INTERFACES
Vol. 7, No. 2
February 1977

AN APPLICATION OF MIXED INTEGER PROGRAMMING IN A SWEDISH STEEL MILL

Carl-Henrik Westerberg*
Bengt Bjorklund*
and
Eskil Hultman**

ABSTRACT. The object is to make a tool for a Swedish steel company (Fagersta AB) to select appropriate ingredients for charges of high-grade steel, specified with regard to weight and percentage of certain chemical elements.

The problem is modelled as a traditional blending problem, with additional constraints expressing that some of the variables must be integer valued. The algorithm used was designed at the Contract Research Group for Applied Mathematics, Royal Institute of Technology in Stockholm. The implemented system is available from the time-sharing terminal of the company and has completely replaced the former manual calculations. About two runs are made every day and the results are directly used in the production process. A substantial improvement in profitability has been demonstrated. The cost for raw materials has been decreased by roughly 6%, which corresponds to $200,000 per year, and the planning work has been considerably simplified.

Background

When producing stainless steel in the HF (high frequency) furnaces of Fagersta AB, up to about 15 types of steel scraps and alloys are melted together. They will form a requested product (charge) where weight (about 10 tons) and analysis (i.e., percentage of nine different chemical elements) are to stay within specified limits. Since alloys often are 30 times as expensive as some sorts of steel scrap, it is of great importance that in each case those sorts and quantities of scrap be used which, together with indispensable alloys, give lowest possible total cost.

Besides alloys and four classes of stainless steel scrap, large pieces of steel (ingots, plate slabs, etc.) with weights of up to three tons are used, with each piece having its weight and analysis individually established.

Since about 150 different components ought to be considered at every charge-calculation and the possible combinations are practically unlimited, the manual planner must have a vast experience in order to come up with acceptable solutions. It has been observed, however, that the solutions often have been far from "optimal".

Problem definition

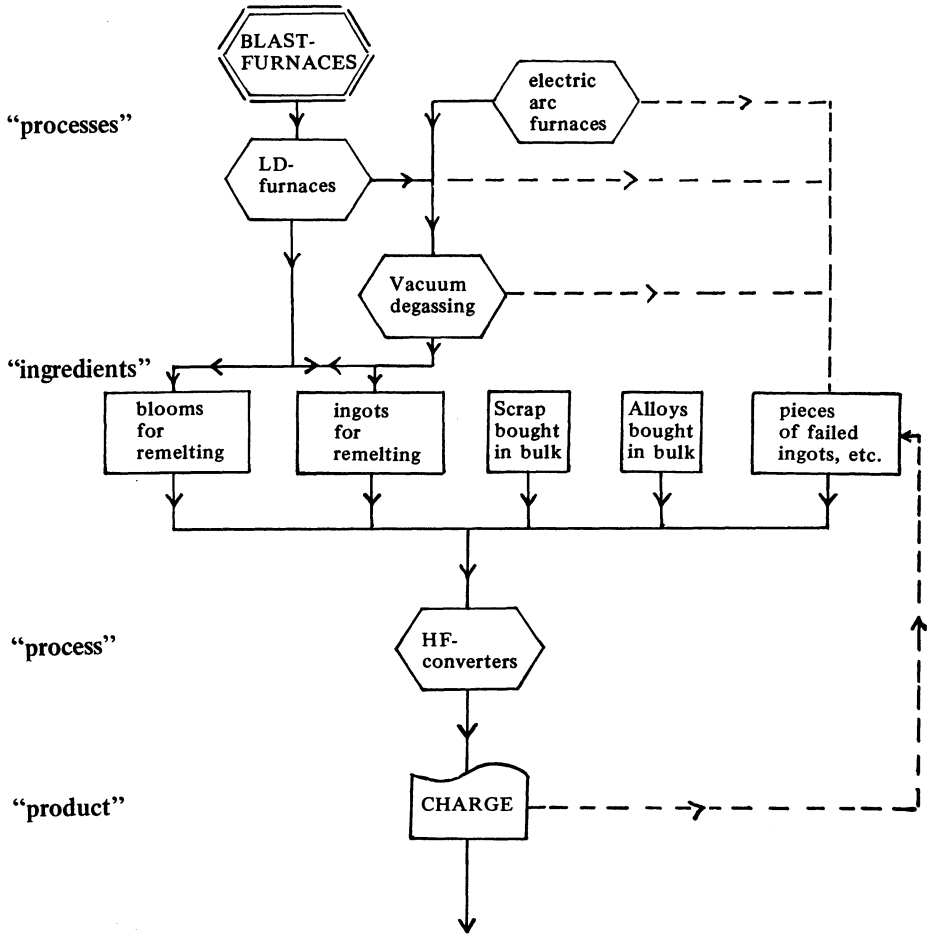
The model is used in the situation when the planner of the HF-converters chooses the appropriate mixture of ingredients that make up a specific requested charge. He knows of the *possible* ingredients (from a file containing weight and

* Contract Research Group for Applied Mathematics, Royal Institute of Technology, Fack, S–100 44 Stockholm 7-, Sweden.

** Fagersta AB, S–77301 Fagersta, Sweden.

analysis of ingredients) and wants to make selections from this file so that the produced charge will have a satisfactory weight and analysis. See Figure I.

FIGURE I. FLOW-CHART OF PRODUCTION IN THE HF-CONVERTERS



In this situation the planner has no possibility to order production of suitable ingredients, but has to select from the available stock. This depends partly on customers' requests for short delivery-times, and partly on the fact that the planner, to a great extent, is permitted to use materials (*pieces* of ingots and slabs) that have been left over in other processes.

As mentioned earlier, it is often a hard combinatorial problem even to achieve a feasible solution. In general, however, there is a large number of such solutions. The goal for the planner is, therefore, to select ingredients so that the ordered charges in the long run are produced as inexpensively as possible.

The model

The model treats charges one at a time. Assuming a realistic cost for each

possible ingredient that reflects its value to the company, the problem is formulated as a mixed integer programming problem. The detailed formulation is found in the Appendix.

To solve the problem, an integer programming code is used which was designed at the Contract Research Group of Applied Mathematics in Stockholm [1]. It is an extension of the 1968 version of Geoffrion's RIP30C [2] to the *mixed* integer case. New features include a primal simplex routine with implicit upper bounds, surrogate constraints, Tomlin penalties and efficient packing of the problem matrix.

A preparatory study indicated that the code could produce solutions within a few percentage points from optimum to realistic test problems with 100-150 binary variables in less than 1 CPU-minute (IBM 360/65).

The final implementation, however, includes an introductory phase where certain ingredients are automatically excluded. In this phase, rules of thumb are used that try to account for the fact that ingredients with least cost may not always be the best choice. Since only one charge is treated at a time, it is of course possible that a very cheap ingredient for the current charge can be even better suited for a later charge. The following are examples of such rules:

> Ingots of high carbon steel are excluded for low carbon charges and vice versa.
> Ingots with a high percentage of molybdenum are excluded for charges with tight upper bounds on this percentage.

In consequence, problem sizes are reduced so that the problems solved with the mixed integer programming algorithm usually have 20-40 binary variables, 5-15 continuous variables and 10-20 constraints (not including simple upper bounds). Solutions within 1% of optimum are usually attained in 20 CPU-seconds (IBM 370/155).

The implemented system with programs and files are described in Figure II. *To initialize the calculations, the steelwork's foreman types the code of the requested charge at the terminal-keyboard.** After that, the generator automatically picks out data from the files, makes the problem-size reduction and generates the input file to the optimizer. Optimization, print-out of solution and updating of the files then take place.

Experiences

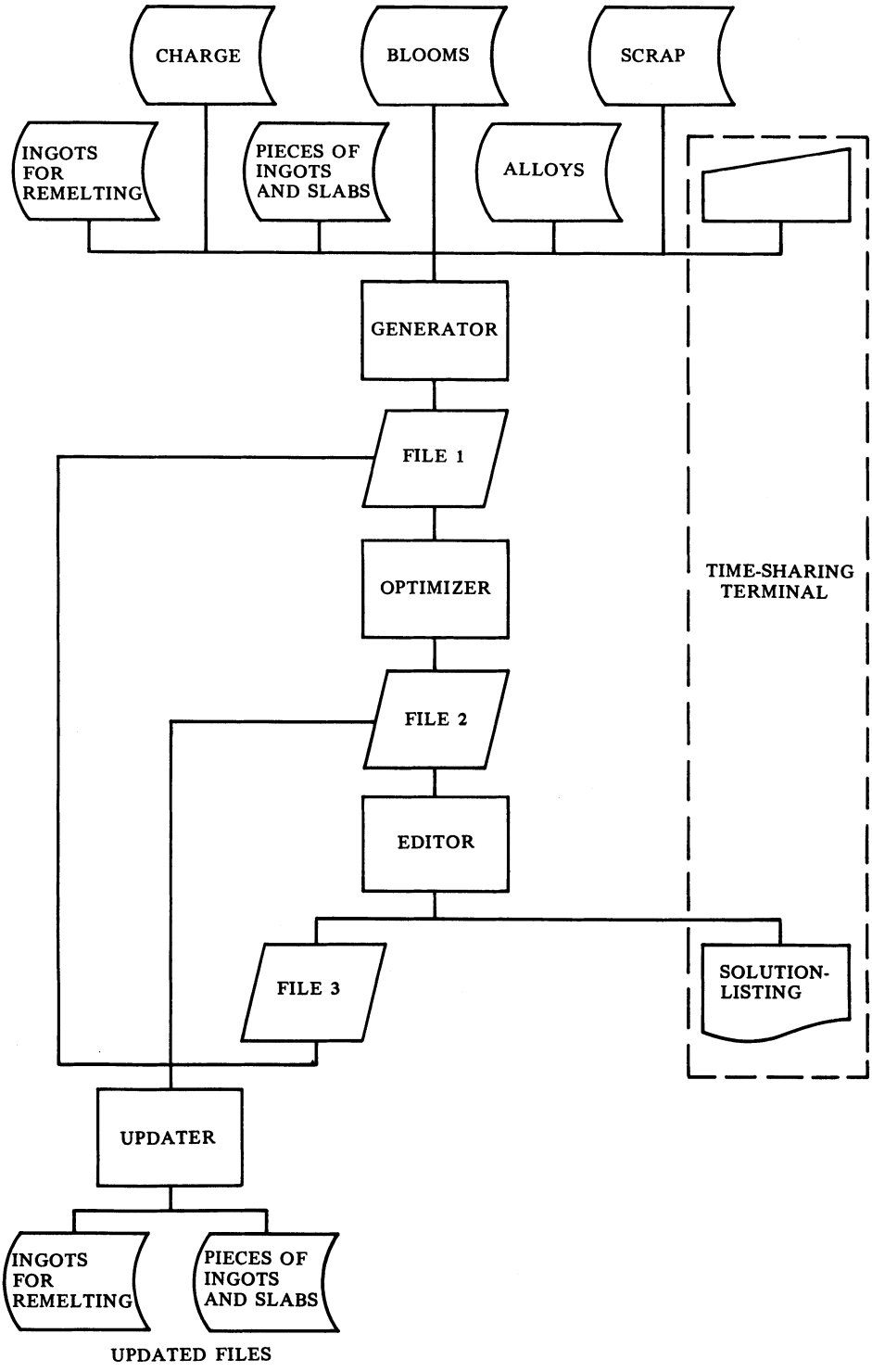
The implementation of this new computerized routine has been highly successful in the company and has completely replaced the former manual calculations. About two runs have been made every day since the end of 1974. A cost accounting made after one year showed that the cost for raw materials had decreased by roughly 6%, which corresponds to a saving of $200,000 a year. This can be compared with the annual cost for data processing (about $10,000) and the costs for developing the routine, which are negligible in this context.

Other advantages have been:

substantial simplification of the planning work;

* Editors Note: It should be emphasized that this system is being operated by *foremen on the shop floor.*

FIGURE II. FLOW-CHART OF THE IMPLEMENTED SYSTEM



elimination of risk for miscalculations, which saves a couple of charges per year from scrapping;

opportunities to put tighter constraints on the analysis of charges without making calculations significantly more difficult.

Appendix

Mathematical formulation

$$(1)\quad \text{Minimize} \sum_j c_j x_j + \sum_k d_k y_k$$

$$(2)\quad \text{s.t.} \sum_j x_j + \sum_k v_k y_k \geqslant \text{weight}^1$$

$$(3)\quad \sum_j x_j + \sum_k v_k y_k \leqslant \text{weight}^u$$

$$(4)\quad \sum_j a_{ij} x_j + \sum_k a_{ik} v_k y_k \geqslant \text{analysis}_i^1 \left(\sum_j x_j + \sum_k v_k y_k \right) \text{ for all } i$$

$$(5)\quad \sum_j a_{ij} x_j + \sum_k a_{ik} v_k y_k \leqslant \text{analysis}_i^u \left(\sum_j x_j + \sum_k v_k y_k \right) \text{ for all } i$$

$$(6)\quad 0 \leqslant x_j \leqslant u_j \text{ for all } j$$

$$(7)\quad y_k = 0 \text{ or } 1 \text{ for all } k.$$

Notations

index i—chemical elements

index j—ingredients available in continuous amounts

index k—"discrete" ingredients

x_j—variable, indicating quantity of ingredient j

y_k—variable, equals 1 if ingredient k shall be in the charge and 0 otherwise

c_j, d_k—cost of corresponding variable

v_k—weight of ingredient k

$\text{weight}^{1,u}$—lower resp. upper bound on charge-weight

a_{ij}, a_{ik}—percentage of chemical element

$\text{analysis}_i^{1,u}$—lower or upper bound resp. on percentage of chemical element i in the charge

u_j—upper bound on weight of ingredient j.

As can be seen from the formulation, variables are of two types. The integer variables correspond to indivisible ingredients. The continuous variables denote entering quantities of ingredients available in bulk.

References

1. Westerberg, C–H., En algorithm for blandad heltalsprogrammering TRITA–MAT–1971–30, Royal Institute of Technology in Stockholm.
2. Geoffrion, A. M., Nelson, A. B., User's instructions for 0–1 integer linear programming code RIP30C. Rand Memorandum RM–5627–PR (1968).