OPERATIONS RESEARCH TECHNIQUES APPLIED TO MINE PRODUCTION PLANNING

by

Franz Michel Posaner
B.Sc.(Hons.)(Adelaide)

Thesis submitted for the degree of
Doctor of Philosophy
in the University of Adelaide,
Department of Applied Mathematics,
May, 1974.

TABLE OF CONTENTS

<u>CONTENTS</u> Page

SUMMARY

This thesis demonstrates how nonlinear optimization techniques such as branch and bound and integer programming can be successfully adapted to solve mathematical models for mine production planning. Application is made to two contrasting mining environments. First we deal with short and medium term production planning from an open pit mine where there are grade constraints on the mill feed. While these constraints are linear, nonlinearities arise from the need to take into account other aspects such as machine movements or precedence relationships in extracting blocks from the mine.

The second application of the techniques relates to long term production planning from a beach sand deposit which is to be mined by a dredge. A network model describing the various dredge movements is formulated and an effective heuristic algorithm is constructed to enable the determination of near optimal dredge paths.

In both applications, computer investigations have proven that the methods are effective and several examples are given in the thesis to demonstrate this.

SIGNED STATEMENT

I hereby declare that this thesis contains no material which has been accepted for the award of any other degree or diploma in any University and, to the best of my knowledge and belief, it contains no material previously published by another person, except where due reference is made in the text of this thesis.

F.M. Posaner

ACKNOWLEDGEMENTS

I would like to express my deep gratitude to my supervisor, Professor R.B. Potts, for his help and encouragement throughout all stages of the development and presentation of this thesis.

In addition I am indebted to staff of the Operations Research/Computer Services Group of the Australian Mineral Development Laboratories for much fruitful interaction and in particular would like to thank Dr. W.J. Howarth, Mr. W.P. Boyce and Mr. G.L. Wood.

I am also grateful for the financial support of a Commonwealth Postgraduate Award and for the use of computing facilities at the University of Adelaide. Finally I would like to thank Miss D.J. Potter for her excellent job in typing this thesis.

CHAPTER 1

INTRODUCTION

The subject of this thesis is the application of mathematical and computer models to mine production planning The past decade or so has witnessed rapid growth in the use of computers in the mining industry. They have proven to be an ideal tool for recording and collating the vast quantities of data which are gathered during mining operations. Many companies have successfully used computers for these purposes, and are now directing their attention towards extending computer usage to enable them to assist more directly in the planning process. Mathematical models are gaining increasing importance in this area since there are often many variables and complex interrelationships which must be taken into account in problems of mine planning.

Coyle [7] has recently reviewed the applications of Operations Research in the mining industry as a whole. He gives many examples of the use of Operations Research techniques, starting with exploration and extending right through to the determination of optimal plant design and process control. A wide range of OR techniques have already found application - to list but a few would include Search Theory, Statistical Theory, Applied Probability and Geostatistics, Optimization Theory, Simulation and Critical Path Analysis.

The techniques applied to the mine production planning problems discussed in this thesis are optimization techniques. To date, the greatest impact of optimization techniques has been more in the area of Open Pit Mine Design. Dynamic Programming, Graph and Network Theory have, for example, all met with substantial application [18,26,16], and currently a considerable number of computer programs based on these methods are available. The state of the art is not so well developed in the field of mine production planning. This is partly due to the increased complexity of the types of scheduling problems which occur in this area and also because there is greater dependence on peculiarities of the mine-mill system under consideration. Whereas quite a universal approach has been possible in open-pit mine design this has not been so in mine production planning.

Linear Programming has been virtually the only optimization technique applied to such problems to date [7,33]. The aim of this thesis is to show that the range of applicable techniques can in fact be extended to include techniques of nonlinear, integer programming and branch and bound. To demonstrate that these methods are effective, application is made to two contrasting mining environments.

In Chapter 2 we apply the techniques to the optimum scheduling of extraction operations in shallow open-pit mines where there are grade constraints on the mill feed, while in Chapter 3 further application is made to the problem of determining an optimal path for a dredge in a beach sand deposit.

A number of authors have already considered the first problem of production planning with grade constraints on the mill feed. Apart from Linear Programming, Simulation has been the only other tool commonly employed, the relative emphasis between the two depending somewhat on the complexity of the problem under investigation and the purposes of the study. The fact that Linear Programming has received wide use is not surprising, since the mill feed is formed by <u>blending</u> together ores which may be obtained from several locations in the mine and which may therefore possess quite large variations in composition. However, the Linear Programming approach is somewhat limited in that it is unable to account in detail for the way in which the mine and mill are interrelated. For instance, it is also necessary to take machine movements into account, and, when mining takes place at mine faces, there are constraints that a block does not become available until a face advances into it or a new one is created therein. The latter constraints are of a logical nature and will be referred to as "mining constraints". Along with machine movements they introduce important nonlinearities into the model and can only be taken into account using more advanced techniques of nonlinear optimization. It is true that in general, nonlinear optimization techniques require considerable amounts of computing time and are often inefficient for large problems. However, the production planning problems considered in this thesis fortunately have special structure which we have been able to exploit to produce computationally efficient algorithms.

Coyle's review [7], with its extensive bibliography of 119 papers, covers the earlier contributions to the literature in great detail. We shall consequently highlight some of the more important ones. Redmon [35], Hewlett [15] and White and Peiker [39] were among the first authors to apply the straightforward Linear Programming approach to optimal ore blending. The applications they described related to underground mining but in fact few features peculiar to underground mining actually appear in the LP formulations. Also the effect of time, that is having to schedule operations to achieve grade control over short consecutive periods of time, was not considered fully. More recently, Williams et al. [40] have succeeded in taking some of the mining constraints into account in the planning of underground copper mining and the computer program they produced has reportedly been a great time saver in their planning department. Others have used Linear Programming in the production planning process but for more general purposes than just grade control, for example, see Manula and Kim [29], Faulkner [11] or Ahlbach [1].

Simulation has been the main tool in which planners, especially those working in open-pit mining, have dealt with the additional mining constraints. Kaas [20], for instance, used simulation in connection with an open-pit mining and milling operation in which blocks of ore were extracted by a certain number of shovels with the objective of minimising variations in the mill feed. Kaas divided the mine up into "shovel areas", one for each shovel. When any shovel completed its block, the strategy was to select the next one to be mined from a smaller set of currently "available blocks" within that particular shovel area, in such a way as to cause the smallest variation in the mill feed. Sheinkin and Julin [37] and Mathias [30] have applied similar simulation approaches to underground mining.

The simulation in use is almost exclusively of the deterministic type - stochastic simulation is a rare luxury since only a limited knowledge of grades and tonnages is ever available. Deterministic simulation fulfills a less responsible role in the planning process than optimization in conjunction with mathematical models. Simulation is concerned more with performing detailed investigation of various "sensible" strategies or system configurations as proposed by the mining engineer. It accepts the burden of detailed computation but relies on the mining engineer to make the decisions. This thesis demonstrates that enabling the computer to participate more directly in decision making through the use of mathematical models, greatly improves the efficiency of the planning process. It is even suggested by the author that, with the right approach and with computer packages which suit the mining engineer or manager, there can be an increase in the enthusiasm with which planning problems are tackled. It should be noted that only the important variables and constraints will be incorporated in these mathematical models and this means tha when the final plans are produced they can still be analysed in greater depth using simulation programs.

Fraser [13] and his colleagues Jutsum and Milner [18 are some of the few authors who have tried to take into account mining constraints with a Linear Programming model for scheduling production from an open-pit mine. That the problem and the resultant mathematical models are quite dependent on the peculiarities of the mining system under consideration, may be seen by contrasting the recent work of Rogado [36] with that of Fraser or Kaas. Rogado uses scrapers to mine the ore whereas the other two use draglines and excavators or shovels and since these operate at mine faces there is considerably less accessibility to blocks of favourable grade. This dependence of the problem on the mining system and on management priorities explains the relative proliferation of different models and computer programs which have been applied to this seemingly universal problem.

A considerable number of important papers not explicitly mentioned in this thesis are published in various proceedings of the annual Symposia on the Applications of Computers in the Mineral Industries. One significant contribution which is not so accounted for, is that of Lutjen [28]. He gives an enlightening account of the production scheduling problem from the point of view of a practising mining engineer faced with the responsibility of specifying day to day operations in the open pit nickel mine at Nicaro, Cuba. Lutjen emphasizes how important the grade constraints are and shows the profound effect of them right back in the mine. This feature is typical of lateritic deposits, of which Nicaro is an example, for they characteristically show sudden and large changes in grade.

Production Planning is a continuing process in the life of a mine. It starts with the initial feasibility study and continues on in the form of detailed production scheduling during advanced stages of development. A significant reason for this is that new information is continually being gathered as mining proceeds. In the initial stages we have a rather limited knowledge of the deposit as a whole and are concerned with determining broad scale (i.e. long term) development and production plans. By contrast, much more information about the deposit is available in the more advanced stages, especially in the immediate vicinity of the machines involved in extracting the ore, and we then have to determine detailed operations of the machines, for example on a daily basis.

In Chapter 2 the problem of mine-mill production planning is considered from both the medium and short term viewpoints. Three models are formulated to plan production from an open pit mine so as to achieve grade control in an optimum manner. The models apply to situations in which ore is extracted at mine faces, for example by draglines, front-end loaders or shovels. In the first model presented, in section 2.2, the task is to perform multiperiod scheduling over a medium term planning horizon (e.g. one or two years) with the prime aim of minimising machine movements between fairly large subareas of the mine. In this case, while the model is nonlinear, an efficient heuristic algorithm, based on repeated applications of linear programming, can be constructed to enable production from the various subareas to be sequenced in time in a near optimal manner. The output of the heuristic algorithm indicates how the mine should be developed and provides general goals to aim at when carrying out shorter term, day to day planning.

The second model, presented in section 2.3, concentrates in detail on the mining constraints and is an extension of the work done by Fraser [13]. It applies to short term day to day planning and the orebody in the immediate vicinity of each machine is divided into much smaller "production" blocks for which a more detailed knowledge of grades and tonnages has been obtained. Each machine will consume several of these production blocks during any period (day or week, for example), the precise manner depending on the grade and mining constraints.

In the short term, we assume that machine movements are not as important as satisfying the mining constraints and we can therefore avoid having to take them explicitly into account in the model. While the mining constraints are still nonlinear, Fraser has been able to use an heuristic procedure to produce locally optimal solutions. In section 2.3, it is shown how branch and bound methods may be easily adapted to produce globally optimal solutions to the mathematical programs. Apart from being particularly efficient from the computational viewpoint, the branch and bound method developed in this section has a great deal of intuitive appeal. As usual branch and bound provides a systematic way of enumerating all possibilities[1], but we do this in a manner which can be quite easily understood and used by a mining engineer faced with the same problem, (or at least smaller versions of it).

It is possible to formulate the problem as a mixed integer program and indeed branch and bound methods are widely used in integer programming algorithms. However, formulating it this way requires the addition of a large number of 0-1 variables and even more supplementary constraints to describe the logic behind the various mining restraints and the problem quickly becomes too large to handle. In contrast, the branch and bound methods used in this section have been specially adapted to utilise the structure of the mining constraints and thus avoid having to introduce these extra variables and constraints into the model. The procedure is nevertheless entirely analogous to the well known approaches used in integer programming. These have been well described by Beale [4], Beale and Small [5] and Tomlin [38], for example. The difference is largely one of format but there are quite startling improve-

---

[1] In general, only a small proportion of all possibilities needs to be considered explicitly though.

ments in computational efficiency when a method based directly on the mining constraints is used.

The approach of Fraser, and of section 2.3, is based on a model for operations over one period (e.g. a day) only. In section 2.4 the formulation is extended to perform multiperiod short term planning e.g. for a week in advance. In doing this many more possibilities have to be considered, but this is of course why the multiperiod approach can obtain an improved solution compared with repeated application of the single period model.

Under certain simplifying assumptions the constraint matrix of the linear program for the multiperiod formulation has the following structure

$$\mathcal{A} = \left[\begin{array}{c|ccccc} E_0 & E_1 & \cdots & & & E_n \\ \hline & B_1 & & & & \\ & & B_2 & & & \\ & & & \ddots & & (0) \\ 0 & & & & \ddots & \\ & & (0) & & & B_n \end{array}\right]$$

This structure is typical of many multiperiod optimization problems [9,16,22]. In this case the submatrix

$$\begin{bmatrix} B_1 & & & & \\ & B_2 & & (0) & \\ & & \cdot & & \\ & (0) & & \cdot & \\ & & & & B_n \end{bmatrix}$$

expresses grade control requirements over separate periods while the upper portion of A consists of reserves constraints which serve to tie the operations over the individual periods together. There are many ways of exploiting the above structure. Johnson [16,17], for example, has applied Dantzig-Wolfe decomposition techniques [9] to multi-period mine production scheduling problems possessing similar structure. In section 2.4, however, it is shown that the contracted basis techniques of Lasdon [25] and Kaul [21] are particularly suitable for use with the new branch and bound enumeration method to provide an efficient solution procedure for the particular production planning problems considered in this thesis.

In Chapter 3 a further application of mathematical and computing techniques is made to long term production planning from a beach sand deposit which is to be mined by a dredge. A necessary part of such planning is the determination of broadly optimal dredge paths and we show how this may be formulated as a longest path problem in a network in which nodes represent blocks in the mine and links represent allowed interblock movements of the dredge. Even though we are dealing with a contrasting mining environment to that discussed in Chapter 2, it is again the spatial distribution of grades and tonnages in the deposit and the movement capabilities of the machines (dredge) which form a key part of the model.

Little work on the problem has appeared in the literature, but this is not surprising since it turns out to be a longest path problem in a network which has positive cycles. The general longest path problem with positive cycles, or, as it more commonly appears in the literature as a shortest route problem with negative cycles, has proven notoriously difficult to solve [34,3] although very efficient algorithms exist for shortest route problems with no negative cycles. For a detailed discussion of the relation between these problems and also the travelling salesman problem, see Kirby [23].

In Chapter 3 we develop an efficient heuristic technique for the mining network, based on partitioning it into smaller subareas and then optimising over these. In this way, sensible solutions can be obtained in a reasonable amount of computing time. In Chapter 4 we conclude the thesis by discussing the various results obtained.

As mentioned earlier, this thesis concentrates on the mathematical formulations behind the various mining problems considered and the OR techniques which may be used to obtain solutions. In his review Coyle [7] points out that papers in the literature generally tend to give minimal description of the problems being investigated and of the assumptions involved in the mathematical models. This thesis has endeavoured to pay greater attention to these aspects and it is hoped that a general improvement will be seen here as mathematical modelling comes to play an increasingly important role in mine planning.

CHAPTER 2

OPERATIONS RESEARCH TECHNIQUES FOR OPEN PIT MINE PRODUCTION PLANNING WITH GRADE CONSTRAINTS ON THE MILL FEED

2.1 GENERAL

As pointed out in Chapter 1 this chapter deals with the problem of mine-mill production planning when there are grade constraints on the mill feed. The discussion applies to shallow open pit mines. However we also assume that, if they consist of more than one level, then successive levels are developed one after the other. The situation in mind is typified by shallow laterite deposits which characteristically possess much larger variations in ore composition then what is required to keep the concentration plant running efficiently.

There are a number of ways of _physically_ increasing the capability of the system to provide acceptable mill feed. For instance, we can use large stockpiles of suitable ore, we can employ special bed and bin blending techniques or we can use several very mobile machines to extract the ore (front-end loaders for example). In addition, the more the combined capacity of these machines exceeds the required mill throughput the easier it will be to produce the right tonnage of acceptable feed.

As mentioned earlier, this chapter shows how mathematical programming can be used to rapidly determine optimal or near optimal production schedules for a given configuration of the mine-mill system. The mathematical and computer model may thus be thought to assume an important place on the "software" side of the system. However its influence extends further than this. Because it allows precise estimates to be obtained for the efficiency with which the various components of the system are used and because it enables the effect of alterations (e.g. to stockpile levels) to be readily evaluated, it can also have significant effect on the "hardware" side of the system.

2.2 MACHINE MOVEMENTS AND MEDIUM TERM PLANNING

We first consider the problem from the medium term viewpoint. Our main aim is to control the development of various subareas of the mine so as to minimise machine movements.[1]

(2.2.1) Mine Mill System

The precise way in which the problem is formulated mathematically will depend on how the system is designed and on the way in which mining is to proceed. We therefore introduce a typical, though hypothetical, mine-mill system and develop the approach in relation to it. The system is illustrated in Fig. 1 and relevant data are specified in Table 1. In particular the following assumptions are made about the system.

(i) The area of the mine to be worked during the planning period is cleared of overburden. Furthermore it is divided into blocks or small subareas and these are grouped together to form larger machine areas. All blocks in a particular machine area can be mined by one machine only.

---

[1]. The machines referred to are those engaged in ore extraction, not haulage units. The machine movements are interblock movements of these machines, not ore movement

(ii) Two types of equipment are employed to mine the ore, namely high volume but slow moving drag lines and lower volume but faster moving front-end loaders. We shall assume just one drag line and one front-end loader and also that sufficient haulage trucks are available to enable these machines to work at their full capacity except when prevented from doing so by grade constraints or plant capacities.

(iii) Grade control is important for only one constituent of the ore.

(iv) Ore may be sent direct to the mill, or, if its grade lies in a suitable range, it may be first stockpiled in a "high" or "low" grade stockpile and then used at a later date. These stockpiles are designed to receive ore which is respectively of a higher or lower in grade than that required by the mill. Fig. 1 illustrates the various ore flows which may thus occur within the system.

Lengthy dragline movements are particularly undesirable and quite a few features of the system are designed to reduce the need for them. Before contemplating a lengthy dragline movement we can first try and achieve grade control by,

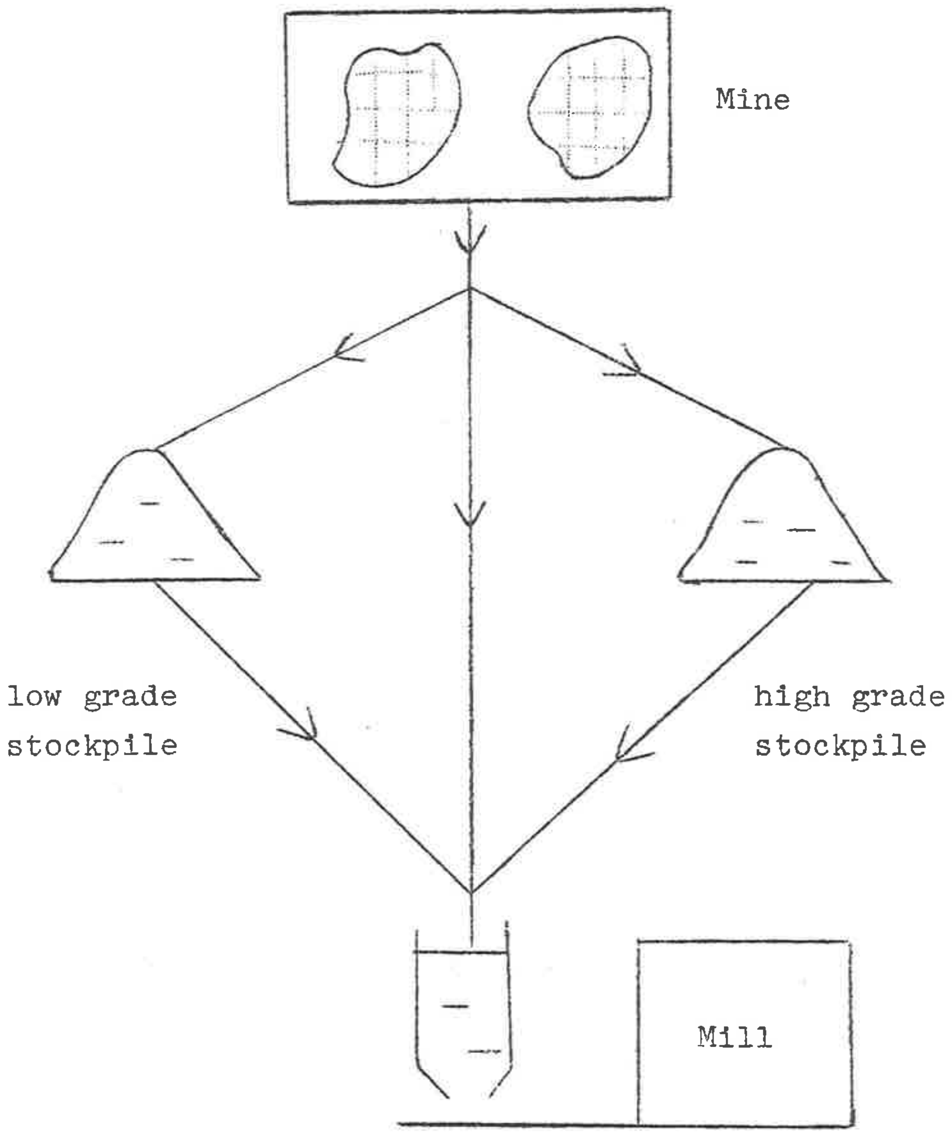


FIGURE 1

Mine-mill system. Ore is mined from two machine areas and may go either direct to the mill or via the low or high grade stockpile.

TABLE I. SPECIFICATIONS FOR THE MINE-MILL SYSTEM:

I MINE:

(i) Size of blocks = 200' × 200' (in plan)

(ii) Average depth ≈ 30' (variable)

(iii) Approximate range of block grades = 0.9%-1.1%

(iv) Bulk Density of ore in mine = 1.79

(v) Average block reserves ≈ 60,000 tons.

II MACHINES:

(i) Scheduled Hours:

No. (available) hours|day = 10 hours|day

No. days|week = 6 days|week

(ii) Machine Capacities:

DL[1] = 30,000 tons|week ≡ 500 tons|hour.

FEL[2] = 22,500 tons|week ≡ 375 tons|hour.

(iii) Minimum Extraction Rates:

DL = 9,000 tons|week

FEL = 6,750 tons|week

(iv) Total Maximum Extraction Rate = 52,500 tons|week

[1] DL = Dragline

[2] FEL = Front-end Loader

[continued on next page]

TABLE I (continued)

III MILL:

(i) Operating hours = 24 hours|day, 7 days|week (continuous operation)

(ii) Grade of feed in range: [0.93%,0.98%]

(iii) Capacity = 42,000 tons|week ≡ 250 tons|hour

(iv) Minimum allowable throughput rate = 31,500 tons|week ≡ 190 tons|hour

IV STOCKPILES:

(i) Low Grade Stockpile only accepts material with grade in range [0.90%,0.93%)

(ii) High Grade Stockpile only accepts material with grade in range (0.98%,1.01%]

(iii) Stockpile Capacities: variable e.g. 78,000 tons

V PLANNING HORIZON:

(i) Length of Planning Period ≈ 1-2 years (medium term).

(i) blending ore which comes direct from the mine with ore from the high or low grade stockpile, or

(ii) reducing the mill throughput slightly, or

(iii) moving the front-end loader to a more suitable block.

If these measures fail larger dragline movements to suitable blocks are unavoidable.

(2.2.2) Blending Intervals

In addition to the above assumptions regarding the mine-mill system, further assumptions are made about the way in which mining is to proceed. In particular we suppose that, whenever the machines start mining a new set of blocks, no machine movements are allowed until at least one machine mines out its block. At this point of time, at least one machine has to move. The ideal solution is, of course, to shift such machines from their current blocks to one of the nearest neighbours. Unfavourable grades may prevent us from employing this strategy and this is what gives rise to the lengthier machine movements.

We shall call an interval of time, which begins with the mining of a new set of blocks and ends when a machine movement takes place, a blending interval. These blending intervals, which are not necessarily of equal duration, will typically last for one or two weeks. Mathematically they are a convenient way of partitioning the whole planning horizon.[1] As suggested by their name, we assume that during any blending interval, each machine mines ore of a fixed grade, namely, the grade of the block in which the machine is located. Given the blocks to be mined during any blending interval, we merely have to determine ore flow rates which will ensure that the average grade of the mill feed is within the prescribed limits. Conversely, at the end of the blending interval, we must reassign the machines to a new set of blocks from which it will be possible to form suitable mill feed during the next blending interval. The essence of the scheduling problem is to perform these two tasks optimally.

(2.2.3) Mathematical Model

We now formulate a network flow model for the medium term mine-mill production scheduling problem, taking into account both machine movements and mining constraints. The formulation is based on the use of a dynamic network [12] to describe all possible ore flows and machine movements. Specifications of the network which is illustrated in Fig.2 are as follows. Apart from special origin and

---

[1] We simply punctuate the time axis at the instants at which changes (in the form of machine movements) occur. The interval between any consecutive pair of such points on the time axis is a blending interval.

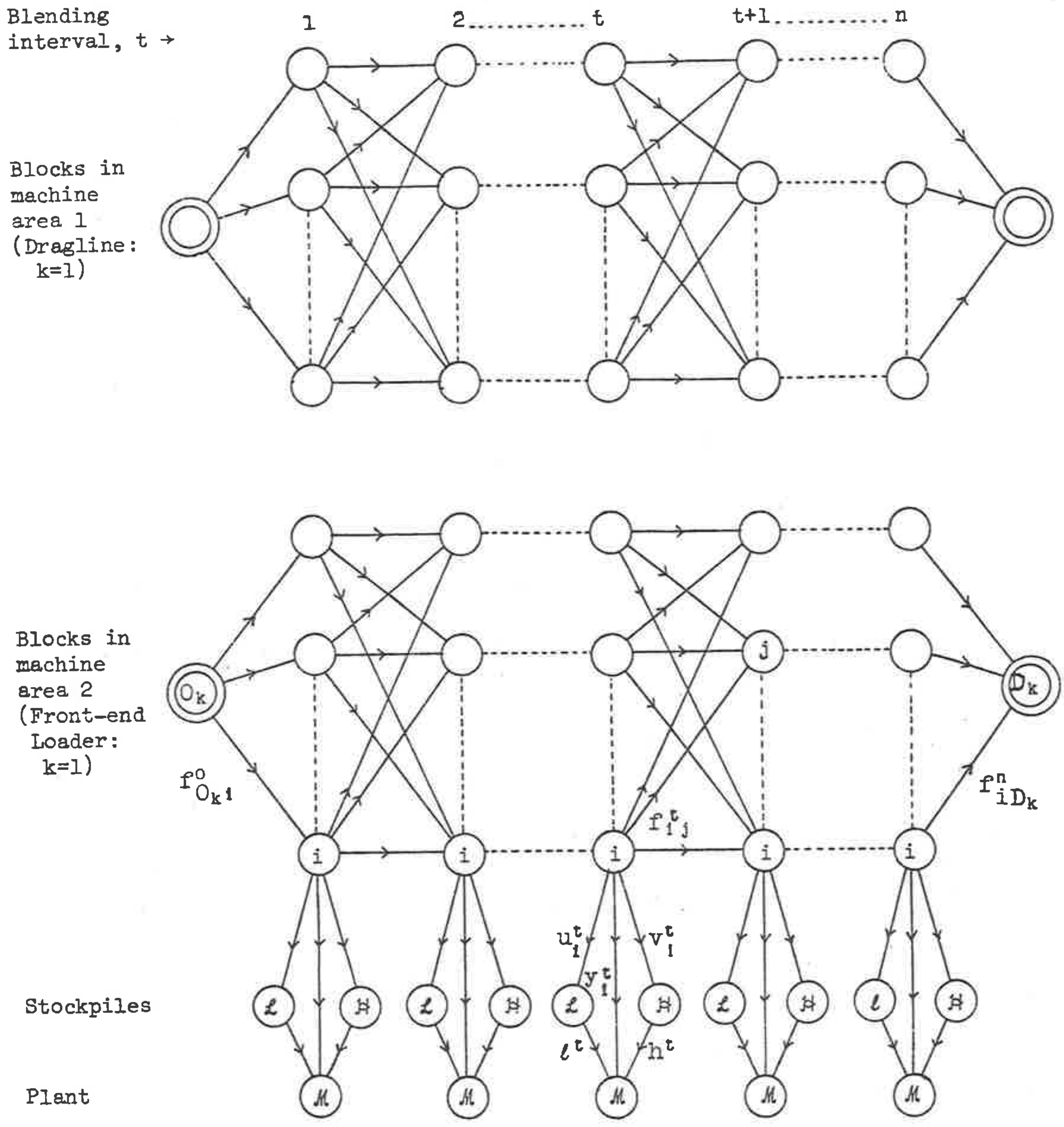


FIGURE 2

Dynamic network model of mine-mill system: Nodes represent blocks in the mine and also plant and stockpiles. Links represent allowed interblock machine movements and also ore flows within the system. Note that not all links have been shown.

destination nodes, it has nodes to represent the mill, the low and high grade stockpiles and every block in the mine. By repeating these nodes n times we can model all operations of the system over a total time horizon which spans n separate blending intervals.

Let (i,t), $(\mathcal{L},t)$, $(\mathcal{H},t)$ and $(\mathcal{M},t)$ denote the nodes representing the block i, the low and high grade stockpiles and the mill during blending interval t. A link from node (i,t) to node $(j,t+1)$ is then denoted by $(i,j;t)$ and allows for a machine movement from block i to block j at the <u>end</u> of period t.[1] Ore flows on the other hand take place <u>during</u> any period t, and along links $(i,\mathcal{M};t)$, $(i,\mathcal{L};t)$, $(i,\mathcal{H};t)$, $(\mathcal{L},\mathcal{M};t)$ and $(\mathcal{H},\mathcal{M};t)$. Finally there are the special origin and destination nodes (denoted by O_k and D_k) for the "flow" (0-1 variables) of each machine k. Links are provided from node O_k to node $(i,1)$ and from node (i,n) to node D_k, where i ranges over all blocks in given machine area k (see Fig.2).

We now introduce the following quantities.

(a) <u>Variables</u>

y_i^t = ore flow rate (e.g. tons|week), during blending interval t, from block i <u>directly</u> to the mill,

---

[1] $j=i$ is permitted since a machine may continue mining block i during the next time interval (if there is ore left in it).

u_i^t, v_i^t = ore flow rates from block i to the low, high grade stockpiles,

ℓ^t, h^t = ore flow rates from the low, high grade stockpiles to the mill,

x_i^t = total ore flow rates from block i,

$Y_k^t, U_k^t, V_k^t, X_k^t$ = total of the ore flow rates $y_i^t, u_i^t, v_i^t, x_i^t$ summed over <u>all</u> blocks in each machine area k,

T^t = duration (in weeks) of bending interval t,

L^t, H^t = low, high grade stockpile levels at the <u>end</u> of blending interval t, (the initial levels L^0, H^0 are given quantities rather than variables),

R_i^t = reserves of block i at the <u>beginning</u> of blending interval t, (the initial reserves R_i^1 are given and are also denoted by $R_i = R_i^1$),

r_k^t = reserves, at the <u>beginning</u> of blending interval t, of the block to be mined by machine k <u>during</u> blending interval t,

$f^o_{O_k i}, f^t_{ij}, f^n_{iD_k}$ = "flow" (0-1 variable) of machine k along the links $(O_k, i; 0)$, $(i, j; t)$, $(i, D_k; n)$.

(b) <u>Given quantities</u>

a, b = lower, upper bounds (in tons|week) on the mill throughput rate,

a_k, b_k = lower, upper mining capacities (in tons|week) for machine k,

C^{ℓ}, C^{h} = low, high grade stockpile capacities,

g_i = grade of block i,

G_1^M, G_2^M = lower, upper grade limits for the mill feed,

g_{ℓ}, g_h = average grade of material drawn from the low, high grade stockpiles,

G_1^L, G_2^L = lower, upper grade limits for ore which is sent to the low grade stockpile,

G_1^H, G_2^H = lower, upper grade limits of ore which is sent to the high grade stockpile,

α_k = net profit (in \$|ton) accruing from each ton of ore mined by machine k and sent directly to the mill,

γ_k = mining, transportation and handling cost (\$|ton) for each ton of ore mined by machine k and sent to the low or high grade stockpile,

β = net profit (\$|ton) accruing from each ton of ore which comes to the mill from the low or high grade stockpile,

c_{ij}^{k} = cost (in \$) of moving machine k from block i to block j,

A = penalty (in \$) for starting a <u>new</u> face in an enclosed block,

K = minimum reserves that any block must have before a machine movement to that block is allowed,

S_k = set of blocks in machine area k,

$\bar{S}_k$ = set of blocks in machine area k which do not possess a mine face at the <u>start</u> of operations, (i.e. at the beginning of blending interval 1),

$\mathcal{N}_i$ = the set of nearest neighbours of any block i.

Also we define $s(x;a)$ to be the following Heaviside unit function of a single variable x and with parameter a;

$$s(x;a) = \begin{cases} 0, & x < a \\ 1, & x \geq a. \end{cases}$$

Its graph is shown in Fig.3.

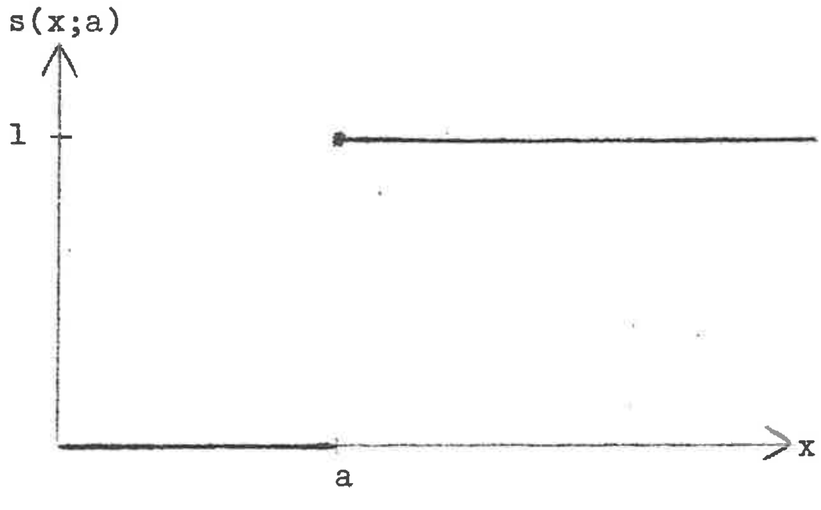


FIGURE 3

Graph of the unit function $s(x;a)$

The problem of finding the optimal production schedule over n blending intervals is now formulated as the mathematical program,

$$\text{Maximise} \sum_{t=1}^{n} T^t \left\{ \sum_{k} \left(\alpha_k \, y_k^t - \gamma_k \, (u_k^t + v_k^t) \right) + \beta \, (\ell^t + h^t) \right\} \cdots$$

$$\cdots - \sum_{t=1}^{n} \sum_{k} \sum_{i \in S_k} \sum_{j \in S_k} c_{ij}^k \, f_{ij}^t \cdots$$

$$\cdots - A \sum_{k} \sum_{i \in \bar{S}_k} f_{0_k i}^0 \cdots$$

$$\cdots - A \sum_{t=1}^{n-1} \sum_{k} \sum_{\ell \in S_k} \sum_{i \in \bar{S}_k} f_{\ell i}^t \, s(R_i^{t+1}; R_i) \prod_{j \in N_i} s(R_j^{t+1}; K) \tag{1}$$

subject to

$$x_i^t = y_i^t + u_i^t + v_i^t \tag{2}$$

$$Y_k^t = \sum_{i \in S_k} y_i^t \tag{3}$$

$$U_k^t = \sum_{i \in S_k} u_i^t \tag{4}$$

$$V_k^t = \sum_{i \in S_k} v_i^t \tag{5}$$

$$X_k^t = \sum_{i \in \ddot{S}_k} x_i^t \tag{6}$$

$$x_i^1 \leqslant b_k \, f_{0_k i}^0, \quad i \in S_k \tag{7}$$

$$x_i^t \leqslant b_k \sum_{j \in S_k} f_{ji}^{t-1}, \quad 2 \leqslant t \leqslant n, \; i \in S_k \tag{8}$$

$$X_k^t \geqslant a_k \tag{9}$$

$$a \leqslant \sum_k Y_k^t + \ell^t + h^t \leqslant b \tag{10}$$

$$G_1^M \leqslant \frac{\sum_i g_i y_i^t + g_\ell \ell^t + g_h h^t}{\sum_i y_i^t + \ell^t + h^t} \leqslant G_2^M \tag{11}$$

$$T^t = \operatorname*{Min}_k \left\{ \frac{r_k^t}{X_k^t} \right\} \tag{12}$$

$$L^t = L^{t-1} + T^t\left(\sum_k U_k^t - \ell^t\right), \; 1 \leqslant t \leqslant n, \tag{13}$$

$$H^t = H^{t-1} + T^t\left(\sum_k V_k^t - h^t\right), \; 1 \leqslant t \leqslant n, \tag{14}$$

$$0 \leqslant L^t \leqslant C_\ell \tag{15}$$

$$0 \leqslant H^t \leqslant C_h \tag{16}$$

$$\sum_{i \in S_k} f^o_{O_k i} = 1 \tag{17}$$

$$f^o_{O_k i} = \sum_{j \in S_k} f^1_{ij}, \quad t = 1 \tag{18}$$

$$\sum_{\ell \in S_k} f^{t-1}_{\ell i} = \sum_{j \in S_k} f^t_{ij}, \quad 2 \leqslant t \leqslant n-1 \tag{19}$$

$$\sum_{j \in S_k} f^{n-1}_{ji} = f^n_{iD_k}, \quad t = n \tag{20}$$

$$\sum_{i \in S_k} f^n_{iD_k} = 1 \tag{21}$$

$$R_i^t = R_i^{t-1} - T^{t-1}\, x_i^{t-1}, \quad 2 \leqslant t \leqslant n \tag{22}$$

$$r_k^1 = \sum_{i \in S_k} f^o_{O_k i}\, R_i, \quad t = 1 \tag{23}$$

$$r_k^t = \sum_{j,i \in S_k} f_{ji}^{t-1} R_i^t, \quad 2 \leqslant t \leqslant n \tag{24}$$

$$\sum_{j \in S_k} f_{ji}^t \leqslant \frac{R_i^{t+1}}{K}, \quad 1 \leqslant t \leqslant n-1 \tag{25}$$

$$(g_i - G_1^L)(G_2^L - g_i)\ u_i^t \geqslant 0 \tag{26}$$

$$(g_i - G_1^H)(G_2^H - g_i)\ v_i^t \geqslant 0 \tag{27}$$

$$f_{ij}^t,\ f_{O_k i}^o,\ f_{iD_k}^n = 0,1 \tag{28}$$

$$y_i^t,\ u_i^t,\ v_i^t \geqslant 0 \tag{29}$$

The model, which takes into account all possible ore flows and machine movements over the n blending intervals, will now be discussed more fully. Apart from the 0-1 variables $f_{O_k i}^o$, f_{ij}^t, $f_{iD_k}^n$, nonlinearities are involved in constraints (12) - (14) and (22) - (24) and the objective function (1).

Constraints (17) - (21) are the conservation equations for the "flow", f_{ij}^t, of each machine. A value of one for f_{ij}^t, indicates a machine movement from block i to block j at the end of period t. For any feasible solution of the problem (1) - (29) we can trace the path taken by each machine k, by following the links in machine area k, for which the variables $f_{O_k i}^o$, f_{ij}^t $(1 \leqslant t \leqslant n-1)$ and $f_{jD_k}^n$ have the value one.

Constraints (7) and (8) ensure that ore production is confined to the blocks on each machine path. Machine k, for example, mines block i during period t if and only if $\sum_{j \in S_k} f_{ji}^{t-1} = 1$, in which case the total ore extraction rate, x_i^t, is limited to the capacity of machine k, namely, b_k tons/week. If machine k does not mine block i during period t, $\sum_{j \in S_k} f_{ji}^{t-1}$ will automatically be zero, and hence so will x_i^t. Thus, during each time interval t, exactly one x_i^t will be greater than zero in each machine area.

It should also be noted that when $i \in S_k$, we have placed an upper bound of b_k tons/week on the ore flow rate x_i^t. The capacity thus depends only on the appropriate machine k and not on the location of the block i. This is a consequence of the assumption that there are sufficient haulage trucks to enable each machine to work at its maximum rate except when prevented from so doing by grade constraints or plant capacities. A more general formulation would allow capacities b_{ik} which also depend on the block i (through its location etc.).

The constraints (11) represent the grade constraints on the mill feed and although they appear nonlinear here they may easily be converted into two linear constraints. Constraints (26) - (27) express further grade requirements for the high and low grade stockpiles - in particular they ensure that they will only accept ore which has a grade in the ranges $[G_1^H, G_2^H]$ and $[G_1^L, G_2^L]$ respectively. Moreover, we have tacitly assumed that ore will be drawn from these stockpiles at a fixed average grade of g_h and g_ℓ respectively $\left(\text{e.g. } g_h = \frac{G_1^H + G_2^H}{2},\ g_\ell = \frac{G_1^L + G_2^L}{2}\right)$. This assumption has been made to simplify the overall model, but is relaxed when we come to construct locally optimal schedules using the heuristic algorithm of §2.2.4. In any case it is not a very serious assumption when only one constituent of the ore requires grade control. If more than one ore constituent requires grade control, it is easy enough to accommodate further grade constraints such as (11). However it soon becomes impractical to provide further stockpiles for all possible "low" and "high" grade combinations. We might only be able to provide stockpiles to help blend one or two of the more important constituents of the ore, and we would then have to keep closer track of the composition of these stockpiles, especially as regards the other constituents.

Unlike most multiperiod models, we do not know, a priori, the duration of each time interval. These are computed individually via the constraints (12). During any period t, each machine k mines ore at a rate of X_k^t tons/week from a block whose reserves are r_k^t tons. Constraint (12) thus expresses the fact that the period concludes as soon as a machine mines out its block. Machine movements at the end of period t are then governed by constraint (25) which ensures that machine k can only move to a block i whose reserves are at least K tons. In the computer program written to implement the heuristic algorithm of §2.2.4, a value of $K>0$ was used to prevent machine movements to blocks which have insignificantly small (i.e.$< K$) amounts of ore left in them.

It is often more desirable to plan operations over a fixed <u>time</u> horizon (e.g. 1-2 years) rather than over a fixed number of blending intervals whose total duration is variable. While it is certainly possible to modify the overall model to achieve this, it can be done much easier when using the algorithm of §2.2.4 and will be discussed more fully then.

The first term in the objective function measures a total profit earnt from the concentrate produced. The value of this profit is determined by the ore flows which take place within the system. The handling charges γ_k present in this term discourage unnecessary stockpiling activities. Also note that we tacitly assume that revenue earnt accrues according to the mill throughput. The second term in the objective function represents the costs of interblock machine movements. Again we assume that monetary values can be assigned to the coefficients c^k_{ij} to take into account factors, such as the location of blocks i and j and the mobility of machine k, which influence the relative desirability of the various machine movements.

The remaining two terms in the objective function compute additional penalties for opening up new faces as a result of shifting machines to blocks which are enclosed by others. Such mining requirements are considered more fully in §2.2.4 and section 2.3. For the moment, we shall merely mention that an extra penalty A is computed if, at the end of any period t, any machine is shifted to mine a new block i for which

(a) $R_i^{t+1} = R_i$ (i.e. mining has not commenced in block i prior to period t+1)

and

(b) $R_j^{t+1} \geqslant K$ for each of the nearest neighbours j of block i (i.e. each $j \in \mathcal{N}_i$).

If the latter requirement holds, that is there is a significant tonnage left in each of nearest neighbouring blocks of block i, we assume that block i is still enclosed by these blocks and that a new face must be commenced in it before mining may proceed. This is illustrated in Fig.4.

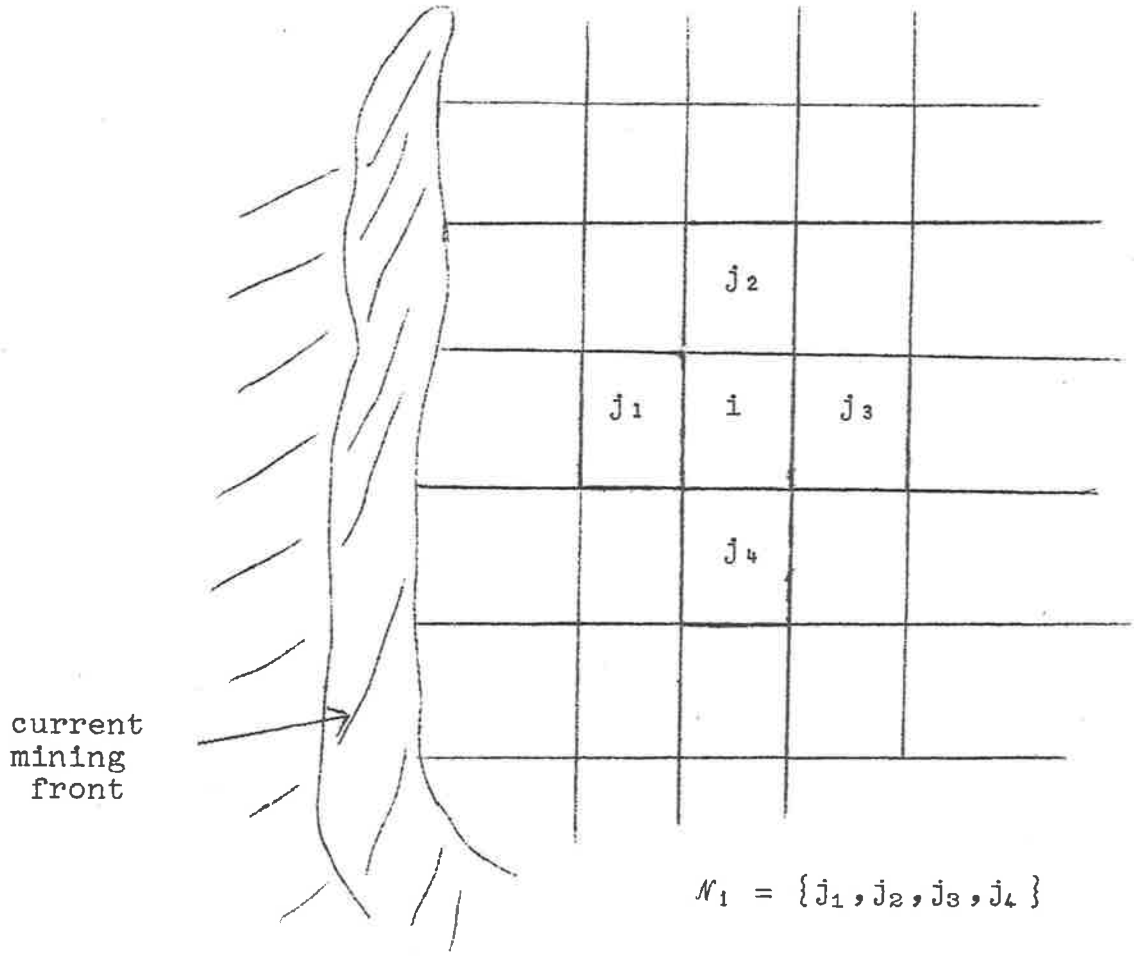


FIGURE 4

A new face must be commenced in order to mine blocks such as block i which are not intersected by the current mining front. We assume this to be the case if none of its nearest neighbours is mined out.

(2.2.4) Heuristic Algorithm

As may be expected, the nonlinear mathematical program (1) - (29) is not amenable to direct solution. We shall therefore determine local optima, constructing them step by step, from period to period. We proceed as follows. Suppose we have already determined operations over blending intervals $1,2,\ldots,t-1$ and have also decided which blocks will be mined during blending interval t. Let i_k denote the block so chosen to be mined by machine k during blending interval t. We now determine the optimal ore flow rates Y_k^t, U_k^t, V_k^t, ℓ^t and h^t subject to satisfying the grade, capacity and production requirements for blending interval t only. When considering the problem from this local viewpoint, it is necessary to treat the objective function in a somewhat different light than before. We discuss this more fully later, but mention for the moment that, in general, we now choose to maximise that component of the mill throughput which comes direct from the mine and to employ any excess mining capacity towards building up stockpiles.

The mathematical program used for determining locally optimal values of the above ore flows is obtained by including only those variables and constraints of the overall model (1) - (29) which relate to blending interval t. For clarity, we give full details of the resulting program - it is

$$\text{Maximise} \quad \sum_k Y_k^t + \varepsilon_1 \sum_k U_k^t + \varepsilon_2 \sum_k V_k^t + \varepsilon_3 \ell^t + \varepsilon_4 h^t \qquad (30)$$

subject to

$$a_k \leqslant Y_k^t + U_k^t + V_k^t \leqslant b_k \qquad (31)$$

$$a \leqslant \sum_k Y_k^t + \ell^t + h^t \leqslant b \qquad (32)$$

$$G_1^M \leqslant \frac{\sum_k g_{1k} Y_k^t + g_\ell \ell^t + g_h h^t}{\sum_k Y_k^t + \ell^t + h^t} \leqslant G_2^M \qquad (33)$$

$$T^t = \min_k \left\{ \frac{r_k^t}{Y_k^t + U_k^t + V_k^t} \right\} \qquad (34)$$

$$0 \leqslant L^{t-1} + T^t\left(\sum_k U_k^t - \ell^t\right) \leqslant C_\ell \qquad (35)$$

$$0 \leqslant H^{t-1} + T^t\left(\sum_k V_k^t - h^t\right) \leqslant C_h \qquad (36)$$

$$(g_{1k} - G_1^L)(G_2^L - g_{1k})U_k^t \geqslant 0 \qquad (37)$$

$$(g_{1k} - G_1^H)(G_2^H - g_{1k})V_k^t \geqslant 0 \qquad (38)$$

$$Y_k^t,\ U_k^t,\ V_k^t \geqslant 0. \qquad (39)$$

Since we have already determined operations for periods $1,2,\ldots,t-1$, the quantities L^{t-1}, H^{t-1} and r_k^t (and g_ℓ, g_h) will now be known. (Note that $r_k^t = R_k^t$.) The only variables are therefore the Y_k^t, U_k^t, V_k^t, ℓ^t, h^t and T^t. The mathematical program (30) - (39) is still nonlinear since it is necessary to keep track of total ore flows within the system during blending interval t, whereas the basic variables Y_k^t etc.

represent ore flow **rates**. However, the nonlinearities are not of a very serious nature and in fact it is possible to solve the mathematical program by solving m linear programs, where m is the number of machines. We obtain the linear program corresponding to machine p, $(p=1,\ldots,m)$, by assuming that it is the first one to mine out its block, that is,

$$T^t = \min_k \frac{r_k^t}{Y_k^t + U_k^t + V_k^t} = \frac{r_p^t}{Y_p^t + U_p^t + V_p^t} . \tag{40}$$

To ensure this we add the constraints

$$\frac{r_p^t}{Y_p^t + U_p^t + V_p^t} \leqslant \frac{r_k^t}{Y_k^t + U_k^t + V_k^t}, \quad k \neq p$$

i.e.

$$r_p^t(Y_k^t + U_k^t + V_k^t) \leqslant r_k^t(Y_p^t + U_p^t + V_p^t), \quad k \neq p. \tag{41}$$

Substituting for T^t of (40) into constraints (35) and (36) yields linear constraints, namely

$$0 \leqslant L^{t-1}(Y_p^t + U_p^t + V_p^t) + r_p^t(\sum_k U_k^t - \ell^t) \leqslant C_\ell(Y_p^t + U_p^t + V_p^t) \tag{42}$$

and

$$0 \leqslant H^{t-1}(Y_p^t + U_p^t + V_p^t) + r_p^t(\sum_k V_k^t - h^t) \leqslant C_h(Y_p^t + U_p^t + V_p^t) \tag{43}$$

The final linear program corresponding to machine p consists of relations (30) - (33), (37) - (39) and (41) - (43).

The nonlinear program (30) - (39) has feasible solutions if and only if at least one of its m associated linear programs has feasible solutions, in which case its optimum is found by selecting the best optimal solution obtained from the various linear programs.[1]

Having determined the ore flows during blending interval t, we update the status of the mine-mill system as a result of the operations that will take place during this period. We then decide which blocks will be mined during the next blending interval, t+1. Of course, the ideal solution is to move machines which have just completed mining their blocks to one of the nearest neighbours. Grade constraints may prevent us from employing this strategy however. In fact machine movements are only allowed to blocks whose grades are such that the mathematical program (30)-(39) possesses feasible solutions.[2] Such movements will be called feasible movements.

If each machine k moves from block i_k to block j_k, at the end of period t, ($j_k = i_k$ is possible), the

---

1. (a) The mathematical program (30)-(39) and its associated linear programs do not have unbounded feasible solutions.
   (b) Not all of the associated linear programs need to have feasible solutions.

2. When the g_i^k are replaced by the grades of the blocks under consideration. (Also t will be replaced by $t+1$).

total movement costs incurred are

$$\sum_{k} \left[c^{k}_{i_k j_k} + A\, s\left(R^{t+1}_{j_k};R_{j_k}\right) \prod_{\ell \in \mathcal{N}_{j_k}} s\left(R^{t+1}_{\ell};K\right) \right] \qquad (44)$$

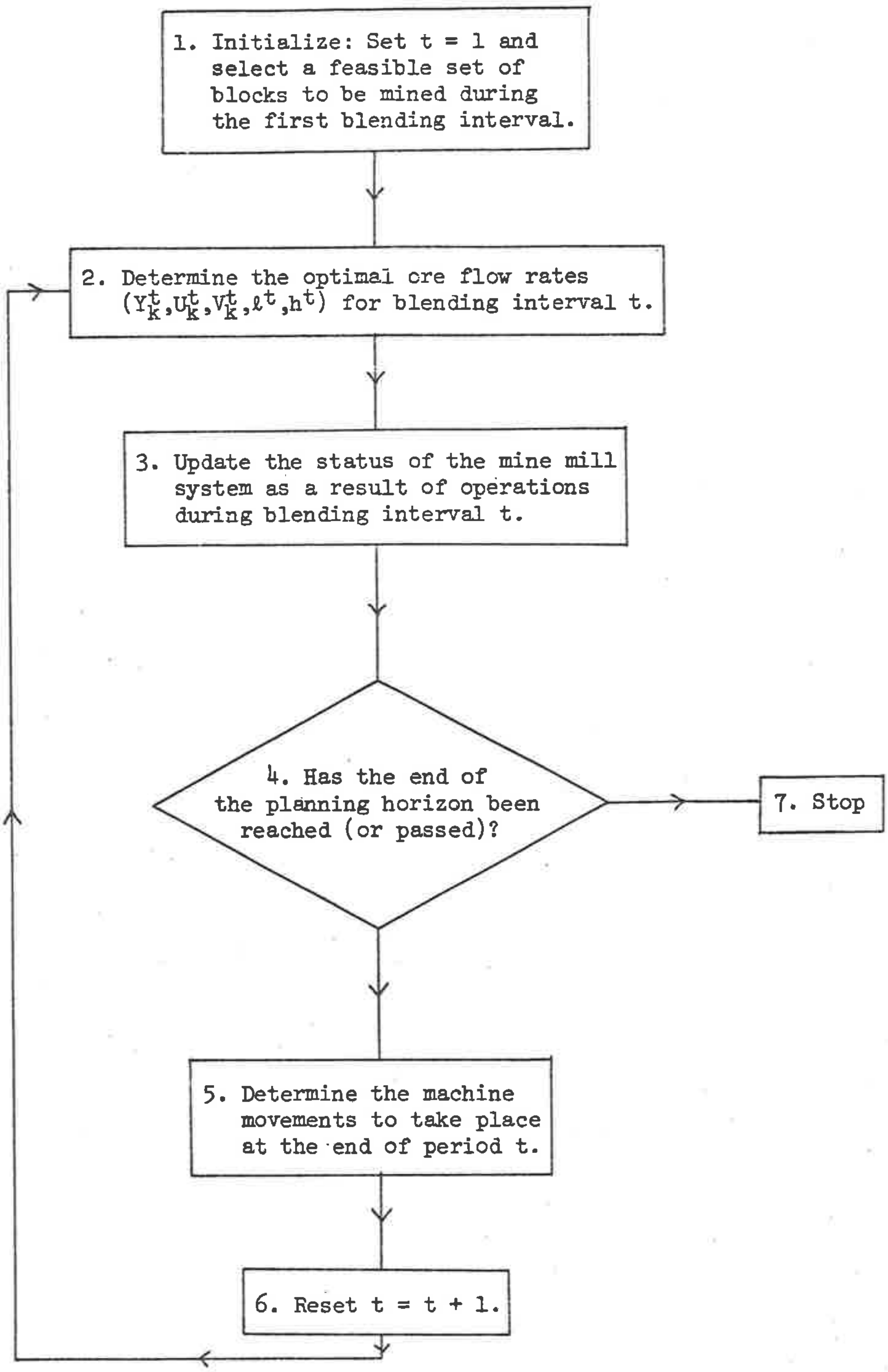
where, as before, the second term in the summand includes a penalty of A dollars for moving machine k to a block j_k which is not yet "available", i.e. it does not yet possess a mining face. The strategy used in the heuristic algorithm is to choose the machine movements from amongst all feasible movements so as to minimise the quantity (44). In the computer program, described in §2.2.5, this was done by brute force enumeration of all possibilities, starting first with movements to nearest neighbours and proceeding further away if unsuccessful. Each possibility considered requires a test to be made to see whether the mathematical program (30) - (39), (with the appropriate grade coefficients), possesses feasible solutions. The objective function at this stage consists of the sum of artificial variables, and, in order that the machine movements be feasible, it must be possible to drive this sum to zero in at least one of the associated linear programs. Optimal solutions to the mathematical program (30) - (39) are not sought until the final selection of blocks to be mined in period $t+1$ is made, in which case the whole process is repeated.

A flowchart of the algorithm is given in Fig.5 while Fig.6 illustrates a complete schedule for an example consisting of 2 machines 8 blocks and 3 blending intervals.

The coefficients $\varepsilon_1 - \varepsilon_4$ in the objective function (30) may be varied from period to period in order to set priorities for adding to or for drawing from the stockpiles as compared with sending ore directly to the mill. From the local viewpoint, there is no incentive to stockpile ore at all; it is carried out more to help create a buffer against large machine movements in the future or to assist in raising mill production to higher levels than might be attainable from the mine only. Machine movements to near-by blocks which have difficult grades, may be feasible if the stockpile levels are high, but infeasible if they are low. It will therefore be desirable to increase the priority given to building up the stockpile levels when they are low. In the overall model (30) - (39) stock-piling would automatically be carried out only to the extent where absolutely necessary.

(2.2.5) Application

A computer program has been written to construct schedules by the algorithm of §2.2.4. The results of five computer runs for the mine-mill system specified in Table 1 and §2.2.1 will now be summarized. The block grades and



<u>FIGURE 5</u>

Flow chart of Heuristic Algorithm

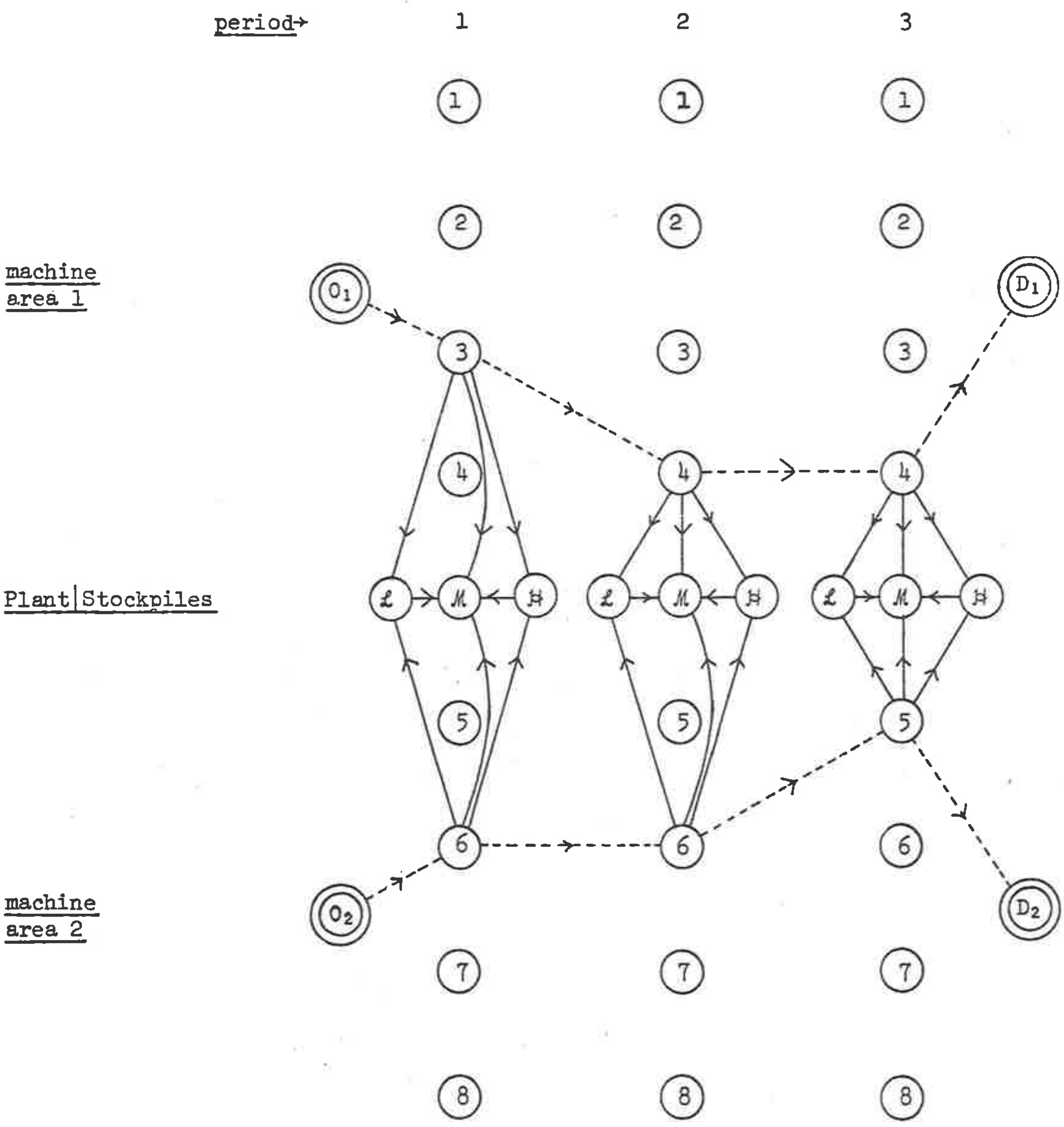


FIGURE 6

Illustration of a complete 3-period schedule for a system consisting of 8 blocks and 2 machines. Broken lines indicate machine movements, solid lines indicate ore flows (some may be zero).

tonnages were generated by random sampling from various statistical distributions and are listed in Appendix I. The probability density functions for these distributions are illustrated in Fig.7. The acceptable ranges of grade for the mill, low and high grade stockpiles were set at 0.93 - 0.98%, 0.90 - 0.93% and 0.98 - 1.01% respectively.

The blocks, which are equidimensional in plan, will now be referred to by their coordinates (i,j) with respect to some square grid system through the deposit. The cost of moving machine k from block (i_1,j_1) to block (i_2,j_2) was computed as

$$c^k_{(i_1,j_1),(i_2,j_2)} = W_k \; F_k(i_2,j_2)(|i_2-i_1| + |j_2-j_1|)$$

where

W_k = weight factor depending on the mobility of machine k. High values (w=20) were used for the slow moving draglines, lower values (w=1) for the front-end loader.

$F_k(i_2,j_2)$= factors, listed in Appendix I, and designed to encourage the machines to move along certain <u>preferred</u> paths unless prevented from so doing by the grade constraints.

The schedules were prepared for an horizon whose total duration was fixed at 100 weeks (2 years). It is usually found that the last blending interval finishes

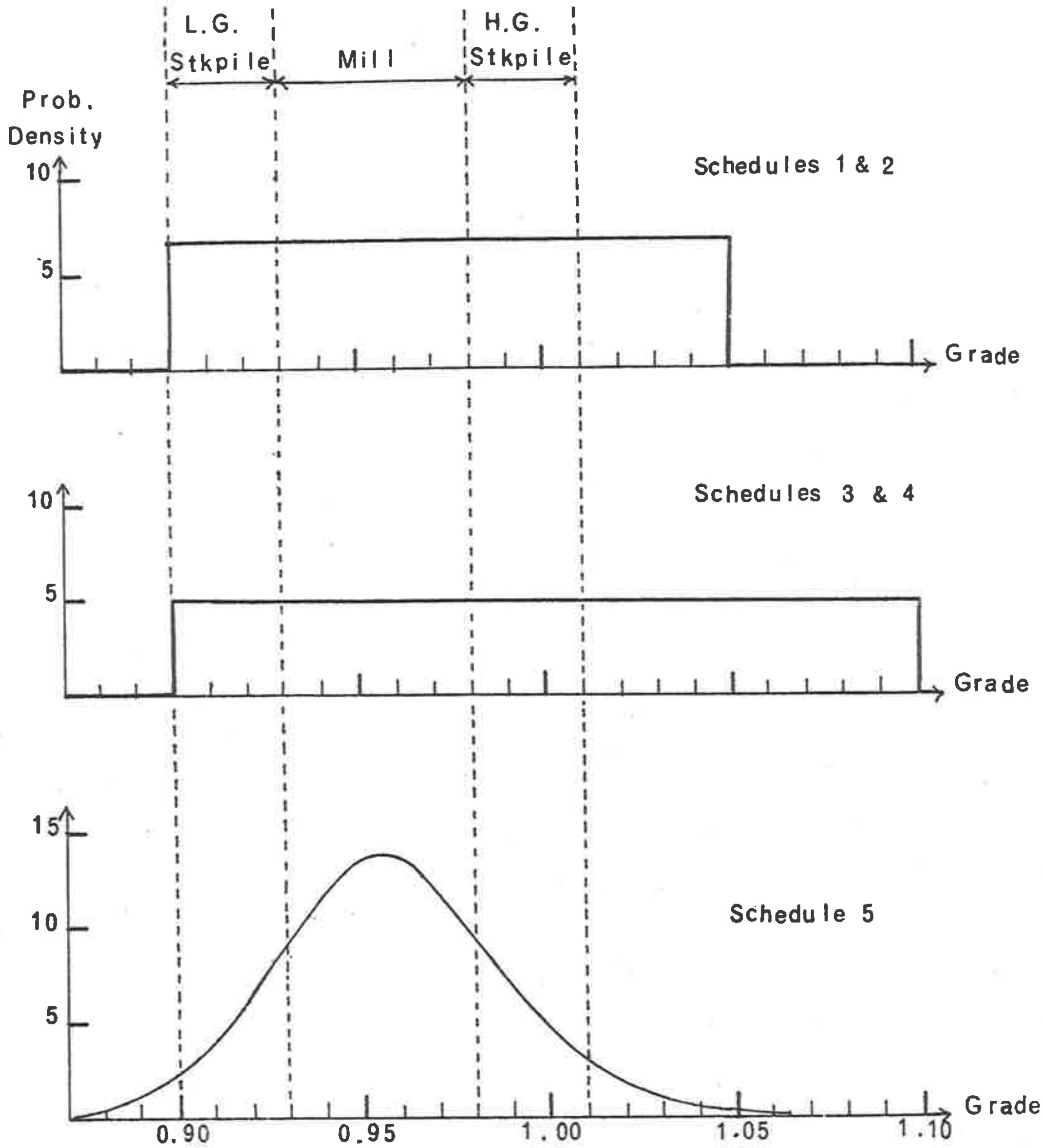


FIGURE 7

Distributions from which block grades were chosen. Also shown are the acceptable ranges of grade for the mill and the stockpiles.

after the conclusion of planning horizons of fixed duration, hence the need for the generalized stopping condition expressed in step 4 of the algorithm (see Fig.5). When this occurs, operations are adjusted to finish exactly at the time desired.[1]

The computer program may be used to investigate variations in

(i) block grades and tonnages
(ii) stockpile capacities
(iii) initial stockpile levels
(iv) mining capacities of the machines
(v) number and types of machines
(vi) grade requirements of plant
(vii) length of planning horizon.

The results presented now deal mainly with (i)-(iii). The relevant input data is given in Table 2, while the corresponding results appear in Table 3. The actual paths taken by the dragline are illustrated in Appendix I.

Schedules 1 and 2 demonstrate clearly how the various features of the system come into play to avoid lengthy dragline movements. In schedule 1 this has been achieved mainly through the buffering effect of the large stockpiles. (There was only one brief stockout). Smaller

[1] The same ore flow rates are used, but for a shorter period of time than the full blending interval.

TABLE 2: INPUT DATA.

	SCHEDULE				
	1	2	3	4	5
Block Grade and Tonnage Data Set[1]	I	I	II	II	III
Stockpile Capacities[2]					
Low Grade	78	18	78	30	0
High Grade	18	18	18	18	0
Initial Stockpile levels					
Low Grade	78	18	78	30	0
High Grade	18	18	18	18	0

[1] see appendix 1.

[2] in units of 1000's tons.

TABLE 3: SUMMARY OF RESULTS

	SCHEDULE 1	2	3	4	5
Number of Periods for 100 weeks of operation	77	68	68	67	57
Cumulative Distances Moved					
D.L.	42	47	66	80	47
F.E.L.	34	72	49	62	34
Total	76	119	115	142	81
D.L. Distance / F.E.L. Distance	1.2	0.7	1.3	1.3	1.4
No. of Machine Movements of more than 1 block					
D.L.	0	3	9	15	0
F.E.L.	1	13	8	14	5
Total	1	16	17	29	5
No. of New Faces Opened up					
D.L.	0	0	0	0	0
F.E.L.	0	0	0	4	0
Machine Efficiencies (%)					
D.L.	78.2	77.8	76.4	73.7	91.8
F.E.L.	78.8	79.6	79.7	80.7	63.4
Total	78.4	78.6	77.8	76.7	79.6
Mill Efficiency (%)	99.5	97.9	98.6	96.1	99.5
Tons Drawn from Stockpiles (as a % of total Mill Throughput)					
Low Grade Stockpile	8.7	2.7	6.5	3.9	0
High Grade Stockpile	0.5	1.7	0.6	1.0	0
Mean Stockpile Levels [1000's tons]					
Low Grade Stockpile	33	9	18	18	0
High Grade Stockpile	16	17	17	18	0

stockpiles were permitted in schedule 2 and greater reliance has been placed on the mobility of the front-end loader. Stockouts were more frequent and there was generally less material available to offset lengthier front-end loader movements. Note that the ratio of D.L. distance/F.E.L. distance was 0.7 whereas, in the absence of grade constraints, we would expect it to be $\frac{30.0}{22.5} \approx 1.3$, based on the relative capacities of the machines.

In schedules 3 and 4, the more difficult block grades have made lengthy dragline movements unavoidable. Schedule 4 would hardly be acceptable, with large dragline movements being frequently required (see Appendix I) after the low grade stockpile (assumed to be initially full) became depleted. In contrast schedule 5 demonstrates the ease with which the heuristic algorithm produces good schedules when the grades are favourable; dragline movements were exactly as desired a priori. As the block grades deviate further from the mill limits, there are two stages of breakdown; the first occurs when acceptable schedules exist, but the heuristic algorithm has difficulty in constructing them due to the fact that it only looks one period ahead. With still worse grades, it becomes impossible for the mine-mill system to produce the mill feed asked for and still keep machine movements at a reasonable level. Alterations to the system itself are then required for the deposit to be mineable.

In each of the above schedules, the priority coefficients $(\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4)$ of the objective function (30) were set at (0.1,0.01,0,0) throughout the planning horizon. Highest priority has therefore been given to maximising the mill throughput which comes direct from the mine; next we give priority to building up the low grade stockpile, since in all schedules, except schedule 5, there is an over-abundance of high grade blocks in the deposit (see Fig.7). These priority settings explain why the mill efficiency has been so high in each case (see Table 3). In schedules 3 and 4, this has been at the expense of larger dragline movements and, depending on the relative values of the costs (which appear in the objective function (30) of the overall model), it may be desirable to give greater priority to building up the low grade stockpile. In this case we would profit by resetting $\varepsilon_2 > 1$ and $\varepsilon_3 < 0$, especially when the stockpile level is low.

2.3 MINING CONSTRAINTS AND SHORT TERM PLANNING

We now concentrate on mining constraints in short term (e.g. day to day) production planning. As mentioned earlier, the work is an extension of Fraser's [13] in which we apply branch and bound techniques to determine globally optimal solutions to blending problems upon which the non-linear mining constraints are superimposed.

(2.3.1) Formulation

Fraser described a mine-mill system which utilised bed blending techniques [41] to assist in grade control. In such systems, ore from various blocks in the mine is laid out by a "stacker" to form long beds of material whose grade conforms as closely as possible to that desired by the mill. Completed beds are subsequently reclaimed by taking cuts across the bed, i.e. perpendicular to the direction in which the material was laid, thus ensuring a good blend of the various ores in the bed. A bucket wheel reclaimer is frequently used for this purpose. A simple picture of day to day operations thus consists of ore on the one hand coming in from the mine to build up new beds, and on the other hand being reclaimed from earlier beds to form the mill feed.

The mining engineer is typically faced with the problem of planning say a day's production operations so that a total of T tons of ore is produced from the mine and placed on the latest bed (or portion of it) such that the final average composition of the bed is as close as possible to the desired mill limits (see Fig.8). The mathematical formulation of this problem proceeds much as before. We now suppose there are I constituents of the ore requiring grade control and we denote these by i, $i=1,\ldots,I$. Also we denote blocks in the mine by an index j, and we introduce the following quantities.

(a) Variables

x_j = tons mined from block j,

α_i = penalty incurred when the average grade of constituent i of the blend deviates from the desired mill limits,

(b) Given Quantities

g_{ij} = grade of the ith constituent of block j,

g_i^o = average grade of the ith constituent of any ore initially in the bed,

G_i^1, G_i^2 = lower, upper limits of grade desired for the ith constituent of the mill feed,

T^o = Initial tonnage of ore in the bed,

T = Desired production (in tons) from the mine during the current period,

p_i, q_i = penalty (e.g. in \$|% deviation) for each % shortfall, excess of constituent i in the blend,

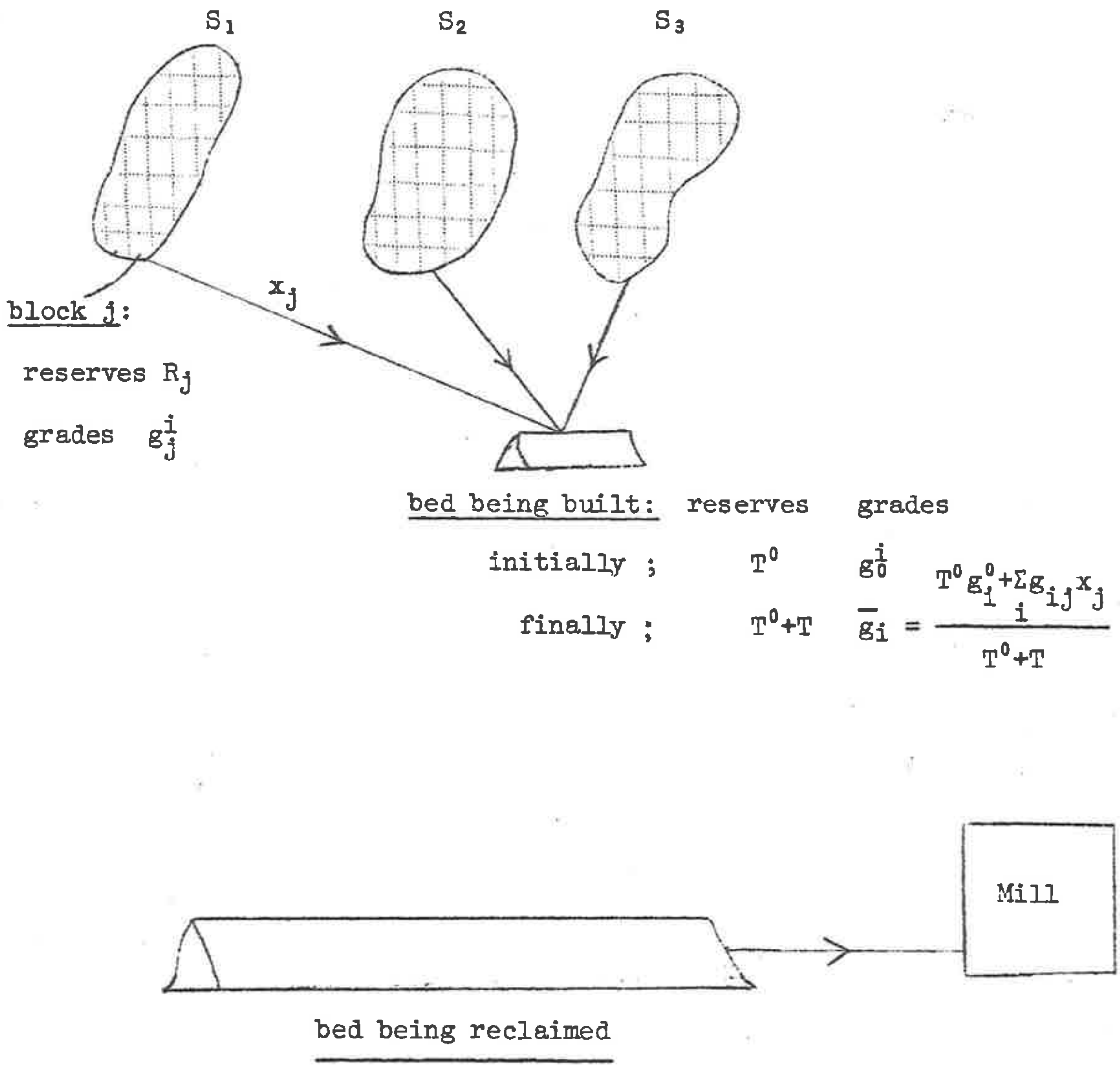


FIGURE 8

Schematic representation of bed blending system and of changes to the latest bed as a result of current production activities from 3 machine areas $S_1,..,S_3$.

a_k, b_k = lower, upper mining capacities of machine k,

R_j = Reserves of block j,

S_k = the set of blocks in machine area k.

The optimum blend is then determined by solving the problem

$$\text{Minimise} \quad z = \sum_i \alpha_i \tag{45}$$

subject to

$$p_i\left(G_i^1 - \frac{T^\circ g_i^\circ + T \sum_j g_{ij} x_j}{T^\circ + T}\right) \leqslant \alpha_i \tag{46}$$

$$q_i\left(\frac{T^\circ g_i^\circ + T \sum_j g_{ij} x_j}{T^\circ + T} - G_i^2\right) \leqslant \alpha_i \tag{47}$$

$$\sum_j x_j = T \tag{48}$$

$$a_k \leqslant \sum_{j \in S_k} x_j \leqslant b_k \tag{49}$$

$$0 \leqslant x_j \leqslant R_j, \quad \alpha_i \geqslant 0. \tag{50}$$

In contrast to the previous model (1) - (29), this formulation allows the average grade of the blend to deviate from the desired mill limits, but computes a penalty for so doing. It is necessary to follow such an approach when there are many constituents requiring grade control and it becomes impossible to ensure that all constituents will simultaneously lie within the desired mill limits. The values assigned to the penalties p_i and q_i

reflect the relative importance of the various constituents with respect to grade control. Also different values are allowed for p_i and q_i according to whether shortfall or excess of constituent i is more detrimental to the concentration process.

The operation of the objective function (45) can be seen as follows. Denote the average grade of constituent i in the final blend by

$$\bar{g}_i = \frac{T^o g_i^o + T \sum_j g_{ij} x_j}{T^o + T}$$

Any optimal solution of (45) - (50) must then satisfy exactly one of the following relations

$$\text{(i)} \quad \bar{g}_i < G_i^1 \quad \text{and} \quad \alpha_i^* = p_i(G_i^1 - \bar{g}_i) \qquad \text{(shortfall)}$$

$$\text{or} \quad \text{(ii)} \quad \bar{g}_i > G_i^2 \quad \text{and} \quad \alpha_i^* = q_i(\bar{g}_i - G_i^2) \qquad \text{(excess)}$$

$$\text{or} \quad \text{(iii)} \quad G_i^1 \leq \bar{g}_i \leq G_i^2 \quad \text{and} \quad \alpha_i^* = 0 \quad \text{(within desired limits).}$$

Thus in the optimal solution, the value of α_i^* is guaranteed to equal the actual penalty for shortfall or excess (if any) of constituent i in the blend. Note that this is only true of the optimal solution, since we may take any feasible solution for which this holds, and <u>increase</u> any α_i. This will still give us a feasible, but non-optimal solution and the property will not hold.

If it is possible to form a blend with <u>all</u> constituents within the desired limits, the optimal value of the objective function becomes identically equal to zero.

In this situation it may be desirable to further control extraction operations, by replanning them with tighter values of G_1^1 and G_1^2. Alternatively it may be preferable to maximise the content of a particularly valuable constituent of the ore subject to the strict imposition of grade constraints as before. Another variation consists of a combination of these two approaches.[1]

(2.3.2) Mining Constraints

As mentioned earlier, a more detailed knowledge of the deposit is required for short term planning purposes. This will be reflected in having available, at least in the immediate vicinity of the machines, estimates of the grades and tonnages of much smaller production blocks and several of these will be consumed by each machine during the blending period. Unlike before, we now insist on strict satisfaction of the mining requirements, that is a block may not be mined until the face advances into it. One good way of approximating this condition mathematically is as follows. Suppose the mining front is advancing into the deposit in a direction as shown in Fig.9. Let j be an arbitrary block <u>within</u> the deposit and let s_1, s_2, s_3 be the three nearest neighbours as shown in Fig.9. The re-

---

1. This in fact was the approach used in most computer runs since it gives rise to fewer dual degeneracies in the branch and bound section of the algorithm (q.v.) than (45).

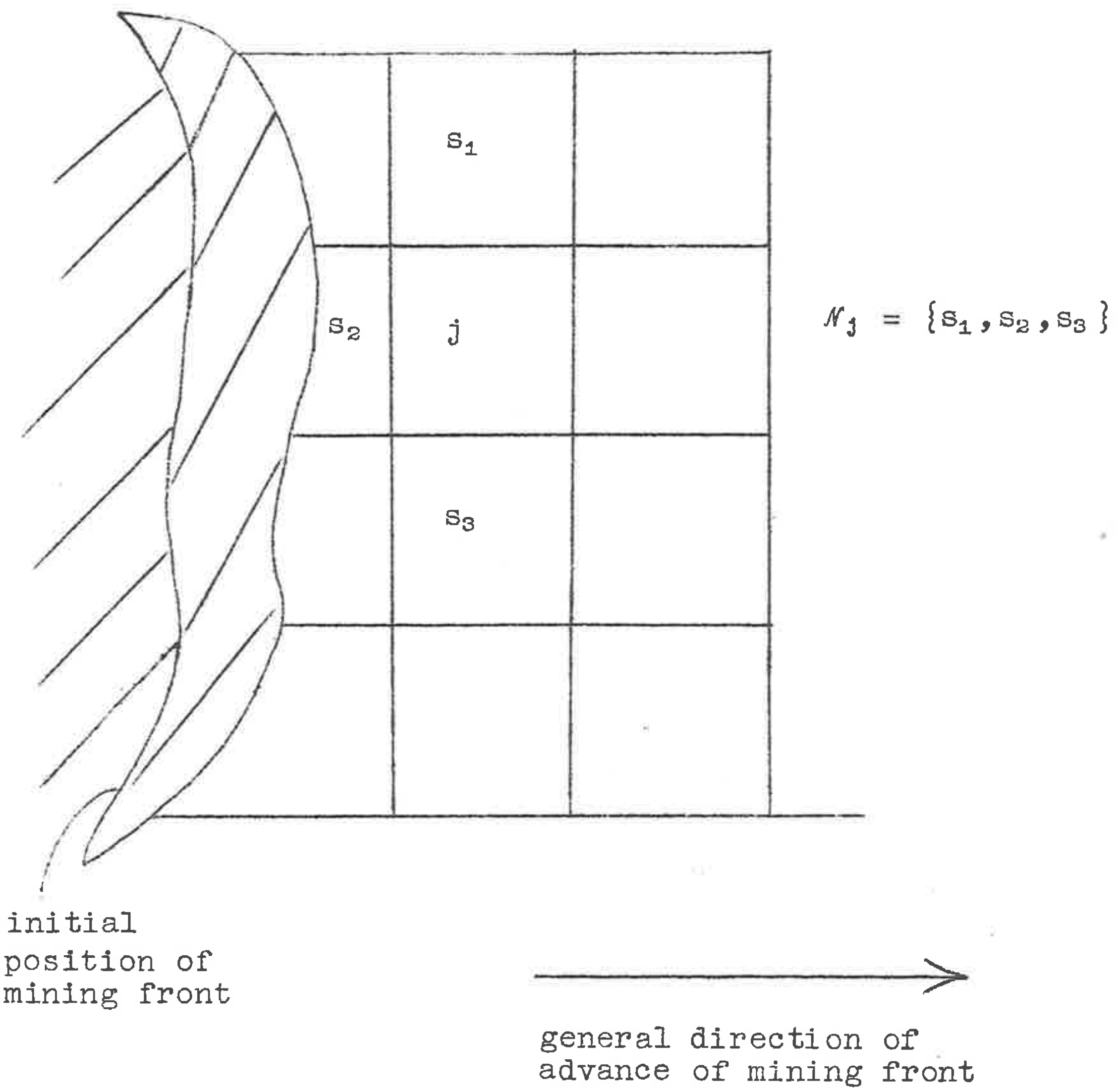


FIGURE 9

Mining constraints dictate that at least one of the blocks s_1, s_2 or s_3 must be completely mined out before mining may commence in block j.

quirement we impose is that x_j may only be positive if at least one of the blocks s_1, s_2 or s_3 is completely mined out. We shall say that block j is "surrounded" by the blocks s_1, s_2, s_3 and shall let $\mathcal{N}_j = \{s_1, s_2, s_3\}$ be the set of all such blocks.[1] The mining constraints may therefore be summarized as

$$x_j \prod_{s \in \mathcal{N}_j} (R_s - x_s) = 0 \tag{51}$$

and they must hold in addition to constraints (46) - (50).

The logical requirements expressed by the mining constraints (51) may be taken into account by the very standard trick of using 0-1 variables. It proceeds as follows. For each block s which surrounds other blocks (see Fig.9) introduce a 0-1 variable ε_s and the constraint

$$x_s \geqslant R_s \varepsilon_s, \tag{52}$$

while for each block j which is surrounded by others, introduce the 0-1 variable δ_j and the constraints

$$x_j \leqslant \delta_j R_j \tag{53}$$

---

1. (a) For short term planning it is usually not desirable to include the remaining nearest neighbour in the set $\mathcal{N}_j$. Mining it out in order to get at block j goes against the general direction of advance of the mining front.
(b) $\mathcal{N}_j$ may contain less than 3 nearest neighbours when block j is on a boundary of the deposit.

and
$$\delta_j \leqslant \sum_{s \in \mathcal{N}_j} \varepsilon_s \, . \tag{54}$$

This works as follows. Suppose we have a feasible solution for which $x_j > 0$. This immediately implies $\delta_j = 1$, otherwise constraint (53) would be violated. Constraint (54) then implies $\varepsilon_s = 1$ for at least one $s \in \mathcal{N}_j$. Inserting this into constraint (52) gives $x_s \geqslant R_s$, which in conjunction with constraint (50) means that $x_s = R_s$; i.e. surrounding block s has been completely mined out, and therefore the mining constraint for block j has been satisfied.

For easier reference we now express the 0-1 requirements explicitly as the constraint

$$\delta_j, \varepsilon_s = 0,1 \tag{55}$$

The complete mixed integer programming formulation of the blending problem with mining constraints then consists of (45) - (50) and (52) - (55). It may be solved using any standard mixed integer programming package. Several computer runs based on this approach have been carried out using the APEX-II package [2]. However CP times have not been encouraging; for instance 49 seconds (on the CDC 6400 computer) for a small problem involving 3 machines, 2 constituents and 23 blocks. The original problem in 32 variables (subject to bounds) and 8 (equality) constraints escalated to one of 116 variables (35 zero-one) and 57 constraints. However, only 17 nodes were required for solution.

In reality, the formulation (52) - (55) only serves to convert the problem into a format which is suitable for solution by the branch and bound procedure of a readily available mixed integer programming package. It is far more efficient to develop a new, but analogous, branch and bound procedure based directly on the mining constraints (28). In this way, we do not have to introduce any new variables or constraints and the linear programs we solve have bases whose size is only $(2I+K) \times (2I+K)$, where I is the total number of constituents requiring grade control and K is the total number of machines mining ore. This results in bases of size 8×8 in the above problem.

An even subtler reason why the new approach is more efficient than the mixed integer formulation is that the constraints (52) - (55) are stronger than the mining constraints (51). While a feasible solution to (52) - (55) guarantees that the mining constraints (51) will be satisfied, the converse is not true and it is easy to construct examples satisfying (51) and (52) - (54) but which do not satisfy <u>all</u> of the 0-1 requirements (55). An acceptable solution, in terms of (51), may thus be at hand, but will not be recognised as such by standard mixed integer programming algorithms until they set many more the fractional δ_j and ε_s values to 0 or 1.

(2.3.3) Branch and Bound Solution Procedure

We now develop the special branch and bound procedure based directly on the mining constraints (51). Consider the optimum solution to the problem (45) - (50) or any subproblem[1] derived from it by altering the bounds on the x_j (in a manner to be described shortly). By analogy with the normal integer programming approach, a block j in such a solution will be called "fractional" if $x_j > 0$ and $x_s < R_s$ for all $s \in \mathcal{N}_j$, that is the mining constraints (51) are violated for block j. To rectify this we must make one of the following decisions,

$$\text{(i)} \quad \text{Set} \quad x_j = 0 \tag{56}$$

or $$\text{(ii)} \quad \text{Set} \quad x_s = R_s \quad \text{for at least one} \quad s \in \mathcal{N}_j. \tag{57}$$

In general a fractional node in the solution tree will therefore generate more than the two subproblems normally associated with mixed integer programming (see Fig.10 for example). The restrictions (56) - (57) are imposed by redefining the upper bound on x_j as 0 and the lower bound on x_s as R_s respectively.

For convenience an outline of the general branch and bound enumeration procedure is reproduced here. The method is based on solving linear programs using variants of both the Primal and Dual Simplex algorithm which allow

---

1. As usual the original problem and subproblems derived from it are represented by nodes in a solution tree [4].

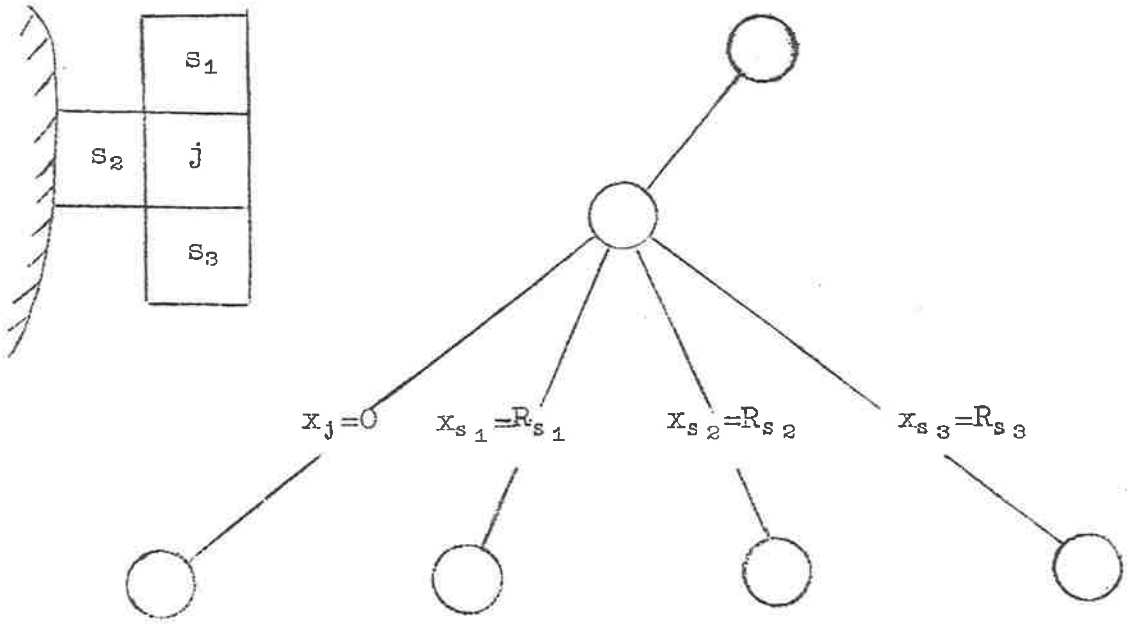


FIGURE 10

Four subproblems obtained from arbitrating fractional block j.

the variables to have arbitrary upper and lower bounds [9,14]. A solution which satisfies the mining constraints (51) is called (by analogy) "integer", and z_I^* is used to denote the objective function value of the currently best known integer solution. We then proceed as follows [31,5].

1. Initial Step

Set $z_I^* = \infty$.

Solve the linear program (45) - (50) using the primal simplex algorithm.

If no feasible solutions are found, stop.

If the optimal solution[1] is integer stop, otherwise,

2. Choose a fractional block for branching

Choose a fractional block j in the optimal solution of the current subproblem and add the new subproblems determined by the arbitrations (56) - (57) of block j to the list of subproblems awaiting solution.

3. Select a subproblem to solve

If the list of subproblems is empty, stop. Otherwise, choose one subproblem from the list, delete the reference to it in the list and test whether the objective function of its parent plus the Penalty is less than z_I^*. If it is go to 4, otherwise reenter 3.

4. Solve Subproblem

Reset the bounds on the x_j according to the arbitrations (56) - (57) which have been made to arrive at (i.e. define) the current subproblem, and, starting from either the current basis or if stored the optimal basis of the parent node, solve the subproblem using the dual simplex algorithm. If the objective function value becomes greater than or equal to z_I^* during this process, exit from the dual routine and go to 3, otherwise,

---

1. The linear program (45)-(50) does not have unbounded feasible solutions. Hence if feasible solutions exist, then so do optimal solutions.

5. Analyse the Solution of the Subproblem

If the subproblem has no feasible solution or the objective function value is greater than or equal to z_I^*, go to 3.

If the solution is integer go to 6; otherwise go to 2.

6. Integer Solution

An improved integer solution is at hand, hence reset z_I^* to equal the optimal objective function value of the current subproblem and then to to 3.

(2.3.4) Penalties

The general branch and bound algorithm contains two heuristic selection rules, namely in steps 2 and 3. As with normal integer programming, these selection rules may be based on penalty calculations. However, the calculations will differ from the more usual ones because of the special nature of the arbitrations (56) - (57).

To illustrate the construction of these selection rules, we first note that after step 1 of the algorithm, we only deal with dual feasible canonical forms of linear programs which consist of relations (45) - (50) but with some of the bounds on the x_j tightened according to the arbitrations (56) - (57). The linear programs are thus always of the form,

$$-z + \sum_{j\in L} \bar{c}_j x_j + \sum_{j\in U} \bar{c}_j x_j = -\bar{z}_0 \tag{58}$$

$$x_{B_i} + \sum_{j\in L} \bar{a}_{ij} x_j + \sum_{j\in U} \bar{a}_{ij} x_j = \bar{b}_i \tag{59}$$

where B is a set of basic variables, L,U are sets of non basic variables at their lower and upper bounds respectively, and

$$\bar{c}_j \geqslant 0 \quad \text{for} \quad j \in L$$
$$\bar{c}_j \leqslant 0 \quad \text{for} \quad j \in U.$$

This is the dual feasibility (primal optimality) requirement. Note that in the above, we have now used further x_j variables to represent the α_i and the slack and surplus variables, and have denoted the reduced cost coefficients, matrix and right hand side elements of (45) - (50) by $\bar{c}_j$, $\bar{a}_{ij}$, $\bar{b}_i$ (and $-\bar{z}_0$) respectively. Denoting the current lower and upper bounds on x_j by $\bar{p}_j$ and $\bar{m}_j$ the basic solution $\{B,L,U\}$ can be immediately written as,

$$\left.\begin{array}{l} x_j = \bar{p}_j , \quad j \in L, \\ x_j = \bar{m}_j , \quad j \in U, \\ x_{B_i} = b'_i = \bar{b}_i - \sum_{j\in L} \bar{a}_{ij}\, \bar{p}_j - \sum_{j\in U} \bar{a}_{ij}\, \bar{m}_j \end{array}\right\} \tag{60}$$

Also,

$$-z = -\bar{z}_0 - \sum_{j\in L} \bar{c}_j\, \bar{p}_j - \sum_{j\in U} \bar{c}_j\, \bar{m}_j. \tag{61}$$

The primal feasibility conditions, $\bar{p}_{B_i} \leqslant x_{B_i} \leqslant \bar{m}_{B_i}$ are satisfied in the final <u>optimal</u> solution of each subproblem.

As mentioned earlier, when we encounter a fractional node (subproblem) of the solution tree and make one of the arbitrations (56) - (57), we redefine exactly one $\bar{p}_j$ or one $\bar{m}_j$. Unlike normal mixed integer programming, however, such bound changes may happen to a <u>non basic</u> as well as a basic x_j variable. In the case of a basic variable, say $x_j = x_{B_r}$, the redefinition causes a primal infeasibility in row r, and the penalty, (minimum change in objective function value during removal of the primal infeasibility), is

$$P_j^D = \eta_r \, b_r', \quad \text{for an } x_j=0 \text{ branch} \tag{62}$$

and

$$P_j^U = \xi_r (R_{B_r} - b_r'), \quad \text{for an } x_j=R_j \text{ branch,} \tag{63}$$

where

$$\eta_r = \text{Min} \begin{cases} \dfrac{\bar{c}_j}{\bar{a}_{rj}}, & \bar{a}_{rj} > 0, \quad j \in L \\ \\ \dfrac{\bar{c}_j}{\bar{a}_{rj}}, & \bar{a}_{rj} < 0, \quad j \in U \\ \infty \end{cases} \tag{64}$$

and

$$\xi_r = \text{Min} \begin{cases} -\dfrac{\bar{c}_j}{\bar{a}_{rj}}, & \bar{a}_{rj} < 0, \quad j \in L \\ -\dfrac{\bar{c}_j}{\bar{a}_{rj}}, & \bar{a}_{rj} > 0, \quad j \in U \\ \infty . \end{cases} \tag{65}$$

In the case of a non basic variable x_j, the bound change has an immediate effect on the objective function value and the values of <u>all</u> basic variables in the basic solution $\{B,L,U\}$. The new values are given by

$$z'' = z - \bar{c}_j R_j$$

$$b_i'' = b_i' + \bar{a}_{ij} R_j$$

for an $x_j = 0$ branch, and

$$z'' = z + \bar{c}_j R_j$$

$$b_i'' = b_i' - \bar{a}_{ij} R_j$$

for an $x_j = R_j$ branch.

This yields immediate penalties of $-\bar{c}_j R_j$ and $\bar{c}_j R_j$ respectively. If some primal infeasibilities are also generated by the above process, an additional penalty may be computed by considering the change in the objective function value during the first dual simplex iteration. If this is based on removing an infeasibility in row i, the additional penalty is

$$\xi_i (\bar{p}_{B_i} - b_i''), \quad \text{if} \quad b_i'' < \bar{p}_{B_i},$$

or

$$\eta_i (b_i'' - \bar{m}_{B_i}), \quad \text{if} \quad b_i'' > \bar{m}_{B_i}.$$

The maximum additional penalty which may be so computed is

$$A = \mathrm{Max}\left\{0,\ \max_{i:b_i''<\bar{p}_{B_i}} \xi_i\left(\bar{p}_{B_i}-b_i''\right),\ \max_{i:b_i''>\bar{m}_{B_i}} \eta_i\left(b''-\bar{m}_{B_i}\right)\right\}$$

and the final penalties are therefore

$$P_j^D = -\bar{c}_j R_j + A, \text{ for an } x_j=0 \text{ branch}, \tag{66}$$

and

$$P_j^U = \bar{c}_j R_j + A, \text{ for an } x_j=R \text{ branch}. \tag{67}$$

Although the calculation of A will involve more effort than the normal choice of a dual simplest pivot, it is well worth while since, as pointed out by Tomlin [38], the computation of the highest possible penalties greatly assists in pruning the solution tree.

A computer program has been written which uses the standard tree development strategies [38] based on the above penalty calculations. For a non-terminal node we (normally) choose to arbitrate the block with the worst penalty namely

$$\underset{j\in F}{\mathrm{Max}}\left\{P_j^D,\ \underset{s\in \mathcal{N}_j}{\mathrm{Max}}\ P_s^U\right\} \tag{68}$$

where F is the set of all fractional blocks in the current solution. This specifies rule 2 of the general algorithm. Rule 3 is specified as follows. The branches descending from the current subproblem are ordered lexicographically according to increasing penalty value and the leftmost branch is selected for development. When a terminal node is reached, the next subproblem selected for solution is obtained by backtracking along the tree until the leftmost branch for which the objective function value of the parent plus the Penalty is less than z_I^*. This locates the leftmost unarbitrated branch of the tree for which it may still be possible to obtain an improved integer solution. We then proceed as before until another terminal node is reached.

It should be noted that the rule (68) may be modified to take into account "flaggable" blocks, similar to normal mixed integer programming [38]. A fractional block is flaggable if exactly one of its branches has a low enough penalty to qualify for development. Since there is thus only one choice which can possibly yield an improved integer solution, it can be made immediately and the rule (68) can be reapplied with F equal to the set of fractional, non-flaggable blocks to enable selection of a normal branch for development as well.

A second computer program incorporating these refinements has also been written and tested. Both programmes have been applied to reasonably small problems and there has not been a great difference in CP times. However there was a vast improvement on the more cumbersome mixed integer programming approach. For example, the execution time for the previously mentioned problem was

4.8 seconds. All tests showed that the new branch and bound procedure was a very efficient and very appealing way of enumerating all the possibilities which arise on account of the mining constraints.

2.4 MULTIPERIOD EXTENSION

We now present a multiperiod extension of the approach given in the previous section. As an example of its application, we have in mind the planning of operations on say a daily basis for maybe a week or two in advance. While this can be achieved by repeated use of the single period model, the more global approach may yield a better solution. The individual periods we shall consider may be thought of as having equal duration, but this is not necessary, so long as we know the production targets and machine capacities during each period. We shall make one important assumption however; namely, that the ore produced from the mine during each period will be used to form separate beds (or perhaps separate portions of the same beds). Consequently we would like each separate production batch to conform as closely as possible to the mill's requirements. This in turn causes the linear programs to assume a particularly simple form which may be exploited to increase the efficiency with which the solutions can be found.

The multiperiod generalization of the linear program (45) - (50) is very readily obtained as

$$\text{Minimise} \quad z = \sum_t \sum_i \alpha_i^t \qquad (69)$$

subject to

$$\sum_j g_{ij} x_j^t + \frac{\alpha_i^t}{p_i^t} - s_i^t = G_i^t \qquad (70)$$

$$\sum_j g_{ij}x_j^t + \frac{\alpha_i^t}{q_i^t} + u_i^t = G_i^2 \qquad (71)$$

$$\sum_j x_j^t = T^t \qquad (72)$$

$$\sum_{j\in S_k^t} x_j^t + v_k^t = b_k^t \qquad (73)$$

$$\sum_t x_j^t + w_j = R_j \qquad (74)$$

$$\alpha_i^t, s_i^t, u_i^t, w_j \geqslant 0, \quad 0 \leqslant v_k^t \leqslant \left(b_k^t - a_k^t\right), \qquad (75)$$

where

x_j^t = tons mined from block j during period t,

α_i^t = penalty (in dollars) for shortfall or excess of constituent i in the blend formed in period t,

a_k^t, b_k^t = lower, upper machine capacities in period t,

T^t = production requirement for period t,

p_i^t, q_i^t = penalties for each % shortfall, excess of constituent i in the blend formed in period t,

S_k^t = blocks available for mining by machine k during period t,

s_i^t, u_i^t, v_k^t, w_j = surplus or slack variables.

The remaining coefficients in (69) - (75) have the same interpretation as before. Note that the penalties p_i^t, q_i^t have now been allowed to depend on t since it may be desirable to weight the earlier coefficients p_i and q_i by the tonnages T^t, if the latter are allowed to differ from period to period.

A significant difference between the single and multiperiod formulations occurs in the reserves constraints (74). For blocks which may be mined in two or more periods, the reserves constraints (74) form <u>generalized</u> upper bound constraints [25] rather than the simple upper bound constraints in (50) and consequently they now appear explicitly in the constraint matrix. For blocks which are available for mining during one period only, it is preferable to revert to the earlier more efficient version of the reserves constraints (50) rather than (74).

The mining constraints are also more involved than before and they are now written as

$$x_j^t \prod_{s \in \mathcal{N}_j} \left(R_s - \sum_{\tau=1}^{t} x_s^\tau\right) = 0, \quad \forall j,t. \tag{76}$$

Again this means that mining may not commence in block j until at least one of the surrounding blocks (in $\mathcal{N}_j$) is mined out. The special branch and bound procedure developed in the previous section for optimizing a single period linear programming blending problem subject to mining constraints may be easily extended to solve the multiperiod problem (69) - (76): Suppose we have an optimal solution to problem (69) - (75) or any subproblem derived from it by altering the bounds on various variables. In such a solution, a block j will now be called fractional in period t if the constraint (76) is violated, i.e.

$$x_j^t > 0 \quad \text{and} \quad \sum_{\tau=1}^{t} x_s^\tau < R_s \quad \text{for all} \quad s \in \mathcal{N}_j.$$

Arbitrating this fractional block gives rise to the following branches;

1. an $x_j^t = 0$ branch and,

2. $\sum_{\tau=1}^{t} x_s^\tau = R_s$, $s \in \mathcal{N}_j$ branches.

An $x_j^t = 0$ branch is made by setting an upper bound of zero on the variable x_j^t, while an $\sum_{\tau=1}^{t} x_s^\tau = R_s$ branch is made by setting an upper bound of zero on the variable w_s and also all variables x_s^τ, with $\tau > t$, which appear in the model. Penalty calculations based on these bound changes may be developed as before.

Computing times for the multiperiod approach increase exponentially with the number of periods and we have to make further use of the structure of the problem to obtain solutions efficiently. There are two reasons for this rapid increase in computing time. Firstly the mining constraints (76) include many more possibilities than are considered by repeated application of the single period model. (This is why the multiperiod approach can of course yield a better solution). Secondly the linear program (69) - (75) is much larger than before.

As hinted at already, the effect of the mining constraints (76) may be reduced by using our mining knowledge to assign the blocks to "logical" periods instead of to all periods; that is we only allow the model to consider logical or preferable alternatives rather than all possibilities. For example, suppose we wish to plan operations for Monday through to Friday. It will then be impossible to mine, on Monday, certain blocks which are located well inside the deposit, i.e. far from the current mining front. (Indeed it may be possible to reach some of them by Friday only). If we indiscriminantly allow an x_j^t variable in the model for these blocks, for the value of t corresponding to Monday, then these variables may quite well assume positive values in the optimal solution of many nodes of the solution tree. The mining constraints will be automatically violated and even though this may be obvious to the mining engineer, the model, if unaided, may have to examine many more nodes before it also discovers this and then correctly fixes the above x_j^t at zero.

Careful selection of the blocks to be mined in any period thus increases efficiency by removing illogical nodes from the solution tree. Further decreases in CP

~~time~~ may be had by utilising the structure of the linear program (69) – (75) to increase the speed with which we can solve the various linear programs for the nodes which are generated by the algorithm. This structure is immediately revealed when we write the linear program (69) - (75) in the matrix form,

$$\text{Min } z = \underset{\sim}{c}_1'\underset{\sim}{x}^1 + \underset{\sim}{c}_2'\underset{\sim}{x}^2 + \dots \underset{\sim}{c}_t'\underset{\sim}{x}^t + \dots \underset{\sim}{c}_n'\underset{\sim}{x}^n \qquad (77)$$

subject to

$$\begin{array}{rlllllll}
m_0 \text{ rows} & \{E_0\underset{\sim}{x}^0 + E_1\underset{\sim}{x}^1 + & E_2\underset{\sim}{x}^2 + \dots & E_t\underset{\sim}{x}^t + \dots & E_n\underset{\sim}{x}^n & = \underset{\sim}{R} & & (78)\\
m \text{ rows} & \{ \quad B_1\underset{\sim}{x}^1 & & & & = \underset{\sim}{b}_1 & \\
m \text{ rows} & \{ & B_2\underset{\sim}{x}^2 & & & = \underset{\sim}{b}_2 & \\
\vdots & & \ddots & & & \vdots & & (79)\\
m \text{ rows} & \{ & & B_t\underset{\sim}{x}^t & & = \underset{\sim}{b}_t & \\
\vdots & & & \ddots & & \vdots & \\
m \text{ rows} & \{ & & & B_n\underset{\sim}{x}^n & = \underset{\sim}{b}_n &
\end{array}$$

$$\underset{\sim}{d}^t \leqslant \underset{\sim}{x}^t \leqslant \underset{\sim}{e}^t. \qquad (80)$$

The vector $\underset{\sim}{x}^0$ consists of the slack variables w_j, while the variables x_j^t, α_i^t, s_i^t and u_i^t are all included in the vector $\underset{\sim}{x}^t$. The reservés constraints (74) which span two or more periods appear in (78), while the blending requirements appear in the $\{B^1,\ldots,B^n\}$ portion of the constraint matrix. The consequences of forming separate blends are immediately reflected in the simple block diagonal structure of this submatrix. As mentioned in Chapter 1, many multiperiod problems have this structure and there are several ways in which it can be exploited. The remainder of this section demonstrates that the contracted basis techniques developed by Kaul [21] and Lasdon [25] are particularly suitable for use with the new branch and bound procedure to solve the problem (69) - (76). These methods rely on representing any full basis of the system (78) - (79) more compactly by a working basis only $m_0 \times m_0$ in size, plus basis matrices, of size $m \times m$ from each B_t.† The simplex method, whether primal or dual, or incorporating arbitrary lower and upper bounds on the variables (as in our case), may be applied with considerable savings in time.

The full algorithm is given by Lasdon but a few details will now be summarized to demonstrate the further savings that may be gained (in both computation and storage) by utilising the additional structure possessed by the matrices $E_0, E_1, \ldots, E_n$. These matrices are in fact just

---

† m_0 denotes the number of rows possessed by the matrices $E_0, E_1, \ldots, E_n$; m denotes the number possessed by $B_1, B_2, \ldots, B_n$.

incidence matrices; their columns are either zero or unit vectors. Corresponding to row r, $(r=1,\ldots,m_0)$, of $E_0,E_1,\ldots,E_n$, there is a generalized upper bound reserves constraint for some block, say block ℓ_r. It follows that for all variables $x^{j}_{\ell_r}$ which appear in the model, the corresponding column of E_t is the unit vector $\underset{\sim}{e}_r$. The column of E_0 corresponding to the slack variable w_{ℓ_r} also consists of $\underset{\sim}{e}_r$. All columns of $E_0,E_1,\ldots,E_n$ not so accounted for consist of zero vectors. The E_t may thus be stored very compactly by recording the row number of the unit element of each column which is a unit vector and a zero for the remaining columns.

Since the submatrices $E_0,B_1,\ldots,B_n$ all have full rank, so does the complete constraint matrix,

$$\mathcal{A} = \left[\begin{array}{c|ccccc} E_0 & E_1 & E_2 & \cdots & E_n \\ \hline & B_1 & & (0) & \\ & & B_2 & & \\ 0 & & & \ddots & \\ & & (0) & & B_n \end{array}\right].$$

Moreover any basis $\mathcal{B}$, chosen from it, can be partitioned to have the form

$$\mathcal{B} = \left[\begin{array}{ccccc|c} E_{11} & E_{21} & \cdots & & E_{n1} & \hat{B} \\ \hline B_{11} & & & & & C_1 \\ & B_{21} & & & & C_2 \\ & & \cdot & & & \cdot \\ & & & \cdot & & \cdot \\ & & & & \cdot & \cdot \\ & & & & B_{n1} & C_n \end{array}\right] \tag{81}$$

where each E_{t1} and B_{t1} are submatrices (of dimension $m_0 \times m$ and $m \times m$ respectively) of E_t and B_t and also the B_{t1} are nonsingular.

Lasdon uses the transformation matrix

$$\mathcal{J} = \left[\begin{array}{ccccc|c} I & & & & & V_1 \\ & I & & & & V_2 \\ & & \cdot & & & \cdot \\ & & & \cdot & & \cdot \\ & & & & \cdot & \cdot \\ & & & & I & V_n \\ \hline & & 0 & & & I \end{array}\right] \tag{82}$$

where $V_t = -B_{t_1}^{-1} C_t$ to reduce $\mathcal{B}$ to block triangular form. Thus

$$\mathcal{B}\mathcal{J} = \left[\begin{array}{cccc|c} E_{11} & E_{21} & \cdots & E_{n1} & B \\ \hline B_{11} & & & & \\ & B_{21} & & & \\ & & \ddots & & 0 \\ & & & B_{n1} & \end{array}\right]$$

where

$$B = (E_{11}\ V_1 + E_{21}\ V_2 + \ldots E_n\ V_n) + \hat{B},$$

and this is the so called working basis. We can now easily invert the matrix $\mathcal{B}\mathcal{J}$ as

$$(\mathcal{B}\mathcal{J})^{-1} = \left[\begin{array}{c|cccc} & B_{11}^{-1} & & & \\ 0 & & B_{21}^{-1} & & \\ & & & \ddots & \\ & & & & B_{n1}^{-1} \\ \hline B^{-1} & -B^{-1}E_{11}B_{11}^{-1} & \cdots & & -B^{-1}E_{n1}B_{n1}^{-1} \end{array}\right],$$

and since the inverse of the original basis matrix $\mathcal{B}$ is then given by

$$\mathcal{B}^{-1} = \mathcal{J}\mathcal{J}^{-1}\mathcal{B}^{-1} = \mathcal{J}\ (\mathcal{B}\mathcal{J})^{-1},$$

we only need to store and maintain the submatrices B^{-1}, $B_{11}^{-1},\ldots,B_{n1}^{-1}$ and $V_1,\ldots,V_n$ in order to be able to generate all quantities needed for any simplex iteration (primal or dual).

For instance, the simplex multipliers $\underset{\sim}{\Pi}' = (\underset{\sim}{\pi}_0', \underset{\sim}{\pi}_1', \ldots\ldots, \underset{\sim}{\pi}_n')$ are obtained by

$$\underset{\sim}{\Pi}' = -\underset{\sim}{c}_{\mathcal{B}}' \mathcal{J}(\mathcal{B}\mathcal{J})^{-1} \tag{83}$$

where $\underset{\sim}{c}_{\mathcal{B}'}'$ are the cost coefficients corresponding to the basic variables of $\mathcal{B}$. $\underset{\sim}{c}_{\mathcal{B}}'$ has the form

$$\underset{\sim}{c}_{\mathcal{B}}' = (\underset{\sim}{c}_1', \underset{\sim}{c}_2', \ldots, \underset{\sim}{c}_n' \vdots \underset{\sim}{c}'),$$

so by defining

$$\underset{\sim}{d}' = (\underset{\sim}{d}_1', \underset{\sim}{d}_2', \ldots\ldots, \underset{\sim}{d}_n' \vdots \underset{\sim}{d}') = \underset{\sim}{c}_{\mathcal{B}}' \mathcal{J}$$

we obtain

$$\underset{\sim}{d}' = \underset{\sim}{c}_1' V_1 + c_2' V_2 + \ldots + \underset{\sim}{c}_n' V_n + \underset{\sim}{c}'$$

and

$$\underset{\sim}{d}_t' = \underset{\sim}{c}_t', \quad t = 1, \ldots, n.$$

Equation (83) then enables the components of $\underset{\sim}{\Pi}'$ to be computed as

$$\underset{\sim}{\pi}_0' = -\underset{\sim}{d}' B^{-1}$$

and

$$\underset{\sim}{\pi}_t' = -(\underset{\sim}{d}_t' + \underset{\sim}{\pi}_0' E_{t1}) B_{t1}^{-1}, \quad t = 1, \ldots, n.$$

The simple structure of the matrices E_{t1} now enables the $\underset{\sim}{\pi}_t'$ to be computed much faster than for general multiperiod problems. In particular, the columns of the product $\underset{\sim}{\pi}_0' E_{t1}$ corresponding to zero vectors of E_{t1} are just zero, while those corresponding to unit vectors of E_{t1} are simply particular elements of $\underset{\sim}{\pi}_0'$. The calculation of $\underset{\sim}{\pi}_t'$ will thus be roughly equivalent to multiplying a row vector with a matrix and coupled with the fact that the matrices are now much smaller since they refer to individual blending periods, substantial savings are gained compared to working with the full basis matrix.

The computation of the transforms, with respect to basis $\mathcal{B}$, of the non-basic costs, the right-hand side elements and an arbitrary column or row of the original constraint matrix $\mathcal{A}$ are outlined in Appendix II. The additional structure derived from the E-matrices persists throughout these computations and again they go much faster than usual. Once completed, they give us all the information required for one iteration of the primal or dual simplex algorithm. The methods for updating B^{-1}, $B_{11}^{-1},\ldots,B_{n1}^{-1}$ and $V_1,\ldots,V_n$ as a result of a pivotal transformation can likewise be easily developed by partitioning the corresponding elementary matrix according to that of $\mathcal{B}$ and further details may be had in Lasdon [25].

The discussion presented so far in this and the previous section has concentrated on the techniques necessary to obtain efficient solutions to the short term planning models (45) - (50) and (69) - (75). Application of the methods to test-problems has proven that they are effective, but the author has not yet had the opportunity to have them implemented in a real-world situation. In practice the mining engineer, if he so desires, will be able to exert considerable influence over the final plan chosen by carefully specifying which blocks are to be incorporated in the model. This opportunity for involvement of the mining engineer is an attractive feature of the models and it is suggested that the greatest benefits will be obtained if the computer programs written for actual use have facilities for easy alteration of the relevant parameters. The ability to do this interactively would particularly enhance the success of the approach.

CHAPTER 3

OPTIMAL DREDGE PATHS

3.1 GENERAL

In this chapter further application of Operations Research techniques is made to long-term production planning (say for five or more years) from a beach sand deposit which is to be mined by a dredge. The dredge may consist of either a suction-cutter or bucket wheel mechanism attached to a floating pontoon. It advances along by slowly eating its way through the deposit. Towed behind it is a floating primary concentration plant whose function is to remove most of the gangue from the ore and eject it to the rear as tailings. The rougher concentrate which it produces is sent (e.g. pumped) to a final concentration plant.

The aspect of the long-term planning which we shall be concerned with is the determination of the dredge path. The critical factors are the extremely limited mobility of the dredge, the continuity of its path and the spatial distribution of block grades and tonnages in the deposit.

3.2 NETWORK MODEL

(3.2.1) The Network

As before we can represent movements of the dredge through the deposit by using a mathematical network in which nodes stand for the blocks in the mine and links represent the allowed interblock movements. The model which we shall now discuss was developed and applied to an actual beach sand deposit. However, we shall use a hypothetical deposit such as shown in Fig.11 to illustrate the approach. We shall suppose the dredge is to start in the North-West corner of this deposit and is to finish in the South; the block in the North-West corner is therefore taken as the origin node and is denoted by 0, but since we shall allow any block on the southern boundary to be the final one to be mined, an extra destination node (denoted by D) is added and connected to all such blocks.

The basic assumption we make is that the movement capabilities of the dredge can be approximated by providing links from each block to just its nearest neighbours. With block sizes of the order of 100×100 metres, the network model gives a good description of the possible dredge movements; however, to illustrate the sorts of networks which arise, we divide the deposit shown in Fig.11 into larger subareas of size 1500m×500m and obtain a correspondingly

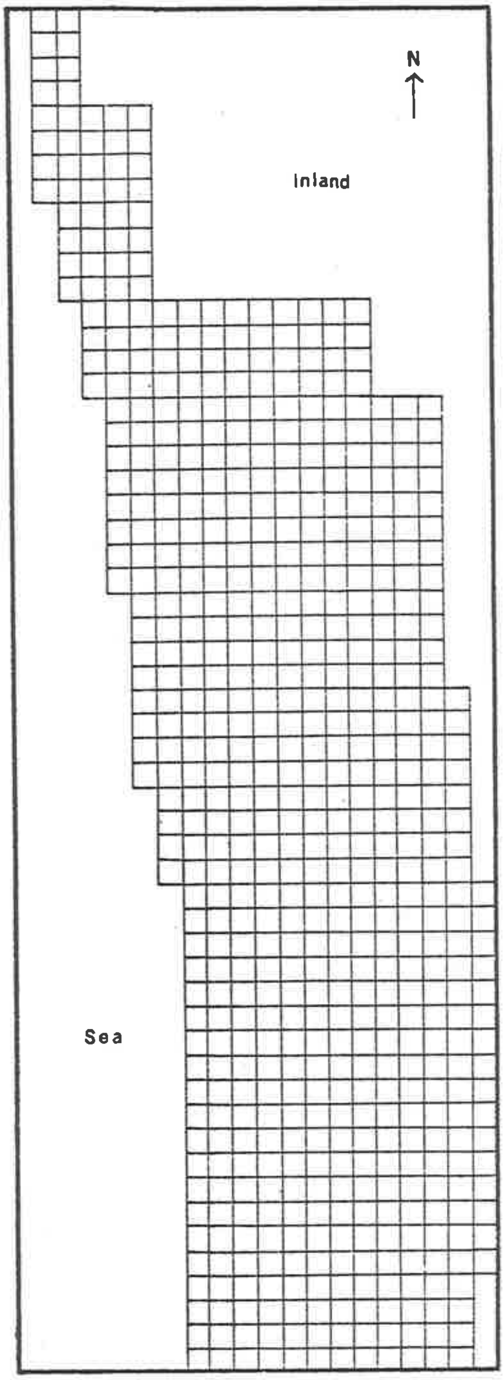


FIGURE 11

Plan View of Beach Sand Deposit. Block Sizes are 100m×100m.

smaller network consisting of nodes $\{1,2,\ldots,14\}$ and links $(1,2)$, $(1,4)\ldots,(13,14)$ as shown in Fig.12.

For our purposes it is convenient to summarize the network by using its "successor matrix" which we shall denote by $S = \{s_{ik}\}$. The elements in row i of S are obtained by listing the successors of node i in some order, the most convenient being lexicographical ordering according to the numbers representing each node: s_{ik} will thus be the kth such successor of node i when node i has k or more successors; otherwise we set s_{ik} to zero. For the network illustrated in Fig.12, the successor matrix is immediately given by

i \ k	1	2	3	4
1	2	4		
2	3	5		
3	2	6		
4	5			
5	2	6		
6	3	5	7	9
7	6	10		
8	5	6	9	
9	6	8	10	12
10	7	9	13	
11	8	12	14	
12	9	11	13	14
13	10	12	14	
14				

where zero elements are represented by blanks.

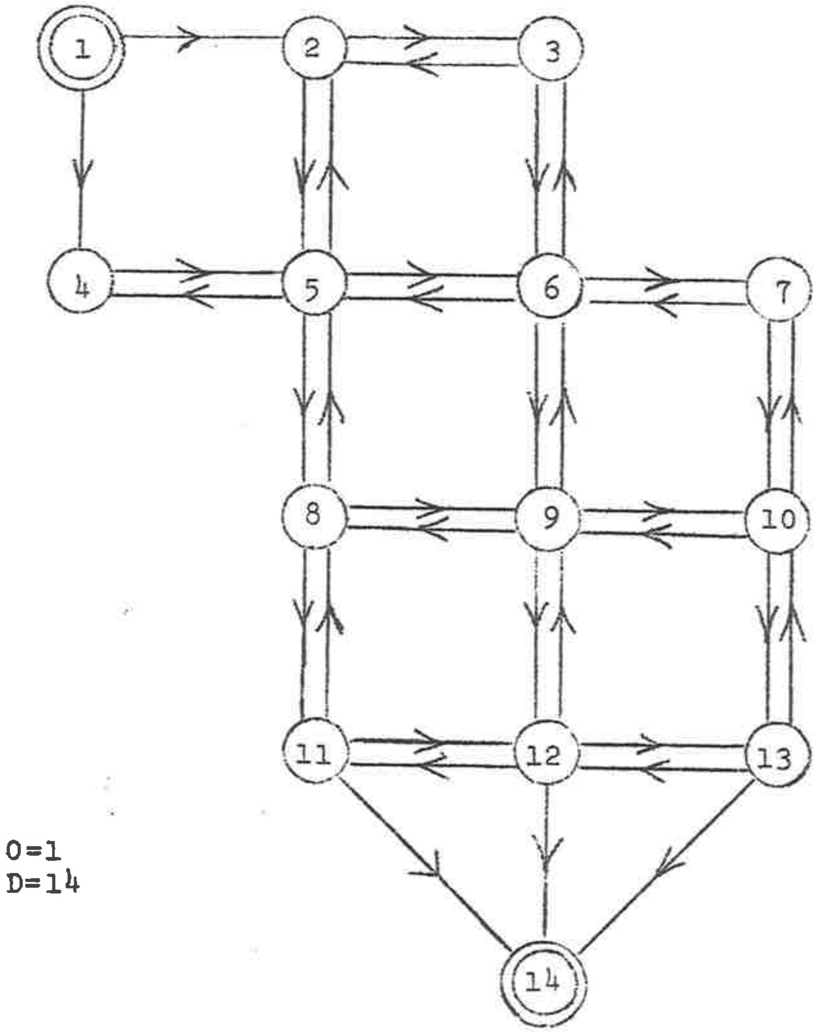


FIGURE 12

Sample Network

(3.2.2) Maximal Chains

In practice it is possible for the dredge to pass through a block more than once. However, on subsequent passages the block will consist only of old tailings and such manoeuvres are therefore very rare. Consequently we restrict ourselves to dredge paths which are chains from the origin 0 to the destination D. A chain in a discrete network is defined formally as follows: If $n_1, n_2, \ldots, n_r$ are distinct nodes such that (n_i, n_{i+1}), $i = 1,2,\ldots,r-1$ are distinct links of the network, then the sequence

$$n_1, (n_1, n_2), n_2, \ldots\ldots, n_{r-1}, (n_{r-1}, n_r), n_r$$

defines a chain[1] from the origin node n_1 to the destination node n_r.

For purposes of evaluating the various 0-D chains we assign to each block in the deposit, a value w_i equal to the net profit obtained from dredging the block. w_i is computed by subtracting all costs of production from the revenue earnt from the block. For example, for an iron-sand deposit mined to give a concentrate assaying 59% Fe for sale at $6.00 per tonne (of concentrate) the block worths w_i (in dollars) are given by

---

[1] The chain can clearly be uniquely defined by the sequence of nodes or the sequence of links alone.

$$w_i = T_i \left[6.00\, R\, \frac{g_i}{59} - d_i - c \right]$$

where

T_i = total tonnage to be dredged,

g_i = average grade (e.g.%Fe) of the block,

R = total recovery (the amount of Fe, say, recovered in the concentrate divided by the total amount originally present in the block),

d_i = dredging cost in \$/tonne dredged,

c = remaining costs (in \$/tonne).

The values of g_i and T_i and hence the economics of mining block i depend considerably on how deep we decide to dredge. This is because the grades usually vary very considerably with depth, higher grades being found above the water-table and lower grades below it. There are also several technological factors which need to be considered when deciding on the level of the dredge pond bottom but we assume that all such decisions have been made beforehand and are reflected in the final values used for T_i, g_i and d_i - typically we may dredge from the surface to sea level.

To perform a proper financial evaluation of any chain, we have to discount the receipts obtained from the various blocks belonging to the chain. The following was found to be a particularly suitable discounting scheme in view of the fact that the times taken to dredge blocks var[y] according to the tonnages of ore contained therein. Suppose dredging of block i starts at time t_1 (expressed in years say) and finishes at time t_2. w_i is then a future receipt which we discount to a present value v_i, given by

$$v_i = \frac{w_i}{(t_2-t_1)\ln(r+1)}\left[\frac{1}{(r+1)^{t_1}} - \frac{1}{(r+1)^{t_2}}\right], \qquad (84)$$

where r is the discount rate (per annum).
This formula is obtained by assuming that the receipt w_i is <u>continuously</u> discounted over the time period $[t_1, t_2]$ during which ore is being obtained from block i. (The constant $\ln(r+1)$ in (84) just ensures that $\lim_{t_2 \to t_1} v_i = w_i$ and $\lim_{r \to 0} v_i = w_i$).

We now let $\{n_1=0, n_2, n_3, \ldots, n_m, n_{m+1} = D\}$ be a chai[n] consisting of $m+1$ nodes (hence m blocks) from 0 to [D]. Its total present value is given by

$$V = \sum_{k=1}^{m} \frac{w_{n_k}}{(t_{k+1}-t_k)\ln(r+1)}\left[\frac{1}{(r+1)^{t_k}} - \frac{1}{(r+1)^{t_{k+1}}}\right] \qquad (85)$$

where t_k is the commencement time for dredging block n_k. These are obtained recursively by

$$t_1 = 0,$$

$$t_{k+1} = t_k + T_{n_k}/\tau, \quad 1 \leqslant k \leqslant m$$

where τ is the rate of dredging (in tonnes/year).

The above scheme has the property that if the dredge had no mobility restrictions whatsoever, the optimal strategy would be,

1. Do not mine blocks with negative w_i values,
2. Mine the remaining blocks in decreasing order of grade.

Because of the very limited dredge mobility, we may, however, have to violate this simple strategy. In any case we seek the O-D chain which maximises the expression (85) When r=0, this reduces to a conventional longest path problem in a network with positive cycles[1] and as noted in Chapter 1, such problems are, in general, notoriously difficult to solve. Fortunately an effective heuristic algorithm can be obtained for the dredge problem, by simply partitioning its full network into smaller networks and then basically using brute force enumeration to optimise the smaller subproblems. The key section of the algorithm was a subroutine used to generate all[2] O-D chains in an arbitrary network. Each O-D chain produced is evaluated separately according to (85) and the maximal chain is saved.

---

1. A cycle is defined as a chain except that n_1 must equal n_r.

2. Except in the refinements to be discussed in §3.2.4 (q.v.

Before describing the full algorithm, we shall briefly outline the method employed in the above-mentioned subroutine. The method was based on a familiar lexicographical enumeration procedure which utilises the orderings (of the successors of any node) as given by the successor matrix S. In what follows it is convenient to call a node j the <u>kth successor of node i</u> (with respect to S) if $s_{ik} = j$. Also we shall use the words "left" and "right" to denote the relative positions in S of different successors of a given node i; thus $j_1=s_{ik_1}$ will be called left of $j_2=s_{ik_2}$ if and only if $k_1 < k_2$ etc. Similarly the words "left" and "right" will be freely used to denote the relative positions of nodes in a chain; node n_i is left of node n_j in the chain $\{n_1,n_2,\ldots,n_m\}$ if and only $i < j$ etc. We now proceed as follows.

1. <u>Initial Step</u>

   Start with a chain consisting of the origin node 0 and its <u>first</u> successor (i.e. node s_{01}).

2. <u>Increment Chain if Possible</u>

   (a) If the final node of the chain has a successor not already used in the chain, add the leftmost such successor to the chain and then re-enter 2.

   (b) If the final node is the destination node, we have constructed the latest O-D chain;

evaluate it, update the incumbent if necessary and then go to 3.

(c) If the final node is not the destination node, go straight to 3.

3. Backtrack to the Rightmost Unfathomed node of the chain

Let j and i denote the last and second to last nodes of the chain respectively.

(a) Delete node j from the chain.

(b) If node i has a valid (i.e. unused) successor to the right of node j (in S), add this successor to the chain and then go to 2.

(c) If node i is the origin node O, stop. (All chains have been enumerated).

(d) If node i is not the origin, re-enter 3.

(3.2.3) Heuristic Algorithm and its Application

As expected, we cannot apply complete enumeration to the deposit shown in Fig.11 when block sizes are in the vicinity of 100m×100m as needed for the network to closely model the dredge's movement capabilities. The network becomes just too large (for example there are 620 100m×100m blocks). The rapid exponential increase in CP time with increasing size of the network (i.e. finer partitions of the deposit) is clearly shown in Table 4.

TABLE 4. CP times for complete enumeration vs Block Size

Block Sizes (m^2)	Network Size			CP times[a]
	No. Nodes	No. Links	No. O-D Chains	
1000 × 1000	12	30	32	0.2 secs
1000 × 500	20	59	1,156	5 secs
1000 × 400	24	74	7,104	37 secs
800 × 400	27	84	22,130	127 secs
500 × 500	37	115	828,738	46[b] mins

[a] On CDC 6400 computer

[b] No discounting

As mentioned earlier, the heuristic algorithm consists of a two stage search procedure in which we first divide the whole deposit into larger subareas (e.g. 800m×400m in size) and then apply the enumeration methods to the resulting network to enable us to sequence production from these subareas in a broadly optimal manner. We then optimise within each subarea in order to produce a proper dredge path. Fig.13 shows how this works in greater detail. We suppose that the first stage has set aside subareas I and II for successive development with subarea I to be dredged first. For purposes of illustration, we have divided these subareas into 16 blocks each. We also assume that we have already determined the dredge path in all

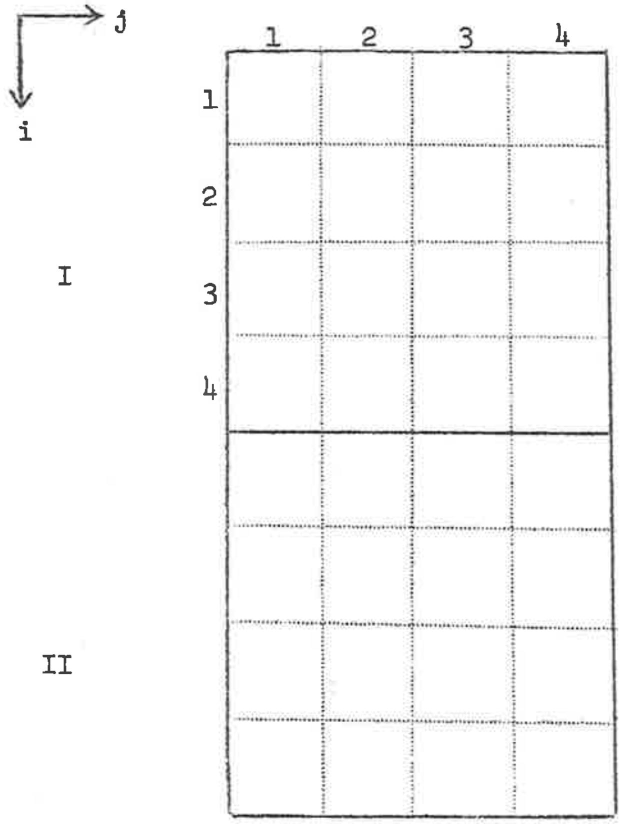


FIGURE 13

subareas mined prior to subarea I (in the order defined by the first stage maximal chain). To determine the path in subarea I, we will therefore be starting at some known block, say block (1,1), and will be desiring to finish on one of the blocks (4,1), (4,2), (4,3) or (4,4) which lie on the boundary between subareas I and II. We construct a network for subarea I which models this situation and apply the enumeration procedure to determine the dredge path through it. The final block on the chosen path is noted and the block adjacent to it in subarea II is taken as the origin node in this subarea, and the whole process is repeated. The algorithm thus constructs local optima in which the dredge has to complete all mining operations within any of these larger subareas before proceeding into the next subarea.

Because the network model corresponding to the coarser subdivision of the deposit made in stage 1, implies less mobility on the dredge than is actually the case, we give special treatment to the block values assigned to the nodes representing these larger subareas. In particular we aggregate only the positive 100×100m block values, tacitly assuming that once in any such subarea, the dredge will exercise its somewhat greater manoeuverability in an attempt to avoid the negatively valued blocks. If there are no such positive 100×100m blocks, a negative value corresponding to a path straight across the largest dimension of the subarea is used to discourage entry into the subarea. Stage 1 of the heuristic algorithm thus sequences production from the larger subareas concentrating on the effect of discounting and imposing broad continuity requirements on the dredge's movements.

Some examples of nonoptimality may occur in the final paths at boundaries between successive subareas, especially when we perform independent optimization over the various subareas. Figure 14 gives a trivial illustration of this. Simple extensions can however be made to discourage such transitions. In fact it was found that a very quick yet effective way of doing this is to look just one block ahead of the boundary. Thus in the

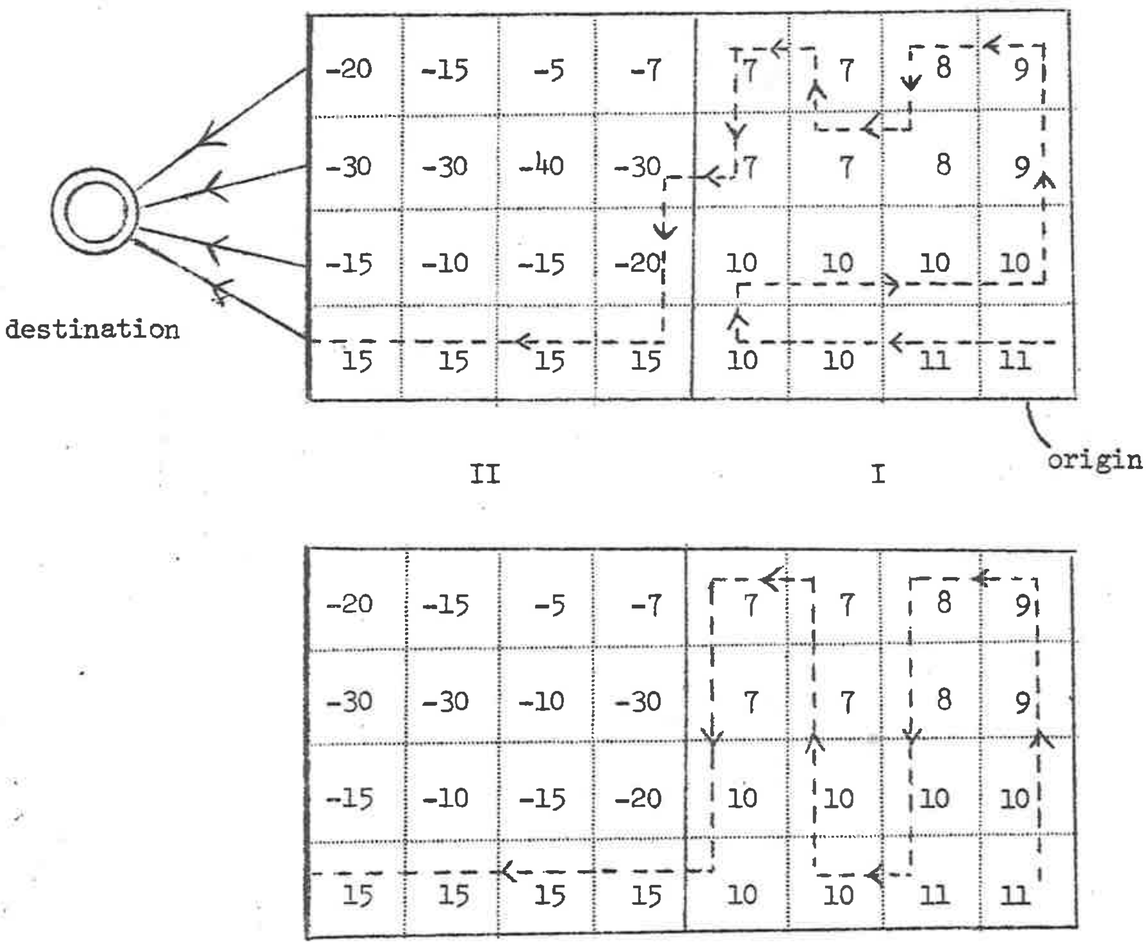


FIGURE 14

Boundary effects: the first path is non optimal for subareas I and II, but will be produced by independent optimization over subarea I. The second path is optimal for subareas I and II. (Numbers represent block values in dollars, dashed line is dredge path).

above example, we let d_i and o_i denote the pairs of blocks straddling the boundary between subareas I and II, with d_i being located in subarea I and o_i in subarea II. If d_i is the last block mined in subarea I, then o_i will be the first block to be mined in subarea II. Thus, denoting by D the destination node of the network representing subarea I, we assign to each link (d_i,D) a value of w_{O_i}, the worth of block o_i when w_{O_i} is negative and a value of zero otherwise. The resulting network for subarea I of the above example is shown in Fig.15.

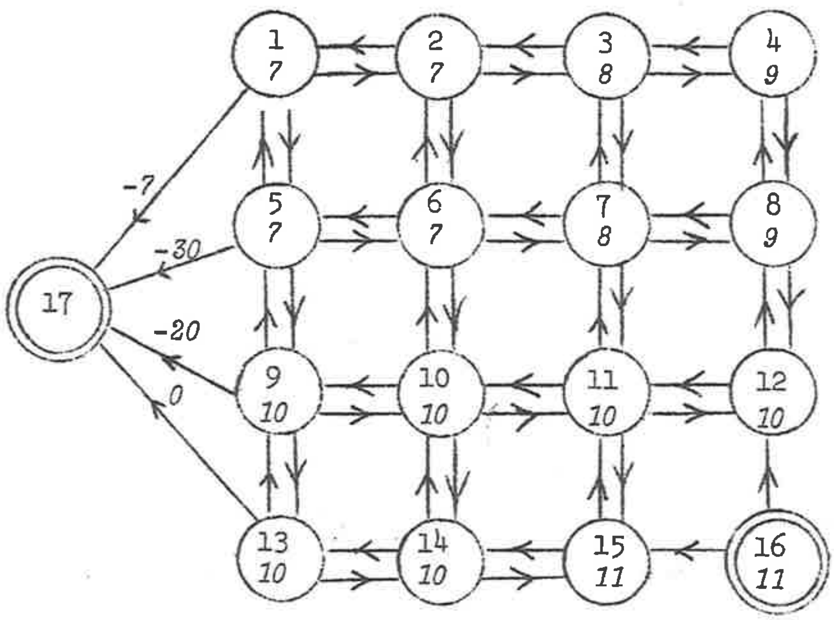


FIGURE 15

Network for subarea I. Values assigned to nodes 1,...,9 correspond to the values of blocks in subarea I. Values assigned to links (7,10), (8,10), (9,10) are determined by the values of the neighbouring blocks in subarea II.

This refinement does not guarantee an overall improvement but certainly gives a strong bias in that direction and was found satisfactory in all computer runs. A more sophisticated but time consuming approach is to look a whole subarea ahead. Thus we perform separate optimizations over subareas I and II for each pair of blocks (d_i, o_i) which straddle the boundary. We take the path which is best for subareas I and II as a whole, commit the dredge to the portion lying in subarea I and then examine paths through subarea II and its successor and so on.

The computer program incorporating either of these refinements is able to rapidly perform the chore of determining near optimal dredge paths through the deposit. This makes it ideally suited for use in feasibility studies where it is necessary to analyse the effect of several possible costs, dredge rates and production requirements. Also, depending on whether the information is available, we can examine the effect of changing the pond bottom or of dozing activities (to be discussed shortly). However, several other technical factors come into consideration when determining the precise details of the path necessary for the day to day operations of the dredge during the actual production stage. Most important is the modification required to the simple concept of the dredge movements as expressed by the network model. While this is entirely consistent with the information available during the feasibility study[1], in the actual production stage the mining front may be allowed to vary anywhere between 50m and 200m in order to pick up what are found to be locally rich pockets of ore and to avoid poorer patches.

To illustrate the use of the model, the results of 3 runs for the hypothetical deposit shown in Fig.11 are summarized in Appendix 3. The relevant block grades and tonnes values (for 400m×100m blocks) are also shown. In all runs the dredge rate was 4.6×10^6 tonnes/year and the discount rate was 20% per annum. Block sizes used in stages 1 and 2 of the optimization process were 800m×500m and 200m×100m respectively. Also the relevant cost data and present values of the final dredge paths are shown in Table 5.

A few further points will now be made with special reference to run 2. Firstly the final path shown in Appendix 3 was obtained by two applications of the basic heuristic algorithm. This is because the first application resulted in losses being incurred in some of the marginal southern subareas whereas a profit was expected to

---

[1]For the actual deposit for which the model was developed, grade and tonnage data was available only for 400×100m blocks.

TABLE 5

Cost Data and Present Values for Runs 1-3

Run	Break Even Block Grade (%Fe)	Total Cost/ Tonne Dredged (c/tonne)	Final Present Value ($'000's)
1	6.5%	31 c/tonne	3290
2	8.0%	38 c/tonne	2080
3	10.0%	48 c/tonne	920

be obtained during stage 1 of the algorithm. The information obtained in stage 2 was thus used to resequence production from the larger subareas and a second stage 2 was then performed to compute the final path. The separate results are compared in Table 6.

Finally it should be noted that there may be small differences between the local optima associated with using different partitions of the deposit in stages 1 and 2 of the heuristic algorithm. Table 7 gives a comparison of the results for four such partitioning schemes applied to run 2. Again the differences in the final present value are usually due to remaining boundary effects and minor manual adjustments may be made to reduce them (for example in subareas (1,1) and (6,4) of Fig.II in Appendix 3)

TABLE 6

Comparison of Results obtained from Double Application of Heuristic Algorithm.

Step	1st Application			2nd Application		
	Subarea	Current Receipt ($'000's)	Cumul. Present Value ($'000's)	Subarea	Current Receipt ($'000's)	Cumul. Present Value ($'000's)
1	(1,1)	50	50	(1,1)	50	50
2	(2,1)	296	334	(2,1)	296	334
3	(3,1)	26	358	(3,1)	26	358
4	(3,2)	836	1109	(3,2)	836	1109
5	(2,2)	216	1281	(2,2)	216	1281
6	(2,3)	68	1332	(2,3)	68	1332
7	(3,3)	114	1416	(3,3)	114	1416
8	(3,4)	6	1421	(3,4)	6	1421
9	(4,4)	126	1511	(4,4)	122	1508
10	(5,4)	212	1656	(4,3)	62	1550
11	(5,3)	52	1686	(4,2)	186	1668
12	(4,3)	62	1720	(5,2)	201	1786
13	(4,2)	186	1817	(5,3)	55	1816
14	(5,2)	201	1914	(5,4)	207	1920
15	(6,2)	194	2001	(6,4)	179	2001
16	(6,3)	-52	1979	(6,3)	-43	1985
17	(6,4)	133	2032	(6,2)	184	2050
18	(7,4)	-12	2028	(7,2)	79	2076
19	(7,3)	-68	2007			
20	(7,2)	68	2028			

TABLE 7

Effect of Partition

Partition: Block Sizes (m)		Network Size Stage 1		Stage 2[a]		CP times (Secs)			Cumulative Present Value ($'000's)
Stage 1	Stage 2	No. Nodes	No. Links	No. Nodes	No. Links	Stage 1	Stage 2	Tot.	
800×400	200×100	27	84	17	52	94	22	116	2090
800×500	200×100	22	65	21	67	8	94	102	2080
1000×400	200×100	24	74	21	67	28	84	112	2060
1000×500	200×100	20	59	26	85	4	708	712	2060

[a]Typical sizes

(3.2.4) Branch and Bound

An important modification can be made to the heuristic algorithm so far described in that it is possible to introduce a simple branch and bound strategy which enables significant savings in CP time to be obtained in stage 2 of the algorithm. Since, in stage 2, the optimization is carried out over small separate subareas which do not require a long period of time to be dredged, discounting does not play a major role. Instead it is more important to just pick up as many positively valued blocks and avoid as many negatively valued ones as possible. In addition to the present value of the incumbent O-D chain, we therefore maintain the total "straight" value, that is the value corresponding to a zero discount rate. Let s* denote this straight value, (the asterisk signifies the incumbent 0-D chain). During the construction of each 0-D chain, subsequent lexicographically to the incumbent, we now keep track of the sum of all negatively valued blocks committed to its various subchains, and we let f denote the value of this variable. Furthermore, let P denote the sum of all positively valued blocks in the subarea, plus the maximal link value (d_1, D) when we use the single block look-ahead procedure, where d_1 ranges over all blocks on the final boundary of the subarea. If it now happens that $f < s^* - P$, then all 0-D chains which are completions of the current subchain, will have a worse straight value than that of the incumbent (s*). We therefore dismiss such chains from consideration, backtracking at least one step to avoid offending blocks, and then recontinue with the lexicographical enumeration procedure. A good early 0-D chain will thus eliminate many succeeding 0-D chains from explicit consideration.

When a complete 0-D chain is produced, it will now have at least as good a straight value as the incumbent. However, we still compute its present value and update the incumbent on this basis. The dramatic effect of this simple branch and bound strategy on CP times can be seen by comparing the times shown in Table 8 with those listed in Table 7.

TABLE 8

Effect of Branch and Bound Strategy on CP times

Partition: Block Sizes		CP times (Secs)			Cumulative Present Value ($'000's)
Stage 1	Stage 2	Stage 1	Stage 2	Total	
800×400m	200×100m	97	3	100	2080
800×500m	200×100m	8	6	14	2080
1000×400m	200×100m	28	9	37	2060
1000×500m	200×100m	4	28	32	2060

(3.2.5) Dozing Activities

Finally we shall show how the basic network model can be extended to allow dozing activities to be taken at least approximately into account. In many beach sand deposits high grades are predominant above the water table and low grades below it and it may therefore be beneficial to also incorporate dozers into the system. This enables richer ore from the upper levels to be pushed into a nearby dredge path and the lower non-paying portions to be left intact. The dual operation of dozers and dredges thus gives considerable flexibility in dealing with vertical variations in grade. They complement each other in that it is infeasible for dozers to operate below the water table and costly to raise the top of the dredge pond above it.

We will again illustrate the approach by referring to the deposit shown in Fig.11. Each block is now divided into the two levels surface - water table and water table - sea level and we assume that the following choices may now be made;

(i) dredge the whole block,

(ii) doze the upper level only,

(iii) do not mine the block.

The relevant grade and tonnage data are now shown separately for the upper and lower levels in Appendix 3 while Fig.16 gives a scatter plot of block grades (%Fe) for the upper level versus those of the lower level.

Material dozed must bear a dozing cost in addition to the straight dredging cost so that the profit earnt from dozing the upper level of an arbitrary block i now becomes

$$p_i = T_i^1 \left[6.00\ R\ \frac{g_i^1}{59} - d - z - c \right]$$

where g_i^1 and T_i^1 are the grade and tonnes values for the upper level, surface-water table, and z is the incremental dozing cost (in $\$|$tonne). In practice the dozing costs depend on the distance of push, the topography and other factors which affect the ease of operations. However, we shall only be able to take dozing into account approximately and for purposes of long term economic

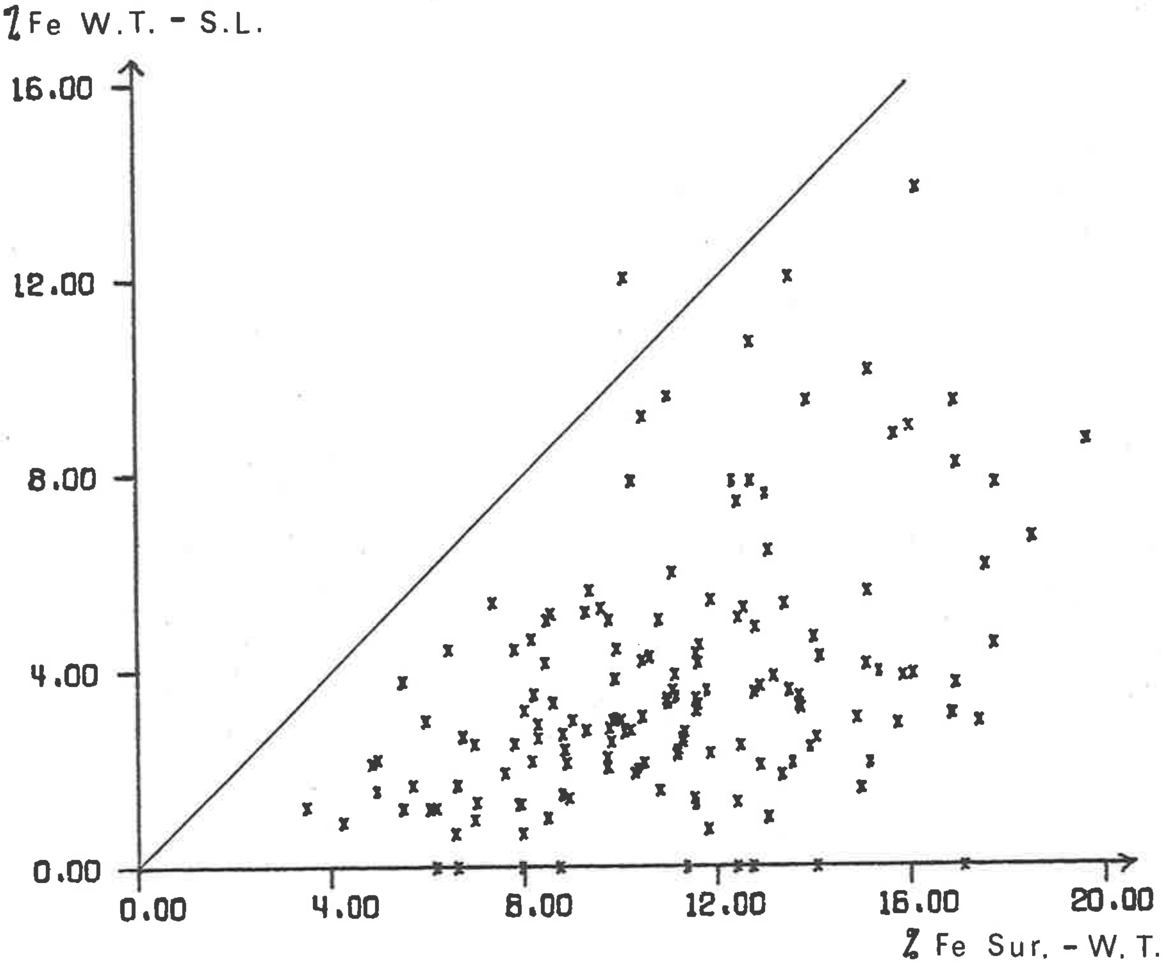


FIGURE 16

Scatter Plot:

%Fe Water Table - Sea Level vs. %Fe Surface - Water Table analysis it is reasonable to base costs on some fixed average length of push, for example 75-100 metres. A scatter plot of dozing profits p versus dredging profits w is given in Fig.17 for the case of $d = 8.0$ cents/tonne, $z = 11.5$ cents/tonne, $c = 30$ cents/tonne and $R = 47\%$.

The preferred a priori decisions for block i can now be stated as

(i) dredge the whole block if $w_i \geq p_i$ and $w_i \geq 0$,

(ii) doze the upper level if $p_i > w_i$ and $p_i \geq 0$,

(iii) do not mine the block otherwise.

Deviations from this preferred strategy may again be necessary due to the dredge's limited mobility.

The way in which we incorporate this information into the model is to simply take the blocks for which dozing is the first preference and instead of the full dredging value w_i, we now assign the values w_i-p_i[†] and then proceed exactly as before. Since this value is negative, the dredging of such blocks will be discouraged although, as mentioned above, it may occasionally be necessary for continuity reasons. In either case an additional profit of p_i will usually be earnt over and above the final value computed by the algorithm. This because if block i is not dredged, a value of zero is present in the computation of the dredge path, whereas an extra profit of p_i will be earnt from dozing it. On the other hand if it is

[†] Note that $p_i-w_i > 0$ is the opportunity loss associated with dredging block i instead of dozing it.

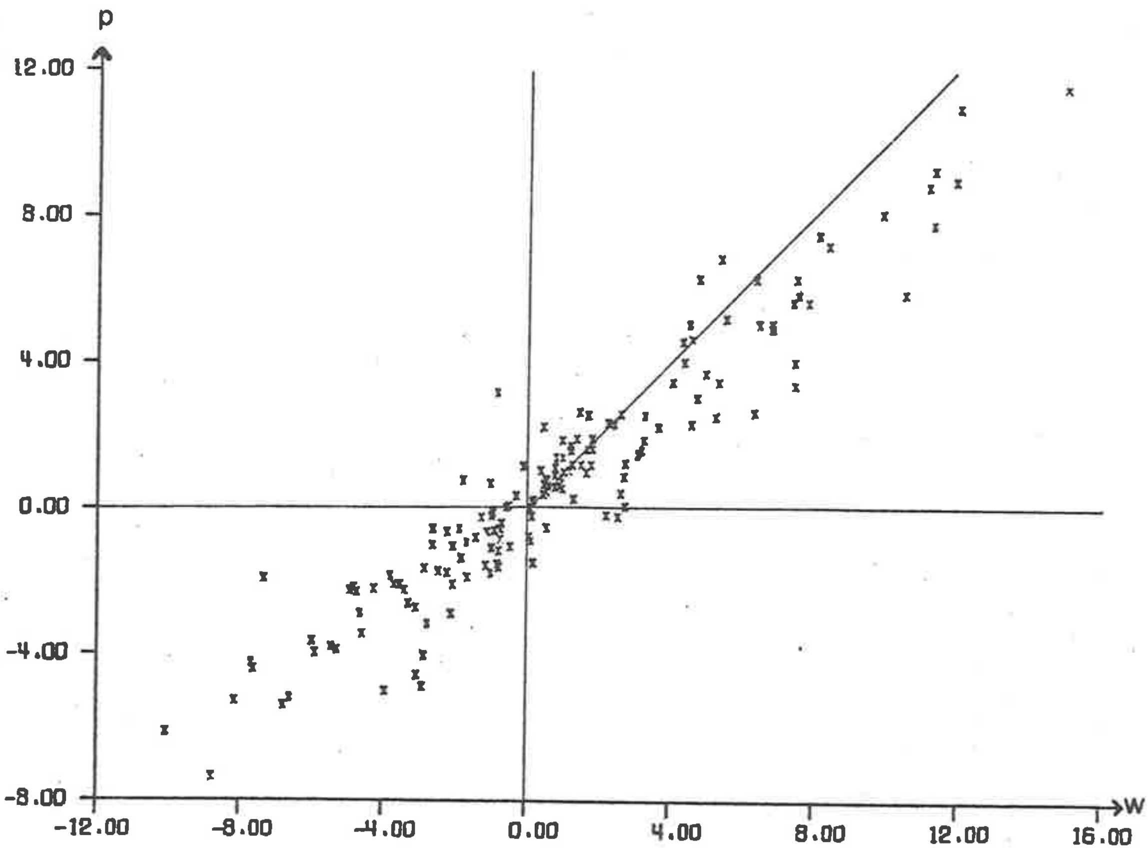


FIGURE 17

Scatter Plot:

Dozing Profits (p) vs. Dredging Profits (w). Units are $'0000.

dredged a value of $w_1 - p_1$ is present in the computation whereas in reality a profit of w_1 will be earnt.

The final connection of dozing activities with nearby dredging activities can be easily done manually. In this regard, it may be found on the first pass of the algorithm, that a few blocks which are anticipated to be dozed actually lie too far away from the dredge path. Practical limits for dozing may for instance be in the vicinity of 150-200 metres. If such blocks are not too insulated by poor areas, minor improvements may be made by replanning the operations using the full dredging values. While the final paths are thus not guaranteed to be globally optimal, they have been sensible in all cases and the approach has again proven especially suitable for rapidly analysing the several possibilities which must be considered in an actual feasibility study. The results of 9 such runs are now summarized in table 9. As before dredging and other costs have been chosen so that break even block grades (for dredging only) are 6.5%, 8.0% and 10.0%. In addition dozing costs have been chosen so that dozing is the first preference for 10%, 20% and 30% of blocks in the deposit. The corresponding values of these costs are also shown in table 9. Further details of the path for the run with break even block grades equal to 8.0% and dozing costs equal to 11.5 c/tonne are given in Appendix III.

Table 9

Present Values of Operations When Dozers are Used[1]

Break Even Block Grade	Percentage of Blocks Preferred for Dozing		
	10%	20%	30%
6.5%	3320 (11.0)	3390 (9.5)	3450 (8.0)
8.0%	2130 (13.5)	2150 (11.5)	2280 (10.0)
10.0%	930 (12.5)	1020 (9.5)	1050 (7.5)

[1]Numbers in parenthesis are dozing costs in cents|tonne.

CHAPTER 4

DISCUSSION

This thesis has employed optimization techniques to solve production planning problems in two quite different mining environments. They have been used to plan production from both an open-pit mine, where there are grade constraints on the mill feed, and from a beach sand deposit which is mined by a dredge. In each case the starting point has been a precise mathematical formulation of the problem, and, except for the material presented in section 2.4, computer programs have been written to test and implement these formulations.

The main contribution has been to demonstrate that the more advanced optimization techniques discussed in this thesis can in fact be successfully applied to mine production planning problems. The state of the art has been rather limited to date, linear programming being virtually the sole optimization technique used in applications described in the literature. By itself linear programming is unable to deal with the more detailed interactions which occur for example between the mine and the mill. Important nonlinearities, such as machine movements and mining constraints which have previously been largely ignored in the optimization process, can now be taken into account more satisfactorily. In practice, CP times often increase to unacceptable limits when we move from linear to nonlinear programming in which case the additional sophistication is not worth the increased computational effort. However, in the case of the mine production planning problems considered in this thesis, it has been shown that it is still possible to come up with computationally efficient algorithms by carefully adapting these more advanced techniques and by using the structure of the problem whenever possible.

The author has not had the opportunity to apply the techniques developed in Chapter 2 for mine-mill production planning to real world problems. Instead extensive computer investigations have been carried out using trial examples and the results have proven logical and in agreement with mining engineering practice. The work presented in Chapter 3 on dredging has been used in a feasibility study for an actual deposit. The resulting computer program enabled the production of acceptable plans and a precise evaluation of them in the space of minutes, whereas a manual approach by a mining engineer required several weeks. However the greatest advantage of the mathematical and computer approach was found to be in the ability to get virtually an instantaneous response to the effect of changing various parameters in the model. The great tedium of going through the whole process again by hand soon deters such a full investigation. The mining engineer is of course the one who must decide what parameters should be changed and how they should be altered, and his interaction with the computer is an integral part of the approach.

As mentioned earlier, the art of applying operations research techniques to mine production planning problems is still in an infant stage and there is considerable scope for further research and development. As a result of work presented in this thesis several areas can now be identified in which it would be particularly beneficial to carry out more research. In section 2.2, for instance, we dealt with medium term mine mill production planning in which the main concern was to minimise interblock machine movements, especially those of the slower moving draglines. Taking these into account led to a highly nonlinear mathematical program which could not be solved directly although an efficient heuristic algorithm was constructed and used to obtain acceptable solutions. While the heuristic approach is entirely consistent with the amount and quality of information likely to be available for input into the model, there is considerable worth in conducting further research into obtaining, if not globally optimal solutions, then improved locally optimal solutions to the mathematical program (1) - (29). In particular, avoiding the necessity for independently optimizing over successive blending intervals would be welcome from both the theoretical and practical viewpoints.

Branch and bound and mixed integer programming are themselves rapidly expanding fields and their application to mine production planning problems can be considered as only just begun. The work presented in sections 2.3 and 2.4 on branch and bound was based exclusively on the calculation of penalties. Selection rules based on penalties appeared very early in the development of branch and bound methods for the standard integer or mixed integer programming problem [10,5]. More recent work [31] has shown that several other heuristic rules may be more effective in terms of tree development and hence reduced CP times. Some of these rules, for example "priority ordering" [31] allow the user to specify parameters based on his knowledge of what variables have greatest bearing on the problem. Rules such as these have allowed significant savings to be obtained for certain types of problems. While the use of penalties has proven very effective in the production planning problems with mining constraints, the investigation of other alternatives also appears to be a fruitful area for further research. Priority ordering rules seem to be particularly promising as the mining engineer will usually have an idea of what blocks might be likely candidates to be mined out in order to get at good blocks which are initially enclosed. In other cases he may have some prior preferences as to how he would like the mining to proceed, and he would like to incorporate these in the model.

The underlying philosophy behind this thesis has been the belief that mathematical models are important tools which the mining engineer will be using more and more in conjunction with his own skills. The rapid advances in technology have made many mining companies aware of this and they are actively encouraging further research and development. It is clear that advanced optimization techniques, such as those discussed in this thesis, will play a key role in the solution of the models that are developed for the planning and control of their mining operations.

APPENDIX I

DATA AND RESULTS FOR MINE MILL PRODUCTION PLANNING

Tables

Figures

TABLE 1.1

Block Grades (%Ni) and Reserves ('000's tons) for Schedules 1-2 (Data Set I)[1]

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	46.5 0.95	57.6 0.91	67.7 0.95	45.4 1.02	47.5 0.93	62.8 0.93	67.9 1.01	59.1 0.98	47.0 0.91	62.7 1.04	65.3 0.92	45.1 1.02	46.8 1.03	46.2 1.00	53.0 1.01	66.8 1.01
2	64.3 0.98	52.9 0.93	48.2 0.97	60.3 0.94	54.1 0.91	63.3 0.91	45.1 0.99	51.8 1.03	60.3 1.01	45.7 1.04	46.7 1.02	53.0 0.92	52.2 0.94	61.0 1.03	46.1 0.92	54.3 0.97
3	66.1 0.97	53.6 1.02	53.4 1.02	52.8 0.93	59.5 1.02	58.7 1.04	50.1 0.98	52.3 1.04	50.5 1.05	56.1 0.91	46.4 1.01	62.8 1.04	63.0 0.95	58.6 0.98	52.8 0.97	48.6 0.94
4	50.0 1.04	68.0 1.05	64.7 0.91			48.6 0.95	50.4 0.97	59.4 1.01	54.6 0.99	60.5 0.95	50.3 1.01	63.5 0.94	54.1 1.01	58.3 1.04	65.9 0.93	47.0 0.97
5	50.3 0.96	46.6 1.04	56.3 0.96			62.7 1.02	47.9 1.01	50.8 0.92	46.0 1.04	64.2 0.99	55.7 0.92	47.8 0.95	65.6 0.97	56.4 0.99	63.7 0.97	54.0 0.93
6	57.0 1.05	55.3 1.05	47.2 0.92	50.4 1.05	67.1 1.01	67.2 0.92	54.4 0.92	57.0 1.05	62.1 0.90	47.5 1.03			63.6 0.98	48.7 0.95	54.0 0.94	63.4 1.03
7	51.5 1.00	59.1 0.97	47.3 0.92	64.1 0.93	52.8 0.98	68.4 1.01	46.0 0.93	64.7 0.94	63.7 1.02	63.2 1.00			49.4 0.97	58.7 1.01	46.0 1.04	64.7 0.93
8	61.7 0.96	56.6 0.94	65.4 1.01	57.5 0.93	50.2 0.98	52.7 1.03	63.1 0.90	64.4 0.98	54.2 0.92	62.3 0.94	60.8 1.02	63.2 0.91	55.9 1.02	50.0 0.92	50.2 0.92	46.6 1.00
9	45.2 0.97	65.6 0.99	59.0 0.91	63.1 1.01	63.1 0.91	52.2 0.91	47.7 0.92	48.2 0.91	51.0 0.99	48.8 0.95	61.6 0.95	67.7 1.04	56.1 1.01	60.4 0.98	54.1 0.93	49.4 1.00
10	56.4 0.93	51.3 0.92	65.8 0.92	62.5 1.03	53.1 0.99	46.6 0.96	67.6 0.95	54.5 0.97	52.1 1.05	60.0 0.93	63.5 0.90	46.2 1.02	48.1 1.02	51.9 0.94	49.6 1.00	46.2 0.98

[1]Upper figure is grade, lower figure is block reserves. Blocks (4,4), (4,5), (5,4), (5,5) and (6,11), (6,12), (7,11), (7,12) are assumed to be initially mined out. Also the dragline and front-end loader start in blocks (4,3) and (6,13) respectively. (These assumptions also apply to Tables 1.2 and 1.3).

TABLE 1.2

Block Grades (%Ni) and Reserves ('000's tons) for Schedules 3-4 (Data Set II)

	1	2	3	4	5	6	7	8	9
1	46.5 0.96	57.6 0.91	67.7 0.97	45.4 1.06	47.5 0.94	62.8 0.93	67.9 1.05	59.1 1.01	47.0 0.91
2	64.3 1.01	52.9 0.94	48.2 1.00	60.3 0.95	54.1 0.91	63.3 0.91	45.1 1.03	51.8 1.07	60.3 1.05
3	66.1 1.00	53.6 1.06	53.4 1.05	52.8 0.93	59.5 1.07	58.7 1.09	50.1 1.01	52.3 1.09	50.5 1.10
4	50.0 1.09	68.0 1.10	64.7 0.91			48.6 0.97	50.4 1.00	59.4 1.05	54.6 1.01
5	50.3 0.98	46.6 1.09	56.3 0.97			62.7 1.06	47.9 1.04	50.8 0.92	46.0 1.08
6	57.0 1.10	55.3 1.10	47.2 0.93	50.4 1.09	67.1 1.04	67.2 0.93	54.4 0.93	57.0 1.10	62.1 0.90
7	51.5 1.03	59.1 1.00	47.3 0.93	64.1 0.94	52.8 1.01	68.4 1.04	46.0 0.94	64.7 0.96	63.7 1.05
8	61.7 0.98	56.6 0.96	65.4 1.05	57.5 0.94	50.2 1.00	52.7 1.07	63.1 0.91	64.4 1.00	54.2 0.92
9	45.2 0.99	65.6 1.01	59.0 0.92	63.1 1.05	63.1 0.91	52.2 0.91	47.7 0.92	48.2 0.91	51.0 1.03
10	56.4 0.94	51.3 0.93	65.8 0.93	62.5 1.07	53.1 1.02	46.6 0.98	67.6 0.97	54.5 0.99	52.1 1.10

Dragline Area

	10	11	12	13	14	15	16
1	62.7 1.08	65.3 0.92	45.1 1.06	46.8 1.08	46.2 1.04	53.0 1.04	66.8 1.04
2	45.7 1.08	46.7 1.06	53.0 0.92	52.2 0.96	61.0 1.08	46.1 0.92	54.3 0.99
3	56.1 0.91	46.4 1.04	62.8 1.09	63.0 0.96	58.6 1.01	52.8 1.00	48.6 0.95
4	60.5 0.96	50.3 1.04	63.5 0.95	54.1 1.05	58.3 1.09	65.9 0.95	47.0 1.00
5	64.2 1.02	55.7 0.92	47.8 0.97	65.6 0.99	56.4 1.02	63.7 1.00	54.0 0.94
6	47.5 1.07			63.6 0.98	48.7 0.97	54.0 0.95	63.4 1.07
7	63.2 1.03			49.4 1.00	58.7 1.04	46.0 1.08	64.7 0.94
8	62.3 0.95	60.8 1.06	63.2 0.92	55.9 1.06	50.0 0.93	50.2 0.93	46.6 1.03
9	48.8 0.97	61.6 0.96	67.7 1.08	56.1 1.05	60.4 1.01	54.1 0.94	49.4 1.03
10	60.0 0.95	63.5 0.90	46.2 1.06	48.1 1.06	51.9 0.96	49.6 1.04	46.2 1.00

Front-end Loader Area

TABLE 1.3

Block Grades (%Ni) and Reserves ('000's tons) for Schedule 5 (Data Set III)

	1	2	3	4	5	6	7	8	9
1	62.1 0.99	54.5 0.98	68.6 0.98	62.5 0.98	63.5 0.98	64.3 0.96	64.3 0.94	54.7 0.94	55.0 1.01
2	63.5 0.90	63.2 0.91	62.7 0.95	63.2 0.91	66.6 0.96	63.9 0.93	57.9 0.95	58.2 0.93	59.0 0.97
3	59.3 0.96	61.7 0.91	49.1 0.94	67.2 0.96	58.2 0.98	65.1 0.95	62.9 0.98	62.3 0.92	59.1 0.95
4	54.2 0.94	57.7 0.90	63.6 0.91			64.5 0.97	54.4 1.00	56.8 0.97	74.6 0.97
5	58.7 0.96	57.9 0.95	49.4 0.96			62.7 0.92	55.3 0.93	59.0 0.93	64.8 0.92
6	63.6 0.96	55.4 0.95	67.3 0.90	70.9 0.94	48.3 0.97	55.1 0.94	49.3 0.94	63.3 0.92	65.3 0.92
7	62.5 0.99	66.4 0.99	50.6 0.91	59.5 0.99	64.2 0.98	61.7 0.96	57.5 0.97	57.9 0.99	56.7 0.94
8	57.2 0.97	52.7 1.00	54.0 0.92	49.1 0.95	61.5 1.02	57.5 0.99	64.7 0.94	65.1 0.97	59.9 0.98
9	60.8 0.99	49.7 0.93	61.8 0.99	65.4 0.92	50.2 0.94	67.4 1.01	68.4 0.92	65.1 0.97	67.8 0.97
10	59.9 0.96	62.1 0.94	58.9 0.94	56.0 1.00	54.5 0.94	53.4 0.92	45.6 0.93	56.2 0.93	56.8 0.92

Dragline Area

	10	11	12	13	14	15	16
1	70.5 0.95	54.7 0.98	74.2 0.95	72.8 0.94	61.0 0.96	54.8 1.00	56.0 0.97
2	49.7 0.93	51.0 0.95	58.4 1.01	69.3 0.90	66.7 0.95	53.7 0.98	64.0 0.95
3	61.9 0.92	60.4 0.92	52.9 0.91	55.8 0.99	75.8 0.91	67.3 0.92	61.7 0.94
4	47.3 0.91	59.8 0.95	78.6 0.98	69.2 0.98	52.9 0.98	70.3 0.94	57.7 0.96
5	60.4 0.92	61.9 0.98	54.9 0.97	72.5 0.89	46.5 0.96	57.1 0.99	61.3 0.96
6	60.3 0.93			71.0 0.98	46.9 0.99	60.1 0.92	59.6 1.01
7	58.6 0.91			54.8 0.95	59.0 0.90	64.7 0.97	64.4 0.95
8	64.2 0.95	62.7 0.94	73.7 0.99	66.1 0.99	58.8 0.92	46.2 0.99	55.7 0.97
9	66.3 0.96	80.2 0.92	59.0 0.97	52.6 0.97	66.2 0.98	61.9 0.96	63.0 0.94
10	61.6 0.92	51.7 0.99	54.4 1.00	59.3 0.95	73.3 0.96	58.2 0.96	61.6 0.93

Front-end Loader Area

TABLE 1.4

Penalty factors $F_k(i,j)$ and a

Preferred Machine Path (Dragline Area).

	1	2	3	4	5	6	7	8	9
1	4.9	4.4	3.8	3.5	3.5	3.8	4.3	4.9	5.7
2	4.3	3.5	3.0	2.5	2.5	2.9	3.5	4.3	5.1
3	3.8	2.9	2.1	1.6	1.6	2.1	2.9	3.8	4.7
4	3.5	2.5	1.6			1.6	2.5	3.5	4.5
5	3.5	2.5	1.6			1.6	2.5	3.5	4.5
6	3.8	2.9	2.1	1.6	1.6	2.1	2.9	3.8	4.7
7	4.3	3.5	2.9	2.5	2.5	2.9	3.5	4.3	5.1
8	4.9	4.3	3.8	3.5	3.5	3.8	4.3	4.9	5.7
9	5.7	5.1	4.7	4.5	4.5	4.7	5.1	5.7	6.4
10	6.5	6.0	5.7	5.5	5.5	5.7	6.0	6.5	7.1

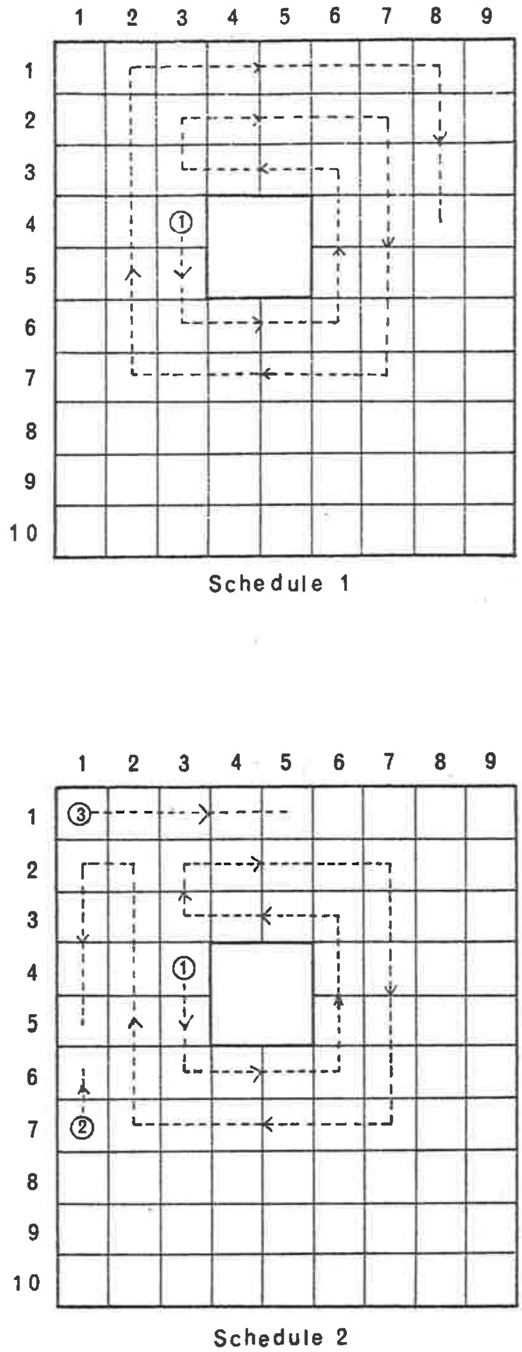
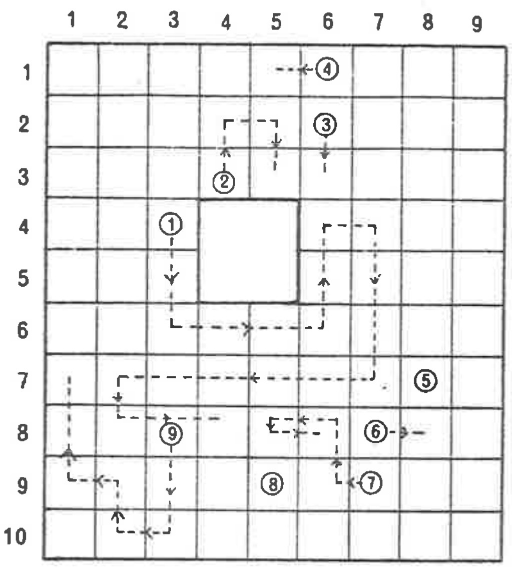
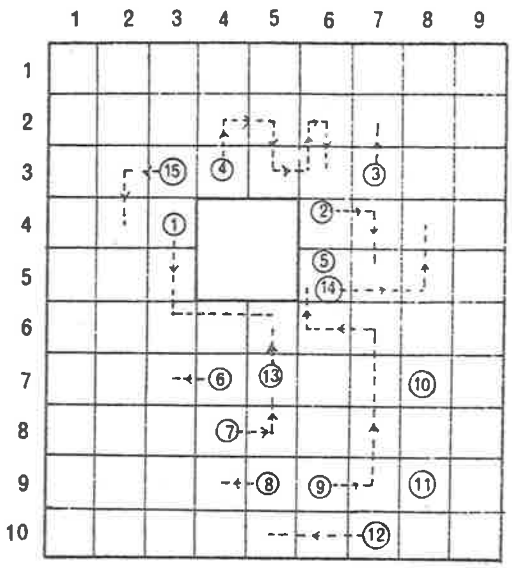


FIGURE I

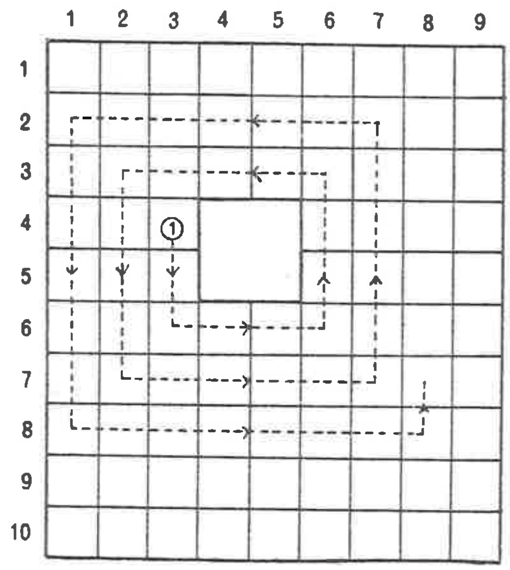
Dragline Movements for Schedules 1-2



Schedule 3



Schedule 4



Schedule 5

FIGURE II.

Dragline Movements for Schedules 3-5

APPENDIX II

CONTRACTED CALCULATIONS OF QUANTITIES REQUIRED BY THE ALGORITHM FOR THE MULTIPERIOD MODEL OF SECTION 2.4

As with the simplex multipliers, $\underset{\sim}{\Pi}'$ the transforms, with respect to basis $\mathcal{B}$, of the non-basic costs, the right hand side elements and an arbitrary column or row of the original constraint matrix $\mathcal{A}$ of the multiperiod planning model (78) - (79) can all be computed in terms of the submatrices $B_1, B_{11}, \ldots B_{n1}$ and $V_1, \ldots, V_n$ which form the compact representation of $\mathcal{B}$ and its inverse. The relevant computations are developed by using the appropriate partitioning schemes of $\mathcal{A}$, $(\mathcal{B}\mathcal{J})^{-1}$ and $\mathcal{J}$ as shown in section 2.4 and are summarized in this appendix.

Firstly consider the transform $(\bar{\mathcal{A}})_{\cdot s} = \mathcal{B}^{-1}(\mathcal{A})_{\cdot s}$ of the s-th column, $(\mathcal{A})_{\cdot s}$ of $\mathcal{A}$. When $(\mathcal{A})_{\cdot s} \in S_t$, for $t \geq 1$, we may write

$$(\mathcal{A})_{\cdot s} = \left[\begin{array}{c} \underset{\sim}{a} \\ \hline 0 \\ \vdots \\ 0 \\ \underset{\sim}{b} \\ 0 \\ \vdots \\ 0 \end{array}\right] \quad \text{(period } t\text{)}$$

where $\underset{\sim}{a}$ is a unit or zero vector. This implies that

$$(\bar{\mathcal{A}})_{.s} = \begin{bmatrix} V_1\underset{\sim}{z} \\ \vdots \\ V_{t-1}\underset{\sim}{z} \\ V_t\underset{\sim}{z}+\underset{\sim}{w} \\ V_{t+1}\underset{\sim}{z} \\ \vdots \\ V_n\underset{\sim}{z} \\ \hdashline \underset{\sim}{z} \end{bmatrix}$$

where $\underset{\sim}{w} = B_{t1}^{-1}\underset{\sim}{b}$

and $\underset{\sim}{z} = B^{-1}(\underset{\sim}{a} - E_{t1}\underset{\sim}{w})$.

However when $(\mathcal{A})_{.s} \in S_0$, $(\bar{\mathcal{A}})_{.s}$ has the simpler form

$$(\bar{\mathcal{A}})_{.s} = \begin{bmatrix} V_1\underset{\sim}{z} \\ \vdots \\ V_n\underset{\sim}{z} \\ \hdashline \underset{\sim}{z} \end{bmatrix}$$

where $\underset{\sim}{z} = B^{-1}\underset{\sim}{a}$.

The values of the basic variables may be maintained from iteration to iteration. Alternatively we can directly transform the right hand side

$$\underset{\sim}{b} = \left[\begin{array}{c} R \\ \hdashline \underset{\sim}{b}_1 \\ \vdots \\ \underset{\sim}{b}_n \end{array}\right]$$

of (78) - (79) to obtain

$$\bar{\underset{\sim}{b}} = \mathcal{B}^{-1}\underset{\sim}{b} = \left[\begin{array}{c} \bar{\underset{\sim}{b}}_1 + V_1\underset{\sim}{z} \\ \vdots \\ \bar{\underset{\sim}{b}}_n + V_n\underset{\sim}{z} \\ \hdashline \underset{\sim}{z} \end{array}\right]$$

where $\qquad \bar{\underset{\sim}{b}}_t = B_{t1}^{-1}\, \underset{\sim}{b}_t$

and $\quad \underset{\sim}{z} = B^{-1}[\underset{\sim}{R} - (E_{11}\bar{\underset{\sim}{b}}_1 + \ldots + E_{n1}\bar{\underset{\sim}{b}}_n)].$

Together with the $\underset{\sim}{\Pi}'$ this gives us all the information needed for the primal simplex algorithm. For the dual simplex algorithm, (which is used to solve all but the first node of each solution tree), it is also necessary to compute the transform, $(\bar{\mathcal{A}})_{r.}$, of an arbitrary row r of $\mathcal{A}$. Since $\bar{\mathcal{A}} = \mathcal{B}^{-1}\mathcal{A} = \mathcal{J}(\mathcal{B}\mathcal{J})^{-1}\mathcal{A}$, we first compute $(\mathcal{B}\mathcal{J})^{-1}\mathcal{A}$ obtaining

$$(\mathcal{B}\mathcal{J})^{-1}\mathcal{A} = \left[\begin{array}{c:cccc} 0 & \bar{B}_1 & & & \\ & & \bar{B}_2 & & \\ & & & \ddots & \\ 0 & & & & \bar{B}_n \\ \hdashline B^{-1}E_0 & B^{-1}(E_1 - E_{11}\bar{B}_1) & \ldots & & B^{-1}(E_n - E_{n1}\bar{B}_n) \end{array}\right]$$

where $\bar{B}_t = B_{t1}^{-1}\, B_t$. Now consider the rth row, $(\mathcal{J})_{r.}$

of $\mathcal{J}$; when it lies in a

$$[0\ldots0\ \underbrace{I}_{\text{period } t}\ 0\ldots0 \,\vdots\, V_t]$$

portion of $\mathcal{J}$, $(\mathcal{J})_{r.}$ has the form

$$[0\ldots0\ \underset{\sim}{e}'0\ldots0 \,\vdots\, \underset{\sim}{v}']$$

where $\underset{\sim}{e}'$ is a unit vector, and hence

$$(\overline{\mathcal{A}})_{r.} = [\overline{\underset{\sim}{v}}'E_0 \,\vdots\, \overline{\underset{\sim}{v}}'\ (E_1-E_{11}\overline{B}_1)\ldots\overline{\underset{\sim}{v}}'(E_t-E_{t1}\overline{B}_t) + \underset{\sim}{e}'\overline{B}_t\ldots$$
$$\ldots\overline{\underset{\sim}{v}}(E_n-E_{n1}\overline{B}_n)]$$

where $\overline{\underset{\sim}{v}}' = \underset{\sim}{v}'B^{-1}$. When row r lies in the portion $[0\ldots0 \,\vdots\, I]$ of $\mathcal{J}$, $(\mathcal{J})_{r.}$ has the form $[0\ldots0 \,\vdots\, \underset{\sim}{e}']$ where $\underset{\sim}{e}'$ is a unit vector, and

$$(\overline{\mathcal{A}})_{r.} = [\underset{\sim}{\beta}'E_0 \,\vdots\, \underset{\sim}{\beta}'(E_1-E_{11}\overline{B}_1)\ldots\underset{\sim}{\beta}'(E_n-E_{n1}\overline{B}_n)]$$

where $\underset{\sim}{\beta}' = \underset{\sim}{e}'B^{-1}$.

As mentioned in section 2.4, the simple structure of $E_0, E_1, \ldots, E_n$ and $E_{11}, \ldots, E_{n1}$ enables very fast computation of all the above quantities. In addition efficient product form representations of B^{-1}, $B_{11}^{-1}, \ldots, B_{n1}^{-1}$ are suitable for use in all the calculations.

APPENDIX III

DATA AND RESULTS FOR DREDGING

Tables

Figures

TABLE 3.1

Block Grades and Tonnes Values (Surface - Sea Level)[1]

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1		180 11.7	104 4.3																	
2		139 9.2	286 7.3	210 8.1	204 8.5	102 9.9														
3			185 5.6	467 13.1	424 10.7	161 8.7														
4				119 11.0	476 13.3	420 11.2	314 14.6	164 13.6	146 7.6	99 11.6	95 8.4	142 8.1	215 12.0	180 9.5	46 13.7					
5					217 10.5	341 15.3	331 15.2	266 14.2	292 15.6	296 10.6	231 7.4	321 7.7	283 7.4	327 9.4	208 14.9	237 4.5	173 4.0	262 5.7		
6					120 3.4	295 12.9	339 14.9	339 17.3	339 9.6	305 13.2	266 4.1	289 4.7	289 7.6	256 7.3	213 10.5	60 11.4	81 3.6	93 6.4		
7						152 12.3	223 9.3	334 11.3	224 9.7	302 11.5	265 8.7	214 7.2	342 10.0	291 6.8	209 10.7	193 9.7	98 9.7	89 2.8		
8						56 4.4	232 9.1	307 8.5	300 8.7	313 7.6	214 6.9	240 5.5	347 8.1	388 7.5	304 8.8	210 13.3	233 12.1	454 7.5	423 8.7	
9							199 8.5	373 9.8	333 12.8	279 8.1	300 8.4	285 9.3	391 9.5	345 5.9	258 8.6	215 12.6	399 10.8	365 5.6	499 9.9	
10								336 10.0	397 11.4	295 7.3	327 6.0	298 9.9	404 7.5	372 4.9	461 7.4	315 12.2	457 8.8	362 5.9	405 6.6	403 10.4
11								303 8.4	366 12.8	292 7.5	356 6.7	286 4.1	373 8.2	423 5.1	484 8.8	443 10.1	445 7.9	440 4.3	488 6.7	570 8.4
12								217 10.4	448 11.8	335 8.3	373 7.2	308 2.4	342 6.5	397 4.0	507 7.2	542 10.9	498 7.2	491 6.0	516 11.0	567 7.7
13								221 6.3	446 9.1	382 7.5	304 6.6	289 3.1	418 8.6	477 6.0	544 6.9	609 8.1	495 6.4	444 3.8	465 5.3	597 8.2
14								263 8.8	506 10.0	412 6.7	377 8.3	379 3.9	437 7.5	490 5.2	458 6.4	530 8.0	481 11.3	452 3.3	525 6.1	

[1]Data is for 400m×100m blocks; upper figure is tonnes×10^3 value, lower figure is grade (%Fe). To determine dredge paths each 400m×100m block was divided into 200m×100m or 100m×100m blocks.

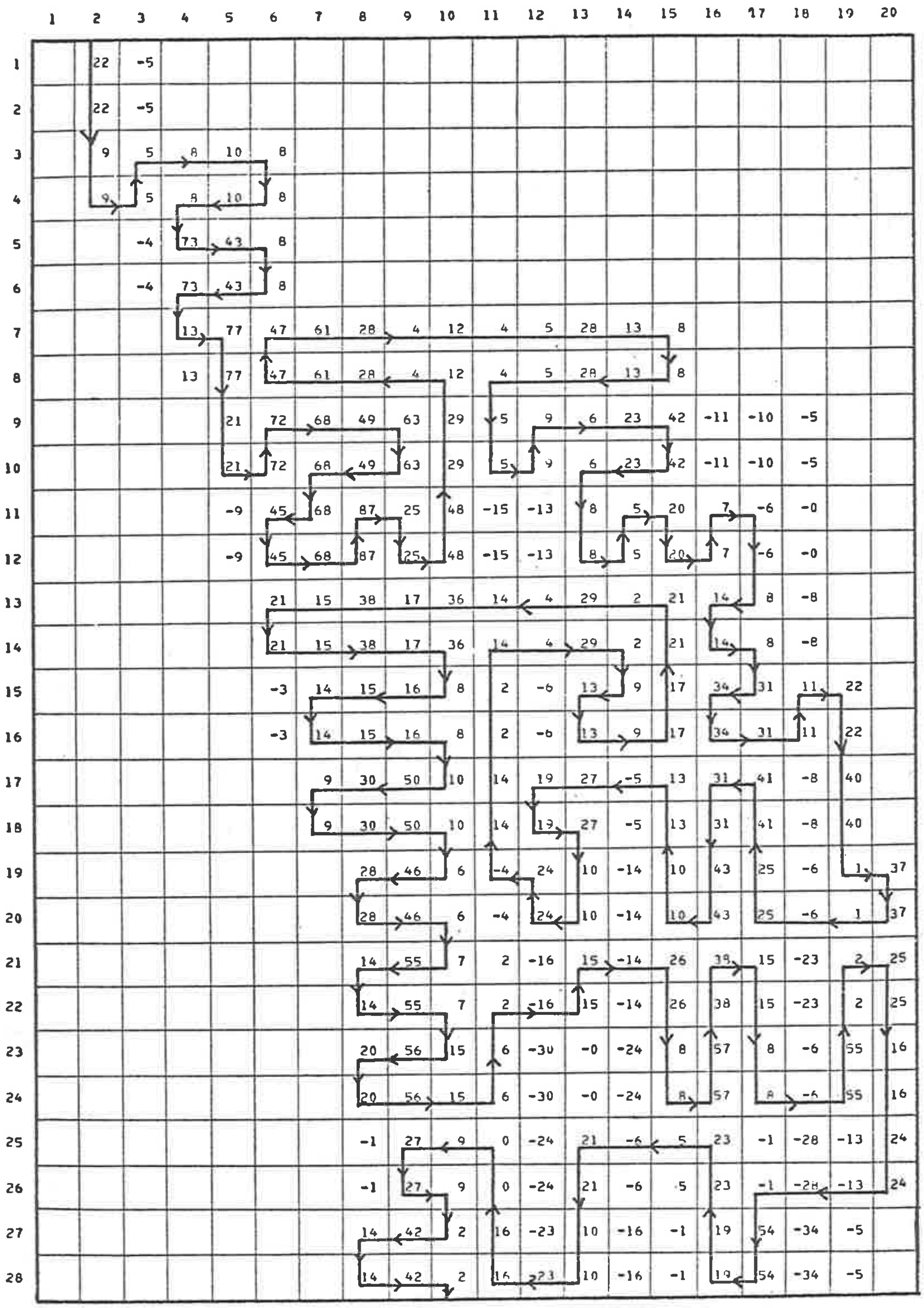


FIGURE I

Path 1: Break Even Block Grades = 6.5%. Block values are the profits (in $'000's) from dredging the block.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
1		16	-9																		
2		16	-9																		
3		4	-5	1	2	5															i
4		4	-5	1	2	5															
5			-10	56	28	3															
6			-10	56	28	3															ii
7				8	60	32	49	22	-1	8	1	0	20	7	6						
8				8	60	32	49	22	-1	8	1	0	20	7	6						
9					13	60	56	39	53	18	-4	-2	-4	11	34	-20	-17	-14			
10					13	60	56	39	53	18	-4	-2	-4	11	34	-20	-17	-14			iii
11					-13	34	56	75	13	37	-25	-23	-3	-4	13	5	-8	-4			
12					-13	34	56	75	13	37	-25	-23	-3	-4	13	5	-8	-4			
13						15	7	26	9	25	5	-4	16	-8	14	8	4	-11			
14						15	7	26	9	25	5	-4	16	-8	14	8	4	-11			iv
15						-5	6	4	5	-3	-6	-14	1	-5	6	27	23	-5	7		
16						-5	6	4	5	-3	-6	-14	1	-5	6	27	23	-5	7		
17							2	16	38	0	3	9	13	-17	4	24	27	-21	23		
18							2	16	38	0	3	9	13	-17	4	24	27	-21	23		v
19								16	32	-5	-15	14	-5	-27	-6	32	9	-18	-14	23	
20								16	32	-5	-15	14	-5	-27	-6	32	9	-18	-14	23	
21								3	42	-4	-11	-27	2	-30	9	22	-1	-38	-15	5	
22								3	42	-4	-11	-27	2	-30	9	22	-1	-38	-15	5	vi
23								12	40	3	-7	-41	-12	-38	-10	38	-9	-24	37	-4	
24								12	40	3	-7	-41	-12	-38	-10	38	-9	-24	37	-4	
25								-9	11	-5	-10	-34	6	-23	-14	1	-19	-44	-30	3	
26								-9	11	-5	-10	-34	6	-23	-14	1	-19	-44	-30	3	vii
27								5	24	-13	2	-37	-5	-33	-18	0	37	-50	-24		
28								5	24	-13	2	-37	-5	-33	-18	0	37	-50	-24		
			i					ii					iii					iv			

FIGURE II

Path 2 : Break Even Block Grades = 8.0%

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1		7	-14																	
2		7	-14																	
3		-3	-19	-9	-7	-0														
4		-3	-19	-9	-7	-0														
5			-19	34	8	-5														
6			-19	34	8	-5														
7				3	37	12	34	14	-8	4	-4	-6	10	-2	4					
8				3	37	12	34	14	-8	4	-4	-6	10	-2	4					
9					3	43	41	26	39	4	-15	-18	-17	-4	24	-31	-25	-27		
10					3	43	41	26	39	4	-15	-18	-17	-4	24	-31	-25	-27		
11					-19	20	40	58	-3	23	-37	-37	-16	-16	3	2	-12	-8		
12					-19	20	40	58	-3	23	-37	-37	-16	-16	3	2	-12	-8		
13						8	-4	10	-2	10	-8	-14	0	-22	4	-2	-1	-15		
14						8	-4	10	-2	10	-8	-14	0	-22	4	-2	-1	-15		
15						-7	-5	-11	-9	-18	-16	-26	-16	-23	-9	17	12	-27	-13	
16						-7	-5	-11	-9	-18	-16	-26	-16	-23	-9	17	12	-27	-13	
17							-7	-1	22	-13	-11	-5	-5	-33	-8	13	8	-39	-1	
18							-7	-1	22	-13	-11	-5	-5	-33	-8	13	8	-39	-1	
19								-0	13	-19	-31	-0	-24	-45	-28	17	-13	-36	-33	4
20								-0	13	-19	-31	-0	-24	-45	-28	17	-13	-36	-33	4
21								-12	24	-18	-28	-40	-16	-50	-14	1	-22	-59	-38	-22
22								-12	24	-18	-28	-40	-16	-50	-14	1	-22	-59	-38	-22
23								2	19	-13	-25	-55	-29	-57	-34	12	-33	-47	12	-31
24								2	19	-13	-25	-55	-29	-57	-34	12	-33	-47	12	-31
25								-19	-10	-23	-25	-48	-14	-46	-40	-28	-43	-65	-52	-26
26								-19	-10	-23	-25	-48	-14	-46	-40	-28	-43	-65	-52	-26
27								-7	-0	-33	-16	-55	-26	-56	-39	-25	14	-72	-49	
28								-7	-0	-33	-16	-55	-26	-56	-39	-25	14	-72	-49	

FIGURE III

Path 3 : Break Even Block Grades = 10.0%

TABLE 3.2

Summary of Results for Path 1:Break Even Block Grades=6.5%

Year	Av. Grade Mined (%Fe)	Tonnes Mined ($\times 10^6$)	Tonnage Concentrate[1] Produced ($\times 10^5$)	Cumul. Present Worth ($'000's)
1	12.2	4.6	9.5	1109
2	11.2	4.6	8.8	1910
3	9.0	4.6	7.1	2270
4	8.8	4.6	6.8	2540
5	9.4	4.6	7.3	2810
6	9.2	4.6	7.2	3030
7	8.5	4.6	6.6	3160
8	7.8	4.6	6.1	3240
9	7.9	3.4	4.5	3290

[1]Assaying 59% Fe, overall recovery 47%.

TABLE 3.3

Summary of Results for Path 1:Break Even Block Grades=8.0%

Year	Av. Grade Mined (%Fe)	Tonnes Mined ($\times 10^6$)	Tonnage Concentrate Produced ($\times 10^5$)	Cumul. Present Worth ($'000's)
1	12.6	4.6	9.8	890
2	11.3	4.6	8.8	1450
3	9.8	4.6	7.6	1700
4	9.6	4.6	7.4	1880
5	8.8	4.6	6.9	1950
6	9.1	4.6	7.1	2040
7	9.7	1.4	2.3	2080

TABLE 3.4

Summary of Results for Path 3:Break Even Block Grades=10.0%

Year	Av. Grade Mined (%Fe)	Tonnes Mined ($\times 10^6$)	Tonnage Concentrate Produced ($\times 10^5$)	Cumul. Present Worth ($'000's)
1	13.1	4.6	10.0	610
2	10.6	4.6	8.2	690
3	11.7	4.6	9.1	930
4	9.7	2.0	3.3	920

TABLE 3.5

Block Grades and Tonnes Values (Surface - Water Table)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1		143 12.7	67 6.1																	
2		99 11.6	212 8.6	154 10.1	137 11.0	74 12.9														
3			112 6.5	359 15.9	301 14.1	115 11.2														
4				93 14.1	316 17.7	295 14.0	210 18.5	121 17.4	82 11.3	89 12.8	65 11.2	97 10.5	162 14.9	139 10.8	37 17.1					
5					143 11.0	257 17.8	253 16.9	182 17.0	215 16.2	216 12.6	153 9.8	196 9.3	187 8.6	214 10.3	194 15.8	195 5.0	112 5.5	234 6.0		
6					56 6.6	225 15.1	260 17.6	265 19.7	180 13.4	227 13.6	139 7.0	149 5.5	190 10.5	169 8.5	172 10.1	55 12.4	47 6.2	74 8.0		
7						65 15.2	153 13.1	257 12.5	173 11.9	236 13.7	176 11.1	160 8.9	222 12.8	179 8.2	134 12.4	93 13.1	45 16.9	37 6.6		
8						28 8.8	130 13.2	195 11.6	167 12.8	171 10.5	158 8.3	135 7.8	209 10.6	202 9.8	187 11.6	130 16.0	140 13.9	249 11.0	265 12.5	
9							149 11.4	284 11.8	262 15.1	182 9.4	193 11.1	184 13.3	265 11.1	196 8.0	165 11.6	120 15.7	224 16.9	206 8.2	340 13.6	
10								260 11.6	290 15.0	205 9.3	232 7.9	171 10.5	269 10.3	236 7.0	288 10.2	237 12.7	277 11.7	207 8.3	257 7.8	279 13.9
11								227 9.9	268 16.1	195 9.8	251 8.9	173 5.7	242 11.3	242 7.0	353 11.6	260 14.1	273 11.3	237 6.6	316 7.4	391 11.2
12								147 13.7	333 15.2	212 11.2	251 9.7	161 3.5	186 8.4	221 6.2	333 9.7	396 13.8	299 10.0	301 6.8	387 13.5	325 12.4
13								112 11.8	242 12.4	251 10.4	242 7.9	186 4.3	298 11.5	316 8.5	391 8.2	391 9.6	279 9.0	279 4.9	275 7.6	428 10.8
14								138 11.9	268 15.4	222 9.9	187 12.9	169 8.0	283 9.9	300 6.8	288 8.9	346 9.9	324 13.0	236 5.0	288 8.8	

TABLE 3.6

Block Grades and Tonnes Values (Water Table - Sea Level)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1		37 7.8	37 1.2																	
2		41 3.3	74 3.4	56 2.7	68 3.4	28 2.1														
3			74 4.4	109 3.8	123 2.6	46 2.4														
4				26 0.0	160 4.5	124 4.7	104 6.7	43 2.9	64 2.8	9 0.0	30 2.3	45 3.1	54 3.0	41 5.0	9 0.0					
5					74 9.6	84 7.8	78 9.5	84 8.2	77 13.8	80 5.3	79 2.5	125 5.2	96 5.1	113 7.9	14 2.9	42 2.2	61 1.2	28 3.0		
6					64 .7	70 5.6	78 6.1	74 8.7	159 5.3	78 12.0	127 .9	140 3.8	99 2.1	87 5.0	41 12.0	5 0.0	33 0.0	19 0.0		
7						87 10.1	69 1.0	77 7.4	51 2.3	66 3.5	89 3.9	54 2.4	121 4.9	112 4.6	75 7.9	100 6.4	54 3.7	52 0.0		
8						28 0.0	102 3.9	112 3.2	132 3.5	142 4.2	57 2.9	105 2.5	138 4.3	186 5.0	116 4.3	80 8.9	93 9.5	205 3.3	158 2.5	
9							50 0.0	89 3.6	71 4.1	97 5.6	106 3.6	101 1.9	126 6.0	150 3.2	93 3.4	95 8.8	175 3.1	158 2.2	159 2.1	
10								75 4.1	107 1.6	90 2.8	94 1.3	127 9.2	135 1.9	136 1.3	173 2.8	79 10.7	180 4.5	155 2.6	148 4.4	124 2.4
11								76 3.8	98 3.9	97 2.8	106 1.4	113 1.7	132 2.5	182 2.5	130 1.3	183 4.3	172 2.6	204 1.7	172 5.4	179 2.3
12								70 3.3	115 2.1	123 3.5	123 2.0	147 1.2	156 4.2	176 1.2	175 2.2	147 3.2	199 3.0	190 1.5	129 3.6	242 1.3
13								110 .7	205 5.1	132 2.0	62 1.3	103 .9	121 1.4	161 1.0	153 3.5	218 5.3	216 3.0	165 2.1	190 1.9	169 1.5
14								126 5.4	238 4.0	190 3.0	190 3.7	210 .6	155 3.0	190 2.7	170 2.1	185 4.4	158 7.6	217 1.5	237 2.7	

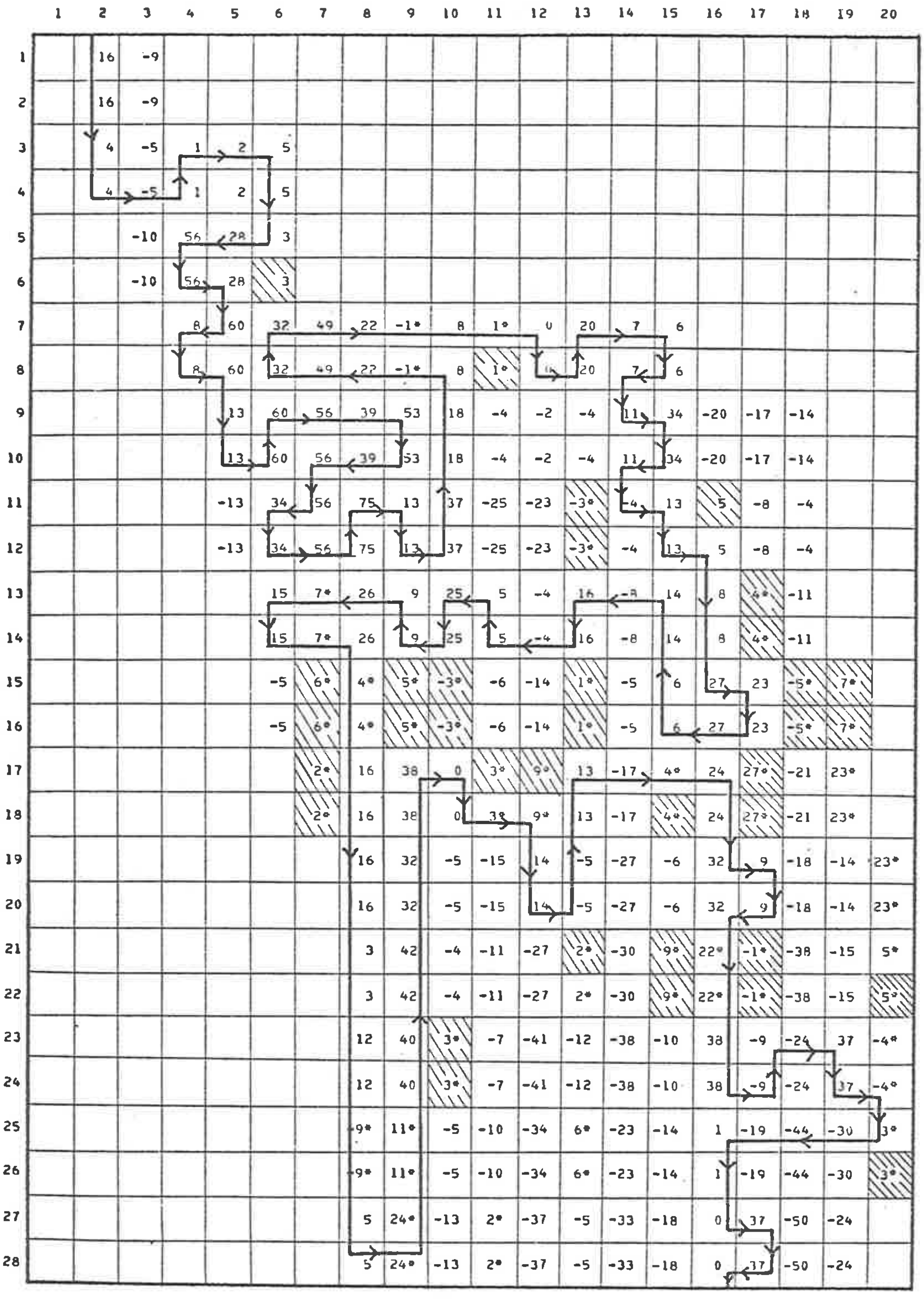


FIGURE IV

Path 4: Break Even Block Grades = 8.0%, Dozing Costs = 11.5 cents/tonne. Block Values are the profits (in $'000's) from dredging the block. An asterisk indicates that dozing is the first preference. The areas actually set aside for dozing are shaded.

TABLE 3.7

Summary of Results for Path 4: Break Even Block Grades = 8.0%, Dozing Costs = 11.5 c/tonne

Year	Av. Grade Mined (%Fe)	Tonnes Mined ($\times 10^6$)	Tonnage Concentrate Produced ($\times 10^5$)	Cumul. Present Worth ($'000's)
1	12.5	4.6	9.8	880
2	12.1	4.6	9.1	1470
3	11.5	4.6	8.1	1720
4	11.2	4.6	8.0	1950
5	11.0	4.6	8.1	2120
6	10.6	4.6	6.7	2140
7	10.6	0.5	0.8	2150

BIBLIOGRAPHY

1. Ahlbach, H., "Long-range Planning in Open-pit Mining," Management Science, Vol. B13, No.10, pp. B549-568, (1967).
2. Apex II - Reference Manual, Control Data Corporation, Minneapolis, U.S.A., (1973).
3. Azpeitia, A.G. and Riley, D., "The Longest Path in a Network," Journal of Mathematical Analysis and Applications, Vol.30, No.3, pp. 636-638, (1967).
4. Beale, E.M.L., "Mathematical Programming in Practice," Pitman, London, (1968).
5. Beale, E.M.L. and Small, R.E., "Mixed Integer Programming by a Branch and Bound Technique," Proceedings IFIP Congress 1965, Vol.2, ed. W.A. Kalenich (Spartan Press, Washington D.C. and Macmillan, London) pp. 450-451.
6. Boldt, J.R., "The Winning of Nickel," Methuen, London, (1967).
7. Coyle, R.G., "Review of the Literature on Operational Research in the Mining Industry," Transactions of the Institution of Mining and Metallurgy, Vol.78, Section A: Mining Industry, pp.A1-A9, (1969).
8. Dakin, R.J., "A Tree Search Algorithm for Mixed Integer Programming Problems," Computer Journal, Vol.8, pp.250-255, (1965).

9. Dantzig, G.B., "Linear Programming and Extensions," Princeton University Press, Princeton, (1963).

10. Driebeek, N.J., "An Algorithm for the Solution of Mixed Integer Programming Problems," Management Science, Vol.12, No.7, pp.576-587, (1966).

11. Faulkner, G.B., "Linear Programming applied to a Mining - Smelting Operation," Canadian Mining and Metallurgical Bulletin, Vol.60, pp.1297-1300, (1967).

12. Ford, L.R. and Fulkerson, D.R., "Flows in Networks", Princeton University Press, Princeton, (1962).

13. Fraser, S., "Computer-Aided Scheduling of Open-Pit Mining Operations," Proceedings Ninth International Symposium on Techniques for Decision Making in the Mineral Industry, Montreal, Canada, June 1970. In: The Canadian Institute of Mining and Metallurgy, CIM Special Vol.12, (1971).

14. Greenberg, H., "Integer Programming," Academic Press, New York, (1971).

15. Hewlett, R.F. and Redmon, D.E., "Design and Applications of Some Mathematical Models for Mine-Systems Analysis," Report of Investigations U.S. Bur. Mines, 6158, (1963).

16. Johnson, T.B., "Optimum Open Pit Mine Production Scheduling," Operations Research Center Report, ORC 68-11, University of California, Berkeley (Ph.D. Thesis), (1968).

17. Johnson, T.B., "Optimum Open-Pit Mine Production Scheduling," in: A Decade of Digital Computing in the Mineral Industry, ed. A Weiss (AIME Inc., New York), pp.539-562, (1969).

18. Johnson, T.B. and Sharp, W.R., "A Three- Dimensional Dynamic Programming Method for Optimal Ultimate Open Pit Design," Report of Investigations U.S. Bur. Mines, 7553, (1971).

19. Jutsum, P.J. and Milner, R.S., "A Linear Programming System for Bauxite Blending," private communication.

20. Kaas, L.M., "A Practical Production Scheduling Model for the Taconite Industry," Transactions of the American Institute of Mining, Metallurgical and Petroleum Engineers, Vol.235, pp.281-285, (1966).

21. Kaul, R.N., "An Extension of Generalized Upper Bounding Techniques for Linear Programming," Rept. ORC 65-27, Operations Research Center, University of California at Berkeley, (1965).

22. Kim, Y.C., "Mathematical Programming Analysis of Mine Planning Problems," Ph.D. Thesis, Pennsylvania State University, (1967).

23. Kirby, R.F., "Mathematical Models for Transportation Planning," Ph.D. Thesis, University of Adelaide, (1968).

24. Land, A.H. and Doig, A.G., "An Automatic Method of Solving Discrete Programming Problems," Econometrica, Vol.28, No.1, pp.30-44, (1960).

25. Lasdon, L.S., "Optimization Theory for Large Systems," MacMillan, London, (1970).

26. Lerchs, H. and Grossman, I.F., "Optimum Design of Open-Pit Mines," Transactions of the Canadian Institute of Mining and Metallurgy," Vol.68, pp. 17-24, (1965).

27. Lawler, E.H. and Wood, D.E., "Branch-and-Bound Methods-a Survey," Operations Research, Vol.14, No.4, pp.699-719, (1966).

28. Lutjen, G.P., "Nicaro Proves Lateritic Nickel can be Produced Commercially," Engineering and Mining Journal, Vol.155, No.6, pp.81-89, (1954).

29. Manula, C.B. and Kim, Y.C., "Optimum Production Planning," Transactions of the American Institute of Mining, Metallurgical and Petroleum Engineers, Vol.241, pp.22-26, (1968).

30. Mathias, A.J., "A Mine Production-Scheduling Model and Critical Path Analysis of Mine Development Work for Long-Range Mine Planning," Report of Investigations U.S. Bur. Mines, 6937, (1967).

31. Mitra, G., "Investigation of Some Branch and Bound Strategies for the Solution of Mixed Integer Linear Programs," Mathematical Programming, Vol.4, No.2, pp.155-170, (1973).

32. Pana, M.T., "The Simulation Approach to Open-Pit Design," Proceedings Fifth Annual Symposium on Computer Applications in the Mineral Industry, Tucson, Arizona, March 1965, 24 pp.

33. Pana, M.T., "Computer Applications in Mining," in: A Decade of Digital Computing in the Mineral Industry, ed. A. Weiss (AIME, New York) pp.539-562, (1969).

34. Pandit, S.N.N., "Some Observations on the Longest Path Problem," Operations Research, Vol.12, No.2, pp.361-364, (1964).

35. Redmon, D.E., "Determining Mine-Production Schedules by Linear Programming," Report of Investigations U.S. Bur. Mines, 6441, (1964).

36. Rogado, J.Q., "An Optimization Method for the Mining and Beneficiation of Ore Blocks," paper presented at the Twelfth International Symposium on Applications of Computers in the Mineral Industry, Golden, U.S.A., April 1974.

37. Sheinkin, M. and Julin, D.E., "Computer Simulation aids in Long-Range Production Planning at Climax," Mining Engineering, Vol.19, No.4, pp.76-81, (1967).

38. Tomlin, J.A., "Branch and Bound Methods for Integer and Non-Convex Programming," in: Integer and Non-linear Programming, ed. J. Abadie (North Holland, Amsterdam) pp.437-450, (1970).

39. White, C.G., and Peiker, E.W., "Analysis of Shaft and Level Development, Stope Production and Planning, and Ore Transportation for a Simulated Copper Mining Operation using Optimization Programming," Colorado School of Mines Quarterly, Vol.59, No.4, pp.969-995, (1964).

40. Williams, J.K., Smith, L. and Wells, M.P., "Planning of Underground Copper Mining," Proceedings of the Tenth International Symposium on the Application of Computer Methods in the Mineral Industry, Johannesburg, 1972, ed. M.D.G. Salamon and F.H. Lancaster (South African Inst. Mining and Metallurgy, J'burg) pp.251-254, (1973).

41. Yu, A.T., "Bed Blending and Quality Control," Proceedings Ninth International Symposium on Techniques for Decision Making in the Mineral Industry, Montreal, Canada, June 1970. In: The Canadian Institute of Mining and Metallurgy, CIM Special Vol.12, pp.416-420, (1971).