FAA-RD-73-126

Project Report
ATC-19

Interrogation Scheduling Algorithms for a Discrete Address Beacon System

A. Spiridon
A.D. Kaminsky

17 October 1973

Lincoln Laboratory
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LEXINGTON, MASSACHUSETTS

Prepared for the Federal Aviation Administration,
Washington, D.C. 20591.

This document is available to the public through
the National Technical Information Service,
Springfield, VA 22161

TECHNICAL REPORT STANDARD TITLE PAGE

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
FAA-RD-73-126		

4. Title and Subtitle
Interrogation Scheduling Algorithms for a Discrete Address Beacon System

5. Report Date
17 October 1973

6. Performing Organization Code

7. Author(s)
A. Spiridon and A. D. Kaminsky

8. Performing Organization Report No.
ATC-19

9. Performing Organization Name and Address
Massachusetts Institute of Technology
Lincoln Laboratory
P. O. Box 73
Lexington, Massachusetts 02173

10. Work Unit No.
Project no. 034-241-012

11. Contract or Grant No.
IAG-DOT-FA72WAI-261

12. Sponsoring Agency Name and Address
Department of Transportation
Federal Aviation Administration
Systems Research and Development Service
Washington, D. C. 20591

13. Type of Report and Period Covered
Project Report

14. Sponsoring Agency Code

15. Supplementary Notes
The work in this document was performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology under Air Force Contract F19628-73-C-0002.

16. Abstract

This report describes several scheduling algorithms that may form part of the interrogation management function of a discrete address beacon system. These include scheduling algorithms that can handle unequal message lengths and types which can schedule a message very rapidly (dynamic scheduling). The algorithms are evaluated in terms of the computation required to execute them and their packing efficiencies.

17. Key Words
Discrete Address Beacon System
Dynamic Scheduling
Interrogation Management
Scheduling Algorithm
Unequal Message Scheduling

18. Distribution Statement
Availability is unlimited. Document may be released to the National Technical Information Service, Springfield, Virginia 22151, for sale to the public.

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	108	4.25 HC 1.45 MF

Form DOT F 1700.7 (8-69)

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (Continued)

LIST OF TABLES

1.0 INTRODUCTION

The Discrete Address Beacon System (DABS) is designed to permit calls (interrogations) to individual aircraft. The DABS system employs an Interrogation Management Function (IMF) to reserve time on the RF channel for calls to, and replies from, these aircraft (referred to as targets), making sure that they do not overlap. The performance of the IMF is important in determining the sensor capacity, the quality of the replies received from the aircraft for direction finding, and the reliability of the surveillance and communication functions performed by DABS.

An important adjunct to the IMF is the scheduling algorithm. (See Figure 1.1.) Its input is a list of targets with their ranges. Its output assigns times on the RF channel for transmission of interrogations to aircraft and reserves times for the expected replies so that messages* will not overlap. The algorithms must also assure a minimum time between interrogations and minimum computation time.

A good scheduling algorithm increases "channel capacity" by efficiently sequencing the non-overlapping DABS messages into the available channel time. To do this, the algorithm makes use of the range of the aircraft to determine when the reply from an aircraft is expected.

*In this report, the term "message" is used to mean the entire RF transmission, either uplink or downlink. Correspondingly, "message length" is used to mean RF transmission duration and is measured in units of time (μsec).

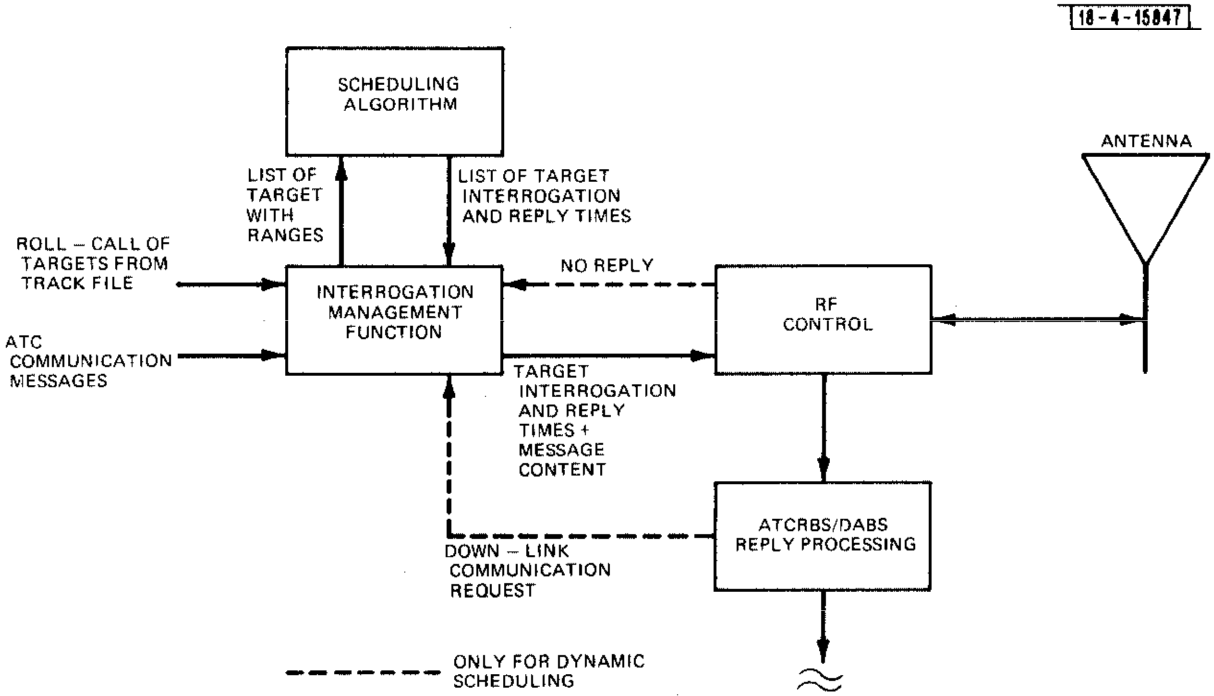


Fig. 1.1. Sensor processing functions (in part).

The first section of this report presents a general description of an IMF including how it interacts with the rest of the sensor software system and how the IMF handles a rotating antenna system versus an agile beam system. The next section describes the performance measures which are important in a scheduling algorithm and how these performance measures are determined. Section 4 describes the constraints and uncertainties which must be considered in designing the scheduling algorithms. Four different scheduling algorithms, meeting the needs of different systems, are described and evaluated in the following sections. The appendices give examples of Interrogation Management Functions. A detailed description is also given of the design of the scheduling algorithms.

2.0 THE INTERROGATION MANAGEMENT FUNCTION

One function of the IMF is the proper scheduling of communication between the sensor and all aircraft assigned to that sensor. The ATC system and the DABS sensor reply processor provide the position and identification of each aircraft so that a roll-call of targets may be maintained in the sensor track file. The uplink and downlink message lengths are also provided. With this information, the IMF produces the next transmission time and next expected reply time for each target. This is the schedule required by the sensor control program.

In the rotating beam sensor, the IMF restricts the interrogation of targets to the interval during which the targets are expected to be in the beam. To do this the IMF divides targets into groups with the scheduling algorithm producing a separate schedule for each group. Such a schedule consists of the interrogation time, as well as expected reply time, for each target in the group. Essentially, a scheduling algorithm accepts from the IMF, a list of targets with their unique identifications and ranges and programs the time of interrogation and reply for each of them. The IMF then sets the beginning of the schedule of each group to coincide with the beginning of an interval of time. The group of targets and the interval of time are chosen such that the beam covers the targets during its interrogation and reply. The schedule produced by the IMF is used by other parts of the sensor software to control the RF transmitter/receiver.

Another function executed by the IMF is the reinterrogation of targets failing to reply to previous interrogations, or those which have requested an additional communication message while the beam is still aligned with the target. To do this, the IMF makes use of a scheduling algorithm to "dynamically" assign an interrogation time for the target and to predict the expected time at which the target will reply. In this case, the interrogation schedule is computed a very short time in advance of the actual interrogation. This makes it important that the scheduling algorithm allow sufficient time of system delays.

The scheduling algorithm which is used in conjunction with the IMF for a rotating beam sensor has to provide dynamic or short term scheduling, as well as long term scheduling. "Dyno" is an example of a scheduling algorithm which performs both these functions. In it, scheduling of each target is carried out by itself with no need for information about other targets. This feature is important for dynamic scheduling when the targets are reinterrogated individually within a short time after the receipt of a previous reply. Dyno also schedules with relatively low computation overhead, an important feature, for this overhead represents inefficiency for every group of targets, and a large number of groups must be scheduled in a rotating beam sensor. A description of Dyno is given in Appendix A.

In the case of an agile beam sensor, a greater degree of freedom exists for the IMF since the beam can be directed to any azimuth at a given time in the scan. The scheduling algorithm function may then accept from the IMF a large group of targets which may be as many as half the targets which may be as many as half the targets on the roll-call. The schedule produced must be accommodated by a number of separate, fixed length intervals. This task is done by two scheduling algorithms in cascade. The first algorithm performs what is called Primary Scheduling [1]. The output of the algorithm is one long schedule which is divided into fixed length intervals by "secondary scheduling" [2]. The IMF delivers the final schedule, along with the azimuth of the targets, to other parts of the sensor processor to control the RF transmitter-receiver. Details of the Interrogation Management Function for an agile beam sensor are given in Appendix A.

The primary scheduling function can be done by either the "Full-Ring" algorithm which has been described in a previous Project Report [3] or the "Close-Fit" algorithm. (See Figure 2.1.) These algorithms both schedule messages to large groups of targets before the antenna beam is pointing toward any of these targets, e.g., schedule once or twice per scan. We refer to this procedure as scheduling ahead of scan. These algorithms have the ability to reorder the target interrogation sequence to attain tight packing of messages. Although they have relatively high computation overhead, this is not critical for the agile beam sensor IMF, as the overhead is encountered only a few times per scan. The Full-Ring algorithm would be used in the DABS system where message lengths were nearly equal. It is discussed in Chapter 5 and in more detail in Appendix B. The Close-Fit algorithm would be used in systems where message lengths may be unequal due to the use of varying numbers of bits in communication messages or different bit rates on the uplink and downlink. The Close-Fit algorithm is discussed in Chapter 6 and in more detail in Appendix C. Loop-Loop is an algorithm that performs the secondary scheduling function in the IMF of an agile beam sensor. It is discussed in Chapter 7 and in more detail in Appendix E.

Algorithm	Computation Overhead (In machine cycles)	Computation Load per Target (In machine cycles)	Packing Efficiency* (In %)	System Applications
Full-Ring	≈ 3500	≈ 58	≈ 94	Agile beam sensor for equal messages performs first part of scheduling.
Close-Fit	≈ 4200	≈ 72	≈ 86	Agile beam sensor with unequal messages; performs first part of scheduling.
Loop-Loop	None	≈ 7	≈ 95	Agile beam sensor; used in tandem with Full-Ring or Close-Fit Algorithm.
Dyno	None	≈ 50	≈ 44 **	Rotating beam sensor; used for dynamic scheduling and regular scheduling.

* Ratio of total message length to channel time between first message and last message.

** Maintains this performance even with a small number of targets.

Figure. 2.1. Representative characteristics of scheduling algorithms.

3.0 PERFORMANCE MEASURES

From a system point of view, packing efficiency, computation effort and core requirements are important measures of performance of a scheduling algorithm. For a set of scheduled targets "packing efficiency" is defined as the ratio of the sum of message lengths scheduled (units of time) to the channel time from the beginning of the first interrogation to the end of the last reply. In order to measure the performance, the algorithms under consideration were implemented in Mix [4], a low level language, that exists on a hypothetical Mix machine and in a corresponding manner in Fortran. In order to measure the computation effort in Mix, counters were incorporated in the Fortran implementation for the corresponding branches of the Mix implementation. In calculating the computation effort in terms of Mix machine cycles, each counter is scaled by the computation cycles required to compute the corresponding branch in the Mix implementation. The algorithms were evaluated by exercising them a large number of times, and building statistics for packing efficiency and computation effort. (See Figure 2.1.) The input to the schedulers were sets of target ranges randomly generated to meet the statistics of the input. The defining parameters for these inputs were the number of targets, the message length, the range over which targets are distributed and the distribution of target ranges.

The core requirements for an algorithm are given indirectly in terms of the list structures required for the algorithms.

The response time of an algorithm may be defined as the processing time required from the receipt of input target list to the time the schedule is available. In the Full-Ring and Close-Fit algorithms, this would be the processing time for the set of all targets, while for Dyno it would be the processing time per target. This measure is important in designing the DABS system software, since it is a measure of the dynamics of the scheduling computation.

4.0 DEFINITION OF PROBLEM

The input to an Interrogation Management Function is a set of targets and their predicted position in (ρ, θ) coordinates as obtained from the sensor tracker. The outputs are the interrogation times and the expected reply times for each target.

Besides this basic set of information, other inputs to the scheduler include the following:

Parameter	Effect or Reason Needed
The range over which the targets are to be interrogated	In some scheduling it may affect the data base.
Range distribution of targets	This may affect the scheduler performance.
Uncertainty in range of target	Equivalent to uncertainty in reply time; accommodated by reserving a longer time for the reply message.
Uplink and downlink message lengths	Needed to determine the amount of time to reserve for a message on the channel. The ratio of message lengths affects the choice of the scheduling algorithm of maximum packing efficiency.
Number of targets	Packing efficiency decreases for small number of targets; for large numbers of targets the efficiency reaches an asymptotic value.
Multiplicity of fixed calls to a target	Increases in proportion to the number of calls to be scheduled ahead of scan.

Additional inputs relevant only to a rotating beam system include:

Target azimuth	Determines the time the target is available for interrogation.

Parameter	Effect or Reason Needed
Azimuth uncertainty	Affects azimuth sector over which the sensor can interrogate the target with certainty.
Azimuth distribution ("bunching") of targets	Bunching, the ratio of the maximum number of targets per unit azimuth sector to the mean number of targets per unit azimuth sector, determines the total target handling capacity in terms of the scheduler handling capacity in a small interval of time.
Sensor beamwidth	Determines target dwell time; usable dwell time decreased by uncertainty in target azimuth accuracy.
Interrogation message separation constraints	To insure that ATCRBS transponders are suppressed it is necessary that successive interrogation are separated by a least 50 μsec. This allows the majority of ATCRBS transponders to go out of suppression before being suppressed again.

Note that for a rotating beam antenna, the scheduler is restricted to scheduling a target in the time interval that the target is covered by the beam. This restriction is not present in an agile beam. In an agile beam, the switching time between two beam positions would have to be accounted for by adding it to the message duration.

Thus, as far as a scheduling algorithm design is concerned, the problem is as follows.

A set of targets with given ranges are to be scheduled in a defined interval of time. The message length used is the actual message length plus an added increment that accounts for uncertainties and system function. A scheduling algorithm is then constructed by one of two distinct methods. In the first, the targets in the set would be scheduled in any order. This would apply in the case of scheduling ahead of scan. In the other, targets are to be scheduled in a fixed order and within short notice, that is, the targets are dynamically scheduled.

An added constraint that may be imposed on scheduling algorithms is the requirement for minimum time between successive interrogations.

Algorithms described in this report are based on a random access memory with a sequential processor.

5.0 FULL-RING ALGORITHM

A. PURPOSE

The Full-Ring algorithm was first proposed in reference [3]. It is designed for scheduling targets ahead of scan. Each target is scheduled a single time. Before scheduling targets, the algorithm reorders the sequence in which targets are scheduled to improve packing efficiency. It is best suited for interrogation and reply messages that are equal or nearly equal to one fixed length.

The algorithm needs information about all the targets to be scheduled before scheduling begins. An appropriate use would be for an agile beam system, since it provides high packing efficiency for a large number of targets, and it requires low computation time per target.

B. BASIC CONCEPT

The Full-Ring algorithm output is in schedule cycles where a schedule cycle is a group of target interrogation times followed by their reply times. An example of a cycle is shown in Figure 5-1. In this scheme, the computation required to schedule a target is simple once the sequence in which targets to be scheduled is established. The scheduling of a target next to a target just scheduled simply entails calculating any delay required so that the target reply would not overlap the last target reply. Also, a check is required so that the interrogation is over before receiving the first reply following the last scheduled interrogation. In case of overlap, the interrogation is placed after the last scheduled target reply to start a new cycle.

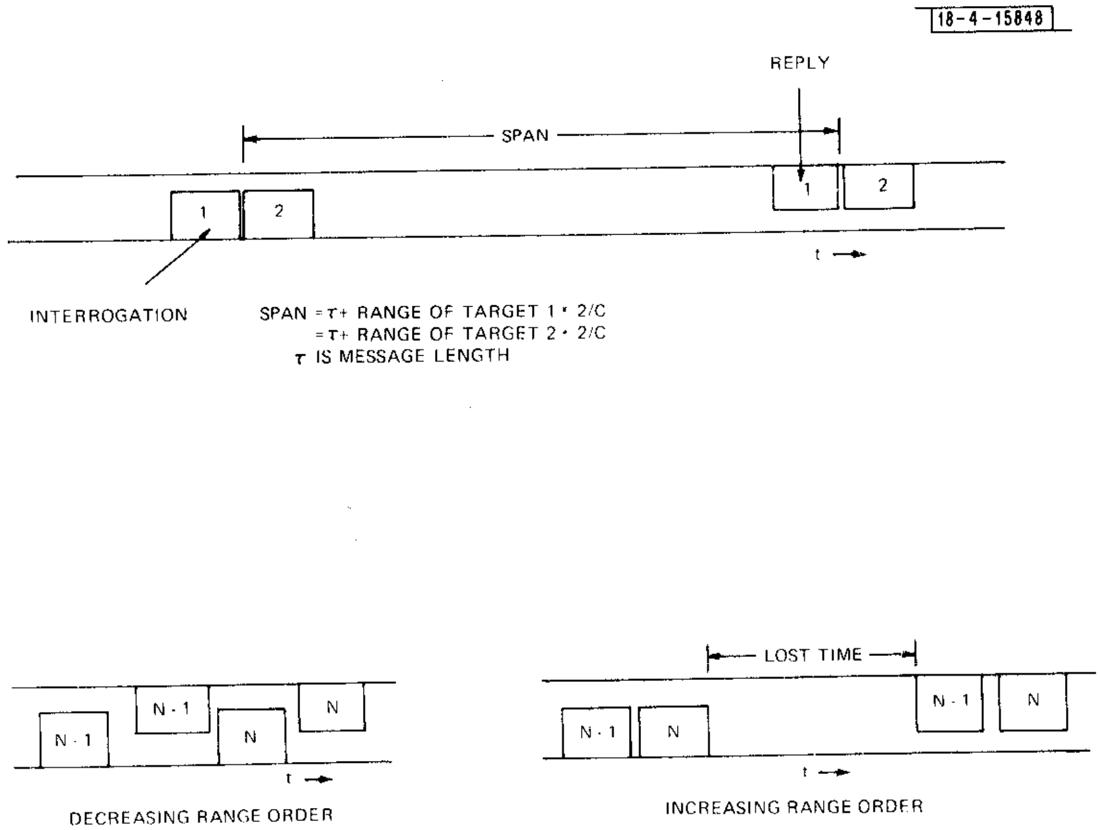


Fig. 5.1. Illustrating a need for range ordering.

To do this, the algorithm assumes that the interrogations and the replies on the channel will be in the sequence the targets are scheduled. It also assumes that no isolated interrogation will occur between the times reserved for replies from other targets.

Targets are scheduled in range order to reduce the loss in time between adjacent targets. Decreasing range order is used to reduce the loss on the channel in scheduling the last few targets. Figure 5-2 illustrates the basic concept.

A compromise method of ordering the target list, over strict sequential ordering, is one which sorts the targets by range into range bins. This decreases the computation effort characteristic from one which increases as the square of the number of targets to one which increases linearly with the number of targets. The propagation delay is derived from the range associated with each target bin, and the target is considered to be at a range corresponding to the beginning of this bin. The interval of time reserved for the reply is increased by an amount equivalent to the width of the bin, so that the actual target range is irrelevant. This method is computationally attractive for a large set of targets where strict sequential ordering becomes computationally expensive.

Thus, at the beginning, the targets are bin sorted. The first target scheduled starts a schedule cycle, an example of which is given in Figure 5-1. The position of its expected reply is used to check that other interrogations in the schedule cycle can fall before it. In scheduling a target (i + 1) in the cycle, the delay required to move the interrogation of target (i + 1) from its initial assumed position next to target (i) interrogation, so that its reply does not overlap the already scheduled target (i) reply, is obtained from Eq. (5.1).

18-4-15849

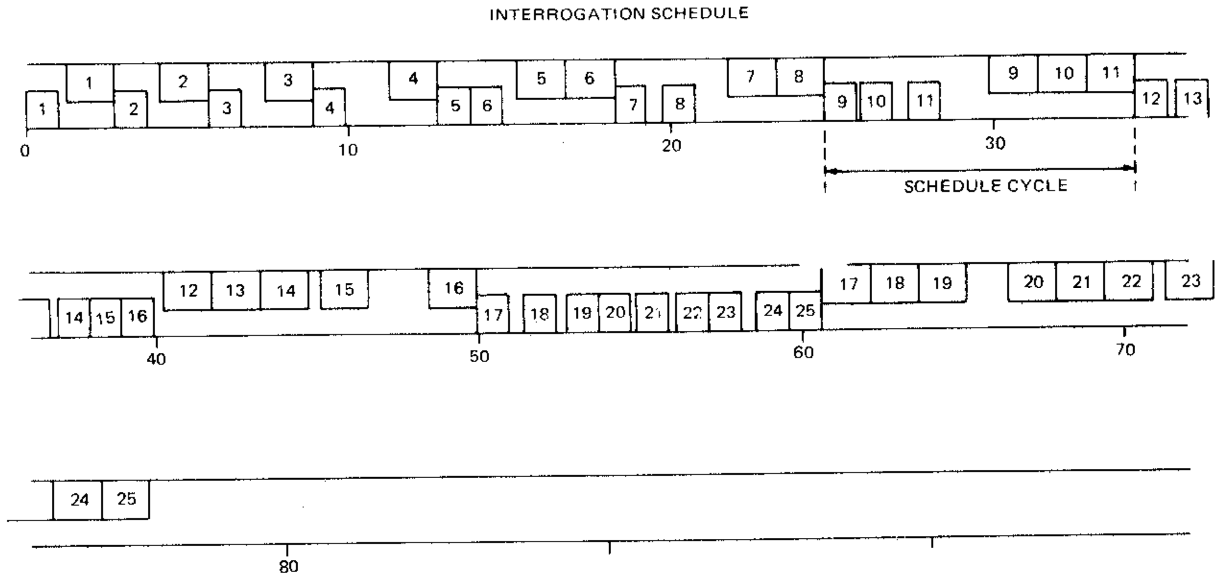


Fig. 5.2. An example of a cycle for Full-Ring.

$$\text{Delay} = 2\,(R_i - R_{i+1})/C - (\tau_I - \tau_R). \qquad (5.1)$$

R_i, R_{i+1} are the ranges of targets i and i + 1. τ_I, τ_R are the message lengths from interrogation and reply. If the delay comes out negative, no delay would be required.

More targets are placed in the group (cycle) until the time span of an interrogation overlaps the start of the reply of the first target in the cycle, at which time a new grouping (cycle) is started.

Linked List structures are used to store information about targets that fall in the same bin. This makes efficient use of core storage, for without the the linked structures it would be necessary to reserve, for each bin, a number of memory words equal to the maximum number of targets that could fall in a bin. This could increase the core required for bin sort by an order of magnitude above the one required using linked lists.

C. STRUCTURE OF THE FULL-RING ALGORITHM

The following sequence of steps defines the structure and logic of the Full-Ring algorithm:

1. Schedule a target from the first (i.e., longest) range bin. Delete it from bin.
2. If link is equal to zero, bin empty, go to the next non-empty bin at shorter range. Get a target and delete it from bin.
3. If this target came from the same range bin as the previous target scheduled, then no delay is needed on the interrogation. Go to (5).

4. If this target came from a successive range bin, then calculate a delay, according to Eq. (5.1).
5. If this target's interrogation, plus its delay, fit into the available gap between the previous interrogation and the reply of the first target in this cycle, go to (7).
6. Start a new cycle in the schedule. Use this target to initiate it.
7. Schedule this target by calculating its interrogation and reply time interval on the RF DABS channel.
8. If this is *not* the last target to be scheduled, go to (2).
9. The schedule is completed.

D. PERFORMANCE

The Full-Ring algorithm performance was tested for various sensor coverage ranges and various target distributions in range. Its performance was observed to be a weak function of these variations. The algorithm was also stable with different message lengths, provided all the targets had equal or nearly equal uplink and downlink messages. Figure 5.3 shows the channel time versus the number of targets scheduled. An uplink message length of 50 μsec, a downlink message length of 50 μsec, and a 60-mile range sensor were used in the runs. Each point in the curve represents the mean channel time in 50 runs. A bin resolution of 1 mile was used. There

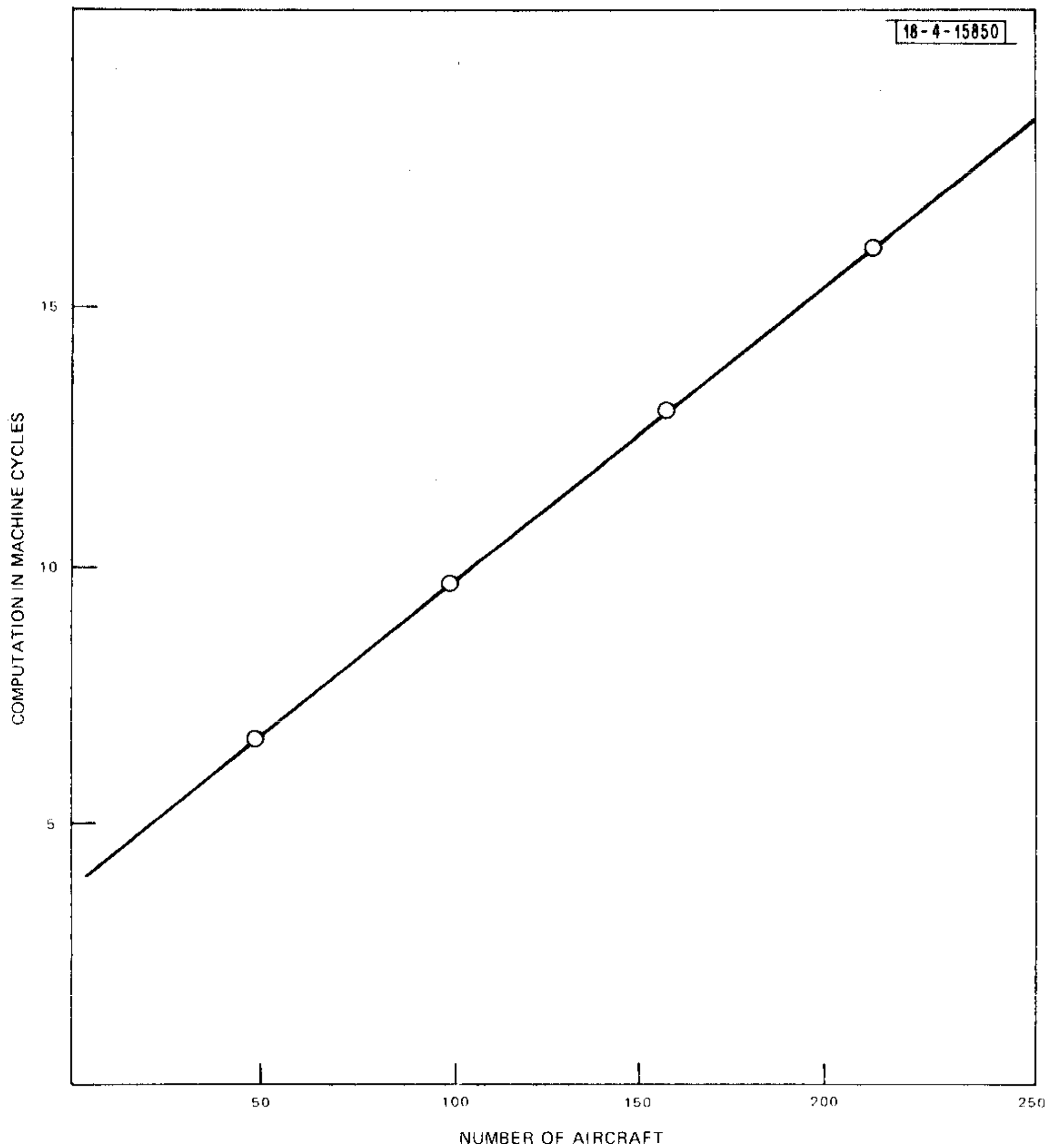


Fig. 5. 3. Full-Ring channel vs number of aircraft.

was little variance observed in the distribution for these runs. It is observed that the packing efficiency of the scheduler reaches 97% for very large numbers of targets.

For the same runs, the computation time required to schedule the targets is given in Figure 5.4. The above two runs are representative of the performance expected from the algorithm.

A constant overhead in computation is used to initialize the list used for the bin sort and in scanning the bins. This overhead increases with the number of bin ranges used. A large bin width could be used for the bin sort for a small number of targets. This would reduce the computation overhead without greatly affecting the packing efficiency. Table 5.1 gives the computation required to schedule 20 targets as a function of bin width. The packing efficiency remained fairly constant; the computation dropped from 4.7 to 1.4 kc, a reduction factor of 3, by increasing the bin width from 0.1 mile to 3 miles.

Table 5.1. Full-Ring Computation Required To Schedule 20 Aircraft vs Bin.

Bin Width (in miles)	Computation (Kilocycles)
0.1	4.7
0.5	1.9
1.0	1.5
3.0	1.4

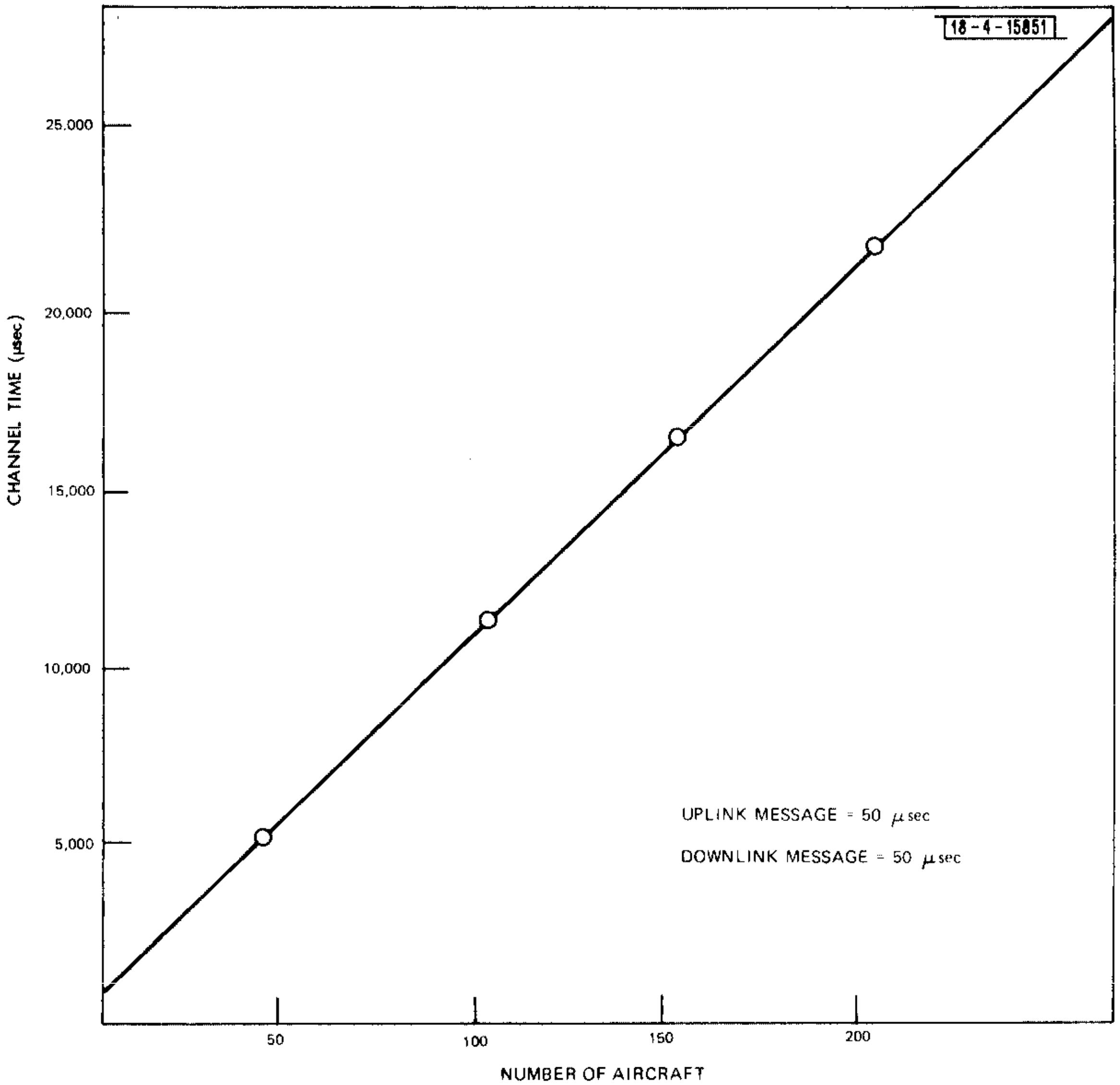


Fig. 5.4. Full-Ring computation effort vs number of aircraft.

The main component of channel time loss incurred between successive targets in a schedule cycle occurs when the targets come from different range bins. This loss is approximated for the whole schedule by the propagation delay to the maximum range target. This assumes that the number of targets in a cycle is reasonably large. This component of the channel time loss is a major part of the channel time overhead. When successive targets come from the same bin, the mean channel time loss per target is a third of the bin width if the targets are assumed uniformly distributed over the bin. This loss is relatively insignificant for bin width of few microseconds and message lengths in the tens of microseconds.

The other channel time loss is the gap left between the last interrogation and the first reply in a schedule cycle. The mean value of this is $\tau/2$, where τ is message length. For the maximum propagation delay that could accommodate MTAK targets per cycle, and for uniform distribution of targets with range, the mean number of targets per cycle M is $\frac{\text{MTAK}}{\log_e \text{MTAK}}$. The mean number of targets per cycle M is sensitive to the distribution of targets with range. The corresponding fraction of channel time loss is $\frac{\tau/2}{2_\tau \times M}$. For a large number of targets distributed uniformly over 60 miles this fraction is 5%. Figure 5.4 shows the channel time with a constant overhead of 750 μsec corresponding to the 60 mile range propagation delay. The slope of the curve is 103 μsec of channel time per target. This corresponds to a mean of 7 targets per cycle.

The algorithm packing efficiency degrades at a low number of targets where the channel time overhead predominates. The performance also degrades with unequal uplink and downlink messages or when a gap is forced between interrogation, as these would increase the time loss on the channel between successive targets. Figure 5.5 shows the channel time for an uplink message of 28 μsec, a downlink message of 46 μsec, and a required gap of 22 μsec between interrogations. In this figure, at 20 targets, the efficiency of packing drops to 45%.

In summary, the algorithm is characterized by a high packing efficiency on the order of 95% or better for scheduling hundreds of targets and for message lengths that are equal. The corresponding computation required to carry out the schedule is 60 cycles/target. The output of the schedule has targets appearing in decreasing range order and in cycles where a group of interrogations is followed by their replies.

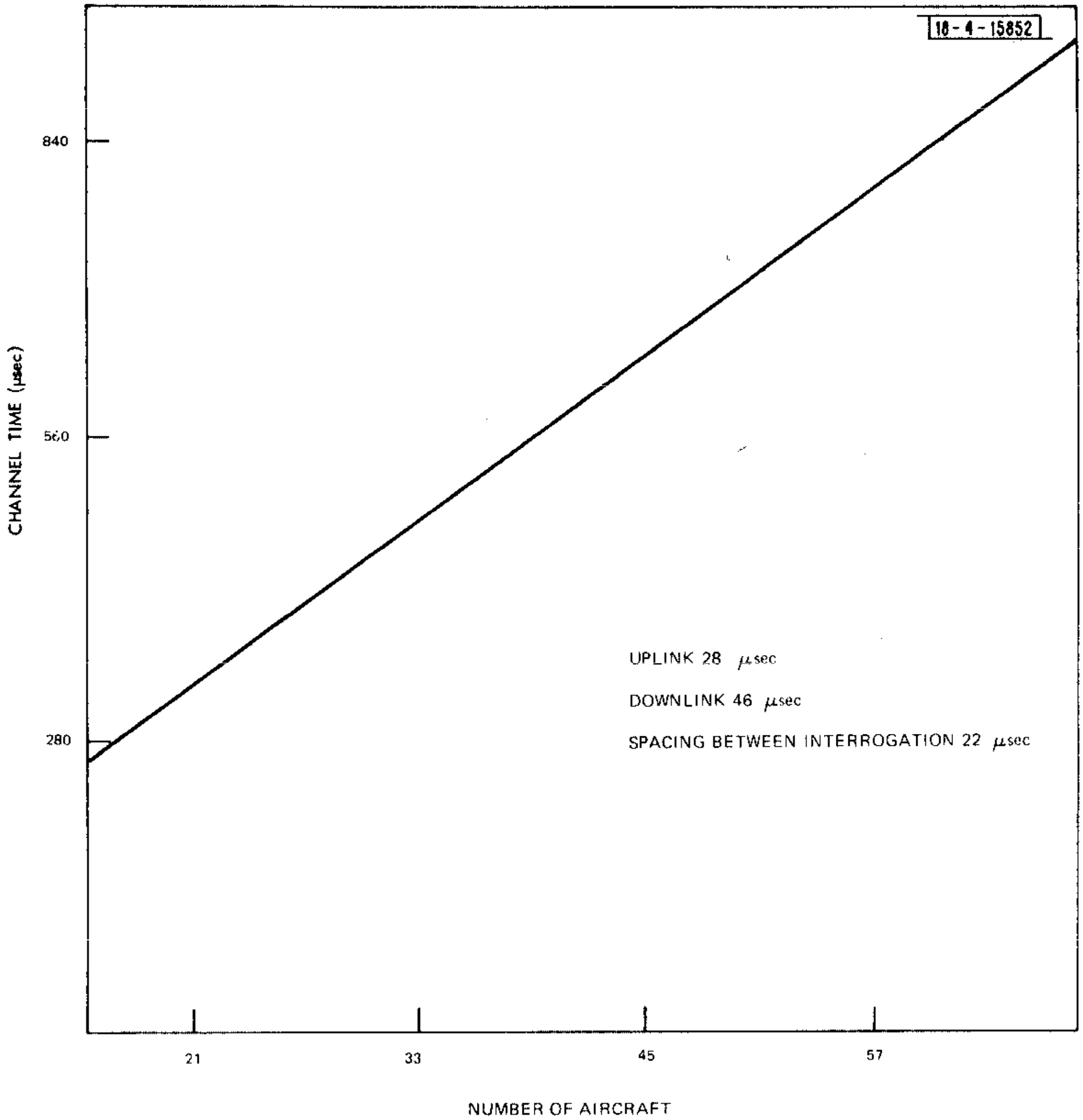


Fig. 5.5. Full-Ring channel time vs number of aircraft.

6.0 CLOSE-FIT ALGORITHM

A. PURPOSE

The packing efficiency of the Full-Ring algorithm degrades when the message become unequal. For example, if the uplink messages are of very short length and the downlink messages are of very long length, the packing efficiency of Full-Ring algorithm would be around 50% for a large number of targets. Also, if the length of the messages uplink or downlink are of different lengths it would reserve the maximum message length for all. This would degrade the packing efficiency of the Full-Ring algorithm to

$$\frac{\overline{\text{uplink message}} + \overline{\text{downlink message}}}{2 \times \text{Maximum Message Length}}$$

possibly a very low value.

The Close-Fit algorithm is designed to schedule unequal length messages while maintaining high packing efficiency and reasonably low computation effort.

It is best suited for an agile beam antenna system using messages which are of unequal length due either to different uplink and downlink bit rates, or to the use of unequal length communication messages. For a large number of targets, the algorithm packing efficiency is high and it has a low computation effort per scheduled target. The algorithm schedules targets in a sequence that is different from the order in which the input is provided. The algorithm needs the information about all the targets before starting to schedule them.

B. BASIC CONCEPT

The Close-Fit algorithm is similar to the Full-Ring algorithm in forming scheduling cycles. However, the sequence in which targets are scheduled is not necessarily range ordered. It depends on the message length of the targets as well as the range of targets; a mechanism necessary to improve the packing efficiency of the algorithm. This alters the mechanism of target information storage and the retrieval of a target.

The data structure again uses bin sort to reduce the sorting effort (which is essentially an information storage mechanism) by making it linear with the number of targets as opposed to increasing as the square of the number of scheduled targets for ordinary sequential sort. Additional arrays for pointers are established to reduce the computation in retrieving the next target to be scheduled. Linked structures are used to reduce the storage requirements for accommodating the data.

An example of a cycle and a typical output of the scheduler is given in Figure 6.1. To reduce channel time loss between messages for adjacent targets in a cycle, the following relation holds between the last assigned target i and the target i + 1 to be assigned next to it in time.

$$Int_{i+1} + 2R_{i+1}/C = Rep_i + 2R_i/C \quad . \qquad (6.1)$$

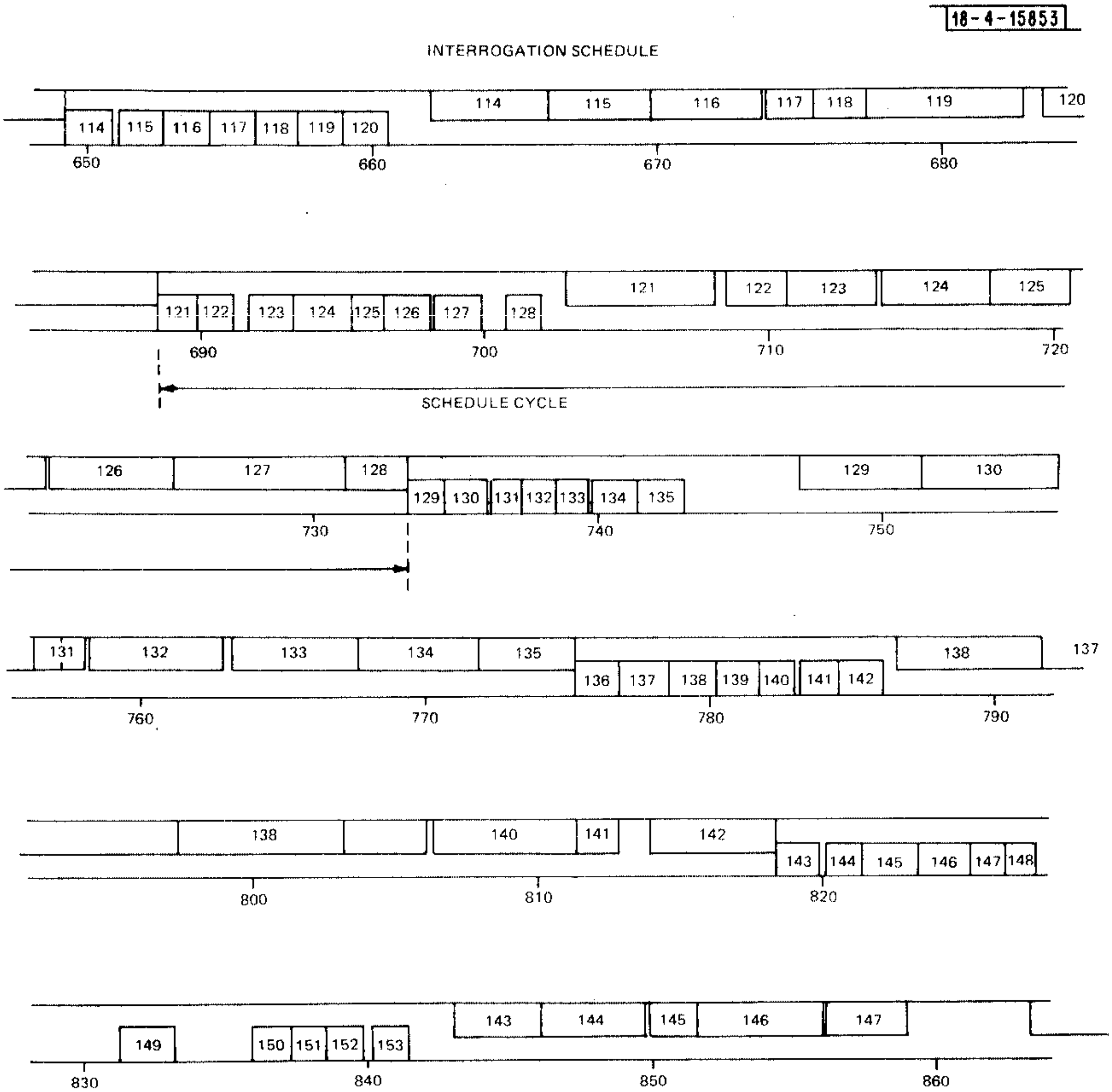


Fig. 6.1. An example of Close-Fit output.

Where Int_{i+1} is the interrogation message length of target i + 1, Rep_i is the time interval to be reserved on the channel for the reply from aircraft i, R_i, R_{i+1} are the ranges of targets i and i + 1 and C is the speed of light. This equation shows the need of using message length besides range in determining the sequence in which to schedule targets as the message length is a part of the equation.

If no available target meets the above condition, the target whose sum of interrogation and propagation delay have the closest value to the desired sum given in Eq. (6.1) would be chosen as this would reduce the channel time loss. Once a target is chosen, the delay required to separate the target interrogation from the end of the last target interrogation is calculated to insure that the target reply falls after the reply of the last scheduled target. This is done according to the equation.

$$\text{Delay} = Rep_i - Int_{i+1} + \frac{2}{C}(R_i - R_{i+1}) \quad . \tag{6.2}$$

If the delay comes out negative, no delay is required. If the interrogation of the target to be scheduled at the required delay overlaps or falls after the reply of the first target in the cycle, a new cycle is started. The target with maximum value of the sum of propagation delay and interrogation among the targets left for scheduling is used to start a new cycle. This would reduce the loss on the channel time in forming the last cycles in the schedule. The above algorithm selects targets based on the sum of their interrogation message length and the propagation delay. (This is essential in determining the sequence in which targets are selected.) <u>Ordering targets</u>

on the sum of their interrogation and propagation delay reduces the effort in retrieving a target with a desired sum. A compromise method of ordering the target list over strict sequential ordering is one which quantizes a given target sum of interrogation message length and propagation delay in time bins. This reduces the computation in storing the information about targets. The sum of propagation delay and interrogation is derived from the value associated with a time bin. Targets in a bin are associated with the time at the beginning of the bin. This introduces a resolution error of the width of a bin in retrieving a target but does not have significant effect on packing efficiency.

For the purpose of discussion, the bins could be pictured strung horizontally with bins associated with the lowest time at the left and the ones with the highest time at the right. For the purpose of accommodating boundary condition in computation without special computations, two guard time bins that are occupied with fictitious targets are created at the extreme left and right of the bins. The extreme left bin is set to represent an arbitrarily low value of time and the extreme right bin an arbitrarily large time value. Actually these bins produce discontinuities in the quantization levels assumed for the bins. These fictitious targets will be encountered in search for targets at the extremes of time bin but will be rejected due to the time values associated with them.

The computation effort in locating an occupied bin is reduced by establishing additional pointers. After the targets are sorted into the time bins a left link and a right link are established between every bin and the adjacent occupied bins. To locate a target to the right of an empty bin the

the thread of right links is followed until an occupied bin is found. The same goes for the left direction. Upon finding these two bins, the target is selected from the bin whose value is closest to the desired bin value. If both bins are equally distant from the desired time bin, an arbitrary choice is made for the right one.

As bins at large time value get empty first the computational effort in locating a target to start a new schedule cycle is reduced by maintaining an index to the occupied bin associated with the largest time value. This index is used for starting the search to the first target in a cycle.

Upon scheduling a target, it is removed from the bin to which it was assigned. If the cycle is started with the last target in a bin, to maintain the index to the occupied bin associated with largest time value the location is found of the time bin indicated as occupied and on the left of the bin just emptied. The index to the maximum occupied bin is set to this bin. Also, the right link of this bin is updated to point to the guard bin on the right. These last two steps are also useful to eliminate from the search in scheduling the last targets; the higher valued time bins which become empty at the beginning of the scheduling. A typical output of the algorithm is given in Figure 6.1.

C. STRUCTURE OF THE ALGORITHM

After establishing the data base described above, the algorithm structure is as described in the following:

1. Fetch the target from an occupied bin associated with highest time value.

Schedule the target.

Delete the target from its bin.

Store its reply as the first reply in the cycle.

Update index to maximum occupied bin.

2. Compute desired time bin for next target using Eq. (6.1).

3. Fetch a target from a bin closest to the desired bin. Calculate delay using Eq. (6.2).

4. If this target's interrogation plus its delay fit into the available gap between the previous target interrogation and the reply of the first target in this cycle, go to step 6.

5. Set the time for the next interrogation at the end of last scheduled reply. End of schedule cycle. Go to (1).

6. Schedule the target at the required delay. Delete target from linked lists.

7. If this is not the last target to be scheduled, go to (2).

8. Schedule is complete.

D. PERFORMANCE

The performance of the algorithm showed little dependence on the distribution of targets in range or the maximum range over which targets are distributed. A significant reduction in the computation effort was obtained by increasing the time bin width from 1.2 μsec to 1/100 of maximum propagation delay (e.g., 60 nmi a value of 7.4 μsec) with little reduction in packing efficiency. Figure 6.2 shows the channel time used to schedule

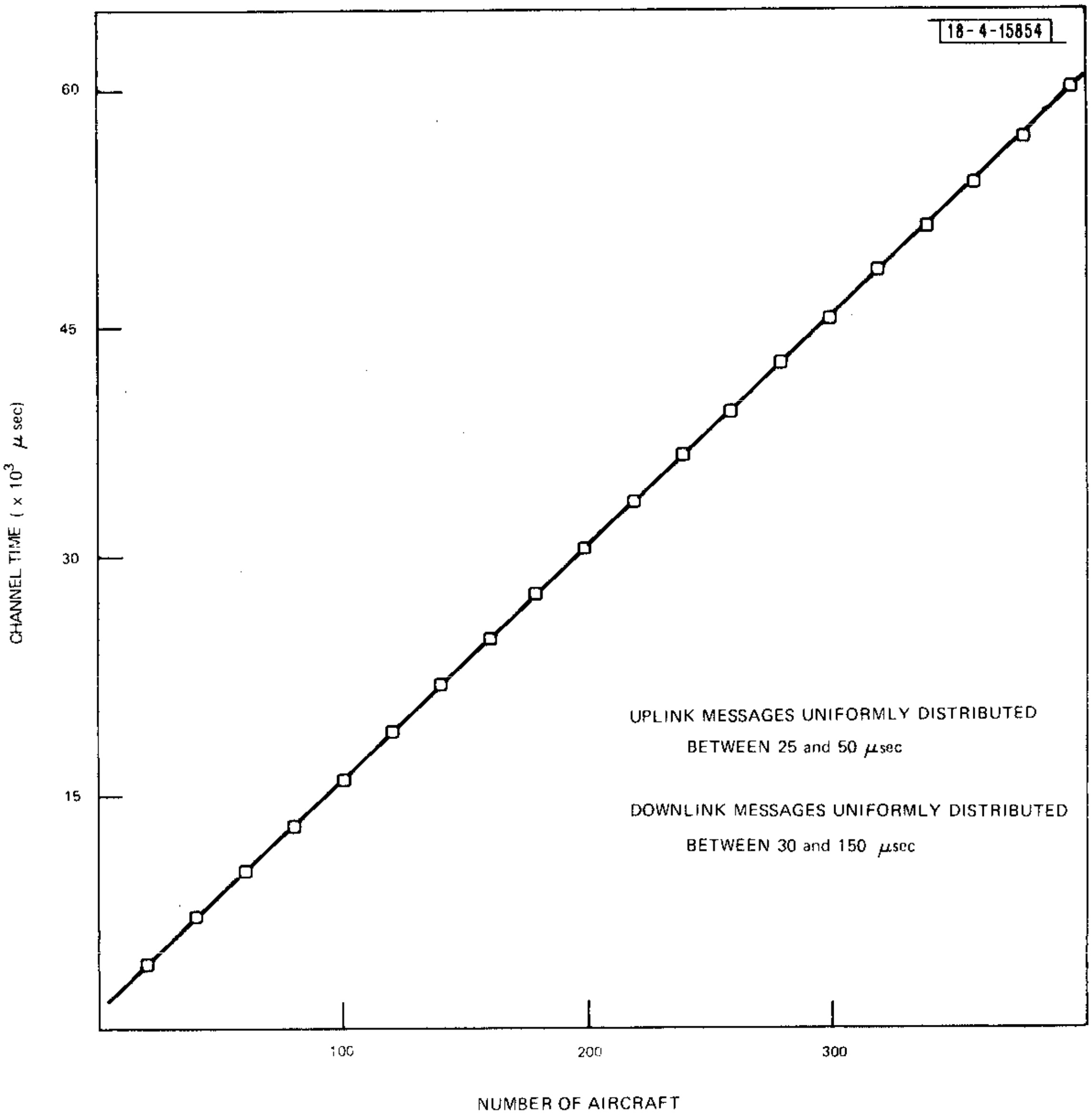


Fig. 6.2. Close-Fit channel time vs number of aircraft.

targets. In these runs, targets' uplink messages were distributed uniformly between 25-50 μsec and downlink messages uniformly between 30-150 μsec. The targets were distributed over 60 miles in range. A bin width of 7.2 μsec was used. Each point in the curve represents the cut-off point for the worst 10% tail in the distribution of channel time for 100 trials. It deviated very little from the mean; this indicates small variance in the performance. The efficiency of packing reaches 87% at a large number of targets. Figure 6.3 gives the computation effort required to schedule the targets mentioned above. For a large number of targets, computation required to schedule a target goes down to 72 computation cycles per target. The computation and channel time given above are representative of the algorithm performance.

The computation time of the algorithm is made up of constant processing overhead and a fixed computation per target. The overhead is used mainly to initialize tables and create the left and right links. An overhead channel time loss is incurred when targets are sought from empty bins. At a small number of targets, this loss is approximated by the maximum propagation delay. At greater numbers of targets, this loss would be a little bit greater as the targets are not scheduled in range order. In the above simulation, the constant overhead obtained, as determined by the best fit line, is 1100 μsec. This is larger than the maximum propagation delay, 720 μsec.

There is a loss in channel time due to a gap between the last interrogation in a cycle and the first reply. This can be modeled as in Figure 6.4.

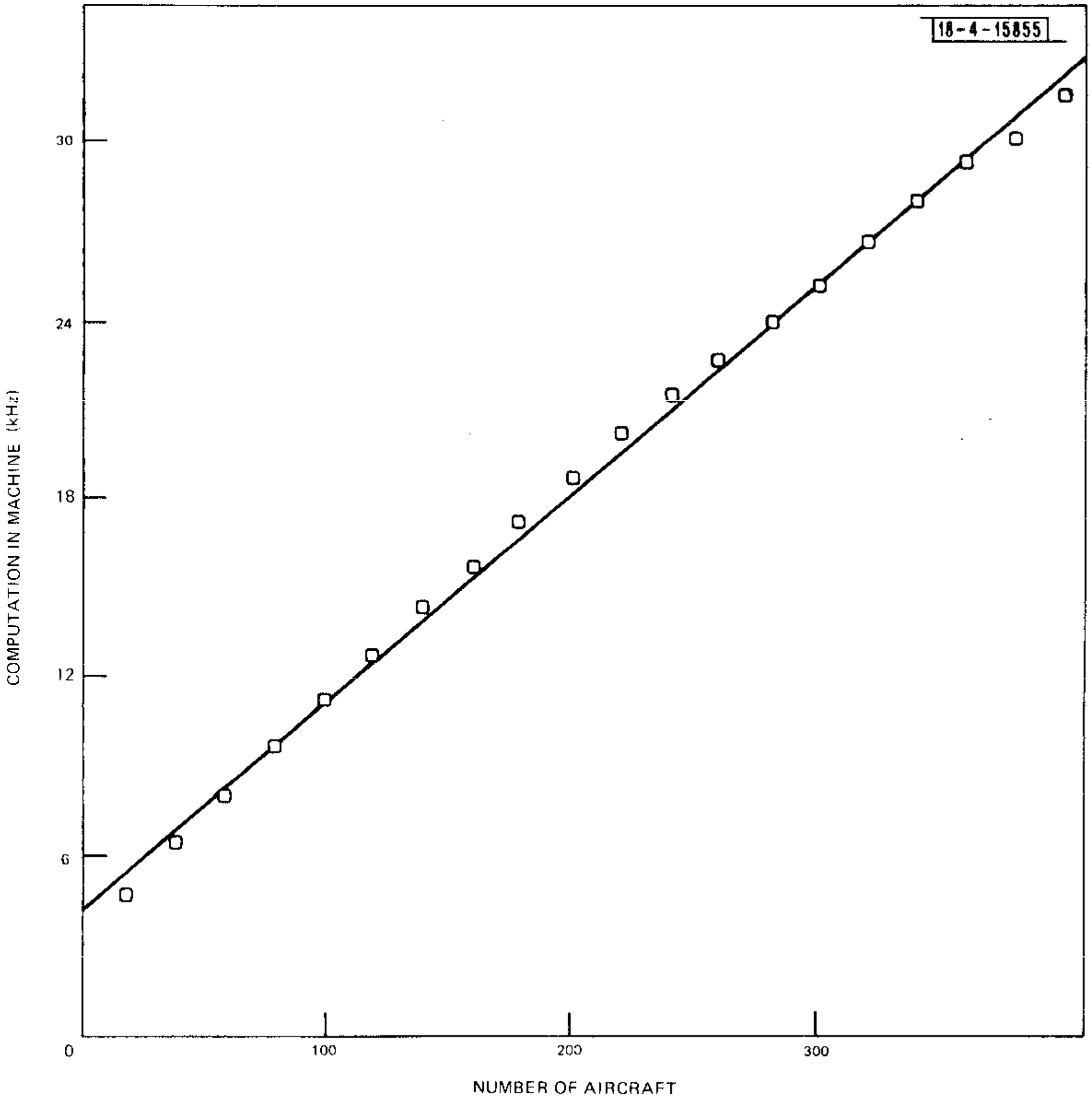


Fig. 6.3. Close-Fit computation effort vs number of aircraft.

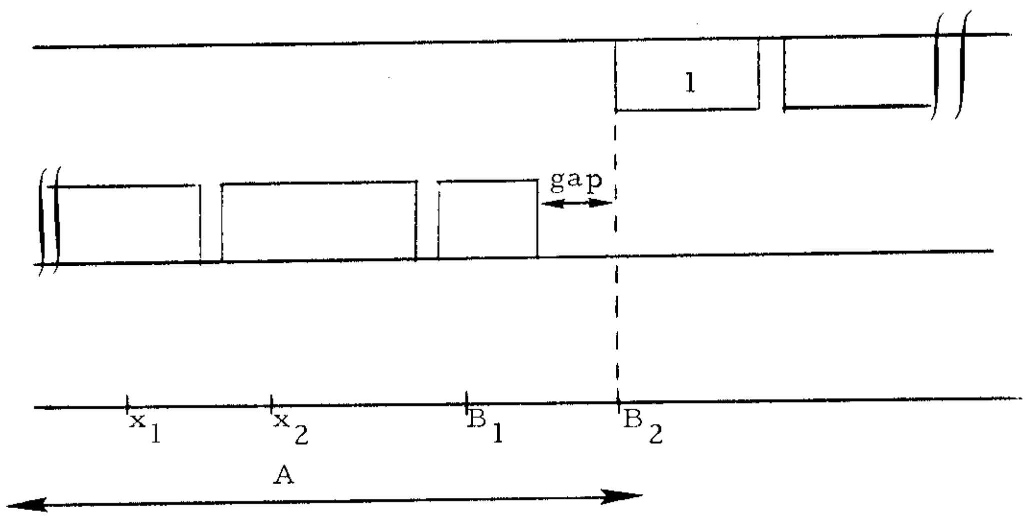


Figure 6.4. Mode of gap between the last interrogation in a cycle and the first reply.

If interrogations were scheduled with no regard to the constraint of overlapping reply 1 as shown in Figure 6.4, then the boundary between interrogations, such as x_1 and x_2 in Figure 6.4, would be uniformly distributed over an interval such as A shown in Figure 6.4. Let the interrogation message length be distributed over values from MIN to MAX with a probability density $P_m(m)$. In Figure 6.4, let the interval A be of length MAX (an interval where potentially the next interrogation cannot fit). Consider the end of an interrogation that is just scheduled to be at x_i somewhere in the interval A. If the next interrogation message length plus the required delay is greater than $B_2 - x_i$, a gap of length $B_2 - x_i$ would be produced.

The mean value of the gap would then be:

$$\overline{\text{Gap}} = \frac{1}{\text{MAX}}\left[\int_0^{\text{MIN}} y\,dy + \int_{\text{MIN}}^{\text{MAX}} y\,dy + \int_y^{\text{MAX}} P_m(m)\,dm\right] .$$

If the message were uniformly distributed over the length between MIN and MAX the value would reduce to:

$$\overline{\text{Gap}} = \frac{1}{\text{MAX}}\left(\frac{\text{MIN}^2}{2} + \frac{1}{\text{MAX-MIN}} * \left(\frac{\text{MAX}^3}{6} + \frac{\text{MIN}^3}{3} - \frac{\text{MAX}*\text{MIN}^2}{2}\right)\right) .$$

Consider the case where the reply messages are distributed uniformly over 30 - 150 μsec for replies and 25 - 50 μsec for interrogations and targets are uniformly distributed over a range of 60 nmi and bin resolution equivalent to 0.6 nmi of range. Then the loss due to bin resulution per target is 2.4 μsec.

The percentage loss in efficiency due to bin resolution would be:

$$\frac{2.4}{\text{Interrogation + Reply}} \times 100 = 1.9\% .$$

The mean gap length is 32 μsec. For targets that are uniformly distributed over range, the mean number of targets per cycle is about 4 targets. The presence of a gap between the last interrogation and the first reply in a cycle gives a percentage packing efficiency loss of 8.9%. The sum of this last loss and the loss due to bin resolution is 11%. This should represent the packing efficiency loss at a large number of targets. From the best fit line in Figure 6.2, the assymptotic loss is loss is 13%.

Table 6.1 gives the packing efficiency of the algorithm for different message length distributions. These results are based on a 60 mile maximum range and a 7.2 μsec bin width. The packing efficiency is relatively stable except that at longer messages it increases by a few percent as the effect of loss due to bin resolution decreases.

In summary, the algorithm performs best with a large number of targets where the packing efficiency could reach 87% for the target message length distribution that could vary by a factor of one to ten. The corresponding computation effort required is 73 machine cycles per target. The output schedule from the algorithm is in the form of schedule cycles where a number of interrogations are followed by their replies. The target sequence is not necessarily in range order.

Table 6.1. Effect Of Message Length Distribution On Packing Efficiency Of 200 Targets.

Interrogation Message Length Distributed Uniformly Over (μsec)	Reply Message Length Distributed Uniformly Over (μsec)	Efficiency (%)
10-20	25- 75	84
10-40	25-125	85
10-60	25-175	86.5
10-80	25-225	87
10-100	25-275	87

7.0 DYNO ALGORITHM

A. PURPOSE

Essential features of the efficient computation and packing operations of the Full-Ring and Close-Fit algorithm is the ability to reorder the sequence in which targets are scheduled from the order they appear in their input list and to have a large number of targets. These features are not available when a target is to be scheduled upon short notice from the request of the sensor for its schedule. For in the dynamic mode, essentially each target is scheduled by itself and the schedule output is produced in the sequence in which targets scheduling requests are received. Furthermore, in a rotating beam sensor computation, effort for the Full-Ring and Close-Fit algorithms becomes relatively high with the large number of separate schedules to be produced as each schedule would have its own computation initialization overhead.

Dyno is an algorithm designed to schedule targets one at a time in whatever sequence they are received from the IMF, rather than scheduling batches of targets as previous schedulers have done. The algorithm requires little computational effort on a per target basis and has practically no intialization overhead. It is therefore useful in a dynamic rescheduling system, where reinterrogation must take place within a short time after an erroneous or missed reply occurs. This usefulness is due to targets being scheduled independently of one another and to the computational efficiency of the algorithm. It is also useful for prescheduling targets for a rotating antenna sensor. This is due to the fact that targets will be grouped into many azimuth sectors for scheduling purposes, the low initialization overhead of the algorithm at the beginning of each scheduling interval is therefore an advantage.

B. BASIC CONCEPT

The arbitrary sequence in which targets are scheduled and the desire for good packing efficiency results in a schedule where interrogations and replies follow no particular structure such as cycle. In this case, it is necessary to search for time intervals to accommodate the interrogation and replies of a target.

To reduce the computation effort in locating available time for messages while maintaining reasonable packing efficiency, time on the channel is quantized and a "quantum" is declared used when a previous message has fallen on it. Also, target messages are represented by integer number of "quanta." A simple link mechanism is used to reduce the search effort for unoccupied channel time.

The algorithm time is divided into equal intervals each represented by a bin. The bins are represented by words in memory in a time file array called TF (N). Discrete time is given by the index of a bin (i.e., of the word in TF (N)). To insure that interrogation messages and reply messages would fall within the bins assigned to them with minimum use of bins, interrogations are represented by a fixed number of bins that cover the message. For example, if the bin width is 30 μsec and the interrogation message length is 25 μsec the interrogation message length is equivalent to one bin or word in TF(N). The propagation delay of a target associated with its range is represented by the maximum number of bins that fits into the propagation delay time after it is reduced by the gap left in the last bin assigned to the interrogation message to the target. Using the above bin width, and if we assume the propagation delay minus the gap left in the last bin assigned to the interrogation is 305 μsec, the propagation delay is put at 10 bins or 10 words in TF (N). The number of bins that represent the reply is the minimum number of bins that cover the reply message plus a bin to accommodate additional time that might be required for the propagation delay beside the number of bins assigned to it. For the above bin size, if the reply is 45 μsec, the reply would be represented by 3 bins or 3 words in TF (N). An interrogation is assumed to start at the beginning of the first bin assigned to it.

To schedule a target, a search is carried out for empty bins to accommodate the interrogation and reply messages. The search starts by attempting to place the target interrogation at a designated time following the last scheduled target interrogation. The separation could be a function of minimum required separation between interrogations as well as allowing enough time for the interrogation command to reach the transmitter before the interrogation is to be transmitted. For example, if a separation of 50 μsec is required between interrogations for a bin width of 30 μsec a bin is skipped after the last interrogation. As the interrogations are placed at the beginning of their bins, this insures the 50 μsec minimum separation between interrogations. If some of the bins where the interrogation and reply fall are occupied, the interrogation time would be increased to a position where the target's interrogation and reply messages fall on empty bins. The search effort required to find empty bins is reduced by use of information stored in occupied bins that points to a potentially empty bin later in time.

The pointer value stored in a bin is set at the time the bin is occupied. It is the distance of the bin from the end of the reply plus the value stored in the bin following the reply. Thus, suppose for purpose of illustration, the reply occupies 3 bins. Assume the bins get filled as in the sequence of Figures 7.1 (a, b, c, d). The figures give the values stored in the words corresponding to the bins.

The length of the time interval represented by one bin is kept large enough to maintain low computational effort and reasonable scheduling efficiency. Use of short bin width tends to increase the packing efficiency and at the same time the computational effort.

The algorithm could accommodate variable length messages. In that case, to check that a message is falling on empty bins, it is necessary and sufficient to check that a subset of the bins in the message are empty. This subset is the first bin; the last bin and bins within the message that are at a multiple of m bins from the start of the message. The value m is the minimum length of a reply in bins. This test is necessary and sufficient to insure that no message of length m or longer would overlap the message investigated. As m is the shortest message length, then no message would overlap the message examined. Attention is confined to replies as they are the only messages in the interval of interest.

In the case where the interrogations are of one fixed length, replies are of another fixed length and the replies are longer than interrogations. To check that the interrogation message is falling on empty bins it is necessary and sufficient to check that the first bin and last bin in the interrogation are falling on empty bins. Similarly, for the reply, a check is required on the first and last bins in it.

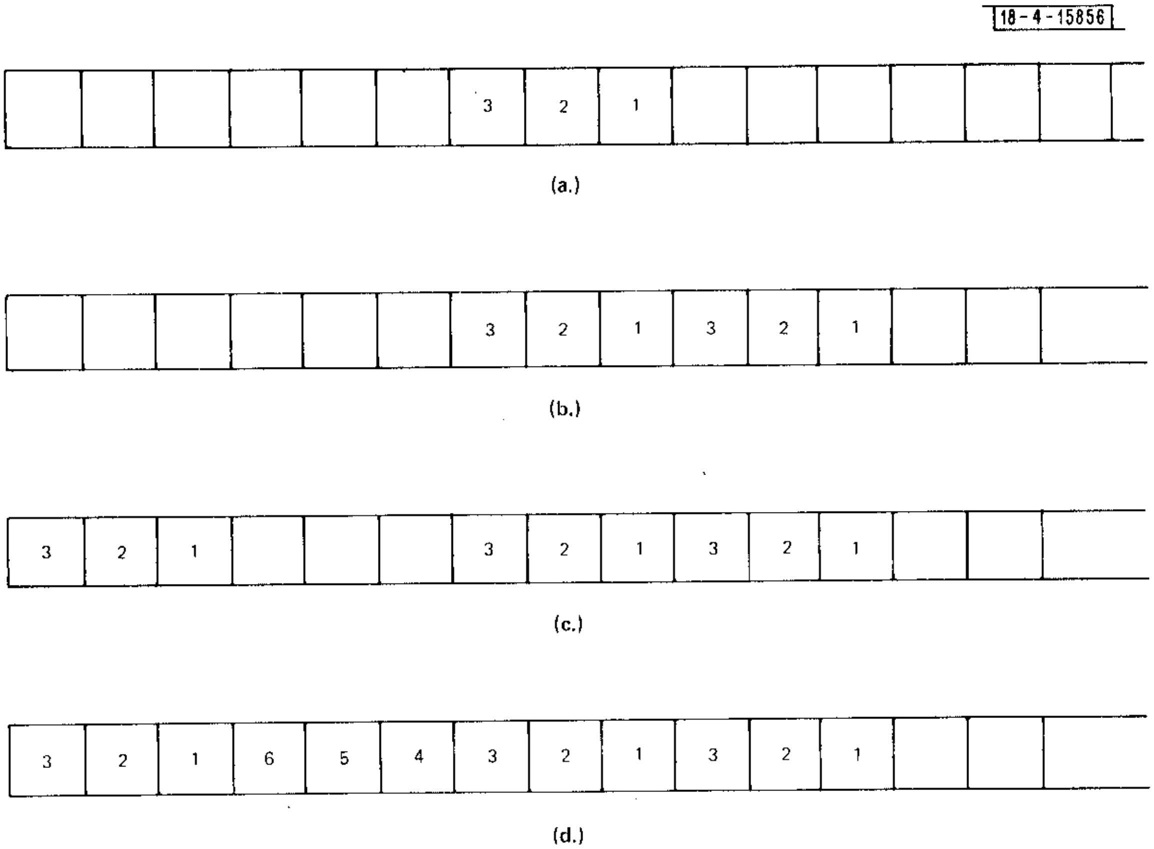


Fig. 7.1. A sequence of filling bins by Dyno.

To reduce the computation effort, the bins corresponding to the reply (which is longer, hence more difficult to accommodate) are checked before the bins corresponding to the interrogation. In this way, a check on the interrogation bins is saved each time the check on the reply bins fails.

C. STRUCTURE OF THE ALGORITHM

The following steps given for the algorithm are specialized to the case where the interrogations are of one constant length that is shorter than the constant length for replies.

1. Set interrogation Bin Index to first bin.
2. Check that interrogation bin index is greater than the time the schedule is computed by required delay. If not, advance bin index to meet condition.*
3. Fetch range of next target. Calculate bin index at start of its reply.
4. Check first bin in reply if occupied go to 13.
5. Check last bin in reply if occupied go to 14.**
6. Check first bin for interrogation if occupied go to 15.
7. Check last bin for interrogation if occupied go to 16.***
8. Store interrogation and reply positions in output schedule.

*Not needed if algorithm is used for scheduling ahead of scan.
**Not needed if reply message length is one bin.
***Not needed if interrogation message length is one bin.

9. Fill bins corresponding to reply.

10. Advance interrogation bin index by required separation between interrogations.

11. Is this last target? If yes, exit.

12. Go to 2.

13. Increment the interrogation and reply bin indexes by the value found in the bin attempted for beginning of reply. Go to 4.

14. Increment the interrogation and reply bin indexes by the value found in the bin at the end of the reply plus number of bins in reply minus one. Go to 4.

15. Increment the interrogation and reply bin indexes by the value found in the bin at the beginning of the interrogation. Go to 4.

16. Increment the interrogation and reply bin indexes by the value found in the last bin of the interrogation minus one. Go to 4.

D. PERFORMANCE

The performance of the algorithm showed little dependence on the maximum target range or the distribution of targets over range. The packing efficiency of the algorithm is not sensitive to message length and is around 57 to 60% when truncation error is neglected. Truncation error is defined as the difference between the sum of the lengths of the uplink and downlink message and the sum of the lengths of the bins assigned to these messages. Truncation error reduces the packing efficiency of the algorithm by the ratio of the sum of interrogation and reply message length to the time occupied by the bins representing these items. This holds only if there is no spacing restriction between interrogations. Typical curves giving the channel occupancy time as a function of number of calls in the schedule output is given in Figure 7-2. For the simulations an uplink message of 28 μsec and a downlink message of 46 μsec were used. The bin size was set at 28 μsec resulting in one bin for the interrogation and three bins for the reply. The results represent the cutoff point for the worst 10% point in the distribution for 50 runs. Two curves are given, one is when the algorithm is used to schedule targets based only on message occupancy on the channel. In the second, a minimum delay time is maintained between receipt of target reply and the reinterrogation to it. This delay time represents the constraint imposed on the scheduler in operating in real-time to insure that the transmission command arrives at the transmitter before the time it is executed.

Figure 7.3 gives the effect of the variation of the minimum gap or delay between the reply and the reinterrogation on the scheduling of 20 calls. The number of different aircraft addressed was set in one curve at 5 and in the other at 8. The size of the minimum delay used at which the algorithm performance begins to deteriorate decreases with the reduction of the number of aircraft actively interrogated. From Figure 7.3 it is seen that the performance is significantly affected when the minimum delay is longer than 280 μsec when 5 targets are addressed and 840 μsec when 8 targets are addressed. Figure 7.4 gives the computation effort to carry out the schedule for the above simulations as a function of number of calls. The computation had little variance in the effort and the points represent the mean of 50 runs. The computation effort increased with decrease of the delay time

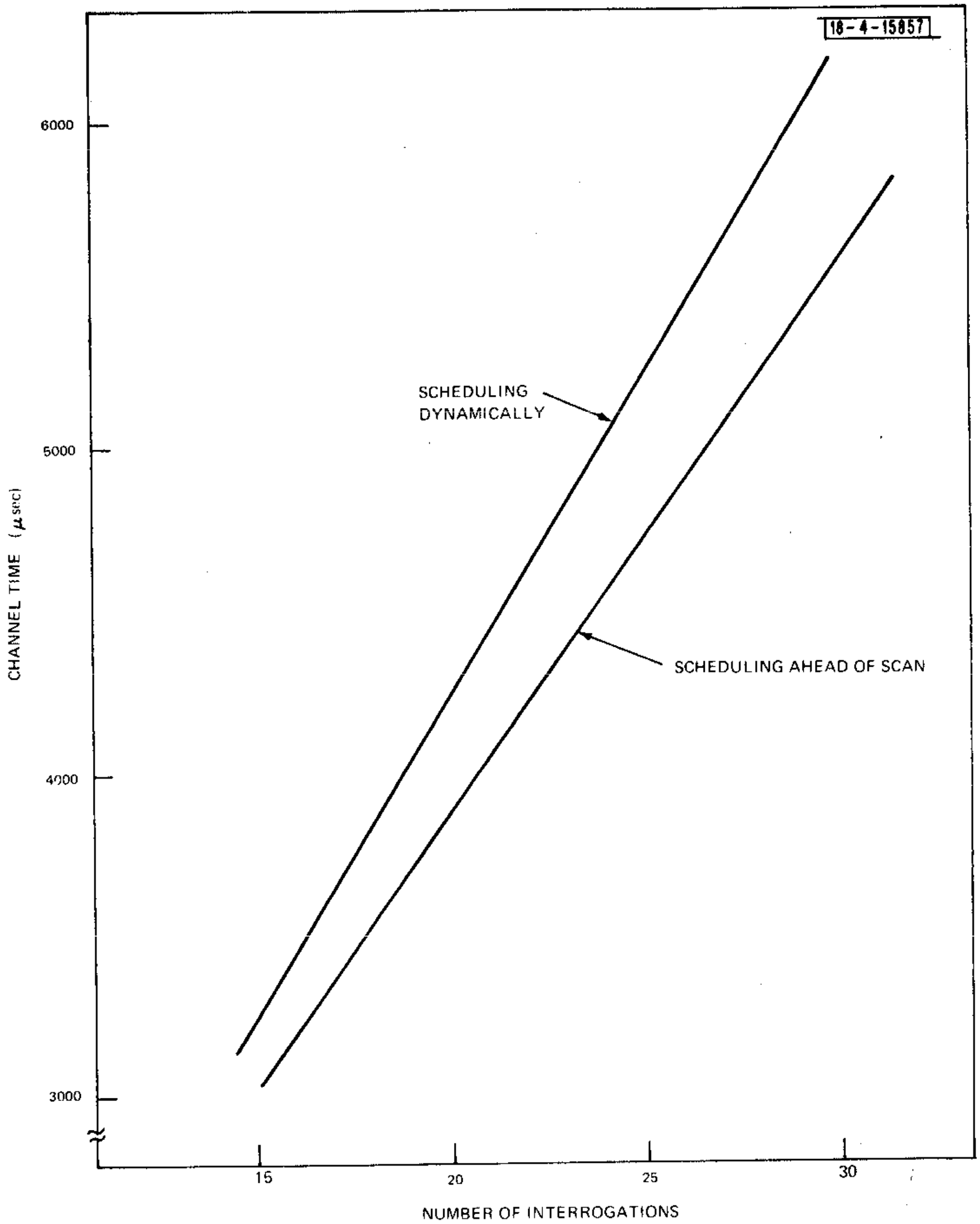


Fig. 7.2. Dyno channel time vs number of calls in the scheduled output.

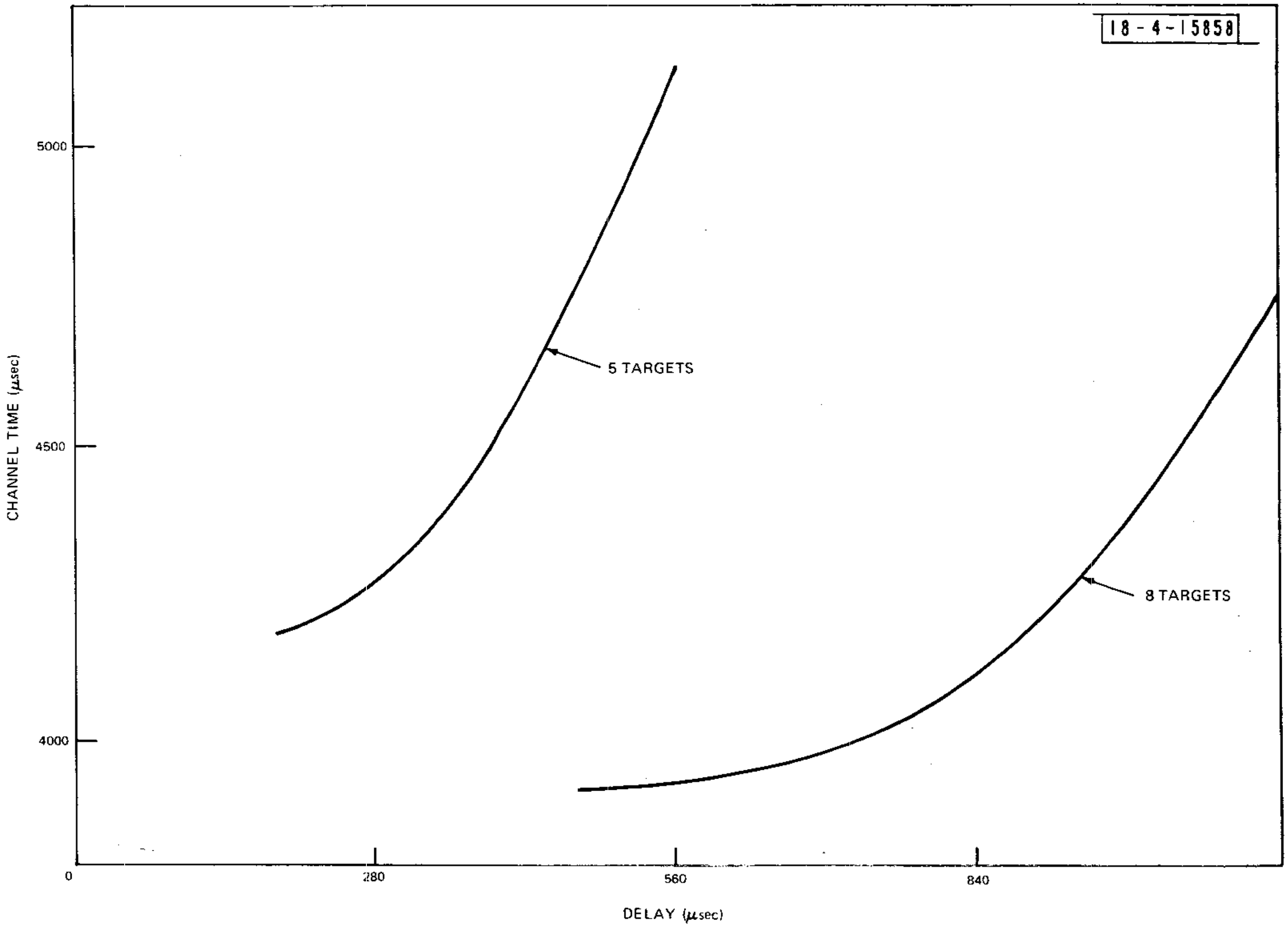


Fig. 7.3. Dyno effect of delay on performance.

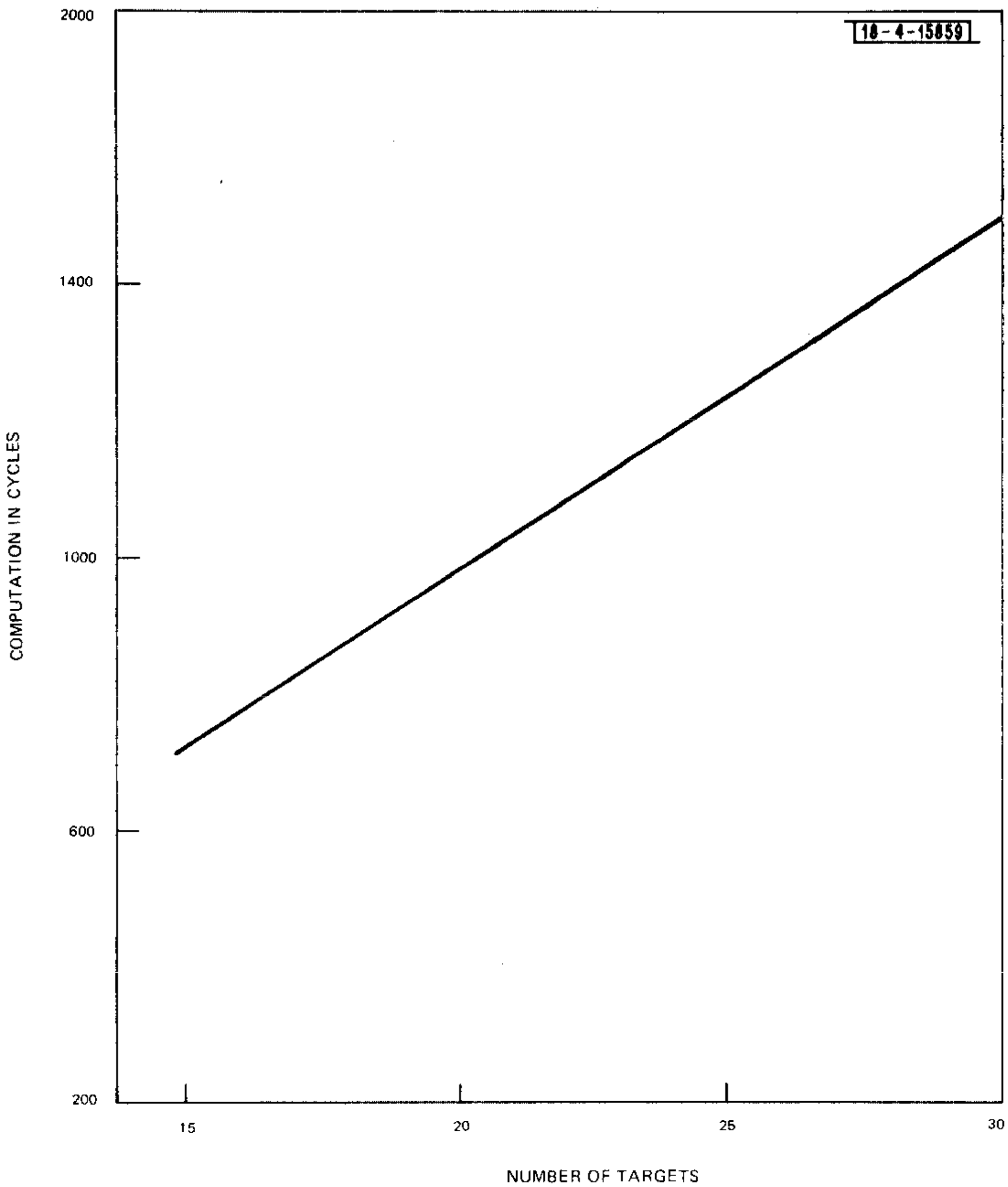


Fig. 7.4. Dyno computation effort vs number of calls.

assigned to processing, as this produces more tightly packed schedules. In general, the computation effort of the algorithm depends only on the number of bins representing the messages. From the results, there is little constant computation overhead associated with the algorithm.

Table 7.1 gives the packing efficiency for different message lengths. Packing efficiency is obtained as the ratio of the sum of message time scheduled to the total channel occupancy by the schedule. This efficiency is dependent on the message length and bin size. If the efficiency is normalized by assuming that the messages fit exactly into the bins used to represent them, the efficiency becomes insensitive to message length and has a value around 60% for the number of targets considered.

Table 7.1. Packing Effiency For Two Different Message Lengths.

	Uplink 28 μsec Downlink 46 μsec Bin Size 28 μsec		Uplink 28 μsec Downlink 96 μsec Bin Size 32 μsec	
No. of Targets	Packing Efficiency	Normalized Packing Efficiency (i.e., no truncation loss)	Packing Efficiency	Normalized Packing Efficiency (i.e., no truncation loss)
15	0.36	0.55	0.44	0.56
20	0.38	0.575	0.45	0.58
25	0.38	0.58	0.46	0.6
30	0.40	0.61	0.47	0.615

Figure 7.5 gives the truncation error as a function of bin size for an message length of 28 μsec and a downlink message length of 46 μsec. The truncation error is given by the additional time required by the bins for interrogation and reply messages as compared to the actual message lengths. Using the bin sizes where the truncation error was minimum, the scheduler was exercised to schedule sets of 20 targets. Figure 7.6 gives the mean channel occupancy vs the computation time. Each point in the results gives the mean of 50 runs at different setting of the bin size. The discontinuity in the values is explained by the fact that the truncation error is not monotonically decreasing with decrease of bin size.

In summary, DYNO can assign targets with packing efficiency better than 40% at a computation effort less than 60 cycles per target on a sequential machine. The close coupling of the algorithm to other activities in real time degrades its performance when the processing of a target schedule, including the scheduling algorithm, exceeds a certain delay time. This delay time depends on the number of targets actively interrogated and decreases as the number of distinct targets interrogated decreases. The algorithm has practically no computational overhead.

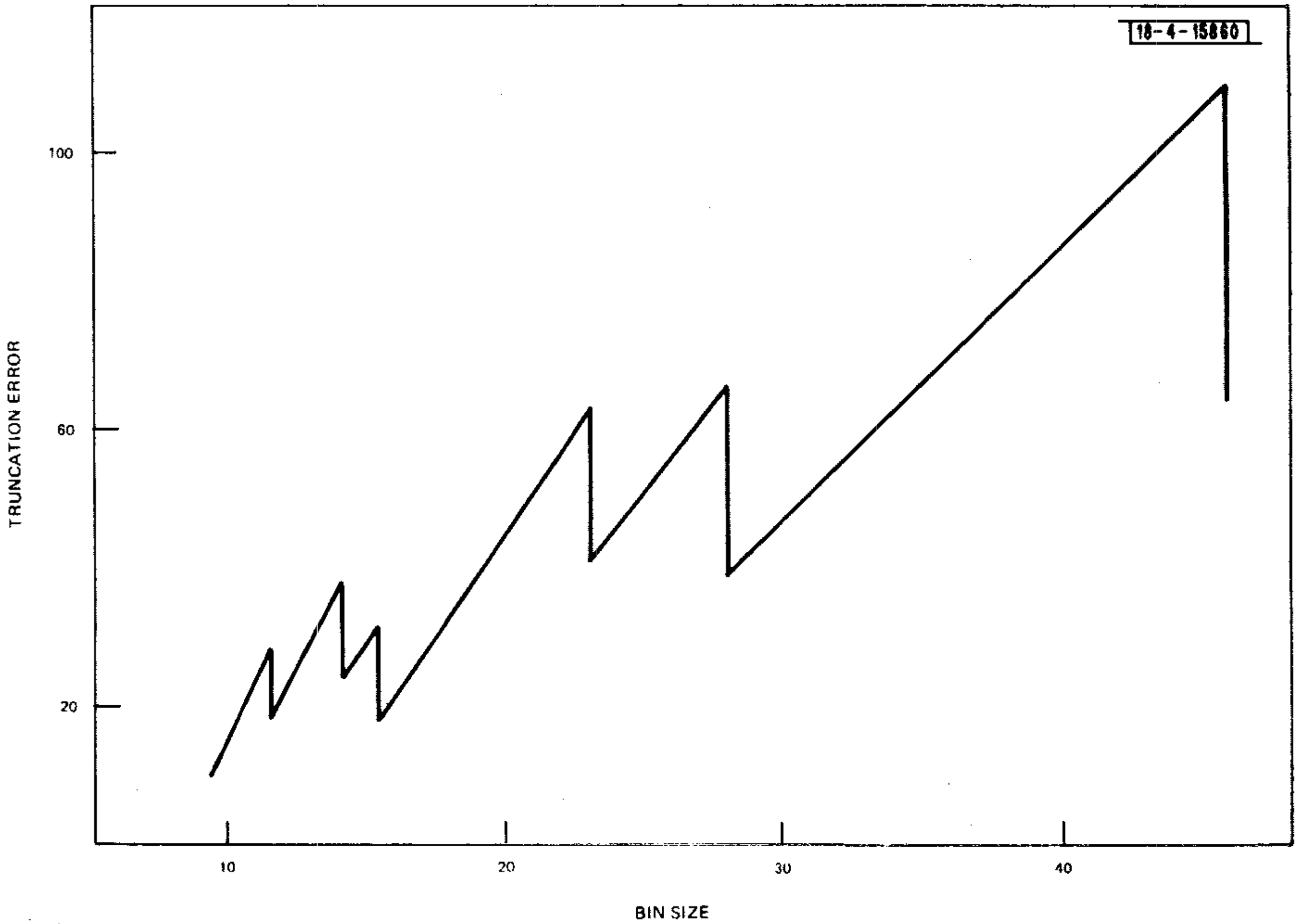


Fig. 7.5. Truncation error vs bin size.

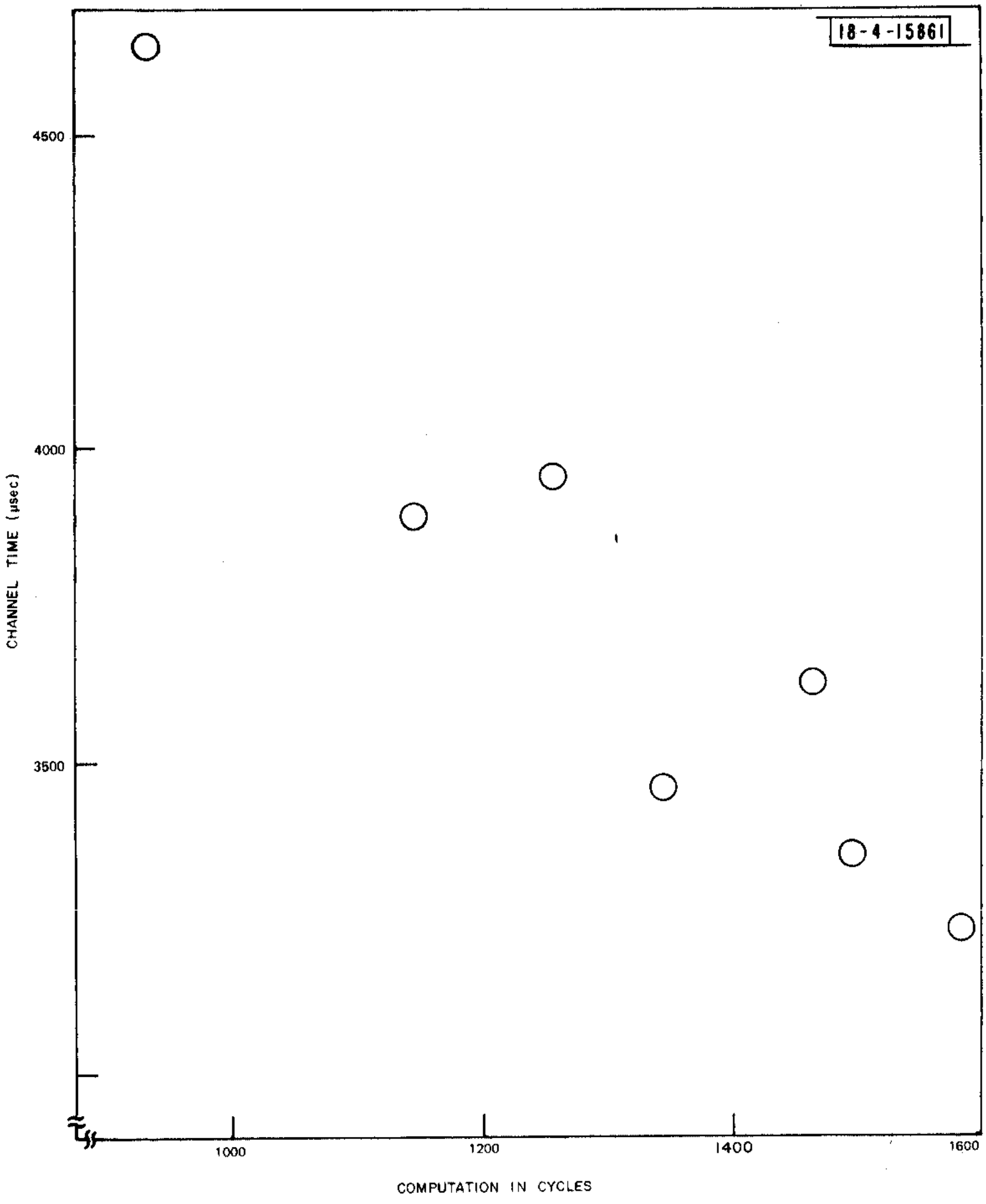


Fig. 7.6. Mean channel occupancy vs computation time.

8.0 LOOP-LOOP ALGORITHM

A. PURPOSE

The output of the Full-Ring and Close-Fit algorithms is not useful for direct output of the Interrogation Management function of an agile beam sensor.

In a DABS sensor using an agile beam, the ATCRBS and DABS function alternate in the use of the sensor. The sensor is available to the DABS function during time intervals called DABS periods. The DABS periods are of a constant duration and alternate with the ATCRBS sweep. Loop-Loop is an algorithm designed to segment the output of the Full-Ring and Close-Fit algorithms to fit into the DABS periods. It could also have the aircraft called a fixed multiple of times although the aircraft was called only a single time in the output of the Full-Ring and Close-Fit algorithms. The algorithm is exercised ahead of the scan. It produces a reordering in the sequence in which aircraft are interrogated from that produced in the output of the Close-Fit and Fixed-Ring algorithms. The algorithm requires full access to this output before it is executed.

B. BASIC CONCEPT

The basic item manipulated by this algorithm is a cycle which is a set of interrogation followed by the set of their replies. This is an inherent output of the Full-Ring and Close-Fit algorithm. For the purpose of the algorithm the cycle is considered a packet of time extending from the beginning of the first interrogation to the end of the last reply. The function of the algorithm is to pack the cycles next to each other into a DABS period reducing the time lost at the end of the interval. The cycles are first sorted in increasing order on length. The algorithm assigns cycles to a DABS interval from the bottom of the list where cycles are long. This is continued until the remaining time at the end of the DABS interval cannot accommodate cycles from the bottom of the list. It then assigns cycles to the DABS period from the top of the list giving up when the next cycle to be assigned is longer than the remaining time in the DABS period. The output of the algorithm is the set of cycles assigned to each DABS period.

In essence, what the algorithm is doing is to initially use long cycles when there is room in the DABS period and at the end when those time packets do not fit into the remaining time and fill the gap with short cycles. If the system is required to call a target, m times the algorithm could be used to do that. The DABS period is divided into m equal intervals and the loop-loop algorithm uses one of these m intervals as the DABS period. The final schedule for a DABS period is the duplication m times of the algorithm output for a DABS period.

C. STRUCTURE OF THE ALGORITHM

The cycles are initially sorted on their length in decreasing order. They are stored in a sequential list that is accessible from the top and bottom. The DABS period length is given to the algorithm as a fixed interval of time. When m calls are to be sent to an aircraft an effective DABS period is obtained as the given DABS period divided by m.

The Algorithm main steps are:

1. Start assignment for new DABS period and initialize the remaining time in it to its length.
2. Can the longest cycle in the input list of cycles fit in the remaining time of the present DABS period? If NO, go to (6).
3. Store the number which is at the top of input list in the list of cycles belonging to the present DABS period. Delete this cycle from input list.
4. Is this the last cycle? If yes, exit.
5. Update the remaining time in this DABS period go to (2).
6. Can shortest cycle in the input list fit in the remaining time of this DABS period? If no, go to (1).
7. Store the cycle number in the list of cycles belonging to this DABS period and delete it from input list.
8. If this is the last cycle, exit.
9. Update the remaining time and go to (6).

D. PERFORMANCE

Cycles from the output of the Full-Ring algorithm were used as input to the algorithm. The packing efficiency of the algorithm is defined as the sum of the cycles durations to the sum of the DABS periods used. For this algorithm, the packing efficiency came out close to 95% for large number of targets. Figure 8.1 gives the output of the algorithm for a DABS period of 2000 μsec, the shaded area is the lost time at the end of a DABS period.

The computation effort of the algorithm was approximately 7 m machine cycles per target. As the DABS period was varied, the packing efficiency of the algorithm varied. Figure 8.2 gives the packing efficiency vs period length. Use of a short DABS period drops the packing efficiency. The drop in the efficiency is explained by the multiplication of the time lost at the end of each DABS period by the great number of the DABS periods used. Use of a long DABS period again drops the efficiency. The drop in efficiency is caused by time lost in the last DABS period which becomes significant. The use of a long DABS period makes efficiency sensitive to the number of targets scheduled, as there is a great variation in the percentage fill of the last DABS period.

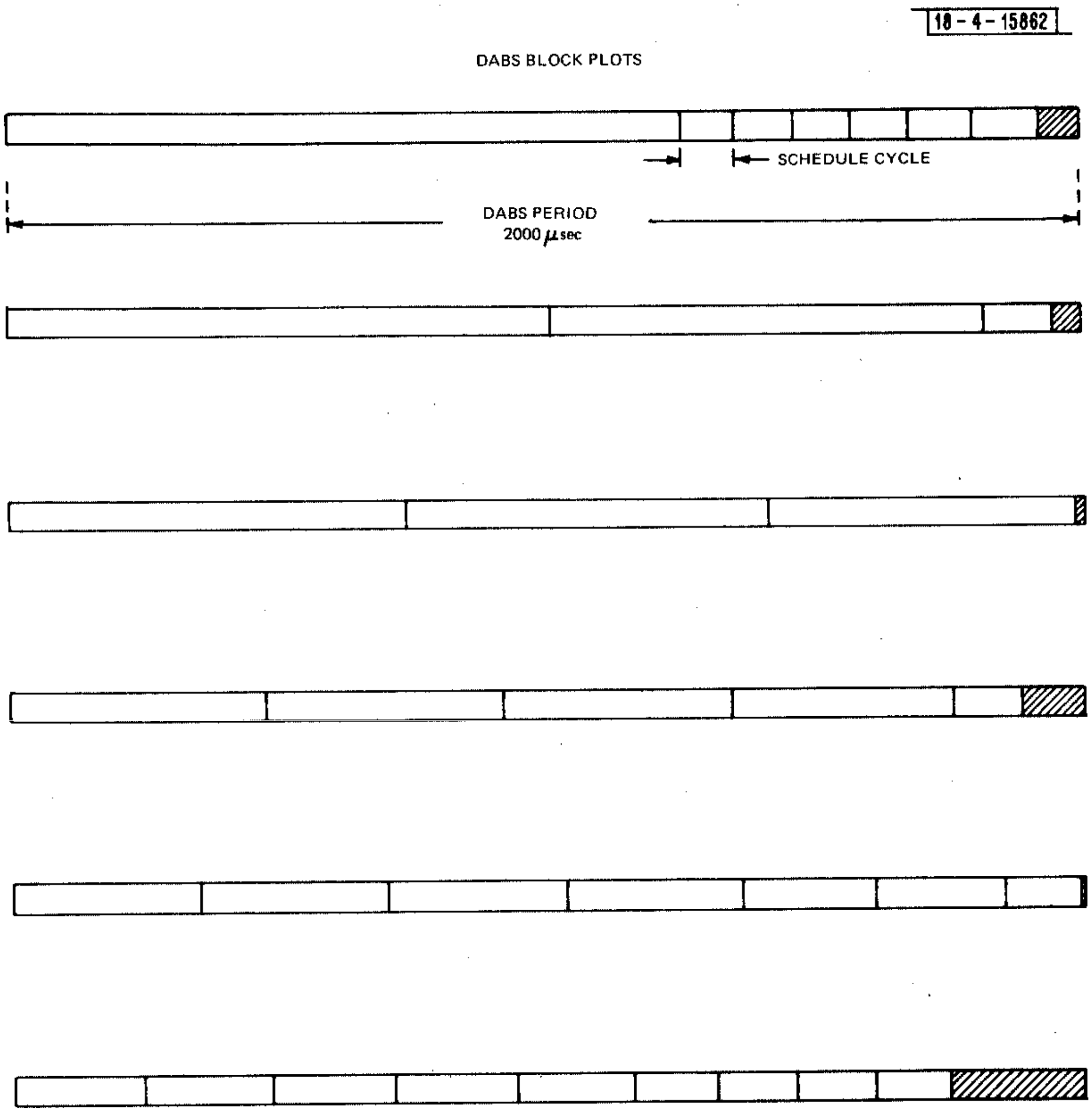


Fig. 8.1. Typical output of the Loop-Loop algorithm.

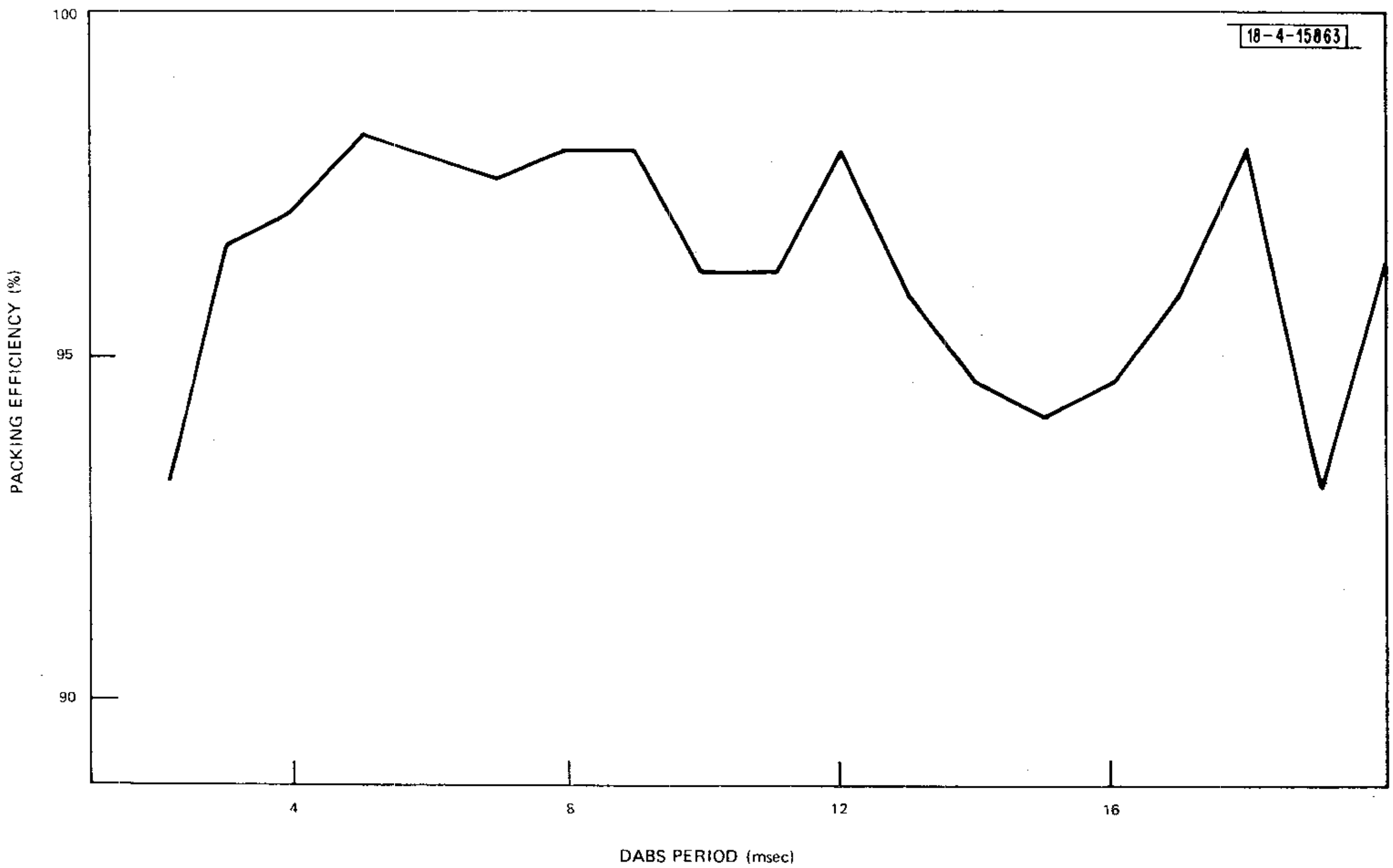


Fig. 8.2. Loop-Loop packing efficiency vs period length.

APPENDIX A

ILLUSTRATIVE EXAMPLES

To illustrate the use of scheduling algorithms in a DABS system, two specific examples are given of interrogation management functions.

1.0 ROTATING BEAM SENSOR

The parameters of the rotating beam sensor example are:

Number of aircraft	1000
3 dB beamwidth	4 degrees
Communication format	uplink 30.5 μsec downlink 63 μsec
Surveillance format	uplink 17.7 μsec downlink 33 μsec
One scan azimuthal prediction uncertainty (3σ, target range >10 nmi)	0.71 degrees
Communication load	5% of surveillance load for both uplink and downlink
Surveillance reliability	99%*
Communication reliability	99.9%*
Discrete calls for DABS direction finding	1
ATCRBS runlength	4
Scan time	4 sec

*This is defined as the probability of success in one scan assuming independent failure probability for each interrogation and good coverage is used.

1.1 BEAMWIDTH AND BUNCHING CONSIDERATIONS

It is desirable to schedule DABS surveillance interrogations such that the replies are received within the 3 dB points of the antenna. This is true since the reply processor is designed to meet its specified angular accuracy within this 4 degree azimuthal wedge. The difficulty in achieving this goal is the uncertainty in the prediction of target azimuth to the next scan based upon track measurements on previous scans. A major contributor to prediction uncertainty is due to aircraft maneuvering, whose effects are particularly troublesome at close ranges. Fortunately, the azimuth prediction uncertainty is at its minimum at longer ranges since it is here that the azimuthal accuracy requirements are most stringent to ensure an acceptable cross range error. For this reason, targets at ranges of greater than 10 nmi are interrogated over an azimuth wedge equal to the 3 dB beamwidth, less an allowance for the azimuth prediction uncertainty subtracted from the leading and trailing edge of the beam.

Using the same technique for targets at ranges less than 10 nmi would greatly reduce the usable azimuth wedge since the magnitude of the azimuth uncertainty approaches (and at very short ranges exceeds) the 3 dB beamwidth. This situation is managed by handling targets between 3 and 10 nmi in the same manner as long range targets except that the 9 dB beamwidth is used. While this results in target replies that fall outside the 3 dB beam, the probability is high that they will fall within the 9 dB points and hence, the reply processor will be able to yield a (degraded) azimuth estimate. The 6 dB reduction in fade margin and the degradation in azimuth accuracy are not significant due to the short target ranges.

At ranges less than 3 nmi, the azimuth uncertainty is greater than even the 9 dB beamwidth. For targets at these very short ranges, several azimuth separated interrogations are scheduled to ensure that at least one of the interrogations will produce a usable reply.

Bunching of targets is defined to be the ratio of the maximum number of targets per azimuth wedge to the mean number of targets per azimuth wedge. This ratio increases with a decrease in the width of the azimuth wedge. Figure A-1 gives an example of the bunching as a function wedge size. This was obtained from traffic models projected for the N.Y. area within 150 nmi from J.F.K. In assessing the capacity of the interrogation management function, bunching values obtained from this curve will be used as nominal levels.

1.2 SCHEDULING PROTOCOL

The azimuth wedge within which a DABS target can be reliable interrogated is the 3 dB beamwidth (4^{o}) less an allowance for the one scan azimuthal prediction uncertainty. We have taken this as the 2σ value or 0.5^{o} which insures that long range targets will be interrogated within the 3 dB beamwidth with a probability of greater than 97% assuming a normal distribution of errors. This uncertainty is subtracted from the leading and trailing edges of the beam and the resulting usable azimuth wedge of 3^{o} is divided into four equal interrogation cycles. Each cycle is composed of a prescheduled period (PR), an ATCRBS/All-Call period (A) and a dynamically scheduled period (DY) as shown in Figure A-2. Each cycle therefore corresponds to a $3/4^{o}$ azimuth wedge, the total dwell time for the four wedges is 33 1/3 ms.

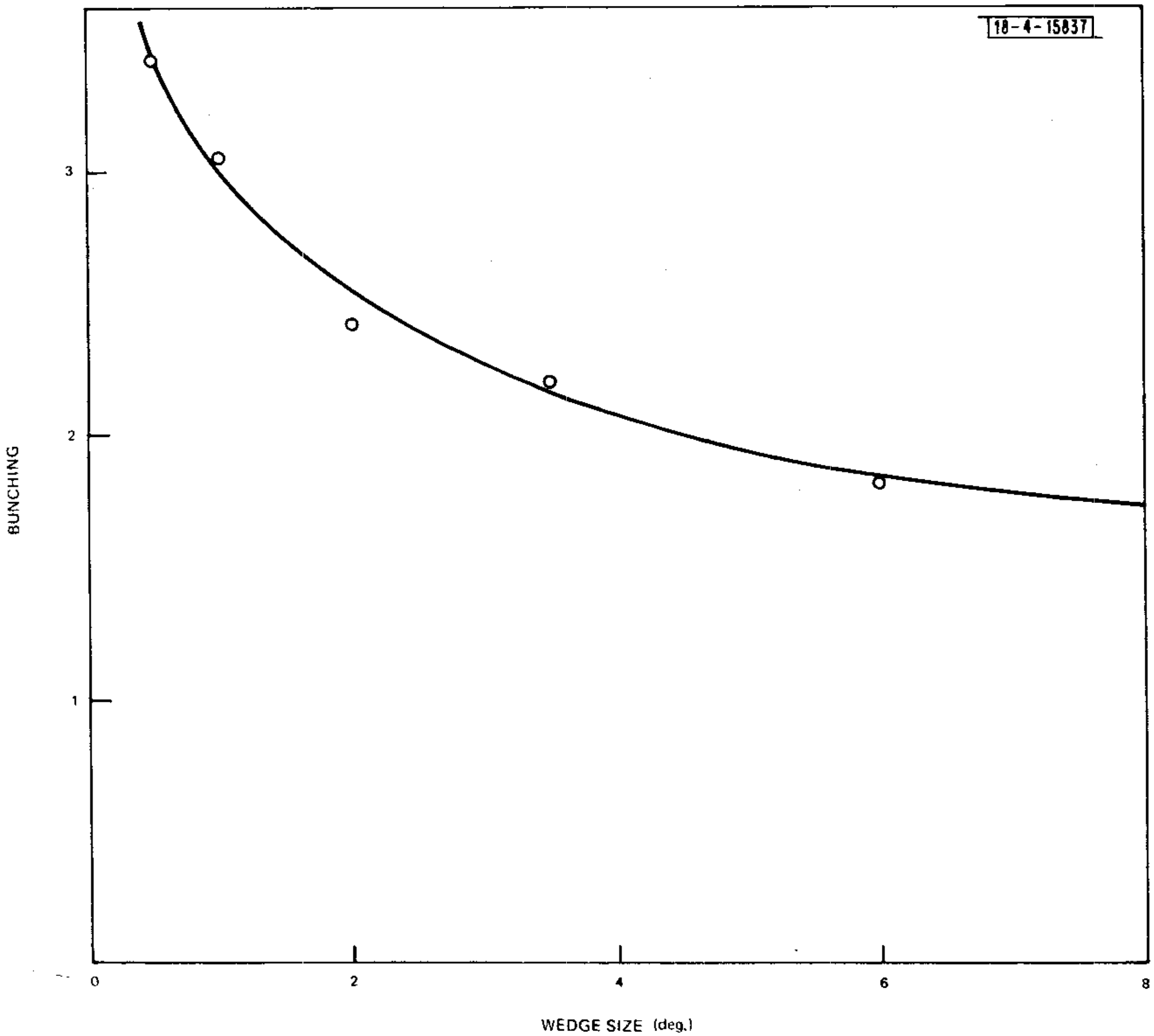
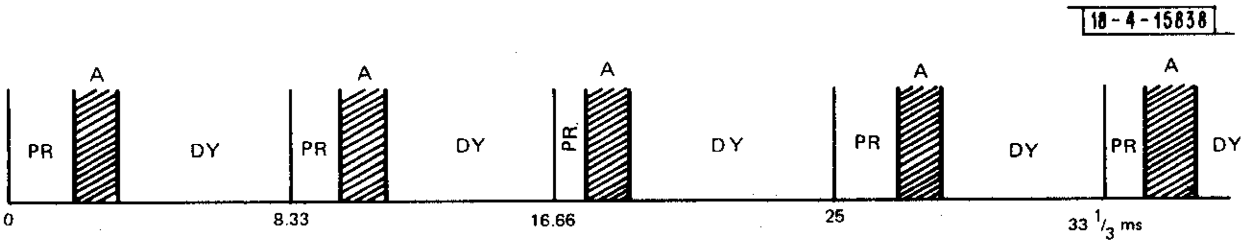


Fig. A-1. Azimuth bunching of aircraft vs wedge size.

18 - 4 - 15838



PR — Prescheduled period

A — ATCRBS/All Call period

DY — Dynamic Scheduled Period

Fig. A-2. Protocol of interrogation management.

Targets that fall in the leading 1/4 beam wedge such as Target ABC, as shown in Figure A-3, are prescheduled ahead of the scan, i.e., prescheduled into the interval of time PR starting at time = 0, as shown in Figure A-2. This prescheduling activity is followed by an ATCRBS sweep whose duration is dependent upon sensor range. This is taken nominally to be 1.24 ms which assumes a sensor range of 100 nmi.

Targets that require reinterrogation due to reply failure are scheduled in the dynamic period following the original prescheduled period. Failure of a dynamically scheduled reinterrogation results in another reinterrogation. This process continues and carries over into succeeding dynamic periods as necessary until a successful reply is received or until the maximum number of interrogations to be allocated to a given target in a scan has been reached. This maximum number is a function of the delivery reliability required and the level of random interference which is responsible for the interrogation or reply failures. Preliminary data indicates that reasonable values for this maximum count are 5 interrogations for routine surveillance interrogations and 10 interrogations for the delivery of an urgent communications message. This insures the reliability specified for the link performance as designed. It should be recalled that fading is not an issue in these limits since ground diversity is assumed, thus at least one of the assigned sensors will have a favorable view of the transponder antenna. In the case of a stand alone sensor operation, the assumption of freedom from fading is no longer valid. Thus it may be desirable to raise the maximum count (perhaps doubling the above limits) in order to improve the delivery reliability for the stand alone sensor.

18-4-15839

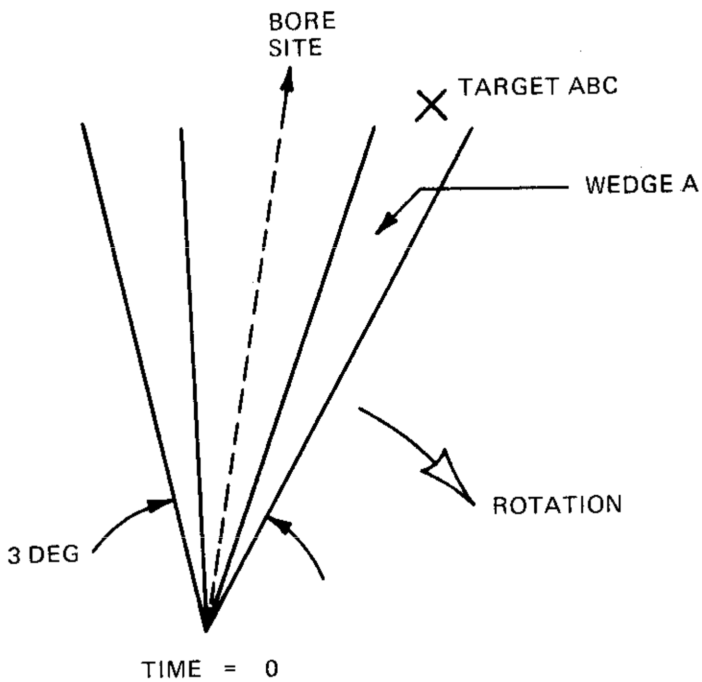


Fig. A-3. Position of beam at start of interval.

The close range (less than 3 nmi) targets that are assigned to more than one 1/4 beamwidth are interrogated at the beginning of the dynamically scheduled period and omitted from the prescheduled interval. They will be limited to 2 surveillance interrogations in each dynamic interval corresponding to each quarter beam to which they are assigned.

In the preceding discussion, only one class of targets requiring dynamic rescheduling was implied; namely, the targets for which the reply processor failed to receive a good reply. A second class of targets placing demands on the dynamic scheduler are those requiring additional interrogations for the movement of uplink or downlink messages. They will be treated in a similar way as a target that failed to reply except that there might be a sequence of communication messages to be transmitted or received instead of a single uplink and downlink message.

The time interval in which either class of reinterrogation is permitted was shown previously in Figure A-2. This is composed of three dynamic intervals (DY) within the antenna dwell time and not four since the last interval cannot be reliably used due to antenna motion during the execution of an interrogation cycle.

Prescheduling of targets is initiated by the interrogation management function at the beginning of each scan for all interrogation cycles in the final two quadrants of that scan. Similarly, prescheduling for the first two quandrants is initiated at the middle of the preceding scan.

1.3 THE SCHEDULING ALGORITHM

DYNO is used for prescheduling as the targets will be grouped into many azimuth sectors for scheduling purposes, and the low computation in initialization overhead of the algorithm at the beginning of scheduling a group of targets is therefore an advantage. It is also used for dynamic scheduling where targets are scheduled one at a time in whatever sequence they are received.

For the messages under consideration, a bin size of 19 μsec gives a good compromise between computation effort and packing efficiency. The number of bins required in the algorithm for the different messages is given as follows:

Message Type	No. of bins
Communication format uplink	2
Communication format downlink	5
Surveillance format uplink	1
Surveillance format downlink	3

For an arbitrary length uplink message, the number of bins required is the multiple of 19 μsec that is equal or just greater than the message. A similar relation holds for the downlink message except that an additional bin is required to accommodate the truncation error of propagation delay.

The algorithm insures that the start of an interrogation is separated by at least 50 μsec from the start of the previous one by separating the beginning of an interrogation from the end of the last scheduled one by a minimum of one bin. The algorithm, when used for dynamic scheduling, has an additional step to allow a minimum of 250 μsec from the time the scheduling is computed for a target to the time the interrogation is supposed to be transmitted. This assumes no delay between the receipt of the reply and the request for an interrogation.

The interrogation management function would then be capable of scheduling about 2000 targets for a 100 mile sensor. This has a factor of two for the assumed load. The capacity of the system as a function of range is given in Figure A-4.

2.0 AGILE BEAM SENSOR

The interrogation management function described in this section is designed for an agile beam sensor. The system is characterized by the parameters given in the previous section with the following modifications:

Switching time	5 μsec
Surveillance reliability	99.9%
Number of aircraft	8000

In an agile beam, it is possible to position the beam at any time within the switching delay time so that the reported position of the target is at boresite. Effectively, this makes all the targets available throughout the scan. To reduce the scan azimuth prediction uncertainty, the azimuth at which a target is interrogated is computed after the schedule is executed. An exception to this are targets at very close range (i.e., less than 4 miles), where the azimuth prediction uncertainty necessitate scheduling several azimuth separated interrogation to ensure that at least one of the interrogations will produce a usable reply.

2.1 SCHEDULING PROTOCOL

A channel protocol similar to the rotating beam sensor is used except that targets are not restricted to be interrogated in specified prescheduling periods or dynamic scheduling periods. As in the rotating beam sensor there is an ATCRBS/ALL-CALL period (A), preschedule period (PR), and dynamically scheduled period (DY). The PR and DY period are set at equal length.

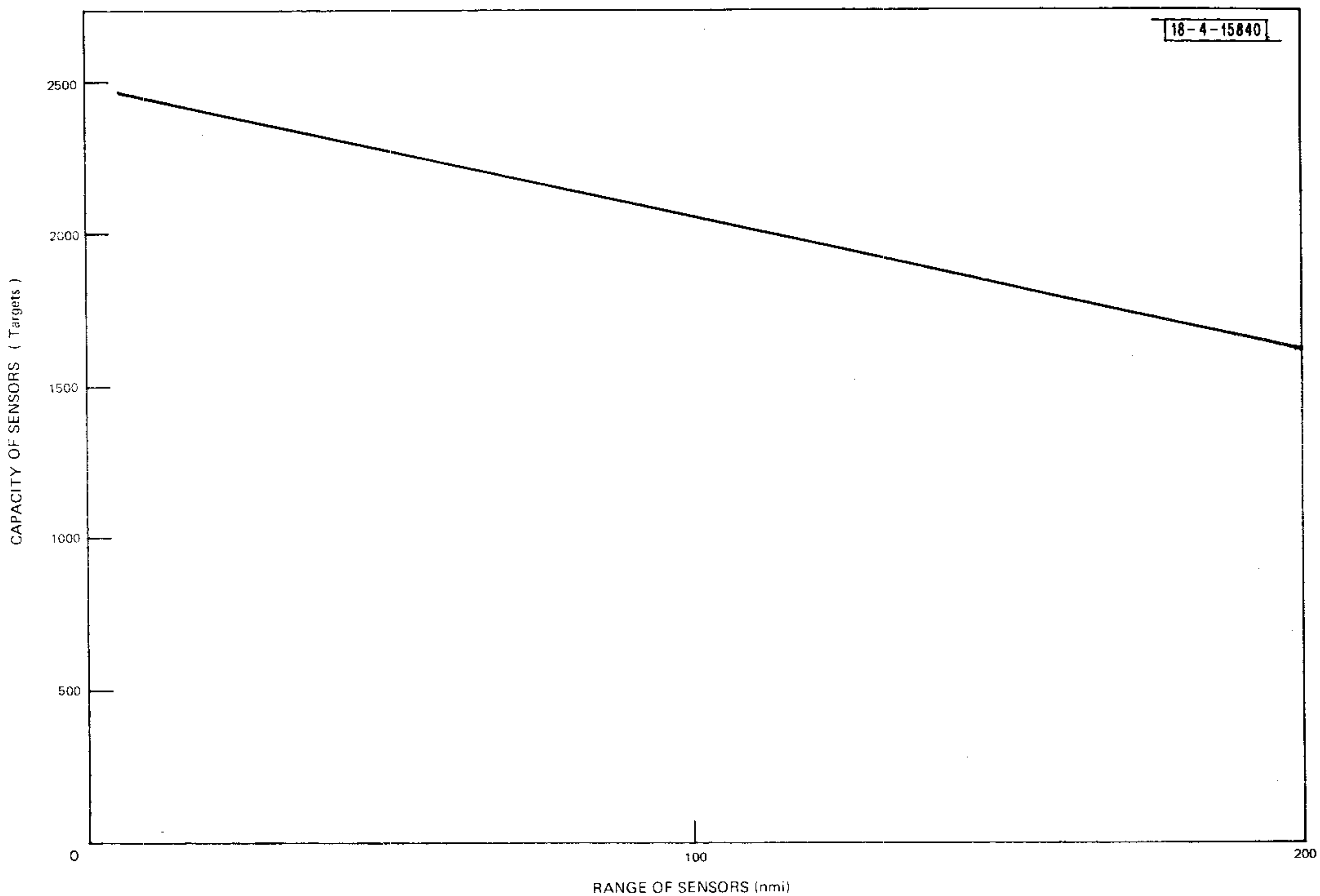
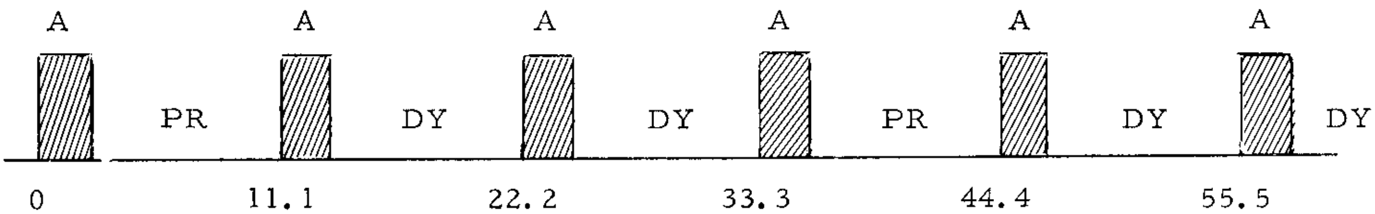


Fig. A-4. Rotating beam sensor capacity.

The interlace pattern of these is given by

PR_A_DY_A_DY_A

as shown in Figure A-5.



The ALL-CALL sweep occurs somewhere in the hashed area.

Figure A-5. Interlace pattern on channel.

The use of two (DY) periods for every (PR) period provides sufficient dynamic scheduling time for the highly packed preschedule period to insure the specified reliability.

The execution of prescheduling in isolated sparse intervals reduces the correlation of the uplink interference of a sensor on a target in its side-lobe. Also, the DABS periods PR and DY are of sufficient length for the scheduling algorithms to have good packing efficiency.

If the load on the sensor is low, every m^{th} DABS period is used for prescheduling where m is in the ratio to 3 as the sensor target capacity to the target load on the sensor.

The ATCRBS ALL-CALL duration is dependent upon sensor range. In Figure A-5 this is taken nominally to be 1.3 msec which assumes sensor range of 100 miles. Successive ATCRBS sweeps are addressed at 1 degree increments in azimuth with a mean pulse repetition frequency of 90. This gives the required ATCRBS run length of 4 on a target every scan.

The time between successive ATCRBS sweeps is varied by pseudo-random value of few microseconds from a period of 11.11 msec to eliminate the synchronization of fruit interference of one sensor on other sensors. For example, in the case of 100 mile sensor 1.6 msec are reserved for the ALL-CALL mode of which 1.3 msec are actually used. The 0.3 msec are used for jitter of the start of the ALL-CALL sweep. This leaves 9.5 msec for DABS preschedule period (PR) or dynamic schedule period (DY).

Targets are assigned one call in the preschedule period for direction finding plus additional calls for communication if required. Targets that require reinterrogation due to reply failure or communication request are scheduled in the dynamic period following the original prescheduled period. Failure of a dynamically scheduled reinterrogation results in another reinterrogation. This process continues and carries over into succeeding dynamic periods as necessary until a successful reply is received or the communication protocol is terminated or until the maximum number of interrogations to be allocated to a given target per message in a scan has been reached. This maximum number is a function of the delivery reliability required and the level of random interference which is responsible for the interrogation or reply failure. Preliminary data indicates a reasonable value for this maximum interrogation is 20. This value is larger than the one used in a rotating beam sensor as the sensor could accommodate it without the constrained access to a target. In the case of close range targets, the maximum number of reinterrogations is restricted to two.

2.2 THE SCHEDULING ALGORITHM

Close-fit is used to preschedule targets ahead of the scan. It is especially suited for the task due to its high packing efficiency and low computation effort for large number of targets with unequal message length. It is to be noted that if the messages were of one fixed value the Full-Ring algorithm would have been used. The switching time of the sensor is added to the interrogation message length. The output of the algorithm which is in the form of cycles is divided by the Loop-Loop algorithm into groups of cycles with each group accommodate in a preschedule period.

The prescheduling function is initiated at the beginning of every half scan for the next half scan. Half the targets are interrogated each half scan.

DYNO is used to schedule targets in the dynamic period (PR). The algorithm is suited to the function due to its ability to schedule targets in the sequence they are received and its low computation effort per target. The above interrogation management can handle 16000 targets which is a factor of two higher than the given load. Its performance as a function of sensor range is given in Figure A-6.

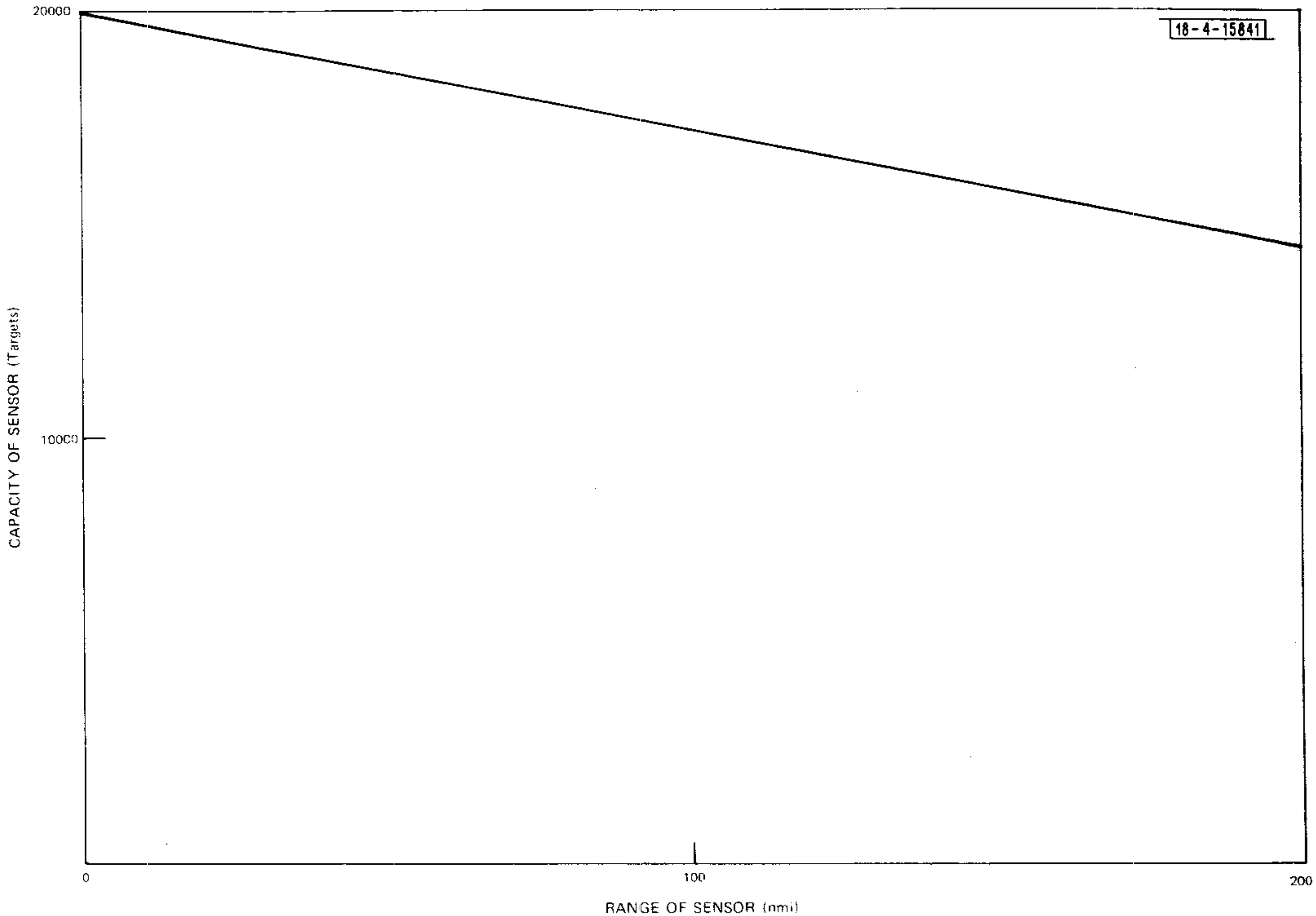


Fig. A-6. Agile beam sensor capacity.

APPENDIX B

"FULL RING" ALGORITHM

1.0 BIN-SORT AND LIST STRUCTURE MECHANISM

Define a set of range bins so that the NBINth bin has maximum range for the system. That is, if the bin size we are interested in is the order of 0.1 nmi, then for a 60 mile system, the 601st bin is at the maximum range.

Next, define a bin address array, where each element in the array corresponds to a range. Call this array BAD(N). Set this array to zero to initialize all elements. Thus, BAD(N) = 0 for any N = 1, . . . , NBIN. This says there are no elements or targets at any range.

Define a target address, or index, link array. Call this array LINKT (NT), were NT = 1, 2, . . . , no. of targets. This array links the indices of the target array, which is the list of input targets as their bin addresses, N, are calculated from the target's true range.

That is, with BAD(1, . . . , NBIN) = 0 initially, calculate which range bin, N, the Jth target will fall into. In sequential order, (1) store the contents of BAD(N) into LINKT(J), and (2) store the index J into BAD(N). Thus, all targets that fall into BAD(N) will have their J indices linked.

At the end of the Bin-Sort and linking operation, the BAD array will contain, for each N, either zero or the index of the last target found in range bin N. In effect, the BAD array can be considered as a series of stacked lists of target indices, linked by LINDT, with each stack found empty, by sensing for BAD(N) = 0. Each list is the target array subscripts for targets which occur in the same bin. The list ends with BAD(N) = 0 (void list) or LINKT (J) = 0.

The following hypothetical system should illustrate the setting up of the arrays and the method of retrieval.

Our system will have a maximum range of 16 nmi, a range bin size resolution of 1 nmi, and a set of 8 targets, TT(1, . . . , 8) with range in nmi. Let, TT(1)=7, TT(2)=3, TT(3)=7, TT(4)=15, TT(5)=12, TT(6)=7, TT(7)=3, and TT(8)=15.

The results of the Bin-Sort and Linking operation follow.

A. Intermediate States

BAD(N)

N =	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Ranges
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Initialize to zero
			2				1					5			4			Indices of targets
			7				3								8			at these ranges
							6											

B. Final States

BAD(N)

N =	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Ranges
	0	0	7	0	0	0	6	0	0	0	0	5	0	0	8	0	0	Indices of targets

LINKT(J)

J =	1	2	3	4	5	6	7	8	Target Count
	0	0	1	0	0	3	2	4	Array of linked target indices

Thus, only BAD(3, 7, 12, 15) contain one or more targets. The remaining bins are zero, meaning there are no targets at these ranges.

To retrieve an element from bin N, one simply addresses that element by BAD(N). Thus, to retrieve a target at Range = 7, BAD(7) contains the index of a target at Range = 7. Addressing the target array by TT(BAD(7)) gets us that target. To acquire other target addresses, we replace BAD(7) by LINKT (BAD(7)). (BAD(7)) ← LINKT (BAD(7)). If the new value of BAD(7) as a result of this operation, equals zero (terminate), we have sensed that there are no more targets to be found at Range = 7.

18-4-15842

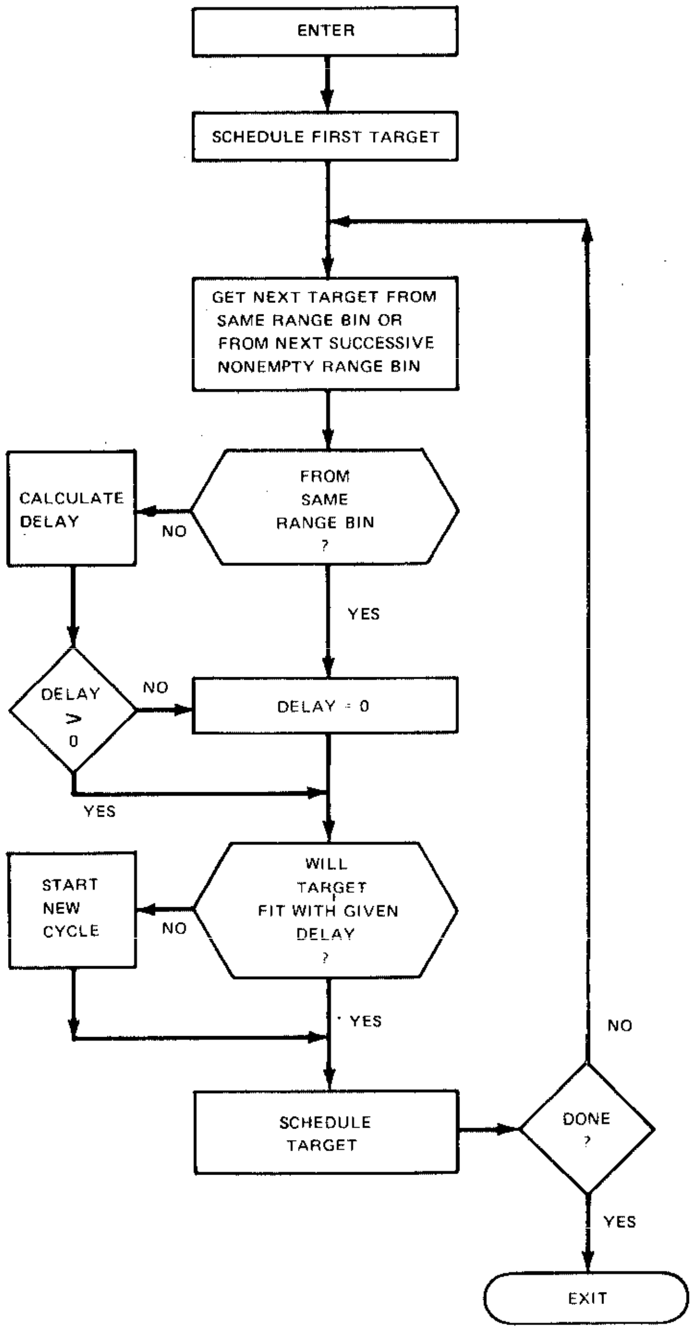


Fig. B-1. Flow diagram of Full-Ring algorithm.

2.0 FORTRAN LISTING

FILE: FULL FORTRAN P1 08/28/73 16:37 M.I.T. LINCOLN LABORATORY

```fortran
 20    CONTINUE
CXX SCHEDULE FIRST TARGET.
        U=0.0
       XINT(NN)=U
       XREP(NN)=U+TR+RR(N)
       NP=N
       M=NN
 40    CONTINUE
CXX GET NEXT TARGET FROM SAME RANGE BIN AS PREVIOUS TARGET SCHEDULED,OR
CXX FROM NEXT SUCCESSIVE NON-EMPTY RANGE BIN.
       J=BAD(N)
       BAD(N)=LINKT(J)
 24    CONTINUE
        IF(BAD(N).NE.0)GO TO 22
       N=N-1
       GO TO 24
 22    CONTINUE
CXX DOES TARGET COME FROM SAME RANGE BIN AS PREVIOUS TARGET SCHEDULED?
       IF(N.NE.NP)GO TO 42
       GO TO 36
 42    CONTINUE
CXX TARGET FROM ANOTHER RANGE BIN THAN PREVIOUS TARGET'S BIN IS TO
CXX BE SCHEDULED.
CXX CALCULATE DELAY.
       DELAY=RR(NP)-RR(N)-DELTA
       NP=N
CXX IS DELAY GREATER THAN ZERO?
       IF(DELAY.GE.0.0)GO TO 38
CXX IF DELAY IS LESS THAN ZERO SET DELAY TO ZERO.
 36    CONTINUE
       DELAY=0.0
 38    CONTINUE
CXX WILL TARGET FIT IN THIS CYCLE WITH THE GIVEN DELAY?
       IF(XREP(M)-U.GE.2.0*TR+DELAY)GO TO 30
       U=XREP(NN)+1.0
       NN=NN+NN1
       M=NN
       GO TO 28
 30    CONTINUE
CXX SCHEDULE THIS TARGET.
       U=U+TR+DELAY
       NN=NN+NN1
 28    CONTINUE
        XINT(NN)=U
       XREP(NN)=U+TR+RR(N)
CXX TEST IF ALL TARGETS HAVE BEEN SCHEDULED.
       IF(NN.EQ.NN2)GO TO 6
CXX GET NEXT TARGET.
       GO TO 40
  6     CONTINUE
CXX DONE WITH SCHEDULE.
       END
```

3.0 VARIABLE DEFINITION

List of Program Variables and Definitions

U	=	The time for the previous target's interrogation.
XINT	=	Array for the time position for the interrogations.
XREP	=	Array for the time positions for the replies.
RR	=	Array of quantized ranges, as large as the number of range bins.
TR	=	The message length ratio. Always ≥ 1.0.
NP	=	Range bin index for previous target scheduled.
N	=	Current range bin for current target to be scheduled.
M	=	Index pointer to the first future reply for a string of consecutive interrogations.
BAD	=	Range bin array which contains the address of a set of targets at the quantized ranges.
J	=	Address of a target at the desired range.
LINKT	=	Array which links the addresses of targets that have the same range quantization attribute.
DELTA	=	The difference between the message length ratios (TR-1.0).
DELAY	=	The amount of time needed to delay the interrogation in order to abut the replies.
NN	=	Index for XINT and XREP arrays.
NN1	=	Increment on NN.
NN2	=	The number of targets.

APPENDIX C

"CLOSE-FIT" ALGORITHM

1.0 BIN SORT AND LINK MECHANISM

Define a set of time bins so that the NBINth bin has the maximum value expected for the sum of the propagation delay and interrogation or reply length. That is, if the bin size is 5 μsec and the maximum message length is 200 μsec, then for 750 μsec maximum propagation system there are 190 time bins. If guard bins are added at the beginning and at the end, the total number NBIN of bins would be 192. Where the 191st bin represents the maximum of the sum of propagation delay and message length for targets under consideration.

Next, define a bin address array the same size as the number of bins required, where each element in the array corresponds to a value of the sum of propagation delay and interrogation. Call this BAD(N). Clear this array to zero to initialize all elements. Thus BAD(N) = 0, for N=1, . . . , NBIN. This says that there are no targets at any range.

Define a bin address value, BINVAL(N) where each element in the array corresponds to an element in the bin address array BAD(N), giving the value of the sum of propagation delay and interrogation associated with the beginning of the bin. Define BINVAL(1) = - ∞, BINVAL(NBIN) = + ∞, and BINVAL(N) = N-1 for N=2.., NBIN-1.

Define a target address or index, link array. Call this array LINK (NT), where NT=1, 2, ..., no. of targets. This array links the indices of the target array as their bin address, N, is calculated from the sum of the target propagation delay and interrogation message length and a one is added to account for the displacement of a guard bin at the beginning.

That is, with BAD(1, ..., NBIN)=0 initially, calculate which range bin, N, the target will fall into. In sequential order, (1) store the contents of BAD(N) into LINK(J), and (2) store the index J into BAD(N). Thus, all targets that fall into BAD(N) will have their J indices linked. BAD(1) and BAD(NBIN) are filled with 10000 as an arbitrary number indicating they are occupied with fictitious targets.

Define LEFT(N) a left link array the size of the BAD(N) array. The elements LEFT(J) of the array gives, for the corresponding element - BAD(J), the index i in the BAD(N) array that has a target in it and is just lower than J. That is, if BAD(14)=0 and BAD(13)=6, then LEFT(15)=13. After the bin sort of targets, and index i is stepped through 2 to NBIN and the following steps are carried sequentially, (1) store in LEFT(i+1) the last value of i, i.e., k where BAD(k)≠0, and (2) increment i by one and check content of BAD(i). If it is not zero, set k=i. The value of k is initially set to one. After this scan the left links are created.

In a simular way, a right link array is established.

At the end of the Bin-Sort and linking operation, the BAD array element BAD(i) for i=2, NBIN-1, will contain for each N either zero or the index of the last target found at the quantized sum of propagation delay and interrogation of N-1. A zero indicates an empty bin. BINVAL(J) gives the value of quantized time associated with BAD(J), LEFT(J) gives index I such that BAD(I) $\neq$ 0, I < J and $|I-J|$ is min, RIGHT(J) gives index P such that BAD(P) $\neq$ 0, P > J and $|P-J|$ is min.

The following hypothetical system should illustrate the setting up of the arrays and method of retrieval.

Our example will have a maximum propagation delay of 40 μsec, a maximum message length of 5 μsec, and a bin resolution of 5 μsec. Let the target propagation delay R and interrogation message length INT be:

R (1) = 7	INT (1) = 5	R (1) + INT (1) = 12
R (2) = 23	INT (2) = 4	R (2) + INT (2) = 27
R (3) = 9	INT (3) = 2	R (3) + INT (3) = 11
R (4) = 7	INT (4) = 1	R (4) + INT (4) = 8
R (5) = 39	INT (5) = 2	R (5) + INT (5) = 41
R (6) = 40	INT (6) = 5	R (6) + INT (6) = 45
R (7) = 29	INT (7) = 4	R (7) + INT (7) = 33

A. Intermediate States

BAD(N)

N =	1	2	3	4	5	6	7	8	9	10	11
	0	0	0	0	0	0	0	0	0	0	0
	10000		4	1			2	7		5	10000
				3						6	

B. Final States

BAD(N)

N =	1	2	3	4	5	6	7	8	9	10	11
	10000	0	4	3	0	0	2	7	0	6	10000

LINKT(J)

J =	1	2	3	4	5	6	7
	0	0	1	0	0	5	0

BINVAL(N)

N =	1	2	3	4	5	6	7	8	9	10	11
	-10^6	1	2	3	4	5	6	7	8	9	10^6

LEFT(N)

N =	1	2	3	4	5	6	7	8	9	10	11
	1	1	1	3	4	4	4	7	8	8	10

RIGHT(N)

N =	1	2	3	4	5	6	7	8	9	10	11
	3	3	4	7	7	7	8	10	10	11	11

If a target is required with quantized value of the sum of propagation delay and interrogation of 5, a check on BAD (6) shows it is empty. Using the thread through the right links BAD (7) is located, and using the thread on the left BAD (4) is located. These are respectively the nearest occupied left and right bins. However, BAD (7) is chosen since BINVAL (7) is closer to BINVAL (6) than BINVAL (4) is to BINVAL (6), and target 2 is selected since it resides in bin address 7. The use of the guard bins at the extremeties becomes evident when a target with a quantized value of 1 is sought; then BAD (1) and BAD (3) are the candidate bin addresses. BAD (3) is chosen, based on the proximity criteria of bin values, giving target 4 which is actually the target with the closest to the desired value.

In a system where reply message lengths are longer than the interrogation message lengths, the lengths of the reply and interrogations for a target are interchanged, before executing the algorithm. The actual time reserved for interrogating a target would be the time reserved for the reply in the output of the above schedule relative to the end of the last reply in the schedule. In a similar way, the position for a reply is the time reserved for the interrogation relative to the end of the schedule. This requires a pass in the output of the algorithm to carry the transformation. This reversal in role between interrogations and replies increases the frequency of the algorithm requests for targets at successively smaller time bins for successive targets in a cycle. This increases the probability of finding targets at the desired bin since the cycle starts with a target from an occupied bin at maximum time.

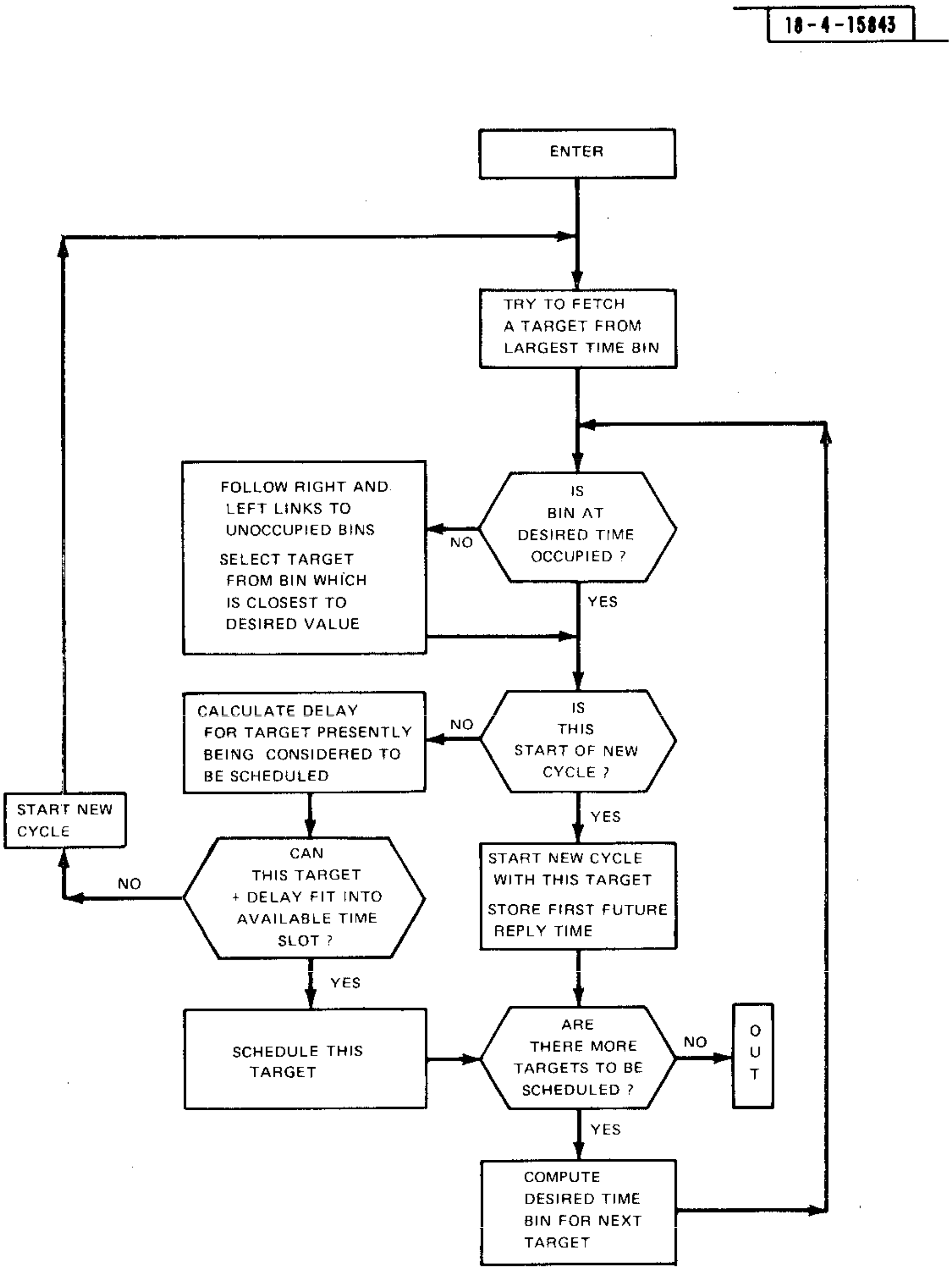


Fig. C-1. Close-Fit algorithm.

2.0 FORTRAN LISTING

FILE: CLOSE FORTRAN P1 08/28/73 16:37 M.I.T. LINCOLN LABORATORY

```
CXX INITIALIZATION; NO ASSIGNMENTS, TIME ZERO
      NASS=0
      U=0.0
      GO TO 2301
 7302 CONTINUE
CXX CALCULATE DELAY FOR TARGET PRESENTLY BEING CONSIDERED TO BE
CXX SCHEDULED.
      DELAY=0.0
      NPRES=BAD(NTENT+DISP)
      DELAY1=TT(NP1)+YINT(NP1)-TT(NPRES)-YREP(NPRES)
      IF(DELAY1.LT.0.0)GO TO 38
      DELAY=DELAY1
 38   CONTINUE
CXX CAN THIS TARGET PLUS DELAY FIT IN THE AVAILABLE TIME SLOT?
      IF(REPM-U.GE.YREP(NP1)+YREP(NPRES)+DELAY)GO TO 30
CXX IF TARGET CAN NOT FIT, START A NEW CYCLE WITH THIS TARGET.
CXX MOVE U TO THE END OF THE LAST REPLY SCHEDULED.
      U=XREP(NN)+YINT(NP1)
      GO TO 2301
 7301 CONTINUE
CXX START A NEW CYCLE.
      NN=NN+NN1
CXX SCHEDULE INTERROGATION.
      XINT(NN)=U
CXX SCHEDULE REPLY.
      XREP(NN)=U+YREP(NP1)+TT(NP1)
CXX STORE FIRST FUTURE REPLY TIME.
      REPM=XREP(NN)
      NASS=NASS+1
      IF(NASS.EQ.NT)GO TO 6
CXX IF NOT THE LAST TARGET TO BE SCHEDULED, COMPUTE THE DESIRED TIME BIN
CXX FOR THE NEXT TARGET.
      NTENT=1.0+(TT(NP1)+YINT(NP1))*GNU
      GO TO 2310
 30   CONTINUE
CXX THIS TARGET FITS WITH COMPUTED DELAY.
CXX COMPUTE NEW POINTER FOR INTERROGATION.
      U=U+YREP(NP1)+DELAY
      NN=NN+NN1
CXX SCHEDULE INTERROGATION.
      XINT(NN)=U
CXX SCHEDULE REPLY.
      XREP(NN)=U+YREP(NPRES)+TT(NPRES)
      NP1=NPRES
CXX STORE THE INDEX OF THE NEXT TARGET IN THE CHAIN OF TERGETS
CXX WITH THIS QUANTIZED RANGE ATTRIBUTE.
      BAD(NTENT+DISP)=LINK(NPRES)
      NASS=NASS+1
      IF(NASS.EQ.NT)GO TO 6
 9001 CONTINUE
CXX COMPUTE THE DESIRED RANGE ATTRIBUTE FOR THE NEXT TARGET.
      NTENT=1.0+(TT(NP1)+YINT(NP1))*GNU
      GO TO 2310
 2301 CONTINUE
CXX FETCH THE TARGET FROM AN OCCUPIED BIN ASSOCIATED WITH THE LARGEST
```

(continued)

```
CXX QUANTIZED RANGE ATTRIBUTE.
      NTENT=IX
      NEWFL=1
 2310 CONTINUE
CXX IF THE DESIRED RANGE ATTRIBUTE BIN IS OCCUPIED, TRY
CXX TO SCHEDULE THIS TARGET.
      IF(BAD(NTENT+DISP).NE.0)GO TO 2302
CXX IF THE DESIRED RANGE ATTRIBUTE BIN IS UNOCCUPIED,
CXX FOLLOW  THE RIGHT AND LEFT LINKS TO UNOCCUPIED BINS.
      IRIGHT=RIGHT(NTENT+DISP)
      ILEFT=LEFT(NTENT+DISP)
 2304 CONTINUE
      IF(BAD(IRIGHT).NE.0)GO TO 2303
      IRIGHT=RIGHT(IRIGHT)
      GO TO 2304
 2303 CONTINUE
      IF(BAD(ILEFT).NE.0)GO TO 2305
      ILEFT=LEFT(ILEFT)
      GO TO 2303
 2305 CONTINUE
CXX SELECT THE TARGET FROM THE BIN WHICH IS CLOSEST TO THE DESIRED
CXX BIN VALUE.
      NTENT1=BINVAL(IRIGHT)
      IF(IABS(NTENT+DISP-BINVAL(IRIGHT)).GT.IABS(NTENT+DISP-BINVAL(ILEFT
     1)))NTENT1=BINVAL(ILEFT)
      NTENT=NTENT1-DISP
 2302 CONTINUE
CXX IF NOT THE START OF A NEW CYCLE, TRY SCHEDULING THIS
CXX TARGET IN THE OLD CYCLE.
      IF(NEWFL.EQ.0 ) GO TO 7302
CXX IF A NEW CYCLE IS TO BE STARTED, SET IX
CXX VARIALBE TO THE INDEX INTO THE BAD ARRAY AT
CXX WHICH A TARGET WITH THE LARGEST RANGE ATTRIBUTE
CXX CAN BE FOUND.
      NEWFL=0
      IX=NTENT
CXX UPDATE RIGHT LINK.
      RIGHT(IX+DISP)=KBIN
CXX STORE TARGET INDEX.
      NP1=BAD(IX+DISP)
CXX UPDATE BAD ARRAY.
      BAD(IX+DISP)=LINKT(NP1)
      GO TO 7301
 6    CONTINUE
      END
```

3.0 VARIABLE DEFINITION

List of Program Variables and Definitions

NASS	=	Number of targets assigned thus far.
U	=	Time for last interrogation.
BAD	=	An array the size of the number of bins, plus two, one on either side for ficticious boundary targets.
N	=	Index into BAD array which corresponds to the time attribute of a target. This time attribute is calculated from the sum of the target's propagation delay and interrogation message length plus 1 to account for the displacement of a guard bin at the beginning.
IX	=	Index into BAD array at which a target with the largest time attribute will be found.
NTENT	=	Index of time bin desired for the next target.
NEWFL	=	Flag which signifies that a new cycle is initiated.
DISP	=	Time attribute displacement which when added to NTENT forms the desired index into the BAD array for a target with the desired time attribute.
RIGHT	=	An array the size of BAD for an index K, such that RIGHT(K) = P with BAD(P) $\neq$ 0, P < K for MIN $\lvert P - K \rvert$.

LEFT	=	An array the size of BAD for an index K, such that LEFT(K) = P with BAD(P) $\neq$ 0, P < K for MIN $\|P - K\|$.
IRIGHT	=	Right index for BAD.
ILEFT	=	Left index for BAD.
BINVAL	=	An array the size of BAD which give the value of the quantized time associated with BAD such that for some index J, BINVAL(J) is the time associated with BAD(J).
KBIN	=	The augmented array size for the discrete time array BAD. This includes the displacement for the boundary conditions.
NP1	=	The index of the last target to be scheduled.
NPRES	=	The index of the target under present scheduling consideration.
TT	=	Array of target ranges. (Propagation delay.)
YINT	=	An array giving the interrogation message lengths of the targets.
YREP	=	An array giving the reply message lengths of the targets.
XINT	=	An array giving the time the interrogation of a target starts.
XREP	=	An array giving the reply message lengths of the targets.
GNU	=	Number of range bins per unit of time measure.
REPM	=	The time of the first reply in the present cycle.

DELAY = The delay of an interrogation to allow replies to be packed back to back if possible.

NT = The number of targets to be scheduled.

NN = The index for the XINT and XREP arrays.

LINK = An array which links the addresses of targets that have the same time quantization attribute.

APPENDIX D

"DYNO" ALGORITHM

1.0 FILE STRUCTURE

The main working file in the algorithm is the one representing the discrete time intervals. The length of the file in words is the length of the time interval into which targets are to be scheduled in units of bins. Thus, if the interval is 2 msec and the bin width is 25 μsec, the number of words needed is 80 words. The File is initialized to zero. Figure D-1 shows the sate of the file at two points in the scheduling activity. The first is before any targets are scheduled. The second after the targets in Table D. 1 are scheduled.

Figure D-2 shows the algorithm output for 14 targets. The first 4 are the ones given in the illustrative example. The TF (N) file is cleared after finishing scheduling. Only the words actually used for replies are cleared. Their position is obtained from the replies position in the schedule output.

The flow diagram for the algorithm is given in Figure D-3.

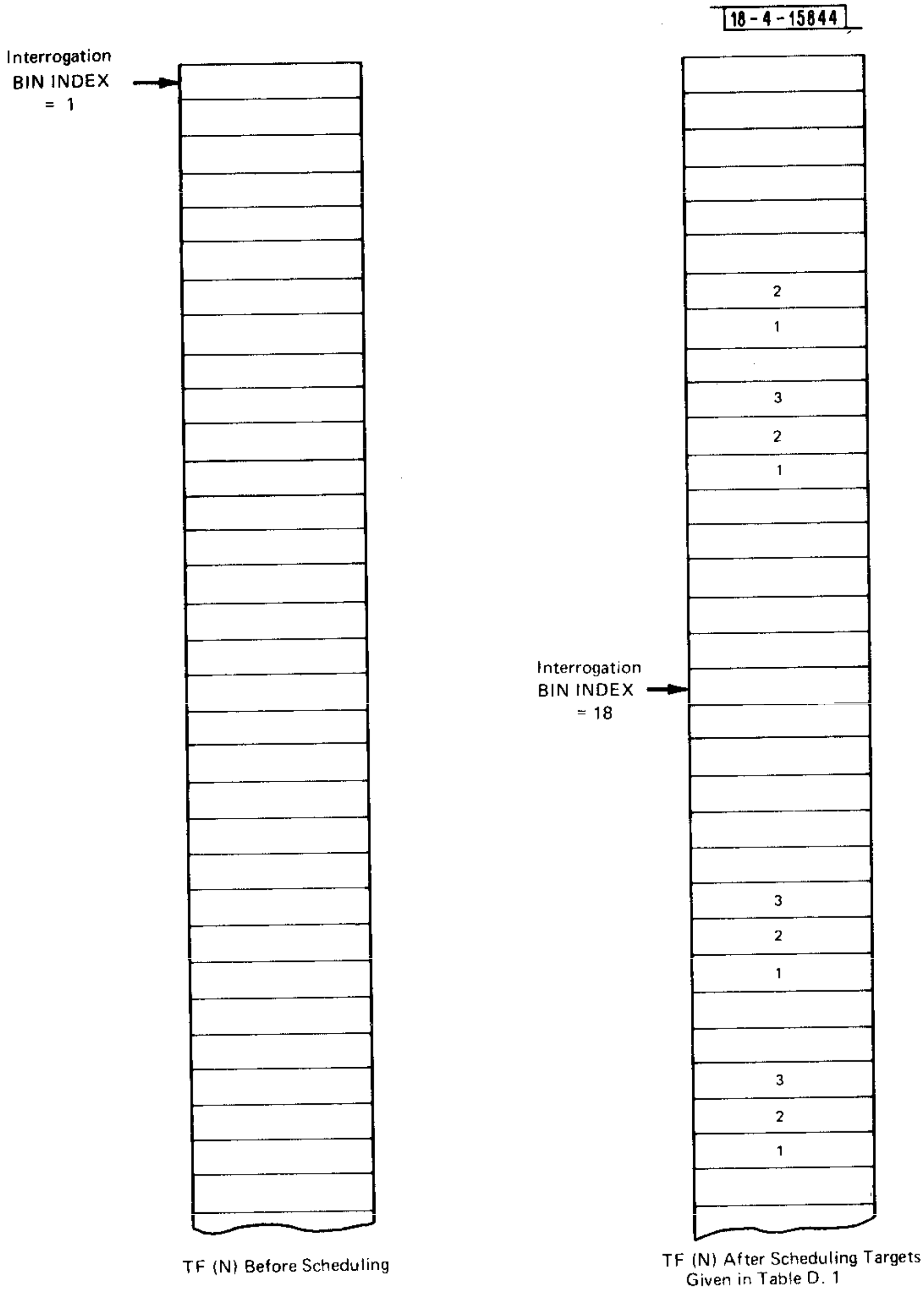


Fig. D-1. Start of time file for Dyno.

Table D.I. Values For Illustrative Example.

Interrogation Message Length	28 μsec
Reply Message Length	46 μsec
Bin Size	28 μsec
Required Separation Between Interrogations	50 μsec
Then Interrogation Message Length	1 bin
Reply Message Length	3 bin

Target Number	Propagation Delay μsec	Propagation Delay Bins
1	115	4
2	20	0
3	425	15
4	227	8

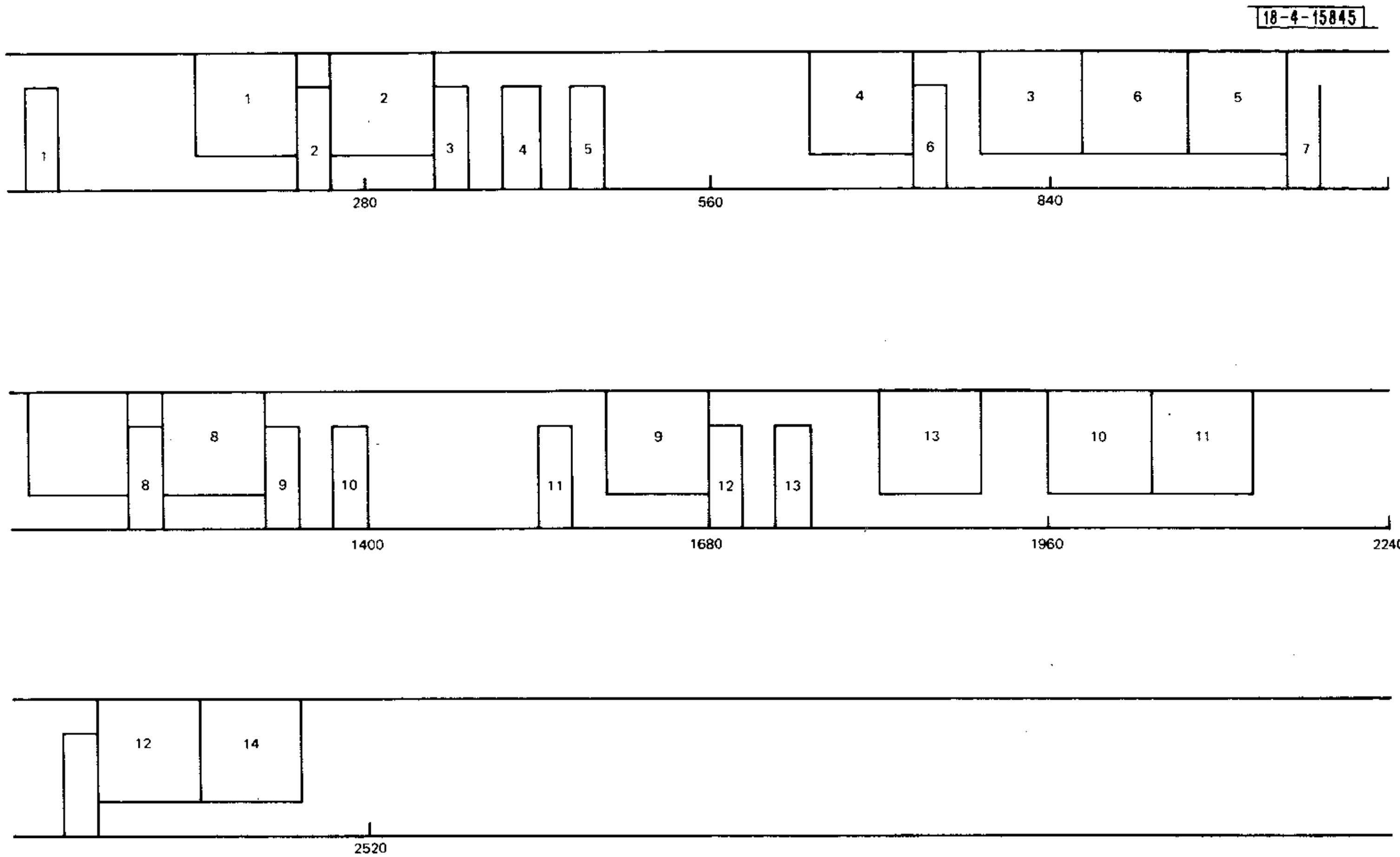


Fig. D-2. Dyno output for 14 targets.

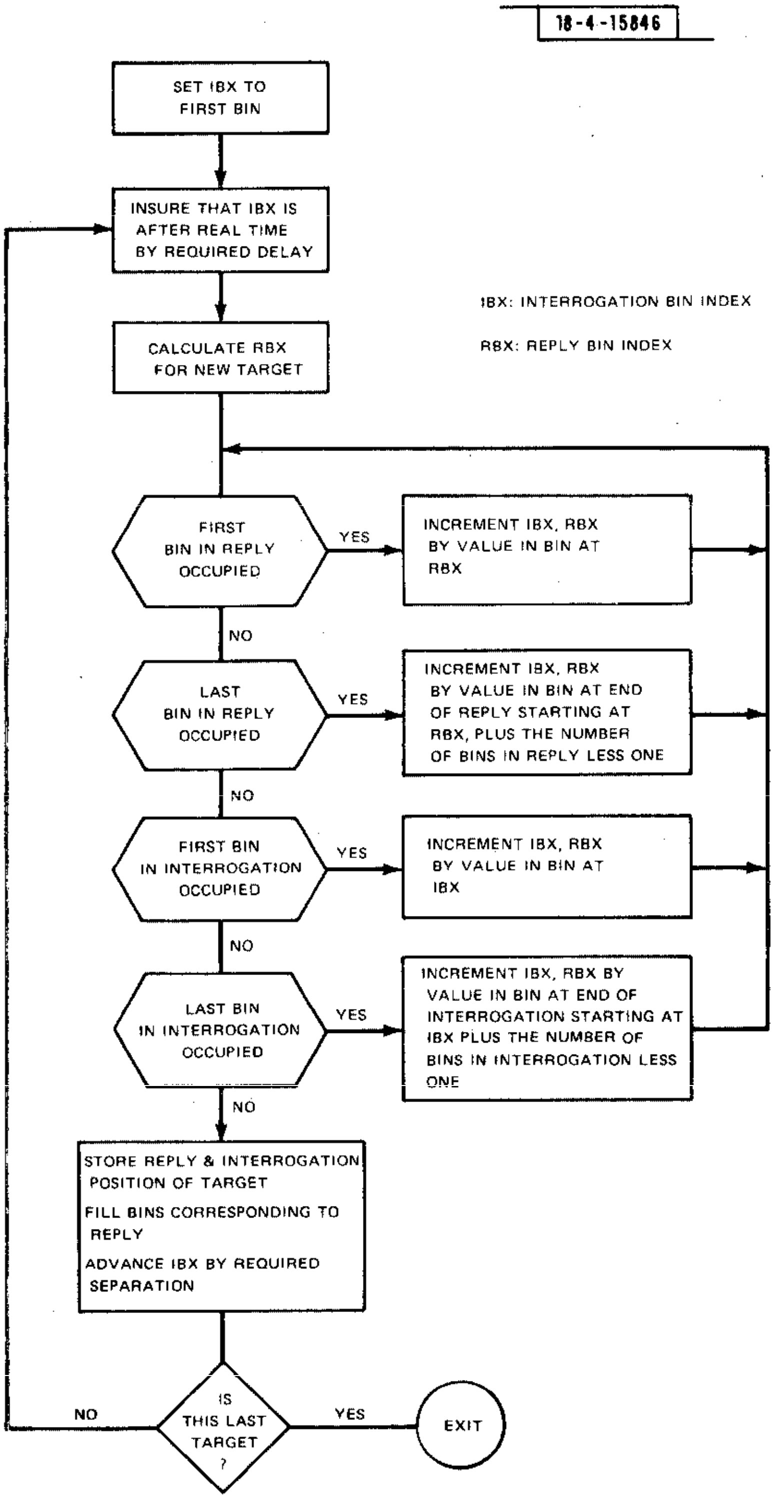


Fig. D-3. Flow diagram of the DABS scheduling algorithm.

2.0 FORTRAN LISTING

```
      J=1
      UI=1
C     ENTRY POINT
 209  UR=UI+NI+RR(J)
C     SEARCH FOR SLOTS
 210  CONTINUE
      LR=UR
      IF(TF(LR).NE.0) GO TO 72
      LR=UR+NR-1
      IF (TF(LR).NE.0) GO TO 73
      LR =UI
      IF(TF(LR).NE.0) GO TO 72
      IF (NI.EQ.1) GO TO 703
      LR=UI+NI-1
      IF(TF(LR).NE.0) GO TO 74
C FILL THE LINK SPOTS
 703  DO 213 IB=1,NR
      TF(IB+UR-1)=TF(NR+UR)+1+NR-IB
 213  CONTINUE
      GO TO 410
 72   CONTINUE
      UR=UR+TF(LR)
      UI=UI+TF(LR)
      GO TO 210
 73   CONTINUE
      UR=UR+TF(LR)+NR-1
      UI=UI+TF(LR)+NR-1
      GO TO 210
 74   CONTINUE
      UR=UR +TF(LR)+NI-1
      UI=UI+TF(LR)+NI-1
      GO TO 210
 410  XINT(J)=UI
      XREP(J)=UR
      IF (J.EQ.NT) GO TO 321
      J = J+1
      UI=UI+NI+IGAP
C     NEXT STEP NOT NEEDED FOR PRESHEDULING
      IF (UI.LT.ICLOCK+IEXE) UI=ICLOCK+IEXE
      GO TO 209
 321  END
```

3.0 VARIABLE DEFINITION

List of Program Variables and Definitions

ICLOCK	=	Real time.
IGAP	=	Minimum number of bins required between end of last interrogation and beginning of the next.
IEXE	=	Minimum time between real time and the time an interrogation is supposed to be transmitted.
J	=	Index of target in input list that is to be scheduled.
LR	=	Index of time bin examined.
NI	=	Length of interrogation in bins.
NR	=	Length of reply in bins.
NT	=	Number of targets to be scheduled.
RR	=	An array for the propagation delay in bins.
UI	=	Bin index for interrogation.
UR	=	Bin index for reply.
XINT	=	Array for target interrogation time in bins.
XREP	=	Array for target reply time in bins.

APPENDIX E

LOOP-LOOP ALGORITHM

1.0

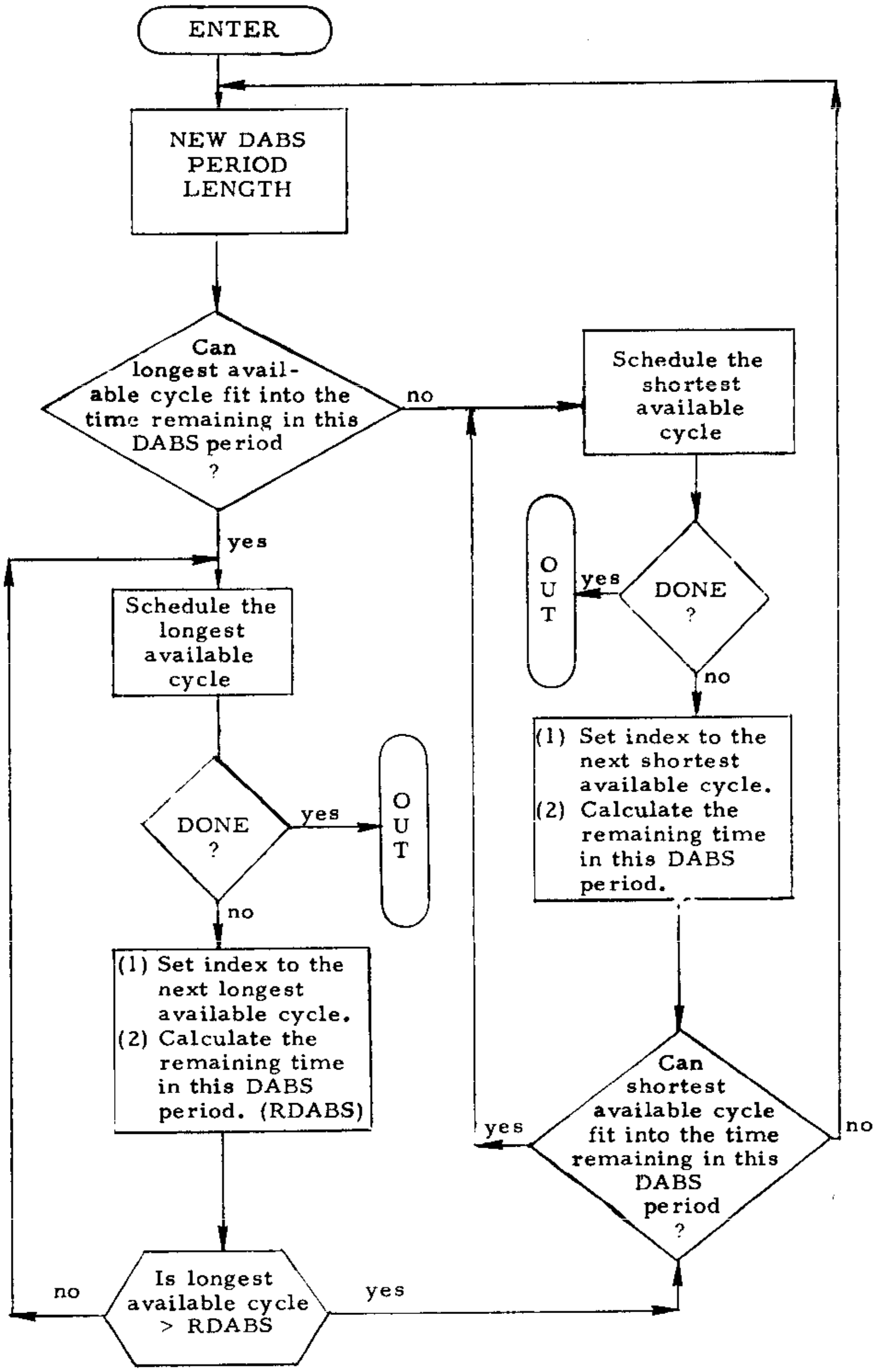


Fig. E-1. Loop-Loop algorithm.

2.0 FORTRAN LISTING


```
      N=0
      NBLOCK=0
      IHIGH=1
      ILOW=NUMCYC
 10   CONTINUE
CXX COUNT DABS PERIODS.
      NBLOCK=NBLOCK+1
CXX CLEAR THE COUNT FOR THE NUMBER OF CYCLES PER DABS PERIOD.
      NCPB=0
CXX RESET DABS PERIOD DURATION, (LENGTH).
      RDABS=DABS
 99   CONTINUE
CXX COUNT THE TOTAL NUMBER OF CYCLES FOR ALL DABS PERIODS.
      N=N+1
CXX STORE THE CYCLE NUMBER OF THIS LONGEST LIVE CYCLE.
      CYCL(N)=IHIGH
      NCPB=NCPB+1
      TOT(NBLOCK)=NCPB
CXX ARE WE DONE?
      IF(N.EQ.NUMCYC)GO TO 100
CXX IF NOT DONE CALCULATE THE REMAINING TIME IN THIS DABS PERIOD.
      RDABS=RDABS-DURAT(IHIGH)
CXX SET INDEX TO NEXT LONGEST LIVE CYCLE.
      IHIGH=IHIGH+1
CXX CAN THE LONGEST LIVE CYCLE FIT INTO THE REMAINING
CXX TIME IN THIS DABS PERIOD?
      IF(DURAT(IHIGH).LE.RDABS)GO TO 99
 14   CONTINUE
CXX CAN THE SHORTEST LIVE CYCLE FIT INTO THE REMAINING DABS PERIOD?
      IF(DURAT(ILOW).GT.RDABS)GO TO 10
CXX IF YES, UP THE COUNT OF THE TOTAL NUMBER OF CYCLES FOR ALL DABS
CXX PERIODS.
      N=N+1
CXX STORE THE CYCLE NUMBER OF THE SHORTEST LIVE CYCLE.
      CYCL(N)=ILOW
      NCPB=NCPB+1
      TOT(NBLOCK)=NCPB
CXX ARE WE DONE?
      IF(N.EQ.NUMCYC)GO TO 100
CXX IF NOT, CALCULATE THE REMAINING TIME IN THIS DABS PERIOD.
      RDABS=RDABS-DURAT(ILOW)
CXX SET INDEX TO NEXT SHORTEST LIVE CYCLE.
      ILOW=ILOW-1
      GO TO 14
 100  CONTINUE
CXX WE ARE DONE.
      END
```

3.0 VARIABLE DEFINITION

List of Program Variables and Definitions

N	=	Number of cycles in all DABS periods.
NBLOCK	=	Index of present DABS period.
IHIGH	=	Index into the cycle length array, for the longest available cycles.
ILOW	=	Index into the cycle length array, for the shortest available cycles.
NCPB	=	Number of cycles in present DABS period.
DABS	=	The DABS period length.
RDABS	=	The remaining time in current DABS period.
CYCL	=	An array which contains the cycle numbers as they are deposited into the DABS periods.
TOT	=	An array which contains the count of the number of cycles per DABS period.
NUMCYC	=	The number of cycles to be scheduled.
DURAT	=	An array which contains the lengths of the cycles produced by the primary scheduling algorithm. (Ordered such that low indexes point to longer cycles and high indexes point to shorter cycles.)

REFERENCES

[1] Quarterly Technical Summary, Development of a Discrete Address Beacon System, Lincoln Laboratory, M.I.T. (1 July 1973).

[2] Ibid.

[3] E. J. Kelly, "Interrogation Scheduling for the Discrete Address Beacon System," Project Report ATC-8, Lincoln Laboratory, M.I.T (24 January 1973).

[4] Knuth, Donald R., The Art of Computer Programming, Vol. 1 (Addison-Wesley, Reading, Mass., 1969).