文章编号：2095-4980(2015)01-0046-06

一种优先级与带宽需求相结合的分组调度算法

江　明[1,2]，刘　锋[1,2]

(1.北京航空航天大学 电子信息工程学院，北京 100191；2.国家空管新航行系统技术重点实验室，北京 100191)

摘　要：优先级队列(PQ)算法虽然能够保证高优先级业务的服务质量，但低优先级业务的性能较差，整体性能不佳，公平性较低。针对这些不足，提出优先级与带宽需求相结合的调度算法(PRQ)，在优先级调度的基础上，使用带宽需求对调度概率进行调整，提高低优先级业务的调度概率，进而改善其服务质量，同时改善整体性能，提高公平性。仿真结果表明，PRQ 算法能够显著改善低优先级业务的性能和整体性能，公平性较 PQ 算法高。

关键词：分组调度；优先级；丢包率；时延；时延抖动；吞吐量；公平性

中图分类号：TN929.5；TP393.02　　**文献标识码：**A　　**doi：**10.11805/TKYDA201501.0046

A packet scheduling algorithm combining priority and bandwidth requirement

JIANG Ming[1,2]，LIU Feng[1,2]

(1.School of Electronic and Information Engineering，Beihang University，Beijing 100191，China；
2.National Key Laboratory of CNS/ATM，Beijing 100191，China)

Abstract：Priority Queueing(PQ) guarantees that traffics with higher priorities get higher Quality of Service(QoS), whereas the performances of those with lower priorities are poor, therefore reducing the total performances and fairness. A scheduling algorithm is proposed combining Priority and bandwidth Requirement Queueing(PRQ), which adopts bandwidth requirement to modulate scheduling probabilities on the base of priority scheduling, aiming to increases the scheduling probabilities of traffics with lower priorities, to improve their QoS and the total performances, and to receive better fairness as well. The simulation results show that PRQ significantly improves the QoS of traffics with lower priorities and the total performances, beating PQ on fairness.

Key words：packet scheduling；priority；packet loss rate；delay；delay jitter；throughput；fairness

随着互联网规模的不断增大，快速增长的业务对网络服务质量(QoS)提出了更高的要求[1-2]。分组调度在网络传输控制中发挥着相当大的作用，是保障 QoS 的核心技术之一。分组调度作用于瓶颈链路上，通过合理地安排分组流入下游节点的顺序来有效分配网络资源，满足各业务对资源的需求，保证各业务所需要的 QoS。

分组调度算法大致分为以下几类：基于优先级、基于轮询、基于通用处理机共享和基于时延等[3]。实际应用中，由于不同业务根据自身的特性可能要求不同的 QoS，同时也由于网络资源的有限性，系统通常会按照一定的原则将众多业务划分为不同的优先级。因此，基于优先级的分组调度算法被应用于这类将不同业务划分为不同优先级的场景中，根据优先级的高低来采取不同的策略分配网络资源，使有限的网络资源得到高效而合理的利用[4]。

基于优先级的分组调度算法中最经典的是优先级队列(PQ)。在 PQ 算法中，假设根据业务的 QoS 需求共分为 n 个优先级，则分别开辟 n 个优先级队列 Q_1~Q_n，用于存储对应优先级的分组。当进行调度时，首先选择优先级最高的队列 Q_1 进行服务，当 Q_1 中的分组全部调度完毕之后，再选择次高优先级的队列 Q_2 进行调度，以此类推。同一队列中则按照先到先服务(First Come First Served，FCFS)的原则进行调度。

PQ 算法的一大优点是它能为不同业务提供不同等级的服务。由于高优先级队列总是优先调度，因此能保证高优先级的业务得到更好的 QoS(如较低的丢包率、较低的时延和时延抖动、较高的吞吐量)等。PQ 算法的另一大优点是容易实现，复杂度较低。然而 PQ 算法的缺点也很明显，那就是必须严格按照优先级的高低来服务各队

收稿日期：2013-12-16；**修回日期：**2013-12-30
基金项目：国家自然科学基金重点资助项目(61231013)；国家自然科学基金重点资助项目(60933012)；新世纪优秀人才支持计划资助项目。

列。如果高优先级队列中一直有分组需要发送，那么低优先级队列中的分组就难以得到服务而导致持续积压，这会造成低优先级业务的 QoS 下降，丢包率和时延大大增加，进而降低整体性能。

在 PQ 算法的基础上，有研究者们提出了概率优先级(Probabilistic Priority，PP)的调度算法[5-6]，以期改善低优先级队列的时延和丢包性能，但其在计算调度概率时使用的参数是静态设置的，不能对网络实际情况进行自适应。PP 虽然也一定程度地保证了低优先级业务的 QoS，但实现较为复杂。且 PP 算法假设节点队列长度是无限的，而在实际中队列所能容纳的分组数目往往是有上限的。也有研究着眼于对优先级的划分问题，提出概率–优先级的分级调度算法[7]。该方法着重探讨如何把对 QoS 要求相近的业务划分优先级，而没有真正改善低优先级的 QoS。另外有研究尝试将 PQ 算法与其他类型的调度算法进行结合，如与基于时延的调度算法进行结合的基于最早截止时间优先的优先级队列(Priority Queue Based on Earliest Deadline First，PQBEDF[8])和与差额轮询(Deficit Round Robin，DRR)进行结合的 PDQ[9]等，但都没有给出确切的性能结果且可扩展性不佳。

本文从不同于上述研究的角度出发，在 PQ 算法的基础上，提出了一种优先级与带宽需求相结合的分组调度算法——PRQ。PRQ 算法在每次调度时以一定概率调度各个优先级业务的分组，在计算调度概率时不仅考虑优先级，同时考虑不同业务排队的带宽需求，对于队列中有大量分组积压的业务，能够相应提升其调度概率，从而解决低优先级业务的分组长时间得不到调度的问题，改善其丢包率、时延和时延抖动、吞吐量等性能，提升公平性。

1　PRQ 算法

PRQ 算法以一定的概率来调度各个优先级业务的分组，而调度概率的计算主要考虑两方面因素：优先级和带宽需求。PRQ 算法是基于优先级的调度算法，因此优先级在决定调度顺序时仍起到重要作用。同时为了解决高优先级业务持续占据资源，导致较低优先级业务分组积压的问题，使用带宽需求调整各优先级的概率，让较低优先级业务的分组也有机会得到调度，从而改善其时延和丢包率等性能。

1.1 PRQ 算法机制

1.1.1 优先级因素

PQ 算法在调度时完全按照优先级的顺序进行，而在 PRQ 算法中，优先级仍然是决定调度顺序的重要因素。在本文中规定优先级越高，对应的数字越小，优先级为 1 的业务具有最高优先级，优先级为 n 的业务具有最低优先级。

定义 1　优先级因素：优先级为 i 的业务，其优先级因素为 $1/2^i$。

设共有 n 个优先级的业务，则所有业务的优先级因素加和为 $\sum_{i=1}^{n}1/2^i$。此加和不等于 1，而是随着优先级个数 n 的增加无限趋近于 1。记此加和与 1 的差额为 D，即 $D=1-\sum_{i=1}^{n}1/2^i$。

1.1.2 带宽需求

在 PQ 算法中，高优先级持续占据输出链路的带宽资源，导致低优先级业务分组积压。而低优先级业务分组积压越多，对带宽的需求就越高。如果持续得不到调度，就会导致很高的时延和很大的丢包率。PRQ 算法在计算调度概率时考虑到带宽需求的因素，当一个业务的分组对带宽需求很高(即积压分组很多)时，其调度概率也会相应提升。

定义 2　带宽需求：设优先级为 i 的业务在某节点上积压的分组数目为 N_i，该节点积压的分组总数为 S，则该业务在该节点上的带宽需求为 N_i/S。

所有业务的带宽需求的加和为 $\sum_{i=1}^{n}N_i/S$，且此加和等于 1。

1.1.3 调度概率

PRQ 算法在进行调度时按照概率来决定调度哪个业务的分组，在计算调度概率时综合考虑优先级因素和带宽需求。设优先级为 i 的业务，其带宽需求为 N_i/S，则其调度概率计算为：

$$P_{\text{schd}}^{i}=\frac{N_i}{S}\alpha+\frac{1}{2^i}(1-\alpha) \tag{1}$$

式中 α 表示队列积压程度，计算为：

$$\alpha = \frac{S}{SUM} \tag{2}$$

式中：S 表示节点上积压的分组总数；SUM 表示该节点可以容纳的分组上限。易见 $\alpha \in [0,1]$，显然节点积压越严重，α 越大，当 $\alpha = 1$ 时，节点队列已经满溢。

1.2 PRQ 算法实现

假设网络中共有 n 个优先级的业务，在瓶颈节点上，维持 n 个队列 Q_1~Q_n，每个队列 Q_i 存储第 i 个优先级的分组，同一队列内的分组按照 FCFS 的原则进行调度。

1.2.1 节点上积压所有优先级业务的分组

在进行调度时，首先按照式(1)计算出各优先级的调度概率 P_{schd}^{i}，$i=1,2,\cdots,n$，然后用所得到的 n 个调度概率将[0,1]区间分割成 n 个区间段：

$$\left[0, P_{\text{schd}}^{n} + \frac{D}{n}\right], \left(P_{\text{schd}}^{n} + \frac{D}{n}, P_{\text{schd}}^{n} + P_{\text{schd}}^{n-1} + \frac{2D}{n}\right], \cdots, \left(\sum_{i=3}^{n} P_{\text{schd}}^{i} + \frac{(n-2)D}{n}, \sum_{i=2}^{n} P_{\text{schd}}^{i} + \frac{(n-1)D}{n}\right], \left(\sum_{i=2}^{n} P_{\text{schd}}^{i} + \frac{(n-1)D}{n}, 1\right]$$

使用随机数生成器生成一个在[0,1]区间内均匀随机分布的随机数 R。若 R 落在 $\left[0, P_{\text{schd}}^{n} + \frac{D}{n}\right]$ 区间内，则调度第 n 优先级的分组，即让 Q_n 队首分组出队；若 R 落在 $\left(\sum_{i=K+1}^{n} P_{\text{schd}}^{i} + \frac{(n-K)D}{n}, \sum_{i=K}^{n} P_{\text{schd}}^{i} + \frac{(n-K+1)D}{n}\right]$ 区间内，则调度第 K 优先级的分组，即让 Q_K 队首分组出队；若 R 落在 $\left(\sum_{i=2}^{n} P_{\text{schd}}^{i} + \frac{(n-1)D}{n}, 1\right]$ 区间内，则调度第 1 优先级的分组，即让 Q_1 队首分组出队。

1.2.2 节点上缺少部分优先级业务的分组

以上讨论的是节点中积压了所有优先级分组的情况。如果节点中缺少某几个优先级业务的分组，设缺少第 $i_1, i_2, \cdots, i_m$ 优先级业务的分组(共缺少 m 个优先级，且 $i_1, i_2, \cdots, i_m$ 不一定为连续自然数)，则把这些业务的优先级因素平均分给其他的优先级，即各优先级的调度概率为：

$$P_{\text{schd}}^{i} = \frac{N_i}{S} \times \alpha + \frac{1}{2^i} \times (1-\alpha) + \frac{1}{n-m} \sum_{i_j=1}^{i_m} \frac{1}{2^{i_j}} \tag{3}$$

式中 $i = 1,2,\cdots,n$ 且 $i \neq i_1, i_2, \cdots, i_m$。

计算出各优先级业务的调度概率之后，再按照 1.2.1 节中所述的随机数落区间的方法进行调度。

1.3 PRQ 算法分析

PRQ 算法按照各业务的优先级因素和带宽需求来计算调度概率，并按照调度概率来调度各个业务的分组。相比于 PQ，PRQ 在保持高优先级有高 QoS 的同时，提升低优先级业务的调度概率，缓解其积压程度，改善低优先级的丢包率、时延和时延抖动、吞吐量等性能，增强了公平性。由于低优先级的时延降低，整体的时延、时延抖动等性能也得到很大提高。

相比于先前的概率调度算法，PRQ 并不假设节点队列长度是无限的，而是认为其有上限 SUM，更符合实际情况，有利于准确分析丢包率等性能。且 PRQ 算法的参数是根据网络情况动态调整的，带宽需求的计算总是由当前队列中的实际分组数来决定，因而能够更好地反映各业务的积压情况。

参数 α 能够实时反映节点上的总体积压情况，进而调整调度概率的计算：当节点总体积压严重时 α 较大，则带宽需求对调度概率的影响较大，让系统优先调度当前积压情况最为严重的业务；当节点总体积压较轻时 α 较小，则优先级对调度概率的影响较大，让系统按照优先级的顺序调度分组，保证高优先级业务的 QoS。

2 仿真结果

仿真使用 NS2 模拟实验平台[10-13]对 FSFC,PQ 和 PRQ 三种算法进行了实现和性能评估，其中主要比较 PQ 算法与 PRQ 算法，FCFS 仅做参考。仿真采用的性能指标有 5 项，分别是：a) 丢包率：在传输过程中丢失的数据分组数与信源总共发送的数据分组数的比率；b) 时延：信源发送一个分组到信宿接收到该分组的时间差；c) 时

延抖动：前后 2 个分组之间的不同时延差；d) 吞吐量：单位时间内信宿接收到的数据量；e) 公平性：引用文献[14]中定义的公平性指数(Fairness Index，FI)。

2.1 仿真场景

仿真使用的场景是队列算法实验典型的哑铃型结构，如图 1 所示。节点 0~4 为 5 个 ftp 流信源，业务代号分别为 ftp1~ftp5，优先级依次为 1~5(ftp1 优先级最高，ftp5 最低)，节点 7~11 分别为对应的信宿。节点 5 和 6 之间的链路为瓶颈链路，带宽为 0.5 MB，延时为 20 ms。其余链路带宽为 2 MB，无延时。所有 ftp 源的发送窗口均为 10 000，所有分组大小均为 500。仿真时间为 1 000 s。

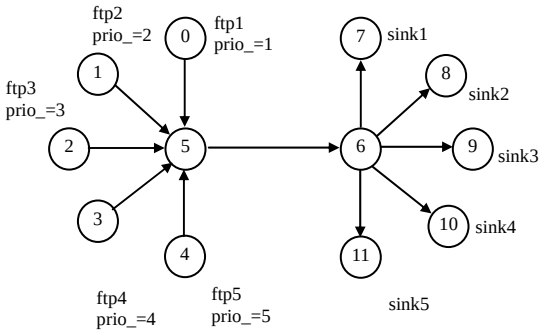


Fig.1 Simulation scenes
图 1 仿真场景

2.2 仿真实验结果

2.2.1 丢包率

表 1 为仿真实验的丢包率统计结果。从表中可以看出，PQ 算法下，不同优先级的业务之间性能差距非常明显，低优先级业务不仅发送数据包非常少，而且丢包率很高。而高优先级业务虽然发送数据包数很多，但丢包率也很高，导致整体的丢包率较高。PRQ 算法中，在保持高优先级业务的 QoS 优势基础上，改善了低优先级业务的性能，使之不仅发送数据包增加，而且丢包率大大降低。尽管总体发送数据包数比 PQ 算法略少，但总体丢包率比 PQ 算法大幅降低。而与 FCFS 算法相比，无论是总发包数还是总体丢包率，都相差无几。

表 1 丢包率统计
Table1 Loss rate statistics

source	FCFS			PQ			PRQ		
	send	loss	loss rate	send	loss	loss rate	send	loss	loss rate
ftp1	12 242	206	0.016 827	34 292	3 210	0.093 608	20 867	340	0.016 294
ftp2	12 309	200	0.016 248	22 930	2 338	0.101 962	14 456	210	0.014 527
ftp3	11 250	216	0.019 200	8 190	842	0.102 808	10 716	164	0.015 304
ftp4	12 919	206	0.015 946	1 068	80	0.074 906	8 157	144	0.017 654
ftp5	12 379	196	0.015 833	90	13	0.144 444	6 468	128	0.019 790
total	61 099	1 024	0.016 760	66 570	6 483	0.097 386	60 664	986	0.016 253

根据表 1 中 3 种算法的“丢包率”数据，绘制丢包率柱状图，如图 2 所示。图中每个 ftp 源的左边柱为 FCFS 数据，中间柱为 PQ 数据，右边柱为 PRQ 数据。从图 2 中可以形象地看出，PRQ 算法中各优先级业务的丢包率比 PQ 算法大大降低，整体丢包率性能也远比 PQ 算法优异。

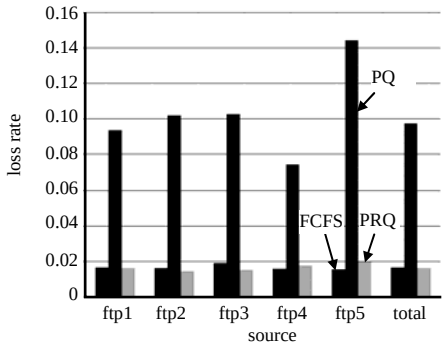


Fig.2 Bar graph of loss rate statistics
图 2 丢包率统计柱状图

表 2 时延及时延抖动统计
Table 2 Delay and delay jitter statistics

source	FCFS		PQ		PRQ	
	delay	delay jitter	delay	delay jitter	delay	delay jitter
ftp1	0.671 952	0.002 160	0.020 960	0.002 997	0.539 091	0.008 512
ftp2	0.671 575	0.002 150	0.128 104	0.009 184	0.660 818	0.017 752
ftp3	0.673 999	0.002 267	0.487 522	0.028 949	0.731 703	0.023 765
ftp4	0.671 170	0.002 096	9.058 256	0.158 158	0.770 774	0.029 731
ftp5	0.671 850	0.002 124	579.689 967	1.433 881	0.811 626	0.035 498
total	0.672 066	0.001 931	1.006 164	0.158 489	0.662 264	0.006 787

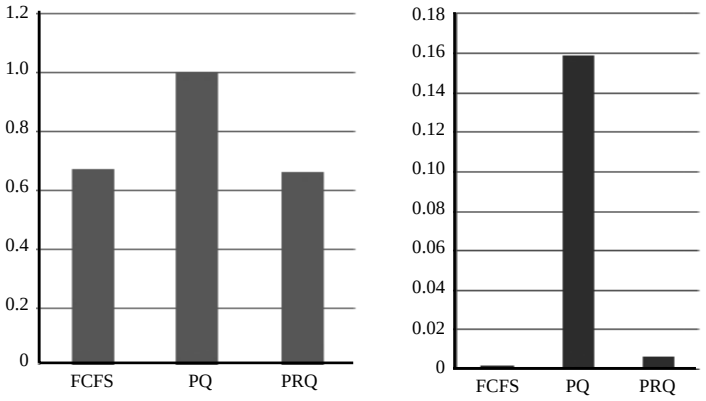
2.2.2 时延与时延抖动

表 2 为仿真实验的平均时延和平均时延抖动统计结果。从表中可以看出，PQ 算法下各优先级业务的时延性能差异很大，低优先级业务的时延非常高。由于各优先级业务的性能差异很大，导致时延抖动很大。PRQ 算法中，低优先级业务的时延性能得到极大改善，总体时延比 PQ 大幅降低，而时延抖动性能则更比 PQ 优异。而相比于 FCFS 算法，PRQ 算法也是注重优先级的，因此时延及时延抖动性能比更为公平的 FCFS 算法略低，但从数量级的对比上来看，这个差值远比 PRQ 与 PQ 算法的差值要小得多。

根据表 2 中的“总体”数据，绘制时延与时延抖动柱状图如图 3 所示。从图 3 中能够形象地看出，PRQ 算法的时延性能和时延抖动性能都远远好于 PQ 算法。

2.2.3 吞吐量

图 4 为仿真实验的吞吐量统计结果。其中，图 4(a)和 4(b)分别为 PQ 算法和 PRQ 算法各业务吞吐量。从图中可以清楚地看到，PQ 算法中低优先级业务的吞吐量非常低，而 PRQ 算法在保证高优先级业务有高 QoS 的同时，也兼顾了低优先级业务的吞吐量性能。

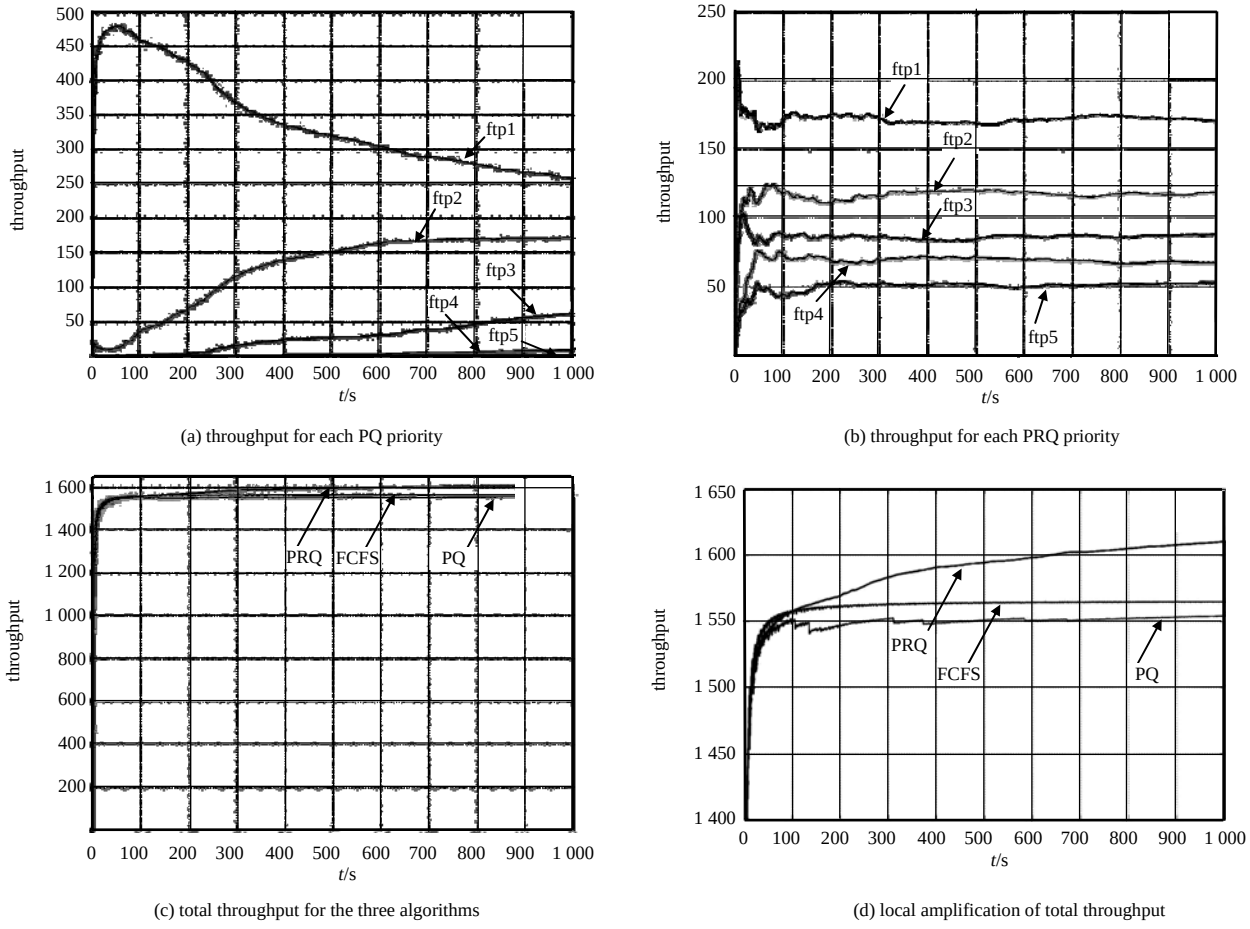


(a) bar graphs of delay statistics (b) bar graphs of delay jitter statistics

Fig.3 Bar graphs of delay and delay jitter statistics

图 3 时延与时延抖动柱状图

图 4(c)和图 4(d)为总体吞吐量统计图。从图中可以看到，PQ 算法的总体吞吐量性能不高，甚至低于 FCFS。而 PRQ 算法能够提升稳态时的总体吞吐量。



(a) throughput for each PQ priority

(b) throughput for each PRQ priority

(c) total throughput for the three algorithms

(d) local amplification of total throughput

Fig.4 Throughput statistics

图 4 吞吐量统计

2.2.4 公平性

为了评估 PRQ 算法的公平性能，引用文献[14]中对算法公平性指标的定义：

$$FI=\left(\sum_{i=1}^{n}x_i\right)^2\Big/n\left(\sum_{i=1}^{n}x_i^2\right) \tag{4}$$

式中：x_i 表示每个优先级信源发送的分组数量；n 表示信源个数。公平性指数 $FI\in[0,1]$，FI 越大，公平性越好。

根据表 1 中各算法的“发送”数据，由式(4)计算得 3 种算法的公平性指数 *FI* 如表 3 所示。容易看出，PQ 算法的公平性能不佳，而 PRQ 算法能够显著地改善公平性。不过，由于 PRQ 算法相比于 FCFS 算法仍是注重优先级的，因此公平性要比 FCFS 算法低。

表 3 公平性统计

Table3 Fairness statistics

algorithm	FCFS	PQ	PRQ
FI	0.998 044	0.500 755	0.848 333

3　结论

在 PQ 算法以优先级顺序调度分组的基础上，PRQ 算法在调度时还考虑了不同业务对带宽的需求，提升低优先级业务的调度概率，缓解其积压程度。在保持高优先级业务的 QoS 较高的基础上，改善低优先级业务的丢包率、时延和时延抖动、吞吐量等性能，同时整体性能也有不同程度的提升。

参考文献:

[1] Gu é rin R,Peris V. Quality-of-Service in packet networks:basic mechanisms and directions[J]. Computer Networks, 1999, 31(3):169–189.

[2] LIN Chuang,SHAN Zhiguang,REN Fengyuan. Quality of service of computer networks[M]. Beijing:Tsinghua University Press, 2004.

[3] WANG Chonggang,LONG Keping,GONG Xiangyang. The study and perspective of queue scheduling algorithms in packet switching networks[J]. Acta Electronica Sinica, 2001,29(4):553–559.

[4] ZHOU Peng,HAO Ming,TANG Zheng. Research on the priority queue scheduling algorithm based on QoS[J]. Electronic Science and Technology, 2013,26(5):122–124.

[5] JIANG Y,Tham C K,Ko C C. A probabilistic priority scheduling discipline for multi-service networks[J]. Computer Comm., 2002,25(13):1243–1254.

[6] Tham C K,YAO Q,JIANG Y. Achieving differentiated services through multi-class probabilistic priority scheduling[J]. Computer Networks, 2002,40(4):577–593.

[7] LUO Huimei,GAO Qiang,SONG Shuang. The study of hierarchical packet scheduling algorithm on probability-priority[J]. Computer Applications and Software, 2011,28(7):57–59.

[8] QIAN Guangming. A diffserv mechanism of multi priority queues based on business[J]. Computer Engineering and Applications, 2006,42(10):118–120.

[9] LUO Zhangqin,LIN Yuhong,LI Ling. A queuing discipline supporting priority and fairness strategies[J]. Communications Technology, 2010(8):32–34.

[10] Popa L,Raiciu C,Stoica I,et al. Reducing congestion effects in wireless networks by multipath routing[C]// Proceedings of the 2006 14th IEEE International Conference on Network Protocols. [S.l.]:IEEE, 2006:96–105.

[11] HUANG Huaji,FENG Shuili, QIN Lijiao. NS network simulator and protocol simulation[M]. Beijing:Post&Telecom Press, 2010.

[12] FANG Luping,LIU Shihua,CHEN Pan. Foundation and application of Network Simulator version 2[M]. Beijing:National Defense Industry Press, 2008.

[13] NS2[EB/OL]. [2013–11–08]. http://www.isi.edu/nsnam/ns/.

[14] Chiu D M,Jain R. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks[J]. Computer Networks and ISDN systems, 1989,17(1):1–14.

作者简介:

江　明(1988–)，男，辽宁省大连市人，在读硕士研究生，主要研究方向为天空地通信网络、网络编码.email:jmjm.47521.gzwgz@163.com.

刘　锋(1970–)，男，湖北省麻城市人，博士，教授，主要研究方向为网络通信理论和技术、网络行为学等.