1982

NASA/ASEE SUMMER FACULTY RESEARCH FELLOWSHIP PROGRAM

MARSHALL SPACE FLIGHT CENTER
THE UNIVERSITY OF ALABAMA

A SCHEDULING ALGORITHM FOR SPACELAB TELESCOPE OBSERVATIONS

Prepared By:	Bob Grone
Academic Rank:	Associate Professor
University and Department:	Auburn University Department of Mathematics
NASA/MSFC:	
(Laboratory)	Systems Analysis & Integration Lab.
(Division)	Mission Analysis
(Branch)	Mission Integration
MSFC Counterpart:	William Askew

Date:	August 1, 1982
Contract No.:	NGT-01-002-009 (University of Alabama)

A SCHEDULING ALGORITHM FOR SPACELAB TELESCOPE OBSERVATIONS

by

Bob Grone, Ph. D.
Associate Professor of Mathematics
Auburn University
Alabama 36849

ABSTRACT

An algorithm is developed for sequencing and scheduling of observations of stellar targets by equipment on Spacelab. The method is a general one, but is motivated by the example of a mission organized by the Office of Space Science denoted by OSS-3. This particular mission, along with interactions with NASA personnel in charge of planning the mission has been the basic model for which this method was developed.

In this paper we define and examine the scheduling problem, exhibit and document the method developed for its solution, and make suggestions for further development and implementation of this method.

TABLE OF CONTENTS

I. INTRODUCTION

The Spacelab OSS-3 mission includes among other experiments a telescope platform having three telescopes. These are referred to as HUT (Hopkins Ultraviolet Telescope), WUPPE (Winconsin Ultraviolet Spectropolarimeter), and UIT (Ultraviolet Imaging Telescope). The telescopes are not independent, but are aimed along a common axis. Targets are specified for each telescope by a scientist in charge referred to a principal investigator. These targets fall into two categories, faint or bright, depending on their magnitude. Furthermore, each target has a viewing requirement consisting of the number of observances requested and the length of an observance. The distribution of the targets is as follows:

UIT faint	-	48
UIT bright	-	0
HUT faint	-	13
HUT bright	-	35
WUPPE faint	-	45
WUPPE bright	-	37

Since none of the targets involve the Earth or Moon, each target may be considered as being in a fixed location. It is expected that all telescopes will function continuously, and so two of the telescopes will be co-observing a target of the third at any given time.

Each target has certain constraints on the times at which it may be viewed. The first type of constraint is geometric, and requires that the target not be occulted by Earth or Moon. Fainter targets may only be viewed while the Earth is occulting the Sun, and may additionally require they be viewed only if they are at least some fixed angular distance from the Earth and/or Moon. When the Orbiter is over a location in the South Atlantic region characterized by high radiation referred to as the South Atlantic Anomaly (SAA), no viewing is to be scheduled. Hence, each target has certain periods at which it can be viewed during the mission, referred to as observation opportunities. The mission is to last 5 days and encompass 72 orbital periods of 90 minutes each. Each orbital period has 36 minutes of shadow and 54 minutes of Sun, which places a premium on the shadow times of the mission.

Basically the problem is to schedule the observations while obeying the constraints and maximizing viewing time. To do this, one wants to minimize the time spent slewing between targets. If we ignore the constraints this would be a single-machine sequencing problem with sequence-dependent setup times, more commonly referred to as the travelling salesman problem.

In the present situation there are some fundamental differences. In the travelling salesman problem one wants to visit a sequence of n clients in n cities in such a way that minimizes the total distance travelled. In the present situation the analogous problem would also have certain hours at which the clients were available and requirements about how many meetings of what length were required by each client. Another basic difference is that we are trying to maximize clients seen in a fixed time rather than minimize the time (or distance) it takes to see all of the clients. Some aspects of a good schedule are not well-defined and involve facts such as; that some clients may have flexible requirements, some clients may be more important than others, and that a balance should be maintained between three types of clients (HUT, UIT, and WUPPE). In this paper we construct a dispatching-type method to be implemented on a VAX computer for this scheduling problem.

The need for such a schedule is two-fold. A method which in a few minutes constructs a near-optimal or very good schedule would clearly save a lot of man-hours and wages. If there is no particular rush in obtaining a schedule, then a machine-produced schedule could be hand-edited to conform to some of the more nebulous criteria applied to such schedules. In the event that the launch of a mission is delayed or there is a malfunction in orbit, this hand-edited schedule becomes so much wishful thinking, and there is an immediate need for a new schedule. For example, if one of the telescopes were to become inoperable after a few hours, a new schedule would be required which only involved targets corresponding to the other two telescopes. It is clear that time is of the essence in such a situation, and that a computerized method is required.

This paper describes an algorithm to accomplish this task. In the section entitled Files, we describe the format of the data which is fed to the program to produce a schedule. In Functions we define certain quantities obtainable from the file which are involved in the algorithm. In Subroutines we identify some components of the algorithm. In the Main Algorithm section we discuss the principal subroutine as well as the overall method. In the Outputs section we describe the desired format in which the results are to be displayed.

II. FILES

To implement this program we must organize the information corresponding to the set of targets we wish to consider into a form which is compatible with the algorithm. In our general discussion we will let {1,...,n} index the set of targets we wish to (try to) schedule. For OSS-3 there were 178 such targets and the indexing scheme actually used was slightly different:

Type of experiment	Number of targets	indices
UIT (faint)	48	101-148
HUT (faint)	13	201-213
HUT (bright)	35	301-335
WUPPE (faint)	45	401-445
WUPPE (bright)	37	501-537

With this indexing format, it is easy to keep track of experiment groups of up to 99 targets each. For most of the rest of this paper we will use the {1,...n} approach for notational simplicity.

For each i=1,...,n, the Observation Requirement File lists the quantities r_i and s_i, where

r_i = the number of observances of target i requested

and

s_i = the time (in minutes) required to complete a single observance of target i.

These quantities are provided by the principal investigators. The quantity s_i is taken to be the minimum time necessary to obtain significant information from observing target i. This quantity is largely a function of the magnitude of target i and the sensitivity of the instrument involved.

For each i = 1,...,n, the Priority File contains an entry p_i which corresponds to the importance of target i. In the algorithm it is assumed that larger numbers denote more important targets so that priority = 2 denotes a more important target than priority = 1. If this were to be reversed the term p_i could be replaced everywhere in the algorithm by $1/p_i$. These priority numbers are decided upon initially by the principal investigators. The following hypothetical situation suggests a question concerning priority values. Suppose a principal investigator groups his targets into four categories of increasing importance. Which of the following assignments of priority values would work best in the algorithm?

1, 2, 3, 4
1, 2, 4, 8
1, 10, 100, 1000

It is hoped that experimentation will suggest an answer to this question. The priority file could also be of use to the scheduling engineer using the algorithm. If a certain target or type of target failed to schedule, then increasing the priority value would cause the algorithm to "try harder" to schedule the target(s).

The Time Periods File consists of an increasing sequence of times:

$$T_0 < T_1 < T_2 < \ldots < T_m.$$

The time To corresponds to the time in the mission when the telescope viewing commences, and T_m corresponds to the time when viewing ends. The intermediate times are obtained in one of two ways; consisting of all the ON times in the Shadow File and OFF times in the SAA File. In actuality, there is no Time Periods File since the program has been written to read the relevant information off of the existing Shadow and SAA files. The user must specify To and T_m, however, since we are only interested in the times in these two files which fall between T_0 and T_m. By a time period we mean $[T\ell-1, T\ell]$, $\ell=1,\ldots,m$. Hence each time period begins at the start of a shadow portion of an orbit or at the end of a SAA portion of an orbit. Due to the nature of the program it will also be necessary to insert a time into the Time Periods File corresponding to the end of any time interval where viewing is prohibited. This allows the program to "start over" when viewing is again permissible.

The Observation Opportunity File lists the times during which targets may be viewed. The actual format is somewhat different in actuality, but for purposes of simplicity in this discussion we will assume that for each (target) i=1,...,178 we have a sequence

$$T_0 \leq t_{1i} < t_{2i} < t_{3i} < \ldots < t_{2n_i-1\ i} < t_{2n_i\ i} \leq T_m$$

where

$$[t_{1i}, t_{2i}], [t_{3i}, t_{4i}], \ldots, [t_{2n_i-1\ i}, t_{2n_i\ i}]$$

are the n_i time intervals during which target i may be observed. This file is computed using other programs available at the MSFC, and takes into consideration geometric availability (whether the target is occulted by the Earth or Moon) and other constraints involving the magnitude of the target, sensitivity of the instrument, and amount of direct or reflected sunlight present. The times

$$t_{1i}, t_{3i}, t_{5i}, \ldots, t_{2n_i-1\ i}$$

are referred to as access times (acc-times) for target i, and the times

$$t_{2i}, t_{4i}, t_{6i}, \ldots, t_{2n_i\, i}$$

are referred to as loss times for target i.

The Weight File is a list of 54 4-tuples, $(\alpha, \beta, \delta, \gamma,)$ where the values of the entries are:

α = 0, .5, 1

β = 0, 5, 10

δ = .01, .1

γ = 0, 1, 10

These are listed in lexicographic order from (0, 0, .01, 0) to (1, 10, .1, 10) in the Weight File as the rows of a 54-by-4 matrix. Each weight produces a schedule for any given time period which gives the program 54 trial schedules to choose from for any time period.

The Slew Time File is regarded as a 178-by-178 matrix

$$D = [d_{ij}]$$

where d_{ij} is the time required to slew from target i to target j. If the slew rate is a constant, then d_{ij} directly proportional to the angular distance between targets i and j, which is available on another file. In our initial program we are assuming a slew rate of .4 degrees/sec., so if a_{ij} is the angular distance (in degrees) between targets i and j, then

$$d_{ij} = \frac{a_{ij}}{1440} \quad \text{(in decimal hours)}.$$

Since D is a symmetric matirx with zero diagonal, the actual file only lists d_{ij} for i<j. In this paper we will refer to d_{ij} without restricting ourselves to i<j.

III. FUNCTIONS

The algorithm requires computation of certain quantities in the process of selecting a next target at any time t. In this section we describe the following five functions related in this idea: Local Availability, Total Availability, Initial Selecter, Successive Selecter, and Reselecter. In addition we define an Objective function which evaluates a schedule over a time period.

The Local Availability function is denoted by A(t,i), and is computed in the following manner:

1. If $t \leq t_{2n_i\, i}$, then $A(t,i)=0$
2. If $t < t_{1i}$, then $A(t,i)=0$
3. If $t_{si} \leq t < t_{s+1\, i}$, s even, then $A(t,i)=0$
4. If $t_{si} \leq t < t_{s+1\, i}$, s odd, then

$$A(t,i)=t_{s+1\, i} - t.$$

This function is 0 unless that t is a time at which target i can be observed. If t is a time at which target i can be observed, then A(t,i) is the amount of time target i can be observed before the observation opportunity ends.

The Total Availability function is denoted by B(t,i) and is the amount of time left in the mission at time t that target i may be observed. It can be computed as follows.

1. If $t \leq t_{2n_i\, i}$, then $B(t,i)=0$.
2. If $t < t_{1i}$, then

$$B(t,i) = \sum_{\ell=1}^{n_i} t_{2\ell i} - t_{2\ell i-1\, i}.$$

3. If $t_{si} \leq t < t_{s+1\, i}$, then

$$B(t,i)=A(t,i) + \sum_{\ell=\left[\frac{s+3}{2}\right]}^{n_i} t_{2\ell\, i} - t_{2\ell-1\, i}.$$

In step 3, the symbol [ ] refers to the greatest integer or step function defined by

$$[x] = \text{the largest integer } n \text{ satisfying } n \leq x.$$

In step 3 the computation of B(t,i) involves the quantity A(t,i). This presents no practical difficulty since the algorithm only requires the computation of B(t,i) after A(t,i) has been computed and found to be positive. Hence it is unnecessary to activate the subroutine for A(t,i) in step 3 since it has already been computed.

In fact, it seems natural to compute A(t,i) and B(t,i) in a single sequence of steps in order to simplify the programming.

The Initial Selecter function is used to choose a target to be the first viewed in a time period. It is denoted by F(t,i), and calculated as follows:

1. If $r_i=0$, then $F(t,i)=0$
2. Compute $A(t,i)$
3. If $A(t,i)<\delta_i$, then $F(t,i)=0$
4. If $A(t,i)\geq\delta_i$, then

$$F(t,i) = \frac{p_i[\alpha+A(t,i)]\ [\gamma+\delta_i r_i]}{\beta+B(t,i).}$$

In other words, A(t,i)=0 unless there is sufficient local availability at time t to conduct at least one observance of target i. In this case, F(t,i) is given by the formula in step 4.

The Successive Selecter function is used to select a next target (j) to follow target i which has been scheduled up until time t. It is denoted by G(t,i,j) and calculated as follows.

1. If $r_j=0$, $G(t,i,j)=0$.
2. Evaluate $A(t+d_{ij},j)$.
3. If $A(t+d_{ij},j)<\delta_j$, then $G(t,i,j)=0$.
4. Evaluate $B(t+d_{ij},j)$.
5. $$G(t,i,j)= \frac{p_i[\alpha+A(t+d_{ij},j]\cdot[\gamma+\delta_i r_i]}{[\beta+B(t+d_{ij},j)]\cdot[\delta+d_{ij}].}$$

Step 1 sets G(t,i,j)=0 if we have already satisfied the requirement of target j.

Step 3 sets G(t,i,j)=0 if there is not sufficient local availability to conduct at least one observance of target j at time t+dij, which is the first time at which viewing of target j can be initiated after viewing target i until time t. If steps 1 and 3 do not establish the value of G(t,i,j), then it is defined by the formula in step 5.

The Reselecter function is denoted by H(t,i,j,k) and is used when target j has been selected to follow target i which finished viewing at time t. In the algorithm, we check to see if target k might be a better choice to follow target i than target j. This could happen if k is not available at time $t+d_{ik}$ but is available at time $t+d_{ij}$.

In such a situation the schedule might be improved by the insertion of idle time. The function is calculated as follows.

1. If $r_k=0$, then $H(t,i,j,k)=0$.
2. Evaluate $A(t+d_{ij},k)$.
3. If $A(t+d_{ij},k)<\delta_k$, then $H(t,i,j,k)=0$.
4. Evaluate $B(t+d_{ij},k)$.
5. $$H(t,i,j,k)=\frac{p_k[\alpha+A(t+d_{ij},k)][\gamma+\delta_k r_k]}{[\beta+B(t+d_{ij},k)][\delta+d_{ij}]}.$$

The careful reader may note the similarities between the functions F, G, and H. In fact, both F and G can be computed by the same subroutine used to calculate H. The relations are as follows.

$$F(t,i) = H(t,i,i,i)$$

$$G(t,i,j) = H(t,i,j,j).$$

For programming purposes the above substitutions would reduce program length and number of subroutines. For purposes of explaining the algorithm we will stick with the various selecters F,G, and H, rather than having a single function (H).

The previous five functions form the basis of the scheduling routine which establishes a schedule for a time period $[T_{k-1},T_k]$. Since the weight $(\alpha,\beta,\delta,\gamma)$ appears in these functions, we obtain one schedule for $[T_{k-1},T_k]$ for each of the 54 weights in the weight file. Suppose we denote the schedule obtained from a particular weight by $S(\alpha,\beta,\delta,\gamma)$ and let us further assume that the schedule is presented as follows.

$$\begin{aligned}&(i_1,\ t'_0,\ t_1)\\ &(i_2,\ t'_1,\ t_2)\\ &(i_3,\ t'_2,\ t_3)\ = S(\alpha,\beta,\delta,\gamma).\\ &\quad\vdots\\ &(i_s,\ t'_{s-1},t_s)\end{aligned}$$

Here it is assumed target i_1 is scheduled from time t'_0 to time t_1, target i_2 is scheduled from time t'_1 to time t_2, etc.. We define an Objective function as follows.

$$O(S) = O(\alpha,\beta,\delta,\gamma)\sum_{\ell=1}^{s} p_{i_\ell}(t_\ell - t'_{\ell-1}).$$

In the algorithm we will choose a schedule $S=S(\alpha,\beta,\delta,\gamma)$ for the time period which corresponds to a maximal value of $O(\alpha, \beta, \delta, \gamma)$. The value of $O(\alpha, \beta, \delta, \gamma))$ (sometimes referred to as the "grade" of $S(\alpha, \beta, \delta, \gamma)$) is simply priority-weighted viewing time, or simply viewing time if $p_i=1$, all $i=1,\ldots,178$.

In this section we identify and describe certain portions of the main algorithm. There are several reasons for doing this. For one, it is easier to understand the main algorithm in a modular fashion than all at once. For another, it is convenient to have the main algorithm broken into subroutines for reprogramming purposes. For example, an improvement could be made by rewriting one of the subroutines rather than rewriting the whole program. Also, subroutines can be tested and debugged individually if problems occur.

The first subroutine is referred to as the Initial Selection Algorithm, or INSLA for short. The input to this subroutine is a time t_o, and the output is (t_o, i_1) where target i_1 has been chosen as the "best" target to schedule at time t_o, or the message STOP. The procedure is as follows.

1. Input t_o.
2. Evaluate $F(t_o, i)$, all $i = 1,\ldots,n$.
3. If $F(t_o, i) = 0$, all $i = 1,\ldots,n$, then STOP.
4. Choose i_1 where $F(t_o, i_1) \geq F(t_o, i)$, all i.
5. Output (t_o, i_1)

In practice we would evaluate $F(t_o, i)$ for $i = 1,\ldots, n$, with instructions to save i_{max} and $F_{max} = F(t_o, i_{max})$. The only time the output is STOP is when there is no target to view at time t_o that is available at time t_o and has a requirement left.

The Experiment Scheduler Algorithm, or EXSA, has as its input (t_o, i_o, i_1) where target i_o has been viewed until time t_o and target i_1, is the next target to view. The EXSA schedules as many consecutive observances of target i_1, as possible as soon as possible. The output is the message (t_1, i_1) which signifies that target i_1, has been scheduled (consecutively) until time t_1. The algorithm is described as follows.

1. Input (t_o, i_o, i_1).
2. Evaluate $A(t_o + d_{i_o i_1}, i_1)$.
3. If $A(t_o + d_{i_o i_1}, i_1) \geq s_{i_1}$, let $t_o' = t_o + d_{i_o i_1}$.
4. If $A(t_o + d_{i_o i_1}, i_1) < s_{i_1}$, let t_o' = the first $t_{\ell i_1}$ where ℓ is odd and $t_o + d_{i_o i_1} < t_{\ell i_1}$.
5. If $t_o' + s_{i_1} > T$, let $t_1 = t_o$.
6. If $t_o' + s_{i_1} \leq T$, schedule one observance of target i_1 from t_o' to $t_o' + s_{i_1}$ and let $r_{i_1} = r_{i_1} - 1$.
7. If $r_{i_1} = 0$, let $t_1 = t_o' + s_{i_1}$.
8. If $r_{i_1} > 0$, go back to 1, replacing t_o with $t_1 = t_o' + s_{i_1}$ and i_o by i_1. (i.e., replace (t_o, i_o, i_1) with (t_1, i_1, i_1)).

The Selecter Algorithm, or SELAG, is used to select a next experiment in the middle of a time period. The input to this subroutine is (t_1, i_1), which represents the fact that target i_1

has been veiwed until time t_1 and a new target is to be chosen. The output is either (t_1, i_1, i_2), which signifies that i_2 has been chosen to be scheduled next, or STOP. The STOP output can occur in the following way. If there is no experiment i_2 which is possible to schedule at time $t_1 + d_{i_1 i_2}$, then STOP occurs. This would happen in a SAA time interval, for example. The Selecter Algorithm is composed of two iterations: one each corresponding to the successive selecter and reselecter functions. It is possible to view the SELAG subroutine as two subroutines, but in this account we will describe the process as one subroutine. The description is as follows.

1. Input (t_1, i_1).
2. Evaluate $G(t_1, i_1, i)$, all $i \neq i_1$.
3. If $G(t_1, i_1, i) = 0$, all $i \neq i_1$, then STOP.
4. If $G(t_1, i_1, i)$ is not identically zero, then select i_2' which maximizes $G(t_1, i_1, i)$.
5. Evaluate $H(t_1, i_1, i_2', i)$, all $i \neq i_1$.
6. Select i_2 which maximizes $H(t_1, i_1, i_2', i)$, and output (t_1, i_1, i_2).

Steps 1-4 make an initial selection for a next target and steps 5-6 examine whether a different target might be a better choice if idle time is inserted. If $i_2 = i_2'$ (i.e., there is no better choice), then this corresponds to the situation $t_o' = t_o + d_{i_o i_1}$ in EXSA. If $i_2 \neq i_2'$ and idle time is inserted, this corresponds to the definition of t_o' in step 4 of EXSA.

The Descheduler Algorithm, or DESAG is used to fit partial scheduler together and requires some explanation of the main algorithm, which can be obtained in the next section. Briefly, suppose that a partial schedule, say S_k, has been obtained over the first k time periods, (T_o, T_k). The algorithm next produces a partial schedule for the (k+I) st time period, (T_k, T_{k+1}). Call this partial schedule S_{k+1}. The Descheduler is designed to combine these two partial schedules into a schedule for (T_o, T_{k+1}) which we will refer to as S_{k+1}. Suppose that the last target in S_k is i_o which finishes viewing at time $t_o \leq T_k$, and the first target in S_{k+1} is L_1, which begins veiwing at time T_k. If

$$T_k - t_o \geq d_{i_o i_1},$$

then there is sufficient time to slew the telescope from target i_o to target i_1, and we combine the two schedules. If

$$T_k - t_o < d_{i_o i_1},$$

then the Descheduler deletes the last observance of i_o until sufficient slew time is available. When DESAG deletes an observance of i_o it is necessary to let

$$r_{i_o} = r_{i_o} + 1.$$

The input to DESAG is S_k, S_{k+1}, and the output is S_{k+1}.

VI. MAIN ALGORITHM

In this section we define the principal subroutine as well as the main algorithm.

The principal subroutine is the Scheduler Algorithm, or SCHALG, and is designed to produce a schedule, for a time period (T_{k-1}, T_k). If we let (t_o, T) play the role of (T_{k-1}, T_k), then SCHALG is described as follows.

1. Input (t_o, T), $(\alpha, \beta, \delta, \gamma)$.
2. Send t_o to INSLA, T to EXSA.
3. If INSLA yields STOP, then $S = \emptyset$.
4. If INSLA yields (t_o, i_1), then send (t_o, i_1, i_1) to EXSA and obtain (t_1, i_1).
5. Send (t_1, i_1) to SELAG and obtain (t_1, i_1, i_2) or STOP.
6. Send (t_1, i_1, i_2) to EXSA and obtain (t_2, i_2).
7. If $t_2 = t_1$, then STOP.
8. If $t_2 > t_1$, go back to 5, letting (t_2, i_2) play the role of (t_1, i_1).

Some comments about step 7 are in order. This step is used when target i, has been viewed until t_1, i_2 is chosen next by SELAG, and EXSA finds it impossible to schedule i_2 within the current time period. A moments reflection yields that this may occur only when the local availability for i_2 extends past the end of the current time period. This cannot occur if the observation opportunities are all contained within a time period, as was indicated would be the case by the NASA planning personnel assigned to this project. Hence, if everything is in order step 7 will not be used. If the Observation Opportunity Files do not obey this constraint with respect to the Time Periods File, then step 7 will prevent the program from falling into a loop where it repeatedly chooses target i_2 and then is unable to schedule an observation of it. If no such difficulties occur, then the SCHALG will exit on step 5.

The Main Algorithm, or MAINALG, produces a schedule for the time interval (T_o, T_m). The method is to generate 54 schedules for the first time period, (T_o, T_1), then pick the best one using the objective function and proceed to the next time period (T_1, T_2). The DESAG then fits these two schedules together to form a schedule for (T_o, T_2). Proceeding in a similar fashion, we obtain a schedule for (T_o, T_m). Before we proceed, we explain the following notation concerning partial schedules.

S_ℓ or S_ℓ $(\alpha, \beta, \delta, \gamma)$ denotes a schedule for $(T_{\ell-1}, T_\ell)$,

S_ℓ denotes a schedule for (T_o, T_ℓ),

$S = S_m$ is the schedule for (T_o, T_m).

The MAINALG is described as follows.

1. Let the time period (t_o, T) be $(T_{\ell-1}, T_\ell)$, $\ell = 1$.
2. Let the weight $(\alpha, \beta, \delta, \gamma)$ be $(\omega_{i1}, \omega_{i2}, \omega_{i3}, \omega_{i4})$, $= 1$.
3. Send (t_o, T) and $(\alpha, \beta, \delta, \gamma)$ to SCHALG and obtain a schedule S_ℓ $(\alpha, \beta, \delta, \gamma)$.

4. Evaluate 0 (S_ℓ (α, β, δ, γ)) over (t_o, T).
5. Go back to 2 and increase i by 1 until i = 54.
6. Choose S_ℓ from among the 54 trial schedules for (t_o, T) which maximizes 0 (S_ℓ (α, β, δ, γ)).
7. Use the DESAG to adjoin S_ℓ to $S_{\ell-1}$ to form S_ℓ.
8. Go back to step 1 and increase until ℓ = m.

A comment on observation requirements is in order. Since we are examining 54 trial schedules for each time period but only using the best one it is important to not write the program in such a way that it thinks all 54 schedules were performed. In other words, the requirement r_i should be decremented during each of the 54 trail schedules but revert to what it was at the start of the time period when the next trial schedule is to be computed. Then, of course, when the best schedule for the time period is obtained and adjoined to the previous partial schedule by DESAG, then the r_is are diminished accordingly.

The algorithm described results in a scheduling sequence, say S, which has been constructed from the subschedules for the m time periods. This needs to be presented in some fashion. One method of presenting S would be as follows.

(i_1, t'_0, t_1)
$(i_2\ t'_1, t_2)$
$(i_3\ t'_2, t_3)$
.
.
.
(i_u, t'_{u-1}, t_u)

In this scheme target i_1 is scheduled between times t'_0 and t_1, target i_2 is scheduled between times t'_1 and t_2, etcetera. The algorithm produces a schedule which automatically satisfies the two important constraints:

$(t'_{\ell-1}, t_\ell)$ is contained in an i_ℓ-window for all $\ell = 1, \ldots, u$.

and

$t'_{\ell-1} - t_{\ell-1} \geq d_{i_{\ell-1} i_\ell}$ all $\ell = 2, \ldots, u$.

It might also be useful to indicate the number of observances as well as the viewing time.

To evaluate the schedule we need to have an objective function to give a grade to the schedule. We define this as follows:

$$\text{Grade} = O(S) = \sum_{\ell=1}^{u} P_{i_\ell} (t_\ell - t'_{\ell-1}).$$

This grade is simply priority-weighted viewing time.

We also wish to have the viewing times displayed. For each $i = 1, \ldots, n$, we define

$$V(i) = \sum_{i=i_\ell} t_\ell - t'_{\ell-1}.$$

These quantities we would not wish to display, but it would be advantageous to be able to call them up if desired. The following six quantities should be displayed along with the schedule.

$$V(100) = \sum_{i=101}^{148} V(i),$$

$$V(200) = \sum_{i=201}^{213} V(i),$$

$$V(300) = \sum_{i=301}^{335} V(i),$$

$$V(400) = \sum_{i=401}^{445} V(i),$$

$$V(500) = \sum_{i=501}^{537} V(i),$$

$$V = V(100) + V(200) + V(300) + V(400) + V(500).$$

Note here that we are referring to the alternate numbering scheme and not to i = 1, ..., 178.

Another desired output is the weight sequence which produced each of the m best subschedules for the time periods. This will show the user which weights were more effective in generating schedules, and will lead to beneficial changes in the weight file. Eventuallythe number of entries in the weight file should be much less than 54. If a weight is never used it would be dropped from the weight file, where one which appeared many times would be bracketed closely by other weights. Finally, an effective weight file will result. This output could be presented as an m-by-4 or 4-by-m matrix. It is worth pointing out that the overall schedule S is not a function of one particular weight as the subschedule for a time period is. Rather, the overall schedule is a function of the whole weight file rather than one particular weight. As a preliminary guess it seems plausible that each of the parameters α, β, δ, γ could have two specified values. This would have the effect of cutting the running time of the program by two-thirds by reducing the number of trial schedules from 54 to 16.

VII. CONCLUSIONS AND RECOMMENDATIONS

In this section we discuss possible pitfalls in the program as well as suggestions for beneficial modifications. It was hoped that these matters may have been investigated after the algorithm was functioning on the VAX computer. This, however, proved to be impossible to achieve within a ten week period. A programmer has been working on putting the algorithm in the computer, but it was not functioning at the time this report was written. The single attempt to run the program to date was less than auspicious. The machine ran for nearly two hours without scheduling one target observance and then stopped due to programming difficulties. An examination of some of the quantities involved revealed, for example, that A (t, i) and B (t, i) were computed to be the same quantity, which was erroneous. Initially it was planned that the method could be tested, then improved for two to three weeks. Hopefully the following comments may be of assistance to anyone attempting to use the methodology presented in this report, or any part thereof.

One possible source of difficulty lies in the SELAG. In step 3 of SELAG, the algorithm stops if there is no target available at the current time. The MAINALG then proceeds to the start of the next time period. Hence, if there was a "blank spot" in the middle of a time period where no target was available, the whole remaining portion of the time period would remain unscheduled. This would become more likely towards the end of the mission when more of the targets have their requirements satisfied. There are various ways around this problem if difficulty occurs in practice. For one thing, if there are many more targets listed than can possibly be observed, then this problem is unlikely to occur since the SELAG can always find a suitable target. Another method would be to include in the files a bogus target labelled "wait" which would be scheduled for one minute if nothing else was available. The "wait" target would of course have to have variable slew times for effective implementation. The comments about step 3 of SELAG apply equally well to step 3 of INSLA.

The DESAG might also cause difficulties. It is conceivable that the DESAG could unschedule a very long observation, thereby ruining the schedule for the previous time period. This would be as a result of the algorithm insisting on starting the initial observance in a time period exactly at the start of the period rather than adding any necessary slew time. This feature was included at the suggestion of the project engineers who indicated it was preferable to do all possible slewing during the Sun portions of the orbits, thereby saving the valuable Shade portions for viewing. There are ways around this problem if the difficulty occur in actual use. For one, the DESAG could be eliminated. Another approach would be for a secondary program that could be used after this program that would fill in any gaps. For example,

any gaps could be fed to SCHALG after the program had been run. A minor modification might be inserted specifying where the telescope was going to end up at the termination of a gap. Such a "scheduling over" method could be applied to the problem previously discussed regarding SELAG and INSLA.

A feature which has been suggested is a variable requested-time capability. It has been remarked that when a principal investigator specifies a value of si, say for example 30 minutes, that what is actually desired is at least 30 minutes, but as much more as can be scheduled consecutively. If that turns out to be a common occurrence, it would be advantageous to have the capability of entering a value of 30+ rather than 30 for si.

Another area of investigation we will label as "grouping". The telescope mount has a capability of gimballing some 17 degrees. It is preferable to use the gimbal mechanism whenever possible to re-aim the telescope since it slews much faster, and it is more efficient to utilize the telescope mount apparatus than to slew by maneuvering the entire Orbiter. To take advantage of this capability it would be first necessary to identify groups of experiments which are in the same portion of the sky (i.e., within 17 degrees of a central point), and all of which can be viewed during some specified time interval. These groups could be identified by the use of some auxiliary software which would examine the Observation Opportunity File and the Angular Distance File. Once these intervals and groups are identified they can be fed as time periods to SCHALG. It would also be necessary to make an adjustment in the Slew Time File to compensate for the faster slew rate of the telescope mount. The identified groups would then be scheduled, using SCHALG, and then the remaining time periods could be scheduled.