下一代网络中基于相邻资源的分布式过载控制机制研究

王开西 杨放春
(北京邮电大学网络与交换国家重点实验室 北京 100876)

摘 要：下一代网络(NGN)不同功能实体由于业务请求形成依赖关系，根据这种依赖关系和 NGN 相对扁平体系结构的特点，该文给出“相邻资源”的概念，查找过载点的相邻资源，在相邻资源实施分布式过载控制。分析可以看出，该算法能有效解决 SIP 协议的分发功能和实施负载平衡带来的过载控制问题，同时尽可能早地拒绝业务请求，提高网络的有效利用率，较好地满足 ETSI TISPAN 对 NGN 过载控制定义的需求。

关键词：下一代网络；体系结构；过载控制；相邻资源
中图分类号：TP393 **文献标识码**：A **文章编号**：1009-5896(2008)03-0690-05

Study on the Overload Control Mechanism Based on the Adjacent Resourse in NGN

Wang Kai-xi Yang Fang-chun
(*State Key Laboratory of Networking & Switching Technology*,
Beijing University of Posts & Telecommunications , *Beijing* 100876, *China*)

Abstract: The dependence is formed between different function entities due to service request. The concept named “adjacent resources” is proposed according to the dependence and the NGN’s flatter architecture than the traditional telecommunication network. A distributed algorithm for overload control is presented, which is based on the proposed concept. It throttles the request at the logical adjacent resources. The algorithm can effectively handle with the overload control issuing from SIP fork or load balance scenarios. The mechanism can improve the network effective throughput by throttling the service request as soon as possible, which satisfies with the overload control requirements presented in the ETSI TISPAN draft.

Key words: Next Generation Network (NGN); Architecture; Overload control; Adjacent resource

1 引言

不管是在传统电信网络中，还是在基于 IP 的 Internet 网络里，以及提供智能业务的 IN 业务网中，过载控制都是保证网络超负荷状态下能正常运行必须解决的问题。下一代网络(Next Generation Network, NGN)是分布式、分层、分平面的网络体系结构[1]，具有多协议、多任务的特点。业务提供在分布式体系结构中要由多个分布功能实体交互协作完成。大量复杂的交互使得实体之间的负荷状态相互影响，这增加了对系统过载控制的难度[2−4]。不同协议的过载控制能力各不相同[3]，多业务运行环境等因素都加剧了过载控制的难度。实施过载控制，除要在过载点进行内部接纳控制外，还需要一种合理有效的外部过载控制机制，以防过载点的资源全部用来拒绝新的业务请求，降低有效业务吞吐量，这已成为过载控制的一个基本原则[3]。文献[5]提出在邻接交换机过滤业务请求的方法实施外部控制，这种方法在垂直业务体系结构的 SS7 网中具有较好的效果，但它并不能解决相对传统电信网来说具有扁平结构特点的 NGN 过载控制问题，如文献[4]所介绍的SIP协议所具有的分发功能以及实施负载平衡后对过载控制的恶化。

根据 NGN 总体体系结构[1]及 IMS 业务提供结构[6]的分层和相对扁平结构特点，本文提出“相邻资源”概念，并基于此概念给出一种适合于 NGN 的分布式过载控制算法。最后以文献[4]中的场景为例说明算法的有效性。

2 相关研究

人们对不同网络中的过载控制都进行了大量研究[7−9]，提出基于请求队列长度、响应时间、处理时间差等不同的过载控制检测方法，并通过自动呼叫限制(Automatic Call Gapping, ACG)、比例限制(Thin Percent, TP)和漏桶算法(Leaky Bucket, LB)等不同方法过滤业务请求。这些方法对 NGN 的过载控制都有借鉴意义。文献[10, 11]在比较 SS7 信令中国际、国内不同过载选项后，认为国际选项具有更好的吞吐量和健壮性；文献[12]研究也表明在 TCP/IP 网络中端到端的过载更有效。端到端的分布式过载控制优势在于：对不能满足需求的业务请求要尽早拒绝，以免网络资源无效使用，这有利于保证完成处理中的业务请求。文献[13]以泊松

2006-8-22 收到，2007-07-06 改回
高等学校博士学科点专项科研基金(20020013004)，国家 973 计划(2003CB314806)，教育部长江学者和创新团队发展计划(通信网的网络理论和技术)(IRT0410) 资助课题

过程刻画呼叫级业务请求到达率，在对比限制到达率和缓冲区大小两种方法后，提出一种不依赖于反馈信息的分布式过载控制。文献[14]分别提出基于代理和市场机制的分布式过载控制机制，这些机制以最终用户为代理实体或市场实体，大量地协商可能会降低网络利用率。从以上的研究可以看出，我们要尽可能早地拒绝不能成功完成的业务请求，以提高系统有效利用率。而文献[5]提出在相邻结点实施外部控制的方法依赖于 SS7 信令网的垂直业务体系结构，不能解决上面提到的 SIP 分发和负载平衡带来的问题。除了针对具体协议的过载控制研究外，目前对 NGN 的通用过载控制标准的研究主要有两个草案，它们是 ETSI TISPAN 组织提出的草案[3]和 MSForum 论坛的技术报告[2]。ETSI TISPAN 草案比较了过载控制组件不同部署方案的优劣，并提出适合 NGN 网络的通用过载控制应用协议(GOCAP)[3]。但它在比较各种部署方案时并未涉及如何选择合适的过载控制点。

3　基于相邻资源的分布式过载控制

3.1 NGN 体系结构

从 NGN 总体体系结构[1]可以看出，控制层由多个业务子层组成，如：IP 多媒体子系统(IMS)，PSTN/ISDN 仿真子系统，流媒体子系统，内容广播子系统等组成。提供多媒体业务的 IMS 子系统包含 6 类主要的功能实体[6]：会话管理和路由控制(CSCFs)，数据库(HSS，SLF)，互联实体(BGCF，MGCF，IM-MGW，SGW)，媒体资源功能实体(MRFC，MRFP)，服务(AS，MRFC，MRFP)和支撑实体(THIG，SEG，PDF)等。它们之间相互协作共同完成用户需求，这使得它们之间具有依赖关系。此外，网络附着子系统(NASS)和资源接纳控制子系统(RACS)向 NGN 的控制层和业务层提供具有 QoS 保证的传输服务，它们和业务层和控制层也具有依赖关系。将业务子层和传输层划分为不同功能实体后，各功能实体可相对独立发展，也使得物理上易于分布式部署，降低了对单个硬件设备的要求，促进商业上的应用。但是这种分离也增加了功能实体间或者说是物理设备间的依赖关系，这种依赖关系加剧了对过载控制的难度。因此，除各类功能实体应具有内部的过载控制机制外，这些功能实体间还要进行过载协调控制。因此，在这些功能实体间需要一种反馈机制，由分布在各功能实体的过载控制组件协作完成过载控制功能，从而有效地保护系统，防止出现系统过载。

3.2 相邻资源

相比 PSTN 网络垂直、分层的实现对过载控制的自然帮助，从上面可以看出相对扁平结构[6]的 NGN 各组件之间存在着大量复杂、自由对等的交互。这些交互增加了过载控制的难度。为刻画不同实体间的依赖关系，我们提出逻辑上“相邻资源”的概念。

设业务请求中的任意两个实体 A，B，$S=\{s_i\}$ 表示 A，B 间路由链路上包括 A，B 在内的服务实体的集合，其中 s_i 表示参与完成业务请求的一个服务实体，则 A，B 间业务请求的任一条路由链路可由分段的服务实体向量表示：

$$\mathbf{route}_{\mathrm{req}}(\mathrm{A,B}) = \langle < \mathbf{servers}_1 >, < \mathbf{servers}_2 >, \cdots, < \mathbf{servers}_n > \rangle \quad (1)$$

其中 $\forall i,j \in [1,n]$ 且 $i \neq j$ 和 $\forall s_x, s_y \in S$，$(< s_x, s_y > \in \mathbf{servers}_i \wedge < s_x, s_y > \in \mathbf{servers}_j) \neq 1$ 成立。这里，除以向量为操作数的 $\in$ 运算表示隶属运算外，其它符号都是数学中的逻辑运算或集合运算。隶属运算的定义如下：

隶属运算 $\in$　若二元组 $< s_x, s_y >$ 是向量 $\mathbf{servers}_i$ 的一个子序列(其中 $i \in N \wedge i \in [1,n]$ 且任意 $s_x, s_y \in S$，n 为 $\mathbf{route}_{\mathrm{req}}(\mathrm{A,B})$ 中的子向量个数)，则运算 $< s_x, s_y > \in \mathbf{servers}_i$ 为真，否则运算为假。

合并运算 $\cup$　若 $\forall i,j,k \in [1,n]$ 且 $i \neq j$，$\forall s_x, s_y, s_z, s_r \in S$ 且 $s_z \neq s_r$，s_y 不过载，下列条件成立：

$$\begin{cases} < s_x, s_y > \in \mathbf{servers}_i \wedge < s_y, s_z > \in \mathbf{servers}_j \\ \exists < s_y, s_r > \in \mathbf{servers}_k \text{ 或 } \exists < s_y, s_r > \in \mathbf{servers}_r \text{ 且所有的 } s_r \end{cases}$$

都不过载，则定义合并运算：$< s_x, s_y > \cup < s_y, s_z > = < s_x, s_z >$。

定义 1　对 $\mathbf{route}_{\mathrm{req}}(\mathrm{A,B})$ 中的 $\mathbf{servers}_i$ 进行合并运算，一直到不能再进行合并运算为止，经合并运算后的结果称为最简化的 $\mathbf{route}_{\mathrm{opt}}(\mathrm{A,B})$，这个过程称为最简化 $\mathbf{route}_{\mathrm{req}}(\mathrm{A,B})$。

模运算 $|\bullet|$　对 $\mathbf{route}_{\mathrm{req}}(\mathrm{A,B})$ 中的 $\mathbf{servers}_i$，$|\mathbf{servers}_i|$ 则表示第 i 段路由链路上参与完成业务的服务实体个数。特别的，则当 $n=1$ 且 $|\mathbf{servers}_1|=1$ 时，表示 A，B 为同一服务实体；若 $\mathbf{route}_{\mathrm{req}}(\mathrm{A,B})$ 未经过合并运算，当 $n=1$ 且 $|\mathbf{servers}_1|=2$ 时，表示 A，B 为物理上的相邻实体。若 $\mathbf{route}_{\mathrm{req}}(\mathrm{A,B})$ 是最简化的，当 $n=1$ 且 $|\mathbf{servers}_1|=2$ 时，表示 A，B 为逻辑上的相邻实体，A，B 之间无过载实体。

定义 2　向量 $\mathbf{servers}_i = < s_{i1}, s_{i2}, \cdots, s_{im_i} > (s_{ij} \in S, j \leq m_i)$ 由完成业务请求的服务实体构成，表示为完成用户请求，s_{i1} 需 s_{i2} 协助，s_{i2} 需 s_{i3} 协助，依次类推，直到 $s_{i(m_i-1)}$ 需 s_{im_i} 协助为止。其中，所有以 A 开始的向量构成的集合 $\{\mathbf{servers}_i \mid \mathbf{servers}_i = < A, s_{i2}, s_{i3}, \cdots, s_{im_i} > \}$ 为链路的起始向量组；所有以 B 结尾的向量构成的集合 $\{\mathbf{servers}_i \mid \mathbf{servers}_i = < s_{i1}, s_{i2}, \cdots, s_{i(m_i-1)}, B > \}$ 为链路的结束向量组。

定义 3　指那些只出现在向量 $\mathbf{servers}_i$ 末尾的元素，它们不会再向其它服务实体发出协助请求，如：媒体处理器，GGSN，以及在控制层和业务层看来的资源和接纳控制系统(RACS)等，这些元素称为终结元素，终结元素是相对于业务请求和观察点而言的。

定义 4　按照上述表示方法和定义，若 $< s_x, s_y > \in \mathbf{servers}_i$ 成立，则说明服务实体 s_x 需要直接依赖服务实体 s_y 的参与才能完成业务请求，将这种因请求/服务而形成的直接依赖关系称为服务实体 s_x, s_y 间的相邻关系。为清楚地表达请求和服务关系，区分相邻关系为前向相邻关系和后向相邻

关系。为一个服务实体 s 提供服务的实体和实体 s 形成前向相邻关系；需要某一实体 s 提供服务的实体与实体 s 形成后向相邻关系。如在上例中，若 s_x,s_y 有相邻关系，则称 $<s_x,s_y>$ 为 s_x 的一个前向相邻关系，而是 s_y 的一个后向相邻关系。

某一服务实体对应于这两种相邻关系可能分别存在前向相邻资源和后向相邻资源。其定义分别如下：

定义 5 某一服务实体 s 的前向相邻资源集是指为完成用户请求，向其提供协助服务的服务实体集合 $\text{Fw\_neighbour}=\{s_{i(x+1)}\big|<s_{ix},s_{i(x+1)}>\in \mathbf{servers}_i \wedge (s_{ix} \underline{\underline{\text{ the same }}} s), i=1,\cdots,n\}$，其含义是指和该实体相邻的参与完成所有用户请求的所有下游服务实体。

定义 6 某一服务实体 s 的后向相邻资源集是指请求其参与完成业务的服务实体组成的集合 $\text{Bw\_neighbour}=\{s_{i(x-1)}\big|<s_{i(x-1)},s_{ix}>\in \mathbf{servers}_i \wedge (s_{ix} \underline{\underline{\text{ the same }}} s), i=1,\cdots,n\}$，其含义是指和该实体相邻的所有参与完成用户请求的上游服务实体。

从过载控制的角度来看，NGN 各功能实体之间的区别仅仅是具有不同的业务请求处理能力。基于当前核心网络带宽相对过剩的现状，做如下假设：认为核心网络带宽充足。NGN 总体体系结构表明，RACS 向控制层和业务层屏蔽接入网和网络互连网关，提供具有 QoS 质量保证的传输服务。

基于上述假设、相邻资源概念和运算以及 NGN 总体体系结构，针对 NGN 核心网络过载控制研究，进行如下抽象：RACS 等提供 IP 连接资源管理功能的实体、NGN 控制层和应用层不同服务实体具有平等的相邻关系，同时，将共享相同物理资源的实体合并一个服务实体。在此抽象基础上，不同资源根据信令关系和过载控制的要求形成某一服务实体 s 的前向和后向相邻资源。如 GPRS 网络中的 GGSN 节点可看作对应 P-CSCF 的一个前向相邻资源，为 P-CSCF 提供服务的 C-CSCF 可看作 P-CSCF 的前向相邻资源，而 P-CSCF 则是该 C-CSCF 的后向相邻资源。

3.3 基于相邻资源的分布式过载控制

基于策略的过载控制体系结构[15]由不同组件构成，它们之间基于反馈机制相互协作实施过载控制。首先过载检测和自适应限制组件检测服务点负载状况，由过载信息分发组件分发过载控制信息，最后过载控制执行组件实施过载控制，其间在分发信息时由策略决策点和控制点实施策略控制。这些组件的划分是逻辑上的，其实际物理部署策略与系统可能出现的过载场景密切相关。不同的组件部署方案对过载控制的作用有着重要影响。正是因为这种过载源和过载点受网络拓扑、网络用户行为等因素的动态变化而变化，所以，本文采用代理技术，基于相邻资源视图实现过载控制。对应于不同组件设计代理实体：过载检测代理 DA，过载控制策略执行点 PEP，过载控制决策点 PDP，更新控制信息代理 UA，过载执行代理 EA，内部接纳控制代理 AA。在 NGN 的每个服务实体上都要部署 DA，EA，AA，而在策略服务器上部署 PDA，而 PEA 与 UA 部署在分发实体上分发过载控制信息。对任一服务实体来说，在某一时刻，其上的代理并不一定全部工作。当其为过载点时，DA，AA 就会运行，然后查找其后向相邻资源 Bw_neighbour 集，由 UA 将过载控制信息分发给向相邻资源集中资源上部署的 EA，实施过载控制。各类代理实体之间遵守 GOCAP 协议，它们首先通过 OPEN/OPEN-ACK 建立对话，并可通过 STATUS/STATUS_ACK 查询服务器或客户端状态，最后用 RESTRICT/RESTRICT_NACK 分发过载控制信息，对请求实施过载控制。

经过前面的抽象以后，对于 A，B 两个服务实体之间路由链路上的任一服务实体 s 有其前向相邻资源和后向相邻资源。在某一服务实体出现过载后，除在该服务实体内部实现过载控制外，以该过载点为 B，根据过载控制条件选择流经 B 的业务请求流 $\mathbf{route}_{\text{req}}$(A,B) 进行合并运算，在最简化的 $\mathbf{route}_{\text{opt}}$(A,B) 中查找该过载资源的后向相邻资源，在后向相邻资源实施接纳控制，从而分担过载资源的拒绝服务处理任务，实现在保证响应时间的前提下，最大化过载资源的有效吞吐量。

由于经过合并运算∪，相比物理上的邻接点，后向相邻资源是更加接近网络边缘、离用户更近的服务实体。因此，网络对被拒绝业务请求的处理更少，网络资源会得到更加有效利用。由于过载控制条件的不同和业务请求流量的动态性，每一个服务实体的前向相邻资源和后向相邻资源在不同时刻也不同。这就需要动态地计算相邻资源并实现过载控制，在上述代理环境下实施过载控制的算法如下：

(1)DA 检测过载点，在过载点启动 AA，进行内部过载控制；

(2)以过载点为 B 点，对所有路由链路 $\mathbf{route}_{\text{req}}$(A,B) 进行合并运算，使其最简化；

(3)查找过载点的后向相邻资源；

(4)UA 根据负载状况请求 PDA、PEP 配合，确定限制级别并分发给其后向相邻资源；

(5)在这些后向相邻资源启动限制器 EA，进行外部过载控制；

(6)并根据负载状况重复上述过程。

其中，后向相邻资源的查找依赖于具体协议，如 SIP 协议中，可通过计算 SIP Invite 消息的 Record-Route 头域来完成。

3.4 算法分析

假定请求消息从请求点 A 经 n 个节点到服务点 B 途径时第 i 点提供的路由转发处理需要的开销为 c_i，而在任何一点提供的拒绝服务处理需要相同的开销 d，该服务只能在服务点 B 得到服务。下面从两个角度分析上述算法，第一，对某个业务请求的处理开销；第二，整个网络的处理能力。

首先，比较在“相邻节点”和“过载节点”实施过载控制对某个业务的处理开销。对被服务业务请求来说，由于“相邻节点”无法提供服务，所以只能由过载点提供服务。对被拒绝的业务请求来说，假设在第 m 点拒绝该业务请求，B 拒绝该业务请求，其开销为 $\sum_1^m c_i + d$ ， 其中 $m \le n$ 。显然，相对于在过载点拒绝业务请求，在“相邻节点”点需要更少的网络开销。

第二，在相邻节点实施过载控制也会提高整个网络的处理能力。网络中某个服务点过载是由于不同节点发出的大量请求超过了其服务能力所致。但对该过载节点来说，可视为从一个节点发来的请求。因此，网络可简化为如图 1 所示结构：

图 1

假定过载服务点 B 的服务处理能力为 S_B，若业务请求的拒绝和服务处理都在该节点，则整个网络的处理能力就是 S_B。若部分或全部拒绝请求能在 R 进行，路由服务点 R 提供的拒绝服务处理为 S_R，则整个网络的处理能力将是 S_B 和 S_R。这样，服务点可有更多的资源用于提供服务请求处理，从而提高了系统的有效吞吐量。

该算法带来的唯一开销是过载控制信息的分发。这虽然要求路由服务点 R 增加接收过载控制信息的能力，但是由于它不是网路的过载点，而且接收过载控制信息的开销也不大，因此对系统带来额外的负担。服务点虽然需要分发过载控制信息，但它无须为拒绝请求服务巨大的开销。同时，基于前面网络带宽不受限的假设，网络传输的开销可忽略不计。另外，当服务点分发自己还可接收的业务请求数量或速率时，控制点未收到控制信息时，就应主动减少转发的请求数量，这样可控制控制分发信息的频率，在服务点过载时减少过载信息的分发，更加有效的利用服务器的资源。

4　场景分析

这里以文献[4]给出的负载均衡或 SIP Fork 带来的过载控制场景为例说明在后向相邻资源实施过载控制的优势。该文献中的一个过载场景如图 2 所示：

图 2 中的 3 个功能服务器具有相同的功能，3 个代理服务器都具有负载均衡的功能或实现了 SIP fork 功能。3 个功能服务器都过载时，若在物理上的相邻节点实现外部控制，即在 P1 限制外部请求，则请求可能被 PA 转至 P2，继续发给功能服务器，功能服务器再次进行拒绝处理，功能服务器的资源没有得到有效利用。因为 P1 和 P2 所具有的负载均衡或 SIP Fork 功能，对业务请求者或上游服务实体来说，3 个功能服务器可抽象为一个功能实体 S_0。若在后向相邻资源实施过载控制，并以 S_0 为抽象后的 B 点，则

$$S = \{S_0, \mathrm{P1}, \mathrm{P2}, \mathrm{PA}\},$$
$$\mathbf{route}_{\mathrm{req}}(\mathrm{A,B}) = \{ < \mathrm{PA}, \mathrm{P1} >, < \mathrm{PA}, \mathrm{P2} >, < \mathrm{P1}, S_0 >, < \mathrm{P2}, S_0 > \},$$

对合并 $\mathbf{route}_{\mathrm{req}}(\mathrm{A,B})$ 进行合并运算得：$\mathbf{route}_{\mathrm{opt}}(\mathrm{A,B}) = \{< \mathrm{PA}, S_0 >\}$。

因此 S_0 的后向相邻资源为 PA，在 PA 点实施外部过载控制。避免了在 P1 返回不能提供服务后继续通过 P2 向功能服务器发送请求带来的不利影响。由此可见，基于最简化 $\mathbf{route}_{\mathrm{opt}}(\mathrm{A,B})$ 的请求限制点是更靠近用户侧的汇聚服务实体，这相对于文献[5]中提出的与过载点直接相连的源交换机方法，提高了网络资源的有效利用率。

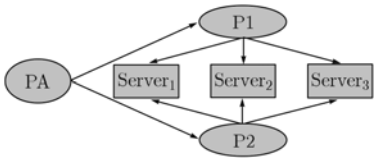

图 2 负载均衡或 SIP Fork 功能带来的负载放大影响

5　结束语

目前对于 NGN 网络过载控制的研究还是一个热点。从自然灾害到恐怖袭击等对人类造成巨大损害的事件来看，NGN 网络过载控制也是必须解决的问题。ETSI TISPAN 组织和 MSForum 论坛等都在加紧分析 NGN 网路过载控制的需求，讨论相关机制，并制定相关规范。本文提出“相邻资源”的概念，基于代理技术给出一种 NGN 过载控制的分布式过载控制，它基于最简化 $\mathbf{route}_{\mathrm{opt}}(\mathrm{A,B})$，在过载点的后向相邻资源实施外部过载控制，能有效解决负载均衡和 SIP Fork 带来的负载放大问题，相对于在过载点直接相连的源交换机实施过载控制，提高了网络资源的有效利用率。

参 考 文 献

[1] ETSI ES 282 001 V1.1.1. TISPAN NGN Functional Architecture Release 1[S]. 2005.

[2] British Telecommunications plc. MSF-TR-ARCH-007-FINAL, NGN Control Plane Overload and its Management [S]. 2006.

[3] ETSI DTR TISPAN-WI02026-NGN V0.0.6d. Next Generation Networks; Architecture for Control of Processing Overload[S]. 2006.

[4] Rosenberg J. Draft-Rosenberg-sipping-overload-reqs-01.txt, Requirements for Management of Overload in the Session Initiation Protocol[S]. 2006.

[5] NICC. ND1007:2006/04(PNO-ISC/SPEC/007 Issue 3.2), ISUP User Part (ISUP)[S]. 2006.

[6] Poikselka M, Mayer G, and Khartabil H, *et al*.. The IMS: IP Multimedia Concepts and Services in the Mobile Domain[M]. The Atrium, Southern Gate, Chichester: John Wiley & Sons

Ltd, 2004: 10–18.

[7] Williams P M and Whitehead M J. Realising effective intelligent network overload controls[A]. In: Telecommunications Performance Engineering[M]. London: The Institution of Electrical Engineers, 2004: 181–220.

[8] Jacobson V and Karels M J. Congestion avoidance and control[J]. *ACM Computer Communication Review* (Proceedings of the SigComm'88 Symposium in Stanford, CA, August), 1998, 18(4): 314–329.

[9] Stallings W著. 齐望东, 薛卫娟和傅麒麟等译. High-Speed Network TCP/IP and ATM Design Principal[M]. 高速网络——TCP/IP和ATM的设计原理. 北京: 电子工业出版社, 1998: 290–322

[10] Manfield D R, Millsteed G and Zukerman M. Congestion controls in SS7 signaling network[J]. *IEEE Communications Magazine*, 1993, 31(6): 50–57.

[11] Rumsewicz M P and Smith D E. A comparison of SS7 congestion control options during mass call-in situations[J]. *IEEE/ACM Transactions on Networking*, 1995, 3(1): 1–9.

[12] Floyd S and Fall K. Promoting the use of end-to-end congestion control in the Internet. *IEEE/ACM Transactions on Networking*, 1999, 7(4): 458–472.

[13] Pillai R R. A distributed overload control algorithm for delay-bounded call setup[J]. *IEEE/ACM Transactions on Networking*, 2001, 9(6): 780–789.

[14] Patel A, Prouskas K, and Barria J, *et al*. A computational economy for IN load control using a multi-agent system[J]. *Journal of Network and Systems Management*, 2000, 8(3): 397–417.

[15] 王开西, 杨放春. 下一代网络中基于策略控制的过载控制体系结构[J]. 计算机工程, 2008, 34(2): 待刊.
Wang Kai-xi, Yang Fang-chun, Study on the overload aontrol architecture based on the policy control in NGN. Computer Engineering. 2008, 34(2):

王开西： 男，1971 生，博士生，研究方向为通信软件、智能网、下一代网络.

杨放春： 男，1957 生，教授，博士生导师，主要研究方向为通信软件、智能网、下一代网络.