计算机研究与发展
Journal of Computer Research and Development
ISSN 1000-1239/CN 11-1777/TP
48(9): 1580-1588, 2011

# Internet 资源动态分配的分布计算模型及其系统支撑技术

彭宇行　吴吉庆　沈　锐
(国防科学技术大学并行与分布处理国防科技重点实验室　长沙　410073)
(jiqingwu@gmail.com)

# Distributed Computing Model and Supporting Technologies for the Dynamic Allocation of Internet Resources

Peng Yuxing, Wu Jiqing, and Shen Rui
(*Key Laboratory of Science and Technology for National Defence of Parallel and Distributed Processing, National University of Defense Technology, Changsha* 410073)

**Abstract** Internet computing becomes more and more popular, such as peer-to-peer, grid and cloud computing. Resource allocation problem is a key problem of all kinds of Internet applications. However, the dynamics of Internet resources makes the resource allocation problem one of the challenging issues on the Internet. Aiming at the issue, this paper presents a computational model of the dynamic Internet resource allocation and some related distributed algorithms. Firstly, the dynamics of the Internet resources is analyzed and the result shows which characteristics are invariable in resource allocation process. Secondly, based on the invariable characteristics during the resource allocation process, an organization model of distributed resources, a computational model of allocating resources and the APIs for use of Internet resources are presented. Thirdly, some distributed resource allocation algorithms on system level, such as publishing resources and requesting resources, are presented to support the models. In addition, the definition of good serving peer and the good serving peer selection algorithm are given. Finally, based on the models, two Internet applications are tested. The experiment results show that the models and the algorithms are effective and that the good serving peer selection algorithm can decrease the ratio of the rejected request drastically.

**Key words** distributed computing; Internet resources; dynamic resource allocation; computational model; distributed algorithms

**摘　要** Internet 资源的动态性使得资源分配问题已成为阻碍 Internet 资源获得充分利用的一大难题.为方便用户进行 Internet 应用开发,提出了一个资源动态分配的分布计算模型以及相关的分布处理算法.首先,通过分析 Internet 资源的动态性,分别从资源申请者和资源提供者的角度给出了资源分配过程中哪些特性是不变的;然后,基于资源分配过程中的不变特性,定义了资源使用时的接口描述,提出了适应资源动态变化的分布资源的组织模型和动态资源分配计算模型;研究了支持上述模型的系统支撑技术,提出了相应的分布式资源分配算法;另外,定义了优质服务节点并给出了优质服务节点选择算法,并且通过实验证明优质节点选择算法可以有效地降低服务请求被拒绝的比例;基于上述模型和系统支撑技术实现了两类 Internet 应用,验证了模型和算法的有效性.

**收稿日期**:2010-02-26;**修回日期**:2011-01-10
**基金项目**:国家"八六三"高技术研究发展计划基金项目(2009AA01Z142);国家"九七三"重点基础研究发展计划基金项目(2011CB302601)
**通信作者**:吴吉庆(jiqingwu@gmail.com)

**关键词** 分布计算;Internet 资源;动态资源分配;计算模型;分布算法

**中图法分类号** TP393

资源分配是并行与分布计算的核心问题,在高性能计算机[1-2]、网格计算[3]和云计算[4-5]领域均有对支撑资源分配的系统技术的研究.当前,分布在 Internet 上的大量计算资源正使其逐步演变为无处不在的计算平台[6],其中,网络资源的分配与应用是一个突出的、急待解决的问题[7].

在高性能计算、网格计算、云计算等并行与分布计算中,由于资源是已知的,因此可以通过记录的资源元信息进行集中的资源分配.但在 Internet 中可用资源是变化的,以已知资源为前提的资源分配方法是不适应的.对等计算(P2P)[8-9]为管理 Internet 上的动态资源提供了一种有效途径.它采用了诸如覆盖网节点组织技术、基于邻居关系的路由技术等,在逻辑操作层面上屏蔽了物理节点的动态变化,较好地解决了动态资源的管理以及相关的资源定位、查找问题.但是,P2P 出现的背景是网络文件共享,其研究局限在以文件共享为特征的应用范畴内[10],不直接支持其他领域的可重用开发.

怎样方便地使用 Internet 上的动态资源已成为制约 Internet 资源综合利用的一大难题.其难点在于在可用资源动态变化的情况下,怎样使得上层应用不必关心资源的变化,方便地使用当前能利用的资源.

本文以 Internet 资源为计算平台,提出了资源动态分配计算模型和基于该模型的分布式算法,支持上层应用方便地使用动态的 Internet 资源.其基本思想是从分布的角度出发,为服务申请者提供一个简单的服务请求接口描述,为服务提供者提供一个简单的服务能力接口描述,并在系统层面上进行服务请求和服务能力的匹配,透明地为服务申请者分配资源.这样,在 Internet 应用开发时,开发者不用过多关注资源的动态性及其分配方式,将开发重点放在应用本身,减轻应用开发难度.应用实验表明本文提出的模型和算法是有效的.

## 1 动态资源分配问题

传统的资源分配问题描述如下:设 $s_1,s_2,\cdots,s_m$ 为 $m$ 个服务节点,其计算能力分别为 $E(s_1),E(s_2),\cdots,E(s_m)$,可服务的类型集合对应为 $\Gamma(s_1),\Gamma(s_2),\cdots,\Gamma(s_m)$;设 $b_1,b_2,\cdots,b_n$ 为 $n$ 个服务请求节点,其申请的服务类型分别为 $q(b_1),q(b_2),\cdots,q(b_n)$,服务所消耗的计算能力对应为 $e(q(b_1)),e(q(b_2)),\cdots,e(q(b_n))$;求解一种分配方案,使请求节点能够按需获得服务节点.

与传统的资源分配问题相比,Internet 上动态资源的分配问题具有以下不同点:①服务节点 $s_1,s_2,\cdots,s_m$ 是动态的且使用时是随时可能掉线的;②计算能力 $E(s_i)$,$i=1,2,\cdots,m$ 也是动态变化的;③服务请求 $q(b_1),q(b_2),\cdots,q(b_n)$ 是随机到达的;④不存在一个固定的计算节点来管理这些请求节点、服务节点及其相关的信息.因此,我们需要从分布的角度来分析 Internet 上动态资源的分配问题.

从请求节点 $b_j$ 的角度看,其可知的特性只有服务需求 $q(b_j)$,而具体由哪个 $s_i$ 为自己服务是受环境变化影响的.因此,在资源分配时 $b_j$ 只需要有途径提出自己的服务申请即可.

同样,从服务节点 $s_i$ 的角度看,其可知的特性是当前的服务能力 $E(s_i)$和可服务类型 $\Gamma(s_i)$,而针对当前到达的服务请求 $q(b_j)$,究竟是否由本服务节点进行服务,也是受环境变化影响的.因此,在资源分配时,$s_i$ 也只需要有途径报告自己的服务能力 $E(s_i)$和可服务类型即可,至于具体分配哪一个 $s_i$ 为 $b_j$ 服务,则需对 $s_1,s_2,\cdots,s_m$ 进行综合比较得到.

**定义 1.** 服务节点的抗耗能力.设 $s$ 为一服务节点,$E(s)$为 $s$ 当前的服务能力,$\Gamma(s)=\{q_j\mid j=1,2,\cdots,l\}$为 $s$ 当前的可服务类型集合,$e(q_j)$ 为 $q_j$ 服务需消耗的计算能力,$j=1,2,\cdots,l$,则 $s$ 的抗耗能力定义为

$$K(s)=\frac{E(s)}{\sum_{j=1}^{l}e(q_j)}.$$

**定义 2.** 优质服务节点.设 $s_1,s_2$ 为两个服务节点,如果满足下述条件,称 $s_1$ 优于 $s_2$:

① $|\Gamma(s_1)|>|\Gamma(s_2)|$,即 $s_1$ 当前的可服务类型数多于 $s_2$ 当前的可服务类型数;

② 若$|\Gamma(s_1)|=|\Gamma(s_2)|$,则 $K(s_1)\geqslant K(s_2)$,即两个服务节点的可服务类型数相同时,$s_1$ 当前的抗耗能力强于 $s_2$.

基于定义 1、定义 2,若已知当前的服务请求为 $q_k$,其消耗的计算能力为 $e(q_k)$,若分配 $s_i$ 为 $q_k$ 服务,$s_i$ 当前的服务能力为 $E(s_i)$,可服务类型为 $\Gamma(s_i)=$

$\{q_j|j=1,2,\cdots,l\}$, $q_k\in\Gamma(s_i)$, 则算法 1 给出了对 $q_k$ 进行服务后 $\Gamma(s_i)$ 和 $K(s_i)$ 的计算方法.

**算法 1.** 可服务类型和抗耗能力的计算.

$E(s_i)=E(s_i)-e(q_k)$
for $j=1$ to $l$ do{
  if $e(q_j)>E(s_i)$ {
    $\Gamma(s_i)=\Gamma(s_i)-q_j$;
  }
}
$sum=0$;
for $j=1$ to $|\Gamma(s_i)|$ do{
  $sum=sum+e(q_j)$;
}
$K(s_i)=E(s_i)/sum$.

若 $s_1,s_2,\cdots,s_m$ 为服务请求为 $q_k$ 当前可选的服务节点, 基于算法 1, 已求得各节点执行 $q_k$ 后的服务类型数和抗耗能力, 则算法 2 给出了优质服务节点的选择过程. 算法基本思想是服务节点选择后, 系统余下的可服务类型最多, 抗耗能力最强.

**算法 2.** 优质服务节点的选择.

$|\Gamma(s)|=0$;
$K(s)=0$;
for $i=1$ to $m$ do{
  if $|\Gamma(s)|<|\Gamma(s_i)|$ {
    $s=s_i$;
  }
  else{
    if $|\Gamma(s)|=|\Gamma(s_i)|$ and $K(s)<K(s_i)$
      $s=s_i$;
  }
}

如上所述, 若要将 Internet 设计为一个方便的计算平台, 其系统支撑环境需要提供下述功能: ①为 $b_j$ 提供请求 $q(b_j)$ 服务的接口; ②为 $s_i$ 提供发布 $E(s_i)$ 和 $\Gamma(s_i)$ 的接口; ③对随机到达的 $q(b_j)$, $j=1,2,\cdots,n$, 支持 $s_1,s_2,\cdots,s_m$ 对其进行服务分配. 然而, 在资源信息动态分布的状态下, 怎样获得上述 $E(s_i)$, $\Gamma(s_i)$ 和 $K(s_i)$ 并不是一件容易的事情, $i=1,2,\cdots,m$. 它需要组织和管理动态的 $s_1,s_2,\cdots,s_m$, 通过 $s_1,s_2,\cdots,s_m$ 的协作求得动态变化的 $E(s_i)$, $\Gamma(s_i)$ 和 $K(s_i)$. 这样, 在每个 $b_j$, $j=1,2,\cdots,n$, 申请 Internet 动态资源时, 以及在每个 $s_i$, $i=1,2,\cdots,m$, 提供服务时, 它们都能够简单地通过调用系统接口达到目的, 不用考虑 $s_1,s_2,\cdots,s_m$ 怎样变化, 以及 $b_1,b_2,\cdots,b_n$ 何时到达等复杂的资源动态性问题.

## 2 分布资源动态分配的计算模型

对请求节点 $b_j$ 申请服务 $q(b_j)$, 定义接口 $request(b_j,q(b_j))$; 对服务节点 $s_i$ 发布服务类型 $q_k$, 定义接口 $bulletin(s_i,q_k)$; 对服务节点 $s_i$ 取消服务类型 $q_k$, 定义接口 $delete(s_i,p_k)$. 为支持请求节点对 $request(b_j,q(b_j))$ 的调用, 支持服务节点对 $bulletin(s_i,p_k)$, $delete(s_i,p_k)$ 的调用, 我们提出分布资源服务分配的资源组织模型和分配计算模型.

**定义 3.** 分布资源的组织模型. 设 $S=\{s_i|i=1,2,\cdots\}$ 为节点集, $D=\{d_j|j=1,2,\cdots\}$ 为数据集, $S$ 中的节点按以下方式进行组织:

① 建立 $D$ 到 $S$ 的映射函数 $F$, 对任意 $d_j\in D$, 确保将其映射到一确定的 $s_k$ 上, $s_k\in S$, 即 $F(d_j)=s_k$;

② 建立 $S$ 中各节点之间的关联关系以及节点间的可达算法 $A$, 确保任一节点 $s_i$ 上的数据 $d_j$, 其索引信息能通过 $A$ 传送到节点 $F(d_j)$ 上.

函数 $F$、算法 $A$ 的实现方式有多种, 如 $F$ 为 Hash 函数 SHA1[11], 关联关系为 Kautz 图连接方式, 算法 $A$ 为 FissionE[12] 路由算法函数等.

**定义 4.** 资源分配的计算模型. 设 $S=\{s_i|i=1,2,\cdots,m\}$ 为服务节点集, $b_j$ 为请求节点, $q(b_j)$ 为服务请求, 则资源分配的计算模型如下:

① 服务节点 $s_i$, $i=1,2,\cdots,m$, 利用函数 $F$ 将自己所能提供的服务类型 $q_k$ 映射到服务节点 $F(q_k)$ 上, 并通过算法 $A$ 将"$s_i$ 能够提供 $q_k$ 服务"公告到 $F(q_k)$ 上;

② 请求节点 $b_j$ 利用函数 $F$ 将请求 $q(b_j)$ 映射到服务节点 $F(q(b_j))$ 上, 并通过算法 $A$ 将"$b_j$ 申请 $q(b_j)$ 服务"导航到 $F(q(b_j))$ 上;

③ $F(q(b_j))$ 用 $q(b_j)$ 匹配公告信息, 获得能够提供 $q(b_j)$ 服务的节点集, 并通过算法 2 从中获得优质服务节点进行服务.

基于定义 3、定义 4, 若 $q_k$ 是服务节点 $s_i$ 的可服务类型, 则可利用函数 $F$ 将信息 $(s_i,q_k)$ 映射到节点 $F(q_k)$ 上, 并利用算法 $A$ 公告到 $F(q_k)$ 节点上; 若 $q(b_j)$ 是请求节点 $b_j$ 所申请的服务, 则可利用函数 $F$ 将信息 $(b_j,q(b_j))$ 映射到节点 $F(q(b_j))$ 上, 并利用算法 $A$ 导航到 $F(q(b_j))$ 节点上. 而对于应用层来说, 其相应工作仅仅是分别调用 $bulletin(s_i,q_k)$ 和 $request(b_j,q(b_j))$ 操作. 具体 $bulletin(s_i,q_k)$, $request$

$(b_j, q(b_j))$怎样运行,则由分布运行于各节点上的算法 3、算法 4 作为系统技术来支撑:

**算法 3.** 服务公告.

when $bulletin(s_i, q_k)$被调用{
  用函数 $F$ 将服务类型 $q_k$ 映射到服务节点 $F(q_k)$;
  用算法 2 求得$|\Gamma(s_i)|$和 $K(s_i)$;
  用算法 $A$ 将$(bulletin, s_i, |\Gamma(s_i)|, K(s_i), q_k, F(q_k))$发送到下一个节点;
}
when 接收到公告信息$(bulletin, s_i, |\Gamma(s_i)|, K(s_i), q_k, F(q_k))${
  if 如果本节点为 $F(q_k)${
    记录公告信息$(s_i, |\Gamma(s_i)|, K(s_i), q_k)$;
  }
  else{
    用算法 $A$ 将$(bulletin, s_i, |\Gamma(s_i)|, K(s_i), q_k, F(q_k))$发送到下一个节点;
  }
}

**算法 4.** 服务请求.

when $request(b_j, q(b_j))$被调用{
  用函数 $F$ 将申请类型 $q(b_j)$映射到服务节点 $F(q(b_j))$;
  用算法 $A$ 将$(request, b_j, q(b_j), F(q(b_j)))$发送到下一个节点;
}
when 接收到请求信息$(request, b_j, q(b_j), F(q(b_j)))${
  if 如果本节点为 $F(q(b_j))${
    用 $q(b_j)$匹配"公告记录"中的 $q_k$;
    调用算法 2 选择一优质服务节点 $s_k$ 发送给 $b_j$;
  }
  else{
    用算法 $A$ 将$(request, b_j, q(b_j), F(q(b_j)))$发送到下一个节点;
  }
}
when 接收到服务节点为 $s_k$ 的通知{
  连接 $s_k$;
  通知 $s_k$ 执行 $q(b_j)$服务;
}

依据算法 3,若多个节点都具有类型为 $q_k$ 的服务能力,则这些信息都公告到节点 $F(q_k)$中;依据算法 4,若请求节点申请类型为 $q_k$ 的服务,则该请求信息也能导航到节点 $F(q_k)$中. 亦即算法 3、算法 4 将不同服务请求的资源分配问题分布到不同服务节点上求解.

## 3 服务能力的动态修改

虽然节点 $F(q_k)$中记录了所有的能够为 $q_k$ 申请提供服务的节点信息,但是要从这些服务节点中选取定义 2 所述的优质服务节点,还需对这些服务节点的服务能力、可服务类型数、抗耗能力等信息进行动态修改.

**算法 5.** 节点服务能力修改.

when 接收到 $b_j$ 传来的执行 $q(b_j)$服务通知{
  调用算法 1 求得新的 $E(s_i)$,$|\Gamma(s_i)|$和 $K(s_i)$;
  for 已公告的服务类型 $q_k${
    if 本节点 $s_i$ 不再具有 $q_k$ 的服务能力{
      call $delete(s_i, q_k)$;
    }
    else{
      call $bulletin(s_i, q_k)$;
    }
  }
  向 $b_j$ 提供 $q(b_j)$服务;
}
when 执行完 $q(b_j)$服务{
  $E(s_i) = E(s_i) + e(q(b_j))$;
  调用算法 1 求得新的 $E(s_i)$,$|\Gamma(s_i)|$和 $K(s_i)$;
  for 已公告的服务类型 $q_k${
    call $bulletin(s_i, q_k)$;
  }
}

**算法 6.** 服务注销.

when $delete(s_i, q_k)$被调用{
  用函数 $F$ 将服务类型 $q_k$ 映射到服务节点 $F(q_k)$;
  用算法 $A$ 将$(delete, s_i, q_k, F(q_k))$发送到下一个节点;
}
when 接收到注销信息$(delete, s_i, q_k, F(q_k))${
  if 如果本节点为 $F(q_k)$ then{
    注销记录$(s_i, |\Gamma(s_i)|, K(s_i), q_k)$;
  }

```
    else{
      用算法 A 将(delete, s_i, q_k, F(q_k))发送到
      下一个节点;
    }
  }
```

算法 5 给出了服务节点服务能力变化时对公告信息的修改过程,即每增加一项服务余留的服务能力减小,每完成一项服务余留的服务能力增加,均需对服务能力的变化重新公告. 算法 6 给出了对服务节点进行注销的分布执行过程.

## 4 实验结果

我们通过文件应用实验比较了优质服务节点选择算法、服务节点随机选择算法[13-14]以及服务节点贪婪选择算法[15-16]的平均服务拒绝比例. 首先,将要下载的文件分为 5 000 个文件块,视为 5 000 个不同的服务类型;在实验初始阶段,只有一个种子节点拥有该文件的所有数据块;当其他节点得到某个数据块后,发布该数据块的可服务信息. 数据块请求节点通过查询得到拥有该数据块的节点集,并分别使用上述 3 种服务节点选择算法选择一个服务节点发送该数据块. 如果没有节点拥有该数据块,或所有拥有该数据块的节点带宽都被耗尽,则视为该请求被拒绝. 图 1 统计了 200 个节点进行文件共享时被拒绝请求数占总请求数的比例随时间的变化值. 实验结果表明,与服务节点随机选择算法、服务节点贪婪选择算法相比,优质服务节点选择算法降低了被拒绝的请求比例:

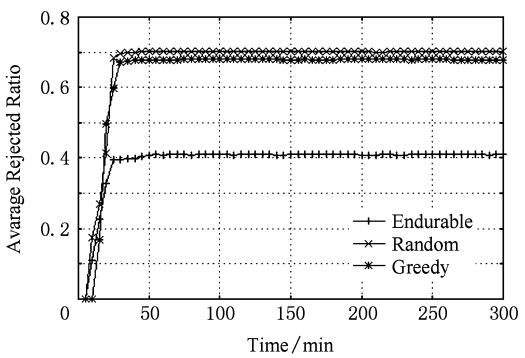



Fig. 1 The average rejected ratio of the three algorithms.

图 1 3 种算法的服务被拒绝比例

## 5 模型应用

视频点播和网络信息爬取是两种典型的 Internet 应用. 基于 P2P 的流媒体传输被认为是当前最有效视频点播方法,其资源分配过程包括节点的组织与动态维护、消息路由、服务节点定位等;网络信息爬取是利用 Internet 获取各种信息的一种手段,基于 Internet 动态资源的爬取方法被认为是对抗网站屏蔽的最有效手段. 两类应用的共同特点是开发难度较大且可重用性不高.

本节中我们基于上述模型和方法重写了上述应用. 可以看出开发过程被显著简化.

### 5.1 视频点播

基于 P2P 视频点播的难点在服务节点定位,因为资源的动态性使得给某一请求提供服务的资源是变化的,它首先表现在节点的自治性使得前一时刻提供服务的资源,下一时刻离开了网络,必须重新寻找服务节点;其次,服务节点能力的变化使得点播节点需要时刻更新服务节点以获得更好的效果. 这种服务节点的不断更换增加了开发的难度.

采用定义 4 所述的计算模型,并基于算法 3、算法 4 所提供的系统支撑程序,视频点播应用程序可以不关心上述工作内容,仅通过简单地调用 $request(b_j, q(b_j))$ 即可实现. 我们采用该计算模型对我们原有的视频点播原型系统进行改写,利用 owlet 语言[17]改写了数据的申请和服务的交互过程.

点播节点数据申请过程的主要流程如下. 很明显,开发过程变得十分简单.

```
role Requester{
  int k; /* 当前申请的数据块号 */
  /* 申请第 1 块数据 */
  when(self:take(File filename)){
  /* 点播开始 */
    k=1;
    request(filename,k);
  }
  /* 申请下一块数据 */
  when(self:notFill(buffer)){
  /* 当前数据块缓冲区不满 */
    if(k is not the last piece){
      k=k+1;
      request(filename,k);
    }
  }
  /* 取数据 */
  when(Provider p: provide(File filename,
  Piece k,string url))  {
```

```
    /*服务节点地址返回*/
      fetchPiece(filename,k,url);
      buf=fetchPiece(filename,k,url);
      fillBuffer(buf);
      bulletin(selfname,(filename,k));
      /*本节点服务公告*/
    }
  }
```

服务节点数据匹配、传送、定点公告等主要流程如下:

```
  role Provider{
    /*数据匹配、路由*/
    when(Requester r: request(File file, int k)){  /*当接收到服务申请*/
      if(piece k of file is bulletined in this node){
        src_url=goodServPeer();
        /*用算法 2 选择优质服务节点*/
        provide(file,k,src_url);
        /*返回选择的服务节点*/
      }
      else{
        routeNext(file,k,r);
        /*路由至下一服务节点*/
      }
    }
    /*数据传送*/
    when(Requester r: fetchPiece(self, File file,int k)){
    /*当接收到数据服务请求*/
      sendPiece(file,k,r);
    }
    /*数据公告、路由*/
    when(Requester r: bulletin(Data (File file,int k),Url target_url)){
    /*当接收到公告消息*/
     if(local url is the target_url){
       store(file,k,r);
     }
     else{
       routeNext(r,(file,k),target_url);
       /*路由至下一服务节点*/
     }
   }
  }
```

系统开发后运行于下述实验环境:由 3 个管理域组成的广域网;每个管理域由 16 台服务器组成的局域网构成;域间通过链路损伤仪相连,由链路损伤仪模拟广域网行为,如图 2 所示. 登录界面如图 3(a)所示,运行效果如图 3(b)所示. 实验结果表明,按照定义 2 所述计算模型所改写的视频点播原型系统与原有系统运行效果一致.

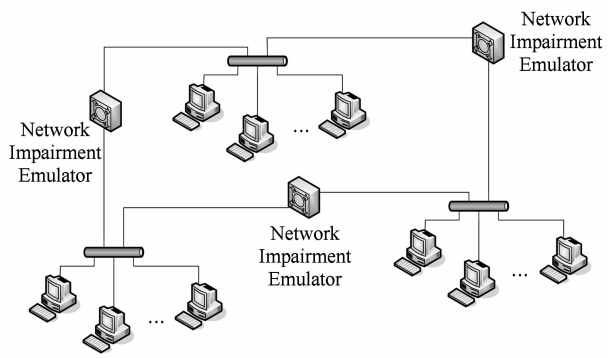



Fig. 2 The experiment environment of the VoD application.

图 2 视频点播应用的实验环境


(a) An agent join the VoD application (b) The screen shot of the client

Fig. 3 The screen shots of the VoD application.

图 3 视频点播应用的运行效果

## 5.2 网络信息爬取

网络信息爬取是利用 Internet 获取各种信息的一种手段. 其基本组成包括任务分配、信息获取和信息分析 3 个部分,如图 4 所示. 信息获取部分由应用

部门的计算节点和互联网自愿贡献资源的用户节点组成,按 Kautz 图方式组织,安装“爬虫”程序[18],协同爬取互联网网页、论坛与博客等信息.

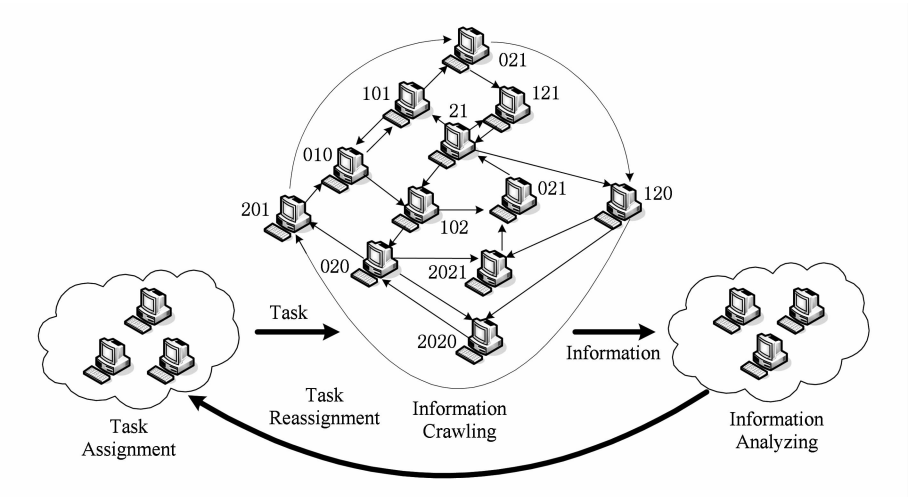



Fig. 4 The structure of network monitoring and management system.

图 4 信息监管系统的结构组成

信息爬取的执行方式如下:

① 任务分配节点在信息获取节点中随机指定几个节点执行具体任务分发;

② 信息获取节点 $s_i$ 通过 Hash 函数 SHA1 将自己的“爬取”能力 $p_k$ 映射到信息获取节点 $s_j$ 上,并通过 $bulletin(s_i, p_k)$ 在 $s_j$ 上公告;

③ 具体任务分发节点 $b_j$ 通过 Hash 函数 SHA1 将“爬取”服务请求 $q(b_j)$ 映射到信息获取节点 $s_i$ 上,并通过 $request(b_j, q(b_j))$ 发出类型为 $q(b_j)$ 的“爬取”服务请求;

④ 信息获取节点通过路由算法,将 $(b_j, q(b_j))$ 发送到公告了 $q(b_j)$ 的 $s_i$ 上;$s_i$ 通过匹配,选取能够执行该“爬取”任务的信息获取节点 $s_k$,将结果 $s_k$ 返回至任务分派节点 $b_j$;

⑤ 具体任务分发节点 $b_j$ 指示 $s_k$ 执行“爬取”任务,并指示将爬取结果发送至信息分析节点;

⑥ 信息分析节点执行分析任务,并根据结果调整或重新安排任务.

具体任务分发及信息获取主要流程的 owlet 程序如下.同样,开发过程也十分简单.任务分发节点的流程如下:

```
role Requester{
  /* 任务申请 */
  when(self:take(Task task)){
  /* 当存在未分配任务 */
    request(task);
  }
  when(Provider p: provide(Task task,
   Type type,string url)){
   /* 当返回爬取节点的地址 */
    crawl(task,url);
  }
}
```

信息获取节点的流程如下:

```
role Provider{
  /* 数据匹配、路由 */
  when(Requester r: request(Task task)){
  /* 当接收到“爬取”服务申请 */
    if(task is bulletined in this node){
      src_url=goodServPeer();
      /* 用算法 2 选择优质服务节点 */
      provide(task,src_url);
      /* 返回优质节点 */
    }
    else{
      routeNext(task);
      /* 路由至下一信息获取节点 */
    }
  }
  /* 爬取 */
  when(Requester r: crawl(Task task)){
    crawl(task);
  }
  /* 数据公告、路由 */
  when(Requester r: bulletin(Task task,
  string target_url)){
```

```
    /* 当接收到公告消息 */
      if(local url is the target_url){
        store(task,r);
      }
      else{
        routeNext(r,task,target_url);
        /* 路由至下一信息获取节点 */
      }
    }
  }
```

程序开发后运行于 PlanetLab[19] 实验环境,同时聚合分布于亚洲、北美、欧洲、南美、大西洋的 150 余台节点协同爬取信息,如图 5 所示,平均每分钟爬取 10 GB 左右信息. 实验结果同样表明,定义 2 所述的计算模型是有效的.

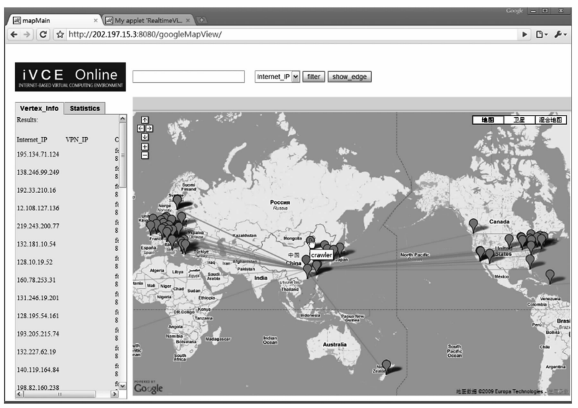



Fig. 5 The experiment of network monitoring and management on PlanetLab.

图 5 PlanetLab 上的网络信息监管实验

## 6 结束语

本文以简化 Internet 应用的开发为目标研究了 Internet 资源分配问题. 针对资源的自治性和动态性,提出了分布资源服务分配的资源组织模型和分配计算模型,以及支撑该模型运行的分布式算法,将复杂的开发过程简化为类 $request(b_j, q(b_j))$, $bulletin(s_i, p_k)$的服务调用. 应用实验表明,本文所提出的模型和算法能够有效简化应用设计.

## 参 考 文 献


[1] Brightwell R. System software R&D at Sandia: To red storm and beyond [R]. Boulder: National Center for Atmospheric Research, Computational and Information Systems Laboratory Seminar, 2007

[2] Bobroff N, et al. A distributed job scheduling and flow management system [J]. ACM SIGOPS Operating Systems Review, 2008, 42(1): 63-70

[3] Siddiqui M, Fahringer T. Grid Resource Management: On-Demand Provisioning, Advance Reservation, and Capacity Planning of Grid Resources [M]. Berlin: Springer, 2010

[4] Schmidt R, Sadilek C, King R. A service for data-intensive computations on virtual clusters [C] //Proc of the 1st Int Conf on Intensive Applications and Services. Los Alamitos, CA: IEEE Computer Society, 2009: 28-33

[5] Tian Wenhong, Su Sheng, Lu Guoming. A framework for implementing and managing platform as a service in a virtual cloud computing laboratory [C] //Proc of the 2nd International Workshop on Education Technology and Computer Science. Los Alamitos, CA: IEEE Computer Society, 2010: 273-276

[6] Lazowska E D, Patterson D A. Distributed computing [OL]. [2010-02-26]. http://www.sciencemag.org/sciext/computers/

[7] Lu X, Wang H, Wang J. Internet-based virtual computing environment (ivce): Concepts and architecture [J]. Science in China Series F: Information Sciences, 2006, 49 (6): 681-701

[8] Pedro Fonseca, Rodrigo Rodrigues, Anjali Gupta, Barbara Liskov. Full-information lookups for peer-to-peer overlays [J]. IEEE Trans on Parallel and Distributed Systems, 2009, 20(9): 1339-1351

[9] Stephanos Androutsellis-Theotokis, Diomidis Spinellis. A survey of peer-to-peer content distribution technologies [J]. ACM Computing Surveys, 2004, 36(4): 335-371

[10] Zhang Guimin, Smith B L, Asce M, et al. Peer-to-peer-based publish/subscribe architecture for advanced infrastructure systems [J]. Journal Computing in Civ. Engrg., 2010, 24(1): 65-72

[11] Eastlake D, Jones P. RFC3174: US secure Hash algorithm 1 (SHA1)[EB/OL]. (2001-09) [2010-02-26]. http://www.faqs.org/rfcs/rfc3174.html

[12] Li Dongsheng, Lu Xicheng. A novel constant degree and constant congestion DHT scheme for peer-to-peer networks [J]. Science in China Series E: Technology Sciences, 2004, 34(12): 1-22 (in Chinese)
(李东升, 卢锡城. P2P 网络中常量度数常量拥塞的 DHT 方法研究[J]. 中国科学 E 辑: 技术科学, 2004, 34(12): 1-22)

[13] Hu S-Y, Huang T-H, Chang S-C, et al. Flod: A framework for peer-to-peer 3d streaming [C] //Proc of IEEE 2008 Int Conf on Computer Communications. Los Alamitos, CA: IEEE Computer Society, 2008: 1373-1381

[14] Liang C, Guo Y, Liu Y. Is random scheduling sufficient in P2P video streaming? [C] //Proc of IEEE ICDCS'08. Los Alamitos, CA: IEEE Computer Society, 2008: 53-60

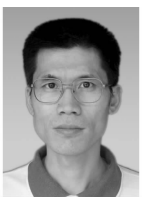
[15] Xu D, Hefeeda M, Hambrusch S, et al. On peer-to-peer media streaming [C] //Proc of IEEE ICDCS'02. Los Alamitos, CA: IEEE Computer Society, 2002: 363-371

[16] Zhou Y, Chiu D M, Lui J C S. A simple model for analyzing P2P streaming protocols [C] //Proc of IEEE ICNP'07. Los Alamitos, CA: IEEE Computer Society, 2007: 226-235

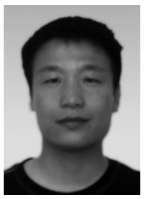
[17] Shen Rui. The owlet programming language [CP/OL]. http://owlet-code.sourceforge.net

[18] Xu Xiao, Zhang Weizhe, Zhang Hongli, et al. A forwarding-based task scheduling algorithm for distributed Web crawling over DHTs [C] //Proc of the 15th Int Conf on Parallel and Distributed Systems. Los Alamitos, CA: IEEE Computer Society, 2009: 854-859

[19] Princeton University. PlanetLab Website [OL]. (2005-01-03). [2010-02-26]. http://www.planet-lab.org/



**Peng Yuxing**, born in 1963. Professor and PhD supervisor of the National University of Defense Technology. His main research interests include distributed computing (pengyuxing1963@yahoo.com.cn).



**Wu Jiqing**, born in 1980. Received his MSc Degree from the National University of Defense Technology in 2004. Since 2005, he has been a PhD candidate in computing science at the National University of Defense Technology. His current research interests include P2P streaming and P2P computing.

**Shen Rui**, born in 1979. Received his BSc and MSc degrees in computing science from the National University of Defense Technology in 2002 and 2004 respectively. Since 2005, he has been a PhD candidate in computing science at the National University of Defense Techonology. His current research interests include distributed program language design(rui.shen@gmail.com).