doi:10.11959/j.issn.1000-436x.2016082

# 基于合作博弈的多虚拟机实时迁移带宽分配机制

崔勇[1,2]，林予松[2]，李润知[2]，王宗敏[2]
（1. 郑州大学信息工程学院，河南 郑州 450001;2. 郑州大学信息网络省重点开放实验室，河南 郑州 450052）

**摘 要：**IaaS 云计算平台采用虚拟机实时迁移技术进行资源动态调度和管理。在实际应用场景下，需要并行实时迁移多个虚拟机。由于实时迁移算法本身以最大利用带宽的方式进行数据传输，存在着迁移进程间竞争带宽的问题，无法保证带宽全局最优分配，影响整体迁移的性能。提出一种基于合作博弈的多虚拟机实时迁移带宽分配机制，将带宽分配问题建模为一个纳什议价，通过求解纳什议价解得到帕累托最优的带宽分配方案，并在实际的虚拟化平台上进行了实现。实验结果表明，相比标准的并行实时迁移，所提出的带宽分配机制能够公平有效地分配带宽，提高了并行实时迁移的性能。

**关键词：**实时迁移；虚拟机；带宽分配；纳什议价；合作博弈
**中图分类号：**TP393 **文献标识码：**A

# Cooperative game based bandwidth allocation mechanism live migration of multiple virtual machines

CUI Yong[1,2], LIN Yu-song[2], LI Run-zhi[2], WANG Zong-min[2]
(1. Institute of Information Engineering, Zhengzhou University, Zhengzhou 450001, China;
2. Key Lab on Information Networking, Zhengzhou University, Zhengzhou 450052, China)

**Abstract:** In IaaS cloud computing platform, live migration of virtual machines plays a significant role in resource dynamic dispatching and management. Under many scenarios, multiple virtual machines in the same server need to be moved online concurrently.Since live migration algorithm utilizes bandwidth as more as possible to send data, there is a bandwidth competition among the concurrent migration processes, which cannot guarantee the optimal allocation for the bandwidth and finally degrades the overall performance of the parallel live migration. A cooperative game based bandwidth allocation mechanism in live migration of multip virtual machines was proposed, which models the bandwidth allocation problem as a Nash bargaining game and attains a desirable bandwidth sharing scheme that guarantees Pareto optimality. Experiment results show that the proposed method can assign the bandwidth fairly and efficiently and improve the performance of the parallel live migration.

**Key words:** live migration, virtual machine, bandwidth allocation, Nash bargaining, cooperative game

## 1 引言

云计算平台采用虚拟化技术提供 IaaS (infrastructure as a service)[1]，将 IT 资源以虚拟机（VM, virtual machine）的形式封装起来进行管理、调度和交付，达到高利用率、动态性和可扩展性的目的。

虚拟机实时迁移作为 IaaS 云计算的底层核心技术，是实现上述特性的关键，也是构成 IaaS 云平台一些高级功能的基础，如在线设备维护、在线负载均衡、自动电源管理等。虚拟机实时迁移是指将

收稿日期：2015-05-11；修回日期：2015-10-27
通信作者：王宗敏，goodaliens@126.com
基金项目：教育部博士点专项科研基金资助项目（No.20114101110007）；河南省科研重点基金资助项目（No.13A520562）；河南省创新人才基金资助项目（No.2011HASTIT003）
**Foundation Items:** The Specialized Research Fund for the Doctoral Program for Higher Education(No.20114101110007), The Key Research Project of Henan Province Department of Education(No.13A520562), The Innovative Talent Project of Henan Province Department of Education(No.2011HASTIT003)

一台或多台虚拟机在线更换宿主机，迁移过程中虚拟机中的任务正常运行。主流的虚拟机管理器（如 Qemu-kvm[2]）采用 Pre-copy 算法[3]进行单虚拟机实时迁移，其基于一个迭代拷贝的过程，首轮拷贝全内存镜像，后轮不断将前轮拷贝过程中产生的脏页传输到目的端，直到剩余脏页足够小或者达到指定迭代次数，最后在一个短暂的停机时间内将虚拟机完全切换过去。

IaaS 云平台中经常会出现并发实时迁移多台虚拟机的情景[4]，其中，各迁移进程只是独立运行 Pre-copy 算法并发执行，缺乏统一管理和协调[5]。因为 Pre-copy 在单个迁移进程内以最大利用带宽方式发送数据，并发迁移进程间竞争带宽资源，导致各迁移进程无法公平均衡地使用带宽。又由于 Pre-copy 对带宽敏感，若无法保证传输带宽将直接影响算法效率[6]。因此从全局来看，需要有效的带宽分配机制来保证整体迁移的性能。本文运用合作博弈理论，将多虚拟机并行实时迁移的带宽竞争建模为一个纳什议价，通过求解纳什议价解(NBS, Nash bargaining solution)得到全局上公平有效的带宽分配方案，保证帕累托最优（Pareto-optimal），进而提高整体迁移性能。

目前，已经有学者采用合作博弈理论解决一些带宽分配问题，如文献[7]。本文和文献[7]虽然都是运用合作博弈理论解决带宽最优分配问题，但是本文的研究背景不同。文献[7]是解决宽带网络中应用服务的带宽占用问题，而本文是解决多虚拟机并行迁移的带宽分配问题，因而虽然与文献[7]所采用的建模方法和数学理论类似，但是解决的是不同的问题，具体体现在建模的前提条件、约束条件和结论都不相同，而且文献[7]的结论是一个分布式方法，本文的结论是一个集中式方法。本文的创新之处是：首次将多虚拟机并行迁移的带宽分配问题建模为一个纳什议价过程，并通过求解凸优化问题得到一个新颖的最优带宽分配方案，提高了整体迁移性能。

## 2 多虚拟机并行实时迁移问题描述

设待迁移的虚拟机个数为 $N$，总带宽为 $B$，总迁移时间为 $T_{\text{all}}$，Pre-copy 停机阈值为脏页率不高于 $Q$ 或达到迭代次数 $R$。虚拟机 $VM_i(i\in\{1,2,\text{L},N\})$ 的虚拟内存为 $M_i$，迁移时间为 $T_i$，实际迁移带宽为 $r_i$(满足 $\sum_{i=1}^{N} r_i \le B$)，迭代拷贝次数为 $K_i$，第 $j$ 轮拷贝产生的脏页率为 $D_{ij}\ (j\in\{1,2,\text{L},K_{i-1}\})$，则有 $K_i=\{j+1\mid D_{ij}\le Q \text{ 或 } j=R\}$。其中，第 1 轮拷贝全内存页，第 $K_i$ 轮进行停机拷贝，即传输剩余脏页和设备状态数据。

令实时迁移 $VM_i$ 的第 $j$ 轮的拷贝时间为 $t_{ij}\ (j\in\{1,2,\text{L},K_i\})$，设备状态传输时间和切换时间总和为 $T_p$（相比迭代拷贝时间，此处可看作一个常数），则 $VM_i$ 的迁移时间为

$$T_i=\sum_{j=1}^{K_i} t_{ij}=t_{i1}+\sum_{j=2}^{K_i} t_{ij}+T_p \tag{1}$$

因为 $t_{ij}=\dfrac{t_{i(j-1)}D_{i(j-1)}}{r_i},\ j\in\{2,\text{L},K_i\}$，而 $t_{i1}=\dfrac{M_i}{r_i}$，则 $t_{ij}=\dfrac{M_i}{r_i^{\,j}}\prod_{m=1}^{j-1}D_{im}=\dfrac{M_i}{r_i}\prod_{m=1}^{j-1}\dfrac{D_{im}}{r_i},\ j\in\{2,\text{L},K_i\}$，则

$$T_i=\frac{M_i}{r_i}\left(1+\sum_{j=2}^{K_i}\prod_{m=1}^{j-1}\frac{D_{im}}{r_i}\right)+T_p \tag{2}$$

那么，对 $N$ 个虚拟机进行并行实时迁移的总时间为

$$T_{\text{all}}=\max\{T_i\},\ i\in\{1,2,\text{L},N\} \tag{3}$$

从上述可知，单个迁移进程的迁移时间由虚拟机内存大小、脏页率以及传输带宽决定。显然，前 2 个因素在迁移时已确定，则传输带宽为优化迁移性能的关键因素。由式（2）可知，传输带宽与迁移时间成反比，且当带宽不低于脏页率时，才能保证实时迁移的性能[6]。式（3）说明多虚拟机并行实时迁移的总时间由迁移时间最长的迁移进程决定，而单机的迁移时间又由其实际传输带宽确定。由于目前并行迁移中缺少带宽分配机制，存在并发进程竞争带宽的问题，因此无法保证总带宽在各迁移进程间的有效分配，必然影响到总迁移时间。

## 3 基于合作博弈的带宽分配机制

本文从全局最优的角度出发，在多虚拟机并行实时迁移中提出一种基于合作博弈的带宽分配机制，提高并发迁移性能。

### 3.1 合作博弈论相关概念

博弈论[8]研究如何有效处理具有利益冲突或竞争资源的多个局中人之间的策略选择及均衡问题。其中，合作博弈论作为博弈论的一个重要分支，侧重于集体理性，强调效率、公平、公正，通过博弈的局中人在达成一个有约束力的协议的基础上进行合作，使

各方的利益都能有所增加，达到共赢的目的。

约翰·纳什提出纳什议价模型来解决合作博弈问题[9]。其首先设置参与博弈的局中人的初始收益及收益函数，然后通过局中人之间议价过程获得博弈的均衡解，即纳什议价解。该解满足纳什公投（个体理性、帕累托最优、线性变换无关、独立于无关选择条件和对称性条件）。纳什采用纳什积代表全局收益，并证明使纳什积最大的解就是纳什均衡解，实现博弈结果的全局利益最大化及公平。

### 3.2　并行实时迁移带宽分配问题建模

在多虚拟机并行实时迁移中，$N$ 个并行迁移进程作为局内人竞争迁移带宽资源，带宽总量为 $B$，带宽分配方案定义为向量 $\boldsymbol{r}=\{r_1, \perp, r_N\}$，则迁移进程 $LP_i(i\in\{1,\perp,N\})$ 所需的带宽为 $r_i$ ，且 $\sum_{i=1}^{N} r_i \le B$。由第 2 节分析可知，单个迁移进程的带宽需求由各待迁移的虚拟机产生的脏页率决定，定义对应 $LP_i$ 的虚拟机的最小和最大脏页率分别为 $DL_i$ 和 $DH_i$，则有 $DL_i \le r_i \le DH_i$。由于每个迁移进程都期望获得对应脏页率的带宽来保证实时迁移性能，则将收益函数直接定义为 $f(\cdot)=r_i$，初始收益由 $DL_i$ 表示，那么每个迁移进程的议价目标就是在获得初始收益（带宽）$DL_i$ 的基础上，尽可能获得接近最大带宽需求 $DH_i$ 的带宽。出于简化，这里假设 $N$ 个迁移进程的 $DL_i$ 之和是严格小于 $B$ 的。当然，若在实际中出现不符合该条件的情况，也可将迁移进程分组进行，保证组内 $DL_i$ 之和满足条件即可。

定义 $X\subset R^N$ 为 $N$ 个迁移进程的带宽分配向量空间，且 $X$ 为一非空有界闭凸集，则 $X$ 表示所有的带宽分配方案。令初始带宽向量为 $\boldsymbol{r}^0=\{DL_i, \perp, DL_N\}$，则定义向量 $\boldsymbol{U}=\{r\mid r\in X, r_i\ge DL_i, i\in\{1,\perp,N\}\}\neq\varnothing$ 为所需要求得带宽分配方案集合，则定义 $G=\{(X,\boldsymbol{r}^0)\}$ 为一个纳什议价问题，其表示根据初始带宽所能获得的带宽分配方案。那么该纳什议价问题的纳什议价解 NBS（即具备帕累托最优及公平的带宽分配方案）定义如下。

**定义 1**　当一个映射 $S$:$G$? $R^N$ 满足如下条件：$S(X, r^0)\in\boldsymbol{U}$、帕累托最优、线性变换无关、独立于无关选择和对称性条件[9]，则称 $S$ 为一个纳什议价解 NBS。

又定义 $J=\{j\mid \boldsymbol{r}\in\boldsymbol{U}, r_j>DL_j, j\in\{1,\perp,N\}\}$ 为分配带宽时能够严格高于初始带宽值的迁移进程的集合。假定 $f_i(\cdot)=r_i$ 是在 $\boldsymbol{U}$ 上的单射函数，若 $J$ 不空，则有以下定理[9]。

**定理 1**　$G$ 存在一个唯一的纳什议价解，且向量 $\boldsymbol{r}=S(X,\boldsymbol{r}^0)$ 是以下优化问题的最优解

$$\max\prod_{j\in J}(r_j-DL_j), \boldsymbol{r}\in\boldsymbol{U}$$

上述问题可以转化为一个等价的凸优化问题[9]为

$$\max\sum_{j\in J}\ln(r_j-DL_j), \boldsymbol{r}\in\boldsymbol{U}$$

后者有唯一解，且两者等价，因此后者的唯一解就是纳什议价解。

以上优化问题是在 $J$ 不为空的假设下成立的，而对于不属于 $J$ 的迁移进程所分配的带宽就是初始带宽，即 $r_j=DL_j$，则在优化问题中不考虑这些迁移进程。也就是说，在那些能够获得高于初始带宽的迁移进程间进行带宽优化分配，而对于不满足该条件的迁移进程直接分配初始带宽。又因为前面假设初始带宽的和严格小于总迁移带宽，那么总存在能够获得高于初始带宽的迁移带宽，因此在以下优化问题中 $J$ 不为空条件总成立。

综合纳什议价模型和实际约束条件，令实际可取的带宽分配向量空间为

$$X_0=\{\boldsymbol{r}\in R^N\}\mid DL_i\le r_i\le DH_i,$$
$$i\in\{1,\perp,N\}\text{且}\sum_{i=1}^{N} r_i\le B$$

则可以得到以下带约束的优化问题 $S$，该问题的解就是并发迁移进程针对获得公平有效的带宽分配方案进行纳什议价的 NBS

$$\begin{aligned}&\max_{\{r\}}\sum_{i=1}^{N}\ln(r_i-DL_i)\\ &\text{s.t.}\quad r_i\le DH_i,\forall i\in\{1,\perp,N\},\\ &\qquad r_i\ge DL_i,\forall i\in\{1,\perp,N\},\\ &\qquad\sum_{i=1}^{N} r_i\le B\end{aligned}$$

**命题 1**　假设 $\sum_{i=1}^{N} DL_i<B$ 成立，则存在 $g\ge 0, b_i\ge 0(i\in\{1,\perp,N\})$ 使

$$\boldsymbol{r}_i^*=\min\{DH_i, DL_i+\frac{1}{g}\}, \forall i\in\{1,\perp,N\} \qquad (4)$$

$$r_i^*\le DH_i, \forall i\in\{1,\perp,N\}$$

$$\sum_{i=1}^{N} r_i^*\le B$$

$$g(\sum_{i=1}^{N} r_i^*-B)=0$$

其中，$r_i^*$ 为优化问题 $S$ 的最优解，也是唯一的 NBS。

**证明** 因为假设 $\sum_{i=1}^{N} DL_i < B$ 成立，则 $X_0$ 是一个非空有界闭凸集。定义

$$q(\boldsymbol{r}) = \sum_{i=1}^{N} \ln(r_i - DL_i)$$

而文献[7]已经证明 $q(\cdot): X_0 \to R^+$ 是一个严格凹函数（因篇幅所限，这里不再引述），又根据约束条件的不等式定义

$$f_1(r_i) = r_i - DH_i,\ f_2(r_i) = DL_i - r_i,\ f_3(r_i) = \sum_{i=1}^{N} r_i - B$$

则 $f_1$、$f_2$、$f_3$ 关于$\{r_i\}$都是线性的，即三者都是凸函数。综上，因为$q$ 是凹函数且 $f_1$、$f_2$、$f_3$均是凸函数，则根据凸优化问题定义，问题 $S$ 是一个凸优化问题[10]。因此，本文定义凸优化问题 $S$ 的拉格朗日函数为 $L(r,a,b,g)$ ,其中， $a_i \geq 0$ ， $b_i > 0(\forall i \in \{1, \text{L}, N\})$ $g \geq 0$ 分别是对应问题 $S$ 中带宽不小于 $DL_i$、不大于 $DH_i$ 和不大于总带宽 3 个约束式的拉格朗日乘子，则有

$$L(r,a,b,g) = q(r) - \sum_{i=1}^{N} a_i (DL_i - r_i) - \sum_{i=1}^{N} b_i (r_i - DH_i) - g\left(\sum_{i=1}^{N} r_i - B\right)$$

又因为$q(r)$ 一阶可导，则问题 $S$ 的最优解 $r^*$ 满足 KKT（Karush-Kuhn-Tucker）条件[10]

$$\nabla L(r^*,a,b,g) = 0 \Leftrightarrow$$

$$\frac{1}{r_i^* - DL_i} + a_i - b_i - g = 0, i \in \{1, \text{L}, N\}$$

且

$$a_i (r_i^* - DL_i) = 0, i \in \{1, \text{L}, N\}$$

$$b_i (r_i^* - DH_i) = 0, i \in \{1, \text{L}, N\}$$

$$g\left(\sum_{i=1}^{N} r_i^* - B\right) = 0$$

在以上条件中，因为假设 $\sum_{i=1}^{N} DL_i < B$ 成立，显然存在 $r_i^* > DL_i$ ，则 $a_i$ 必然为 0。另外，若有 $r_i^* < DH_i$ 存在，则 $b_i$ 必为 0，否则 $r_i^* = DH_i, \forall i \in \{1, \text{L}, N\}$ ，此时各迁移进程都能分配到最大需求带宽，是一种特殊情况。

综上所述，根据取得最优解的条件可推出命题 1 的结论，证毕。

### 3.3 带宽分配方案的求解

式(4)给出了问题 $S$ 最优解的形式，可以看出最优解的取值的关键在于求解拉格朗日乘子。本文采用对偶分解（dual decomposition）方法[11]来求解最优解。首先定义与优化问题等价的原问题 $P$(primal problem）为

$$\min_{\{r\}} G(r) = -\sum_{i=1}^{N} \ln(r_i - DL_i)$$

$$\text{s.t.} \quad r_i \leq DH_i, \forall i \in \{1, \text{L}, N\},$$

$$r_i \geq DL_i, \forall i \in \{1, \text{L}, N\},$$

$$\sum_{i=1}^{N} r_i \leq B$$

这里依然假设 $\sum_{i=1}^{N} DL_i < B$ 成立，则如第 3.2 节所述，约束条件 $r_i \geq DL_i$ 对应的拉格朗日乘子为 0。同样，对于 $r_i = DH_i, \forall i \in \{1, \text{L}, N\}$ 成立的这种特殊情况，直接分配最大需求带宽，不在优化问题中处理。则定义原问题 $P$ 的拉格朗日函数 $L(\cdot): X_0 \times R \to R$ 定义为

$$L(\boldsymbol{r},g) = -\sum_{i=1}^{N} \ln(r_i - DL_i) + g\left(\sum_{i=1}^{N} r_i - B\right)$$

其中，拉格朗日乘子 $g$ 也称作对偶变量，则原问题 $P$ 的对偶函数（dual function） $d(\cdot): R \to R$ 定义为

$$d(g) = inf_{r \in X_0} L(\boldsymbol{r},g)$$

因为 $P$ 是个可分解问题，其有唯一解，则 $d(g)$ 可通过以下推导求得

$$d(g) = inf_{r \in X_0} \{\sum_{i=1}^{N} [-\ln(r_i - DL_i) + g r_i] - g B\}$$

$$= \sum_{i=1}^{N} inf_{r \in X_0} [g r_i - \ln(r_i - DL_i)] - g B$$

令 $W(r_i) = g r_i - \ln(r_i - DL_i)$ , $W(\boldsymbol{r}_i)$ 在定义域内一阶、二阶可导，有

$$W'(r_i) = g - \frac{1}{r_i - DL_i}$$

$$W''(r_i) = \frac{1}{(r_i - DL_i)^2}$$

此时若

$$g \leq \frac{1}{DH_i - DL_i} \leq \frac{1}{r_i - DL_i}$$

则 $W(r_i)$ 为单调减函数，则 $r_i = DH_i$ 时有极小值。相反，若 $g \geq \frac{1}{DH_i - DL_i}$ ，可令 $W'(r_i) = 0$ 求出驻点，该驻点值为 $r_i^0 = DL_i + \frac{1}{g}$ 。因为 $W''(r_i^0) > 0$ ，则 $r_i^0$ 为 $W(r_i)$ 的极小值点。综合这 2 种情况得出

$$d(g)=\begin{cases}\sum_{i=1}^{N}[g\,DH_i-\ln(DH_i-DL_i)]-gB, & g\le\dfrac{1}{DH_i-DL_i}\\ \sum_{i=1}^{N}(g\,DL_i+\ln g+1)-gB, & g\ge\dfrac{1}{DH_i-DL_i}\end{cases} \tag{5}$$

则原问题 $P$ 的对偶问题 $D$ 可定义为

$$\max_{\{g\ge 0\}} d(g)$$

因为 $X_0$ 是凸集，$G(r)$是在 $X_0$ 上的凸函数，又显然存在一个 $r\in X_0$ 满足 $DL_i<r_i<DH_i$ 和 $\sum_{i=1}^{N} r_i<B$，也就是说存在 $r$ 在约束条件空间的相对内部。因此，根据 Slater 条件[11]，原问题和对偶问题之间没有对偶间隙，是强对偶关系，则对偶问题 $D$ 存在最优解 $g^*$，也是原问题 $P$ 的解。

根据式（5），对于第一种情况，由于

$$g\le\frac{1}{DH_i-DL_i}\Leftrightarrow DL_i+\frac{1}{g}\ge DH_i$$

根据式（4）可得 $r_i^*=DH_i$；对于第 2 种情况，由于 $d(g)$ 在定义域上有一阶、二阶导数，有

$$d'(g)=\sum_{i=1}^{N}DL_i+\frac{N}{g}-B$$

$$d''(g)=\frac{-N}{g^2}<0$$

则令 $d'(g)=0$ 可求出对偶问题 $D$ 的最优解 $g^*$，将其代入式(4)可求出 $r_i^*$。至此，可获得实现帕累托最优的并行实时迁移带宽分配方案。

### 3.4　算法与实现

基于以上求得的带宽分配方案，多虚拟机并行实时迁移算法如下。

1) 多虚拟机并行实时迁移开始。

2) 统计 $N$ 个待迁移的虚拟机的脏页率，获得 $VM_i$ 的最小、最大脏页率为 $DL_i$、$DH_i$。

3) 分配每个并行迁移进程的发送带宽为

$$r_i=\min\left\{DH_i,DL_i+\frac{B-\sum_{i=1}^{N}DL_i}{N}\right\}$$

4) 每个并行迁移进程以 $r_i$ 为发送带宽进行 Pre-copy。其间若迁移进程 $j$ 结束，则回收 $j$ 的带宽，进行带宽重分配：重新计算进行中的迁移进程 $i$ 的带宽为 $r_i=\min\left\{DH_i,DL_i+\frac{B-\sum_{i=1}^{M}DL_i}{M}\right\}$，其中，$M\in[1,N)$为 $j$ 结束时还未完成的迁移进程个数。

5) 重复步骤 4)直到所有迁移进程完成。

可以看出，一旦某个迁移进程完成，算法就回收其带宽，并剩余运行中的迁移进程进行带宽重分配，达到有效利用带宽的目的。

本文基于虚拟机管理器 Qemu-kvm 0.14.0 和虚拟化平台管理库 Libvirt 0.7.5 实现了以上算法，并对 2 个平台的代码进行扩充，主要增加了控制器和执行器 2 个模块。系统框架如图 1 所示，其中，控制器模块采用单独进程运行，包括如下功能。

1) 在 LibvirtAPI 中添加 *virNodeGet MemoryDirtyRate* 函数，给执行器下发“获取脏页率”指令，并作为回调函数来响应各执行器发来的脏页率数据，以此获取各虚拟机的最小、最大脏页率。

2) 利用 LibvirtAPI 的 *virConnectListDomains* 函数获取迁移进程总数 $N$，并注册一个回调函数 *virNodePrecopyOver* 来响应迁移进程结束事件，在

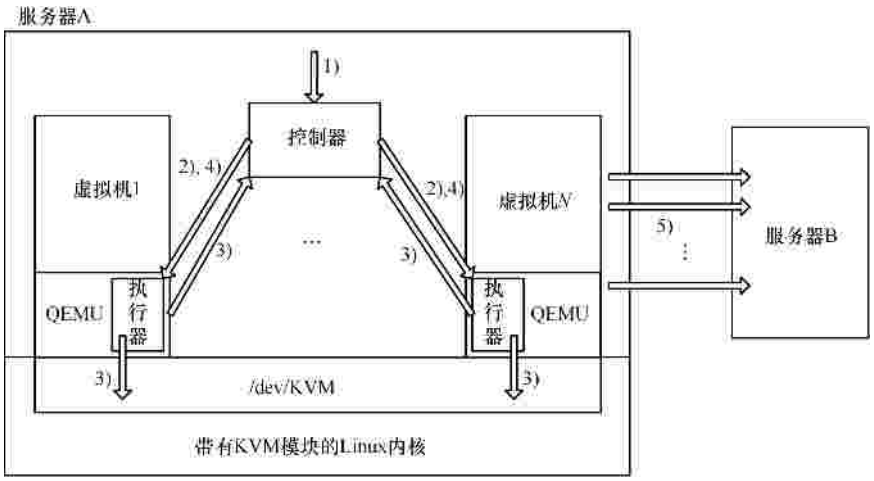


图 1　带有带宽分配机制的并行实时迁移系统框架

其中更新 $N$。

3) 维护每个虚拟机的<最小脏页率，最大脏页率>键值对 *vm_pair*。

4) 根据总带宽、*vm_pair* 和 $N$ 给各迁移进程分配带宽。通过在 LibvirtAPI 中添加 *virNodeSetPrecopyRate* 函数，将带宽值下发给各个迁移进程的执行器。这里总带宽通过 Iperf 工具获得，并在迁移前手工输入到控制器中。

5) 利用 LibvirtAPI 的 *virDomainMigrate* 函数同时给各虚拟机进程下发实时迁移指令。

执行器模块在 Qemu-kvm 的用户空间程序（即 Qemu）中实现，各迁移进程都对应启动一个执行器，主要完成以下功能。

1) 执行控制器的 *virNodeGetMemoryDirtyRate* 调用。利用 Qemu 提供的 *cpu_physical_memory_ set_ dirty_tracking* 函数给 KVM 内核发送 *KVM_MEM_ LOG_DIRTY_PAGES* 调用，打开 KVM 的脏页跟踪机制。接着在一段时间 $P$ 内，以 $T$ 为周期调用 *cpu_ physical_sync_dirty_bitmap* 函数，将内核空间跟踪的脏页映射到用户空间的 *ram_ listphys_dirty* 数组中，从中统计脏页数，计算出脏页率。最后将 $P$ 时间内的最小、最大脏页率以异步消息传给控制器。

2) 当迁移进程结束时，给控制器发送“迁移结束”的异步消息。

3) 执行控制器的 *virNodeSetPrecopyRate* 调用，获得控制器分配的传输带宽，将其赋值给 Pre-copy 模块中的传输带宽限制变量 *max_throttle*。

整个系统的运行流程如下（对应图 1 中的序号）。

1) 启动控制器，输入总带宽值，并行迁移开始。

2) 控制器获取所有虚拟机个数，并向所有虚拟机进程下发“获取脏页率”指令。

3) 各虚拟机进程中的执行器收到指令后进行脏页率统计，将最小、最大脏页率传给控制器。

4) 控制器执行带宽分配方案，给各个迁移进程分配传输带宽。

5) 并行实时迁移正式启动。

## 4 实验与评估

本文以并行迁移总时间为性能度量标准，通过对比实验对所提出的带宽分配机制进行验证评估。为了保证性能评估的准确性和全面性，实验分为定制场景评估和真实场景评估。前者采用可控实验环境和定制运行负载，侧重性能数据的准确跟踪；后者基于实际的云平台和真实运行负载，侧重真实场景下的性能评估。

### 4.1 定制场景评估

在该实验场景下，本文对有、无带宽分配机制的多虚拟机并行迁移的性能进行跟踪和对比（以下称为“标准迁移”和“优化迁移”）。另外，由于目前在一些研究中采用平均分配带宽这种简单的策略[5]，本文在实验中也加入了与该带宽分配方案（以下称“均分迁移”）的对比。

#### 4.1.1 实验配置及方法

实验采用 3 台配置相同物理服务器（Intel Dual Core 2.93 GHz 的 CPU，内存 4 GB，硬盘 500 GB，操作系统为 Ubuntu Server 11.10），其中，2 台服务器安装标准的 Qemu-kvm 0.14.0 或有带宽分配机制的优化版本作为虚拟化平台，另 1 台服务器作为 NFS 共享存储。3 台服务器通过 100 Mbit/s 以太网互联。一台虚拟化服务器同时运行 4 台虚拟机（配置均为 1 个 1 GHz 的 vCPU、512 MB 虚拟内存、10 GB 虚拟机磁盘及 Ubuntu Server 11.10 的操作系统）。另一台虚拟化服务器作为实时迁移的目的端，空载运行。

本文采用开源的系统测试套件 LMbench 中的内存测试程序 bw_mem 作为虚拟机的运行负载，即对一个 20 MB 的内存区域反复进行写操作。同时，为了能够控制各个虚拟机的脏页率，修改 bw_mem.c 源文件中的 *wr* 函数，利用 *usleep* 库函数对内存写速率进行调整，实现虚拟机脏页率在一定范围内变化。经过反复调试，4 台虚拟机的脏页率变化范围如表 1 所示，其中，最大值、最小值是在 20 s 内以 2 s 为间隔对虚拟机的脏页率进行统计所得，平均值是对这些统计值的算术平均值。

表 1 实验中各虚拟机负载的脏页率情况

| 虚拟机 | 脏页率/(Mbit·s$^{-1}$) | | |
|---|---|---|---|
| | 最小值 | 最大值 | 平均值 |
| $VM_1$ | 10 | 60 | 48 |
| $VM_2$ | 5 | 23 | 17 |
| $VM_3$ | 4 | 25 | 15 |
| $VM_4$ | 1 | 17 | 6 |

实验中，待各个虚拟机稳定运行后，将 4 台虚拟机并发迁移至目的端服务器。在此过程中，通过 Linux 下的流量监控工具 Nethogs 观察每个进程的带宽利用情况。另外，利用 Linux 下的流量控制工具 TC 调整 2 台服务器间的总带宽为 20、

30,…,70, 80 Mbit/s。以下实验数据是 10 次结果的算术平均值。

#### 4.1.2　实验结果及分析

图 2 和图 3 分别显示了在不同的总带宽配置下，标准迁移及优化迁移中 4 个迁移进程的实际带宽利用情况。从表 1 可知，总带宽都小于各虚拟机脏页率之和，因此各总带宽条件下都存在带宽竞争。图 2 显示标准迁移中各迁移进程竞争占用带宽，其中，$VM_4$ 因脏页率高而占用了大量带宽，在低带宽下更严重，使并行迁移被迫变成串行迁移。随着总带宽升高，其他 3 个迁移进程的可用带宽略有增加，但依然无法满足脏页传输需求。图 3 中，优化迁移中各迁移进程按脏页传输需求均衡使用带宽，在低带宽即可以进行并行迁移。随着总带宽增加，$VM_2$、$VM_3$ 和 $VM_4$ 能够快速获得所需传输带宽。由于均分迁移采用均享带宽方案，对应各总带宽下的各虚拟机占用的带宽分别为 5 Mbit/s、7.5 Mbit/s、10 Mbit/s、12.5 Mbit/s、15 Mbit/s、17.5 Mbit/s、20 Mbit/s。该方案虽然能各迁移进程公平享有带宽，在低带宽下也能进行并行迁移，但缺乏考虑各虚拟机迁移负载的实际需求，使高负载进程没有分配的带宽过低而低负载进程带宽过剩，特别在高带宽下这种问题更明显。

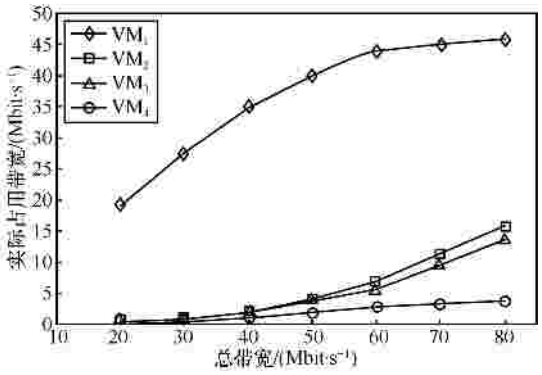


图 2　标准迁移中各迁移进程带宽利用情况

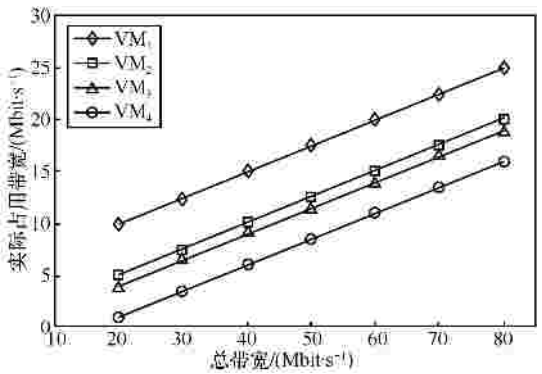


图 3　优化迁移中各迁移进程带宽利用情况

本文抽取 40 Mbit/s 和 70 Mbit/s 总带宽配置下的实验结果分别作为低、高带宽的典型，对比 3 种带宽分配方案下各迁移进程的完成情况。如图 4 所示，40 Mbit/s 总带宽下标准迁移的完成次序为 $VM_1$-$VM_3$-$VM_2$-$VM_4$，优化迁移和均分迁移都为 $VM_4$-$VM_3$-$VM_2$-$VM_1$。根据三者的带宽利用情况可知，标准迁移中各迁移进程竞争带宽，在低带宽下由重负载的 $VM_1$ 抢占带宽，完成迁移后再由负载相当的 $VM_2$、$VM_3$ 抢占，待两者完成后再由 $VM_4$ 进行迁移；优化迁移严格限制了各迁移进程的传输带宽，在保证并行迁移的同时优先完成低负载进程；均分迁移和优化迁移情况相似，只是低负载进程完成得稍快，而高负载进程完成得稍慢。图 5 显示在 70 Mbit/s 的总带宽下，标准迁移中按 $VM_1$-$VM_4$-$VM_3$-$VM_2$ 次序完成。此时 4 个迁移进程虽然可并行执行，但由于带宽竞争，$VM_2$、$VM_3$ 和 $VM_4$ 的实际带宽依然无法满足其脏页传输需求；优化迁移中因 3 个低负载进程获得满足脏页传输需求的实际带宽，依然保持低负载进程优先完成；均分迁移此时虽然也是保持低负载进程优先，但是高负载进程所需带宽均摊给了低负载进程，而低负载进程所占带宽过剩，使高负载进程迁移时间相对要长。

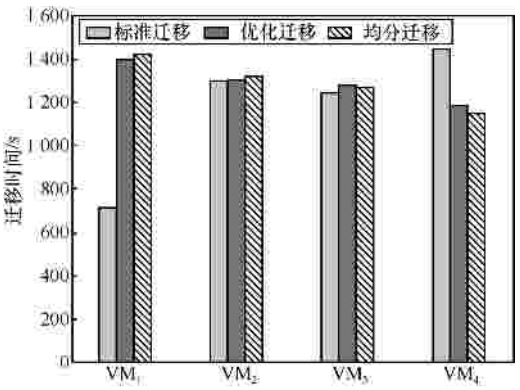


图 4　40 Mbit/s 总带宽下各迁移进程完成情况对比

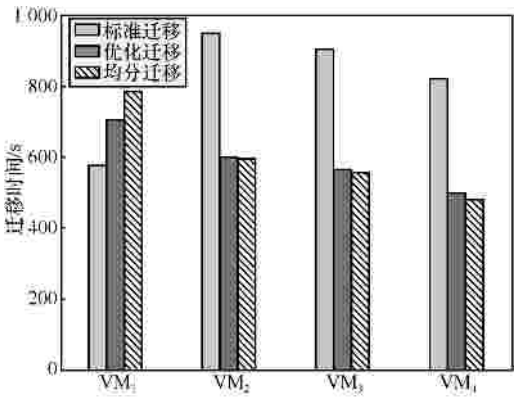


图 5　70 Mbit/s 总带宽下各迁移进程完成情况对比

图6对比了标准迁移、均分迁移和优化迁移在不同总带宽下的整体性能，即总迁移时间。可以看到，低带宽下三者性能相近，甚至在20 Mbit/s的极低带宽下，优化迁移和均分迁移的性能还略低。结合图2~图4，本文分析在低带宽下优化迁移中各迁移进程虽能公平获得带宽，却仅能分配到最小需求带宽，各迁移进程是在并发低速运行。同样，此时均分迁移中平均带宽过低，甚至还无法满足所有迁移进程的最小需求带宽，如$VM_1$，导致并行迁移时间长。而此时标准迁移中各进程实际是串行执行，虽有进程间等待时间但单个进程运行较快。随着总带宽升高，超过40 Mbit/s后，优化迁移和均分迁移性能优势明显。结合图2~图5可知，在高带宽下，标准迁移下各迁移进程虽可并行执行，但大多数进程还是无法获得所需带宽，无法保证整体迁移效率；优化迁移中大多数进程都能获得所需带宽，少数进程也能在满足最小需求带宽之上最大利用带宽，因此整体迁移效率较高；而均分迁移此时虽然大多数进程也都能获得所需带宽，但因高负载进程所得带宽较低，而低负载进程带宽过剩，导致其性能不如优化迁移。总之，优化迁移在70 Mbit/s时较标准迁移性能提高约25%，较均分迁移性能提高约10%；在80 Mbit/s时较标准迁移提高约40%，较均分迁移提高约20%。

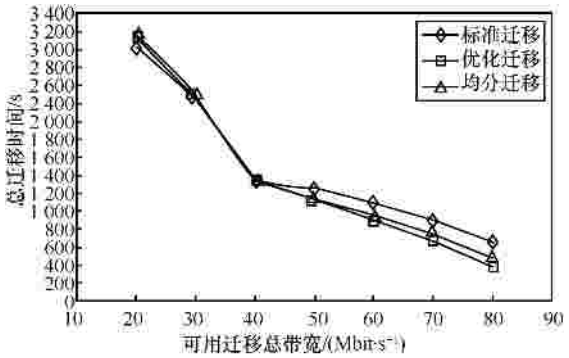


图6 标准迁移、均分迁移和优化迁移整体性能对比

### 4.2 真实场景评估

本文采用Openstack云平台进一步评估所提方案的实际性能。首先，本文将之前实现的带宽分配机制移植到Openstack上；然后在Openstack上开启多个虚拟机，运行真实的应用程序；最后，分别在标准的Openstack和优化的Openstack上进行虚拟机并行迁移，对比两者的迁移性能。

#### 4.2.1 实验准备

由于Openstack也是采用Libvirt API底层的Qemu-kvm实现实时迁移，因此本文可以很方便地将以上对所提带宽分配机制的实现移植到Openstack上。本文在Openstack的Nova-API中增加了一项“multi-migrateLive”，对应在Nova-API实现文件admin_actions.py中增加了一个wsgi接口“multi-migrateLive”，里面主要包括一个*_multi_migrate_live*函数来启动多虚拟机并行迁移。在该函数中，本文基于Python调用C的方式直接调用之前实现的控制器程序，并配合底层增加了执行器模块的Qemu-kvm实现本文所提的带宽分配机制。只是在启动各迁移进程时调用Nova-API中现有的*_migrate_live*函数。

#### 4.2.2 实验配置和方法

实验采用4台物理主机，其中，2台安装Openstack的Nova、Neutron及Horizon等组件，作为计算节点，配置除了内存变为8 GB外，其余和4.1.1节中一样。剩余2台中，1台安装Swift、Glance及Cinder等组件作为存储节点，另1台作为客户端使用，配置与4.1.1节中一样。同样，在Openstack中启动4台虚拟机，其配置除了内存变为1 GB外，其他与4.1.1节中一样。网络为吉比特以太网，操作系统都是Ubuntu Server 11.10。

本文采用2种真实的应用程序作为虚拟机的运行负载：一种是编译Linux内核，另一种是动态Web应用程序，即采用PHP和Mysql部署开源的微博应用程序WordPress。在客户端本文对WordPress人工录入微博，并采用Web服务测试程序Siege，设置多个连接进程对Wordpress进行并发访问。对于虚拟机$VM_1$，运行WordPress服务，客户端发起15个访问进程；对于$VM_2$、$VM_3$，运行Linux内核编译；对于$VM_4$，运行WordPress服务，客户端发起5个访问进程。

实验中，在客户端上访问Openstack控制台，将4个虚拟机在运行负载时进行并发迁移，对比Openstack在默认带宽分配方案（简称“Openstack标准迁移”）和采用本文所提方案(简称“Openstack优化迁移”)下的总迁移时间。实验结果为10次实验数据的算术平均值。

#### 4.2.3 实验结果及分析

通过分析源代码，本文发现Openstack默认没有考虑并行迁移时的带宽分配问题，即对应4.1节

中的标准迁移方式。图 7 对比了 Openstack 标准迁移和 Openstack 优化迁移下各虚拟机迁移进程的完成时间。Openstack 标准迁移的完成次序为 $VM_4$-$VM_1$-$VM_2$-$VM_3$，即在并行迁移状态下，带宽需求量小的轻负载虚拟机 $VM_4$ 和抢占大部分带宽的重负载虚拟机 $VM_1$ 优先完成迁移；Openstack 优化迁移仍然是保证低负载进程优先完成。图 7 也反映出了在总迁移时间上，Openstack 优化迁移要比 Openstack 标准迁移性能提高了近 22%。和前面实验分析原因相似，Openstack 优化迁移中因采用了带宽分配机制，解决了各迁移进程竞争带宽资源导致带宽无法有效分配的问题。

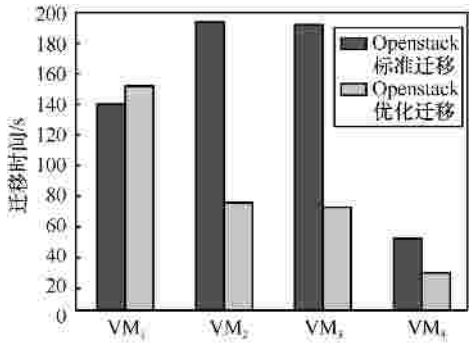


图 7　真实场景下各迁移进程完成情况对比

## 5　相关工作

针对单虚拟机实时迁移的带宽分配问题，文献[3]针对实时迁移影响网络应用服务，提出在 Pre-copy 算法中设定了一个“动态限速”机制：首轮拷贝采用一个最小带宽值，之后拷贝轮的带宽为上轮的脏页率加上一个根据经验确定出的固定常数。文献[12]基于排队论建立了以网络带宽来量化网络应用服务质量的代价函数模型，通过最小化该代价函数来求得在不影响网络服务情况下的实时迁移带宽。文献[13]提出了一种面向业务特征的自适应虚拟机实时迁移带宽分配算法，在 Pre-copy 中通过分析历史迭代轮中脏页率，进而自适应地调整后轮迭代拷贝的脏页发送速率。和以上工作不同，本文研究多虚拟机的实时迁移问题。

针对多虚拟机实时迁移方面的研究，文献[14]在多虚拟机并行实时迁移中采用重复数据删除技术，识别出并行迁移的多个虚拟机间相同或相似的内存页，在迁移时只对这些冗余页面传输一次，降低了迁移时间和网络负载。文献[5]在假设的前提下，分别给出了串、并行多虚拟机实时迁移的迁移时间及停机时间的量化模型，并根据模型对 2 种迁移策略进行对比。文献[15]研究运行多层应用程序的相关多虚拟机的并行迁移问题，提出一个多虚拟机实时迁移调度系统 VMbuddies，其中，包括一个带宽分配方案来给各迁移进程分配最优迁移带宽。方案以总迁移时间为代价函数，将带宽分配建模为一个分布约束优化问题进行求解。以上研究不涉及带宽分配问题，或没有从带宽资源竞争的角度考虑带宽分配方案。

在运用合作博弈论解决网络领域中的带宽分配问题的方面，文献[7]针对高速宽带网络支持弹性应用服务数据传输速率的问题，基于纳什议价模型，提出了一个实现全局最优和公平的分布式的最优带宽分配算法。文献[16]针对 IaaS 云计算数据中心中保证多租户虚拟机的网络性能，将虚拟机间的带宽共享问题建模为一个纳什议价，进而设计出一个分布式带宽分配算法，保证虚拟机获得最小带宽的基础上能够公平使用带宽。本文侧重于虚拟机实时迁移方面的研究，与这些工作的研究领域不同。

## 6　结束语

本文针对多虚拟机并行实时迁移中存在的带宽竞争问题，提出一种基于合作博弈的带宽分配机制，将多个并行迁移进程竞争迁移带宽的问题建模为一个纳什议价，通过求解纳什议价解获得具备兼顾全局最优及公平的带宽分配方案，并在实际的虚拟化平台上对带宽分配机制进行了实现。对比实验的结果表明，所提出的带宽分配机制能够在多个迁移进程间公平有效地分配迁移带宽，提高并行实时迁移的性能。下一步工作将考虑带有虚拟机磁盘迁移的多虚拟机全系统实时迁移的带宽分配机制。

参考文献：

[1] 董健康，王洪波，李阳阳，等. IaaS 环境下改进能源效率和网络性能的虚拟机放置方法[J]. 通信学报, 2014, 35(1): 72-81.
DONG J K, WANG H B, LI Y Y, et al. Improving energy efficiency and network performance in IaaS cloud with virtual machine placement[J]. Journal on Communications, 2014, 35(1): 72-81.

[2] KVM. Kernel based virtual machine [EB/OL]. http://www.linux-kvm.org/page/Main_Page.

[3] CLARK C, FRASER K, HAND S, et al. Live migration of vi ual machines[C]//The Second Symposiumon Networked Systems Design

and Implementation. c2005: 273-286.
[4] YE K, JIANG X, MA R, et al. VC-Migration: live migration of virtual clusters in the cloud[C]//The ACM/IEEE 13th International Conference on Grid Computing. c2012: 209-218.
[5] CALLEGATI F, CERRONI W. Live migration of virtualized networks: Analytical modeling and performance evaluation[C]//IEEE Workshop on Software Defined Networks for Future Networks and Services(SDN4FNS 2013). Trento, c2013: 1-6.
[6] LIU H, JIN H, XU C Z, et al. Performance and energy modeling for live migration of virtual machines [J]. Cluster comput ng, 2013, 16(2): 249-264.
[7] YAÏCHE H, MAZUMDAR R R, ROSENBERG C. A game theoretic framework for bandwidth allocation and pricing in broadband networks[J]. IEEE/ACM Transactions on Networking (TON), 2000, 8(5): 667-678.
[8] FUDENBERG D，TIROLE J．Game theory[M]. MIT Press,1991.
[9] MUTHOO A. Bargaining theory with applications[M]. Cambridge University Press, 1999.
[10] STEPHEN B, LIEVEN V. Convex optimization[M].Cambridge University Press,2004.
[11] BERTSEKAS D.Nonlinear programming[M].Athena Scientific, 1995.
[12] BREITGAND D, KUTIEL G, RAZ D. Cost-aware live migration of services in the cloud[C]//The 3rd Annual Haifa Experimental Systems Conference(SYSTOR 2010). c2010.
[13] **刘诗海，孙宇清，刘古月．面向业务特征的自适应虚拟机迁移带宽分配算法**[J]. **计算机学报**, 2013, 36(9): 1816-1825.
LIU S H, SUN Y Q, LIU G Y. An adaptive bandwidth allocatio lgorithm for virtual machine migration based on service features[J]. Chinese Journal of Computers, 2013, 36(9): 1816-1825.
[14] DESHPANDE U, WANG X, GOPALAN K. Live gang migration of virtual machines[C]//The 20th ACM International Sympos um on High Performance Distributed Computing. c2011: 135-146.
[15] LIU H, HE B. VMbuddies: coordinating live migration of multi-tier applications in cloud environments[J]. IEEE Transactio n Parallel and Distributed Systems, 2015, 26(4):1192-1205.
[16] GUO J, LIU F, ZENG D, et al. A cooperative game based allocation for sharing data center networks[C]//IEEE Internationa ference on Computer Communications (INFOCOM). c2013: 2139-2147.

**作者简介：**

**崔勇（1983-），男，河南郑州人，郑州大学博士生，主要研究方向为云计算、虚拟化技术。**

**林予松（1973-），男，河南郑州人，博士，郑州大学副教授，主要研究方向为下一代互联网、互联网医疗。**

**李润知（1978-），女，河南洛阳人，博士，郑州大学讲师，主要研究方向为** P2P **流媒体、社交网络。**

**王宗敏（1964-），男，河南荥阳人，博士，郑州大学教授，主要研究方向为下一代互联网。**