Crop Rotation Planning Using Simulated Annealing

by

Arif Shakoor

Project Report submitted to the faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

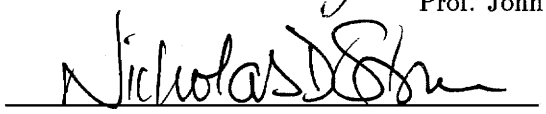
MASTER OF SCIENCE

in

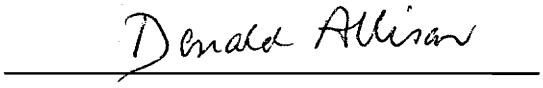
Computer Science

APPROVED:

Prof. John Roach, Chairman



Prof. Nicholas Stone



Prof. Donald Allison

February, 1995

Blacksburg, Virginia

c.2

Crop Rotation Planning Using Simulated Annealing

by

Arif Shakoor

Committee Chairman: Prof. John Roach

Department of Computer Science

(ABSTRACT)

The goal of this project is to explore the potential of *simulated annealing* in solving a combinatorial optimization problem in agriculture. The specific problem addressed in this project is the generation of optimal crop rotation plans for farms.

The whole-farm planning problem is a practical challenge faced by the farmers and environmental and federal agencies. The goal of farm planning is to come up with crop rotation plans that are environmentally safe and economically feasible. The farmers are encouraged, and in some cases mandated, to follow National Resource Conservation Services (NRCS) guidelines in cropping and tillage practices in order to reduce the loss of top-soil due to water erosion, reduce the risk of pesticide and nutrient leaching, and reduce surface runoff risk. The conflicting nature of these goals with the farmers' goals makes this a difficult problem and neither the agencies nor the farmers have the necessary tools to generate plans that satisfy the farmers' need while adhering to environmental restrictions.

This project uses *simulated annealing*, a method derived from statistical mechanics, to find acceptable plans for farms with a wide variety of local and global constraints. We define the farm planning problem, lay the framework of the annealing algorithm, use the annealing algorithm to produce plans for a specific farm, compare the results with those obtained by another AI-based heuristic technique, and determine the plan's feasibility in actual farming practice.

The framework upon which *simulated annealing* is based is very simple and most optimization problems can be formulated to fit this framework quite easily. Results obtained in this project are comparable with those obtained by another AI-based heuristic technique. Increasing the search space decreased the speed of convergence somewhat but for problems

like *farm planning* where time is not extremely crucial, *annealing* seems to be a viable optimization tool.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

This work would never have been completed without the infinite patience and encouragement of many people. I am greatly indebted to all of those who did not lose faith even though they had every reason to write me off.

I would like to express my gratitude to Dr. John Roach for being there whenever I needed help and guidance. He has done a tremendous job in trying to keep me focused during the long time it took me to finish this project.

I would also like to thank Dr. Nicholas Stone for giving me the opportunity to work with him on this project. His patience and suggestions were invaluable to me throughout the duration of the project.

I want to thank Dr. Donald Allison for agreeing to be on my committee and taking time out of his busy schedule to review my writeup on very short notice.

I would like to thank my friends, especially Rajiv and Randy, for having faith in me and encouraging me to finish this project.

Chapter 1

INTRODUCTION

This is the age of environmental consciousness. From the expanding ozone hole to the depleting of Amazon rain forests, from the planned toxic dumps to the accidental Exxon Valdez oil spill - the public has never before been more vociferous in its awareness and desire to preserve the environment. It is no wonder that more stringent legislation is being enacted in every facet of life in order to stem the ever-increasing pollution that is slowly but surely shifting the balance of nature so as to make this world a little less hospitable and habitable with each passing day. As a consequence of the public's growing concern over the environmental pollution issues, agricultural practices and management have come under scrutiny, and farmers are being required to file resource management plans. The goal of these plans is to strike a balance between the economic and practical needs of farmers and the environmental regulations of various federal and local agencies. Laws like the Federal Food Security Act of 1985, the Farm Bill of 1990, and Chesapeake Bay Preservation Act of Virginia are examples of central and local regulations that are being enacted in order to force farmers to abide by environmentally safe farming practices. United States Department of Agriculture (USDA) and Environmental Protection Agency (EPA) are mandating that farmers submit conservation plans for farms that have highly erodible lands in order to be eligible for USDA incentive programs. Agencies like the National Resource Conservation Services(NRCS), the Virginia Department of Conservation and Recreation (a division of Department of Soil and Water Conservation), and Agricultural Extension Services are burdened with the responsibility of evaluating, approving, and in most cases developing these plans. However, both farmers and government agencies lack the tools required to generate farm plans that meet farmers' goals while abiding by the regulations.

This project evolved out of the need to develop an efficient planning engine to solve the farm planning problem.

The goal of farm planning is to assign crop rotations to each field in the farm so that yearly production and economic objectives are met without violating environmental conservation rules. The problem is characterized by the presence of a two-level constraint hierarchy, local field-level constraints and global farm-level constraints. The global constraints are associated with the overall farm and are the required yearly yields, acreages for target crops and required profit from the entire farm. The local constraints determine the crop rotations that are possible for any particular field. We can calculate the maximum allowable level of soil erosion, pesticide and nitrate leaching, and surface runoff for each field, and using these local constraints we can build a list of valid crop rotations for each field. Generating a plan for the farm consists of choosing a crop rotation from the valid rotation list associated with each field such that the global constraints are satisfied. The crop rotation planning problem is characterized by an enormous search space resulting from all possible assignments of crops to individual fields. The number of plans to choose from grows exponentially and exhaustive search of the entire search space becomes infeasible even when the number of fields is in the low tens. The farm planning problem falls into the category of NP-complete problems for which no known polynomial time solution exists.

The goal of this project is to find an efficient solution to the farm planning problem that is a "good" estimate of the "optimal" plan. To measure the "goodness" of a solution we must have a way of assigning a quantitative measure to the solution. We shall call this quantitative measure the "score" of the particular solution. If the *score* of any optimum solution in the entire search space is "zero," our goal will be to find a solution whose *score* is as close to "zero" as possible.

In the farm planning problem there are many conflicting goals. From the farmers' perspective the primary goal is to maximize profit, while the government regulatory agencies aim at reducing soil erosion, chemical leaching, and surface runoff. The farmers may have preferences as to how much of certain crops they want to grow for domestic consumption,

while the agencies might mandate a minimum amount of the crop that the farmer has to grow in order to participate in federal farm programs. The terrain of the farm, climate, and soil types of the fields play an important part in determining which crops can be grown on a farm. The critical aspect of farm planning is to assign to each field a crop rotation such that:

- Adequate amounts of certain crops (corn, for example) are grown so as to feed the livestock
- Crops follow one another in an appropriate sequence so that leaching of pesticides and fertilizers is kept below mandated levels. Management practices include appropriate use of cover crops, reduced tillage, and controlling the timing and amount of pesticides and fertilizers to reduce risks of leaching and runoff.
- Soil erosion in each field is kept within the NRCS threshold
- Base acreage requirement is met for each target crop
- Storage capacity of the farm is not exceeded
- Fields with high yield potential are not used inappropriately (i.e., for pasture or conservation)
- The yearly profit is maximized.
- Special consideration should be given to the uniformity of production level of crops such that the variance remains small from year to year.

The duration for which a plan is generated is also important because if the *planning horizon* is too short, a good plan for this brief period could lead to long term disaster by putting the planner in a situation where plans for the next period will be severely restrictive. By using crop rotations that span a fixed number of years we can break down the planning process into manageable sub-plans. Once a good plan is found for the number of years in the rotation, we can repeat the plan multiple times to get plans of longer duration.

Even though farmers possess a great deal of common sense knowledge about what crop to grow in which field it is impossible for them to come up with a plan that meets their goals while satisfying all the constraints now being imposed on them by governmental agencies. For large farms with more than 20 fields it is often impossible to determine what the target yield/acreage for each crop should be. Underestimating the production capacity of the farm may lead to inefficient utilization of farm resources. Therefore, the goal of this research is to come up with a plan for farms for which we do not have an initial target in addition to being able to generate plans for farms for which targets are known.

Using traditional operations research (OR) optimization methods, that is, linear programming, to solve the farm planning problem becomes infeasible due to interdependencies of the various variables. Approaches to solving the farm planning problem that have been tried in the past include case-based planning, heuristic search and pruning, and constraint-based planning [15, 16]. This project takes an entirely different approach to the solution. By realizing that for large farms it is not always possible to determine the optimal plan, we adopted a method that looks for a "good" solution to the problem in a limited amount of time. We use the simulated annealing algorithm to solve the farm planning problem and get results that compared very well with results obtained from other methods, both in terms of quality of solution and time complexity.

The fundamental idea behind the annealing algorithm is derived by analogy from statistical mechanics where we want to predict the behavior of a system in thermal equilibrium at any given temperature. We are interested in predicting the state of the system as the temperature is gradually lowered starting from a very high temperature. At each successive lowering of the temperature, the atoms in the system will realign in order to lower the overall energy of the system. If the cooling is done slowly enough we can expect to reach *ground state* with a perfectly crystalline solid at thermal equilibrium. Metropolis et al.[7] introduced an algorithm to simulate the behavior of a collection of atoms in equilibrium at a given temperature. At each step of the algorithm an atom is given a small random displacement and the change in system energy, $\Delta\varepsilon$, is computed. If the change results in

a lower overall global system energy, i.e. if $\Delta\varepsilon \leq 0$, then the change is "accepted." If the change results in an increase in the global energy of the system, then this change is accepted with probability p given by

$$p(\Delta\varepsilon) = e^{\frac{-\Delta\varepsilon}{k_B T}}, where\ \Delta\varepsilon = \varepsilon_{new} - \varepsilon_{old}$$

and where ε_{new} is the energy of the new configuration, ε_{old} the energy of the old state, k_B the Boltzman constant, and T the system temperature. A random number is selected in the interval (0,1) and compared with $p(\Delta\varepsilon)$. If it is less than $p(\Delta\varepsilon)$, the new configuration is accepted, else, the original configuration is used to begin the next step. If we only accepted transitions for which the energy of the system decreases, we might end up in a state that is *meta-stable*. In a meta-stable state energy of part of the system will be locally minimized but the entire system will have a higher energy than in the ground state. In other words the solid will have imperfections frozen into the final crystalline form at various parts.

Combinatorial optimization can be viewed as a process analogous to the simulation of cooling a solid down by the Metropolis algorithm. The analogy between combinatorial optimization and the physical system (cooling a solid to its crystalline form) is shown in Table 1.1.

Table 1.1: Analogy between annealing and combinatorial optimization

Physical system (Annealing)	Optimization problem
Particles	Variables
Allignment of particles	Assignment of values to variables
State	Feasible solution
Energy	Cost
Ground state	Optimal solution
Rapid quenching	Local search
Careful annealing	Simulated annealing
Temperature	Control parameter

The search space of most optimization problems resembles a hilly landscape where the

height of each state is proportional to the score. The valleys correspond to local minima. By adding a random component to the selection process we escape from high local minima. In most cases, the system still takes a step downhill, but occasionally it will take a step uphill instead.

Annealing has been successfully applied to solve the problem of placement and global wiring of integrated circuits, folding, test pattern generation, designing error correcting codes, image restoration, logic minimization and compaction. Annealing can also be used to approximate solutions to graph theoretic problems like graph partitioning, graph coloring, and the steiner tree problem[10].

Chapter 2

LITERATURE REVIEW

This chapter is divided into three parts. The first part is devoted to the problem of planning and scheduling in AI and work done in these areas. The second deals with previous work in farm planning and the third consists of previous work in simulated annealing.

2.1 Planning

Planning in AI is defined as the process of sequencing a set of operators so that an initial state is transformed to the goal state by successive application of the ordered set of operators. The General Problem Solver (GPS) is an early search method that can be modified to generate plans. GPS uses means-end analysis to move closer to the goal. It compares the current state with the goal-state and finds the differences between the states. If there are no differences, we have reached the goal; otherwise GPS tries to apply an operator to the current state to reduce the differences. Each operator has a set of preconditions that must be true before it can be applied to the current state. The application of the operator changes the state of one or more attributes of the current state and takes us to a new state; one that is closer to the goal state than the current state. This classical approach to planning requires that we know how to evaluate the difference between states and are capable of deciding if the difference is leading in the correct direction - towards the goal. This is not always easy due to the interdependencies between the various attributes of the state. The Strips planner [4] is an adaptation of the GPS model. Later systems, like NOAH [12], work with partial plans instead of navigating through the entire search space. MOLGEN [13] uses constraint posting to coordinate the various interactions among

subplans. MOLGEN, like GPS, compares goals, finds differences, and chooses operators to reduce the differences. MOLGEN adopts a least-commitment strategy of deferring decisions for as long as possible. Recognizing the fact that subplans in a system interact, MOLGEN works to keep its options open, and reasons by elimination when constraints from other subplans become known. Initially it sets up an abstract plan and then refines that to a plan of specific steps. New constraints of increasing complexity are formulated as the planning process progresses. By not committing to a solution early on MOLGEN reduces the amount of unnecessary work required to backtrack from a dead-end. Planning in a temporal domain [2] applies constraints to time maps. Case-based planning [11] mimics human thinking in solving new problems by altering known solutions to the problem at hand. This requires the added capability to perform reasoning by analogy in order to find situations that are similar.

Planning is primarily concerned with constructing a linear chain of action with no emphasis on parallel execution of activities (Fox 1990). In many cases where the sequence of actions necessary to achieve the goal is known, the problem is to coordinate the actions in time so that conflicts between resource usage by different actions is resolved. For example, in a factory with a certain number of machines that manufactures a certain number of products, the sequence of actions needed to produce each product is known. But the difficult problem is to assign the machines to different products in such a way that the deadline is met for each product and idle time is minimized. In this scenario the planning part of the problem is known but instantiation of the plan is what we have to determine. Given a plan, *scheduling* selects among the alternative plans, and assigns resources and times for each activity so that they obey the temporal restrictions for the activities and the capacity limitations of a set of resources. ISIS job shop scheduling [5] uses hierarchical constraint satisfaction approach to the scheduling problem.

Farm planning involves both the planning and scheduling aspect of problem solving. Given a target acreage and yield for a number of target crops, we must decide when the various crops are to be grown subject to the constraints imposed by the farm resources.

Moreover, the duration between cultivating and harvesting is different for different crops, making the problem more temporal in nature. Earlier work in farm plannning involved OR type optimization techniques. CROPS [16] is a farm planner that uses three different approaches to generating whole-farm plans.

2.2 CROPS - a Crop Rotation Planning System

The first approach taken by CROPS was to use case-base planning. In this method the problem was tackled by applying known solutions to similar situations. Suppose, a farm has fields of sizes 10, 15, 25, 25, 50, 75, and 100 acres and we want to grow 25 acres of corn each year. We can see that this can be achieved by applying a rotation that has corn every other year to the fields of size 25 acres by offsetting the rotation by one year every second year. We can also achieve the same result by realizing that the fields of sizes 10 and 15 acres could be grouped together to get 25 acres of corn. So, in order to find a plan for a certain amount of corn we need to find two groups of fields each having an aggregate acreage that was equal to the required corn production. We can then use the rotation that has corn every other year in one group and use the same rotation, offset by one year, in the other group. Similarly, if we wanted to use a rotation that has corn every three years, we need to find three groups of fields with total acreage equal to the production goal. We could also use a combination of the two rotations by the following method. Use rotation with corn every other year on two groups of fields with total acreage of $\mathbf{X}$ and use crop rotation with corn every three years on three groups of fields with total acreage of $\mathbf{Y}$ such that $(\mathbf{X} + \mathbf{Y})$ was equal to the production goal. This method works well if the variables are limited and we are interested in only one crop. When multiple crops were introduced into the production goal this method fails to perform well and was ultimately abandoned.

The next method employed by CROPS was *brute-force and heuristic pruning.* This approach treated the problem as a rectangular matrix with the rows corresponding to fields and the columns corresponding to years and seasons. Heuristics were used to reduce the

number of valid rotations for each field. The goal was to fill the matrix with valid rotations for each field and then check to see if the plan was acceptable. This required that every valid rotation for each field be permuted with all the valid rotations for the rest of the fields to check for plans. The search space for this approach became enormous for even small farms.

The last method used by CROPS was based on the previous two methods and uses both heuristics and constraint-based reasoning. The planner is divided into three heirarchical levels in the spirit of Stefik's MOLGEN. In level 1, a list of all crop rotation is formed and valid rotation lists are formed for each field by eliminating rotations that violate field-level constraints like soil erosion, leaching, and surface runoff. In level 2, fields are grouped to form combinations that achieve the production goals of the farmer. Level 3 is where the actual assignment of crop rotations to individual fields take place. CROPS uses the field combination lists formed at level 2 and the valid rotation lists formed at level 1 to fill a rectangular matrix. At each step of the algorithm, a field combination or a crop rotation is placed in the two-dimensional template. Once a combination or crop rotation is placed in the template the remaining choice for combination and rotation is constrained. This two-dimensional constraint-propagation is continued until all of the template is filled. This process can be viewed as filling in a crossword puzzle. Each vertical or horizontal entry into the puzzle restricts the available choice of words for the remaining grids. In this scheme, if a failure happens at any stage of filling the template the planner backtracks to the last successful overlay of crop rotation and field combination and starts again. To further improve the efficiency of the algorithm, a *most-constrained-first* method is used to reduce the amount of back-tracking done by the system. Fields with few rotations and years with the minimum number of combinations are considered first. Currently, the planner uses partial arc-consistency methods that incorporate automatic constraint relaxation [14]. In this approach the variables are represented as nodes and the constraints as directed arcs between nodes. The goal is to instantiate all the variables without violating any of the constraints.

2.3 Simulated Annealing

Simulated annealing has been applied to the physical design of computers [6] ,including *partitioning*, *placement*, and *wiring*. *Partitioning* is the process of dividing the logic into small packages, *placement* is the process of actually assigning a logic to a specific location on chip, and *wiring* is the process of interconnecting the several logic components to one another. These three stages are very closely related and the choice of a design at any stage affects the design in the subsequent stages. Partitioning must be done such that the number of circuits in each partition can be easily accomodated in the available space while minimizing the number of connections that cross the partition boundary. Placement within a partition must try to minimize the length of connecting wires while avoiding congestion in any area.

Carnevali et al. [3] used simulated annealing to estimate the parameters necessary to describe a geometrical pattern corrupted by noise, to smooth a noise-corrupted bitmap image, and to create a bi-level image from a continuous one.

Annealing has also been used to solve graph theoretical problems like graph partitioning, graph coloring, traveling salesman, and the steiner tree problem. A comparative study of simulated annealing and other heuristic methods used to solve combinatorial optimization problem appears in [8].

The application of simulated annealing to combinatorial optimization problems was first envisioned by Kirkpatrick et al. [6] and has gained increased attention and application over the past decade. Theoretically, simulated annealing can be used to solve any optimization problem. It is well suited to problems for which a quantitative objective function containing the trade-offs between various variables can be formulated.

Chapter 3

THE FARM PLANNING PROBLEM

This chapter explains the farm planning problem in detail.

3.1 Problem Definition

The farm planning problem consists of three major components - the resources, the constraints, and the goals. The objective is to use the resources in accordance with the constraints to achieve the goals.

3.2 Resources

Land, equipment, capital, and labor are resources that are limited and have to be utilized optimally to achieve the goal.

3.3 Constraints

Constraints can be divided into two categories - local field-level constraints and global farm-level constraints.

Field-level constraints are factors that are dependent on individual field characteristics (e.g. soil type, slope length, angle, field size, soil composition, soil fertility, and previous or current crops). The local constraints are soil erosion caused by water, nitrate leaching, and surface runoff thresholds for each field that have to be adhered to. These constraints in turn limit the crop rotations that can be applied to the field. Soil erosion due to wind is not considered in this project.

The farm-level constraints are associated with the entire farm. Target acreages and yields for crops are two farm-level constraints that the farmer may want to satisfy. The target acreage could be the base acreage that the farmers are required to grow of a certain crop, henceforth called *target crop*, to be eligible for federal funding. The storage capacity of the farm dictates the amount of any crop the farmer may want to grow. Another farm-level constraint is to be profitable each year and the farmer may state a minimum amount of profit that must be maintained each year. The global constraints can also be viewed as the goals. Given the local constraints, the goal is to meet the yield and/or acreage requirements for each *target crop* and make a certain amount of profit each year.

3.3.1 Pesticide leaching and surface runoff risks

Pesticides used in fields pose the risks of being carried by rainwater along slopes to nearby waterways and/or percolating down to groundwater level. Contamination of waterways is harmful to aquatic organisms and groundwater contamination makes drinking water hazardous to humans. The soil types of a field and the pesticides used in that field determine the risk of surface runoff and pesticide leaching into adjoining waterways. Each field can be viewed as being composed of different soil types that constitute a certain percentage of soils in the entire field. Each soil type has associated with it indices for pesticide leaching, surface runoff, and nitrate leaching. These indices are used to determine whether a particular sequence of crops can be grown in a particular field without the risk of surface runoff and/or leaching.

3.3.2 Nitrate leaching

Fertilizers used in a field can cause surfacewater and groundwater contamination through soluble nitrate leaching. Each soil type in a county belongs to a particular hydrological group and each group is assigned a nitrate leaching index. The weighted nitrate leaching index of a field is used to classify fields as low, medium, or high risk. The planner must exclude rotations that use a high quantity of nitrogen fertilizer in a field with high or medium nitrate

leaching risk.

3.3.3 Soil erosion

The propensity of any field to be eroded by water is a function of aggregate soil types, slope length and slope percent of the field, crop cover and management practices, erosion control practices, and the amount of rainfall in the area. The total annual soil loss E in any field can be expressed by the following equation[16]:

$$E = RKL_sCP$$

where, R is the rainfall intensity in the area

K is the aggregate erosivity of the soil

C is the cropping factor that depends on the crop rotation

P is the special erosion control practice factor

L_s is a function of field slope length λ and slope angle θ

$$L_s = \left(\frac{\lambda}{72.6}\right)^m (65.41sin^2\theta + 4.56sin\theta + 0.065)$$

$$\text{where } m = \begin{cases} 0.2 & for\ slopes\ <\ 1\% \\ 0.3 & for\ 1-3\%\ slopes \\ 0.4 & for\ 3.5-4.5\%\ slopes \\ 0.5 & for\ \geq\ 5\%\ slopes \end{cases}$$

NRCS classifies a field as highly erodible (HEL) if

$$\left\{ \frac{RKL_s}{T}\ \geq\ 8 \quad \begin{array}{l} for\ \geq\ 33.33\%\ of\ the\ field\ acreage \\ OR\ for \geq\ 50\ acres\ in\ one\ field \end{array} \right\}$$

where, T is the estimated annual rate of soil formation. The numbers appearing in the above equations were determined empirically by the National Resource Conservation Services (NRCS).

3.3.4 Crop Rotations

Crop rotation is the sequence of crops that can be grown in a field. Table 3.1 shows some valid crop rotations. Each individual rotation extends over a particular period of time and can be combined together to form rotations that cover an arbitrary number of years. Rotation A is a one-year rotation that consists of corn followed by a grass/legume winter cover crop. It can be repeated to form rotations for any number of years. Rotation B consists of corn followed by wheat, millet and winter cover crop and lasts for 2 years. Rotation B can be repeated to form rotations that last a multiple of 2 years. We can also combine rotations to get hybrid rotations that can be repeated to form longer rotations. By combining rotations A and B we get a three-year rotation with two consecutive years of corn, followed by wheat, millet, and a winter cover crop. This rotation can be repeated, too, for other durations that are a multiple of three. Various factors limit the selection

Table 3.1: Corn-based crop rotation examples

Rotation	Sequence	Notes	Duration
A	Corn	Continuous Corn with a grass/legume winter cover crop.	1 year
B	Corn-Wheat	Corn every other year, alternating with a winter wheat followed by foxtail millet and a green manure cover crop	2 year
C	Corn-Barley(M)	Corn every other year, alternating with a winter barley silage, followed by foxtail and a green manure cover crop millet	2 years
D	Corn-Barley(S)	Corn alternating with barley silage double-cropped with sorghum followed by a green manure cover crop	2 years
E	Corn-Wheat-Alfalfa-Alfalfa	Rotation as in (B) but following millet with 2 years of alfalfa hay before going back to a green manure cover crop	4 years

of crops that can follow any crop in a sequence. The climate of the region and terrain of a field may preclude a particular crop from being grown. The soil type and risk to

pesticide leaching of the crops in a rotation may limit certain crops from being grown in consecutive years. For example, if the pesticide leaching and surface runoff risks are high for a particular field, corn can be grown for only one year followed by another crop in the rotation. The reason for this is, with continuous corn larger amount of pesticides is required to combat insects and weeds. This results in a higher risk of leaching and runoff. Furthermore, each rotation is assigned pesticide leaching and surface runoff indices that depend on the crops in the rotation. Table 3.2 shows how these indices and corresponding indices for the field determine if a particular rotation is acceptable for a field. The nitrate leaching index of a particular field and the amount of nitrogen fertilizer used for a rotation determines if that rotation is acceptable. If the nitrate leaching index of a field is high or medium and a crop rotation uses more than a certain amount of nitrogen fertilizer (700 lbs over 6 years), that rotation is rejected. By associating a penalty with each rotation we can discourage the planner from choosing rotations that are acceptable but not desirable. Penalty is assigned to a rotation based on the severity of constraint violation. Introducing rotations with penalty into the system allows the planner to automatically relax constraints that are deemed relaxable by the farmer.

Table 3.2: Planning strategies for combined soil and crop rotation risks to pesticide leaching and surface runoff

Field Soil	Crop Rotation	Is Rotation Acceptable	Penalty
Low	Low	Yes	0.00
Low	Medium	Yes	0.00
Low	High	Yes	0.05
Medium	Low	Yes	0.00
Medium	Medium	Yes	0.05
Medium	High	Yes	0.10
High	Low	Yes	0.00
High	Medium	Yes	0.10
High	High	No	N/A

Table 3.3: Planning strategies for combined soil and crop rotation risks to nitrogen usage

Field Nitrate Leaching Index	Nitrogen Usage	Is Rotation Acceptable	Penalty
Low	Low	Yes	0.00
Low	High	Yes	0.00
Medium	Low	Yes	0.00
Medium	High	Yes	0.10
High	Low	Yes	0.05
High	High	No	N/A

Each crop rotation is assigned a cropping factor, C, that represents the influence of rotation and tillage practice on soil erosion. The maximum rate of soil formation, T, is determined by the soil type of a field. A field consists of various soil types with different soil formation rate. The rate of soil formation for each type of soil is published by NRCS in its survey. The aggregate soil formation for a particular field, T, is the weighted average of all soil types present in the field. The composition of a field according to soil type is determined from County Soil maps. For each field in the farm the rate of soil formation, T, must be greater than or equal to the rate of soil loss, E. Using this relationship we can determine the maximum allowable C value for each field as follows:

$$C_{max} = \frac{T}{RKL_sP}$$

Any rotation with a C value greater than the C_{max} value of any particular field is also rejected. In effect, the planner must choose a rotation for a field such that the expected rate of soil formation is greater than or equal to the soil loss.

3.4 Goal

The goal is to assign crop rotations to each field in the farm such that the constraints are not violated and the yearly yield, acreage, and profit requirements are fulfilled.

3.5 Example

Consider the following simplified farm that has 6 fields with characteristics shown in Table 3.4.

Table 3.4: Field characteristic of a simplified farm

Field #	Acreage	C_{max}	L_s	T	K
1	50.0	0.019	4.3	3.135	0.24871
2	25.0	0.133	0.6	3.500	0.28015
3	25.0	0.039	2.1	3.500	0.28556
4	20.0	0.133	0.6	3.500	0.28015
5	5.0	0.184	0.4	3.500	0.30700
6	30.0	0.133	0.6	3.500	0.28015

We can formulate the following constraints:

Global - Grow 50 acres of Corn and 30 acres of Wheat each year. The acreage requirement can also be specified as a range.

Local - Table 3.5 shows the allowed rotations for the example farm.

Goal - The goal is implicit in the constraints. We must choose rotations such that the acreage requirements for the target crops are met.

In practice a lot of the constraints mentioned above cannot be fulfilled exactly. For example, it might not be possible to achieve the target acreage for a crop each year if leaching and C_{max} constraints are stringently adhered to. It is up to the farmer to relax one or more constraints so that the goal can be met. Relaxing one of the leaching indices or surface runoff constraints will result in more rotations being considered for each field and may result in a plan. The NRCS may have a threshold beyond which these constraints cannot be relaxed. Relaxing the C_{max} constraint allows rotations that result in soil loss that is greater than the rate of soil formation. C_{max} cannot be relaxed for fields classified as

HEL. If none of the above relaxations result in a plan the farmer has to change the acreage requirement for the target crops.

A valid plan for the example farm is shown in Table 3.6 on page 19.

Table 3.5: Valid crop rotations for example farm

Rotation #	C Value	6 year crop sequence
1	0.102667	*CCRCCWWMRCCRCCWWMR*
2	0.102667	*WMRCCRCCWWMRCCRCCW*
3	0.102667	*CCWWMRCCRCCWWMRCCR*
4	0.063667	*CCRCCBBSRCCRCCBBSR*
5	0.063667	*BSRCCRCCBBSRCCRCCB*
6	0.063667	*CCBBSRCCRCCBBSRCCR*
7	0.038000	*CCWWLLLLLCCWWLLLLL*
8	0.038000	*LLLCCWWLLLLLCCWWLL*
9	0.038000	*WLLLLLCCWWLLLLLCCW*
10	0.005000	*PPPPPPPPPPPPPPPPPP*

where C = corn grain, R = rye + hairy vetch cover crop W = wheat grain,
M = millet hay, P = pasture, B = barley silage,
S = sorghum silage L = Clover.
Each letter corresponds to a season, three seasons to a year.

Table 3.6: A plan for the example farm

Field #	6 year crop sequence
1	*PPPPPPPPPPPPPPPPPP*
2	*WMRCCRCCWWMRCCRCCW*
3	*CCRCCWWMRCCRCCWWMR*
4	*CCWWMRCCRCCWWMRCCR*
5	*CCWWMRCCRCCWWMRCCR*
6	*PPPPPPPPPPPPPPPPPP*

Chapter 4

THE ANNEALING ALGORITHM

4.1 Combinatorial optimization and NP-completeness

Optimization problems are concerned with finding the best configuration from a large set of feasible alternatives. There are two parts to any optimization problem - a set of feasible options and an *objective function* that is to be minimized(or maximized). The feasible options are obtained by imposing certain constraints on the configuration set. If the configuration set is finite the problems are called *combinatorial.* In combinatorial optimization problems, each configuration is assigned a *scoring* function or objective function that is a quantitative measure of the "goodness" of the configuration. The scoring function is dependent on a number of variables that constrain the set of feasible solutions. The combinatorial optimization problem can be formulated as follows:

Let	S	be the set of all feasible configurations,
	s	be an element of S, representing a particular configuration,
	ε(s)	be a function that provides a score for state s.

The goal is to find a state s_0 in S such that

$\forall s\prime \in S,\ [\varepsilon(s_0) \leq \varepsilon(s\prime)]$.

In the farm planning problem we are interested in assigning a crop rotation to each field in the farm such that the yield for each target crop is met. Let us suppose that our farm

consists of N fields and each field can be assigned one of R different rotations. The state set S consists of all R^N different assignments of individual rotations to the fields. The score ε of any state can be calculated by determining how close the actual yield is to the target yield. The computational time required to determine the optimal assignment of rotations increases exponentially with N, so that an exact solution to the farm planning problem with any practical value of N becomes intractable in any realistic time for computation. The farm planning problem belongs to the class of NP-complete (nondeterministic polynomial time complete) problems whose worst-case solution time cannot be bounded by a power of the size of the problem instance (N).

The nature of NP-complete problems that arise in practical life, like the farm planning problem, is such that deterministic optimization algorithms are not feasible due to the enormous search space resulting from the presence of very many independent variables. In applications where solution time is critical, one sometimes sacrifices optimality for speed and settles for a "good" approximation instead of an "optimal" solution. Therefore, solutions to these problems can be viewed as the task of finding a good or if possible best configuration from a large set of feasible options in a reasonable computation time. Algorithms that give an incorrect result with very small probability but are guaranteed to converge to a result within a reasonable computation time are called Monte Carlo algorithms. These algorithms are probabilistic in the sense that they include steps that do not depend only on the input but also on the outcome of some random events. The following sections describe a few Monte Carlo algorithms that are often used to solve NP-complete problems. All of these algorithms can be used to generate approximate solutions to optimization problems for which no polynomial-time algorithm exists that will generate the optimal solution.

4.1.1 Blind random search

A simple and naive approach is blind random search. In this algorithm, a generation procedure produces a random configuration from the set of all possible configurations. The score of each state is evaluated and if the score of a new state is lower than the best score

encountered so far, the new score replaces the previously best score. This is repeated a predetermined number of times or until a certain amount of time has elapsed.

4.1.2 Iterative improvement

Iterative improvement is a modified version of blind random search. In this approach, a neighborhood structure is defined for each state. The neighborhood of any state is the set of states that can be reached from that state through a small modification of the state. A *move* procedure is defined that produces a neighbor given any state. Iterative improvement proceeds as follows. First generate a random configuration as in the blind random search. This configuration is stored as the initial configuration. Then use the move procedure to produce a neighboring configuration and evaluate the new configuration. If the score of the new configuration is lower than the score of the current configuration it takes the place of the current configuration. The process of modifying the current configuration is continued until no improvement occurs after a certain number of modifications. The current score is compared with the score of the best configuration encountered, and if it is lower, the current configuration takes the place of the best configuration. If the maximum allowable time is not used up another current configuration is generated randomly, and the modification process starts again.

4.1.3 Probabilistic hill climbing

In the above algorithms a new state is selected as the current configuration only if the score of the new configuration is lower than the score of the current configuration. Another group of algorithms, known as probabilistic hill climbing algorithms, allows transitions that result in an increase in the scoring function. Probabilistic hill climbing algorithms work in the following manner. States are generated as in the iterative improvement algorithm, but instead of comparing the score of the current configuration with the score of the newly generated one, an acceptance function is evaluated on the basis of the two scores and a random number. If a true value is returned by the acceptance function the newly generated

configuration becomes the current configuration with a non-zero probability that the new score is greater than the current score. Thus the algorithm has the ability to escape local minima. By changing the acceptance function dynamically, we can restrict the "uphill movement" during the course of the program execution. The acceptance function can be influenced through some control parameters. The name "hill climbing" is suggestive of the nature of the search space. We can visualize the search space as a hilly landscape where the height of each state is proportional to the scoring function. The valleys correspond to local minima and in iterative improvement no change in height is possible when we are at a valley. Hill climbing algorithms allow us to escape local minima by allowing transitions that result in an increase in the scoring function, or in other words let us climb out of a valley eventually to find a deeper valley [10].

Simulated Annealing is a variety of probabilistic hill climbing algorithm that is used in this project. In the next section we discuss the *annealing algorithm* in detail and show how the farm planning problem can satisfactorily be addressed by using *simulated annealing.*

4.2 Simulated annealing

In condensed matter physics, annealing is the thermal process by which the low energy state of a solid is obtained by first heating the solid until it melts and then gradually cooling the temperature of the solid. In the liquid phase all particles of the solid arrange themselves randomly resulting in a high energy state. In the ground state the particles are arranged in a highly structured lattice and the corresponding energy of the system is minimal. The minimal energy state is reached only if the solid is heated to a very high temperature initially and the cooling is done very slowly. Otherwise, the solid will be frozen into a meta-stable state possessing a higher energy than the ground state [6]. At each successive lowering of temperature the configuration of the system changes by a small amount resulting in a small change in the energy of the state. In annealing , the global process of minimizing the energy of the system is achieved by a localized process of energy modification.

In simulated annealing we start with a state in the search space and use a scoring function in place of the energy of the system (See Table 1.1). The temperature is simply a *control parameter* that has the same units as the scoring function. The idea is to start at a high value of the control parameter, T, and gradually lower T until no change in the scoring function occurs. At each successive lowering of the control parameter the Metropolis simulation is performed long enough for the system to reach a steady state. Simulated annealing can be used to solve problems in combinatorial optimization in the following way[9].

1. Randomly select an initial state s_i. Select a large value for the initial control parameter T. Choosing a large initial value of T lets the system move around freely during the initial stages of the process by accepting all proposed transitions. Usually T is chosen such that $exp(-(\varepsilon_{new}-\varepsilon_{old})/T)$ is greater than or equal to 0.999 for all energy changes.

2. Select a neighbor state s_n of the current state by modifying the current state by a small amount.

3. Calculate $\Delta\varepsilon = \varepsilon(s_n) - \varepsilon(s_i)$.

4. If $\Delta\varepsilon$ is < 0, then s_n becomes the current state. Go to step 6.

5. If $\Delta\varepsilon$ is ≥ 0, then s_n becomes the current state with probability $exp(-\Delta\varepsilon/T)$. This is done by selecting a random number R between 0 and 1 using a uniform probability density function. If R is $< exp(-\Delta\varepsilon/T)$, then s_n is accepted as the current configuration, else the current configuration is unchanged.

6. If M transitions are accepted for which the energy ε of the system dropped or N (N $\gg$ M) total modifications to the system have been made since the last change in the control parameter then update T by multiplying it by a, which is typically a constant between 0.8 and 0.999 else go to 2.

7. If the value of ε has not decreased by more than epsilon (a small constant) in the last L(L $\gg$ N) iterations, then stop; else go to step 2.

Steps two through six above constitute the Metropolis loop. During execution of each Metropolis loop the value of the control parameter remains unchanged.

It can be seen from the above discussion that annealing can be used to solve any combinatorial optimization problem that has a unique representation of each state, a scoring function that adequately reflects the goodness of any state, and a neighbor relation that allows smooth transition between adjacent states. However, it must be noted that not all optimization problems can be solved efficiently by the annealing algorithm. For some problems the control parameter has to be lowered so slowly that other heuristics are more efficient. The traveling salesman problem is one such problem for which heuristics exist that consistently outperform simulated annealing.

4.3 Crop Rotation Planning

This section discusses how we can formulate the crop rotation problem to fit the simulated annealing technique. We can summarize the critical elements of the simulated annealing algorithm as follows :

1. Definition of the search space.
2. A neighborhood structure that allows a state to move to another state. This requires the formulation of a move function.
3. A scoring function, depicting the goodness of a state.
4. A selection function that randomly selects a part of the system to be changed.
5. An acceptance function that stochastically accepts the change to the selected component.
6. A termination criterion for the Metropolis loop.
7. A schedule for updating the temperature of the system at each iteration of the inner loop.

8. A stop criterion for the termination of the algorithm.

4.3.1 Definition of the Search Space

Each state in the search space for the crop rotation problem corresponds to the assignment of a rotation from the valid rotation list of each field. The search space consists of all possible permutations of crop rotation assignments to the fields. Let N be the number of fields in the farm, R_i be the list of valid rotations for field i, and r_i $(1 \leq r_i \leq M)$ be the number of valid rotations for field i $(1 \leq \text{i} \leq \text{N})$. Then the search space, S, is given by,

$$R_i = \{r_{i1}, r_{i2}, \cdots r_{ik}\}; \quad 1 \leq k \leq M \tag{4.1}$$

$$r_i = |R_i| \tag{4.2}$$

$$S = \prod r_i; 1 \leq i \leq N \tag{4.3}$$

A state, s, of the search space S is formed by choosing a rotation r_{ik} $(1 \leq k \leq r_i)$ for each field i $(1 \leq \text{i} \leq \text{N})$ of the farm. In the worst case each field can have a maximum of M valid rotations that can be assigned to it. In this case the search space is M^N. For a farm with 10 fields and 10 valid rotations for each field the search space will have 10 billion states. In equation 4.2 r_i corresponds to the number of valid rotations for field i. The planner starts with a basic set of rotations that spans the planning horizon. A comprehensive list of crop rotations can be attained by shifting the crops in a rotation by one or more years to form new rotations. For example, a 3-year crop rotation can have three starting points leading to 3 eligible rotations. The valid rotation list is a list of all eligible rotations that are accepted for a particular field after applying field-level constraints.

4.3.2 Neighborhood of a State and the Move Function

The move function generates a new state $s\prime$ from the current state s by choosing a new rotation for a randomly selected field. The function first selects a field i at random and picks a rotation from the set of valid rotations for field i. There is a finite probability that the new state generated is identical to the current state since we can select the current rotation

for field i with probability $1/r_i$. The neighborhood of state s is defined as the set of all states that can be reached from state s via a single move. So we can define a *neighborhood* as

$$neighborhood = \sum_{i=1}^{N} r_i \tag{4.4}$$

4.3.3 Scoring Function

The annealing algorithm is very much dependent on the scoring function, since transition between states and the final state is guided by the changes in the score of the intermediate states. Because the scoring function reflects the "goodness" of a state we would like it to mirror the constraints of the system. The local field-level constraints are used in the system to generate the valid rotations for each field and together with the global farm-level constraints are used to evaluate the score of any state. Given target yields yt_j and target acreages at_j we would like the scoring function ε to be small for states with actual yields and acreages (ys_{jk} and as_{jk}) close to the target. On the other hand, we would like the score to be large for states for which the yields and acreages vary widely from the targets. The field-level constraints are incorporated into the scoring function as penalties associated with valid crop rotations. A rotation that does not violate any conservation regulation receives no penalty points. A rotation is awarded penalty points if the acceptable level of erosion, pesticide and nitrate leaching, and surface runoff is exceeded for any field. If a rotation violates any of the field-level constraints beyond an acceptable threshold that rotation is eliminated from the valid rotation list of that particular field. For the rotations that are accepted with a penalty a different weighting factor is used for each violation in order to reflect the relative importance of the violations. The penalty for income is an exponentially decaying function so that plans with very low income are severely penalized but as we approach the theoretical maximum the penalty flattens out. This is done so that we do not over-emphasize the economic goal and sacrifice other goals such as minimizing

leaching and soil erosion. We define the score of a state as follows:

$$\varepsilon = \frac{1}{Y}\sum_{k=1}^{Y}\left(\sum_{j=1}^{C}\left(\frac{|yt_j - ys_{jk}|}{yt_j} + \frac{|at_j - as_{jk}|}{at_j}\right) + \exp\left(is_k \times \frac{\ln(p_{pmax})}{it_k}\right)\right) + \sum_{i=1}^{N} p_i$$

where C is the number of target crops

N is the number of fields in the farm

Y is the number of years in planning horizon

yt_j is the annual target yield for crop j

at_j is the annual target acreage for crop j

ys_{jk} is the actual yield of crop j in year k

as_{jk} is the actual acreage of crop j in year k

it is the target profit in year k

is_k is the actual profit in year k

p_{pmax} is the penalty when actual profit is equal to target

p_i is the penalty associated with the rotation for field i

The importance of profit in the scoring function can be adjusted by changing the p_{pmax} value. If we set this value to a very small number, profit greater than the target amount does not positively influence the score. Setting the p_{pmax} to a large value increases the importance of profit in the calculation of the score. p_{pmax} was experimentally set to 0.000001 during the project. In the experiment either the yield component or the acreage component was used in the calculation of the score , as the two terms are closely related .

At present we can attempt to maximize any of the goals of yield, acreage, or profit but do not have a framework to allow farmers to specify a target for one or more of the goals while maximizing others. For example, the farmers may want to keep yield of a target crop within a certain range while receiving maximum return from the farm.

Scores for plans with no target

If the target yield or acreage is not known for a farm the planner uses a scoring function that tries to maximize yield for each *target crop* while maintaining uniformity over the years. The scoring function is now modified to reflect the standard deviation of yield. We want the standard deviation to be as small as possible. Keeping yield uniform accross the years is important to farmers as they have to find adequate storage, harvesting equipment, labor, transportation, and fertilizer. Storage requirement that does not vary widely over the years is desirable because the farmer can plan ahead of time and build extra storage facilities if needed. If we make the scoring function to be just the standard deviation we could end up with a plan that is uniform over the years but fails to utilize the resources by not producing enough of the target crops. In order to force the planner to maximize production, we penalize scores with low yield by adding an amount by which the yield falls short of the maximum attainable yield. The maximum yield for a target crop is determined by the total area of the fields that are arable (have at least one rotation other than pasture in their valid rotation lists) divided by the number of target crops. The scoring function for farms with target yields not known is shown below :

$$\varepsilon_{no\_goal} = \frac{1}{C}\sum_{j=1}^{C}\frac{|amax - \bar{E}_j| + \sigma_j}{amax} + \sum_{i=1}^{N} p_i \tag{4.5}$$

where,

$$\sigma_j = \sqrt{\frac{\sum_{k=1}^{Y} as_{jk}^2 - \frac{1}{Y}\left(\sum_{k=1}^{Y} as_{jk}\right)^2}{Y-1}} \tag{4.6}$$

$$\bar{E}_j = \frac{1}{Y}\sum_{k=0}^{Y} as_{jk} \tag{4.7}$$

and C is the number of target crops

N is the number of fields in the farm

Y is the number of years in planning horizon

$\bar{E}_j$ is the mean acreage of crop j in Y years

σ_j is the standard deviation of acreage for crop j

$amax$ is the maximum attainable acreage for any crop

as_{jk} is the actual acreage of crop j in year k

p_i is the penalty associated with the rotation for field i

Although equation 4.5 was used in the project to maximize yield of a target crop we can easily modify the equation to maximize profit. Experiment with this scoring function for maximizing profit led to plans that had yields that varied widely from year to year.

Penalty For Crop Rotation

Each field has a list of rotations that can be used in a plan. Some of these rotations have a penalty associated with them to discourage the planner from using them in place of better rotations. Table 3.2 and Table 3.3 show how rotations are either rejected or accepted with a penalty based on nitrate leaching, pesticide leaching, and surface runoff risks. Rotations are also penalized if they violate the C_{max} constraint for a field. The following algorithm is used to assign a penalty to a rotation based on the C factor of the rotation and the field:

```
{
    if (C_rot ≤ C_max)
        Accept Rotation ;
    else if ( (field is HEL) and (C_rot > C_max) )
        Reject Rotation ;
    else if (C_rot > C_max × 1.15)
        Reject Rotation ;
    else
        Add 0.05 to Penalty for rotation
}
```

4.3.4 Selection Function

The selection function or *move* function lets the system migrate from the current state to a new state. Given a current state, all of its neighbors have an equal probability of being chosen as the next state. The *move* function chooses a field at random and then assigns

a new rotation to the chosen field from the valid rotation list of that field. The selection probability, β, is given by,

$$\beta = \frac{1}{\sum_{i=1}^{N} r_i} \tag{4.8}$$

4.3.5 Acceptance Function

Once we have selected a new state from the current state, we accept the transition from the current state to the new state if the score of the new state is lower than the score of the current state. If the selected state has a higher score than the current state, we accept the transition with probability $exp(-(\varepsilon(s_n) - \varepsilon(s_i))/T)$, where $\varepsilon(s_n)$ is the score of the selected state, $\varepsilon(s_i)$ is the score of the current state, and T is the temperature. To decide whether to accept the proposed transition, we generate a random number R between 0 and 1 and compare it with $exp(-(\varepsilon(s_n) - \varepsilon(s_i))/T)$. If R is smaller we accept the transition and s_n becomes the current state, else the current state is unchanged. If the transition is accepted, we compare the score with the best score encountered so far and store the new state as the best state if the score is lower than the best score.

The overall acceptance probability can be summarized as follows:

$$\text{p(accept } \varepsilon_{new}) = \begin{cases} 1 & if \Delta\varepsilon \leq 0, \\ e^{\frac{-\Delta\varepsilon}{T}} & if \Delta\varepsilon > 0. \end{cases}$$

4.3.6 Metropolis Loop(Inner Loop) Termination Criterion

The inner loop of the simulated annealing algorithm generates a new state from the current state using the move function and either accepts the transition to the new state or remains at the current state depending on the acceptance function described above. The quality of the final result depends on the number of times the loop is executed for any value of the control parameter, T. Initially, when the value of T is high, most of the transitions are accepted and so we loop for a small number of times. As T gets smaller, fewer and fewer transitions are accepted and the variance in the score gets smaller as we approach a

minimum and we have to execute the loop more times to get to a lower energy state. This is accomplished by keeping track of the number of accepted transitions and the variance in score for a particular value of the control parameter. As fewer transitions are accepted and the score variance decreases, the loop is executed more times. An upper limit on the number of times the inner loop is executed is set so that if we near the global minimum, when almost none of the proposed transitions are accepted, we do not cycle through the loop an arbitrarily large number of times.

The *metropolis loop* is characterized by k_n, the number of new states generated in the n_{th} iteration; k_n can be parameterized by using the local and global accessibility of the search space. The global accessibility (H_n) is a measure of the size of the likely search space and the local accessibility (h_n) is a measure of the access to the neighboring states[10]. k_n is defined to be the ratio of global and local accessibility multiplied by a constant (C) :

$$k_n = \frac{H_n}{h_n} \times C \tag{4.9}$$

Initially, all states are equally likely and $H_0 = \ln(|S|)$ and $h_0 = -\ln\beta$ where β is the selection probability. After each execution of the *metropolis loop* the values of H_n, h_n, and k_n are recalculated. H_{n+1} and h_{n+1} are updated using the following equations[10]:

$$H_{n+1} = H_n + \frac{\bar{E}_n - \bar{E}_{n-1}}{T_n} \tag{4.10}$$

$$h_{n+1} = a \times \ln(\frac{1}{\beta}) + \frac{\bar{U}}{T_n} - (1-a)\ln(1-a) \tag{4.11}$$

where, T_n = control parameter for n_{th} metropolis loop

a = $\frac{|s_i|}{k_n}$ such that $s_i \neq s_{i-1}$

$\bar{E}_n$ = average score for nth loop

$\bar{E}_{n-1}$ = average score for (n-1)th loop

$\bar{U}$ = $\overline{max(0, \Delta\varepsilon)}$

In equation 4.11 "a" is the fraction of moves for which the state changed and $\bar{U}$ is the mean of $\Delta\varepsilon$ for transitions that resulted in an increase in the system energy.

We experimented with keeping k_n fixed throughout the simulation. For this case k_n was chosen to be equal to the cardinality of the neighborhood of a state. The fixed k_n value is given by:

$$k_n = \sum_{i=1}^{N} r_i \tag{4.12}$$

4.3.7 Initial Value of Control Parameter

The initial value T_0 of the control parameter must be chosen such that virtually all proposed transitions are accepted. This corresponds to the initial high temperature of a physical system. The acceptance ratio χ is defined as the ratio of accepted transitions to proposed transitions. We want the initial acceptance ratio χ_0 to be very high to simulate the initial state of the system where almost all transitions are accepted. The initial value of the control parameter, T_0, is determined by the following method. Assume that a sequence of trials is generated at a certain value T of the control parameter. Let m_1 denote the number of proposed transitions from i to j for which $\varepsilon(j) \leq \varepsilon(i)$, and m_2 the number of transitions for which $\varepsilon(j) > \varepsilon(i)$. Also let $\overline{\Delta\varepsilon}^{(+)}$ be the average difference in score for the cost-increasing transitions. Then the acceptance ratio χ can be approximated by the following expression:

$$\chi \approx \frac{m_1 + m_2 \times exp(\frac{-\overline{\Delta\varepsilon}^{(+)}}{T})}{m_1 + m_2}, \tag{4.13}$$

from which we obtain:

$$T = \frac{\overline{\Delta\varepsilon}^{(+)}}{ln(\frac{m_2}{m_2 \times \chi - m_1(1-\chi)})}. \tag{4.14}$$

The initial value of T_0 can be calculated from 4.14 in the following way. Initially, T_0 is set equal to zero. Next, a sequence of m_0 trials is generated. After each trial, a new value of T_0 is calculated from 4.14, where X is set X_0, the *initial acceptance ratio.* In this

project the size of the neighborhood of the searc space was used as the value of m_0. The value of T at the end of m_0 trials was used as the initial value of the control parameter, T_0.

4.3.8 Cooling Schedule

The values of the control parameter (T) used during the simulation in the inner Metropolis loop constitute the cooling schedule. The metropolis loop inherits its name from statistical mechanics where T represents the temperature of the solid and the simulation consists of gradually cooling the solid to its ground state. The value of the control parameter, T, is lowered for successive runs of the inner loop. Initially, when the current state moves around freely in the search space, T is reduced by half. The free motion in the search space is determined by the high variance in score calculated for the inner loop during this phase. As we start approaching the global minimum, the variance in score drops and accordingly T is reduced by a smaller fraction. The following cooling schedules were used in the project:

Schedule 1 $T_{n+1} = T_n \times 0.95$

Schedule 2 $T_{n+1} = \begin{cases} T_n - \frac{T_n}{2} & if\ 2 \times \gamma \times T_n > \sigma, \\ T_n - \frac{\gamma \times T_n^2}{\sigma} & else. \end{cases}$ (See reference [10])

Schedule 3 $T_{n+1} = \frac{T_n}{1+(T_n \times \ln(1+\gamma))/3\times\sigma}$ $(See reference [1])$

where, T_n = control parameter for n_{th} metropolis loop

T_{n+1} = control parameter for (n+1)th loop

γ = 0.03 (chosen so that $\gamma \times t_0 \ll \sigma_0$)

σ^2 = variance in the scoring function in n_{th} loop

4.3.9 Stopping Criteria for Termination of Algorithm

The algorithm stops if no improvement in the score is encountered or no transition is accepted during a predefined number of runs of the inner loop. The following methods were used in the experiment:

Stopping criterion 1 No improvement for 100 inner metropolis loop.

Stopping criterion 2 $\sigma \times \sigma < \theta \times t_T \times (\bar{E} - \bar{E}_n)$ (*See reference*[10])

Stopping criterion 3 No change in current state for three inner metropolis loop iterations[1]

where T_n = control parameter for n_{th} metropolis loop

T_{n+1} = control parameter for $(n+1)_{th}$ loop

$\bar{E}$ = average score during initial run of 1000 random configurations

$\bar{E}_n$ = average score of n_{th} run of the metropolis loop

θ = 0.01 (1% precision of optimum)

Chapter 5

RESULTS

The simulated annealing algorithm was applied to a particular farm in Montgomery County, Virginia. This farm will henceforth be called the MC farm. The primary products of the MC farm are corn, hay, and small grains to support a cattle-finishing operation of approximately 700 head per year. One tract of the complete farm, with varying degrees of slope, soil type, size and erodibility, was focused upon in this experiment. The tract under scrutiny consists of 405 acres of arable land divided into fourteen fields. The data used in this experiment are shown in Appendix A. The primary objective of a plan for this farm has been to grow enough corn to feed the livestock. Several of the fields in the MC farm were labelled as HEL by the NRCS in 1988 and in order to be eligible for federal funding programs, the farmer was mandated to file a plan for the farm that would reduce the soil erosion rate in the HEL fields. The farmer's attempts to reduce soil erosion have resulted in a decrease in the yield of the farm.

A planner using simulated annealing was used on the MC farm to generate plans to produce 5000, 7800, and 9500 bushels of corn. The planner was run on a Decstation 5000/240 running under Ultrix 4.3A. Initially sixteen different combinations of cooling schedule, metropolis loop termination criterion, and program termination criterion were used in the experiment for a target yield of 7800 bushels of corn and the results are shown in Table B.1 of Appendix B. Optimization is viewed as a minimization problem in this project and the smallest score reflects the best solution generated by the annealer. Combination 4 of Table B.1 performed the best on an average and was used to generate plans for the other goals. Performance is measured in terms of execution time, low average score, and low standard deviation of score.

For a target yield of 5000 bushels of corn, soil erosion, pesticide leaching, and nitrate leaching were all deemed important. The plan generated for 5000 bushels of corn is shown in Table B.2. For 7800 and 9500 bushels of corn the planner was run with the pesticide leaching constraint as unimportant. The results are shown in Tables B.3 and B.4. Table B.5 shows the result of specifying two target crops - corn and wheat while Table B.6 shows a plan for which no target acreage was specified for corn. In this case the planner tries to generate the maximum acreage attainable while being consistent over the years. Table B.7 shows a plan for two target crops - corn and wheat without any stated acreage goal. All the plans met the goal while being profitable and reducing soil erosion. The penalty scheme also worked well in keeping rotations with high risk of pesticide and nitrate leaching from being assigned to plans.

The simulated annealing approach of this project compared well with results from CROPS [16] in terms of quality of solution. The comparisons between the result obtained by using simulated annealing and CROPS are shown in Figures B.1- B.6. Both methods were employed on a larger farm (WC farm) in Wythe county, Virginia and the results are shown in Figure B.7. The search space for the WC farm is $\approx 1.5 \times 10^{29}$. Plans generated by CROPS have received favorable responses from the farmers and meet conservation guidelines. The results from the annealer were always as good and in some cases better than the ones generated by CROPS and so are equally attractive to the farmers. For fixed targets the plans generated by the annealer were closer to the target while being more profitable. When the target was specified as a range both methods generated plans that were within the range but the annealer always generated plans that made more profit.

A planner using iterative improvement was also used to generate plans for the MC farm. Schedule 4 was used to generate plans for 7800 bushels of corn and the time needed by the annealing algorithm was used as the time allowed for an iterative improvement algorithm to generate plans. For the MC farm, where the search space is $\approx 1.5 \times 10^{16}$, the plans generated by the annealing algorithm had an average score of 0.0238 and the average score of plans generated by iterative improvement is 0.030322. The search space can be calculated

by using equation 4.3. The same process was carried out on the WC farm for 54-56 acres of corn.

Both the MC and WC farms are severely constrained due to the high slope percentage of most of the fields . As a result a lot of the fields are classified as HEL and valid rotations for most fields are quite limited. In order to explore the effect of larger search space on the quality of result and time required to generate plans, we changed the field data of the WC farm so that some of the highly constrained fields allowed more rotations resulting in a larger search space. The results obtained for various search spaces using both the annealer and the iterative improvement algorithm is shown in Table B.9. The annealing algorithm always generated better plans than the iterative improvement algorithm. For very large search spaces ($\gg 10^{35}$), the quality of solutions for the annealer deteriorated somewhat in terms of speed. For the search space of 1.3×10^{49} the annealer was rerun for the combinations in Table B.1 and combination 5 gave better result than combination 4 but took longer time. The result of this run is shown in Table B.10. This result suggests that the overall schedule to be used on a particular instance of the problem can be chosen at runtime depending on the size of the search space. We can calculate the search space of the instance at hand using equation 4.3 and depending on the size of the search space we can use different annealing schedule. As the size of the search space grows larger we may have to sacrifice speed to obtain results that are acceptable.

Chapter 6

CONCLUSIONS

The simulated annealing algorithm is sensitive to a number of control parameters and stopping rules. The control parameters used in this experiment for the inner loop termination criterion, the cooling schedule, and stopping criterion for termination of the algorithm were determined experimentally on a trial and error basis. For search spaces less than 1.5×10^{20}, annealing did not perform much better than iterative improvement. When the search space was increased the annealing algorithm had a clear performance edge over iterative improvement. As the search space was increased beyond a certain point (1.5×10^{40}), the quality of solutions deteriorated very little but the time required by the annealer increased somewhat. The result is plotted on a logarithmic scale in Figure B.8 and clearly shows that the time of execution grows linearly as the search space grows. In practical problems like farm planning, it is not always crucial to find a solution very quickly. A farmer is quite likely to be patient with the plan generation if it assures him of a better plan in terms of resource usage and profit. As long as execution time is not very critical *annealing* can be used to find *good* solutions in a reasonable computation time. The results obtained by applying *annealing* to larger search spaces show that it can be used for farms that are not as constrained as the farms used in the project. This project demonstrates that *simulated annealing* does find high quality plans that incorporate factors that are attractive and desirable to the farmers while attempting to abide by environmental regulations. The method employed by this project uses field characteristics to determine suitable crop rotations and came up with plans that are both profitable and environmentally sound.

The scoring function used in this project can be adjusted to reflect the needs of the farmers. Farmers are mostly concerned with the yearly profit of the farm and the scoring

function should be adjusted to maximize profit while keeping yields uniform over the years. At present the scoring function penalizes both over-production and under-production by the same amount. A separate weighting factor can be introduced into the scoring function to penalize under-production more severely than over-production. The present planner does not use weighting factors to emphasize the importance of profit, yield, and penalties due to environmental violations. A more complete scoring function will allow farmers to specify their preferences and let them state the relative importance of the various components of the function.

This project does not attempt to schedule a farm's resources to realize plans. Labor and equipment have not been incorporated into the constraint hierarchy. If the labor and equipment requirement for each crop is known we can add a function to evaluate the plan at the end and generate total resource needed to implement the plan. It can also be incorporated into the scoring function to make the resources act as constraints while generating plans. By severely penalizing plans that violate resource restrictions we can generate plans that will take labor and equipment into consideration.

Future work can be done in finding a more robust and dynamic way of dealing with the parameters that control the cooling schedule and the inner loop termination criterion. Experimenting with different density functions for the probability of accepting an increase in the system energy, incorporating randomness in accepting a small decrease in the energy of the system in addition to probabilistically accepting an increase in the energy, limiting the probabilistic heuristics only in the vicinity of a local minimum, and experimenting with larger farms to evaluate the algorithm and compare the results with those obtained by other AI methods - are topics that can be further explored to investigate the viability of simulated annealing as a general purpose optimization technique.

Appendix A

Data For MC Farm

Table A.1: Crop economic data

Crop	Cost(Dollars/Acre)			Income(Dollars/Acre)		
	Spring	Fall	Winter	Spring	Fall	Winter
Corn	129.86	51.59	0	0	2.55†	0
Wheat	39.01	0	72.08	3.45†	0	0
Barley	45.14	0	71.38	180	0	0
Rye	0	0	47.88	0	0	0
Sorghum	0	160.46	0	0	360	0
Millet	0	101.48	0	0	168	0
Alfalfa (First year)	0	0	210.08	0	508.5	0
Alfalfa (Successive years)	90.21	0	70.16	0	508.5	0
Clover (First year)	0	61.36	18.48	0	0	168
Clover (Successive years)	0	0	36.06	0	0	252
Pasture	0	48.248	0	0	0	0 0

† Income per bushel.

Table A.2: Crop fertilizer data

Crop	Yearly fertilizer requirement (Pounds)			
	Nitrogen	Phosphorus	Potassium	Boron
Corn	100	50	50	0
Rye	0	0	0	0
Sorghum	115	50	50	0
Barley	70	40	60	0
Wheat	70	40	40	0
Millet	60	35	35	0
Alfalfa	0	65	165	3
Clover	30	30	30	0
Pasture	0	40	40	0

Table A.3: Crop pesticide data

Crop	Common Names	Active ingredient	Leaching risk	Surface runoff risk
Corn	Atrazine-AAtrex	atrazine	Large	Medium
	Dual	metalochor	Medium	Medium
	Furadan 15G	carbofuran	Large	Small
Wheat	2,4-D	2,4-D amine/ester	Medium	Small
	Banvel	dicamba	Large	Small
Barley	2,4-D	2,4-D amine/ester	Medium	Small
	Banvel	dicamba	Large	Small
Rye	Gramoxone	paraquat	Small	Large
Sorghum	Atrazine-AAtrex	atrazine	Large	Medium
	Dual	metalochor	Medium	Medium
	Furadan 15G	carbofuran	Large	Small
Millet	Gramoxone	paraquat	Small	Large
Alfalfa	Gramoxone	paraquat	Small	Large
(First year)	Furadan 15G	carbofuran	Large	Small
	2,4-D	2,4-D amine/ester	Medium	Small
Alfalfa	Lexone/Sencor	Metribuzin	Lareg	Medium
(Subsequent years)	Gramaxone	paraquat	Small	Large
	Furadan 15G	carbofuran	Large	Small
Clover	Gramoxone 2 x	paraquat	Small	Large
	2,4-D	2,4-D amine/ester	Medium	Small
Pasture	Gramoxone 2 x	paraquat	Small	Large
	2,4-D	2,4-D amine/ester	Medium	Small

Table A.4: Corn-based crop rotations

Six-year crop rotation	C value
P P P P P P P P P P P P P P P P P P	0.005
C C B B S R C C B B S R C C B B S R	0.0855
C C W W M R C C W W M R C C W W M R	0.1455
C C R C C W W M R C C R C C W W M R	0.102667
C C R C C B B S R C C R C C B B S R	0.063667
C C W W M R C C W W M R C C B B S R	0.1255
C C W W M R C C B B S R C C B B S R	0.1055
C C W W M R C C R C C B B S R C C R	0.082667
C C R C C R C C W W M A A A A A A R	0.054667
C C R C C R C C B B S R C C B B S R	0.062667
C C R C C R C C B B S R C C W W M R	0.082667
C C R C C R C C W W M R C C B B S R	0.062667
C C R C C R C C W W M R C C W W M R	0.102667
C C R C C W W M A A A A A A A A A R	0.0525
C C W W M R C C W W M A A A A A A R	0.0975
C C B B S R C C W W M A A A A A A R	0.0775
C C W W M A A A A A A A A A A A A R	0.051833
C C W W L L C C W W L L C C W W L L	0.05
C C W W L L L L L C C W W L L L L L	0.038

where C = corn grain R = rye + hairy vetch cover crop W = wheat grain
M = millet hay P = pasture B = barley silage
S = sorghum silage A = Alfalfa hay L = Clover.

Each letter corresponds to a season, three seasons to a year.

Appendix B

Results

The following *cooling schedules* were used:

Schedule 1 $T_{n+1} = T_n \times 0.95$

Schedule 2 $T_{n+1} = \begin{cases} T_n - \frac{T_n}{2} & if,\ 2 \times \gamma \times T_n > \sigma, \\ T_n - \frac{\gamma \times T_n^2}{\sigma} & else. \end{cases}$ (See reference [10])

Schedule 3 $T_{n+1} = \frac{T_n}{1+(T_n \times \ln(1+\gamma))/3 \times \sigma}$ (See reference [1])

The following *program stopping criteria* were used:

Stopping criterion 1 No improvement for 100 inner metropolis loop.

Stopping criterion 2 $\sigma \times \sigma < \theta \times T_n \times (\bar{E} - \bar{E_n})$ (See reference [10])

Stopping criterion 3 No Change in current state for three inner metropolis loop [1].

where, T_n = control parameter for n_{th} metropolis loop

T_{n+1} = control parameter for $(n+1)_{th}$ loop

$\bar{E}$ = average score during initial run of 1000 random configurations

$\bar{E_n}$ = average score of n_{th} run of the metropolis loop

γ = 0.03

σ^2 = variance in the scoring function in n_{th} loop

θ = 0.01

Table B.1: Results for various combinations of cooling schedules and stop criteria

Combination #	Inner loop iterations	Cooling Schedule	Program stopping criterion	Average Score	Standard Deviation	Time min:sec
1	fixed†	schedule 1	criterion 1	0.029169	0.010042	0:29
2	fixed	schedule 1	criterion 3	0.024953	0.005082	0:52
3	fixed	schedule 2	criterion 1	0.023191	0.002779	0:32
4	fixed	schedule 2	criterion 2	0.023844	0.003198	0:25
5	fixed	schedule 2	criterion 3	0.024349	0.003021	0:42
6	fixed	schedule 3	criterion 1	0.025121	0.004904	0:40
7	fixed	schedule 3	criterion 2	0.024296	0.002358	0:44
8	fixed	schedule 3	criterion 3	0.022151	0.001242	1:10
9	variable‡	schedule 1	criterion 1	0.0379	0.018977	0:39
10	variable	schedule 1	criterion 3	0.038422	0.023524	7:06
11	variable	schedule 2	criterion 1	0.038157	0.011718	1:21
12	variable	schedule 2	criterion 2	0.032282	0.010102	0:25
13	variable	schedule 2	criterion 3	0.028710	0.010920	1:35
14	variable	schedule 3	criterion 1	0.032896	0.017065	0:30
15	variable	schedule 3	criterion 2	0.032785	0.011922	0:11
16	variable	schedule 3	criterion 3	0.024877	0.003234	1:56

†number of iteration equals cardinality of neighborhood (equation 4.9).
‡number of iterations given by equation 4.12.

Table B.2: Plan for 5000 bushels of corn. Soil erosion, pesticide leaching, and nitrate leaching are all important.

Field #	six-year crop rotation	Soil	
		Formation	Loss
1	P P P P P P P P P P P P P P P P P P	3.13	0.82
2	P P P P P P P P P P P P P P P P P P	3.68	0.17
3	W M A A A A A A A A A A A A A R C C W	3.00	2.99
4	A A A A A R C C W W M A A A A A A A	3.50	1.36
5	C C W W M R C C R C C B B S R C C R	3.38	3.06
6	P P P P P P P P P P P P P P P P P P	3.58	0.21
7	W M A A A A A A A R C C B B S R C C W	3.50	1.47
8	A A A A A A A A A A A A R C C W W M A	3.53	2.60
9	C C R C C B B S R C C R C C B B S R	3.70	3.12
10	P P P P P P P P P P P P P P P P P P	3.01	1.21
11	P P P P P P P P P P P P P P P P P P	3.20	0.64
12	L L L C C W W L L L L L C C W W L L	3.50	3.47
13	A A R C C W W M R C C W W M A A A A	3.50	2.55
14	C C W W M A A A A A A A A A A A A R	3.22	2.81

where C = corn grain R = rye + hairy vetch cover crop W = wheat grain
M = millet hay P = pasture B = barley silage
S = sorghum silage A = Alfalfa hay L = Clover.

Each letter corresponds to a season, three seasons to a year.

Yearly yield in bushels:

Crop	Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
Corn	4975	4817	4940	5030	4736	5167

Yearly profit in dollars:

Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
7962.23	15711.54	17983.59	11409.43	18885.09	14157.81

Score of plan = 0.02

Penalty = 0.0
Time of execution : 0 min 39.1 sec

Table B.3: Plan for 7800 bushels of corn. Soil erosion, pesticide leaching, and nitrate leaching are all important.

Field #	six-year crop rotation	Soil Formation	Loss
1	P P P P P P P P P P P P P P P P P P	3.13	0.82
2	P P P P P P P P P P P P P P P P P P	3.68	0.17
3	A A R C C W W M A A A A A A A A A A	3.00	2.99
4	W M A A A A A A A A A A A A A R C C W	3.50	1.36
5	W L L L L L C C W W L L L L L C C W	3.38	1.41
6	C C R C C B B S R C C R C C B B S R	3.58	2.62
7	A A A A A A A A R C C R C C W W M A	3.50	1.00
8	A A A A A R C C R C C W W M A A A A	3.53	2.63
9	P P P P P P P P P P P P P P P P P P	3.70	0.24
10	P P P P P P P P P P P P P P P P P P	3.01	1.21
11	P P P P P P P P P P P P P P P P P P	3.20	0.64
12	W L L L L L C C W W L L L L L C C W	3.50	3.47
13	C C W W L L L L L C C W W L L L L L	3.50	0.99
14	C C W W L L C C W W L L C C W W L L	3.22	2.71

where C = corn grain R = rye + hairy vetch cover crop W = wheat grain
M = millet hay P = pasture B = barley silage
S = sorghum silage A = Alfalfa hay L = Clover.

Each letter corresponds to a season, three seasons to a year.

Yearly yield in bushels:

Crop	Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
Corn	8218	7637	7831	8102	8065	7976

Yearly profit in dollars:

Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
21673.39	22590.34	15299.10	28779.07	20221.60	19703.54

Score of plan = 0.0141

Penalty = 0.0
Time of execution : 0 min 35.5 sec

Table B.4: Plan for 9500 bushels of corn. Soil erosion, pesticide leaching, and nitrate leaching are all important.

Field #	six-year crop rotation	Soil Formation	Loss
1	P P P P P P P P P P P P P P P P P P	3.13	0.82
2	W M A A A A A A A R C C W W M R C C W	3.68	3.38
3	A A A A A A R C C W W M A A A A A A A	3.00	2.99
4	A A A A A A R C C W W M A A A A A A A	3.50	1.36
5	C C R C C R C C W W M A A A A A A R	3.38	2.02
6	C C W W M A A A A A A R C C B B S R	3.58	3.19
7	C C B B S R C C W W M A A A A A A R	3.50	1.47
8	C C R C C W W M A A A A A A A A A R	3.53	2.63
9	A A R C C W W M A A A A A A A A A A	3.70	2.54
10	P P P P P P P P P P P P P P P P P P	3.01	1.21
11	P P P P P P P P P P P P P P P P P P	3.20	0.64
12	L L L C C W W L L L L L C C W W L L	3.50	3.47
13	L L L C C W W L L L L L C C W W L L	3.50	0.99
14	A A R C C R C C W W M A A A A A A A	3.22	2.85

where C = corn grain R = rye + hairy vetch cover crop W = wheat grain
M = millet hay P = pasture B = barley silage
S = sorghum silage A = Alfalfa hay L = Clover.

Each letter corresponds to a season, three seasons to a year.

Yearly yield in bushels:

Crop	Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
Corn	9727	9614	9207	9542	9596	9542

Yearly profit in dollars:

Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
34442.68	52009.06	54173.09	32702.04	57933.84	54140.86

Score of plan = 0.0071

Penalty = 0.0
Time of execution : 0 min 43.3 sec

Table B.5: Plan for 75-95 acres of corn and 70-90 acres of wheat. Soil erosion and nitrate leaching are important, but pesticide leaching is not.

Field #	six-year crop rotation	Soil Formation	Loss
1	P P P P P P P P P P P P P P P P P P	3.13	0.82
2	C C W W L L C C W W L L C C W W L L	3.68	1.73
3	A A R C C W W M A A A A A A A A A A	3.00	2.99
4	A A A A A A A A R C C W W M A A A A	3.50	1.36
5	W M A A A A A A A A A A A A R C C W	3.38	1.92
6	W L L C C W W L L C C W W L L C C W	3.58	2.06
7	W M A A A A A A R C C B B S R C C W	3.50	1.47
8	A A A A A A A A A A A A R C C W W M A	3.53	2.60
9	W L L C C W W L L C C W W L L C C W	3.70	2.45
10	P P P P P P P P P P P P P P P P P P	3.01	1.21
11	P P P P P P P P P P P P P P P P P P	3.20	0.64
12	P P P P P P P P P P P P P P P P P P	3.50	0.46
13	C C W W M A A A A A A A A A A A A R	3.50	1.36
14	A A R C C W W M A A A A A A A A A A	3.22	2.81

where C = corn grain R = rye + hairy vetch cover crop W = wheat grain
M = millet hay P = pasture B = barley silage
S = sorghum silage A = Alfalfa hay L = Clover.

Each letter corresponds to a season, three seasons to a year.

Acreage by year:

Crop	Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
Corn	85	86	78	80	84	80
Wheat	80	85	86	78	73	84

Yearly profit in dollars:

Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
38320.48	42270.05	37448.75	48649.42	39695.59	45017.39

Score of plan = 0.0

Penalty = 0.0
Time of execution : 0 min 33.5 sec

Table B.6: Plan for corn with no target acreage. Soil erosion, nitrate leaching, and pesticide leaching are all important.

Field #	six-year crop rotation	Soil Formation	Loss
1	P P P P P P P P P P P P P P P P P P	3.13	0.82
2	C C W W M R C C R C C B B S R C C R	3.68	2.87
3	A A R C C W W M A A A A A A A A A A	3.00	2.99
4	C C W W M R C C W W M R C C R C C R	3.50	2.70
5	C C B B S R C C R C C R C C W W M R	3.38	2.32
6	L L L C C W W L L L L L C C W W L L	3.58	1.56
7	A A A A A R C C W W M R C C W W M A	3.50	1.85
8	C C R C C B B S R C C R C C B B S R	3.53	3.19
9	W L L C C W W L L C C W W L L C C W	3.70	2.45
10	P P P P P P P P P P P P P P P P P P	3.01	1.21
11	P P P P P P P P P P P P P P P P P P	3.20	0.64
12	L L L C C W W L L L L L C C W W L L	3.50	3.47
13	W M R C C W W M A A A A A A R C C W	3.50	2.55
14	W L L C C W W L L C C W W L L C C W	3.22	2.71

where C = corn grain R = rye + hairy vetch cover crop W = wheat grain
M = millet hay P = pasture B = barley silage
S = sorghum silage A = Alfalfa hay L = Clover.

Each letter corresponds to a season, three seasons to a year.

Acreage by year:

Crop	Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
Corn	127	126	127	128	126	128

Yearly profit in dollars:

Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
33372.36	31564.71	33949.53	33593.69	38445.97	37968.51

Score of plan = 0.0127

Penalty = 0.0
Time of execution : 0 min 30.3 sec

Table B.7: Plan for corn and wheat with no target acreages specified. Soil erosion, nitrate leaching, and pesticide leaching are all important.

Field #	six-year crop rotation	Soil	
		Formation	Loss
1	P P P P P P P P P P P P P P P P P P	3.13	0.82
2	C C W W L L C C W W L L C C W W L L	3.68	1.73
3	W L L C C W W L L C C W W L L C C W	3.00	2.89
4	C C R C C W W M R C C R C C W W M R	3.50	2.70
5	W L L C C W W L L C C W W L L C C W	3.38	1.85
6	W L L C C W W L L C C W W L L C C W	3.58	2.06
7	C C W W L L L L L C C W W L L L L L	3.50	0.72
8	W L L C C W W L L C C W W L L C C W	3.53	2.51
9	W L L C C W W L L C C W W L L C C W	3.70	2.45
10	P P P P P P P P P P P P P P P P P P	3.01	1.21
11	P P P P P P P P P P P P P P P P P P	3.20	0.64
12	W L L L L L C C W W L L L L L C C W	3.50	3.47
13	C C W W L L C C W W L L C C W W L L	3.50	1.31
14	C C W W L L C C W W L L C C W W L L	3.22	2.71

where C = corn grain R = rye + hairy vetch cover crop W = wheat grain
M = millet hay P = pasture B = barley silage
S = sorghum silage A = Alfalfa hay L = Clover.

Each letter corresponds to a season, three seasons to a year.

Acreage by year:

Crop	Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
Corn	127	120	127	127	121	127
Wheat	127	106	121	127	106	121

Yearly profit in dollars:

Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
38085.35	37504.04	36261.52	38128.07	37461.30	36304.25

Score of plan = 0.09455

Penalty = 0.0
Time of execution : 0 min 32.3 sec

Table B.8: Plan for maximizing profit with no goal specified. Soil erosion, nitrate leaching, and pesticide leaching are all important.

Field #	six-year crop rotation	Soil Formation	Soil Loss
1	P P P P P P P P P P P P P P P P P P	3.13	0.82
2	W M A A A A A A A A A A A A A R C C W	3.68	1.80
3	W L L C C W W L L C C W W L L C C W	3.00	2.89
4	L L L C C W W L L L L L C C W W L L	3.50	1.00
5	C C R C C R C C W W M R C C B B S R	3.38	2.32
6	A A R C C B B S R C C W W M A A A A	3.58	3.19
7	W M A A A A A A A R C C B B S R C C W	3.50	1.47
8	A A R C C R C C W W M A A A A A A A	3.53	2.63
9	C C W W L L C C W W L L C C W W L L	3.70	2.45
10	P P P P P P P P P P P P P P P P P P	3.01	1.21
11	P P P P P P P P P P P P P P P P P P	3.20	0.64
12	W L L L L L C C W W L L L L L C C W	3.50	3.47
13	C C W W M R C C W W M A A A A A A R	3.50	2.55
14	W M A A A A A A A A A A A A A R C C W	3.22	2.81

where C = corn grain R = rye + hairy vetch cover crop W = wheat grain
M = millet hay P = pasture B = barley silage
S = sorghum silage A = Alfalfa hay L = Clover.

Each letter corresponds to a season, three seasons to a year.

Yearly yield in bushels:

Crop	Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
Corn	4250	13340	8047	8418	5880	16844
Wheat	10614	1138	2972	4934	4864	2331

Yearly profit in dollars:

Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
39023.39	52294.46	55361.02	49421.63	52209.41	38441.85

Score of plan = 0.0

Penalty = 0.0
Time of execution : 0 min 11.7 sec

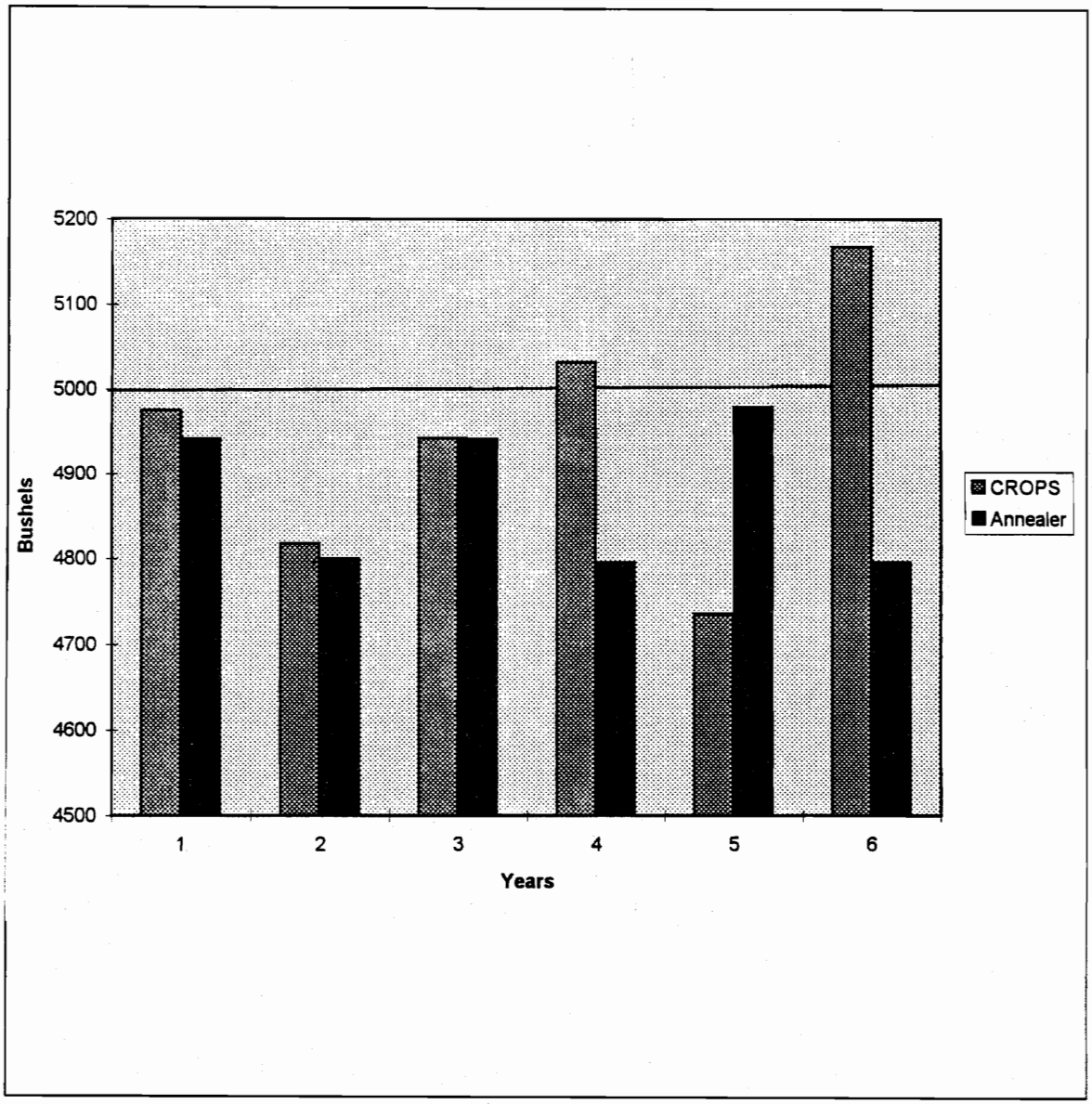


Figure B.1: Comparison between results from CROPS and annealing. Target is 5000 bushels of corn on MC farm

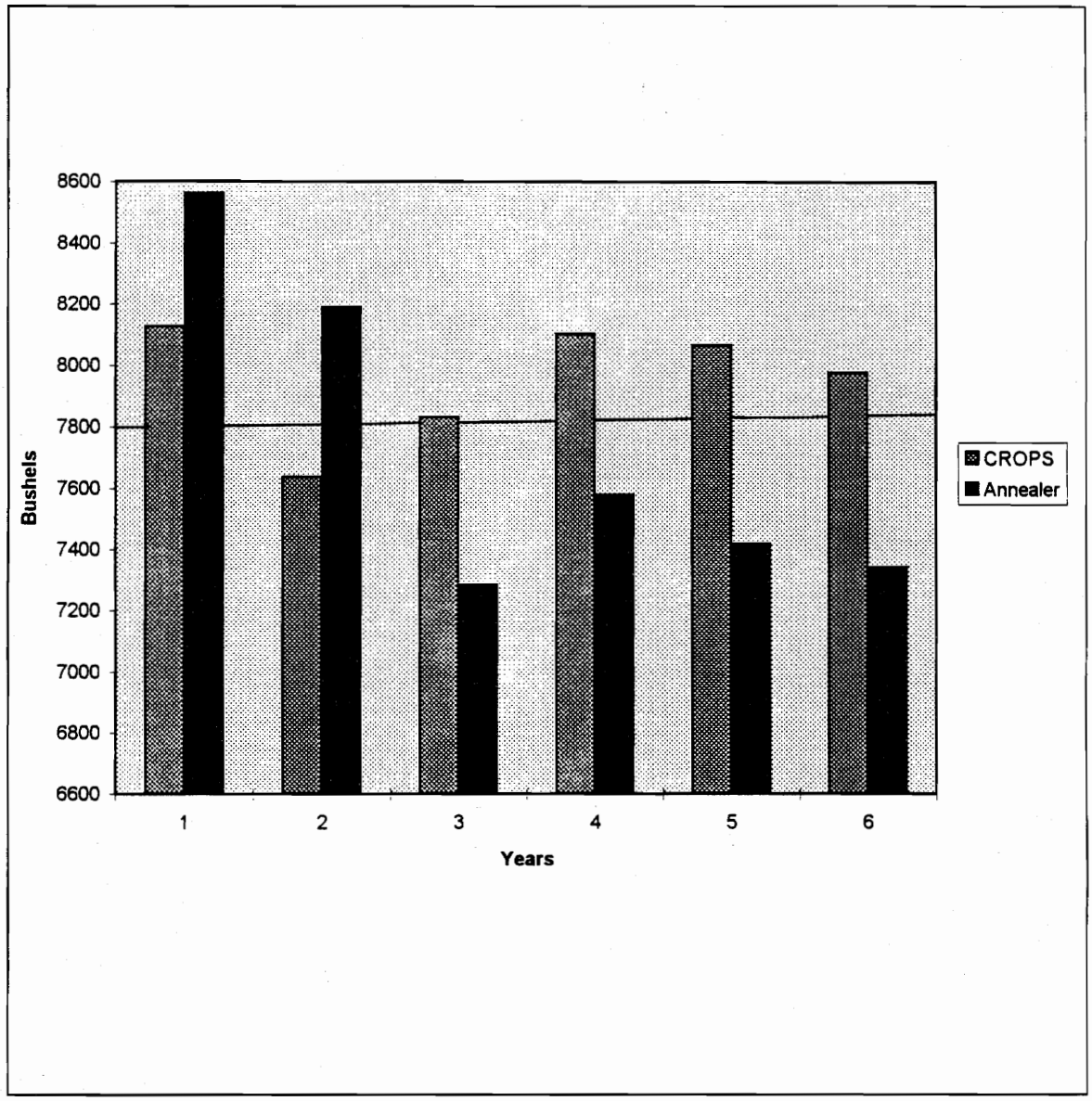


Figure B.2: Comparison between results from CROPS and annealing. Target is 7800 bushels of corn on MC farm. Pesticide leaching constraint relaxed

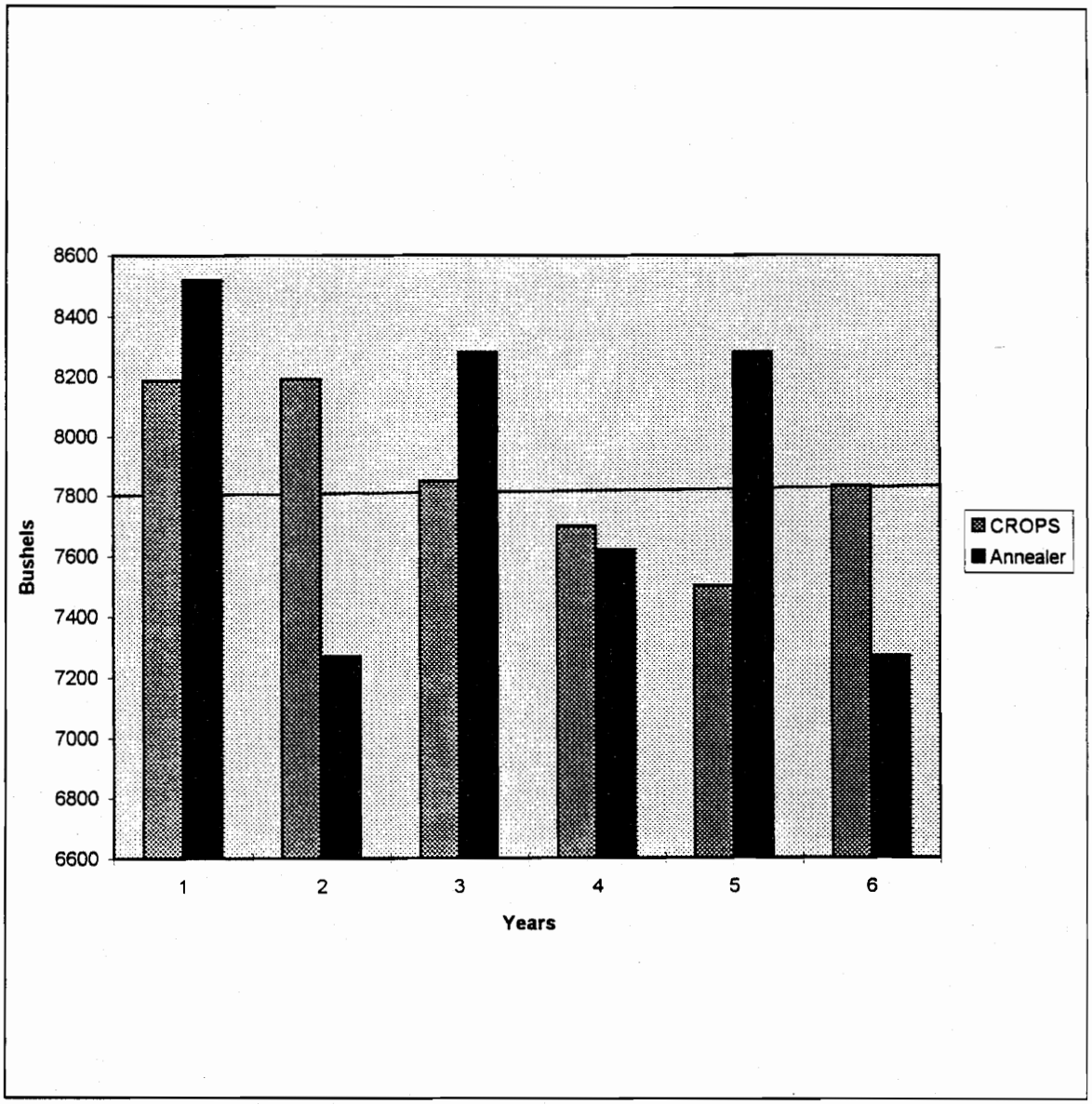


Figure B.3: Comparison between results from CROPS and annealing. Target is 7800 bushels of corn on MC farm. Surface runoff, pesticide and nitrate leaching constraint relaxed

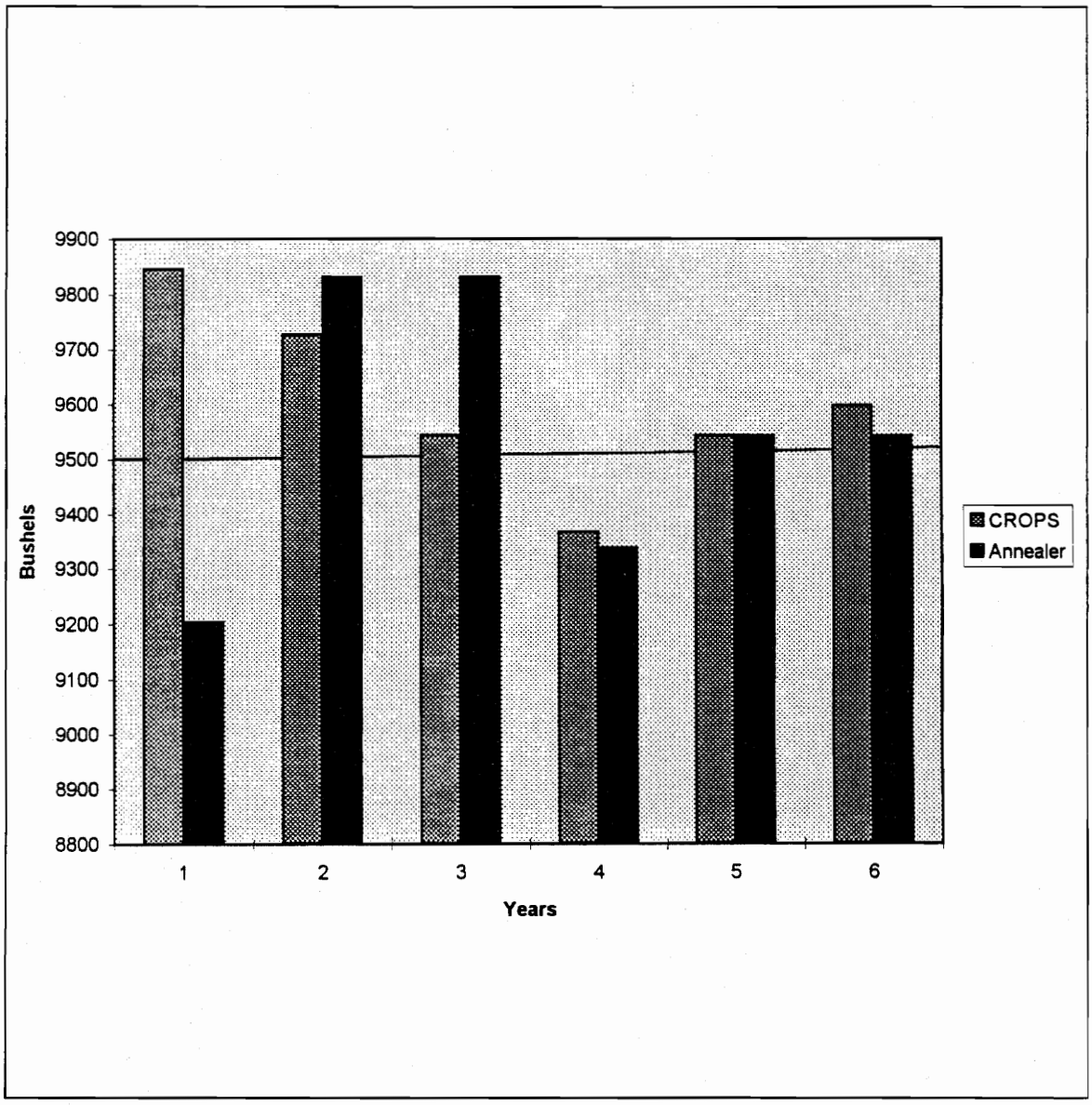


Figure B.4: Comparison between results from CROPS and annealing. Target is 9500 bushels of corn on MC farm. Soil erosion and pesticide leaching constraint relaxed

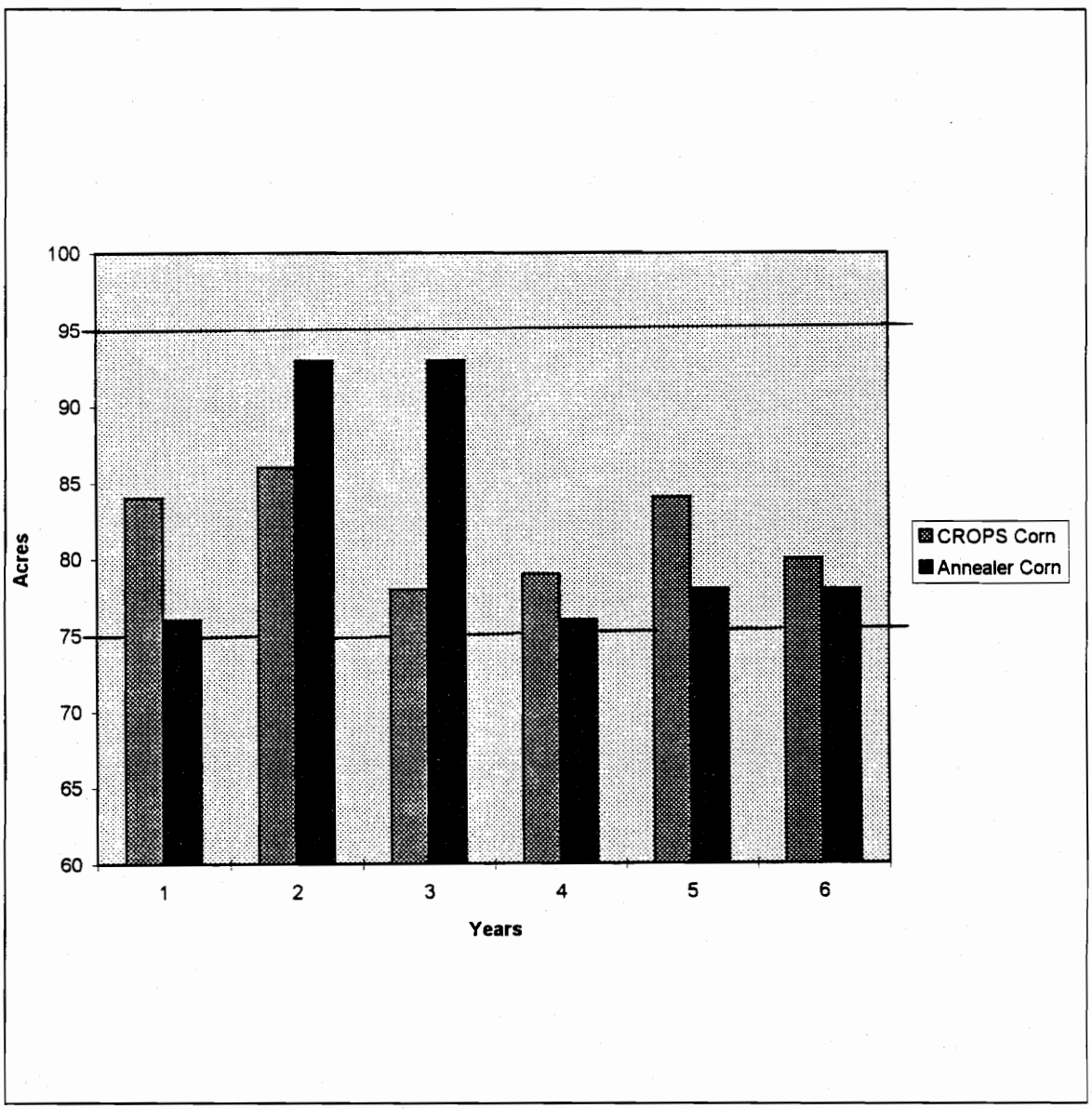


Figure B.5: Comparison between results from CROPS and annealing. Target is 75-95 acres of corn and 70-90 acres of wheat on MC farm. Pesticide leaching constraint relaxed

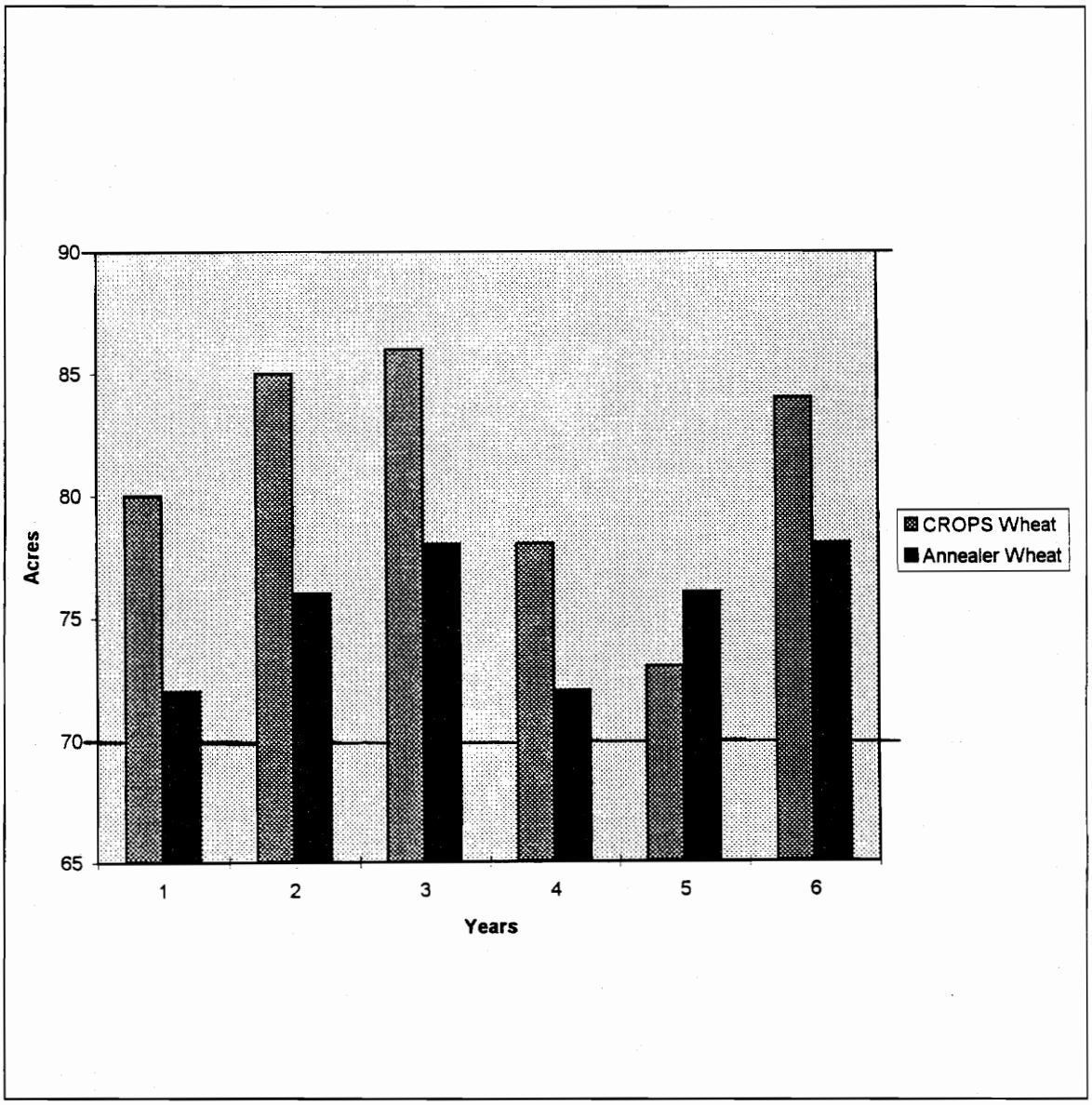


Figure B.6: Comparison between results from CROPS and annealing. Target is 75-95 acres of corn and 70-90 acres of wheat on MC farm. (continued)

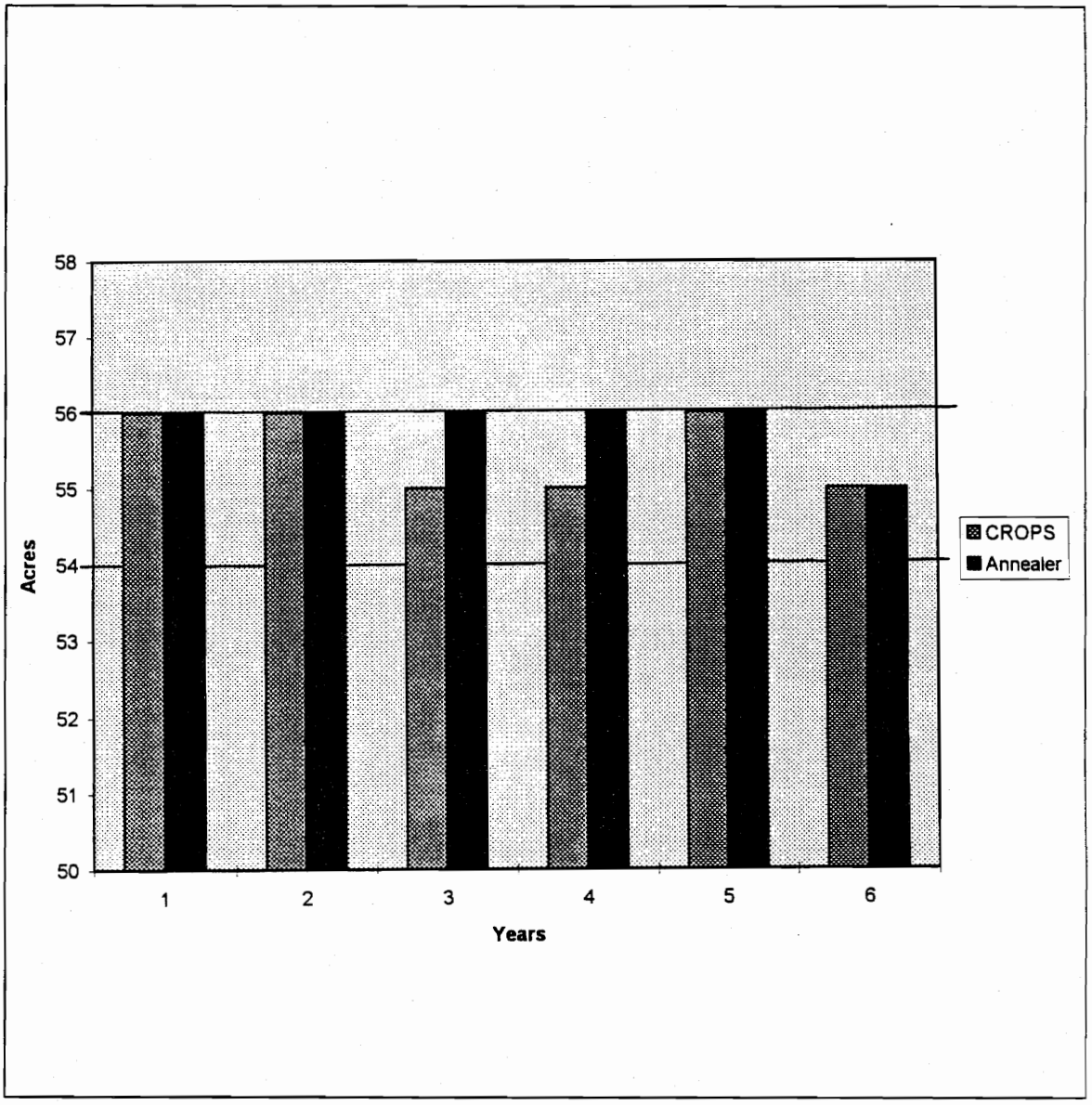


Figure B.7: Comparison between results from CROPS and annealing. Target is 54-56 acres of corn on WC farm

Table B.9: Impact of size of search space on annealing

Search space	Iterative improvement		Annealing		Time
	Score	standard deviation	Score	Standard deviation	min:sec
1.5×10^{29}	0.002191	0.001590	0.000031	0.000098	1:31
1.3×10^{31}	0.002191	0.002586	0.000772	0.001021	1:42
1.1×10^{33}	0.005895	0.002649	0.000432	0.000701	2:06
1.1×10^{44}	0.016018	0.005558	0.000586	0.001150	4:00
1.3×10^{49}	0.031019	0.012491	0.000710	0.000797	5:25

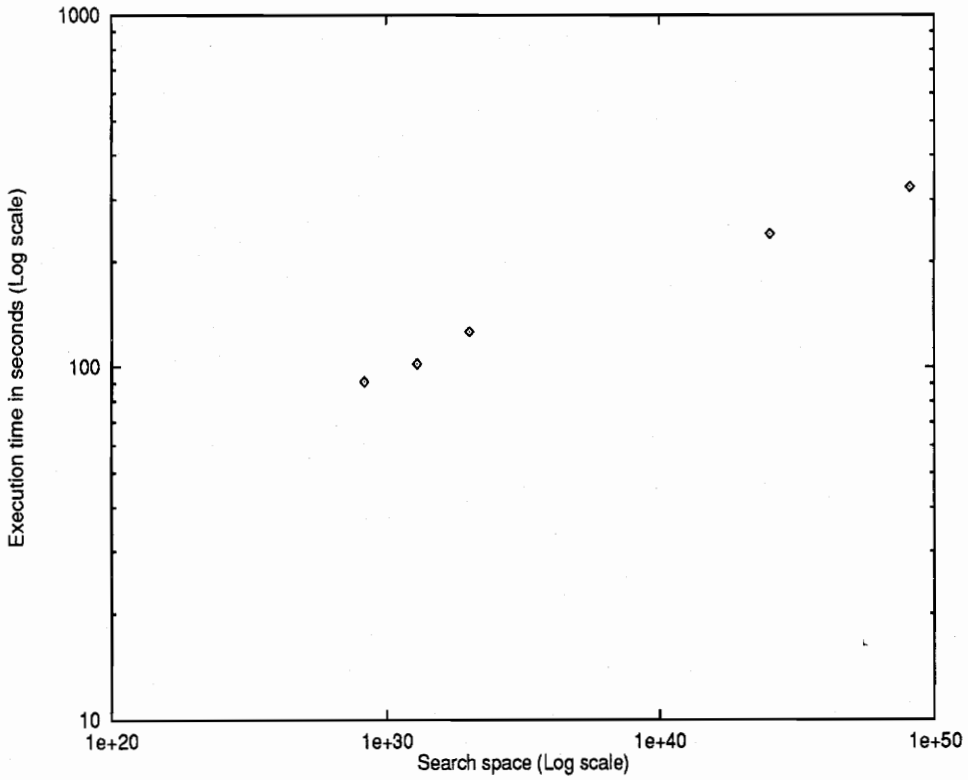


Figure B.8: Plot of execution time versus search space.

Table B.10: Results of different schedule for search space of 1.3×10^{49}.

Combination #	Inner loop iterations	Cooling Schedule	Program stopping criterion	Score	Time min:sec
1	fixed†	schedule 1	criterion 1	0.014321	2:05
2	fixed	schedule 1	criterion 3	0.019784	1:21
3	fixed	schedule 2	criterion 1	0.078500	0:46
4	fixed	schedule 2	criterion 2	0.018790	3:48
5	fixed	schedule 2	criterion 3	0.007086	4:21
6	fixed	schedule 3	criterion 1	0.105617	0:49
7	fixed	schedule 3	criterion 2	0.017074	6:26
8	fixed	schedule 3	criterion 3	0.008099	7:01
9	variable‡	schedule 1	criterion 1	0.011420	1:58
10	variable	schedule 1	criterion 3	0.013506	1:08
11	variable	schedule 2	criterion 1	0.059469	0:47
12	variable	schedule 2	criterion 2	0.016747	3:42
13	variable	schedule 2	criterion 3	0.009086	4:36
14	variable	schedule 3	criterion 1	0.049494	0:43
15	variable	schedule 3	criterion 2	0.012772	6:06
16	variable	schedule 3	criterion 3	0.010235	7:12

†number of iteration equals cardinality of neighborhood (equation 4.9).
‡number of iterations given by equation 4.12.

REFERENCES

[1] E. Aarts and J. Korst. *Simulated Annealing and Boltzman Machines.* John Wiley & Sons, New York, 1989.

[2] J. Allen and J. Koomen. Planning using a temporal world model. *Proceedings of IJCAI*, 8:741–747, 1983.

[3] P. Carnevali, L. Coletti, and S. Patarnello. Image Processing by Simulated Annealing. *IBM J. Res. Develop.*, 29(6):569–579, 1985.

[4] R. E. Fikes and N. J. Nilsson. Strips: a new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208, 1971.

[5] M. S. Fox. *Constraint-directed search: a case study of job-shop scheduling.* PhD thesis, Carnegie-Mellon University, Pittsburg, Pennsylvania 15213, 1983.

[6] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by Simulated Annealing. *Science*, 220:1087–1092, 1983.

[7] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of State Calculations by Fast Computing Machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

[8] S. Nahar, S. Sahni, and E. Shragowitz. Simulated Annealing and Combinatorial Optimization. *23rd Design Automation Conference*, pages 293–299, 1986.

[9] R. H. Nielson. *Neurocomputing*, chapter 6, pages 192–195. Addison-Wesley Publishing, Massachusetts, 1990.

[10] R. H. J. M. Otten and L. P. P. P. Ginneken. *The Annealing Algorithm.* Kluwer Academic Publishers, Massachusetts, 1989.

[11] C. K. Riesbeck and R. C. Schank. *Inside case-based reasoning.* Lawrence Erlbaum Associates, New Jersey, 1989.

[12] E. D. Sacerdoti. *A structure for plans and behavior.* Elsevier Press, New York, 1977.

[13] Mark Stefik. Planning with constraints (MOLGEN: Part 1). *Artificial Intelligence*, 16:111–140, 1981.

[14] N. D. Stone. Object-Oriented Constraint Satisfaction Planning for Whole Farm Management. *AI Applications*, 9(1), 1995.

[15] N. D. Stone, R. D. Buick, J. W. Roach, R. K. Scheckler, and R. Rupani. Crops: a Whole-Farm Crop Rotation Planning System to Implement Sustainable Agriculture. *AI Applications*, 6(3):29–50, 1992.

[16] N. D. Stone, R. D. Buick, J. W. Roach, R. K. Scheckler, and R. Rupani. The Planning Problem in Agriculture: Farm-level Crop Rotation Planning as an Example. *AI Applications*, 6(1):59–75, 1992.