This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2007 proceedings.

To Repair or Not To Repair: Helping Ad Hoc Routing Protocols to Distinguish Mobility from Congestion

Manoj Pandey, Roger Pack, Lei Wang, Qiuyi Duan and Daniel Zappala
Computer Science Department
Brigham Young University
Provo, UT 84602
Email: {manoj, rdp, lei, qiuyi, zappala}@cs.byu.edu

***Abstract*—In this paper we consider the problem of distinguishing whether frame loss at the MAC layer has occurred due to mobility or congestion. Most ad hoc routing protocols make the faulty assumption that all frame loss means the destination node has moved, resulting in significant overhead as they initiate the repair of routes that have not been broken. We design a mobility detection algorithm, MDA, that properly detects the cause of a lost frame, then coordinates with the routing protocol so that it reacts properly. This approach dramatically reduces routing protocol overhead and significantly increases application throughput. We use a simulation study to demonstrate the effectiveness of MDA and to determine the proper setting for MDA parameters.**

I. Introduction

One of the difficult problems in mobile ad hoc networks involves distinguishing whether frame loss at the MAC layer has occurred due to mobility or congestion, and then having the transport and routing protocols react properly. Frame loss occurs when the MAC layer is unable to transmit a frame from one node to another. In current IEEE 802.11 standards, a node will try to send a frame up to 7 times for smaller frames and 4 times for larger frames before deciding the transmission can't be completed.

The difficulty lies in the fact that from the perspective of a wireless node, the exact cause of the dropped frame cannot be determined. If the destination node has moved, then clearly the current route is broken. In this case, the routing protocol should repair the route, and in the meantime the transport protocol should stop transmitting. Once the route is repaired, the transport protocol may need to restart its congestion control algorithm to determine the appropriate sending rate for the new path. On the other hand, if the loss is due to congestion, then the routing protocol should do nothing. Instead, the transport protocol should adapt to the proper sending rate and minimize lost frames.[1]

Unfortunately, most ad hoc routing protocols have a major design flaw in that they react to all frame loss as a sign of mobility, without any regard to congestion. Both AODV [1] and DSR [2] assume that a single dropped frame is a sign that all paths using the MAC layer destination as a next hop have failed. Similar behavior exists with routing protocols that use passive ACKs or HELLO messages [3] to determine link availability. Congestion can cause these messages to be lost, fooling the routing protocol into thinking the route is broken. This design flaw is even present in multicast routing protocols; ADMR [4] assumes that the sender transmits at a constant rate and infers that a route is broken if a number of consecutive packets are lost.

The costs incurred due to this design flaw are significant because finding a new route involves flooding the network to some extent. For example, whenever it determines that a route has failed, AODV drops all packets queued for that next hop and uses flooding to try to repair the route. The price of this mistake is increased routing protocol overhead, leading to reduced application throughput. In this paper we demonstrate that this is a significant concern, with routing overhead consuming as much as 700 Kbps in a 2 Mbps wireless network.

The key question then is how can a wireless node properly determine the cause of a lost frame? In this paper, we solve this problem by designing a cross-layer Mobility Detection Algorithm, MDA, that uses MAC-layer statistics to distinguish between mobility and congestion-based losses. Using MDA can significantly reduce routing overhead, leading to corresponding increases in throughput.

One of the primary advantages of MDA is that it is completely independent of the routing protocol. We are able to use MDA to significantly improve the performance of both AODV and DSR. Because of its independence, MDA is easily adapted to work with any kind of link availability mechanism. We report results showing that our performance improvements for AODV hold whether it uses packet loss, HELLO messages, passive acknowledgments, or network-layer acknowledgments to determine whether a route is alive.

To demonstrate the advantages of MDA, we conduct a simulation study that examines routing protocol performance in conjunction with a congestion-controlled transport protocol.

[1] If the achievable rate for the current path is too low, then QoS routing could be used to carefully route different flows. However, even with QoS routing, the transport protocol must be given a chance to converge before the system can determine whether an alternate path is needed.

0743-166X/07/$25.00 ©2007 IEEE

This study is unique in that it is the first to show the impact of routing overhead on application throughput. Prior studies of routing protocols send constant bit rate (CBR) traffic, usually only a small number of packets per second, and measure just the percentage of packets delivered correctly. We use ATP for a transport protocol because it has been designed specifically to overcome some of the shortcomings of TCP in an ad hoc wireless network [5].

Our results show that MDA successfully differentiates between frame loss due to mobility versus loss due to congestion. As a result, the routing protocol only rebuilds routes when the current one has truly been broken, significantly lowering routing overhead. In a scenario dominated by congestion, MDA improves ATP throughput by 50 to 100%, with similar gains for CBR flows. In a scenario dominated by mobility, MDA properly notifies the routing protocol, with no negative impact on application throughput. In more mixed scenarios, MDA improves ATP throughput by 10-100%, depending on the routing protocol. We also evaluate appropriate settings for MDA parameters and show that its behavior is very close to that of an omniscient routing protocol.

```
1  credibility = threshold;
2  if transmission failure to node X then
3      if received CTS from X during attempt then
4          credibility = 0;
5      else
6          timer.set(t, X);
7          if credibility == threshold then
8              notify routing protocol node X has moved;
9          end
10     end
11 end
12 if timer for node Y expires then
13     credibility = min(threshold, credibility++);
14 end
15 if hear from any node Z then
16     if timer.isset(Z) then
17         timer.cancel(Z);
18         credibility = 0;
19     end
20 end
```

Fig. 1. Mobility Detection Algorithm

II. Mobility Detection Algorithm

MDA is based on the observation that proving a node has moved is not possible, but it is easy to conclude that a node has *not* moved. MDA listens to all frame transmissions in the area, and if it hears from a node it can conclusively determine that this node has not moved. MDA is run locally on each node and does not require any communication or cooperation among nodes.

The complete MDA algorithm is shown in Figure 1. The main state MDA keeps at each node is a credibility value, initially set to *threshold*, typically 1 or 2. High credibility means that MDA assumes the loss is due to mobility and it notifies the routing protocol so that it can find a new route. Any credibility below the *threshold* indicates the loss is due to congestion, in which case the routing protocol is not notified.

Credibility is increased or decreased based on observations of the wireless medium during and after a failed transmission. If any CTS is received during a failed transmission, then that neighbor must still be within transmission range; this causes credibility to be immediately set to zero. Otherwise, MDA starts a credibility observation on this suspect neighbor by starting a timer. If the timer for a node expires before MDA observes a frame from this neighbor, then the neighbor is assumed to have moved, and credibility is increased. If MDA hears from a neighbor with an outstanding timer, then it sets the credibility to zero cancels the timer.

One of the main decisions we faced in designing MDA was whether to keep a separate credibility value for each destination a node communicates with. After extensive testing, we decided to keep a single credibility value for all frames sent by the node, regardless of the destination. This is particularly effective for detecting congestion-induced losses; in a wireless medium, if any frame is dropped due to congestion then it is highly likely that subsequent lost frames are also due to congestion.

The performance of MDA depends on the setting of two parameters – the *threshold* and the duration of the timer. If the *threshold* is set too high, MDA will not react quickly enough to mobility based losses. However, a very small *threshold* may cause misinterpreted losses, and hence incorrect repair responses. The timer value should be set fairly high for two reasons. First, if congestion is low, a node may simply be sending at a slow rate. Second, during congestion it will be difficult to hear from a node to determine whether it has moved. However, a very large timer will make it difficult for MDA to react quickly to route failures, since the worst-case reaction time is equivalent to the threshold times the timer value. We evaluate a wide range of timer settings in our simulations to determine the appropriate value.

Our initial simulations of MDA focused on the performance of MDA in purely mobile conditions (without congestion) and highly congestion conditions (without mobility). These simulations enabled us to confirm that MDA correctly identifies the cause of frame loss in a situation when we know what the correct answer should be. When mobility is high and congestion is low, broken routes are detected quickly because congestion will not have reduced the credibility value. During conditions of high congestion, losses will always be reported as due to congestion, even though a node may have moved. This is acceptable because the transport protocol should lower its sending rate first, to prevent congestion collapse. It will then be easier for MDA to determine which nodes have moved during this period, and the routing protocol can react to route changes when the network is not as heavily congested.

The rest of this paper examines the performance of MDA in more general situations and explores the correct setting for its timer and threshold parameters.

III. Methodology

Our main objective in evaluating MDA is to determine how much it can reduce routing overhead and increase application throughput. To properly evaluate throughput, we use ATP, a transport protocol with a congestion control algorithm designed especially for ad hoc wireless networks. Because ATP is new and still has some flaws, this has the effect of limiting the throughput gains that MDA can achieve. However, ATP is one of the best available choices at this time. Future transport protocols may take better advantage of the limited available bandwidth in wireless networks, and in that case MDA's performance enhancements should be even greater.

In our simulations, we use the $ns2$ simulator [6], version 2.28. For routing protocols we use the default implementations for DSR (with minor, bug-fixing changes) and AODV. Because no version of ATP was available, we implemented ATP in $ns2$ based on the original publication [5]. In our experiments all nodes communicate using IEEE 802.11 at the MAC layer, with a $250m$ radio range, free-space radio signal propagation, and a maximum data rate of 2 Mbps. We run each simulation for 600 seconds and take the average of 5 experiments for each data point.

Unless otherwise specified, MDA uses a credibility timer of 1 second. The simulations we report here use a threshold of 1 or 2, though others we have conducted use a threshold as high as 10.

In our simulations we collect the following metrics:

- **Correct Route Failure Decisions**: The percentage of correct decisions made by MDA each time a frame is lost. To calculate this metric, we use an *omniscient* version of MDA that determines, for each frame lost, whether the destination node is in range of the source node, and thus accurately ascribes the loss to either mobility or congestion. We then report the percentage of correct decisions made by the standard version of MDA.
- **Routing Overhead**: All control messages originated or forwarded by any node, converted to Kbps. In most cases we report total routing overhead, summed over the entire simulation. We also use instantaneous overhead, calculated with an Exponentially Weighted Moving Average that is updated every second. For DSR, control packets consist of ROUTE REQUEST, ROUTE REPLY, and ROUTE ERROR messages. For AODV, control packets consist of ROUTE REQUEST, ROUTE REPLY, and ROUTE ERROR messages.
- **Throughput**: We report the total throughput summed over all ATP flows in Kbps. We calculate instantaneous throughput using an EWMA that is updated every second.

IV. Differentiating Between Congestion and Mobility

To test MDA and its ability to differentiate between congestion and mobility, we generate a random topology of 100 nodes in a square field of $(1000m)^2$. We conducted extensive simulations that vary both mobility (speeds from 0 to 30 m/s) and congestion (number of senders from 0 to 100). Here we report results for the general case where all nodes move using the random waypoint mobility model, with a speed of 20 m/s and a pause time of 10 seconds. We then vary the number of ATP flows from 0 to 50.

Figures 2 and 3 show that MDA provides performance improvements for both DSR and AODV. Although MDA makes correct route failure decisions only about 60% of the time, it still reduces routing overhead and improves throughput to levels that are nearly as good as the omniscient version. Throughput gains are more modest when mobility is the dominant factor, since the routing protocol is doing the right thing even without MDA. Improvements are more evident at higher loads, with throughput gains as high as 300%.

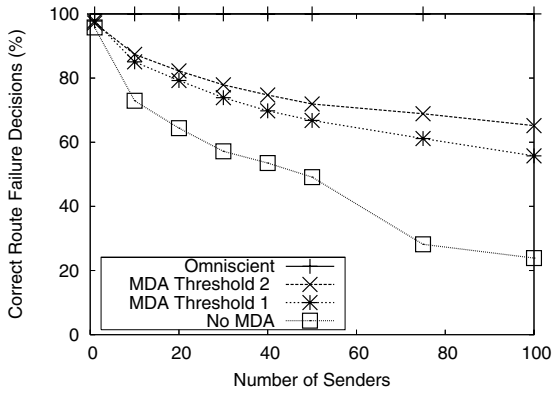
MDA's incorrect route failure decisions stem from how it handles loss due to mobility. When nodes are mobile, there will be many times when MDA's credibility metric is at the threshold, in order to react to mobility properly. At these times, MDA will make a mistake on the first frame that is lost due to congestion and assume it is instead due to mobility. The higher MDA's threshold, the better it will properly categorize congestion losses, but the longer it will take to recognize mobility. Likewise, there will be times when MDA's credibility is low due to congestion, causing it to make a mistake on packets lost due to mobility. As shown in these results, MDA's imperfection does not have a significant impact on system performance.

We achieve similar results for constant-rate flows in the random scenario. As congestion increases in the network, both DSR and AODV make correct route failure decisions only about 5 to 20% of the time. MDA makes the correct decision about 60% of the time. This ability to make an informed decision significantly decreases routing overhead, but provides only a small gain in the packet delivery ratio. This is because a flow that is not using congestion control can artificially increase its packet delivery ratio by sending as fast as it can, while ignoring packet loss.

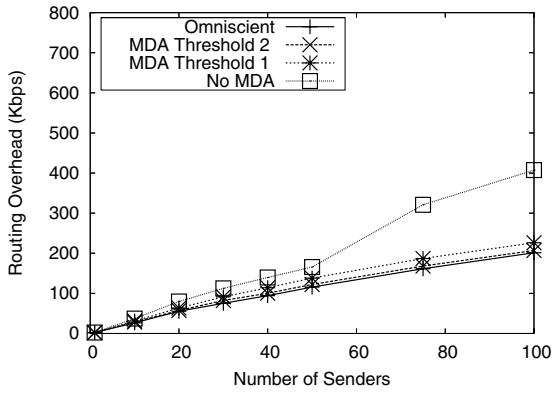
V. MDA Timer Setting

To determine the appropriate setting for MDA's timer, we repeated the random scenario used in the previous section with a range of timer settings. Our results, shown in Figure 4, indicate that MDA needs a timer larger than 1 second, otherwise it frequently and incorrectly notifies the routing protocol of a route failure. This substantially increases routing overhead, particularly for AODV, and correspondingly reduces throughput. On the other hand, a very large timer value prevents the routing protocol from reacting to legitimate route failures, which of course lowers routing overhead. This also increases overall throughput because fewer flows are competing for the available bandwidth.

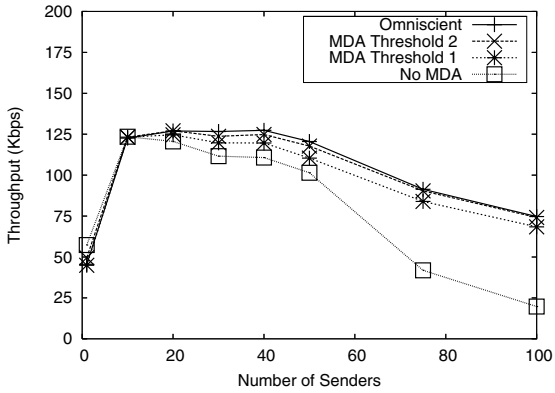
These results indicate that our timer setting of 1 second works well. If desired, a shorter timer of $\frac{1}{3}$ or $\frac{1}{2}$ second could reduce the latency required to react to route changes, while sacrificing some of the improvement in routing overhead and throughput.



(a) Correct Route Failure Decisions

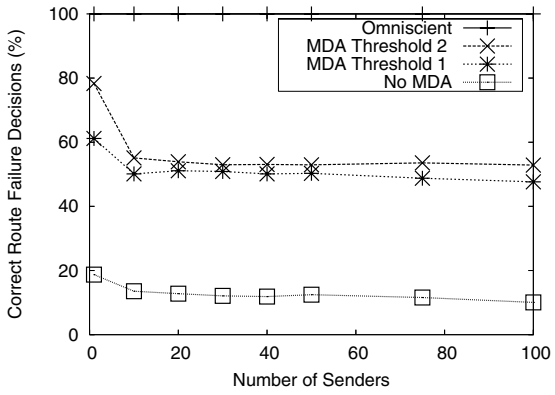


(b) Routing Overhead

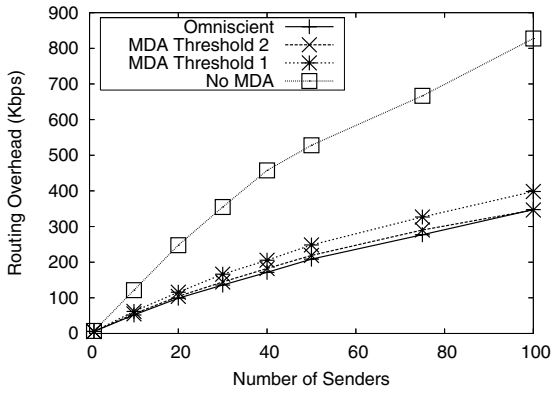


(c) Throughput

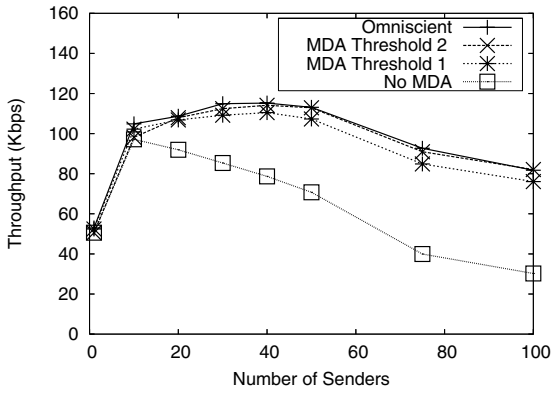
Fig. 2. Random Scenario with ATP flows over DSR



(a) Correct Route Failure Decisions



(b) Routing Overhead



(c) Throughput

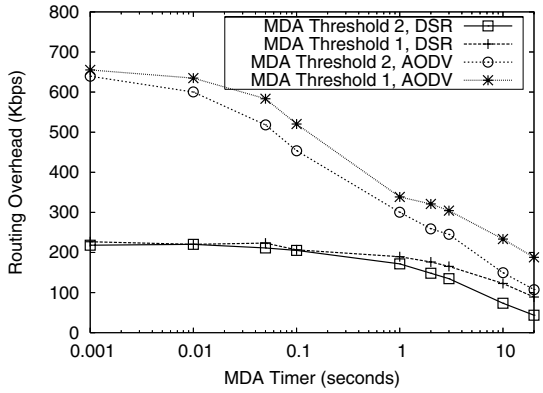
Fig. 3. Random Scenario with ATP flows over AODV

VI. Cross-Layer Architecture

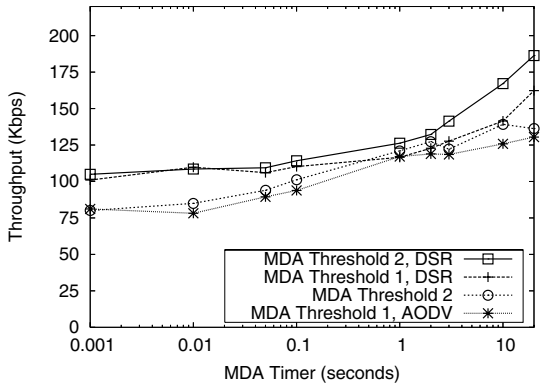
To demonstrate the utility of MDA as part of a cross-layer architecture, we implemented several additional route failure detection mechanisms for AODV in *ns2*. The mechanisms we use include HELLO messages, passive acknowledgments, and explicit network-layer acknowledgments. We then repeated our above experiments with each of these mechanisms. While some mechanisms, such as HELLO messages, lead to substantially longer latency in detecting a failed route, in most cases MDA was able to reduce routing overhead and increase throughput, as in our previous results. To avoid repetition, we do not include the detailed results here.

VII. Related Work

One alternative to MDA is to use signal strength measurements to determine whether a node has moved [7], [8]. In this work, a node may initiate a search for an alternate path when a next hop along the current path begins to move out of range, as determined by signal strength. One of the complications is that the mechanism must compensate for variations in signal strength due to fading, multipath effects, or power conservation



(a) Routing Overhead



(b) Throughput

Fig. 4. Mobile Random Scenario with ATP flows

mechanisms. This approach is shown to significantly decrease the number of broken paths and to improve latency for TCP.

While our work focuses on routing protocols, there has also been a great deal of work on helping transport protocols react properly to lost frames. Most of the work in this area takes an end-to-end approach to solving this problem and does not interact directly with the MAC layer at each node. For example, ATP [5] ignores dropped frames as a sign of congestion, and the sender instead collects explicit rate information from each node along its path to the destination. One enhancement of TCP [9] uses out of order packets detected by the receiver as a sign of a route change caused by mobility, allowing TCP to respond differently to these losses. TCP Casablanca [10] assigns priorities to packets, and assumes that lower priority packets are dropped first. A receiver can then use the dropping pattern that it observes to determine when loss is due to congestion and when it is due to mobility. Cen et al. explore a range of end-to-end mechanisms for distinguishing congestion from other types of packet loss [11]. As an exception to this approach, TCP-ELFN [12], [13] interacts directly with the routing protocol. When TCP-ELFN receives a route failure notification from the routing protocol, it freezes its congestion window and its state and then restarts once a new route has been installed.

VIII. Conclusion

In this paper we have demonstrated the benefits of using a cross-layer Mobility Detection Algorithm to determine whether a lost frame in a wireless network is due to mobility or congestion. By determining when a lost frame is truly a sign of a route failure, MDA significantly reduces routing overhead and can increase throughput by 10 to 100%, depending on the routing protocol and the mobility scenario. MDA can be used with any mobile ad hoc routing protocol, even those that use passive acknowledgments or HELLO messages to maintain routes.

We plan to work in several additional related areas. First, we plan to test MDA with other congestion control algorithms. Many of our results show a dramatic decrease in routing overhead, with significant but more modest gains in throughput. It is possible that a different congestion control algorithm may be able to take better advantage of the reduce routing overhead afforded by MDA. Second, MDA currently considers only mobility and congestion as sources of loss. Frames may also be lost due to interference from other technologies such as Bluetooth, microwave ovens, and cordless phones. A complete architecture should also detect interference from these sources and use alternative frequencies or other methods to avoid the interference when possible. Finally, we are working on adapting MDA so that it can also be used to help transport protocols distinguish between mobility and congestion.

References

[1] C. Perkins and E. Royer, "Ad Hoc On Demand Distance Vector (AODV) Algorithm," in *IEEE Workshop on Mobile Computing Systems and Applications*, 1999.

[2] D. Johnson and D. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," in *Mobile Computing*, Imielinski and Korth, Eds. Kluwer Academic Publishers, 1996, vol. 353.

[3] S. Murthy and J. J. Garcia-Luna-Aceves, "An Efficient Routing Protocol for Wireless Networks," *Mobile Networks and Applications*, vol. 1, no. 2, pp. 183–197, 1996.

[4] J. Jetcheva and D. Johnson, "Adaptive Demand-Driven Multicast Routing in Multi-Hop Wireless Ad Hoc Networks," in *ACM MobiHoc*, October 2001.

[5] K. Sundaresan, V. Anantharaman, H.-Y. Hsieh, and R. Sivakumar, "ATP: A Reliable Transport Protocol for Ad-Hoc Networks," *Mobihoc*, 2003.

[6] LBL, X. PARC, UCB, and USC/ISI, "ns Simulator, http://www.isi.edu/nsnam/ns/."

[7] D. S. P. Tom Goff, Nael B. Abu-Ghazaleh and R. Kahvecioglu, "Preemptive Routing in Ad Hoc Networks," in *ACM MOBICOM*, 2001.

[8] F. Klemm, Z. Ye, S. V. Krishnamurthy, and S. K. Tripathi, "Improving TCP Performance in Ad Hoc Networks Using Signal Strength Based Link Management," *Ad Hoc Networks*, vol. 3, no. 2, pp. 175–191, 2005.

[9] F. Wang and Y. Zhang, "Improving TCP Performance over Mobile Ad-Hoc Networks with Out-of-Order Detection and Response," in *Mobihoc*, 2002.

[10] S. Biaz and N. H. Vaidya, "De-Randomizing Congestion Losses To Improve TCP Performance over Wired-Wireless Networks," *IEEE/ACM Transactions on Networking*, vol. 13, no. 3, June 2005.

[11] S. Cen, P. Cosman, and G. Voelker, "End-to-end Differentiation of Congestion and Wireless Losses," in *Proceedings of ACM Multimedia Computing and Networking 2002.*, 2002.

[12] G. Holland and N. H. Vaidya, "Analysis of TCP Performance Over Mobile Ad Hoc Networks," in *Proceedings of IEEE/ACM MOBICOM '99*, August 1999, pp. 219–230. [Online]. Available: citeseer.ist.psu.edu/article/holland99analysis.html

[13] J. Monks, P. Sinha, and V. Bharghavan, "Limitations of TCP-ELFN for Ad Hoc Networks," in *MOMUC 2000*, 2000.