动态可配置的两阶段虚拟机容错分配方法

陈 晓，江建慧

(同济大学软件学院 上海 嘉定区 201804)

【摘要】提出了一种随着虚拟机资源请求和应用程序可用性水平不断变化的两阶段虚拟机容错分配方法。第一阶段根据虚拟机资源请求变化求解不同的虚拟机初始分配方案集合，第二阶段通过虚拟机在线迁移与虚拟机热备份技术，根据应用程序可用性水平不断变化求解虚拟机容错分配方案。实验结果表明，与现有的方法相比，该文提出的两阶段虚拟机容错分配方法性能更好，系统可用度更高。

关 键 词 可用度；数据中心；动态可配置；虚拟机容错分配；两阶段算法

中图分类号 TP302.8 **文献标志码** A doi:10.3969/j.issn.1001-0548.2016.02.022

A Dynamic Configurable Two-Phase Virtual Machine Fault-Tolerance Allocation Method

CHEN Xiao and JIANG Jian-hui

(School of Software Engineering, Tongji University Jiading Shanghai 201804)

Abstract A two-phase virtual machine (VM) fault-tolerance allocation method is proposed according to the constantly change of resources request of VMs and the availability levels of applications. In the first stage, the initial allocation plans of VMs are solved according to the change of resources request of VMs. In the second stage, through live migration of VMs technology and hot standby of VMs technology, the fault-tolerance allocation plan of VMs is solved according to the constantly change of the availability levels of applications. Experimental results demonstrate that the proposed method shows up better performance and higher availability compared with the existing methods.

Key words availability; data center; dynamic configuration; fault-tolerance allocation of VMs; two-phase algorithm

云计算系统结构包括组织层、统一资源层、平台层与应用层[1]。虚拟化技术是云计算系统结构统一资源层中最关键的技术，可以提高主机的利用率，降低构建数据中心的成本。同时也会降低数据中心的可靠性，从2006年～2013年亚马逊云计算服务(amazon web services, AWS)宕机数据统计分析可知，排名前三位的分别是电源、存储以及虚拟机[2]。因此在数据中心服务器整合过程中，研究虚拟机管理的可靠性问题是当前的研究热点。虚拟机容错技术与数据中心服务器整合中虚拟机容错分配技术也随之被关注。

当前虚拟机容错技术主要分为两类：一类是将云计算系统失效节点上放置的虚拟机在线迁移到其他节点[3]。另一类是利用增量式检查点思想，实现虚拟机热备份系统，主机在运行过程中在一定的时间间隔内将修改后的数据同步到备份机[4-5]。

数据中心服务器整合主要分为虚拟机初始(静态)分配[6]和虚拟机动态管理[7]。已有工作研究以可靠性为目的的虚拟机容错分配问题。文献[8]研究在虚拟机初始分配阶段多种服务部署到不同的虚拟机时，怎样找到最小的物理节点数量，同时保证系统能容忍多节点失效问题，但是未考虑在动态管理阶段虚拟机分配方法。文献[9]研究多个物理节点连续失效时虚拟机的分配序列，在任意时刻保证总有一定数量的虚拟机正常运行，但是未考虑虚拟机动态分配过程中新的约束引入。文献[10]提出一种基于云计算应用组件排序的容错框架，首先找出云计算应用的关键组件，然后对这些关键组件选择合适的软

收稿日期：2015－02－11；修回日期：2015－10－27

基金项目：国家自然科学基金重点项目(61432017)；国家自然科学基金青年项目(61404092)；江苏省产学研联合创新资金项目子课题(BY2013095-5-06)

作者简介：陈晓(1987－)，男，博士，主要从事容错计算、软件可靠性等方面的研究.

件容错结构，提高云计算应用可靠性，但未考虑部署组件的虚拟机分配问题。文献[11]研究基于约束编程(constraint programming, CP)的可配置的虚拟机分配方法，提出了14种可配置应用程序可用性要求的约束，但是该方法只考虑了基于虚拟机在线迁移的容错结构，同时未考虑虚拟机初始分配对动态分配的影响。

为了解决上述问题，本文提出一种动态可配置的两阶段虚拟机容错分配方法，不仅考虑虚拟机在线迁移，还考虑虚拟机热备份，并提出一种基于CP的两阶段算法，研究虚拟机初始分配对动态可配置的虚拟机容错分配的影响。

1　虚拟机容错管理结构

1.1　系统结构

动态可配置虚拟机容错分配管理结构构建在云计算系统统一资源层，如图1所示。

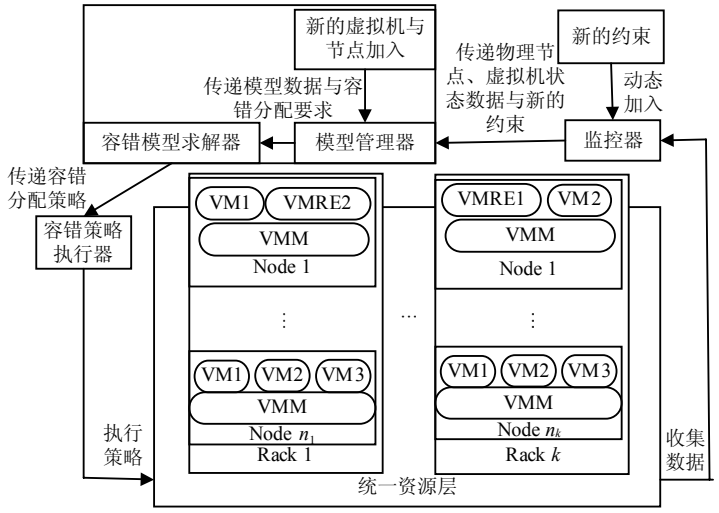

图1　动态可配置虚拟机容错分配管理结构

统一资源层上由 k 个机架(rack)构成集群，第 k 个机架放置 n_k 个物理节点(node)。每个物理节点的操作系统之上是虚拟机管理器(virtual machine manager, VMM)，其上放置的虚拟机有两类，一类是提供服务的虚拟机，另一类是服务虚拟机对应的冗余虚拟机(virtual machine replication, VMRE)。监控器负责接收VMM传递过来的数据，同时动态接收新的约束。全局控制器包含容错模型求解器与模型管理器。模型管理器接收来自监控器传递的数据和新的物理节点与虚拟机加入请求，并将这些数据按照容错模型求解器需要的格式处理，然后周期性地传递给容错模型求解器。容错模型求解器求解虚拟机分配策略，并将其传递给容错策略执行器。容错策略执行器将容错分配策略解析成VMM可执行的指令，然后传递给需要再次分配虚拟机的物理节点上的VMM并执行。

1.2　状态迁移

虚拟机容错分配策略是物理节点与虚拟机的一系列动作集合，这些动作使物理节点与虚拟机从一种状态变成另一种状态。

物理节点状态比较简单，共有运行(online)与关机(offline)两种状态，通过执行关闭与启动动作改变物理节点状态，物理节点状态迁移如图2所示。

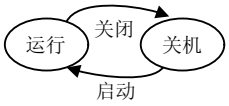

图2　物理节点状态迁移

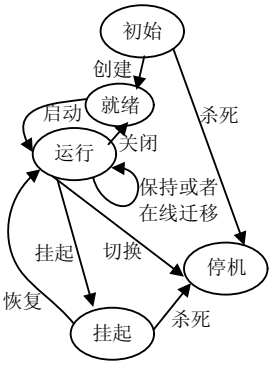

图3　虚拟机状态迁移

虚拟机的状态与操作系统进程的状态较类似，根据虚拟机管理软件(如Libvirt)对虚拟机定义状态，共包含5个状态，分别是：初始、就绪、运行、挂起

与停机，虚拟机状态迁移如图3所示。设虚拟机动作集合 $A=\{\text{Suspend, Launch, Migrate, Switch, Unmove, Close}\}$，分别表示虚拟机挂起、启动、迁移[12]、切换[4]、保持不变以及关闭，时间开销根据实际环境测定。

2 动态可配置虚拟机容错分配模型

2.1 虚拟机容错分配模型

根据图1中监控器反馈的数据，对那些分配条件发生变化的虚拟机，需要重新找出虚拟机行向量($\boldsymbol{V}$)与物理节点行向量($\boldsymbol{N}$)之间的映射关系，要求满足虚拟机资源请求，模型求解目标是使重新分配所花费的时间开销最小。为了求解该模型，将虚拟机容错分配分成两个阶段，第一阶段为虚拟机初始分配(virtual machine initial allocation, VMIA)，第二阶段为动态可配置虚拟机容错分配(dynamic configurable virtual machine fault tolerance allocation, DCVMFTA)。

根据1.2小节虚拟机状态迁移描述，可以将虚拟机的迁移动作看作从初始(initial)状态到最终(final)状态的迁移。设 n 表示物理节点数量、v 表示虚拟机数量、p 表示物理资源的种类(资源维度)；$\boldsymbol{N}_i$ 表示物理节点 i 的 p 维资源向量，$i=1,2,\cdots,n$；$\boldsymbol{V}_j^{\text{ini}}$ 表示虚拟机 j 初始状态对资源的需求，$j=1,2,\cdots,v$；$\boldsymbol{V}_j^{\text{fin}}$ 表示虚拟机 j 最终状态对资源的需求；a_j 表示虚拟机 j 的执行动作；d_{tj} 表示虚拟机 j 执行状态转移动作 t 的时间，$t\in\boldsymbol{A}$；c_{tj} 表示虚拟机 j 执行状态转移动作 t 的时刻；w_{tj} 表示虚拟机 j 执行状态转移动作 t 的权重(Unmove的权重为0，其他动作的权重均为1)；X 表示已使用物理节点数量，x_i 表示若存在虚拟机放置在物理节点 i，其值为1，否则为0。为了形式化描述两阶段问题，给出两个定义。

定义 1 虚拟机行向量 $\boldsymbol{V}$ 与物理节点行向量 $\boldsymbol{N}$ 的映射函数为 g，即 $g:\boldsymbol{V}\to\boldsymbol{N}$。

定义 2 四元组 $\boldsymbol{T}=(\boldsymbol{V},g_{\text{ini}}(\boldsymbol{V}),\boldsymbol{A},g_{\text{fin}}(\boldsymbol{V}))$ 表示 $\forall$ 虚拟机 $j\in\boldsymbol{V}$，初始被分配到物理节点 $g_{\text{ini}}(j)$，然后经过执行动作 $a\in\boldsymbol{A}$，最终被分配到物理节点 $g_{\text{fin}}(j)$，这一执行过程表示虚拟机在动态管理过程中的状态转移。

1) VMIA。VMIA主要研究 $\boldsymbol{V}$ 与 $\boldsymbol{N}$ 的映射方案，要求满足虚拟机资源请求与物理节点的资源匹配限制条件，目标是使用的物理节点数量最少。式(1)给出VMIA问题的目标函数，即使用的物理节点数量最小。式(2)给出VMIA的约束条件，即放置在物理节点上的所有虚拟机每个维度的资源请求总和不能超过该物理节点所拥有的资源。

$$\text{目标：}\quad \text{Min}(X)=\text{Min}\left(\sum_{i=1}^{n}x_i\right) \tag{1}$$

$$\text{约束：}\quad \sum_{j\in g_{\text{ini}}^{-1}(i)}\boldsymbol{V}_j^{\text{ini}}\leqslant\boldsymbol{N}_i \tag{2}$$

根据定义1可知，$g_{\text{ini}}^{-1}(i)$ 表示放置在物理节点 i 上的虚拟机集合。

2) DCVMFTA。DCVMFTA主要研究在VMIA求解方案基础上，随着虚拟机分配条件的变化，求解 $\boldsymbol{V}$ 与 $\boldsymbol{N}$ 的再次映射方案。虚拟机分配条件发生变化主要有两类情况：一类是随着应用程序的变化，某些虚拟机对物理资源请求的增加或者减少；另一类是虚拟机可用性水平的动态配置。式(3)给出了求解DCVMFTA问题的目标函数，表示所有虚拟机的状态迁移执行动作的时间开销之和最小。式(4)给出DCVMFTA的约束条件，表示任意时刻对任意虚拟机 $j\in\boldsymbol{V}$，在执行状态迁移动作之前时刻($l<d_{tj}+c_{tj}$)的资源请求与执行状态迁移动作之后时刻($l\geqslant c_{tj}$)的资源请求之和小于放置之上的物理节点 i 所拥有的资源。其中，l 表示任意时刻。

$$\text{目标：}\quad M=\text{Min}\sum_{j\in\boldsymbol{V}}w_{tj}\times(d_{tj}+c_{tj}) \tag{3}$$

$$\text{约束：}\quad \sum_{\substack{j\in g_{\text{ini}}^{-1}(i)\\ l<d_{tj}+c_{tj}}}\boldsymbol{V}_j^{\text{ini}}+\sum_{\substack{j\in g_{\text{fin}}^{-1}(i)\\ l\geqslant c_{tj}}}\boldsymbol{V}_j^{\text{fin}}\leqslant\boldsymbol{N}_i \tag{4}$$

2.2 可配置虚拟机容错分配描述

根据图1可知，监控器动态接收新的约束，这些约束主要为提高数据中心可靠性虚拟机与物理节点映射的约束规则。共有两类约束，分别是分散(spread)与冗余(redundancy)。

1) Spread表示给定的虚拟机集合中任意两个虚拟机不能放置在同一物理节点上，主要是为了防止单点故障而导致虚拟机与其对应的冗余虚拟机同时失效。Spread的描述为spread($\boldsymbol{V}$)，其约束模型为：

$$g_{\text{fin}}(j_1)\neq g_{\text{fin}}(j_2)\quad\forall j_1,j_2\in\boldsymbol{V} \tag{5}$$

2) Redundancy表示给定的虚拟机集合的冗余水平，主要是对提供关键服务的虚拟机进行冗余备份并同步。Redundancy的描述为redundancy($\boldsymbol{V},k$)，k 为虚拟机冗余数量，一般 $k=2$ 时，属于虚拟机双机热备份，其约束模型如下：

$$\forall j\in\boldsymbol{V},\boldsymbol{V}_j=\{j_b\mid b=1,2,\cdots,k\},\forall j_1,j_2\in\boldsymbol{V}_j,$$
$$g_{\text{ini}}(j_1)\neq g_{\text{ini}}(j_2)\,\text{and}\,g_{\text{fin}}(j_1)\neq g_{\text{fin}}(j_2) \tag{6}$$

2.3　系统可用度(availability)

虚拟机容错分配系统周期性地对无法满足需求的虚拟机进行再次分配，因此，可以量化动态可配置的虚拟机容错分配系统的可用度[11]。完成一次虚拟机容错分配所需要的时间为求解时间(Ts)与预测分配时间(Tr)之和，用(Ts+Tr)表示。定义系统平均修复时间(mean time to repair, MTTR)：$\mathrm{MTTR}=(\mathrm{Ts}+\mathrm{Tr})\times\sigma$，$\sigma$ 表示需要重新分配的虚拟机比例。设虚拟机重新分配的时间周期为Tp，那么系统可用度 A 如式(7)所示，Tp一般取值为1小时。

$$A=\frac{\mathrm{Tp}}{\mathrm{Tp}+\mathrm{MTTR}}=\frac{\mathrm{Tp}}{\mathrm{Tp}+(\mathrm{Ts}+\mathrm{Tr})\times\sigma} \tag{7}$$

3　基于CP的两阶段算法

由2.1节的描述可知，VMIA与DCVMFTA属于约束满足问题，因此可以用CP方法求解该类问题[9,11]。两阶段算法流程如图4所示。第一阶段基于CP方法求解VMIA问题，产生虚拟机分配方案 $\boldsymbol{X}$，然后进入第二阶段循环选择 $\boldsymbol{X}$ 并对DCVMFTA问题求解，输出虚拟机容错分配方案 $\boldsymbol{P}$。

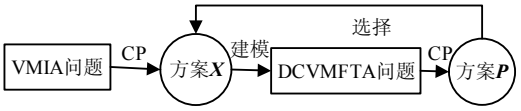

图4　两阶段算法流程

3.1　VMIA问题求解

基于CP方法求解VMIA问题一般需要搜索较大的解空间，因此求解时间非常大，为了减少算法搜索空间，可以采用等价类与上下界限定优化方法。在VMIA问题求解中，等价虚拟机指任意两个虚拟机的请求资源完全相同；等价物理节点指任意两个物理节点所拥有的资源完全相同。

在VMIA问题求解中，所需物理节点数量的下界为大于所有虚拟机请求的每个维度资源总和除以对应的所有物理节点资源总和的最大值，设下界为 X_{lower}，那么 $X_{\mathrm{lower}}\geqslant \mathrm{Max}\left\{\left\lceil\frac{\sum_{\forall j\in \boldsymbol{V}}\boldsymbol{V}_j^{\mathrm{ini}}}{\sum_{\forall i\in \boldsymbol{N}}\boldsymbol{N}_i}\right\rceil\right\}$。

上界的选取规则为：若启发式算法求解结果存在，则为上界，否则选择虚拟机数量与物理节点数量的最小值，以降序首次适应(first fit descending, FFD)算法为例，X_{ffd} 为FFD算法求解结果，设上界为 X_{upper}，有：

$$X_{\mathrm{upper}}\leqslant\begin{cases}X_{\mathrm{ffd}} & \text{if } \exists X_{\mathrm{ffd}}\\ \mathrm{Min}(n,v) & \text{else}\end{cases} \tag{8}$$

基于CP的VMIA问题求解步骤如下：

1) 初始化相关资源，包括 $\boldsymbol{N}$、$\boldsymbol{V}$、$\boldsymbol{N}_i$、$\boldsymbol{V}_j^{\mathrm{ini}}$、计数变量 z 以及迭代次数阈值 $N_{\max}$；

2) 对物理节点向量 $\boldsymbol{N}$、虚拟机向量 $\boldsymbol{V}$ 按等价类方式从大到小排序，排序规则是按照内存、CPU资源依次从大到小，排序后物理节点向量 $\boldsymbol{N}'$、虚拟机向量 $\boldsymbol{V}'$；

3) 将VMIA问题约束转换成CP约束，并设置搜索空间的上下界，即 X_{upper}、X_{lower}，进入模型器求解；

4) 若找到可行解，则将其放入方案向量 $\boldsymbol{X}$，更新计数变量 $z=z+1$；

5) 若 $z<N_{\max}$，则转到步骤3)，否则转步骤6)；

6) 输出分配方案集合 $\boldsymbol{X}$。

3.2　DCVMFTA问题求解

DCVMFTA问题是在VMIA问题的方案之上求解。DCVMFTA问题等价物理节点与VMIA问题相同，而等价虚拟机指对任意两个虚拟机，它们在同一个物理节点上并且请求的资源完全相同。DCVMFTA问题求解的搜索空间上下界与VMIA问题相同。基于CP的DCVMFTA问题求解步骤如下：

1) 根据监控器的数据获取需要重新分配的虚拟机向量 $\boldsymbol{V}$，初始化虚拟机动作向量 $\boldsymbol{A}$、相应的时间 d_{ij}，时间计数变量 $o=0$、计算时间阈值 $t_{\max}$，以及预测最优分配时间 $M_{\mathrm{best}}=0$；

2) 从方案 X 中选取未被选择的方案 x，若 x 存在，则转步骤3)；否则，转步骤6)；

3) 若 $o<t_{\max}$，则将DCVMFTA问题约束转换成CP约束；否则设置本次分配方案 pl 为空，转步骤2)；

4) 运行基于CP的模型求解器，如果分配方案 pl 不存在，则转步骤5)，更新时间计数变量 $o=o+o_{\mathrm{pl}}$，o_{pl} 为本次求解时间；否则，保存 pl，更新 M_{best}，若 $M_{\mathrm{best}}>M_{\mathrm{pl}}$，则 $M_{\mathrm{best}}=M_{\mathrm{pl}}$，重置 $o=0$，转步骤2)；

5) 扩充 $\boldsymbol{V}$，若 $\boldsymbol{V}$ 非全集，转步骤3)；否则设置本次分配方案 pl 为空，转步骤2)；

6) 输出分配方案集合 $\boldsymbol{P}$ 与预测最优分配方案执行时间 M_{best}。

4　实验与分析

本文在BtrPlace[11]基础上实现了动态可配置的两阶段虚拟机容错分配原型系统HABtrPlace。选取

了基准程序RUBiS(rice university bidding system)[11]模拟真实云计算服务实例评估其性能，实例基于RUBiS负载的三层结构。模拟3 000～7 000个虚拟机在1 500个物理节点上动态可配置容错分配过程，这些虚拟机对应的实例个数依次为174、233、295、335、402，比较FFDPlace[8]、BtrPlace与HABtrPlace，求解DCVMFTA问题的时间与系统可用度。

实验求解的硬件配置为：CPU采用双路E5-2620 V2，主频： 2.1 GHZ，内存：32 GB，操作系系统：centos 6.4 x86_64，Java SDK版本1.8 update 5。

图5给出了不同虚拟机容错分配方法求解DCVMFTA问题的时间。可以看出，随着待分配虚拟机数量增多，DCVMFTA问题求解时间增加，HABtrPlace求解时间最小，说明随着求解问题规模增加，本文提出的方法具有更小的求解时间。

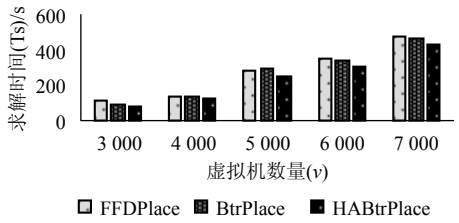

图5 不同分配方法的DCVMFTA问题求解时间

图6给出不同虚拟机容错分配方法的系统可用度。从图中可以看出，随着待分配虚拟机数量的增多，基于HABtrPlace的虚拟机容错分配方法可用度最高，说明随着求解问题规模的增加，本文提出的虚拟机容错分配方法具有更高系统可用度。同时，从图中可知，系统可用度均在0.99以上，并且随着待分配的虚拟机数量增加，系统可用度随之下降，这是因为待分配的虚拟机数量越多，则求解时间与预测再次分配的执行时间越大，系统可用度随之下降。

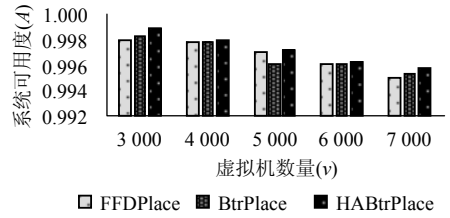

图6 不同虚拟机容错分配方法的系统可用度

图7给出VMIA问题求解方案对DCVMFTA问题求解时间影响。从图中可以看出，随着物理节点放置虚拟机数量限制增多，DCVMFTA问题求解时间反而下降，这是由于使用的物理节点越多，虚拟机在物理节点的放置越分散，随着新的可靠性约束条件动态配置，需要再次分配的虚拟机数量会随之减小。

图8给出VMIA问题求解方案对系统可用度影响，可看出，随着物理节点放置虚拟机数量限制增多，系统可用度随之增大，这是由于随着使用的物理节点数量增多，DCVMFTA问题求解时间与预测执行时间随之减小，这样系统可用度则随之增大。

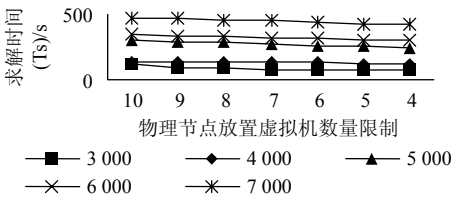

图7 VMIA对DCVMFTA问题求解时间影响

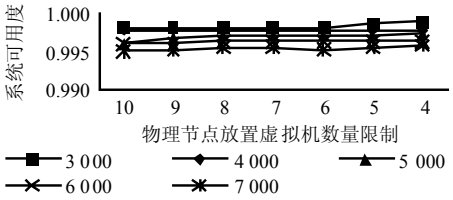

图8 VMIA问题求解方案对系统可用度影响

从图5～图8的分析可知，本文提出的虚拟机容错分配方法求解时间更小，系统可用度更高。同时，DCVMFTA问题求解时间与系统可用度不仅与问题规模相关，还受VMIA问题求解方案的影响。

5 结 束 语

针对数据中心服务器整合的可靠性问题，本文提出一种动态可配置的两阶段虚拟机容错分配方法。随着虚拟机请求资源和用户对应用程序可用性水平的不断变化，通过虚拟机在线迁移与虚拟机热备份技术，构建了一种动态可配置的两阶段虚拟机容错分配模型，提出了一种基于CP的两阶段算法求解该模型。实验结果表明，与现有的方法相比，本文提出的方法求解时间更少，系统可用度更高。下一步研究工作考虑其他因素对虚拟机容错分配的影响，例如，功耗、云计算系统服务等级合约等。

参 考 文 献

[1] FOSTER I, YONG Z, RAICU I, et al. Cloud computing and grid computing 360-degree compared[C]//Proceedings of Grid Computing Environments Workshop. Austin, TX: IEEE, 2008: 1-10.

[2] CHEN Huai-lin. A qualitative and quantitative study on

availability of cloud computing[EB/OL]. [2013-10-22]. http://www.valleytalk.org/w p-content/uploads/2013/10/.

[3] XU F, LIU F M, LIU L H, et al. iAware: Making live migration of virtual machines interference-aware in the cloud[J]. IEEE Transactions on Computers, 2014, 63(12): 012-3025.

[4] BRENDAN C, GEOFFREY L, DUTCH M, et al. Remus: High availability via asynchronous virtual machine replication[C]//Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation. San Francisco: USENIX Association, 2008: 161-174.

[5] ZHU J, JIANG Z F, XIAO Z, et al. Optimizing the performance of virtual machine synchronization for fault tolerance[J]. IEEE Transactions on Computers, 2011, 60(12): 1718-1729.

[6] WANG Y F, WANG X R. Virtual batching: Request batching for server energy conservation in virtualized data centers[J]. IEEE Transactions on Parallel and Distributed Systems, 2013, 24(8): 1695-1705.

[7] XIAO Z, SONG W J, CHEN Q. Dynamic resource allocation using virtual machines for cloud computing environment[J]. IEEE Transactions on Parallel and Distributed Systems, 2013, 24(6): 1107-1117.

[8] MACHIDA F, KAWATO M, MAENO Y. Redundant virtual machine placement for fault-tolerant consolidated server clusters[C]//Proceedings of 2010 IEEE Network Operations and Management Symposium. Osaka: IEEE, 2010: 32-39.

[9] BIN E, BIRAN O, BONI O, et al. Guaranteeing high availability goals for virtual machine placement[C]// Proceedings of the 31st International Conference on Distributed Computing Systems. Minneapolis: IEEE, 2011: 700-709.

[10] ZHENG Z B, ZHOU T C, LYU M R, et al. Component ranking for fault-tolerant cloud applications[J]. IEEE Transactions on Services Computing, 2012, 5(4): 540-550.

[11] HERMENIER F, LAWALL J, MULLER G. BtrPlace: a flexible consolidation manager for highly available applications[J]. IEEE Transactions on Dependable and Secure Computing, 2013, 10(5): 273-286.

[12] LIU H K, JIN H, XU C Z. Performance and energy modeling for live migration of virtual machines[C]// Proceedings of the 20th International Symposium on High Performance Distributed Computing. San Jose: IEEE, 2011: 171-182.

编 辑 蒋 晓