A Genetic Algorithms Programming Application in Natural Cheese Products

ANANT OONSIVILAI[1] , RATCHADAPORN OONSIVILAI[2],
[1] Alternative and Sustainable Energy Research Unit, School of Electrical Engineering,
Institute of Engineering
[2]School of Food Technology,
Institute of Agricultural Technology
Suranaree University of Technology
111 University Street, Muang District, Nakhon Ratchasima, 30000
THAILAND
roonsivi@sut.ac.th, anant@sut.ac.th http://www.sut.ac.th

Abstract: - Application of genetic algorithms to attain the parameters of the processed cheese products is used in this paper. As in the recent years, model integrates cheese manufacturing has had considerable appeal. The economical evaluation of standaized milk for cheese making is established, thus an objective is given; to make the profit of the natural chesse. Genetic algorithms are highly eligible for investigation of multimodal space of discreteness, noisiness and complexity. Majority of Genetic Algorithms (GA) modifications to solve problematic optimization issues, find some mathematical support to the GA. Explotation of Genetic Algorithms modified optimization give advanced results as recognized.

Key-Words: Genetic Algorithms, Natural Cheese, Process, Optimization, Economic, Profit

1 Introduction

Cheese is one of the main products from milk in the dairy industry.Cheese manufacturing is essentially a Process of concentrating the protein(casein) and fats contained in milk by complicated microbial and biochemical reactions (Smith, 1996).The main steps in cheese manufacturing are: (1)the acidication of milk by the microbial conversion of lactose to lactic acid by lactic acid bacteria,(2) the coagulation of casein and fats by the combination of proteolysis using chymosin (rennet) and the acidication, (3) the dehydration and shaping of the coagulated casein and fats(curd),and additionnally(4) maturing of the dehydrated curd as required as shown in Figure 1 [1].

Speficialization in dairy product manufacture has produced more easily managed production system, but in the case of processed cheese production the results is not necessarily the most efficient use of milk resources. An intregated approached to managing cheese making resources allows the processor to get the most from incoming materials. In this case "integrated" means having some control of the manufacture or purchase of intermediate products used in processed cheese manufacture, including especially the incoming natural cheeses that may be used in this processing. Controlling the manufacture of these products for overall organization objective can lead to manufacturing efficiencies and to increased profitability for the organization [2]

The emphasis of the published literature on process cheese typically has focused on the manufacturing process. The study of the contribution of the ingredients has generally been limited to their principle effects on melting properties, textural and flavor contributions, microorganism contamination, and browning problems. There are several good reviews of the industry and of the manufacturing process. The economies involved in the manufacture of natural cheese and the manufacure of process cheese have traditionally been analyzed independently. A linear programming model is described herein that integrates these two production systems for the purpose of maximizing net returns from natural cheese and processed cheese products. Although the usual restrictions of a linear programming model apply, the advantage of developing such a model is that machine optimal solutions provide a place to begin the analysis and adjustment of production plans and methods. Further, the model provides tools that make the economic and resource

use consequences of adjusting production plans available to the modeler [1, 9-11, 13-14, 16].

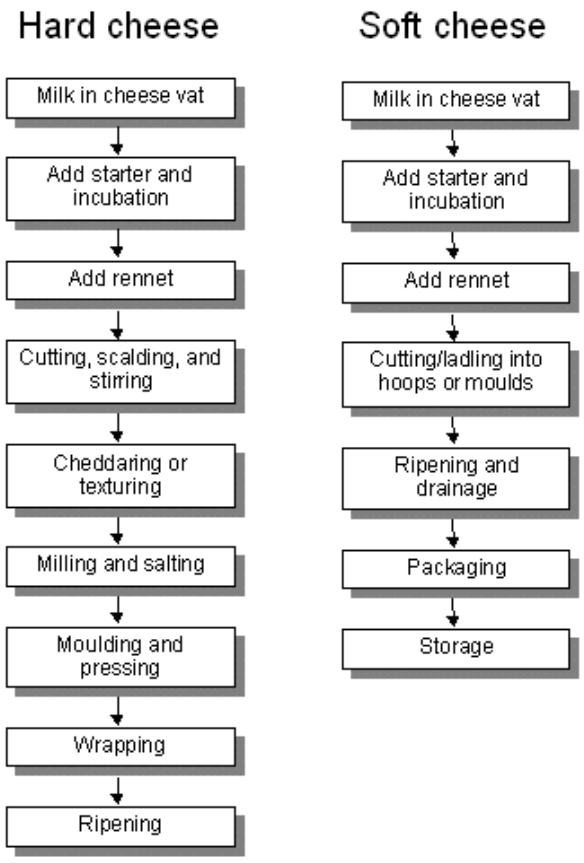

Figure 1 Flow diagram of cheese production [1].

3. Genetic Algorithms

A genetic algorithm (GA) is a search technique used in computing to find exact or approximate solutions to optimization and search problems. Genetic algorithms are categorized as global search heuristics. Genetic algorithms are a particular class of evolutionary algorithms (also known as evolutionary computation) that use techniques inspired by evolutionary biology such as inheritance, mutation, selection, and crossover (also called recombination).

Genetic Programming
In artificial intelligence, genetic programming (GP) is an evolutionary algorithm-based methodology inspired by biological evolution to find computer programs that perform a user-defined task. It is a specialization of genetic algorithms where each individual is a computer program. Therefore it is a machine learning technique used to optimize a population of computer programs according to a fitness landscape determined by a program's ability to perform a given computational task.

Genetic operators
The main operators used in evolutionary algorithms such as GP are crossover and mutation.

Crossover
Crossover is applied on an individual by simply switching one of its nodes with another node from another individual in the population. With a tree-based representation, replacing a node means replacing the whole branch. This adds greater effectiveness to the crossover operator. The expressions resulting from crossover are very much different from their initial parents.
The following code suggests a simple implementation of individual deformation using crossover:

```
individual.Children[randomChildIndex]=
otherIndividual.Children[randomChildIndex];
```

Mutation
Mutation affects an individual in the population. It can replace a whole node in the selected individual, or it can replace just the node's information. To maintain integrity, operations must be fail-safe or the type of information the node holds must be taken into account. For example, mutation must be aware of binary operation nodes, or the operator must be able to handle missing values.

A simple piece of code:

```
individual. Information = randomInformation;
```

or

```
individual = generateNewIndividual;
```

Meta-Genetic Programming
Meta-Genetic Programming is the proposed meta learning technique of evolving a genetic programming system using genetic programming itself. It suggests that chromosomes, crossover, and mutation were themselves evolved, therefore like their real life counterparts should be allowed to change on their own rather than being determined by a human programmer. It is a recursive but terminating algorithm, allowing it to avoid infinite recursion.

Critics of this idea often say this approach is overly broad in scope. However, it might be possible to constrain the fitness criterion onto a general class of results, and so obtain an evolved GP that would more efficiently produce results for sub-classes. This might take the form of a Meta evolved GP for producing human walking algorithms which is then used to evolve human running, jumping, etc. The fitness criterion applied to the Meta GP would simply be one of efficiency.
For general problem classes there may be no way to show that Meta GP will reliably produce results more efficiently than a created algorithm other than exhaustion. The same holds for standard GP and other search algorithms.

Methodology
Genetic algorithms are implemented as a computer simulation in which a population of abstract representations (called chromosomes or the genotype of the genome) of candidate solutions (called individuals, creatures, or phenotypes) to an optimization problem evolves toward better solutions. Traditionally, solutions are represented in binary as strings of 0s and 1s, but other encodings are also possible. The evolution usually starts from a population of randomly generated individuals and happens in generations. In each generation, the fitness of every individual in the population is evaluated, multiple individuals are stochastically selected from the current population (based on their fitness), and modified (recombined and possibly randomly mutated) to form a new population. The new population is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population. If the algorithm has terminated due to a maximum number of generations, a satisfactory solution may or may not have been reached.

Genetic algorithms find application in phylogenetics bioinformatics, computational science, engineering, economics, chemistry, manufacturing, mathematics, physics and other fields.

A typical genetic algorithm requires two things to be defined:

1. a genetic representation of the solution domain,
2. a fitness function to evaluate the solution domain.

A standard representation of the solution is as an array of bits. Arrays of other types and structures can be used in essentially the same way. The main property that makes these genetic representations convenient is that their parts are easily aligned due to their fixed size, that facilitates simple crossover operation. Variable length representations may also be used, but crossover implementation is more complex in this case. Tree-like representations are explored in Genetic programming and graph-form representations are explored in Evolutionary programming.

The fitness function is defined over the genetic representation and measures the *quality* of the represented solution. The fitness function is always problem dependent. For instance, in the knapsack problem we want to maximize the total value of objects that we can put in a knapsack of some fixed capacity. A representation of a solution might be an array of bits, where each bit represents a different object, and the value of the bit (0 or 1) represents whether or not the object is in the knapsack. Not every such representation is valid, as the size of objects may exceed the capacity of the knapsack. The *fitness* of the solution is the sum of values of all objects in the knapsack if the representation is valid, or 0 otherwise. In some problems, it is hard or even impossible to define the fitness expression; in these cases, interactive genetic algorithms are used.
When have the genetic representation and the fitness function defined, GA proceeds to initialize a population of solutions randomly, then improve it through repetitive application of mutation, crossover, inversion and selection operators.

Initialization
Initially many individual solutions are randomly generated to form an initial population. The population size depends on the nature of the problem, but typically contains several hundreds or thousands of possible solutions. Traditionally, the population is generated randomly, covering the entire range of possible solutions (the *search space*). Occasionally, the solutions may be "seeded" in areas where optimal solutions are likely to be found.

Selection
During each successive generation, a proportion of the existing population is selected to breed a new generation. Individual solutions are selected through a *fitness-based* process, where fitter solutions (as measured by a fitness function) are typically more likely to be selected. Certain selection methods rate the fitness of each solution and preferentially select

the best solutions. Other methods rate only a random sample of the population, as this process may be very time-consuming.

Most functions are stochastic and designed so that a small proportion of less fit solutions are selected. This helps keep the diversity of the population large, preventing premature convergence on poor solutions. Popular and well-studied selection methods include roulette wheel selection and tournament selection.

Reproduction

The next step is to generate a second generation population of solutions from those selected through genetic operators: crossover (also called recombination), and/or mutation.

For each new solution to be produced, a pair of "parent" solutions is selected for breeding from the pool selected previously. By producing a "child" solution using the above methods of crossover and mutation, a new solution is created which typically shares many of the characteristics of its "parents". New parents are selected for each child, and the process continues until a new population of solutions of appropriate size is generated.

These processes ultimately result in the next generation population of chromosomes that is different from the initial generation. Generally the average fitness will have increased by this procedure for the population, since only the best organisms from the first generation are selected for breeding, along with a small proportion of less fit solutions, for reasons already mentioned above.

Termination

This generational process is repeated until a termination condition has been reached. Common terminating conditions are:

- A solution is found that satisfies minimum criteria Fixed number of generations reached
- Allocated budget (computation time/money) reached
- The highest ranking solution's fitness is reaching or has reached a plateau such that successive iterations no longer produce better results
- Manual inspection
- Combinations of the above

Simple Generational Genetic Algorithm Pseudocode

1. Choose initial population
2. Evaluate the fitness of each individual in the population
3. Repeat until termination: (time limit or sufficient fitness achieved)
    1.Select best-ranking individuals to reproduce
    2.Breed new generation through crossover and/or mutation (genetic operations) and give birth to offspring
    3.Evaluate the individual fitnesses of the offspring
    4. Replace worst ranked part of population with offspring

Observations

There are several general observations about the generation of solutions via a genetic algorithm:

In many problems, GAs may have a tendency to converge towards local optima or even arbitrary points rather than the global optimum of the problem. This means that it does not "know how" to sacrifice short-term fitness to gain longer-term fitness. The likelihood of this occurring depends on the shape of the fitness landscape: certain problems may provide an easy ascent towards a global optimum, others may make it easier for the function to find the local optima. This problem may be alleviated by using a different fitness function, increasing the rate of mutation, or by using selection techniques that maintain a diverse population of solutions, although the No Free Lunch theorem proves that there is no general solution to this problem. A common technique to maintain diversity is to impose a "niche penalty", wherein, any group of individuals of sufficient similarity (niche radius) have a penalty added, which will reduce the representation of that group in subsequent generations, permitting other (less similar) individuals to be maintained in the population. This trick, however, may not be effective, depending on the landscape of the problem. Diversity is important in genetic algorithms (and genetic programming) because crossing over a homogeneous population does not yield new solutions. In evolution strategies and evolutionary programming, diversity is not essential because of a greater reliance on mutation.

Operating on dynamic data sets is difficult, as genomes begin to converge early on towards solutions which may no longer be valid for later data. Several methods have been proposed to remedy this by increasing genetic diversity

somehow and preventing early convergence, either by increasing the probability of mutation when the solution quality drops (called *triggered hypermutation*), or by occasionally introducing entirely new, randomly generated elements into the gene pool (called *random immigrants*). Again, evolution strategies and evolutionary programming can be implemented with a so-called "comma strategy" in which parents are not maintained and new parents are selected only from offspring. This can be more effective on dynamic problems.

GAs cannot effectively solve problems in which the only fitness measure is a single right/wrong measure, as there is no way to converge on the solution (no hill to climb). In these cases, a random search may find a solution as quickly as a GA. *However*, if the situation allows the success/failure trial to be repeated giving (possibly) different results, then the ratio of successes to failures provides a suitable fitness measure.

Selection is clearly an important genetic operator, but opinion is divided over the importance of crossover versus mutation. Some argue that crossover is the most important, while mutation is only necessary to ensure that potential solutions are not lost. Others argue that crossover in a largely uniform population only serves to propagate innovations originally found by mutation, and in a non-uniform population crossover is nearly always equivalent to a very large mutation (which is likely to be catastrophic). There are many references that support the importance of mutation-based search, but across all problems the No Free Lunch theorem holds, so these opinions are without merit unless the discussion is restricted to a particular problem.

Often, GAs can rapidly locate *good* solutions, even for difficult search spaces. The same is of course also true for evolution strategies and evolutionary programming.

For specific optimization problems and problem instances, other optimization algorithms may find better solutions than genetic algorithms (given the same amount of computation time). Alternative and complementary algorithms include evolution strategies, evolutionary programming, simulated annealing, Gaussian adaptation, hill climbing, and swarm intelligence (e.g.: ant colony optimization, particle swarm optimization) and methods based on integer linear programming. The question of which, if any, problems are suited to genetic algorithms (in the sense that such algorithms are better than others) is open and controversial.

As with all current machine learning problems it is worth tuning the parameters such as mutation probability, recombination probability and population size to find reasonable settings for the problem class being worked on. A very small mutation rate may lead to genetic drift (which is non-ergodic in nature). A recombination rate that is too high may lead to premature convergence of the genetic algorithm. A mutation rate that is too high may lead to loss of good solutions unless there is elitist selection. There are theoretical but not yet practical upper and lower bounds for these parameters that can help guide selection.

The implementation and evaluation of the fitness function is an important factor in the speed and efficiency of the algorithm [15].

to achieve the pasteurized process cheese food product. In the example developed here, Table 1 lists some of the input resources that could be used in the manufacture of natural cheese. We will assume that Cheddar cheese is to be manufactured and some or all of it will be processed.

The Cheddar cheese is to be made with a fat on a dry basis (FDB) of 53.5%. This is accomplished by regulating the casein to fat ratio to be .6925. Moisture of the cheese is assumed to remain at 37%. The values used in the cheese yield formula are 1.09 salt solids retention factor, 93% fat retention, and 96% casein retention. It is assumed that the fat content of whey cream removed is 45%.

Table 1. Several input resources available for use in the manufacture of natural cheddar cheese.

Resource	Amount available	Fat %	Casein %	Cost/ pound
Silo 1	300,000	3.55	2.44	0.1178
Silo 2	300,000	3.57	2.49	0.1181
NDM	As needed	1.00	28.00	0.79
Cream	As needed	45.0	1.39	0.80
Condense Skim milk	As needed	0.37	9.20	0.228

Whey fat is recovered at a rate of 100%. All of the raw milk is used for cheese production. The purchasable quantities of nonmilk resources is assumed to be immeasurable. Potential inputs for process cheese manufacture are listed in Table 2.

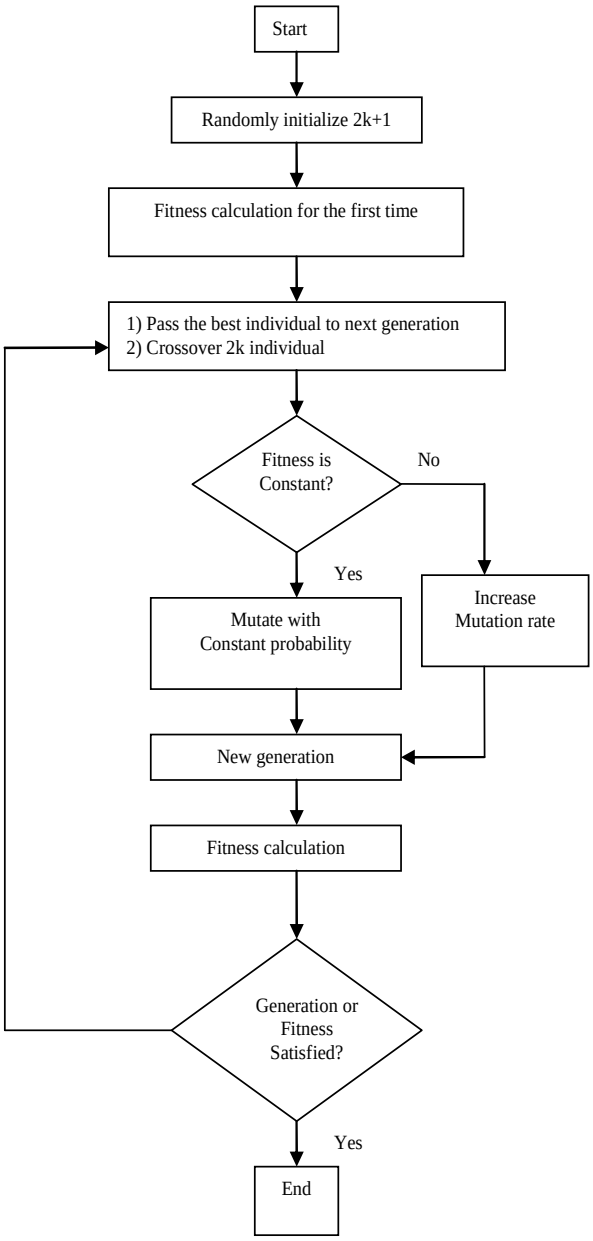

Fig. 4. Flowchart of Genetic Algorithms [10].

All of the manufactured cheese is either sold or processed. It is possible to divide the manufactured cheese, i.e., one-third can be sold and two-thirds processed or half sold and half processed. The value of the output products (cheese, whey cream, and separated whey) and the resources used for process cheese manufacture are related to the value

Table 2. Several input resources available for use in the manufacture of process cheese.

Resource	Fat %	Moisture %	Cost/ pound
Cheese	Determined by LP.	37.0	Determined by LP
Whey protein concentrate	3.5	1.5	0.4825
Emulsifiers	0	0	0.535
Fat	83.0	17.0	1.53
Whey cream	45.0	55.0	0.78
Water	0	100.0	0.01

of the final process cheese product. The end product is pasteurized process block cheese valued at a block wholesale price of $1.50/lb.

3.1 Decision Variables

The decision variables for the natural cheese manufacture are identified with the resources that may be used and the amount of cream that can be removed from a milk resource during a standardization [1]. Other variables are the direct inputs to the process cheese food.

3.2 Constraints

In the model presented here in the casein: fat ratio of the cheese mik is 0.6925. A summing up of contraints for this model could be found in Table 3. Studies have shown relationships between the casein: fat ratio in milk, manufacturing conditions, and the resulting FDB, moisture in the nonfatty substance (MNFS), and fat and moisture percentage in the cheese. The greater control is needed when range is superior allowed during the manufacturing operation to maintain a consistent quality product. The general format for the constraint is:

$$\frac{Casein\ perentage\ of\ standardized\ milk}{fat\ percentage\ of\ standardized\ milk} = 0.6925$$

This ratio was chosen to establish the FDB of the natural cheese at 53.5%, which is the FDB of unstandardized milk in this example. Other ratios could be calculated from cheese yield formula:

$$FDB = \frac{FR(F)}{[FR(F) + CR(C)]SR}$$

where: FR = fat retention percentage divided by 100, F = fat percentage of standardized milk, CR = casein retention percentage divided by 100, C = casein percentage of standardized milk, and SR = salt solids retention factor.

The determination of final form of this constraint is considering all additions and subtractions separately from the cheese milk of casein and fat. (We may remove cream from the batch, for instance.)

The result for this case is:

Table 3. Constraints for the model.

1)	0.0002x1 + 0.0002x2 + 0.2977x3 + 0.2974x4 + 0.2731x5 + 0.0894x6 - 0.2977x7 ≥ 0 casein to fat ratio constraint
2)	X1 = 300,000.
3)	X2 = 300,000 use all milk constraints
4)	0.0789x1 – x3 ≥ 0
5)	0.0793x2 – x4 ≥ 0, limiting the amount of removed constraints.
6)	0.0977x1 + 0.0988x2 – 0.747x3 – 0.7472x4 + 0.4812x5 + 0.1588x6 + 0.7472x7 = M+S
7)	0.0055x1+ 0.0056x2 – 0.73x3 – 0.07x4 + 0.0013x5 + 0.0005x6 + 0.07x7 = Z2+Z1
8)	M+ B1 + B3+ B5 ≥ 0.70P. where P=(M+ B1 + B3+ B5+ y3+y4+y5+y6 Natural heese in the blend constraint
9)	y3= 0.03P emulsifier constraint.
10)	y4 ≤ 0.10P whey protein conentrate constraint
11)	0.0y3 + 0.015y4 + 0.17y5 + 1.0y6 + 0.55Z1 + 0.37B1+ 0.36B3 + 0.35B5 ≤ 0.43P moisture in the processed cheese onstraint
12)	0.0y3 + 0.35y4 + 0.83y5 + 0y6 + 0.45Z1 + 0.33B1+ 0.33B3 + 0.33B5+ 0.3371M ≥ 0.24P fat content onstraint
13)	-0.005x1 – 0.005x2 – 0.307x3 – 0.307x3 -0.307x4 + 0.85x5 + 0.15x6 + 0.307x7 ≤ 0 allowable solids constraint
14)	M + B1 ≥ 0.2 (M + B1 + B3 + B5) M + B1 ≤ 0.6 (M + B1 + B3 + B5) B3 ≥ 0.2 (M + B1 + B3 + B5) B3 ≤ 0.35 (M + B1 + B3 + B5) B5 ≥ 0.15 (M + B1 + B3 + B5) B5 ≤ 0.20 (M + B1 + B3 + B5) chees blend constraints

$$\frac{0.0244x1 + 0.0249x2 - 0.0139x3 - 0.142x4 + 0.28x5 + 0.092x6 + 0.0139x7}{0.0355x1 + 0.0357x2 - 0.45x3 - 0.45x4 + 0.01x5 + 0.0037x6 + 0.45x7} = 0.6925$$

Rearranging the equation yields:
-0.0002x1 + 0.0002x2 + 0.2977x3 + 0.2974x4 + 0.2731x5 + 0.0894x6 – 0.2977x7 ≥ 0.

The same format would be used for constraining the casein:fat ratio between two limiting values. The optimum casein: fat ratio is that which maximizes the net return of the process cheese product. accordingly, the cheese milk may or may not be standardized, depending on the profit contributions of the milk components and the natural cheese toward the final process cheese product. The casein: fat ratio has been constrained here to show the difference that could occur during manufacturing of natural cheese to maximize the value of the processed cheese product. A manufacturer might test several different casein: fat ratios to ascertain the impact different fat cheese have on the final processed product.
Constraint 2. xl = 300,000 and x2 = 300,000. These constraints require all the milk to be used.

Constraint 3. x3, x4, x5, x6 ≥ 0.

Constraint 4. 0.0789xl- x3 ≥ 0 and .0793x2 - x4 ≥ 0. These two constraints limit amount of cream that could be removed from the two milk resources. Silo 1 has 3.5% fat and 45% fat cream may be removed. In this case there are 3.5/45 = 0.078 lb of 45% fat cream available from each pound of 3.5% milk from silo 1. The constraint to limit the amount of cream removed is:

$$0.078x1 \geq x3,$$

and rearranged:

$$0.078xl - x3 \geq 0.$$

Constraint 5. The connecting link between the raw cheese manufacture and the blending of the process cheese ingredients is the cheese yield formula represented by the equation:

.0977xl + .0988x2 -.7472x3 -.7472x4 + .4812x5 + .1588x6 + .7472x7 =M + S

where: M = pounds of cheese manufactured and processed, S = pounds of cheese manufactured and sold, and M + S = total pounds of cheese produced.

Constraint 6. The whey cream by-product resulting from the cheese manufacture is:

$$WCY = \frac{(F - FR(F)) * WFR}{WF}$$

where: WCY = whey cream yield per 100 lb., WFR = whey fat recovery percentage divided by 100, and WF = fat percentage of whey cream divided by 100.

Constraint 7. There must be a constraint to maintain a certain amount of cheese is the rocess cheese blend. The total process cheese batch is represented by P, where:

P = M + B1 = B3 + B5 + y3 + y4 + y5 + y6.

Constraint 8. It is assumed that the process cheese batch size is known and constant. Thus P = batch size of process cheese blend.

Constraint 9.The amount and type of emulsifiers used for processing must be incorporated into the program in a similar manner as that done for cheese (Constraint 7).

Constraint 10. WPC is a supplement to the raw cheese and is generally accepted for its lower cost and the improved sensory qualities it could impart to the final process cheese food product. Because sensory qualities can be adversely affected by too much WPC, limits must be set (24). Management and experience play a major role in deciding the quantity to be used. In this example, WPC is limited to less than or equal to 10% of the total batch, where y4 is pounds of WPC.

$$y4 \leq 0.10P$$

Constraint 11. The moisture in the process cheese product must be constrained to comply with the egal standards of identity and the characteristics of the product as set by management. Because moisture adds to the yield of the product with next to no cost, the LP model will add as much moisture as it is
allowed by the constraints. therefore, only an upper limit constraint is needed. To ensure the final product has less than or equal to the legal moisture of 43% requires the following constraint:

$$0y3 + .15y4 + .17y5 + 1.0y6 + .55zl + .37B1 + .36B3 + .35B5 + .37M \leq .43P.$$

Constraint 12. Another constraint is necessary to ensure the fat content of the process cheese food meets the standard of identity for the specific product. Pasteurized process cheese foods must contain at least 23% fat. An extra I% is added for a safety margin. The general inequality is:

pounds fat in process cheese $\geq$ .24 (total proc-
ess cheese batch).

Constraint 13. A constraint on the allowable solids content of the raw milk used for cheese making has been included in this model [labeled (*)]. This has been done for several reasons. If milk concentrated by reverse osmosis or ultrafiltration is used, it is necessary to set a limit on the solids content of the cheese milk. Using reverse osmosis, Barbano and Bynum (1, 7) have shown that at about a 15% reduction in volume (about 14.17% solids content), the increased lactose of the cheese may become the limiting factor in producing a good quality aged Chedar cheese. They also go on to say that a low moisture barrel cheese used within 60 d may tolerate a reduction in volume of greater than 15%.

Constraint 14. Finally, there are several onstraints that must establish the acceptable age blends used in the processed cheese product. These age blends may be set as absolute or given as a range. For xample, an absolute age blend could consist of 60% l-mo-old cheese, 30% 3-mo-old cheese, and 10% 5-mo- old cheese. Alternately, an acceptable range could be 50 to 60% 1-mo-old cheese, 20 to 35% 3-mo-old cheese, and 15 to 20% 5-mo- old cheese. The advantage of a range of values is that it allows some flexibility in the program depending on cost, availability, grade of cheese, etc.

Thus, the 1-mo-old cheese, M and B1, could be greater than 50% but less than 60% of the total cheese used in processing.
Rewritten:

$$M + BI \geq .5\ (M + BI + B3 + B5)$$
$$M + B1 \leq .6\ (M + B1 + B3 + B5)$$

Likewise,

$$B3 \geq .2\ (M + B1 + B3 + B5)$$
$$B3 \leq .35(M + B1 + B3 + B5)$$

and

$$B5 \geq .15\ (M + B1 + B3 + B5)$$
$$B5 \leq .20\ (M + B1 + B3 + B5).$$

3.3 Objective funtion

The objective function is to maximize the net returns of a pasteurized process cheese product. Net return is defined as the difference between revenue and cost.

2.3.1 Net Process Cheese Profit. The process cheese batch is represented by the following equation:

$$M + B1 + B3 + B5 + Y3 + Y4 + Y5 + Y6 + Z1$$

If the selling price of block process cheese is $1.50/lb. the expression gives the revenue from its sale:

$$\$1.50\ (M + B1 + B3 + B5 + Y3 + Y4+Y5 + Y6 + Z1)$$

The ingredient costs for the processing materials are just their purchase prices. Whey cream produced from the Cheddar cheese manufacturing process has a "cost" value calculated from its foregone fat revenue. If whey cream fat is valued at $1.73/lb and 45% of the cream is fat, then:

$1.73/lb. fat * .45 fat in cream
=$.78 (value of fat/lb, cream)

The expression representing the cost of the processing ingredients is:

$$.535y3 + .4825y4 + 1.535y5 + .01y6 + .78zl.$$

The direct labor and overhead cost estimate is based on total pounds of process cheese produced. It is necessary to include these costs because process cheese production and natural cheese production are separate operations requiring different equipment, facilities, and labor. The inclusion of the labor and overhead estimate is written as:

$$\$.35\ (M + B1 + B3 + B5 + Y3 + Y4 + Y5 + Y6 + Z1).$$

Combined, this part of the objective function is written as:

$$1.15M + 1.15B1 + 1.15B3 + 1.15B5 + .615y3 + .6675y4 - .385y5 + 1.14y6 + .37zl.$$

2.3.2 Cost to Manufacture Natural Cheese. The cost to produce Cheddar cheese was not included directly in the profit from processed cheese to allow for the option of selling the manufactured cheese or processing it. There are the ingredients costs to consider as well as the direct labor and overhead that go into the manufacturing operation. The sweet cream represented by x3 and x4 have positive "costs" since any cream removed and sold is revenue (18). The sweet cream fat is valued at

$1.82/lb with 45% fat in the cream.
$1.82/lb fat * .45 fat in cream
=$.82 (value of fat/lb cream)

Direct labor and overhead costs are calculated on the total pounds of natural cheese produced, whether sold or processed.

$$.11278xl + .l181x2-.8x3 -.8x4 + .79x5 + .228x6 + .82x7 + .25\ (M+ S).$$

2.3.3 Cost to Buy Natural Cheese. Cost values are based on the estimated cost to buy Cheddar cheese of varying ages:

$$1.51B1 + 1.55B3 + 1.59B5.$$

2.3.4 Revenue from Natural Cheese (and Whey Cream).

$$1.49S + .78z2$$

Note that the sold cheese [S] value is $.02 less than the 1-mo-old cheese that could be purchased. It is assumed that it costs more to buy than is gained through sale of similar products. The objective function to be maximized is a combination of all four components:

$$-.1178xl -.1181x2 + .8x3 + .8x4 -.79x5 -.228x6 -.82x7 +.615y3 +.6675y4 -.38y5 + 1.14y6 +.37zl + 1.24S +.9M-.36B1 -.4B3 -.45B5.$$

4. Results

Results of applying genetic algorithms to the model described are presented in Tables 4. There

are several underlying assumptions common to the scenario.
First is that all raw milk resources are used in their entirety to make natural Cheddar cheese. A second assumption is that there is no waste. Third, the process cheese batch size is known and constant. Fourth, all ingredients are of known quality and composition and there is no problem with availability. Fifth, marginal revenue and costs are not applicable over the stated batch sizes.
The optimum input resources to maximize the profits of the process cheese manufacturing operation, their amounts and some costs are listed in Table 4.
Genetic Algorithms gives an economic evaluation of standardizing milk for cheese making. The objective is to maximize the profit of the natural cheese.
The constantly changing economic environment regularly influences the optimal solution to the model. A small change in the cost of an input resoure can dramatically change the optimal solution. An important question is how sensitive the solution is to these changes. It is important to know how close the unused decision variables (those held a value 0) came to be included in the optimal solution. If using a resource corresponding to a decision variable is obviously not profitable, then little extra effort is needed to accurately estimate its cost. Conversely, if a small change in a decision variable results in a new optimal solution, that decision variable and the constraints limiting ts use should be analyzed.

Table 4. The optimum formulation and profit potential that results from modeling the natural cheese operation.

	GA Results
Process cheese input resources	
y3 = Emulsifier	3,000
y4 = WPC	9,382
y5 = Fat	329
y6 = Water	17,288
z1 = Whey cream used	0
z2 = Whey cream sold	3,322
M = Manufacturing cheese	58,935
S = Cheese sold	16,935
B1 = Purchased cheese, 1 mo.	0
B2 = Purchase cheese, 3 mo.	17,500
B3 = Purchase cheese, 5 mo.	10,500
Total processed batch size, lb	100,000
Cheddar cheese yield, lb	58,935
Total Cheddar cheese cost	$ 85,522
Cost/lb Cheddar cheese	$ 1.4511
Objective function value	$ 4,365

5. Conclusion

The model is a guideline to assist in decision making. Its assumptions and solutions must be regularly tested as economic and manufacturing conditions change. Human judment is needed to evaluate the proposed solution and adjust it to the specific situation.
This paper illustrates the genetic algorithms to estimate parameters of the natural cheese process. The obtained parameters from genetic algorithms are investigated. GAs efficiently exploit past information to explore new regions of the decision space with a high probability of finding improved performance.

References:

[1] G.L. Kerrigan and J.P. Norbak. 1986. Linear programming in the alloation of milk resources for cheese making. J. Dairy Si. 69: 1432.
[2] K.L. Craig, J. P. Norback, and M. E. Johnson. 1989. A linear programming model intregated resource allocation and product acceptibility for processed chesse products. Journal of dairy science. 72: 3098 – 3108.
[3]A. Akramizadeh, A. Akbar Farjami, and H. Khaloozadeh, Nonlinear Hammerstein model identification using Genetic algorithms, *Proceeding of the 2002 IEEEInternational Conference on Artificial Intelligence Systems ZCAIS'02), 2002.*
[4] K.Benatchba, M. Kondil., H. Drias., and R. Belkacem. 2003. An adapted genetic algorithms for solving Max-Sat problems. WSEAS TRANSACTIONS on SYSEMS. Issue 4, Volume 2, October, ISSN: 1109 – 2777.
[5] D.E. Goldberg. 1989. The genetic algorithms insearch, optimization, and machine learning. New York; Addison-Welsey.
[6] A. Oonsivilai and B. Marungsri. 2008. Solving the unit commitment problem using an adaptive immune genetic algorithm. Int. Conf. on Power Control and Optimization, ICPCO. Chiang Mai, July 18-20. Thailand

[7] A. Oonsivilai and R. Oonsivilai. 2008 Genetic Algorithm Approach to Twin-Screw Food Extrusion Process Frequency Domain Parameter Estimation. WSEAS April 6-8, Hangzhou, China.

[8] A. Oonsivilai, and R. Oonsivilai. 2008 Parameter Estimation of Frequency Response Twin-Screw Food Extrusion Process using Genetic Algorithm. WSEAS TRANSACTIONS on SYSTEMS. Issue 7 Volume 7, July, ISSN: 1109 – 2777.

[9] S. Gunasekaran, and M. Mehmet Ak. 2003. Cheese rheology and texture. CRC Press. London.

[10] M. Mohebbi, J. Barouei, M. R. Akbarzadeh-T, A.R.Rowhanimanesh, M.Yavarmanesh, M.B.Habibi-Naja Modelling and optimization of viscosity in enzyme-modified cheese by fuzzy logic and genetic algorithm. Computers and Electronic in Agriculture. 62: 260 – 265.

[11] D. W. Everett , M. A. E. Auty. Cheese structure and current methods of analysis. Internation dairy journal. 18 : 759 – 773.

[12] Smith, J. P., B. Ooraikul, and E. D. Jackson. 1984. Linear programming: a tool in reformulation studies to extend the shelf life of English-style crumpets. Food Technol. Aust. 36:454.

[13]Marungsri, B., Meeboon, N., and Oonsivilai, A. 2006. “Dynamic Model Identification of Induction Motors using Intelligent Search Techniques with taking Core Loss into Account”. WSEAS TRANSACTIONS on POWER SYSTEMS Issue 8, Volume 1, August ISSN: 1790-5060.

[14]Pao-La-Or. P, Kulworawanichpong, T., and Oonsivilai, A. 2008. Bi-objective intelligent optimization for frequency domain parameter identification of a synchronous generator. WSEAS TRANSACTIONS on POWER SYSTEMS. Isuue 3, Volume 3, March, ISSN: 1790 – 5060.

[15]Oonsivilai, A. and Oonsivilai, R. 2008 Parameter Estimation of Frequency Response Twin-Screw Food Extrusion Process using Genetic Algorithm. WSEAS TRANSACTIONS on SYSTEMS. Issue 7 Volume 7, July, ISSN: 1109 – 2777.

[16]Oonsivilai, A. and Marungsri, B. 2008. Stability Enhancement for Multi-machine Power System by Otimal PID Tuning of Power System stabilizer using particle swarm optimization. WSEAS TRANSACTIONS on POWER SYSTEMS. Issue 5 Volume 3. May. ISSN: 1790-5060.