应用系统的快速创建和部署提供了条件。用户能以适当的方式编排多个兼容的虚拟应用,构建满足其功能需求的虚拟应用网络,并按部署需求及约束自由组合基础设施服务(IaaS)资源并进行部署,快速构建所需的分布式应用系统。

为虚拟应用网络组合配置IaaS部署资源时,既要满足各个虚拟组件的需求,又要满足应用系统总体需求及组件间的有关约束。因此,必须提供一种良好的虚拟应用网络部署描述方式,才能以自动方式快速组合配置IaaS资源。另外,满足需求的IaaS组合通常很多,需要基于用户的服务质量

本文受江苏省自然科学基金项目(BK20131069)资助。
朱华旻(1978-),男,博士生,主要研究方向为云计算资源管理,E-mail:zhuhuamin2001@163.com;吴礼发(1968-),男,教授,博士生导师,主要研究方向为网络与信息安全;赵 鹏(1983-),男,博士生,主要研究方向为云计算资源管理。

(QoS)目标需求对 IaaS 组合进行优化，优化目标通常包括费用、性能、安全、可用性等，目标数量往往并不唯一。现有的单目标优化模型不能较好地满足需求[5-6]，将多目标加权聚合转换成单目标的方法[2,7]仅适合各目标比较独立且性质相同的情形，而 IaaS 组合的优化目标往往性质不同且并不独立，例如费用和性能。

我们在之前的研究[8]中给出了单个 IaaS 实例的匹配发现方法，能够满足单个虚拟应用组件的部署需求。在此基础上，本文对多云环境的虚拟应用网络部署决策方法进行了深入研究，主要工作及贡献如下：

1)通过扩展开放虚拟格式(OVF)[9]文档，设计了一种虚拟应用网络的拓扑及部署需求描述方法；

2)研究确立了多云 IaaS 组合的几个常见 QoS 指标，在此基础上，定义了 IaaS 组合的多目标优化模型，使用第二代非支配排序遗传算法 NSGA-II[10]和多目标粒子群优化 MOPSO[11]算法对模型进行求解；

3)针对获得的多目标最优解集，给出了一种基于模糊决策的最终满意解确定方法。

2　相关工作

已有一些学者对多云环境的虚拟应用网络部署资源决策方法进行了深入研究，然而，这些方法仍存在一些不足和缺陷。Pawluk 等[12]设计了一个业务驱动的多云经纪服务——STRATOS，初始分配资源后，根据运行时信息和资源弹性规则进行资源再分配，但独立决策各节点资源未考虑节点间部署约束及资源组合的总体效用。Tordsson[5]利用整数规划建模、求解了多云虚拟机调度问题，只考虑了将费用、云间负载平衡作为约束，将总计算容量作为目标这一种情形。Petcu 等[13]阐述了欧盟的多云中间件项目 mOSAIC，使用了基于服务等级协议(SLA)及语义技术等的资源动态协商机制，未给出组合优化的模型和方法。Son 等[2]研究了在提供商多个数据中心分配负载的方法，以资源节省、负载平衡及减小 SLA 违反为目的，创建了价格和响应时间指标的效用函数。Kessaci[14]针对“客户-经纪人-提供商”的多云环境，设计了最小化费用和虚拟机响应时间的组合优化模型，但仅考虑了部分虚拟机实例类型。Qavami 等[6]利用整数规划建模多云调度问题，分别创建了费用和性能的优化调度策略。Menzel[7]给出了一种 Web 应用簇的多云部署决策方法，以最大化性能、最小化费用为目标，通过加权将两目标聚合成单目标，由于性能与费用通常相互冲突，方法结果未必可靠。

3　虚拟应用网络部署描述方法

开放虚拟格式 OVF[9]是虚拟系统封装和分发的一种常用格式标准，由分布式管理任务组(DMTF)制订并发布为 ISO17203。OVF 文档用于描述一个虚拟系统的组成、资源配置和部署操作，我们使用扩展的 OVF 文档来描述虚拟应用网络拓扑及部署资源的配置、约束和 QoS 目标需求。

OVF 文档采用 XML 格式，其 Envelope 根目录下包含 4 类主要元素，其中引用元素 References element 来描述 OVF 引用的文件列表；Content elements 表示抽象的内容元素，具体包括虚拟系统 VirtualSystem 或虚拟系统集 VirtualSystemCollection；节元素 Section elements 定义 OVF 的主要描述数据；使用 Strings element 规范 0 个或多个区域的消息资源包。一般可以使用 VirtualSystemCollection 描述同一功能层的相关虚拟应用，用 VirtualSystem 描述一个虚拟应用组件。我们通过组合嵌套 VirtualSystem 和 VirtualSystemCollection 两类元素来描述虚拟应用网络的拓扑结构，在这两类元素内部使用节元素来描述相关应用的资源需求。图 1 给出了一个 OVF 文档的主要内容，它描述了一个玩具商店电子商务应用的拓扑及部署需求，也是本文实验中案例虚拟应用网络的部署描述文档。

```
〈ovf:PlacementGroupSection ovf:id=“PG1” ovf:policy=“affinity”〉〈/ovf:PlacementGroupSection〉
〈ovf:PlacementGroupSection ovf:id=“PG2” ovf:policy=“availability”〉〈/ovf:PlacementGroupSection〉
〈VirtualSystemCollection ovf:id=“ToyStore” otherns:objective=“cost,performance,security:high,high,high”〉
〈VirtualSystemCollection ovf:id=“WebTier”〉
〈VirtualSystem ovf:id=“IDS”〉〈ovf:PlacementSection ovf:group=“PG1”〉〈/ovf:PlacementSection〉…〈/VirtualSystem〉
〈VirtualSystem ovf:id=“WebServer”〉〈ovf:PlacementSection ovf:group=“PG1”〉〈/ovf:PlacementSection〉
〈OperatingSystemSection ovf:id=“97”〉〈Description〉Linux〈/Description〉〈/OperatingSystemSection〉
〈VirtualHardwareSection〉
〈System〉〈vssd:ElementName〉Virtual Hardware Family〈/vssd:ElementName〉
〈vssd:VirtualSystemType〉VmWare〈/ vssd:VirtualSystemType〉〈/System〉
〈Item〉〈rasd:ElementName〉virtual CPU〈/rasd:ElementName〉〈rasd:VirtualQuantity〉2〈/rasd:VirtualQuantity〉 … 〈/Item〉…〈/VirtualHardwareSection〉
〈/VirtualSystem〉
〈/VirtualSystemCollection〉
〈VirtualSystemCollection ovf:id=“AppTier” 〉
〈VirtualSystem ovf:id=“LoadBalancer”〉… 〈/VirtualSystem〉
〈VirtualSystem ovf:id=“AppServer1” 〉〈ovf:PlacementSection ovf:group=“PG2”〉〈/ovf:PlacementSection〉…〈/VirtualSystem〉
〈VirtualSystem ovf:id=“AppServer2” 〉〈ovf:PlacementSection ovf:group=“PG2”〉〈/ovf:PlacementSection〉…〈/VirtualSystem〉
〈/VirtualSystemCollection〉
〈VirtualSystem ovf:id=“DataServer”〉 〈/VirtualSystem〉
〈VirtualSystem ovf:id=“EmailServer”〉〈/VirtualSystem〉
〈/VirtualSystemCollection〉
```

图 1　实验案例电子商务应用的 OVF 描述文档

该应用包括 Web 层、应用层以及数据库和电子邮件服务。Web 层和应用层表示为嵌套的 VirtualSystemCollection 元素。Web 层包含 1 个入侵检测系统(IDS)、1 个 Web 服务器，业务层包含 1 个负载平衡器、应用服务器 1 和应用服务器 2。作为示范，在 Web 服务器 VirtualSystem 下使用 OperatingSystemSection来描述 Web 服务器的操作系统，使用 VirtualHardwareSection 来描述虚拟应用所需 IaaS 实例的配置情况，每个 Item 项分别描述某方面的具体需求，例如虚拟机 CPU、内存等。组合使用 PlacementGroupSection 和 PlacementSection 两个节元素，为 VirtualSystem 或 VirtualSystemCollection 指定部署约束。通过在 OVF 文档根目录下使用 PlacementGroupSection 定义不同部署约束，赋值给该元素的 ovf:policy 属性即可指定其约束策略，之后就可在 VirtualSystem 或 VirtualSystemCollection 内部添加 PlacementSection 元素，通过指定该元素的 ovf:group 属性引用定义好的部署

约束。图 1 中定义了部署约束 PG1，在 IDS 和 WebServer 组件上进行了引用；定义了部署约束 PG2，在 AppServer1 和 AppServer2 组件上进行了引用。

此外，为描述部署资源组合优化的 QoS 目标，为 VirtualSystemCollection 扩展定义了一个字符串类型的 QoS 目标属性 objective，在此用户可以指定对 IaaS 组合的 QoS 偏好，例如该 OVF 文档中用户指定：objective＝“cost，performance，security；high，high，high”，表示要求同时优化费用、性能和安全性，且对 3 个目标的偏好均为高度满意，这将作为 IaaS 组合多目标优化及模糊决策最终解的依据。最后，为 VirtualSystem 扩展了 num 属性，用以描述应用组件的虚拟机数量。

4 IaaS 组合的服务质量指标

卡内基梅隆大学的云服务度量指标协会(CSMIC)[15]提出了一套云服务 QoS 指标目录，包括可审计性、灵活性、费用、性能、服务保证、安全隐私、易用性等。Li 等[16]将 IaaS 的 QoS 指标分类为性能、经济性和安全性三方面。基于现有工作[8,15-17]，确立了 IaaS 组合的主要 QoS 指标及其获取方法。

(1)费用：对于虚拟应用网络的各个虚拟组件，基于我们在文献[8]中给出的 IaaS 特征模型匹配方法，为其获取到满足需求的 IaaS 实例，之后匹配实例的费用也随即确定。实例费用包括实例购置费用和网络费用，假定虚拟应用网络有 n 个虚拟应用组件，第 i 个虚拟应用组件有一个规模为 k_i 的可选实例群，num_i 表示虚拟组件 i 所需同类型 IaaS 实例的个数，x_{ij} 表示虚拟组件 i 的可选实例群中第 j 个可用实例，将 x_{ij} 的单位月购置费表示为 $ICost_{ij}$，单位月网络费用表示为 $NCost_{ij}$，则任一可用实例组合 $\{x_1,x_2,\cdots,x_n\}$ 的费用 $TCost$ 可表示为：

$$TCost=\sum_{i=1}^{n}(ICost_{ij}+NCost_{ij})\times num_i,1\leqslant j\leqslant k_i \tag{1}$$

(2)性能：各应用组件的虚拟机性能决定整个 IaaS 组合的性能，来自不同提供商的同配置虚拟机的性能测试往往也表现出不小的差异。标准性能评估协会(SPEC)[18]推出的 SPEC CPU2006 是一套较通用的 CPU 子系统标准测试套装，包括 SPECint 和 SPECfp 两个子项，前者测试整数处理性能，后者测试浮点数处理性能。云服务性能测试比较服务 CloudHarmony[19]免费提供了许多云提供商 IaaS 实例的 SPECint，SPECfp 测试数据[19]。我们将 SPECint，SPECfp 测试分值之和作为虚拟机性能测试的最终分值，采取直接求和的方法聚合各个 IaaS 实例的性能值。对于任意可用实例组合 $\{x_1,x_2,\cdots,x_n\}$，虚拟组件总数为 n，第 i 个虚拟组件的虚拟机个数为 num_i，实例 x_i 的性能为 $Performance_i$，那么有：

$$Performance=\sum_{i=1}^{n}Performance_i\times num_i \tag{2}$$

(3)安全性：利用文献[20]给出的云安全度量和评价方法估算 IaaS 安全性，依据提供商向云安全联盟(CSA)提交的共识倡议评估问卷(CAIQ)报告计算其安全值。CAIQ 依据云控制矩阵(CCM)设计，CCM 是 CSA 参考许多权威安全标准和框架，提出的最佳云安全控制实践规范。采取求取组合中提供商安全评价平均值的方法，假设 k 为参与组合的提供商数量，$Security_i$ 为提供商 i 的安全评价值，那么有：

$$Security=(\sum_{i=1}^{k}Security_i)/k \tag{3}$$

(4)可用性：它是目前多数提供商给出 SLA 公开承诺的一个 QoS 指标，一定周期内，IaaS 实例可用性一般可计算为：可用性＝可用时间长度/(可用时间长度＋不可用时间长度)。基于我们的 IaaS 特征模型匹配方法[8]，为虚拟组件获取到满足需求的 IaaS 实例后，其可用性也随即确定。IaaS 组合的可用性需基于 OVF 文档对应用拓扑的描述进行聚合计算。为此，定义了顺序、分支和并行 3 种基本拓扑结构的可用性聚合方法，实际应用中组合使用这 3 种方法即可。

对于顺序、并行结构，令 l 为结构中实例总数，$Availability_i$ 为实例 i 的可用性，则有：

$$Availability=\prod_{i=1}^{l}Availability_i \tag{4}$$

对于分支结构，令 l 为结构中实例总数，p_i 表示分支 i 执行概率的估计值：

$$Availability=\sum_{i=1}^{l}p_iAvailability_i \tag{5}$$

(5)信誉度：它是 IT 服务的一项重要指标，尽管学术界提出了许多云服务信誉度计算模型，但均处于理论研究阶段，尚未有一种模型得到较广泛的推广和商用，很难获得可参考的服务信誉度值。市场占有率尽管不能代替服务信誉度，但能够基本反映用户的选择倾向，一定程度上间接地反映了服务信誉度，因此，选用 CloudHarmony 网站[19]提供的云提供商市场占有率数据作为一个度量提供商信誉度的指标。IaaS 组合的聚合方式与安全性的聚合方式相同，即取组合中各提供商市场占有率的平均值。

5 多目标优化模型

以结构化标准方式完整描述 IaaS 服务的实例空间并创建较完备的 IaaS 服务库，是自动匹配及组合优化 IaaS 资源的前提。文献[8]提出了一种基于特征模型的 IaaS 结构化描述及匹配方法，通过创建 IaaS 特征模型对 IaaS 服务及需求进行描述，并使用特征模型自动分析技术搜索满足需求的 IaaS 匹配实例。目前，我们通过人工方式收集了一些提供商的 IaaS 信息，创建其 IaaS 服务模型，如此来建立和维护 IaaS 服务库。为提高效率，未来考虑使用网络爬虫方法搜集 IaaS 服务信息，逐步实现 IaaS 服务库的自动化创建和维护。

基于 OVF 文档对各虚拟组件的资源需求进行描述，使用 IaaS 匹配实例发现方法，可为各虚拟组件自动搜索获取到满足部署需求的 IaaS 匹配实例，详细过程参见文献[8]。根据该匹配方法，对每个成功匹配的 IaaS 服务仅返回了费用最优的 IaaS 匹配实例，这样一个虚拟组件若与 t 个 IaaS 服务成功匹配，则将总共获得 t 个可用 IaaS 实例，组成一个规模为 t 的候选 IaaS 实例群。

假定虚拟应用网络有 n 个虚拟应用组件，用户同时持有 m 个 QoS 目标，那么 IaaS 实例组合的多目标优化模型的一般形式可表示为：

$$\begin{cases}\max Y=F(X)=(F_1(X),F_2(X),\cdots,F_m(X))\\ \text{s.t. } G_j(X)\leqslant 0,j=1,2,\cdots,p\\ \quad H_k(X)=0,k=1,2,\cdots,q\\ \quad X\in[X_{\min},X_{\max}]^n\end{cases} \tag{6}$$

其中，$X=(x_1,x_2,\cdots,x_n)\in X'\subset R^n$ 为 n 维决策向量，表示虚拟应用网络的任一可用 IaaS 实例组合，X' 为所有可用组合构成的 n 维决策空间。目标函数 $F(X)$ 定义了 m 个由决策空间向目标空间映射的函数，$Y=(y_1,y_2,\cdots,y_m)\in Y'\subset R^m$ 表示

任意决策向量 X 通过函数 $F_1(X),F_2(X),\cdots,F_m(X)$ 映射到目标空间的一个 m 维目标向量，Y' 为 m 维目标空间。$G_i(X)\leqslant 0$ 和 $H_k(X)=0$ 分别表示不等式约束和等式约束，用来表示虚拟应用网络的部署约束。考虑到第 4 节的 5 个 QoS 指标，相应定义了 5 个目标函数 $Co(X),Av(X),Pe(X),Se(X),Re(X)$，分别表示费用、可用性、性能、安全和信誉，可根据用户偏好的 QoS 目标自由组合。另外，分别定义了 AVAILABILITY（可用性），AFFINITY（亲近）和 LATENCY（延迟）3 个约束策略。首先指定了两个约束函数：$Res1(x_1,x_2,\cdots,x_l)=\{y\mid y=0 \text{ 或 } 1\}$，其中 $2\leqslant l\leqslant n$；$Res2(x_i,x_j)\leqslant e_{ij}$，其中 i,j 为 1 到 n 的整数，且 $i\neq j,e_{ij}\in R^+$ 为正整数。

定义 1（AVAILABILITY 约束）　令 $x_1,x_2,\cdots,x_l$ 分别表示虚拟应用网络第 $1,2,\cdots,l$ 个虚拟组件对应的任意一组可用实例，若要求这 l 个实例必须分别来自不同的云数据中心，则定义约束函数 $Res1(x_1,x_2,\cdots,x_l)=0$，称这 l 个实例存在 AVAILABILITY 约束。

定义 2（AFFINITY 约束）　令 $x_1,x_2,\cdots,x_l$ 分别表示虚拟应用网络第 $1,2,\cdots,l$ 个虚拟组件对应的任意一组可用实例，若要求这 l 个实例必须全部来自同一个云数据中心，则定义约束函数 $Res1(x_1,x_2,\cdots,x_l)=1$，称这 l 个可用实例存在 AFFINITY 约束。

定义 3（LATENCY 约束）　令 x_i,x_j 分别表示虚拟应用网络第 i,j 个虚拟应用对应的任意两个可用实例，如果这两个实例间的通信延迟不能超过指定值 $e_{ij},e_{ij}\in R^+$，那么定义约束函数 $Res2(x_i,x_j)\leqslant e_{ij}$，称这两个可用实例存在 LATENCY 约束 e_{ij}。

6　模型求解

6.1　相关定义及方法分析

多目标优化问题不存在唯一最优解，即不可能找到一个解使所有目标值同时最优，多个目标间通常相互制约、互相冲突，只能在其间协调，找出折中解。在此给出多目标优化的几个重要定义。

定义 4（可行解）　对于 $\forall X\in X'$，X' 为多目标优化问题的决策变量空间，如果 X 满足所有约束条件，则称 X 为可行解。

定义 5（Pareto 支配）　假定 X_1,X_2 是多目标优化问题的任意两个可行解，$F(X)$ 为效益型目标函数，m 为目标个数，称解 X_1 支配 X_2，记为 $X_1\succ X_2$，当且仅当：

$$\forall i=1,\cdots,m,F_i(X_1)\geqslant F_i(X_2)\wedge\exists j=1,\cdots,m,F_j(X_1)\succ F_j(X_2) \tag{7}$$

定义 6（Pareto 最优解）　假设 Q 是多目标优化问题的可行解集合，称一个可行解 X^* 为 Pareto 最优解，当且仅当：

$$\neg\exists X\in Q,\text{使 } X\succ X^* \tag{8}$$

定义 7（Pareto 最优解集）　假设 Q 是多目标优化问题的可行解集合，所有 Pareto 最优解的集合称为 Pareto 最优解集 P^*，定义如下：

$$P^*=\{X^*\mid\neg\exists X\in Q,X\succ X^*\} \tag{9}$$

定义 8（Pareto 前沿）　所有 Pareto 最优解在目标空间的映射构成 Pareto 前沿 PF^*，定义如下：

$$PF^*=\{F(X^*)\mid X^*\in P^*\} \tag{10}$$

多目标优化即是获取尽可能完整的 Pareto 最优解集的过程，一个主要操作是根据各维目标值判断可行解之间的支配关系，搜索非支配解。由于决策空间通常很大，随目标维数增多，计算过程相当复杂，一般采用启发式方法求解，已有研究表明进化算法在多目标优化方面表现出色[21]。NSGA-II[10] 和 MOPSO[11] 是目前两个比较优秀的代表性多目标优化算法，选择采用这两个算法来求解 IaaS 的组合优化模型。

6.2　NSGA-II 算法

NSGA-II 是 NSGA 的改进版[10]，主要优点是采用基于分级的快速非支配排序法，将复杂度降低到 $O(mN^2)$，其中 N 为种群大小，m 为目标数；引进精英保留机制，使子代、父代共同竞争产生新一代种群，有利于提高种群进化水平。

(1)编码：NSGA-II 通常使用模拟二进制交叉和多项式变异，均针对实数编码，为此本文也采用实数编码。任意决策变量表示为一个实数向量 $X=(x_1,x_2,\cdots,x_n)$，n 为虚拟应用组件数，$1\leqslant x_i\leqslant k_i$，$k_i$ 表示第 i 个虚拟组件的可用 IaaS 实例数。通过 Matlab 的取整函数 round(x)，可将实数决策向量转换为整数向量，映射到一个 IaaS 实例组合。

(2)算子：选择机制采用常用的随机联赛方法，即每次随机选择几个个体并从中取适应度最高的个体参与组成父代种群，一般联赛规模取 2，交叉、变异算子分别选择常用的模拟二进制交叉、多项式变异，具体参见文献[10]。

(3)主要参数设置：按照一般经验并结合问题实际，设置 NSGA-II 的主要参数如下：种群规模为 200，最大进化代数为 250，交叉概率为 0.9，变异概率为 0.1。

6.3　MOPSO 算法

粒子群优化(PSO)算法源于对鸟群觅食活动的研究，通过共享个体信息使群体运动从无序到有序演化，获得最优解[22]。Coello 等提出的 MOPSO 算法[11] 将自适应超网格机制作为外部档案维护和领导者选择策略，并引入概率逐渐衰减的变异机制。粒子在搜索空间的速度和位置计算如下：

$$\begin{cases}V_i(t+1)=wV(t)+c_1\times rand()\times(Pbest_i-X_i(t))+\\\qquad c_2\times rand()\times(Gbest-X_i(t))\\X_i(t+1)=X_i(t)+V_i(t+1)\end{cases} \tag{11}$$

$X_i=(x_{i1},x_{i2},\cdots,x_{in})$ 和 $V_i=(v_{i1},v_{i2},\cdots,v_{in})$ 分别表示粒子 i 的位置和速度，w 为惯性权重系数，c_1 和 c_2 为学习因子，$rand()$ 为[0,1]随机数，$Pbest_i$ 是粒子 i 的历史最优位置，$Gbest$ 是当前全局最优位置。通过式(11)调整粒子的飞行速度和当前位置，速度调整公式右端的第一部分是粒子先前行为惯性，第二部分通过记忆历史最优位置实现自学习，第三部分通过感知全局最优位置共享群体信息不断调整速度，从而引导粒子群向潜在最优解区域飞行。

(1)编码：方法与 NSGA-II 的编码相同。

(2)变异操作：迭代初期粒子不易陷入局部最优，无需对粒子变异；随着迭代的进行，粒子逐渐向最优位置聚集，可能陷入局部最优，此时可进行一定概率变异，若粒子群在迭代早期连续多次保持不变，则应加大变异概率；到后期则应逐渐减小变异概率，使粒子逐渐收敛。为此，定义了粒子变异概率 P_m 的计算公式：

$$P_m=\begin{cases}l/G_{cu}, & G_{cu}\leqslant I_{\max}/2\\1-G_{cu}/I_{\max}, & G_{cu}>I_{\max}/2\end{cases} \tag{12}$$

其中，l 为粒子群连续不发生变化的代数，G_{cu} 为当前代数，$I_{\max}$ 为总迭代次数。

(3)主要参数设置:按通常情况并结合反复实验经验,设置 w 初始值为 0.5,衰减率 $wdamp=0.99$,设置加速常数 $c_1=c_2=2$,粒子群规模 $nPop=200$,最大迭代次数 $IT_{max}=250$,目标网格数 $nGrid=7$,领导者选择压力 $beta=2$ 等。

7 最终满意解的模糊决策

多目标优化算法获取的一系列 Pareto 非支配解已经无法直接比较优劣,必须在各目标间折衷协调,选取一个最终满意解,为此,设计给出了一种基于模糊决策的最终解选取方法。首先,模糊评价每个 Pareto 解的各子目标满意度;然后,根据用户的多目标模糊偏好,从 Pareto 解集中匹配搜索最贴近用户目标模糊偏好的子集;最后,从返回解集中选出中心个体作为最终满意解。

7.1 模糊评价 Pareto 解的子目标满意度

假设各子目标均为效益型(成本型可转换为效益型),合并定义各个子目标的满意度函数 $D_i(x)$ 如下:

$$D_i(x)=(f_i(x)-f_i^{min})/(f_i^{max}-f_i^{min}),i=1,2,\cdots,k \quad (13)$$

其中,$f_i(x)$ 是任一 Pareto 解 x 的子目标 i 的实际值,f_i^{min} 和 f_i^{max} 分别是子目标 i 的最小值和最大值,显然,$D_i(x)\in[0,1]$。将子目标满意度作为语言变量,定义了该语言变量的一个模糊子集:{低、较低、中等、较高、高},同时定义了相应的三角隶属度函数,如图 2 所示。基于隶属度函数即可模糊评价各子目标满意度,即根据子目标 $D_i(x)$ 值,将其评价或描述为高满意度、…、较低满意度、低满意度。

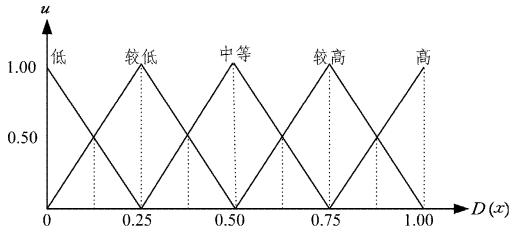

图 2 目标满意度模糊子集的隶属度函数

7.2 基于模糊偏好的最终解选取算法

通过模糊评价 Pareto 解各子目标满意度,可以得到一些具有典型特征的 Pareto 解类别,例如有费用、性能两个目标,可以得到 5×5=25 个类别的 Pareto 子集,如“费用高度满意、性能高度满意”,“费用高度满意、性能较为满意”等,根据用户的多目标模糊偏好,即可选择匹配的类别。根据聚类思想,一群个体中与其他所有个体距离和最小的个体为中心个体,它最能代表整个群体的特征。因此,可以从匹配用户目标偏好的 Pareto 子集中选出中心体,作为用户最终满意解。然而,用户多目标模糊偏好很有可能因过于严格或出于巧合,找不到恰好匹配的 Pareto 子集。为此,设计了算法 1 来模糊评价 Pareto 解集,并从中选择使各子目标满意度与用户偏好恰好匹配或最接近匹配的最终解。首先,根据 OVF 文档描述的用户多目标模糊偏好,创建多目标模糊偏好等级向量 $ObjL$;然后,将 $ObjL$ 与所有 Pareto 解进行子目标满意度等级匹配,若返回空集,则按照从次要目标到优先目标的顺序,通过循环、逐级降低子目标满意度等级的方式,与 Pareto 最优集进行匹配。过程中,只要获得了非空匹配子集,则退出循环,选择该匹配集的中心个体作为最终满意解。

算法 1 多目标优化最终解选择算法

输入:Pareto 最优解集 P*,有优先顺序的目标模糊偏好等级向量 ObjL(ObjL[i] > ObjL[i+1]),目标数 n,满意度等级数 L

输出:最终满意解 p

```
1. L←5;
2. 模糊评价 P* 中各解的子目标满意度,分配相应等级;
3. 基于 ObjL 从 P* 中搜索最佳匹配子集 P;
4. if P==∅ then
5.   for i from 1 to L−1 //逐渐调低满意度等级
6.   for j from 0 to n−1 //按逆优先顺序调低目标满意度
7.       if ObjL[n−j] >1 then
8.       ObjL[n−j]←ObjL[n−j]−1;
9.       基于 ObjL 从 P* 中搜索最佳匹配子集 P;
10.      end if
11.      if P ~=∅ then goto line 15;end if
12.  end for
13.  end for
14. end if
15. 选出 P 的中心个体 p;
16. return p;
```

8 实验及分析

8.1 实验准备

考虑一个典型的多层 Web 分布式应用——电子商务作为多云虚拟应用网络部署案例。这个玩具商店电子商务系统包括 Web 层、应用层以及数据库层和电子邮件服务,使用扩展的 OVF 文档描述定义了其应用拓扑及部署需求,主要内容如图 1 所示,细节参见第 3 节。

首先,基于真实云提供商原型,创建了几个提供商的 IaaS 服务模型作为备选 IaaS 服务库,包括 RackSpace[23],DigitalOcean[24],Linode[25] 等 9 个 IaaS 服务模型,使用数字 1~9 对这 9 个提供商进行依次编号。然后,基于玩具商店 OVF 文档对各虚拟组件的部署进行描述,生成相应的 IaaS 配置需求,使用文献[8]给出的 IaaS 特征模型匹配方法,为每个虚拟组件匹配搜索得到一组可用 IaaS 实例。使用数字 1~7 依次对 IDS, WebServer, LoadBalancer, AppServer1, AppServer2, DataServer, EmailServer 虚拟组件进行编号,其对应的备选实例群数据如图 3 中矩阵 1~7 所示。

$$\begin{bmatrix}4&107&0.9990&111&65.948&0.05\\1&55&0.9999&101&54.243&0.01\\2&60&0.9990&115&73.328&0.02\\6&94&0.9990&120&52.121&0.01\\9&75&0.9990&100&68.910&0.06\end{bmatrix}$$

可用实例矩阵 1,3

$$\begin{bmatrix}6&212&0.9990&220&52.121&0.01\\4&175&0.9990&212&65.948&0.05\\9&168&0.9990&195&68.910&0.06\\1&185&0.9999&200&54.243&0.01\\2&170&0.9990&175&73.328&0.02\\7&210&0.9999&216&67.992&0.04\end{bmatrix}$$

可用实例矩阵 2,7

$$\begin{bmatrix}4&175&0.9990&212&65.948&0.05\\1&185&0.9999&200&54.243&0.01\\2&170&0.9990&175&73.328&0.02\\6&212&0.9990&220&52.121&0.01\\7&210&0.9999&216&67.992&0.04\\9&168&0.9990&195&68.910&0.06\\3&187&0.9990&199&59.011&0.01\\5&200&0.9990&190&45.641&0.03\end{bmatrix}$$

可用实例矩阵 4,5

$$\begin{bmatrix}4&285&0.9990&305&65.948&0.05\\1&274&0.9999&301&54.243&0.01\\2&250&0.9990&285&73.328&0.02\\6&290&0.9990&295&52.121&0.01\\7&280&0.9999&300&67.992&0.04\\9&240&0.9990&275&68.910&0.06\\3&255&0.9990&295&59.011&0.01\\8&270&0.9999&303&53.459&0.02\end{bmatrix}$$

可用实例矩阵 6

图 3 虚拟组件 1~7 对应的备选实例数据矩阵

组件 1 和 3、组件 2 和 7、组件 4 和 5 的部署需求分别相同，得到相同的可用实例矩阵。矩阵每行代表虚拟组件的一个可选实例，6 列依次表示实例的提供商编号、月费用、可用性、性能、安全性和市场占有率。费用、可用性、性能在完成实例匹配后即已获得；利用文献[20]给出的安全性评价方法计算得到提供商 1～9 的安全值，为了直观，将其值转换成百分制，依次为：54.243，73.328，59.011，65.948，45.641，52.121，67.992，53.459，68.910；通过 CloudHarmony[19] 网站获取到提供商 1～9 的市场占有率，分别为：0.01，0.02，0.01，0.05，0.03，0.01，0.04，0.02，0.06。另外，由图 1 的 OVF 描述文档可知，应用系统包含部署约束 AVAILABILITY 和 AFFINITY，表示为 $Res1(x_1,x_2)=1$ 和 $Res1(x_4,x_5)=0$，意味着组件 1 和组件 2 必在相同数据中心，而组件 4 和组件 5 不在同一个数据中心。最后，通过 Matlab 编程实现了 NSGA-II 和 MOPSO 算法，分别针对目标向量 $obj_1=[1,2]$，$obj_2=[1,2,3]$，$obj_3=[1,2,3,4]$，$obj_4=[1,2,3,4,5]$ 使两个算法独立运行 30 轮，统计分析实验结果（目标 1～5 分别代表实例组合的月费用、可用性、性能、安全性和市场占有率）。根据用户在 OVF 文档中指定的 QoS 目标偏好，从费用、性能、安全性的多目标优化结果中为其选择并确定一个最终满意解。

8.2　算法性能指标及参考的 Pareto 最优集

多目标优化算法一般通过求取所得 Pareto 非支配解集的几个综合指标或专项指标，来评价算法收敛性、多样性和解分布均匀性。这里选择实践常用的 4 个指标[21-22]：世代距离（GD）[26]、间距（SP）指标[27]、反世代距离（IGD）[28]、算法执行时间。

（1）世代距离（GD）：它是算法收敛指标，度量所得解集 P 与真实最优集 P^* 的接近程度，即所得 Pareto 解集 P 映射到目标空间的点集与真实 Pareto 最优集 P^* 在目标空间点集的间隔，计算公式为 $GD=\sqrt{\sum_{i=1}^{num}d_i^2}/num$。$num$ 是所得解数量，d_i 是在目标空间内 P 的任意解 i 到 P^* 的最短欧氏距离。

（2）间距指标（SP）：它是在目标空间中所得 Pareto 解集 P 内相邻个体间距离的标准偏差，用以描述所得 Pareto 解集 P 在目标空间的分布均匀性，$SP=\sqrt{\sum_{i}^{n}(d_i-\bar{d})^2/(n-1)}$。$d_i$ 为个体 X_i 到 P 中其余个体在目标空间的最短距离，$\bar{d}$ 为所有 d_i 的平均值。SP 值越小，表明所得 P 中个体分布越均匀，SP 值等于 0 表明 P 所有解等距离分布在目标空间。

（3）反世代距离（IGD）：它是个综合性评价指标，既反映所得解集收敛性，也反映其解分布均匀性。IGD 与 GD 的计算方法类似，只是它描绘的是真实 Pareto 最优前沿到算法所得 Pareto 前沿的间隔。

（4）参考的 Pareto 最优集：实验案例的真实 Pareto 最优集事先并不知道，但评价算法必须确定一个参考最优集，借鉴文献[7]的遍历方法，获取到完整 Pareto 非支配解集作为参考集。2～5 个目标的 Pareto 最优解耗时在 6～9h 之间。

8.3　实验结果对比

对不种目标场景重复运行算法各 30 次，使用统计软件 minitab[29] 描绘了指标箱线图，如图 4－图 8 所示。箱线图方框底部为下四分位数（1/4 的数据小于或等于此值），方框顶部为上四分位数，＊号表示异常值，方框中线上端延伸至最大值，下端延伸至最小值，框内横线表示中位数。

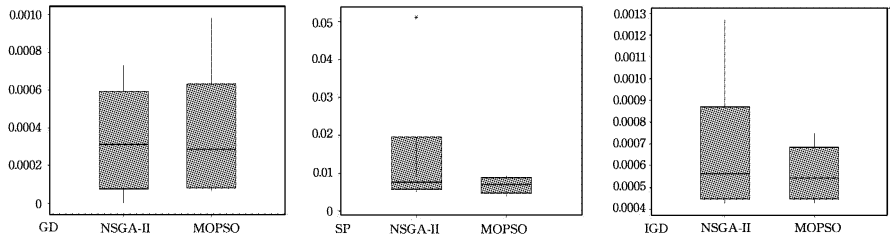

图 4　2 个目标时 GD，SP，IGD 指标分布对比

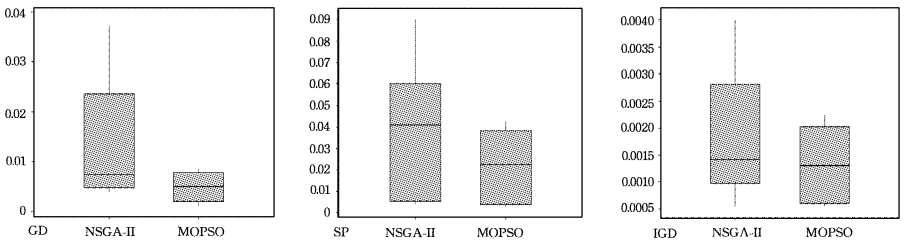

图 5　3 个目标时 GD，SP，IGD 指标分布对比

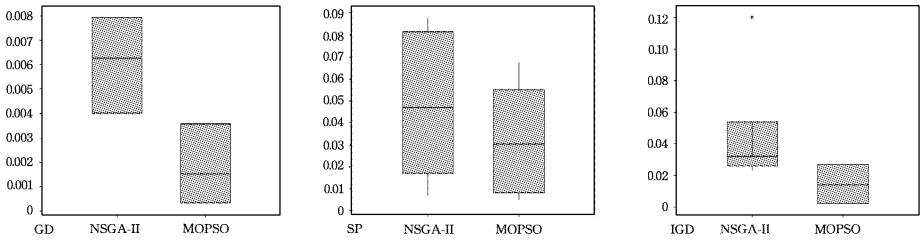

图 6　4 个目标时 GD，SP，IGD 指标分布对比

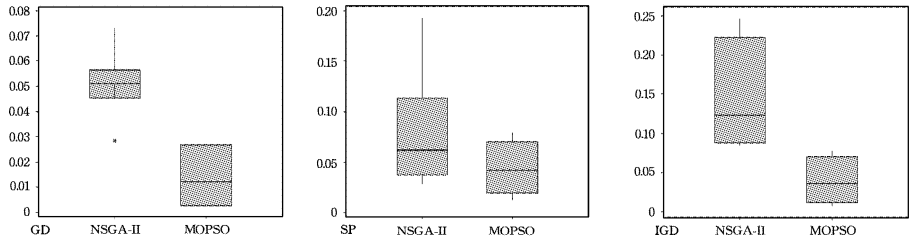

图 7 5 个目标时 GD,SP,IGD 指标分布对比

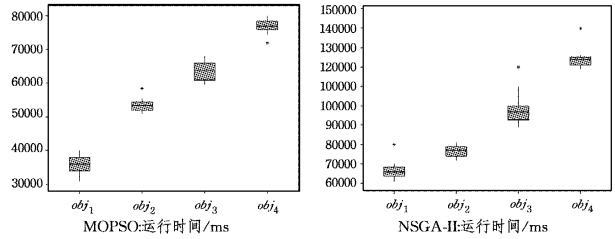

图 8 2～5 个目标时两个算法的运行时间的对于比情况

(1)obj_1=[1,2]:如图 4 所示,对于均匀性指标 SP 和综合指标 IGD,MOPSO 稍好于 NSGA-II,而收敛性指标 GD 无明显差异,两算法中位值差异不大,总体上 2 个目标时两个算法的性能无明显差别。

(2)obj_2=[1,2,3]:如图 5 所示,就中位值而言,MOPSO 均优于 NSGA-II,MOPSO 的绝大部分收敛指标 GD 优于 NSGA-II,MOPSO 的所有 SP 值基本都位于 NSGA-II 中位值以下。就综合指标 IGD 来看,MOPSO 的分布比 NSGA-II 更好。因此,3 个目标时 MOPSO 性能稍好于 NSGA-II。

(3) obj_3=[1,2,3,4]:如图 6 所示,MOPSO 的收敛指标 GD 和综合指标 IGD 要好于 NSGA-II,两个算法的均匀指标 SP 有一些重叠,但 MOPSO 的中位值更低,且对于 SP,其总体上比 NSGA-II 分布稍好些,综合来看,4 个目标情形下 MOPSO 的性能较 NSGA-II 要好。

(4)obj_4=[1,2,3,4,5]:如图 7 所示,就 GD 和 IGD 的表现,MOPSO 明显好于 NSGA-II;就 SP 而言,MOPSO 几乎有近 75%的值处于 NSGA-II 的 SP 中位值以下,MOPSO 算法的 SP 值明显低于 NSGA-II。因此,5 个目标时 MOPSO 性能明显好于 NSGA-II。

(5)运行时间对比:MOPSO 与 NSGA-II 运行时间的对比如图 8 所示,横轴分别表示 2～5 个目标的场景。随着目标数增加,两个算法处理时间均有所增长,总体上,MOPSO 的处理时间未超过 80s,目标为 2 个时,运行时间只需 38s 以下;NSGA-II 的处理时间在 140s 以下,2 个目标时,处理时间在 69s 以下。

(6)最终满意解的获取:基于图 1 OVF 文档描述的用户“费用、性能、安全性均高度满意”目标需求偏好,首先通过 MOPSO 算法获取到费用、性能、安全性上的多目标优化 Pareto 最优解集;然后使用第 7 节的模糊决策方法,获取到用户目标偏好的最佳匹配子集,部分如图 9 所示。各解已按照与其他解的距离之和升序排序,据此得到用户最终满意解:X=[3 5 3 1 5 2 2]或[3 5 3 5 1 2 2],二解均为最佳匹配子集的中心个体。

实例组合	费用、性能、安全性	费用、性能、安全性的满意度	偏好等级	距其他解
3 5 3 1 5 2 2,	1124 1346 67.730714,	0.677083 0.642857 0.730239,	4 4 4,	161.980007
3 5 3 5 1 2 2,	1124 1346 67.730714,	0.677083 0.642857 0.730239,	4 4 4,	161.980007
3 5 3 1 4 5 2,	1132 1349 67.427571,	0.649306 0.657143 0.713772,	4 4 4,	165.772619
3 5 3 4 1 5 2,	1132 1349 67.427571,	0.649306 0.657143 0.713772,	4 4 4,	165.772619
3 5 3 1 5 5 2,	1130 1345 69.694857,	0.656250 0.638095 0.836931,	4 4 4,	174.847681
3 5 3 5 1 5 2,	1130 1345 69.694857,	0.656250 0.638095 0.836931,	4 4 4,	174.847681
3 5 3 1 5 8 2,	1120 1348 67.618714,	0.690972 0.652381 0.724155,	4 4 4,	177.621358

图 9 Pareto 最优集的多目标模糊决策结果子集

结束语 本文针对多云虚拟应用网络的部署,研究给出了 IaaS 组合的优化决策方法。首先,给出了虚拟应用网络拓扑及部署需求的描述方法,研究确立了多云 IaaS 组合的 5 个主要 QoS 优化指标,并给出了获取与聚合的方法;然后,定义了多云 IaaS 组合的多目标优化模型,并选择 NSGA-II 及 MOPSO 算法进行了求解;最后,设计了一种模糊决策方法,从获取的 Pareto 最优集中确定最终满意解。不同多目标场景的多次实验显示,两个算法均能够有效求解问题,NSGA-II 更适合 2～3 个目标的场景,而 MOPSO 在 2～5 个目标的场景下均有更好的性能表现,MOPSO 算法不仅运行时间均在 80s 以下,其他各项算法指标均比 NSGA-II 表现更好;另外,使用模糊决策方法获取的最终解能够最佳匹配用户的多目标偏好。

参 考 文 献

[1] GROZEV N,BUYYA R. Inter-Cloud architectures and application brokering:taxonomy and survey[J]. Software:Practice and Experience,2014,44(3):369-390.

[2] PETCU D. Multi-Cloud:expectations and current approaches[C]// Proceedings of the 2013 International Workshop on Multi-cloud Applications and Federated Clouds. ACM,2013.

[3] FERRER A J. Inter-cloud Research:Vision for 2020[J]. Procedia Computer Science,2016,97:140-143.

[4] ASSIS M,BITTENCOURT L. A survey on cloud federation architectures:Identifying functional and non-functional properties [J]. Journal of Network and Computer Applications,2016,72:51-71.

[5] TORDSSON J,MONTERO R S,MORENO-VOZMEDIANO R,et al. Cloud brokering mechanisms for optimized placement of virtual machines across multiple providers[J]. Future Generation Computer Systems,2012,28(2):358-367.

[6] QAVAMI H R,JAMALI S,AKBARI M K,et al. Dynamic resource provisioning in cloud computing:a heuristic markovian approach[C]// International Conference on Cloud Computing. Springer,2013.

(下转第 328 页)

tion of Web sandboxes[J]. Journal of Computer Security, 2014, 22(4): 511-565.

[4] MAASS M, SALES A, CHUNG B, et al. A systematic analysis of the science of sandboxing[J]. BMC Evolutionary Biology, 2016, 2(3): e43.

[5] JavaScript and the Document Object Model[OL]. http://www.ibm.com/developerworks/web/library/wa-jsdom.

[6] The Browser Object Model[OL]. http://msdn.microsoft.com/en-us/library/ms952643.aspx.

[7] ECMAScript Language Specification 2015[OL]. http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf.

[8] DOUGLAS C. ADsafe[OL]. http://www.adsafe.org.

[9] MARK S, MILLER S, BEN L, et al. Caja: Safe active content in sanitized JavaScript[OL]. http://google-caja, googlecode.com/files/caja-spec-2008-06-07.pdf.

[10] Facebook. FBJS[OL]. http://developers.facebook.com/docs/fbjs.

[11] SERGIO M, MITCHELL J C, TALY A. Isolating JavaScript with filters, rewriting, and wrapper [C]// European Symposium on Research in Computer Security. Springer Berlin Heidelberg, 2009: 505-522.

[12] TALY A, ERLINGSSON Ù, MITCHELL J C, et al. Automated analysis of security-critical javascript apis[C]// 2011 IEEE Symposium on Security and Privacy (SP). IEEE, 2011: 363-378.

[13] POLITZ J G, SPIRIDON A E, ARJUN G, et al. ADsafety: Typed-based verification of JavaScript sandboxing[C]// Proceedings of the 20th USENIX Conference on Security. 2011.

[14] POLITZ J G, ARJUN G, SHRIRAM K, Semantics and Types for Objects with First-Class Member Names[M]// Workshop on Foundations of Object-Oriented Languages (FOOL). 2012: 15-22.

[15] SAXENA P, AKHAWE D, HANNA S, et al. A symbolic execution framework for javascript[C]// 2010 IEEE Symposium on Security and Privacy (SP). IEEE, 2010: 513-528.

[16] LI Y F, PARAMJIT K D, DAVID L D. Two decades of Web application testing-A Survey of recent advances[J]. Information System, 2014, 43: 20-54.

[17] DANIEL M, JAMES W. Choosing Scrapy[J]. Journal of Computing Sciences in Colleges, 2015, 31(1): 83-89.

(上接第 292 页)

[7] MENZEL M, RANJAN R, WANG L, et al. CloudGenius: a hybrid decision support method for automating the migration of web application clusters to public clouds[J]. IEEE Transactions on Computers, 2015, 64(5): 1336-1348.

[8] ZHU H M, WU L F, HUANG K Y, et al. Research on Methods for Discovering and Selecting Cloud Infrastructure Services Based on Feature Modeling[OL]. http://downloads.hirdawi.com/journals/mpe/2016/8194832.pdf.

[9] CROSBY S, DOYLE R, GERING M, et al. Open virtualization format specification[R]. Distributed Management Task Force, 2010.

[10] DEB K, PRATAP A, AGARWAL S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II[J]. IEEE transactions on evolutionary computation, 2002, 6(2): 182-197.

[11] 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.

[12] PAWLUK P, SIMMONS B, SMIT M, et al. Introducing STRATOS: A cloud broker service[C]// 2012 IEEE 5th International Conference on Cloud Computing (CLOUD). IEEE, 2012.

[13] PETCU D, DI MARTINO B, VeENTICINQUE S, et al. Experiences in building a mOSAIC of clouds[J]. Journal of Cloud Computing: Advances, Systems and Applications, 2013, 2(1): 12.

[14] KESSACI Y, MELAB N, TALBI E G. A pareto-based genetic algorithm for optimized assignment of vm requests on a cloud brokering environment [C]// 2013 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2013.

[15] SIEGEL J, PERDUE J. Cloud services measures for global use: the Service Measurement Index (SMI)[C]// 2012 Annual SRII Global Conference (SRII). IEEE, 2012.

[16] LI Z, O'BRIEN L, ZHANG H, et al. On a catalogue of metrics for evaluating commercial cloud services[C]// Proceedings of the 2012 ACM/IEEE 13th International Conference on Grid Computing. IEEE Computer Society, 2012.

[17] GARG S K, VERSTEEG S, BUYYA R. A framework for ranking of cloud computing services[J]. Future Generation Computer Systems, 2013, 29(4): 1012-1023.

[18] SPEC[OL]. http://www.spec.org.

[19] Cloudharmony[OL]. http://www.cloudharmony.com.

[20] 朱华旻, 吴礼发, 康红凯. 基于 SecLA 的云服务商选择方法研究[J]. 计算机科学, 2016, 43(5): 100-107.

[21] 公茂果, 焦李成, 杨咚咚, 等. 进化多目标优化算法研究[J]. 软件学报, 2009, 20(2): 271-289.

[22] 胡旺, YEN G, 张鑫. 基于 Pareto 熵的多目标粒子群优化算法[J]. 软件学报, 2014(5): 1025-1050.

[23] Rackspace[OL]. http://www.rackspace.com.

[24] Digitalocean[OL]. http://www.digitalocean.com.

[25] Linode[OL]. http://www.linode.com.

[26] ZITZLER E, DEB K, THIELE L. Comparison of multiobjective evolutionary algorithms: Empirical results [J]. Evolutionary computation, 2000, 8(2): 173-195.

[27] SCHOTT J R. Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization[J]. Cellular Immunology, 1995, 37(1): 1-13.

[28] SBALZARINI I F, MÜLLER S, KOUMOUTSAKOS P. Multiobjective optimization using evolutionary algorithms[C]// Proceedings of the Summer Program. CiteSeer, 2000.

[29] Minitab[OL]. http://www.minitab.com/en-us.