基于协同进化蚁群算法的多播 QoS 路由算法

李昌兵[1,2],胡 华[2],吴 建[1],曹长修[2]
LI Chang-bing[1,2],HU Hua[2],WU Jian[1],CAO Chang-xiu[2]

1.重庆邮电大学 电子商务与现代物流重点实验室,重庆 400065
2.重庆大学 自动化学院,重庆 400044
1.Electronic Commerce and Modern Logistics Key Laboratory,Chongqing University of Post and Telecommunication,Chongqing 400065,China
2.Automation Academy,Chongqing University,Chongqing 400044,China
E-mail:lcbss@126.com

LI Chang-bing,HU Hua,WU Jian,et al.Multicast QoS routing algorithm based on cooperative evolution ant colony optimization.Computer Engineering and Applications,2008,44(24):139-142.

Abstract: A new QoS(Quality of Service) multicast routing problem based on cooperative evolution ant colony algorithm is presented.The route selection strategy and the control parameters of the ant colony are adjusted adaptively in different iteration steps.It makes comprehensive consideration of all the constraints of QoS routing problem,and takes into account the individual characteristics of each constraint.Simulation results show that this algorithm has higher convergence speed,it can meet the service quality requirement in real network.

Key words: multicast;routing algorithm;cooperative evolution;ant colony algorithm

摘 要:提出一种基于协同进化蚁群算法的求解 QoS(Quality of Service)多播路由问题的新算法。算法中控制参数及路由选择策略根据迭代过程所处的不同阶段自适应调整。综合考虑 QoS 路由中所有约束条件的同时,也充分考虑各个约束自身的独立特性。仿真结果证明了算法收敛速度快,能满足实际网络服务质量的要求。

关键词:多播;路由算法;协同进化;蚁群算法
DOI:10.3778/j.issn.1002-8331.2008.24.041 **文章编号:**1002-8331(2008)24-0139-04 **文献标识码:**A **中图分类号:**TP393.07

1 引言

随着 Internet 规模的不断扩大，网上的实时业务量也在不断增长,由于实时业务对网络传输的延时、延时抖动、带宽、包丢失率、花费等特性较为敏感,当突发性高的多媒体业务数据在网络上传输时,实时业务会受到影响。因此应在网络上导入 QoS 技术,以确保实时业务的通信质量。

QoS 多播路由问题是指在分布的网络中寻找最优路径,要求从源节点出发,历经所有的目的节点,找到一条满足所有的约束条件且花费最小的网络路径。QoS 多播路由问题是 NP 完全问题。近年来,各国学者开始在这方面探索,提出了一些快速有效的算法。大多数启发式算法本质上是集中式的算法(源路由算法)。集中式路由算法有许多问题，如路由信息的更新开销、计算瓶颈与扩展性等问题。而在分布式路由算法中,路径的计算是通过分布式计算完成的,路由计算的控制消息在节点间互相交换,路由计算需要的状态信息则保存在每个节点上。路由响应时间很短,扩展性更强。但分布式算法存在算法复杂性及回路问题。

无论是对于源路由还是分布式路由，都面对着同一个问题,就是如何使网络资源利用优化,避免拥塞和资源使用过度。对于网络营运者来说,不仅要保证用户的 QoS 请求,而且还要维护网络的有效运行,避免多个请求对某些链路的挤占所造成的拥塞和带宽资源过度使用。由于 QoS 多播路由问题的复杂性,近来也有研究人员利用蚁群算法解决多播 QoS 路由计算这类多目标优化问题。但是单纯采用蚁群算法的解决方案存在计算时间以及收敛速度的问题。

本文提出基于协同进化蚁群算法的多播 QoS 路由算法,试图克服采用传统蚁群算法的路由方案的局限性。同时由于该算法具有较高的并行性，一次运行可以求得多个 Pareto 最优解,具有单目标优化方法不可比拟的优势。

2 问题描述

就组播路由的选择而言,通常用带权图 $G=(V,E)$ 表示网络,其中 V 为网络节点集,E 为任意两相邻节点 i,j 间通信链路 $(i,j)\in E$ 的集合。$s\in V$ 表示源节点,$M\subseteq V-\{s\}$ 表示目的节点集

基金项目:重庆市教委科研基金(the Science Research Foundation of Chongqing Munipal Education Commission under Grant No.KJ070509)。
作者简介:李昌兵(1970-),男,博士生,讲师,主要研究领域为计算智能,控制优化;胡华(1970-),男,博士生,高级工程师,主要研究领域为网络控制优化;吴建(1969-),男,工程师,主要研究领域为数据处理;曹长修(1937-),男,教授,博导,主要研究领域为智能控制。
收稿日期:2007-11-16 **修回日期:**2008-01-07

合,也称组播组。M 中的每个节点 $m \in M$ 称为组成员,$|M|$表示目的节点集的大小。组播树 T 是 G 的子图,它是根为 s 并包含目的节点集合 M 的一棵树,用 $T=(VT,ET)$表示,其中 $VT \in V$,$ET \in E$,T 中存在由源节点 s 到每个目的节点 m 的一条路径,用 $p(s,m)$表示。假设对于任意一条链路 $e \in E$,可以定义关联代价($cost$)、时延($delay$)、带宽($width$)及丢包率($packet\_loss$)4种 QoS 度量函数,则组播树 $T=(VT,ET)$存在下列约束关系:

$$delay(p(s,m))=\sum_{e\in p(s,m)} delay(e),\ \forall m\in M \tag{1}$$

$$cost(p(s,m))=\sum_{e\in p(s,m)} cost(e),\ \forall m\in M \tag{2}$$

$$width(p(s,m))=\min\{width(e),e\in p(s,m)\},\ \forall m\in M \tag{3}$$

$$packet\_loss(p(s,m))=1-\prod_{e\in p(s,m)}(1-packet\_loss(e)),\ \forall m\in M \tag{4}$$

以代价 $cost$ 最小化为优化目标,且 $cost$ 为所有链路代价和:

$$cost(T)=\sum_{e\in E_T} cost(e) \tag{5}$$

时延(定义为源节点 s 到各目的节点路径时延的最大值)约束为:

$$delay=\max_{n\in N}\left\{\sum_{e\in p(s,m)} delay(e)\right\}=\max_{\forall m\in M}\{delay(p(s,m))\} \tag{6}$$

时延抖动(定义为任意两条到达组播目的节点的路径的时延之差)约束为:

$$delay\_jitter(T)=\left|\sum_{e\in p(s,u)} delay(e)-\sum_{e\in p(s,v)} delay(e)\right| \tag{7}$$

其中,$u,v \in M$。QoS 组播路由就是以网络拓扑结构和网络状态为基础,构建一棵组播树 T,包含源节点和所有目的节点,并且在满足给定的约束条件下目标函数最优[1]。

3 基于协同进化蚁群算法的多播路由策略

多播路由优化的目标是找到一种算法或策略,在给定的网络和多播需求的情况下,寻求一种链路连接方式,使网络资源能够得到有效利用。近年来,各国学者开始在这方面进行探索,提出了一些快速有效的多播路由算法[2-4],都是以服务质量指标中的带宽、时延、时延约束和包丢失率等为优化选路准则。由于多播 QoS 路由计算问题已被证明是 NP 完全问题,近来有许多研究者致力于将蚁群算法用于解决路由中的优化问题,相关文献提出了不同的解决约束的算法,但都没有用蚁群算法来解决多播中的多 QoS 约束路由的优化问题。

蚁群算法的生物学背景决定了它适合于求解离散空间中的组合优化问题,因为 QoS 多播路由问题与 TSP 问题相似。所以可以用求解 TSP 问题的蚁群算法来求解 QoS 多播路由问题。但在求解多目标优化问题时存在一些困难。多目标优化问题由于解的多样性,不仅要求所得的解能够收敛到 Pareto 前沿,而且要有效地保持群体的多样性。而在基本蚁群算法中,信息素浓度越高的地方,蚂蚁选择作为寻优路径的概率越大。在求解多目标优化问题时,蚂蚁之间的这种信息素交流方式,会使所求得的解集中在解空间的某一区域内,不利于群体多样性的保持。为了解决算法控制参数难以确定和早熟停滞两个问题,相关文献以蚁群算法(ACS)为基础提出了进化蚁群算法(EACS)。进化蚁群算法的具体描述参见文献[5]。

协同进化(Co-evolution)是紧密相关的种群之间的补充进化。最早的协同进化思想是由 Ehrlich 和 Raven 在讨论植物和植物性昆虫相互作用对各自进化的影响时提出的。根据种群之间相互作用的协同进化思想,Hillis 将一种寄生虫和寄主共生的机制应用于优化搜索中,提出了协同进化遗传算法(Co-evolutionary Genetic Algorithm,CGA)理论,并用于多目标函数优化问题。针对多目标离散优化问题,以 EACS 算法为基础,开发出多目标协同进化蚁群优化算法 CACSM。CACSM 中的多个蚁群有着共同的搜索空间,每个群体搜索活动、信息激素更新策略、群体内部的进化机制都与 EACS 算法类似。CACSM 的关键在于进化 NM 代后,群体之间发生信息交换,信息交换的策略、内容和频率,这些都直接关系到算法的效率和解的质量。信息交换完成后,根据求得的解的质量进行信息激素的奖励与惩罚。

现有的 QoS 多播路由算法大都只考虑了一个或两个限制,并且有的需要在每个节点保留整个网络状态信息。本文提出一种基于协同进化蚁群算法的全局分布式多播 QoS 路由计算方法。并且考虑了多个 QoS 约束限制,结合多播路由选择的具体要求构造新颖的全局优化蚂蚁算法来解决多约束问题。首先为各个种群分配一个目标,在这里是搜索到特定目的节点的满足 QoS 约束的最优路径。然后采用本文设计的方法将各种群的最优解合并为满足 QoS 约束的多播树路径,并用 Pareto 最优性的相关进行评价,据此进行信息激素的奖励或惩罚。该算法是完全的分布式算法。每个节点均依据自身保存的局部路由信息参与算法的相应操作。算法设计的关键是,在最终的多播树中要有效避免任何回路的产生。而这恰恰是分布式多播路由算法设计的难点问题。

4 算法原理及实现步骤

4.1 算法原理

基于蚁群算法的原理,可以创建基于人工蚂蚁群体的网络模型。为了把通信网络和蚁群算法的理论结合起来,可以用"概率表"来取代网络节点的路由选择表,又称信息素表,如表 1 所示。表中的信息素值以概率值表示(n 为目的节点数,m 为邻接节点数目)。"信息素表"给出了以某个节点为目的节点时选择下一节点的概率,同时表中的概率值按照某种规则周期性地刷新,公式(8)给出了在节点 u 中的蚂蚁 k 选择到相邻节点 v 的概率。

表 1 信息素表

目的节点	邻接节点			
	A_1	A_2	…	A_m
D_1	P_{11}	P_{12}	…	P_{1m}
D_2	P_{21}	P_{22}	…	P_{2m}
…	…	…	…	…
D_n	P_{n1}	P_{n2}	…	P_{nm}

$$p_k(u,v)=\frac{[\tau(u,v)]\cdot[\theta(u,v)]^{\mu}\cdot[B(u,v)]^{\delta}}{\sum_{n\in J(u)}[\tau(u,n)]\cdot[\theta(u,n)]^{\mu}\cdot[B(u,n)]^{\delta}} \tag{8}$$

其中 τ 为信息素,$J(u)$代表节点 u 的邻接点集合,$\theta=1/C(u,v)$,$C(u,v)$可以是代价或者其它网络资源利用测度值。$B(u,v)$代表链路(u,v)的可用带宽。μ 和 δ 分别是表示信息素总量与代价和可用带宽相对重要程度的关系。式(8)表示我们倾向于选择代价较小和可用带宽较大的链路。此外还定义信息素的局部和全局更新策略如下:

(1)信息素的局部更新策略[6]

对第 i 只蚂蚁,如果节点 u、v 是它选择路径上的两个相邻节点,信息素 $\tau(u,v)$ 用公式:$\tau(u,v)\leftarrow(1-\alpha)\tau(u,v)+\Delta\tau(u,v)$ 来调节,否则不调节,其中 $0<\alpha<1$,α 表示信息素挥发强度参数,$\Delta\tau=\sum_{i=1}^{m}\Delta\tau_k(u,v)$,$\Delta\tau_i=Q/L_i$,$L_i$ 是第 i 只蚂蚁在链路 (u,v) 的时延,Q 是常数。采用此方法蚂蚁可以更好地利用链路的时延动态地调整链路上的信息素强度,当时延大于某个值时(即大于 $\Delta\tau$ 时),链路上的信息素强度完全处于"挥发状态",这时随着链路 (u,v) 时延的增大,链路的信息素强度随之减少,此外若没有局部更新,所有蚂蚁将在前一次的最好路径的有限相邻区域内搜寻。

(2)信息素的全局更新策略

当一只蚂蚁从源节点 s 到达目的节点 d,所经过的所有的链路上的信息素 $\tau(r,s)$ 按公式 $\tau(r,s)\leftarrow(1-\alpha_1)\cdot\tau(r,s)+\alpha_1\Delta\delta$ 来调节,否则按公式 $\tau(r,s)\leftarrow(1-\alpha_1)*\tau(r,s)$ 来调节,其中 $0<\alpha_1<1$,此规则的目的是搜寻全局最优解,因此 σ 的定义如公式(9)所示。其中,LC_{ij},节点 i 到 j 的链路的费用;$P_{ij}^{d}=1$,如果节点 i 到 j 的边是第 d 个蚂蚁所选的路由的边,否则 $P_{ij}^{d}=0$;LB_{ij},节点 i 到 j 的链路的带宽;LD_{ij},节点 i 到 j 的链路的时延;如果节点 r 是第 d 个蚂蚁所选路由的节点 $N_r^d=1$,否则 $N_r^d=0$;ND_r,节点 r 的处理时延;NL_r,节点 r 的丢包率。

$$\delta=-\delta_1+\delta_2$$

$$\delta_1=\sum_{i=1}^{v}\sum_{j=1,j\neq i}^{v}LC_{ij}*P_{ij}^{d}$$

$$\delta_2=A*\sum_{i=1}^{v}\sum_{j=1,j\neq i}^{v}H(Z_{ij})+B*H(Z_2)+C*H(Z_3)\qquad(9)$$

如果 Z,$H(Z)=0$,否则,$H(Z)=Z$。

$$Z_{ij}=P_{ij}^{d}*LB_{ij}-B$$

$$Z_2=D-(\sum\sum LD_{ij}*P_{ij}^{d}+\sum N_r^d*ND_r)$$

$$Z_3=\prod_{i=1}^{v}(1-N_r^d*NL_r)-(1-L_n)$$

4.2　算法实现步骤

基于协同进化蚁群算法的多播 QoS 路由算法实施步骤:

(1)初始化。为每个群体分配一个目的节点,每个群体拥有的蚂蚁数相同。在源节点处,以时间间隔 Δ 交替发送各群体的蚂蚁分组。

(2)每个群体按照 EACS 的法则搜索源节点到目的节点满足 QoS 约束的路由。

其中,单个蚂蚁分组搜索路径的步骤如下:

每个蚂蚁分组使用五个字段 ant.delay、ant.hop、ant.node、ant.cost、ant.bw 分别表示蚂蚁分组的从源节点开始的时延、经历的跳数、所访问的节点、代价、带宽。

在源节点均以一定的时间间隔向选定的目标节点发送蚂蚁分组。

以节点 d_1 作为目的节点,初始化蚂蚁分组各字段值,然后从源节点发送蚂蚁分组。设每个蚂蚁群体蚂蚁分组的数量为 l。人工蚂蚁根据信息素最大的原则在邻接节点中选择下一个节点(如满足条件的节点有多个,则按照随机方式选择),每经过一个节点,均要更新蚂蚁分组上述相应字段的值。

当蚂蚁到达节点 U 时,如果 U 为所选定的目的节点,而且满足 QoS 约束的要求则按原路径返回一个蚂蚁分组(该返回蚂蚁分组按高优先级分组处理),同时根据公式更新原路径上各条链路的信息素值。否则要丢弃该蚂蚁分组。与 U 点的信息素表做比较,如果该节点不是所选定的目的节点,而且 U 不是该蚂蚁已经访问过的节点,则根据信息素最大的原则移动到下一个邻接节点 V。如果 U 是已经访问过的节点,为避免出现回路,执行丢弃蚂蚁操作,同时该链路上的信息素的值重设为 0(信息素挥发)。当蚂蚁到达下一个节点 V 时,如果满足时延要求,则该链路上的信息素值根据公式进行更新,蚂蚁分组的各字段值的更新原则如下:如果不满足时延要求则丢弃该蚂蚁分组(蚂蚁死亡)。在节点 V 又根据该点的信息素表重复上述过程。

(3)每个群体所搜索的解空间中信息激素的局部更新和全局更新。

(4)每个群体内部按照 EACS 中的法则进化。

(5)若步骤(2)~步骤(4)已经循环 NM 次,进行信息交换、多播树的生成和信息激素的奖励或惩罚。将上述各个群体所获得的路径进行合并,即可得到所求的多播树。在合并过程中,为了避免回路的产生,需要采取以下措施,如果各个子路径相互之间均无交叉,则进行简单合并操作就可形成满足要求的多播树,如果任意一对路径在合并过程中有交叉节点存在,而且交叉节点的顺序与两条路径的顺序相同,则无回路,否则就有回路,解决方法是,先任意选两条有交叉点的路径,在第二条路径中,删除从第一个乱序的交叉点的前一个交叉点开始,到最后一个乱序的交叉点之间的路径,将其剩余的路径在最后一个乱序交叉点处加入第一条路径,形成一棵子树(或者直接将最后一个交叉点之后的路径加入第一条路径即可)。如果其余路径与该子树有交叉点,则逐次将这些路径加入该子树,路径加入子树的具体方法是,将待加入路径最后一个交叉点之后的路径加入该子树。采用上述方法就可将各蚂蚁群体搜索的较优路径合并为一棵多播树。当然采用这种方法形成的多播树不是唯一的。由这些多个非支配解就可以形成 Pareto 最优解集及其相应的节点序列。

5　算法仿真及性能评价

5.1　仿真模型

利用 ns2 仿真平台验证上述路由计算方法。设定参数值:$D=50$,$B=70$,$\alpha=1$,$\beta=1$,$\alpha_0=0.5$,$\alpha_l=0.5$;$\tau_{ij}=100$,$A=0$,$H=10$ 和 $M=20$(蚂蚁数目),染色体数目为 20,这些参数都是通过大量仿真实验获得的,这样选取的参数可以获得较好的结果,利用这些参数初始化每个节点的信息素,蚂蚁影响着每个节点的信息素,同时信息素影响连接的路由选择。

仿真中使用 Waxman 随机网络模型。用网络费用的大小来衡量生成多播树质量的好坏,网络费用用仿真运行总次数的平均值来测量,每个仿真节点,运行 50 次,每次模拟节点是从组的节点中随机选取,在模拟的实际过程中,组大小通常设为小于总节点数的 30%,因为在广域网中运行的多播应用经常只涉及到网络中的一小部分节点,诸如视频会议系统和协作编辑系统等。

5.2　算法性能分析

应用蚁群算法求解 QoS多播路由树的结果如图 1 所示。蚁

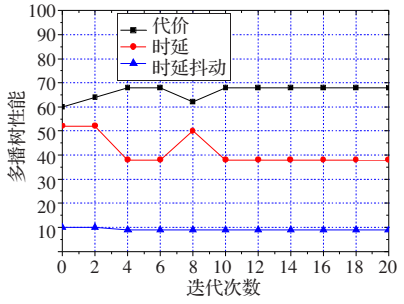

图 1 蚁群算法所求多播树随进化代数变化曲线

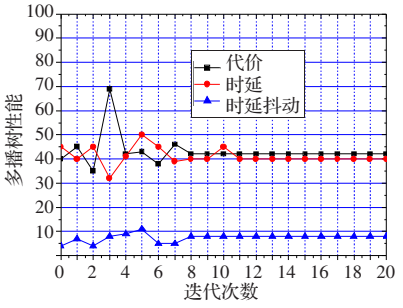

图 2 协同进化蚁群算法所求多播树随进化代数变化曲线

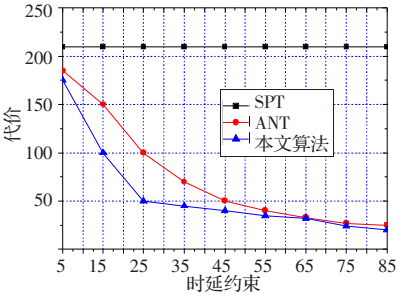

图 3 网络代价随时延变化比较

群算法的代价、延迟、延迟抖动曲线都比较平稳,几乎每一代的蚁群操作均能得到最佳多播树。说明蚁群算法有较好的收敛性。但在仿真实验中,蚁群算法操作的时间复杂度较大,耗费的计算时间较长,影响算法的性能。利用协同进化蚁群算法求解多播树,得到的结果如图 2 所示。其代价曲线比遗传算法的代价曲线波动要略大,说明寻径刚开始时,链路上的信息素信息比较匮乏,蚁群进行随机的全局搜索。但是经过几代的信息累积后,能很快收敛于最优解。延迟曲线相对平稳,表示能较快地找到最优解(或次优解)。延迟抖动曲线起伏不大,表示在最优解附近波动。

图 3 表示网络代价随时延变化的关系。协同进化蚁群算法比蚁群算法在同样的时延约束下所花费的代价要少,这是因为协同进化蚁群算法是在蚁群算法的基础上提出的一种优化算法。

6 结束语

本文提出了一种基于协同进化蚁群算法的 QoS 多播路由优化算法,该算法除了具有分布式算法的特点和低费用外,还能产生较好质量的多播树,且具有较好的稳定性。通过网络仿真实验证明,该算法不仅可以有效地提高网络数据包的传输质量,而且还可以大大节约路由选择时间,很快找到 Pareto 最优解。

参考文献:

[1] RFC 3317 Differentiated services quality of service policy information base[S].

[2] 王颖,谢剑英.一种基于蚁群算法的多媒体网络多播路由算法[J].上海交通大学学报,2002,36(4):526-531.

[3] Lu Guoying,Liu Zemin.Qos multicast routing based on ant algorithm in Internet[J].The Journal of China Universities of Posts ant Telecommunication,2000,7(4):12-17.

[4] Hu G M,Li L M,An H Y.Fast routing algorithm for group multicast with bandwidth reservation[J].Acta Electronica Sinica,2003,31(4):569-572.

[5] 陶振武,肖人彬.协同进化蚁群算法及其在多目标优化中的应用[J].模式识别与人工智能,2005,18(5):588-595.

[6] 许毅.基于 IP 网络 QoS 路由技术的研究[D].武汉:武汉理工大学,2005.

(上接 105 页)

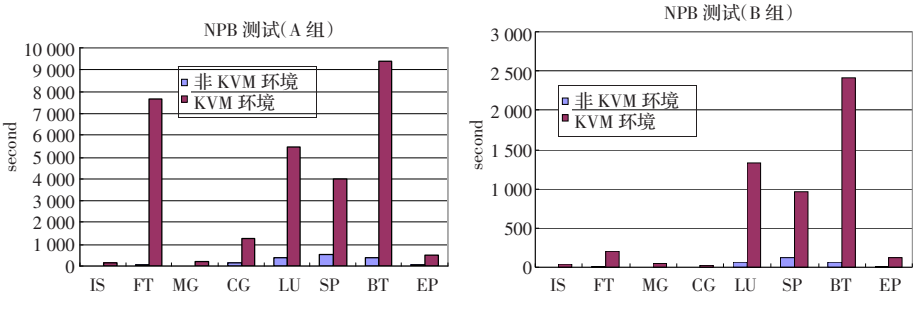

图 5 系统时间对比

分时调度策略简单,效率不是很理想。这个在未来的工作中会进一步改进,以提高整个系统的效率。另外,由于虚拟机具有可迁移的特点,如果能在后续工作中采用这样的技术,将能大大提高系统的效率。

参考文献:

[1] Li J,Dabek F,Berkeley U,et al.F2F:reliable storage in open networks[C]//Proc of the 5th Int'l Workshop on Peer-to-Peer Systems,2006.

[2] Lian Q,Chen W,Zhang Z.On the impact of replica placement to the reliability of distributed brick storage systems[C]//Proc of the 25th IEEE Int'l Conf on Distributed Computing Systems,2005:187-196.

[3] Ian P,Keir F,Steve H,et al.Xen 3.0 and the art of virtualization[C]//Proceedings of Linux Symposium,Volume Two,2005.

[4] Garfinkel T.Terra:a virtual machine-based platform for trusted computing[C]//Proc of the 19th ACM Symp Operating Systems Principles.[S.l.]:ACM Press,2003:193-206.

[5] Bhagwan R,Tati K,Cheng Y,et al.Total recall:system support for automated availability management[C]//Proc of the 1st ACM/Usenix Symp on Networked Systems Design and Implementation,2004.

[6] MPICH2:a new start for MPI implementations[C]//Proceedings of the 9th European PVM/MPI Users' Group Meeting on Recent Advances in Parallel Virtual Machine and Message Passing Interface Table of Contents.London,UK:Springer-Verlag,2002,2474:7.

[7] Kozuch M,Satyanarayanan M.Internet suspend/resume[C]//Proc of the 4th IEEE Workshop Mobile Computing Systems and Applications.[S.l.]:IEEE Press,2002:40.