Proceedings of the 1992 Winter Simulation Conference
ed. J. J. Swain, D. Goldsman, R. C. Crain, and J. R. Wilson

OPTIMIZATION OF A CORN-PROCESSING SIMULATION MODEL

David Humphrey
Department of Industrial Engineering
North Carolina State University
Raleigh, North Carolina 27695, U.S.A.

Julius Chu
The Pillsbury Company
Minneapolis, Minnesota 55414, U.S.A.

ABSTRACT

Due to the fact that studies are often performed with the use of computer simulation but little or no effort is made to optimize the results, this work was undertaken in an attempt to find an automated method of optimizing simulation input parameters in order to improve overall performance of the corn-processing facility being studied. The Pillsbury Company operates a corn processing facility. The operation of this facility has recently been studied using computer simulation. Pillsbury requested that the issue of optimizing performance of this facility be investigated. As a result, a flexible polyhedral search algorithm was implemented and integrated into a combined discrete-continuous simulation model of a corn processing facility. The software system is provided with an initial set of input parameters which represent the size of the problem to be evaluated, a starting point for the evaluation, and a relative measure of the precision of the solution. The software system automatically changes values of the input parameters in an effort to optimize the performance of the facility under study.

1 INTRODUCTION

The Pillsbury Company operates a corn processing facility that was the focus of this work. This facility purchases corn from growers in the surrounding region during the corn growing season. Several different varieties of corn are available to be purchased and the corn is used to produced two products of interest at the facility. The facility managers have a great deal of flexibility in deciding the quantities of different varieties of corn to purchase and process. The goal of this work was to find a method of determining the quantities of the available varieties of corn that should be purchased in order to be of greatest financial benefit to the processing facility.

A decision was made that computer simulation would be used to examine this area of interest. A combined discrete-continuous simulation model of the current corn processing operations was developed to analyze the facility performance for a given mix of varieties of corn to be processed. After becoming apparent that simulation of the current corn operations alone would not be sufficient to produce the desired results, an effort was made to integrate the simulation model with a stochastic optimization system. The optimization system needed to be sufficiently robust to work well with functions which were non-differentiable and would likely contain an extreme amount of variability.

2 SETTING

The corn processing facility operates during and at the end of the corn growing season. During this time period, the management of the facility will choose from available fields of mature and nearly mature corn in order to select the different varieties and quantities of corn that the facility will process. Several different varieties of corn are available for consideration. Each variety of corn planted performs differently under varying conditions of temperature, rain fall and sunny days during the growing season (among other factors). Different varieties of corn can have different growth periods needed to reach maturity. Several different varieties of corn, in different quantities, must be planted and grown each season in order to be reasonably sure that an adequate supply of acceptable corn for processing will be available at a reasonable price. It is from these different varieties of corn that the selection is made of the corn to be processed in the facility.

The nature of the corn processing is such that once the processing begins, continuous operation is desired. Every few days the facility will shut down for several hours for extensive cleaning of the equipment.

These shut downs for cleaning are scheduled well in advance. Other than the scheduled shut downs, the processing of the corn is designed to operate on a continuous basis. Any non-scheduled stops in the processing create large inefficiencies in the operation and therefore great effort is taken to avoid such stoppages. The corn processing at this facility is analogous to the processing operations of an oil refinery. Raw material enters the facility and is continually processed in one area and transported to the next until finally emerging from the entire operation as finished product. A flow of material from one area of processing to the next is always desired. Any time the flow of material from one area to another is completely stopped, a danger of expensive (in terms of time and money) start up procedures exists. This continuous nature of the operation helped to convince us that the continuous portion of the combined discrete-continuous simulation model was critical to understanding the problem.

Sufficient buffers have been placed between different operations so that individual pieces of equipment may occasionally stop working for short periods of time without causing the stoppages described above. These occasional stops may be caused by breakdowns in the equipment or by the buffer at the next operation being over capacity. Such events were modeled in the discrete event portion of the model. Additionally, the workers in the facility can adjust the rates at which individual pieces of equipment operate as dictated by the level of product in the buffers. These adjustments (which are critical to balancing the flow of material among operations) were incorporated into the model as well.

This facility operates two product lines that were of interest to our study. The products produced on these lines will be referred to as corn product one (CP1) and corn product two (CP2). The production of these two products actually takes place on only one production line for part of the operation. The raw material corn is delivered to the facility and is stored temporarily on a large concrete slab. The corn is then fed into the single production line where several operations take place (e.g., removing the husk from the corn). The flow from this one line is then split and corn for CP1 proceeds through one set of operations while corn for CP2 proceeds through a different set of operations. Splitting the flow of the corn is accomplished by making a decision as to whether each ear of corn is better suited for the CP1 line or the CP2 line. Figure 1 represents the flow of the corn through the facility. The different varieties of corn that have been picked and delivered to the facility will affect the proportion of corn being sent to either of the production lines. Part of the ability to produce enough of CP1 and CP2 depends upon having acceptable quantities of different varieties of corn planted, grown, picked and delivered to the plant. If these quantities are not acceptable, the product flow in the facility can not be adequately split and one of the two production lines will be starved (and the other will lack capacity to process all it is being sent).

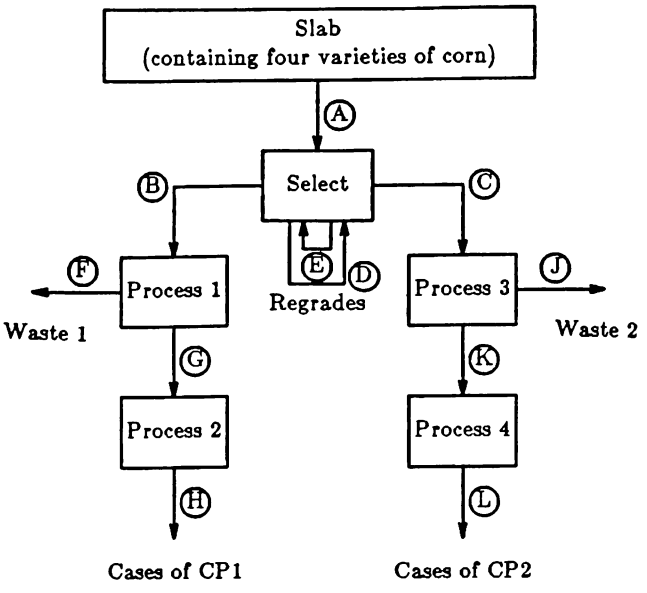

Figure 1: Product Flow through Facility

3 METHODOLOGY

Before beginning work on the simulation model, consideration was given to the choice of language which should be used. Our understanding of the problem indicated that a simulation language supporting continuous variables was required. However, a language with continuous variables but no discrete event logic wound not be acceptable due to the variety of discrete events which needed to be modeled as well. We were working primarily in a personal computer environment and the simulation language needed to be supported at this level. The Pillsbury Company also had a strong desire for animations to accompany the simulation modeling. With this information taken into account, a decision was made to use SLAM II (Pritsker 1986) within the SLAMSYSTEM software package (Pritsker Corporation 1989).

The goal of this work was to develop a software system that would address the issue of which varieties of corn should be purchased and the quantities

in which to purchase those varieties. This software system was developed in two parts. The first part of this system is a simulation model which uses specified quantities of the different varieties of available corn (an experimental design point) to estimate the processing facility performance for that design point. The second part of the system is a flexible polyhedral search algorithm which uses the output from the simulation model to determine a new design point (i.e., a new combination of quantities of the varieties of corn to be processed).

3.1 Simulation Model

The elements of the simulation model are of two primary types. There is a continuous portion of the model which represents the continuous flows of material from one area to another. The rates of flow from one area to another and the level of product accumulated at any given area may change continually over time and are modeled accordingly. Each operation in the process is in effect simulated at a micro level insuring that everything that flows into an area also flows out of the area. The rates of product flow are typically updated every five simulated minutes through calls to the SLAM `subroutine state`. The rates of flow are modeled as being constant between these five minute updates. The rates of product accumulation or depletion at an operation are also modeled as being constant between the updates occurring every five simulated minutes.

The continuous portion of the simulation model also monitors the process for a variety of threshold crossings. A threshold may be specified as the maximum or minimum quantity of product allowed to be in a buffer at a given operation at any given point in time. Once the quantity in the buffer goes above the specified maximum or below the specified minimum, the threshold has been crossed and some type of state event occurs (in these two cases, perhaps the increase or decrease, respectively, in the rate at which product is processed at the operation corresponding to the buffer in question). In other words, if too much product accumulates in a queue, a threshold is crossed and changes in some of the rates of the operation must be made.

The discrete event portion of the simulation is used to model events taking place at random times during the operation of the facility. These discrete events are occurrences of phenomena which directly impact the flow of product through the processing operations of the facility but can not be adequately modeled in a strictly continuous fashion. Such occurrences may be machine breakdowns, machine repair completions or the delivery of more raw material corn now available for processing. These randomly occurring events cause instantaneous changes in the status of the facility and must be modeled as discrete events so their impact on the flow of product (the continuous portion of the model) may be accounted for accurately.

3.2 Search Algorithm

The algorithm used to determine the best combination of quantities of the varieties of corn to be processed is a flexible polyhedral search algorithm put forth by Nelder and Mead (1965). The algorithm is a simplex method for minimizing a function which is dependent on the value of the function at various vertices of a simplex. The method, when trying to minimize over n variables, starts with $n+1$ vertices (a general simplex in n-dimensional space) and continues replacing the vertex with the highest objective function value with a new point. The method continues to adjust to the local landscape until reasonably close to the minimum (as specified by certain initial criteria). The method was shown to perform well with a low number of variables (up to n=10).

For this work, the algorithm is initially provided the combination of quantities of the varieties of corn processed (a design point) and the value of an objective function based upon the simulated performance of the facility at the given design point. Additional design points are specified by the algorithm so that a simplex is formed and the simulation is run (and the objective function evaluated) for each of these design points. The algorithm then removes the worst point from the simplex (based upon objective function value), replaces the removed point with a new design point and causes the facility performance to be simulated (and the objective function to be evaluated) for this new design point. This procedure of removing the worst design point from implex and replacing it with a new design point continues until an acceptable level of precision (as indicated by the user) is reached.

This algorithm is attractive and was selected for this work because of its ability to quickly move through the topology of the problem. In determining new design points for the simplex, the algorithm repeatedly determines the centroid of the current simplex and steps through a series of five operations (referred to as reflection, expansion, contraction, reduction and replacement) which effectively scale the next simplex to the local topology. The scaling of each successive simplex allow the algorithm to make faster progress in moving through steep regions of the n-dimensional polytope. At the same time, the scal-

ing also allows the algorithm to contract and terminate at an (hopefully global) optimum without many unnecessary iterations. In short, this algorithm has been found to be very quick and relatively robust.

3.3 Integration of Simulation Model and Search Algorithm

For the software system to accomplish the established goal, the simulation model and the search algorithm need to be integrated. Figure 2 depicts the relationship between the Nelder-Mead search algorithm (NELMIN) and the simulation model. The

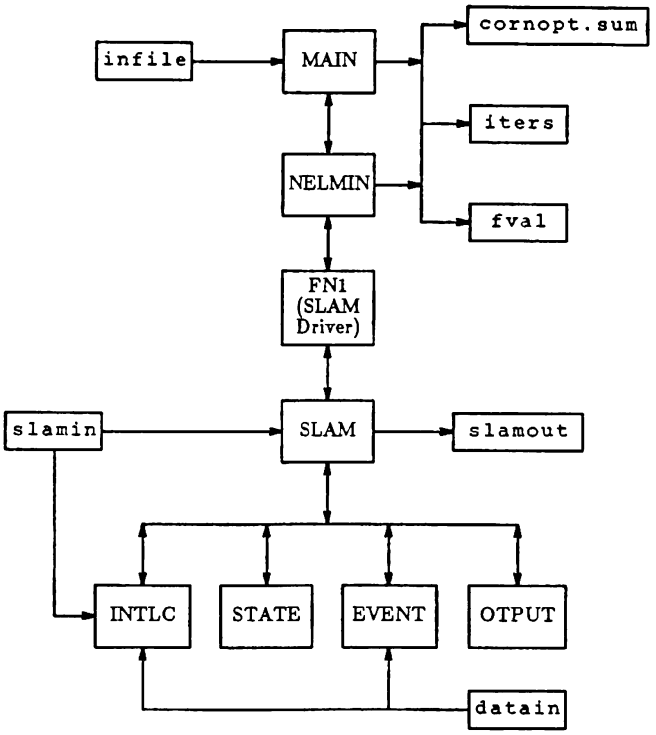

Figure 2: Software System Control

MAIN program reads from input file `infile`the number of parameters over which to optimize, the initial values of the n parameters, the maximum value allowed for each parameter, a tolerance value for stopping the NELMIN procedure (discussed below) and the maximum number of iterations the NELMIN procedure will be allowed to perform. This information is also written to the output summary report, `cornopt.sum`. The MAIN program then calls subroutine `NELMIN` and passes to NELMIN the initial parameter values (i.e., the quantities of the different varieties of corn to be processed) and other values needed to start the search algorithm. NELMIN then evaluates the objective function at the (initial) design point by calling SLAM and performing a simulation experiment at the (initial) design point. The simulation of the design point continues until the number of replications performed satisfies the relative precision stopping rule built into the simulation model (discussed below). At this point the average objective function value computed by the simulation experiment is passed back to `function FN1` and the information is written to the summary report `cornopt.sum`. Additionally, the average objective function value and the NELMIN iteration number are written to files `fval` and `iters`, respectively. (These two files are used to create plots of the system performance.) The function FN1 then passes the average function value back to NELMIN as the value of the function to be optimized at the current (initial) vertex. This then completes one iteration of the overall optimization procedure.

NELMIN then completes the initial simplex by calling `function FN1` to evaluate the objective function at n additional vertices. The coordinates of each of the additional n vertices are determined by increasing each of the individual initial values by their initial step size while holding the other n-1 parameters at their initial values. For each of these n vertices, the function FN1 call SLAM and performs the needed simulation runs until an objective function value can be passed back to NELMIN.

Once NELMIN has the first n+1 vertices and their corresponding function values (as determined by the simulation runs), the procedure then goes about performing its five operations (reflection, expansion, contraction, reduction and replacement) in order to add vertices to the current simplex, delete vertices from the current simplex, and evaluate the new vertices by performing simulation runs. The procedure continues to add new points to the simplex in hope of improving on the best objective function value until no significant progress is being made or the maximum number of iterations hs been reached. Once this occurs, the best function value (and the corresponding coordinates of the vertex) is returned to the MAIN program, that information is written to the summary report, and the optimization run is complete.

In order to determine when no significant progress is being made by the search algorithm, a stopping rule is applied after each new design point has been simulated to find an objective function value. The stopping rule consists of determining the square root of the sum of the squared differences between the objective function value at each point in the simplex and the objective function value at the centroid

of the simplex divided by n. This quantity is then compared to epsilon, a user specified arbitrarily small positive constant. If the calculated quantity is less than epsilon, then the search algorithm terminates having sufficiently located the (hopefully global) optimum.

The simulation portion of the software system was also automated to determine how long it should run. A relative precision stopping rule was placed in the SLAM subroutine OTPUT. This relative precision stopping rule was evaluated after each simulation replication (or run) was completed (after having completed a required minimum number of replications). This stopping rule requires that the standard error of the mean objective function value should not exceed a prespecified percentage of the mean objective function value observed at the current design point. If this condition is not met, an additional simulation replication at this point is performed and the stopping rule is evaluated once again. When the stopping condition is met, the average objective function value is returned to NELMIN as the estimate of the expected simulation response at this point of the simplex.

3.4 Data Collection

Our data collection effort focused in two primary areas: data related to the corn being processed by the facility and data related to the equipment doing the processing. Several measures of the quality of corn to be processed were obtained from historical data, from personnel who inspect the corn in the field, from personnel responsible for purchasing the corn and from personnel working directly in the processing operations. These quality measures include the following and are estimated for each load of raw material corn arriving at the facility:

- percentage of corn which may ultimately be used for CP1
- percentage of corn which may ultimately be used for CP2
- percentage of damaged corn which must be removed from the processing operations
- efficiencies at which certain pieces of equipment may process the corn (these values will change with generally larger or smaller ears of corn)
- percentage of corn (by weight) which will be lost during production of CP1
- percentage of corn (by weight) which will be lost during production of CP2

We also obtained information on machine capacities, frequency of breakdowns, duration of repair times and other relevant information relating to the equipment in the facility. This information was generally obtained from historical data and from production and maintenance personnel working at the facility.

3.5 The Experimental Frame

The production from an entire growing season was modeled with each replication of the simulation. With each replication the simulation starts "empty and idle" as does the processing facility at the beginning of each growing season. Each replication ends after the required production demand has been met (very similar to the facility as well). Replications of a design point are performed (as discussed earlier) until sufficient "noise" has been removed from the estimate of the objective function value.

The performance measures of interest from the simulation experiment include the following:

- quantity of CP1 produced
- quantity of CP2 produced
- total time facility operated
- cost to purchase the corn for processing

With knowledge of the hourly costs associated with operation the facility and the revenues generated from each unit of CP1 and CP2 produced, the performance measures above allow us to estimate the expected profitability of the facility for a given design point. This is done by way of the objective function which is evaluated after each design point is simulated.

3.6 Verification and Validation

The verification of the model and software code took place in several steps. First of all, care was taken to build the model and code in several steps and perform debugging along the way. Print statements and write statements were temporarily placed in the code to insure that values passed between subroutines were those which were intended to be passed. After it was determined that the simulation model and the rest of the code were interacting correctly, several short runs were performed to insure that reasonable results were being found; this was done by comparing results against past performance of the system.

The validation of the simulation model actually took place over several months. The simulation

fval=617546	xfin(1)= 1.6
iters=44	xfin(2)=240.3
av runs=3.9	xfin(3)=272.6
	xfin(4)=210.7

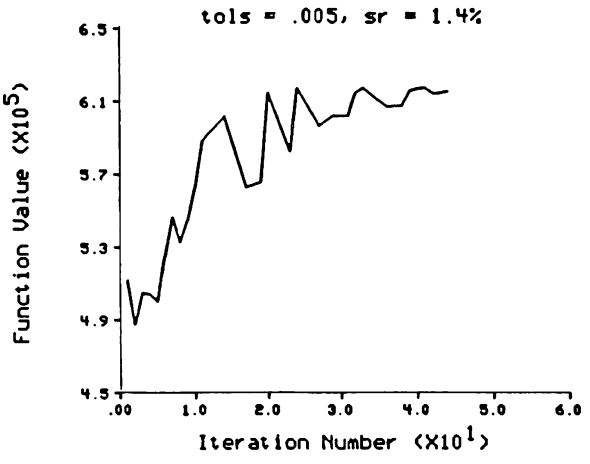

Figure 3: System Improvement

fval=626035	xfin(1)= 6.4
iters=138	xfin(2)=248.1
av runs=3.7	xfin(3)=278.6
	xfin(4)=179.1

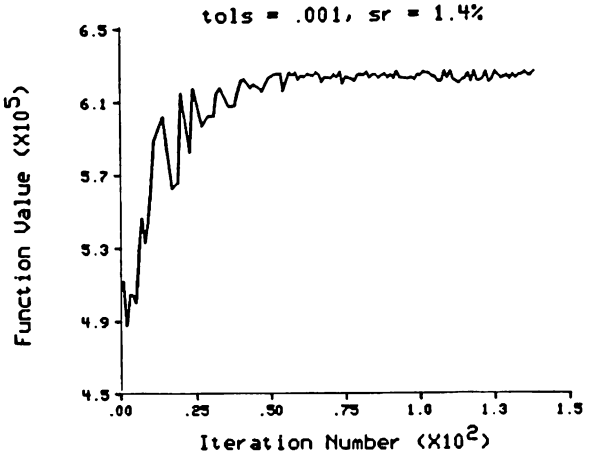

Figure 4: System Improvement

model was developed with a great deal of input from many representatives of Pillsbury. Assumptions were discussed with personnel in knowledgeable positions. Available data were studied in detail and additional data were gathered as needed.

4 RESULTS

The experiments involved examining a situation where four different varieties of corn are to be used for processing in the facility during the season. The quantities of corn used for starting values for the simulation runs were all set initially to 200 units. That is, the four decision variables were initially equal to 200. The relative precision stopping rule used to determine when an adequate number of simulation replications had been completed was set at a value of 1.4%. The tolerance (used by the Nelder-Mead procedure to determine if sufficient progress was being made) was set at values of .005 and .001. In both cases the objective function value started near 510,000 and was around 620,000 upon completion of the run. The final objective function value (fval), number of iterations of the Nelder-Mead procedure (iters) and the average number of simulation replications per design point (av run) are shown below. The final value of each of the decision variables (which started at a value of 200) is also shown (the array xfin). A graphical representation of the objective value improvement as the Nelder-Mead iterations increased is also shown below.

Both tolerance levels seem to provide the same early improvement in the objective function value although the tighter tolerance level provides a marginally better final result at the cost of more iterations through the Nelder-Mead procedure.

The next experiments examined a two phase approach. The first phase consisted of starting at the same initial design point and using "loose" tolerance levels to quickly find a good starting point for the second phase. In the second phase, this good starting point was used with tighter tolerances to allow the system to reach an even better objective function value. This two phase approach appeared to work well by allowing the first phase to move from a relatively poor starting point to a relatively good point and then allowing the second phase to do some "fine tuning" in the local landscape around the optimal solution.

The final experiments to be discussed deal with some anomalous behavior experienced with the Nelder-Mead procedure. When the relative precision stopping rule (used to determine when sufficient statistical noise from simulation replications had been

eliminated) was set much below 1.4% (say, to 1.3%), the number of required simulation replications at a given design point increased dramatically. There seemed to exist some type of threshold between relative precision levels of 1.4% and 1.3% for this particular model. Once this threshold was crossed, 20 to 25 simulation replications were often required at each vertex rather than the three or four replications which were previously satisfactory. Additionally, and more importantly, the entire software system appeared to get hung-up or trapped in the local landscape; and only minimal progress was made in improving the average objective function value.

5 CONCLUSIONS

A study was performed of the operations of a corn processing facility owned by the Pillsbury Company. The problem of improving the mix of varieties of corn to be processed at the facility was examined by way of a combined discrete-continuous simulation model of the corn processing facility integrated with a flexible polyhedral search algorithm. The developed software system starts with an initial design point to simulate (some reasonable starting values) and automatically determines new design points to simulate in an effort to optimize the performance of the facility. The new design points are determined by the Nelder-Mead procedure based upon the simulated performance of the facility at other design points. The system performs well in a two phase approach wherein the first phase is used to move from a relatively poor initial design point to an improved starting point and the second phase uses the results from the first phase to move much closer to the optimal solution. The second phase is performed with the stopping rules at much more restrictive levels than the first phase. Results were presented showing improved facility performance in terms of objective function values. Anomalous behavior experienced with excessively tight controls was also discussed.

ACKNOWLEDGMENTS

The authors would like to thank the Pillsbury Company for support of this work.

REFERENCES

Nelder, J. A., and R. Mead. 1965. A simplex method for function minimization. *The Computer Journal* 7:308-313.

Pritsker, A. A. B. 1986. *Introduction to simulation and SLAM II.* 3d ed. New York: Halsted Press.

Pritsker Corporation 1989. SLAMSYSTEM: Total Simulation Project Support, Pritsker Corporation, Indianapolis, Indiana.

AUTHOR BIOGRAPHIES

DAVID HUMPHREY is a Ph.D. student in the Department of Industrial Engineering at North Carolina State University. He received B.S. and M.S. degrees in industrial engineering from Purdue University in 1989 and 1991 respectively.

JULIUS CHU is a Senior Process Engineer with the Pillsbury Company in Minneapolis, Minnesota. He received a B.S. degree in mechanical engineering from National Taiwan University in 1977, a M.S. degree in industrial engineering from the University of Rhode Island in 1981 and a Ph.D. degree in industrial engineering from Purdue University in 1984.