JET PROPULSION LABORATORY INTEROFFICE MEMORANDUM

SFB: 363-88-001

August 1, 1988

Time Warp Measurements on Message Times

Steven Bellenot

Recently, we (Mike DiLoreto and myself) were able to "synchronize the 2 microsecond clocks" across the Mark III Hypercube. This allows a number of accurate measurements of communication time. These statistics are the first published fruit of this instrumentation. Rather than means and standard deviations we offer a large number of graphs. We feel the shape of the graphs is more telling than averages distorted by the *noise* at the very slow (large delay) end.

Definitions

We time stamp each Time Warp message at four times during its life:

(1) As soon as it is a message. This occurs roughly just before the routine deliver is called to mail the message.

(2) Just before it is given to Mercury. This occurs just before the Mercury routine send_msg is called.

(3) When Mercury receives a message it calls the interrupt routine ioint which gives the third time stamp. (The message then goes into the Mercury receive queue.)

(4) The last time stamp (and logging of all the time stamps) is done in the Time Warp routine nq. If we waited any longer the message could be annihilated.

Using the four time stamps above we define the latencies below:

A. **Time Warp End to End Time:** as time $stamp_4$ - time $stamp_1$.
B. **Time Warp to Mercury Time**: as time $stamp_2$ - time $stamp_1$.
C. **Mercury to Mercury Time**: as time $stamp_3$ - time $stamp_2$.
D. **Mercury to Time Warp Time**: as time $stamp_4$ - time $stamp_3$.

Observe that on node messages have neither time stamp$_2$ nor time stamp$_3$. Only during the Mercury to Mercury time is the message really in transit. The B and D times are queueing delays, the time the message sits in some queue. However, the Time Warp end to end time is roughly the transit time of a message from the application's point of view.

The Two Runs of CTLS

We collected data on two runs of CTLS on TW111. Both were 32 node runs made on the cpc5 hypercube. Over 65,000 messages were send on each run with over 50,000 messages being off node messages. In most of the graphs these two runs are named "acks=2" and "acks=20". The term acks (or max_acks) is a parameter which stops the sending of a new (positive) message when the number of unacknowledged messages exceeds max_acks. However, the two runs differed in other ways, with acks=20 run also having plimit=50, reward=1 and penalty=2. The parameter plimit is how far into the Mercury queue Time Warp will look for a message with higher priority than the object it is about to run. The penalty parameter "slows" objects which send negative messages and the reward parameter undoes the "slowing".

These two runs seem typical of CTLS and (with one exception) yield close to the same graphs. The acks=20 run was first and it was conjectured (incorrectly) that reducing acks to 2 would decrease the length and time in Mercury's receive queues. However, the two runs are quite similar.

The times for GVT messages have been filtered out. (The "collects" can have a long (13 millisecond) Mercury to Mercury delay. Also the GVT printf was turned off. (This can slow the GVT node by about 50 milliseconds per GVT message). Also each off node application message has a short returning acknowledgement message which is not logged. Thus Mercury had about twice as many messages as logged.

The Graphs

The graphs in the appendices were all made with MicroSoft's Excel which has a limit of about 100 data points per series. Sometimes fewer data points were used to better show the shape of the curve. Thus some graphs appear in three pieces: the fastest 95% of the messages, the 95% to 99.9% range and the slowest 0.1%. Often the graphs appear in both linear scales and semi-log paper to better show its features. The programs mstat and msgstat (on the Sun) were used to filter the over 50,000 messages into histograms with near 100 data points. The Excel Macro File "Hist.MS" was used to help speed the graphing.

Appendices A, B, C and D contain graphs for the times A, B, C and D above. (For example Appendix C is about Mercury to Mercury Times.) In addition, Appendix E contains graphs on queueing delays and queue lengths of Mercury's receive queue.

Time Warp will sometimes unlink a message not at the front of Mercury's receive queue. Usually these messages are system messages but about 100 application messages in each run were processed ahead of earlier arriving messages. Hence the strange wording on some of the graphs in Appendix E.

Conclusions

1. The major factor in the Time Warp end to end time is the Mercury to Time Warp time, the time a message waits in the Mercury receive queue. "Most messages spent most of their time waiting on the receiving node."

2. The decrease of max acks from 20 to 2 does not seem to reduce either the Time Warp end to end time nor the Mercury to Time Warp time. However, it does increase the Time Warp to Mercury Time by increasing the number of messages with large Time Warp to Mercury times. Indeed, we claim the second peak near 7 milliseconds in graph B-2 is evidence that there is a similar peak in the execution times of object and Time Warp "pieces". (Roughly, the time it takes the code to return to the "Tester main loop" so it can try to send the message again.)

3. The application CTLS does not push Mercury to its limits. Indeed, it seems that message length rather than the number of hypercube

hops is the important factor in determining transit time. There are messages with long Mercury to Mercury delays, but they all could be behind a GVT "collection". In a GVT collection, half of the cube (16 of 32 nodes in this case) sends its reply down paths whose last hop is the channel which connects nodes one and zero. Delays on the order of a dozen milliseconds are not surprizing, even when there are no other messages in the hypercube.

4. Although Time Warp was never designed to run in "real time," it has delays which seem to be too long. (See below, "Story of a Message.") Garbage collection on the Slooow benchmark (with one object per node and a three second GVT interval) takes 60 to 80 milliseconds. The execution time of Time Warp "pieces" clearly needs to be identified and shortened.

Story of a Message

This is the history of one message which we will call "M." The message M is the right most "dot" on the graphs A-3, A-6, D-3, D-6, E-3, E-4 and E-9. M's Time Warp end to end time is 385 milliseconds of which 383 milliseconds is spent waiting in Mercury's receive queue. However, when M arrives on the receiving node there are no other messages in the Mercury receive queue. It seems that "Time Warp code" is executing during all of this time. Time Warp is busy doing "zip forward," that is it is repeating executing "go forward" for the object red_div22. The object red_div22 sends messages to itself, and lots of them are now being cancelled. There are 64 messages from red_div22 to itself which are being cancelled while message M waits in Mercury's receive queue. The irony is that M is a message for red_div22 with a receive time earlier than the send time of all but five of the 64 cancelled messages.

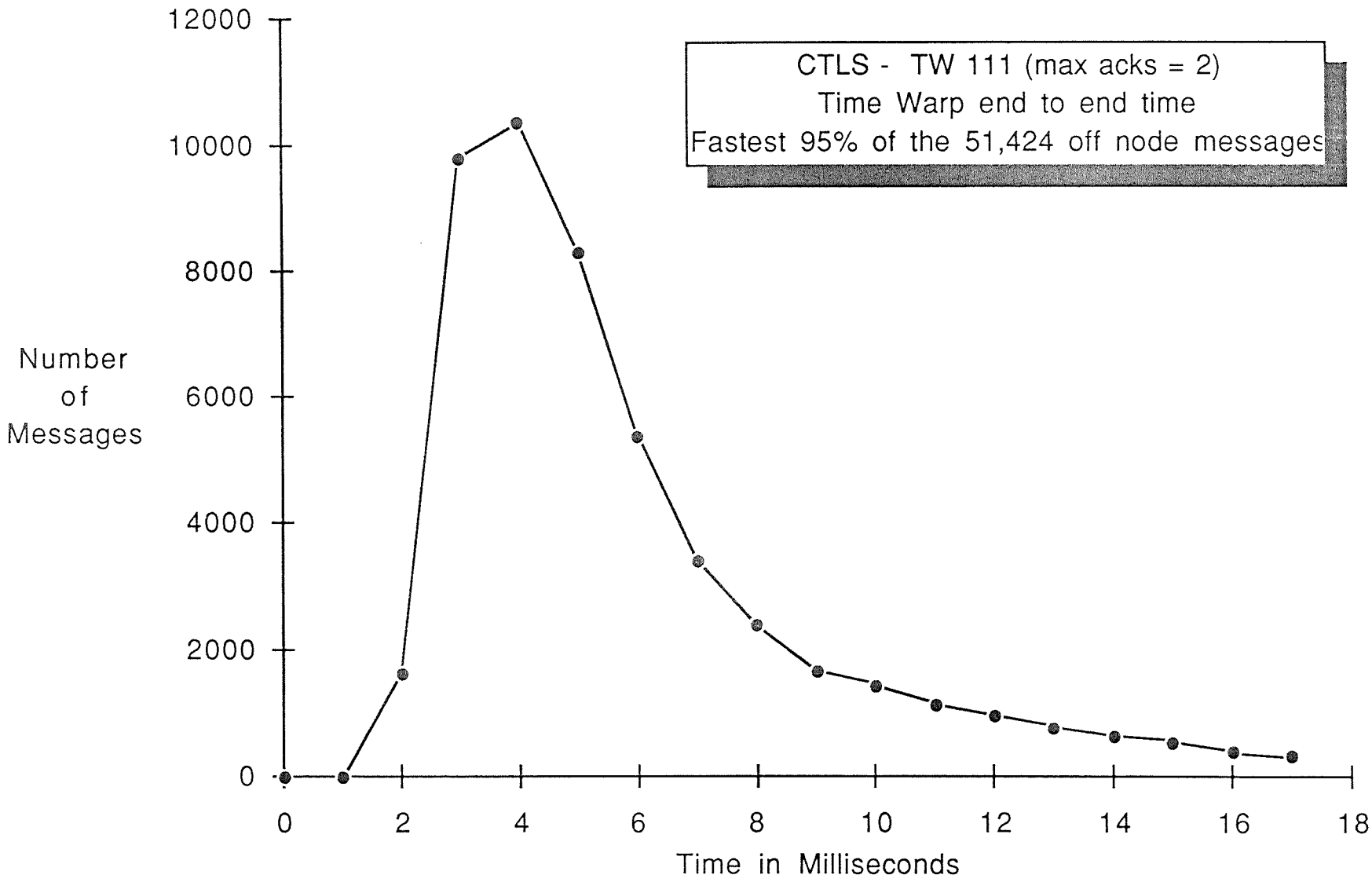
Appendix A
Time Warp End to End Times

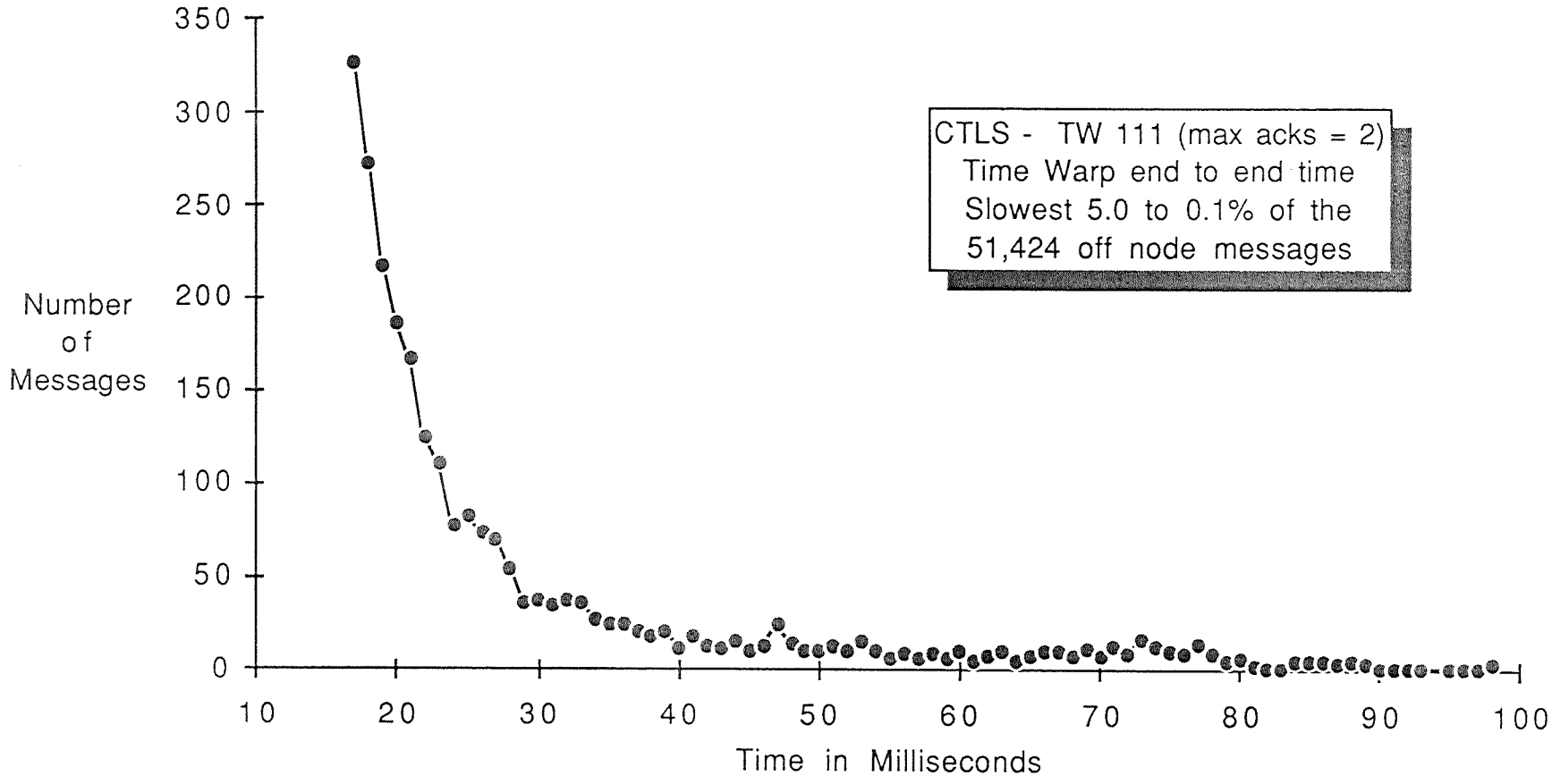
A-1: Fastest 95%, acks=2, graph (see A-4).
A-2: The 95% - 99.9% range, acks=2, graph (see A-5).
A-3: Slowest 0.1%, acks=2, graph (see A-6).

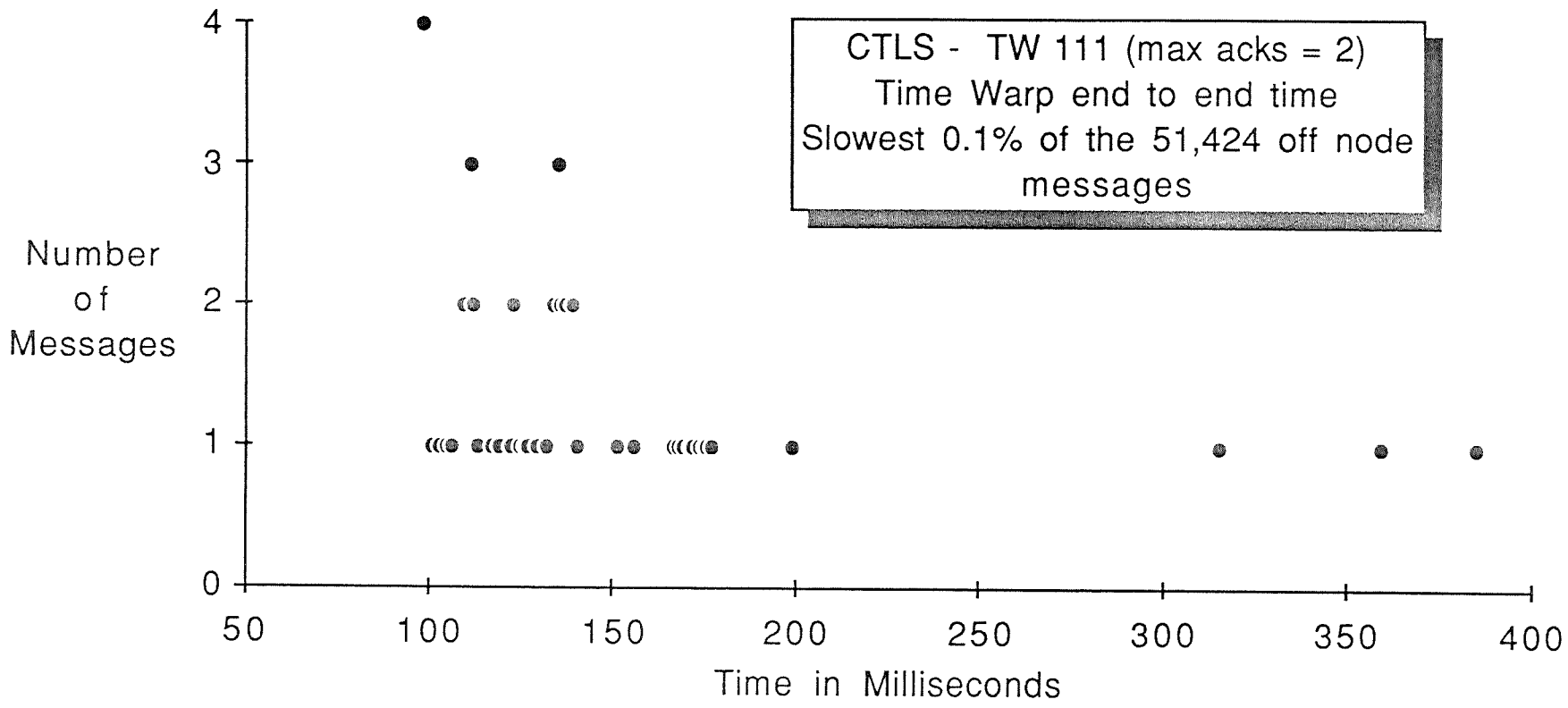
A-4: Fastest 95%, acks=2, semi-log graph (see A-1).
A-5: The 95% - 99.9% range, acks=2, semi-log graph (see A-2).
A-6: Slowest 0.1%, acks=2, semi-log graph (see A-3).

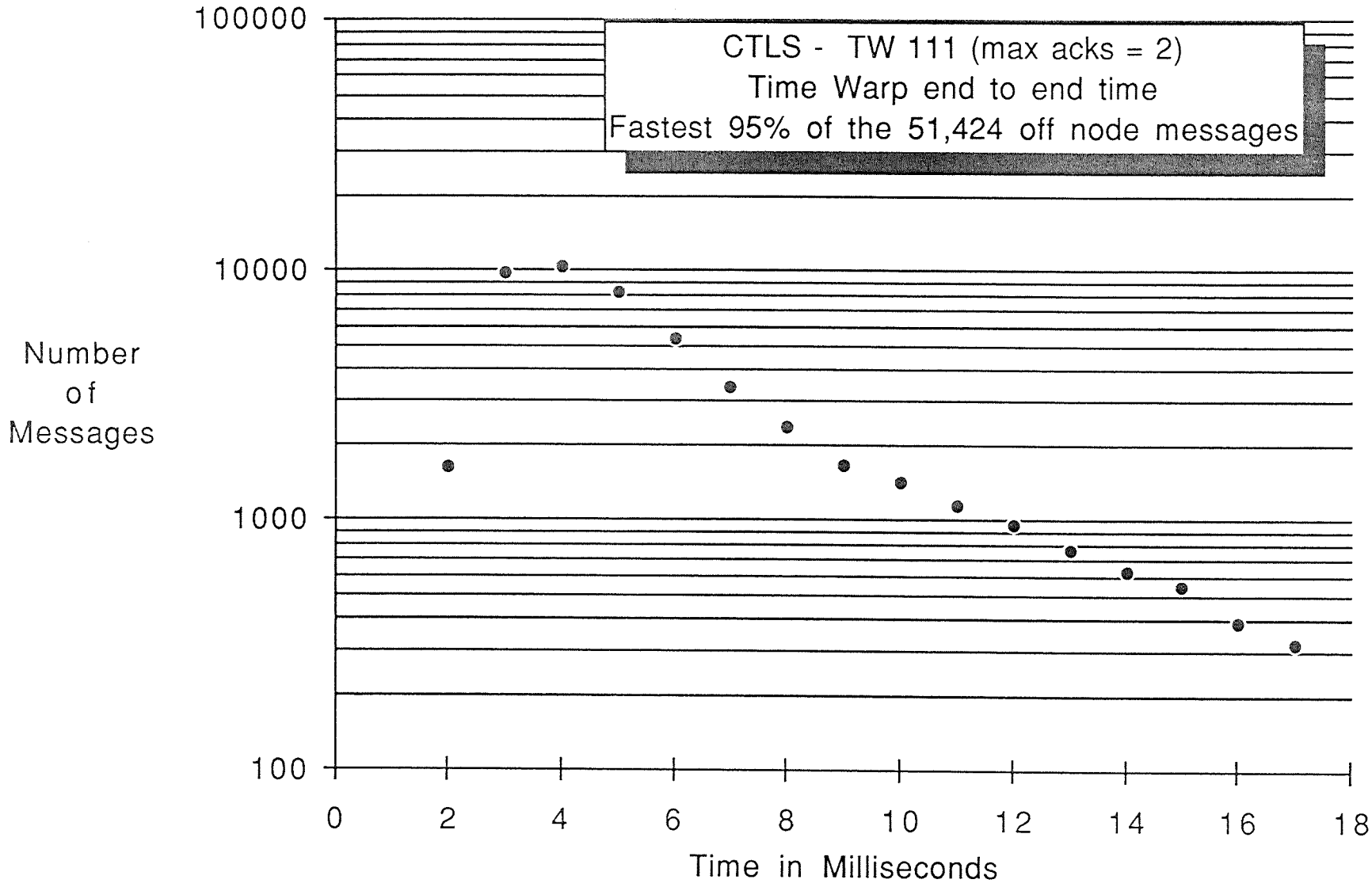
A-7: Fastest 98%, acks=2 vs acks=20, graph (see A-8, A-9).
A-8: Fastest 98%, acks=2 vs acks=20, semi-log graph (see A-7, A-9).
A-9: Fastest 98%, acks=2 vs acks=20, log-log graph (see A-7, A-8).

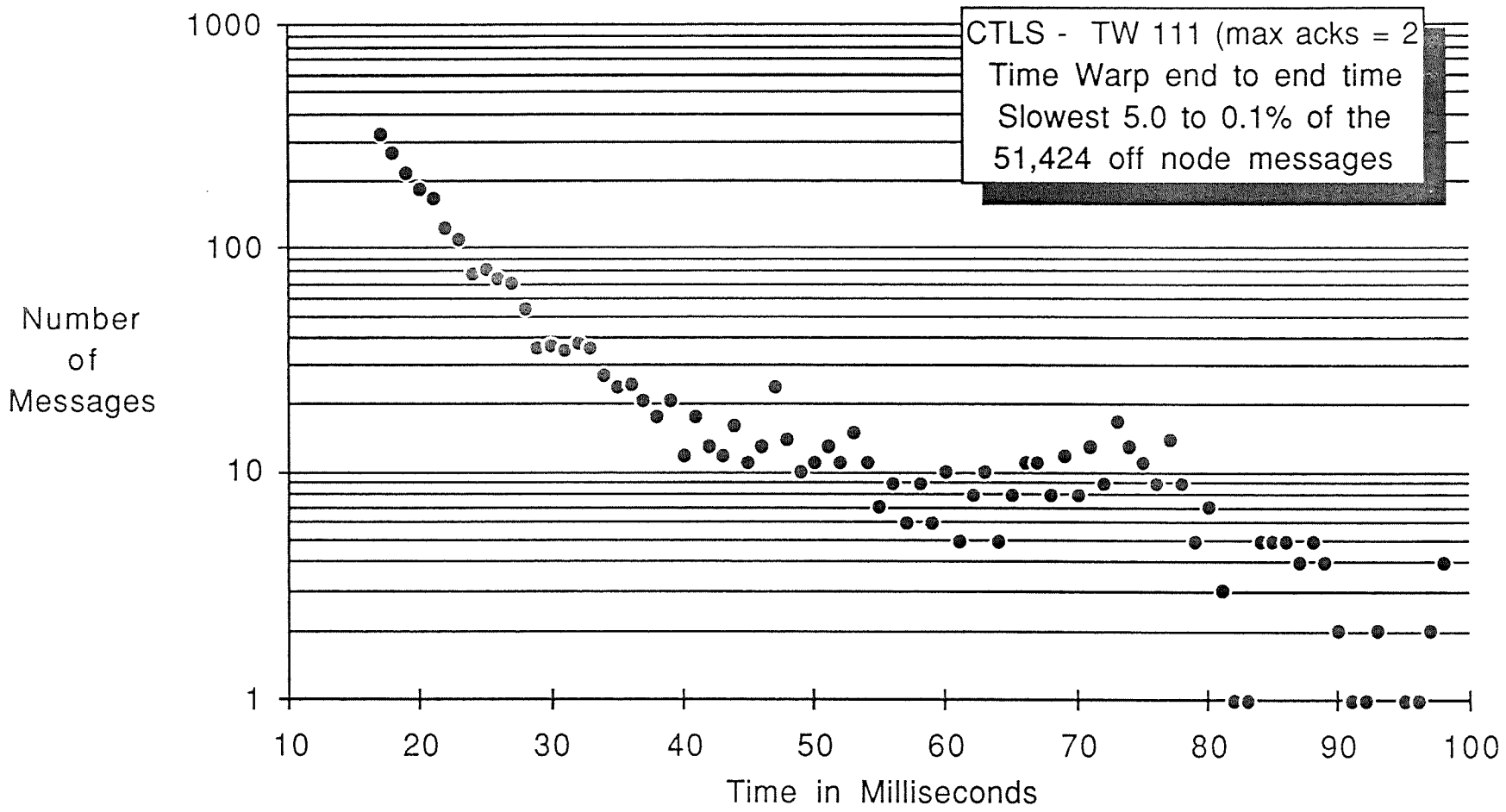
A-10:Two Queueing Delays? (see A-11).
A-11:Fastest 99.9%, acks=2, graph (see A-10).

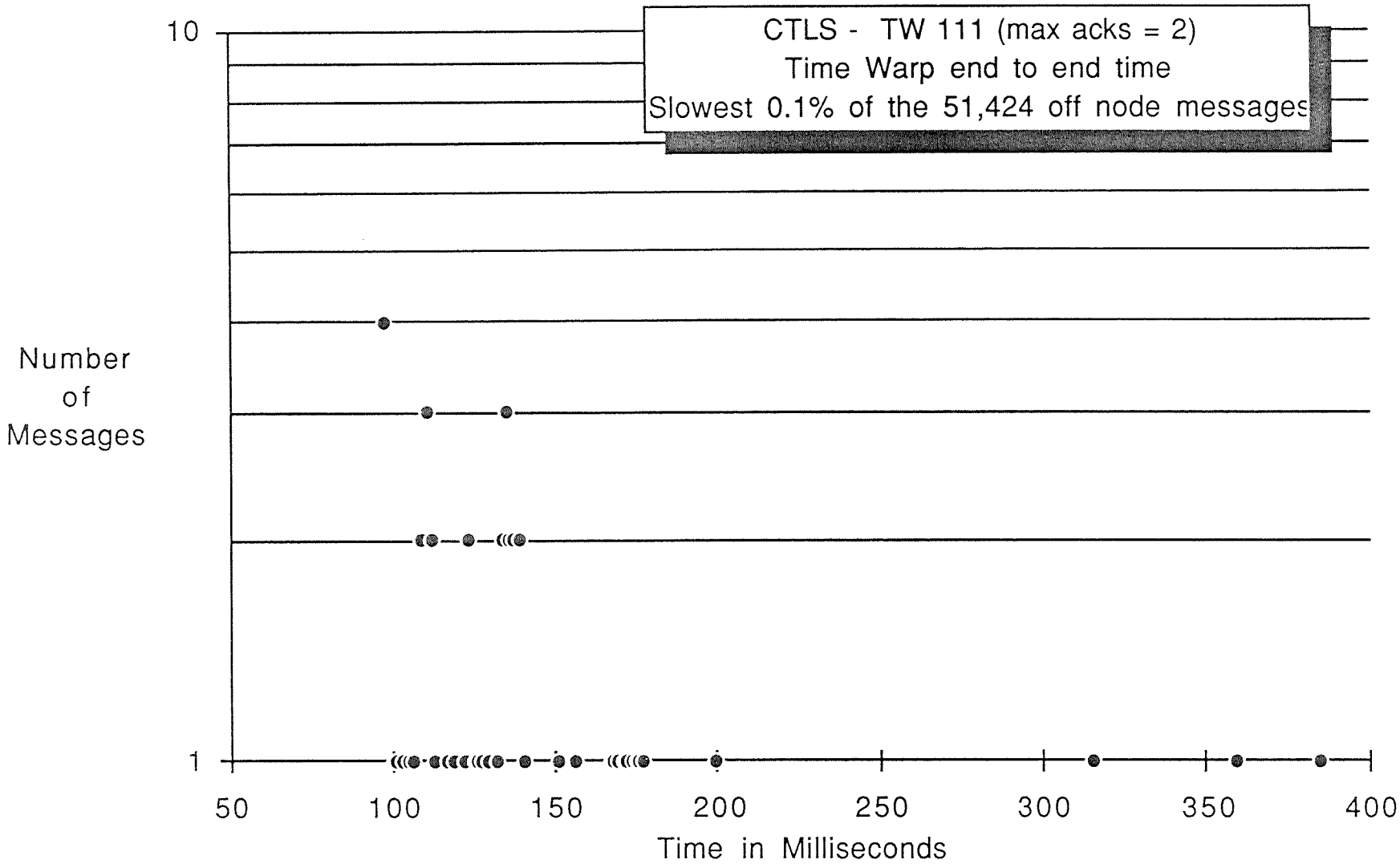

CTLS - TW 111 (max acks = 2)
Time Warp end to end time
Fastest 95% of the 51,424 off node messages
Number of Messages
12000
10000
8000
6000
4000
2000
0
0
2
4
6
8
10
12
14
16
18
Time in Milliseconds

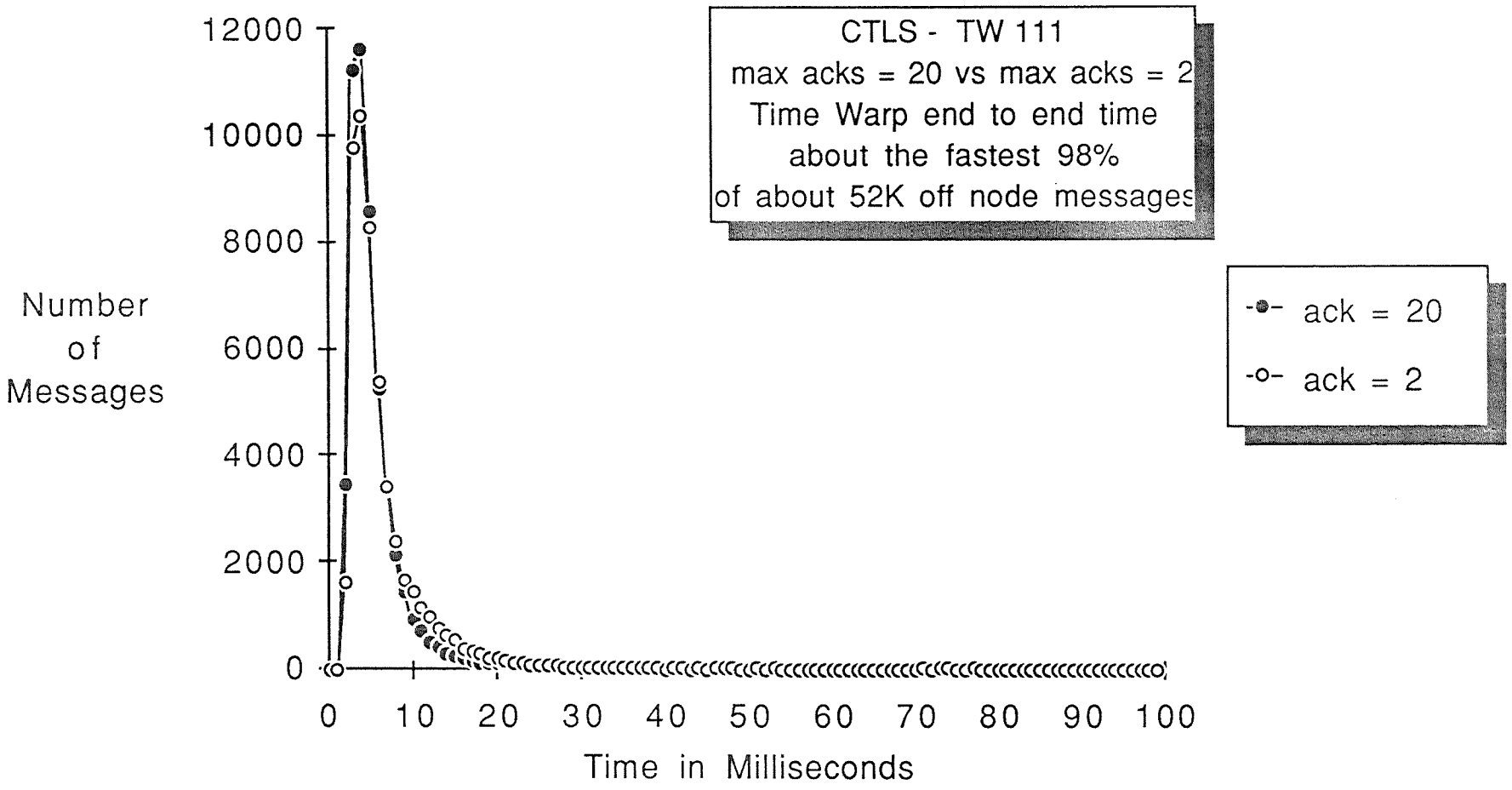

CTLS - TW 111 (max acks = 2)
Time Warp end to end time
Slowest 5.0 to 0.1% of the
51,424 off node messages
Number of Messages
350
300
250
200
150
100
50
0
10
20
30
40
50
60
70
80
90
100
Time in Milliseconds

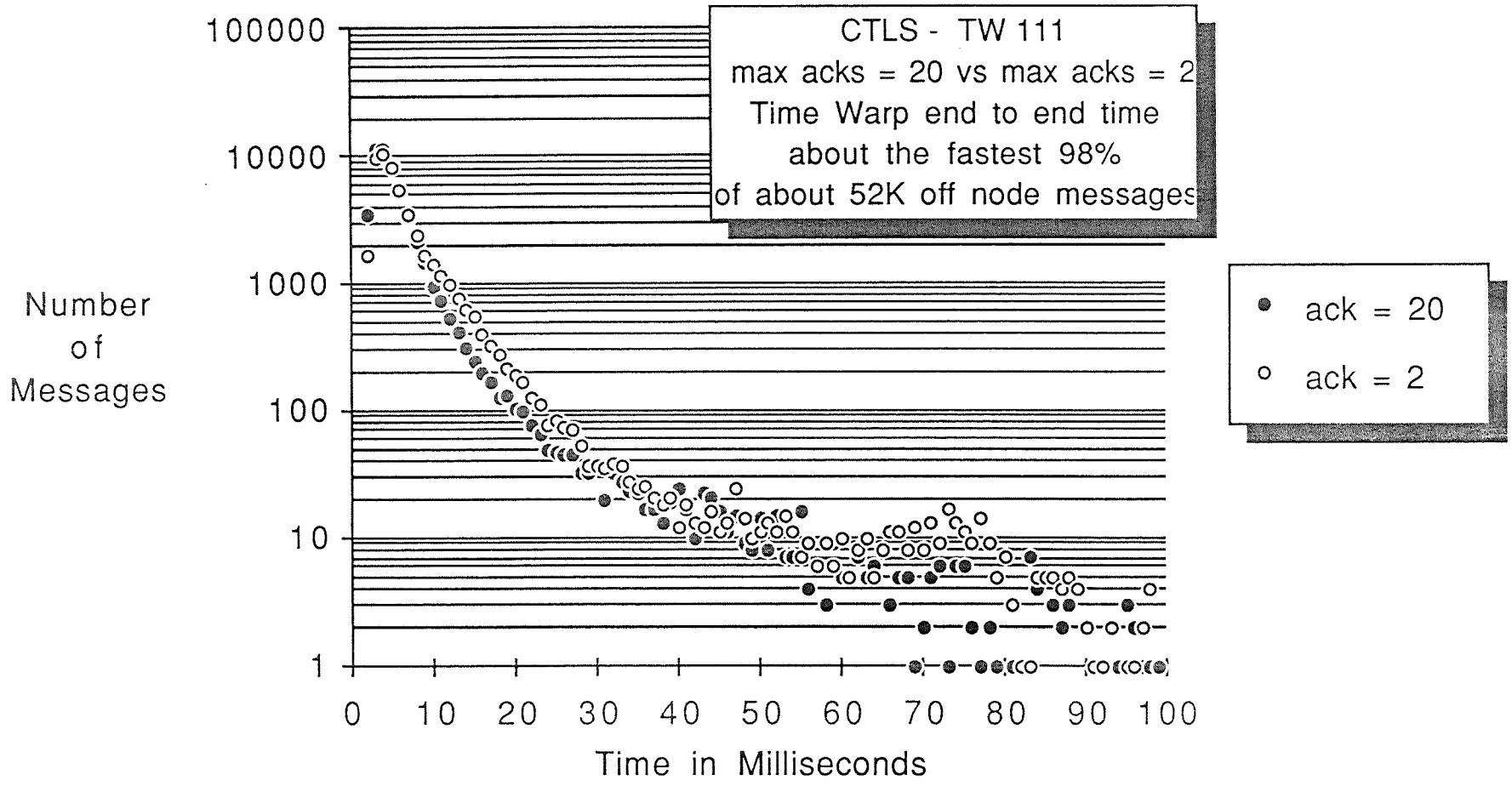

CTLS - TW 111 (max acks = 2)
Time Warp end to end time
Slowest 0.1% of the 51,424 off node messages
Number of Messages
0
1
2
3
4
50
100
150
200
250
300
350
400
Time in Milliseconds

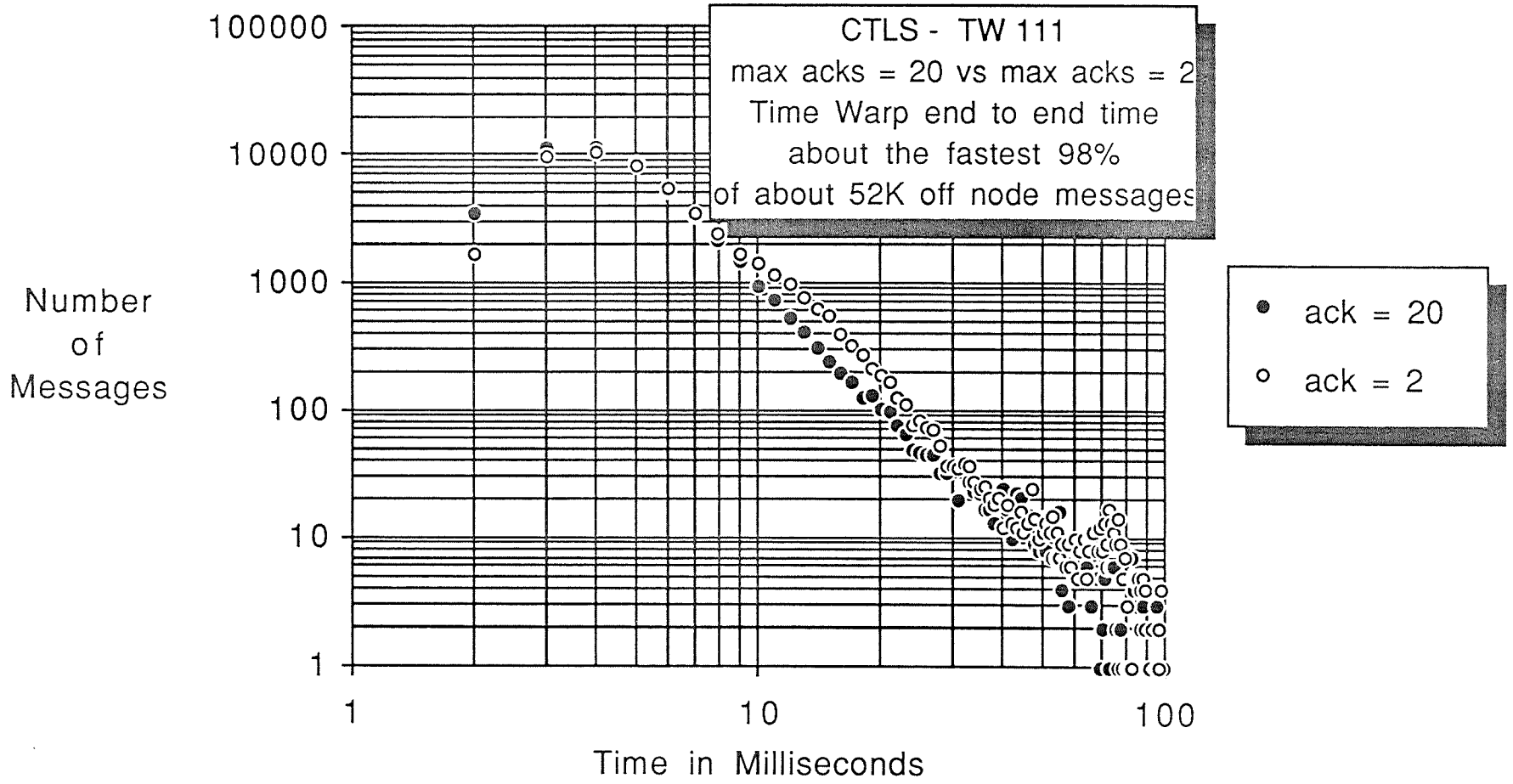

CTLS - TW 111 (max acks = 2)
Time Warp end to end time
Fastest 95% of the 51,424 off node messages
100000
10000
1000
100
Number of Messages
0
2
4
6
8
10
12
14
16
18
Time in Milliseconds


CTLS - TW 111 (max acks = 2
Time Warp end to end time
Slowest 5.0 to 0.1% of the
51,424 off node messages
1000
100
10
1
Number
of
Messages
10
20
30
40
50
60
70
80
90
100
Time in Milliseconds

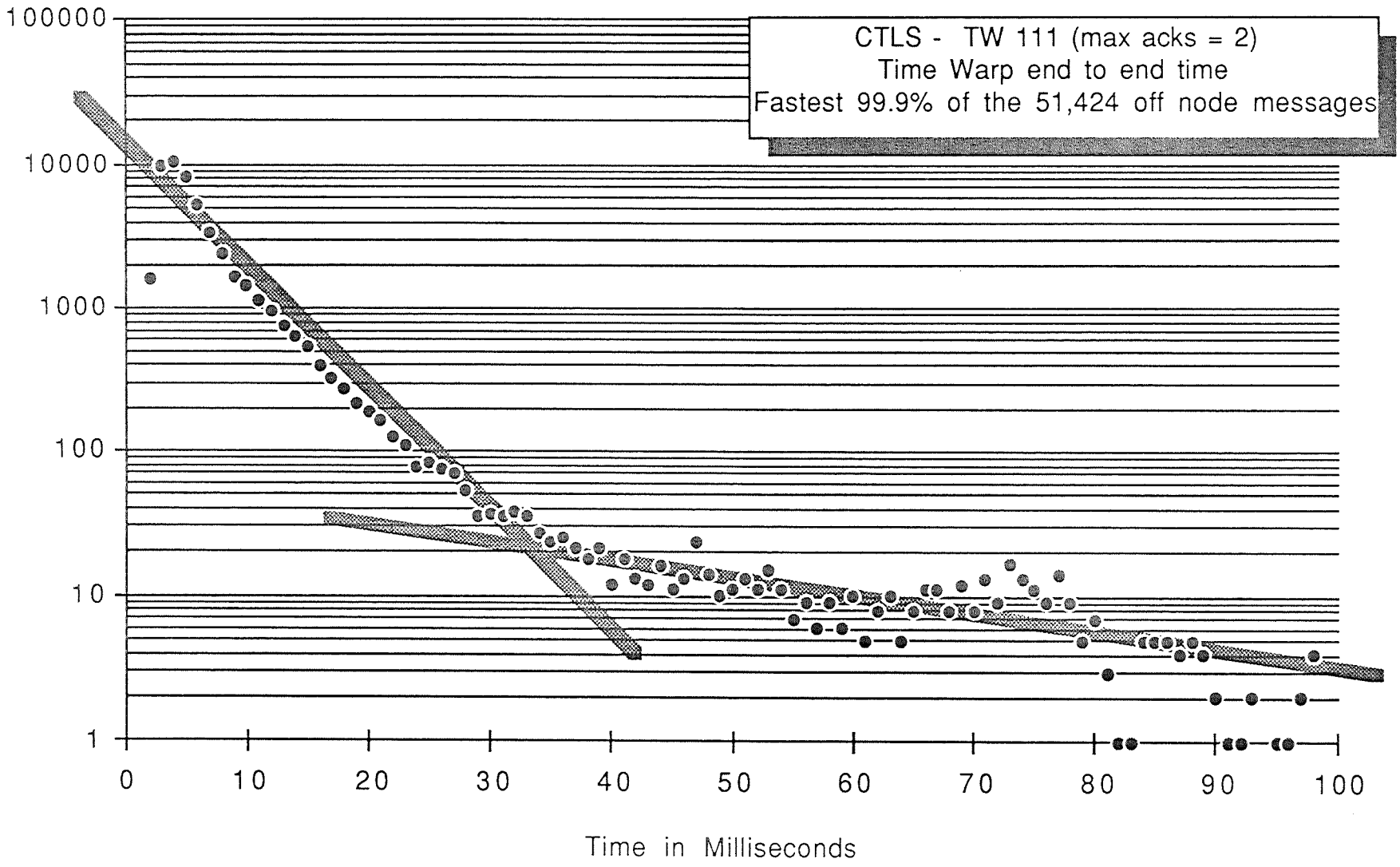

CTLS - TW 111 (max acks = 2)
Time Warp end to end time
Slowest 0.1% of the 51,424 off node messages
10
1
Number
of
Messages
50
100
150
200
250
300
350
400
Time in Milliseconds

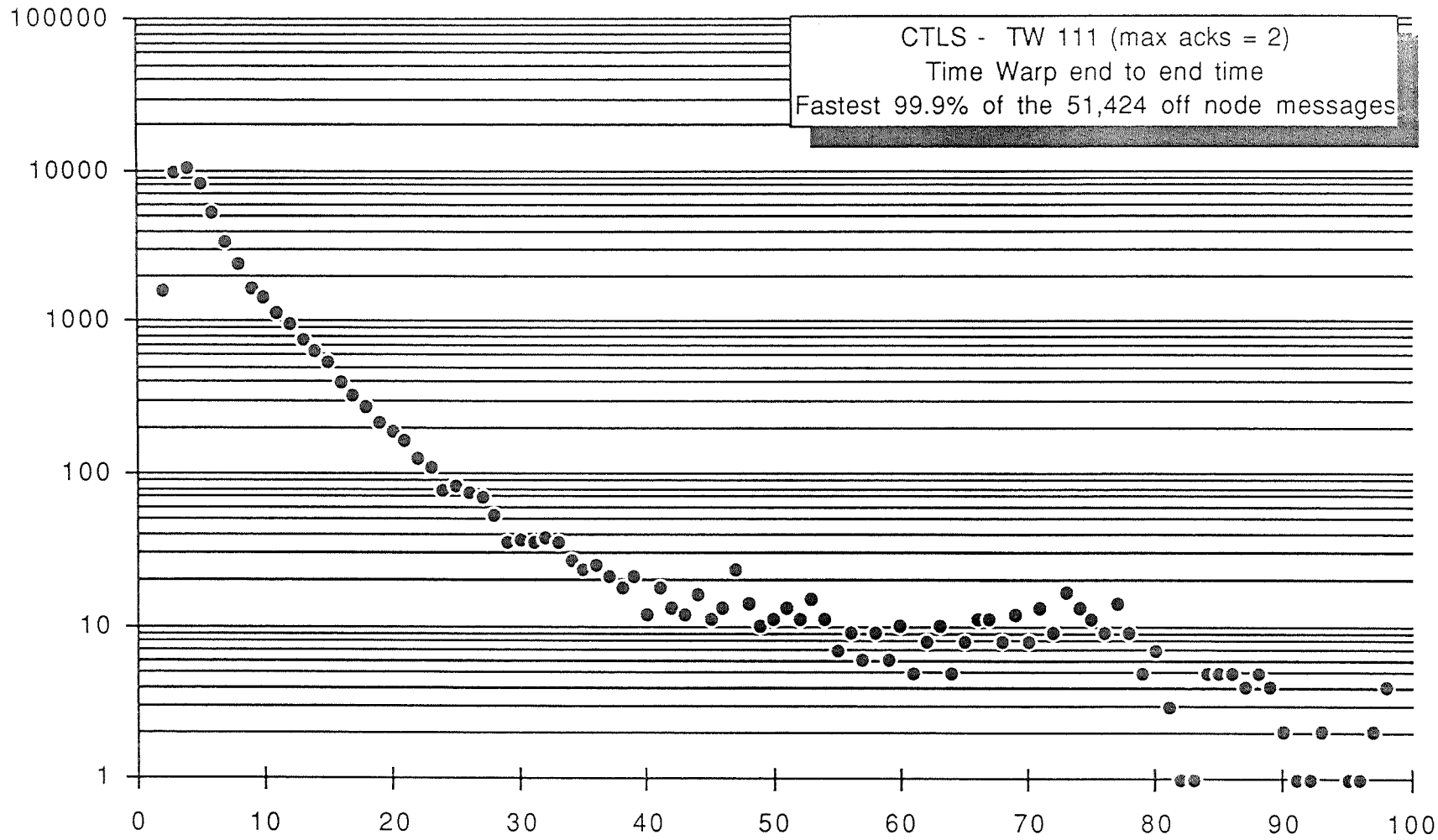

CTLS - TW 111
max acks = 20 vs max acks = 2
Time Warp end to end time
about the fastest 98%
of about 52K off node messages
ack = 20
ack = 2
Number of Messages
12000
10000
8000
6000
4000
2000
0
0 10 20 30 40 50 60 70 80 90 100
Time in Milliseconds


CTLS - TW 111
max acks = 20 vs max acks = 2
Time Warp end to end time
about the fastest 98%
of about 52K off node messages
Number of Messages
100000
10000
1000
100
10
1
0
10
20
30
40
50
60
70
80
90
100
Time in Milliseconds
ack = 20
ack = 2


CTLS - TW 111
max acks = 20 vs max acks = 2
Time Warp end to end time
about the fastest 98%
of about 52K off node messages
ack = 20
ack = 2
Number of Messages
100000
10000
1000
100
10
1
1
10
100
Time in Milliseconds

Two Queueing Delays?




CTLS - TW 111 (max acks = 2)
Time Warp end to end time
Fastest 99.9% of the 51,424 off node messages
100000
10000
1000
100
10
1
0
10
20
30
40
50
60
70
80
90
100

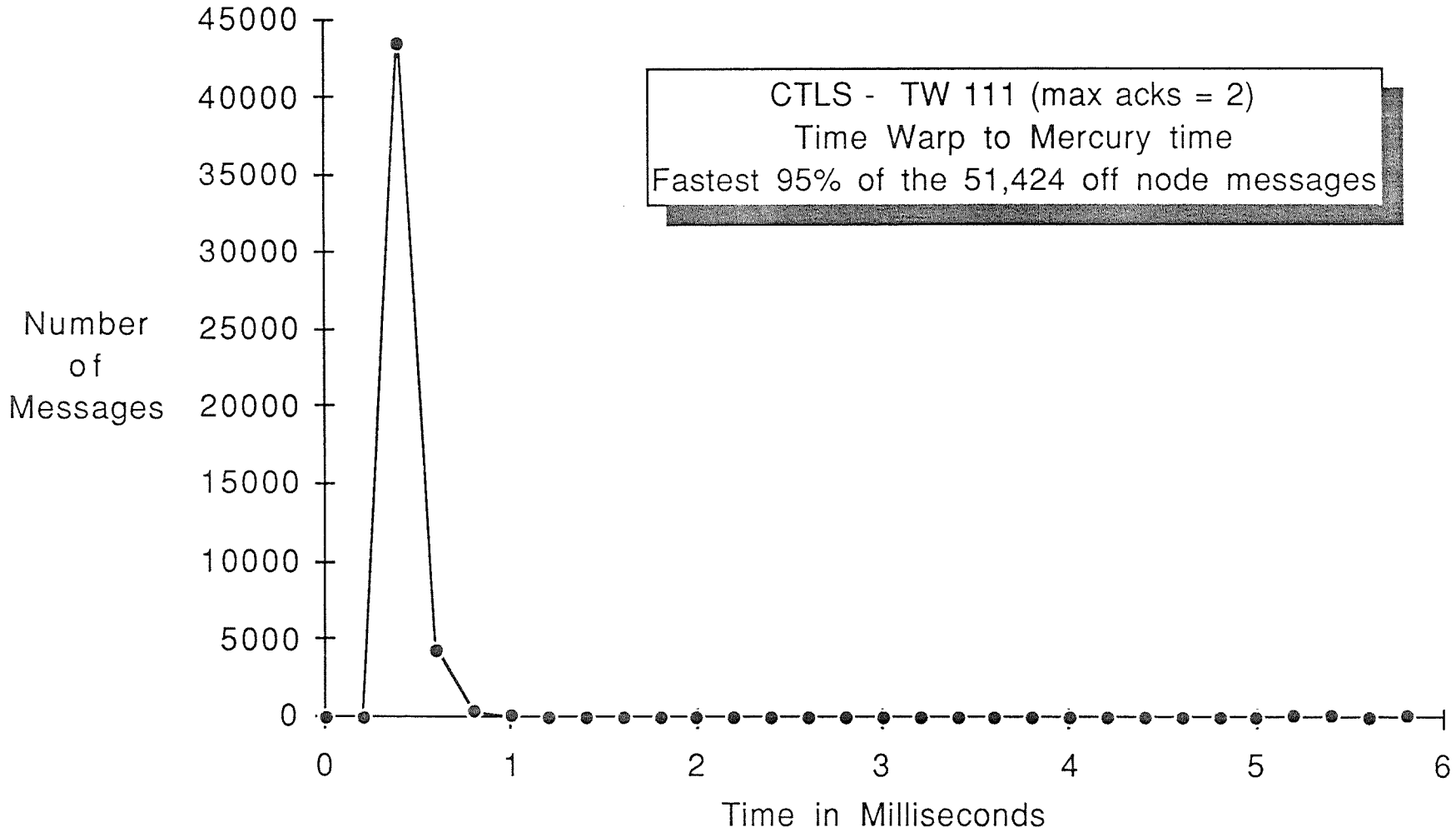
Appendix B
Time Warp to Mercury Times

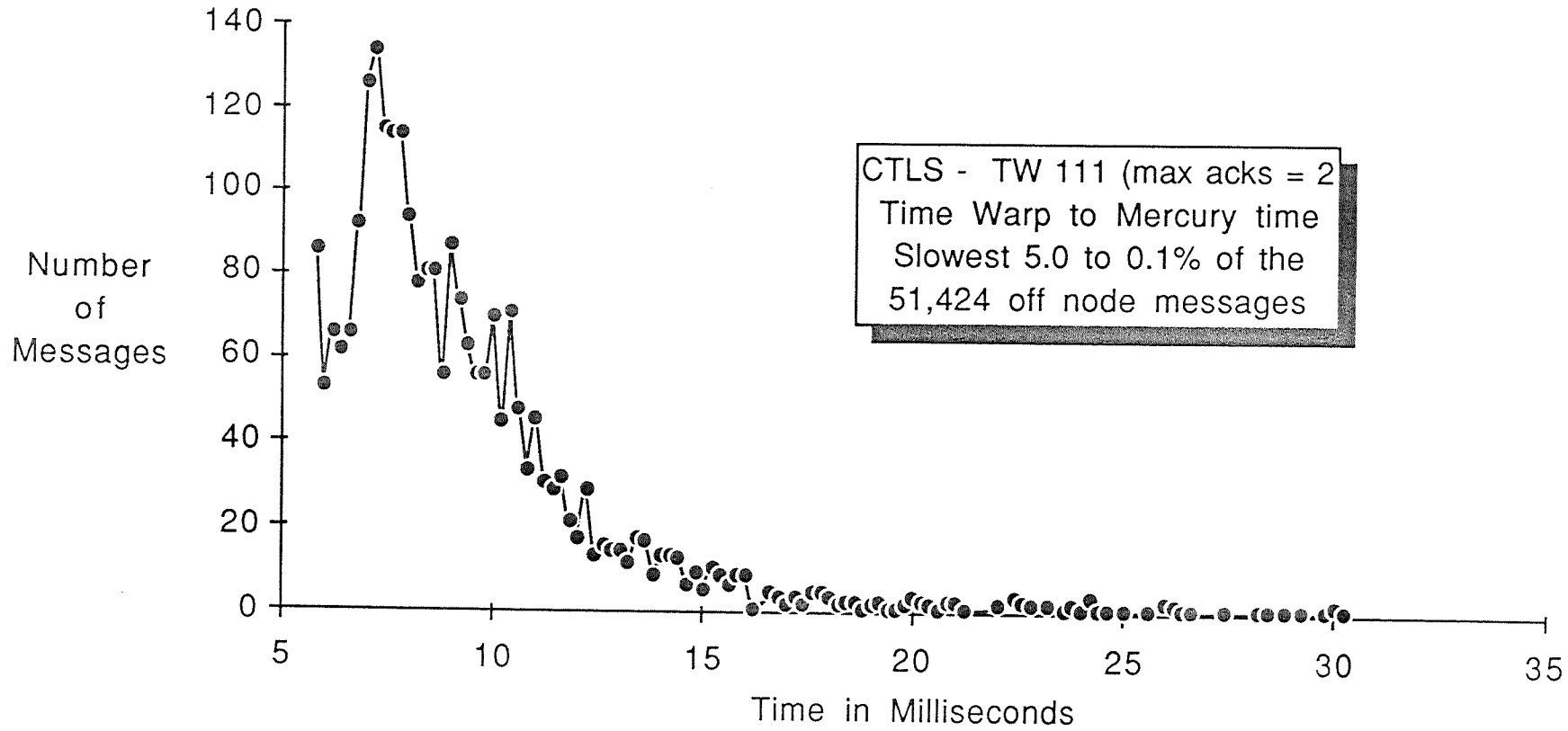
B-1: Fastest 95%, acks=2, graph (see B-4).
B-2: The 95% - 99.9% range, acks=2, graph (see B-5).
B-3: Slowest 0.1%, acks=2, graph (see B-6).

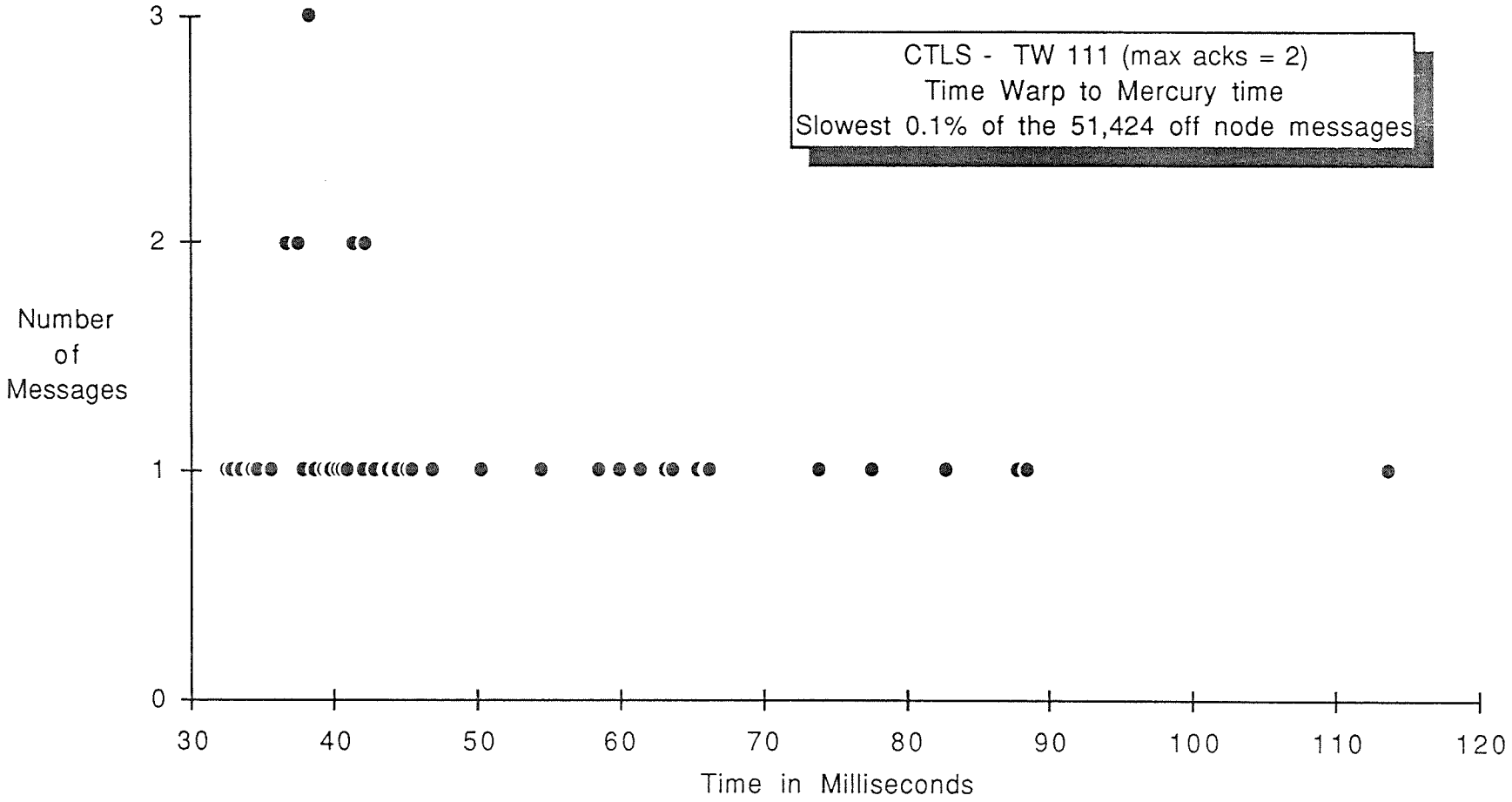
B-4: Fastest 95%, acks=2, semi-log graph (see B-1).
B-5: The 95% - 99.9% range, acks=2, semi-log graph (see B-2).
B-6: Slowest 0.1%, acks=2, semi-log graph (see B-3).

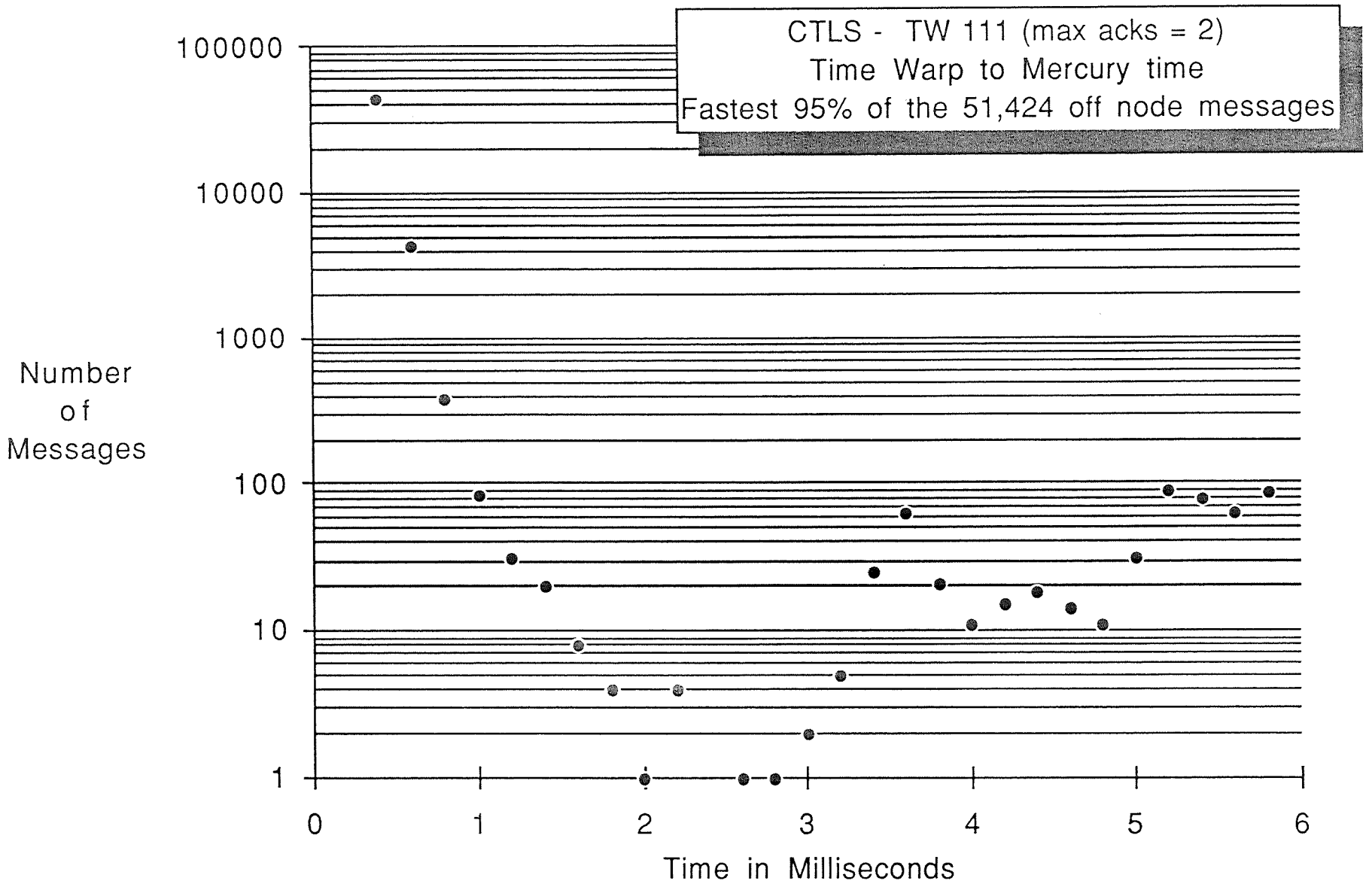
B-7: 100%, acks=20, graph (see B-8).
B-8: 100%, acks=20, semi-log graph (see B-7).

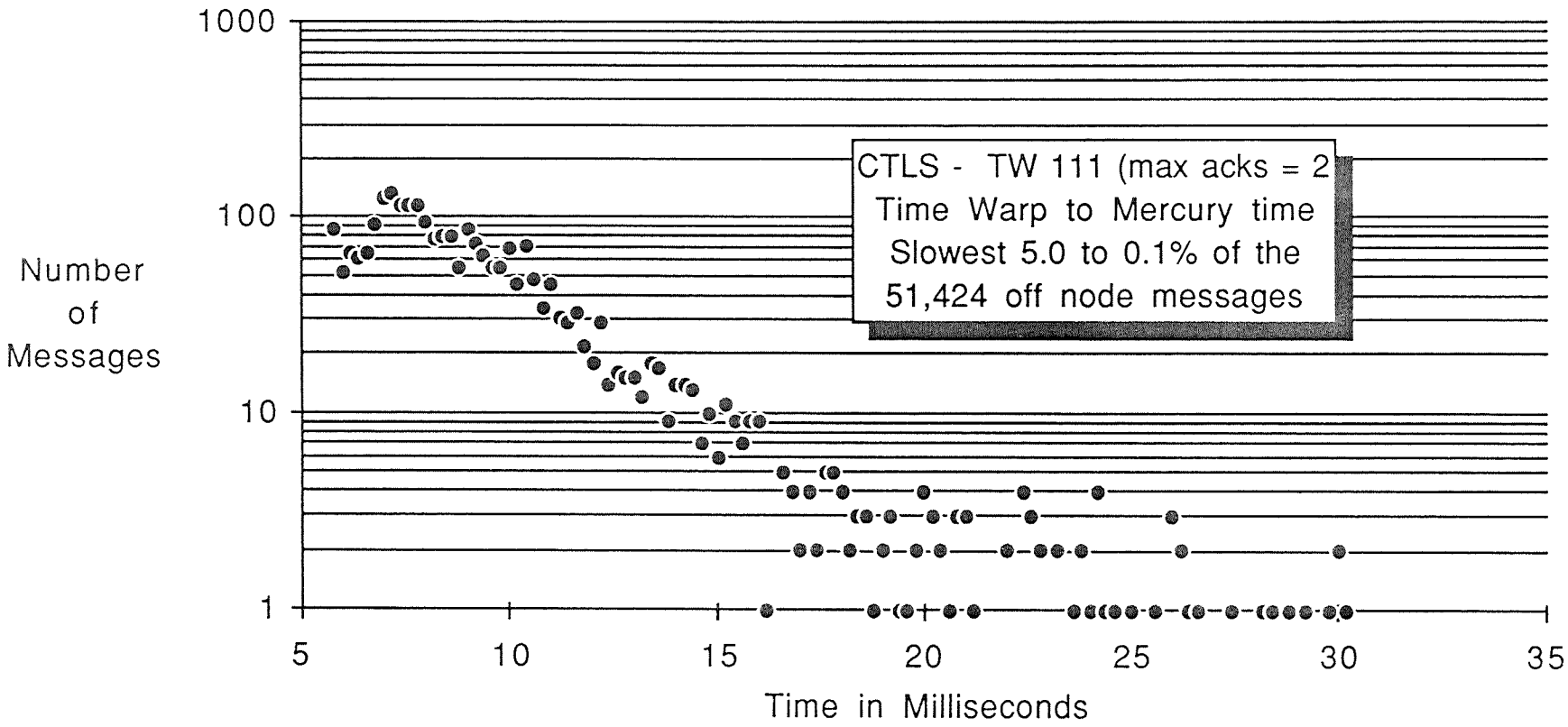
B-9: Fastest 99.96%, acks=20, graph (see B-10).
B-10: Fastest 99.96%, acks=20, semi-graph (see B-9).

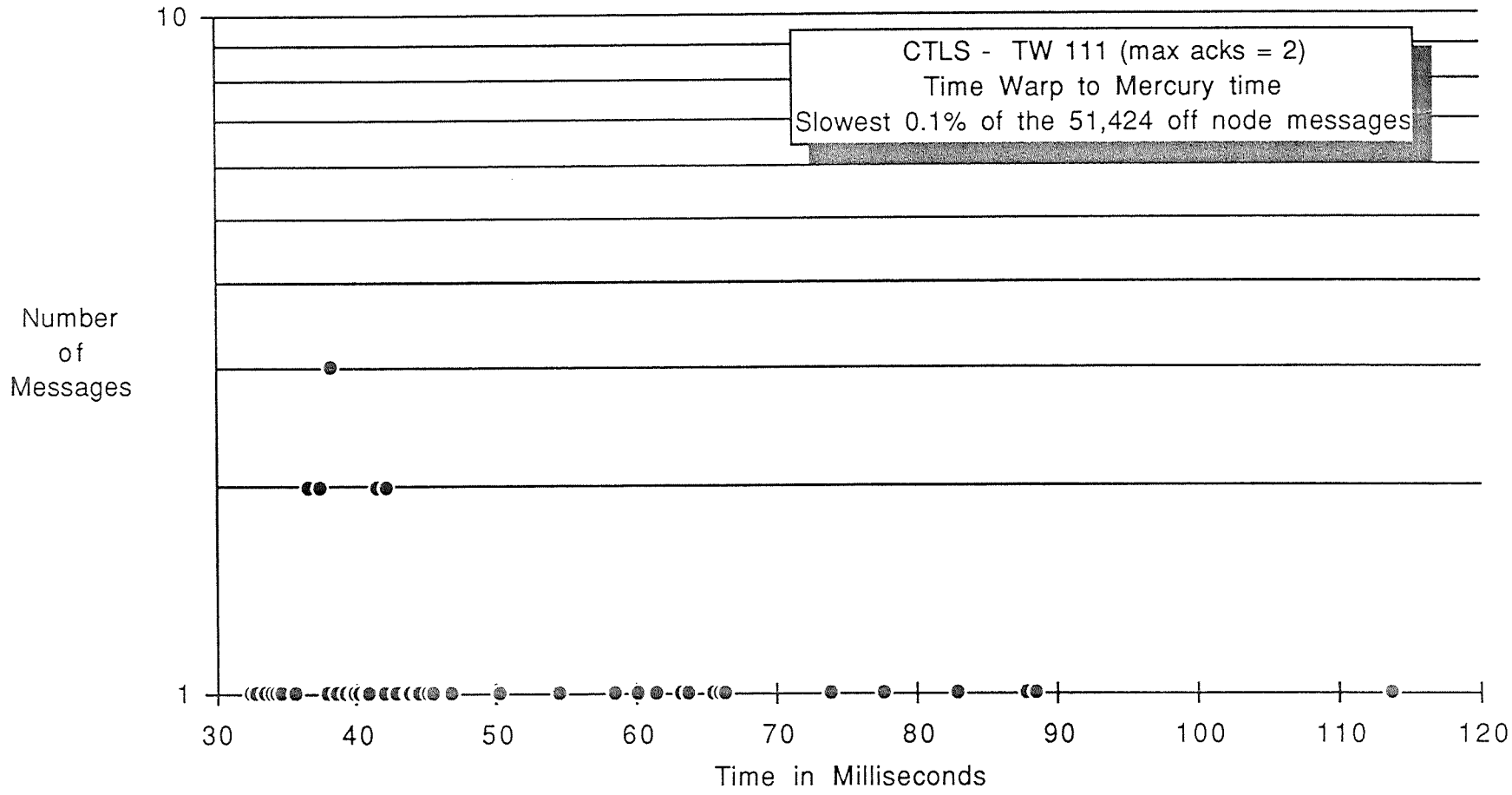

CTLS - TW 111 (max acks = 2)
Time Warp to Mercury time
Fastest 95% of the 51,424 off node messages
Number of Messages
45000
40000
35000
30000
25000
20000
15000
10000
5000
0
0
1
2
3
4
5
6
Time in Milliseconds

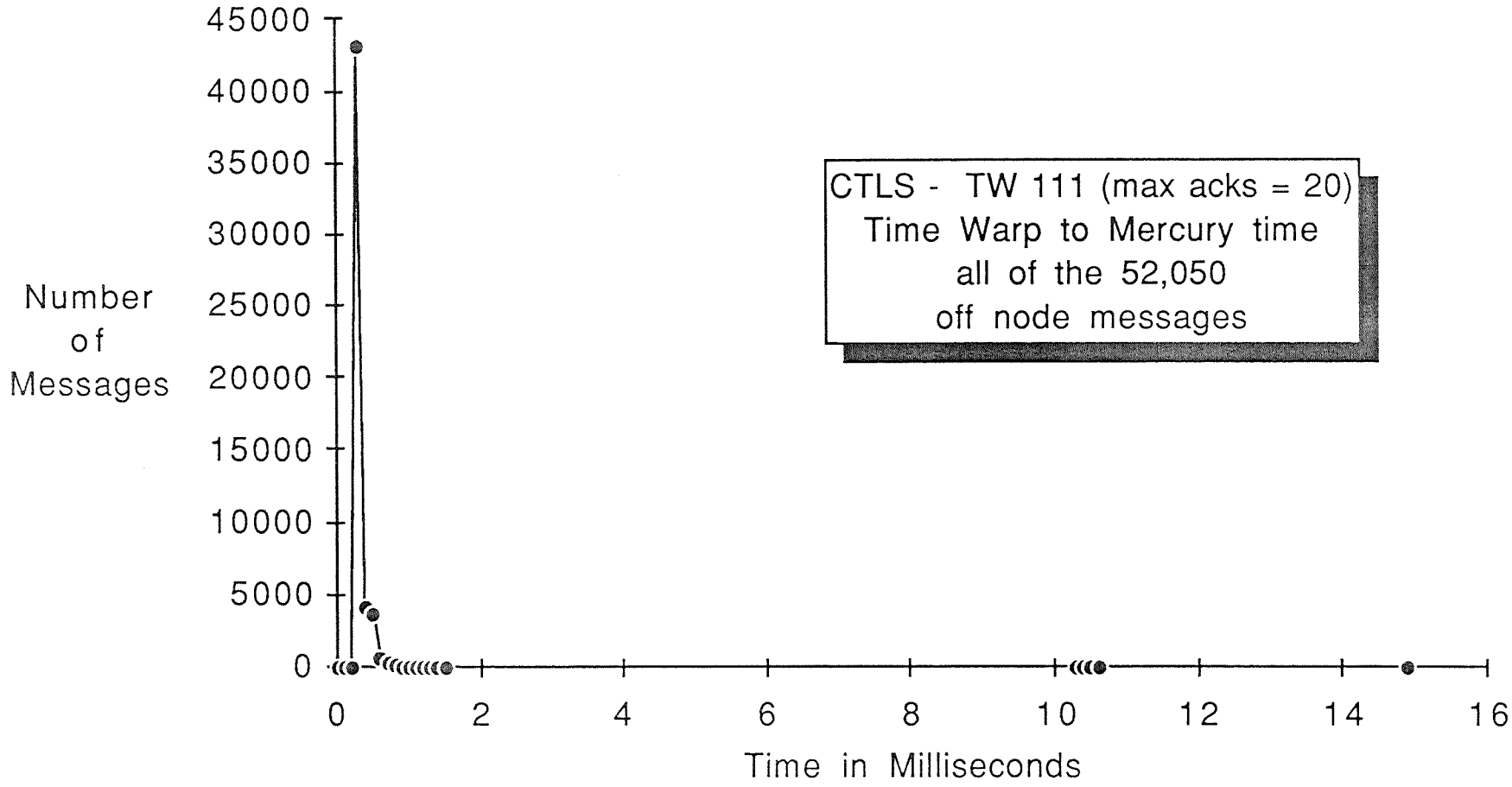

140
120
100
80
60
40
20
0
Number of Messages
5
10
15
20
25
30
35
Time in Milliseconds
CTLS - TW 111 (max acks = 2
Time Warp to Mercury time
Slowest 5.0 to 0.1% of the
51,424 off node messages

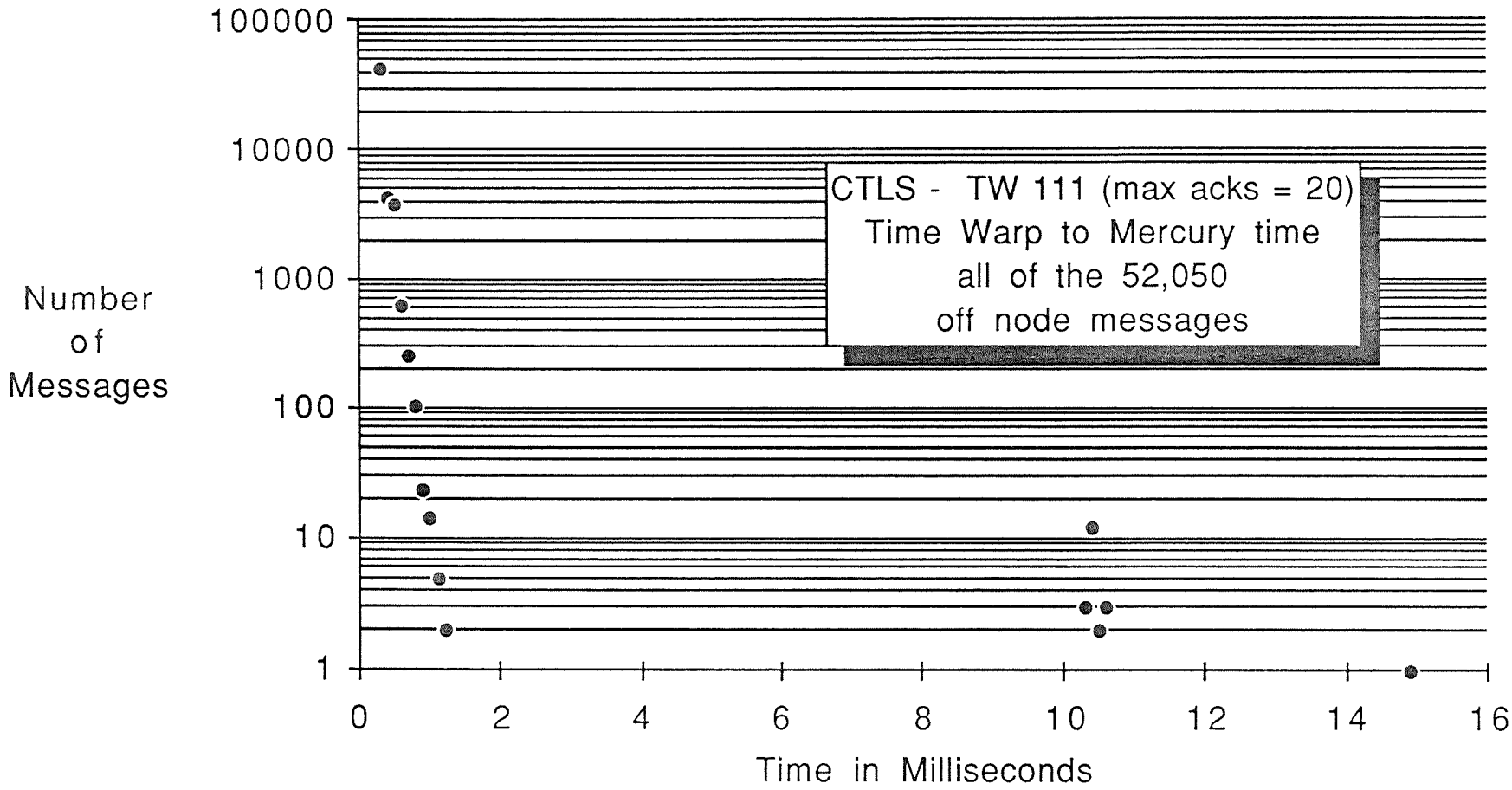

CTLS - TW 111 (max acks = 2)
Time Warp to Mercury time
Slowest 0.1% of the 51,424 off node messages
Number of Messages
0
1
2
3
30
40
50
60
70
80
90
100
110
120
Time in Milliseconds

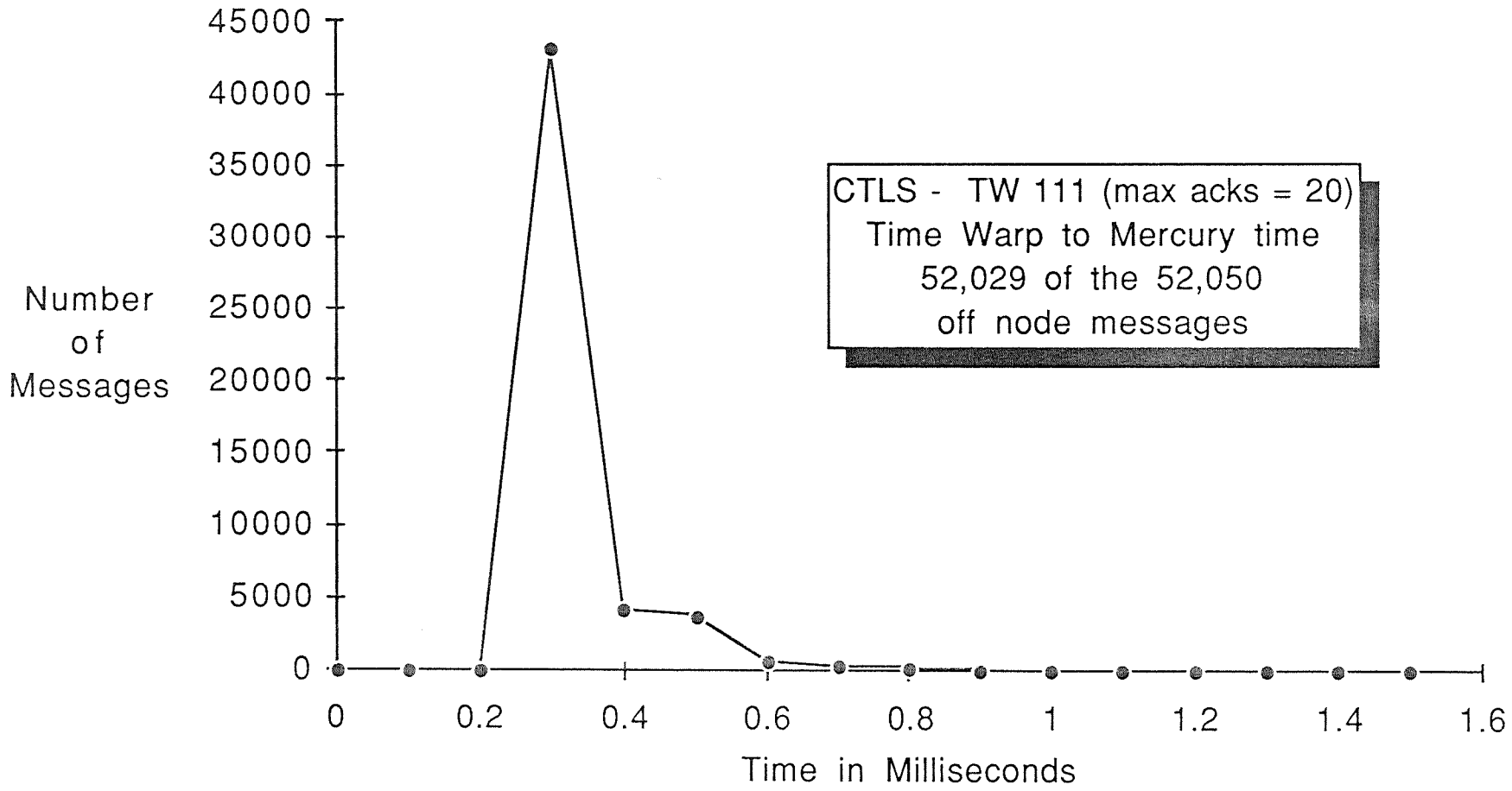

CTLS - TW 111 (max acks = 2)
Time Warp to Mercury time
Fastest 95% of the 51,424 off node messages
100000
10000
1000
100
10
1
Number of Messages
0
1
2
3
4
5
6
Time in Milliseconds

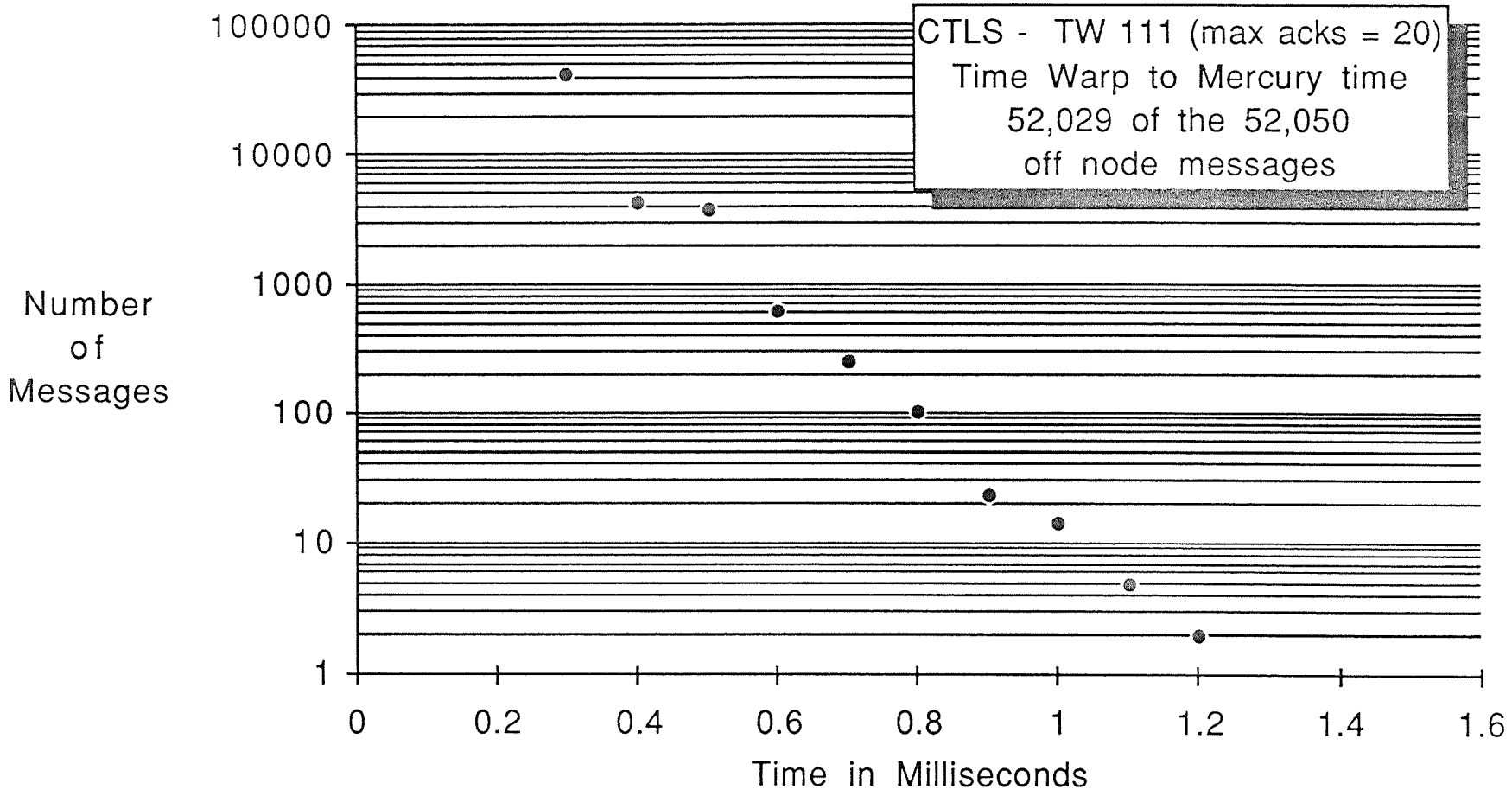

1000
100
10
1
Number
of
Messages
CTLS - TW 111 (max acks = 2
Time Warp to Mercury time
Slowest 5.0 to 0.1% of the
51,424 off node messages
5
10
15
20
25
30
35
Time in Milliseconds


10
1
CTLS - TW 111 (max acks = 2)
Time Warp to Mercury time
Slowest 0.1% of the 51,424 off node messages
Number
of
Messages
30
40
50
60
70
80
90
100
110
120
Time in Milliseconds


45000
40000
35000
30000
25000
20000
15000
10000
5000
0
Number of Messages
CTLS - TW 111 (max acks = 20)
Time Warp to Mercury time
all of the 52,050
off node messages
0
2
4
6
8
10
12
14
16
Time in Milliseconds


100000
10000
1000
100
10
1
Number of Messages
CTLS - TW 111 (max acks = 20)
Time Warp to Mercury time
all of the 52,050
off node messages
0
2
4
6
8
10
12
14
16
Time in Milliseconds


45000
40000
35000
30000
25000
20000
15000
10000
5000
0
Number of Messages
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
Time in Milliseconds
CTLS - TW 111 (max acks = 20)
Time Warp to Mercury time
52,029 of the 52,050
off node messages


CTLS - TW 111 (max acks = 20)
Time Warp to Mercury time
52,029 of the 52,050
off node messages
Number of Messages
100000
10000
1000
100
10
1
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
Time in Milliseconds

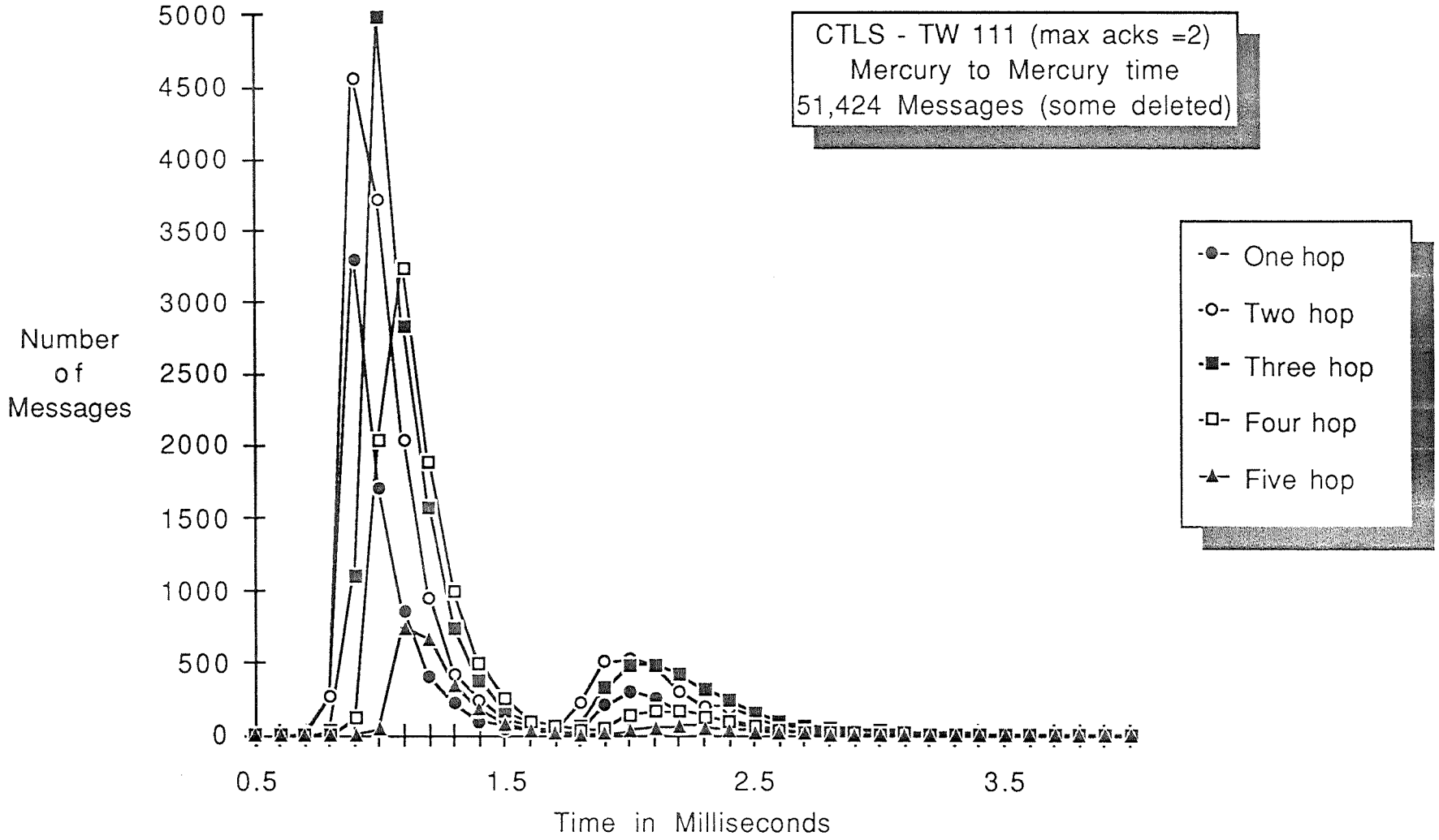
Appendix C
Mercury to Mercury Times

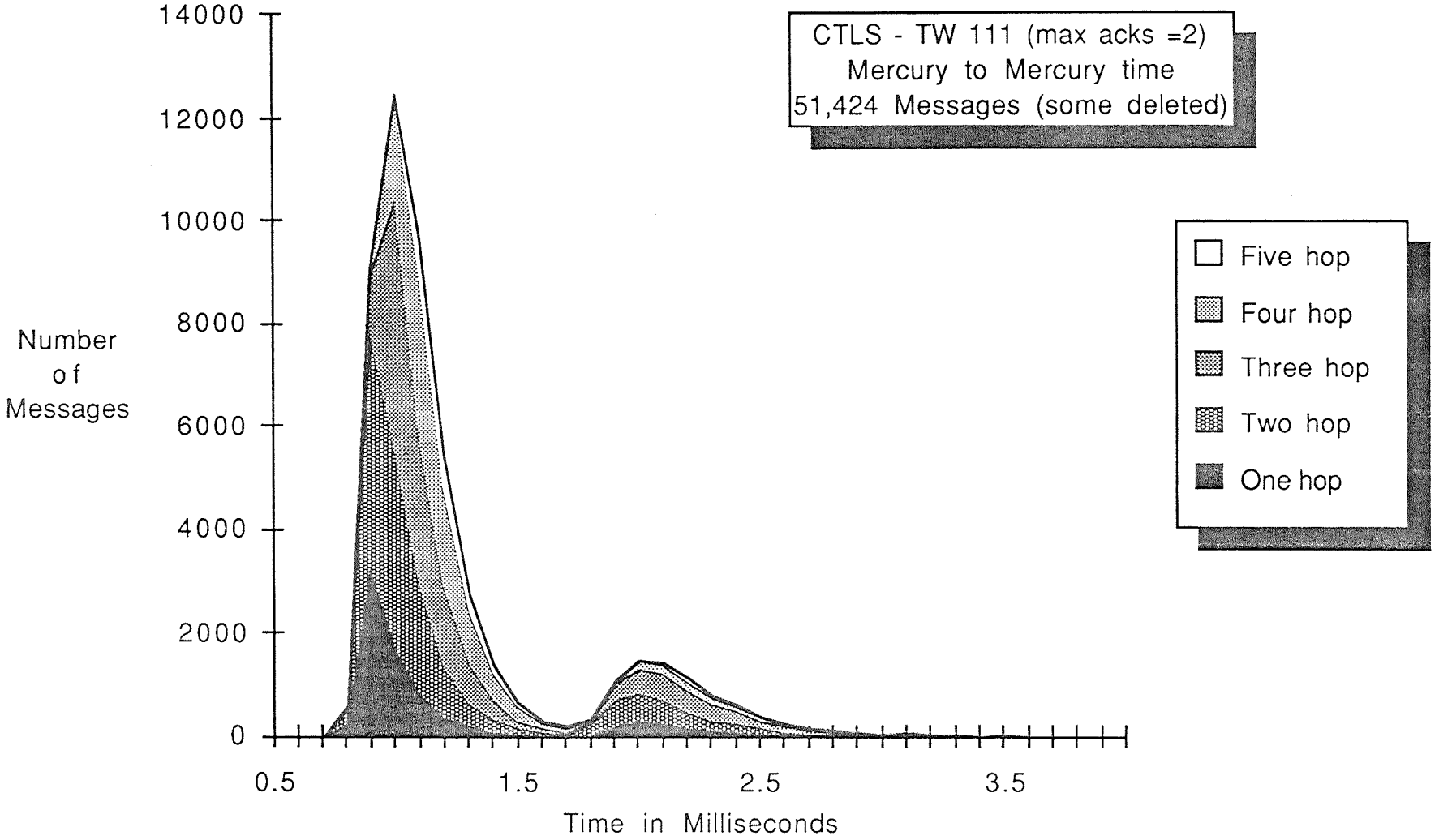
C-1: Fastest 99.66%, acks=2, by hops, graph (see C-2, C-3).
C-2: Fastest 99.66%, acks=2, by hops, area graph (see C-1, C-3).
C-3: Fastest 99.66%, acks=2, by hops, semi-log graph (see C-1, C-2).

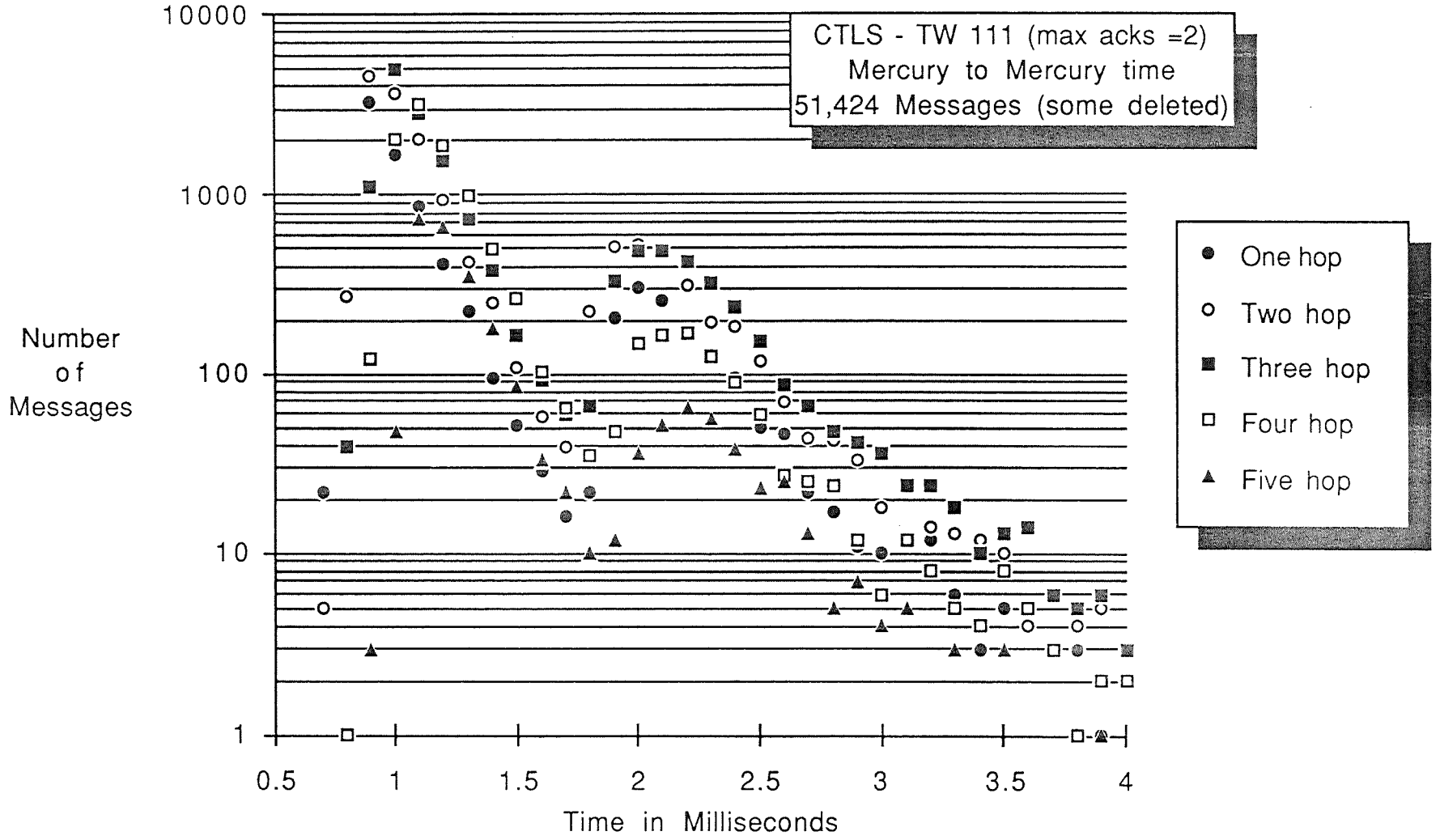
C-4: Fastest 99.64%, acks=20, by hops, graph (see C-5, C-6).
C-5: Fastest 99.64%, acks=20, by hops, area graph (see C-4, C-6).
C-6: Fastest 99.64%, acks=20, by hops, semi-log graph (see C-5, C-6).

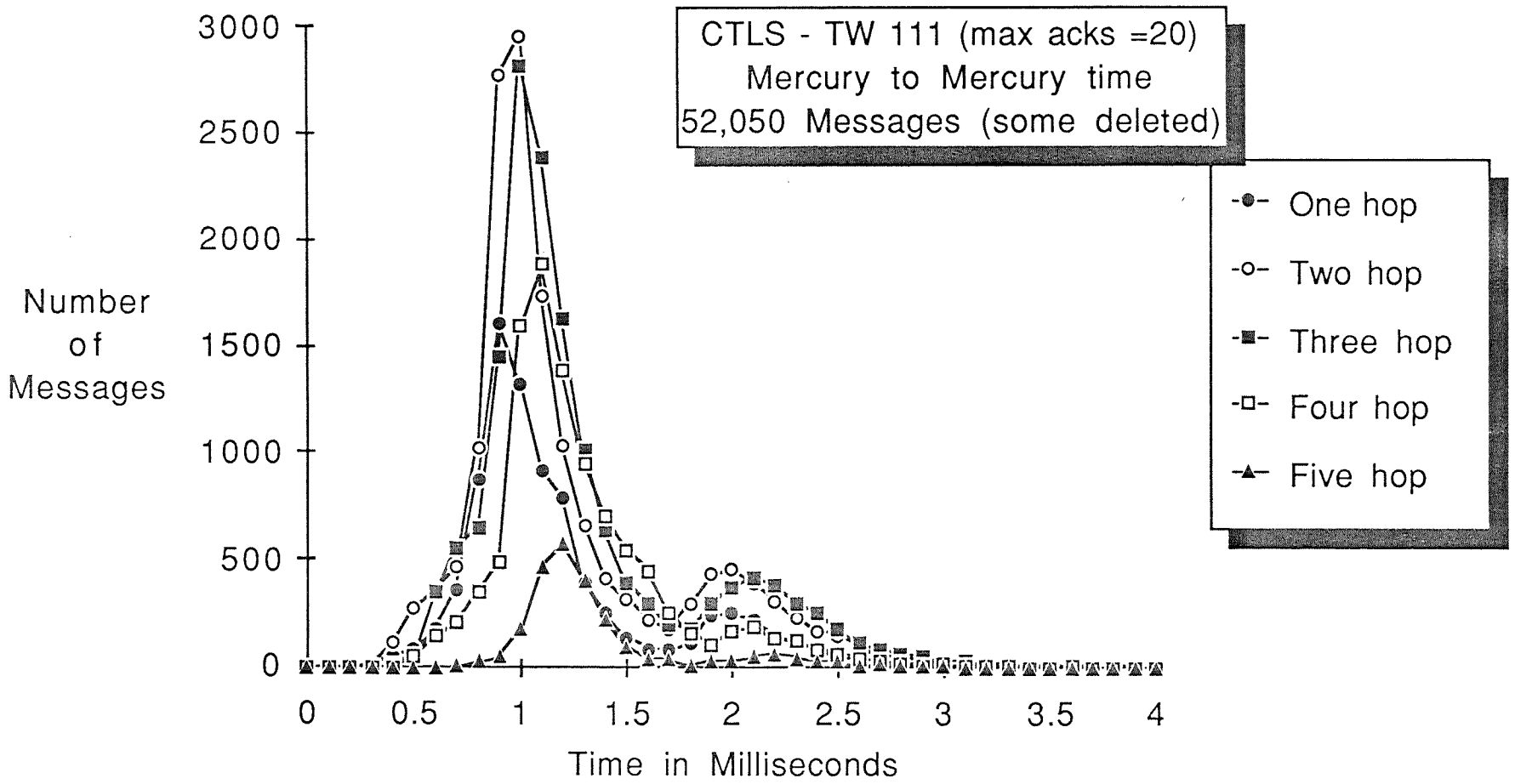
C-7: Slowest 0.36%, acks=20, by hops graph.

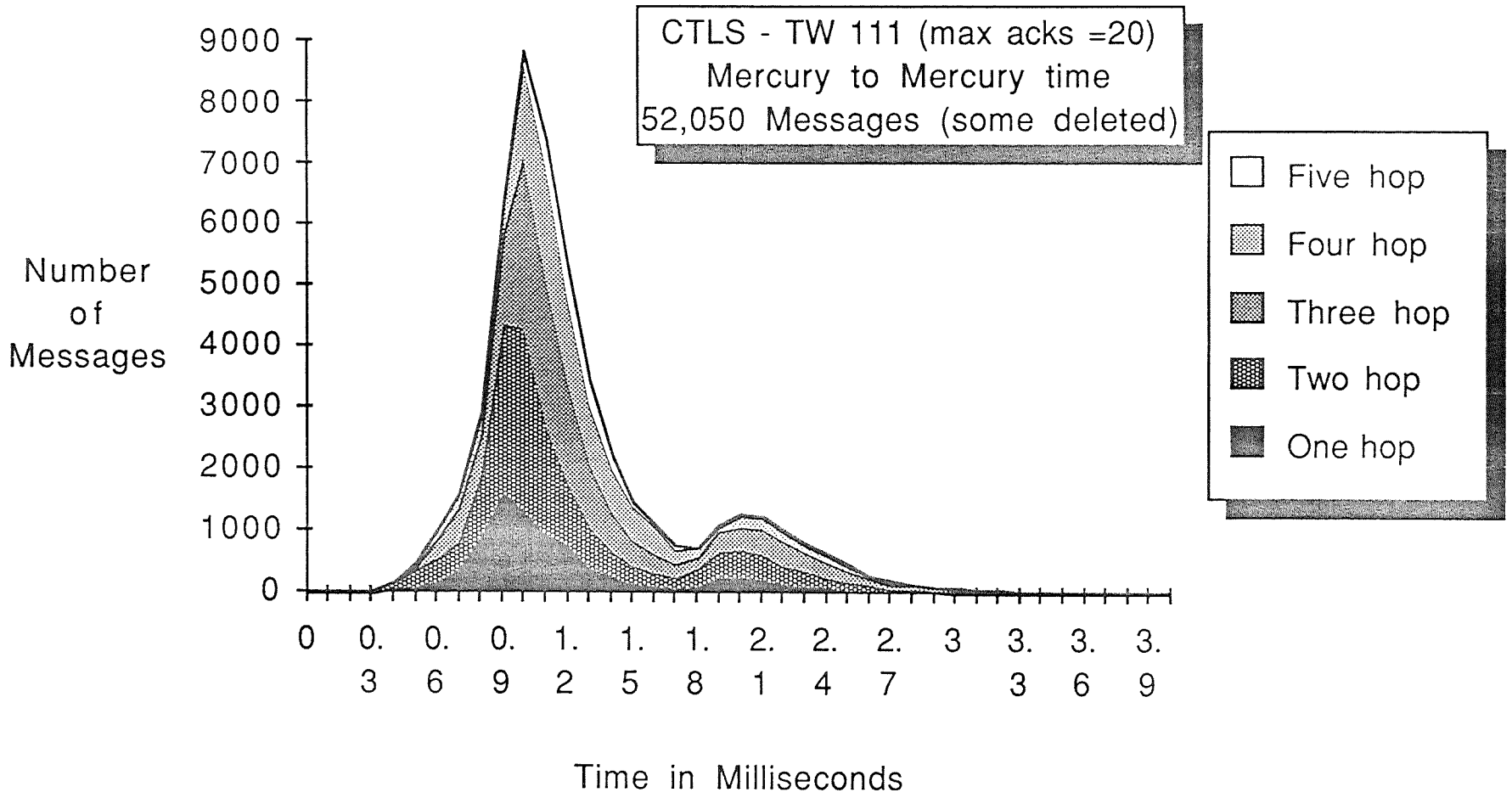
C-8: Fastest 3 hop, acks=2, by packets, area graph (see C-9).
C-9: Fastest 3 hop, acks=2, by packets, semi-log graph (see C-8).

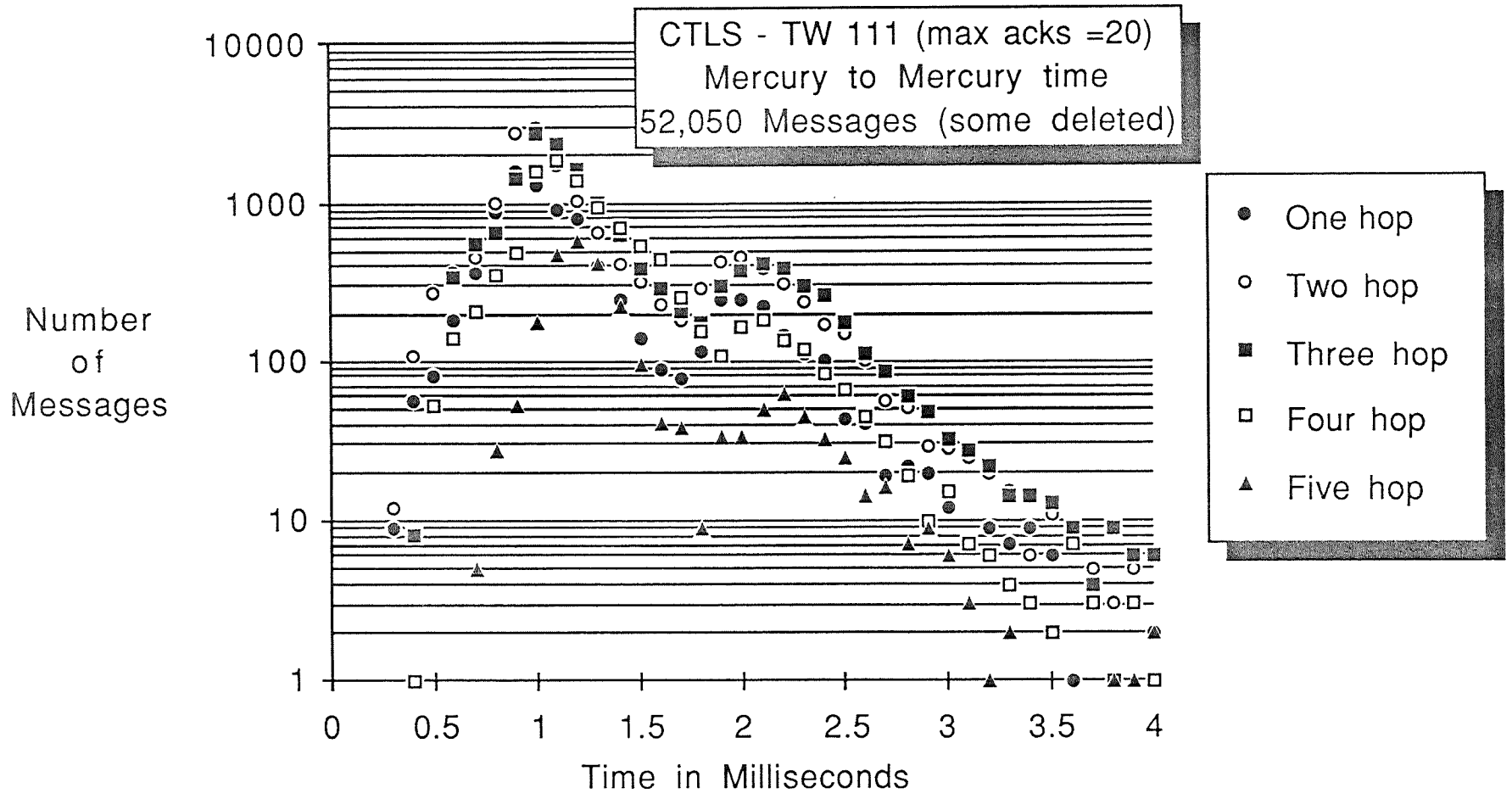

CTLS - TW 111 (max acks =2)
Mercury to Mercury time
51,424 Messages (some deleted)
One hop
Two hop
Three hop
Four hop
Five hop
Number of Messages
5000
4500
4000
3500
3000
2500
2000
1500
1000
500
0
0.5
1.5
2.5
3.5
Time in Milliseconds

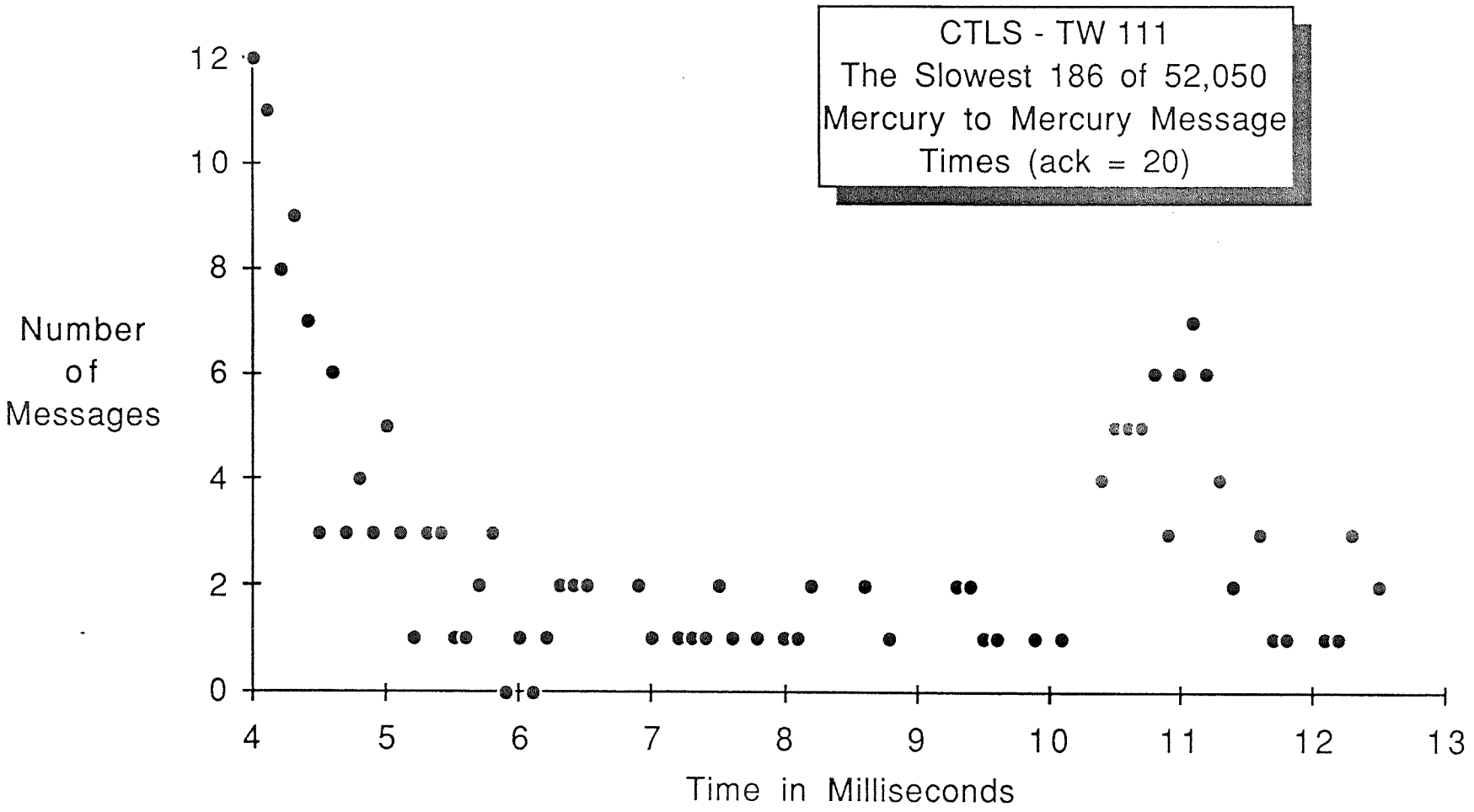

CTLS - TW 111 (max acks =2)
Mercury to Mercury time
51,424 Messages (some deleted)
Five hop
Four hop
Three hop
Two hop
One hop
Number of Messages
14000
12000
10000
8000
6000
4000
2000
0
0.5
1.5
2.5
3.5
Time in Milliseconds

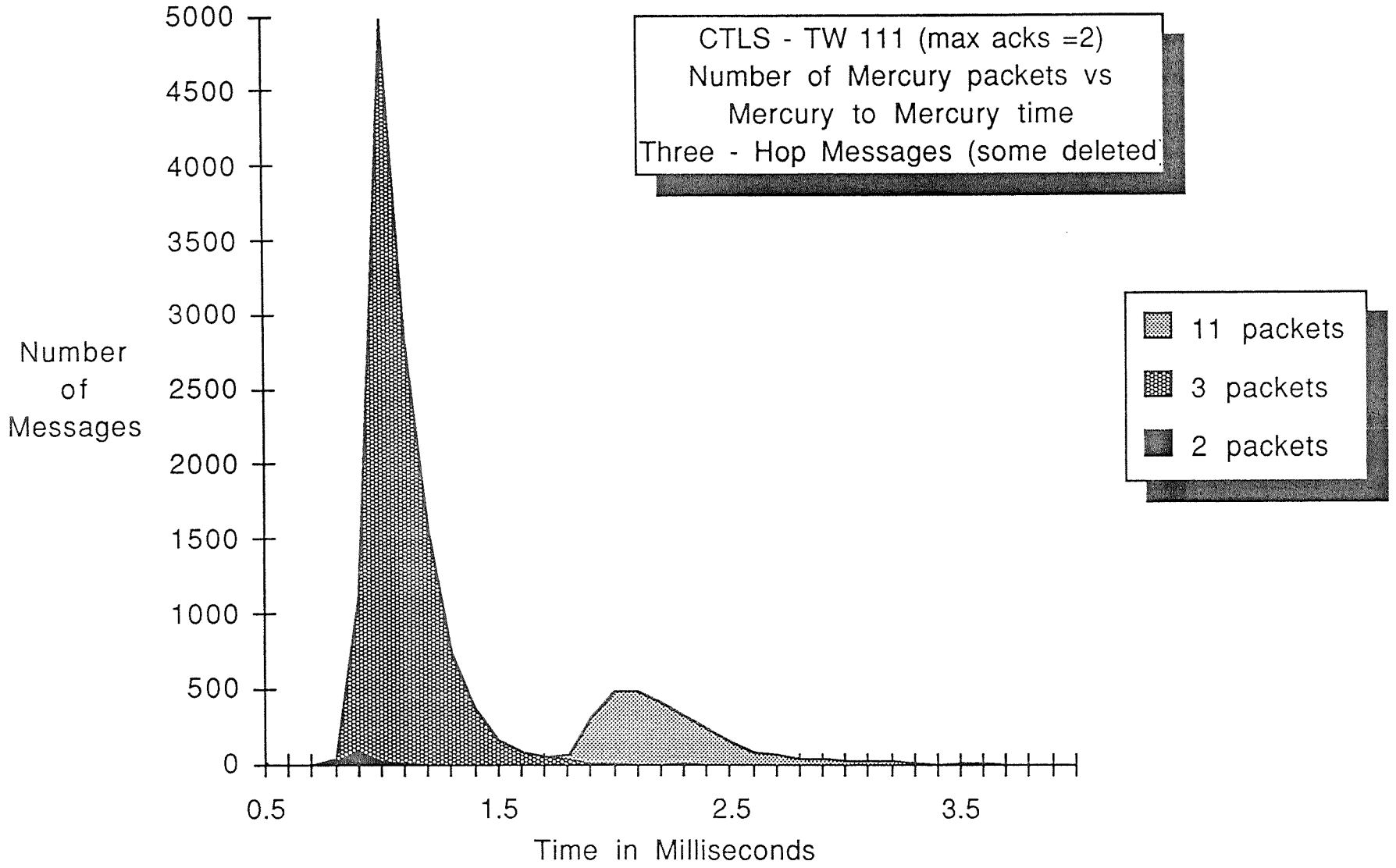

CTLS - TW 111 (max acks =2)
Mercury to Mercury time
51,424 Messages (some deleted)
10000
1000
100
10
1
Number of Messages
0.5
1
1.5
2
2.5
3
3.5
4
Time in Milliseconds
One hop
Two hop
Three hop
Four hop
Five hop

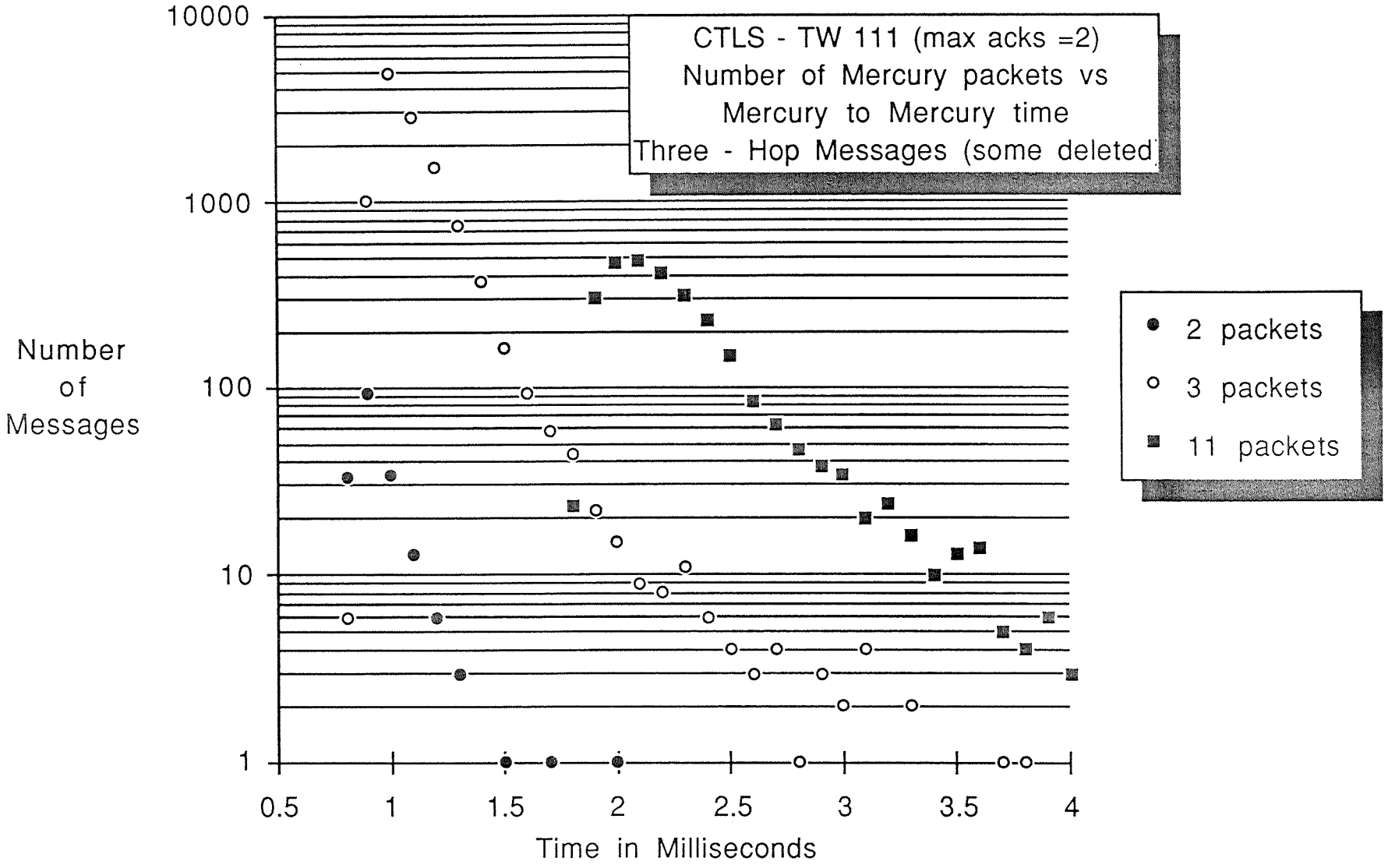

CTLS - TW 111 (max acks =20)
Mercury to Mercury time
52,050 Messages (some deleted)
One hop
Two hop
Three hop
Four hop
Five hop
Number of Messages
3000
2500
2000
1500
1000
500
0
0
0.5
1
1.5
2
2.5
3
3.5
4
Time in Milliseconds


CTLS - TW 111 (max acks =20)
Mercury to Mercury time
52,050 Messages (some deleted)
Number of Messages
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
0
0.3
0.6
0.9
1.2
1.5
1.8
2.1
2.4
2.7
3
3.3
3.6
3.9
Time in Milliseconds
Five hop
Four hop
Three hop
Two hop
One hop


CTLS - TW 111 (max acks =20)
Mercury to Mercury time
52,050 Messages (some deleted)
10000
1000
100
10
1
Number of Messages
0
0.5
1
1.5
2
2.5
3
3.5
4
Time in Milliseconds
One hop
Two hop
Three hop
Four hop
Five hop


CTLS - TW 111
The Slowest 186 of 52,050
Mercury to Mercury Message
Times (ack = 20)
Number
of
Messages
0
2
4
6
8
10
12
4
5
6
7
8
9
10
11
12
13
Time in Milliseconds

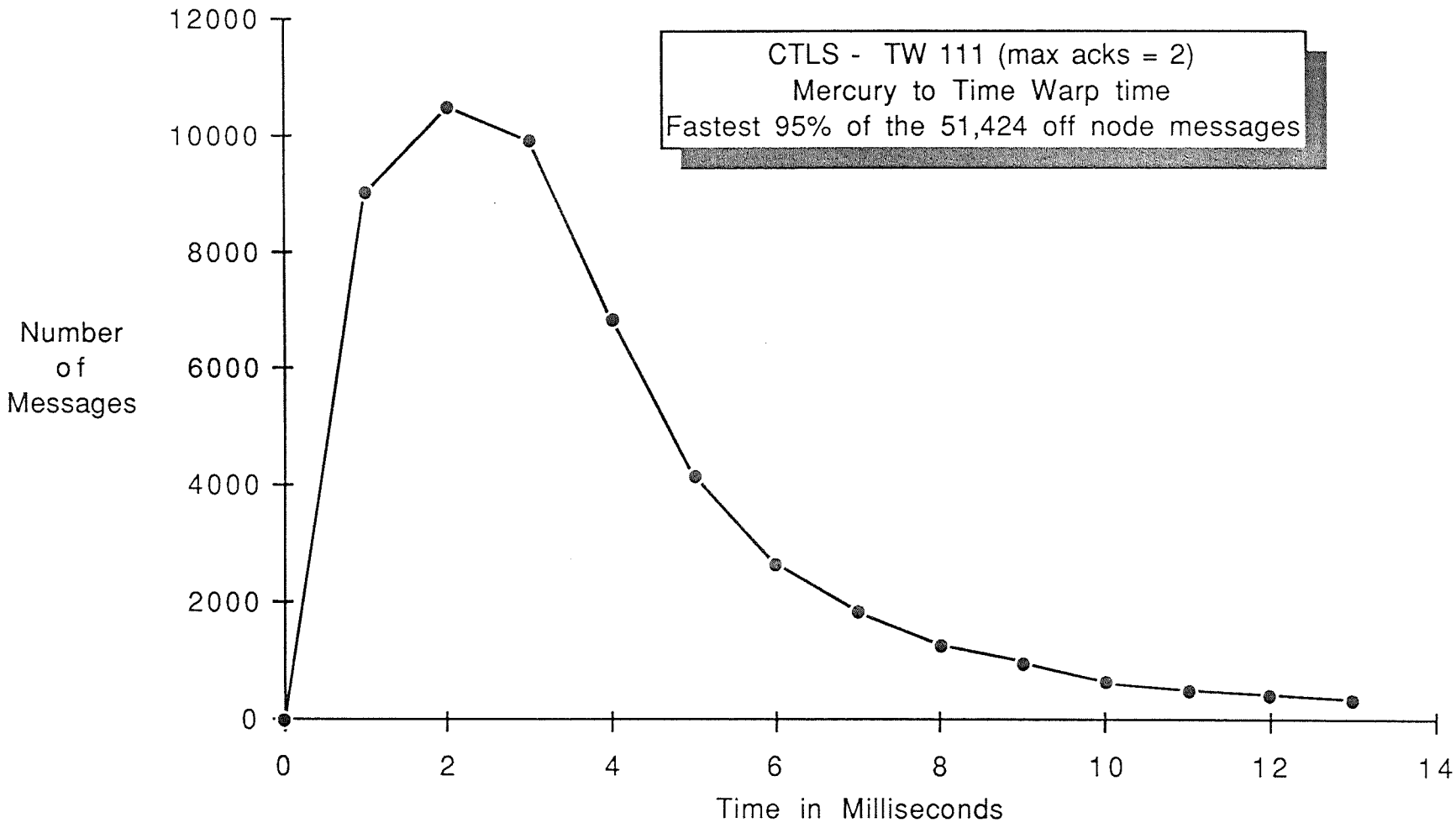

CTLS - TW 111 (max acks =2)
Number of Mercury packets vs
Mercury to Mercury time
Three - Hop Messages (some deleted)
11 packets
3 packets
2 packets
Number of Messages
5000
4500
4000
3500
3000
2500
2000
1500
1000
500
0
0.5
1.5
2.5
3.5
Time in Milliseconds

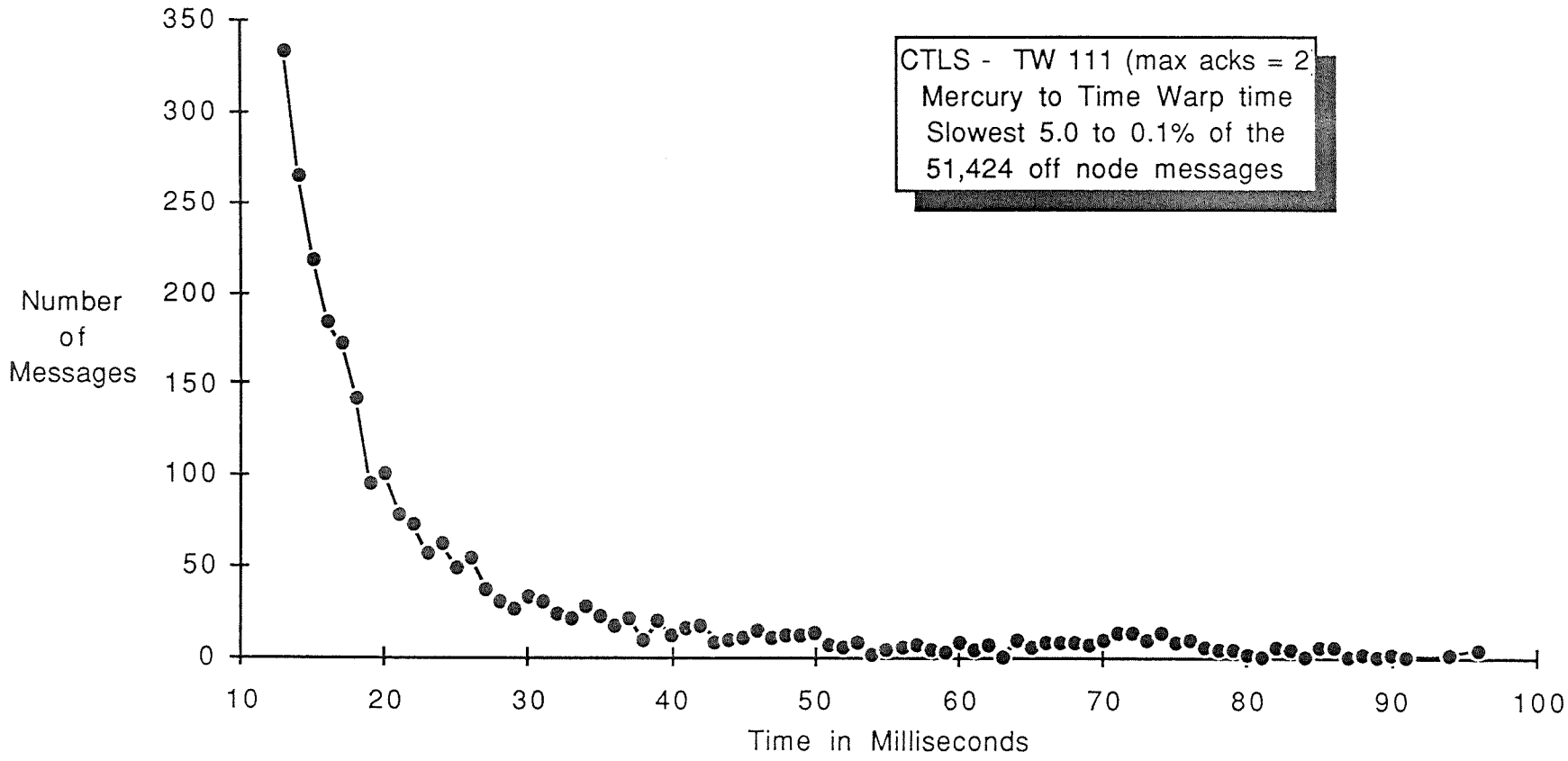

CTLS - TW 111 (max acks =2)
Number of Mercury packets vs
Mercury to Mercury time
Three - Hop Messages (some deleted)
10000
1000
100
10
1
Number
of
Messages
0.5
1
1.5
2
2.5
3
3.5
4
Time in Milliseconds
2 packets
3 packets
11 packets

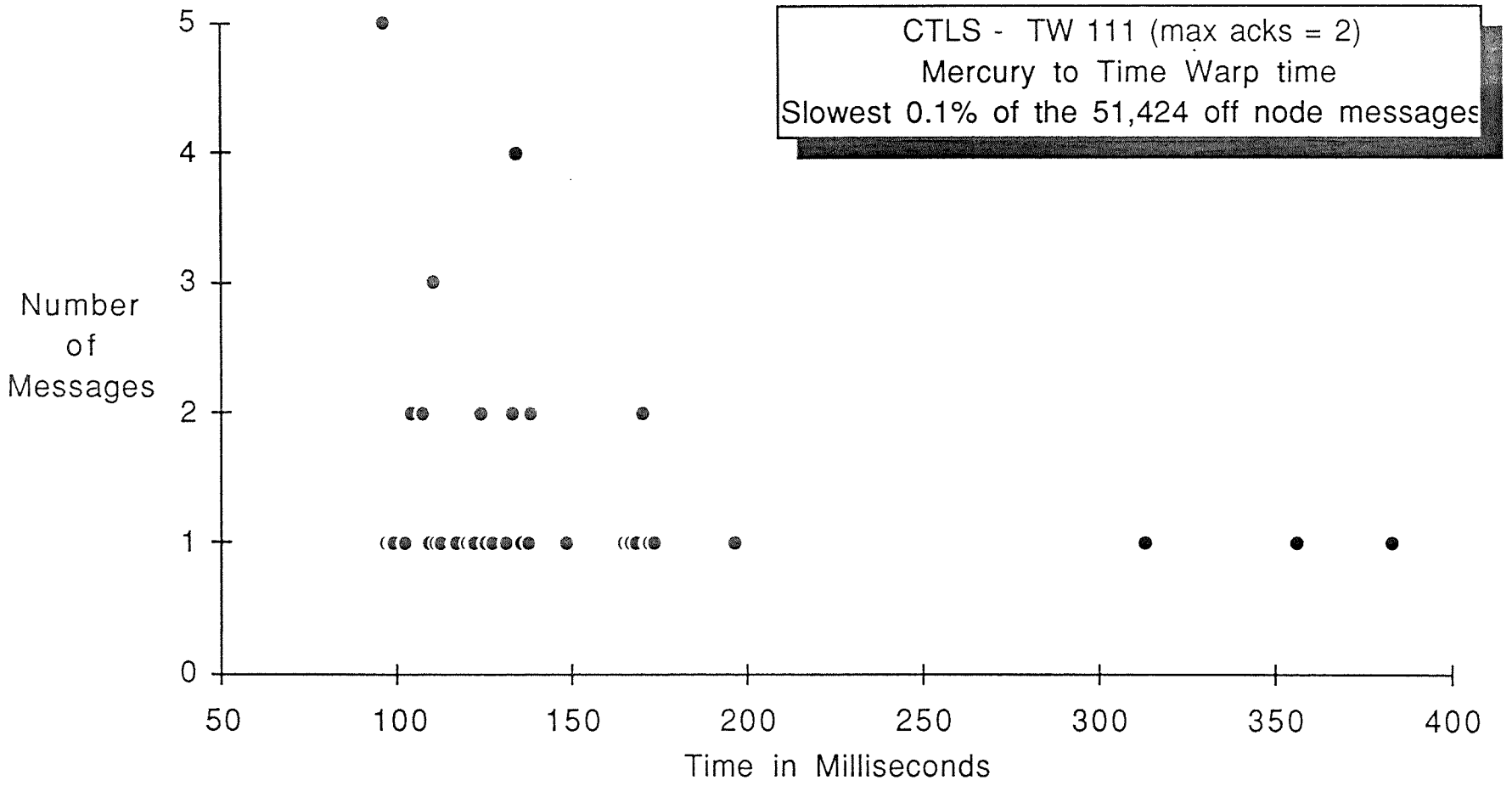
Appendix D
Mercury to Time Warp Times

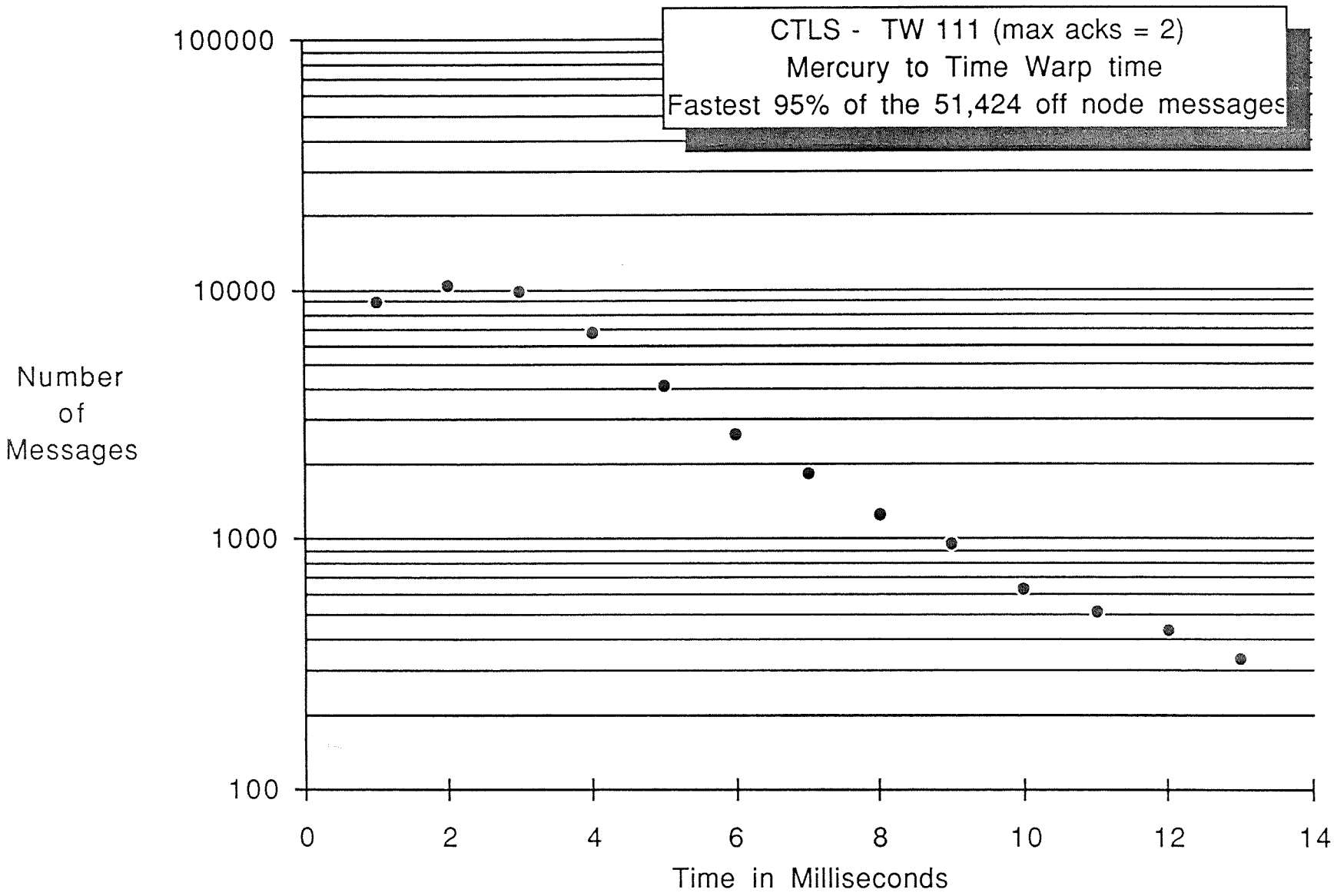
D-1: Fastest 95%, acks=2, graph (see D-4).
D-2: The 95% - 99.9% range, acks=2, graph (see D-5).
D-3: Slowest 0.1%, acks=2, graph (see D-6).

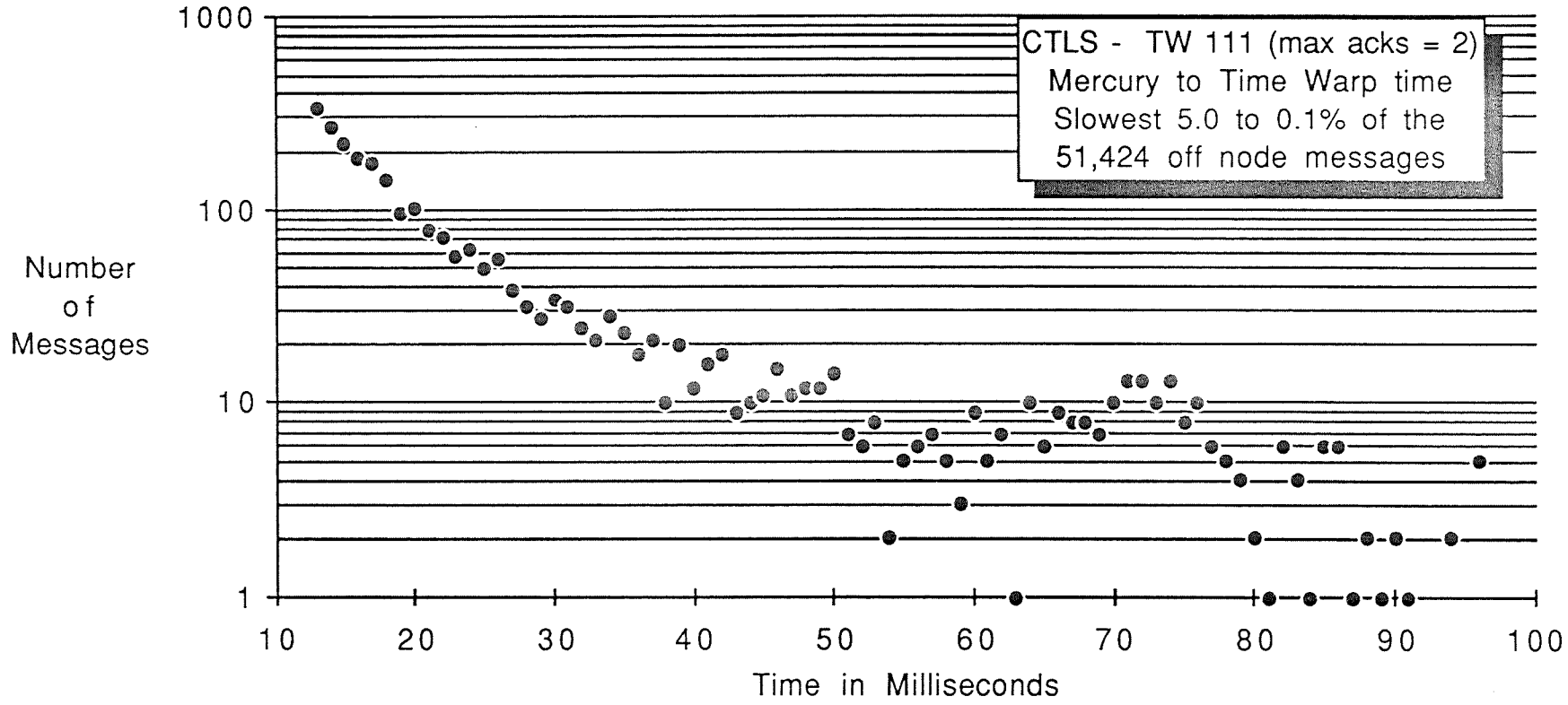
D-4: Fastest 95%, acks=2, semi-log graph (see D-1).
D-5: The 95% - 99.9% range, acks=2, semi-log graph (see D-2).
D-6: Slowest 0.1%, acks=2, semi-log graph (see D-3).

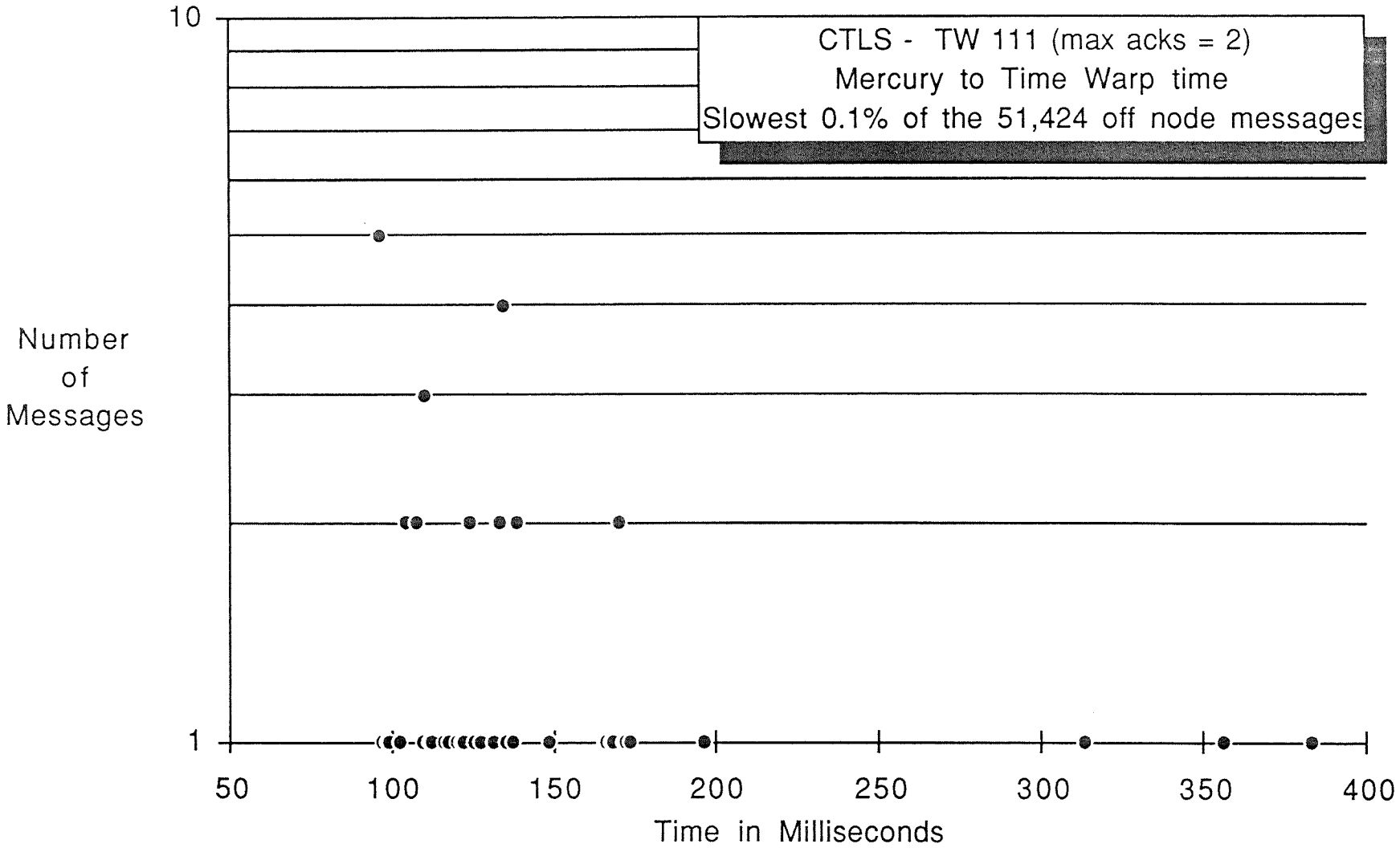

CTLS - TW 111 (max acks = 2)
Mercury to Time Warp time
Fastest 95% of the 51,424 off node messages
Number of Messages
12000
10000
8000
6000
4000
2000
0
0
2
4
6
8
10
12
14
Time in Milliseconds


CTLS - TW 111 (max acks = 2
Mercury to Time Warp time
Slowest 5.0 to 0.1% of the
51,424 off node messages
Number of Messages
350
300
250
200
150
100
50
0
10
20
30
40
50
60
70
80
90
100
Time in Milliseconds


CTLS - TW 111 (max acks = 2)
Mercury to Time Warp time
Slowest 0.1% of the 51,424 off node messages
Number of Messages
0
1
2
3
4
5
50
100
150
200
250
300
350
400
Time in Milliseconds


CTLS - TW 111 (max acks = 2)
Mercury to Time Warp time
Fastest 95% of the 51,424 off node messages
100000
10000
1000
100
Number of Messages
0
2
4
6
8
10
12
14
Time in Milliseconds


CTLS - TW 111 (max acks = 2)
Mercury to Time Warp time
Slowest 5.0 to 0.1% of the
51,424 off node messages
1000
100
10
1
Number of Messages
10
20
30
40
50
60
70
80
90
100
Time in Milliseconds


CTLS - TW 111 (max acks = 2)
Mercury to Time Warp time
Slowest 0.1% of the 51,424 off node messages
10
1
Number of Messages
50
100
150
200
250
300
350
400
Time in Milliseconds

Appendix E
Queueing Delays
Time in Mercury's Receive Queue
Length of Mercury's Receive Queue

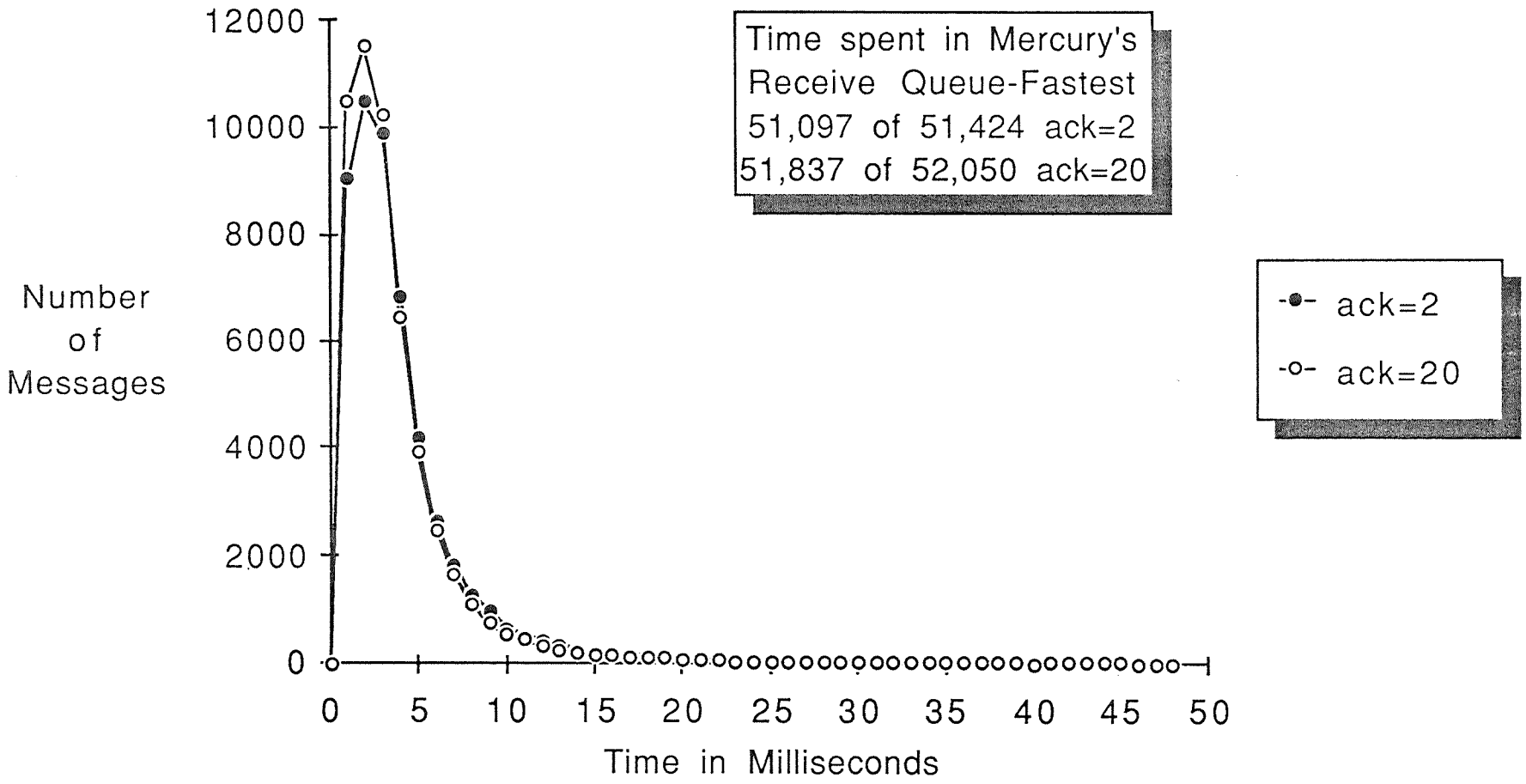
E-1: Fastest times, acks=2 vs acks=20, graph (see E-2).
E-2: Fastest times, acks=2 vs acks=20, semi-log graph (see E-1).

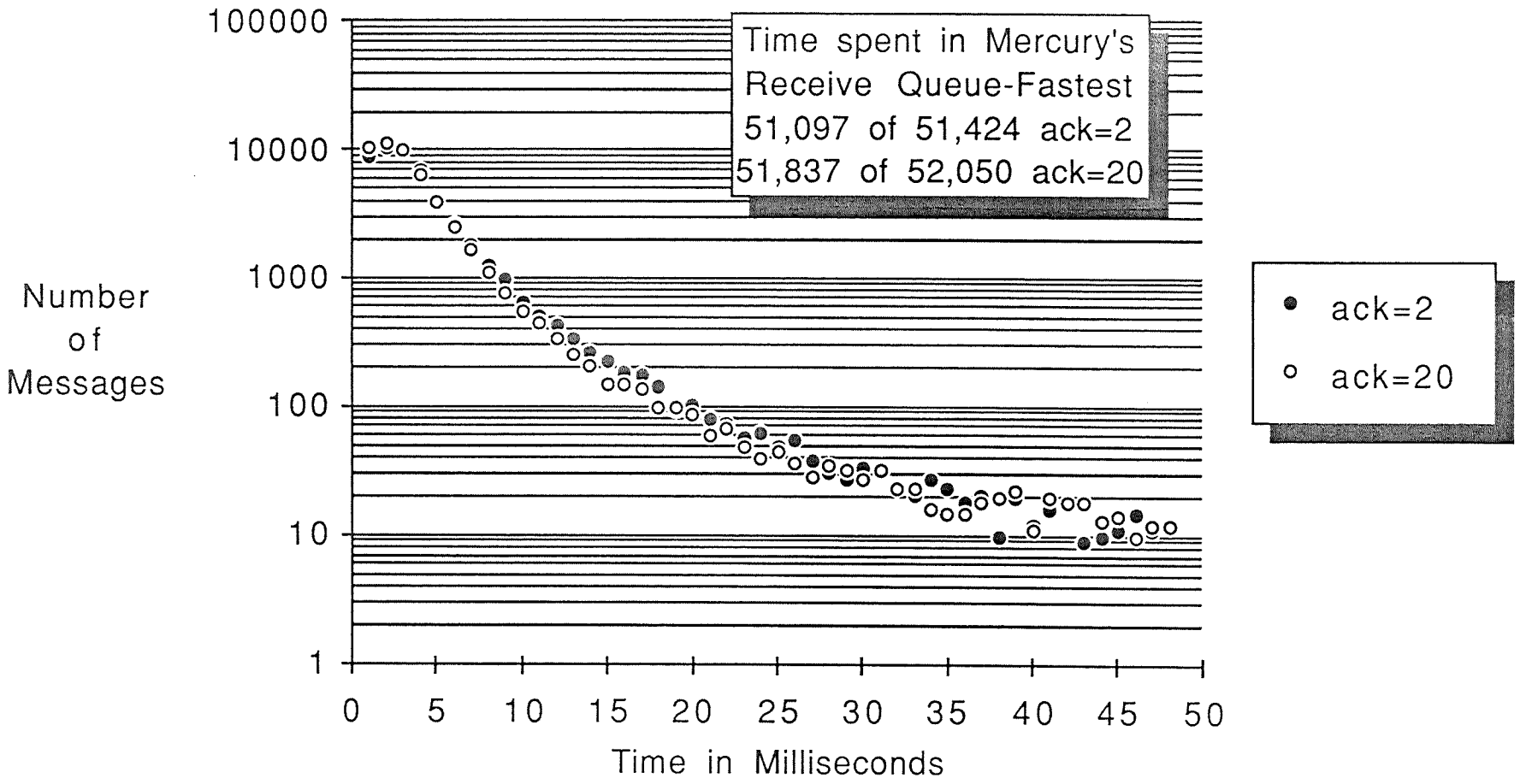
E-3: Slowest times, acks=2 vs acks=20, graph (see E-4).
E-4: Slowest times, acks=2 vs acks=20, semi-log graph (see E-3).

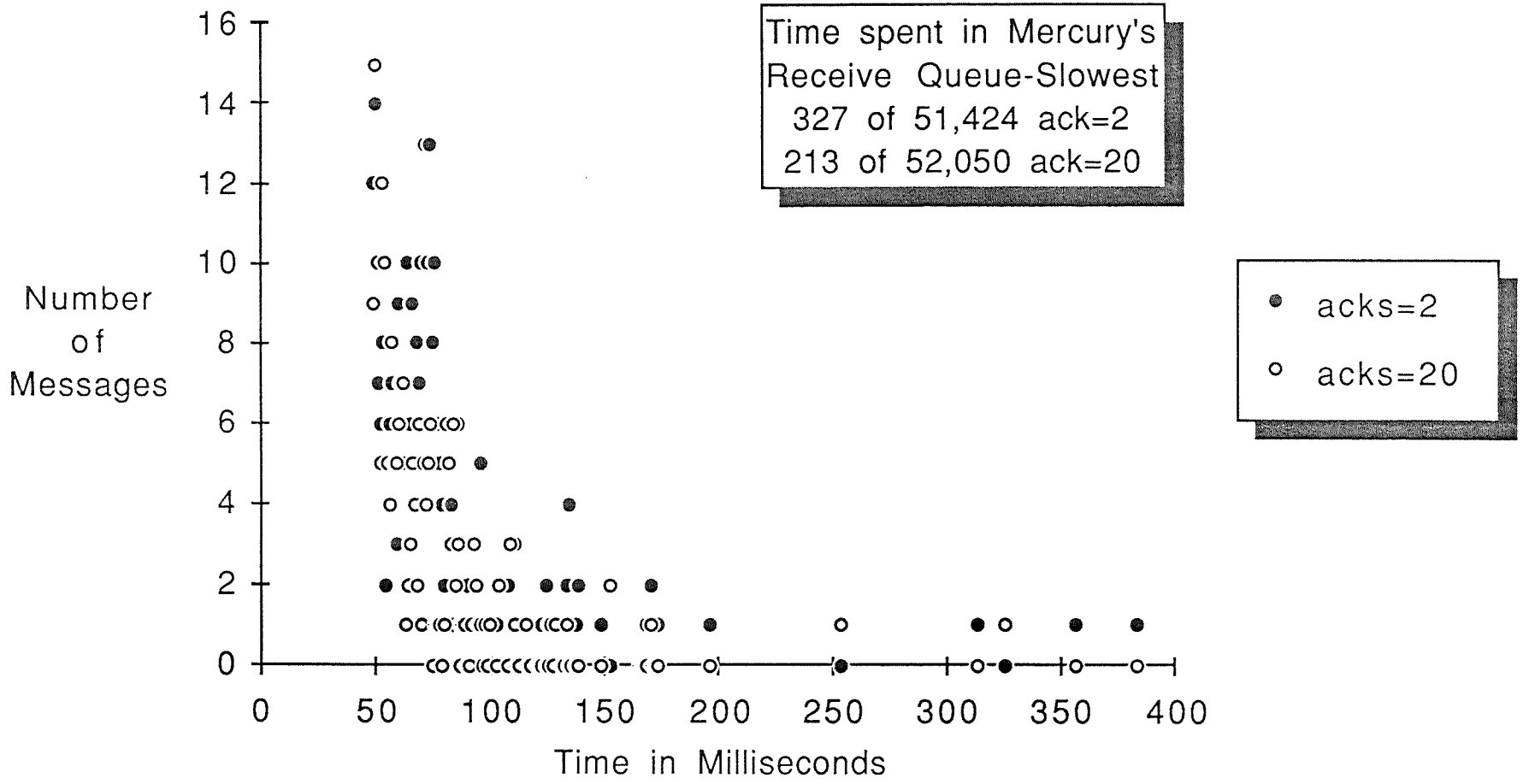
E-5: Length, acks=2, graph (see E-6).
E-6: Length, acks=2, semi-log graph (see E-5).

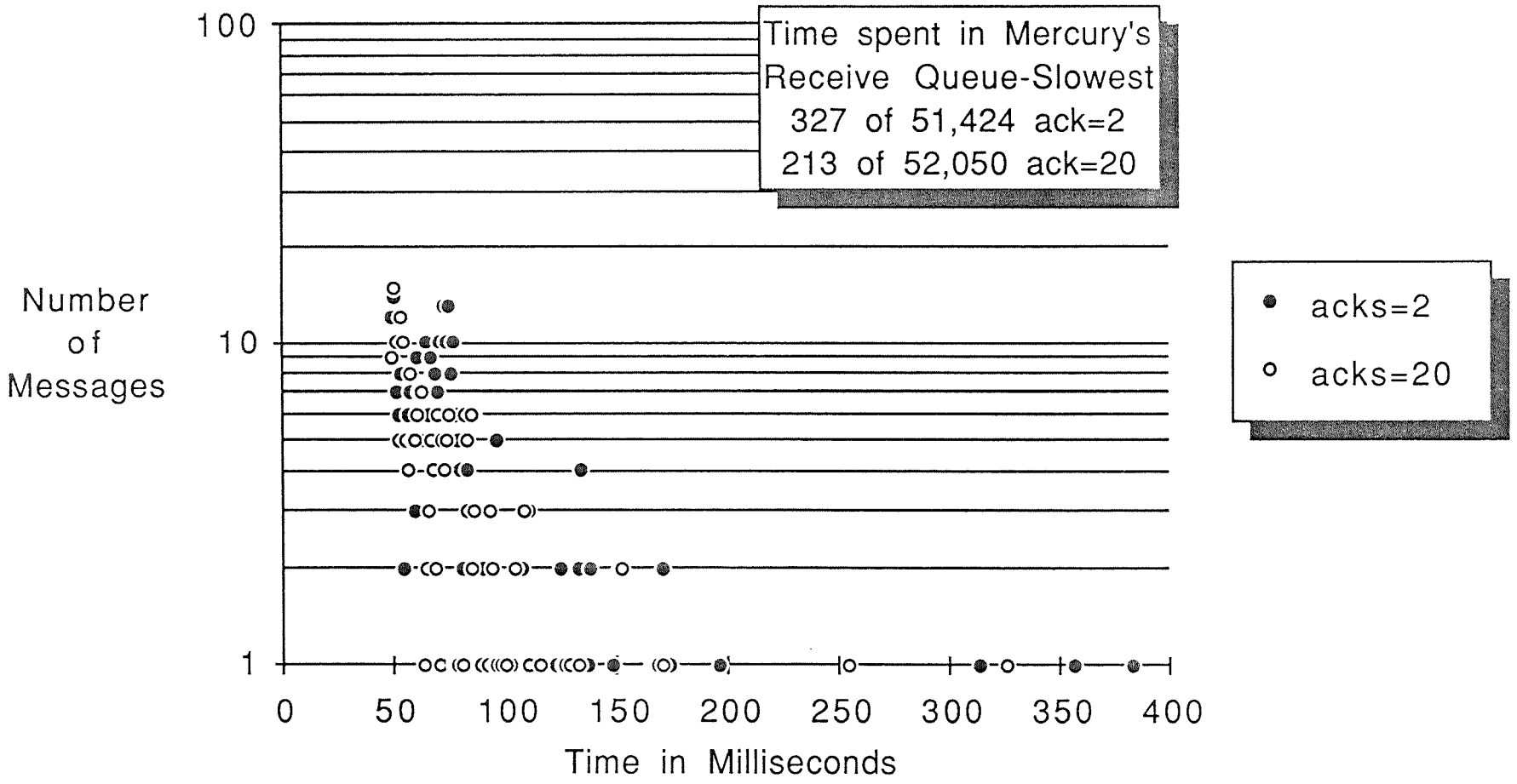
E-7: Length, acks=2 vs acks=20, semi-log graph.

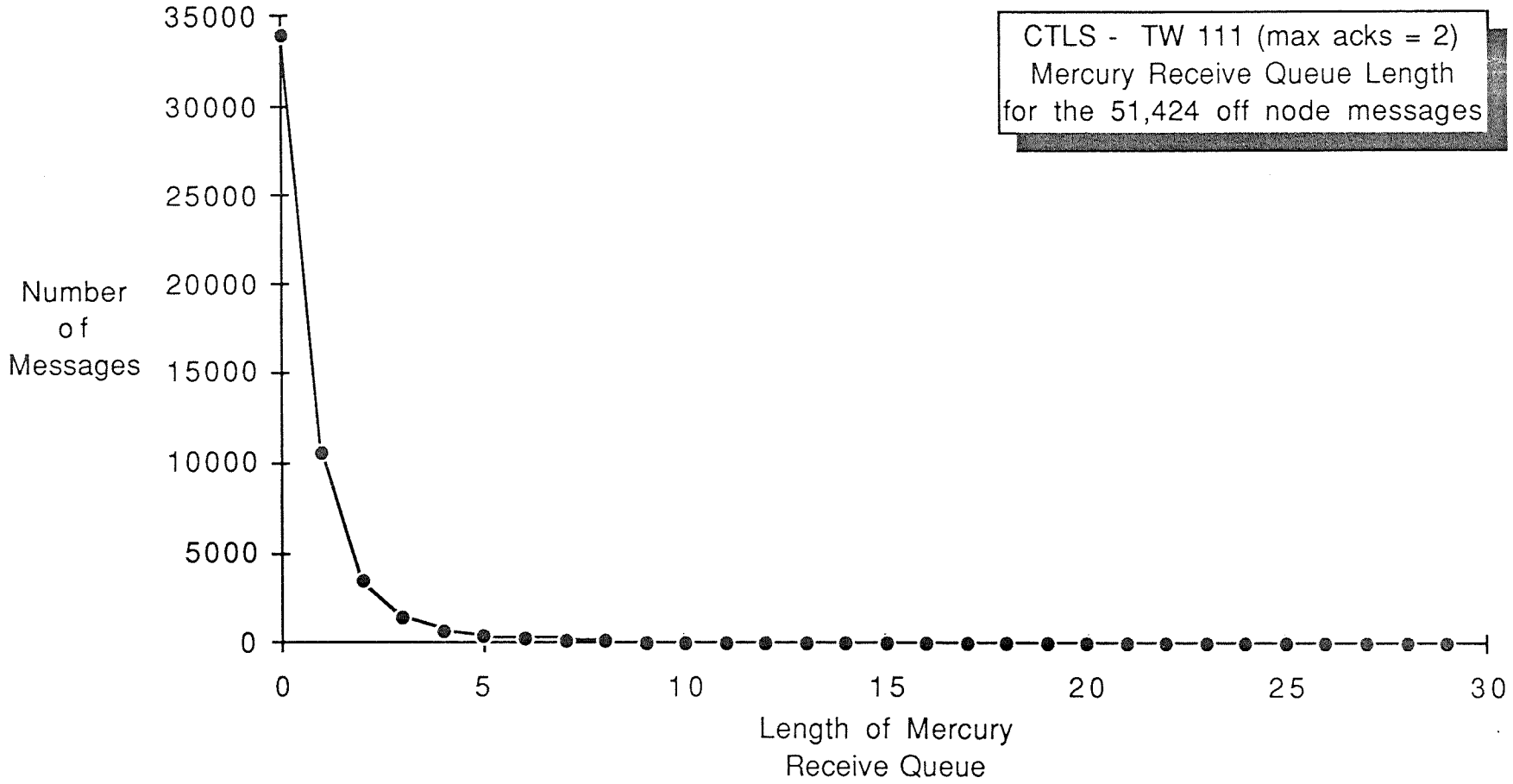
E-8: On node length, acks=20, semi-log graph.
E-9: On node length, acks=2, semi-log graph.

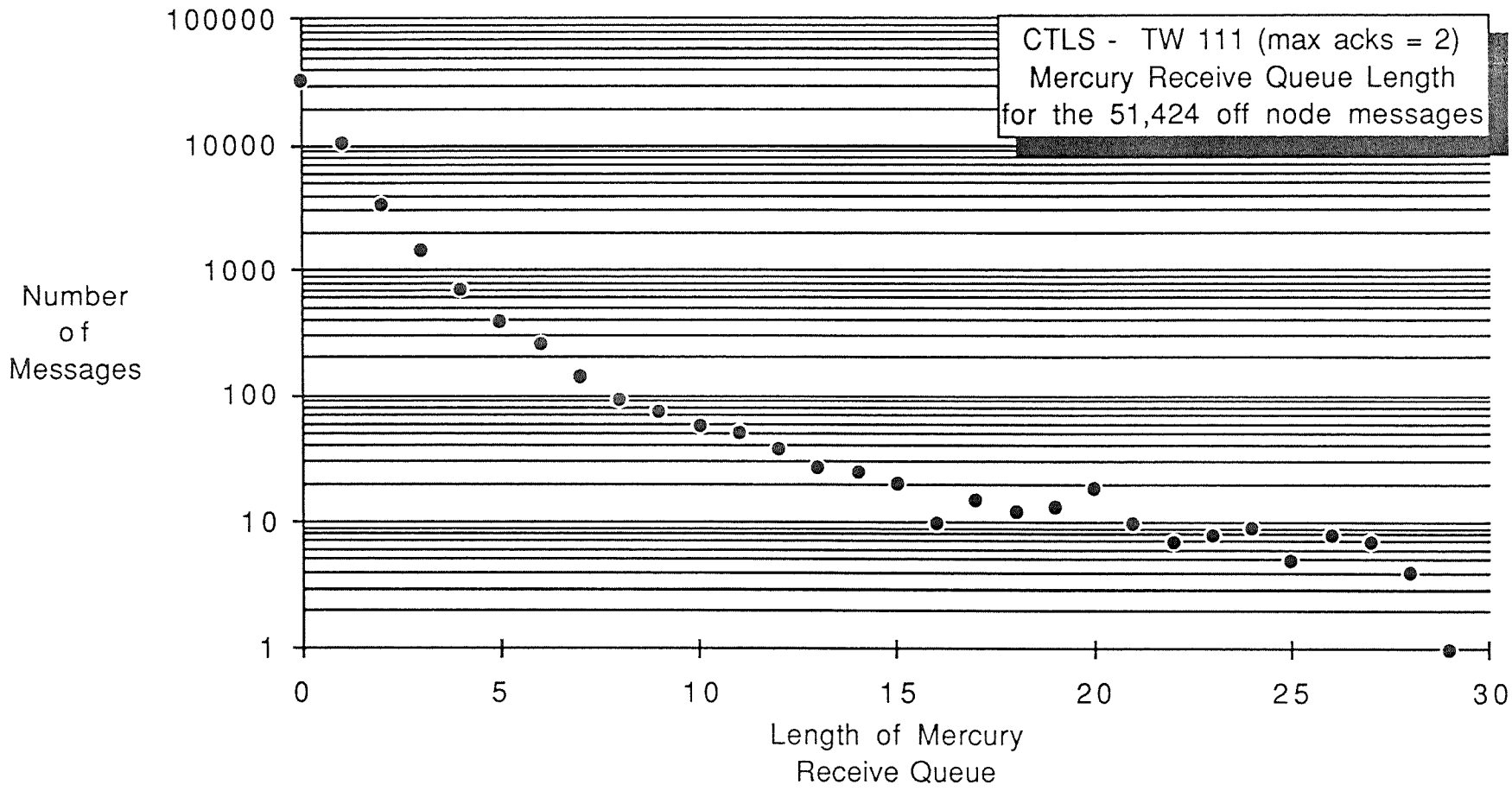

Time spent in Mercury's
Receive Queue-Fastest
51,097 of 51,424 ack=2
51,837 of 52,050 ack=20
ack=2
ack=20
Number
of
Messages
12000
10000
8000
6000
4000
2000
0
0
5
10
15
20
25
30
35
40
45
50
Time in Milliseconds

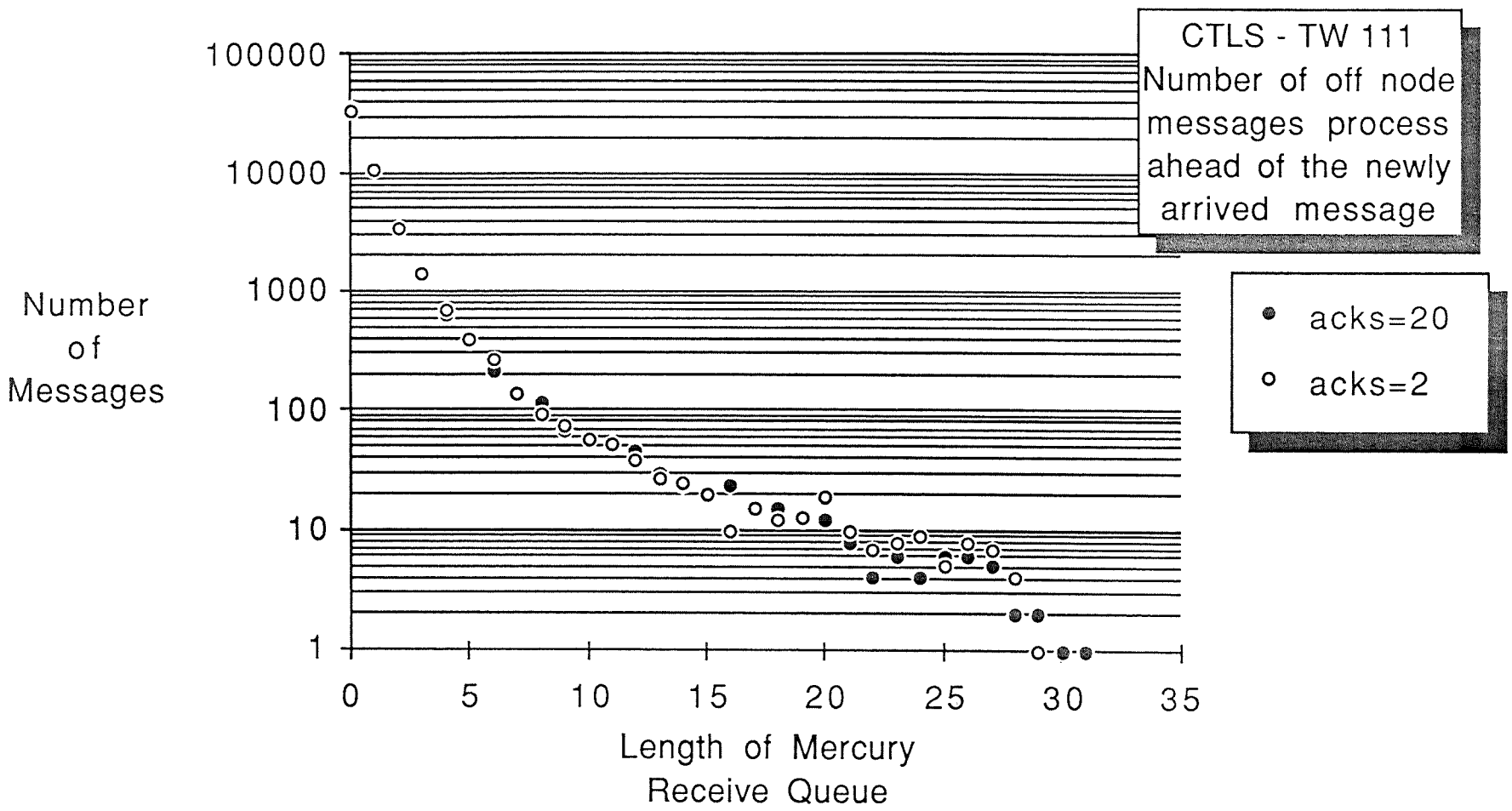

Time spent in Mercury's Receive Queue-Fastest
51,097 of 51,424 ack=2
51,837 of 52,050 ack=20
Number of Messages
100000
10000
1000
100
10
1
0
5
10
15
20
25
30
35
40
45
50
Time in Milliseconds
ack=2
ack=20

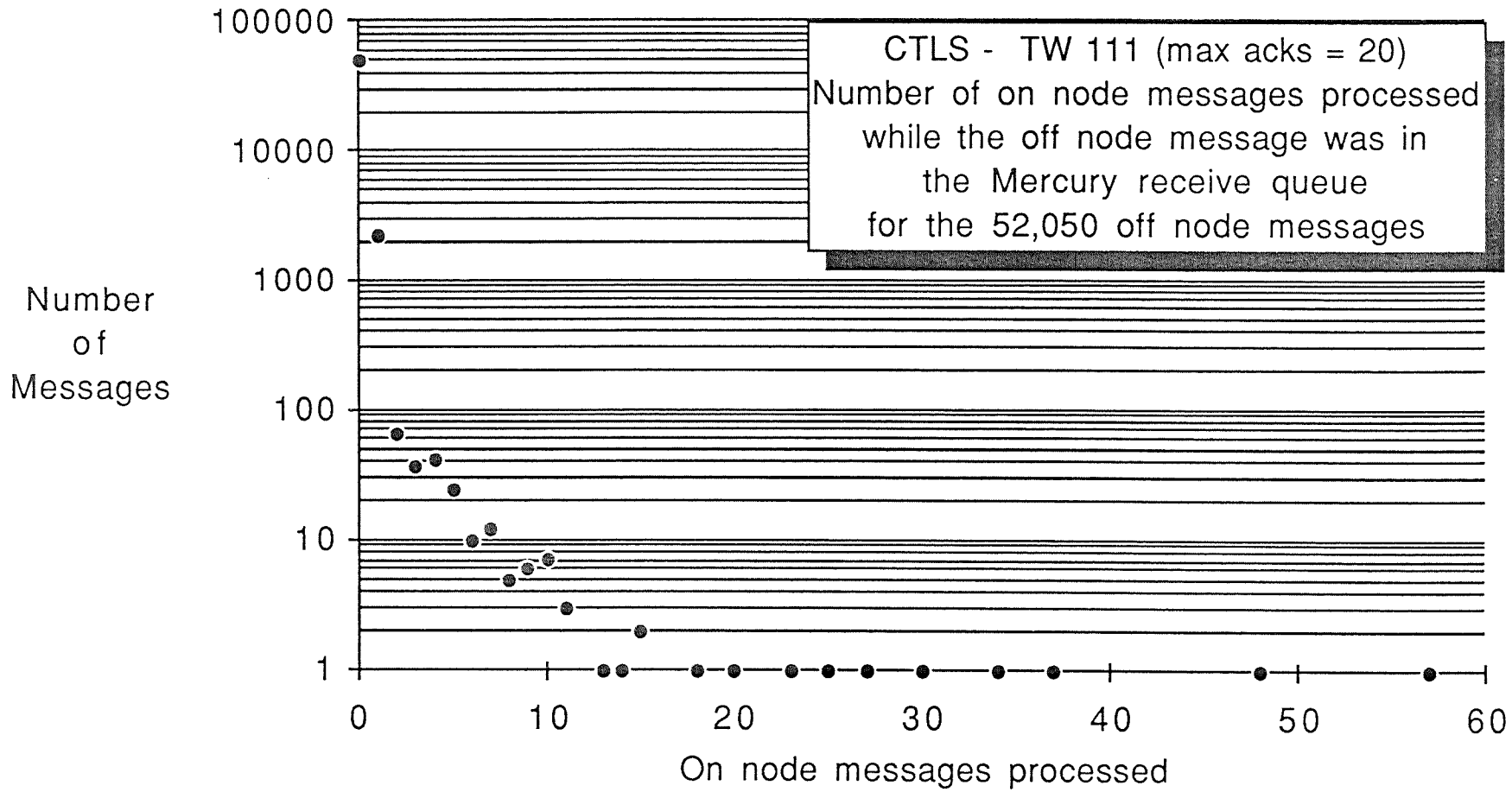

Time spent in Mercury's
Receive Queue-Slowest
327 of 51,424 ack=2
213 of 52,050 ack=20
acks=2
acks=20
Number
of
Messages
16
14
12
10
8
6
4
2
0
0
50
100
150
200
250
300
350
400
Time in Milliseconds

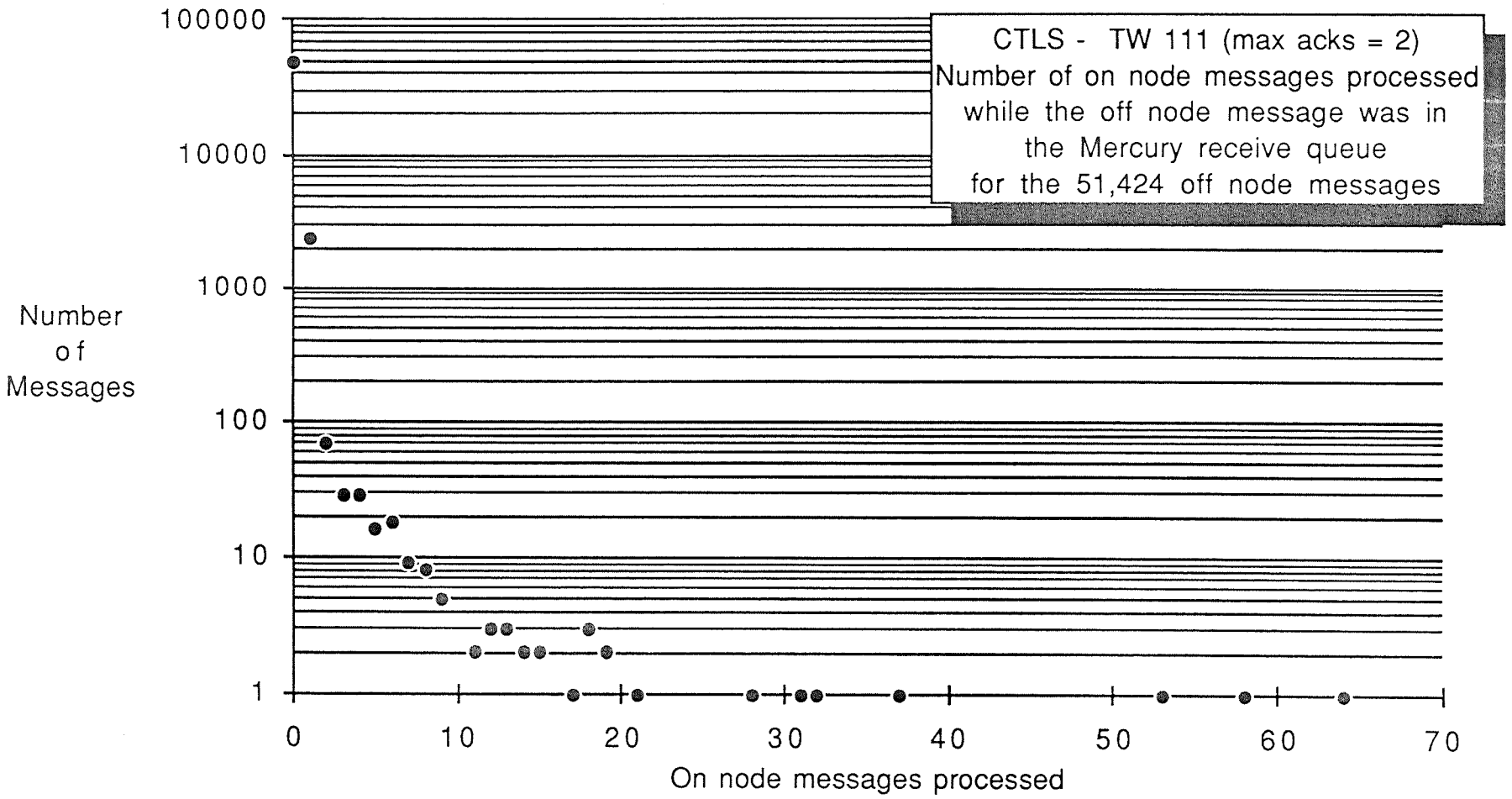

Time spent in Mercury's
Receive Queue-Slowest
327 of 51,424 ack=2
213 of 52,050 ack=20
100
10
1
Number
of
Messages
0
50
100
150
200
250
300
350
400
Time in Milliseconds
acks=2
acks=20


CTLS - TW 111 (max acks = 2)
Mercury Receive Queue Length
for the 51,424 off node messages
35000
30000
25000
20000
15000
10000
5000
0
Number of Messages
0
5
10
15
20
25
30
Length of Mercury Receive Queue


CTLS - TW 111 (max acks = 2)
Mercury Receive Queue Length
for the 51,424 off node messages
100000
10000
1000
100
10
1
Number of Messages
0
5
10
15
20
25
30
Length of Mercury Receive Queue


CTLS - TW 111
Number of off node messages process ahead of the newly arrived message
acks=20
acks=2
Number of Messages
100000
10000
1000
100
10
1
0
5
10
15
20
25
30
35
Length of Mercury Receive Queue


CTLS - TW 111 (max acks = 20)
Number of on node messages processed
while the off node message was in
the Mercury receive queue
for the 52,050 off node messages
Number
of
Messages
100000
10000
1000
100
10
1
0
10
20
30
40
50
60
On node messages processed


CTLS - TW 111 (max acks = 2)
Number of on node messages processed
while the off node message was in
the Mercury receive queue
for the 51,424 off node messages
100000
10000
1000
100
10
1
Number
of
Messages
0
10
20
30
40
50
60
70
On node messages processed