Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2013, Article ID 158538, 14 pages
http://dx.doi.org/10.1155/2013/158538

Research Article

Three New Stochastic Local Search Metaheuristics for the Annual Crop Planning Problem Based on a New Irrigation Scheme

Sivashan Chetty and Aderemi Oluyinka Adewumi

School of Mathematics, Statistics and Computer Science, University of KwaZulu-Natal, University Road, Westville, Private Bag X 54001, Durban 4000, South Africa

Correspondence should be addressed to Aderemi Oluyinka Adewumi; laremtj@gmail.com

Received 6 February 2013; Accepted 19 April 2013

Academic Editor: Sabri Arik

Copyright © 2013 S. Chetty and A. O. Adewumi. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Annual Crop Planning (ACP) is an NP-hard-type optimization problem in agricultural planning. It involves finding optimal solutions concerning the seasonal allocations of a limited amount of agricultural land amongst the various competing crops that are required to be grown on it. This study investigates the effectiveness of employing three new local search (LS) metaheuristic techniques in determining solutions to an ACP problem at a new Irrigation Scheme. These three new LS metaheuristic techniques are the Best Performance Algorithm (BPA), Iterative Best Performance Algorithm (IBPA), and the Largest Absolute Difference Algorithm (LADA). The solutions determined by these LS metaheuristic techniques are compared against the solutions of two other well-known LS metaheuristic techniques in the literature. These techniques are Tabu Search (TS) and Simulated Annealing (SA). The comparison with TS and SA was to determine the relative merits of the solutions found by BPA, IBPA, and LADA. The results show that TS performed as the overall best. However, LADA determined the best solution that was the most economically feasible.

1. Introduction

Increases in population growth have increased the need for more food to be produced throughout the world. At present, the shortages in food supply have resulted in the problem of starvation, being a hard-felt reality in the lives of millions of people. This is particularly true in the 4th world countries. To combat this problem for the future, the productivity of food needs to increase.

The sector, that is, the primary supplier of food in the world, is the agricultural sector [1]. To try and meet the growing demands for food, the agricultural sector needs to increase its output. Optimizing the production of food at the current agricultural practices is important but is not enough to meet the *future* demands. To produce more food in the future, more land must be made available for agricultural production.

The allocations of land, for agricultural production, will depend on the decisions made by the local authorities. For land to be allocated, it needs to be accessed to determine its feasibility for agricultural production and if the crops grown on it will be sustainable in the future. This is important for economic development.

To determine the agricultural potential of a given area of land, several factors will need to be considered. The main factors considered are the soil characteristics and climate conditions [2]. For crop production, these factors will determine the types of crops that will most suitably adapt to the given environmental conditions. Other important factors considered are the natural land resources and agricultural trends, amongst others.

Natural land resources such as lakes and rivers are very valuable commodities. They can be used to source irrigated water. Irrigated water and rainfall are important in determining the full agricultural potential of a given area of agricultural land. The agricultural trends will determine the types of crops that will be the most suitable for economic benefits.

When an area of land gets allocated for the development of a new Irrigation Scheme, and it has been finalized which crops will be cultivated, then solutions need to be found concerning the hectare allocations amongst the competing crops. In determining the hectare allocations, it needs to be considered that different types of crops grow in different seasons, grow for different lengths of time, and have different plant requirements. These factors must be considered in order to determine feasible solutions.

The problem of trying to optimize the seasonal hectare allocations, of a given area of agricultural land amongst the various competing crops that are required to be grown within a year, is an NP-hard-type optimization problem in agricultural planning called Annual Crop Planning (ACP). ACP aims at determining solutions that seek to maximize the total gross profits that can be earned from a given area of agricultural land, in making the most efficient uses of the limited resources available for agricultural production. Limited resources include land, irrigated water supply, and the various costs associated with agricultural production. The solutions found must satisfy the multiple land and irrigation water allocation constraints that are associated with ACP, in order to be feasible.

This research introduces a new Annual Crop Planning mathematical model. The model has been formulated by the authors of this paper. It is intended to be used to determine solutions to the ACP problem at a *new* Irrigation Scheme.

Previous studies in crop and irrigation planning have used both single and multiobjective mathematical models. Many optimization techniques have been used to provide solutions to these models. These include Linear Programming (LP), Simulated Annealing (SA), Particle Swarm Optimization (PSO), and Evolutionary Algorithms (EA's), amongst others.

Pant et al. [3] employed the Differential Evolution (DE) algorithm to provide solutions to a crop planning problem under adequate, normal and limited irrigated water supply. The objective was to maximize the net benefits gained under these conditions. It was found that DE performed better than the programming tool LINGO. In [4], Pant et al. investigated the performances of four EA's in providing solutions to a crop planning problem. These algorithms included the Genetic Algorithm (GA), PSO, DE, and Evolutionary Programming (EP). Solutions were also determined using LINGO. The solutions found showed that, from all heuristic algorithms, GA performed poorly and that DE, PSO, and EP were all comparable. Georgiou and Papamichail [5] used SA in combination with the Stochastic Gradient Descent Algorithm to determine solutions concerning the optimized water release policies of a reservoir. The released water needed to be allocated efficiently amongst the various crops being grown. To maximize profits, the "optimal" cropping pattern needed to be determined. Wardlaw and Bhaktikul [6] used GA to solve a problem of irrigated water scheduling, using a 0-1 approach. The research found that GA performed well in being able to distribute irrigated water to several farm plots in satisfying the soil moisture content levels, under water stress conditions. The water allocations were done on a rotational basis. Sarker and Ray [7] proposed an improved EA known as the Multiobjective Constrained Algorithm (MCA). MCA was used to provide solutions to a multiobjective crop planning problem. The research found that MCA performed relatively better compared to the two other optimization techniques used. These techniques included the ε-constrained method and the Nondominated Sorting Genetic Algorithm (NSGAII). Raju and Kumar [8] compared the performances of GA and LP in providing solutions to a crop planning problem. The objective was to maximize the net benefits gained. The performances of GA and LP were relatively close. It was concluded that GA is an effective heuristic algorithm that can be used for irrigation planning. Reddy and Kumar [9] studied the effectiveness of using Elitism-Mutation Particle Swarm Optimization (EMPSO) in determining the short-term release policies of irrigated water from a reservoir, under water scarce conditions. The study concluded that the heuristic algorithm is effective in providing short-term solutions for multicrop irrigation.

This research introduces three new local search (LS) metaheuristic algorithms in the literature. These algorithms are called the Best Performance Algorithm (BPA), the Iterative Best Performance Algorithm (IBPA), and the Largest Absolute Difference Algorithm (LADA). These algorithms are used to provide solutions to an ACP problem at a new Irrigation Scheme. To determine the relative merits of the solutions provided by these algorithms, their solutions have been compared against the solutions of two traditional LS metaheuristic algorithms in the literature. These popular metaheuristic algorithms are Tabu Search (TS) and Simulated Annealing (SA). The solutions determined and comparisons made will indicate the possible strengths and/or weaknesses of the LS algorithms, in determining solutions to this ACP problem. The solutions found will be valuable in making suggestions concerning the seasonal hectare allocations for the crops that are required to be grown.

The rest of this paper is structured as follows. Section 2 describes and presents the formulation of the ACP mathematical model. Section 3 describes the case study of the Taung Irrigation Scheme. Section 4 describes the SI metaheuristic algorithms used. Section 5 presents and discusses the experimental results obtained. Finally, Section 6 draws conclusions and outlines possible future work.

2. The Annual Crop Planning Mathematical Model

This Annual Crop Planning (ACP) mathematical model has been formulated by the authors of this paper. It is intended to be used to determine solutions to the Annual Crop Planning problem at a *new* Irrigation Scheme. The feasible solutions found must allocate the limited amount of agricultural land amongst the various competing crops that are required to be grown within the year. These solutions must satisfy all the constraints associated with the objective function. The objective in determining an optimal solution is to maximize the total gross profits that can be earned, in making the most efficient usage of the limited resources available. The limited resources include land, irrigated water supply, and

the variable costs associated with agricultural production. To determine feasible solutions, it must be taken into account that the different types of crops grow in different seasons, grow for different lengths of time, and have different plant requirements. To make efficient use of irrigated water supply, precipitation must be taken into account.

The crops cultivated for agricultural production include those that are grown all year around. These are the tree bearing crops and perennials. Other crop types include the seasonal crops such as the summer, autumn, and winter crops, amongst others. Single-crop plots of land are allocated to those crops that are grown all year around. Double-crop plots of land are allocated to two different types of crops that are grown in sequence within the year. Triple-crop plots of land are allocated to three different types of crops that are grown in sequence within a year and so on.

Soil characteristics are also a factor in crop planning. Certain crops may adapt well only to certain types of soils. Therefore, the utilization of land is important for optimal yields. Irrigation application is also important. Too much or too little applications of water will lead to suboptimal plant growth. This will affect the yield of the crop. Soils are also sensitive to leaching due to excessive water applications [1]. Therefore, the seasonal irrigated water allocations amongst the various crops need to be well planned.

The ACP mathematical model for determining solutions at a new Irrigation Scheme is formulated as follows.

2.1. Indices

(i) k: plot types (1 = single-crop plots, 2 = double-crop plots, 3 = triple-crop plots, etc.).

(ii) i: indicative of the groups of crops that are grown in sequence throughout the year, on plot type k ($i = 1$ represents the 1st group of sequential crops, $i = 2$ represents the 2nd group of sequential crops, $i = 3$ represents the 3rd group of sequential crops, etc.).

(iii) j: indicative of the individual crops grown at stage i, on plot k.

2.2. Input Parameters

(i) l: number of different plot types.

(ii) N_k: number of sequential groups of crops grown within a year, on plot k.

(iii) M_{ki}: number of different types of crops grown at stage i, on plot k.

(iv) F_{kij}: average fraction per hectare of crop j, at stage i, on plot k, which needs to be irrigated (1 = 100% coverage, 0 = 0% coverage).

(v) R_{kij}: averaged rainfall estimates that fall during the growing months for crop j, at stage i, on plot k.

(vi) CWR_{kij}: crop water requirements of crop j, at stage i, on plot k.

(vii) T: total hectares of land allocated for the irrigation scheme.

(viii) A: volume of irrigated water that can be supplied per hectare (ha^{-1}).

(ix) P: price of irrigated water m^{-3}.

(x) O_{kij}: other operational costs ha^{-1} of crop j, at stage i, on plot k. These costs exclude the cost of irrigation.

(xi) YR_{kij}: the amount of yield that can be obtained in tons per hectare ($\mathrm{t\ ha}^{-1}$) from crop j, at stage i, on plot k.

(xii) MP_{kij}: producer prices per ton (t^{-1}) for crop j, at stage i, on plot k.

(xiii) Lb_{kij}: lower bound for crop j, at stage i, on plot k.

(xiv) Ub_{kij}: upper bound for crop j, at stage i, on plot k.

(xv) Lb_P_k: lower bound for plot type k.

(xvi) Ub_P_k: upper bound for plot type k.

2.3. Calculated Parameters

(i) IR_{kij}: volume of irrigated water estimates that should be applied to crop j, at stage i, on plot k ($IR_{kij}m^3 = (\mathrm{CWR}_{kij}m - R_{kij}m) * 10000m^2 * F_{kij}$).

(ii) TA: total volume of irrigated water that can be supplied to the given area of land, within a year ($TA = T * A$).

(iii) C_IR_{kij}: the cost of irrigated water ha^{-1} of crop j, at stage i, on plot k ($C\_IR_{kij} = IR_{kij} * P$).

(iv) C_{kij}: variable costs ha^{-1} of crop j, at stage i, on plot k ($C_{kij} = O_{kij} + C\_IR_{kiij}$).

(v) B_{kij}: gross margin that can be earned ha^{-1} for crop j, at stage i, on plot k ($B_{kij} = MP_{kij} * YR_{kij} - C_{kij}$).

2.4. Variables

(i) L_k: total area of land allocated for agricultural production for plot type k.

(ii) X_{kij}: area of land, in hectares, that can be feasibly allocated to crop j, at stage i, on plot k.

2.5. Objective Function.

Maximize

$$f = \sum_{k=1}^{l} \sum_{i=1}^{N_k} \sum_{j=1}^{M_{ki}} X_{kij} B_{kij}. \tag{1}$$

In (1), k represents the plot types. $k = 1$ indicates the single-crop plots, $k = 2$ indicates the double-crop plots, and so on. For each plot type k, i is indicative of the number of groups of crops that are grown in sequence throughout the year. For $k = 1$, N_k (or N_1) will be equivalent to 1. This will represent the group of crops that are grown all year around. For $k = 2$, $N_k = 2$. This will represent two groups of crops that are grown in sequence throughout the year. These are the summer and winter crop groups. The explanation is similar for $k = 3$, and so on. For each sequential crop group i, grown on plot k, j will represent the individual crops grown. For

$k = 1$ and $i = 1$, j will be indicative of all the tree bearing and perennial crops grown. For $k = 2$ and $i = 1$, j will be indicative of all the summer crops grown. For $k = 2$ and $i = 2$, j will be indicative of all the winter grown, and so on.

Equation (1) is subjected to the land and irrigated water allocation constraints given in Sections 2.6 and 2.7. The gross benefits B_{kij} that can be earned per crop must also satisfy the nonnegative constraint given in Section 2.8.

2.6. Land Constraints. Feasible solutions must satisfy the lower and upper bound constraints of the plot types k. This constraint is given in

$$Lb\_P_k \leq L_k \leq Ub_{Pk}, \quad \forall k, i, j. \tag{2}$$

The sum of the hectares allocated for each plot type k must be less than or equal to T. This constraint is given by

$$\sum_{k=1}^{l} L_k \leq T. \tag{3}$$

The sum of the hectares allocated for each crop j, at stage i, on plot k, must be less than or equal to the total area of land allocated for agricultural production on plot type k. This constraint is given by

$$\sum_{j}^{M_{ki}} X_{kij} \leq L_k, \quad \forall k, i. \tag{4}$$

The lower and upper bound constraints for each crop must be satisfied. This constraint is given by

$$Lb_{kij} \leq X_{kij} \leq Ub_{kij}, \quad \forall k, i, j. \tag{5}$$

2.7. Irrigation Constraints. The total volume of irrigated water that is required for the production of all crops, within the year, must be less than or equal to the total volume of irrigated water that can be supplied to the given area of land. This constraint considers that some crops may require more irrigated water than what is supplied ha^{-1}. It is therefore the responsibility of the farmer to distribute his supply of irrigated water efficiently. This constraint is given by (6) below:

$$\sum_{k}\sum_{i}\sum_{j} IR_{kij} \leq TA. \tag{6}$$

2.8. Nonnegative Constraints. The gross profits that can be earned per crop must be greater than zero. This constraint is given by

$$B_{kij} > 0, \quad \forall k, i, j. \tag{7}$$

3. Case Study

The Taung Irrigation Scheme (TIS) is situated in the Taung District, in the North West Province of South Africa. It is a neighbouring Irrigation Scheme to the Vaalharts Irrigation Scheme (VIS). The VIS is one of the largest Irrigation Schemes in the world. TIS currently consists of a total of 3,764 ha of irrigated land [2].

The irrigated water currently supplied to the TIS is drawn from the Vaal River and is supplied via the Vaalharts Canal System. The Vaalharts Canal System also supplies irrigated water to the VIS. The irrigated water supplied to the TIS is supplied at a basic quota of 8,417 $\text{m}^3\,\text{ha}^{-1}\,\text{annum}^{-1}$ to the farmers [2].

Located in the area of the TIS is the Taung Dam. At full capacity the dam consists of a total volume of 62.97 million m^3 of water. The dam was originally constructed to supply irrigated water to the TIS, but no infrastructure had been built to do so.

A recent survey [2] had been done to determine if extending the existing TIS would be feasible in developing new irrigated areas. If it is found that the adjacent portions of land are feasible, then the irrigated water supplied to the TIS will be drawn from the Taung Dam.

The survey found that 3,315 ha are acceptable for agricultural production. It is also believed that agricultural production on this portion of land will match the high agricultural output of the neighbouring VIS.

The current expansion of the TIS will cater for 175 people that had been previously excluded from the land. A total of 1,750 ha (10 ha per person) will now be allocated to them for restitution. According to the choices of the local department of agriculture and the local farmers, the most suitable crops to be cultivated on this portion of land are those listed in Table 1 [2].

The crops consist of Lucerne, which is grown all year around (y). The rest of the crops are the summer (s) and winter (w) crops. Lucerne will be grown on single-crop plots of land. The summer and the winter crops will be grown on double-crop plots of land.

To determine solutions concerning the seasonal hectare allocations, amongst the various competing crops that are required to be grown, the Crop Water Requirements (CWR) and the Average Rainfall (AR) statistics need to be determined. The AR values are the average amounts of rain that is expected to fall during the growing months of each crop. The CWR is provided by [2]. The average rainfall statistics are obtained from [10].

The producer prices per ton (ZAR t^{-1})of yield are determined from [11, 12] (ZAR stands for Zuid-Afrikaanse Rand which is the Dutch translation of "South African Rand." The Rand is the currency in South Africa). The yield expected (t ha^{-1}) per crop is determined from [13]. The water quota of 8,417 $\text{m}^3\,\text{ha}^{-1}\text{annum}^{-1}$ will remain the same. The cost of irrigated water is 8.77 cents/m^3 [14].

4. Methodology

Heuristic algorithms are decision algorithms that use trial and error techniques in determining the next solution from the solution space. Without using "intelligent" techniques,

Table 1: Crop and average rainfall statistics.

Crops	CWR (mm)	AR (mm)	ZAR t^{-1}	$t\,ha^{-1}$
Lucerne (y)	1,445	444.7	1,185.52	16.0
Tomato (s)	1,132	350.8	4,332.00	50.0
Pumpkin (s)	794	279.0	1,577.09	20.0
Maize (s)	979	279.0	1,321.25	9.0
Ground Nut (s)	912	339.5	5,076.00	3.0
Sunflower (s)	648	314.9	3,739.00	3.0
Barley (w)	530	58.3	2,083.27	6.0
Onion (w)	429	177.0	2,397.90	30.0
Potato (w)	365	152.8	2,463.00	28.0
Cabbage (w)	350	152.8	1,437.58	50.0

the heuristic algorithms can suffer from premature convergence. Premature convergence occurs when the algorithm gets stuck within a local neighbourhood structure of the solution space, where the local optima are not close enough to the global optima. To reduce the possibility of premature convergence, many heuristic algorithms have been developed using more intelligent techniques. Some of these techniques include using memory abilities, having the ability to randomly "jump" to other neighbourhood structures of the solution space and having the ability to learn from other "agents," amongst others. Heuristic algorithms that use intelligent techniques, in determining stochastic solutions, are called metaheuristic algorithms.

There are two types of metaheuristic algorithms. These are the global search (GS) and local search (LS) metaheuristic algorithms. GS algorithms aim at exploring the domains of the solution space in the hope of trying to find the global optimum solution. LS metaheuristic algorithms exploit the local neighbourhood structures of the solution space in the hope of trying to find the local optimum solutions. The best local optimum solution found by both the GS and LS techniques represents the best solution found by the algorithms. Both GS and LS metaheuristic algorithms have provided effective solutions to many real-world optimization problems that are NP-hard in nature.

This research introduces three new LS metaheuristic algorithms in the literature. These algorithms have been developed by the authors of this paper. The algorithms are called the Best Performance Algorithm (BPA), the Iterative Best Performance Algorithm (IBPA), and the Largest Absolute Difference Algorithm (LADA). To investigate the effectiveness of these algorithms, they have been used to try and determine solutions to the ACP problem at the TIS. To determine the relative merits of their solutions, their solutions will be compared against the solutions of two traditional LS metaheuristic algorithms in the literature. These algorithms are Tabu Search (TS) and Simulated Annealing (SA). Descriptions of these algorithms are given in the following subsections.

4.1. Best Performance Algorithm. The Best Performance Algorithm is modelled on the competitive nature of professional athletes. Professional athletes desire to push the boundaries of their best performances within competitive environments. This occurs for several reasons. The reasons could be personal and/or financial, amongst others. However, to give off their best performances, the athletes need to strategize and practice. Strategizing and practice will help them improve their talents, in developing refined skills. These refined skills will enable the athletes to perform at their best within competitive environments, irrespective of their sporting disciplines.

An effective strategy used in improving performances is to make use of technology. Technology can be used to identify the weaknesses and strengths of the athletes in them delivering a performance. By identifying and then strengthening their weaknesses or even developing new techniques, in delivering a performance, an athlete could possibly register improved performances in being competitive. One way to identify an athlete's weaknesses and strengths is to maintain an archive or a collection of the athlete's best registered performances. This collection will provide a reference to which the athlete can go back to in order to review the way a previous best performance was delivered. Once weaknesses are identified, appropriate changes can be made to the techniques used in delivering a performance. This will help the athlete develop refined skills which will improve the chances of the athlete delivering improved performance. Best performances can include those performed within competitive environments and even those performed during training sessions. Modelled on the idea of an athlete maintaining a collection of a limited number of his/her best performances is how BPA is implemented.

BPA is implemented by maintaining a *sorted* list of the best performances of an individual athlete. This list is called the Performance List (PL). The PL only maintains a limited number of the best recorded performances of an athlete, as the athlete will only be interested in working with a limited number of his/her best recorded performances. Performances are arranged according to the "quality" of the performances delivered. The quality of a performance is a measure of the result obtained in executing that performance. The better the quality of a performance is the higher up on the PL will be its ranking.

In trying to develop refined skills or possibly determining a new technique, which may lead to improved performances being delivered, the athlete will review a performance from the PL and will seek to make appropriate changes. By making slight changes (performing local search) in the way that the reviewed performance was delivered, an improved technique may be determined which may lead to a better quality performance. If an improved technique is found, then the PL will be updated with this performance, provided that it at least improves on the *worst* registered performance on the PL. When improved performances get inserted into the PL, the worst performance gets removed. The sorted order of the PL must always be maintained. Any improved technique found which produces a performance that results in the quality of that performance being identical to the quality of another performance, that is, already registered on the PL, will *not* be considered.

Upon making slight changes to the technique used in delivering a previous performance, which results in an

(1) Set the index variable, $index = 0$
(2) Set the size of the Performance List, $listSize$
(3) Initialize probability, p_a
(4) Populate the Performance List (PL) with random solutions
(5) Calculate the fitness values of the solutions in PL, i.e. $PL\_Fitness$
(6) Sort PL and $PL\_Fitness$ according to $PL\_Fitness$
(7) Initialize $working$ to PL_{index}
(8) **for** i to $noOfIterations$ **do**
  (8.1) $working$ = Perform_Local_Search ($working$)
  (8.2) $f\_working$ = Evaluate ($working$)
  (8.3) **if** $f\_working$ better than $PL\_Fitness_{listSize-1}$ **then**
    (8.3.1) Update PL with $working$
    (8.3.2) Update $PL\_Fitness$ with $f\_working$
  (8.4) **end if**
  (8.5) **if** random$[0, 1] > p_a$ **then**
    (8.5.1) $index$ = Select index, e.g. Random$[0, listSize]$
    (8.5.2) $working = PL_{index}$
  (8.6) **end if**
(9) **end for**
(10) **return** PL_0

Algorithm 1

updated technique, the athlete may want to continue making slight changes to the updated techniques if he/she desires to do so. If the athlete wants to work with another performance from the PL, then the athlete will choose to do so. If improved techniques are found along the way, then the PL will get updated accordingly. After a sufficient amount of time, with enough strategizing and implementation, the athlete will determine the best technique to use, which will allow him/her to perform at best.

From a heuristic perspective, the best performances recorded on the PL refer to the best solutions found by the heuristic algorithm. The performance/solution that the athlete will consider working with is called the "working" solution. Local changes (slight changes) are made to this working solution in the hope of trying to determine an improved solution within the local neighbourhood structures of the solution space. If *updated* working solutions *at least* improve on the *worst* solution found on the PL, then the PL will get updated. The athlete will continue working with this updated working solution for the next iteration or choose another solution from the PL to be its new working solution, given a certain probability. The probability symbolizes the athletes' willingness to continue working with an updated working solution or not.

PL will always only get updated with solutions that give unique performance results. This will prevent the algorithm from working with solutions that produce identical results. After a predetermined number of iterations is completed the best solution found will be representative of the best technique determined by the athlete. This best solution will be the first solution registered on the PL.

The algorithm for the BPA is as Algorithm 1.

4.2. Iterative Best Performance Algorithm. With the BPA, an athlete determines improved techniques by making slight changes to the techniques used in delivering a limited number of the athletes best recorded performances (refer to Section 4.1). At different iterations of the algorithm, the performance/solution chosen to be worked with will either be a new performance selected from the Performance List (PL) or the updated performance worked with from the previous iteration. Working with an updated performance determines the "willingness" of the athlete to continue working with that previous worked with performance. This willingness is represented by a predetermined probability variable in the algorithm. Given this probability, the algorithm either works with a previous worked with performance or not.

The Iterative Best Performance Algorithm (IBPA) is modelled on the same principles as BPA. However, with IBPA, the athlete will continue to work with the *same* performance for a specified amount of time. This performance is viewed as a reference performance. Using this reference performance, the athlete will make slight changes to the technique used in delivering that performance in the hope of trying to determine improved techniques. The athlete will continue to do this for a specified amount of time, in order to be satisfied that enough attempts were made in working with an individual performance. After the athlete completes working with a reference performance, another reference performance will be chosen to be worked with from the Performance List (PL). In working with these reference performances improved techniques may be determined along the way. These improved techniques may lead to improved performances being delivered. If improved performances are delivered, then the PL will get updated accordingly.

In the implementation of IBPA, the reference performance is considered the "current" solution. This current solution remains the same for a predetermined number of iterations. This iteration count will be referred to as the "steps per change." The steps per change remain constant for the current solution worked with, for the *number* of current solutions that the athlete is willing to work with. The number of current solutions that the athlete is willing to work with is also specified by a predetermined number of iterations. This iteration count is referred to as the "number of iterations."

For each step per change, local search (slight changes) is performed on the current solution. This will generate a "working" solution. Similar to BPA, if the working solution *at least* improves on the *worst* solution on the PL, then the PL will get updated accordingly. After the number of steps per change complete, in working with the current solution, another current solution will get chosen from the PL for the next set of steps per change. This process will continue until the number of iterations complete. After the number of iterations is completed, the best solution determined will be the first solution on the PL. This solution is representative of the best technique determined by the athlete.

The algorithm for the IBPA is as Algorithm 2.

4.3. Largest Absolute Difference Algorithm. Difference, in mathematical terms, is the amount which remains after one

(1) Set the index variable, $index = 0$
(2) Set the size of the Performance List, $listSize$
(3) Populate the Performance List (PL) with random solutions
(4) Calculate the fitness values of the solutions in PL, i.e. $PL\_Fitness$
(5) Sort PL and $PL\_Fitness$ according to $PL\_Fitness$
(6) Initialize $current$ to PL_{index}
(7) **for** i to $noOfIterations$ **do**
  (7.1) **for** j to $stepsPerChange$ **do**
    (7.1.1) $working$ = Perform_Local_Search ($current$)
    (7.1.2) $f\_working$ = Evaluate ($working$)
    (7.1.3) **if** $f\_working$ better than $PL\_Fitness_{listSize-1}$ **then**
      (7.1.3.1) Update PL with $working$
      (7.1.3.2) Update $PL\_Fitness$ with $f\_working$
    (7.1.4) **end if**
  (7.2) **end for**
  (7.3) $index$ = Select $index$, e.g. Random[0, $listSize$]
  (7.4) $current = PL_{index}$ ($current_i \neq current_{i-1}$)
(8) **end for**
(9) **return** PL_0

Algorithm 2

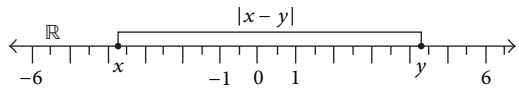


Figure 1: The absolute difference between the values x and y.

quantity is subtracted from another. An example is when the number 3 is subtracted from the number 6. The remainder is equivalent to −3. The remainder is negative because 3 is less than 6.

The *absolute difference* between two real numbered values x and y is the absolute value of their difference. It is denoted by $|x - y|$ and is mathematically defined as follows:

$$|x - y| = \begin{cases} (x - y), & \text{if } (x - y) \geq 0, \\ -(x - y), & \text{if } (x - y) < 0. \end{cases} \quad (8)$$

The absolute difference will always be either positive or zero (if $x \equiv y$). On a real line, it can be seen as the magnitude or difference between points x and y. This can be seen in Figure 1.

The Largest Absolute Difference Algorithm (LADA) is modelled on the ability to calculate an absolute difference between real numbers.

During an optimization process [15], a solution vector $x \in \theta \subseteq \mathbb{R}^p$ is the input vector to the objective function f. x is the p-dimensional vector of *design variables* of f; that is, $x = \{x_1, \ldots, x_p\}$. Design variables can be continuous or discrete depending on the type of optimization problem. The *values* of the design variables will determine the state (or quality) of the objective function within the domain of the solution space. Several solutions can exist depending on the different values of the design variables. By taking two of these solutions x_i and x_j, a vector of absolute differences (d) can be determined by calculating the absolute differences of the values of the *adjacent elements* of vectors x_i and x_j. d is determined by using

$$d_k = \left| x_{i,k} - x_{j,k} \right| \quad \text{for } k = 1, \ldots, p. \quad (9)$$

The elements of d are indicative of how far away from each other are the adjacent elements of the solution vectors x_i and x_j. The indices of d, which are indicative of the smallest absolute differences, represent the indices of x_i and x_j that are *most* similar. The indices d with the largest absolute differences represent the indices of x_i and x_j that are *least* similar. By performing local search on the adjacent elements of x_i and x_j, indexed by the largest absolute differences of d, new solution vectors x_i' and x_j' can be determined. If these new "child" solutions improve on their "parent" solutions, then these solutions will get drawn closer together in moving towards the global optimum. By performing this local search technique on a population of solutions, the population will converge towards the global optimum in an iterative way.

LADA is implemented by maintaining a population of solutions in a list called the Solutions List (SL). SL must at least be greater than or equal to 2. Also, the best solution found in SL must be recorded in a variable called *best*. LADA is executed for a specified number of iterations. At each iteration l, two solutions x_i and x_j will be randomly selected from SL ($i \neq j$). x_i and x_j get copied respectively into their "working" variables $working_i$ and $working_j$. Using $working_i$ and $working_j$, the vector of absolute differences d_l can be determined. To implement local search, using d_l, the number of largest absolute differences to be worked with must be specified. This is given by the variable m, where $0 < m \leq n$. Having determined d_l and knowing m, two new child solutions are generated by making permissible changes to $working_i$ and $working_j$. If $working_i$ provides a better quality solution than SL_i, then SL_i will be replaced by $working_i$. Similarly, if $working_j$ improves on SL_j, SL_j will be replaced by $working_j$. If $working_i$ or $working_j$ improves on *best*, then *best* must be updated accordingly. The quality of the solutions of $working_i$ and $working_j$ must *not* be identical to the quality of another solution found in SL. Disallowing identical quality solutions ensures the uniqueness of the solutions listed on the SL. After the specified number of iterations is completed the best solution found will be recorded in *best*.

The algorithm for the LADA is as Algorithm 3.

4.4. Tabu Search. Tabu Search (TS) is based on the idea of something that should not be interfered with [16, 17]. TS implements this idea by recording a specific number of unique best solutions found in a list called the Tabu List (TL). If a new solution is found, which improves on the solutions recorded in the TL, the new solution gets added to the TL. Any new solutions found, that is, identical to those that are already registered in the TL, will not be considered. This eliminates the possibility of exploiting identical moves.

(1) Set the size of the Solutions List, *listSize*
(2) Populate the Solutions List (SL) with random solutions
(3) Calculate the fitness values of the solutions in *SL*, i.e. *SL_Fitness*
(4) Set the no. of absolute differences to consider, *m*
(5) Set the best solution (*best*) and best fitness (*f_best*) using *SL_Fitness*
(6) **for** *i* to *noOfIterations* **do**
  (6.1) *index*1 = Select *index*1, e.g. Random[0, *listSize*]
  (6.2) *index*2 = Select *index*2, e.g. Random[0, *listSize*] (*index*1 ≠ *index*2)
  (6.3) $working\_1 = SL_{index1}$
  (6.4) $working\_2 = SL_{index2}$
  (6.5) $d = |working\_1 - working\_2|$
  (6.6) Perform_LS (*working_1*, *working_2*, *d*, *m*)
  (6.7) *f_working_1* = Evaluate (*working_1*)
  (6.8) *f_working_2* = Evaluate (*working_2*)
  (6.9) **if** *f_working_1* better than $SL\_Fitness_{index1}$ **then**
    (6.9.1) $SL_{index1} = working\_1$
    (6.9.2) $SL\_Fitness_{index1} = f\_working\_1$
    (6.9.3) **if** *f_working_1* better than *f_best* **then**
      (6.9.3.1) *best* = *working_1*
      (6.9.3.2) *f_best* = *f_working_1*
    (6.9.4) **end if**
  (6.10) **end if**
  (6.11) **if** *f_working_2* better than $SL\_Fitness_{index2}$ **then**
    (6.11.1) $SL_{index2} = working\_2$
    (6.11.2) $SL\_Fitness_{index2} = f\_working\_2$
    (6.11.3) **if** *f_working_2* better than *f_best* **then**
      (6.11.3.1) *best* = *working_2*
      (6.11.3.2) *f_best* = *f_working_2*
    (6.11.4) **end if**
  (6.12) **end if**
(7) **end for**
(8) **return** *best*

Algorithm 3

TS also maintains a record of the "best" overall solution. Using a "current" solution, TS generates a list of candidate solutions, which are local to the current solution. The new candidate solutions determined must be cross-referenced against the TL. This will eliminate the possibility of repeating identical moves. Once the candidate list is determined, the best candidate solution from the list can-found. This best candidate solution becomes the new current solution for the next iteration. If this new current solution improves on the best solution found so far, then it also gets recorded as the best solution and gets inserted into the TL. The TL is usually updated using the last in first out technique.

Generating new solutions is done in a deterministic way, using local search. This process continues iteratively for a specific number of iterations.

The algorithm for TS is as Algorithm 4.

4.5. Simulated Annealing. Simulated Annealing (SA) [18, 19] models the annealing process, when heated metal begins to cool. The hotter the metal gets, when heated, the more volatile its atomic structure will become. This will result in a weakened and more unstable structure. However, when the heated metal begins to cool, the highly energized metallic atoms loose energy and the structure begins to stabilize. When the metal is completely cooled, an equilibrium state is reached. The cooling process must be slow for the annealing to be successful. Reaching an equilibrium state is symbolic of an "optimal" solution being found for optimization problems.

(1) Generate an initial random solution = *best*
(2) Set *current* = *best*
(3) Evaluate the fitness of *best* = *f_best*
(4) Set the fitness of *current*(*f_current*) = *f_best*
(5) Set the size of the Tabu List, *tabuListSize*
(6) Set the size of the Candidate List, *candidateListSize*
(7) Initiate the Tabu List *TL* and the *CandidateList*
(8) **for** *i* to *noOfIterations* **do**
  (8.1) *CandidateList* = Generate_List (*current*)
  (8.2) *current* = Find_Best_Candidate (*CandidateList*)
  (8.3) *f_current* = Evaluate (*current*)
  (8.4) **if** *f_current* better than *f_best* **then**
    (8.4.1) *f_best* = *f_current*
    (8.4.2) *best* = *current*
    (8.4.3) Update *TL* with *current*
  (8.5) **end if**
(9) **end for**
(10) **return** *best*

Algorithm 4

SA starts off with randomly generated, but equivalent, "best," "current" and "working" solutions. It starts off with an initial temperature (T) and then decreases by a constant factor (α) until it reduces its final temperature (F). At each reduced temperature ($T \times \alpha$), SA iteratively searches for local solutions to the current solution. This constitutes the working solution. If the working solution is better than the current solution, the current solution is replaced by this working solution. If this current solution is better than the best solution, then the best solution becomes this current solution. Worst working solutions can replace the current solution, given a certain probability. This strategy reduces the chances of premature convergence.

This process continues until F is reached. F symbolizes an equilibrium state being reached where the best solution found will be given.

The algorithm for SA is as Algorithm 5.

5. Testing and Evaluation

The nonheuristic specific parameters, required for the execution of the algorithms, had been set according to the values given in Tables 2 and 3. The lower and upper bound settings for the different plot types are given in Table 2.

Table 3 gives the lower and upper bound settings, the land coverage fraction values, the cost of irrigated water, and the operational costs for each crop. The large differences in the lower and upper bound values were to investigate the ability of the heuristic algorithms in determining solutions in a larger solution space $F_{kij} \in [0, 1]$. C_{IRkij} is the cost of the

```
(1) Generate an initial random solution = best
(2) Set current = working = best
(3) Evaluate the fitness of best = f_best
(4) Set the fitness of current(f_current) and the fitness of
    working(f_working) = f_best
(5) Initiate starting temperature T and final temperature F
(6) while T ≥ F do
   (6.1) for i to stepsPerChange do
      (6.1.1) working = Generate_Solution (current)
      (6.1.2) f_working = Evaluate (working)
      (6.1.3) if f_working better than f_current then
         (6.1.3.1) use_solution = true
      (6.1.4) else
         (6.1.4.1) Calculate acceptance probability P
         (6.1.4.2) if P > random[0, 1] then
            (6.1.4.2.1) use_solution = true
         (6.1.4.3) end if
      (6.1.5) end else
      (6.1.6) if use_solution then
         (6.1.6.1) use_solution = false
         (6.1.6.2) f_current = f_working
         (6.1.6.3) current = working
         (6.1.6.4) if f_current better than f_best then
            (6.1.6.4.1) best = current
            (6.1.6.4.2) f_best = f_current
         (6.1.6.5) end if
      (6.1.7) end if
   (6.2) end for
   (6.3) Update T according to cooling schedule
(7) end while
(8) return best
```

Algorithm 5

Table 2: Lower and upper bounds for each plot type.

Plot types	Bounds (ha)	
	Lb_P_k	Ub_P_k
Single crop	10	1,700
Double crop	50	1,740

irrigated water per hectare per crop (ZAR ha^{-1}). O_{kij} is set to a third of the producer prices per ton of yield (ZAR ha^{-1}).

The initial parameters for the heuristic algorithms were set as follows:

(i) BPA—the *listSize* was set at 20. The *noOfIterations* was set at 100,000. p_a was set at 0.2.

(ii) IBPA—the *listSize* was set at 20. The *noOfIterations* was set at 5,000. The *stepsPerChange* was set at 20.

(iii) LADA—the *listSize* was set at 20. The *noOfIterations* was set at 50,000. m was set at 3.

(iv) TS—the *tabuListSize* was set at 7. The *candidateListSize* was set at 20. The *noOfIterations* was set at 5,000.

(v) SA—the *stepsPerChange* was set at 100. T was set at 230. F was set at 0.01. α was set at 0.99.

Table 3: Nonheuristic specific parameters required for the execution of the algorithms.

Crops	Lb_{kij}	Ub_{kij}	F_{kij}	C_IR_{kij}	O_{kij}
Lucerne (y)	10	1,700	1	877.26	6,259.52
Tomato (s)	10	1,740	1	685.11	71,478.00
Pumpkin (s)	10	1,740	1	451.66	10,408.80
Maize (s)	10	1,740	1	613.90	3,924.09
Groundnut (s)	10	1,740	1	502.08	5,025.24
Sunflower (s)	10	1,740	1	292.13	3,701.61
Barley (w)	12.5	1,740	1	413.68	4,124.88
Onion (w)	12.5	1,740	1	221.00	23,739.30
Potato (w)	12.5	1,740	1	186.10	22,758.12
Cabbage (w)	12.5	1,740	1	172.94	23,720.00

Table 4: The average execution times, in milliseconds, and the 95% Confidence Interval values for each heuristic algorithm.

Methods	AVG (ms)	95% CI
BPA	229	AVG ± 3
IBPA	223	AVG ± 3
LADA	147	AVG ± 2
TS	184	AVG ± 5
SA	212	AVG ± 3

To compare the heuristic algorithms fairly, the heuristic specific parameter settings ensured that each algorithm is executed for 100,000 objective function evaluations. Each algorithm was then run 100 times using different population sets for each run.

The population sets had been initially randomly generated. Each population set contained *listSize* number of solutions, that is, 20. For explanation, we mathematically denote each population set as pop_i, for $i = 1, \ldots, 100$. Then, for each run i, pop_i was used as the input population set for BPA, IBPA, and LADA. This was to set the Performance List (PL) for BPA and IBPA and the Solutions List (SL) for LADA. The best solution from each pop_i was also used to initialize *best* for TS and SA.

From the 100 best solutions determined, by each heuristic algorithm, their overall best solutions and (where applicable) their average results have been documented. Using the populations of the 100 best solutions determined by each heuristic algorithm, the 95% Confidence Interval (CI) have been calculated. These were for the execution times and for the fitness values (total gross profits earned). The results are explained in the following.

Table 4 gives the statistics of the average execution times (AVG) in milliseconds (ms) and the 95% Confidence Interval (95% CI) values of each heuristic algorithm. It can be observed that LADA executed the fastest on average. This was followed by TS, SA, IBPA, and BPA. The relatively fast average execution time of LADA is due to its ability to work with two solutions per iteration.

The 95% CI values, from Table 4, mean that we can be 95% certain that the 100 execution times of each heuristic algorithm have fallen within those interval estimates.

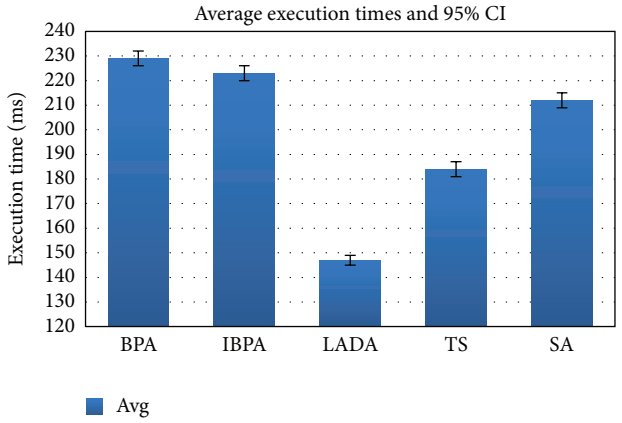


Figure 2: The average execution times, in milliseconds (ms), and the 95% CI values of each heuristic algorithm.

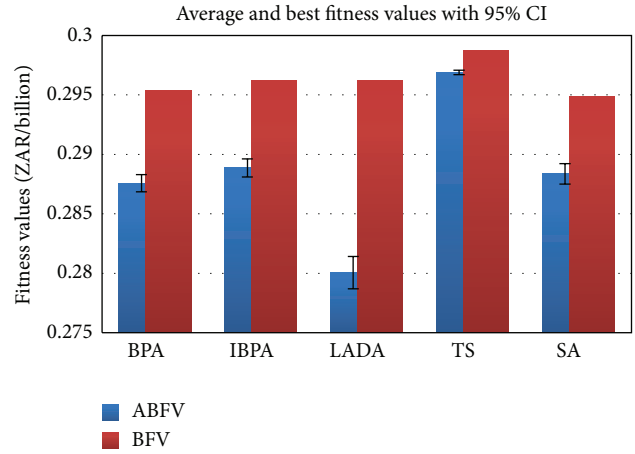


Figure 3: A comparison of each algorithm best and average fitness values determined along with the 95% CI estimates.

Table 5: Statistics for the Best Fitness Values (BFVs), Average Best Fitness Values (ABFVs), and 95% Confidence Interval (95% CI) values.

Methods	BFV (ZAR)	ABFV (ZAR)	95% CI
BPA	295,382,093	287,575,514	ABFV ± 732,543
IBPA	296,166,629	288,864,091	ABFV ± 756,861
LADA	296,241,511	280,062,612	ABFV ± 1,352,737
TS	298,765,873	296,886,105	ABFV ± 185,479
SA	294,824,404	288,363,133	ABFV ± 866,622

By observing those CI values, we conclude that the execution times of each algorithm have been fairly consistent. A visual representation of the statistical values from Table 4 is given in Figure 2 below. In Figure 2, the 95% CI values are represented by the black interval estimates.

Table 5 gives the statistical values of the overall best BFV and average best ABFV fitness values for each heuristic algorithm. The fitness values are the total gross profits earned. The 95% CI values for the fitness value populations, of each algorithm, are also given.

From Table 5, it is observed that TS determined the highest BFV. This was followed by LADA, IBPA, BPA, and then SA. On average, TS was also the best. This was followed by IBPA, SA, BPA, and then LADA. Although LADA's BFV was higher than IBPA, BPA, and SA, its average performance was the worst overall. This proves that LADA had the ability to determine good solutions, although it performed relatively poorly on average.

A graphical comparison of the algorithms best and average fitness values, as determined from Table 5, is given in Figure 3. The 95% CI values are represented by the black interval estimates over the average fitness values.

The solutions found by the algorithms were in a solution space of constantly changing plot-type hectare allocations. The hectare allocations for each plot-type needed to be determined first before the hectare allocations of the crops. The hectare allocations needed to satisfy the land constraints given in Section 2.6.

For each algorithm, the best solution determined from the "population" of solutions at iteration t, for plot-type hectare allocations p, will not necessarily be the best solution at iteration $(t + 1)$ for plot-type hectare allocations $(p + 1)$. The change in the plot-type hectare allocations at iteration $(t + 1)$ will change the crop hectare allocations accordingly, so the land constraints do not break. The constantly changing dimensions of the solution space make it very difficult for the algorithms to perform exploitation. This makes the problem difficult, in determining effective solutions.

Under the circumstance of the constantly changing dimensions of the solution space, TS had performed most consistently. This is confirmed by its low 95% CI fitness value. BPA had the second lowest 95% CI fitness value. This is followed by IBPA, SA, and LADA. By observing and comparing each algorithm BFV, ABFV, and 95% CI fitness value solutions, we conclude that TS had been the strongest heuristic algorithm, in providing solutions to this particular optimization problem.

The strength of TS, in performing as the overall best, is due to its strong exploitation ability. At iteration t, generating a candidate list of solutions allows for TS to maximize its exploitation within the local neighbourhood structure of the solution space for plot-type hectare allocations p. The best candidate solution determined at iteration t will be the best solution found for plot-type hectare allocations p, but as explained earlier, it will not necessarily be the best "working" solution at iteration $(t + 1)$ for plot-type hectare allocations $(p + 1)$. However, if $(p + 1)$ is very similar to p, then the working solution at iteration $(t + 1)$ will become very valuable in trying to effectively exploit the local neighbourhood structure of the solution space even further. The possibility of $(p + 1)$ being similar to p and in using the best candidate solution, from iteration t, as the working solution at iteration $(t + 1)$ encourages further exploitation. This is the reason why TS has performed well.

Similar to TS, IBPA uses a "current" solution to perform exploitation at each iteration t for a certain number of "steps per change." The solution chosen as the current solution

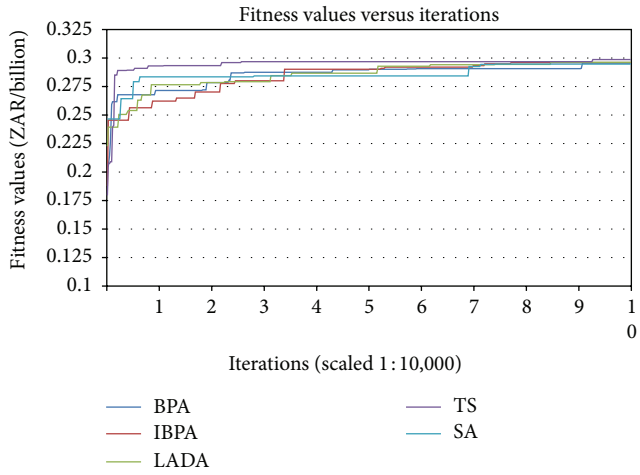


FIGURE 4: The performance of the heuristic algorithms in determining their overall best fitness value solutions.

TABLE 6: Statistics of the irrigation water requirements (IWR) and variable costs of production (VCP) for the best solutions found.

Methods	IWR (m^3)	VCP (ZAR)
BPA	16,922,183	147,701,718
IBPA	16,961,536	148,093,316
LADA	17,244,651	74,544,333
TS	17,142,919	149,397,333
SA	17,070,610	147,446,530

at iteration t is restricted to the solutions listed on the Performance List (PL). Any "working" solution generated from the current solution, at iteration t, will therefore not necessarily be related to the current solution chosen at iteration $(t + 1)$. This holds even if any working solution generated updates the PL. The possibility of further exploiting a local neighbourhood structure of the solution space if p is very similar to $(p + 1)$ is therefore minimized.

The purpose of maintaining updated lists of their best solutions found, for BPA, IBPA, and LADA, is to facilitate exploration of the solution space. Performing local search facilitates exploitation. For this particular optimization problem, IBPA and BPA determined a better balance in performing exploration and exploitation, compared to LADA. This is concluded in comparing their performances to SA. SA has a naturally good balance in its ability to perform exploration and exploitation. LADA seems to be stronger in its explorative ability. This explains its relatively high BFV solution found and its relatively low ABFV performance.

Figure 4 shows the performances of the heuristic algorithms in them determining their best fitness value (BFV) solutions. It is seen that TS had clearly outperformed all heuristic algorithms in determining its BFV. SA had initially progressed at a very fast rate, up to about 10,000 objective function evaluations, compared to BPA, IBPA, and LADA. BPA, IBPA, and LADA performed very similarly in progressively improving on their BFV performances.

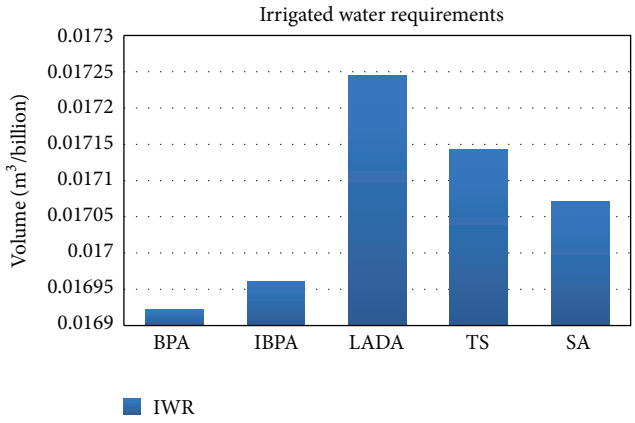


FIGURE 5: Irrigated Water Requirements (IWR) of the best heuristic solutions.

Table 6 gives the statistics of the irrigation water requirements (IWR) and the variable costs of production (VCP) for the best solution determined by each of the heuristic algorithms. BPA's solution required the least amount of irrigated water. At a cost of ZAR 0.0877 m^{-3}, the cost of this irrigated water is ZAR 1,484,075. The IWR of IBPA, SA, TS, and LADA was a volume of 39,353, 148,427, 220,736, and 322,468 m^3 more than BPA's IWR, respectively. At a water quota of 8,417 $m^3\,ha^{-1}annum^{-1}$, BPA's IWR value would have supplied irrigated water to 4, 17, 26, and 38 ha's *less* than the IWR of IBPA, SA, TS, and LADA, respectively.

A graphical representation of the IWR's, as determined from Table 6, is shown in Figure 5.

From Table 6, it is also observed that the variable costs of production (VCP) of SA, BPA, IBPA, and TS are similar. From these four algorithms, SA's VCP value is the lowest and TS's VCP value is the highest. Interestingly enough, LADA's VCP value is about half in comparison to the VCP values of each of the other heuristic algorithms. Compared to SA, LADA's VCP value is ZAR 72,902,197 less. In comparison to TS, LADA's VCP value is ZAR 74,853,000 less. Although TS determined a best solution overall that earned an extra gross profit of ZAR 2,524,362 and required a volume of 101,732 m^3 less of irrigated water, in comparison to LADA's best solution, the remarkable saving in LADA's VCP value means that LADA determined the most economically feasible solution, from all heuristic algorithms.

A graphical representation of the VCP values, as determined from Table 6, is shown in Figure 6.

Table 7 gives the plot-type hectare allocations for the best solution found by each heuristic algorithm. BPA, IBPA, TS, and SA determined that the total gross profits will be greater in allocating more land for the double-crop plots of land. LADA's best solution determined that allocating more land to the single-crop plots would be better. This is despite Lucerne's relatively high IWR and relatively low producer price t^{-1} values, compared to the other crop types.

Figure 7 gives a graphical comparison of the seasonal hectare allocations of each crop, for the best solution determined by each heuristic algorithm. For the single-crop

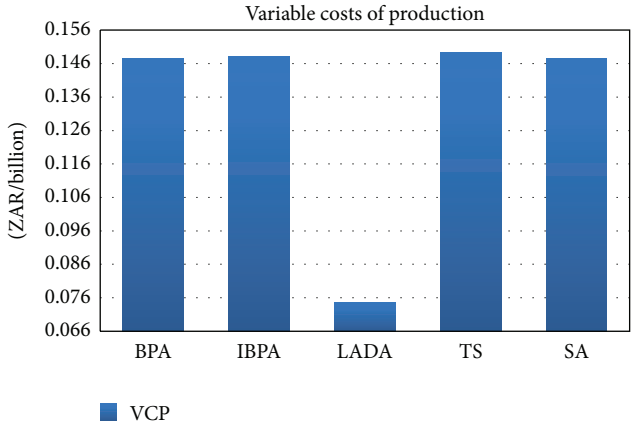


Figure 6: The variable costs of production (VCP) values of the best heuristic solutions.

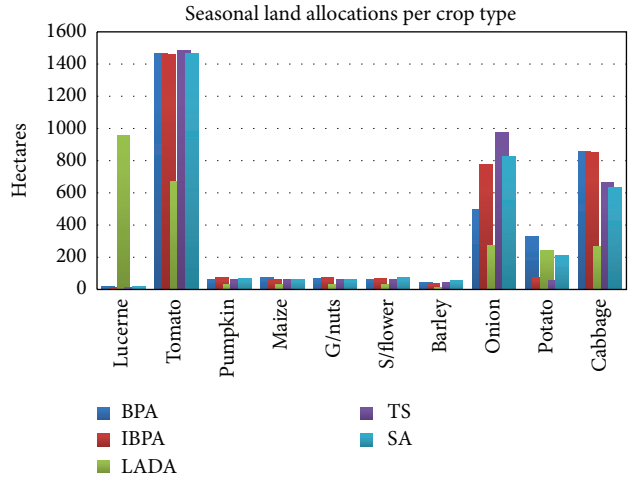


Figure 7: A comparison of the hectare allocations, per crop, for the best solution found by each heuristic algorithm.

Table 7: Plot-type hectare allocations for each heuristic algorithm.

Methods	Single-crop plots	Double-crop plots
BPA	17	1733
IBPA	12	1738
LADA	956	794
TS	14	1736
SA	18	1732

plots of land, BPA, IBPA, TS, and SA determined similar hectare allocations for Lucerne. LADA's hectare allocation was clearly higher. For the double-crop plots of land, all heuristic algorithms allocated the most amount of land to Tomato, Onion, and Cabbage. BPA, LADA, and SA allocated a relatively higher percentage of land for Potato, compared to IBPA and TS. The large hectare allocations for Tomato are due to its high yield ha^{-1} and high producer price t^{-1} values. Similar hectare allocations were determined for Pumpkin, Maize, Ground Nuts, and Sunflower, by each algorithm.

Table 8 gives the statistical values of each crop hectare allocation (ha's $crop^{-1}$), irrigated water requirements (IWR), and variable costs of production (VCP) for the best solution determined by each heuristic algorithm.

The program was written in the Java programming language. It was programmed using the Netbeans 7.0 Integrated Development Environment. All simulations were run on the same platform. The computer used had a Windows 7 Enterprise operating system, an Intel Celeron Processor 430, 3 GB of RAM, and a 500 GB hard drive.

In developing object-oriented versions of these LS metaheuristic algorithms, each algorithm was relatively easy to implement. Each algorithm also requires few parameter settings.

6. Conclusion

The shortages in food supply, and the increases in population growth, have increased the need for more food to be produced. To try and meet this *growing* demand for food, it is important that new Irrigation Schemes be developed to increase the agricultural output.

The planning of new Irrigation Schemes requires that optimized solutions be found concerning the seasonal hectare allocations of the crops that are required to be grown within the year. The solutions found must seek to maximize the total gross profits that can be earned, in making the most efficient usage of the limited resources available for agricultural production. Determining solutions to this problem is referred to as Annual Crop Planning (ACP). ACP is an NP-hard-type optimization problem in agricultural planning.

This research has introduced a new ACP mathematical model. The model is intended to be used to determine solutions to the ACP problem at a new Irrigation Scheme.

The case study in this paper is the Taung Irrigation Scheme (TIS), situated in the North West Province of South Africa. The Irrigation Scheme is currently being expanded to cater for an extra 1,750 hectares of irrigated land. This portion of land is required to grow 10 different types of crops.

To determine solutions for this ACP problem, three new Local Search (LS) metaheuristic algorithms have been introduced in the literature. These algorithms have been investigated in trying to determining near-optimal solutions for this problem. The new algorithms introduced are the Best Performance Algorithm (BPA), the Iterative Best Performance Algorithm (IBPA), and the Largest Absolute Difference Algorithm (LADA). To determine the relative merits of their solutions found, their solutions have been compared against the solutions of two other well-known LS metaheuristic algorithms in the literature. These popular metaheuristic algorithms are Tabu Search (TS) and Simulated Annealing (SA).

To ensure fairness in the performances of the heuristic algorithms, their parameter specific settings had been set to execute for the same number of objective function evaluations. The list sizes for BPA, IBPA, and LADA were also set to be the same. Each heuristic algorithm was then run

Table 8: Crop statistics of the best solution determined by each heuristic algorithm.

Crops	Methods	ha's crop^{-1}	IWR (m^3)	VCP (ZAR)
Lucerne	BPA	17	169,016	120,587
	IBPA	12	123,883	88,386
	LADA	956	9,560,965	6,821,407
	TS	14	138,942	99,130
	SA	18	175,621	125,299
Tomato	BPA	1,465	11,442,080	105,695,864
	IBPA	1,461	11,416,473	105,459,327
	LADA	671	5,242,001	48,422,824
	TS	1,483	11,587,560	107,039,732
	SA	1,463	11,426,259	105,549,719
Pumpkin	BPA	62	318,126	670,872
	IBPA	73	375,268	791,375
	LADA	31	159,882	337,164
	TS	62	319,971	674,763
	SA	67	343,852	725,124
Maize	BPA	75	522,969	339,033
	IBPA	63	443,563	287,555
	LADA	30	211,339	137,008
	TS	63	437,786	283,810
	SA	65	454,319	294,528
Ground Nuts	BPA	69	392,341	378,794
	IBPA	73	416,717	402,328
	LADA	32	184,256	177,894
	TS	64	363,636	351,080
	SA	61	351,911	339,759
Sunflower	BPA	63	211,220	253,245
	IBPA	67	223,812	268,341
	LADA	30	99,098	118,815
	TS	65	215,246	258,072
	SA	77	255,319	306,118
Barley	BPA	46	216,110	207,935
	IBPA	36	171,475	164,988
	LADA	12	57,471	55,297
	TS	41	195,287	187,899
	SA	59	278,448	267,915
Onion	BPA	499	1,258,048	11,961,592
	IBPA	775	1,952,043	18,560,133
	LADA	276	695,752	6,615,253
	TS	974	2,455,286	23,345,004
	SA	827	2,084,191	19,816,604
Potato	BPA	329	699,027	7,558,257
	IBPA	73	155,092	1,676,941
	LADA	241	512,445	5,540,829
	TS	57	121,629	1,315,123
	SA	211	448,211	4,846,302
Cabbage	BPA	859	1,693,246	20,515,539
	IBPA	854	1,683,210	20,393,942
	LADA	264	521,442	6,317,842
	TS	663	1,307,576	15,842,720
	SA	635	1,252,479	15,175,162

100 times. For each run, a different population set was used as the input "population" for each heuristic algorithm. From the 100 solutions determined, by each algorithm, the overall best solution and the average performances had been documented.

The solutions found by the heuristic algorithms were in a solution space of constantly changing dimensions. This circumstance made it very difficult for the algorithms to determine effective solutions. Having stronger exploitation abilities would have been beneficial to the heuristic algorithms, for this particular optimization problem.

Our results show that TS performed as the overall best technique. It determined the best overall solution and was the best on average. Its consistent performance in determining good solutions was confirmed by its low 95% Confidence Interval (CI) fitness value. The second best solution was determined by LADA. This was followed by IBPA, BPA, and then SA. On average, IBPA performed the second best. This was followed by SA, BPA, and then LADA. The average execution times of BPA, IBPA, TS, and SA were similar. The execution time on LADA was clearly the fastest.

Although LADA performed the worst on average, its best solution required the least financial investment. This investment was nearly half of the investment required by the best solution determined for each of the other heuristic algorithms. This point, combined with LADA's relatively good best fitness value, meant that LADA's best solution was the most economically feasible.

The advantage of each algorithm is its ease of implementation. Each algorithm also only requires few parameter settings.

Possible future work will be to investigate the effectiveness of employing BPA, IBPA, and LADA in providing solutions to other NP-hard-type optimization problems in the literature.

References

[1] G. H. Schmitz, N. Schutze, and T. Wohling, "Irrigation control: towards a new solution of an old problem," in *International Hydrological Programme (IHP) of UNESCO and the Hydrology and Water Resources Programme (HWRP) of WMO*, vol. 5 of *IHP/HWRP-Berichte*, Koblenz, Germany, 2007.

[2] Department of Water Affairs and Forestry, "Vaal River System: Feasibility Study for Utilization of Taung Dam Water," Irrigation Planning and Design, http://www.dwaf.gov.za/.

[3] M. Pant, R. Thangaraj, D. Rani, A. Abraham, and D. K. Srivastava, "Estimation using differential evolution for optimal crop plan," in *Hybrid Artificial Intelligence Systems*, vol. 5271 of *Lecture Notes in Computer Science*, pp. 289–297, Springer, 2008.

[4] M. Pant, R. Thangaraj, D. Rani, A. Abraham, and D. K. Srivastava, "Estimation of optimal crop plan using nature inspired metaheuristics," *World Journal of Modelling and Simulation*, vol. 6, no. 2, pp. 97–109, 2010.

[5] P. E. Georgiou and D. M. Papamichail, "Optimization model of an irrigation reservoir for water allocation and crop planning under various weather conditions," *Irrigation Science*, vol. 26, no. 6, pp. 487–504, 2008.

[6] R. Wardlaw and K. Bhaktikul, "Application of genetic algorithms for irrigation water scheduling," *Irrigation and Drainage*, vol. 53, no. 4, pp. 397–414, 2004.

[7] R. Sarker and T. Ray, "An improved evolutionary algorithm for solving multi-objective crop planning models," *Computers and Electronics in Agriculture*, vol. 68, no. 2, pp. 191–199, 2009.

[8] K. S. Raju and D. N. Kumar, "Irrigation planning using genetic algorithms," *Water Resources Management*, vol. 18, no. 2, pp. 163–176, 2004.

[9] M. J. Reddy and D. N. Kumar, "Optimal reservoir operation for irrigation of multiple crops using elitist-mutated particle swarm optimization," *Hydrological Sciences Journal*, vol. 52, no. 4, pp. 686–701, 2007.

[10] R. J. Maisela, *Realizing agricultural potential in land reform: the case of Vaalharts Irrigation Scheme in the Northern Cape Province [M.S. thesis]*, University of the Western Cape, Cape Town, South Africa, 2007.

[11] Department of Agriculture, Forestry and Fisheries, "Trends in the agricultural sector 2012," http://www.daff.gov.za/docs/statsinfo/Trends2011.pdf.

[12] Department of Agriculture, Forestry and Fisheries, "Abstract of agricultural statistics 2012," http://www.nda.agric.za/docs/statsinfo/Ab2012.pdf.

[13] Department: Agriculture & Environmental Affairs, "Expected Yields," http://www.kzndae.gov.za/.

[14] B. Grove, *Stochastic efficiency optimisation analysis of alternative agricultural water use strategies in Vaalharts over the long- and short-run [Ph.D. thesis]*, Department of Agricultural & Economics, University of the Free State, Bloemfontein, South Africa, 2008.

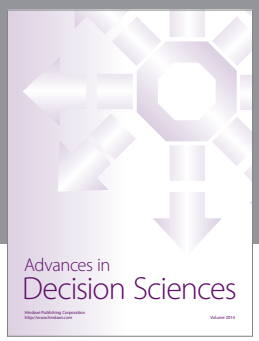
[15] T. Weise, "Global Optimization Algorithms—Theory and Application," Self-published electronic book, 2006–2009, http://www.it-weise.de/projects/book.pdf.

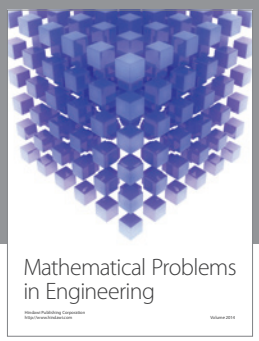
[16] F. Glover, "Tabu search—part 1," *ORSA Journal on Computing*, vol. 1, no. 2, pp. 190–206, 1989.

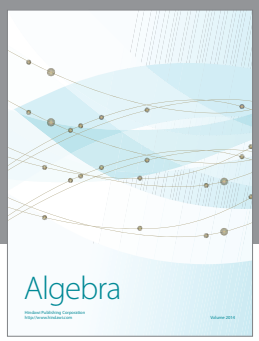
[17] F. Glover, "Tabu search—part 2," *ORSA Journal on Computing*, vol. 2, no. 1, pp. 4–32, 1990.

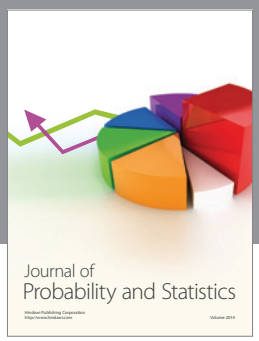
[18] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

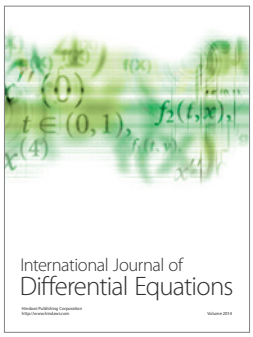
[19] C. M. Tan, *Simulated Annealing*, In-Tech, 2008.

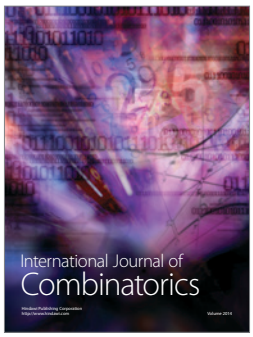


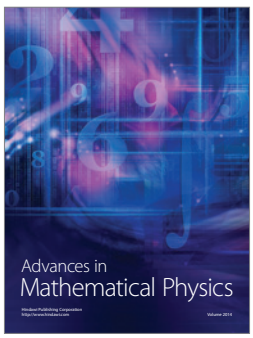


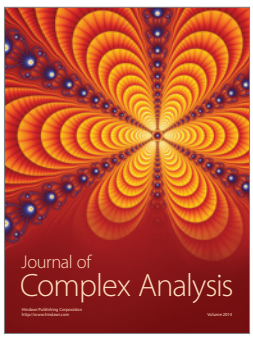


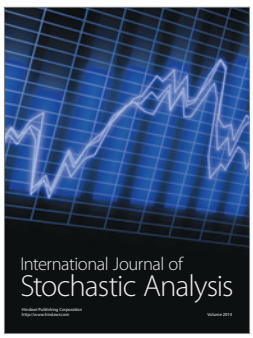


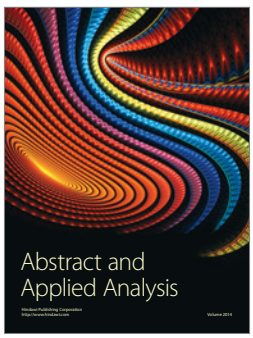


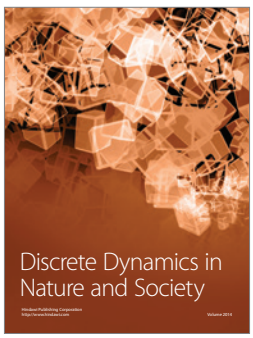


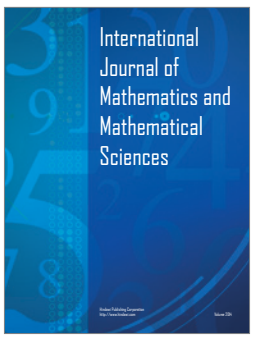


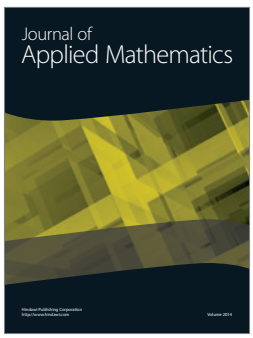












Hindawi
Submit your manuscripts at
http://www.hindawi.com