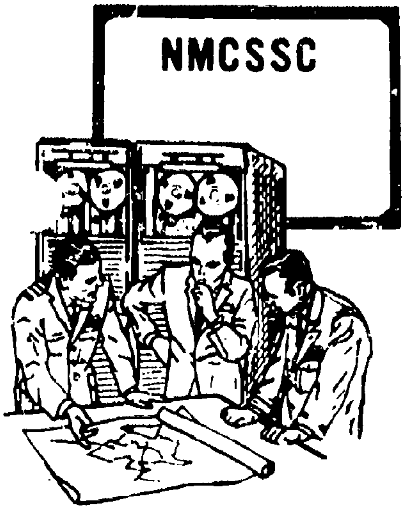
AD 742782

NATIONAL MILITARY COMMAND SYSTEM SUPPORT CENTER



COMPUTER SYSTEM MANUAL
CSM AM 9A-67
VOLUME II
29 FEBRUARY 1972

THE NMCSSC QUICK-REACTING GENERAL WAR GAMING SYSTEM (QUICK)

PLAN GENERATION SUBSYSTEM

Vol. I - AD740763

ANALYTICAL MANUAL

DEFENSE COMMUNICATIONS AGENCY

305

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) National Military Command System Support Center (NMCSSC) Defense Communications Agency (DCA) The Pentagon Washington, DC 20301	2a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED 2b. GROUP

3. REPORT TITLE
The NMCSSC Quick-Reacting General War Gaming System (QUICK) Analytical Manual, Volume II, Plan Generation Subsystem

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)
N/A

5. AUTHOR(S) (First name, middle initial, last name)

NMCSSC: Robert R. Hardiman, Cathy L. Sacassas, Donald F. Webb
Lambda Corp: Paul D. Flanagan, Patricia M. Parish, Jack A. Sasseen

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
29 February 1972	307	7b C

8a. CONTRACT OR GRANT NO. DCA 100-70-C-0065	9a. ORIGINATOR'S REPORT NUMBER(S) NMCSSC COMPUTER SYSTEM MANUAL CSM AM 9A-67
b. PROJECT NO. NMCSSC Project 631	
c.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report) None
d.	

10. DISTRIBUTION STATEMENT
This document is approved for public release; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	National Military Command System Support Center/Defense Communications Agency The Pentagon, Washington, DC 20301

13. ABSTRACT

This is one of three volumes describing the analytical techniques used in the Quick-Reacting General War Gaming System (QUICK). This volume describes the Plan Generation Subsystem. Initially, the general concept of operation and the functions performed by this subsystem are presented. Subsequently, the detailed analytical concepts and algorithms are discussed.

Based upon a suitable data base, and user control parameters, QUICK will generate individual bomber and missile plans suitable for war gaming. The generated plans are of a form suitable for independent review and revision. Subsequently, execution of the planned events can be simulated. Various statistical summaries can be produced to reflect the results of the war game. A variety of force postures and strategies can be accommodated.

QUICK is documented extensively in a set of Computer System Manuals (series 9-67) published by the National Military Command System Support Center (NMCSSC), Defense Communications Agency (DCA), The Pentagon, Washington, DC 20301.

DD FORM 1 NOV 65 1473 REPLACES DD FORM 1473, 1 JAN 64, WHICH IS OBSOLETE FOR ARMY USE.

UNCLASSIFIED
Security Classification

NATIONAL MILITARY COMMAND SYSTEM SUPPORT CENTER

Computer System Manual Number CSM AM 9A-67

29 February 1972

THE NMCSSC QUICK-REACTING GENERAL WAR GAMING SYSTEM (QUICK)

Analytical Manual

Volume II - Plan Generation Subsystem

Submitted by:

DONALD F. WEBB
Major, USAF
Project Officer

REVIEWED BY:

R. E. HARSHBARGER
Technical Director
NMCSSC

APPROVED BY:

BRUCE MERRITT
Colonel, USA
Commander, NMCSSC

ACKNOWLEDGMENT

This document was prepared under the direction of the Chief for Development and Analysis, NMCSSC, in response to a requirement of the Studies, Analysis and Gaming Agency (SAGA), Organization of the Joint Chiefs of Staff. Technical support was provided by Lambda Corporation under Contract Number DCA 100-70-C-0065.

CONTENTS

Chapter		Page

Chapter Page

ILLUSTRATIONS

TABLES

ABSTRACT

QUICK is a two-sided nuclear exchange war gaming system. It is designed to assist the military planner in examining various facets of strategic nuclear war involving a variety of forces, strategies, and starting conditions. Based on suitable input data, QUICK will automatically generate global strategic nuclear war plans, simulate the planned events, and provide statistical output summaries.

This document is one of three volumes of the Analytical Manual which provides a description of the QUICK system methodology for the non-programmer analysts. This volume describes the QUICK Plan Generation subsystem. The general concept of operation and the functions performed by this subsystem are presented in the introductory chapter. Subsequent chapters provide a detailed explanation of the analytical concepts, techniques, and algorithms employed in plan generation. In addition, applicable accuracy considerations are described in the final chapter.

The following is a list of associated documents pertaining to the QUICK system.

GENERAL DESCRIPTION
Computer System Manual CSM GD 9A-67
A nontechnical description for senior management personnel

PROGRAMMING SPECIFICATIONS MANUAL
Computer System Manual CSM PSM 9A-67 (three volumes)
Detailed information required for system maintenance and modification

USER'S MANUAL
Computer System Manual CSM UM 9-67 (two volumes)
Detailed instructions for applications of the system

OPERATOR'S MANUAL
Computer System Manual CSM OM 9A-67
Instructions and procedures for the computer operators

CHAPTER 1
INTRODUCTION

This second volume of the Analytical Manual describes the QUICK Plan Generation subsystem, hereafter referred to as the Plan Generator. The Plan Generator uses information from the Data Input subsystem of QUICK to develop a global nuclear war plan suitable for manual interpretation or input to the Simulation subsystem. A single pass through the Plan Generator produces a plan for one side only. If plans for both sides are required, two runs must be made. If such plans are intended to be used together in the QUICK Simulator, the plans must be based on a common data base; otherwise, indexing incompatibilities may occur.

The Plan Generator operates using the target system and weapon resources supplied to it from the indexed data base INDEXDB prepared by program INDEXER of the Data Input subsystem. Table 1 provides an example of the type of data maintained in the data base. Table 2 indicates targetable-type installations currently included in QUICK's 15 target classes.

The Plan Generator does not make judgments about the appropriateness of either the target system or the resources specified. It accepts given inputs and produces a plan using the weapon resources specified to maximize the expected target value destroyed (subject to any requirements for specific kill probabilities on specified targets).

Table 1. Typical Data Base Elements Included in QUICK

OFFENSIVE WEAPONS
- Types
  - Strategic Bombers
  - Strategic Missiles
  - Tactical Nuclear Bombers
  - Tactical Nuclear Missiles
- Numbers
- Characteristics
  - Numbers and Yield of Warheads
  - Accuracy
  - Reliability
  - Range
  - Speed
  - Electronic Countermeasures (ECM)

DEFENSIVE WEAPONS
- Types
  - Manned Interceptors
  - Surface-to-Air Missiles
  - Antiballistic Missile (ABM) Systems

TARGETS
- Types
  - Offensive Weapon Launch Bases
  - Defensive Weapon Bases
  - Command and Control Sites
  - Early Warning Stations
  - Military Support Installations
  - Urban/Industrial Complex
- Characteristics
  - Geographic Location
  - Vulnerability
  - Value

Table 2. QUICK Classes

CLASS	DATA CATEGORY
1	Offensive missiles
2	Offensive bombers
3	Tankers
4	Defensive command and control
5	Interceptor aircraft
6	Offensive command and control
7	Nuclear storage sites
8	Airfields
9	Naval targets
10	Troops
11	Communications
12	Miscellaneous (e.g., engineer facilities)
13	Urban/industrial targets
14	Area ABM defense components
15	Reserved for future use

The Plan Generator can be used to serve two distinct purposes. It can generate war plans for one or both sides which can be fed directly into the QUICK Simulator for detailed evaulation; or, the Plan Generator alone can be used to produce a one-sided expected-value war game.

The remainder of this chapter first presents a brief summary of the QUICK system as a whole. It then discusses the methodology of the Plan Generation subsystem and presents the procedural and information flow through the subsystem. Chapter 2 presents an in-depth description of the analytical techniques employed within this subsystem. Chapter 3 provides a detailed mathematical explanation of the more complex and sophisticated algorithms included in the Plan Generator. In addition, comments relevant to the accuracy considerations are included in chapter 4.

QUICK SYSTEM

The following describes the general concept of operation for the QUICK system and establishes the relationship of the Plan Generator to the other major subsystems.

Figure 1 illustrates the processing sequence and information flow within the QUICK system. The procedural flow is shown by solid lines and the information flow by dashed lines. As indicated, magnetic computer tapes are utilized to pass information between the four subsystems.

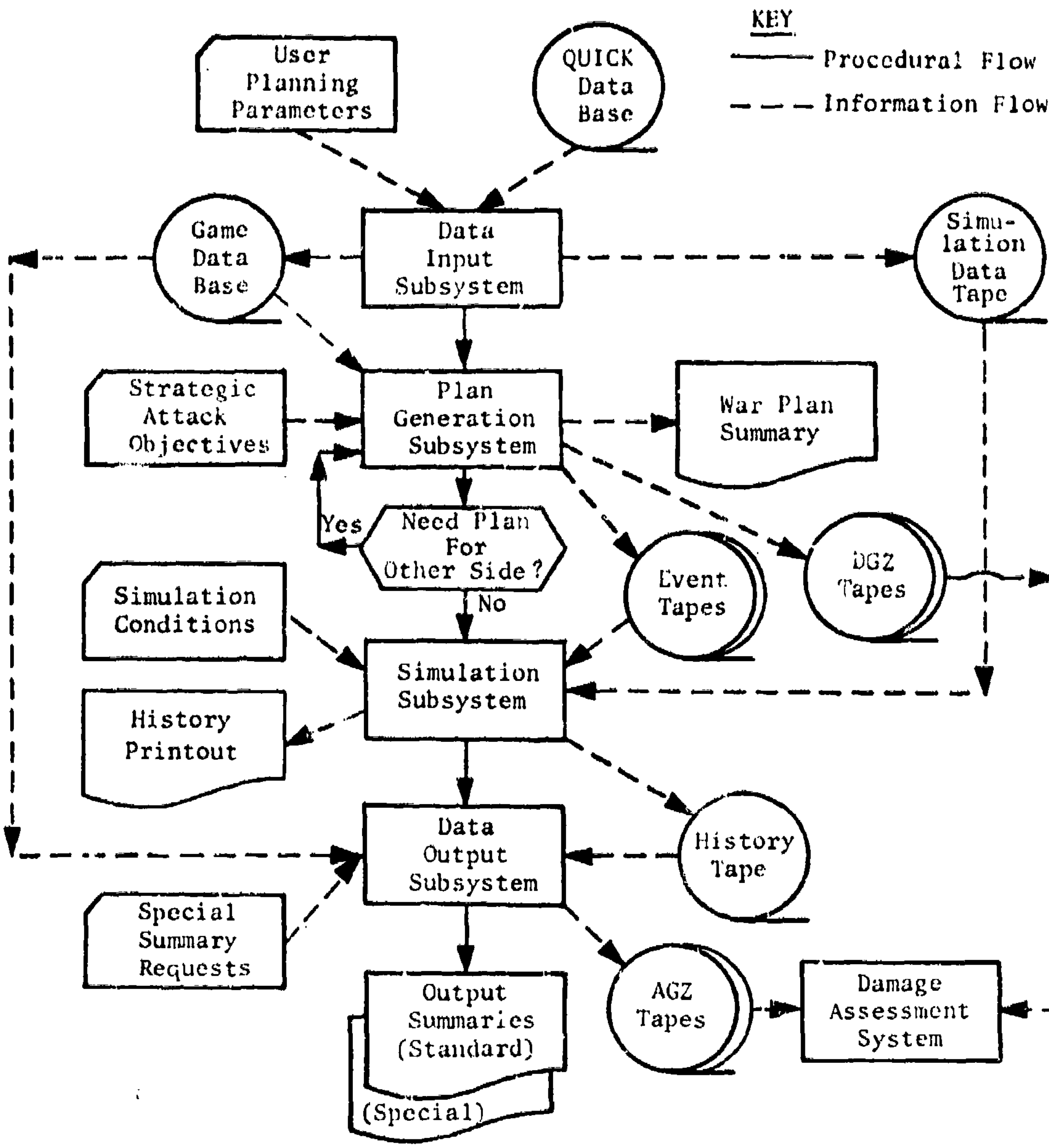


Fig. 1. Procedure and Information Flow in QUICK

Processing is initiated by inputting the parameters which identify the Red and Blue forces and the potential targets which are to be extracted from the QUICK data base. In addition, any desired data base modifications are specified. The Data Input subsystem then processes the QUICK data base and prepares a game data base which reflects the selected forces and targets.

The next step is to prepare an attack plan for one of the opposing forces. Since a single run of the Plan Generator produces a plan for only one side, the Plan Generator must be cycled twice to produce the Red and Blue plans. Two major inputs are required to initiate this phase of processing: (1) the game data base prepared by the Data Input subsystem; and (2) a set of parameters which relate to the strategy associated with the plan which is to be developed. These parameters are supplied by the planner. They reflect his views as to the strategic attack objective, in terms of the relative values of the various targets being considered, the forces to be withheld, the targeting constraints to be observed, and the side which attacks first.

The target values which are computed on the basis of these parameters reflect in a very significant way the major strategic objectives of the war plan which is to be generated by the Plan Generator. These values are relative values and are partially contained in the data base itself. There are several specific classes of targets in QUICK, as shown in table 2. The relative value of the targets contained in any one class is generally included in the data base, and then the strategic objectives of the planner who wants to run the Plan Generator are expressed in how the value scales of these various classes of targets are related to one another. The user has the option of putting more or less relative importance on any one of the classes of targets in accomplishing the desired strategic objectives. This, of course, will be related to the kind of strategy contemplated for the particular war game; e.g., first or second strike.

Having established a value for each target, the Plan Generator then allocates the weapons (e.g., Red weapons to Blue targets) and prepares the detailed missile and bomber attack plans. If desired, the plans may be printed out, inspected, and altered by changing the attack objectives and repeating the process. The event tape, which reflects the series of missile and bomber events corresponding to the sortie plan, is prepared in a form suitable for input to the Simulator. As a user option, a war plan summary is provided which includes an expected-value estimate of the results of the attack. In addition, the desired ground zero (DGZ) for each planned weapon can be output for subsequent evaluation using an external damage assessment system. A second (e.g., Blue) war plan is then prepared in the same manner as the first war plan. With the two event tapes available, the system is ready to proceed with the simulation.

The simulation conditions, specifying the starting time for each side and various defense capabilities, are read in from cards. The scheduled missions on the event tapes are then processed in the Simulator. For each event that transpires, a record is made on the History tape of all information that might later prove of interest.

When the last event in the game has been simulated, this History tape is processed to prepare the actual ground zero (AGZ) tapes listing the latitude, longitude, and yield of all successful weapons, and formatted History tapes which are in a form suitable for game output summarization. The AGZ tapes are subsequently processed by an external system to produce detailed damage assessments. The formatted History tapes are processed by the QUICK Data Output subsystem to provide two outputs: a standard summary of the game, and special summaries prepared in response to specific user request for information concerning the results of the simulation.

The system can proceed automatically through all steps if desired. However, it is generally halted at the end of each subsystem, and the available output is inspected for correctness and adequacy.

CONCEPT OF OPERATION

Procedural Flow

The Plan Generator accepts as input the indexed data base tape INDEXDB prepared by the Data Input subsystem and proceeds by a series of steps to produce a detail plan for general nuclear war. This plan is prepared in the form of the EVENTAPE for use by the Simulation subsystem of QUICK, and as a PLANTAPE which is used to prepare inputs for other subsystems. Two complete runs of the Plan Generator (one Red, one Blue) from the same indexed data base file are required to provide the plans required for the operation of the Simulator.

The flow of information within the Plan Generator is summarized in the succeeding section, "Information Flow." The series of steps performed by the Plan Generator is as follows.

Weapon and Target Selection: The first step of the Plan Generator is to select from the input file the weapons from one side and the targets from the other side, as specified by the user. The weapons are selected by type; e.g., B-52H. The various target classes are assigned relative values (see chapter 2, Target Value), reflecting the user's ideas of strategic priorities.

Formation of Weapon Groups: Weapons of the same type and alert status and in geographical proximity are grouped together (see chapter 2, Weapon Grouping) so that they may be initially treated as identical for purposes of arriving at a general allocation. Thus a group consists of a number of warheads, any of which would arrive at a given target at essentially the same time (see also chapter 2, Approximations).

Modification of Planning Factors: It may be desirable to prepare a number of different plans, modifying such planning factors as weapon reliability. This can be done in the Plan Generator, so that it is unnecessary to modify the data base and run through the Data Input subsystem for each modification (see chapter 2, Planning Factor Processing).

Precomputation of Data: A large amount of data, such as times of flight [illegible] group to a specific target and kill probabilities, are needed to prepare the plan. These data are precomputed and stored on a reference file for later use.

Allocation of Weapons to Targets: Using a Generalized Lagrange Multiplier method (see chapter 2, Weapon Allocation), an optimal allocation is generated subject to several forms of user-input allocation constraints. These constraints include specification of minimum and maximum desired damage levels on specified targets, restriction of certain weapon types to specified subsets of the target system, and specification of certain weapons to certain targets. Within these constraints, the Plan Generator develops the allocation which maximizes expected damage to the target system.

Selection of Desired Ground Zeros: For those targets which may have offset aim points, the DGZs are selected to optimize damage (see chapter 2, DGZ Selection).

Sorting the Allocation by Weapon Group: The output of the allocation is, for each target, the number of warheads in each group assigned to the target. This assignment is sorted to obtain, for each group, those targets which will be struck by the group.

Assignment of Multiple Independent Re-Entry Vehicles: For those missiles that possess a multiple independently targetable re-entry vehicle (MIRV) capability, the individual re-entry vehicles with each payload are assigned to aim points with the geographical constraints (footprint) of the system (see chapter 2, Basic Sortie Generation, MIRV Missile Plans).

Assignment of Strikes to Individual Vehicles: In the initial assignment, the warheads are aggregated into groups. It is next necessary to identify the bases and individual vehicles which carry the assigned warheads (see chapter 2, Basic Sortie Generation). For bombers which carry multiple weapons, missions are made up within the vehicles' range constraints. Penetration and depenetration corridors are selected.

Preparation of Detailed Sorties: The details of bomber plans are next added. These details include selection of where altitude changes are made and where air-to-surface missiles (ASMs) and decoys are launched. Specific tanker sorties are also prepared at this time.

Preparation of Plans for Simulation: The final major function of the Plan Generator is to prepare the plans on files with formats appropriate for evaluation. An additional set of processors is necessary if evaluation by programs other than the QUICK Simulator is desired.

Summarization of Plan: As an option, the plan may be summarized, giving the expected damage to each class and type of target. In addition, the plan can be evaluated in terms of the effect of varying input values for certain weapon and target parameters. The summarization may be made either after the basic weapon-to-target allocation, or after the detailed sortie plans are prepared (the allocation is adjusted slightly to take into account geographical constraints in striking a set of targets from a single vehicle).

The major steps in plan generation are summarized in figure 2. The QUICK programs which perform the steps are also shown on the figure.

Information Flow

The flow of information through the Plan Generator is as illustrated in figure 3. The basic information carried by the various files is indicated below. The last three files described are not used within the QUICK system. They are prepared for plan evaluation by other systems.

Indexed Data Base (INDEXDB or INMODDB): This tape contains the basic indexed information on both sides, as prepared by the Data Input subsystem. INDEXDB is prepared by program INDEXER and is the usual input to the Plan Generator. INMODDB, prepared by program BASEMOD, is an optional modification of INDEXDB in which targets in specified countries are deleted.

Target Input File (TINFILE): This temporary file contains all the target information required by the Plan Generator. Targets may be grouped (see chapter 2, Target List Preparation).

Weapon Input File (WINFILE): This temporary file contains all the offensive weapon information required, including information on penetration and depenetration corridors, refueling locations, recovery points, and air defense zone boundaries.

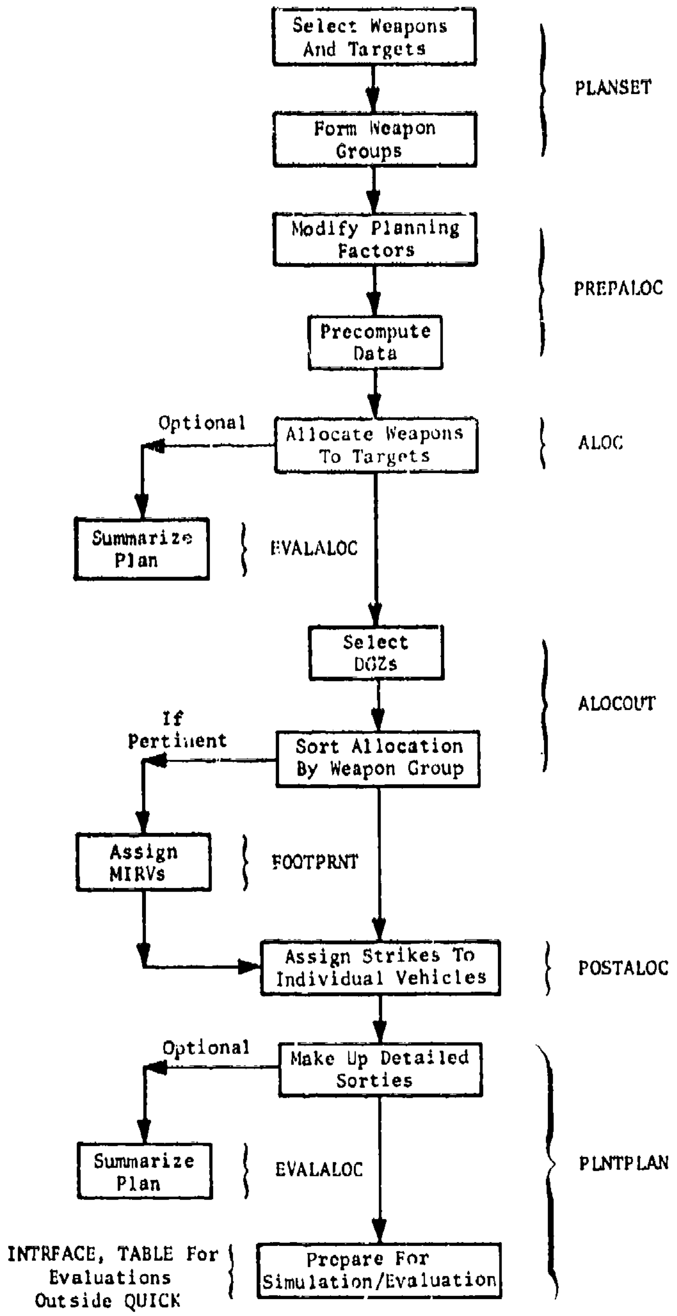


Fig. 2. Steps in Plan Generation

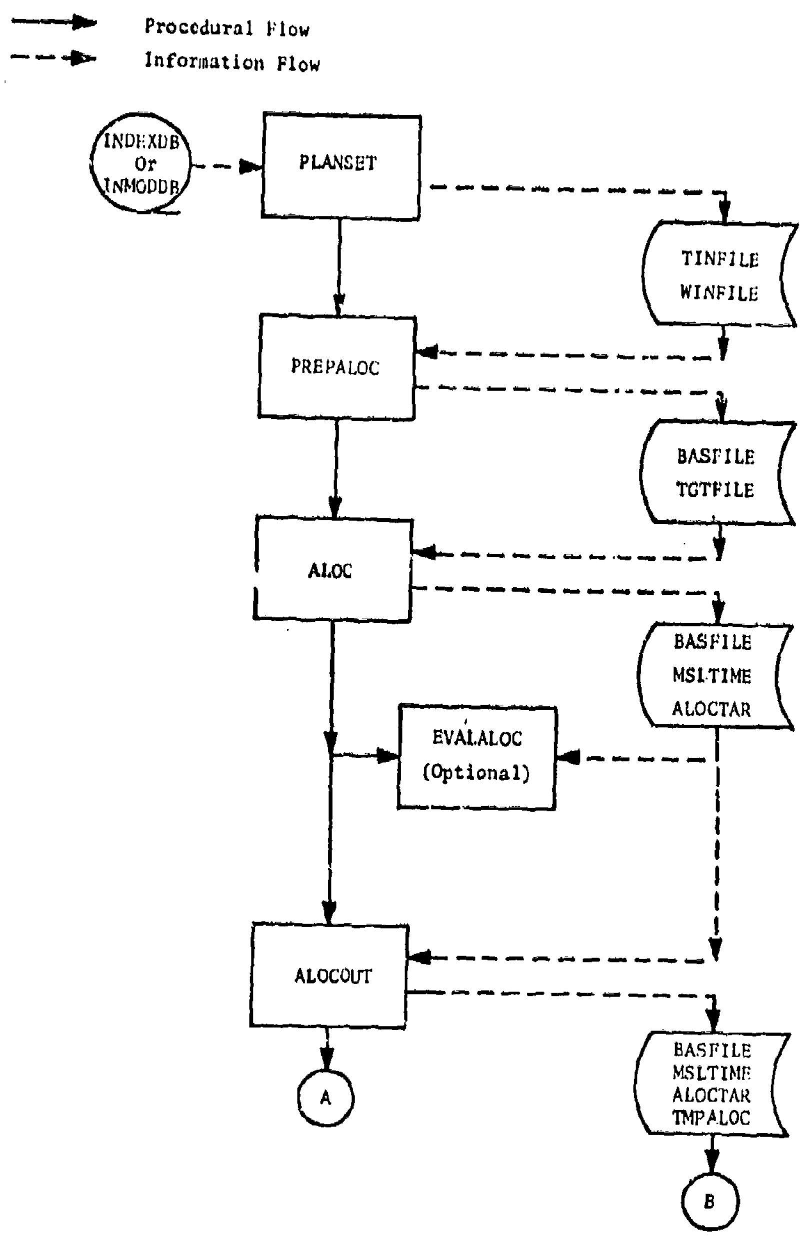


Fig. 3. Flow Within Plan Generator
(Sheet 1 of 2)

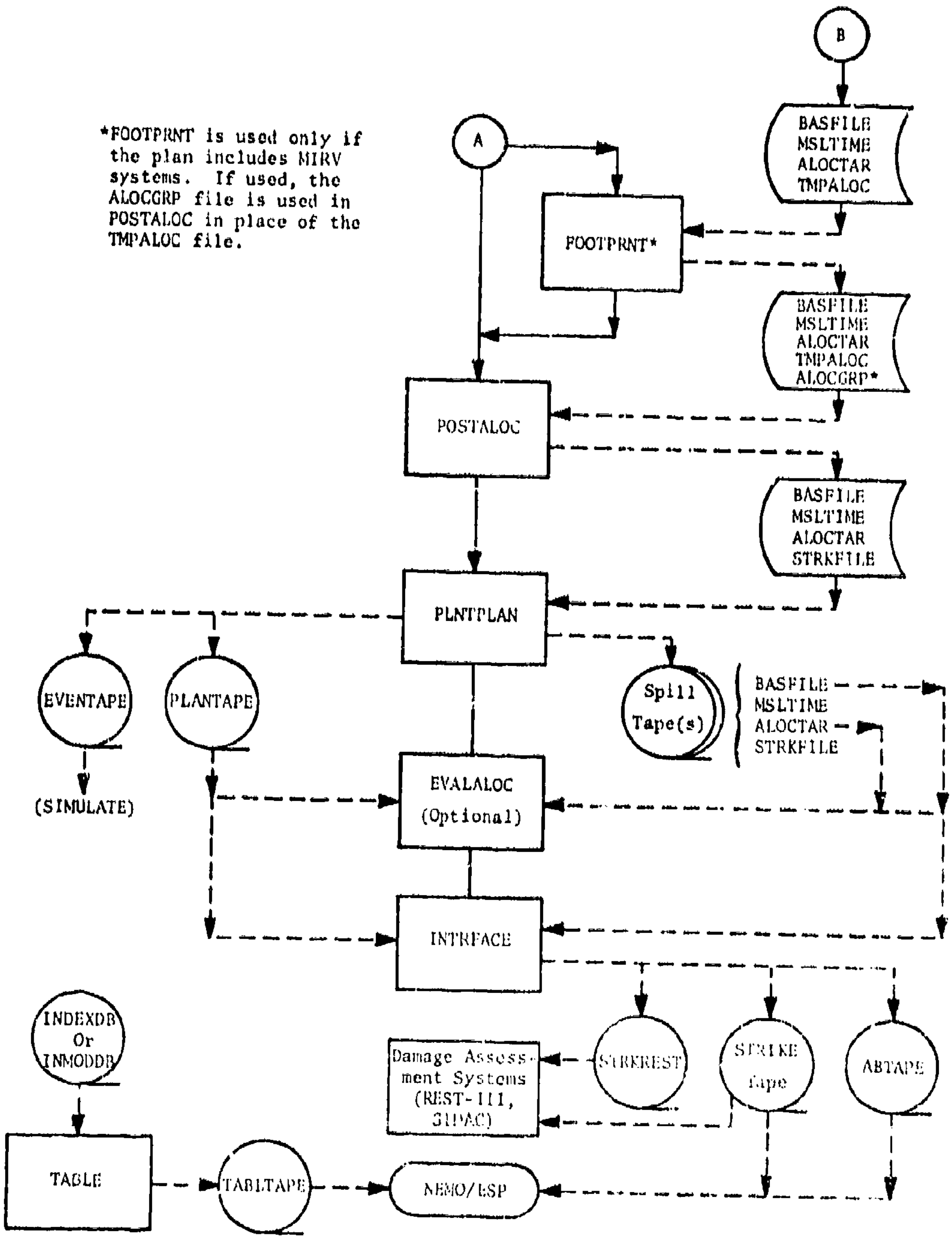


Fig. 3. (cont.)
(Sheet 2 of 2)

Base File (BASFILE): This temporary file contains base information required by all the succeeding processors.

Target File (TGTFILE): This temporary file contains the target information required by program ALOC.

Missile Time-on-Target File (MSLTIME): This temporary file contains the times of arrival of all missiles whose mission is prespecified by the user.

Allocation by Target File (ALOCTAR): This file contains the basic allocation of weapons to targets. It is used and modified by subsequent programs and finally saved as the ALOCTAR tape by program PLNTPLAN.

Temporary Allocation File (TMPALOC): This temporary file acts as interface between programs ALOCOUT and POSTALOC. It contains a list of all the strikes assigned to each weapon group. This file is also referred to by FOOTPRNT.

Allocation by Group File (ALOCGRP): This file contains the data from TMPALOC, as modified by program FOOTPRNT. If FOOTPRNT is unnecessary because there are no MIRV missiles, the ALOCGRP file is not prepared.

Strike File (STRKFILE): This temporary file contains the specifications for the missile and bomber sorties.

Planned Event File (EVENTAPE): This tape contains the final plan in a form suitable for input to the QUICK Simulator. Tanker sortie specifications are added by subroutine PLNTPLAN.

Detailed Sortie Plan File (PLANTAPE): This tape contains the final plan in a form suitable for review or for use in programs EVALALOC or INTRFACE.

Spill Tape: At this stage of processing, the spill tape contains the BASFILE and MSLTIME files, for use by EVALALOC and INTRFACE.

DGZ Targeting Tapes (STRKREST and STRIKE Tapes): These tapes contain the weapon delivery data required for detail damage assessment using the NMCSSC REST-III and SIDAC systems (REsource STatus Damage Assessment Model III and Single Integrated Damage Analysis Capability System, respectively).

Sortie Specifications Tape (ABTAPE): This tape contains the flight route and weapon delivery data required to simulate the execution of the missile and bomber plans using the NEMO and ESP simulation systems (Nuclear Exchange MOdel and Event Sequenced Program, respectively).

Weapon and Target Data List Tape (TABLTAPE): This tape contains various data tables, obtained from the INDEXDB or INMODDB tape, which pertain to the weapon systems and targets reflected on the ABTAPE.

CHAPTER 2
ANALYTICAL CONCEPTS AND TECHNIQUES

This chapter describes the major analytical concepts, techniques, and algorithms employed within the Plan Generator to accomplish the system functions described in chapter 1. For ease of reference, the detailed mathematical explanations of the more complex and sophisticated algorithms are not included in this chapter, but are presented in Chapter 3, Calculations.

WEAPON GROUPING

The initial phase of plan development provides an allocation of weapons* to targets. To reduce the amount of processing required during this phase, the offensive weapons are aggregated into "weapon groups."

Grouping Criteria

On the basis of user input which specifies the type (TYPE) weapons to be considered, program PLANSET processes the indexed data base INDEXDB, prepared by program INDEXER, and assembles the individual missile and bomber units (items in classes MISSILE and BOMBER) into weapon groups.

A weapon group is defined as a set of weapons which are assigned to delivery vehicles that are located in the same geographic area and have like characteristics. Specifically, to be in the same group, these

*A weapon is defined here as a warhead plus the characteristics of its delivery vehicle.

weapons must be of the same type; i.e., the attribute TYPE* must have the same value and the weapons must have the same alert status (alert or nonalert); and must be located in the same geographic region. Bombers must have the same refueling index (IREFUEL), and missiles must be carrying the same payload. Missile systems with a multiple independently targetable re-entry vehicle (MIRV) capability must also be assigned the same value of the attribute IMIRV (MIRV system identification).

In order for missiles or nonrefueling bombers to be grouped, they must lie within a geographic area which, for alert weapons, has a radius equal to a certain percentage of the range of the weapon. This percentage is a parameter RANGEMOD specified by the user for input to program PLANSET. If RANGEMOD is not specified, it is assumed to be 15%. The RANGEMOD value used for alert weapons is automatically doubled for nonalert weapons (to reduce the proliferation of groups). Under this criterion, it is appropriate to think of the weapons of a given group as being capable of attacking the same set of targets.

If the weapons are to be used exclusively against naval targets (a player option), all the weapons in the group must have the same value for the attribute PKNAV (the single shot kill probability for these weapons against targets of class NAVAL).

---

*A single set of delivery vehicle characteristic (e.g., speed and range) is associated with all weapons of a given type.

Group Data

When a new group is started, certain data, including the total numbers of warheads and vehicles, the average yield per warhead, and the group centroid, are stored in memory for each group as the data base is processed; additional data for the weapons which belong to the groups are stored as the weapons are assigned to the groups. The allocation of weapons to targets is subsequently carried out in terms of these "weapon groups"; for the purpose of the allocation, all weapons within a group are treated identically. This phase of processing is then followed by the sortie generation phase during which the specific missile and bomber plans are developed.

The allocations developed during the allocation phase may not be completely feasible for bombers and MIRV missiles. The allocation phase does not consider serial bombing constraints or MIRV footprint constraints, which limit a weapon system's ability to deliver warheads from one vehicle to geographically separated targets. Thus, the sortie generation phase may be required to omit certain targets from its assignments in order to create sorties which can be flown by the delivery vehicles.

For this reason, a number of weapons are artificially added to each weapon group. The formula used to add these weapons is as follows:

NEX = NWOLD * (PEX + EXB/NVOLD)

where NEX = number of weapons added to group

NWOLD = original number of weapons in group

NVOLD = original number of vehicles in group

PEX = percentage extra factor

EXB = extra vehicle factor

There is one set of increase factors (PEX and EXB) each for bombers, non-MIRV missiles, and MIRV missiles. These increase factors are user-input parameters (see User's Manual, Program PREPALOC). As a default, for bomber groups, three vehicle loads of weapons are added (PEX = 0, EXB = 3); for non-MIRV missile groups, no extra weapons are added; and for MIRV missile groups, two vehicle loads of weapons plus ten percent of the original number of weapons are added (PEX = 0.1, EXB = 2). (See the Over-Allocation subsection of the Approximations section of this chapter.)

This excess of weapons appears as an over-allocation of weapons from the weapon allocation phase. The sortie generation phase removes this over-allocation in creating the sorties. Thus the final number of weapons for which plans are generated closely approximates the number requested in the data base. (In some extreme cases, some weapons may be omitted.)

In order that the allocation phase, which uses expected-value analysis, will perceive the correct number of weapons, the probability of survival before launch (SBL) is modified for all groups which contain an over-allocation. The actual SBL is lowered by the ratio of actual weapons to the total of actual and added weapons. When the excess weapons are removed, the SBL is restored to its original value.

TARGET VALUE

The Plan Generator allocates weapons so as to maximize the target value destroyed. To accomplish this, the relative importance or value of the targets to be considered must be established. These target values reflect the major strategic objectives of the war plan which is to be generated. They must, therefore, be established by the user within the context of a specific game scenario.

Value Calculations

The QUICK system uses a two-step procedure to input the user judgmental data required for target value calculations.

1. In the data base, each potential target is assigned a value calculated to reflect its relative worth within its assigned class.
2. To generate a specific plan, the user must also provide data to the Plan Generator (program PLANSET) which determine the relative value of the target classes, and hence of all targets, for the current plan.

For the data base, a reasonably good judgment can be made of the relative values of the targets within each target class (such as missile, bomber, urban/industrial, or naval classes). The values may be based, for instance, on relative population or industrial importance for urban/industrial targets. For missile and bomber classes, the user will probably select target values which take into account each weapon's effective megatonnage, range, and CEP. Each potential target in the data base must be assigned the attribute VAL, and the value associated with this attribute must establish the target's relative worth within the class to which it is assigned.

The value input is completed with data cards input to program PLANSET. Here, when generating a specific plan, the user must input his judgment as to the relative values of the target classes. This is communicated to the Plan Generator by the selection of an exemplar (or typical) target from each target class which is to be included in the plan. To that exemplar target, the user assigns a new value (NEWVAL). NEWVAL, then, is used as follows.

$$\text{Let VALCLASS(J)} = \frac{\text{NEWVAL for the exemplar target in class J}}{\text{VAL for the examplar target in class J}}$$

and CUMVAL(J) = the sum of the VALs of all the targets in class J

Then the total value of the targets in class J is

$$\text{CUMVALF(J)} = \text{CUMVAL(J)} * \text{VALCLASS(J)}$$

These target class values are then scaled so that the sum of all target values is 1,000, thus facilitating comparative analyses of differing plans. This scaling is done by setting

$$\text{SUMVALX} = 1000 \Bigg/ \left(\sum_{j=1}^{15} \text{CUMVALF}(J) \right)$$

and establishing the final value factor for all items in class J by

$$\text{VALFAC}(J) = \text{SUMVALX} * \text{VALCLASS}(J)$$

VALFAC (J), then, is the multiplier used to derive the new value for each target in class J from its data base value, VAL; i.e., the target's value for this plan = VALFAC(J) * VAL.

The QUICK value scheme allows the user to reflect a relative judgment between the worth of two specific targets in different classes, rather than to decide the total distribution of VALUE which is to be apportioned between those two classes. This judgment is much more analogous to the usual strategic decisions. It is generally easier to specify the relative worth of Moscow vs. an SS-9 missile site than it is to specify the fraction of value that will be associated with urban/industrial targets vs. missile sites. In order to better illustrate this exemplar value scheme, a simple set of four targets is shown in table 3. In this table, one exemplar target from each class is assigned a value. The final calculated values used in the allocator sum to 1,000 and maintain the original data base ratios within each class. Also, the ratio of values between the exemplar targets is the same as the ratio between the user inputs.

Table 3. Sample Exemplar Target Value Calculation

TARGET CLASS	TARGET NAME	DATA BASE VALUE (VAL)	USER INPUT EXEMPLAR VALUE (NEWVAL)	FINAL CALCULATED VALUE
U/I	Moscow	80	16	400
U/I	Kiev	60		300
Missile	Ipich	5	10	250
Missile	Aag	1		50
			Total	1,000

The intermediate calculations used to derive the final calculated values above are:

	U/I CLASS	MISSILE CLASS
VALCLASS	16/80 = .2	10/5 = 2
CUMVAL	80 + 60 = 140	5 + 1 = 6
CUMVALF	.2(140) = 28	2(6) = 12
VALFAC*	.2(25) = 5	2(25) = 50

*Where SUMVALX = 1000/(28 + 12) = 25

TARGET LIST PREPARATION

The information provided to the Plan Generator consists of information on the target system which is to be attacked and on the available weapon systems which have been provided to deal with the target system. The weapon allocator (program ALOC) receives its targets as a shuffled target list: that is, a list of targets that are arranged in a random order.

Target Categories

From a computational point of view, QUICK considers three categories of targets: simple targets, multiple targets, and complex targets. Target numbers are assigned to all simple targets, multiple targets, and complex targets in classes 1-15 for both sides, one side at a time. A simple target is a single data base item with a single unique geographical location.

The concept of a multiple target was added to the system to increase its speed in dealing with missile squadrons. For example, a Minuteman squadron may have as many as 50 separately targetable points. From the targeting point of view, all these points have essentially the same geographic location, the same value, and the same characteristics. For efficiency in processing, therefore, QUICK allows multiple targets. A multiple target is defined as several independent, identical missile targets (such as separate missile silos in a Minuteman squadron) that are close together relative to the range of the weapon systems, but far enough apart that each target element must be treated as an independent aim point. For such targets, the right targeting for one of them is undoubtedly the right targeting for them all. Thus, the Plan Generator determines the targeting of all elements of a multiple target through a single calculation of targeting for a representative target (of the appropriate multiplicity).

The third category of target, the complex target, allows the Plan Generator to deal with targets consisting of several elements and to treat them as a single simple target during the weapon allocation phase. Complex targets are formed by the Data Input subsystem (program INDEXER) and consist of target elements (up to 40 data base items) in which each element is separated from some other element in the complex by a distance not greater than one-half the sum of the lethal radii of the two elements from a one-megaton weapon, considering the vulnerabilities for each of the elements. Under this criterion, the complex target is input to program ALOC as a single element target with characteristics which are representative of the complete complex. The procedures used in identifying and describing this representative target element are discussed later in this chapter (see Approximations - Complex Targets).

With the above simplifications, the method of allocation used by program ALOC can be essentially the same for all three types of targets.

Target Shuffling

During the allocation phase of plan generation, the rate of allocation for each weapon group is monitored as the targets are processed. To prevent these rates from being biased by a large number of similar targets considered consecutively, the basic target list is shuffled.

Since similar targets appear together in the data base (by class and type), target shuffling randomizes the order in which various types of targets are encountered. Thus the rate of allocation provides a good estimate of whether a group is being over-allocated or under-allocated. The algorithm used to achieve the required shuffling is described in Chapter 3, Calculations - Target Shuffling.

MISSILE REPROGRAMMING

Each missile type in the data base has an associated attribute IREP which indicates its reprogramming capability. Missiles may be retargetable, for instance, if other weapons in the squadron have been destroyed before launch, during launch, or in powered flight. The reprogramming capabilities considered within the Plan Generator* are:

No reprogramming capability (IREP=1)

Reprogramming for not in commission (IREP=2)

Reprogramming for destruction before launch (IREP=3)

Reprogramming for failure through launch (IREP=4)

Reprogramming for failure through powered flight (IREP=5)

*This is in contrast with the latent "reprogramming" capability of the QUICK Simulator. See QUICK Analytical Manual, Volume III, NMCSSC CSM AM 9A-67, Chapter 2, Missile Events.

During QUICK plan generation, this reprogramming capability is exercised only if the user specifies a RETARGET option in program PLANSET. The effects of missile reprogramming during plan generation are to: 1) decrease the number of vehicles per squadron; 2) reduce the DBL probability for alert vehicles to zero for those missiles which reprogram for this failure mode; and 3) increase the reliability factor for reprogrammable missiles. In computing replacement values for these parameters, the data base value associated with the following attributes is considered.

PINC:	Probability that the missile is in commission
ALERTDBL:	Probability of DBL for alert vehicles
PLABT:	Probability of a launch abort
PFPF:	Probability of failure during powered flight.

Table 4 shows the method of calculating replacement values for each level of reprogramming capability. To illustrate the reprogramming calculations, let N be the original number per squadron, R the original reliability for any missile squadron, and S be the probability of survival before launch. If N' is the reduced number of weapons, R' the increased squadron reliability resulting from reprogramming calculations, and S' the modified survival probability, N'*R'*S' will still equal N*R*S. The new values, however, reflect the probability, with retargeting, of striking the N' highest priority targets to be assigned to the squadron.

For example: for a non-SLBM (submarine-launched ballistic missile) missile squadron with attributes IREP=3, PINC=.8, ALERTDBL=.1, PLABT=.2, and PFPF=.3, and a number per squadron of 30, the new attribute values assigned (see table 4) are:

Table 4. Computations for Reprogrammable Missiles

Reprogramming Capability Index		New number per squadron (N = original number)	New ALERTDBL for this type	Reliability for this type
IREP = 1		N	ALERTDBL	PINC *(1-PLABT)*(1-PFPF)
IREP = 2		PINC * N	ALERTDBL	(1-PLABT)*(1-PFPF)
IREP = 3	Non SLBM	PINC *(1-ALERTDBL) *N	0	(1-PLABT)*(1-PFPF)
	SLBM	N	#	#
IREP = 4	Non SLBM	PINC *(1-PLABT) *(1-ALERTDBL)*N	0	1-PFPF
	SLBM	PINC *(1-PLABT)*N	ALERTDBL	1-PFPF
IREP = 5	Non SLBM	PINC *(1-ALERTDBL) *(1-PLABT)*(1-PFPF)*N	0	1
	SLBM	PINC *(1-PLABT) *(1-PFPF)*N	ALERTDBL	1

#Reprogramming for destruction before launch is not applicable to submarine-launched weapons since the destruction of one launch site destroys all remaining missiles in the squadron.

Number per squadron = PINC(1-ALERTDBL)(N)

= (.8)(1-.1)(30) = 21.6, truncated to 21

New ALERTDBL = 0

Reliability = (1-PLABT)(1-PFPF)

= (1-.2)(1-.3) = .56

Had reprogramming not been considered, the values would have been:

Number per squadron = 30

ALERTDBL = .1

Reliability = .8(1-.2)(1-.3) = .448

CORRIDOR ROUTING

Penetration/Depenetration Corridors

In QUICK, bomber routing for penetration and depenetration of enemy territory is controlled by the use of flight corridors as reflected in figure 4. These corridors are established by the user and are defined in the data base. The user is permitted to specify a number (up to 30 per side) of alternative penetration corridors that can be used by the bomber force. A penetration corridor is defined by an entrance point and a corridor origin. From the corridor origin, the aircraft is permitted to fly in a direct route to the target. The corridor also has a specified orientation or axis, which is used to indicate the general direction of the defense suppression effort. There will be a tendency for bombers to penetrate more deeply parallel to the direction of the penetration axis than at right angles to it, since the attrition rate

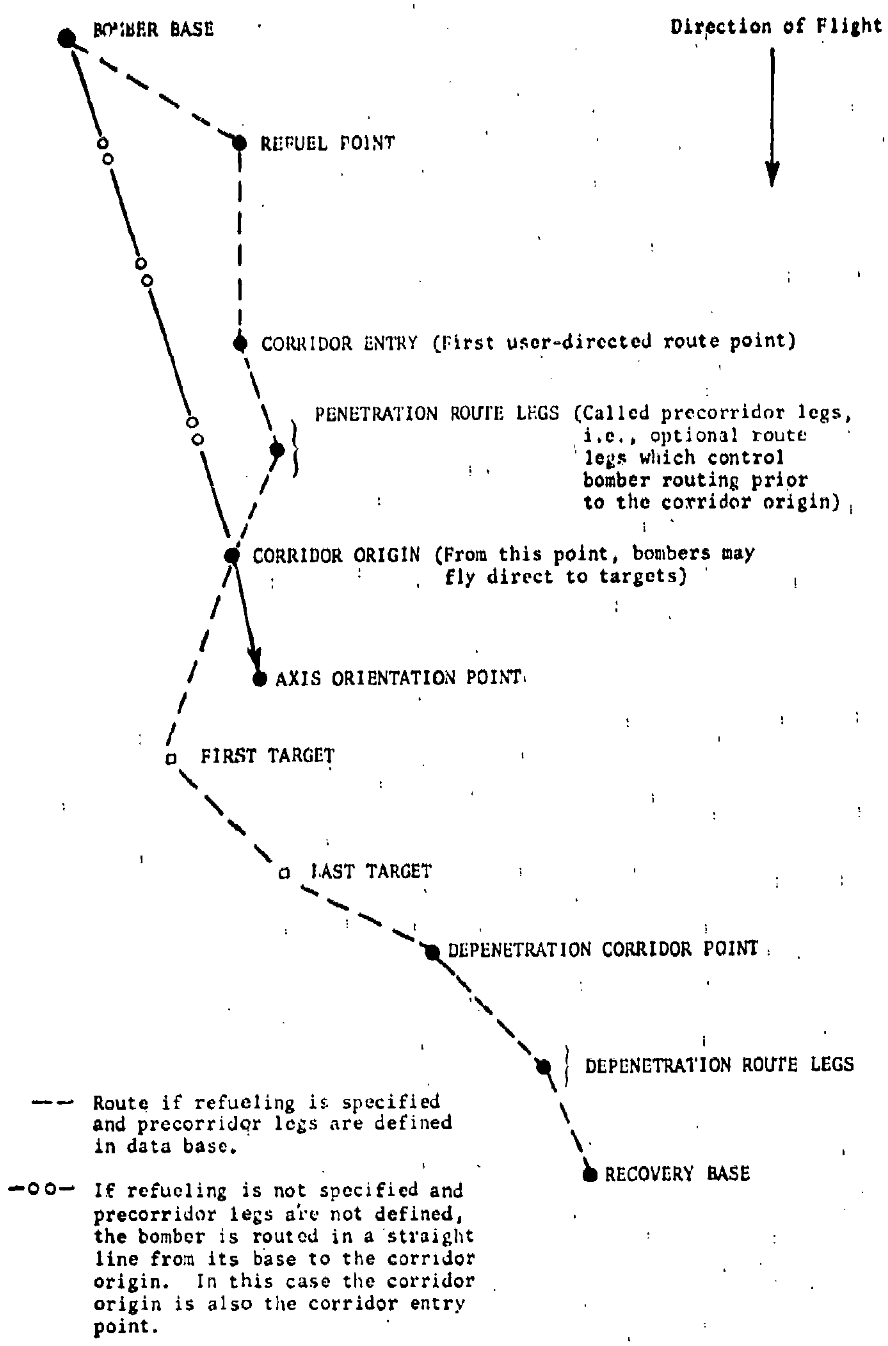


Fig. 4. Typical Bomber Flight Route

will be less (see Bomber and Missile Defenses, this chapter). The corridor axis is specified in the data base by a coordinate for the origin and a coordinate for the axis orientation point (denoted by the arrowhead in figure 4). In addition, the user may establish precorridor legs. This may be useful in order to avoid areas in which the expected attrition is high.

The user must also establish depenetration corridors which define the routing from enemy territory to a recovery base. A maximum of 50 depenetration corridors, each with up to four recovery bases, may be defined for each side. The system seeks, for each target, the most convenient depenetration corridor and associates it with the target. The depenetration corridor is specified in the data base by a depenetration point and one or more depenetration legs. The system will search from the last leg of the depenetration route and select an appropriate recovery base (see Detailed Sortie Specifications, this chapter).

Under the corridor concept, the routing of long-range strategic bombers is as follows. The aircraft is programmed to launch from its launch base; fly to a refueling area, if there is one; fly to the entrance of the penetration corridor; and fly down the corridor until it reaches the corridor origin. From this point, the bomber is permitted to fly in a direct route to the target. After the last target, the bomber is programmed to fly to the depenetration corridor entry point and fly down the depenetration corridor to a recovery base.

In actuality, not all bombers travel through geographic corridors to reach their targets. Two types, tactical bombers (those carrying nuclear weapons) and naval bombers (those restricted to attacking targets in class NAVAL), fly directly from their launch point (or refuel point) to their targets. However, to facilitate the creation of flight plans for these two types of aircraft, two dummy corridors (one for each type) are defined in the data base. While these corridors have no geographic significance, their assigned parameters do reflect the attrition to which aircraft will be subjected as they fly to their targets.

Corridor Attributes

The QUICK System allows up to 30 corridors per side to be used in a war game. However (for each side) each corridor must be defined as belonging to one of five possible corridor types designated by the user; e.g., TYPE ATTRLO (attrition low), TYPE ATTRHI (attrition high). Each type of corridor is associated with a set of type characteristics (attributes). These type characteristics, with exception of attrition on precorridor legs (Attribute KORSTYLE), are used within the Plan Generator to establish the area attrition rates for bombers (See Bomber and Missile Defense, this chapter). Following is a description of the corridor attributes.

ATTRCORR — Normal attrition rate for high-altitude aircraft using the corridor

ATTRSUPF — A reduced attrition rate for high-altitude aircraft applicable near the main axis of the corridor

DEFRANGE — Typical range of interceptor aircraft on bases near a corridor (nautical miles)

HILOATTR	The ration of low-altitude attrition to high-altitude attrition (decimal fraction)
KORSTYLE	Attribute used to control the mode of corridor penetration (referred to as parameter k when used in the calculation of curvilinear coordinates--see Basic Sortie Generation, this chapter).

BOMBER AND MISSILE DEFENSES

The modeling of the effect of enemy defense operations on weapon survival during penetration is divided into two parts: area and terminal. Area defense considers those defenses which affect weapons without regard to their assigned targets. Terminal (or local) defenses affect only those weapons attacking specific targets.

Bomber Defenses and Corridor Selection

In the case of bomber/area defenses, the penetration probability is estimated on the basis of the nominal attrition rates ascribed to the penetration corridors. Each corridor is ascribed at least two attrition rates:

ATTRCORR	Normal attrition rate for high-altitude aircraft using the corridor
ATTRSUPF	A reduced attrition rate for high-altitude aircraft applicable near the main axis of the corridor.

In addition, attrition rates can be specified if desired for any prescribed legs between the entrance and origin of the corridor, and attrition can be specified in connection with penetration to defended targets (TARDEFs). These attrition rates are used to estimate the penetration probability. However, it is also assumed that the attrition rates can be reduced by the factor HILOATTR for portions of the route where the aircraft can fly low. Any excess range available to the aircraft at high altitude is used to provide a low-altitude flight -- assuming a conversion factor RANGEDEC between low-altitude and high-altitude fuel consumption. The estimated low-altitude range is then allocated among the legs of the mission to minimize attrition.

To represent the effect which area and terminal defense will have upon the successful execution of any bomber attack plan, a probabilistic approach is used. The level of defense in a given area will directly affect the probability that a bomber which travels through this area will successfully reach its subsequent flight points. Therefore, each section of geography over which bombers fly is characterized by attrition parameters which reflect the level of area and local defenses for that section. These parameters will, in turn, determine SURV(I), the probability that the bomber will survive to reach flight point I. Finally, VALSORTY, the total value of a sortie, is defined as follows:

$$\text{VALSORTY} = \sum_{\substack{\text{all flight} \\ \text{points}}} \text{SURV(I)*V(I)}$$

where V(I) = estimated value of reaching flight point I. This value V(I) is the relative value RVAL generated during weapon allocation by program ALOC (see Basic Sortie Generation).

The computation of SURV(I) for the formula is based on a simple exponential attrition law. If the integrated attrition probability on each individual leg to a point J is given by ATLEG(J), then the survival probability for the bomber to the point I will be given by:

$$\text{SURV(I)} = \text{EXPF} \left[- \sum_{J=1}^{J=I} \text{ATLEG(J)} \right]$$

The attrition ATLEG(J) includes both area and terminal attrition for the leg. Figure 5 illustrates the attrition attributes and variables used in the program POSTALOC.

The area attrition for each leg is computed by integrating the assumed area attrition rate over the length of each leg. After the first target, this assumed area attrition rate per nautical mile is a constant, equal to the data base variable ATTRCORR supplied for the corridor. Prior to the first target, the assumed attrition rate decreases exponentially toward the limiting value ATTRSUPF which is also a data base variable for the corridor. Thus the variable representing the assumed area attrition rate between the origin and the first target is given by:

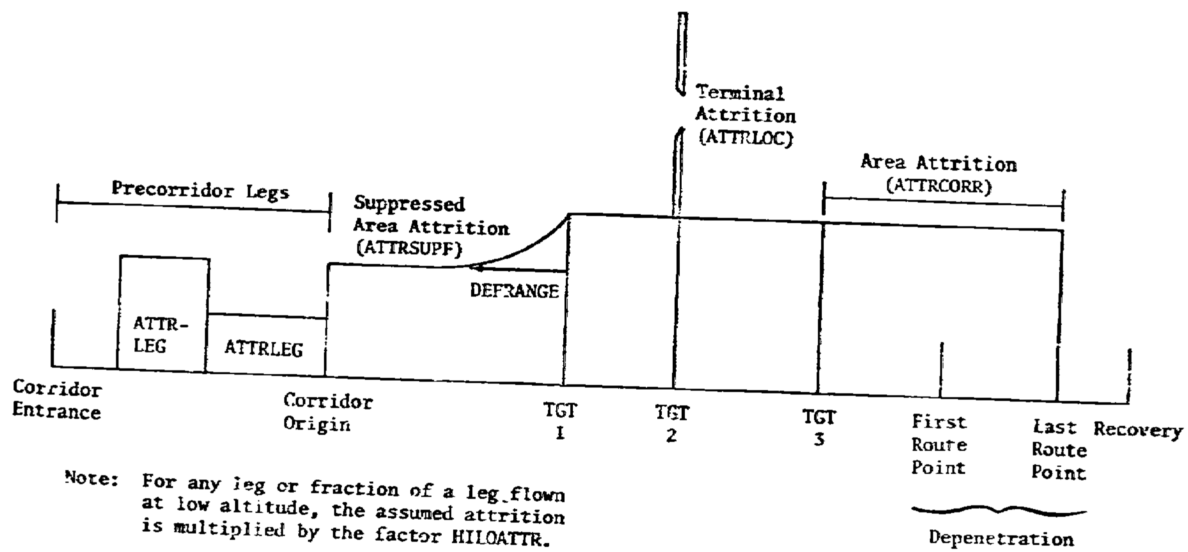


Fig. 5. Illustration of Attrition Attributes and Variables (Used in Program POSTALOC)

Rate=ATTRSUPF + (ATTRCORR - ATTRSUPF)*EXPF(-X/DEFRANGE)

where X = the distance in nautical miles between the corridor origin and the first target and DEFRANGE is the typical range in nautical miles of interceptors on bases near the corridor. Attrition rates (ATTRLEG) may also be specified for the precorridor legs leading in to the corridor.

The terminal attrition ATTRLOC (see TGT2 in figure 5) is estimated directly from the data base variable TARDEF. Each potential target with a local (terminal) surface-to-air missile (SAM) defense is assigned the attributes TARDEFHI and TARDEFLO. The value assigned these attributes reflects the level of bomber defense, at high and low altitudes, provided by local SAM units. Considering the bomber's altitude (e.g., high) the local attrition ATTRLOC is estimated as follows:

ATTRLOC = .1*TARDEFHI

Naturally, this local attrition is of concern only when the route point characterized by this local attrition is itself a target for a bomb. It produces no effect if the target with which it is associated is attacked by an ASM (air-to-surface missile) that is launched from another route point. Moreover, even if the sortie definition indicates that the ASM is launched at the target from the vicinity of the target itself, it is assumed that the actual launch point will be such that the aircraft will not be required to penetrate the local defenses. Thus, any local attrition associated with the ASM target is again ignored. Finally, it is assumed that all local attrition is applied only to the incoming leg to the target and that on any leg or fraction of a leg flown at low altitude the attrition rates will be reduced by the factor HILOATTR. In order to estimate the expected value of the sortie, therefore, an estimate must be made of how the available low-altitude range should be applied (discussed under Basic Sortie Generation, this chapter). Notice that a change in the assumed attrition rate for any leg or part of a leg will change the integrated attrition for the leg ATLEG(J). This in turn will change the probability of survival to any point I (SURV(I) which is required to evaluate VALSORTY.

During the weapon allocation phase (program ALOC), detailed sortie information (i.e., routing and sequential targeting) has not yet been generated. Therefore, bomber penetration of area defenses is treated as follows.

In weapon allocation, only one target is under consideration per vehicle. Therefore, in allocating low-altitude range among the legs of a mission to minimize attrition, much less weight on attrition is placed after the target has been reached. The algorithm assumes that the normal corridor attrition ATTRCORR applies to the entire route from the target to depenetration, and to a portion of the route prior to the target equal to the perpendicular distance of the target from the main axis of the penetration corridor. The suppressed attrition ATTRSUPF is assumed to apply for the remainder of the route from the corridor origin to the target.

In computing the range of the aircraft, the normal range RANGE is used starting from the centroid of the weapon group for nonrefueled aircraft (IREFUEL=0) and from the specified refueling area for area type refueling (IREFUEL $\geq$ 0). In the case of buddy refueling, the refueled range RANGEREF is used, but distances are again measured from the weapon group centroid.

The penetration calculation is implemented by dividing the aircraft attrition elements into four "LEGS."

LEG = 1	Corridor entrance to origin (distance equal to sum of all such legs with attrition specified -- attrition equal to sum of attrition on all such legs)
LEG = 2	Corridor origin toward target as far as suppressed attrition (ATTRSUPF) is applicable
LEG = 3	End of LEG 2 to target -- ATTRCORR applies but is augmented by any local attrition at a defended target TARDEF
LEG = 4	Target to depenetration -- ATTRCORR still applies but value of mission and seriousness of attrition (RATE) is assumed to be less by a factor of approximately .25.

The available low altitude is then distributed among these legs, and the penetration probability is estimated. To select the preferred penetration corridor, a weight, .75, is given to reaching the target; the remaining weight, .25, is assigned to reaching the depenetration corridor. The corridor showing the highest value (Σ weight*penetration probability) is chosen, and the penetration probability to the target via that corridor is recorded for the group. If the group has been specified for nonrecovery (IRECMODE = -1, the recovery distance is simply set to zero.

On leg 3, the terminal attrition parameter TARDEF is modified by two factors, TARFAC and EXPASM. TARFAC is a user-input parameter which allows adjustment of the perceived terminal bomber defenses during program ALOC. EXPASM is the fraction of weapons in a group that are air-to-surface missiles (ASMs) rather than gravity bombs. The modified terminal bomber defense attrition is therefore defined as:

$$\text{TARDEF} \times \text{TARFAC} \times (1.0 - \text{EXPASM})$$

This use of EXPASM reflects the fact that a bomber delivering an ASM to a target does not penetrate the target's terminal defenses.

Missile Defenses

Ballistic missile defenses involve a simpler model. Only a random defense is considered for area attrition of missiles. Each warhead, regardless of its assigned target, has the same probability of being destroyed by the random area defenses. One random area kill probability is input for each side. (The QUICK Simulation subsystem also considers a preferential area defense against ballistic missiles.)

Terminal defenses are modeled by a subtractive model. Each target with terminal defenses is assigned a number of terminal ballistic missile interceptors. This number of interceptors (variable MISDEF) is input in the data base via the attribute NTINT which must be defined for each defended target.

The input variables describing the target's terminal defense capability allow uncertainties to be introduced in the number of interceptors present. MISDEF is the "nominal" number of interceptors on the target, each with kill probability PKTX against an unhardened warhead. In addition, four other parameters are defined (the same for all targets) which introduce uncertainties in MISDEF. RXLOW is a factor which, when multiplied by MISDEF, gives a lower estimate of interceptors which has probability PXLOW of occurring. Likewise, RXHIGH and PXHIGH define the overestimate of interceptor availability. Thus, if there is imperfect knowledge of the defense capability, the allocator can hedge against these uncertainties when assigning weapons.

In addition to the target-associated defense data, it is possible to describe penetration aids suitable for the various missiles by means of the Payload Table. For a particular payload index, the following variables* describe the penetration aids:

*NWHD is data base attribute NWHDS; NTDECOYS is attribute NDECOYS; XDEC is a user-input parameter to program ALOC.

NWHD = Number of warheads per independent re-entry vehicle package.

NTDECOYS = The number of "aim points" the terminal defense sees for each independent re-entry vehicle (in addition to the warheads).

XDEG = A factor by which the PKTX is multiplied to obtain terminal interceptor kill probability against this weapon type. It reflects additional hardening of the warhead or electronic penetration aids which can degrade interceptor effectiveness.

An independent re-entry vehicle package is a set of warheads and terminal decoys that can be guided to a target point (or points) independently. For missile boosters with a multiple independently targetable re-entry vehicle capability (MIRV), there may be several independent RVs per booster. Otherwise, each booster delivers one set of warheads and decoys.

The penetration probability of any warhead is a function of all the missiles allocated to the target. The model computes the total number of objects allocated to the target, NOBJ, as the sum of all warheads and decoys* allocated to the target. The number of perfect interceptors, variable PINT, is defined as:

*For each weapon, this is the sum of NWHD and NTDECOYS multiplied by the product of the survival before launch probability, weapon system reliability, and command and control reliability.

PINT=PKTX*[(PXLOW*RXLOW)+(PXHIGH*RXHIGH)+(1-PXLOW-PXHIGH)]*MISDEF

This variable is the expected number of objects to be removed by the terminal defense interceptors.

The penetration probability for any warhead is defined as:

$$1.0 - \left[XDEG^{*} \frac{PINT}{NOBJ} \right]$$

If this probability is less than (1.0 - PKTX*XDEG), it is reset to that value.

BOMBER REFUELING

Refueling Modes

The QUICK design provides for modeling two kinds of bomber refueling capabilities: "buddy" and area. In buddy refueling, two aircraft take off together and fly to the refuel point; one then provides fuel to the second and recovers. Fuel can be provided by either a tanker or another bomber of the same squadron as the aircraft being refueled.

There are two types of area refueling: directed and automatic refueling. In the directed mode, the user establishes, in the data base, a specific refueling area (up to 20 per side may be defined in the data base) and manually assigns the appropriate bombers and tankers to this area. In the automatic mode, the Plan Generator (program PLNTPLAN) develops the refueling plan on the basis of information provided in the data base. The data base reflects the bomber squadrons which require refueling and the tankers which are available. Program PLNTPLAN then selects the refueling area (up to 30 additional refueling areas may be added) and assigns the bombers and tankers accordingly. To reflect the refueling requirements associated with a specific plan, the user defines the attribute IREFUEL for all bomber and tanker units defined in the data base. The codes which may be assigned as the value of IREFUEL are as follows:

IREFUEL Setting	Definition
-5	Automatic refueling -- two refuelings required.
-4	Automatic refueling -- one refueling required.
-3	This code is used to flag air-breathing missiles which are to be treated as aircraft when calculating attrition rates - no refueling involved.
-2	Buddy refueling -- a bomber from the same squadron is used in a tanker role.
-1	Buddy refueling in which support is provided by a tanker. Tanker units associated with buddy refueling need not be defined in the data base.
0	No refueling required.
≥1	Directed area refueling -- refuel area and bomber/tanker assignments are directed by user.

For the weapon allocation process to reflect accurately the appropriate range of all available aircraft, it is necessary to decide prior to the allocation which aircraft have their refueled range and which do not. If the user has specifically assigned the refuel area and/or buddy refueling capabilities, the program assumes that the aircraft can be refueled and so indicates to the weapon allocation portion of the program. If the user selects the automatic refueling capability, there may not be enough tankers, and therefore a decision must be made in program PLANSET as to which bombers are to be refueled and which are not. If a count of the bombers requiring automatic refueling and the tankers available to perform this refueling indicates a deficiency of tankers, the aircraft are given the refueled range on the basis of a set of priorities built into the program. Alert aircraft are always given priority over nonalert; aircraft with the least unrefueled range are given priority over those with a greater range. Thus, when the weapons are allocated, the range capability has been completely determined, and the sorties generated by program POSTALOC assume either the refueled or unrefueled range generated by program PLANSET. Where bombers are used as tankers in buddy refueling (i.e., a bomber unit is assigned the refuel index IREFUEL=-2), the number of bombers available for the strike is cut in half.

Selection of Refueling Areas

For the directed area mode of refueling, the user assigns refuel areas for both bombers and tankers, and the vehicles are scheduled accordingly.

Where buddy refueling is to occur, tankers are ignored by the system. Bombers are scheduled to refuel at the "buddy point," which is at maximum range (as defined below) or at the corridor entry, whichever is earlier. The maximum range is determined by:

1. Let REFDIF = the bomber's refueled range minus range.

   Let DIS = the distance (in nautical miles) from base to corridor entry.

2. If DIS $\leq$ REFDIF, let FACTOR = the greater of $\frac{DIS-5}{DIS}$ or zero.

   If DIS > REFDIF, let FACTOR = $\frac{REFDIF}{DIS}$.

3. Now using FACTOR, the desired point is found by an interpolation along the great circle route between launch base and penetrated corridor entry point if the longitudinal difference between base and entry point is greater than 2.8 degrees. Otherwise, the desired point is determined, using a straight line or Mercator interpolation.

For the third case, in which a bomber is to be automatically assigned a refuel area by PLNTPLAN, the buddy refuel point X is first computed as for buddy refueling. The list of tanker bases is then scanned to see whether the point X is within range of any of them. If not, the closest tanker base is chosen, and a new buddy point is calculated by interpolation. The new point will fall between the tanker base and the original buddy point and will be within range of the tanker base.

Next, the refuel area nearest the buddy point (if one exists within a predetermined radius) is selected. Let REFDIF be the difference between refueled range and range (see figure 6). If there already exist refuel areas which are within REFDIF of the base and within the specified distance D of the buddy point X, the area nearest X is assigned as the bomber's refuel area. Otherwise, the point X is assigned and added to the list of refuel areas. Available tankers will later be assigned and scheduled by PLNTPLAN in such a way as to service all automatically assigned bombers (see Detailed Sortie Specifications, this chapter).

PLANNING FACTOR PROCESSING

Modifications

In order to allow minor corrections to data base values for planning factors and to provide the capability to rerun the allocation phase rapidly using alternative values for these planning factors, the Plan Generation subsystem allows for planning factor modification just prior to weapon allocation. The modified factors are considered during weapon allocation, but other phases of plan generation use the original values (if the factors are required after weapon allocation). The factors that may be changed are target value (VALUE), minimum required destruction fraction (MINKILL), maximum desired destruction fraction (MAXKILL), weapon unrefueled range (RANGE), and weapon refueled range (RANGEREF).

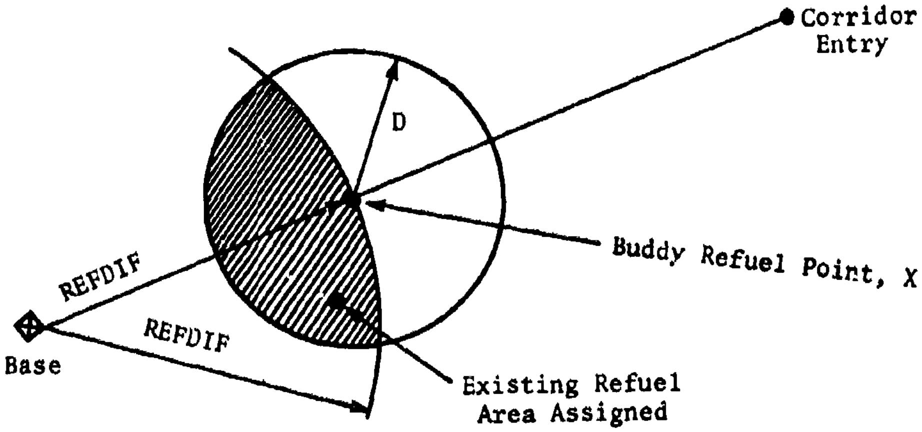


Fig. 6. Assigning a Refuel Area (Automatic)

Each change request for VALUE, MINKILL, or MAXKILL specifies the set of targets for which the change is to be effected. This set (which may consist of only target) may be defined by specifying the target class (CLASS), target type (TYPE), target name (TGTNAME), target identifier index number (INDEXNO) or designator code (DESIG), or any combination of these. Any target that fits the requirements of the request (e.g., appropriate class or type) is given a modified value for the specified planning factor. If a target characteristic (e.g., class) is not specified in a request, the characteristic is not checked in determining the range of the change request. If a target fulfills the requirements of a change request and is a component of a complex target, the planning factors for the entire complex are modified to reflect the change to the component. If the target is a component of a multiple target, the planning factors of all the components of the multiple target are changed. If target value, VALUE, is modified, the target values for all targets are renormalized so that the sum of values in the target system remains constant at 1,000.

Weapon planning factors may also be modified. The system allows the user to change range (RANGE), refueled range (RANGEREF), and set a minimum range (RANGEMIN) by weapon group. These new ranges are used only in program ALOC in assigning targets to the weapons. The minimum range specifies the minimum distance from group centroid to target for any assignment of weapon to target.

Uncertainty Considerations

In the past, automatic "plan generators" have tended to produce plans that were deficient from a military point of view because they did not take into account the uncertainties in parameters used to define the target system and the delivery systems. Because these uncertainties have been ignored, the resulting plans (while they may have been optimum or near optimum for the specific mathematical assumptions) have often been very deficient in the face of any variation of the assumptions. This is not a trivial point, since the hallmark of a satisfactory, realistic war plan is its ability to function in an acceptable way in the face of very large uncertainties. A war plan which does not explicitly take such uncertainties into account can be useless for application in a realistic war gaming system, even though it may be rigorously optimized for apparently very reasonable assumptions.

In the present system, such factors will be incorporated in the war plans if, and only if, the factors are explicitly included in the payoff function. Since the plans that are produced are optimized with respect to the payoff function, it is necessary to give careful attention to the payoff function itself. The following paragraphs illustrate the importance of various types of uncertainties in the development of realistic military plans.

Uncertainties in Probability of Destruction Before Launch and Overall System Reliability: A very real concern in the design of any strategic war plan is the possibility of total or almost total failure of an entire weapon system. Such failure could occur because of failure of command and control in a given region, or because of unexpectedly heavy and effective enemy targeting of a system. A less complete but nevertheless important failure to meet expectations could occur because of an erroneous estimate of system reliability or accuracy. Because of such uncertainties, real war plans place heavy emphasis on cross-targeting of critical targets, so that the destruction of such targets is not dependent on the survival of a single base, or a single type of weapon system. Unless such correlations in survival probability are taken into account, an automatic war plan generator may simply select the single most effective type of weapon against any given target and allocate enough of these weapons to achieve the desired kill probability. Such a plan would, of course, be unacceptable because of its failure to recognize the uncertainty in the overall survival probability for the system chosen.

In a theoretical sense, any factor which individual delivery systems may have in common should be taken into account in the development of a cross-targeting plan, and the same theoretical techniques could be used to deal with them all. However, in practice, a simple prohibition on the use of weapons from a single bomber or single missile base against the same target is much simpler and therefore more practical (where it is appropriate) than providing for explicit analysis of more general alternatives.

In the case of missiles from the same launch base, and in the case of multiple bombs delivered by the same bomber, such a simple prohibition is used. There are a sufficiently large number of alternative launch bases of each type that it should always be possible to use an alternative without much loss of efficiency.

However, in the case of other common factors, such as the same type of delivery system, or the same region of origin, there will almost certainly be cases when it is appropriate to use more than one of the same type of weapon despite the common factors. The delivery system groups, as considered by program ALOC, are identified by class, type, region of origin, and alert status, so that the advantages and disadvantages of cross-targeting can be explicitly considered. The treatment of this type of uncertainty is discussed in the section entitled, Weapon Correlations, in this chapter.

These considerations illustrate some of the reasons why plan generation schemes based on simplified mathematical models have typically failed to produce adequate or realistic war plans. The Plan Generator for the QUICK system, however, does take these factors into account. It also takes account of range constraints and variations in penetration probability (even when these are a function of the range to the target, or the position of the target relative to specified penetration corridors). On the other hand, it is only reasonable to anticipate that there may be additional criteria which must be incorporated to produce fully satisfactory plans. Therefore, an essential feature of a satisfactory plan generator is the ability to incorporate additional targeting rules and new targeting criteria without excessive difficulty. This was one of the important considerations in the selection of the present design.

Target Vulnerability Uncertainties: In a real war plan, it may be worth allocating one weapon to a target even when it is supposedly too hard to be very vulnerable to such attack. There is always a possibility of a design flaw in the hardening of a class of targets, so that (despite intent) a single weapon might nevertheless have a substantial chance of putting the target out of operation. An automatic plan generation system intended to maximize the value of enemy targets destroyed would very likely ignore such targets entirely unless provision is made to reflect the uncertainty in target vulnerability.

In the QUICK weapon allocator, provision is made to include a probability mix of target hardness which reflects such uncertainty about target vulnerability. The kill probabilities used by program ALOC thus explicitly take into account the possibility that targets are more or less vulnerable than they seem. The resulting optimization, of course, automatically reflects this possibility.

There are a number of alternative ways in which the hardness uncertainty can be treated. The technique used here is chosen because it is also compatible with the treatment of correlations in delivery probability.

Uncertainty in target hardness is treated by permitting each target to be represented by two "hardness components" rather than one. That is, the total target value VTO is distributed among the components with hardness H(J). Each component is assumed to have some probability of occurring. This probability is reflected in the expected values VO(J) for the separate components. In effect, the calculation of target survivability is then carried out independently for each hardness component, and the surviving values are totaled to give the overall expected probability of survival for the target.

For soft targets, a single component seems adequate; for hard targets, two will often be desirable. The increase in calculation involved is proportional to the number of components used, but the inclusion of hardness components does not add to the theoretical complexity of the payoff function. The "component" approach also provides a better capability to deal with separate targets of different hardness located at the same point, or with targets that are known to be soft only for short time intervals, but at unknown times.

Uncertainties in Time-Dependent Target Values: Of course, any useful war plan generator must take into account the time dependence of target values. Otherwise, slow bombers might be the only weapons targeted against fleeting missile and bomber bases. However, it is also important to reflect the uncertainty in the time dependence. Clearly, it is impossible to know whether enemy launches will go on schedule, or even how they are scheduled. Thus, if the value of targets is linked exclusively to a single estimated departure time, the importance of hitting a target even after the estimated departure time may be ignored. To reflect these uncertainties, the time-dependent target values supplied to program ALOC are smoothed to reflect the "expected" value of the target as a function of time when the uncertainties in the time of departure and probability of departure are included. The exact nature of the smoothing function is described in the Time-Dependent Target Value section of this chapter. The use of the smoothed value is described in the following paragraphs.

The inclusion of the time dependence of the target value implies that the target value which can be affected by a weapon will depend on the weapon time of arrival. These target values (at the time of arrival of the weapons) are computed for each weapon group G and for each hardness component J. The resulting values are stored in an array V(G,J).

To understand the time-dependent aspect of the payoff, consider all the weapons actually allocated against a target, arranged in order of time of weapon arrival. Let the index (NI) represent the $(NI)^{th}$ time in this ordered sequence, and let the index (NN) represent the final or last time in the sequence. Now consider that portion V(NI,J)-V(NI + 1,J) of the target value for the J^{th} hardness component which will disappear between the arrival of weapon (NI) and (NI + 1). This portion of the target value will be subject to destruction by weapons at time (NI) and all those that precede it. It will not be hazarded by weapons at time (NI + 1) or any weapons which arrive later. If we represent by S(NI,J) the probability that the J^{th} hardness component will survive all weapons arriving up to and including the time (NI), then we can express the total surviving value of the J^{th} hardness component as follows:

$$\sum_{NI=0}^{NI=NN} \left[V(NI,J) - V(NI+1,J) \right] * S(NI,J)$$

The total residual target value in all M hardness components is then

$$VT = \sum_{J=1}^{J=M} \sum_{NI=0}^{NI=NN} \left[V(NI,J) - V(NI+1,J) \right] * S(NI,J)$$

where V(NN+1,J)=0, V(0,J)=VO(J) and S(0,J)=1.0. The payoff is, of course, just the initial target value VTO minus the residual target value VT.

This relationship is used throughout in the allocation for the calculation of payoff in connection with any combination of weapons.

If all the weapons were independent, the quantity S(NI,J) would be simply equal to the product of the survival probabilities SSSP(G,J) for all weapons on the target up to and including the time NI. However, in order to deal with the problem of correlations in delivery probabilities, a more general formula is used for S(NI,J). To explain this formula, it is necessary to develop a mathematical model for dealing with correlations in delivery probability. This explanation is contained in the Weapons Correlation section of this chapter.

Approximations

Group Centroid: To reduce the amount of processing required during the allocation phase, the offensive weapon launch bases for the attacking side are aggregated to form weapon groups. For simplicity in the allocation, a single group centroid is specified from which timing and distance calculations are made. Program PLANSET processes the indexed data base INDEXDB and assembles the individual missile and bomber units into groups (see Weapon Grouping). As a new base is added to a group, the latitude and longitude of the group centroid are adjusted so that the final values reflect the true group centroid. That adjustment is effected as follows.

Let NG = The number of bases included in the group prior to this addition

LSTLAT = Latitude of centroid before addition

LSTLONG = Longitude of centroid before addition

LAT = Latitude of the weapon being added

LONG = Longitude of the weapon being added

Then for the new centroid latitude (NEWLAT),

$$NEWLAT = \frac{(NG*LSTLAT)+LAT}{NG+1}$$

To determine the new centroid longitude (NEWLONG) an intermediate quantity (GLONG) is calculated. If GLONG < 0, NEWLONG = GLONG + 360; otherwise NEWLONG = GLONG. GLONG is calculated as follows:

1. If -180 ≤ (LSTLONG - LONG) ≤ 180 then GLONG = [(NG * LSTLONG) + LONG]/[NG + 1]
2. If (LSTLONG - LONG) > 180 then GLONG = [(NG * [LSTLONG - 360]) + LONG]/[NG + 1]
3. If (LSTLONG - LONG) <-180 then GLONG = [(NG * LSTLONG) + (LONG - 360)]/[NG + 1]

<u>Average Yield (Bombers)</u>: One of the composite characteristics calculated for a bomber group is its average yield per warhead. That value is obtained as follows. As each bomber squadron is to be added to the group, the squadron's total yield, which equals

(Yield for a type 1 bomb)*(# of type 1 bombs carried)
+(Yield for a type 2 bomb)*(# of type 2 bombs carried)
+(Yield for the ASM type carried)*(# of ASMs carried)

is added to the current total yield for the group.

When all groups have been formed, this total yield for each group (G_i) is replaced by the average yield per warhead for that group:

$$\text{Group } G_i \text{ average yield} = \frac{\text{Total yield, group } G_i}{\text{Total \# warheads, group } G_i}$$

It is this average yield which is used for all weapons of the group during the allocation phase (program ALOC).

MRV/MIRV Payloads: In QUICK, those missiles equipped with a multiple re-entry vehicle (MRV) capability are allocated to a single target. For allocation purposes, the component RVs (re-entry vehicles) are considered to be a single warhead; however, the added effect of the MRV's pattern is reflected in the formula used to determine its expected yield:

$$\text{MRV yield} = (\text{yield for one warhead of the given type}) * (\text{the number of warheads, or RVs})^{3/2}$$

The number of warheads (re-entry vehicles) is raised to the 3/2 power in order to accommodate the "2/3 rule" for comparing the yield of N MRV warheads delivering X megatons each against the yield of one warhead of NX megatons striking the target center.

Multiple independently targetable re-entry vehicles (MIRVs), on the other hand, are allocated as separate weapons, subject to footprinting constraints. Hence, for the case in which the independently targetable re-entry vehicles (IRVs) of a missile with MIRV capability are in turn equipped with MRVs, the expected yield calculated is:

Yield for missile with MIRV capability =

(yield for one warhead of the given type)

*(the number of IRV's)

*(the number of warheads, or RV's, per IRV)$^{3/2}$

Overallocation: The QUICK weapon allocator is designed to assign the individual weapon of a group to specific targets. In developing this allocation, program ALOC does not consider serial bombing constraints or MIRV footprint constraints. These constraints reflect the physical limitations on a delivery vehicle's ability to deliver warheads to geographically separated targets. In addition, in allocating bomber weapons, the number of weapons associated with a given penetration corridor may not correspond to an integral number of delivery vehicles.

The above constraints are considered in the sortie generation phase of plan development. To provide some flexibility in developing feasible weapon assignments for each delivery vehicle, a few extra weapons are added to each MIRV and bomber weapon group for allocation by program ALOC. Subsequent processing by programs POSTALOC (for bombers) a FOOTPRNT (for MIRV's) removes this overallocation in creating the sortie specifications.

The extent of excess weapon addition has been determined through experience in the use of the heuristic algorithms used for sortie generation. For bomber weapon groups, the number of excess weapons is equal to three times the number of weapons carried on a single bomber. For missile groups with a MIRV capability, the number of excess weapons (re-entry vehicles) is equal to twice the number carried on each booster plus ten percent of the total number in the weapon group.

Survival Before Launch Probability (SBL): In order to provide for efficient operation of the sortie generation phase, a few extra weapons are added to each bomber and MIRV missile weapon group (see Over-Allocation, this chapter). In order that the Plan Generator will perceive the correct number of expected weapons, the survival before launch probability (SBL) is modified to reflect this change.

If: NACTUAL = actual number of weapons in a group

NEXCESS = number of weapons added to the group

then:
$$SBL = SBLREAL* \left[\frac{NACTUAL}{NACTUAL + NEXCESS}\right]$$

The actual survival before launch probability (SBLREAL) is used after the excess weapons have been removed in the sortie generation phase of plan development.

Command and Control Reliability: Each weapon item in the data base is assigned to a command and control region (IREG) by the user. This command and control region is an arbitrary designation for the extent of command and control functions and has no geographic meaning. The reliability for command and control (CC) is a function of this region IREG. Thus, the user must divide the offensive weapons systems into these "regions" according to the command and control which is appropriate for the plan being developed. The maximum number of command and control regions is 20. The use of command and control reliability (CC) is discussed in the Weapon/Target Interaction section of this chapter.

Groups with Time-Dependent DBL: The aggregation of weapons into weapon groups is a straightforward process unless the weapons have a time-dependent destruction before launch probability (DBL). If the weapons do not have a time-dependent DBL, the DBL probability for all weapons assigned to the group is considered to be a weapon type characteristic. In this case, the DBL associated with the group is obtained directly from the data base. For weapons assigned a time-dependent DBL (this feature is recognized by a value of the attribute IDBL greater than zero) the program computes the time of the first and last launches from the group. The time of first launch is the appropriate delay time (alert or nonalert). The time of last launch is computed via the attribute DELTA which is the time between successive launches from the same base. (Each base in the data base may be assigned a value for DELTA.) Clearly, the time of last launch is equal to the time of first launch plus the product of DELTA and one less than the number of vehicles to be launched. Using the DBL data table specified by IDBL, the initial and final DBL probabilities are calculated. If the difference between these probabilities exceeds the user-input value for the maximum intragroup difference in DBL (DMAXDBL), the number of weapons in the group is reduced until this criterion is met. The DBL for the group is then computed as the average DBL of all the weapons in the group. The excess weapons removed from consideration because of the DBL difference are now considered for inclusion in the other groups or formation of a new group.

Time-Dependent Target Value: The relative value of the targets considered during plan generation is established on the basis of two sets of input data supplied by the user. In the data base each potential target is assigned a value (VAL) which establishes its relative worth within its assigned class. Then, the user provides data, for input to program PLANSET, which establish the target's value relative to all other potential targets in the game base (see Target Value, this chapter).

Since the relative strategic worth of a target may degrade over time (e.g., the value of a missile launch site before and after launch), the time dependence of target value must be considered in developing the attack plan. In QUICK, this relationship is established on the basis of data supplied by the user and included in the data base. The user can specify up to three separate time components which represent specified fractions of the total target value. For each of the three components, the user specifies the time (in hours) at which the value changes, T(I), and the fraction of the target value that is removed at that time, FVALT(I). (The latter factor is not specified for the third time component in the data base, since the sum of the values of FVALT(I) must be equal to 1.0.)

In using these data, the system (program PLANSET) automatically assumes a standard uncertainty in times specified. Figure 7 illustrates the relationship between the step function time dependence of target value

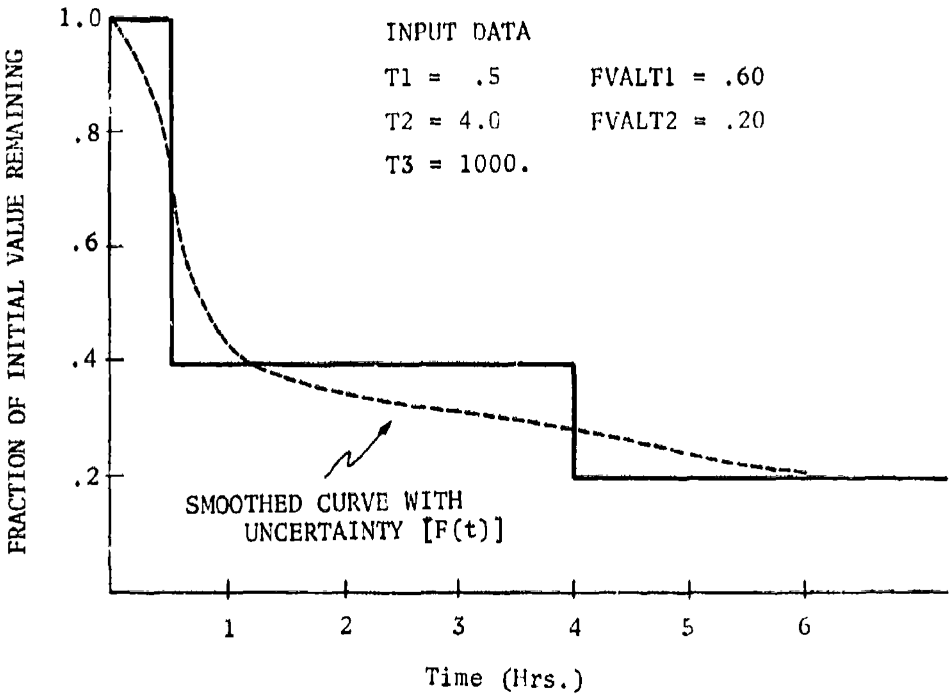


Fig. 7. Uncertainty Effects on Time-Dependent Target Value

accepted as input, and the smoothed time-dependent target values used to reflect uncertainty. If we define F(t) to be the fraction of target value that exists at time t (in hours), then a smoothing function can be defined that provides the uncertainty in target value. The specific smoothing function is given by:

$$F(t) = \sum_{I=1}^{I=3} \frac{FVALT(I)}{1 + [t/T(I)]^4}$$

The resultant total target value at the time of arrival is given simply by substituting the computed time of arrival in the above equation and summing over the separate time components. The dashed line in figure 7 is the curve which defines F(t) for a target with the specified input data. (The solid line is shown merely for reference.)

Complex Targets: The potential target list input to program ALOC reflects the complex target as a single element. The target attributes for this representative target, calculated in program PLANSET, are derived from the target data associated with the individual elements of the complex. The largest target radius associated with any element of the complex is assigned as the radius* (TGTRAD) of the representative target. Similarly,

---

*This technique represents an oversimplification. However, computing an adjusted radius, based on the geographic locations and dimensions of each target element, would not necessarily be an improvement. In order to provide a significantly more accurate treatment, a much more detailed analysis would be required of each complex target, which should take into account the yield and accuracy of the available weapons as well as the number, hardness, and geographic distribution of target elements.

the maximum value of TARDEF (local bomber defense potential) is assigned to represent the complex. The target value (VAL) and the number of terminal defense missile interceptors (NTINT) assigned each element are accumulated and their totals assigned to the representative element. MINKILL (the minimum kill probability required) and MAXKILL (the maximum kill probability desired) are weighted (by VAL) averages of the element attributes. The time dependence of the value of the complex, which is due to the time components of its elements, is approximated by at most three time components. That approximation is accomplished as follows.

First, the list of time components is checked for equal values. If any are found, the corresponding values are added together, and all but one of the equal components are removed from consideration, along with any zero components. If the number of remaining entries does not exceed three, the time dependence of the complex is approximated by these time components. Otherwise, an elimination procedure to reduce the number of entries to three is performed. For this, successive values (VALs) are accumulated (in V), and the average, weighted by VAL, of the corresponding time components is stored (in TAU). As successive entries are combined, all but the first are removed from consideration. When V becomes $\geq .35$ (VTOT), where VTOT is the total complex value, the component list is again collapsed. If more than three entries still remain, the above accumulation process is repeated, until no more than three time components remain.

Similarly, the hardness components (H1,H2) and the corresponding fractional value (FVALH1) which represent the complex are determined by first taking, for each target of the complex, its VAL, FVALH1, the number of hardness components (1 or 2), and the lethal radius corresponding to each hardness. The complement of FVALH1 is found to represent the second hardness component. If either fractional value is nonzero, it is multiplied by VAL to obtain the actual value at that hardness. After all targets have been considered, the lethal radii are separated into radii belonging to hard targets (radii less than 1.5 miles) and radii belonging to soft targets. The average lethal radius, weighted by the actual value at the corresponding hardness, is calculated from both hard and soft targets for those radii, and the result (HHARD or HSOFT) is assigned to the complex. Similarly, the actual value at each hardness (VHARD or VSOFT) is accumulated. If there are no hard targets (i.e., VHARD = 0), FVALH1 for the complex is set to 1; otherwise the fraction of actual value for hard targets (VHARD/VTOT) is assigned to FVALH1. This FVALH1, then, and the corresponding number of hardness components are assigned to the complex.

The index number (INDEXNO) and the target designator code (DESIG) associated with the representative target will normally be the INDEXNO and DESIG assigned to the first member of the complex (i.e., the first element of the complex encountered when processing the game data base). The user may, however, establish criteria for selecting the representative INDEXNO and DESIG (a control feature used in RISOP development). The procedures for exercising this option are presented in the User's Manual, (Program PLANSET).

WEAPON/TARGET INTERACTION

The quality of the plans, in terms of realism and sophistication, will be a direct reflection of the realism incorporated in the payoff function. In order to produce plans of maximum realism, the payoff function should reflect <u>all</u> the major factors that would be considered by an experienced military planner. The design incorporates:

1. Time of arrival of weapons
2. Time dependence of target values, which can reflect a planner's uncertainty in the time of arrival of weapons relative to change in target value
3. Weapon range limitations
4. Uncertainty in target vulnerability
5. Correlations in the effectiveness of weapons of similar nature reflecting such factors as reliability, DBL probability, and defense effectiveness.

To evaluate the capability of any weapon group against any target, program ALOC requires six basic numbers. These are:

SBL(G) The probability assumed that weapons in group G are not destroyed before launch

CC(KR) The assumed command and control reliability associated with the region for group G

REL(K) The assumed reliability for the weapon type K used by group G

PEX(G) The estimated penetration probability for weapons from group G to the target

STK(G,JH) The estimated kill probability of warheads in group G if delivered against the JH hardness component of the target

TVALTOA(G) The estimated target value at the time of weapon arrival for weapon from group G (this factor is computed from the time of arrival for a weapon from group G, TOA[G]).

These numbers reflect the planning factors the user has specified for the plan generation and do not necessarily reflect the values that the user specifies for the simulations. (See Chapter 2, Analytical Concepts and Techniques and Bomber and Tanker Events, and Missile Events, Analytical Manual, Volume III, Simulation and Data Output Subsystems.) The number is noted as "assumed" where it is a direct user input supplied in the data base. It is described as "estimated" where it is a derived quantity, based on other input data.

Actually, the numbers reflect only two types of information -- the time of arrival information, and the kill probability data. The single shot kill probability is simply a product of the first five items. The breakdown of the single shot kill probability into these five separate factors, however, is required in order to estimate correlations in delivery probability between several warheads delivered to the same target.

Most of the processing of weapon/target interactions deals with the six quantities given above. These quantities are then used in the calculation of weapon payoff.

The basic payoff calculation is modified by the inclusion of weapon correlation considerations. For each single weapon, four factors are calculated: the single shot kill probability and three auxiliary quantities required by the correlation model (see Weapon Correlations, this chapter).

If we define the overall single shot kill probability on one hardness component J as: SSK = REL * CC * SBL * PEX * STK

then MUP(G,J) = - LOGF(1.0 - SSK)

and SSIG(G,J) = MUP(G,J)/LOGF(SSK)

If the option to use the square root damage law is selected, MUP is defined in a different manner. It is defined so that:

$$(1.0 - SSK) = \left(1 + \sqrt{MUP(G,J)}\right) * \exp\left(-\sqrt{MUP(G,J)}\right)$$

The use of the square root damage function is further explained in a later section (see Multiple Weapon Attacks -- Square Root Law, this chapter).

MUP is in effect a measure of the effectiveness of the weapon against the specified hardness component. If all weapons were independent, the survival probability for the component with respect to multiple weapons IG would be simply:

$$\text{EXPF} -\left(\Sigma \text{ MUP(IG,J)}\right)$$

(This is called the exponential damage law.)

If the square root law option is selected, then the survival probability would be:

$$\left(1 + \sqrt{\Sigma \text{MUP(IG,J)}}\right) * \exp \left(- \sqrt{\Sigma \text{MUP(IG,J)}}\right)$$

The actual formula, using correlations, reduces to this form in the limit or no correlations but requires the array SSIG(G,J) as an auxiliary quantity.

Estimation of Correlation Factors, RISK(A,G,J)

The mathematics of the correlation calculation will be treated in detail below. Qualitatively, however, the technique requires an estimate of the extent to which the probability of failure for each weapon system is correlated with other weapons of the same class, type, alert status, etc.

The RISK array provides an estimate of this information. For any weapon system, the importance (or risk involved) in each failure mode (e.g., SBL, REL) can be represented in an additive form by taking the logarithm of the associated reliability. Thus, the total risk of failure for the weapon system -- LOGF(SSK) -- is given by: $\sum_{L=1}^{5} SM(L)$ where:

SM(1) = - LOGF(SBL)

SM(2) = - LOGF(CC)

SM(3) = - LOGF(REL)

SM(4) = - LOGF(PEX)

SM(5) = - LOGF(STK)

An array SMAT(A,L) is input by the user at the beginning of the allocation to provide a nominal estimate of the fraction of each risk SM(L) that is correlated with other weapons sharing each attribute A, where the attributes A represent:

A = 1	All weapons
A = 2	Weapons in the same group
A = 3	Weapons in the same region
A = 4	Weapons in the same class
A = 5	Weapons in the same type
A = 6	Weapons in the same alert status

For each weapon group G the RISK array by class, type, etc., is estimated (for each hardness component J) simply as:

$$RISK(A,G,J) = \sum_{L} SM(L)*SMAT(A,L)$$

This simple technique for considering weapon correlation is used because it is a reasonable representation of correlation and the allocations do not seem very sensitive to the details of the correlations. Additionally, input data for a more detailed representation would be difficult to develop.

Adaptability of Input Data

The foregoing three arrays are derived from the basic six variables listed earlier: SBL(G), CC(KR), REL(K), PEX(G), STK(G,JH), and TVALTOA(G).

The techniques used to calculate these six basic quantities allow a great deal of flexibility to adapt to new concepts in timing and penetration strategy. Thus it can be expected that the specific form of their computations will change as experience is gained in actual applications of the program.

The computations now in use illustrate both the factors involved and the flexibility that is available. We will now consider the present techniques for computing these six variables.

Planning Factors -- (SBL, CC, REL)

Two of the six (CC and REL) are contained directly in the data base. SBL is also in the data base -- except that the meaning there is probability of destruction before launch. To retain mathematical parallelism with other reliabilities, the SBL used here is defined as a probability of surviving and is obtained simply as (SBL = 1.0 - DBL). Obviously the specific value of DBL supplied in the data base should depend on both the alert status and the probability distribution of warning times for which the planner wishes to design the plan.

Evaluation of Value at Time of Arrival (TVALTOA(G))

The estimated target value at the time of weapon arrival for a weapon from group G, TVALTOA(G) is computed using the formula shown in the Time Dependent Target Value Subsection of the Planning Factor Processing Section of this chapter. TVALTOA(G) is equal to F(t) as calculated in the equation of that section, where t is the time of arrival of a weapon from group G, called TOA(G).

The time of arrival is computed differently depending on whether an initiative or a reactive plan is desired.

In the case of a reactive plan it is assumed that all weapon systems launch as soon as possible (subject to their specified delays) after a decision to launch is made. The time of arrival in this case is computed simply as FLIGHT TIME + DELAY where FLIGHT TIME = FLIGHT DISTANCE/SPEED and the delay is either the alert or nonalert delay (ALERTDLY or NALRTDLY) specified in the data base. For missiles, the flight distance is computed as the great circle distance from the weapon group centroid to the target. For aircraft, the distance is the sum of the great circle distances for each leg on the following path:

1. Weapon group centroid
2. Specified refueling area (if appropriate)
3. Entrance to chosen penetration corridor
4. All specified intermediate route points for the penetration corridor (if any)
5. Origin of penetration corridor*
6. Target.

In the case of buddy refueling or nonrefueling, the second point on the path is omitted. (Note that the times of arrival used at this point are approximate in that they use a constant nominal speed and, in the case of bombers, do not allow for excursions to other targets on the way.)

In the case of an initiative strike, the times of launch are coordinated to reduce warning time. This is accomplished by coordinating the plan relative to an assumed warning time. In the case of alert missiles, the user specifies (in the parameter CORMSL) what fraction of the flight time should have elapsed at the coordination time. With CORMSL = 1.0 all missiles impact at the coordination time. With CORMSL = 0.0 all missiles launch at the coordination time. This parameter is used in the weapon allocation phase The sortie generation phase, which

*Aircraft must fly to the origin of the corridor, but are not required to fly along the corridor axis to the corridor axis orientation point itself.

constructs the detailed plans, may use more sophisticated CORMSL data to achieve more highly coordinated missile attacks.

In the case of bombers, the user specifies (in the parameter CORBOMB) the remaining flight distance to the entrance of the penetration corridors at the coordination time. For alert vehicles, launch times are coordinated to make good this position at the coordination time -- except that no alert aircraft are held on the ground after the coordination time. The launch time and time of arrival for nonalert vehicles differ from that for the alert vehicles by just the difference in the alert and non-alert delays.

Penetration Probability (PEX)

The computation of this factor is discussed in the section entitled Missile/Bomber Defenses, this chapter.

Evaluation of Warhead Kill Probability (STK)

The warhead kill probability is estimated as follows.

The lethal radius H(J) for a one-megaton weapon against the J^{th} target hardness component is computed using the VN function in program PLANSET and is scaled to the actual yield using the 1/3 power yield-area scaling law. The kill probability is computed using the formula

$$P_R = 1- \left[\frac{\sigma_D^2}{\sigma_D^2 + \frac{1}{2W} \sigma_R^2} \right]^W$$

where $\sigma_R = H(J)$ and $\sigma_D^2 = \sigma_{CEP}{}^2 + \sigma_{Tgt}{}^2$, in which $\sigma_{CEP} = .8943*CEP$, $\sigma_{Tgt} = 2.448*R_{95} = 2.448*RADIUS$ (R_{95} = Radius containing .95 of total target value).

This kill function is computed from a very general actual-range/kill-probability law described in the Algorithms section, chapter 3. When the parameter W equals 3, sigma-30 damage curves are closely approximated, appropriate to soft targets (below 15 psi); for W equal to 6, sigma-20 curves are approximated, appropriate for hard targets. The use of these sigmas is inherent to the VN system as outlined in Physical Vulnerability Handbook -- Nuclear Weapons (U), Defense Intelligence Agency (CONFIDENTIAL).

For weapons restricted to targets in class NAVAL, this calculation is not performed. The value of the attribute PKNAV is used as the single shot kill probability. (Note that these weapons are identified by a value of PKNAV greater than zero.)

Multiple Weapon Attacks -- Square Root Law

When a number of weapons attack a single target, there are two ways to consider the total expected kill probability: the exponential (or power) law and the square root damage function.

The exponential, or power, law considers the total survival probability to be the product of the individual survival probabilities. This law is not as appropriate for area targets as for point targets. The user therefore has the option to use a square root damage function on area targets; i.e., targets with a radius greater than zero. The square root law operates as follows: For each weapon i, define a factor K_i as follows:

$$P_S = \exp\left(-\sqrt{K_i}\right) * \left(1 + \sqrt{K_i}\right)$$

where P_S = probability that target survives one weapon of type i. (This K_i factor is called MUP in this program.) If we have N_i weapons of type i, then the survival probability of the target, assuming independent weapons, is

$$P_{S,N_i} = \exp\left(-\sqrt{N_i K_i}\right) * \left(1 + \sqrt{N_i K_i}\right)$$

If we have N_j weapons of type j also allocated to the target, the survival probability, again assuming complete independence, is

$$P_{S,N_i,N_j} = \exp\left(-\sqrt{N_i K_i + N_j K_j}\right) * \left(1 + \sqrt{N_i K_i + N_j K_j}\right)$$

The weapons are not usually considered to be completely independent. Thus, the sums, $N_i K_i$ + ..., must be modified to consider interweapon correlations. The method of modifying this sum is discussed in the Weapon Correlations section of this chapter (also see chapter 3, Derivation of Formula for Correlations in Weapon Delivery Probability).

WEAPON CORRELATIONS

A basic consideration underlying the need for cross targeting is the existence of "shared risks" between weapons -- not only of the same type, but also between weapons of similar or related types. For example, if the enemy air defense is better than expected, the actual penetration probability of all bombers will be lower than that planned. If ballistic missile guidance systems prove to be operationally less accurate than expected, the target kill probability will be lower for all such missiles. These possibilities are illustrative of risks that are "shared" by large numbers of weapon systems. Cross targeting is intended to avoid "putting all eggs in one basket." It is designed to increase the probability that important targets will be destroyed even if most or all of the weapons with certain identical characteristics fail to perform as planned. Cross targeting recognizes the fact that operational percentages of success or failure for weapon systems cannot be predicted in advance.

The basic model used for cross-targeting analysis therefore recognizes that operational performance reliabilities are uncertain, and treats them as random variables. War plans are then developed on the assumption that the actual reliabilities that may be encountered in practice are unknown, and that they will in effect be selected at random for each weapon type from appropriate probability distributions. Moreover, it must be recognized that the reliabilities are not independently random for each weapon type, because certain risks are shared by many weapon types. Thus, on a specific Monte Carlo selection, when one success percentage is low, certain other percentages should tend to be low also. A satisfactory plan generation model also should be capable of considering these relationships between the success percentages for various weapon types.

To provide input data for the generation and evaluation of a cross-targeting plan, it is convenient to express these relationships in terms of risks that are shared in various degrees by similar weapon systems. The QUICK Plan Generator deals with five possible failure modes (table 5): survival before launch, launch or in-flight failure, command and control failure, penetration failure, and failure to kill the target even if delivered successfully. Each such failure mode can involve certain risks that are shared with similar weapons. For each such mode of failure, the user can specify the extent to which he feels risks will be of a type that are shared by all weapons of the same group, type, class, region, and alert status. Residual risks that are not specified to be shared in this way are treated as independent from weapon to weapon. Two weapons that share any attribute, such as type or alert status, can have a certain amount of shared risk. The failure correlation model used in the QUICK system considers each weapon to have seven attributes over which to distribute the effects of the five failure modes. Table 6 shows the seven weapon attributes.

Associated with the attributes and modes is a matrix which specifies the fraction of the risk in each mode that is shared by weapons with the

Table 5. Failure Modes

MNEMONIC	DESCRIPTION
SBL	Probability of survival before launch
CC	Reliability of command and control system
REL	Weapon system hardware reliability
PEN	Penetration probability
STK	Probability of target kill by warhead

Table 6. Weapon Attributes

NAME	DESCRIPTION
ALL	Shared by all weapons in the stockpile
ALERT	The alert status of the weapon, either alert or nonalert
CLASS	Weapon class, either bomber or missile
TYPE	Weapon type (e. g., B-52G, Poseidon)
REGION	Region of launch base
GROUP	Weapons of same class, type, region, and alert status whose launch bases are close to one another
INDEPENDENT	Shared by no two weapons in the stockpile

same attribute. This failure mode/attribute matrix, the SMAT array, defines the amount of risk shared by similar weapons and was referred to previously as the correlation array.

The entries in the matrix are the fraction of the risk of failure in the failure mode that is assumed to be shared by weapons with like attributes; e.g., class, type, region, and alert status. The sum of each row of the matrix must be 1.0. Two weapons in the same group that are identical with respect to all of these attributes will share identical risks except for the independent component. This array is used in the QUICK Plan Generator to compute weapon delivery probabilities and expected target damage when multiple weapons are assigned. The method for these computations is discussed in chapter 3.

Nature of Uncertainties

The basic objective of cross targeting (using more than one weapon type against a target) is to increase the probability that the target will be destroyed even if most or all of the weapons of any given type fail to operate as planned. In other words, the cross targeting is intended to hedge against the fact that the operational target kill probability for any weapon type is uncertain. In the conventional oversimplified calculation of expected target destruction, uncertainty in the percentage of targets destroyed is assumed to arise only as a consequence of the random selection of statistically independent individual weapon successes and failures (which are assumed to be drawn from an ensemble of known overall reliability). However, in realistic planning situations, these individual weapon-to-weapon statistical variations account for only a very small portion of the total uncertainty in the percentage of successes that will actually occur.

There are numerous other factors over and above this simple statistical variation that introduce uncertainty in the actual percentage of weapon successes. In the present model, all of these factors, regardless of their actual cause, are lumped as contributors to a single uncertainty which represents total uncertainty in each of the various planning factors. Thus, within the model, the overall uncertainty is divided into two separate parts. First, for each planning factor (such as in-flight reliability, launch reliability, penetration probability, or probability of surviving destruction before launch), the uncertainty is modeled by defining a probability distribution for the reliability factor. For any specific war game, the actual reliabilities are considered to be drawn at random from these distributions. After the random selection of these reliabilities, there still remains uncertainty in the actual success percentage. This second uncertainty derives from simple statistical fluctuations in the success percentages that occur when independent successes and failures are drawn from an ensemble of specified overall reliability. However, in realistic planning situations, this latter cause of uncertainty is usually relatively minor. The really serious uncertainties and, in particular, the uncertainties that give rise to the need for cross targeting, are above and beyond this simple statistical variability. The following are examples of some of these important factors that contribute to the uncertainty represented in the model by the probability distribution for each of the planning factors.

1. The enemy strategy and tactics are unknown and these can have major effects on the probability of penetration and the probability of destruction before launch both for individual weapon types and the force at large.
2. The basic system reliabilities in an operational environment may differ from those estimated in a test environment, and even the test environment reliabilities are not known exactly.
3. The actual success or failure percentages for one weapon may <u>physically</u> influence the success or failure probabilities of others--for example, in defense suppression attacks and in saturation tactics.

<u>Weapon Failure Modes and Target Survivability</u>

A programmed weapon can fail to destroy a target for a variety of reasons (failure modes) such as destruction before launch, launch failure, in-flight failure, penetration failure, or delivery inaccuracy. Assuming that these various failure modes are statistically independent, the overall reliability of the weapon h (from group i(h)) will be simply the product of the reliabilities over all the possible failure modes j;

$$R_h = \prod_j R_{i(h)j}$$

where

R_h = reliability for weapon h

$R_{i(h)j}$ = reliability for weapon h with respect to failure mode j

The target will survive the weapon h with probability

$$s_h = 1 - R_h = 1 - \prod_j R_{i(h)j}$$

Assuming for the moment that all weapons programmed against the target are statistically independent, the total probability of target survival is given by

$$S = \prod_h s_h = \prod_h \left(1 - \prod R_{i(h)j}\right)$$

In simplified analysis models where the reliability with regard to various modes of failure is assumed to be independent from weapon to weapon (i.e., where the operational reliabilities are assumed to be exactly predictable), this relation gives rise to a very simple law for target survivability with regard to multiple weapons. Specifically, relative to any target, one can define a single parameter X_h for each weapon h, where

$$X_h = -\ln s_h$$

The X_h in this equation can be thought of as a measure of the strength of the weapon against the target. The probability of target survival is then given by

$$S = \exp\left(-\sum_h X_h\right)$$

This relationship is widely used in military analysis work. It has the advantage that the effectiveness of weapons against a target can be measured in terms of a single additive quantity, and the efficiency of a weapon relative to its value can be measured simply by comparing this quantity, X_h, with the weapon cost or shadow value.

However, as soon as one admits the possibility of uncertainty in the reliability factors or of dependence of the reliabilities between weapon types, the simplicity of this relationship is lost. Since the X_h are related, a simple sum will no longer suffice to determine target survival. The incremental effectiveness of each weapon depends in part upon the other weapons which have been programmed against the target. It is no longer correct to increase the sum in the exponent as each weapon is added. The entire expression for target survival must be completely reevaluated with each weapon addition. Thus, the previous equation must be expanded to the form

$$S = \exp\left[\sum_h \ln\left(1 - \prod_h R_{i(h)j}\right)\right] = \prod_h \left(1 - \prod_h R_{i(h)j}\right)$$

The computational complexity of this expression for S in terms of the $R_{i(h)j}$, although undesirable, seems to be unavoidable in a practical cross-targeting model.

One obvious and superficially attractive way of avoiding the complexity, however, may require some comment. It has been suggested that the complexity can be avoided simply by considering the X_h as the random variables, and allowing the user to specify the statistical dependence between the X_h rather than the $R_{i(h)j}$. Unfortunately, because of the complex and unintuitive relationships between the X_h that result from mutually shared risks, this approach appears to place an impossible burden on the user.

A simple example will serve to illustrate this point. Consider two weapons, A and B, that share an identical risk of destruction before launch. Weapon A is otherwise completely reliable, and weapon B has numerous other more important failure modes. The small risk of prelaunch destruction is the only risk that prevents the X_h for the reliable weapon from being infinite. Thus, the destruction before launch risk completely determines the value of the X_h for the reliable weapon, but this same risk will have very little effect (even on a percentage basis) on the X_h for the less reliable weapon. Thus an identical shared risk produces grossly different effects on the X_h for the two weapons.

It seems clear that if a model is to successfully deal with the statistical dependence between weapons, the user must be permitted to express the relationships in terms of the sharing of risks, and the consequences in terms of the X_h must be derived by the model. It is unrealistic to expect the user to supply information directly in terms of the X_h, even though this might simplify the mathematics.

<u>Correlation Input Information</u>

The preparation of correlation information for the QUICK Plan Generator is simplified for the user through the use of a hidden variable approach. The specific hidden variables employed are generalized so that they can represent broad aggregations of risk elements. This has the advantage that a standardized set of risk elements can be used, and it is not necessary to redefine a new set of hidden variables for each application of the system.

For the purpose of dealing with these risks, the QUICK system classifies all possible ways a weapon can fail (to destroy its target) into the five generalized failure modes described previously.

Each weapon in the QUICK system is considered to be a member of a homogeneous group of weapons which are considered to be identical with regard to all parameters used in the development of a war plan. The "weapon group" in turn is categorized as being of a particular: Class (bomber or missile); Type (Minuteman, B-52, Polaris, etc.); Alert Status (alert or nonalert); and Command and Control Region. The various specific risk factors that can contribute to each of the five failure modes are also further classified as to whether they represent risks that might be shared in some degree: by all weapons of the same class; by all weapons of the same type; by weapons of the same alert status; or by weapons which share any other weapon attribute. Thus for each generalized failure mode, the QUICK system operates as if there is a hidden risk variable for each weapon attribute (see table 6, Weapon Attributes). By the conventions used in QUICK, the risks represented, for example, by the hidden random variable "Penetration Risk - Class Bomber" are available to be used only in the calculation of penetration risk for weapons that are members of the class "Bomber." Another risk variable is available to be used for penetration uncertainties by all weapons that are of class "Missile." If there are penetration risks that are relevant only for a subset of weapons within a class, there is another hidden variable for each type and even for each group that can be used.

The risk correlation information supplied for the QUICK system thus takes the following form. For each failure mode j and each weapon group i, an expected reliability $\bar{R}_{ij}$ is specified. The total risk, or variance, associated with this reliability factor is thought of as being divided into two parts, an independent risk and a shared risk. The shared risk is shared by all weapons in the group and is a result of the variance of the actual reliability R_{ij} relative to the expected reliability $\bar{R}_{ij}$. The remaining variance is identified as an "independent" risk which is completely independent from weapon to weapon in the group. The division of variance between "shared" and "independent" thus determines the width or uncertainty assumed by the Plan Generator for the probability distribution of R_{ij} relative to $\overline{R}_{ij}$. The larger the percentage of independent risk, the lower the uncertainty in R_{ij}.

The portion of the variance that is assumed to be shared within the group is then further subdivided into portions that are attributed to the hidden variables for weapons of that particular class, group, type, etc. Thus for each failure mode, the risk attribution required by the QUICK system consists simply of a specification of the portion of the total risk that is to be associated with each of a number of weapon attributes. Specifically, the user must specify the portion of the risk associated with each of the seven weapon characteristics previously described.

The summation of risk percentages attributed to each of the above factors must of course equal 100%. The following table illustrates a typical risk attribution array (SMAT) used as input to the QUICK system.

	ALL	GROUP	REGION	CLASS	TYPE	ALERT	INDEPENDENT
SBL	0	.10	.10	.40	.10	0	.30
CC	0	.20	.30	.10	.10	0	.30
REL	0	.05	0	.10	.20	0	.65
PEN	0	0	.10	.20	.20	0	.50
STK	0	0	0	0	0	0	1.00

The fact that 100% of the STK risk variable is treated as independent in this example implies zero uncertainty in STK; thus in this example we are ignoring any uncertainty in weapon yield or CEP. The choice of

.30 for the independent component of SBL as opposed to .65 for REL implies the assumption of greater relative uncertainty in any SBL reliability than is assumed in corresponding launch or in-flight reliabilities, REL.

Since, by definition, each row of this array must add to 1.0, the final column is obviously implied by the numbers in the other six columns. The actual input format for QUICK therefore omits the final column, so the correlation or risk attribution data are actually supplied in the form of a 5 X 6 array, known as SMAT. By convention, in supplying these data for QUICK, the array is normally filled with numbers intended to represent the maximum amount of uncertainty or shared variance that it seems reasonable to consider.

One other important simplifying assumption is made concerning the risk attribution data supplied. In principle, one might think that the user would like to specify different risk attributions by class, type, alert status, etc., for every individual weapon group. This approach would provide maximum flexibility to control the factor weightings for each group, but it would require a separate SMAT array for each of the groups used (up to 200) in the QUICK system. To avoid this data burden, the QUICK system actually uses only one SMAT array and the values used in the array are chosen to be a reasonably good compromise for all weapon groups.

For missiles with a MIRV capability, a different weapon correlation array is created. The user specifies what fraction of the variance attributed to the INDEPENDENT attribute is to be added to the variance attributed to the GROUP attribute for all MIRV groups. This specification has the effect of increasing intragroup correlations for these groups. Since this increased correlation is applicable only to those events which precede booster burnout, only the failure modes which affect the booster are modified. These modes are survival before launch (SBL), command and control reliability (CC), and weapon system reliability (REL). Two SMAT arrays are stored, one for MIRV groups and one for non-MIRV groups. As each group is processed, the appropriate array is used in computing weapon/target interaction parameters.

WEAPON ALLOCATION

Program ALOC allocates weapons over the specified target system, using input data concerning the structure of the target system, the inventory and capabilities of available forces, and the war objectives and strategy. It produces as output a detailed specification of the weapons assigned to each target.

The structure of the target system is represented by the location, value, and estimated vulnerability and defense capability of each target element. The available forces are represented by such factors as range, yield, accuracy, reliability, penetration parameters, response time, speed, survivability, location of deployment, and inventory.

The allocator (ALOC) uses generalized Lagrange multiplier optimization techniques. With this approach, it is practical to use comparatively detailed payoff functions reflecting realistic uncertainties and planning contingencies that are usually ignored in automatically generated plans. The approach provides sufficient flexibility to include targeting objectives and constraints which may not have been foreseen in the original formulation of the payoff function.

The objectives and strategy reflected by the plan will be determined by:

- The relative values assigned to various elements of the target system, and the time dependence (if any) of these values
- Any minimum required kill probabilities which may be specified for particular targets or groups of targets
- The portion of the available force specified (such specification is optional) for allocation*

*The same types of information are used to control the resources allocated for defense suppression. In principle, the allocation of effort to defense suppression targets should be chosen to maximize the destruction of other elements of the target system -- and should follow as an automatic consequence of the values assigned to these other targets. However, such a fully automatic treatment of defense suppression is beyond the present state-of-the-art. Consequently, the user must specify equivalent values or required kill probabilities for defense suppression as well as primary targets.

The realism and sophistication of the plans produced by such an optimization depend in large measure on how completely the intended objectives (with realistic contingency or uncertainty considerations) are reflected in the payoff function. The design objective has been to provide the flexibility needed for any reasonable payoff function. Some of the factors included in the payoff function by the QUICK Allocator are:

1. The time dependence of target values
2. The uncertainties in target vulnerability
3. Correlations in delivery probability between weapons which share the same uncertainties of accuracy, reliability, penetration probability, and weapon survivability (for the second-strike applications)
4. The uncertainty in target value and time dependence -- as a consequence of the unpredictability of enemy actions
5. Uncertainty in the level of ABM interceptors defending the target.

In addition, program ALOC computes the marginal value of each weapon allocation. This value (RVAL), whose calculation is described in the Basic Sortie Generation section of this chapter, is used in the sortie generation process to determine the worth of including a target in a sortie.

Concept of Operation

The efficient targeting of a limited inventory of weapons is a combinatorial problem primarily because of inventory constraints. The fact that weapons used against one target are not available for others introduces a resource interaction between targets that are otherwise independent. The Lagrange optimization technique provides an exact representation of this interaction, which permits the allocation of weapons to be accomplished one target at a time. In the Lagrange technique, the detailed resource interaction is represented by a single "price" or value established for each type or group of weapons. This "price" represents the value of the weapons in each group in relation to the specific requirements and objectives of each war plan. This "price" (or Lagrange multiplier) corresponds to the minimum payoff (in target value destroyed) that will justify the use of the weapon.

The QUICK Allocator utilizes a resource allocation technique published in Operations Research* which permits the application of Lagrange multipliers to discontinuous or nondifferentiable functions (such as the payoff targeting problems).

As applied to the targeting problem, the technique consists of assigning a trial "weapon price" for each "group" of weapons in the

*H. Everett III, "Generalized Lagrange Multiplier Method for Solving Problems of Optimum Allocation of Resources," Operations Research, Vol. II, No. 3, May-June 1963. p. 399-417. For ease of reference, an excerpt from this publication is contained in appendix B.

inventory to be allocated. (A "group" is defined here as a set of weapons which are so nearly identical both in characteristics and location that no distinction between them is necessary during the allocation.) The attacker's "profit" on each target is then defined as the target value destroyed minus the total "price" of the weapon or weapons expended. Weapons are allocated against any target in such a way that this "profit" is maximized. (When the allocation against any target is complete, there are no weapons in the total inventory which could achieve an added payoff on the target in excess of their assigned "weapon price." Also, there are no weapons actually assigned to the target which do not achieve a payoff in excess of their assigned "weapon price.")

If the allocation were carried out this way for all targets, a certain total number of weapons from each group would be assigned. This number could be more or less than the actual inventory available. However, the resulting allocation would be a true optimum allocation for a hypothetical stockpile consisting of the weapons actually used in this allocation. If the number of weapons allocated from any group were larger than the actual group inventory, then the trial "weapon price" is too low, and the use of these weapons should be limited to those places where a higher return is achieved. If too few were allocated, the trial "price" is too high, and the weapons could be fruitfully employed where the payoff is somewhat less.

The trial "weapon prices" could then be adjusted accordingly and a new allocation could be carried out until a satisfactory approximation to the actual inventory is achieved. Many iterations throughout the target list would thus be required to establish the correct prices which would cause the desired stockpile to be consumed.

In the QUICK Allocator, the basic process described above is speeded up in several ways:

1. The targets are processed in a random order, so that serious errors in the initial trial "weapon prices" are detected promptly and are corrected by observing the rate of allocation for each group of weapons. Thus, it is not necessary to carry an allocation to completion before correcting the trial "weapon prices."
2. Initial allocation rates are monitored for aggregated categories of weapons (i.e., weapons which share identical attributes), rather than individual groups. Thus, statistically useful information on the allocation rates is obtained from small samples of targets, and corrections are applied to the "weapon prices" for all the weapon groups within the aggregated categories.
3. Ordinarily, in such a process, it would be difficult to estimate the size of the error in the "weapon prices" from the size of the

error in the allocation rates. For example, a trivial difference in "weapon prices" between essentially identical weapons could cause the one with the lower "weapon price" to be used to the complete exclusion of the other. The QUICK Allocator therefore incorporates a small "premium" which prevents such large and unnecessary deviations from the desired allocation rates, where the difference in profit is small. With the premium, a large error in the allocation rates can occur only if the error in prices is substantial. In this way, the magnitude of the error in the "weapon prices" can be estimated from the allocation rates, and corrections of the proper size in the "weapon prices" can be efficiently made.

4. The iteration process in trial "weapon prices" is terminated when "weapon prices" are approximately correct (typically within a few percent) even though the resulting allocation does not accurately fit the available stockpile. The allocation is then adjusted to fit the stockpile by removing weapons excessively allocated and substituting weapons underallocated. This adjustment of the allocation is done by adjusting the "premiums" in the closing phase in such a way that the loss in "profit" is kept as small as practical. It has been mathematically proven in the preceding reference that the <u>payoff</u> for the resulting allocation will not be degraded by this closing phase by more than the <u>observed</u> loss of "profit."

This final approximation technique provides a powerful method for converging rapidly on war plans which are near optimum. The extent of the observed loss of "profit" provides a valuable gauge of the efficiency of any such approximation. (If a rigorous bound on deviations from optimality is desired, it can be obtained by a final pass over the target list in which all premiums are removed.)

Adjustment of Multipliers

To understand the operations of the allocator (program ALOC), it is helpful to think of the set of all targets arranged in random order around a circle. Processing will continue for several "passes" around the circle until the multipliers have converged to acceptable values, and the weapon stockpile constraints are met. To start the process, initial values for the multipliers (i.e., "weapon prices") are selected, and an initial pseudo allocation is made in which the weapons are distributed uniformly (without regard for integer weapon constraints) over the target set. Thus, in the beginning it appears that weapons have been allocated at exactly the right rate. As each new target is encountered, the pseudo allocation is removed, and actual trial allocation is made using the current values of the multipliers. Since the initial multipliers are not correct, this gradually produces an error in the estimated rate of allocation. This error is then used to determine how to adjust the Lagrange multipliers. Of course, statistically significant information on errors in the allocation rates becomes available most quickly for those groups where the number of weapons is large. To accelerate the adjustment of the multipliers, ALOC monitors the allocation rates for large collections of weapons (i.e., weapons which share weapon attributes, see table 6) which include many groups. When it is observed that the overall allocation rate for such a collection is in error, the Lagrange multipliers for all the groups involved are adjusted simultaneously. To simplify this, the Lagrange multiplier, LAM(G), for each individual group of weapons is expressed as a product of collective "local multipliers," LA(J). Specifically, the Lagrange multiplier for a group of weapons is represented as the product of the local multipliers for all weapons; all weapons of the same class; the same type; the same region; the same alert status; and a final local multiplier unique to the specific group; i.e.,

$$LAM(G) = LA\left(J_{all}\right) * LA\left(J_{class}\right) * LA\left(J_{reg}\right) * LA\left(J_{alert}\right) * LA\left(J_{group}\right)$$

(To facilitate this bookkeeping, an index table is maintained for each weapon group which specifies these local multipliers.)

The concept for monitoring the allocation rates is as follows. If there are a total of NTGTS targets, and the total number of weapons in a particular collection of weapons indexed by J (e.g., J_{all}, J_{class}) is NOWPS(J), then the expected number of these weapons to allocate per target is just

Expected Rate = NOWPS(J)/NTGTS

If the observed rate is less, the associated multiplier LA(J) should be lowered; if it is greater, it should be raised.

Particularly during the early phase of the allocation, when the Lagrange multipliers ("weapon prices") are changing rapidly, the allocation rate will also change rapidly. Thus, in evaluating the allocation rate, it is appropriate to place more weight on the allocation rate for more recently processed targets. The estimators of allocation rate used by the allocator, therefore, allow a variable weight to be assigned to the targets. The estimated allocation rate R for any collection of weapons J is computed as follows:

$$R(J) = \frac{\sum_i \left(N(i,J) * W(i) \right)}{\sum_i W(i)} = \frac{RUNSUM}{WTSUM}$$

where W(i) is the weight assigned to the i^{th} target* and N(i,j) is the number of weapons from the collection J assigned to the i^{th} target. The summation is always taken over all targets. However, in the early stages of the allocation, the weight attached to each successive target is increased quite rapidly, so that the estimated allocation rate is determined almost entirely by the most recently processed 10 to 20 targets. As the Lagrange multipliers come closer to correct values, the target weights are increased more slowly and the allocation rate, in

*Target weight is initialized at 1.0 and modified during processing, as described in Chapter 3, Calculations, Lagrange Multiplier Adjustment.

effect, is averaged over a larger number of targets. Ultimately, the weight attached to succeeding targets is held fixed. Obviously, after all targets have been processed with identical weights, the above estimator of the allocation rate becomes an exact measure of the average allocation rate and if multiplied by the number of targets would give the exact number of weapons on all targets. Thus, the same estimating machinery can be used in the final stage of the allocation as a guide in converging to the exact stockpile.

Actually, for each collection of weapons J, three separate estimators of the allocation rate are maintained. These estimators differ in the rate of change of the target weights that are used in computing the estimates. In effect, they correspond to averaging the allocation rate over different numbers of targets. The algorithm requires that all three estimates provide the same sign of the estimated error rate before it will change the value of the Lagrange multipliers. This feature provides a conservative approach to changes in the multipliers and reduces the chance of overcorrecting.

The allocation process evaluates its own progress in converging the multipliers and determines when to terminate the process. The variable which reflects this evaluation is called PROGRESS. PROGRESS is an arbitrary variable set internally by program ALOC to monitor the allocation state. The values 0, .4, .5, .75, 1.0, and 2.0 are arbitrarily assigned by the program according to procedures specified in Chapter 3, Calculations, Multiplier Adjustment. Qualitatively, the PROGRESS states are as follows:

1. Progress = 0 This is the initial state. Its main purpose is to prevent the allocator from terminating very quickly because the pseudo allocation seems satisfactory.
2. PROGRESS = .4 This state indicates that the estimated allocation rates reflect primarily the actual rather than the pseudo allocation.
3. PROGRESS = .5 From this point on, the rate of change of the target weight is not permitted to increase -- i.e., the allocation estimators are required to move monotonically toward the state where all targets are weighted equally.
4. PROGRESS = .75 Target weights have stopped increasing -- multipliers are assumed to be nearly stable.
5. PROGRESS = 1.00 This occurs only after at least one full pass of the target set with PROGRESS = .75. At this point the multipliers are frozen, and the premium (see below) for meeting the exact allocation is gradually increased. During this phase, multiple targets previously allocated as a unit may be split to receive independent allocations, if this will aid in meeting stockpile constraints.
6. PROGRESS = 2.00 Allocation is complete. Three options for further processing are provided depending on value of IVERIFY supplied by user.

IVERIFY [illegible] Current allocation simply transferred to normal output file, and process halts.

IVERIFY = 1 Allocation transferred as above, but a verification allocation (not recorded on file) is made to obtain a bound on the maximum theoretical payoff if convergence had been continued indefinitely.

IVERIFY = 2 Allocation transferred as above but the current allocation is reevaluated assuming a revised value of the correlation factor which is user-input at the start of the run (CORR2).

The details of multiplier adjustment are contained in Chapter 3, Calculations, Lagrange Multiplier Adjustment.

Closing Factors -- Premiums

The Lagrange multiplier for each weapon is modified by a premium. This factor is used to force closure of weapon allocations to the available stockpile. It acts as a bonus for using under-allocated weapons and a penalty for using over-allocated weapons. The parameters which are used to calculate the premiums are:

SURPWP(G) An estimate of the number of surplus (or un-allocated weapons) in the group. This number is based on estimated allocation rates in the early phase and the actual allocation later.

NWPNS(G)	The actual number of weapons in group G.
CTMULT	The current multiplicity of the target being processed.
LAMEF(G)	The Lagrange multiplier for the group.

The premium depends also on three control parameters: PROGRESS, PRM, and CLOSE.*

The effect of PROGRESS (described earlier) is as follows:

1. If PROGRESS is greater than 1.0, this indicates that a verification allocation is desired to obtain a theoretical upper bound on the payoff without regard to meeting the actual stockpile constraints. For this purpose, the premiums are simply set to zero.
2. If PROGRESS is less than 1.0, a small premium is computed which is intended only to avoid large deviations from the desired allocation rate of small errors in the Lagrange multipliers. (Otherwise, a trival change in the multipliers for two competing weapons could result in a complete change from always allocating one to always allocating the other.)
3. If PROGRESS is equal to 1.0, this is a signal that the closing phase has been reached and the object is to close in on an exact allocation of the available weapons. In this case, a larger step function premium is computed, and the size of the step function is gradually increased until final closure occurs.

---

*PROGRESS is set internally by the program as described in Chapter 3, Calculations, Multiplier Adjustment. PRM and CLOSE are user-input parameters.

During the early allocation phase, superimposed on the actual payoff is a small negative quantity (called a premium) that is proportional to the value of each weapon group and quadratic in the size of the error in allocation. In effect, the actual payoff, H(X), for any allocation, X, is adjusted to $H(X)_{ADJ}$:

$$H(X)-PRM*\sum_{G}\left\{NWPNS(G)*LAMEF(G)*\left(\frac{SURPWP(G)}{NWPNS(G)}\right)^{2}\right\}$$

This quadratic addition to the payoff function has the effect of introducing a preference for allocations where the absolute value of SURPWP is small.

The addition or deletion of a weapon from group G will give rise to a difference in SURPWP equal to the current target multiplicity. Thus, the change in this quantity (per unit multiplicity) with the addition of a weapon G is:

$$PREMIUM(G)=PRM*LAMEF(G)*\frac{SURPWP(G)\ -\ .5*CTMULT}{NWPNS(G)}$$

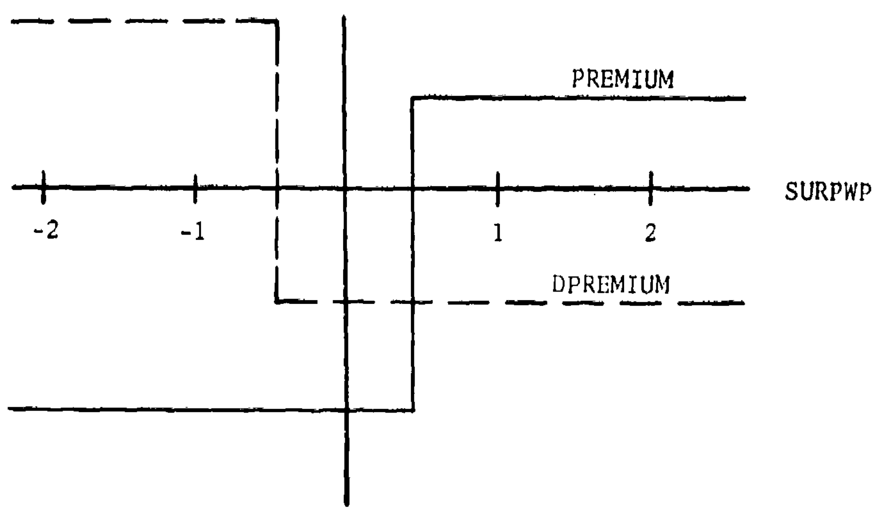
and the change with deletion of a weapon is:

$$DPREMIUM(G)=PRM*LAMEF(G)*\frac{-SURPWP(G)\ -.5*CTMULT}{NWPNS(G)}$$

The value of PRM is a user-input parameter. The value should be less than 1.0. Otherwise, in cases when no weapons from some group have been used, the premium for allocation of a weapon could exceed the cost of the weapon LAMEF(G) and weapons could be allocated even if the payoff were zero or even negative. Experience has shown that values between .5 and .9 work very well.

When PROGRESS reaches 1.0, PRM is set to .9 by the program to accelerate convergence. In addition, a small step function is added.

The following sketch illustrates the value of these step function premiums as a function of their SURPWP:



Notice that when SURPWP is in the desired area, that is |SURPWP|<.5, the premiums for either addition or deletion of a weapon are negative, making the current allocation seem most desirable. If there is a surplus of weapons (right side of figure), the premium for addition is positive, and the premium for deletion is negative. In the limit, if closure is long delayed, these premiums approach the value of the weapons. In this limit unallocated weapons seem free. The formula for these premiums is approximately:* LAMEF(G)*[1.0 - 1.0/CLOSE] where CLOSE starts at 1.0 and gets larger geometrically. The adjustment of CLOSE is controlled by another user-input parameter. CLOSE is adjusted linearly at a rate such that at the end of one pass it will have increased by the amount CLOSER (which is also a user-input parameter).

On the left-hand side of the figure, where weapons are over-allocated, the premium for deletion is positive and the premium for addition is negative. These premiums can grow large without limit to provide incentive if necessary to remove a weapon from a very attractive target. The formula for these premiums is: LAMEF(G)*(CLOSE - 1).

Whereas the first set of premiums is linear and can be thought of as representing a negative quadratic addition to the payoff, these premiums are a step function and can be thought of as an upside down "V"-shaped addition to the payoff, which will strongly favor allocations that exactly match the stockpile.

Actually, it has been found desirable to add a very small quantity equal to 1/2 the smallest value of LAMEF(G) for any G multiplied by (CLOSE - 1.0). This provides an incentive for [SMALLAM(CLOSE - 1.0)] using weapons with very low marginal value even if the payoff is essentially zero.

Single Target Allocation -- Targets Without Terminal Ballistic Missile Defenses

The problem is to select the best combination of weapons against each target as the targets are processed. The problem therefore is really a combinatorial problem. However, to calculate the payoff for all possible combinations of weapons and then select the best on each target would clearly be impossible. Consequently the method approaches the problem by adding one weapon at a time. After a weapon is added, the program estimates the additional payoff to be obtained by adding or, where relevant, deleting one weapon from any one of the available weapon groups. A decision must then be made whether to terminate the allocation or whether to add or delete additional weapons. In its effort to maximize profit, the program operates initially on a form of steepest ascent basis. That is, it selects those weapons which provide the highest payoff per unit cost. It also removes any weapon which shows a negative profit after other weapons are added. There is a constraint, however, that every weapon on target destroy a minimum fraction of the target's original value. This minimum fraction is read in with the other control data. Ultimately it works solely on the basis of marginal profit and seeks any change in the allocation that will increase the profit.

Thus in effect the program needs to know the marginal profit for a potential weapon, the efficiency or payoff per unit cost, and the marginal profit of each weapon already on the target so that weapons which become unprofitable after others are added can be recognized.

The data required for these decisions are:

VT	The current surviving target value
VTP(G)	The potential surviving target value if a weapon from group G were added
VTD(N)	The potential surviving target value if the N^{th} weapon now on the target were deleted.

The inputs required for their calculation include:

PREMIUM(G)	The current premium for adding a weapon from group G to the target
DPREMIUM(G)	The current premium for deleting from the target a weapon from group G together with the Lagrange multiplier
LAMEF(G)	The current Lagrange multiplier or cost associated with the utilization of a weapon from each group.

Using these input arrays, the program computes the potential "BENEFIT" associated with the addition of a weapon from any of the weapon groups. The BENEFIT is interpreted simply as the payoff plus the premium; i.e., for potential weapons, BENEFIT = VT-VTP+PREMIUM. Similarly, for each weapon that might be deleted, there is computed the BENEFIT that would be lost if the weapon were deleted, BENEFIT = VTD-VT-DPREMIUM. Notice that if the premiums are small (as they usually are) the benefit is essentially the same as the payoff. It is, therefore, convenient to think of the BENEFIT as simply a modified payoff that is to be maximized. The PREMIUM is added simply to speed the convergence to the desired stockpile.

The program scans the potential BENEFIT associated with all weapon groups that might be added and finds that group IPPMX for which the "modified potential profit," PP, is greatest; i.e., PPMX, PP = BENEFIT - LAMEF.

Similarly it reports the group IPVRMX for which the "efficiency," PVR is greatest, PVRMX. The "efficiency" is here interpreted as the rate of BENEFIT per unit cost; i.e., PVR = BENEFIT/LAMEF. (It is necessary for the single target allocator to know the "efficiency" of alternative weapons. If it were guided only by "profit" (i.e., (BENEFIT - LAMEF), it would always select those individual weapons showing the largest profit, whereas it is often better (especially on very valuable targets) to select several less costly weapons so long as the benefit <u>per unit cost</u> is higher.)

Finally, the program scans all weapons, already on the target, to determine which weapon IDPMN shows the smallest DPMN marginal modified profit DP where DP = BENEFIT - LAMEF.

These quantities:

VALUE	INDEX	DEFINITION
PPMX	IPPMX	Maximum potential profit
PVRMX	IPVRMX	Maximum potential efficiency
DPMN	IDPMN	Minimum current marginal profit

constitute the primary input for determination of weapon allocation on single targets. Their calculation is modified, however, by the minimum and maximum damage constraints placed on each target. MINKILL is the minimum required damage level. MAXKILL is the maximum desired damage level. MAXCOST is the maximum factor by which value may be multiplied to obtain MINKILL (these three factors are established in the data base: MAXKILL and MINKILL are defined as attributes; MAXCOST is set equal to the attribute MAXFRACV). MINDAMAG, a program user-input parameter, is the minimum fraction of damage required from an individual weapon.

To implement the MINKILL and MAXKILL responsibility, the VT, VTP, and VTD are replaced by effective values VTEF, VTPEF, VTPEF, and VTDEF. The relationships are:

VTEF = ALPHA*MAX1F(VT,VTMIN)

VTPEF = ALPHA*MAX1F(VTP,VTMIN)

VTDEF = ALPHA*MAX1F(VTD,VTMIN)

(Note: MAX1F implies "Maximum of")

where: VTMIN = VTD*(1.0 - MAXKILL)

ALPHA = Local control variable defined below.

If neither MINKILL nor MAXKILL has been explicitly specified for the target then the default values apply (ALPHA=1.0 and VTMIN=0.00) and the effective values of VT, VTP, and VTD are identical with the actual values. If MAXKILL has been specified as less than 1.0, it implies there is no value in reducing the target value below VTMIN. This point of view is built into the payoffs simply by not allowing the effective value to reflect any surviving target value less than VTMIN.

The variable ALPHA is increased above 1.0 when necessary to motivate the algorithm to achieve the specified MINKILL (minimum acceptable fraction of expected value destroyed). A quantity VTMAX is defined

VTMAX = VTO*(1.0 - MINKILL)

which reflects the largest acceptable expected surviving target value. If the computed surviving target value VT exceeds VTMAX, and at the same time the output does not show any additional potentially profitable weapons, then the process will not terminate immediately. It will instead increase the value of ALPHA above 1.0 by whatever factor necessary to make at least one more weapon seem profitable. It then recycles and reevaluates all the output parameters. Since ALPHA multiplies all the target values, increasing ALPHA is equivalent to increasing the value of the target until more weapons can be justified against it. Once the value has been raised so that the required kill is achieved, ALPHA remains fixed (for this pass) during the remainder of the allocation to the target, so that the program automatically proceeds to do a complete optimum allocation for the revised target value.

There is a protection feature MAXCOST that is designed to prevent excessive waste of warheads against a target where it is simply not practical to achieve the prescribed destructive level required by MINKILL. If the current cost (of the allocation to the target) divided by the total target value already exceeds the ratio prescribed by MAXCOST, the value of ALPHA will not be increased any further. For the same reason, if it is necessary to raise the target value by a factor of 100 or more to justify the specified MINKILL, the ALPHA will not be further increased.

Experience with the allocator has shown that if the efficiency PVR is used in its pure form, PVR = BENEFIT/LAMEF, the program will sometimes arrive at its allocation in a very inefficient way. What happens is that during the initial laydown of weapons on the target it will use large numbers of very cheap but not very effective weapons. Then as soon as a more efficient weapon is used, the target value is drastically reduced and many of the weapons initially allocated cease to be worthwhile and have to be removed. Consequently, the program now incorporates a revised version of the efficiency PVR'. This is defined as follows:

$$PVR' = \begin{cases} PVR & \text{if } PP < 0 \\ \left[1.0 + \frac{PP}{LAMEF} * \frac{1 + \gamma(VTEF/(VTEF-PP+PREMIUM))}{1 + \gamma}\right] & \text{if } PP \geq 0 \end{cases}$$

If γ is zero this gives the pure value of PVR. However if γ is set above zero, as it usually is, then the value of PVR will reflect the magnitude of the profit as well as the efficiency. (This coefficient, γ, is a user-input parameter.) Notice that as the potential profit PP becomes comparable to the remaining target values, the coefficient of γ in the numerator becomes large and PRV' is increased above PVR. In the limit where the potential profit PP is negligible relative to the remaining target value VTEF, PVR' is equal to PVR. The single target weapon allocation procedure consists of three parts:

1. A set-up and single weapon allocation phase
2. A multiple weapon laydown loop
3. A multiple weapon refinement loop.

The initial laydown operations are handled using the "efficiency" as the criterion for selecting weapons. This is necessary because if the "profit" were used at this stage, effective individual weapons which could produce a large single weapon profit would always be selected in preference to less effective but less expensive weapons where two or three such weapons added in succession might provide a better payoff at lower cost. However, before exiting from the routine, provision is made to test the allocation to determine whether a higher total profit is possible. So, the final refinement of the allocation is always done using total "profit" as the criterion.

An immediate exit is made if there are no potential weapons that show a profit. Otherwise, the weapon which shows the highest "efficiency" is added. A test is then made to determine whether more weapons are needed on the target. If so, control passes to the multiple weapon laydown loop. If not, it is clear that a single weapon allocation is needed. In this case, if the single "efficient" weapon just tested is not also the most profitable weapon, then it is removed and replaced with the most "profitable" single weapon before exiting from the routine.

On the other hand, if several weapons are indicated, the multiple weapon laydown loop takes over. This loop simply keeps adding the most efficient next weapon until there are no more potential weapons that show a profit; i.e., have an efficiency greater than one. (For a profitable weapon, (BENEFIT/COST) must exceed 1.0.) As new weapons are added, however, it often occurs that some of the old weapons cease to be profitable; provision is therefore made to remove any unprofitable weapons after each new weapon is added. When this part of the process is complete, all weapons on the target must be "profitable" and there must be no potential weapons that would show a profit if added.

At this point, there is a remote possibility that there is again only on weapon in the allocation. If so, it is replaced with the most profitable single weapon. Otherwise, control passes to the allocation refinement loop.

Basically, the allocation refinement loop is intended to start back with the first weapon placed on the target and successively remove each weapon to determine if there is any more profitable weapon that can be substituted. If, in each case, the same weapon proves to be the most profitable the allocation is considered complete. If, in any case, a substitution occurs, the testing of the other warheads starts over again from that weapon until all weapons on the target have been tested.

It is possible during this process, as in the preceding loop, that as more profitable weapons are substituted, some of the other weapons that formerly were profitable will cease to be so. Therefore, after each weapon is added, a check is made and any unprofitable weapons are deleted. If such deletion leaves a situation where some other weapon would be profitable, it is immediately added before re-entering the testing loop. Any such change that interrupts the testing process requires that the testing start over again. To avoid unnecessary operations, the pointer which selects successive weapons to be deleted for testing is set to skip over weapons which are from a weapon group that has already been tested.

Single Target Allocation -- Targets With Terminal Ballistic Missile Defenses

The allocator (program ALOC) considers two possibilities for targets with terminal BMD. It first attempts a leakage attack. A force, possibly mixed between bombers and missiles, is allocated without trying to exhaust the missile defense. Any bomber or missile weapons that leak through their respective terminal defenses are considered in evaluating damage. Second, the allocator attempts an exhaustion attack. A force of missiles large enough to exhaust the terminal missile interceptors is allocated. After exhaustion of the defenses, missiles are added until the damage done by each incremental missile is less than the value of the Lagrange multiplier for that missile. The profit from these two attacks is compared and the more profitable allocation is chosen.

The rate of return for a missile against a target with terminal BMD is defined as follows:

RATE = (VT - VTDX)/(LAMEF + PREMIUM)

- VT = Surviving target value prior to latest allocation
- VTDX = Surviving target value including latest allocation
- LAMEF = Lagrange multiplier
- PREMIUM = Bonus for allocation (see Closing Factors, above).

The surviving target value VTDX is computed as follows. Let PWK be the probability of warhead kill by the terminal defense (PKTX in Bomber and Missile Defenses, above).

Define SSSP(G,J) = Single shot survival probability of the target from group G on hardness component J

NOWEP(G) = Number of weapons allocated from group G

VTOA(N1,J) = Value of target hardness component J at time of arrival index N1

S(G,J) = Probability that target component J survives attack of NOWEP(G) weapons from group G

NWHD(G) = Number of warheads per weapon in group G

NN = Number of weapon groups

M = Number of hardness components

Set: VTOA(0,J) = VO(J) = original value of component J

VTOA(NN+1,J) = 0

Then: S(G,J)=(SSSP(G,J)+PWK-PWK*SSSP(G,J))(NWHD(G)*NOWEP(G))

If the weapons are ordered by increasing time of arrival, then

$$VTDX = \sum_{J=1}^{M} \sum_{L=0}^{NN} \left[VTOA(L,J) - VTOA(L+1,J)\right] * \prod_{G=1}^{L} S(G,J)$$

The innermost sum over L must be carried out in order of weapon time of arrival.

Since the payoff function for a defended target is generally not concave, one cannot look at only the rate of return of the next missile to determine whether the target is to be attacked. Rather, it is necessary to allocate weapons beyond the exhaustion point and then search for that allocation which yields the highest average rate of return. If this average rate is greater than one (i.e., a profit is realized by attacking the defended target), then the allocation can actually proceed.

The missile allocation proceeds as follows. First, those missiles with the cheapest terminal objects (warheads and terminal decoys) are allocated until the terminal interceptors are exhausted. Then, each missile type in turn is tried to determine which type has the greatest payoff per unit cost when added to this exhaustion mix of weapons.

If it is determined that saturating the terminal defense does not yield a profit, the leakage allocation is restored. In any event, the more profitable allocation, leakage or saturation, is used.

Other Constraints

Several other constraints may be imposed on the weapon allocation. These constraints will reduce the payoff but allow more realistic modeling of special cases. Weapon groups may be restricted in the set of targets they are allowed to strike in the following manner.

FLAG Restrictions: The user may restrict the allocation of weapons from any group according to the attribute FLAG. (This attribute is set in the data base by program BASEMOD.) Weapon groups may be permitted or forbidden to strike targets according to the FLAG value for the targets.

Country Location: The user may specify at program execution time the acceptable target country location codes (CNTRYLOC) for weapon allocation by weapon group.

MIRV Restriction: The user may specify at program execution time the acceptable target classes (CLASS) for allocation of MIRV weapons. These constraints are input by MIRV system type (IMIRV).

Naval Restriction: While naval forces can appear as targets within QUICK, there are specific limitations on the kind of weapons that can attack the aircraft carriers. All the targets which are included under class NAVAL should be moving ships. Certain weapon types can then be designated to attack only NAVAL targets. Since the mechanism of interaction of these naval strategic weapons with the aircraft carriers is essentially different from the normal kill mechanisms used in QUICK, an attribute (PKNAV) is defined for this type of weapon which specifies its single shot kill probability against an aircraft carrier. Thus, in the allocation process if a particular target is class NAVAL, the only weapons which can be allocated against that target are those which have the attribute PKNAV defined to be greater than zero. The kill probability of such a weapon, if successfully delivered through the area defenses against the carrier, is equal to PKNAV. These naval attack aircraft are handled like the tactical aircraft, since they do not pass through penetration corridors.

User-Specified Damage Levels (MINKILL/MAXKILL): The QUICK Plan Generator allows the user to specify the maximum (MAXKILL) and/or minimum (MINKILL) desired level of damage for any particular target. MINKILL specifies the minimum level of damage the allocator is to attain (if not attainable, the user is informed by the message MINKILL Too High). MAXKILL precludes the assignment of additional weapons once the specified level of damage is attained. Because only an integral number of weapons can be assigned to a target, the level of damage specified by MAXKILL may be slightly exceeded, unless there exists a combination of weapons which exactly meets the required damage level.

This slightly greater level of damage is intensified when the damage is evaluated using procedures which ignore the interweapon correlations and planning factor modifications used in QUICK. In order that the user can specify whether or not the application of damage constraints considers these factors, two options are available to the user for implementing these constraints. As a default option, these constraints are applied to damage calculations which include degradations for correlations in weapon delivery probabilities and considerations of the time dependence of target value. Since the evaluation programs to be used in conjunction with QUICK did not take these factors into account and since the output of these programs was to be compared to the QUICK-generated analysis, an optional computational procedure was desirable. Thus, the user has the option of specifying that the variables MAXKILL and MINKILL be applied to target damage which was calculated by ignoring the correlations and weapon delivery probabilities and the time degradation of value of the target. (User-input parameter IMATCH is used for this purpose.)

Combined Fixed, Optimum Assignment Capability

In order to provide for more precise user control of weapon allocations, there is a capability in the Plan Generation subsystem to allow the user to specify certain particular weapon-to-target assignments and then allow the automated plan generation process to allocate the residual of the weapon stockpile so as to maximize destruction of the remaining target value. The user can specify at his option certain fixed weapon assignments in the form of card inputs at the point where the actual weapon-to-target allocation occurs. This allows the user to examine the output of all of the preceding programs before committing himself to a particular fixed assignment. The user must specify the target identifier (either index number or target designator) of each target for which weapons are going to be forced-assigned. Also, the group of the weapon or weapons which is to be assigned to each of those targets, as well as the number from those groups, must be input.

This particular capability is made possible by the flexibility of the generalized Lagrange multiplier technique for performing optimum weapon allocations. Since any constraints can be imposed on the allocation to an individual target without seriously affecting the Lagrange multiplier allocation procedure, it is necessary only to modify the damage calculations for each target to reflect the damage created by the user-specified weapons prior to calculating the return for new potential weapons additions. Thus, when the allocator initiates the first pass, the only target value that has to be considered is that which is unaffected by the fixed assigned weapon. Also, the assigned weapons are subtracted from the stockpile available for automatic assignment.

In addition to the fixed assignment capability, the user may also specify the precise impact time of a fixed missile assignment. This allows the user to externally plan a time saturation attack against a BMD installation and be assured that the final QUICK plan will execute the tactic. The <u>only</u> use for this impact time specification is to calculate the correct missile launch time. If an impact time is fixed, this calculation overrides the other factors which would normally determine weapon launch time. However, the use of attribute DELTA for a missile base will modify the launch time in the Simulation subsystem; and the user-input parameters DELMIS or DLMIS (in program INTRFACE) will modify the launch time used in other simulators and damage-assessment systems.

If the target does not have terminal ballistic missile defenses, a maximum of 30 weapons can be assigned. On targets with terminal BMD, weapons from a total of 30 weapon groups may be assigned with no limit on the maximum number of weapons. In this latter case no bomber weapons may be fixed assigned if more than 30 missiles have been fixed assigned.

For missiles with a MIRV capability the assignment and timing of a fixed assignment may be changed by the application of the MIRV footprint parameter constraints.

The weapon allocator (program ALOC) supplies program ALOCOUT with a file ALOCTAR which contains data for each target, specifying the weapon groups assigned to each target together with the associated targeting data. ALOCOUT extracts from these records the data relevant for the post-allocation phase and reorganizes the extracted data by weapon group, giving for each weapon group the number of strikes and the specific targets assigned through each penetration corridor, plus associated data relating to these targets.

In addition, ALOCOUT is responsible for selecting optimum DGZs (desired ground zeros, also called weapon aim points) for weapons allocated to target complexes and for computing any aim point offsets required by the plan. In the case of simple or multiple targets, these offsets are simply set to zero. In the case of complex targets, which can have several elements at slightly different coordinates, program ALOCOUT selects optimum aim points within the target complex.

Multiple Targets

A multiple target represents two to five missile targets of the same type whose geographic locations are in the same vicinity (and whose index numbers, as game elements, are consecutive). These targets are represented as a multiple target, with a single set of coordinates, in the input to the allocator, so that the allocator can save time by making only one assignment of weapons for all elements of the multiple target. However, to develop detailed sortie plans, separate coordinates must be specified for each target element and specific missiles or aircraft must be assigned to each target from the weapon groups specified. Therefore, when processing a multiple target, ALOCOUT prepares a strike data record for each individual target which contains the index number and coordinates of the target element. From this point on in the data flow, the individual targets of a multiple target are treated just as if they were separate simple targets.

Complex Targets

A complex target (or target complex) is a combination of target elements sufficiently close in geographic location that a weapon on any one of them has some probability of killing other elements in the complex. Such target complexes are targeted as a unit, not as individuals. Thus, program ALOC allocates weapons against their total value, using one set of coordinates. To maximize targeting efficiency against such a complex, one must select optimum aim points among the target elements. The aim point offsets are specified relative to the first target element only and are output in that form for use in subsequent programs.

When ALOCOUT encounters a complex target, the program first assembles the target data in a form that can be efficiently used for DGZ selection.

Each target component of the complex generates a standardized target element. Targets with more than one hardness component generate more than one such target element, and targets with a specified target radius generate several elements, spread over the area of the target, to represent a value spread over the area. If the number of target elements so generated reaches the maximum program dimension (50), elements with similar properties and coordinates are combined. Finally, specific aim points (or aiming offsets) for each weapon allocated to the complex are selected using the target element data.

Optimization of Aim Points

The optimization of DGZs explicitly considers the time dependence of target value and the time of arrival of warheads. It does not reanalyze the correlation of delivery probabilities which is assumed to have been treated in the cross targeting provided by program ALOC.

The selection of DGZs is a two-step process. First, the prescribed warheads are assigned initial coordinates through a laydown process in which each successive warhead is targeted directly against that target element where the highest payoff is achieved, taking into account collateral damage to all other target elements. Second, the derivatives of the payoff as a function of x and y coordinates of each weapon are calculated, and the coordinates are adjusted to minimize the surviving target value. A test is included to help ensure that a global minimum has been determined rather than a local minimum which could occur as a result of the mathematical process.

This refinement procedure terminates after either a maximum number of iterations, or after it finds that it can no longer make significant improvements in the payoff. Further details of the mathematical theory upon which the selection of DGZs is based is presented in chapter 3.

BASIC SORTIE GENERATION

The development of the QUICK strategic war plan may be viewed as incorporating two major planning tasks. The initial task involves that processing required to establish an allocation of weapons to target which maximizes target destruction within the scenario and weapon system constraints established for the plan. Then, to implement this allocation, specific missile and bomber plans (i.e., sortie specifications) must be generated for each delivery vehicle. The latter task, referred to as "sortie generation" includes the preparation of a set of basic sortie specifications and the subsequent expansion/refinement of the data contained therein to produce a set of detailed sortie specifications. This section addresses the development of the basic sortie data. The preparation of detailed specifications is discussed in the following section.

The optimum allocation developed by program ALOC specifies only the weapon type and approximate base location (the group centroid) of the weapons allocation to each target; it does not specify the precise bomber or missile which is allocated to each target. In addition, when allocating bombs and MIRVs (multiple independently targetable re-entry vehicles), ALOC does not consider the requirement for geographically grouping targets for attack by a single delivery vehicle, bomber or MIRV.

The development of the basic sortie data for the individual missiles and bombers (i.e., the generation of the basic sortie* specifications for these vehicles) is primarily performed by program POSTALOC. In the case of missiles, the task is less complex since the missile flight plans (as required by the Simulator) are basically determined once a specific target or target set (provided by FOOTPRNT) is associated with a specific type of missile and the launch and target coordinates are known. In the case of bombers, the process is more complicated. The development of basic bomber sorties requires the association of several strikes in a single sortie. Moreover, it is necessary to associate each sortie with specific launch and recovery bases and to select a flight profile which specifies

*As used in QUICK the term sortie refers to an operational flight or flight plan associated with one delivery vehicle, missile or bomber.

where low-altitude capability should be used. Since the allocator (ALOC) does not distinguish between bombs and air-to-surface missiles (ASMs) carried by the same aircraft, it remains for program POSTALOC to determine which targets should be targeted with bombs and which with air-to-surface missiles.

Prior to being input to program POSTALOC, the weapon-to-target assignment data developed by program ALOC are processed by program ALOCOUT and, if required, program FOOTPRNT. The major functions performed by these programs are described in other sections of this manual but are summarized here for purpose of continuity.

The weapon allocator ALOC supplies program ALOCOUT with data for each target, specifying the weapon groups assigned to each target together with associated targeting data. ALOCOUT extracts from these records the data relevant to sortie generation and reorganizes the extracted data by weapon group, giving for each weapon group the number of strikes and the specific targets assigned through each penetration corridor, plus associated data relating to these targets. ALOCOUT is also responsible for computing any aiming offsets required by the plan. In the case of simple targets or multiple targets, these offsets are simply set to zero. In the case of complex targets, which can have several elements at slightly different coordinates, ALOCOUT selects optimum aim points within the target complex. A complex target (or target complex) is a combination of target elements sufficiently close that a weapon on any one of them will have some probability of killing other elements in the complex. Such target complexes must be targeted as a unit -- not as individuals. Thus, program ALOC treats them as a unit, allocating weapons against their total value, using one set of coordinates. In order to maximize targeting efficiency against such a complex, one must select desired ground zeros (DGZs) or aim points among the target elements (see DGZ Selection in this chapter).

If the plan includes missile weapon groups equipped with MIRVs (multiple independently targetable re-entry vehicles), program FOOTPRNT must be included in the plan development cycle. This program processes the individual weapon-to-target assignments and constructs the specific booster loads (the re-entry vehicle-to-target assignments to be associated with a single MIRV-capable missile) for each weapon group with a MIRV capability (see MIRV Missile Plans in this chapter).

Bomber Plans

The sortie definitions developed in program POSTALOC are generated separately for each weapon group and, within each weapon group, separately for each penetration corridor. For tactical bombers or naval bombers (i.e., PKNAV > 0.0), a penetration corridor is not used. However, to preserve the logic of the program, a dummy corridor index is defined to indicate no corridor usage. This corridor index is tested before performing distance calculations and strike assignments so that the appropriate substitutions are made in the method of processing. The basic sortie plan consists of ordered lists of the targets to be struck by each bomber, an indication of whether a target is to be struck with a bomb or an ASM, and an estimate of the distances between successive flight points that are flown at low altitude. The sortie definition does not, however, include the actual coordinates for the various events; e.g., launch, refuel, and drop bomb. These, together with the release points for ASMs and the times of entry into defense zones, are calculated in program PLNTPLAN.

The bomber sorties are actually constructed in the following fashion. First, the program reads in the strikes assigned to a given group. However, it reads them one corridor at a time. This division of strikes forms a raid; i.e., the aircraft from one group routed by way of not more than one corridor. Next, the strikes in the raid are roughly divided among the available vehicles and bases. Then, each sortie is evaluated in considerable detail, taking into account bomber range, estimated attrition rates, low altitude capability, and the option to use either bombs or ASMs on a given target. During this process, provision is made to omit strikes that seem unprofitable. Each strike omitted may be assigned to another sortie, so that this phase usually includes some refinement of the initial rough allocation of strikes. Only after all of the sorties for the given corridor are defined are the strike data for the next corridor read in.

A more detailed discussion of initial raid generation and sortie optimization is included below.

Initial Raid Generation: As indicated above, the first step in the generation of the sorties for a given weapon group and corridor is to ascertain the portion of the vehicles and warheads in the group that should be allocated to each raid. For a first approximation, the number of warheads assigned to each penetration corridor is proportional to the number of strikes assigned in each corridor in program ALOC. However, if this number of warheads does not correspond to an integral number of delivery vehicles, the necessary additional warheads required to produce an integral number of delivery vehicles are assigned to each corridor as it is processed. Since the corridors are delivered for processing in order of decreasing number of strikes assigned, this rule puts a slightly higher ratio of bombers to targets in corridors with large raids. In this way, bombers assigned to corridors where there are few other bombers will have more flexibility to select from the geographically sparse target set assigned. In the extreme case where a corridor happens to have only one or two isolated strikes assigned, the corridor will probably be skipped in the assignment of bombers from the group, so that isolated individual bombers are less likely to be assigned to such a corridor.

The next necessary task is to assign strikes within the raid to individual sorties. This requires the assignment of individual weapons to individual targets in accordance with the location of the targets relative to the penetration corridor. The assignment is accomplished through the use of curvilinear coordinate systems chosen to parallel typical flight paths within the penetration corridor.

Figure 8 illustrates two examples of the coordinate system employed in the planning of corridor penetrations. For strategic bombers, the coordinate system shown is established with the x=0, y=0 position corresponding to the origin of the penetration corridor (see figure 4). The y axis is parallel to the axis defined by the corridor origin and the coordinates of the corridor orientation point. For tactical or naval bombers, the x=0, y=0 position is defined as follows. Consider the centroid of the group of launch bases and the centroid of the group of target bases; and define the distance between the two centroids to be DISTC. The origin of the coordinate system is located at the end of the directed line segment which originates at the target centroid, passes through the launch base centroid, and has a magnitude of 2 × DISTC. Thus, in this coordinate system is is possible to locate both the targets and the launch bases.

The equations which describe the transformation from the Cartesian coordinates x, y to the curvilinear coordinates ρ, ϕ, are as follows:

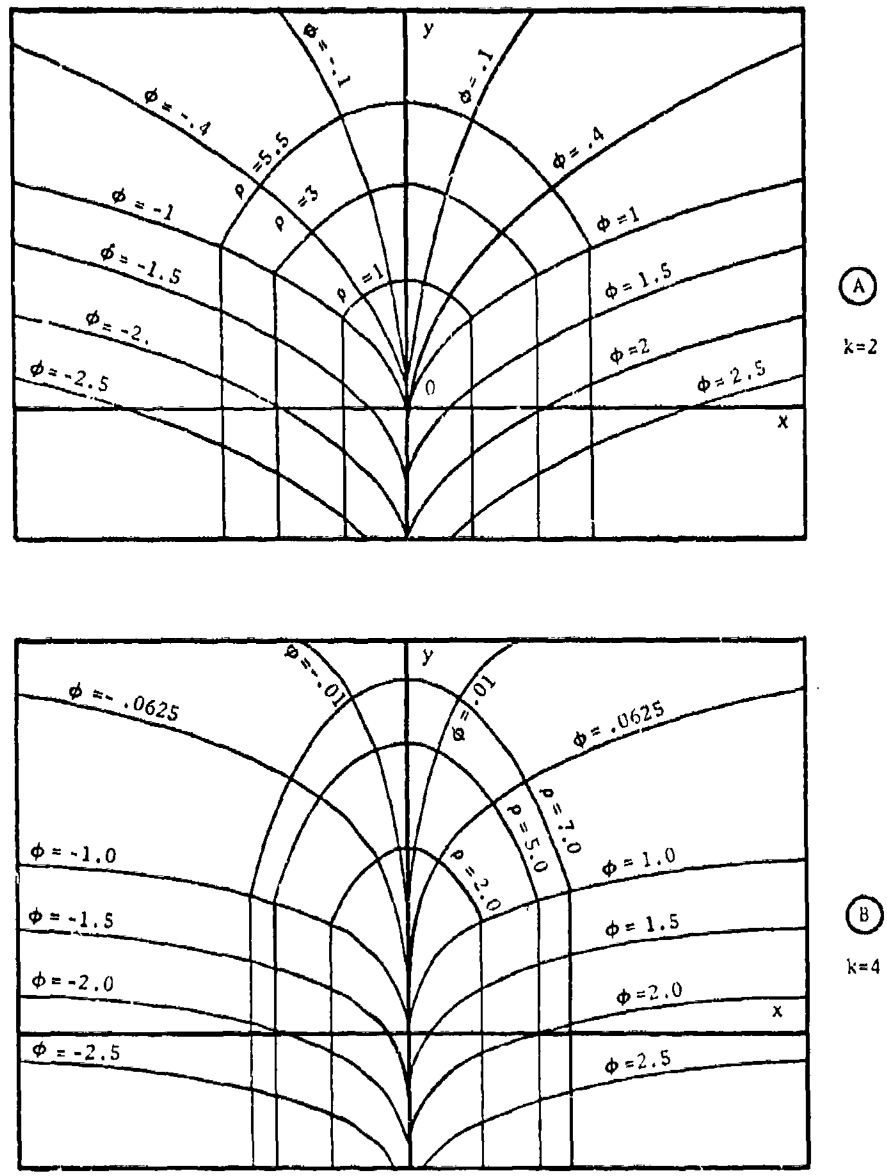


Fig. 8. Illustrative Curvilinear Functions

$$\phi = \frac{x}{y^k}$$

$$\rho = y^2 + kx^2 \qquad \text{for } |\phi| \leqq 1.0$$

$$\phi = \begin{cases} 1 + |x|^{1/k} - y & x \leqq 0 \\ -(1 + |x|^{1/k} - y) & x < 0 \end{cases}$$

$$\rho = |x|^{2/k} + kx^2 \qquad \text{all } x$$

Investigation of the two graphs presented reveals that lines corresponding to constant values of ϕ roughly parallel the type of flight paths which should be followed by penetrating bombers. Thus, in the assignment of sorties, a single bomber should be assigned targets which have approximately the same values of ϕ. Further consideration of the graphs indicates the alteration of the parameter k can be used to reflect certain planning objectives into the sorties. For example, higher values of k should be used when saturation of defenses is desired, while lower values should be used if greater importance is attached to minimizing the flight distances to targets (k is the corridor parameter KORSTYLE).

The procedure used to assign strikes within the raid to individual sorties now becomes clear. First, all strikes are arranged in increasing order of their ϕ coordinates. Then, the flights from each launch base are processed in order of the distance from the base to the corridor entry point, thus causing the vehicles to be processed in order of time of arrival.

To provide an approximation of saturation and roll-back tactics, each flight is assigned, as a unit, to either one side of the corridor or the other. The first flights are usually assigned to shallow targets (for which the absolute values of ϕ are higher), while later flights are assigned to deeper targets (for which the absolute values of ϕ are lower). Even if the density of strikes on the two sides of the corridor is quite different, the flights going to opposite sides are kept roughly in balance by comparing the value of ϕ before deciding to which side to assign the next flight. In order to maintain this balance, it is desirable to have at least five or six flights. Thus if there are four or fewer bases, two flights are sent from each base. If there is no penetration corridor defined, the launch bases are processed in order of their absolute values of ϕ alternating from one side of the coordinate system to the other, in an attempt to make the sortie paths approximate as closely as possible the direction of the lines of constant ϕ.

Within each flight, strikes are assigned to one sortie at a time by working through the list of unassigned strikes. Before any strike is assigned, however, all strikes previously assigned to the sortie are checked to be sure that it would not duplicate a previously assigned target (where multiple strikes may be allocated to the same target). If such duplication would occur, the strike is skipped, and later strikes on the list are processed to get the specified quota for the sortie. Processing for the next sortie in the flight always begins with the first unassigned strike and continues from there. Strikes actually assigned to each sortie are always arranged in order of increasing values of ρ thus corresponding to the initial time order or sequence of the strikes.

Sortie Value (VALSORTY): The optimization of the sortie is accomplished by a heuristic programming technique. To determine the effectiveness of modifications to the initial bomber sortie, the concept of the overall value of the sortie must be defined. The total value of the sorties is a function of the value associated with each of the flight points in the sortie and of the probability that each of these flight points is successfully executed. Specifically, VALSORTY, the total value of the sortie, is expressed as follows

$$\text{VALSORTY} = \sum \text{SURV(I)*V(I)}$$

where the summation is over all flight points including recovery. SURV(I) is the estimated probability of the bomber surviving to reach the flight point I, and V(I) is the estimated value of reaching that point.

The value V(I) attached to the target, I, depends on whether it is to be attacked by a bomb or an ASM.

1. If I is target for a bomb then: V(I) = RVAL(tgt)
2. If I is target for an ASM then: V(I) = RVAL(tgt)*[1.0 + TIMEPREM(tgt)]

   In the second relation, TIMEPREM is a bonus factor that is given for using an ASM on certain classes of targets. At present TIMEPREM is set to one for air defense targets (target classes 4 and 5), and to zero otherwise. This bonus is intended to reflect the advantage of destroying these targets before the aircraft and others in the same flight have to pass the target.

3. If I is a recovery point then we define: $V(I) = .5*\sum RVAL(tgt)$, the third equation, the summation is over all targets in the ...ie, which implies that the value of recovery is equal to 1/2 the value of all targets in the mission.

The variable RVAL as calculated in program ALOC is actually a measure of the marginal utility of each weapon. For weapon allocations not directed by the player (not allocated through the use of the "fixed assignment" capability), the marginal utility RVAL is computed as

$$RVAL_J = \left[\frac{(VTD_{I-1} - VTD_I)}{\lambda_I} \right] / PEN_J$$

where:

VTD_I = Residual target value after the allocation of the Ith weapon(s) on target J [VTD_0=VTO= original target value]

λ_I = Lagrange multiplier for the Ith weapon

PEN_I = Aggregate penetration probability for Ith weapon

This formula applies for targets with no terminal missile defenses. In this instance, VTD_{I-1} is equal to the residual target value prior to the allocation of the I^{th} weapon. However, for targets with terminal ballistic missile defenses, VTD_{I-1} is defined to be the residual target value if all weapons from the same group as weapon I are removed. This affords an accurate representation of missiles which are used for defense suppression.

For all weapons assigned by the fixed weapon assignment capability, the marginal utility is computed as

$$RVAL = VTO/\lambda_i$$

The computation of SURV(I) for the formula is based on a simple exponential attrition law. If the integrated attrition probability on each individual leg to a point J is given by ATLEG(J), then the survival probability for the bomber to the point I will be given by:

$$SURV(I) = EXPF\left[-\sum_{J=1}^{J=1} ATLEG(J)\right]$$

The attrition ATLEG(J) includes both area and local attrition for the leg (see Bomber and Missile Defenses).

Application of Low-Altitude Range: In selecting low-altitude range, QUICK assumes that on any leg or fraction of a leg flown at low altitude the attrition rates will be reduced by the factor HILOATTR. In order to estimate the expected value of the sortie, therefore, an estimate must be made of how the available low-altitude range should be applied. Notice that a change in the assumed attrition rate for any leg or part of a leg will change the integrated attrition for the leg ATLEG(J). This in turn will change the probability of survival to any point I (SURV(I)) which is required to evaluate VALSORTY.

The program therefore begins by summing the total distance for the sortie as specified. This distance is subtracted from the aircraft range to give the surplus range RNGSURP available for the mission. Using the conversion factor RANGEDEC, this surplus range is used to estimate the available low-altitude distance AVAILOW for the mission. Finally, AVAILOW is allocated to the various legs in a manner intended to maximize the value of the sortie VALSORTY.

During this allocation of available low-altitude range, the following alternatives are provided:

1. Allocate low-altitude range to that remaining precorridor leg that has the highest attrition
2. Extend the low-altitude flight from the first target one more leg toward the depenetration point (where the attrition is assumed to end)

3. Extend the low-altitude flight a little further in front of the first target toward the corridor origin.

Choices among these alternatives are made on the basis of which one will produce the largest rate of increase in VALSORTY per nautical mile of low-altitude range required.

To illustrate how the priorities for this allocation work out mathematically, we note that the cumulative survival probability SURV to route point i can be represented as a product of the survival probabilities S_j for each leg j up to and including the i^{th}. Thus we can rewrite the equation for VALSORTY as follows:

$$V = \sum_{i=1}^{i=n} \left\{ \prod_{j=1}^{j=i} S_j \right\} V_i$$

where V is the value of the sortie and V_i is the value of successfully reaching the i^{th} route point. (This is referred to as the value done, or VALDONE, in the program.)

We also note that $S_j = e^{-\alpha_j}$ where α_j is the total attrition on the j^{th} leg. Obviously α_j is a function of L_j, the low-altitude distance allocated to the j^{th} leg.

Differentiating V with respect to L_k, the low altitude allocated to some specific leg k, we obtain

$$\frac{\partial V}{\partial L_k} = \frac{\partial V}{\partial S_k} \frac{\partial S_k}{\partial \alpha_k} \frac{\partial \alpha_k}{\partial L_k}$$

while

$$\frac{\partial V}{\partial S_k} = \sum_{i=k}^{i=n} \frac{1}{S_k} \left\{ \prod_{j=1}^{j=i} S_j \right\} v_i$$

$$\frac{\partial S_k}{\partial \alpha_k} = - e^{-\alpha_k} = \cdot \, S_k$$

Thus

$$\frac{\partial V}{\partial L_k} = - \left[\sum_{i=k}^{i=n} \left\{ \prod_{j=1}^{j=i} S_j \right\} v_i \right] \frac{\partial \alpha_k}{\partial L_k}$$

Now separating out the common factors S_j for j=i,k, and noting that

$$\prod_{i=1}^{i=k} S_i = \mathrm{SURV}(k)$$

we obtain

$$\frac{\partial V}{\partial L_k} = - \,\mathrm{SURV}(k) \left[\sum_{i=k}^{i=n} \left\{ \prod_{j=k+1}^{j=i} S_j \right\} v_i \right] \frac{\partial \alpha_k}{\partial L_k}$$

The term in the square bracket is the estimated value of the remainder of the mission, assuming that the aircraft arrives successfully at the point k. (This is called VALON(k) in the program.) Since α_k is the total attrition for the k^{th} leg, the quantity $\partial\alpha_k/\partial L_k$ is simply the difference between high-altitude and low-altitude rates per nautical mile. Moreover, since we are assuming a constant ratio HILOATTR between high-altitude and low-altitude attrition rates, this quantity is proportional to the attrition rate. Therefore, we can write:

$$\frac{\alpha V}{\partial L_k} = - \text{SURV(k)} * \text{VALON(k)} * (\text{Attrition Rate (k)}) * \text{CONSTANT}$$

Thus the leg where additional low-altitude range will do the most good can be selected by comparing the product of the first three factors in the above expression for $\partial V/\partial L_k$.

This is the technique used in determining whether the next increment of low-altitude range is to go into the precorridor legs, the leg to the first target, or in extending the low-altitude flight to additional legs or fractions thereof beyond the first target.*

The attrition rate used in this decision process for legs beyond the first target is simply ATLEG(k)/DISTLEG(k); thus the effective attrition rate also reflects any local attrition associated with the k^{th} route point.

The assumed position-dependent attrition rate per nautical mile is used on the leg to target one so that low-altitude range is added to this leg only as far ahead of the target as is justified by the assumed attrition rate.

*Actually the values of SURV used in the subroutine during the allocation of the low-altitude flight are all divided by the value of SURV to the first target. This speeds up the operation of the routine, since changes in the survival probability in the precorridor legs or on the way to the first targets, as allocations are made to these legs, do not affect the value of SURV which must be used in later legs.

The attrition rate used in the precorridor legs is the constant value specified in the data base.

It is also worth noting that regardless of which leg k receives the final allocation of low altitude, this allocation will correspond to some value for the quantity $\partial V/\partial L_k$. This value, of course, is the marginal value of additional low-altitude range. It can be converted (using the conversion factor RANGEDEC) to obtain a marginal value of additional range or the marginal value of saving distance in the sortie definition. This marginal value of distance, known as VALDIST, is computed by program POSTALOC and used to estimate the value of the distance saved in alternative sortie definitions.

The above allocation procedure produces a rigorously optimum allocation of the low-altitude range to the sortie so long as there is no local attrition. However, where local attrition is present at specific targets late in the sortie, a theoretically optimum allocation might allocate limited low-altitude range explicitly for each such target. If this were permitted, it could lead to sorties which unrealistically go low for each defended target and fly high between such targets. To avoid this difficulty, the requirement has been imposed that after passing the corridor origin a flight is allowed to go low only once.

Moreover, for simplicity of computation during the development of the sortie definitions, the flight is required to go low before the first target, if it is going to fly low at all. Obviously, if there is local attrition at a target toward the end of the mission but not at the first target, it might be better to stay high past the first target and save the low-altitude capability to be used in the vicinity of later defended targets. While this possibility is ignored (for computational speed) during the development of the sortie definitions, after the sortie definition is complete a final check is made and, if such a change would increase the estimated value of VALSORTY, the change is incorporated in the final version of the flight plan.

If there are no defended targets where the bomber is scheduled to fly high after using its low-altitude range, no changes in the sortie are considered. Otherwise, QUICK tries extending the low-altitude range to include the next defended target. When any low-altitude capability is left prior to the first target of the sortie, the excess is allocated as before between the leg to the first target and the precorridor legs. If there is no such excess, the point where the aircraft first goes low is set as soon after the first target as possible. The resulting value of VALSORTY is then computed. If the sortie value is increased over that previously obtained, the revised sortie is used. If not, the prior version is retained. This process is repeated until a version of the sortie is tested in which the low-altitude flight is extended to the last defended target. That version of the sortie which produced the best value of VALSORTY is then selected. There is a possibility that in the original version of a given sortie, the total range may be inadequate to execute the sortie as defined, even if the entire mission were carried out at high altitude. In this case, low altitude is not assigned to any of the legs. Moreover, VALSORTY is computed so that it receives no contribution from any route point beyond the maximum range of the aircraft. In this case, later operations usually result in the omission of some targets that cannot be reached or the elimination of recovery, so that a revised sortie definition is developed which constitutes a feasible sortie.

Depenetration Routing: Each bomber for which a recovery is planned must exit via a depenetration corridor. These corridors, while having no attrition associated with them, serve to define the geographic route to be flown while leaving enemy territory. When a bomber leaves a depenetration corridor, it recovers at a base which is associated with that corridor. The bomber chooses the depenetration corridor according to the last target struck in the sortie. If D1 is the distance from that target to the depenetration point, and D2 is the distance from depenetration point to the nearest recovery base associated with that point (or corridor), then the depenetration corridor used is the corridor which minimizes

(2*D1) + D2

Sortie Modifications: All decisions on the modifications of the sortie definition are based on the estimated effect the changes will produce in the value of VALSORTY.

The initial sortie definition may not even be feasible. It may require too many warheads; it may require too much range; or it may specify all bombs whereas the aircraft may carry ASMs. Thus, the task of program POSTALOC is to revise the sortie definition to produce a feasible sortie with the highest possible expected value of VALSORTY.

In accomplishing this, the program estimates the marginal value of using bombs in a sortie and the potential advantage of using ASMs instead, performing one or more of the following functions.

- Determine which targets assigned bombs should be converted to ASMs when not all ASMs are assigned
- Determine which remaining bombs are of least value and should be deleted if too many strikes are assigned
- Determine which route points (recovery or bomb targets) are of negative value to the sortie and should be deleted.

In so doing, it analyzes each route point in succession down to and perhaps including the recovery point. The processing of each route point is handled in two parts. First, the marginal value of the route point as a target for a bomb is evaluated. Then, the value of the same route point is calculated as a potential ASM target, and the marginal value of changing it to an ASM target is estimated. For these computations the recovery point is not included in the evaluation.

When all ASMs have been assigned, there may still be too many strikes for the available warheads. The next step may then be, still excluding the recovery point, to select the least valuable remaining bomb which could be deleted. Finally, the sortie is evaluated again, this time including the recovery point to be sure that all route points including the recovery make a positive contribution to the payoff.

The marginal value of each route point is also evaluated. The value of reaching the route point, multiplied by the probability of surviving to reach it, is compared with the cost of doing so.

This cost consists of two elements:

- Change in the probability of reaching succeeding targets because of local attrition, if any, at this target, or because of additional area attrition over the added distance required to fly to this target
- Reduction in the amount of low-altitude flight available because of the extra distance to the target, which in turn can affect penetration probability to all targets.

In analyzing each target, the program considers an alternative flight route which bypasses the target and goes directly from the preceding to the succeeding target. The effect of this route on the expected payoff for succeeding route points can be directly evaluated. The change in attrition is known, so the change in the cumulative survival probability SURV to the succeeding target can be computed, and the value VALON of the remainder of the sortie is made available.

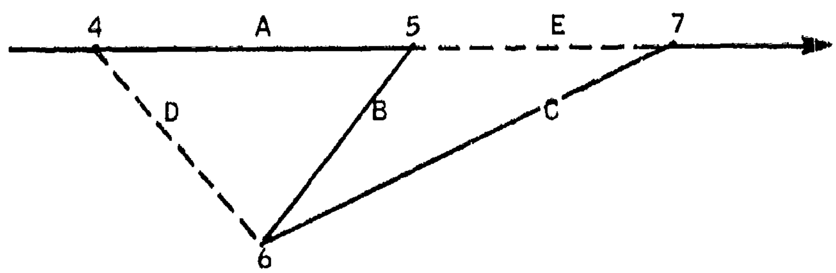
The change ΔV in VALSORTY, due to change in available low-altitude capability, is only estimated. The estimate is based on the amount of distance saved by skipping the target DISTSV multiplied by the quantity VALDIST, the marginal value of distance. However, where the saving in distance is very large, this type of linear extrapolation with a constant VALDIST can be quite misleading and could even exceed the full value of all targets in the sortie. Obviously, the value of the sortie can never exceed the actual value VALMAX of all route points, and with one target k omitted could not exceed VALMAX-V(k). Consequently, the value VALO of omitting a target k cannot exceed POTVALO=VALMAX-V(k)-VALSORTY. This quantity POTVALO is therefore used to establish a limiting value for the value of saving distance. The quantity VALDIST is used to give the derivative for small values of DISTSV. The actual form used for estimating ΔV for distance saved is:

$$\Delta V = POTVALO * [1.0 - 1.0/(1.0 + TEMP)]$$

where TEMP = VALDIST * DISTSV/POTVALO.

In the second phase of the process -- to estimate the value of the target as an ASM target -- the time premium for using an ASM on the target is added into the basic value RVAL of the target, and the survival probability used is that for the earliest possible launch point in range of the target.

Determining the value of omitting a route point requires calculation of the distance saved. Once this information has been computed for two successive route points, the next computations are distances that are necessary to determine whether the two points are out of order on the route. The following figure illustrates the method used.



The figure illustrates a route:

4 via leg A to 5

5 via leg B to 6

6 via leg C to 7.

Consider the possibility of reversing the order of points 5 and 6 on the route. The present distance is A + B + C; the revised distance would be D + B + E, using dashed alternative legs D and E.

If the reversed path is shorter, then D + B + E < A + B + C or A + C - D - E > 0. When we consider omitting 5, we compute DISTSV = A + B - D. When we consider omitting 6, we compute DISTSV = B + C - E.

Adding the two values of DISTSV and subtracting 2B we obtain A + C - D - E. Therefore, if this value is positive the two route points are out of order, and the flag JSEQERR is set to indicate one of the two targets for possible temporary omission. Usually the first target is flagged. (The presumption is that a later evaluation will result in the replacement of such a target in its proper position in the sortie.) However, if the first target is also a launching point for ASMs, even temporary omission would be complicated; thus, rather than seek an alternative launch point for the ASMs, the second target will be flagged instead. If both route points are also ASM launching points no flag is set, and the current order of targets is not changed.

The problem of route points serving double duty as ASM launch points also arises when the marginal value of omitting route points is being estimated. Therefore, after the original value VALO is estimated, a check is made to see if the point is used as an ASM launch point. If so, the value VALO of omitting the point is decremented to reflect changes in the marginal value of the ASM, for which a new and probably inferior launch point must be found. If such an alternative launch point cannot be found, the entire value of the ASM is charged to VALO. Except in the most extreme cases this is sufficient to preclude omission of this target.

If the program (POSTALOC) is to delete a bomb where the same route point is used as a launch point for an ASM, it first seeks an alternative launch point for the ASM. However, if it cannot find one, the ASM is omitted also.

The desirability of using an ASM on one of the omitted targets is also estimated. This can be done either to find a target for an unused ASM or to evaluate the value of substituting an omitted strike point as the target for an ASM already assigned.

Changes in the bomber route are not considered at this point. In this way, the values of changes considered can be evaluated exactly.

The operation is divided into two portions. First, the program scans all targets in the mission currently assigned for ASMs, skipping any target used as its own launch point, since its omission would change the bomber route. The marginal value of the others is determined by multiplying the value of the strikes as ASM targets by the survival probability for the aircraft to the launch point. During this phase, the strike JDEL with the lowest marginal ASM value MINDA is determined.

In the second portion of the operation, all omitted strikes are evaluated as ASM targets. The method of evaluation is exactly the same, except that a suitable launch point must be found. The first route point within range of each target is taken as the potential launch point. As it proceeds through this part of the program, it keeps a record of the strike JADDA with the highest marginal ASM payoff MAXDA and the associated launch point IAIM. Of course, strikes are disqualified for such consideration if another strike on the same target is already in the sortie definition.

The program (POSTALOC) also estimates the value of strikes in the omit list as potential targets for bombs. It does this by finding an additional target or an omitted target that is more profitable for a bomb than the least valuable in the sortie. In turn, each target in the omit list is processed. Each potential target is tried first in a position just before the first target with a higher value of RHO. The distance added to the sortie is then evaluated. The target is then tried in a position on the other side of its nearest neighbor (nearest in value of RHO). If this position produces a lower value for the distance added, this position is accepted instead of the original position.

The marginal contribution of the bomb in the preferred position is then computed. The method parallels the calculation of the marginal value of bombs in EVALB. The effect of the extra attrition on following targets is evaluated. Then the effect on low-altitude range is estimated using (VALDIST * DISTAD). These quantities are added to get the total benefit VALO of not flying to this new route point. The value of the target, times the probability of surviving to reach it, is then computed to get the net marginal value of adding the target DVALB.

The index for the target with the highest DVALB is then recorded as JADDB, and the route point it should follow is recorded as JAF. Of course, any strike on a target already in the sortie is excluded from consideration to avoid duplicate strikes on the same target by the same bomber.

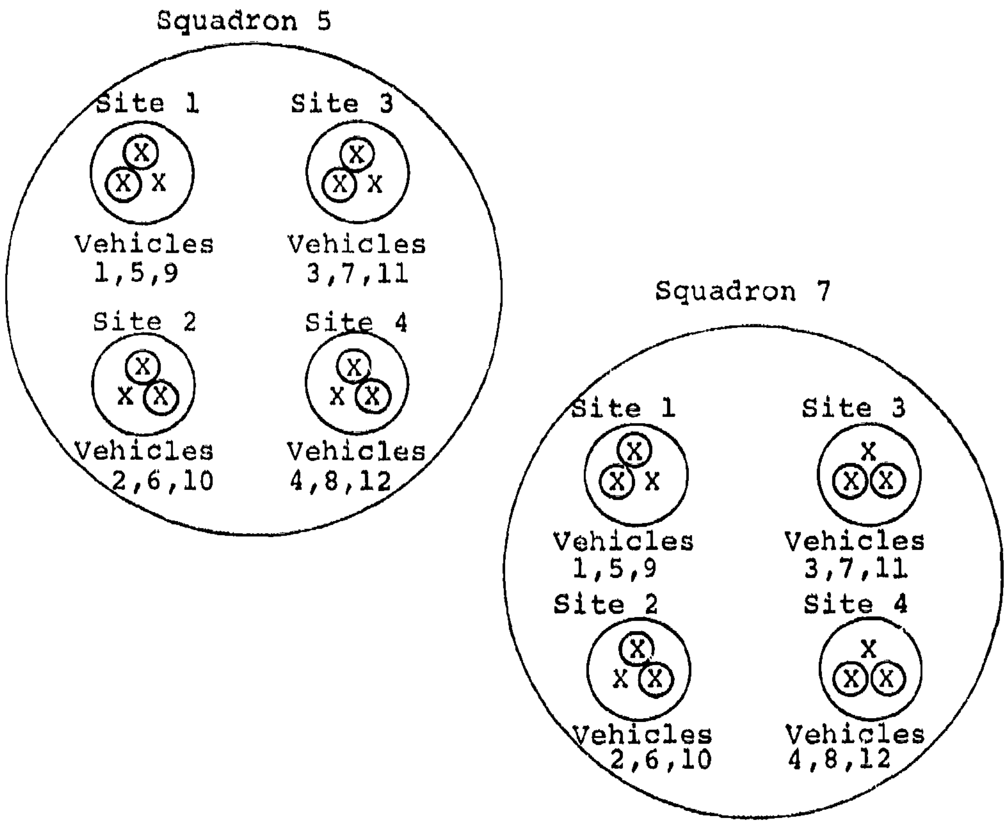
Missile Plans

Program POSTALOC generates missile sortie specifications for each missile weapon group and its assigned targets. The weapon group data contained on the BASFILE prepared by program PREPALOC are read in and stored. In addition, the target data are obtained from: (1) the ALOCTAR file prepared by ALOC, if the plan does not contain MIRV weapons, or (2) from the TMPALOC file output by FOOTPRNT, if MIRVs are included. Since MIRV missiles are a special case of missiles, the description of the additional processing for these weapons is deferred to the next section. For non-MIRV missiles, individual targets are assigned to individual vehicles. For MIRV missiles, an ordered set of targets is assigned to each vehicle. From these data, specific strikes are assigned to specific delivery vehicles within the weapon group. The development of the missile plans is relatively straightforward. With the exception of the timing computations (e.g., launch time, performed in PLNTPLAN), the missile plans are complete as output by POSTALOC.

Figure 9 illustrates the structure of a typical missile group. The group may include several squadrons (two shown) and a squadron may include several sites (four per squadron shown). Each site may have one or more vehicles (three shown). Vehicles are considered to occupy the same site if they are so close together that they would have to be targeted as a simple target. For example, the Polaris squadron of 16 missiles on one submarine is considered to occupy one site, while the Minuteman squadron of 50 missiles occupies 50 separate sites.

On the other hand, any nonalert missiles in a squadron will constitute a separate weapon group. Since the vehicle indices within a squadron may not start from 1, the starting vehicle index ISTART for each squadron is supplied as an input to the missile assignment phase. This and the other input parameters defining the available weapons for the program are also shown in figure 9.

In POSTALOC a maximum of 18 missiles can be assigned to a single launch event. (To facilitate preparation of sortie plans for use in other simulators (e.g., NEMO), each missile launch is treated as a separate event in the final plan output by program PLNTPLAN.) If a squadron contains more than 18 weapons (or re-entry vehicles in the case of MIRV groups), the number of events required to output all the weapons is computed. This number times the number of squadrons in the group gives the total number of events to be generated for the group. This computation, however, is not performed at the start of the processing for each group, but rather during processing. Since missile groups with



Ⓧ Vehicles in Group

X Vehicles not in Group

NOPERSQN = Total Vehicles in Squadron.
NBASE = Number of Bases (or Squadrons) in Group.
NWPSITE = Number of Weapons per Site.
ISTART = Lowest Vehicle Index in Group for Each Squadron.
NWPNS = Total Vehicles in Group.

Fig. 9. Exemplar Configuration of Missiles in a Group

a MIRV capability have a variable number of strikes per booster, the number of missiles in each event can be determined only dynamically.

The input strikes (allocated weapons) assigned to the group are ordered by decreasing values of RVAL (the marginal utility of the weapons computed as described for bomber weapons). For weapon groups with a MIRV payload, the strikes are ordered by decreasing value of the total value associated with each booster (i.e., the sum of the values of RVAL for each target assigned to the booster). In order that each event to be output to PLNTPLAN will contain a mix of values for its strikes, the strikes are not assigned to launch events in simple serial order. The strikes are distributed over the events to attempt equalization of strike value between events. The method for this is to skip certain strikes when constructing an event. The algorithm selects a strike to start an event, skips a number of high value strikes, selects another for inclusion in the event, and so on. Thus, the first event may be composed of strike numbers 1, 11, 21, 31, ..., in the input list, and the second event may have strikes 2, 12, 22, ... The number of strikes to be skipped is computed as a function of the number of squadrons in the group.

Before assigning the strikes to each vehicle, the number of vehicles in the group and the number of vehicle assignments are computed. If the number of vehicle assignments is less than the number of vehicles, the number of vehicles for which a plan will be processed is decreased until it matches the number of assignments. If the number of vehicle assignments exceeds the number of vehicles, QUICK determines if the vehicles are carrying a MIRV payload. If so, then program FOOTPRNT has erred in generating the footprint assignments. An error message is printed to this effect and processing proceeds. The result will be the omission of some target sets from the final plan. If the group does not have a MIRV payload, the least valuable assigned targets are removed until the number of targets equals the number of vehicles. However, targets assigned through the fixed assignment capability of program ALOC are not omitted, unless there are more fixed assignments for this group than there are vehicles. In that case (an input error), fixed targets are omitted in order of increasing value (RVAL) until the number of targets matches the number of vehicles. In addition, an error message is printed to this effect.

MIRV Missile Plans

Technological developments in guidance have made possible the introduction of multiple missile warheads on a single missile which can be directed at geographically separate targets. Although the original QUICK General War Gaming System was not designed to accommodate multiple independently targetable re-entry vehicles (MIRVs), the introduction of MIRVs into operational weapons made it very desirable to incorporate into the QUICK system the changes required to enable the consideration of these weapons.

A major ramification of the addition of the MIRV capability to the system was the necessity to consider the effect upon the target assignments of "footprint" constraints: that is, constraints on the geographic configuration of targets assigned to a single missile equipped with MIRVs. In order to minimize the amount of system alteration required to introduce the MIRV capability, it was decided not to alter the basic weapon allocation process, but rather to introduce these footprint constraints into the plan generation process subsequent to the initial assignments of targets to weapon groups as effected by program ALOC. Hence, the development of the general strike plan now entails, in order of occurrence, the initial allocation of targets to weapon groups in program ALOC, the refinement of the target point locations for complex area targets and the reordering of the assignments according to weapon group in program ALOCOUT, and the construction of specific booster loads (i.e., the weapon-to-target point assignments to be associated with a single MIRV-capable missile) for each weapon group with a MIRV capability in program FOOTPRNT. Once program FOOTPRNT has determined the assignment of targets to booster, this information is passed to program POSTALOC. In that program, the booster load assignments are distributed to the individual boosters in each squadron according to the method discussed previously in the section Basic Sortie Generation (Missile Plans). In the MIRV case, however, the value of the sortie is defined to be the sum of all the marginal utility values (RVAL) for the targets assigned to the booster. Program

FOOTPRNT orders the booster load assignment information in order of decreasing values of sortie before passing the information to program POSTALOC.

Throughout this discussion, the term "target point" will refer to a "desired ground zero" (DGZ) selected either in program ALOC for simple targets, or in program ALOCOUT for complex and area targets, as the aim point for a single re-entry vehicle (RV). Although, depending upon the value of a given target, two or more RVs may be allocated to target points with the same geographic coordinates, these target points will be considered as being distinct in all the succeeding processing of the target assignments.

When a weapon group with a MIRV payload is located, the detailed target point assignments for each of the boosters in the group must be formulated. The initial attempt at creating a set of feasible booster assignments consists of arbitrarily dividing all target points assigned to the group in program ALOC equally among all of the boosters in the group, such that targets of similar launch azimuth will have a greater chance of being assigned to the same booster than targets of different launch azimuth.

Two important characteristics of this initial assignment should be noted. First, it is possible that the assignment will not satisfy the footprint constraints. Second, it is usually the case that the number of target points initially assigned to a booster is greater than the number of RVs which the booster will actually carry. This latter phenomenon results from the "over-allocation" policy utilized for the assignment of MIRV weapons. To minimize the chance that the elimination from booster assignments of targets which will not fit into a feasible footprint will cause an under-utilization of the available weapon stockpile, additional RVs are created for MIRV weapon groups in program PREPALOC for assignment in program ALOC (see Weapon Grouping). However, after the processing in program FOOTPRNT, the expected number of RVs actually utilized will not exceed the number that are available.

Preliminary Calculations: After the initial RV-to-target assignments have been made, it is necessary to refine them to insure that the final assignments satisfy the various footprint constraints. To accomplish this task, the target assignments for each booster are processed individually. To facilitate the creation of a feasible set of booster assignments, each target point in the set assigned to the weapon group is classified as either a potential target or a non-potential target, for the booster under consideration. Only targets classified as potential ones are eligible to be included in the set of assignments for this booster. The set of potential targets for any given booster consists of all of the targets in the current assignment for the booster, a certain portion of the targets which were in the potential target list for the previous booster but which were not assigned to that booster, and a certain user-specified fraction of the current target assignment of the next booster in the list. The latter targets are included in the potential target list to enable the program to function efficiently in cases where the distance between certain targets of different launch azimuth is less than that between certain targets of the same azimuth.

After the potential target list has been defined and the appropriate target data introduced into the potential target arrays, all intertarget distances for the potential targets are calculated and stored in an array D(i,j) as follows:

$$D(i,j) = \begin{cases} \begin{bmatrix} + (\text{downrange distance})^2 & j \text{ downrange of } i: \\ - (\text{uprange distance})^2 & j \text{ uprange of } i: \end{bmatrix} & i<j \\ - (\text{crossrange distance})^2 & i>j \\ \text{worth of keeping the target in the potential target list} & i=j \end{cases}$$

Downrange distances are measured along an axis which is parallel to the shorter of the two great circle routes from the launch point to the first target point to be hit; crossrange distances are measured along an axis which is perpendicular to this route. The uprange direction is defined to be parallel but oppositely directed to the downrange direction.

Two concepts which are extensively used in the remainder of the processing of the target assignments are that of equivalent downrange distance (EDD) and that of the value of including a given target in a particular assignment.

Equivalent Downrange Distance: To determine the worth of keeping a target in the potential target arrays, as well as to determine overall footprint feasibility, use is made of the equivalent downrange distance of a target, an approach similar to that used in program FOOTCALL (not part of the QUICK system).* The major premise of this method is that all downrange, crossrange, and uprange distances can be converted into an equivalent downrange distance, EDD,

---

*"Strategic Offensive Weapons Employment in the Time Period About 1975 (U)," (Top Secret) Weapons Systems Evaluation Group Report R-160, August 1969, Volume VI, Allocation of MIRV System.

which is equal to the downrange distance that could be traversed by the payload if the same amount of energy were expended as would be required to traverse the distance under consideration. In practice, the EDD from point i to point j, as depicted in figure 10, may be expressed by the following relationship:

$$\left(EDD_{ij}\right)^2 = \begin{cases} \left(DOWN_{ij}\right)^2 + \left(\frac{DR}{CR}\right)^2 \left(CR_{ij}\right)^2 & \text{if j is downrange of i} \\ \left(\frac{DR}{UR}\right)^2 \left(UR_{ij}\right)^2 + \left(\frac{DR}{CR}\right)^2 \left(CR_{ij}\right)^2 & \text{if j is uprange of i} \end{cases}$$

where:

$\frac{DR}{CR}$ $\triangleq$ downrange-crossrange ratio

$\frac{DR}{UR}$ $\triangleq$ downrange-uprange ratio

$DOWN_{ij}$ = downrange distance from i to j

CR_{ij} = crossrange distance from i to j

UR_{ij} = uprange distance from i to j

Thus, it is now possible to define V_{in_i}, the worth of keeping the i^{th} target in the potential target list, as follows:

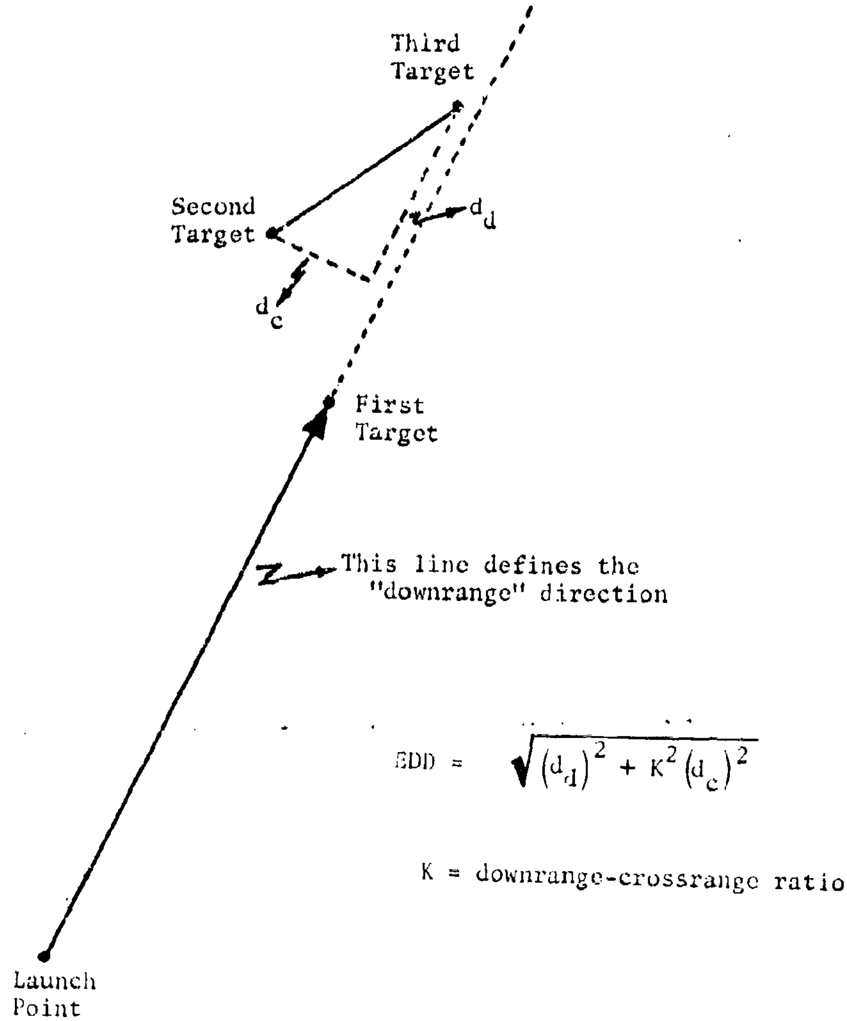


Fig. 10. Graphical Representation of the Concept of Equivalent Downrange Distance (EDD)

$$V_{in_i} = \sum_{j=1}^{n} \left[\frac{1}{\left(EDD_{ij}\right)^2} \right] \Big/ AGE(i)$$

where:

$AGE(i) = (DELAGE)^{NBIN_i}$

$NBIN_i$ = number of boosters for which the i^{th} target has been in the potential target list

DELAGE = user-introduced weighting parameter

n = number of targets in potential target list.

The variable AGE(i) is proportional to the amount of time that the i^{th} target has resided in the potential target arrays. This relationship reflects the fact that the probability of successfully incorporating a target into a feasible assignment diminishes as the distances between it and the other targets in the assignment increase.*

Value of Assigning a Target to a Booster: The main objective of the processing in program FOOTPRNT is to create feasible booster assignments which include as many as possible of the target points that were originally assigned to the weapon group in program ALOC. Therefore, two factors must be considered when determining the value of adding a given target to a booster assignment -- first, the

*The details of entry and removal of targets into the potential target arrays are covered in Chapter 6, Program FOOTPRNT, Subroutine BOOSTIN, Programming Specifications Manual, Volume II, Plan Generation Subsystem.

number of targets which have already been assigned to the booster, and second, the amount of energy which would be required to travel from the target under consideration to each of the remaining targets in the "miss list," the list containing all of the targets in the potential target list which have not been assigned to the booster. Specifically, W_i, the worth of adding the i^{th} target to the booster assignment, is defined as follows:

$$W_i = \sum_{\substack{j \\ \text{targets in} \\ \text{miss list}}} \left[\text{VALF}\,(\alpha_{ij}, N) \right] * \text{RVAL}_j$$

where

$$\alpha_{ij} \equiv \frac{\text{EDD}_{ij}}{\text{EDD}_{max}}$$

EDD_{max} = maximum EDD which could be achieved before the addition of the i^{th} target to the booster assignment

RVAL_j = relative target value of target j

and where it is assumed that the i^{th} target will be inserted into that position in the assignment which would require the smallest increase in total energy expended by the booster. The graphical relationships which define $\text{VALF}(\alpha_{ij}, N)$ and N are presented in figures 11 and 12, respectively. The value of VALF (α_{ij}, N) is a function of both EDD_{ij}, where j is an arbitrary target in the miss list ($j \neq i$), and N, a weighting parameter, is a function of both NHIT, the number of targets already assigned to the booster, and PN, a user-specified weighting factor.

The calculation of the value of making each of the potential targets the first in the assignment makes use of the above equation for W_i where the miss list now contains all of the targets in the potential target list and EDD_{max} is equal to THROWMAX, a user-specified input defining the maximum possible distance between any two target points. Specifically, $VALFIRST_i$, the value of making the i^{th} target the initial one in the booster assignment. is defined as follows:

$$VALFIRST_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} \left[VALF(\alpha_{ij}, N) \right] * RVAL_j \qquad (5)$$

Target Assignment: The initial task in the construction of an assignment is to choose as the first target to be hit that target from the potential target list which has the maximum value of VALFIRST. When this target is located, it is moved from the miss list to the "hit list," where the "hit list" contains all of the targets in the potential target list which have been assigned to the booster.

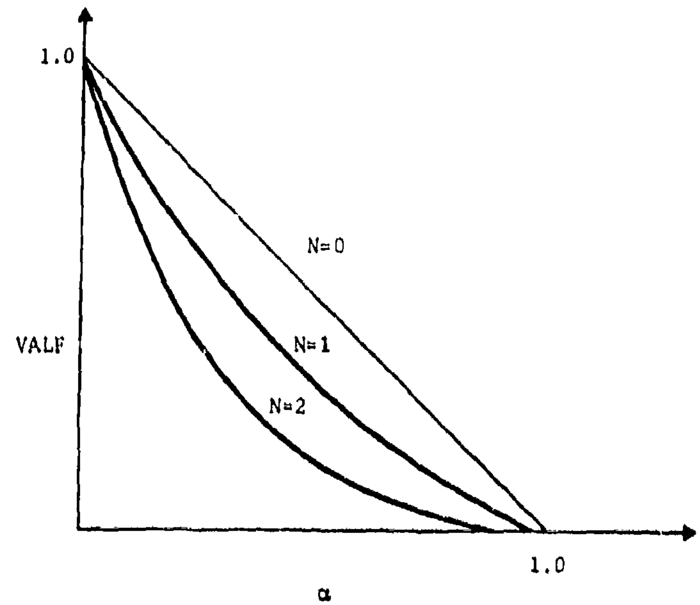


Fig. 11. Graph Indicating VALF as a Function of α, for Various Values of the Parameter N

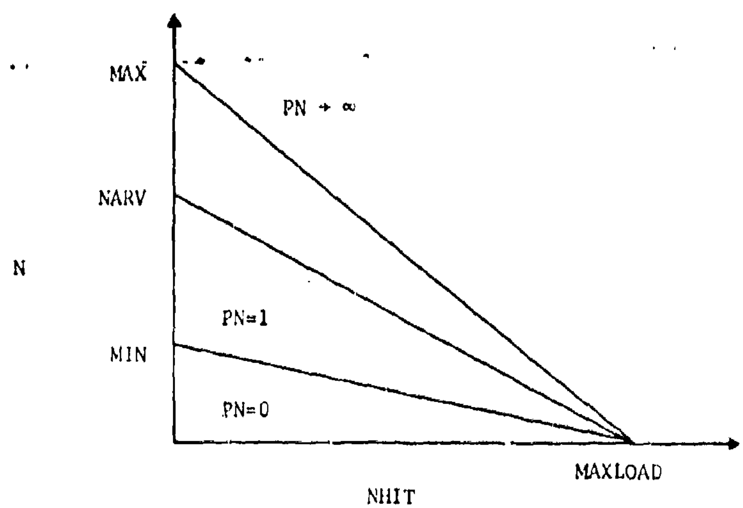


Fig. 12. Graph Indicating N as a Function of NHIT for Various Values of the Parameter PN

The next task, of course, is to choose that target from the miss list which would be most valuable if added to the assignment. Thus, after the first target has been assigned, values of W are calculated for each target in the miss list. The target with the largest value of W is then chosen to be added to the booster assignment, and it is moved from the miss list to the hit list. If this target is incorporated as the new first one in the ordered assignment, the intertarget distances and the values of W are recalculated.

This process is continued until either the maximum booster load has been attained, or the addition of any other of the remaining targets in the miss list would result in a violation of the footprint constraints. At this point, a final improvement stage is entered. The target which requires the greatest marginal use of fuel is temporarily deleted from the assignment. Attempts are then made to add one or more targets to the assignment to replace the one removed. If it is possible to add more than one target, or if one target may be added which is worth more than the one which has been temporarily deleted, the new assignment is kept, and the deleted target is returned to the miss list. This policy is pursued until no further improvement is possible, at which point the construction of the target assignments for the next booster is begun.

In every case, when only one target can be added to a booster assignment, the target with the greatest value of RVAL, the marginal damage level, is added. The usual worth calculations are bypassed in this case, and all feasible additions are examined to select the target with the highest relative value.

Loading Requirements and Options: Program FOOTPRNT attempts, first, to construct detailed, ordered booster assignments which assign every available RV to a target point specified for a group in program ALOC, and second, to create these assignments such that each booster contains between MINLOAD and MAXLOAD RVs. In some cases, however, the footprint constraints preclude the possibility of accomplishing both or even one of these tasks. Therefore, in addition to being able to specify the values of MINLOAD, the minimum number of RVs to be assigned to any one booster, and MAXLOAD, the maximum number of RVs to be assigned to any one booster, the user has the option of specifying which of three alternative policies should be followed if the above objectives are not achieved.

The first of these policies, the free-loading option, specifies that, if all attempts have been made to satisfy the two requirements indicated above, the resulting allocations will be utilized, regardless of the number of RVs assigned to each booster. The second option stipulates that, when the final assignments contain boosters with fewer than MINLOAD RVs apiece, an attempt should be made to assign additional RVs to these boosters by targeting them at points already assigned to these boosters. The resulting allocation is then deemed acceptable, even if certain boosters have assigned to them less than the minimum specified load. The third and final option, however, specifies that if, after all attempts have been made to satisfy the minimum load requirements, certain boosters still have assigned to them fewer than MINLOAD RVs, these boosters should then be dropped entirely from the strike plan, and that the RVs which were originally assigned to them should not be used.

If the loading option requires addition of re-entry vehicles to the assignment, they are assigned after the program has constructed the best possible assignment disregarding the loading constraint. Assume that to meet this constraint a total of NTOADD vehicles must be added to the assignment.

The process begins by adding NTOADD re-entry vehicles to the first target in the footprint. If this allocation is not feasible, the program decrements the number of RVs added to the first target until it reaches a feasible allocation. There is no further processing for this allocation, since if a re-entry vehicle cannot be added to the first target of a footprint, it cannot be added to any later target.

If the total number of re-entry vehicles (NTOADD) could be added to the first target, the process searches for an alternative allocation with less variance in the number of RVs allocated to each target point. (The optimal allocation would have the same number of vehicles assigned to each target in the footprint.) The alternative allocations are constructed by examining the number of vehicles on each target. The targets are examined in order of delivery of their RVs by the final stage of the booster. At the first target where this number decreases (a decreasing step), a vehicle is removed and placed at the last target which has a number allocated less than the preceding target. Figure 13 demonstrates the construction of a series of alternative allocations. If at any time an alternative allocation is infeasible, the process reduces the number of targets to be investigated for addition of RVs to the current feasible number and continues processing.

Footprint Testing: The testing of potential target assignments for footprint feasibility involves solution of the physical equations which define the flight paths of MIRV systems. Since solution of these equations for all possible assignments would be prohibitively inefficient in the QUICK system, other approximating equations are used. These equations were generated for program FOOTCALL, which is not part of the QUICK system.* The equations were generated by a curve-fitting program so that their values most nearly match the results of the actual physical equations. The parameters for these equations, as well as their derivation, are discussed in the reference.

There are three MIRV systems for which testing equations are implemented in QUICK. The first is a long-range system similar to the MM-III system. The second is a short-range system, similar to the POSEIDON

*See "Strategic Offensive Weapons Employment in the Time Period About 1975 (U)," (Top Secret) Weapons Systems Evaluation Group Report R-160, August 1969, Volume VI, Allocation of MIRV Systems.

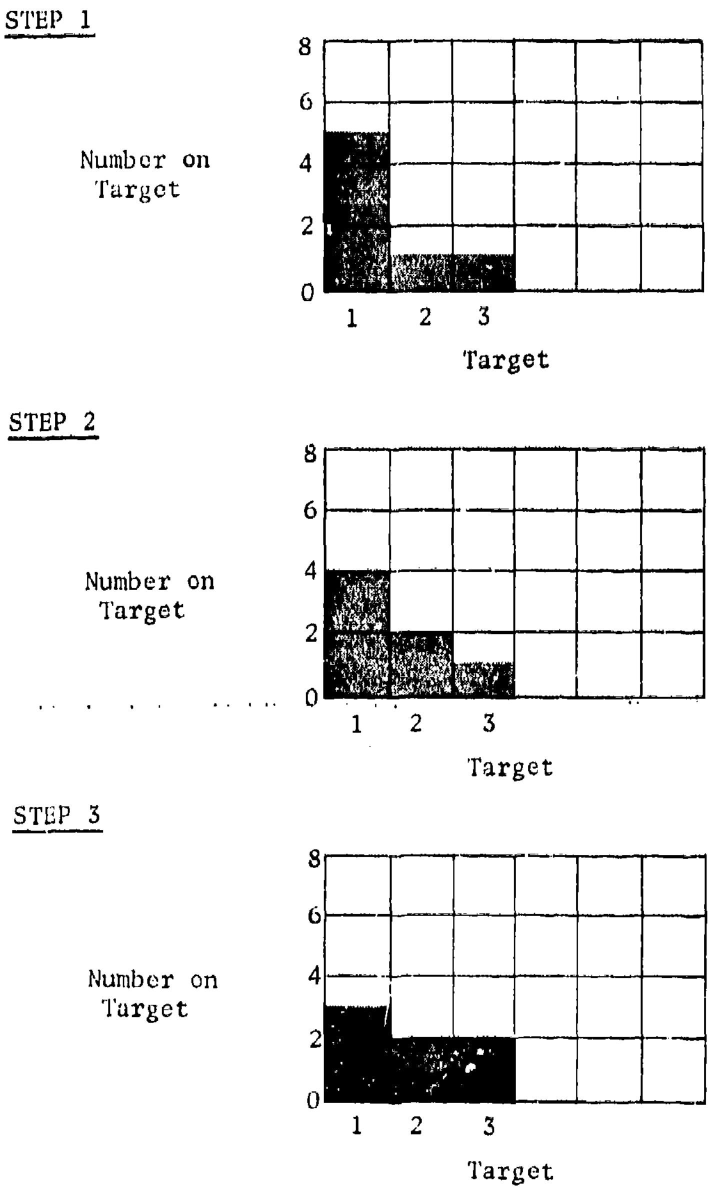


Fig. 13. Extra Re-Entry Vehicle Allocation Example

system. The third is a long-range system with area penetration aids carried. This last system is similar to the first, but contains somewhat different equations.

Each system has four sets of equations which define footprint feasibility. These sets are equations for:

1. Determining maximum booster range
2. Determining fuel load available for footprinting
3. Determining fuel consumption per mile of equivalent downrange distance
4. Determining factors for converting crossrange and uprange distances to equivalent downrange distances.

The specific form of the equations is discussed in Chapter 3, Calculations, Feasibility Testing for MIRV Footprints.

DETAILED SORTIE SPECIFICATIONS

Program PLNTPLAN processes the bomber and missile plans prepared by program POSTALOC and writes them with tanker plans in a format required by the QUICK Simulator. In addition, a detailed plan is output which reflects the plan in a form more suitable for hard-copy output. The detailed plan is also used as input to programs INTRFACE and EVALALOC.

Table 7 indicates the type of sortie information supplied to program PLNTPLAN for each sortie. Besides sortie identification, launch base, and vehicle information, it describes the target area part of the sortie by listing the target events. It lists the targets to be attacked, their location, and index numbers. It also lists ASM targets, decoy launches, and whether the bomber recovers or aborts the mission.

The major functions performed by PLNTPLAN in processing the input sortie data and creating the detailed sortie specifications include: assigning refuel areas to bombers and allocating tankers to service them; calculating ASM launch points; determining where zone crossing, change altitude, and launch decoy events should occur; coordinating launch times according to user parameters; and calculating distances and times between all events of each plan. The techniques associated with each of these functions are discussed below.

Bomber Plans

Figure 14 shows a typical flight route for a long-range bomber sortie from launch to recovery. After launching from its base, the bomber flies first to a refuel area if refueling is called for, then to a corridor entry point. It may then fly one or more prespecified doglegs (called corridor legs) which define a penetration route before reaching the point labelled corridor origin. From the origin, it flies over a target area and its assigned targets in their proper order. Finally, it enters the depenetration corridor, which may also consist of one or more doglegs,

Table 7. List of Information Supplied PLNTPLAN by POSTALOC for Each Sortie on STRKFILE

CATEGORY	ITEM
Sortie Identification	Group index Corridor index Sortie index
Base Information	Base index Base location (lat., long) Regional index Payload index Weapon type
Vehicle Information	Vehicle index Vehicle speed (at high and low altitude) Vehicle range (with and without refueling)
Sortie Information	Refuel index Depenetration corridor Alert status Delay before takeoff Number of targets Target list Low-altitude range available

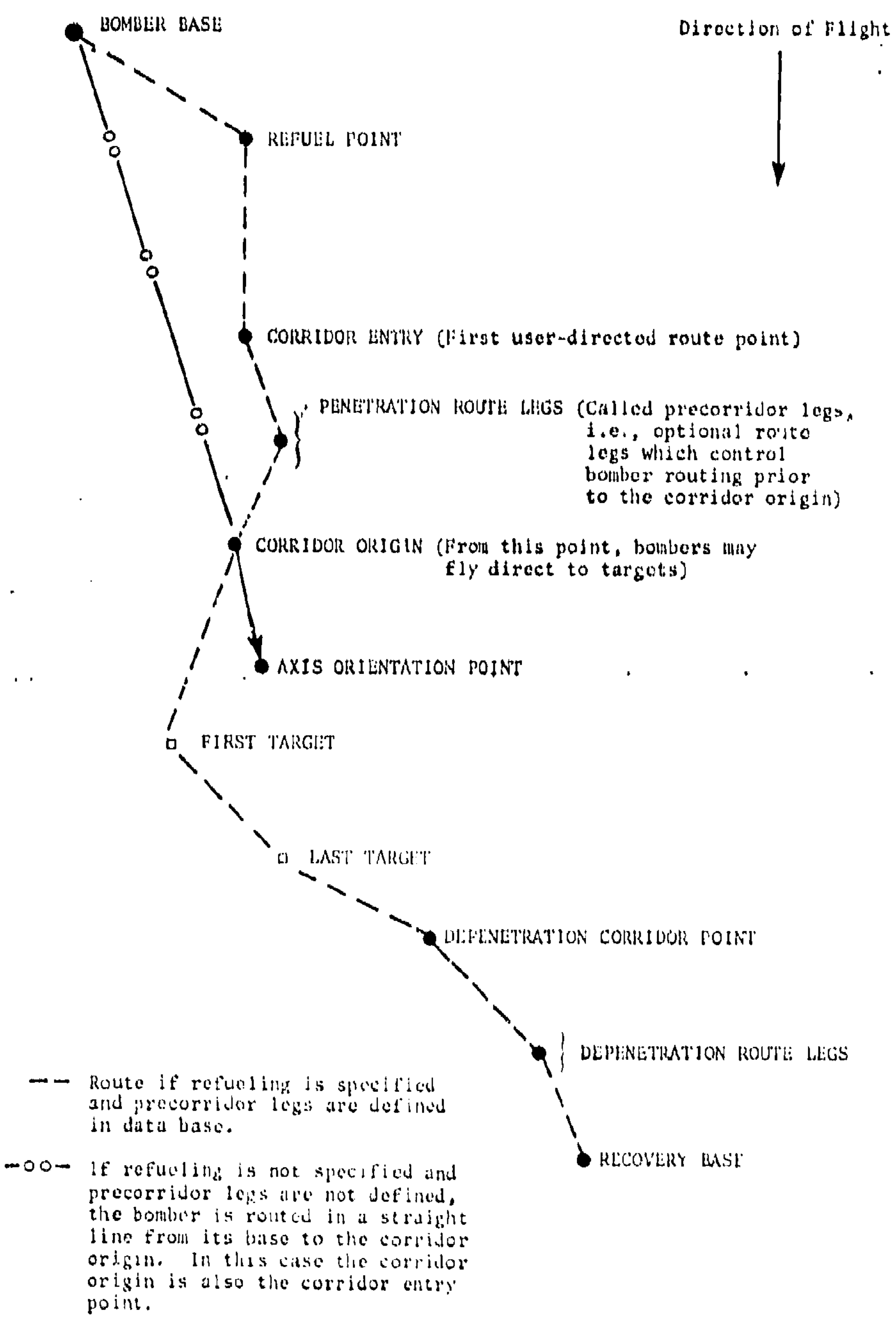


Fig. 14. Typical Bomber Flight Route

before going on to the recovery base. Depending on the bomber's range, a portion of the flight route may be flown at low altitude.

The air defense zones (to which the area air defense interceptors are assigned) are not shown in figure 14; however, between corridor entry and depenetration, the bomber may enter one or more of these zones. As indicated below, PLNTPLAN calculates the time and location of these zone crossing events.

Program PLNTPLAN generates a detailed sortie plan for each bomber which defines the flight route, altitude profile, and offensive operations. For bombers scheduled to refuel, an alternative plan is prepared to be used should the refueling be unsuccessful. The major PLNTPLAN functions and techniques involved in preparing the detailed bomber sortie data are discussed in the following paragraphs.

Distance Calculations: Each event of the bomber sortie is assigned a place of occurrence in latitude and longitude. Later, the great circle distances between all events are computed in nautical miles.

If the difference in longitude is less than ∿2.8 degrees, a straight-line approximation to the great circle route is used. Otherwise, the standard law of cosines for a spherical triangle is applied to compute the great circle distance. The radius of the earth is assumed to be 3437.74677 nautical miles.

This computation is sufficient for all events except for zone crossings, since zone crossings are located or determined only approximately on a Mercator projection. The adjustment to the distances in the case of zone crossings may be described by the illustration in figure 15. This shows the two zone crossing events Z_1 and Z_2 located between events E_1 and E_2. The distances between events are d_1, d_2, and d_3 as indicated. The great circle distance between E_1 and E_2 is D. In this case the distance d_1 would be replaced by $d'_1 = d_1D'$ where $D' = D/(d_1 + d_2 + d_3)$. Similarly $d'_2 = d_2D'$ and $d'_3 = d_3D'$.

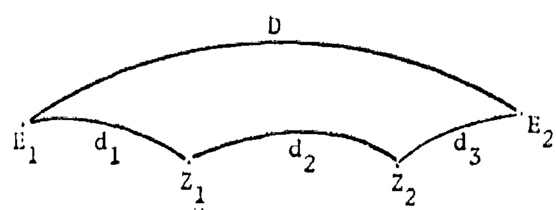


Fig. 15. Distance Adjustments for Zone Crossings

Bomber Timing: Using the calculated distances between events together with bomber (high or low altitude) speed or ASM speed, the time intervals between successive bomber events are computed. For the purposes of QUICK, each event of a plan is assumed to be carried out instantaneously at a specified time; i.e., a bomber is assumed to be launched in no time, to refuel without delay, and to change altitudes instantly. The calculated time between events takes into account the fact that zone crossing locations are known only approximately (as described for distance calculations); hence the uncertainty of the zone crossing points is not reflected in the corresponding time increments.

Actual times are determined in the first-strike case by coordinating the entire sortie with the user-input parameter CORBOMB which specifies the distance from the corridor entry point at which the bomber is to be at time zero. In the second-strike case, the sortie begins at the earliest possible moment, considering any user-specified launch delays. Tactical aircraft launch at time = 0.

Employment of Decoys: As each bomber plan is processed by the program, any flight situation which could use a decoy launch (see table 8) is flagged, and its associated launch priority is stored. The possible decoy launch events then are arranged by PLNTPLAN according to priority and are allocated available decoys in the order of this priority. If there are sufficient decoys available to cover all possible launches, double coverage is begun, again according to launch priority. Coverage continues until all decoys have been allocated or until six decoys have been launched at each possible site.

If the distance to be covered by a decoy launch event is greater than the range of one decoy, sufficient decoys are allocated to cover the entire distance. It is assumed that another decoy is launched as

Table 8. Launch Priority

LAUNCH PRIORITY	CIRCUMSTANCES OF LAUNCH
1	R_L* miles before first low-altitude gravity bomb attack on a SAM-defended target
2	Immediately before changing from high to low altitude
3	Immediately before penetrating defended airspace if flying at high altitude
4	R_H** miles before first high-altitude gravity bomb attack on a SAM-defended target
5	Coverage when flying at high altitude over defended airspace before priority 4 launch
6	R_L miles before subsequent low-altitude gravity bomb attacks on SAM-defended targets
7-8***	Coverage when flying at high altitude over defended airspace after priority 4 launch

*R_L = range of decoy at low altitude

**R_H = range of decoy at high altitude

***Priority 8 is used if the coverage is to begin at the point where the priority 4 decoy terminates. Priority 7 is used if the bomber has changed altitude between the priority 4 and the priority 7 launch.

soon as the previous decoy terminates. However, only the first launch event and the last termination event are posted, since intermediate launch-termination events essentially cancel.

Decoys launched at low altitude are assumed to terminate at their associated target. For high-altitude launches, either one or two termination events are required in addition to the launch event.

Changes in Bomber Altitude: The low-altitude range available to the bomber in executing the planned mission is allocated in program POSTALOC so as to maximize the value of the sortie VALSORTY (see Basic Sortie Generation). The actual latitudes and longitudes of the altitude change events (GOHIGH and GOLOW), and the associated time of the event, are calculated in PLNTPLAN.

The bomber's low-altitude range capability is specified to PLNTPLAN in three separate amounts: the amount during the precorridor legs (G_1), the amount immediately prior to the first target (G_2), and finally the amount following the first target (G_3). For realism, values of G_1, G_2, or G_3 equivalent to less than 15 minutes are not applied.

G_1 is measured backward from the corridor origin toward the corridor entry points. Since corridor attrition may or may not be associated with the precorridor legs, the low-altitude range capability is applied against only those precorridor legs where the bomber would experience attrition. Any G_1 remaining is added to G_2.

The initial go-low point after the precorridor legs is determined from the value of G_2:

1. If $G_2 > 0$, the go-low event will occur G_2 miles before the first target. Here, the first target is defined to mean the first bomb target on the first ASM launch point after the corridor origin.
2. For plans in which $G_2=0$, the bomber will go low at the first target, provided that the range to be flown at low altitude after the first target $(G_3) > 0$. If G_3 also equals 0, it will fly the entire mission after the corridor origin at high altitude.
3. If $G_3 < 0$, the bomber will fly $-G_2$ miles beyond the first target before going low; the total low-altitude range in this case is $G_3 - (-G_2) + G_1$ miles.

G_3 is always measured out beginning at the first target, and any G_3 remaining after the target area is applied during depenetration.

The location of the change-altitude points are initially determined by applying G_1, G_2, and G_3 as outlined above. Once the initial processing is completed, the GOHIGH and GOLOW locations are checked to ensure that the bomber does not change altitude in an unrealistic manner. If necessary, as explained below, the location of these points is modified.

For the purposes of the QUICK system, each event of a plan is assumed to be carried out instantaneously at the indicated time; i.e., a bomber is assumed to be launched in zero time, to refuel with no delay, and to change altitude instantaneously. Thus, if the bomber is asked to go high or go low in the immediate neighborhood of a target or ASM launch point, the order of these events does not matter. However, the detailed plan appears more realistic if the bomber climbs immediately after, rather than immediately before, a target and goes to a low altitude immediately before, rather than immediately after, a target.

Program PLNTPLAN adjusts the plan to make certain that this is the case. The adjustment performed is seen by referring to figure 16 where the high-altitude adjustment is shown. If a bomber is found to climb within THB minutes before a target (in which case it might be thought of as flying a path shown by the solid line in the figure), then the altitude change event is moved so that it occurs THA minutes after the target (in which case it might be thought of as flying the path shown by a dotted line). Similarly, the low-altitude adjustment is indicated in figure 17. Here, if the bomber is scheduled to go low within TLA minutes after the target, this event is moved so that it goes low TLB minutes prior to the target. The parameters shown in the figure may be preset to any value. The current setting of these parameters in program PLNTPLAN follow.

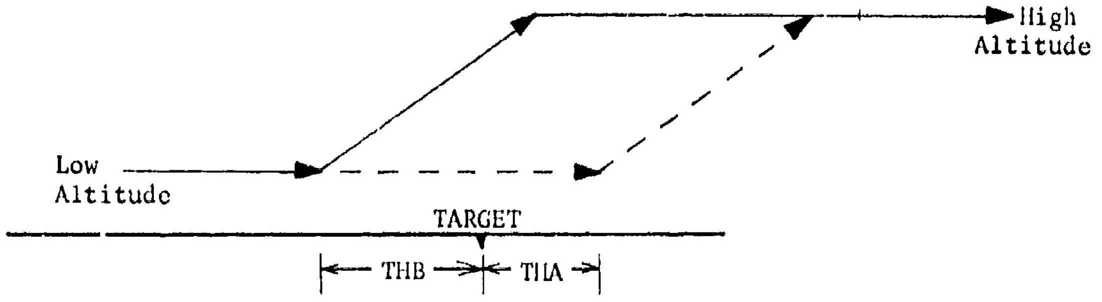


Fig. 16. High-Altitude Adjustment

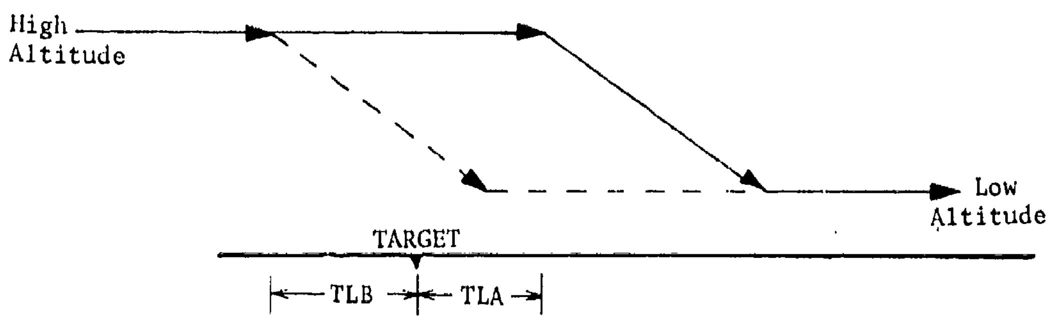


Fig. 17. Low-Altitude Adjustment

PARAMETER	DESCRIPTION
THB=15 THA=2	The time before (THB) and after (THA) a target or ASM launch point during which the bomber may not change from low to high altitude
TLB=10 TLA=3	The time before (TLB) and after (TLA) a target or ASM launch point during which the bomber may not change from high to low altitude

In making these adjustments, the amount of low-altitude flight is never decreased, but it may be increased as illustrated in figure 18. It shows two targets labeled T_1 and T_2 with associated values of the parameters THB and THA. A section of bomber path is shown by dashed lines. In this case, a GOHIGH event found, say, at point p would be moved first to point q and finally to point r. The time of low-altitude flight would be increased in this case at almost twice the sum of THB + THA. For this to occur, the targets would have to be within THB + THA minutes of flying time.

ASM Launch: Whenever an ASM target is processed (as indicated in the basic plan), PLNTPLAN computes the aim or launch point at which the ASM assigned to the target is to be fired. The situation is illustrated in figure 19.

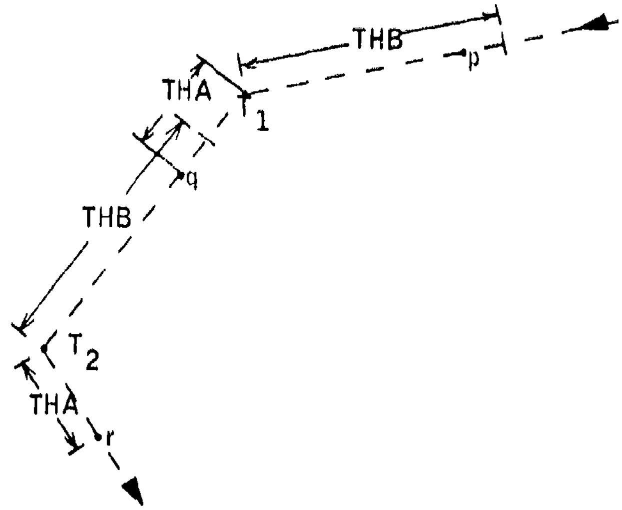


Fig. 18. Increase in Low-Altitude Flight

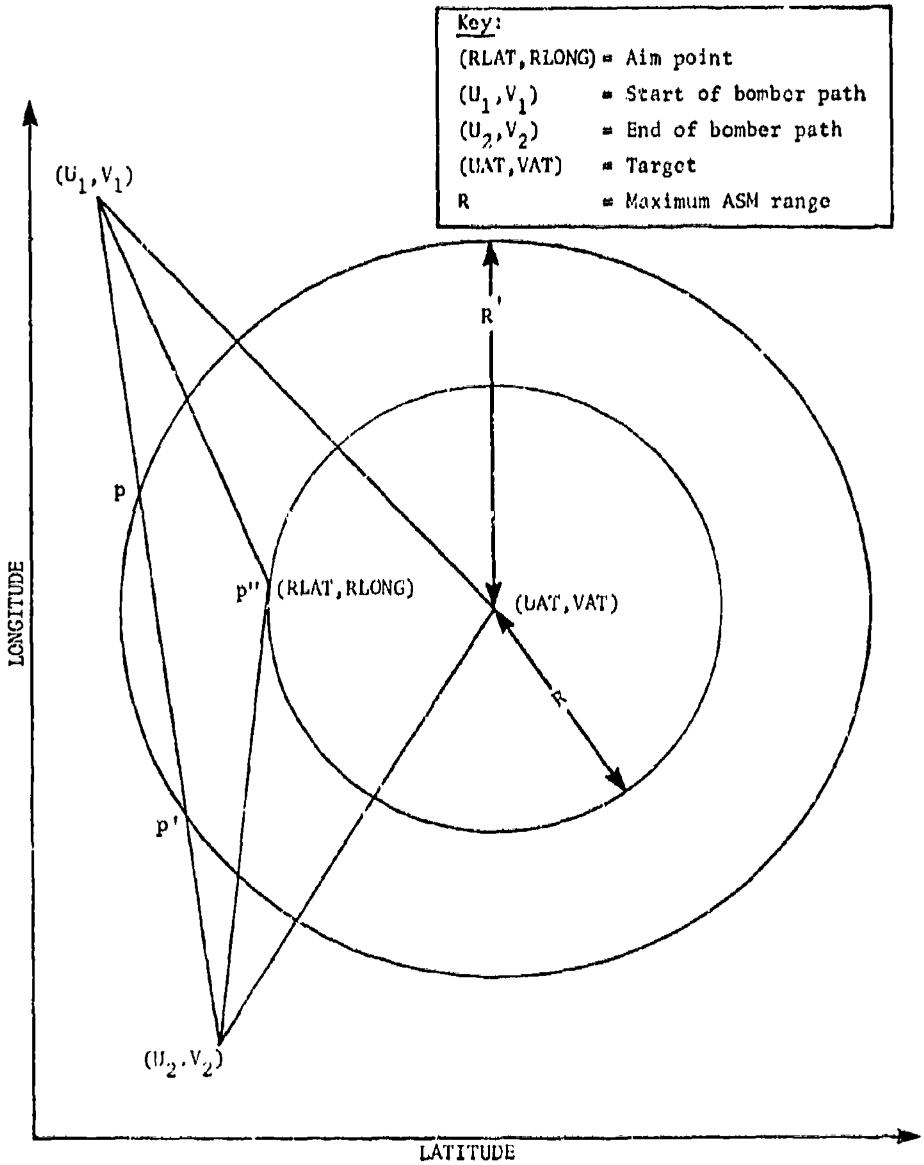


Fig. 19. Illustration of ASM Launch Point Calculation

The bomber is flying from a specified point (U_1,V_1) to point (U_2,V_2) and is to fire an ASM at a target (UAT,VAT) enroute, at maximum range R if possible. The aim point to be determined is (RLAT,RLONG). In determining the point (RLAT,RLONG), two cases occur:

1. The simpler case exists when the range of the ASM is sufficient for it to be launched while the bomber is proceeding in a straight-line path from point (U_1,V_1) to (U_2,V_2). This would be the case if the range of the ASM were R' (figure 19). The ASM target is then said to be "in range." Since it could be launched at maximum range from either point p or p' shown in the figure, the point p would be chosen as the desired launch point. Since point p is a point enroute, it is not considered to be a flypoint.
2. The more interesting case occurs when the range of the ASM is equal to R in figure 19. Here, the bomber must deviate from its course and fly to the point p" to fire the ASM. The ASM target is noted as "out of range," and the point p" is now a flypoint.

Subsequently, during allocation of low-altitude range, any ASM launch scheduled to occur at the corridor origin will be rescheduled to occur five minutes later if the aircraft is also to change to a low altitude at the origin.

Refueling: See Bomber Refueling.

Recovery: The list of targets for a bomber terminates in either of two ways:

1. With a DEPEN event, indicating normal recovery to the most distant of the four recovery bases associated with its depenetration point, that the bomber can reach. The depenetration corridor description is obtained from the system's input data, and the bomber's dogleg events, if any, are posted in the proper order. Any remaining low-altitude range (G-3) is applied at this time. Checks for zone crossings are also made, and zone events are posted as appropriate. If a zero zone is encountered in zone processing, indicating that the bomber has left the area in which there are defense zones, the zone crossing check is turned off, and no further check for zone crossings is made.
2. With a LAND event, which indicates the aircraft does not have sufficient range to recover. In this case an ABORT event is posted for the bomber five minutes after the last target in the direction of the assigned depenetration corridor.

Missile Plans

The input missile plans prepared by POSTALOC are complete with the exception of the launch and flight times associated with the mission.

These calculations are performed in program PLNTPLAN, and the appropriate data are added to the basic missile plan.

Missile flight times and launch times are calculated from user inputs. The main timing parameters used are the minimum flight time (FLTMIN) and the coordination time for missiles (CORMSL).

FLTMIN, the minimum flight time for a missile type, may be equal to or greater than zero. All flight times less than FLTMIN will be raised to FLTMIN before the launch and impact times are posted to the missile plan.

The user may specify a CORMSL* for each missile type. This parameter will control the launch timing for initial strikes (INITSTRK=1). There are two kinds of CORMSL: a "FLIGHT" CORMSL and a "LINE" CORMSL.

A "FLIGHT" CORMSL is the fraction of the missile's flight which is completed at time 0.0. Clearly, such a CORMSL must lie between 0 and 1. If it is 0, the missile is launched at time = 0. If it is 1.0, the missile impacts at time = 0.

The "LINE" CORMSL requires another user input. The user first specifies a sequence of straight-line segments (not necessarily connected). The "LINE" CORMSL is then the time at which the missile first crosses any line. If the flight path does not cross any line, then the missile will

---

*In program ALOC, a single value for the parameter CORMSL is applied to all missile types.

impact at time = 0. Because of the great length of missile paths, great circle routes are used for the lines and the flight paths, rather than a Mercator projection of coordinates. The timing calculations which involve "LINE" CORMSLs are described in Chapter 3, Calculations.

If the missile is a fixed weapon with a specific time of arrival specified from program ALOC, this time is used to compute the launch time, regardless of any CORMSL. The CORMSL parameter is also ignored for second-strike plans.

In the case of missiles with a MIRV capability, if there are several targets assigned to the missile and more than one has a "fixed time" assigned, only the first fixed-time assignment encountered will be considered. Thus, if a previous fixed-time assignment has determined the launch time for the missile, no further calculations are done to compute the launch time for later re-entry vehicles on the missile. If there are no fixed assignments (with timing) on a missile with a MIRV payload, the launch time is computed by considering only the data for the target assigned to the first re-entry vehicle on the booster.

Tanker Plans

In addition to defining the basic missile and bomber plans developed by POSTALOC, PLNTPLAN generates the tanker plans for tankers used in an area refueling mode. The input data for tankers are obtained from the

BASFILE (prepared by PREPALOC) and include:

1. Tanker base latitude and longitude
2. An index specifying either the refuel area to which it is to be directed or its availability for automatic allocation by PLNTPLAN
3. Number of tankers per squadron
4. Number of tankers on alert per squadron
5. Tanker speed (referred to below as V_t)
6. Alert delay
7. Nonalert delay
8. Total time on station (TTOS)
9. Tanker type
10. Tanker range.

After all bomber plans have been formulated, a plan for each tanker is generated consisting of the seven events shown below:

EVENT TYPE	TIME BETWEEN EVENTS	PLACE
Launch	Delay	Tanker Base
Enter Refuel Area	$DIST/V_t$	Refuel Area
Leave Refuel Area	TTOS	Refuel Area
$Recover_1$	DI_1/V_t	Nearest recovery base
$Recover_2$	DI_2/V_t	2nd nearest recovery base
$Recover_3$	DI_3/V_t	3rd nearest recove y base
$Recover_4$	DI_4/V_t	4th nearest recovery base

where DIST = Distance from tanker base to refuel area

DI_x = Distance from refuel area to recovery $base_x$

First, PLNTPLAN assigns a refuel area to each tanker that is not user-directed to a specific area. This is done in such a way as to minimize the total tanker miles flown while servicing all bomber requests. The time of arrival at the refuel area differs depending on whether the plan is for a first or second strike.

In the second-strike case, all tankers are sent to their assigned refuel areas at the earliest possible moment, considering delays before launch due to alert or nonalert status as well as the travel time required between base and refuel area.

In the first-strike case, each tanker is scheduled to enter its assigned refuel area .1 hour prior to the arrival of the bomber that it is to service. The tanker launch time, then, is computed by:

$$\text{Launch time} = (\text{time due at refuel area}) - \left(\frac{DIST}{V_t}\right)$$

Each tanker is scheduled to leave the refuel area TTOS (total time on station) hours after arriving. The four recovery bases closest to the refuel area are found, ordered by ascending distance, and posted for each tanker as alternate recovery events. The flight times from refuel area to each recovery base are determined and the tanker plan is complete.

Program EVALALOC was created to enable the user to determine the effectiveness of a given war plan without the execution of a complete and detailed simulation. It develops an expected-value estimate of the results of the allocation which reflects only the effects of blast damage. Except for program SIMULATE in the Simulation subsystem of QUICK, EVALALOC is the only damage assessment program in QUICK.

The expected-value estimates of damage provided by EVALALOC are realized as two distinct kinds of numbers; the first is a non-time-dependent value of target destroyed, and the second is the corresponding time-dependent value of target destroyed. Directly related to the calculations of these two numbers are the calculations of expected values of delivered weapons and of delivered megatonnage, both of which affect the time-dependent value of target destroyed. Other weapon and target characteristics are used by EVALALOC in its calculations of expected target damage. Weapon characteristics are time of arrival, reliability, circular error probability, penetration probability, yield, and function of the weapon (alert long-range aircraft, tactical bomber, intercontinental ballistic missile, etc.). Target characteristics are the terminal ballistic missile defense capability and the related probability of penetrating such a target defense, the weapon aim points, and the type of target (simple, multiple, or complex) including its radius. Such factors as collateral damage, weapon correlations, and other electromagnetic and radioactive phenomena are not reflected in the QUICK system estimate of expected damage provided by EVALALOC. However, other damage assessment systems for which QUICK can produce output do provide estimates which reflect these kinds of factors.

CHAPTER 3
CALCULATIONS

This chapter provides a detailed mathematical explanation of the more complex algorithms associated with the Plan Generation subsystem. It should be noted that the general description of these techniques, included in chapter 2, is not repeated in this chapter. Therefore, the reader should be familiar with the preceding chapters of this manual.

ALGORITHMS

Target Shuffling

To achieve efficient performance of program ALOC, the input target list is shuffled in program PLANSET to distribute the order in which the various type targets will be encountered. This is accomplished in the following manner.

Consider the target indices (I) as equally spaced points on a circle, with targets in a particular class occurring consecutively. If the I^{th} point is displaced along the circle to the index

$$\frac{1}{2}\left(3 - \sqrt{5}\right) C * \left(I\right) \qquad \text{mod } C$$

where C is the number of points on the circumference of the circle, the result will be the desired distribution. To accomplish the corresponding reordering of the discrete list of targets, each index must be multiplied by

$$\frac{1}{2}\left(3 - \sqrt{5}\right) N$$

where N is the number of elements in the list, and then reduced by modulo N. A direct application of this procedure, however, will result in some cases with the same final index being assigned to more than one element.

Therefore, the following algorithm is used by program PLANSET to assign new indices to the elements of the list in such a way that the above criterion is satisfied and each index from 1 to N is assigned exactly once.

Let L be the greatest integer such that $L \leq \frac{1}{2}\left(3 - \sqrt{5}\right) N$

where N is the total number of targets.

Let P = J = L ,

where P = the beginning index of the current cycle, and

J = the index number currently being assigned.

As each list element is processed, J is replaced by J + L to obtain the next index number. If J becomes > N, J is replaced by J - N. If J becomes = P, a new cycle is to be started; 1 is added to P and to J, and the procedure continues as before.

Lagrange Multiplier Adjustment

Define the following variables:

SURPWP(G) = number of excess weapons allocated in group G

CURSUM	=	sum of the target weights
NOWPS(J)	=	number of weapons sharing attribute J
NTGTS	=	number of targets
SNSTVTY FSNSTVTY	=	user-input parameters which control rate of multiplier adjustment
LAMEF(G)	=	Lagrange multiplier for group G
PRM	=	local internal control variable which governs size of premiums (closing factors)
NWPNS(G)	=	number of weapons in group G
CTMULT	=	current target multiplicity

The method used to adjust the Lagrange multipliers proceeds as follows.

At every fourth target or so, when it is decided to recompute the multipliers, control passes to an adjustment routine. The first step is to recompute all the allocation error estimates, ALERREST. At the same time SURPWP is reevaluated, based on the new value of ALERREST. Although SURPWP is continuously updated by the operating program, it is useful--especially in the early phases of the program--to base it on the projected allocation-rate estimates rather than the actual weapons allocated, which at that time could be very misleading. This provides a more rational basis for calculating the premiums at this early stage of the program.

The adjustment phase processing is determined in part by an internal variable, PROGRESS. This variable is assigned the arbitrary values 0.,

.4, .5, .75, 1., and 2. by the program as a flag for various stages of the allocation process. PROGRESS is initially set to 0. at the start of processing by program ALOC. When the sum of target weights, WTSUM, exceeds half the number of targets PROGRESS is set to 0.4. When the weight change rate (WRATE, described later in this section) first decreases, PROGRESS is set to 0.5. When the weight change rate decreases to zero value, PROGRESS is set to 0.75. A user-input parameter, SETTLE, determines the next change. SETTLE is the number of passes the process continues with PROGRESS equal to .75. After this time PROGRESS is set to 1.0. PROGRESS remains at this value until one of three conditions is met:

1. More than 1.5 passes over the target set are made while PROGRESS = 1.0;
2. The sum of the Lagrange multipliers for the under- or over-allocated weapons (VALERR) is less than a fraction (ERRCLOS, a user input parameter) of the sum of the Lagrange multipliers for all the weapons in the stockpile (VALWPNS);
3. The sum of the squares of the allocation error estimates (SUMSQERR, the sum of the squares of ALERREST, described later in this section) is less than $1/(10 * NTGTS^2)$, where NTGTS is the number of targets.

When any of these three conditions is met, the allocation process is complete and PROGRESS is set to 2.0.

If PROGRESS = 1.0 the change of local multipliers is omitted so that the same values of the multipliers are retained. Otherwise the program determines the change in the local multipliers. Each multiplier is changed only if all the estimates of error rate have the same sign. In the early phases of the program (PROGRESS < .75) better stability is achieved by requiring, in addition, that the average allocation rate to the last 2 to 4 targets, as computed from CURSUM, show the same sign. This limitation is later removed, since it clearly would not work well for weapon groups with very small numbers of weapons that might only be allocated 2 to 10 times during a pass over the target system.

An estimate is made of CORRATE, the rate at which it is desired to correct the allocation rate. If the allocation rate is corrected too rapidly there will be a tendency to over-correct before the effects of the correction become observable in the values of the allocation error estimates. This can produce oscillations. To estimate how rapidly to correct the error, an estimate is made of the number of targets that would have to be observed before an error of the observed size would be statistically significant. Even if the multipliers were exact, and the average allocation rate was correct, statistical fluctuations would be observed in the allocation of each weapon group when the allocation rate was sampled for a small number of targets.

Let n equal the expected or average number of weapons from a group available per target; i.e., n = NOWPS(J)/NTGTS. Then in M targets the expected number of weapons allocated should be just n(M). Suppose the actual number observed, however, is n'(M). Then our estimate of the error in the allocation rate ALERREST would be

$$ALERREST = n' - n$$

Assuming a Poisson distribution, the statistically expected error in a number of expected value n(M) is equal to $\sqrt{n(M)}$. That is,

$$\langle (n'(M) - n(M))^2 \rangle = n(M)$$

$$\langle (n' - n)^2 \rangle = n/M$$

Solving for the number of targets M, we have:

$$M = n / \langle (n' - n)^2 \rangle$$

or

$$M = (NOWPS(J)/NTGTS)/(ALERREST(J))^2$$

as the number of targets we should expect to sample to get a statistical error estimate of size, ALERREST. If we wish to reduce the indicated error by 1 part in M per target, our fractional correction in the allocation rate per target should be:

1/M = ALERREST ** 2/(NOWPS(J)/NTGTS)

This, multiplied by a sensitivity factor SNSTVTY, is the first term in the value of CORRATE. However, if the entire set of targets were observed, the estimate would not be a sample but would be exact. Therefore, even a very small value of ALERREST becomes statistically significant if it is based on a sample of size NTGTS. Therefore, errors should always be corrected at a rate at least equal to one part in NTGTS.

This explains the second term in CORRATE, which is just 1.0/NTGTS multiplied by a sensitivity factor FSNSTVTY (final sensitivity). This factor controls the sensitivity of corrections to the allocation rate in the final phase of the allocation where the errors are small. Thus the desired correction rate is just:

```
CORRATE = SNSTVTY*ALERREST**2/(NOWPS(J)/NTGTS) + FSNSTVTY/NTGTS
```

This is multiplied by the number of targets processed between corrections, MULSTEP, to determine the fraction CORFAC of the error to correct. In addition, a safety limit of 1/2 is used to avoid ever making a correction larger than 1/2 the estimated error rate.

However, even when it is known what fraction of the error in the allocation rate we wish to correct, an estimate must be made of the relationship of the allocation rate to changes in the Lagrange multipliers before the size change to make in the multiplier can be estimated. For this purpose it is useful to have a model of the dependence of the allocation rate on the value of the multipliers. We have assumed a dependence as follows:

$$\text{Rate} = k\,\lambda^{-n}$$

Consider now two rates, the current rate R_0 associated with a multiplier λ_0 and a predicted rate R_1 associated with a new multiplier λ_1. Thus we find

$$R_1 \lambda_1^{\,n} = R_0 \lambda_0^{\,n} = k$$

or

$$R_1/R_0 = (\lambda_1/\lambda_0)^{-n}$$

so

$$\frac{\alpha(R_1/R_0)}{\alpha(\lambda_1/\lambda_0)} = -n$$

For small differences between λ_0 and λ_1 this implies:

$$\frac{R_1 - R_0}{R_0} = -n \frac{\lambda_1 - \lambda_0}{\lambda_0}$$

Solving for the new value λ_1 of λ

$$\lambda_1 = \lambda_0 (1 + \frac{(R_1 - R_0)/(-n)}{R_0})$$

If we now identity a new variable R_2 as the ultimately desired allocation rate, R_1 as the new rate we hope to obtain with λ_1, and R_0 as the current allocation rate--then the above variables can be associated with information already available as follows:

$$R_1 - R_0 = \text{CORFAC}*(R_2 - R_0) = \text{CORFAC}*\text{ALERREST}$$

$$R_0 = \text{ALERREST} + (\text{NOWPS}/\text{NTGTS})$$

If we now associate the variable PARTIAL with n and the local multiplier LA with λ this gives rise to the following procedure for updating LA:

$$LA_1 = LA_0 * \left[1.0 + \frac{\text{CORFAC*ALERREST(J, INTPRD)/(- PARTIAL)}}{\text{ALERREST(J, INTPRD) + (NOWPS(J)/NTGTS)}}\right]$$

This formula is well-behaved if ALERREST is large and positive, but if it is negative and as large as the expected rate (NOWPS(J)/NTGTS) (i.e., if the actual allocation rate is zero), then the denominator goes to zero. In this case an infinite correction would be indicated. To avoid this, the expected rate in the denominator is multiplied by 2 giving:

$$LA_1 = LA_0 * \left[1.0 + \frac{\text{CORFAC*ALERREST(J, INTPRD)/(- PARTIAL)}}{\text{ALERREST(J, INTPRD) + 2*(NOWPS(J)/NTGTS)}} \right]$$

This is the function used.

In the present version of the program the value of PARTIAL(J) has been set equal to 1.0 for all the local multipliers LA(J). This choice is based on the effect of the premium on the sensitivity of the allocation rate to the value of LAMEF or λ. When the multipliers are almost correct, it is usually the case that most weapon groups are in close competition with many other groups with very similar properties. Then a small change in the multiplier LAMEF will produce a very large change in the allocation rates, as the weapon group in question almost totally replaces, or is replaced by, its competitors.

However, such a large error in the allocation rate will not actually occur because as the error builds up the estimated value of the payoff will be automatically changed by the premium. Thus for constant values of LAMEF, when an equilibrium allocation rate is reached, it must be approximately true that the error in LAMEF is compensated by the premium. That is, if λ_0 is the correct value for LAMEF then:

$$\text{LAMEF} - \text{PREMIUM} \cong \lambda_0$$

Since:

$$\text{PREMIUM} = \text{PRM*LAMEF*}\frac{\text{SURPWP} - .5\text{*CTMULT}}{\text{NWPNS}}$$

we can define a relation between LAMEF and (SURPWP/NWPNS)

$$\text{LAMEF*}(1 - \text{PRM*}\frac{\text{SURPWP} - .5\text{*CTMULT}}{\text{NWPNS}}) \cong \lambda_0$$

Since this relationship is the same for all groups it is reasonable simply to use the same value 1.0 of partial derivative for all local multipliers.

The values of LAMEF(G) are recomputed using the new values of the local multipliers LA(J). At the same time it is necessary to reevaluate the summation of the value of all the weapons VALWPNS = $\sum$LAMEF(G)*NWPNS(G) and the summation of the value of the error in weapons allocated

VALERR = $\sum$LAMEF(G)*ABSF(SURPWP(G))

using the updated values of LAMEF. The average number of targets over which allocation rates are averaged (the integration period) is determined by the rate at which the target weights are increased.

In estimating the rate with which to correct multipliers, it was computed on a statistical basis that even if the allocation rates were correct an estimated error of size ALERREST would be expected if the allocation rates were monitored only over a small sample of M targets where:

$$M = (\text{NOWPS(J)/NTGTS})/(\text{ALERREST(J)})^2$$

Thus if separate integration periods could be used for each local multiplier, M as defined above might provide a reasonable basis for determining the period. However, in fact, the same three periods (INTPRD = 1, 2, 3) must be used for all local multipliers LA(J). Consequently the value of the integration period used must be based on an estimate of overall error rate. The corresponding relation is:

$$M = \left(\sum_G \text{NOWPS(J)/NTGTS}\right) / \sum_G (\text{ALERREST(J)})^2$$

where the summations are taken over all weapon groups. The quantity $\sum_G$NOWPS(J), is identical with NOWPS(2) (Note: LA(J) for J = 2 is used for all weapon groups) and so for efficiency the variable NOWPS(2) is used. While the expected value of $(\text{ALERREST(2)})^2$ is the same as $\sum_G (\text{ALERREST(J)})^2$, the variance of the latter version is much less, and it is therefore preferable as an estimator of the expected integration period, EXPINTPD.

To allow the possibility of using integration periods either longer or shorter than the theoretical EXPINTPD, a desired longest integration period DESINTPD is defined:

DESINTPD = EXPINTPD*RATIOINT

where RATIOINT is an adjustable input parameter.

If this period were used exactly in setting the rate of change of the target weight (i.e., WRATE = 1.0/DESINTPD), the WRATE would never become exactly zero as is required for a constant target weight. Obviously when the change in the target weight becomes small over a full pass, the

WRATE should be allowed to go to zero. Therefore in:

WRATE = (1.0/DESINTPD) - (2.0/NTGTS)

the term 2.0/NTGTS is subtracted, and if the resulting WRATE is negative it is set to zero. To avoid a situation where large errors cause the integration period to become ridiculously small, a limit that WRATE $\leq$ .07 is set.

Moreover, after the allocation is well under way, PROGRESS $\geq$ .5, the value of WRATE is not allowed to increase. In the program, WRATE(INTPRD) is used as a multiplier of the target weight; therefore we add 1.0 to WRATE to obtain a suitable multiplier for the longest period NINTPRD.

The values of the WRATE for the shorter periods are then derived from this value to give a ratio of integration periods roughly equal to RINTPRD, another input parameter.

Derivation of Formula for Correlations in Weapon Delivery Probability

An exact calculation of the probability of target survival when it is subject to attack by correlated weapons is very lengthy. Both the conventional statistical analysis and the Bayesian incremental information approach have been examined. Both approaches for each time and hardness require the calculation component of the interaction terms between each weapon to be added with all possible combinations of the weapons already on the target. Thus the completely rigorous calculation would be impractical in a rapid response allocator. The method used here is based on an approximation derived from the properties of the log-gamma distribution.

When a group of weapons share a common failure risk the probability of success is likely to be either high or low for all weapons collectively. Thus the probability of success can itself be thought of as a random variable. For any chance value of this overall random variable there will exist the usual independent probabilities for individual weapons. However, on one trial the overall success probability for the group of weapons may be 90%, while in another trial it may be 50% depending on the particular success probability drawn for the trial.

The following mathematical model has been developed to deal with this type of problem. We assume that the probability of survival of a target with respect to the i^{th} weapon is itself a random variable S of the form

$$S_i = e^{-X_i}$$

where the X_i are random variables drawn from a known distribution.

If two weapons are involved, then the probability of survival with respect to both can be represented by the random variable S_T:

$$S_T = S_i S_j = e^{-(X_i + X_j)}$$

However, the random variables X_i and X_j may or may not be independent. If they are not independent then of course

$$\langle S_i S_j \rangle \neq \langle S_i \rangle \langle S_j \rangle$$

If the X_i are independently drawn from a known two-parameter family of distributions with a convolution property,* then the distribution of $X_i + X_j$ will of course be a member of the same distribution family. Moreover, since any probability distribution for the X_i implies a distribution for the corresponding S_i, the distribution for $S_i S_j$ can be calculated and the value for $\langle S_i S_j \rangle$ can be computed.

The gamma distribution given by:

$$P(X)dx = \frac{X^a e^{-X/b}}{b^{a+1}\Gamma(a+1)} dx \text{ for } X \geq 0$$

$$P(X) = 0 \text{ for } X \leq 0$$

is a well known two-parameter distribution with the required convolution property.

The gamma distribution is unique among convolving two-parameter distributions in that the expected value of e^{-X} is easily computed. This property is particularly important for QUICK since the damage function performs a

*A probability distribution is said to "convolve" when the convolution of any two distributions in the family (i.e., the distribution of the sum of the two random variables) is itself a member of the same family.

computation of this value many times during the allocation. The expected value of e^{-X} is given by:

$$\langle e^{-X} \rangle = \int_0^\infty P(X)e^{-X}dX$$

which can be written

$$\langle S \rangle = \langle e^{-X} \rangle = \left(\frac{1}{b + 1}\right)^{a + 1}$$

This distribution is valid for $b > 0$ and $a > -1$. It has a mean $\mu = b(a + 1)$ and a variance $\sigma^2 = b^2(a + 1)$.

Since this distribution is completely defined by the mean and variance, the actual probability distribution of S can be computed at any time so long as a record of the mean and variance of the distribution is maintained. We now observe that:

$$a + 1 = \mu^2/\sigma^2$$

and

$$b = \sigma^2/\mu$$

so the expected value of S can be written

$$\langle S \rangle = \left(\frac{1}{\frac{\sigma^2}{\mu} + 1}\right)^{\mu^2/\sigma^2}$$

or

$$-\ell n \langle S \rangle = \frac{\mu^2}{\sigma^2} \ell n \left(\frac{\sigma^2}{\mu} + 1\right)$$

This distribution is sufficiently flexible to include almost any shape distribution of interest. For σ small the distribution in S approximates a gaussian centering on some specific survival probability. As the σ is increased the distribution widens, so that it can approximate a uniform probability from zero to one, or a sloping probability with more weight on zero or one. In the limit of very large σ the distribution consists essentially of spikes of different weight at zero and one.

If we were dealing with independent weapons we could calculate the parameters for the multiple weapon distribution from those for the single weapon distributions simply by making use of the additivity of the mean and the variance. Specifically the mean, μ_T, for the new distribution and the variance σ_T^2 would be given by:

$$\mu_T = \sum_i \mu_i$$

$$\sigma_T^2 = \sum_i \sigma_i^2$$

The expected value of target survivability S_T for the new distribution would then be obtainable through the equation:

$$- \ell n \langle S_T \rangle = \frac{\mu_T^2}{\sigma_T^2} \ell n \left[\left(\frac{\sigma_T^2}{\mu_T} \right) + 1 \right]$$

However, the variance is directly additive as above only if the weapons are really independent. To introduce the possibility of correlations we will write the variance as follows:

$$\sigma_T^2 = \sum_i \sum_j \sigma_i \Gamma_{ij} \sigma_J$$

where the quantity Γ_{ij} represents the correlation between the weapons. In the special case of uncorrelated weapons, $\Gamma_{ij} = 0$ for $i \neq j$ and 1 for $i = j$, which is identical with the previous form.

This approach of arbitrarily introducing the cross terms in this formulation to approximate the actual correlations is exact so long as the correlations are of such a form that the distribution of X remains a gamma distribution. To the extent that the actual correlations cause departures from the Γ distribution the approximation is in error. The correlation model thus amounts to the assumption that correlations can be adequately modeled without going outside the log-gamma distribution.

For implementation it seems appropriate to introduce an additional simplification. In the foregoing formulation the magnitude of the penalty for using correlated weapons will depend not only on the size of the correlation and the kill probability for the correlated weapons, but also on the shape of the distribution for the success probability for each weapon. This shape dependence introduces a complicating variable which undoubtedly exists, but for which it would not be easy to get data. It therefore seems desirable to eliminate this factor.

This can be done by standardizing on a single shape factor for all calculations of the effects of correlations. It is easiest to do this by considering only distributions with a very large σ, which are essentially spikes on zero and one. This choice tends to exaggerate the importance of correlations (and this fact should be borne in mind in assigning the correlations for the war game) but it significantly simplifies the data required, as well as the computation of the payoff.

In the limit of large σ the quantity σ_i^2/μ_i approaches infinity while the quantity μ_i^2/σ_i^2 compensates to maintain the correct value of $-\ell n \langle S_i \rangle$

To illustrate the transition to this limit we let $b_i = \sigma_i^2/\mu_i$ and define

$$\beta_i = b_i/\ell n(b_i + 1)$$

Then

$$-\ell n \langle S_i \rangle = \mu_i/\beta_i$$

so:

$$\mu_i = \beta_i \left[-\ell n \langle S_i \rangle \right]$$

and

$$\sigma_i^2 = \mu_i b_i = b_i \beta_i \left[-\ell n \langle S_i \rangle \right]$$

The formula for obtaining the expected value of S_T can now be written

$$-\ell n \langle S_T \rangle = \frac{\mu_T^2}{\sigma_T^2} \ell n(b_T + 1)$$

and substituting,

$$\mu_T = \Sigma \mu_i \text{ and } \sigma_T^2 = \sum_{ij} \sigma_i \, \Gamma_{ij} \sigma_j$$

we obtain:

$$-\ln \langle s_T \rangle = \frac{\left(\Sigma \beta_i \left[-\ln \langle s_i \rangle\right]\right)^2 \left(\ln(b_T + 1)\right)}{\sum_i \sum_j b_i \beta_i \left[-\ln \langle s_i \rangle\right]^{1/2} \Gamma_{ij} \left[-\ln \langle s_j \rangle\right]^{1/2}}$$

We now assign to all weapons the same value of b_i, so that all b_i are equal and all β_i are equal and we obtain:

$$-\ln \langle s_T \rangle = \frac{\ln(b_T + 1)}{\ln(b_i + 1)} \; \frac{\left[\Sigma \left(-\ln \langle s_i \rangle\right)\right]^2}{\Sigma \, \Sigma \left(-\ln \langle s_i \rangle\right)^{1/2} \Gamma_{ij} \left(-\ln \langle s_j \rangle\right)^{1/2}}$$

If we now let b_i approach infinity the ratio of the two logarithmic quantities will approach 1. Note that

$$b_T = \frac{\sigma_T^2}{\mu_T}, \qquad \text{so} \quad b_T = \frac{\Sigma \Sigma \sigma_i \Gamma_{ij} \sigma_j}{\Sigma \mu_i}$$

It follows that $b_T \geq b_i$ and $b_T \leq n^2 b_i$, where n is the number of weapons. The limiting case $b_T = n^2 b_i$ occurs when all $\Gamma_{ij} = 1$ and all μ_i are equal. Therefore so long as $b_i >> n^2$ the ratio of the logarithms will be essentially 1, and in the limit as b_i approaches infinity we obtain simply:

$$-\ell n \langle S_T \rangle = \frac{\left[\Sigma -\ell n \langle S_i \rangle\right]^2}{\Sigma\,\Sigma \left(-\ell n \langle S_i \rangle\right)^{1/2} \Gamma_{ij} \left(-\ell n \langle S_j \rangle\right)^{1/2}}$$

For compactness of notation let us identify the quantities

$$\mu_i = (-\ell n \langle S_i \rangle) \quad \text{and} \quad \mu_T = (-\ell n \langle S_T \rangle)$$

Then since $\Gamma_{ij} = 1$ if $i = j$ we obtain

$$\mu_T = \frac{\left[\sum_i \mu_i\right]^2}{\sum_i \mu_i + \sum_i \sum_{j \neq i} \left(\mu_i\right)^{1/2} \Gamma_{ij} \left(\mu_j\right)^{1/2}}$$

or equivalently

$$\mu_T = \frac{\left[\sum_i \mu_i\right]^2}{\sum_i \mu_i + \sum_i \sum_{j<i} \left(\mu_i\right)^{1/2} 2\,\Gamma_{ij} \left(\mu_j\right)^{1/2}}$$

This form has the basic properties desired. Notice there is only one interaction term between each pair of weapons. In addition, only two sums need to be maintained to compute μ_T. These are:

$$\text{MU} = \sum_i \mu_i$$

$$\text{SIG} = \sum_i \sum_{j<i} \left(\mu_i\right)^{1/2} 2\Gamma_{ij} \left(\mu_j\right)^{1/2}$$

From these the value μ_T is given simply:

$$\mu_T = (\text{MU})^2 \,/\, (\text{MU} + \text{SIG})$$

The addition of any new weapon adds one term to the MU sum, and several terms to the SIG sum.

The computation of the first sum is trivial; however, before the second one can be used it is necessary to provide a practical method of estimating Γ_{ij}.

We recall that the array RISK (A,G,J) was computed as an estimate of shared risk, and that:

$$RISK(A,G,J) = \sum_{L=1,5} SM(L)*SMAT(A,L)$$

For a particular weapon G and hardness component J, this relation might look as follows: (A is a weapon attribute index; L is a failure mode index.)

		SMAT(A,L)						
	A =	1	2	3	4	5	6	
L	SM(L)	All	Group	Reg	Class	Type	Alert	Independent Risk
1	-LOGF(DBL) = .20	.00	.10	.10	.10	.10	.40	.20
2	-LOGF(CC) = .00	.00	.10	.30	.10	.10	.30	.10
3	-LOGF(REL) = .05	.00	.05	.00	.10	.20	.00	.65
4	-LOGF(PEX) = .20	.00	.00	.10	.20	.20	.00	.50
5	-LOGF(STK) - .02	.00	.00	.00	.00	.00	.00	1.00
	RISK(A,G,J)	.000	.0225	.040	.065	.070	.08	.1925

Thus the SMAT array, a user input estimate of shared risk, is used simply to divide the five types of risk SM(L) between the independent weapon risk, and the six factors A that any two weapons might have in common. The total RISK over all A plus the independent risk is of course equal to the sum of SM(L). We are now interested in using the RISK array to derive reasonable values for the correlation coefficients Γ_{ij}.

The RISK array thus represents the amount of the risk for each weapon that is likely to be correlated with other weapons of the same class, type, etc.

The correlation coefficients should reflect the <u>shared</u> risk. If two weapons have only two attributes A in common then the shared risk should come only from these two common attributes. Moreover, the amount of risk that can be shared on the basis of one attribute cannot exceed the minimum risk associated with that attribute for either weapon. Therefore, to estimate the maximum risk, γ_{ij}, that can be shared by two weapons, i and j, we define:

$$\gamma_{ij} \text{ or } GAM(i,j) = \sum_{A} \delta\,(A_i,A_j)*\text{Min } RISK(A_i,G_i,J)RISK(A_j,G_j,J)$$

where $\delta = 0$ if $A_i \neq A_j$ and $\delta = 1$ if $A_i = A_j$.

The coefficients Γ_{ij} however must never exceed 1.0. Therefore it is appropriate to divide the shared risk GAM(i,j) by $\sum_{L} SM$ to obtain a normalized fraction guaranteed to be less than 1.0.

Thus the form of the second summation

$$SIG = \sum_{i} \sum_{j<i} 2(\mu_i)^{1/2}\, \Gamma_{ij}\, (\mu_j)^{1/2}$$

would become

$$SIG = \sum_i \sum_{j<i} 2(\mu_i)^{1/2} \frac{GAM(i,j)}{\sum_L SM} (\mu_j)^{1/2}$$

However, this form involves square roots which are inconvenient. Moreover, it represents an upper limit of correlation. We can reduce the size of the overestimate by using the largest (or maximum) $\sum_L SM$; i.e., using the least reliable weapon for normalization. In addition, we can simplify the form and provide for the removal of square roots if we also multiply by $(\mu_{min} / \mu_{max})^{1/2}$. (This is a factor less than 1.0 that has the effect of reducing slightly the assumed correlation between weapons of very different overall effectiveness.)

With these changes, the equation for SIG takes the form of

$$SIG = \sum_i \sum_{j<i} 2(\mu_i)^{1/2} \left\{ \frac{GAM(i,j)}{Max \sum_L SM} * \left(\frac{\mu_{min}}{\mu_{max}}\right)^{1/2} \right\} (\mu_j)^{1/2}$$

The form in braces is still guaranteed to fall between zero and 1.0. It represents the actual form for Γ_{ij} used in the present version of the Allocator. This form has a computational advantage in that it simplifies the calculation of SIG. Assume that $\mu_i < \mu_j$. Then

$$\sum_L SM_i > \sum_L SM_j \quad \text{and so}$$

$$SIG = \sum_i \sum_{j<i} 2(\mu_i)^{1/2} \left\{ \frac{GAM(i,j)}{\sum_L SM_i} * \left(\frac{\mu_i}{\mu_j}\right)^{1/2} \right\} (\mu_j)^{1/2}$$

This reduces to:

$$SIG = \sum_i \sum_{j<i} 2*GAM(i,j)*\underset{i,j}{MIN}\left\{\frac{\mu_i}{\sum_L SM_i}\right\}$$

This is the actual form used computationally. (For each weapon group G the quantity $\mu / \sum_L SM$ is identified in the FORTRAN as SSIG(G,J).)

The specific formula used for the terms in SIG is of heuristic origin and is obviously somewhat arbitrary. It is justified, in the final analysis, by the fact it is fairly simple and that it works. The resulting kill probabilities produce realistic cross targeting, and in cases where these probabilities can be compared with a rigorous statistical model of correlations, it produces a satisfactory approximation to the kill probability.

In summary, the mathematics is as follows:*

For a single weapon let

SSK = single shot kill probability, and let

SSS = single shot target survival probability

then SSK is given by

$$-LOGF(SSK) = \sum_L SM(L)$$

---

*The displayed mathematics for the calculation of MUP are for the exponential damage. law. The derivation of the quantity, MUP, required for use of the square root damage law is discussed in the Derivation of Square Root Damage Function section of this chapter and are not of any importance in this discussion of correlation effects.

As usual, SSS = 1.0 - SSK, and we define μ_i or MUP for group G_i relative to hardness component J as:

$$\underline{\text{MUP(G,J) = -LOGF(SSS)}}$$

We also define SSIG(G,J) as:

$$\underline{\text{SSIG(G,J) = LOGF(SSS)/LOGF(SSK) = MUP(G,J)/} \sum_L \text{SM(L)}}$$

Finally we define RISK(A,G,J) as:

$$\underline{\text{RISK(A,G,J)} = \sum_{L=1,5} \text{SM(L)*SMAT(A,L)}}$$

The preceding three arrays (underlined for emphasis) are the main input for the estimation of kill probabilities.

The target survivability relative to multiple weapons S_T is given by

$$S_T = e^{-\mu T}$$

where $\mu T = (\text{MU})^2 / (\text{MU} + \text{SIG})$

and where $\text{MU} = \sum_i \mu_i = \sum_i \text{MUP}(G_1, J)$

and $\text{SIG} = \sum_i \sum_{j<i} 2(\mu_i)^{1/2} \Gamma_{ij} (\mu_j)^{1/2}$

The individual terms in SIG for specific i and j can be thought of as:

$$\text{DSIG}(i,j) = 2(\mu_i)^{1/2} \Gamma_{i,j} (\mu_j)^{1/2}$$

which we identify computationally as

$$DSIG(i,j) = 2*GAM(i,j)*\min_{k=i,j} \{SSIG(G_k,J)\}$$

where GAM(i,j), the maximum risk shared by i and j, is estimated as

$$GAM(i,j) = \sum_A \delta(A_i,A_j)*\mathrm{Min}\{RISK(A_i,G_i,J),\ RISK(A_i,G_j,J)\}$$

where δ, the Kroniker δ, is 0 if $A_i \neq A_j$, and 1 if $A_i = A_j$.

The simple form used for DSIG above implies that Γ_{ij} has the form:

$$\Gamma_{i,j} = \frac{GAM(i,j)}{\max_{i,j}\left[\sum_L SM(L)\right]} * \left(\frac{\mu\,\mathrm{Min}}{\mu\,\mathrm{Max}}\right)^{1/2}$$

however, this form never enters explicitly into the calculations.

To combine this treatment for the analysis of weapon correlations with the preceding treatment of time-dependent target values we simply use the S_T evaluated above to supply the S(NI,J) required in the formula

$$VT = \sum_{J=1}^{J=M} \sum_{NI=0}^{NI=NN} \left[V(NI,J) - V(NI+1,J)\right] * S(NI,J)$$

The weapons to be included in the evaluation S_T for any NI are of course those on the target up to and including the time NI.

This, of course, requires that separate sums for MU and SIG be maintained for each relevant time interval, NI, and each hardness component J. Thus these variables are actually two-dimensional arrays MU(NI,J) and

SIG(NI,J). Moreover, every potential payoff estimate (both for each weapon that might be added, and for each that might be deleted) requires a separate complete set of sums.

Derivation of Damage Functions

A Universal Damage Function: Consider the situation for which the lethal radius and CEP of a single weapon are small compared to the target dimensions. This case becomes quite pertinent under any of the following circumstances:

Very large cities

Targets whose uncertainty of location is larger than the area of influence of a weapon

Employment of large numbers of small weapons (e.g., cluster warheads)

Hardening which reduces effective weapon radius below target size (e.g., blast shelters for urban population).

In such a situation, where the value density of the target does not vary significantly over the area of effect of a single weapon, one can usefully employ the concept of weapon density (weapons targeted per unit area) and seek the weapon density as a function of value density which optimizes the total target destruction for a given total number of weapons.

Before such an optimization can be effected, however, it is necessary to obtain the relationship between the weapon density applied to a sub-region, expressed for convenience as the fraction of the original value surviving. In the most general case, this function can vary with position in the target, reflecting the possibility of varying degrees of vulnerability over the target.

We introduce the following notation:

X	Position within target (x, y coordinates)
$\omega(X)$	Density of weapons targeted in vicinity of X (number/unit area)
$V(X)$	Target value density in vicinity of X (value/unit area)
$F(\omega)$	Fraction of destruction produced by weapon density ω, in the absence of hardening
$\mu(X)$	Vulnerability (hardening) factor ($0 \leq \mu \leq 1$) expressed as effective degradation of weapon density
W	Total number of weapons intended against target.

The total payoff for a given weapon density distribution is then given by:

$$H = \int_A VF(\mu\omega)\,dA \qquad (1)$$

where the integration is understood to be over the whole target area, and dA is the area element.

Similarly, the total number of planned weapons is given by:

$$W = \int_A \omega\,dA \qquad (2)$$

We seek now the weapon density distribution which maximizes the payoff for a given W. Introducing a Lagrange multiplier $\lambda \geq 0$, and applying the generalized method described above,* we seek the weapon density function which maximizes the unconstrained Lagrangian.

$$L = H - \lambda W \tag{3}$$

This is equivalent to maximizing:

$$L = \int_A \left[VF(\omega\mu) - \lambda\omega \right] dA \tag{4}$$

The density function ω_λ^* which maximizes this Lagrangian for a given λ is obtained simply by maximizing the expression inside the integral at each point (see cell problem discussion in Everett's paper, appendix B). The optimum density at any point is therefore a solution of:

$$\underset{\omega}{\text{MAX}} = \left\{ VF(\mu\omega) - \lambda\omega \right\} \tag{5}$$

For the case where F is monotone increasing, concave (diminishing returns), and differentiable, an internal maximum of (5) can be sought by zeroing its derivative:

$$\frac{d}{d\omega} \left[VF(\mu\omega) - \lambda\omega \right] = VF'(\mu\omega_\lambda^*)\mu - \lambda = 0 \tag{6}$$

Letting $G = (F')^{-1}$ stand for the inverse function of the derivative of F leads to:

*See the Weapon Allocation section in chapter 2.

$$\omega_\lambda^* = \frac{1}{u} G\left(\frac{\lambda}{V\mu}\right) \qquad (7)$$

Equation (7) gives the internal maximization of (5). To complete the solution we must account for the constraint $\omega^* \geq 0$ (negative densities are not allowed). Thus the optimum is given by (5) only if $\omega_\lambda^* \geq 0$ and if $VF(\mu\omega^*) - \lambda\omega^* \geq 0$, since otherwise (5) is maximized by $\omega = 0$. The complete solution can therefore be stated:

$$\omega_\lambda^* = \begin{cases} \frac{1}{\mu} G\left(\frac{\lambda}{V\mu}\right) & \text{if } \omega_\lambda^* \geq 0 \text{ and } VF(\mu\omega^*) - \lambda\omega \geq 0 \\ 0 & \text{otherwise} \end{cases} \qquad (8)$$

(This solution is also valid even if F is not concave -- a situation in which G may be multivalued -- provided that one uses that value of $G(\lambda/V\mu)$ for which $VF(\mu\omega) - \lambda\omega$ is a maximum.)

Observe that the optimum density given by (8) is a function only of V and μ, and is explicitly independent of position. If we can further assume that the vulnerability μ is a function only of the value density V and is otherwise independent of position,* then we can simplify the formulation and solution somewhat. In this case, all pertinent target characteristics are summarized by two functions:

---

*Which seems generally quite plausible, and is in any case certainly true if the variation of μ arises from optimization of shelter deployment, for example.

A(V) = total area of those areas whose value density is greater than V

μ(V) = vulnerability factor as a function of value density

The optimum weapon density ω_λ^* given by (8) becomes then a function only of the value density V:

$$\omega_\lambda^*(V) = \begin{cases} \frac{1}{\mu(V)} G\left(\frac{\lambda}{V\mu(V)}\right) & \text{if } \omega_\lambda^* \geq 0 \text{ and } VF(\mu\omega^*) - \lambda\omega^* \geq 0 \\ 0 & \text{otherwise} \end{cases} \quad (9)$$

and the total payoff and total weapons are given in the simple form of Stieltjes integrals:

$$H_\lambda = -\int_0^\infty VF(\omega_\lambda^*\mu(V))\,dA(V)$$

$$W_\lambda = -\int_0^\infty \omega_\lambda^*\,dA(V) \quad (10)$$

This completes the general optimization of weapon density. For explicit solutions we require specific functions for the target value distribution function A(V), the destruction function F(ω), and the vulnerability distribution μ(V). We shall now consider several pertinent cases.

Locally Random Impact Model: When the CEP is not significantly smaller than the lethal radius, or when the delivery probability of individual weapons is low, the situation over any homogeneous part of the target can be closely approximated by regarding the weapons as having been dropped uniformly at random over that part.

Consider, therefore, a region of area A (large compared to the lethal area of a single weapon) into which N weapons each with lethal area πR_K^2 and delivery probability P are delivered uniformly and independently at random. The probability that any given point in the region will survive one weapon is:

$$S(1) = 1 - \frac{P\pi R_K^2}{A} \tag{11}$$

and, since weapon arrivals are independent events, the probability of surviving N is:

$$S(N) = \left(1 - \frac{P\pi R_K^2}{A}\right)^N \tag{12}$$

Introducing the parameters K and ω :

$$K = P\pi R_K^2 = \text{expected lethal area of one weapon}$$

$$\omega = N/A \quad = \text{weapon density} \tag{13}$$

allows (12) to be written as:

$$S(\omega) = \left(1 - \frac{K\omega}{N}\right)^N \tag{14}$$

This gives for the destruction function:

$$F_N(\omega) = 1 - S(\omega) = 1 - \left(1 - \frac{K\omega}{N}\right)^N \tag{15}$$

Equation (15) still contains an extra parameter, N, which is the number of weapons in the area A used to derive (12)--presumed large compared to the effects of a single weapon and small compared to the total target size. We are currently interested in the limit as this area A becomes infinite compared to the effects of a single weapon, hence in the limit as $N \to \infty$:

$$F_\infty(\omega) = \lim_{N \to \infty} F_N(\omega) = 1 - e^{-K\omega} \tag{16}$$

which becomes our final destruction function for the locally random impact model.

"Perfect" Weapon Model: At the other extreme from the locally random impact model is the hypothetical situation where the weapons have zero CEP, delivery probability of unity, and completely destroy a hexagonal region of area K with no damage outside the region.

This situation closely resembles the case of "cookie-cutter" weapons of zero CEP and unit delivery probability, and deviates from the latter only when the area covered is so densely packed that the "cookie-cutter" circles begin to overlap--which does not occur until the fractional coverage exceeds $\pi/(2\sqrt{3})$ or about .91.

For such "perfect" weapons the destruction fraction is given by:

$$F = \begin{cases} K\omega & \omega < 1/K \\ 1 & \omega \geq 1/K \end{cases} \qquad (17)$$

Intermediate Cases: We have considered two extremes, locally random impact, and perfect weapons. For actual situations, the targeting will not be random, but some optimum pattern of DGZs.

As the CEP becomes larger than the lethal radius, or the delivery probability becomes small, the situation -- even though based on a pattern of DGZs -- approaches a situation described by the random impact moded. On the other hand, for high delivery probability and small CEP, the situation begins to approach the "perfect" weapon case -- particularly as the weapon effect radius becomes sharp (close to "cookie-cutter" -- e.g., the conventional σ_{20} model).

Returning to the destruction function given by (15) containing the extra parameter N (from which the random model was obtained by letting $N \rightarrow \infty$), we observe the remarkable fact that for N = 1 this function is precisely the damage function (17).

Since this function contains, for the extreme values of N, the two limits we have considered, it seems reasonable to suppose that any actual intermediate case could be adequately approximated by this function for some intermediate value of N.

We shall accordingly adopt this general function as our destruction function, subject to subsequent empirical verification.

The general law therefore becomes:

$$F_N(\omega) = \begin{cases} 1 - \left(1 - \frac{K\omega}{N}\right)^N & \omega < \frac{N}{K} \\ 1 & \omega \geq \frac{N}{K} \end{cases} \tag{18}$$

For purposes of determining the optimum distribution of weapon density over a target of varying value density we wish to employ Eq. (9), for which we require the function $G = (F')^{-1}$. Accordingly,

$$F'_N(\omega) = \frac{d}{d\omega} F_N(\omega) = \begin{cases} K\left(1 - \frac{K\omega}{N}\right)^{N-1} & \omega < \frac{N}{K} \\ 0 & \omega \geq \frac{N}{K} \end{cases} \tag{19}$$

for which the inverse function is easily determined to be:

$$G_N(X) = \frac{N}{K}\left[1 - \left(\frac{X}{K}\right)^{1/(N-1)}\right] \tag{20}$$

Thus from (9), the optimum weapon density is given by:

$$\omega_\lambda^*(V) = \begin{cases} \frac{1}{\mu(V)}\frac{N}{K}\left[1 - \left(\frac{\lambda}{KV\mu(V)}\right)^{\frac{1}{N-1}}\right] & \frac{\lambda}{KV\mu} < 1 \\ 0 & \frac{\lambda}{KV\mu} \geq 1 \end{cases} \tag{21}$$

and for which the destruction fraction is easily calculated:

$$F_N(\omega_\lambda^*\mu) = \begin{cases} 1 - \left(\frac{\lambda}{KV\mu(V)}\right)^{N/N-1} & \frac{\lambda}{KV\mu} < 1 \\ 1 & \frac{\lambda}{KV\mu} \geq 1 \end{cases} \tag{22}$$

This completes the general treatment for arbitrary target value distributions.

Gaussian Targets: A particularly important special case is that of a Gaussian target, for which the value density distribution is given by:

$$V(x,y) = \frac{1}{2\pi\sigma^2} e^{-r^2/2\sigma^2} \tag{23}$$

(The total value is here normalized to unity.) From (23) we determine the relationship between radius and value to be :

$$r^2(V) = -2\sigma^2 \ \ell n(2\pi\sigma^2 V) \tag{24}$$

and hence the cumulative area distribution function to be:

$$A(V) = \pi r^2(V) = -2\pi\sigma^2 \ln(2\pi\sigma^2 V) \quad \text{for } V \leq \frac{1}{2\pi\sigma^2} \tag{25}$$

and the differential element is:

$$dA(V) = -\frac{2\pi\sigma^2}{V} dV \tag{26}$$

Solution nstant Vulnerability: Combining Eq. (10) with (26) and (22), and letting $\mu = 1$:

$$H_\lambda = \int_{\lambda/K}^{1/(2\pi\sigma^2)} V \left[1 - \left(\frac{\lambda}{KV}\right)^{\frac{N}{N-1}}\right]\left(\frac{2\pi\sigma^2}{V}\right) dV$$

$$= 1 - \frac{2\pi\sigma^2\lambda}{K} - (N-1)\left[\left(\frac{2\pi\sigma^2\lambda}{K}\right)^{\frac{N}{N-1}} - \frac{2\pi\sigma^2\lambda}{K}\right] \tag{27}$$

Transforming the Lagrange multiplier λ to a new multiplier β:

$$\beta = \left[\frac{2\pi\sigma^2\lambda}{K}\right]^{1/(N-1)} \tag{28}$$

we can rewrite (27) as:

$$H_\beta = 1 - \beta^{N-1}\left[1 + (N-1) \cdot (1-\beta)\right] \tag{29}$$

The total number of weapons as given by (10), (21), and (26):

$$W_\lambda = \int_{\lambda/K}^{1/(2\pi\sigma^2)} \frac{N}{K} \left[1 - \left(\frac{\lambda}{KV}\right)^{\frac{1}{N-1}}\right]\left(\frac{2\pi\sigma^2}{V}\right) dV \tag{30}$$

leads, in terms of β, to:

$$W_\beta = \frac{N(N-1)2\pi\sigma^2}{K} \left[\beta - \ln(\beta - 1)\right] \tag{31}$$

In order to permit explicit exhibition of payoff as a function of number of weapons, it is necessary to define a new function, Υ, which is the inverse of

$$y - \ln y - 1 = x \tag{32}$$

that is, $y = \Upsilon(x)$. It is defined for all non-negative arguments, with values on the interval zero-one. With this function, (29) and (31) can be rewritten, in terms of surviving value:

$$S = \beta^{N-1}\left[1 + (N-1)\ (1 - \beta)\right]$$
$$\beta = \Upsilon\left(\frac{KW}{2\pi\sigma^2 N(N-1)}\right) \tag{33}$$

Equations (33) summarize the relationship between surviving fraction, S, and number of weapons targeted, W, for Gaussian targets, and with a model parameter N, which can range from 1 to ∞.

The two limiting forms of (33), corresponding to $N = 1$ and $N \to \infty$ are interesting and important, and are easily shown to be:

$$S_1 = \exp(-KW/2\pi\sigma^2)$$

$$S_\infty = \left(1 + \frac{\sqrt{KW}}{\pi\sigma^2}\right) \exp\left(-\frac{\sqrt{KW}}{\pi\sigma^2}\right)$$

These are often termed the power law (or exponential law) and the square root law, respectively.

Derivation of Kill Probability Function

A variety of kill probability functions are in general use. The "normal model" employs a function of the form:

$$P_K(r) = e^{-r^2/2\sigma_K^2} \tag{34}$$

The "cookie-cutter" model employs a discontinuous function:

$$P_K(r) = \begin{cases} 1 & R_K \geq r \geq 0 \\ 0 & r > R_K \end{cases} \tag{35}$$

where R_K is the so-called "lethal radius." The relation between R_K and σ_K is obtained by equating lethal areas

$$\pi R_K^2 = \int_0^{2\pi} \int_0^\infty e^{-r^2/2\sigma_K^2} \, r dr d\theta \tag{36}$$

leading to the relation

$$\sigma_K^2 = .5R_K^2 \tag{37}$$

Other functions have often been used and, indeed, it has occasionally been found convenient to employ a generalized kill function of the form:

$$G_K(r) = e^{-K} \sum_{j=0}^{W-1} \frac{K^j}{j!} \tag{38}$$

where

$$K = \frac{Wr^2}{a^2}$$

Again, we can equate lethal areas to relate a with R_K:

$$\pi R_K{}^2 = \int_0^{2\pi} \int_0^{\infty} G_K(r) \; r dr d\theta \tag{39}$$

so that

$$R_K{}^2 = a^2 \text{ for all } W \tag{40}$$

The parameter W serves to alter the shape of this kill probability curve. Thus, $G_K(r)$ reduces to the normal curve for $W = 1$ and the cookie-cutter for $W \to \infty$. Standard kill curves, such as the σ_{20} and σ_{30} curves of AFM 200-8, representing, respectively, ground burst and optimal air burst blast damage probabilities as a function of distance, can readily be approximated. $W = 6$ approximates closely the σ_{20} curve, and $W = 3$ approximates the σ_{30} curve.

Integration of a kill probability function over appropriate density functions allows the representation of such factors as delivery error, geodetic error, extended targets, etc.

Assume an extended target with Gaussian normal value distribution as follows:

$$V(r) = \frac{1}{2\pi\sigma_{Tgt}^2} e^{-r^2/2\sigma_{Tgt}^2} \tag{41}$$

$V(r)$ = value per unit area at distance r from center

σ_{Tgt} = standard deviation of value distribution

Clearly:

$$1.0 = \frac{1}{2\pi\sigma_{Tgt}^2} \int_0^\infty e^{-r^2/2\sigma_{Tgt}^2}\, dr \tag{42}$$

Define a radius, R95, such that 95% of the value of the target is contained within this distance of the target center. (This R95 is the target radius used in the QUICK system.)

Then

$$\int_0^{R95} e^{-r^2/2\sigma_{Tgt}^2}\, dr = .95 \int_0^\infty e^{-r^2/2\sigma_{Tgt}^2}\, dr \tag{43}$$

Solving this equation for σ_{Tgt} in terms of R95, we get:

σ_{Tgt} = 2.448 * R95

Assume a CEP, the radius of a circle with center at an aiming point which will contain 50% of the centers of impact of weapons aimed at the aiming point. Assuming a circular normal (Gaussian) distribution of the aiming errors:

$$p(r) = \frac{r}{\sigma_{CEP}^2} e^{-r^2/2\sigma_{CEP}^2} \tag{44}$$

where

$p(r)$ = probability aiming error is r

σ_{CEP} = standard deviation of aiming errors

By definition of CEP

$$\int_0^{CEP} p(r)\, dr = 0.5 \tag{45}$$

Solving for σ_{CEP} in terms of CEP

$\sigma_{CEP} = .8943 * CEP$

Assume a weapon is aimed at the center of the target. From the nature of the Gaussian distribution we can define a standard deviation $\sigma_D^2 = \sigma_{CEP}^2 + \sigma_{Tgt}^2$ such that the circular normal distribution characterized by σ_D^2 is the convolution of the distributions characterized by σ_{CEP}^2 and σ_{Tgt}^2.

Therefore if

$P_K(W)$ = probability of target kill

W = kill function parameter

$G_K(r)$ = kill function from Eq. (38)

then

$$P_K(W) = \frac{1}{2\pi\sigma_D^2} \int_0^\infty \int_0^{2\pi} \exp\left[-\frac{r^2}{2\sigma_D^2}\right] G_K(r) \; r d\theta \qquad (46)$$

Evaluating the integrals

$$P_K(W) = 1 - \left(\frac{2WX^2}{1 + 2WX^2}\right)^W \qquad (47)$$

where $X = \sigma_D/R_K$

or

$$P_K(W) = 1 - \left(\frac{\sigma_D^2}{\sigma_D^2 + \frac{1}{2W} R_K^2}\right)^W \qquad (47a)$$

which is the function used in QUICK.

Optimization of DGZs for Complex Targets

Program ALOCOUT is responsible for selecting optimum desired ground zeros (DGZs) for weapons allocated to complex targets. The complex target may contain several component target elements, each with specific coordinates, hardness, and some given time dependence of value. To place this diverse target element information on a commensurate basis for efficient DGZ selection, each target component of the complex is represented as a series of simple point value elements. Complex elements with more than one hardness component generate more than one such target element, and area targets generate several elements, spread over the area of the target, to represent a value spread over the area. A (DGZSEL) Desired Ground Zero Selector then uses the data to select optimum aim points within the target complex.

The selection of DGZs is a two-step process. First, the prescribed warheads are assigned initial coordinates through a "lay-down" process in which each successive warhead is targeted directly against that target element where the highest payoff is achieved, taking into account collateral damage to all other target elements. Second, a general-purpose function optimizer, FINDMIN, calculates the derivatives of the payoff as a function of x and y coordinates of each weapon and adjusts the coordinates to minimize the surviving target value. FINDMIN terminates either after a maximum number of iterations (which can be specified by the NMCSSC analyst) or after it finds that it can no longer make significant improvements in the payoff.

The mathematical representation used is as follows.

The weapons allocated to a complex target are to be placed in a manner which attempts to minimize the total escaping target value. To simplify discussion, the notation below is introduced. A second subscript, j, referencing the j^{th} target element, is used when needed.

$\overline{V}_j$ = value of j^{th} target element remaining immediately following arrival of the i^{th} weapon

S_j = probability of survival of j^{th} target element associated with weapon i

E_j = amount of value of j^{th} target element that "escapes" between arrival of weapons i - 1 and i

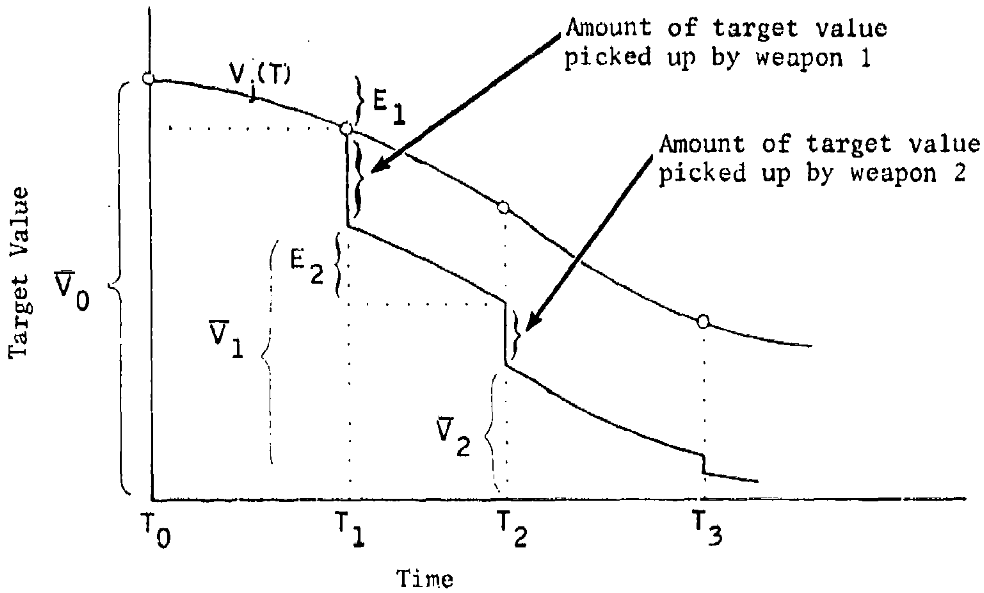
T_i = time of arrival of weapon i (T_0 is an initial time when the full target value is applied) ($T_i \leq T_{i+1}$ all i)

$V_j(T_i)$ = value of j^{th} target, at time T_i

N = number of weapons

NT = number of targets

The following sketch illustrates the treatment of the time-dependent values of the j^{th} target



From this sketch, the following relationships should be apparent. The equations immediately below refer to a single target (j), but for simplicity the j subscript is omitted.

$$\bar{V}_i = V(T_i) S_i \bar{V}_{i-1} / V(T_{i-1}) \qquad (i = 1,2, \ldots, N)$$

$$E_i = \bar{V}_{i-1} \left[1 - V(T_i)/V(T_{i-1}) \right] \quad (i = 1,2, \ldots, N + 1)$$

From the previous equations,

$$\bar{V}_i = \left[\prod_{k=1}^{i} S_k \right] V(T_i) \quad \text{and} \quad E_i = \left[\prod_{k=1}^{i} S_k \right] \left[V(T_{i-1}) - V(T_i) \right]$$

(For $i = 1$, the product $\left(\prod_{k=1}^{i-1} S_k \right)$ is understood $= 1$. Also $V(T_{N+1}) = 0$.)

The total escaping value associated with target j is

$$\sum_{i=1}^{N+1} E_{ij} = \sum_{i=1}^{N+1} \left(\left[\prod_{k=1}^{i-1} S_{kj} \right] \left[V_j(T_{i-1}) - V_j(T_i) \right] \right)$$

The value on target j which escapes after arrival of weapon i is given by

$$\sum_{p=i+1}^{N+1} E_{pj}$$

The effective value of target j associated with weapon i is defined by

$$F_{ij} = \left(\sum_{p=i+1}^{N+1} E_{pj} \right) \Big/ S_{ij}$$

This value is introduced for computational efficiency and may be thought of as the total value available for weapon i, the effect of all other weapons having been taken into account.

The marginal value picked up on target j due to weapon i is given by

$$F_{ij}(1 - S_{ij})$$

where S_{ij} is a function of, among other things, the position of weapon i. For a fixed weapon configuration, weapon i can be moved from (x,y) to (x',y') and the marginal escaped value is given by:

$$\sum_{j=1}^{NT} F_{ij}(S_{ij} - S'_{ij})$$

To establish an initial weapon configuration, a lay-down is performed as follows. Initially, set $S_{ij} = 1$ for all i, j. Denote by S^j_{ik} the survival probability of the k^{th} target, relative to the i^{th} weapon, when this weapon is placed on the j^{th} target. Now the i^{th} weapon is placed on that target, j, which yields a maximum value for the expression

$$\sum_{k=1}^{NT} F_{ik}(S_{ik} - S^j_{ik})$$

The S_{ik} are now set equal to S^j_{ik} (k = 1,2, ..., NT), the F_{ik} (all i, k) are redetermined, i is increased by one, and the process repeated until all weapons have been allocated.

This weapon configuration can now be input as the initial position to a "hill climber" routine, based on a steepest descent algorithm, which attempts to optimize further by replacing the discrete set of possible weapon positions with the two-dimensional continuum. The function to be minimized is:

$$\sum_{j=1}^{NT} \sum_{i=1}^{N+1} E_{ij}$$

Processing by the optimizer will be terminated either when the optimum has been achieved or when a specified number of iterations have been completed. In either case, to insure that the local optimum obtained cannot be further improved, the value of removing, in sequence, each of the weapons from its final location and placing it on one of the target points is explored. If the results obtained by this method are better than those achieved with the previous configuration, this new assignment will be used as an initial one for a second utilization of subroutine FINDMIN. If not, the results of the first use of subroutine FINDMIN will be kept.

Feasibility Testing for MIRV Footprints

This section presents the equations which are used to approximate the physical characteristics of MIRV delivery systems. These equations were derived by a curve fitting program to match the physical data of current MIRV systems.* This section presents the functional form for the equations for three types of systems. The values for the equation parameters must be obtained from the reference.

For all equations, define the following variables:

R = Great circle distance from launch point to first target in footprint (nautical miles)

RM = Maximum booster range (nautical miles)

AZ = Launch aximuth of booster (radians)

MPU = Number of nautical miles traversed per unit of fuel

TF = Total fuel carried for footprinting

DCR = Ratio of equivalent downrange distance to crossrange distance

DUR = Ratio of equivalent downrange distance to uprange distance

---

*See "Strategic Offensive Weapons Employment in the Time Period About 1975 (U)," (Top Secret) Weapons Systems Evaluation Group Report R-160, August 1969, Volume VI, Allocation of MIRV Systems.

Long-Range System: The equations for this system are as shown below.

1. Fuel Load at Booster Separation Available for Footprinting

$$TF = T - SRF$$

T is a constant, representing total fuel

SRF is a parameter which depends only on the number of re-entry vehicles carried on the booster at launch time. It represents fuel required for spacing and release of RVs.

2. Maximum Booster Range

$$RM = RBASIC + RADD * \sin(AZ)$$

RBASIC and RADD are parameters which depend on the number of RVs carried on the booster at launch and the sine of the azimuth

3. Range Extension

This equation refers to the capability to deliver the first RV at a distance greater than maximum booster range. The footprinting fuel* will be used, if necessary, to extend the booster range.

$$MPU = RX + RAXX * \sin(AZ)$$

*Footprinting fuel is the fuel used by the final stage ("bus") of the delivery system to position, space, and release the re-entry vehicles. It may also be used to extend the basic booster range.

RX and RAXX are parameters which depend on the number of RVs originally on board the booster and the sine of the azimuth.

4. Re-entry Vehicle Delivery

This equation refers to the capability to "toss" re-entry vehicles from the current target point to the next target point.

$$MPU = G * (TC_1 + TC_2 * \sin(AZ)$$

where

$$G = \exp\left\{ TE_1 * \left(\frac{RM-R}{TD_1}\right)^{TE_2} \right\}$$

TC_1 and TC_2 depend on the number of RVs originally on board, the number of RVs currently on board, and the sine of the launch azimuth.

TE_1 and TE_2 depend on the number of RVs originally on board and the number of RVs currently on board.

TD_1 is a function of the number of RVs currently on board.

5. Downrange-Crossrange Ratio

$$DCR = G * (C_1 + C_2 * \sin(AZ)$$

where

$$G = \exp \left\{ E_1 * \left(\frac{RM-R}{TD_2} \right)^{E_2} \right\}$$

C_1 and C_2 depend on the number of RVs currently on board and the sine of the azimuth.

E_1 and E_2 depend on the number of RVs currently on board.

TD_2 is a constant.

6. Downrange-Uprange Ratio

$$DUR = \infty$$

The long-range system has no uprange capability.

Short-Range System: The equations for this system are as shown below.

1. Fuel Load at Booster Separation

$$TF = \beta_2 R^2 + \beta_1 R + \beta_0$$

β_2, β_1, and β_0 all depend on the number of RVs originally on board the booster.

2. Maximum Booster Range

$$RM = MAXR$$

MAXR depends upon the number of RVs originally on board the booster.

3. Range Extension*

This equation is of the same form as the equation for re-entry vehicle delivery. The parameters are evaluated using the number of RVs originally on board the booster.

4. Re-entry Vehicle Delivery

$$MPU = \alpha_2 R^2 + \alpha_1 R + \alpha_0$$

α_2, α_1, and α_0 depend on the number of RVs currently on board the booster.

5. Downrange-Crossrange Ratio

$$DCR = \gamma_2 R^2 + \gamma_1 X + \gamma_0$$

γ_2, γ_1, and γ_0 are constants.

6. Downrange-Uprange Ratio

$$DUR = \delta_1 R + \delta_0$$

δ_1 and δ_0 are constants.

*The use of this equation is the same as for the long-range system.

Long-Range System with Area Penetration Aids: The equations for this system are of the same functional form as for the basic long-range system, except for the calculation of fuel available for footprinting. The parameter values for each equation have values different from those used in the basic long-range system.

1. Fuel Load at Booster Separation

$$TF = T - \left\{ G * \left(SRFC_1 + SRFC_2 * \text{sine}(AZ) \right) \right\}$$

where

$$G = \exp \left\{ SRFE_1 * \left(\frac{RM-R}{SRFD} \right)^{SRFE_2} \right\}$$

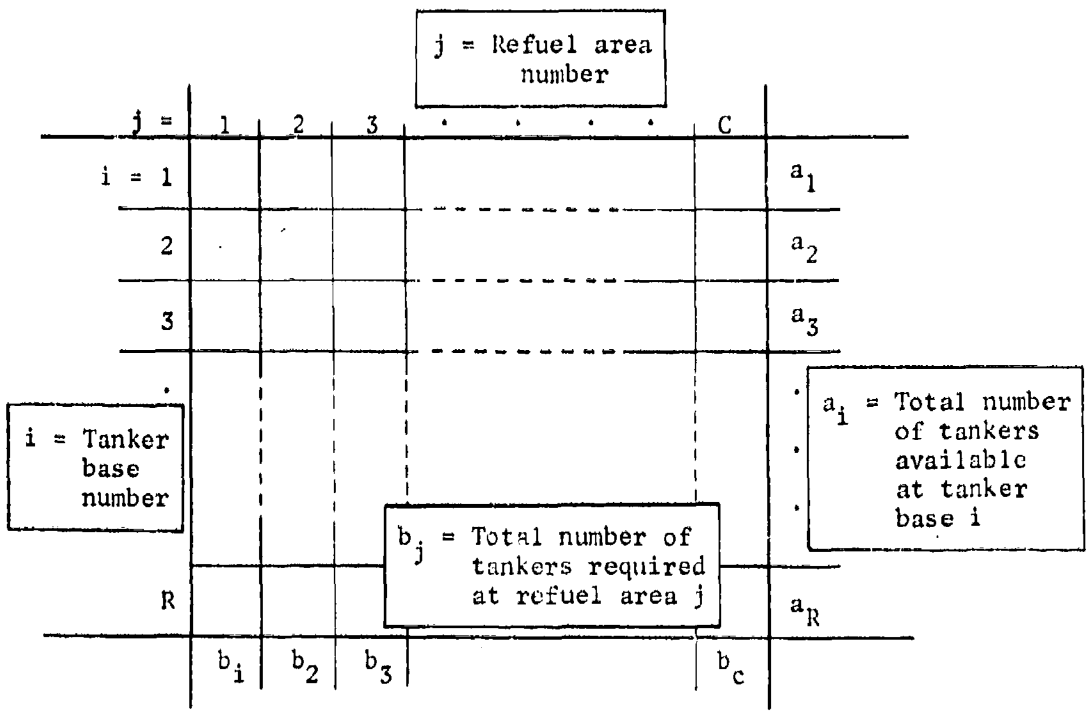
T is the constant total fuel parameter as used in the basic long-range system. The remainder of the equation represents calculation of the fuel required for spacing and release of the RVs and the area penetration aids.

$SRFC_1$, $SRFC_2$, and $SRFE_1$, $SRFE_2$ depend on the number of RVs originally carried on the booster.

SRFD is a constant.

Tanker Allocation Technique

The task of allocating tankers to refuel areas in such a way as to service all bombers is considered by PLNTPLAN to be a form of the classical transportation problem. The variables involved are considered as follows:



Each cell in the above table has two entries associated with it.

1. COST(i,j) = distance from base i to refuel area j + safety factor of .5 miles.

2. X_{ij} = number of tankers at base i to be assigned to refuel area j.

The statement of the transportation problem to be solved is:

Given: all i, j, a_i, b_j, and COST(i,j) ,

Find: all X_{ij} such that the total number of tanker miles flown

$$\left(\sum_{i=1}^{R} \sum_{j=1}^{C} \left[COST(i,j) * X_{ij} \right] \right)$$

is minimized, subject to the constraints that

1. The total number of tankers assigned from base i must equal the total number of tankers available at base i

$$\sum_{j=1}^{C} X_{ij} = a_i \quad \text{for } 1 \leq i \leq R$$

2. The total number of tankers assigned to refuel area j must equal the total number required at refuel area j

$$\sum_{i=1}^{R} X_{ij} = b_j \quad \text{for } 1 \leq j \leq C$$

A dummy refuel area is created to handle extra tankers, which are later reassigned.

The solution is found using Vogel's Approximation Method. This method will be illustrated below by use of an example: additional information may be found in a basic operations research text, such as Introduction to Operations Research by F. S. Hillier and G. J. Lieberman, published by Holden-Day, Inc.

Figure 20 illustrates the formulation of a tanker allocation problem. There are three refuel areas and three tanker bases. We notice, for example, that there are eight tankers at tanker base 3 and 20 tankers are needed at refuel area 2. The distance from tanker base 1 to refuel area 2 is 200 miles, and the distance from tanker base 3 to refuel area 1 is 500 miles.

We wish now to allocate the tankers from the tanker bases to the refuel areas in such a way that all the tankers at the bases are used, all the requirements at the refuel areas are met, and so that the total mileage that all the tankers fly is as small as possible.

Suppose we look at tanker base 1 and try to allocate the 20 there to the refuel areas. There are many possibilities. We could send five tankers to refuel area 1 and 15 to refuel area 2. We could send all 20 to refuel area 2. We could send 10 to refuel area 1 and 10 to refuel area 3. Or we could make many other allocations. Our first impulse would be to send all 20 tankers to refuel area 2 because then each tanker would have to fly only 200 miles for a total of 4,000 miles. If we did this, however, refuel area 2 would be saturated and the tankers from bases 2 and 3 would have to be sent in some order to refuel areas 1 and 3, a distance for each tanker of 500 miles or for all 20 tankers a total distance of 10,000 miles. This allocation, then, of all 40 tankers would give a total mileage of 14,000.

Tanker Base \ Refuel Area	#1 10 Tankers Needed	#2 20 Tankers Needed	#3 10 Tankers Needed
#1 20 Tankers Available	210 Miles	200 Miles	210 Miles
#2 12 Tankers Available	500 Miles	220 Miles	500 Miles
#3 8 Tankers Available	500 Miles	220 Miles	500 Miles

Fig. 20. Formulation of a Tanker Allocation Problem

If, however, we started all over again and sent 10 of the tankers on base 1 to area 1, the other 10 tankers on base 1 to area 3, and all of the tankers on bases 2 and 3 to area 2, the total mileage would be only 8,600, which is a considerable saving.

The problem with the first allocation is that even though the shortest route for sending the tankers on base 1 is to area 2, this forces us to send the tankers from bases 2 and 3 on a much longer route.

To be more specific, the penalty for not sending the tankers from base 1 on the shortest route to a refuel area is much smaller than the penalty for not sending the tankers from bases 2 and 3 on the shortest route. The idea is that if the tankers are not sent on the shortest route to a refuel area, they can probably be sent on the next shortest route. Therefore, if the distance along the shortest route is not significantly different from the distance along the next shortest route, there is no great penalty for sending the tanker on the second shortest route.

We formalize this idea by defining for a transportation matrix (as in figure 20), a row penalty, which is the difference between the second shortest distance in each row. For figure 20 the row penalties are 10 miles, 280 miles, and 280 miles for rows 1, 2 and 3. We see immediately from these numbers that the penalty for not allocating tankers from row 1 to the closest area is very small compared with the penalty for not allocating from rows 2 and 3 to the closest area. We would naturally then allocate from rows 2 and 3 first.

In general we would first allocate from the row with the largest penalty, then from the row with the second largest, and so on. Although the actual algorithm is much more complicated, using column as well as row penalties and using elimination of rows and columns with subsequent recomputation of penalties, the above example gives the basic idea.

The Vogel Approximation Method has been tested against full-blown transportation algorithms and has been found quite accurate for small matrices.

Missile Timing

The algorithm for determining the intersection of the timing line and the flight path for missiles with a LINE CORMSL uses the nature of the vector cross product to determine possible crossings.

Each great circle segment is the shorter great circle path between two points on the surface of the earth. By the nature of great circles, this path lies completely in the plane defined by the two end points and the center of the earth. Given two such segments, the algorithm will calculate the point of intersection of the segments if they do cross.

In order to do this, we must define a three-dimensional Cartesian coordinate system and define a position vector.

We assume a right-handed coordinate system as shown in figure 21. The origin of the system is the center of the earth. The earth is assumed to have unit radius.

Define a position vector $r_i = (x_i, y_i, z_i)$ to be the vector originating at the origin and terminating at some point on the earth's surface. Since this vector has unit length, we derive the following relationships between the end point's latitude and longitude and the Cartesian coordinates.

Define: β = latitude of end point (+ for North, - for South)

α = longitude of end point, if East (360 - longitude, if West)

Then:

$$\vec{r_i} = \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} = \begin{pmatrix} \cos\alpha\cos\beta \\ \sin\alpha\cos\beta \\ \sin\beta \end{pmatrix}$$

Therefore, each great circle can be defined by two position vectors.

Define: $\vec{R}_{ij} = \vec{r_i} \times \vec{r_j}$

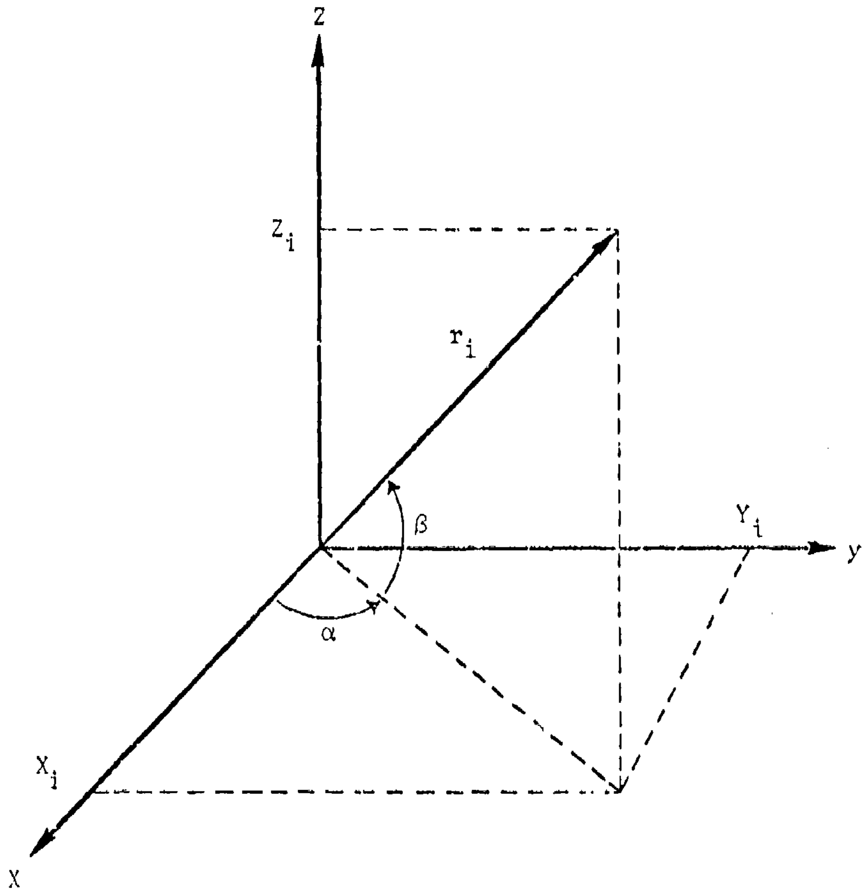


Fig. 21. Coordinate System for Missile Timing Calculations

R_{ij} is the cross product of two position vectors. This vector, R_{ij}, is perpendicular to the plane defined by the great circle. (See figure 22) Any vector in that plane will be perpendicular to R_{ij} and any vector with base at the origin and perpendicular to R_{ij} will lie in the plane.

Define: $\vec{r_1}$ = position vector for first point on timing line

$\vec{r_2}$ = position vector for second point on timing line

$\vec{r_3}$ = position vector for launch point

$\vec{r_4}$ = position vector for target.

$\vec{R_{12}}$ is perpendicular to the plane of timing line

$\vec{R_{34}}$ is perpendicular to plane of flight path.

Let:

$$\vec{T} = \vec{R_{12}} \times \vec{R_{34}}$$

$$= (\vec{r_1} \times \vec{r_2}) \times (\vec{r_3} \times \vec{r_4})$$

If we normalize $\vec{T}$ to have unit length, then $\vec{T}$ and $-\vec{T}$ are position vectors. In fact, they are the position vectors for the points of intersection of the planes of the timing line and the flight path.

Since $\vec{T}$ is perpendicular to $\vec{R_{12}}$, it lies in the first plane. Since it is perpendicular to $\vec{R_{34}}$, it lies in the second plane. Therefore, its end point must lie on both great circles. (See figure 23) The end point does not necessarily lie on the segments defining the timing line or the flight path.

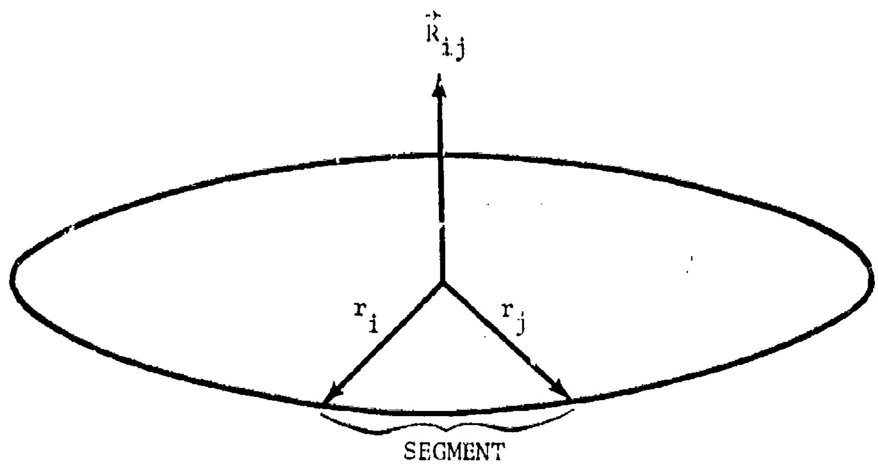


Fig. 22. Relation of R_{ij} to Great Circle Plane

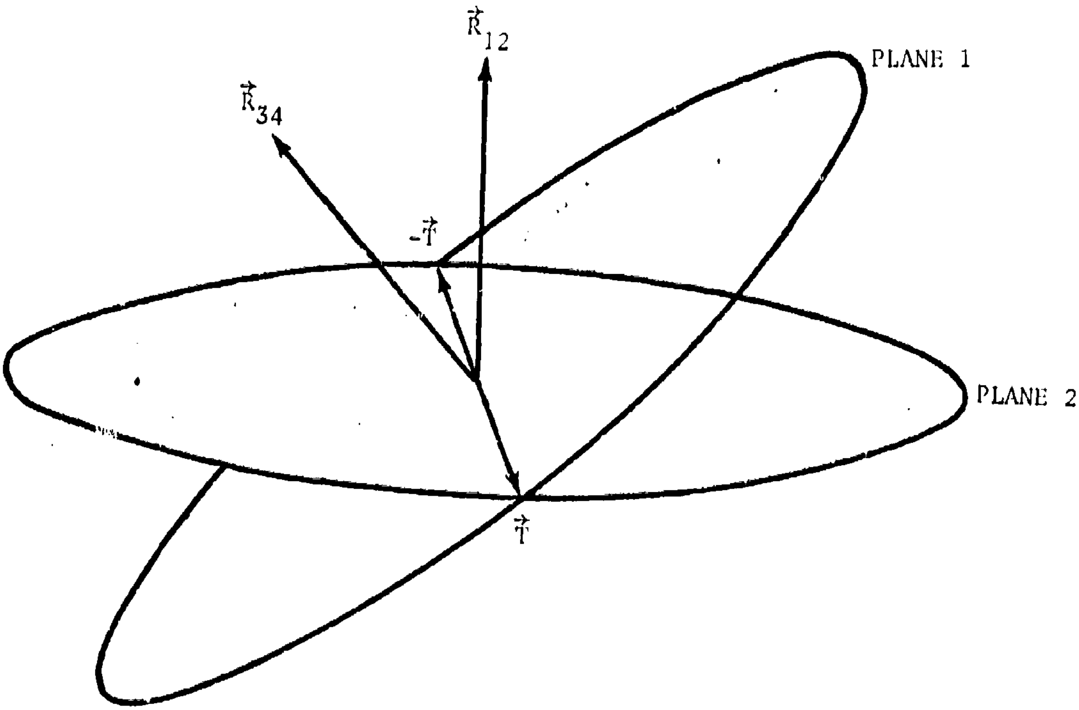


Fig. 23. Diagram of T Vector

With the coordinates of the $\vec{T}$ and $\overrightarrow{-T}$ vectors we can compute the latitude and longitude of the possible intersections.

The line data input is restricted so that the line crosses the flight path from left to right as the missile would see it. This restriction eliminates $\overrightarrow{-T}$ as a possible intersection.

If

$$\vec{T} = \begin{pmatrix} X_T \\ Y_T \\ Z_T \end{pmatrix}$$

then

$$\beta = \sin^{-1}(Z_T)$$

$$\alpha = \tan^{-1}(Y_T/X_T)$$

where the value of the arc tangent is not necessarily the principal value.

We now test these possible intersections to see if they lie on the segment as well as in the plane

Define: $D(\vec{r}_i,\vec{r}_j)$ = shorter great circle distance between end points of $\vec{r}_i$ and $\vec{r}_j$

The possible intersection defined by $\vec{T}$ lies on both segments if

$$D(\vec{r}_1,\vec{R}_2) = D(\vec{r}_1, \vec{T}) + D(\vec{r}_2, \vec{T})$$

and

$$D(\overrightarrow{r_3},\overrightarrow{r_4}) = D(\overrightarrow{r_3}, \overrightarrow{T}) = D(\overrightarrow{r_4}, \overrightarrow{T})$$

If both these relations are true, then the point defined by $\overrightarrow{T}$ is the intersection of the segments and that point is a crossing of the flight path and the timing line.

PLNTPLAN finds the time of the first crossing and uses that time to calculate the launch time so that the missile crosses the line at time equal to CORMSL. If the missile does not cross any line, it will be launched to impact at game time equal to zero.

CHAPTER 4
ACCURACY

The nature of the programs used in the Plan Generation subsystem and the capabilities of the CDC 3800 computer are such that the accuracy of computations performed has negligible impact on the quality of the offensive attack plan generated. The quality of the generated allocation is limited by the following major factors:

1. The accuracy of information contained in the data base supplied by the user to the Data Input subsystem
2. The accuracy of additional user-input information supplied to the programs of the Plan Generator; e.g., weapon delivery correlation
3. The validity of assumptions made, regarding the independence of the targets to be attacked, in the allocation program ALOC.

A more specific discussion of the major factors related to the quality of plans generated by the Plan Generator is presented below.

Correlations

The development of the equations used to model interweapon correlations involves several approximations required for operating efficiency. The effect of these approximations is to slightly increase the estimates of shared risk between weapons. A more precise estimate of the level of increase is not possible, since there does not exist a generally accepted measure of "correlation" between weapons. (Such measures as linear correlation coefficients are inadequate for the multivariate distributions which characterize interweapon correlations for various failure modes.)

The effect on target damage of the weapon cross targeting (produced by consideration of correlations) can be measured. For a data base consisting of 6,000 weapons and 1,500 targets, the maximum decrease in payoff caused by including correlations is less than 2% of the payoff produced by the uncorrelated plan. Various levels of correlations have been used in plan generation, and the resulting decrease in payoff is always minimal. The plans which consider correlations, however, do perform better than uncorrelated plans when evaluated under conditions different from those used in plan generation.*

Optimal Allocations

The generalized Lagrange multiplier method used to determine the weapon allocations produces a theoretically optimal allocation, considering the constraints placed upon the allocation. Constraints such as minimum and maximum destruction levels, country restrictions, and fixed assignments

*A more detailed analysis of the effects of weapon cross targeting on target destruction is contained in "Analysis of Cross Targeting" (UNCLASSIFIED), Lambda Paper 34, Lambda Corporation, 1501 Wilson Boulevard, Arlington, Virginia 22209, September 1969. This study was carried out in response to a request from the Air Force, Office of the Assistant Chief of Staff, Studies and Analysis, under contract F44-620-69-C-0046.

will decrease the payoff from the theoretical optimum. The extent of this decrease is greatly dependent on the range of these constraints.

The allocation produced, however, is nearly optimal considering the constraints. The major factors preventing optimality are the correlation considerations and the closing forces (premiums) used to improve running efficiency. The allocator will produce on user request a rigorous upper bound on the optimal payoff. The allocator can reevaluate the allocation without the effects of either correlations or closing forces. Dr. Everett (see appendix B) proves that the observed loss of profit in this reevaluation is a strict upper bound to the loss in payoff caused by the closing factor. Thus, every run of the allocator can include a verification pass to determine the accuracy of the final weapon allocation.

MIRV Footprint Feasibility

The equations used to determine the feasibility of MIRV footprints were derived as the best fitting curves to actual MIRV system data. To date, there has not been a significant amount of research to determine the accuracy of these equations in operational use. Preliminary research performed in determining the feasibility of using these equations and their associated parameters showed that approximately 98% of the footprints considered feasible using these equations were feasible when tested using physical equations. (The equations become less accurate as the range extension capability is used extensively.) Further research and testing of actual physical MIRV systems and the equation parameters used in these equations should refine the accuracy of the MIRV footprint testing equations.

Planning Factors

The planning factors used for complex targets are, of necessity, approximations. The factors which characterize each component are evaluated to generate the factors characterizing each complex. Such factors as minimum and maximum damage required are averaged, weighted by value, to obtain these factors for the complex. Vulnerabilities are divided into two classes (above and below about 15 psi) and the value divided into hardness components appropriately. The time dependence of target value is a value-averaged approximation which returns most of the characteristics of the original components time dependence.

The major approximations in planning factors which affect plan generation are:

1. A constant speed (specified for each weapon type) is assumed for each weapon regardless of the mode of flight (e.g., high altitude, low altitude, penetration, etc.).
2. Refueling is considered to take no time and to add no distance to the mission. Each bomber is completely refueled by one tanker. Each tanker can refuel only one bomber.

APPENDIX A
QUICK ATTRIBUTE NAMES AND DESCRIPTIONS

ATTRIBUTE NAME	DESCRIPTION
ABRATE	Probability of aircraft in-flight abort per hour of flying time
ADBLI	ALERTDBL probability for initiative attack
ADBLR	ALERTDBL probability for a retaliatory attack
ADEFCMP	Area ballistic missile defense (BMD) component index (radar or missile launch site)
ADEFZON	Area ballistic missile defense (BMD) zone number
AGX	Offset X-coordinate of AGZ (fiftieths of nautical miles)
AGY	Offset Y-coordinate of AGZ (fiftieths of nautical miles)
AHOB	Actual height of burst of weapon (air or ground)
ALERTDBL	Probability of destruction before launch (DBL) of alert delivery vehicle (missile or bomber)
ALERTDLY	Delay of alert vehicle before commencing launch (hours)
AREA	Area of a bomber defense ZONE (millions of nautical miles^2)
ASMTYPE	Air-to-surface missile type
ATTRCORR	Attrition parameter for a bomber corridor (probability of attrition per nautical mile)
ATTRLEG	Attrition parameter for each route leg in bomber sortie (probability of attrition per nautical mile)
ATTRSUPF	Amount of original attrition that remains after defense suppression

ATTRIBUTE NAME	DESCRIPTION
AZON1	First area defense zone covered by a BMD long-range radar
AZON2	Second area defense zone covered by a BMD long-range radar
AZON3	Third area defense zone covered by a BMD long-range radar
BCODE	Code indicating the outcome of a simulated bomber event
BENO	Bombing encyclopedia number
BLEGNO	Index to boundary line segment
CATCODE	Category Code as reflected in Joint Resource Assessment Data Base (JAD)
CCREL	Regional reliability of offensive command and control (probability)
CEP	Circular error probable (CEP), delivery error applicable to bomber and missile weapons (nautical miles)
CLASS	Class name assigned identify sets of TYPES in data base
CLASST	Target CLASS
CNTRYLOC	Country code for country where item is located
CNTRYOWN	Country code for country which owns the item
CNTYLOCT	Target country code for country where the target is located
CNTYOWNT	Target country code for country which owns the target
CODE	Outcome code for a general event used in simulation
CPACTY	Capacity of a bomber recovery base (number of vehicles)

ATTRIBUTE NAME	DESCRIPTION
DATEIN	Earliest date in inventory (year)
DATEOUT	Latest date in inventory (year)
DEFRANGE	Typical range of interceptors at defense bases near a corridor (nautical miles)
DELAY	Delay time (e.g., launch delay time) (hours)
DELTA	Time interval between successive vehicle launches from the same base (missile or bomber) (hours)
DESIG	Target designator code, e.g., AB100, which uniquely identifies each target element included in the data base
DGX	Offset X-coordinate of desired ground zero (DGZ) (fiftieths of nautical miles)
DGY	Offset Y-coordinate of DGZ (fiftieths of nautical miles)
DHOB	Height of burst of weapon (0-ground, 1-air)
EFECNES1 } EFECNES2 }	Attributes assigned to fighter interceptor units (ICLASS = 5 in the data base): the value EFECNES1 or EFECNES2 is assigned to the attribute EFECTNES depending on value of BASEMOD input parameter POSTURE (if POSTURE=1, EFECNES1 is used; otherwise EFECNES2 value is assigned)
EFECTNES	Air defense capability (arbitrary scale) established by user to indicate relative effectiveness of air defense command and control installations and fighter interceptor bases
EVENT	Index to event type
EVENTN	Index to type of event which did not occur
FFRAC	Fission fraction (fission yield/total yield)
FLAG	Numeric code (1 through 9 permitted) used to impose restrictions on the allocation of weapons within QUICK

ATTRIBUTE NAME	DESCRIPTION
FLTNO	Flight number for a sortie
FUNCTION	Operational application code for a weapon system (e.g., ICBM)
FVALH1	Fraction of value of target in first hardness component
FVALT1	Fraction of target value that disappears by T1 (percent)
FVALT2	Fraction of target value that disappears by T2 (percent)
H1	First hardness component of a target (VULN)
H2	Second hardness component of a target (VULN)
HILOATTR	The ratio of the low-altitude attrition rate to the high-altitude rate (decimal fraction)
IALERT	Alert status; 1 = alert, 2 = nonalert
IALT	Altitude index (1 = high, 0 = low)
IATTACK	Selection index for preferential area BMD; 1 forces target selection for defense.
ICLASS	Class index assigned for game
ICLASST	Target class index
ICOMPLEX	Complex index
ICORR	Bomber corridor index number assigned in program PLANSET: 1 - Tactical (FUNCTION=TAC) aircraft corridor (TYPE name DUMMY in the data base) 2 - Naval attack corridor (TYPE name NAVALAIR in the data base) used by bomber units with PKNAV greater than zero >2 - Other corridors used by long range bombers (FUNCTION=LRA)

ATTRIBUTE NAME	DESCRIPTION
IDBL	Index to data tables for time-dependent destruction before launch probability
IDUD	Dud warhead indicator; assigned to weapons which arrive at the target but fail to detonate; 1=dud warhead
IGIW	Indices of General Industrial Worth (IGIW) (dollars)
IGROUP	Group index assigned for weapon grouping during game
IMIRV	Identifying index for system with multiple independently targetable re-entry vehicles
INDEXNO	Index of a data base item (potential target) used during processing to identify the item
INDV	Vehicle index within base
INTAR	Target index (corresponds to INDEXNO)
IPENMODE	Penetration mode; 1 = aircraft uses penetration corridor, 0 = penetration corridor not used
IPOINT	Index to a geographic point
IRECMODE	Recovery mode; 1 = aircraft should plan recovery, 0 = aircraft recovery not planned
IREFUEL	Bomber refueling code
IREG	Index to identify a geographic region
IREP	Reprogramming index (capability of missile squadron)
ISITE	Site number
ITGT	Target index number assigned by Plan Generation subsystem
ITIME	Index to time periods in time dependent DBL data tables

ATTRIBUTE NAME	DESCRIPTION
ITYPE	Type index assigned for game
ITYPET	Target type index
IVULN	Index to vulnerability number table
IWTYP2	Second warhead type
JTYPE	Type index within class
JTYPET	Target type index within class
KORSTYLE	Parameter to adjust mode of corridor penetration
LAT	Latitude (degrees)*
LEGNO	Index to line segment
LINK	The index of a leg linked to the current point
LONG	Longitude (degrees)*
MAJOR	Major reference number as reflected in the Joint Resource Assessment Data Base (JAD)
MAXFRACV	Maximum value of weapon resources to be used relative to target value (in processing MAXCOST=MAXFRACV)
MAXKILL	Desired maximum damage expected for a target
MINKILL	The required minimum damage established for a target

---

* Latitude and longitude are carried internally in the QUICK system in the following format:

North latitude	0. (equator) to +90. (North Pole)
South latitude	0. (equator) to -90. (South Pole)
East longitude	180. to 360. (Greenwich Meridian)
West longitude	0. (Greenwich Meridian) to 180.

These attributes may be input in either the above format or in standard degree, minute, second, direction format.

ATTRIBUTE NAME	DESCRIPTION
MINOR	Minor reference number as reflected in JAD to identify an item
MISDEF	Number of terminal ballistic missile interceptors for a target
MVA	Manufacturing value added (MVA); indicates the amount of value added by manufacture within a specific area (expressed in U.S. dollars)
MWHDS	Number of missile warheads penetrating area defenses to terminal defense
NADBLI	NALRTDBL for initiative attack
NADBLR	NALRTDBL for retaliatory attack
NAINT	Number of area ballistic missile interceptors at an interceptor launch base
NALRTDBL	Probability of destruction before launch (DBL) of non-alert vehicle
NALRTDLY	Delay of non-alert vehicle before commencing launch (hours)
NAME	Arbitrary alphameric descriptor for any item included in the data base
NAREADEC	Number of decoys per independent re-entry vehicle for area BMD
NASMS	Number of ASMs carried by a bomber
NCM	Number of countermeasures carried by vehicle
NDECOYS	Number of decoys on a bomber or number of decoys per independent re-entry vehicle for terminal BMD
NDET	Number of warheads detonating in current event
NEXTZONE	The adjacent zone to a side of a defense zone
NMPSITE	Number of missiles per site

ATTRIBUTE NAME	DESCRIPTION
NOALERT	Number of vehicles on alert at a base
NOBOMB1	Number of first bomb type carried by vehicle
NOBOMB2	Number of second bomb type carried by vehicle
NOINCOM	Number of delivery vehicles in commission
NOPERSQN	Number of weapon vehicles per squadron
NOPERSQ1 NOPERSQ2 NOPERSQ3	Attributes used in program BASEMOD to compute the value of the attribute NOPERSQN for bomber units; numbers 1, 2, and 3 specify surprise, initiative, and retaliatory attack plans respectively
NPEN	Number of warheads penetrating in current event
NTARG	Number of targets in missile launch event
NTINT	Number of terminal BMD interceptors at target
NWHDS	Number of warheads per independent re-entry vehicle (missiles)
NWPNS	Number of weapons in a group
NWTYPE	Warhead type
PARRIVE	Probability of bomber arrival in current event
PAYLOAD	Index which identifies entire weapon and penetration aid complement on a vehicle
PDES	Probability that launch failure destroys missile
PDUD	Probability a warhead will fail to detonate
PEN	Penetration probability for a weapon
PFPF	Probability of failure during powered flight (missiles)
PINC	Probability that a missile is in commission

ATTRIBUTE NAME	DESCRIPTION
PKMIS	Probability a missile fails to penetrate terminal defense
PKNAV	Single shot kill probability of a weapon against a naval target (a value greater than zero restricts weapon use to naval targets)
PLABT	Probability of vehicle launch abort
PLACE	Index to geographic location of an event
PLACEN	Index to geographic location of an event which did not occur
POP	Population (cities) (thousands)
POSTURE	Force readiness condition
PRABT	Probability of refueling abort
PRIMETAR	Prime target flag; 1 signifies priority target in a complex
PSASW	Destruction before launch probability assigned a weapon for a specified time period
RADIUS	Size descriptor for area targets (nautical miles)
RANGE	Vehicle range (nautical miles)
RANGEDEC	Range decrement for low-altitude aircraft flight (high range/low range)
RANGEREF	Range (nautical miles) of bomber with refueling
REL	Reliability - probability that weapon system will arrive at target given successful launch
RESERVE	Technique used to remove certain targets from weapon allocation when RESERVE = 0
SIDE	Item side name, currently either "RED " or "BLUE"

ATTRIBUTE NAME	DESCRIPTION
SITENO	Site number (currently for individual missile sites)
SPDLO	Speed at low altitude (knots)
SPEED	Speed (knots)
SQNNO	Squadron number
T1	Time of departure of first value component of a target
T2	Time of departure of second value component of a target
T3	Time of departure of third value component of a target
TAIM	Number of aim points perceived by terminal defense in current event
TARDEFHI	Level of local bomber defense at high altitude*
TARDEFLO	Level of local bomber defense at low altitude*
TASK	Target task code indicating targeting priority
TGTSTAT	Indicates target status as dynamic or nondynamic; in simulation status (alive/dead) is maintained for dynamic targets
TIME	Game time at which event occurred (hours)
TIMEN	Time planned for event which did not occur (hours)
TMDEL	Mean delay time to relaunch after a nondestructive aircraft abort (hours)

* Arbitrary units scaled by user-input parameter in Plan Generation subsystem. Minimum value 0 for no defense. Highest allowed defense level is + 7.

ATTRIBUTE NAME	DESCRIPTION
TPASW	Time at which a time period ends for DBL data tables; there may be up to 10 time periods for each table
TRETARG	Time required to retarget for known in-flight missile aborts (hours)
TTOS	Total time on station (for a tanker) (hours)
TVUL	Time a missile remains within vulnerable range of launch site (hours)
TYPE	Arbitrary alphameric designator (type name) to identify smallest sets in data base
TYPET	Target TYPE
TYPE1 } TYPE2 }	Attributes assigned fighter interceptor units (ICLASS=5 in the data base): attribute TYPE is assigned the TYPE1 or TYPE2 value based on BASEMOD input parameter POSTURE (POSTURE=1 TYPE1 is used; otherwise TYPE2 value used)
VAL	Relative value of an item within its CLASS as established in the data base by the user
VALU	Game value of an item (assigned in plan generation based on user-input parameters)
VAL1 } VAL2 }	Attributes assigned fighter interceptor units (ICLASS=5 in the data base): attribute VAL is assigned the VAL1 or VAL2 value based on BASEMOD input parameter POSTURE (POSTURE=1, VAL1 is used; otherwise VAL2 value is assigned)
VULN	Vulnerability number
WACNO	World aeronautical chart number
WHDTYPE	Warhead type index assigned in the data base
WHDTYPEN	Warhead type index (used with EVENTN)
YIELD	Yield (MT)
ZONE	An area bomber defense zone enclosed by a set of linked boundary points

APPENDIX B
GENERALIZED LAGRANGE MULTIPLIER METHOD FOR SOLVING PROBLEMS OF OPTIMUM ALLOCATION OF RESOURCES

Reprinted from OPERATIONS RESEARCH
Vol. 11, No. 3, May-June, 1963
Printed in U.S.A.

GENERALIZED LAGRANGE MULTIPLIER METHOD FOR SOLVING PROBLEMS OF OPTIMUM ALLOCATION OF RESOURCES

Hugh Everett III

Weapons Systems Evaluation Division, Institute for Defense Analyses, Washington, D. C.

(Received August 20, 1962)

The usefulness of Lagrange multipliers for optimization in the presence of constraints is not limited to differentiable functions. They can be applied to problems of maximizing an arbitrary real valued objective function over any set whatever, subject to bounds on the values of any other finite collection of real valued functions defined on the same set. While the use of the Lagrange multipliers does not guarantee that a solution will necessarily be found for all problems, it is 'fail-safe' in the sense that any solution found by their use is a true solution. Since the method is so simple compared to other available methods it is often worth trying first, and succeeds in a surprising fraction of cases. They are particularly well suited to the solution of problems of allocating limited resources among a set of independent activities.

IN MOST textbook treatments, Lagrange multipliers are introduced in a context of differentiable functions, and are used to produce constrained stationary points. Their validity or usefulness often appears to be connected with differentiation of the functions to be optimized. Many typical operations-research problems, however, involve discontinuous or nondifferentiable functions (integral valued functions, for example), which must be optimized subject to constraints.

We shall show that with a different viewpoint the use of Lagrange multipliers constitutes a technique whose goal is *maximization* (rather than location of stationary points) of a function with constraints, and that in this light there are no restrictions (such as continuity or differentiability) on the functions to be maximized. Indeed, the domain of the function to be maximized can be any set (of any cardinal number) whatever.

The basic theorems upon which the techniques to be presented depend are quite simple and elementary, and it seems likely that some of them may have been employed previously. However, their generality and applicability do not seem to be well understood at present (to operations analysts at least). The presentation will consequently place primary emphasis on the implications and applications of the basic theorems, as well as discussion of a number of techniques for extending the usefulness of the methods.

FORMULATION

FOR CLARITY of presentation, we shall develop the subject in a language of problems concerning the optimal allocation of resources. Other applications of the theorems will suggest themselves.

Let us suppose that there is a set S (completely arbitrary) that is interpreted as the set of possible strategies or actions. Defined on this strategy set is a real valued function H, called a *payoff function*. $H(x)$ is interpreted as the payoff (or utility) which accrues from employing the strategy $x \epsilon S$. In addition, there are n real valued functions $C^k(k=1\cdots n)$ defined on S, which are called *Resource* functions. The interpretation of these functions is that employment of the strategy $x \epsilon S$ will require the expenditure of an amount $C^k(x)$ of the kth resource.

The problem to be solved is the maximization of the payoff subject to given constraints $c^k, k=1\cdots n$, on each resource; i.e., to find

$$\max_{x \epsilon S} H(x)$$

subject to $C^k(x) \leqq c^k$, all k.

A particular subclass of this general problem with wide application is what will be called a *cell problem* (or separable problem) in which there are a number, m, of independent areas into which the resources may be committed, and for which the over-all payoff that accrues is simply the sum of the payoffs that accrue from each independent venture (cell). In this type of problem we have as before, for each cell, a strategy S_i, a payoff function H_i defined on S_i, and n resource functions C_i^k defined on S_i. $H_i(x_i)$ is the payoff in the ith cell for employing strategy $x_i \epsilon S_i$, and for each k, $C_i^k(x_i)$ is the amount of the kth resource expended in the ith cell by employing strategy x_i in that cell. In this case the problem to be solved is to find a strategy set, one element for each cell, which maximizes the total payoff subject to constraints c^k on the total resources expended; i.e.,

$$\max_{\substack{\text{all choices of } \{x_i\} \\ x_i \epsilon S_i}} \sum_{i=1}^{i=m} H_i(x_i)$$

subject to $$\sum_{i=1}^{i=m} C_i^k(x_i) \leqq c^k \quad \text{for all } k.$$

This type of problem is simply a subclass of the previous general problem since it can be translated to the previous problem by the following identifications:

$$S = \prod_{i=1}^{i=m} S_i \text{ (direct product set)},$$

[where a strategy $x \epsilon S$ consists of an ordered m-tuple $(x_1, \cdots, x_n)$ of strategies, one for each S_i]

$$H(x) = \sum_{i=1}^{i=m} H_i(x_i),$$
$$C^k(x) = \sum_{i=1}^{i=m} C_i^k(x_i), \quad \text{all } k$$

MAIN THEOREM AND SOME OF ITS IMPLICATIONS

We now present the main theorem concerning the use of Lagrange multipliers, and discuss its meaning and implications. The proof will be supplied in a later section.

THEOREM 1

1. λ^k, $k=1, n$ *are nonnegative real numbers,*
2. $x^* \epsilon S$ *maximizes the function*

$$H(x) - \sum_{k=1}^{k=n} \lambda^k C^k(x) \text{ over all } x \epsilon S,$$

→3. x^* *maximizes* $H(x)$ *over all those* $x \epsilon S$ *such that* $C^k \leq C^k(x^*)$ *for all* k.

Discussion

This theorem says, for any choice of nonnegative λ^k, $k=1, n$, that if an *unconstrained* maximum of the new (Lagrangian) function

$$H(x) - \sum_{k=1}^{k=n} \lambda^k C^k(x)$$

can be found (were x^*, say, is a strategy which produces the maximum), then this solution is a solution to that *constrained* maximization problem whose constraints are, in fact, the amount of each resource expended in achieving the unconstrained solution. Thus if x^* produced the unconstrained maximum, and required resources $C^k(x^*)$, then x^* itself produces the greatest payoff which can be achieved without using more of any resource than x^* does.

According to Theorem 1, one can simply choose an arbitrary set of nonnegative λ's, find an unconstrained maximum of the modified function, $H(x) - \sum_{k=1}^{k=n} \lambda^k C^k(x)$, and one has as a result a solution to a constrained problem. Notice, however, that the particular constrained problem which is solved is not known in advance, but arises in the course of solution and is, in fact, the problem whose constraints equal the resources expended by the strategy that solved the unconstrained problem.

In general, different choices of the λ^k's lead to different resource levels, and it may be necessary to adjust them by trial and error to achieve any given set of constraints stated in advance.

However, it is noteworthy that in most operations-research work one is not simply interested in achieving the optimum payoff for some given resource levels, but rather in exploring the entire range of what can be obtained as a function of the resource commitments. In this case it matters little whether this function is produced by solving a spectrum of problems with constraints stated in advance, or by simply sweeping through the λ^k's to solve a spectrum of problems whose constraint levels are produced in the course of solution. The method when applicable is therefore quite efficient if the whole spectrum of constraints is to be investigated. Even in the case where only a single constraint set is of interest the use of this method, and adjustment of the λ^k's until the constraint set is achieved, is often more efficient than alternative procedures.

A limitation of the Lagrange multiplier method arises from the fact that it does not guarantee that an answer can be found in every case. It simply asserts that if an answer can be found it will indeed be optimum.

In cases where multiple constraints are involved that are not completely independent it may not be possible to simultaneously utilize all resources to the full allowance of the constraints. This can happen if the utilization of one resource requires the utilization of others, or equivalently in cases where some constraints may involve various combinations of others. These cases are analogous to problems in linear programming where certain constraints prove to be irrelevant in the optimum solution.

In such cases one might actually find the optimum solution but be unable to establish the optimality of the result because of incompletely utilized resources. Nevertheless, there is a large class of allocation problems in which the constraints really are independent (i.e., the resources can be consumed independently in the region of interest). In such cases solutions can usually be obtained that give consumption values adequately close to the constraint values. The existence of optimum solutions that can be found by this method actually depends upon an approximate concavity requirement in the region of the solution that will be discussed more carefully later.

At this point we wish to remind the reader of the generality of Theorem 1. *There are no restrictions whatever on the nature of the strategy set S, nor on the functions H and C^k other than real-valuedness.* The strategy set may therefore be a discrete finite set, or an infinite set of any cardinality. Furthermore, the payoff function and the resource functions can take on negative as well as positive values. [$C^k(x)$ negative may be interpreted as *production* rather than expenditure of the kth resource.]

Application to Cell Problem

One of the most important applications of Theorem 1 is in the solution of cell problems. As shown in the Formulation Section, these problems are a subclass of the general problem to which Theorem 1 is applicable. In this case, maximizing the unconstrained Lagrangian function

$$H(x) - \sum_{k=1}^{k=m} \lambda^k C^k(x)$$

is equivalent to finding

$$\max_{x \in \prod_{i=1}^{i=m} S_i} \left[\sum_{i=1}^{i=m} H_i(x_i) \right] - \sum_{k=1}^{k=n} \lambda^k \left[\sum_{i=1}^{i=m} C_i^k(x_i) \right],$$

which (interchanging summation order) is the same as:

$$\max_{x \in \prod_{i=1}^{i=m} S_i} \sum_{i=1}^{i=m} \left[H_i(x_i) - \sum_{k=1}^{k=n} \lambda^k C_i^k(x_i) \right].$$

But, since the choices x_i may be made independently in each cell as a consequence of $S \equiv \prod_{i=1}^{i=m} S_i$, the sum is obviously maximized by simply maximizing

$$H_i(x_i) - \sum_{k=1}^{k=n} \lambda^k C_i^k(x_i)$$

in each cell independently of strategy choices in other cells, and summing the payoffs and resources expended for each cell (for the strategy that maximized the Lagrangian for that cell) to get the total payoff and resource expenditures. Theorem 1 then assures us that the result of this process is a solution to the over-all constrained problem with constraints equal to the total resources expended by the strategy produced by this procedure.

Observe that there is no possibility that just a local maximum to the over-all problem has been obtained. If the Lagrangian in each cell has been correctly maximized (i.e., is not itself merely locally maximized), then theorem 1 guarantees that the result is a *global* maximum to the over-all problem.

Theorem 1 says nothing about the manner in which one obtains the maxima of the unconstrained Lagrangian functions, but simply asserts that if one can find them, then one can also have maxima of a problem with constraints. The Lagrange multipliers therefore are not a way in themselves of finding maxima, but a technique for converting optimization problems with constrained resources into unconstrained maximization problems.

This conversion is especially crucial for cell problems with constraints on total resource expenditures, where the conversion to unconstrained maximization of the Lagrangian function uncouples what was an essentially combinatorial problem (because of the interaction of choices in each cell through total resource constraints) into a vastly simpler problem involving independent strategy selections in each cell.

The present treatment of Lagrange multipliers was motivated, in fact, by a cell problem involving continuous, differentiable payoff functions, the solution of which was attempted by a classical Lagrange multiplier approach. In this case, the resulting (transcendental) equations had in many circumstances a multiplicity of solutions, and the embarrassing problem arose as to which of several solutions to select for each cell. It appeared as though it might be necessary to try all combinations of choices of solutions an impossible task in this case which involved several hundred cells. As a result of this difficulty, a closer look was taken at the role of Lagrange multipliers, and the present treatment is the result. The original problem of multiple solutions is, of course, easily solved by simply selecting that solution in each cell which gives the largest value for the Lagrangian.

It is the recognition that the objective is to maximize the Lagrangian, by whatever means, not to zero its derivative, which is decisive. In many cases it *is* expeditious to maximize the Lagrangian by finding zeroes of its derivative. One can then easily select a final value by testing each solution (if there is more than one) to find which gives the largest (global) maximum. This procedure automatically excludes any solutions that correspond to minima or saddle values, and also facilitates taking into account any boundary conditions (such as nonnegative resource constraints) by testing the boundary cases as well.†

In other cases (particularly cases of nonnumerical strategies, or discrete strategy sets such as integers), the Lagrangian may best be maximized by trial and error procedures, or even direct computer scanning of all possibilities.

Another possibility is illustrated by cases wherein resources may be applied only in integral numbers. Often in such cases one can define a continuous differentiable payoff function that attains its correct value on the integers. A useful trick applicable to many such cases is to maximize analytically the Lagrangian based upon the continuous function, and then test the integer on each side of the solution, selecting the one that maximizes the Lagrangian.

PROOF OF MAIN THEOREM

THE PROOF of the main theorem presented and discussed in the previous section is quite elementary and direct:

Proof of Main Theorem. By assumptions (1) and (2) of Theorem 1, λ^k, $k=1\cdots n$, are nonnegative real numbers, and $x^*\epsilon S$ maximizes

$$H(x)-\sum_{k=1}^{k=n} \lambda^k C^k(x)$$

over all $x\epsilon S$ (the x^* producing the maximum may very well not be unique all that we require is that x^* be *some* element that maximizes the Lagrangian). This means that, for all $x\epsilon S$,

$$H(x^*)-\sum_{k=1}^{k=n} \lambda^k C^k(x^*) \geqq H(x)-\sum_{k=1}^{k=n} \lambda^k C^k(x),$$

† This type of constraint (e.g., nonnegativity of resources), which holds independently for each cell rather than over-all as with total resources, is handled by simply restricting the strategy set for the cell appropriately. The Lagrange multipliers are reserved for over-all constraints.

and hence, that

$$H(x^*) \geq H(x) + \sum_{k=1}^{k=n} \lambda^k [C^k(x^*) - C^k(x)]$$

for all $x \epsilon S$. But if the latter inequality is true for all $x \epsilon S$, it is necessarily true for any subset of S, and hence true on that subset S^* of S for which the resources never exceed the resources $C^k(x^*)$. Notationally: $x \epsilon S^* \Leftrightarrow$ for all k, $C^k(x) \leqq C^k(x^*)$. However, on the subset S^* the term

$$\sum_{k=1}^{k=n} \lambda^k [C^k(x^*) - C^k(x)]$$

is nonnegative by definition of the subset and the nonnegativity of the λ^k's, hence our inequality reduces to $H(x^*) \geqq H(x)$ for all $x \epsilon S^*$, and the theorem is proved.

LAMBDA THEOREM

THEOREM 2

1. Let $\{\lambda_1^k\}, \{\lambda_2^k\} k = 1 \cdots n$ *be two sets of* λ^k*'s that produce solutions* x_1^* *and* x_2^**, respectively. Furthermore, assume that the resource expenditures of these two solutions differ in only the jth resource.*

$$C^k(x_1^*) = C^k(x_2^*) \text{ for } k \neq j$$

and that $C^j(x_1^*) > C^j(x_2^*)$.

2. Then: $\lambda_2^j \geqq [H(x_1^*) - H(x_2^*)]/[C^j(x_1^*) - C^j(x_2^*)] \geqq \lambda_1^j$.

This theorem states that, given two optimum solutions produced by Lagrange multipliers for which only one resource expenditure differs, the ratio of the change in optimum payoff to the change in that resource expenditure is bounded between the two multipliers that correspond to the changed resource.

Thus the Lagrange multipliers, which were introduced in order to constrain the resource expenditures, in fact give some information concerning the effect of relaxing the constraints.

In particular, if the set of solutions produced by Lagrange multipliers results in an optimum payoff that is a differentiable function of the resources expended at some point, then it follows from Theorem 2 that the λ^k's at this point are in fact the partial derivatives (or total derivative in case of one resource) of the optimum payoff with respect to each resource (all other resources kept constant):

$$[\partial H^* / \partial C^j]_{C_k \text{ constant}, k \neq j} = \lambda^j.$$

Proof. The proof of Theorem 2 is also quite elementary. By hypothesis x_1^* is the solution produced by $\{\lambda_1^k\}$, hence x_1^* maximizes the Lagrangian for $\{\lambda_1^k\}$, which implies:

$$H(x_1^*) \geqq H(x) + \lambda_1^j [C^j(x_1^*) - C^j(x)] + \sum_{k \neq j} \lambda_1^k [C^k(x_1^*) - C^k(x)]$$

holds for all $x \epsilon S$, and hence in particular holds for x_2^*. But since by hypothesis $C^k(x_1^*) = C^k(x_2^*)$ for $k \neq j$, we can deduce that

$$H(x_1^*) \geqq H(x_2^*) + \lambda_1^j [C^j(x_1^*) - C^j(x_2^*)],$$

which, since by hypothesis $C^j(x_1^*) > C^j(x_2^*)$, implies that:

$$[H(x_1^*) - H(x_2^*)]/[C^j(x_1^*) - C^j(x_2^*)] \geqq \lambda_1^j,$$

which proves one side of the conclusion of Theorem 2. Interchanging the roles of x_1^* and x_2^* [and observing the reversal of the sign of

$$C^j(x_1^*) - C^j(x_2^*)]$$

produces the other side of the inequality to complete the proof of Theorem 2.

An obvious consequence of Theorem 2 is the fact that, if all but one resource level is held constant, the resource that changes is a monotone *decreasing* function of its associated multiplier. This fact indicates the direction to make changes when employing a trial and error method of adjusting the multipliers in order to achieve some given constraints on the resources.

The Lambda Theorem also suggests a potentially useful technique for choosing a starting set of multipliers for such a trial-and-error method of achieving given constraint levels in a cell problem. Beginning with any reasonably good allocation of the given resources, one can often calculate easily what the effect on the payoff is for a small additional increment of each resource, optimally placed within the cells. The differential payoff divided by the increment of resource is then taken as the starting λ for that resource. The λ's are then adjusted by trial and error until the Lagrange solution corresponds to the given constraints, producing the optimum allocation.

THE EPSILON THEOREM

A NATURAL question with respect to the practical application of the Lagrange method concerns its stability—supposing that as a result of methods of calculation or approximation one cannot precisely maximize the Lagrangian, but can only guarantee to achieve a value close to the maximum. Such a solution can very well be at a drastically different resource level and payoff than that which actually achieves the maximum, and yet produce a value of the Lagrangian very near to the maximum. For the method to be practical, it is required that in this situation a solution that nearly maximizes the Lagrangian must be a solution that also nearly maximizes the payoff for the resource levels that *it itself* produces (which may be quite different than those of the solution that actually maximizes the Lagrangian). Only in such a circumstance would it be safe to assert that the solutions produced by any nonexact procedures (such as numerical computation with finite accuracy, or methods based upon approximations) were in fact approximately optimal solutions to the constrained problem. Such required assurance of insensitivity is supplied by the following ('epsilon') theorem.

THEOREM 3

1. $\bar{x}$ comes within ϵ of maximizing the Lagrangian, i.e., for all $x \epsilon S$:

$$H(\bar{x}) - \sum \lambda^k C^k(\bar{x}) > H(x) - \sum \lambda^k C^k(x) - \epsilon.$$

→2. $\bar{x}$ is a solution of the constrained problem with constraints $c^k = C^k(\bar{x})$ that is itself within ϵ of the maximum for those constraints.

The proof of this theorem, which is a simple extension of Theorem 1, exactly parallels the proof of Theorem 1 (with an added ϵ) and will not be repeated.

ADDITIONAL REMARKS, CONCLUSIONS, AND COMPUTATIONAL PLOYS

Gaps or Inaccessible Regions

Theorem 1 assures us that any maximum of the Lagrangian necessarily is a solution of the constrained maximum problem for constraints equal to the resource levels expended in maximizing the Lagrangian.

The Lagrange multiplier method therefore generates a mapping of the space of lambda vectors (components λ^k, $k = 1, \cdots, n$) into the space of constraint vectors (components c^k, $k = 1 \cdots n$). There is no a priori guarantee, however, that this mapping is onto—for a given problem there may be inaccessible regions (called *gaps*) consisting of constraint vectors that are not generated by any λ vectors. Optimum payoffs for constraints inside such inaccessible regions can therefore not be discovered by straightforward application of the Lagrange multiplier method, and must hence be sought by other means.

The basic cause of an inaccessible region is nonconcavity in the function of optimum payoff vs. resource constraints (convexities in the envelope of the set of achievable payoff points in the space of payoff vs. constraint levels). This possibility, and several methods for dealing with it, will now be investigated.

Before beginning this investigation, however, we wish to point out that even though the Lagrange multiplier method is not certain to obtain the desired solutions in all cases, any solutions that it does yield are guaranteed by Theorem 1 to be true solutions. The procedure is therefore 'fail-safe,' a very reassuring property. It has been our experience over the last several years, which includes application of this method to a variety

of production and military allocation problems, that the method has been extremely successful, and nearly always has directly yielded all solutions of interest. The few situations in which the direct method failed were readily solved by simple modifications to the procedure, some of which will now be mentioned.

Source of Gaps

Consider the $(n+1)$ dimensional space of payoff vs. resource expenditures. This space will be called PR space for brevity. Every strategy $x \epsilon S$ maps into a point in this space corresponding to $H(x), C^k(x)$ $(k=1\cdots n)$. The entire problem is therefore represented by this set of accessible points in PR space. The problem of finding the maximum of H subject to constraints c^k, $k=1\cdots n$, is simply the problem of selecting that point of our set in PR space of maximum H that is contained in the subspace of PR space where the resources are bounded by the c^k's. The set of all such points (corresponding to all sets of values in the c^k's) will be called the *envelope*, and constitutes the entire set of solutions for all possible constraint levels.

Consider now any solution x^* produced by a set of Lagrange multipliers (λ^k). By definition x^* maximizes the Lagrangian; consequently we have that

$$H(x^*) - \sum \lambda^k C^k(x^*) \geqq H(x) - \sum \lambda^k C^k(x)$$

for all $x \epsilon S$. Rearranging terms slightly, we have:

$$H(x) \leqq H(x^*) - \sum \lambda^k C^k(x^*) + \sum \lambda^k C^k(x)$$

for all $x \epsilon S$. If we consider now the hyperplane in PR space defined by $H=\sum \lambda^k C^k + \alpha$ where $\alpha = H(x^*) - \sum \lambda^k C^k(x^*)$, we see that, because of the previous inequality, none of the accessible points in PR space lies above this hyperplane, and at least one point, $H(x^*), C^k(x^*)$ $k=1\cdots n$, lies on it.

Each solution produced by Lagrange multipliers therefore defines a bounding hyperplane that is tangent to the set of accessible points in PR space at the point corresponding to the solution (hence tangent to the envelope), and which constitutes an upper bound to the entire set of accessible points. It is clear that, since no such tangent bounding hyperplanes exist in regions where the envelope of accessible points in PR space is not concave, the Lagrange multiplier method cannot produce solutions in such a region. Conversely, for any point on the envelope (solution) where a tangent bounding hyperplane *does* exist (envelope concave at the point), it is obvious that there exists a set of multipliers (namely the slopes of the hyperplane) for which the strategy corresponding to the point in question maximizes the Lagrangian.

Thus the Lagrange method will succeed in producing all solutions that correspond to concave regions of the envelope (optimized payoff vs. constraint level), and fail in all nonconcave regions.

A fortunate feature of cell problems with many cells is the fact that, even though there may be large convexities in the envelope in the PR space for each cell, the result of over all optimization is an envelope in the PR space for the total problem in which the convexities are vastly reduced in significance.† This property is the major reason for the general success of the Lagrange method in solving cell problems.

Some Methods for Handling Gaps

Despite the general success of Lagrange multipliers (at least for the problems we have encountered), occasions may arise where gaps occur in regions of critical interest. Under such circumstances there are several useful techniques that can be attempted before abandoning the procedure altogether.

First, all solutions that can be obtained outside the gaps contribute a good deal of information and can be used to bound the solution in the gap region. As was previously shown, each solution that can be obtained by Lagrange multipliers defines a bounding hyperplane that gives an upper bound to the maximum payoff at all points, and hence inside the gap as well. For any point inside a gap, therefore, an upper bound can be obtained by finding the minimum payoff for that point over the set of bounding hyperplanes corresponding to the solutions that one could calculate.

On the other hand, every solution that can be obtained that has the property that none of its resource expenditures exceeds the resources of a point in a gap for which one is seeking bounds, obviously constitutes a *lower bound* to the optimum payoff at the point in question, and the maximum of these lower bounds can be selected as a lower bound to the payoff in question. Thus the set of solutions that *can* be obtained by Lagrange multipliers can be used to obtain bounds on the optimum payoff for inaccessible regions.

There is another technique that is often successful in reducing gaps in instances where the bounds one can compute leave too large a region of uncertainty, and where the gap is caused by degeneracy in which a number of cells have gaps corresponding to the same multiplier. A gap is char-

† In fact, the gap structure for the over all problem obviously simply reflects faithfully the gap structure in the individual cells, with each gap in a cell corresponding to a given multiplier value occurring with the same magnitude (same jump in payoff and resources) in the over-all optimization at precisely the same multiplier value. Only degeneracies in which several cells have gaps corresponding to the same multiplier can cause a larger gap in the over all problem, and such degeneracy is easily removed by techniques to be discussed in the following section

acterized by the behavior that, as the λ's are continuously varied, there are abrupt discontinuities in the resource levels generated. These discontinuities can often be filled in cell problems by the following technique.

Given two sets of λ's, (λ_1^k), (λ_2^k), which are very close, but for which the generated resource levels markedly differ, one can make a *mixed* calculation in a cell problem using the set (λ_1^k) in some cells and the set (λ_2^k) in the others. If the two sets of λ's are close together, maximizing the Lagrangian in any cell for one set will necessarily result in a solution that nearly maximizes the Lagrangian for the other set, hence by the Epsilon Theorem will yield a result that is guaranteed to be nearly optimum.

Somewhat more generally, one can simply exploit the Epsilon Theorem directly in a cell problem, working with a given set of λ's but deliberately modifying the choices in some or all cells in a way which moves in the direction of the desired expenditure of resources. By summing the deviations from maximum of the Lagrangian in each cell (epsilons) in which the strategies are so modified, a bound on the error of the result is obtained (which can be kept quite small in most cases by judicious choice of deviations). This appears to be a quite powerful strategem.

•   •
•   •
•   •

EFFECTIVE PAGES - 31 August 1972

This list is used to verify the accuracy of CSM AM 9A-67, Volume II after change 1 pages have been inserted. Original pages are indicated by the letter O, and change 1 by the numeral 1.

Page No.	Change No.
Title Page	0
ii-ix	0
1-42	0
43-45	1
46-295	0
296	1
297-298	0

each side. (The QUICK Simulation subsystem also considers a preferential area defense against ballistic missiles.)

Terminal defenses are modeled by a subtractive model. Each target with terminal defenses is assigned a number of terminal ballistic missile interceptors. This number of interceptors (variable MISDEF) is input in the data base via the attribute NTINT which must be defined for each defended target.

The input variables describing the target's terminal defense capability allow uncertainties to be introduced in the number of interceptors present. MISDEF is the "nominal" number of interceptors on the target, each with kill probability PKTX against an unhardened warhead. In addition, four other parameters are defined (the same for all targets) which introduce uncertainties in MISDEF. RXLOW is a factor which, when multiplied by MISDEF, gives a lower estimate of interceptors which has probability PXLOW of occurring. Likewise, RXHIGH and PXHIGH define the overestimate of interceptor availability. Thus, if there is imperfect knowledge of the defense capability, the allocator can hedge against these uncertainties when assigning weapons.

In addition to the target-associated defense data, it is possible to describe penetration aids suitable for the various missiles by means of the Payload Table. For a particular payload index, the following variables* describe the penetration aids:

---

*NWHD is data base attribute NWHDS; NTDECOYS is attribute NDECOYS: | XDEG is not currently implemented as a feature in the QUICK system.

NWHD	=	Number of warheads per independent re-entry vehicle package.
NTDECOYS	=	The number of "aim points" the terminal defense sees for each independent re-entry vehicle (in addition to the warheads).
XDEG+	=	A factor by which the PKTX is multiplied to obtain terminal interceptor kill probability against this weapon type. It reflects additional hardening of the warhead or electronic penetration aids which can degrade interceptor effectiveness.

An independent re-entry vehicle package is a set of warheads and terminal decoys that can be guided to a target point (or points) independently. For missile boosters with a multiple independently targetable re-entry vehicle capability (MIRV), there may be several independent RVs per booster. Otherwise, each booster delivers one set of warheads and decoys.

The penetration probability of any warhead is a function of all the missiles allocated to the target. The model computes the total number of objects allocated to the target, NOBJ, as the sum of all warheads and decoys* allocated to the target. The number of perfect interceptors, variable PINT, is defined as:

+XDEG is not currently implemented as a feature in the QUICK system.

*For each weapon, this is the sum of NWHD and NTDECOYS multiplied by the product of the survival before launch probability, weapon system reliability, and command and control reliability.

PINT=PKTX*[(PXLOW*RXLOW)+(PXHIGH*RXHIGH)+(1-PXLOW-PXHIGH)]*MISDEF

This variable is the expected number of objects to be removed by the terminal defense interceptors.

The penetration probability for any warhead is defined as:

$$1.0 - \left[\text{XDEG}* \frac{\text{PINT}}{\text{NOBJ}} \right]$$

If this probability is less than (1.0 - PKTX*XDEG), it is reset to that value.*

BOMBER REFUELING

Refueling Modes

The QUICK design provides for modeling two kinds of bomber refueling capabilities: "buddy" and area. In buddy refueling, two aircraft take off together and fly to the refuel point; one then provides fuel to the second and recovers. Fuel can be provided by either a tanker or another bomber of the same squadron as the aircraft being refueled.

There are two types of area refueling: directed and automatic refueling. In the directed mode, the user establishes, in the data base, a specific refueling area (up to 20 per side may be defined in the data base) and manually assigns the appropriate bombers and tankers to this area. In

*XDEG is not currently implemented in QUICK. In these formulas it should be replaced by 1.0.

the automatic mode, the Plan Generator (program PLNTPLAN) develops the refueling plan on the basis of information provided in the data base. The data base reflects the bomber squadrons which require refueling and the tankers which are available. Program PLNTPLAN then selects the refueling area (up to 30 additional refueling areas may be added) and assigns the bombers and tankers accordingly. To reflect the refueling requirements associated with a specific plan, the user defines the attribute IREFUEL for all bomber and tanker units defined in the data base. The codes which may be assigned as the value of IREFUEL are as follows:

IREFUEL Setting	Definition
-5	Automatic refueling -- two refuelings required.
-4	Automatic refueling -- one refueling required.
-3	This code is used to flag air-breathing missiles which are to be treated as aircraft when calculating attrition rates - no refueling involved.
-2	Buddy refueling -- a bomber from the same squadron is used in a tanker role.
-1	Buddy refueling in which support is provided by a tanker. Tanker units associated with buddy refueling need not be defined in the data base.
0	No refueling required.
≥ 1	Directed area refueling -- refuel area and bomber/tanker assignments are directed by user.

acterized by the behavior that, as the λ's are continuously varied, there are abrupt discontinuities in the resource levels generated. These discontinuities can often be filled in cell problems by the following technique.

Given two sets of λ's, (λ_1^k), (λ_2^k), which are very close, but for which the generated resource levels markedly differ, one can make a *mixed* calculation in a cell problem using the set (λ_1^k) in some cells and the set (λ_2^k) in the others. If the two sets of λ's are close together, maximizing the Lagrangian in any cell for one set will necessarily result in a solution that nearly maximizes the Lagrangian for the other set, hence by the Epsilon Theorem will yield a result that is guaranteed to be nearly optimum.

Somewhat more generally, one can simply exploit the Epsilon Theorem directly in a cell problem, working with a given set of λ's but deliberately modifying the choices in some or all cells in a way which moves in the direction of the desired expenditure of resources. By summing the deviations from maximum of the Lagrangian in each cell (epsilons) in which the strategies are so modified, a bound on the error of the result is obtained (which can be kept quite small in most cases by judicious choice of deviations). This appears to be a quite powerful strategem.

• •

• •

• •

DISTRIBUTION

Addressee	Copies
NMCSSC Codes	
B121	3
B122 (stock)	6
B200	1
B210	2
B220	29
B600	1
DCA Codes	
920	1
950	1
System Engineering Facility, ATTN: T221 Reston, Virginia 22070	1
OJCS Studies, Analysis and Gaming Agency, ATTN: SFD, Room 1D957, Pentagon, Washington, D.C. 20301	5
Commander-in-Chief, North American Air Defense Command ATTN: NPPG, Ent Air Force Base, Colorado 80912	2
Commander, U.S. Air Force Weapon Laboratory (AFSC) ATTN: AWL, Kirtland Air Force Base, New Mexico 87117	2
Director, Strategic Target Planning Offutt Air Force Base, Nebraska 68113	2
Chief of Naval Operations, ATTN: OP963G Room 5E531, Pentagon, Washington, D.C. 20350	2
Defense Documentation Center, Cameron Station, Alexandria, Virginia 22314	12
	70