独立智能外设的性能建模与分析

杨孟辉 廖建新 沈奇威 张奇支
(北京邮电大学 网络与交换国家重点实验室 北京 100876)

摘 要 该文建立了独立智能外设的多任务类型封闭排队网络模型。对独立智能外设上开展的彩铃业务进行了分析，并使用近似平均值分析(MVA)算法求解出了系统的相关性能指标。通过分析这些性能指标，可以找到制约系统性能的瓶颈。最后，用 OPNET 进行了仿真，验证了分析的有效性。

关键词 智能网络，智能外设，排队网络， 平均值分析
中图分类号：TN915.5 **文献标识码**：A **文章编号**：1009-5896(2006)08-1422-07

Performance Modeling and Analysis of Independent Intelligent Peripheral

Yang Meng-hui Liao Jian-xin Shen Qi-wei Zhang Qi-zhi
(*State Key Laboratory of Networking and Switching, Beijing University of Posts and Telecommunications, Beijing* 100876, *China*)

Abstract Aiming at the analysis for the service performance of independent intelligent peripheral, this paper presents the performance model based on queueing network. With the analysis of the service features and the basic information flow of the CRBT (Color Ring Back Tone) service deployed on the independent intelligent peripheral, Some performance metrics are obtained through using the approximate Mean Value Analysis(MVA) algorithm. These metrics help to find the bottleneck of the system and improve the quality of service. Finally, OPNET is used to simulate the system, and verified the accuracy of this model.

Key words Performance analysis, Intelligent peripheral，Queue network, Mean value analysis

1 引言

智能外设(Intelligent peripheral)是智能网中的重要功能实体之一，它作为资源服务器为智能网提供专用资源功能(SRF)。智能外设有两种实现方式[1]：一种是将SRF与SSF/CCF(Service Switch Function/Call Control Function)功能实体映射到同一物理实体上，称为综合IP；另一种是将SRF映射到一个单独的物理实体上，称为独立IP。随着智能网业务种类的增加，业务要求的提高，综合IP越来越不能满足业务的快速提供需求，这就需要独立IP的支持。独立IP系统是一个集计算，交换，通信功能为一体大型通信系统[2]。它有丰富的资源，各类通信接口及交换模块。它拥有支持实时和并发控制的软硬件环境，可以生成，加载和执行SRF业务逻辑。

独立 IP 在中国电信，中国移动和中国联通的电信网络上得到了广泛的应用，丰富多彩的智能业务与独立 IP 的灵活多样功能是密不可分的。研究智能外设的文章较多，文献[1]研究了移动智能网独立 IP 的设计和实现。文献[2]研究了固定智能网独立 IP，并使用 $M/M/1$ 排队模型对独立 IP 的性能作了分析。文献[3]研究了固定智能网独立 IP 的设计和实现。文献[4]提出了一种用于智能外设的分布式负载控制算法。

对智能外设的性能分析主要是为了得到系统的总体性能与系统各组成部分性能之间的关系。系统的总体性能主要是智能外设的 Caps 能力，即每秒处理的呼叫数。制约系统 Caps 能力的因素可能很多。本文对北京邮电大学网络与交换国家重点实验室研制的移动智能网独立 IP 进行了性能建模和分析。尽管文献[2]使用 $M/M/1$ 排队模型研究了独立 IP 的 Caps 能力，但它假设控制节点(Control Node, CN)的处理能力就是整个系统的处理能力，并专门针对 CN 进行了建模和分析，并通过实验得到了 CN 的处理能力。实际上，作为一个大型通信系统，其中的每一个物理节点以及将这些物理节点联系的网络都可能成为系统性能的瓶颈。本文使用一个排队网络模型，对移动智能网独立 IP 中的各个物理节点以及通信网络进行了性能建模，给出了不同消息在不同物理节点的平均逗留时间，平均排队长度以及平均吞吐量等性能指标，根据这些性能指标可以定位系统性能的瓶颈。本文后续的内容是这样组织的：第 2 节对移动智能网独立 IP 的系统结构作了简单的描述。第 3 节提出了独立 IP 的性能分析模型，即排队网络模型。第 4 节通过实验验证了性能模型中的有效性。第 5 节对本文中尚待进一步解决的问题进行了说明，对全文进行了总结。

2004-11-25 收到，2005-04-18 改回
高等学校博士学科点专项科研基金（20030013006），国家移动通信产品研究开发专项基金：下一代移动智能网络系统的开发及应用和电子信息产业发展基金：移动通信增值服务平台及应用系统资助课题

2　系统结构

独立 IP 主要由控制节点，资源节点(Resource Nodes, RN)和信令节点(Signaling Nodes, SN)组成，各节点通过 100Mbyte 高速以太网相连，如图 1 所示。(1)控制节点采用高性能计算机服务器，工作在负荷分担方式。它在独立 IP 中起控制中心作用，负责资源管理，呼叫处理，业务逻辑执行等功能。(2)资源节点由高可靠性的工控机构成，它在独立 IP 中起资源驱动作用。它接受控制节点的指令，通过调用各种资源的 API 来控制资源，完成系统所需的资源功能，如语音播放，DTMF 信号接受，播送信号音，文-语转换等。(3)信令节点是独立 IP 与电话网的接口，包括信令接口和话路接口两部分。独立 IP 通过电路连接提供服务，为充分利用资源，提高系统的稳健性，在信令节点引入了电路交换矩阵，在控制节点的控制下，灵活地选择特定资源提供服务。该节点的信令处理部分负责处理 TCAP 消息和 ISUP(Integrated Services digital network User Part)消息，工作在负荷分担方式。在实际应用中，如果需要支持高的话务负荷，可以增加信令节点，控制节点和资源节点中处理机的个数。对于每个节点有多个处理机的情况，我们可以有如下两种方式进行分析：我们使用一个先来先服务(First Come First Serve，FCFS)排队系统来表示一个处理机，第 1 种方式是将每个节点中多个处理机看成为一个处理能力更大的单服务窗排队系统，处理机的多少反映在排队系统的处理能力上。第 2 种方式是将每个节点中多处理机映射为一个多服务窗的排队系统，这样，每个节点用一个多服务窗排队系统表示。在本文中，我们使用第 2 种方式来对独立 IP 进行性能建模和分析。

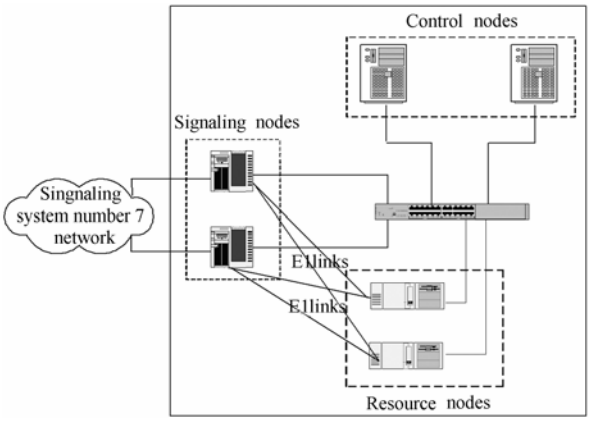

图 1　独立 IP 的物理结构

Fig.1 Independent intelligent peripheral physical architecture

3　系统性能分析模型

独立IP是一个复杂的通信系统，影响系统性能的因素很多，如信令节点处理信令的能力，中继线的数量，资源的多少，控制节点的处理能力以及通信网络速度等。根据图 1，我们建立独立IP的性能分析模型，使用这个模型试图找到制约系统性能的瓶颈，提高系统的服务质量。对排队网络的研究已经有了很多的成果[5-12]。图 2 是根据图 1 建立的独立IP的排队网络模型。这是一个多任务类型的封闭排队网络，有 5 个节点，每个节点是一个单服务窗或者多服务窗排队系统。节点 1 代表独立IP的外部消息输入源，它可以是智能网中的业务控制点(Service Control Point，SCP)或业务交换点(Service Switch Point，SSP)，在这个节点的时间花费是SCP的思考时间，即SCP收到一个应答，直到SCP本身或SCP指示SSP又发送一个新的消息需要的时间。节点 2 表示独立IP的信令节点。节点 3 代表独立IP中的通信网络。节点 4 表示独立IP的控制节点。节点 5 代表独立IP的资源节点。由于独立IP处理的消息既有CAP(CAMEL Application Part)消息，又有ISUP消息，因此，我们需要使用能处理多任务类型的封闭排队网络模型的MVA(Mean Value Analysis)算法[5,6,12-14]。图 2 所示的排队网络为一个FCFS节点的网络，每个节点为一个单服务窗或者多服务窗排队系统，假设排队网络中有 N 个节点，R 类任务，μ_{ir} 表示 r 类任务在节点 i 的服务率，$\bar{K}_{ir}$ 为 r 类任务在节点 i 的平均个数，$\bar{T}_{ir}$ 为 r 类任务在节点 i 的平均响应时间，λ_r 为 r 类任务的吞吐量，e_{ir} 为 r 类任务对 i 节点的访问率，m_i 为节点 i 的服务窗个数。设网络中共有 $k-1$ 个任务，假设节点 i 有 $j-1$ 个任务，考察一个到达这个节点的任务，这个事件发生的概率假设为 $\pi_i(j-1|k-1)$。以系统中的任务个数为状态量。下面给出多任务类型的封闭排队网络的MVA算法[5,13,14]：

步骤 1　初始化：$\bar{K}_{ir}(0,0,\cdots,0)=0$，$\pi_i(0|0)=1$，$\pi_i(j|0)=0$，$\lambda_{ik}=0$；

步骤 2　开始迭代：

(a)按照式(1)计算任务 r 在节点 i 的平均响应时间或逗留时间：

$$\bar{T}_{ir}(k)=\begin{cases}\dfrac{1}{\mu_{ir}}\left[1+\sum_{s=1}^{R}\bar{K}_{is}(k-1)\right], & m_i=1\\ \dfrac{1}{\mu_{ir}}\left[1+\sum_{s=1}^{R}\bar{K}_{is}(k-1)+\sum_{j=0}^{m_i-2}(m_i-j-1)\pi_i(j|k-1)\right], & m_i>1\end{cases} \quad (1)$$

(b)按照式(2)计算系统处于状态 k 时，节点 i 有 j 个任务的概率：

$$\pi_i(j\mid k)=\frac{1}{j}\left[\sum_{r=1}^{R}\frac{e_{ir}}{\mu_{ir}}\cdot\lambda_r(k)\cdot\pi_i(j-1\mid k-1)\right] \quad (2)$$

对于 $j=0$，按照式(3)计算 $\pi_i(0|k)$：

$$\pi_i(0|k)=1-\frac{1}{m_i}\left[\sum_{r=1}^{R}\frac{e_{ir}}{\mu_{ir}}\lambda_r(k)+\sum_{j=1}^{m_i-1}(m_i-j)\pi_i(j|k)\right] \quad (3)$$

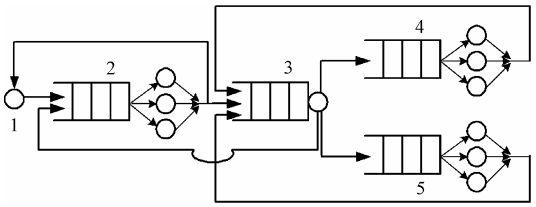

图 2 独立 IP 的排队网络模型

Fig.2 Independent intelligent peripheral queueing network model

(c)按照式(4)计算 r 类任务的吞吐量：

$$\lambda_r(k)=\frac{k_r}{\sum_{i=1}^{N} e_{ir}\cdot \bar{T}_{ir}(k)} \tag{4}$$

(d)按照式(5)计算 r 类任务在节点 i 的平均个数：

$$\bar{K}_{ir}(k)=\lambda_r(k)\cdot \bar{T}_{ir}(k)\cdot e_{ir} \tag{5}$$

步骤 3 回到步骤 2 继续迭代。迭代过程直到队列长度足够精确。

为了开始进行数值迭代，首先假设队列的长度值，然后逐步调整到与实际精确值更接近的值。一个好的方法是首先假设把每一类的任务平均分配到该类任务所访问的所有队列上。即如果一个类 r 只访问 K 个队列中 K_r 个，那么初始值即为 N_r/K_r，其中 N_r 为 r 类任务的总个数，这就是Schweitzer提出的一个近似MVA算法[5,12]。该算法基于如下的假设：在每一个队列中的第 r 任务的数目的增长是与整个排队网络中的第 r 类任务的数目成正比的。

为了用近似 MVA 算法求解图 2 所示的排队网络，首先需要计算 r 类任务在节点 i 的服务时间需求 D_{ir}，由定义有 $D_{ir}=1/\mu_{ir}$。这里考虑的任务主要是独立 IP 处理的 CAP 消息和 ISUP 消息，其中 CAP 消息主要考虑 PA(Play Announcement)，P&C(Prompt and Collect user information)，SRR(Specialized Resource Report)；ISUP 消息主要考虑 IAM(Initial Address Message)，ACM(Address Complete Message)，ANM(ANswer Message)，REL(RELease)，RLC(ReLease Complete)。独立 IP 的一个服务过程包括承载连接的建立、对 PA/P&C/CANCEL 消息的处理、承载连接的拆除 3 个连续的子过程, 其中, 在第 2 个子过程中可以有多个 PA,P&C,CANCEL 消息的交互。图 3 为独立 IP 服务过程的 3 个子过程，它们形成了一个完成的呼叫流程，当然该流程与正统的 IP 服务有一定的差距。

为了方便，我们对上述的 CAP 消息和 ISUP 消息按照任务类进行如下编号：IAM 为第 1 类任务，该消息依次进入信令节点，交换网络，控制节点，交换网络，资源节点，然后返回应答消息 ACM。尽管该消息在不同节点已经变为不同的内部消息进行传输和处理，但我们都认为是一类消息。该消息在独立 IP 的服务过程完成了呼叫中承载连接的建立。

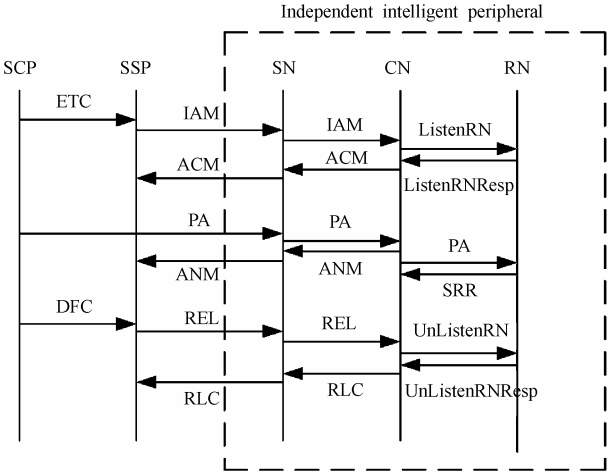

图 3 呼叫流程

Fig.3 Call flows

PA 为第 2 类任务，ANM 为该任务的应答消息。该任务在独立 IP 的服务过程完成了呼叫中消息的处理。该服务过程可能重复若干次，根据不同业务需求而定。在本文中使用的彩铃业务只使用一次消息处理过程。

REL 为第 3 类任务，RLC 为该任务的应答消息。该任务在独立 IP 的服务过程完成了呼叫中承载连接的拆除。

节点 1 为独立 IP 的外部节点，它作为消息输入源向节点 2(信令节点)发送消息。这些消息在节点 1 的服务时间需求按照下面的公式计算：$D_{11}=1/\lambda$， $D_{12}=1/\lambda$， $D_{13}=1/\lambda$；其中 λ 为呼叫到达率。

对于节点 2，它处理的任务为节点 1 发出的任务和节点 3 返回的任务。这些任务包括如下消息：(1)节点 1 发出的消息 IAM，PA，REL；(2)节点 3 返回的消息 ACM，ANM，RLC。我们按照下面的公式计算这些消息的服务时间。$D_{21}=2\cdot i\text{sup\_time}$，$D_{22}=4\cdot i\text{sup\_time}$，$D_{23}=2\cdot i\text{sup\_time}$，其中 $i\text{sup\_time}$ 为节点处理一个 ISUP 消息的平均时间。尽管各类消息长度不一致，但编解码操作对消息长度是不敏感的。因此对这些消息而言，它们的平均处理时间基本相同；由于 PA 包括对话和成分两个消息，因此需要 2 个 $i\text{sup\_time}$；$i\text{sup\_time}$ 一般都折算为 TPS(Transaction Per Second)来进行计算。

对于节点 4，它处理的任务为节点 2 发出的任务经过节点 3 后到达该节点，以及节点 5 返回的任务经过节点 3 后到达该节点进行处理。这些任务包括如下消息：(1)节点 2 发出的消息 IAM，PA，REL；(2)节点 5 返回的消息 ListenRN Response，SRR， UnListenRN Response。我们按照下面的公式计算各类任务的服务时间需求。$D_{41}=$ iam_time+acm_time, $D_{42}=$ pa_process_time+anm_time, $D_{43}=$ rel_process_time+rlc_

process_time 。在控制节点(节点 4)各类消息服务时间主要包括这些消息处理数据库操作，搭建时隙和拆除时隙所需要的时间。同样，不同操作的时间也需要折算成 TPS 来计算。

对于节点 5，它处理的任务为节点 4 发出的任务经过节点 3 后到达该节点。控制节点(节点 4)收到 IAM 消息后，向资源节点(节点 5)发送 ListenRN 消息，请求资源节点为该消息规定的呼叫实例占用相应的特殊资源。ListenRNResp 为资源节点向控制节点返回的应答消息，表明资源占用成功。控制节点(节点 4)收到 REL 消息后，向资源节点(节点 5)发送 UnListenRN 消息，请求资源节点释放该消息规定的呼叫实例占用的特殊资源。UnListenRNResp 为资源节点向控制节点返回的应答消息，表明资源释放成功。控制节点(节点 4)收到 PA 消息后，向资源节点(节点 5)发送 PA 消息，资源节点返回 SRR 给控制节点作为应答。我们按照下面的公式计算各类任务的服务时间需求。 $D_{51}=\text{alloc\_resource\_time}$, $D_{52}=\text{play\_tone\_time}$, $D_{53}=\text{release\ \_resource\_time}$ 。消息在资源节点的消耗主要是 CPU 和磁盘操作；对于放音等，由于是由单独的硬件芯片完成，这里暂不考虑其开销。

对于节点 3，它是通信网络，它处理结点 2，4，5 的所有发出消息和应答消息。由于独立 IP 内部各节点之间的消息通信协议采用 TCP/IP 协议，所以在计算节点 3 的服务时间需求时需要考虑 TCP 协议，以及它的底层 IP，还有其上层的 LAN Link-level 协议的附加的头部。我们用 NetworkTime(m) 代表通信网络对一个长为 m byte 消息的服务时间： $\text{NetworkTime}(m)=\dfrac{8[m+\text{Overhead}(m)]}{10^6\cdot\text{LANBandwidth}}$ ，其中 Ovrehead(m) 为附加的头部， LANBandwidth 为网络带宽，单位为 Mbps。发送一个长为 m byte 消息所需要的 TCP 和 IP 头为：$\text{Ovrehead}(m)=\text{TCPOvhd}+\text{NDatagrams}(m)\cdot(\text{IPOvhd}+\text{FrameOvhd})$ ，其中 TCPOvhd 为 TCP 协议的头部，大小为 20byte，IPOvhd 为 IP 协议，大小为 20byte，NDatagrams(m) 为数据报的数目，FrameOvhd 为 LAN 的链路层的帧的头部，大小为 18byte。发送一个长为 m 字节消息所需要的数据报的数目为 $\text{NDatagrams}(m)=\dfrac{m+\text{TCPovhd}}{\text{MaxPDU}-\text{IPOvhd}}$ ，其中 MaxPDU 为 LAN 在网络层协议最大的 PDU 大小，一般为 65535byte。

按照上面的公式计算各类任务的服务时间需求:

$D_{31}=\text{NetworkTime}(\text{Length}(\text{IAM+ACM+ListenRN+ListenRNResp}))$

$D_{32}=\text{NetworkTime}(\text{Length}(\text{PA+PA+SRR+ANM}))$,

$D_{33}=\text{NetworkTime}(\text{Length}(\text{REL+RLC+UnListenRN+UnListenRNResp}))$

此外，我们还需要给出每一类任务的数量和误差控制参数。

4　仿真实验和结果分析

使用第 3 节中提出的排队网络模型和近似平均值分析算法来分析独立 IP 上开展的彩铃业务。我们在实验室一台配置较低的 PC SERVER(HP DL380， Linux 操作系统，1 个主频为 667MHz 的 CPU，2Gbyte 内存，数据库为 Informix 7.31)上运行 CN，RN 为 CPCI 工控机，配置有 32×20 个资源。通信网络为 100Mbyte 以太网。计算出相应消息的长度分别为(单位: bytes): Length(IAM)=128, Length(ACM)=44, Length(ANM)=44, Length(RLC)=44, Length(REL)=44, Length(PA)=116, Length(SRR)=60, Length(ListenRN)=60, Length(ListenRNRep)=60, Length(UnListenRN)=60, Length(UnListenRNRep)=60。这些消息在系统中都被编码成二进制码流，消息长度为二进制码流的长度。操作需要的时间与硬件环境相关，表 1 为根据上面配置得到的各类任务在不同节点的服务时间需求。

我们使用OPNET对独立IP的排队网络模型进行了仿真。使用多个消息产生器，产生图 3 所示的呼叫流程，来模拟多个呼叫。使用多个队列分别模拟信令节点，通信节点，控制节点，资源节点，其中通信网络使用单服务窗队列，信令节点，控制节点和资源节点使用多服务窗队列；节点中每类消息的服务时间设置如表 1 所示。仿真结果与模型的理论分析结果对比如下：(令误差控制参数 $\theta=10^{-6}$。)

表 1　服务时间需求(秒)

Tab.1 Service time requirement (s)

	第 1 类任务	第 2 类任务	第 3 类任务
信令节点	0.0006	0.0012	0.0006
通信网络	0.000029	0.000033	0.000022
控制节点	0.104764	0.169862	0.002533
资源节点	0.00738	0.15965	0.00564

4.1　Case 1

在该 Case 中，独立 IP 同时接受 10 个呼叫的处理。使用本文提出的近似 MVA 算法得到的分析结果如表 2 所示；OPNET 的仿真结果如表 3 所示。所有节点中服务窗个数均为 1。图 4 为各类消息在网络中平均逗留时间，其中分析结果中的平均逗留时间为该类消息在所有节点中的平均逗留时间的总和。图 5 为各类消息在各个节点中的平均排队长度。图 4，图 5 表明，分析结果与仿真结果基本一致。同时从图 5 可以看出，与其它节点相比，控制节点中消息出现了较多的排队，因此，可以看出，控制节点是性能瓶颈点。

表 2　同时处理 10 个呼叫的近似 MVA 分析结果

Tab.2 Approximate MVA analysis results with 10 calls processed simultaneously

	平均逗留时间(消息在各节点的平均逗留时间) (s)	平均队长(消息个数)

	第 1 类消息	第 2 类消息	第 3 类消息	
信令节点	0.000616	0.001281	0.000637	0.061524
通信网络	0.000029	0.000033	0.000022	0.002135
控制节点	2.484517	4.060492	0.061026	23.092264
资源节点	0.051348	1.080494	0.037282	5.610357
吞吐量: 104.467956 (消息/秒)				

表 3 同时处理 10 个呼叫的仿真结果

Tab.3 Simulation results with 10 calls processed simultaneously

	平均逗留时间(消息在网络中平均逗留时间) (s)			平均队长(消息个数)
	第 1 类消息	第 2 类消息	第 3 类消息	
信令节点	3.521126	5.29577	0.236619	0.010
通信网络				0.0002
控制节点				28.75
资源节点				8.12
吞吐量: 90(消息/秒)				

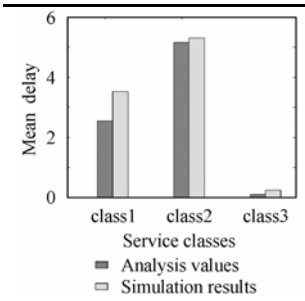

图 4 消息的平均时延

Fig.4 Messages mean delay

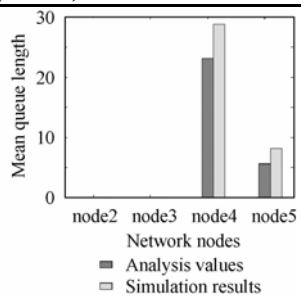

图 5 消息的平均排队长度

Fig.5 Messages mean queue length

4.2 Case 2

在该 Case 中，独立 IP 同时接受 20 个呼叫的处理。使用本文提出的近似 MVA 算法得到的分析结果如表 4 所示；OPNET 的仿真结果如表 5 所示。图 6 为各类消息在网络中平均逗留时间，其中分析结果中的平均逗留时间为该类消息在所有节点中的平均逗留时间的总和。图 7 为各类消息在各个节点中的平均排队长度。图 6，图 7 表明，分析结果与仿真结果基本一致，同时可以看出，消息在控制节点中的平均逗留时间增加，排队现象加剧，但系统的吞吐量几乎没有什么变化。可见，控制节点就是限制系统性能提高的瓶颈点。

表 4 同时处理 20 个呼叫的近似 MVA 分析结果

Tab.4 Approximate MVA analysis results with 20 calls processed simultaneously

	平均逗留时间(消息在各节点的平均逗留时间) (s)			平均队长(消息个数)
	第 1 类消息	第 2 类消息	第 3 类消息	
信令节点	0.000643	0.001287	0.000641	0.068857
通信网络	0.000029	0.000033	0.000022	0.002372
控制节点	5.179403	8.424459	0.126046	48.761554
资源节点	0.082705	1.764011	0.061476	9.899976
吞吐量: 110.626601(消息/秒)				

表 5 同时处理 20 个呼叫的仿真结果

Tab.5 Simulation results with 20 calls processed simultaneously

	平均逗留时间(消息在网络中平均逗留时间) (s)			平均队长(消息个数)
	第 1 类消息	第 2 类消息	第 3 类消息	
信令节点	5.29577	9.7887	0.38873	0.015
通信网络				0.0002062
控制节点				52.5
资源节点				15.0
吞吐量:90(消息/秒)				

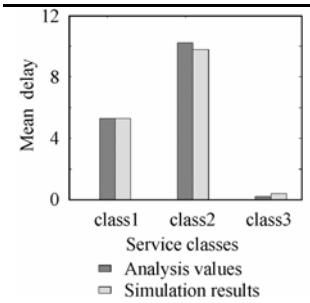

图 6 消息的平均时延

Fig.6 Messages mean delay

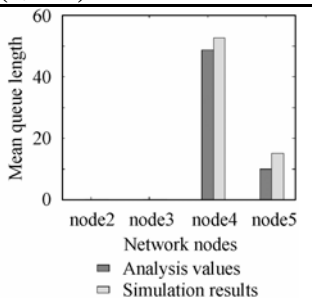

图 7 消息的平均排队长度

Fig.7 Messages mean queue length

4.3 Case 3

为了提高系统的性能，将控制节点的服务窗设为 2 个。使用本文提出的近似 MVA 算法得到的分析结果如表 6 所示；OPNET 的仿真结果如表 7 所示。图 8 为各类消息在网络中平均逗留时间，其中分析结果中的平均逗留时间为该类消息在所有节点中的平均逗留时间的总和。图 9 为各类消息在各个节点中的平均排队长度。图 8，图 9 表明，分析结果与仿真结果基本一致，同时可以看出，消息在控制节点中排队减少，消息时延得到明显改善；资源节点出现较多的消息排队，消息时延明显增加。此时，资源节点成为了瓶颈点。

表 6 同时处理 20 个呼叫，且控制节点的服务窗个数为 2 的近似 MVA 分析结果

Tab.6 Approximate MVA analysis results with 20 calls processed simultaneously and 2 servers in control node

	平均逗留时间(消息在各节点的平均逗留时间) (s)			平均队长(消息个数)
	第 1 类消息	第 2 类消息	第 3 类消息	
信令节点	0.000639	0.001280	0.000638	0.065968
通信网络	0.000029	0.000033	0.000022	0.002328
控制节点	2.405915	2.240494	0.034869	24.85968
资源节点	0.257112	5.475543	0.192654	33.15864
吞吐量: 112．267418 (消息/秒)				

表 7 同时处理 20 个呼叫，且控制节点的服务窗个数为 2 的近似仿真结果

Tab.7 Simulation results with 20 calls processed simultaneously and 2 servers in control node

	平均逗留时间(消息在网络中平均逗留时间) (s)			平均队长(消息个数)
	第 1 类消息	第 2 类消息	第 3 类消息	
信令节点	3.9577	7.475	0.304	0.025
通信网络				0.000337
控制节点				27.5

资源节点				30.0
吞吐量: 90(消息/秒)				

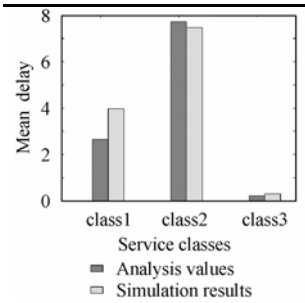

图 8 消息的平均时延

Fig.8 Messages mean delay

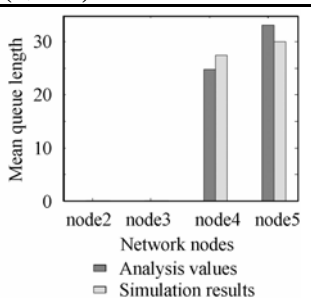

图 9　消息的平均排队长度

Fig.9 Messages mean queue length

4.4 Case 4

为了进一步提高系统性能，将资源节点的服务窗个数也增加为 2。使用本文提出的近似 MVA 算法得到的分析结果如表 8 所示；OPNET 的仿真结果如表 9 所示。图 10 为各类消息在网络中平均逗留时间，其中分析结果中的平均逗留时间为该类消息在所有节点中的平均逗留时间的总和。图 11 为各类消息在各个节点中的平均排队长度。图 10，图 11 表明，分析结果与仿真结果基本一致，同时可以看出，消息在资源节点中排队减少，消息时延得到明显改善，系统的吞吐量增加为原来的 2 倍，系统的服务性能得到提高。

表 8　同时处理 20 个呼叫，且控制节点和资源节点的服务窗个数均为 2 的近似 MVA 分析结果

Tab.8 Approximate MVA analysis results with 20 calls processed simultaneously and 2 servers in control node and resource node

	平均逗留时间(消息在各节点的平均逗留时间) (s)			平均队长(消息个数)
	第 1 类消息	第 2 类消息	第 3 类消息	
信令节点	0.000689	0.001373	0.000632	0.149012
通信网络	0.000029	0.000033	0.000022	0.004831
控制节点	2.574993	4.185757	0.061406	47.15798
资源节点	0.044426	1.027479	0.033021	10.97443
吞吐量: 214.749124　(消息/秒)				

表 9　同时处理 20 个呼叫，且控制节点和资源节点的服务窗个数均为 2 的仿真结果

Tab.9 Simulation results with 20 calls processed simultaneously and 2 servers in control node and resource node

	平均逗留时间(消息在网络中平均逗留时间) (s)			平均队长(消息个数)
	第 1 类消息	第 2 类消息	第 3 类消息	
信令节点	4.0563	4.47	0.315	0.11
通信网络				0.000425
控制节点				55.0
资源节点				15.625
吞吐量:180(消息/秒)				

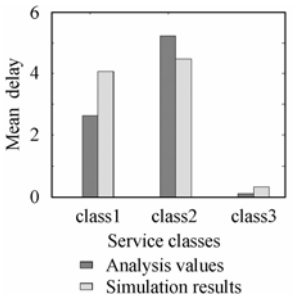

图 10 消息的平均时延

Fig.10 Messages mean delay

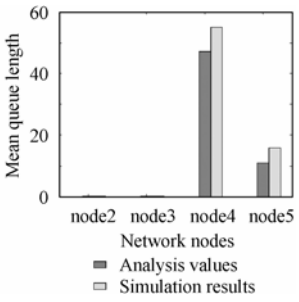

图 11 消息的平均排队长度

Fig.11 Messages mean queue length

从上面的分析结果可以看出，增加影响系统性能的节点的服务窗个数，可以提高系统的服务质量，包括降低消息时延和提高系统的吞吐量。根据这些分析结果，可以很容易知道系统性能瓶颈所在。消息的平均时延越大，平均排队长度越长，表明该节点为影响系统性能的节点。从上面的分析结果可以看出，通过增加瓶颈所在节点的服务窗个数，系统的性能，如吞吐量，平均时延得到了明显改善。

在 4 个 Cases 中，性能模型的理论分析结果与用 OPNET 的仿真结果均表明，分析结果与仿真结果基本吻合。同时我们也对两种结果之间存在的误差进行了定量分析，分析表明：近似 MVA 算法得到的平均时延与使用 OPNET 仿真得到的平均时延之间的平均相对误差为 55.17%；使用近似 MVA 算法得到的平均排队长度与使用 OPNET 仿真得到的平均排队长度之间的平均相对误差为 58.03%；使用近似 MVA 算法得到的吞吐量与使用 OPNET 仿真得到的吞吐量之间的平均相对误差为 17.12%。

上面的分析结果还表明，本文分析的独立 IP 系统，在运行彩铃业务时，影响系统性能的节点主要是控制节点和资源节点。通信网络和信令节点的处理能力还很宽裕。

5　结束语

本文使用排队网络理论分析了独立IP系统中每个节点处理消息的情况，包括不同消息在不同物理节点的平均逗留时间，平均排队长度以及平均吞吐量。本文求解排队网络使用的近似MVA算法是一个较快的算法，但在分析独立IP系统时，我们也发现对于存在多服务器的节点以及有多类任务的网络，该算法存在一些缺点，如稳定性，累计误差等。其中稳定性问题尤其突出，在很多情况下算法都无法收敛到给定的误差控制范围。对于累计误差，主要表现在求系统处于状态 k 时，节点 i 有 j 个任务的概率 $\pi_i(j|k)$ 当 $j \geq 1$ 时有时出现大于 1 的现象。这种现象出现在迭代开始时期，主要是由于给定的参数不合理导致的。一般情况下，该算法有小于 5% 的误差，但在某些情况下也可能有 20%的误差[6]。同时，仿真本身也可能产生较大误差。因此，该算法大大减少算法的计算量而较快获得计算结果，是以牺牲精度为代价的。但在网络较大，节点服务窗较多，同时任务种类也较多的情况下，能较快地得到计算结果往往非常重要，因为精确算法的计算

量与网络中节点的个数，每个节点的服务窗个数，以及任务种类的多少都存在指数关系。本文的分析表明，近似MVA算法与仿真结果之间存在的误差较大。误差可能由算法本身产生，也可能由仿真实验产生，或同时由算法本身和仿真试验产生。如何提高该算法的稳定性，如何抑制算法实现中概率大于 1 的现象的出现，以及量化该算法与精确MVA算法之间的误差等是作者下一步解决的问题。

参 考 文 献

[1] 朱晓民, 黄晖, 廖建新, 沈奇威. 移动智能网独立智能外设的设计与实现[J]. 北京邮电大学学报, 2003, 26(4) : 75−79.

[2] 杨飞, 李晓峰, 詹舒波, 陈俊亮.一种智能外设的结构和性能分析[J]. 北京邮电大学学报, 2000, 23(2): 52−56.

[3] Li Xiaofeng, Zhan Shubo, Li Qinag. Design and implementation of independent intelligent peripheral in CIN system [J]. APCC/OECC '99 , Fifth Asia-Pacific Conference on Communications , and Fourth Optoelectronics and Communications Conference, 1999, vol.2: 1264−1268.

[4] Kihl M, Rumsewicz M P. On overload control of intelligent peripherals in intelligent networks[J].Global Telecommunications Conference, GLOBECOM '96，Communications: The Key to Global Prosperity, 1996，vol.3: 1539−1543 .

[5] 李圣军, Lackman Robert. 基于排队网络的容量分析与模拟[J]. 计算机应用研究, 2002, (4): 99−104.

[6] 林闯. 计算机网络和计算机系统的性能分析[M]. 北京: 清华大学出版社, 2001: 124−125.

[7] Iakovos Venieris, Heinrich Hussmann. Intelligent Broadband Networks[M]. New York, John Wiley & Son, Inc., 1999, Chapter 4.

[8] Onyuksel IH, Irani KB. Markovian queueing network models for performance analysis of a single-bus multiprocessor system[J]. *IEEE Transactions on Computers*, 1990，39(7): 975−980.

[9] Bertsimas D, Gamarnik D, Tsitsiklis J. Performance of multiclass Markovian queueing networks[J]. Proceedings of the 39th IEEE Conference on Decision and Control, 2000，vol.1: 534−539 .

[10] Sauer C, MacNair E, Kurose J. Queueing network simulations of computer communication[J]. *IEEE Journal on Selected Areas in Communications*, 1984，2(1): 203–220.

[11] Cheng-Shang Chang. Stability, queue length, and delay of deterministic and stochastic queueing networks[J]. *IEEE Transactions on Automatic Control*, 1994，39(5): 913–931.

[12] Cremonesi P, Schweitzer PJ, Serazzi G. A unifying framework for the approximate solution of closed multiclass queuing networks[J]. *IEEE Transactions on Computers*, 2002, 51(12): 1423–1434.

[13] Conway A E, de Souza e Silva E. Lavenberg S S. Mean value analysis by chain of product form queueing networks[J]. *IEEE Transactions on Computers*, 1989, 38(3): 432–442.

[14] Akyildiz. IF. Mean value analysis for blocking queueing networks[J]. *IEEE Transactions on Software Engineering*, 1988, 14(4): 418–428.

杨孟辉： 男，1970 年生，博士生，研究方向为智能网下一代网络.

廖建新： 男，1965 年生，教授，博士生导师，研究方向为智能网、下一代网络.

沈奇威： 男，1976 年生，博士生，研究方向为智能网、下一代网络.

张奇支： 男，1976 年生，博士生，研究方向为智能网、下一代网络.