Resampling for Wireless Access

Ming-Young You   Cheng-Shang Chang*

Dept. of Electrical Engineering
National Tsing Hua University
Hsinchu 30043
Taiwan, R.O.C.

June, 1996

Abstract

The well known problem among most random access protocols in wireless networks is that the throughput drops rapidly in heavy load. To cope with this problem, one has to control to the load offered to a network. Unlike the traditional backoff policy in Ethernet where backoff occurs after collision, in this paper we propose various control schemes based on the new idea of resampling, where carrier sensing ability is used to determine whether a backoff command should be issued or not. We show from various experiments that our schemes are capable of controlling the offer load to the optimal offer load. As a result, the throughputs of these schemes are kept close to the network capacity in heavy load.

Key words: wireless networks, multiple access, random access, throughput, capacity

*Corresponding author. E-mail:cschang@ee.nthu.edu.tw. Fax: +886-35-715971

1 Introduction

In most random access protocols, including the ALOHA family in [1, 2] and CSMA [7], one common problem is that the throughput drops rapidly in heavy load. To cope with this problem, one has to control the load offered to the system. In Ethernet, offer load is controlled by a binary backoff policy. Each time a collision is detected, the mean of the random time for next retransmission is increased by a factor of 2. Recently, a new control scheme based on resampling is proposed in [3]. In such a scheme, backoff might occur before a collision is detected. The idea is to use the carrier sensing ability to estimate the offer load before actual data are transmitted. If the estimated offer load is not satisfactory, then a backoff command is issued and the offer load is re-estimated (resampled) after the backoff command. Repeat the procedure until the estimated offer load is satisfactory. Then actual data are transmitted. The intuition behind resampling is as follows: once the estimated offer load is not satisfactory, it is expected there will be many collisions if actual data are transmitted. It might be better to backoff first to avoid possible collisions.

Our objective of this paper is to provide efficient and practical schemes that implements the idea of resampling. We do this based on the Group Randomly Addressed Polling (GRAP) protocol proposed by Chen and Lee [5] (see Section 2 for details). Since GRAP and nonpersistent CSMA are shown to have the same throughput-offer load characteristic [3], GRAP has the same problem as CSMA in heavy load. To control the offer load in GRAP, we implement a resampling scheme into the GRAP protocol, called GRAP with resampling in Section 2.2. We also propose two variants of GRAP with resampling to ease implementation. The first one is GRAP with binary backoff in Section 2.3 and the second one is GRAP with a limited polling group size in Section 2.4. To verify our schemes, we perform various experiments. These experimental results, reported in Section 3, show that our resampling schemes perform very well. In may cases, the throughputs in heavy load are very close to the network capacity.

2 Group randomly addressed polling

In a wireless multiple access network, all sources share a common medium to communicate with each other. In our model, we assume there is a base station that negotiates the access process. The base station is capable of sending packets to all users through a broadcast channel (down link). On the other hand, the base station is capable of receiving packets from sources through a multiple access channel (up link). Assume that all packets are lost when these packets are sent by different sources at the same time in the up link. We call this specific condition *collision*. To investigate this collision problem further, we concentrate ourselves on the analysis for the up link.

Suppose there are N sources. A source is called active if it has a packet to send. As discussed in [3], an ideal scheme for these N sources to share the channel fairly is to create a deterministic mapping so that every active source takes turn to send its packet. To achieve this, every source requires two pieces of information: (i) the set of active sources, and (ii) the deterministic mapping. One way to implement this is to reserve each source a packet-to-send slot. Active sources mark their corresponding slots so that the base station can have the information of the number of active sources. Then the base station polls active sources one by one to create the deterministic mapping (see e.g., the bit-map protocol [6]).

In a wireless environment, the number of active sources is time varying. Moreover, at any moment, only a small portion of sources are active. In such an environment, it may not be economical and practical to reserve each source a packet-to-send slot. To deal with the problem that only a small portion of sources are active, one can partition active sources into groups and each group owns only one packet-to-send slot. Each active source randomly joins a group with equal probability and marks the corresponding packet-to-send slot. A scheme that does this efficiently is the Group Randomly Addressed Polling (GRAP) in [4, 5] that we will introduce in the next section.

2.1 GRAP with a fixed number of group

In this section, we briefly review the GRAP protocol with a fixed number of groups.

Group Randomly Addressed Polling (GRAP):

1. At the beginning of every GRAP cycle, the base station broadcasts a READY signal to all sources.

2. (Marking cycle) After receiving the READY signal, every active source joins randomly one of the groups with equal probability, and marks the corresponding packet-to-send slot for that group.

3. (Polling cycle) The base station then polls the marked groups one by one. Active sources in the group being polled send their packets. If there is only one active source in a group, then this active source will transmit its packet successfully. However, if there are more than one active sources in a group, then more than one (different) packets will be transmitted after being polled. This results in a collision and all collided packets are lost.

4. After the polling cycle being completed, repeat the procedure from Step 1.

We note that there are various implementations for the marking cycle. For instance, an active source might transmit a carrier to mark a packet-to-send slot, and the base station might use the carrier sensing ability to detect whether a packet-to-send slot is marked or not. In this case, the time of a packet-to-send slot is the time for the base station to sense a carrier.

When there are n active sources and b groups in the system. We define the offer load, $G = \frac{n}{b}$, which is the average number of active sources per group. Also define the throughput to be the proportional amount of time that successful packets are transmitted. Let τ_m, τ_p, τ_d, and τ_c be the time of a packet-to-send slot, the time of polling a group, the time to transmit a packet and the time to detect a collision. The following theorem, regarding the throughput-offer load characteristic, is shown in [3].

Theorem 2.1 When $b \rightarrow \infty$, the throughput-offer load characteristic is given as follows:

$$S(G) = \frac{Ge^{-G}\tau_d}{\tau_m + (\tau_p + \tau_d)Ge^{-G} + (\tau_p + \tau_c)(1 - e^{-G} - Ge^{-G})}, \tag{1}$$

where S is the throughput and G is the offer load. The capacity of the GRAP is $S(G^*)$, where $0 < G^* < 1$ is the unique solution of $G = 1 - \frac{\tau_p + \tau_c}{\tau_m + \tau_p + \tau_c} e^{-G}$.

To gain some insight, we plot in Fig. 1 the throughput-offer load characteristic $S(G)$ with $\tau_d = \tau_c = 0.1 msec$ and $\tau_p = \tau_m = 5\mu sec$.

2.2 GRAP with resampling

As shown in Fig. 1, the throughput of the GRAP with a fixed number of groups drops significantly when the offer load exceeds the optimal offer load G^*. To keep the throughput high in heavy load, we need to have GRAP operate close to G^*. To achieve this, a resampling scheme is proposed in [3]. Note that the marking cycle is corresponding to throwing balls into urns. The base station only knows how many urns are occupied. It turns out this piece of information is enough to yield a good estimate for the total number of balls that are thrown into urns (the number of active sources). As a result, one has a good estimate for the offer load $\hat{G}$ by $\hat{G} = \log\frac{b}{b-N_2}$ (see [3]), where b is the number of urns (groups) and N_2 is the number of occupied urns (marked groups). If $\hat{G} \gg G^*$, then the offer load is high and it is expected to have many collisions if the polling cycle is carried out. In this case, it might be better for the base station to run the marking cycle again with a larger group size. We call this process the resampling process. Clearly, resampling will reduce the offer load. However, as the marking cycle is done twice, the throughput-offer load characteristic is modified as follows:

$$S_2(G) = \frac{Ge^{-G}\tau_d}{2\tau_m + (\tau_p + \tau_d)Ge^{-G} + (\tau_p + \tau_c)(1 - e^{-G} - Ge^{-G})}. \tag{2}$$

To find the criterion for resampling, note that $S(\hat{G})$ is the estimated throughput for the current sample and that $S_2(G^*)$ is the best throughput for the next sample. If $S(\hat{G})$ is larger than $S_2(G^*)$, then resampling is not needed. Thus, we modify GRAP as follows:

GRAP with resampling:

2. After receiving the READY signal, every active source randomly joins a group with equal probability and marks the corresponding packet-to-send slot. The base station uses the information of the number of marked groups N_2 to form the estimate of the offer load $\hat{G} = \log\frac{b}{b-N_2}$.

If $\hat{G} > G^*$ and $S(\hat{G}) < S_2(G^*)$, go to step 1.

2.3 GRAP with binary backoff

To implement GRAP with resampling efficiently, one may have to limit the group size b to a set of finite numbers. As the truncated binary backoff policy in CSMA/CD [7], the group size can be chosen as $b = 2^k$ for some integer k. When resampling is needed, the group size b is increased by a factor of 2. To implement this idea, we first choose an interval (G_{s1}, G_{s2}) (with $G_{s1} \leq G^* \leq G_{s2}$) within which we would like to keep the offer load. At the beginning of a GRAP cycle, the base station not only broadcasts the READY signal but also the EXPONENT of the group size to inform all the sources to adjust their group sizes to 2^k if EXPONENT=k. After the marking cycle, the base station knows the number of marked groups N_2 and uses that to estimate the offer load $\hat{G} = \log \frac{b}{b-N_2}$. If the estimated offer load $\hat{G}$ is larger than G_{s2}, then resampling is performed. In this case, the base station broadcasts the READY signal and increases the EXPONENT by 1. This informs all the sources to double their group sizes, and initiates a new marking cycle. If $\hat{G}$ is smaller than G_{s2}, then there is no need for resampling and the base station proceeds the polling cycle. If $\hat{G}$ is also smaller than G_{s1}, the base station will decrease the EXPONENT by 1 at the beginning of the next GRAP cycle. The protocol is now modified as follows:

GRAP with binary backoff:

1. At the beginning of every cycle, the base station broadcasts not only the READY signal but also the EXPONENT of the group size.

2. After receiving the READY signal and the EXPONENT of the group size. If the EXPONENT=k, all sources (including the base station) adjust the group size b to 2^k. Then, every active source joins one of the groups with equal probability and marks the corresponding packet-to-send slot. The base station knows the number of marked groups N_2 after the marking cycle and forms the estimate, $\hat{G} = \log \frac{b}{b-N_2}$.

If $\hat{G} \geq G_{s2}$, increase the EXPONENT by 1 and go back to Step 1.

If $\hat{G} \leq G_{s1}$, decrease the EXPONENT by 1 and go Step 3. Otherwise, go to Step 3 directly.

Note that one might like to keep the minimum of the EXPONENT moderate so that the estimation of the offer load can be accurate. In our experiments below, for simplicity we set the minimum of the EXPONENT to be 0.

2.4 GRAP with a limited polling group size

From the previous section, we know the delay due to the marking cycle could be very large when the EXPONENT is large. This might not be practical for a wireless channel shared with real-time traffic. The idea to solve this problem is to limit the marking cycle to a maximum time while allowing the EXPONENT to go without bound. We call this implementation GRAP with a limited polling group size.

Let b_{max} be the maximum group size that can be polled in a GRAP cycle. Thus, the maximum marking cycle is $\tau_m b_{max}$. In each marking cycle, the base station stops the cycle when the marking time is larger than $\tau_m b_{max}$. At this moment, the base station doesn't know the total number of marked groups because it hasn't completed the whole marking cycle yet. The number of marked groups $\hat{N}_2$, which the base station collects during the stopped marking cycle, is only a portion of the total number of marked groups. Since the base station knows the group size used in the marking cycle, i.e., $\hat{b} = min(b, b_{max})$, and the group size b broadcast to the sources, one can form a linear estimate for the total number of marked groups, i.e., $N_2 = \hat{N}_2 * \frac{b}{\hat{b}}$.

At the end of the marking cycle, the base station polls the marked groups one by one. Note that active sources with their group number larger than $\hat{b}$ will not be polled as the marking cycle is stopped before they have a chance to mark their corresponding packet-to-send slots.

We modify the adaptive algorithms for GRAP as follows:

GRAP with a limited polling cycle

2. After receiving the READY signal, every active source randomly joins a group with equal probability and marks the corresponding packet-to-send slot. The base station stops the marking cycle if the marking time is larger than $\tau_m b_{max}$. Estimate the number of marked groups by $N_2 = \hat{N}_2 * \frac{b}{\hat{b}}$, where $\hat{N}_2$ is the number of marked groups in the marking cycle and $\hat{b} = min(b, b_{max})$ is the group size used in the marking cycle.

3 Numerical results

For our experiments, the length of a packet is 1000 bits. The length of packet-to-send slot is 50 bits. The length of polling a group is 50 bits and the length of collision detection is 1000 bits. The channel capacity is 10 Mbits/sec. Thus, $\tau_d = \tau_c = 0.1m$sec and $\tau_p = \tau_m = 5\mu$sec, where τ_d, is the time to transmit a packet , τ_c is the time to detect a collision, τ_p is the time to poll a group, and τ_m is the time of a packet-to-send slot. In all our simulations, the amount of time for a nonactive source to become active is exponentially distributed with mean $\frac{1}{\mu}$ and it is independent of all else. An active source remains active until its packet is transmitted successfully. According to Theorem 2.1, we plot the throughput-offer load characteristic of GRAP in Fig. 1. For our setting. the capacity is found to be 0.6906. This occurs when the offer load $G = 0.27$.

We perform the experiment of GRAP with binary backoff with 20 sources in Fig. 2. In the experiment of GRAP with binary backoff, the interval (G_{s1}, G_{s2}) (with $G_{s1} \leq G^* \leq G_{s2}$) that we would like to keep the offer load in is (0.15,0.5). We also plot in Fig. 3 the throughput of GRAP with binary backoff for 200 sources. In both figures, we also plot the simulation results for fixed GRAP and GRAP with resampling in [3]. Clearly, both GRAP with resampling and GRAP with binary backoff outperform the fixed GRAP. Moreover, the performance of GRAP with binary backoff is almost the same as that of GRAP with resampling. They both are also very close to the capacity when μ increases.

In a wireless environment, the number of active sources is time varying. To model this phenomenon, we vary μ with respect to time in Fig. 4 and 5. We plot in Fig. 4 and 5 the throughputs of GRAP with resampling and GRAP with binary backoff when μ is varying with respect to time. In Fig. 4 (resp. Fig. 5), there are 20 sources (resp. 200 sources) and μ is normalized by 10000 (resp. 1000). From Fig. 4 and 5, we note that the adaptive speed of GRAP with binary backoff is not as fast as that of GRAP with resampling. Furthermore, even when μ drops rapidly in Fig 5, GRAP with resampling still maintains high throughput for a while. This is due to the fact that backlogged sources are not resolved completely when μ drops. Thus, the number of sources that has packets to transmit or retransmit does not decrease as rapidly as μ, and the throughput does not drop accordingly.

We assume the maximum polling group size, b_{max}, is 20 in all the experiments of GRAP with a limited polling group size. In Fig. 6, we show that the adaptive ability of GRAP with a limited polling group size is almost as good as that without limited polling group size. However, When we vary μ with respect to time, the throughput of GRAP with a limited polling group size decays, as shown in Fig 7. This can be explained as follows: since the base station is not capable of knowing the total number of marked groups when $b > b_{max}$ (b is the broadcast group size), the base station only estimates the total number of marked groups N_2 by the linear estimate, $N_2 = \hat{N}_2 * \frac{b}{\hat{b}}$, where $\hat{N}_2$ is the number of marked groups collected during the stopped marking cycle and $\hat{b}$ is the number of groups that the base station polls during the stopped marking cycle, i.e., $\hat{b} = min(b, b_{max})$. When the number of active sources increases, the base station should broadcast a large group size b to track the offer load to optimal offer load G^*. Under such situation, the error of estimation of N_2 is also increased. This implies the estimation of offer load is less precise than that without limited polling group size.

4 Conclusions

Based on GRAP, in this paper we proposed various schemes to implement the idea of re-sampling for random access in a wireless environment. It is shown in our experiments that using carrier sensing ability to collect the information of number of marked groups N_2, one can track the offer load to the optimal offer load G^*. This results in high throughput and fast adaptive speed.

To ease implementation, we proposed GRAP with binary backoff, where the base station only needs to broadcast a much smaller set of numbers. To further reduce the delay caused by marking cycle, we proposed GRAP with a limited polling group size. From our simulations, the performance of GRAP with a limited polling group size is slightly lower than that of GRAP with resampling, but the difference is small. Thus, GRAP with a limited polling group size might be of practical importance.

References

[1] N. Abramson, "The ALOHA System- another alternative for computer communications," *Proc. FJCC*, pp. 281-285, 1970.

[2] N. Abramson, "Packet Switching and Satellites," *Proc. NCC,* pp. 695-702, 1973.

[3] C.S. Chang, J.F. Chang, K.C. Chen and M.Y. You, "Guaranteed Quality-of-Service Wireless Access to ATM Networks", accepted by *IEEE Journal on Selected Areas of Communications*, 1996.

[4] K.C. Chen, "Medium Access Control of Wireless Local Area Networks for Mobile Computing", *IEEE Network*, Sep./Oct., 1994.

[5] K.C. Chen and C.H. Lee, "Group randomly addressed polling for multi-access wireless data networks," *IEEE International Conference on Communications*, New Orleans, pp. 913-917, 1994. "An updated version of GRAP," *IEEE Documentation P802.11/93-106.*

[6] A.S. Tanenbaum, *Computer Networks*. New Jersey: Prentice-Hall, Inc. 1981.

[7] *Carrier Sense Multiple Access with Collision Detection (CSMA/CD), IEEE Std 802.3-1985 (ISO/DIS 8802/3).*

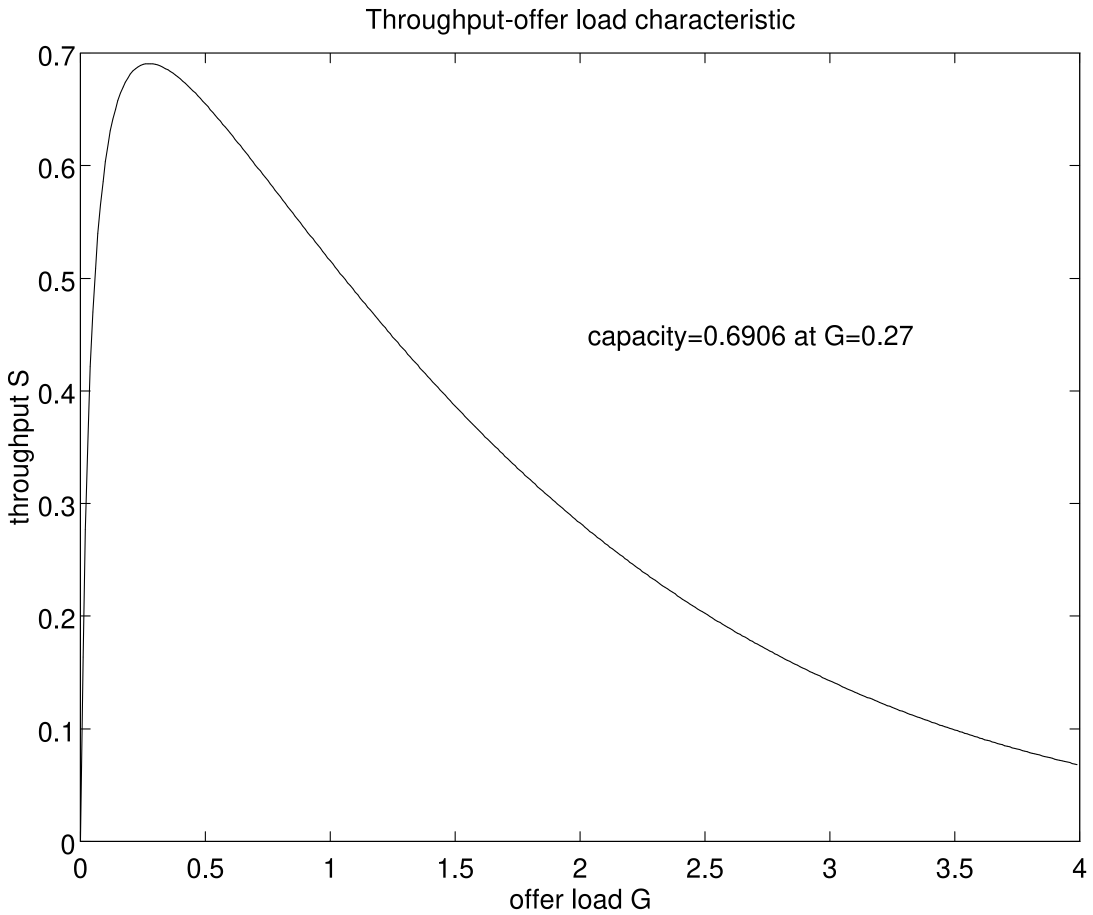

Figure 1: Throughput-offer load characteristic

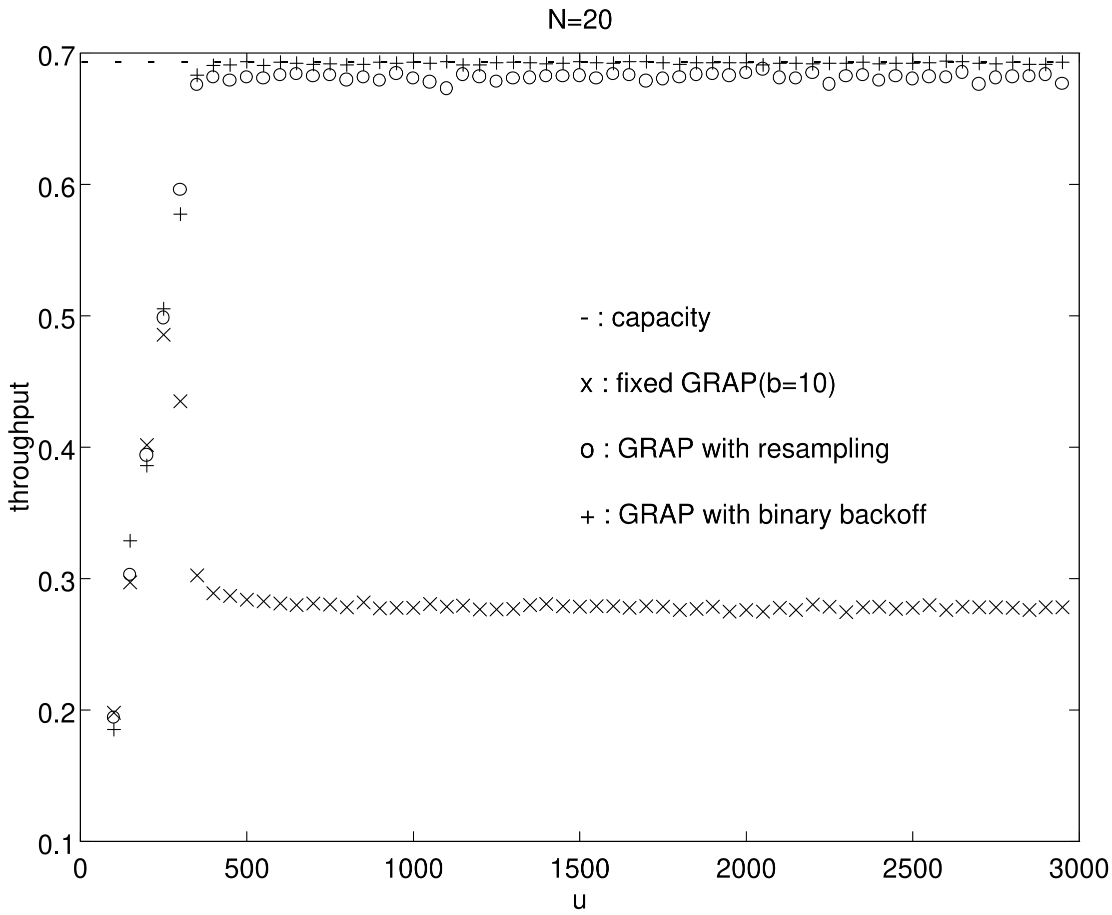

Figure 2: Throughput of resampling and binary backoff with 20 sources

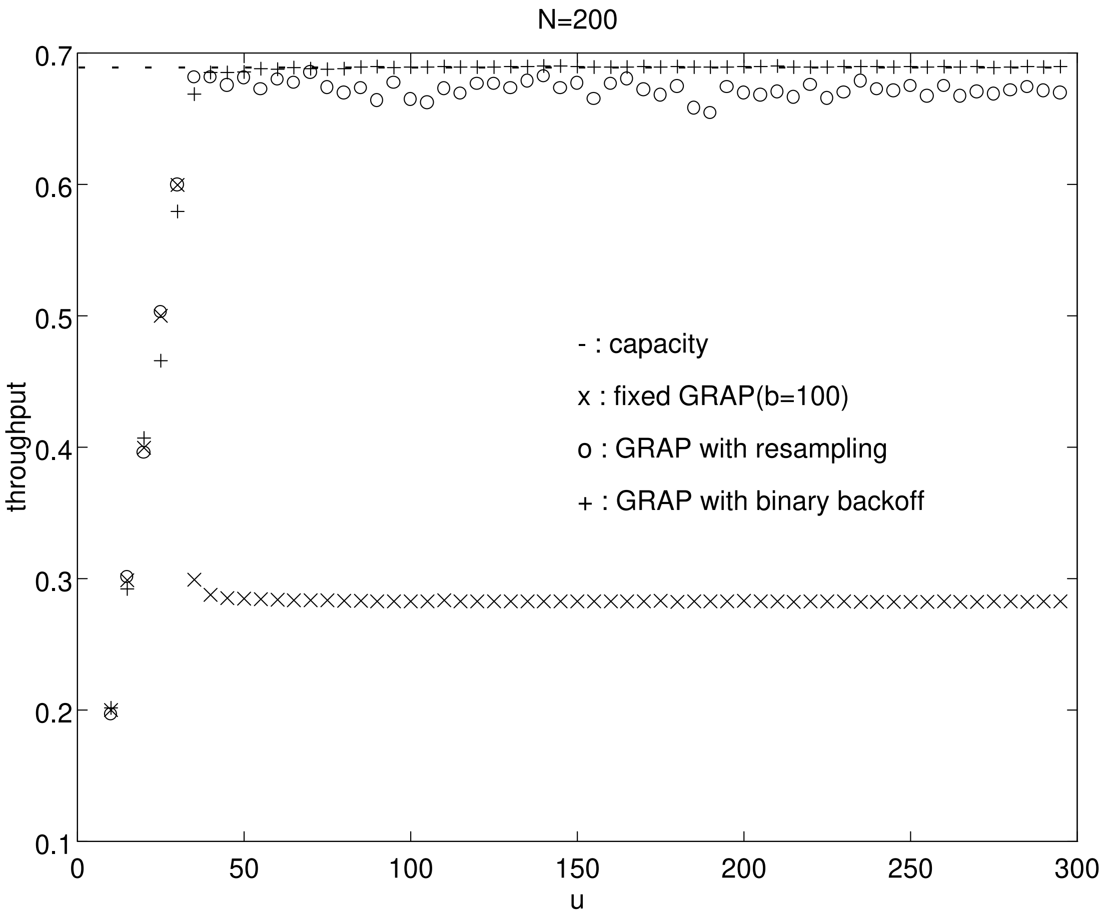

Figure 3: Throughput of resampling and binary backoff with 200 sources

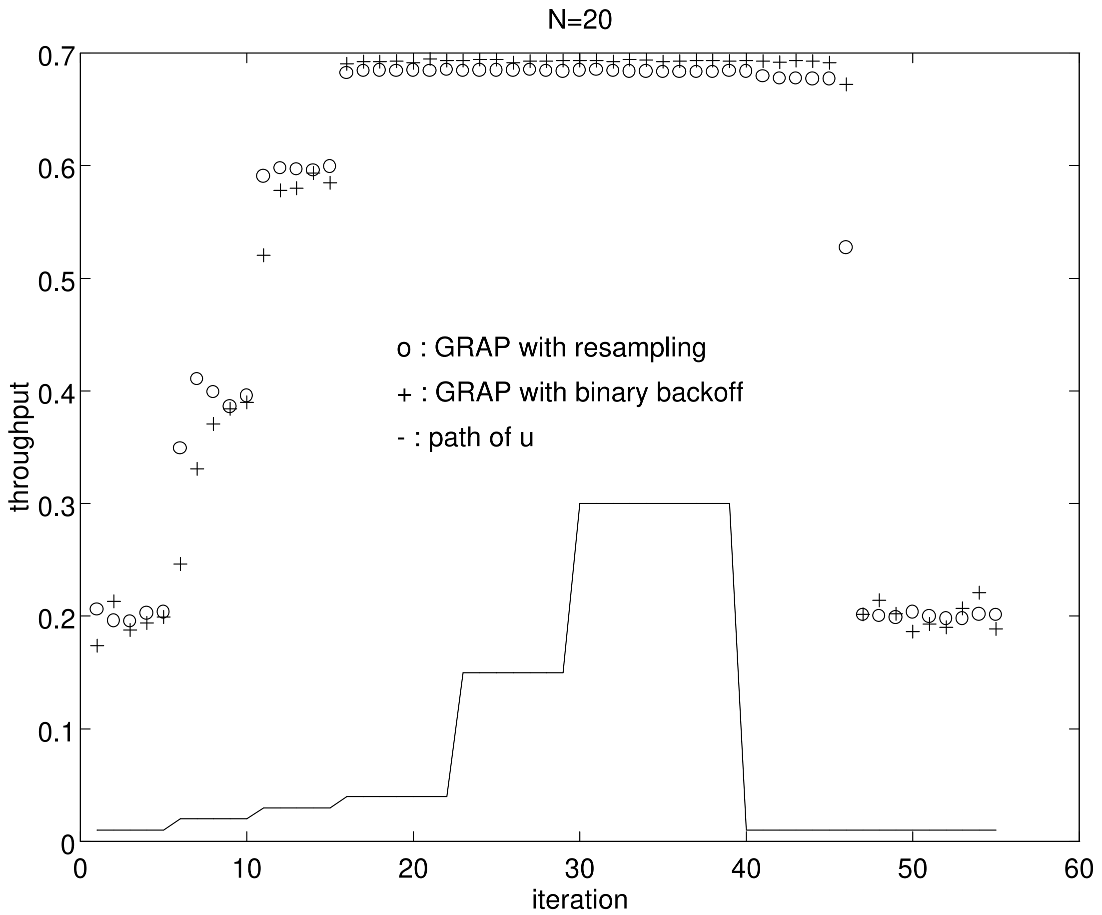

Figure 4: Adaptive speed of resampling and binary backoff with 20 sources

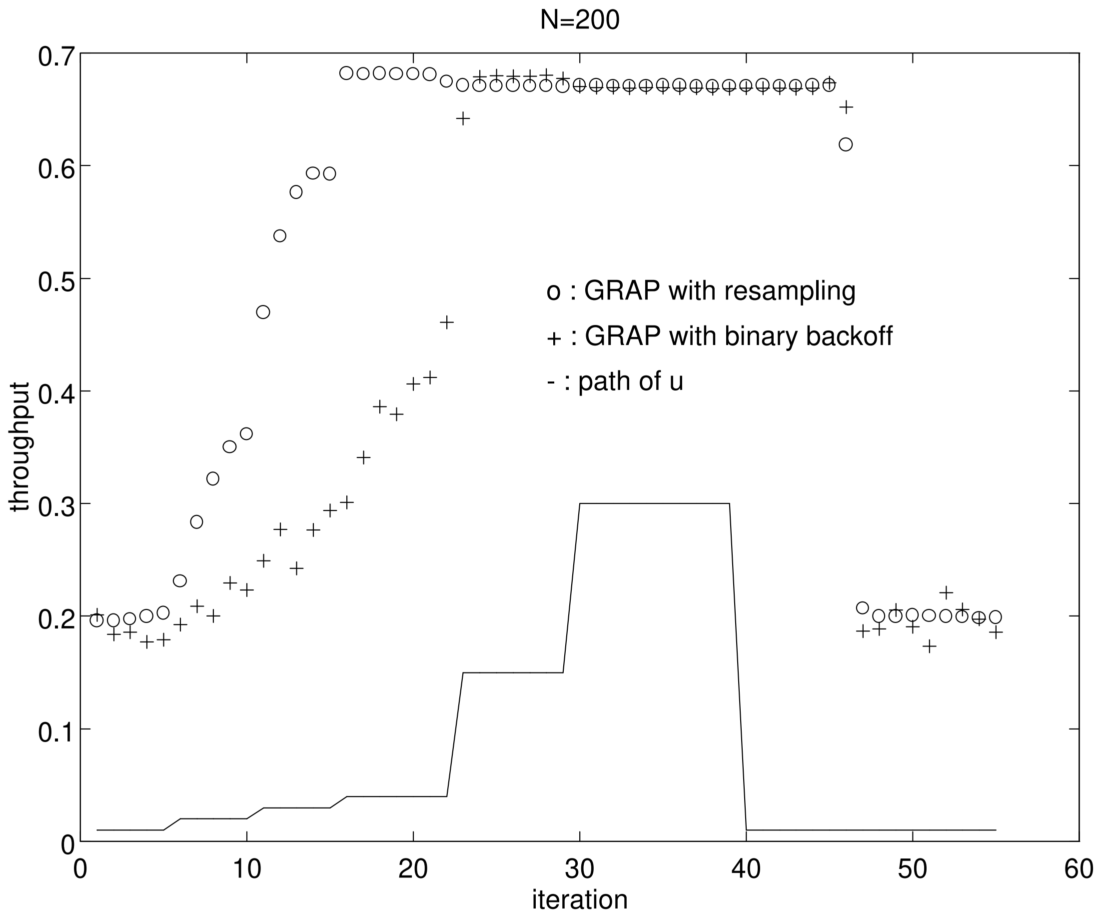

Figure 5: Adaptive speed of resampling and binary backoff with 200 sources

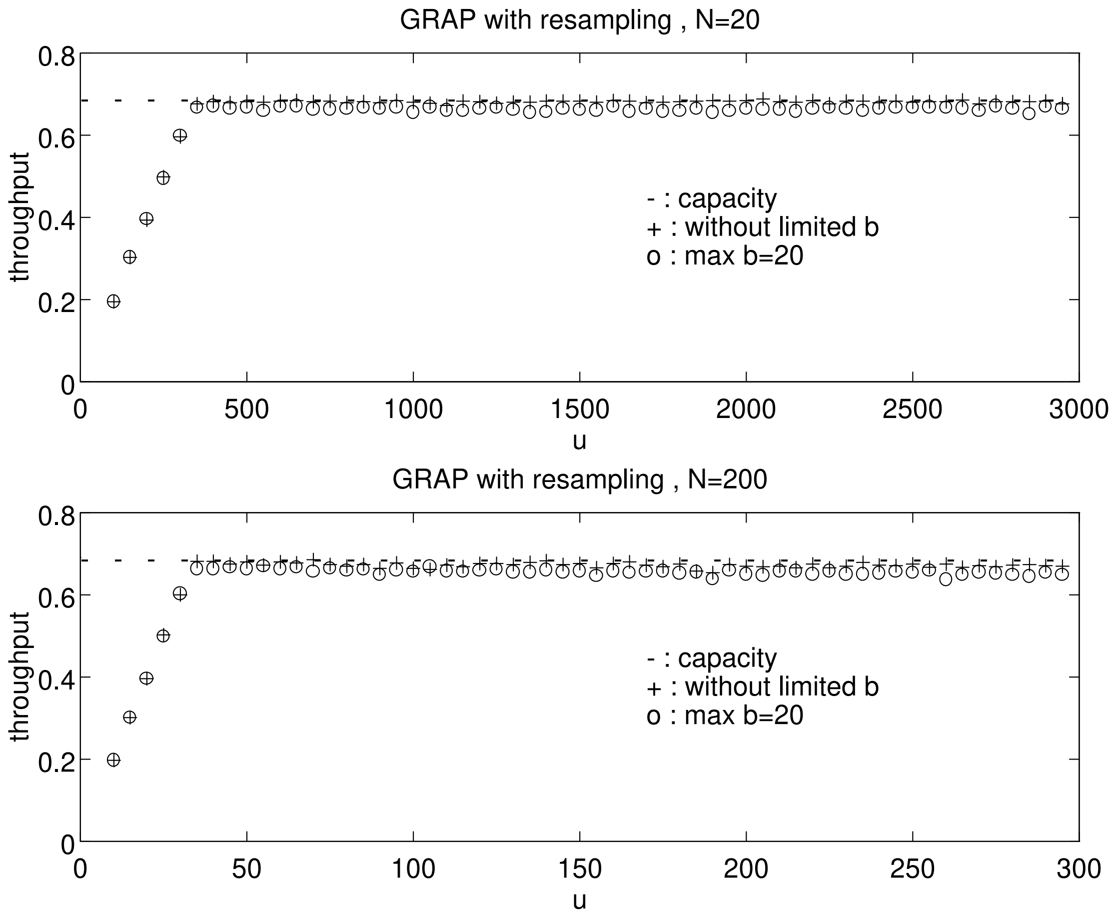

Figure 6: Throughput of GRAP with a limited polling group size

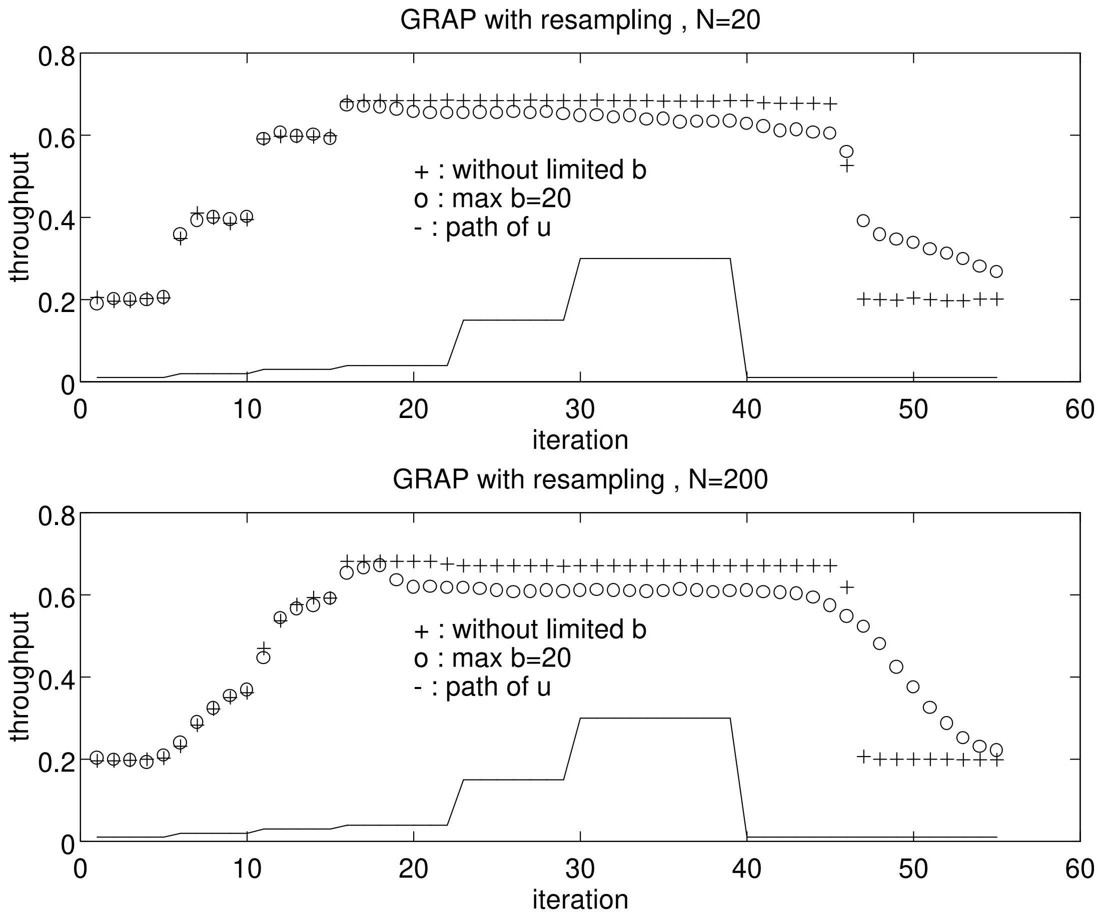

Figure 7: Adaptive speed for GRAP with a limited polling group size