一种新的软实时应用的接纳允许控制和 QoS 控制算法*)

A New Admission Control and QoS management Algorithm

陈庶樵　邬江兴

(国家数字交换系统工程技术研究中心　郑州450002)

Abstract　With the development of computer system and network, soft real-time application becomes more and more important. In this paper, the authors propose a new admission control and QoS management algorithm, which is called Quixote. This algorithm fixes on system benefit function and QoS. Extensive simulation proves that Quixote is better than Worst case algorithm.

Keywords　Soft real-time, Admission control, QoS

1. 引言

多媒体应用及 Internet 的各种新业务对包括操作系统和网络系统的支撑环境的要求与传统意义上的各类实时业务有着显著的不同，这种不同之处往往表现在衡量支撑环境服务质量 QoS(Quality of Service)的标准上。传统意义上的实时业务，现在一般称为硬实时业务，其衡量支撑环境 QoS 的标准一般单纯依赖于执行时限 deadline 的丢失率上；而多媒体应用和一些新的 Internet 应用，现在一般称为软实时业务，其衡量支撑环境 QoS 的标准并不单纯依赖于 deadline 丢失率的统计数字上，在通常意义上讲，丢失 deadline 并不会造成这类应用的执行失败以及系统的崩溃，而其它一些指标，如系统的接纳允许率、QoS 的稳定性和整个系统的综合利用率则变得重要起来。本文所提出的 Quixote 算法，正是一种新的软实时应用的接纳允许控制和 QoS 控制方法。

2. 软实时业务支撑系统基本模型的建立

一般来说，当一个要求一定资源的应用到达系统时，系统根据所能提供的资源总量来决定是否接纳该应用，这种算法被称为接纳允许控制算法。当然，在某些特定的场合下，系统还要衡量诸如安全一类的因素，但是这些因素不在本文考虑的范畴之内。为了简化系统建模的复杂性，本文假设系统所考虑的资源只有一种，比如当影响软实时应用 QoS 指标最为关键的因素是该应用的 CPU 占用率时，接纳允许控制算法就应当以系统 CPU 的空余作为根据；同时，操作系统的调度算法对软实时应用 QoS 指标所起的作用也最大。

2.1　系统资源分类

系统资源可以分为以下三类：①R_{total}：系统的资源总量；②$R_{available}$：系统空余资源数量；③$R_{reserved}$：系统已分配资源数量。明显地，

$$R_{total}=R_{available}+R_{reserved}$$

另外，一个软实时应用所要求的资源表示为 $R_{required}$，能够满足最低要求的资源表示为 R_{endure}，系统实际为该业务分配的资源表示为 $R_{assigned}$。明显地，

$$R_{endure}\leqslant R_{assigned}\leqslant R_{required}$$

2.2　软实时应用表示方法

软实时应用与硬实时应用不同之处表现在许多方面，其中最为显著的是软实时应用在系统资源不满足其要求时也可以在一定的 QoS 程度上执行，而不会导致失败以及系统的崩溃。因此，软实时应用可以有多个 QoS 执行水平。

一个软实时应用可以用一个三元组表示：$\{App_{id}, QoS_{level}, R_{assigned}\}$。其中，$App_{id}$表示应用的标识，$QoS_{level}$表示应用执行的 QoS 水平，$R_{assigned}$表示系统为该应用所分配的资源。对于每一个软实时应用来说，QoS_{level}由 $R_{assigned}$决定，可以由以下函数表示：

$$QoS_{level}=Quality(R_{assigned})$$

明显地，$QoS_{level}\leqslant Quality(R_{required})$，并且有，

$$QoS_{level,i}<QoS_{level,j}\Rightarrow R_{assigned,i}<R_{assigned,j}$$

2.3　支撑系统的资源分配问题分析

如果从整个软实时应用系统的角度来看，整个系统的最佳利用率取决于系统中的每一个软实时应用的 QoS 执行水平。如果假设系统中存在 N 个软实时应用，则可得到系统的获益方程：

$$System_{benefit}=\sum_{id=1}^{N}Benefit(QoS_{id.level})$$

其中，$Benefit(App_{id}, QoS_{level})$是软实时应用的获益函数。为简化接纳允许算法和调度算法的复杂度，获益函数一般定义为 QoS_{level}的线性函数。

为了追求整个系统的最佳利用率，即使 $System_{benefit}$达到最大，系统资源的分配算法可以归纳为以下的规划问题：

目标函数　$Max(System_{benefit})$

$$\text{约束条件}\begin{cases}System_{benefit}=\sum_{id=1}^{N}Benefit(QoS_{id.level})\\ QoS_{id.level}=Quality(R_{id.assigned})\\ R_{total}\geqslant\sum_{id=1}^{N}R_{id.assigned}\\ R_{id.assigned}\geqslant R_{id.endure},\text{对于每一个 } id\end{cases}$$

3. Quixote 算法描述

Quixote(QoS requirement fixed on negotiation and ad-

*)本文获863计划通信技术主题专家组成员课题(863-317-04-05-99)、863重大课题(863-300-01-03-99)、河南省自然科学基金(0111061300)资助。陈庶樵　博士研究生，主要从事 IP 网络体系结构分析研究工作。邬江兴　教授，博导，国家863高技术研究发展计划通信技术主题专家组成员，主要从事信息系统和计算机应用研究工作。

mission control)算法是一种基于软实时应用支撑系统获益方程的QoS协商和接纳允许控制算法。与传统的QoS协商和接纳允许控制算法相比,Quixote算法具有以下优点:

·接纳允许率高:根据软实时应用的特点,允许动态降低某些应用的 QoS_{level},增加系统的剩余资源 $R_{available}$,从而可以接纳更多的软实时应用;

·系统资源利用率高:从整个系统的角度出发,基于软实时应用支撑系统获益方程,而不是简单地考虑一个或几个软实时应用的 QoS_{level},可以使系统的资源利用率达到最高;

·允许定义不同应用的优先权:通过改变 $QoS_{level}=Quality(R_{assigned})$ 的定义,可以改变应用的优先权,从而使较为重要的应用得到较多的系统资源和较高的接纳允许率。

Quixote算法的简单描述如下:

```
Switch 事件 of
Case 新应用 App_i 到达:
    AA 读取系统可资源 R_available;
    if (R_required,i ≤ R_available) then
        AA 接纳 App_i;
        AA R_assigned,i ← R_Rrequired;
    else
        AA 重新计算 Max(System_benefit,new);
        if(Max(System_benefit,new) > Max(System_benefit,current))
            AA 接纳 App_i;
            AA 重新分配系统资源;
        else
            AA 拒绝 App_i;
case 应用 App_i 结束
    AA R_available ← R_available + R_assigned,i;
    AA 重新计算 Max(System_benefit,new);
    AA 重新分配系统资源;
End switch
```

4. 实验结果和分析

为了评估Quixote算法的性能指标,我们建立以下模型进行计算机仿真。

·软实时应用到达支撑系统的过程是服从参数lembda的泊松过程;

·软实时应用在的支撑系统中的执行过程是服从参数u的泊松过程;

·$QoS_{level}=W\times R_{assigned}$,其中w是服从平均分布的随机变量;

·每个软实时应用向支撑系统所申请的资源是服从平均分布的随机变量;

·能够满足每个软实时应用最低要求的资源是服从平均分布的随机变量。

主要的评估方法是将Quixote算法与传统的worst case算法[5]进行比较,比较主要从以下三个方面进行:支撑系统的接纳允许率;支撑系统的资源利用率;支撑系统的总获益。下面给出仿真结果。

4.1 支撑系统的接纳允许率比较

表1 支撑系统接纳允许率比较

u	lambda	Worst case算法接纳允许率	Quixote算法接纳允许率
100	10	97%	98%
120	10	95%	96%
150	10	78%	86%
200	15	85%	93%

从表1可以看出,在支撑系统的负载u/lambda较低时,Quixote算法与Worst case算法的接纳允许率比较接近,而当支撑系统的负载较大时,Quixote算法的接纳允许率明显高于Worst case算法的接纳允许率。

4.2 支撑系统的资源利用率比较

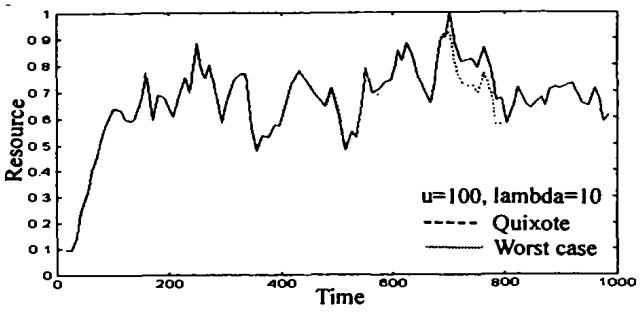

图1 支撑系统负载较小时的资源利用率比较

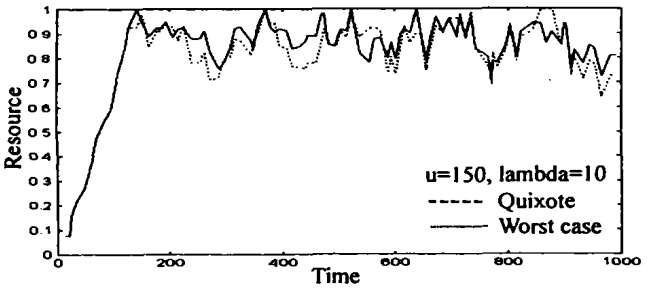

图2 支撑系统负载较大时的资源利用率比较

从图1和图2可以看出,Quixote算法资源利用率一般要高于Worst case算法,特别是支撑系统负载较大时,而且总是能够接近于系统的资源总量。当Worst case算法资源利用率高于Quixote算法时,其主要原因是Worst case算法没有考虑各个软实时应用的优先权所致。

4.3 支撑系统的总获益比较

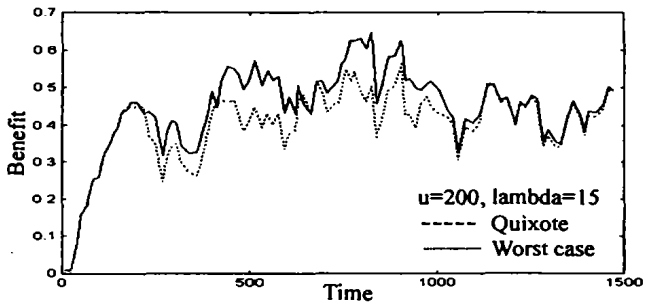

图3 支撑系统负载较小时的总获益比较

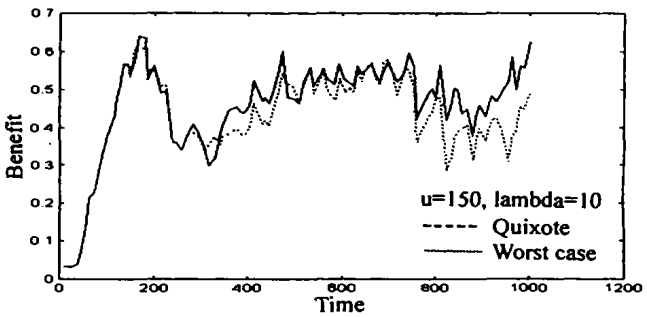

图4 支撑系统负载较大时的总获益比较

从图3和图4可以看出,因为Quixote算法利用规划理论

(下转第55页)

行物化选择，使用算法3进行维护）做了一个比较，系统平均查询反应时间如图2所示，系统查询数据的 Freshness 对比如图3所示。

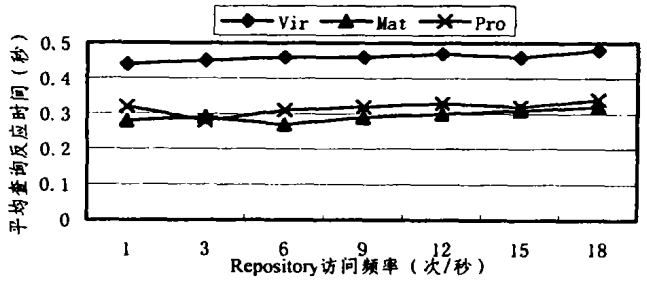

图2 几种方法的平均查询反应时间比较

从图2中可以看到，Virtual 方式查询反应时间最长，而 Materialized 方式和 Profit-based 方式效率相差不多，前者要稍稍强些。

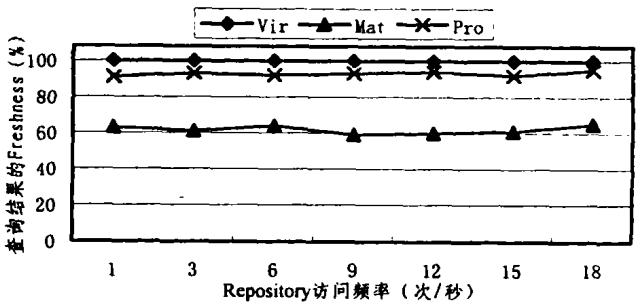

图3 几种方法查询结果的 Freshness 比较

从图3中可以看到，Virtual 方式的查询结果肯定是最新的（Freshness＝100%），而 Profit-based 方式的页面 Freshness 达到90%以上，远远超过 Materialized 方式的60%。

结论 本文介绍了在 Web Repository 集成环境下的 WebView 物化选择方案和自适应维护方案。方案基于效益模型并且高度自适应。通过实验可以看出，我们的方案对于访问非常频繁的 Web 环境具有良好的适应性，在 Web Repository 的访问频率增大时系统性能没有明显降低，查询结果的 Freshness 也没有明显变化。

未来的工作是，我们希望在一个大型的 Web 集成环境中验证这些算法，并结合用户对数据 Freshness 的容忍程度（tolerance），进一步改进物化选择和维护方案。

参 考 文 献

1 Arlitt M F, Williamson C. Internet Web Servers: Workload Characterization and Performance Implications. IEEE/ACM Transactions on Networking, 1997, 5(5)

2 Cluet S, et al. Your Mediators Need Data Conversion!. In: Proc. of ACM SIGMOD Intl. Conf. on Management of Data, June 2-4, 1998, Seattle, Washington, USA

3 Cho J, Garcia-Molina H. Estimating Frequency of Change. Submitted for publication, Feb. 2000

4 Fernandez M, et al. STRUDEL: A Web-site Management System. In: ACM Intl. Conf. on management of Data (SIGMOD), 1997. 549～552

5 Gupta A, Mumick I S. Maintenance of Materialized Views: Problems, Techniques and Applications. Data Engineering Bulletin, 1995, 18(2): 3～18

6 Ludascher B, et al. Managing Semistructured Data with C: A Deductive Object-Oriented Perspective. Information Systems, 1998, 23(8)

7 Ludäscher B, Papakonstantinou Y, Velikhov P. A Framework for Navigation-Driven Lazy Mediators. ACM SIGMOD Workshop on The Web and Databases (WebDB'99), Philadelphia, Pennsylvania, USA, Informal Proceedings, INRIA A. 1999. 85～90

8 Labrinidis A, Roussopoulos N. Web View Materialization. In: Proc. of the ACM SIGMOD Conf. Dallas, Texas, USA, May2000

9 Labrinidis A, Roussopoulos N. Adaptive WebView Materialization. In: Proc. of the 4th Intl. Workshop on the Web and Databases (WebDB'2001), Santa Barbara, California, USA, May24-25, 2001, held in conjunction with ACM SIGMOD'2001

10 Labrinidis A, Roussopoulos N. On the Materialization of WebViews. In: Proc. of the ACM SIGMOD Workshop on the Web and Databases (WebDB'99), Philadelphia, USA, June 1999

11 McHugh J, et al. Lore: A Database Management System for Semistructured Data. SIGMOD Record, 1997, 26(3): 54～66

12 Roussopoulos N. Materialized Views and Data Warehouses. SIGMOD Record, 1998, 27(1)

13 Abiteboul S. Querying semi-structured data. In: Foto Afrati, Phokion Kolaties ed. Lecture Notes in Computer Science 1186, Database Theory —ICDT' 97. New York: Springer-Verlag, 1997. 1～18

14 Ullman J D. Information Integration Using Logical Views. In: Proc. of the 6th Int. Conf. on Database Theory (ICDT'97), volume 1186 of Lecture Notes in Computer Science, pages 19～40

（上接第38页）

进行系统资源的分配，可以系统的总获益要高于 Worst case 算法，这尤其表现在支撑系统负载较大时。在系统启动初期，由于系统空余资源较多，所以 Quixote 算法与 worst case 算法几乎一致，这也是在预料之中的。

小结 本文根据现代网络中大量出现软实时应用的特点，给出了一种基于软实时应用支撑系统获益方程的 QoS 协商和接纳允许控制算法。在 Quixote 算法中，充分考虑到软实时应用与传统的硬实时应用的区别，利用规划理论进行系统建模，并给出计算机仿真结果。仿真数据充分说明，Quixote 算法在支撑系统的接纳允许率、支撑系统的资源利用率、支撑系统的总获益等性能指标上要明显地优于传统的 Worst case 算法。下一步的工作是将 Quixote 算法应用于实际系统中，包括网络主机或者路由器，相信会得到较满意的结果。

参 考 文 献

1 Shenker S, Partridge C, Guerin R. Specification of Guaranteed Quality of Service, RFC 2212, Sept. 1997

2 Shenker S, Wroclawski J. General Characterization Parameters for Integrated ServiceNetwork Elements. RFC 2215, Sept. 1997

3 Shenker S, Wroclawski J. Network Element Service Specification Template, RFC 2216, Sept. 1997

4 Rangan P V, Vin H M. Designing a Multiuser HDTV Storage Server. IEEE Journal on Seleted Areas in Communications, 1993, 11: 153～164

5 Braden R, et al. Resource ReSerVation Protocol (RSVP)-Version 1 Functional Specification. RFC 2205, Sept. 1997