# A Multi-Criteria Approach for Nurse Scheduling

## Fuzzy Simulated Metamorphosis Algorithm Approach

Michael Mutingi
Faculty of Engineering and the Built Environment
University of Johannesburg
Johannesburg, South Africa
mmutingi@gmail.com

Charles Mbohwa
Faculty of Engineering and the Built Environment
University of Johannesburg
Johannesburg, South Africa
cmbohwa@uj.ac.za

***Abstract*—Motivated by the biological metamorphosis process and the need to solve multi-objective optimization problems with conflicting and fuzzy goals and constraints, this paper proposes a simulated metamorphosis algorithm, based on the concepts of biological evolution in insects, such as moths, butterflies, and beetles. By mimicking the hormone controlled evolution process the algorithm works on a single candidate solution, going through initialization, iterative growth loop, and finally maturation loop. The method is a practical way to optimizing multi-objective problems with fuzzy conflicting goals and constraints. The approach is applied to the nurse scheduling problem. Equipped with the facility to incorporate the user's choices and wishes, the algorithm offers an interactive approach that can accommodate the decision maker's expert intuition and experience, which is otherwise impossible with other optimization algorithms. By using hormonal guidance and unique operators, the algorithm works on a single candidate solution, and efficiently evolves it to a near-optimal solution. Computational experiments show that the algorithm is competitive.**

***Keywords*—Simulated metamorphosis, fuzzy set theory, multi-objective optimization, nurse scheduling, evolutionary algorithm***

## I. INTRODUCTION

The most desired practical objective in nurse scheduling is to produce high quality work schedules, so that (i) individual nurse preferences are satisfied and workload is balanced, (ii) patients are satisfied with the quality of service, and (iii) management goals are satisfied. Since the desires are often conflicting, imprecise, and uncertain in a non-stochastic sense, decision making is difficult. This is commonplace in healthcare organizations [1][2]. In a fuzzy environment, addressing conflicting multi-criteria decision problems requires interactive tools that are fast, flexible, and easily adaptable to specific problems. Decision makers often desire to use judicious approaches that can find a cautious tradeoff between the many goals, which is a common scenario in real world problems [3]. Addressing ambiguity, imprecision, and uncertainties of the desired goals is highly desirable in practice [4][5]. For instance, in a hospital setting, where nurses are often allowed to express their preferences on shift schedules, the decision maker has to incorporate the imprecision in preferences and management goals and choices. To achieve shift fairness and equity among the nursing staff, it is important to balance workload assignment. Patient preferences and expectations have to be considered as well [1][6]. In view of these issues, this paper presents a fuzzy simulated metamorphosis algorithm, inspired by the biological metamorphosis evolution. The algorithm is motivated by the need for interactive, fuzzy multi-criteria, and fast optimization approaches to solving problems with fuzzy multi-criteria problems. Thus, the specific objectives are:

1. To present the basic biological metamorphosis evolution process;
2. To derive from the metamorphosis concepts, a multi-criteria fuzzy evolutionary algorithm; and,
3. To apply the algorithm to typical nurse scheduling problems, demonstrating its effectiveness.

The rest of the paper is as follows. The next section introduces the nurse scheduling problem and the basic concepts of metamorphosis evolution. Section III presents a simulated metamorphosis algorithm. In Section IV, a fuzzy simulated metamorphosis is proposed for solving the nurse scheduling problem. Computational analyses are provided in Section V. Section VI concludes the paper.

## II. PRELIMINARIES

### A. The Nurse Scheduling Problem

The NSP is a hard optimization problem that involves assignment of different types of shifts and off days to nurses over a period of up to one month. The decision maker considers a number of conflicting objectives, choices, and preferences associated with the healthcare organization and individual nurses [7][8][9]. In practices, contractual work agreements govern the assignable shifts and off days per week. Imprecise personal preferences should be satisfied as much as possible. Typically nurses are entitled to day shift *d*, night shift *e*, and late night shift *n*, and holidays or days-off *o* [10][11]. Table I lists typical shifts and their time allocations.

TABLE I. TYPICAL SHIFTS

| Shift | Shift Description | Time allocation |
|---|---|---|
| 1 | *d*: day shift | 0800 - 1600 hrs |
| 2 | *e*: night shift | 1600 - 2400 hrs |
| 3 | *n*: late night shift | 0000 - 0800 hrs |
| 4 | *o*: off days as nurse preferences | |

The primary aim is to search for a schedule that satisfies a given set of hard constraints while minimizing a specific cost function [8][12]. However, in practice, individual nurse

978-1-4799-6065-1/15/$31.00 ©2015 IEEE

preferences, which are often imprecise, should be satisfied to the highest degree possible; the higher the degree of satisfaction, the higher the schedule quality. This ensures healthcare service quality and job satisfaction.

In this study, we classify constraints into sequence, schedule and roster constraints as listed in Table II. A sequence constraint pertains to the successive order of shifts in an individual nurse schedule or shift pattern. A schedule constraint relates to the restrictions on the complete nurse schedule covering the planning period, based on criteria such as workload and number of night shifts. On the other hand, a roster constraint controls the combination of nurse schedules based on criteria such as shift coverage and congeniality.

TABLE II. TYPICAL CONSTRAINTS TYPES

| Constraints | Description of the constraint |
|---|---|
| Sequence Constraints | A1: Shift sequences (*n-d*), (*n-e*), and (*e-d*) not permissible |
| | A2: Minimum rest time between night shift n |
| | A3: Maximum and minimum working time |
| Schedule constraints | B1: Fair or equal total workload assignment |
| | B2: Interval between night shifts should ≥ 1 week |
| | B3: Fair number of requested days-off or holiday assigned |
| Roster Constraints | C1: Shift coverage requirements to fulfil service quality |
| | C2: Tutorship - a trainer has to work with a specific trainee |
| | C3: Congeniality, compatibility of workmates |

## B. Metamorphosis: Basic Concepts.

Metamorphosis is an evolutionary process common in insects such as butterflies [13], as illustrated in Fig. 1. The process begins with an egg that hatches into an instar larva (instar). Subsequently, the first instar transforms into several instar larvae, then into a pupa, and finally into the adult insect [14]. The process is uniquely characterized with radical evolution, hormone controlled growth and maturation.

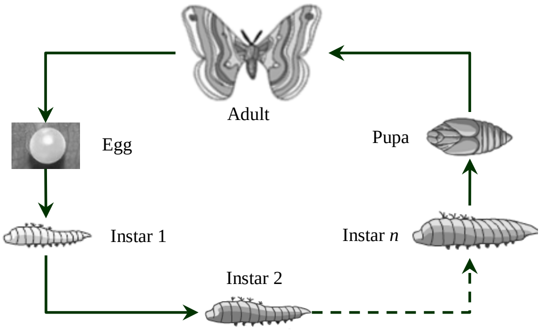

Fig. 1. Metamorphosis evolution

Metamorphosis implies change of physical form, structure, or substance; a marked and more or less abrupt developmental change in the form or structure of an animal (such as a butterfly or a frog) occurring subsequent to hatching or birth [14]. A species changes body shape and structure at a particular point in its life cycle, such as when a tadpole turns into a frog.

Insect molting and development is controlled by several hormones. The hormones trigger the insect to shed its exoskeleton and, at the same time, grow from smaller juvenile forms (e.g., a young caterpillar) to larger adult forms (e.g., a winged moth). The hormone that causes an insect to molt is called ecdysone. The hormone, in combination with a juvenile hormone, determines whether the insect will metamorphose.

## III. A SIMULATED METAMORPHOSIS ALGORITHM

Simulated Metamorphosis (SM) is an evolutionary approach to metaheuristic optimization inspired by the natural biological process of metamorphosis in many insect species. The approach is motivated by several fuzzy multi-criteria decision problems in the operations research and operations management community, such as vehicle routing problems [15], nurse scheduling [5][2][6], and task assignment [10]. Such fuzzy decision problems are associated with conflicting imprecise goals, and the need to incorporate choices, intuitions and expert judgments of the decision maker [1]. As a fuzzy multi-criteria heuristic approach, SM seeks to bridge this gap.

There are three basic phases in the simulated metamorphosis algorithm: initialization, growth, and maturation. Each of these phases has specific operators. Fig. 2 outlines the simulated metamorphosis algorithm.

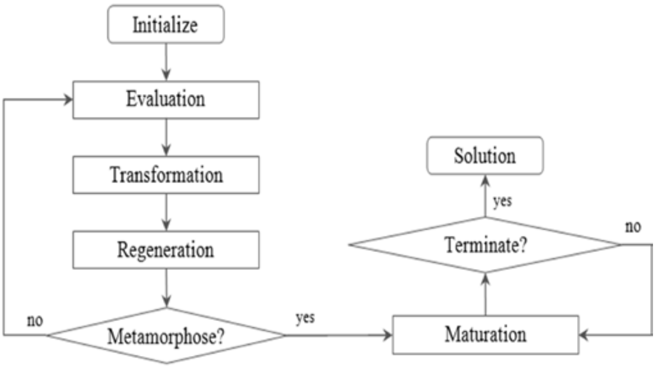

Fig. 2. Metamorphosis evolution

## A. Initialization

In the initialization stage, an initial solution is created as a seed for the evolutionary algorithm. In our approach, we use a problem specific heuristic that is guided by hard constraints of the problem. This ensures generation of a feasible initial solution. Alternatively, a decision maker can enter a user-generated solution as a seed. The initial candidate solution $s_0$ consists of constituent elements $e_i$ ($i = 1,\ldots,I$) where $I$ is the constituent number of elements in the candidate solution.

## B. Growth

The growth phase comprises the evaluation, transformation, and the regeneration operators.

### 1) Evaluation

The choice of the evaluation function is crucial to the evaluation procedure. First, the evaluation function should ensure that it measures the relevant quality of the candidate solution. Second, the function should capture the actual problem characteristics, particularly the imprecise, conflicting and multi-objective nature of the goals and constraints. Third, the fitness function should be easy to evaluate. The function $F_t$, at iteration $t$, is a normalized function of normalized functions

$\mu_h$ ($h = 1,\ldots,n$), where $n$ is the number of constituent objective functions. Thus, using multi-factor evaluation,

$$F_t(s_t) = \sum_h w_h \mu_h(s_t) \quad (1)$$

where, $s_t$ is the current solution at iteration $t$; and $w_h$ denotes the weight of the function $\mu_h$.

*2) Transformation.*

The growth mechanism is achieved through selection and transformation. Selection determines whether a constituent element ei of the candidate solution st should be retained for the next iteration, or selected for transformation operation. The goodness or fitness ηi of element ei ($i = 1,\ldots,I$) is compared with probability pt∈[0,1], generated at each iteration $t$. That is, if $\eta_i \leq p_t$, then $e_i$ is transformed, otherwise, it will survive into the next iteration. Deriving from the biological metamorphosis, the magnitude of $p_t$ should decrease over time to guarantee convergence. From preliminary empirical computations, $p_t$ should follow a decay function,

$$p_t = p_0 e^{-at/T} \quad (2)$$

where, $p_0 \in [0,1]$ is a randomly generated number; $T$ is the maximum number of iterations; $a$ is an adjustment factor.

It follows that the higher the goodness, the higher the likelihood of survival in the current solution. Therefore, elements with low fitness are subjected to growth. The magnitude of $p_t$ controls the growth rate, emulating the inhibition/juvenile hormone. To avoid loss of performing elements, new elements are compared with the rejected ones, keeping the better ones. A pre-determined number of rejected elements are kept in list $Q$ for future use in regeneration.

*3) Regeneration.*

The regeneration operator has a repair mechanism that considers the feasibility of the candidate solution. All infeasible elements are repaired using problem domain specific heuristics. Elements in the reject list $Q$ are used as food for the repair mechanism. After regeneration, the candidate solution is tested for readiness for transition to the maturation phase. This is controlled by the dissatisfaction level (juvenile hormone) $m_t$,

$$m_t = 1 - \mu_1 \wedge \mu_2 \wedge \ldots \wedge \mu_n \quad (3)$$

Here, $\mu_1,\ldots,\mu_n$, represent the satisfaction level of the respective objective functions; "∧" is the min operator. This implies that the growth phase repeats until a pre-defined acceptable dissatisfaction $m_0$ is reached. However, the algorithm proceeds to maturation if there is no significant change in $m_t$ after a pre-defined number of trials.

*4) Maturation*

The maturation phase is a loop consisting of intensification and post-processing to bring the candidate solution to maturity.

*5) Intensification*

The aim of the intensification operator is to ensure complete search of an improved solution in the neighborhood of the current solution. This helps to improve the current solution further. Howbeit, at this stage, the juvenile hormone has ceased to control or balance the growth of the solution according to the constituent fitness functions.

*6) Post-processing*

The post-processing operator is user-guided; it allows the user to interactively make expert changes to the candidate solution, and to re-run the intensification operator. As such, the termination of the maturation phase is user determined. This also ensures that expert knowledge and intuition are incorporated into the solution procedure. This enhances the interactive search power of the algorithm.

### *C. SM and Related Algorithms*

The proposed algorithm has a number of advantages over related metaheuristics. Contrary to Simulated Annealing (SA) which makes purely random choices to decide the next move, SM employs intelligent selection operation to decide which changes to perform. Furthermore, SM takes advantage of multiple transformations on weak elements of the solution, allowing for more distant changes in successive iterations.

FSM, like Genetic Algorithm (GA), uses the mechanics of evolution as it progresses through generations. GA necessarily keeps a number of candidate solutions in each generation as parents, generating offspring by a crossover operator. Conversely, SM evolves a single solution under hormonal control. In addition, domain specific heuristics are employed to regenerate and repair the emerging solution, developing it into an improved complete solution. Thus, SM reduces the computation time needed to maintain a large population of candidate solutions in GA.

The selection process in the SM is quite different from GA and other related evolutionary algorithms. While GA uses probabilistic selection to retain a set of good solutions from a population of candidate solutions, SM selects and discards inferior elements of a candidate solution according to the goodness of each element, enhancing the computational speed of SM. At the end of the growth phase, the algorithm goes through maturation where intensive search process is performed to refine the solution, possibly obtaining an improved solution. The algorithm allows the decision maker to input expert choices to guide the search process.

The algorithm uses hormonal control to enhance and guide its global multi-criteria search process. This significantly eliminates unnecessary search in regions with inferior solutions. Thus, these advantages provide SM algorithm enhanced convergence that enables it to perform fewer computations than other algorithms.

## IV. FUZZY SIMULATED METAMORPHOSIS FOR NURSE SCHEDULING

### *A. FSM Encoding Scheme*

A unique coding scheme is proposed. Fig. 3 shows an example for 8 nurses to be scheduled into day (*d*), evening (e), night (*n*), and day-off shift (*o*). The coding scheme covers a period of 7 days. The coding allocates nurses one of the four shifts in each day, subject to shift sequence, schedule and roster constraints.

|  | Days |  |  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|---|
| Nurse | 1 | 2 | 3 | 4 | 5 | 6 | 7 | *d* | *n* | *l* |
| Nurse 1 | *n* | *n* | *e* | *e* | *d* | *n* | *e* | 1 | 3 | 3 |
| Nurse 2 | *o* | *d* | *e* | *n* | *d* | *d* | *n* | 3 | 1 | 2 |
| Nurse 3 | *d* | *d* | *d* | *d* | *o* | *e* | *d* | 5 | 1 | 0 |
| Nurse 4 | *e* | *n* | *n* | *o* | *d* | *d* | *d* | 3 | 1 | 2 |
| Nurse 5 | *d* | *d* | *o* | *e* | *e* | *n* | *e* | 2 | 1 | 1 |
| Nurse 6 | *d* | *o* | *d* | *d* | *n* | *e* | *d* | 4 | 1 | 1 |
| Nurse 7 | *n* | *e* | *d* | *d* | *n* | *d* | *o* | 3 | 1 | 2 |
| Nurse 8 | *e* | *e* | *n* | *n* | *e* | *o* | *n* | 0 | 3 | 3 |
| *d* | 3 | 3 | 3 | 3 | 3 | 3 | 3 |  |  |  |
| *e* | 2 | 2 | 2 | 2 | 2 | 2 | 2 |  |  |  |
| *n* | 2 | 2 | 2 | 2 | 2 | 2 | 2 |  |  |  |

Fig. 3. An example of a nurse schedule table

### B. *Initialization Phase*

The initialization algorithm generates a good initial solution, avoiding violation of sequence constraints. Fig. 4 presents the enhanced initialization algorithm that generates an initial shift *s*. Successively, the algorithm generates shift $s_{k+1}$, and tests whether or not the additional sequence is not a subset of forbidden shifts F. An example of a forbidden set is $F = \{nd, ne, ed\}$. In addition, the workload of the current sequence $[s_1 s_2 \ldots s_{k+1}]$ should not exceed the maximum workload $w_{max}$.

*Algorithm* 1. *FSM Initialization Procedure*

```
Initialize, counter $i = 1$;
Repeat
    Initialize $k = 1$
    Randomly generate an initial shift $s_1$
    Repeat
        Select shift $s_{k+1}$ = rand ($d, e, n, o$) with a probability
        If sequence $[s_k s_{k+1}] \not\subset$ Forbidden set $F$, Then
            Add shift $s_{k+1}$ to shift pattern $P_i$
            If workload $w_i$ of sequence $[s_1 s_2 \ldots s_k+1] \geq w_{max}$ Then
                $s_{k+1} = o$
            End If
            Increment counter $k = k+1$
        End If
    Until (Shift Pattern $P_i$ is complete)
    Increment counter $i = i + 1$
Until (Required schedules, $I$, are generated)
Return solution
```

Fig. 4. FSM Initialization Procedure

The initialization algorithm terminates when the required number of the individual nurse schedules *I* are generated.

### C. *Growth Phase*

#### 1) *Evaluation.*

The fitness or quality of a solution is a function of how much it satisfies soft constraints. As such, fitness is a function of the weighted sum of the satisfaction of soft constraints. Thus, each constraint is represented as a normalized fuzzy membership function. Two types of membership functions are used: (a) triangular, and (b) interval-valued functions, as in Fig. 5.

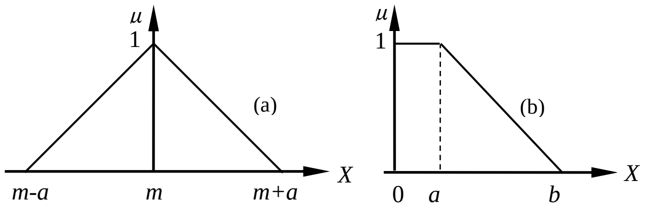

Fig. 5. Linear membership functions

In (a), the satisfaction level is represented by a fuzzy number $A\langle m,a \rangle$, where *m* denotes the centre of the fuzzy parameter with width *a*. Thus, the membership function is,

$$\mu_A(x) = \begin{cases} 1 - \dfrac{|m-x|}{a} & \text{If } m-a \leq x \leq m+a \\ 0 & \text{If otherwise} \end{cases} \tag{4}$$

In (b), the satisfaction level is represented by a decreasing linear function where [0,*a*] is the most desirable range, and *b* is the maximum acceptable. Therefore,

$$\mu_B(x) = \begin{cases} 1 & \text{If } x \leq a \\ (b-x)/(b-a) & \text{If } a \leq x \leq b \\ 0 & \text{If otherwise} \end{cases} \tag{5}$$

##### a) *Membership Function 1 - Workload Variation.*

For fair workload assignment, the workload hi for each nurse *i* should be as close as possible to the mean workload *w*. Therefore, the workload variation $x_i = h_i - w$ should be minimized. Assuming symmetrical triangular membership,

$$\mu_1(x_i) = \mu_A(x_i) \tag{6}$$

where, $x_i$ = workload variation for nurse *i* from mean *m* of the fuzzy parameter with width *a*.

##### b) *Membership Function 2 - Allocated Days Off.*

This membership function measures the variation of the allocated days off from the mean. Thus,

$$\mu_2(x_i) = \mu_A(x_i) \tag{7}$$

where, $x_i$ = the actual variation of days off for nurse *i* from the mean *m* of the fuzzy parameter with width *a*.

##### c) *Membership Function 3 - Variation Night Shifts.*

For shift fairness the variation xi of the number of night shifts (shifts *e* and *n*) allocated to each nurse *i* should be as close as possible to the mean allocation *m*, therefore,

$$\mu_3(x_i) = \mu_A(x_i) \tag{8}$$

where, $x_i$ = variation of number of nights shifts allocated to nurse *i* from mean *m* of the fuzzy parameter, with width *a*.

##### d) *Membership Function 4 – Congeniality.*

This membership function measures the compatibility (congeniality) of staff in similar shifts; the higher the congenialities, the higher the schedule quality. In practice, a decision maker sets limits to acceptable uncongenial shifts $x_i$ for each nurse *i*. Therefore,

$$\mu_4(x_i) = \mu_B(x_i) \tag{9}$$

where, $x_i$ = actual number of uncongenial allocations; *a* = the upper limit to the preferred uncongenial shifts; *b* is the maximum uncongenial shifts.

##### e) *Membership Function 5 – Understaffing.*

High quality schedule minimize as much as possible the understaffing for each shift *k*. In practice, the level of understaffing $x_j = \sum u_k$ in each day $j$ should be within acceptable limits. This is represented;

$$\mu_5(x_j) = \mu_A(x_j) \tag{10}$$

where, $x_j$ = staffing variation from mean $m$ of the fuzzy parameter, with width $a$.

### f) Membership Function 6 – Overstaffing.

For high quality schedule, overstaffing ok for each shift k should be minimized as much as possible. In a practical setting, the level of overstaffing $x_j = \sum o_k$ for all shifts in each day $j$ should be within acceptable limits, which is represented,

$$\mu_6(x_j) = \mu_A(x_j) \tag{11}$$

where, $x_j$ = staffing variation from the mean $m$ of the fuzzy parameter with width $a$.

### g) Membership Function 7 - Forbidden Shift Sequences.

The number of shifts in the forbidden set affects the quality of the schedule for each nurse. If the number of forbidden sequences for each nurse $i$ is $x_i$, then the desirable goal is to reduce the forbidden shifts as much as possible,

$$\mu_6(x_i) = \mu_B(x_i) \tag{12}$$

where, $x_i$ = actual number of forbidden shift sequence; $a$ and $b$ are the fuzzy parameters of the function.

### h) Membership Function 8: Shift Variation.

For each nurse $i$, a schedule with a continuous sequence or block of similar shifts is desirable. For instance, shift [*d d d o o*] with shift variation $x_i = 1$ is more desirable than shift [*d o d o d*] with a variation $x_i = 4$. Therefore,

$$\mu_6(x_i) = \mu_B(x_i) \tag{13}$$

where, $x_i$ = actual number of shift variation; and $a$ and $b$ are the fuzzy parameters of the function.

### i) The Overall Fitness Function.

For each nurse $i$, schedule fitness is obtained from the weighted sum of the first four membership functions. As such, the fitness for each shift pattern (or element) $i$ is;

$$\eta_i = \sum_{z=1}^{4} w_z \mu_z(x_i) \qquad \forall i \tag{14}$$

where, $w_z$ is the weight of each function $\mu_z$, such that condition $\sum w_z = 1.0$ is satisfied. Similarly, the fitness according to shift requirement and congeniality in each day $j$ is given by,

$$\lambda_j = \sum_{z=5}^{6} w_z \mu_z(x_j) \qquad \forall j \tag{15}$$

where, $w_j$ = weight of each function $\mu_j$, with $\sum w_z = 1.0$. The overall fitness of the candidate solution is,

$$f = \left(\frac{\eta}{\omega_1} \wedge 1\right) \wedge \left(\frac{\lambda}{\omega_2} \wedge 1\right) \tag{16}$$

where, $\eta = \eta_1 \wedge \eta_2 \wedge \ldots \wedge \eta_I$; $\lambda = \lambda_1 \wedge \lambda_2 \wedge \ldots \wedge \lambda_J$; $\omega_1$ and $\omega_2$ are weights associated with $\eta$ and $\lambda$, respectively.

The weights $w_z$, $w_j$, $\omega_1$ and $\omega_2$ offer the decision maker an opportunity to incorporate expert choices.

### 2) Transformation.

In NSP, elements are two-fold: one that represents horizontal shift patterns, denoted by $e_i$, and another representing the vertical shift allocations for each day, denoted by $e_j$. Fitness $\eta_i$ and $\lambda_j$ of each element are probabilistically tested for transformation by comparing with a random number $p_t \in [0,1]$, generated at each iteration $t$. A decaying transformation probability limit $pt = p_0 e^{-t/T}$ is used.

*Algorithm 2: Column-wise transformation heuristic*

```
Initialize iteration $t = 1$;
    While ($t \le t_{max}$) do
        While (termination condition) do
            With probability $p_c = \min[1 - \lambda, p_t]$;
            Randomly select $c_1$ = cell with conflict;
            Randomly select $c_2$ = cell with conflict, same column;
            Swap ($c_1$, $c_2$);
        Select the best from neighbourhood;
        End While
        $t = t + 1$;
    End While
```

Fig. 6. Pseudo-code for column-wise transformation heuristic

*Algorithm 3: Row-wise transformation heuristic*

```
Initialize iteration $t = 1$;
    While ($t \le t_{max}$) do
        While (termination condition) do
            With probability $p_r = \min[1 - \lambda, p_t]$;
            Randomly select $r_1$ = cell with conflict;
            Randomly select $r_2$ = cell with conflict, in same row;
                Swap ($r_1$, $r_2$);
        Select the best from neighbourhood;
        End While
        $t = t + 1$;
    End While
```

Fig. 7. Pseudo-code for row-wise transformation heuristic

The column-wise heuristic searches for improved shift sequences and schedules in the neighborhood of the current schedule for each nurse. Again, the dynamic transformation probability pt is used to control the transformation process.

The row-wise transformation heuristic searches for improved roster structure in the neighborhood of the current schedule for each nurse.

### 3) Regeneration.

Regeneration repairs infeasible elements using a mechanism similar to the initialization algorithm which incorporates hard constraints. Based on the juvenile hormone level mt at iteration $t$, the candidate solution is then tested for readiness for maturation,

$$m_t = 1 - (\eta_1 \wedge \eta_2 \wedge \ldots \wedge \eta_I) \wedge (\lambda_1 \wedge \lambda_2 \wedge \ldots \wedge \lambda_J) \tag{17}$$

The growth phase repeats until a pre-defined acceptable dissatisfaction $m_0$ is reached. However, the algorithm proceeds to the maturation phase if there is no significant change $\varepsilon$ in $m_t$, with the value of change ε set in the order of $10^{-6}$.

## D. Maturation Phase

Intensification ensures complete search of a near-optimal solution in the neighbourhood of the current solution. In the post-processing stage the user interactively makes expert changes to the candidate solution, and to execute intensification. Expert knowledge and intuition are coded in form of possible adjustments through weights $w_1,\ldots,w_4$ and $\omega_1$, $\omega_1$. Illustrative computations are presented next.

## V. COMPUTATIONAL ANALYSIS.

The proposed FSM algorithm was coded in JAVA and tested on a 3.06 GHz speed processor, with a 4GB RAM.

### A. *Computational Experiments*

To illustrate the effectiveness of the proposed FSM algorithm, computational experiments were carried out on typical nurse scheduling problems in the literature. Two sets of problem cases were used for the experiments: (i) experiment 1, a preliminary experiment adapted from Jan et al., 2000, (ii) experiment 2 comprising a set of 20 benchmark problem cases in the literature [11], Problem cases in experiment 2 were obtained from real life situations in healthcare organizations reported in [11]. Each experiment includes constraints on shift sequences, length of shift sequences, and length of work and days-off. The number of employees (or groups) for the problems ranges from 7 to 163, to be scheduled over day, evening and night shifts.

The termination criteria are controlled by two conditions: (i) the maximum number of iterations, set at $T_m$ = 300, and (ii) the maximum number of iterations with no improvement, set at $T_I$ = 30. This implies that the algorithm terminates when either of the conditions is met. Generally, each experiment was executed 50 independent times.

### B. *Results and Discussion*

#### 1) *Experiment 1*

The first experimental problem was adapted in [2]. In this problem, there are 15 nurses to be scheduled over a planning horizon of 30 days. In this experiment, the day-off *o* and congeniality preferences were not considered. The initial schedule with this setup is shown in Fig. 8 (a). The fitness values for individual nurses are very low. Fig. 8 (b) shows the final optimal schedule obtained in the preliminary experiments. The overall fitness for the best solution is 1.00, which is desirable to patients, staff and management.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Nurse 1 | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* |
| Nurse 2 | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* |
| Nurse 3 | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* |
| Nurse 4 | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* |
| Nurse 5 | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* |
| Nurse 6 | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* |
| Nurse 7 | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* |
| Nurse 8 | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* |
| Nurse 9 | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* |
| Nurse10 | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* |
| Nurse11 | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* |
| Nurse12 | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* |
| Nurse13 | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* |
| Nurse14 | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* |
| Nurse15 | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* | *e* |

(a)

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | Fitness $\eta_i$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Nurse 1 | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | 1.000 |
| Nurse 2 | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | 1.000 |
| Nurse 3 | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | 1.000 |
| Nurse 4 | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | 1.000 |
| Nurse 5 | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | 1.000 |
| Nurse 6 | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | 1.000 |
| Nurse 7 | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | 1.000 |
| Nurse 8 | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | 1.000 |
| Nurse 9 | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | 1.000 |
| Nurse10 | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | 1.000 |
| Nurse11 | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | 1.000 |
| Nurse12 | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | 1.000 |
| Nurse13 | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | 1.000 |
| Nurse14 | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | 1.000 |
| Nurse15 | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | *n* | *e* | *e* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *d* | *n* | 1.000 |

(b)

Fig. 8. Initial and final nurse schedule for experiment 1

TABLE III. COMPARATIVE PERFORMANCE BASED ON EXPERIEMTN 1

| Approach | Ref. | Best Fitness | Success Rate (%) | CPU Time(s) | Iterations |
|---|---|---|---|---|---|
| Basic CGA | [2] | 1.00 | 8.33 | ** | ** |
| CGA | [2] | 1.00 | 100 | 49.00 | 100 |
| FSM | - | 1.00 | 100 | 32.40 | 40 |

** value not provided

Table III compares the performance of FSM against basic Cooperative Genetic Algorithm (basic CGA) and improved CGA algorithms reported in [2]. Out of 50 independent runs, the success rate of FSM was 100%, which is comparable to 100% for CGA with 12 independent runs. In each successful run, the FSM algorithm was able to obtain the optimal solution in less than 40 iterations, compared to 100 iterations for CGA. The average computational time was 32.40 seconds, indicating that FSM is computationally superior than CGA.

To further demonstrate the performance of FMS, a plot of the intermediate solutions arrived at during the algorithm execution is presented. The overall fitness value *f* is plotted against number of iterations *t*. Fig. 9 shows a plot of the intermediate solutions during the iterative process of the algorithm. The fitness value increased from 0.02 at the initialization stage to 1.00 at the 40th iteration, which implies that the algorithm obtained the optimum solution at the 40th iteration, though the user intended computations up to 300.

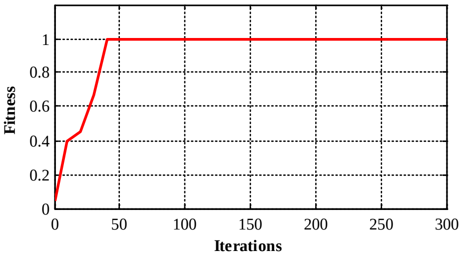

Fig. 9. Illustrative computations based on problem case 1

#### 2) *Experiment 2*

In this experiment, computational results for 20 benchmark problems are reported. For comparative analysis, the success rate and the computational time (CPU time) are taken into consideration. For each problem, 10 independent runs were executed using the FSM algorithm. The maximum number of iterations for each run was $T_m$ = 300.

Table IV provides a summary of the comparative computational results, in terms of search success rate and average CPU time. FSM is compared with min-conflicts heuristic (MC) and MC with tabu search mechanism (MC-T), as well as FSEA. It can be seen that FSM was able to find satisfactory solutions for all the problems, hence 100% mean success rate, even for large scale problems 15, 19 and 20. The success rate of FSM is comparable to MC-T, but is much better than MC and FSEA. In terms of computational efficiency, FSM outperformed all the other algorithms, with a mean time 8.17 sec, compared to 95.70 sec for MC, 20.15 for MC-T and

TABLE IV. COMPARISON BETWEEN FSM AND OTHER ALGORITHMS

| Problem | Success Rate (%) | | | | CPU Time (sec) | | | |
|---|---|---|---|---|---|---|---|---|
| | MC | MC-T | FSEA | FSM | MC | MC-T | FSEA | FSM |
| 1 | 100 | 100 | 100 | 100 | 4.77 | 0.07 | 0.1 | 0.09 |
| 2 | 100 | 100 | 100 | 100 | 1.48 | 0.07 | 0.1 | 0.08 |
| 3 | 100 | 100 | 100 | 100 | 69.36 | 0.42 | 0.18 | 0.14 |
| 4 | 100 | 100 | 100 | 100 | 0.12 | 0.11 | 0.08 | 0.1 |
| 5 | 100 | 100 | 100 | 100 | 15.78 | 0.43 | 0.31 | 0.33 |
| 6 | 100 | 100 | 100 | 100 | 2.89 | 0.08 | 0.09 | 0.07 |
| 7 | 100 | 100 | 100 | 100 | 62.51 | 52.79 | 4.38 | 3.16 |
| 8 | 100 | 100 | 100 | 100 | 32.52 | 0.74 | 0.88 | 0.73 |
| 9 | 50 | 100 | 100 | 100 | 84.17 | 15.96 | 4.87 | 2.14 |
| 10 | 100 | 100 | 100 | 100 | 11.40 | 0.60 | 0.78 | 0.66 |
| 11 | 10 | 100 | 100 | 100 | 254.82 | 13.15 | 10.3 | 7.12 |
| 12 | 100 | 100 | 100 | 100 | 74.26 | 1.17 | 5.33 | 3.27 |
| 13 | 100 | 100 | 100 | 100 | 68.32 | 0.87 | 2.34 | 1.2 |
| 14 | 100 | 100 | 100 | 100 | 8.77 | 0.76 | 2.85 | 1.95 |
| 15 | 15 | 100 | 80 | 100 | 331.11 | 159.04 | 46.34 | 33.12 |
| 16 | 100 | 100 | 100 | 100 | 14.48 | 0.54 | 3.15 | 2.19 |
| 17 | 100 | 100 | 100 | 100 | 54.79 | 2.16 | 7.59 | 5.54 |
| 18 | 100 | 100 | 100 | 100 | 60.58 | 6.83 | 8.35 | 8.13 |
| 19 | 70 | 100 | 100 | 100 | 577.96 | 75.83 | 72.62 | 62.2 |
| 20 | 100 | 100 | 100 | 100 | 183.82 | 71.38 | 27.78 | 31.22 |
| Mean | 87.25 | 100.00 | 99.00 | 100.00 | 95.70 | 20.15 | 9.92 | 8.17 |

9.92 for FSEA. From these comparative analyses, it can be seen that FMS can produce good solutions satisfying patient, staff, and management expectations and prefences.

## VI. CONCLUSIONS

This paper presented a fuzzy simulated metamorphosis algorithm, based on the concepts of biological evolution in insects (e.g., moths and beetles). The algorithm is motivated by the need to solve multi-objective optimization problems with fuzzy conflicting goals and constraints. It mimics the hormone controlled evolution process going through initialization, iterative growth loop, and finally maturation loop.

The suggested method offers a practical approach to optimizing fuzzy multi-objective problems such as the nurse rostering, homecare nurse scheduling, vehicle routing, job shop scheduling, and task assignment. Equipped with the facility to incorporate the user's choices and wishes, the algorithm offers an interactive approach that can accommodate the decision maker's expert intuition and experience, which is otherwise impossible with other optimization algorithms.

FSM is an invaluable addition to the operations research and management community, specifically to researchers concerned with multi-objective global optimization. Learning from the preliminary experimental tests of the algorithm, the application of the proposed approach can be extended to a number of practical hard problems such as task assignment, vehicle routing, home healthcare nurse scheduling, job sequencing, and time tabling, and other industrial problems.

## ACKNOWLEDGMENT

The authors appreciate the constructive comments by anonymous reviews.

## REFERENCES

[1] S. Topaloglu, S. Selim, "Nurse scheduling using fuzzy modeling approach," Fuzzy Sets and Systems, vol. 161, 2010, pp. 1543–1563.

[2] A. Jan, M. Yamamoto, A. Ohuchi, "Evolutionary algorithms for nurse scheduling problem," IEEE Proceedings of the 2000 Congress on Evolutionary Computation, vol. 1, pp. 196 – 203, July 2000.

[3] P. Vasant, Meta-heuristics optimization algorithms in engineering, business, economic, and finance. IGI Publisher, 2012.

[4] P. Vasant, I. Elamvazuthi, T. Ganesan, J.F: Webb, Iterative fuzzy optimization approach for crude oil refinery industry. Scientific Annals of Computer Science 8(2), 2010, pp. 261–280.

[5] M. Mutingi, C. Mbohwa, "Healthcare staff scheduling in a fuzzy environment: A fuzzy genetic algorithm approach," Proceedings of the 2014 International Conference on Industrial Engineering and Operations Management, Bali, Indonesia, pp. 303-312, January 7 – 9, 2014.

[6] T. F. Inoue, T. H. Maeda, M. Takaba, "A proposal of combined method of evolutionary algorithm and heuristics for nurse scheduling support system," IEEE Trans on Ind Elect, vol.50, no.5, 2003, pp. 833-838.

[7] M. Mutingi, C. Mbohwa, "A fuzzy genetic algorithm for healthcare staff scheduling," International Conference on Law, Entrepreneurship and Industrial Engineering (ICLEIE'2013), 2013, pp. 239-243.

[8] S. Shaffer, "A rule-based expert system for automated staff scheduling. IEEE International Conference on Systems, Man, and Cybernetics, Decision Aiding for Complex Systems, vol.3, 1991, pp. 1691-1696.

[9] Burke, E. Causmaecker, P. Berghe V. G. Landeghem, H. "The state of the art of nurse rostering," Jour. of Scheduling, vol. 7, 2004, pp.441-499.

[10] M. Cheng, H.I Ozaku,. N. Kuwahara, K. Kogure, J. Ota, "Nursing care scheduling problem: Analysis of staffing levels," IEEE Proceedings of the 2007 International Conference on Robotics and Biomimetics, vol. 1, December 2007, pp. 1715-1719

[11] N. Musliu,"Heuristic methods for automated rotating workforce scheduling". International Journal of Computational Intelligence Research, 2 (4), 2006, pp. 309-326

[12] B. Cheang, H. Li, A. Lim, and B. Rodrigues, "Nurse rostering problems–a bibliographic survey," European Journal of Operational Research, vol. 151, pp. 447–460, 2003.

[13] J.W. Truman, and L.M. Riddiford, "Endocrine insights into the evolution of metamorphosis in insects," The Annual Review of Entomolog, vol. 47, pp. 467–500, 2002.

[14] G. Tufte, "Metamorphosis and Artificial Development: An Abstract Approach to Functionality," In G. Kampis, I. Karsai, and E. Szathm´ary (Eds.): ECAL 2009, Part I, LNCS 5777, Springer-Verlag Berlin Heidelberg, pp. 83–90, 2011.

[15] C.D. Tarantilis, C.T. Kiranoudis, V.S.A. Vassiliadis, "A threshold accepting metaheuristic for the heterogeneous fixed fleet vehicle routing problem," European J. of Op. Research, vol. 152, pp. 148–158, 2004.

[16] T. Ganesan, I. Elamvazuthi, K.Z.K. Shaari, P. Vasant, An Algorithmic Framework for Multiobjective Optimization, The Scientific World Journal, Vol 2013 , (2013).

## BIOGRAPHY

MICHAEL MUTINGI is with the School of Engineering at the Namibia University of Science & Technology, Namibia. He is a doctoral student with the University of Johannesburg, South Africa. He has been a Research Associate at the National University of Singapore, and a Lecturer in Industrial Engineering at the National University of Science & Technology, Zimbabwe. Michael Mutingi is a member of the Southern African Institution of Industrial Engineers (RSA), and the System Dynamics Society (USA). His research interests are in healthcare operations management, engineering management, and manufacturing systems. He has published more than 70 international

CHARLES MBOHWA is a Professor at the University of Johannesburg. He has a DEng from Tokyo Metropolitan Institute of Technology, masters in operations management and manufacturing systems from the University of Nottingham and a BSc (honours) in Mechanical Engineering from the University of Zimbabwe. He has been a British Council Scholar, Japan Foundation Fellow, a Heiwa Nakajima Fellow, a Kubota Foundation Fellow and a Fulbright Fellow. His research interests are in operations management, engineering management, energy systems and sustainability assessment. He has published a book, several book chapters and more than 150 articles.