2015 IEEE Symposium Series on Computational Intelligence

Enhancing State-of-the-art Multi-objective Optimization Algorithms by Applying Domain Specific Operators

Seyyedeh Newsha Ghoreishi*, Jan Corfixen Sørensen* and Bo Nørregaard Jørgensen*
* Center for Energy Informatics
University of Southern Denmark
Campusvej 55, 5230, Odense, Denmark
Emails: {ng,jcs,bnj}@mmmi.sdu.dk

***Abstract*—To solve dynamic multi-optimization problems, optimization algorithms are required to converge quickly in response to changes in the environment without reducing the diversity of the found solutions. Most Multi-Objective Evolutionary Algorithms (MOEAs) are designed to solve static multi-objective optimization problems where the environment does not change dynamically. For that reason, the requirement for convergence in static optimization problems is not as time-critical as for dynamic optimization problems. Most MOEAs use generic variables and operators that scale to static multi-objective optimization problems. Problems emerge when the algorithms can not converge fast enough, due to scalability issues introduced by using too generic operators. This paper presents an evolutionary algorithm CONTROLEUM-GA that uses domain specific variables and operators to solve a real dynamic greenhouse climate control problem. The domain specific operators only encode existing knowledge about the environment. A comprehensive comparative study is provided to evaluate the results of applying the CONTROLEUM-GA compared to NSGAII, ε-NSGAII and ε-MOEA. Experimental results demonstrate clear improvements in convergence time without compromising the quality of the found solutions compared to other state-of-art algorithms.**

I. INTRODUCTION

MOEAs have been applied in a diversity of different scientific fields spanning from biology to complex control systems. Complex systems are pervasive and address a challenging class of problems in terms of multi-objective optimization [1]. Dynamic optimization problems have complex fitness landscapes that change over time [2]. The algorithms have to converge rapidly and continuously, when an environmental change emerge. Such characteristics make the real-world dynamic control optimization problems hard to solve compared to static optimization problems [2] . An obstacle for MOEAs is to generate and evaluate all possible solutions in computational tractable time and to find feasible solutions for dynamic problems. The inherent design of MOEAs requires a large number of generations and eventually a large number of function evaluations. A too generic representation of the decision variables affects the probability of generating improvement and lead to a larger search space [3].

In this paper, we address dynamic control optimization problems for systems that have fitness landscapes that change dynamically over time. It is the hypothesis of this work, that it's possible to find diverse Pareto optimal solutions in a dynamic environment in tractable time by introducing domain specific operators into a MOEA.

Research in MOEAs, tends to focus on new algorithms that can provide potential solutions that support Pareto optimality for static problems. This work presents a MOEA CONTROLEUM-GA enhanced by domain specific operators and variables to solve a domain specific dynamic greenhouse climate control problem modelled by Jørgensen et al. [4].

The greenhouse climate control problem is characterized by nonlinearity, stochasticity, non-convexity, high dimension of decision variables and an uncertain dynamic environment. Together, these mathematical properties motivate us to use a MOEA for discovering and exploiting critical trade-offs when optimizing the greenhouse climate [5]. An example of a critical trade-off, is reducing supplemental light energy consumption without compromising the quality and growth of the ornamental plants.

Comparisons between CONTROLEUM-GA, NSGAII, ε-NSGAII and ε-MOEA are performed using performance indicators, to understand the effect of domain specific operators and to study the quality of the final solution sets [6], [7]. The result of executing one algorithm is called the solution set. In an experimental setting, solution sets obtained by executing each of the algorithms over a number of settings and a number of seeds is called approximation sets. All the found approximations sets are merged into a combined reference set. Theoretically, an ideal or near-ideal algorithm should be able to support proximity, diversity and consistency. By definition, proximity of an algorithm indicates the adjacency of the final solution set towards the combined reference set found by all algorithms. Diversity indicates the variation of solutions provided by each algorithm compared to the the variation of solutions from the combined reference set. Lastly, consistency expresses the ability of the algorithm to discover all regions of the ideal solution space in different system states.

The paper is organised as follows. First, a brief overview of the work is presented in Section I. Section II describes the problem domain of the case study, the objectives and other system properties. The domain specific operators for the CONTROLEUM-GA are presented in Section III. Section V provides a detailed description of the experimental setup, such as the algorithm properties, the performance indicators and the

978-1-4799-7560-0/15 $31.00 © 2015 IEEE
DOI 10.1109/SSCI.2015.129

TABLE I: The greenhouse climate control subsystems.

Subsystem	Description
Lighting	Activates artificial light when natural light is not sufficient.
Heating	Maintains the sufficient level of temperature.
CO_2	Supplies the required level of CO_2 to support a desired photosynthesis rate.
Ventilation	Open and closes the windows to achieve the required level of temperature and humidity.
Screen	Shades to protect the plants in summer time and to isolate the greenhouse in winter time.
Irrigation	Provides water to achieve the required level for the plant growth.

TABLE II: The greenhouse climate control objectives and constraints for an optimal light plan.

Name	Type	Description
PARSumBalance	Objective	Ensures that plants obtain an optimal photosynthesis by achieving sufficient supplementary light
FixedLightPlan	Objective	Ensures that the lamps are lit in fixed light time intervals
CheapLightPlan	Objective	Ensures the minimum price for the light plan decision variable based on predicted electricity prices
PosPARSum	Constraint	Ensures a positive value for the light sum balance objective (PARSumBalance)
LightInterval	Constraint	Ensures the same light switch status during a specific light time interval

settings of greenhouse climate control system. Experimental results and evaluation of performance indicators are described in Section VI. Section VII presents the findings of the research and provides a discussion about the quality of the solutions. Finally, conclusions and future research are written in Section VIII.

II. CASE STUDY

A greenhouse climate control system uses sensors and actuators to control the greenhouse climate automatically [8]. Input parameters of the control system is read by sensors and the climate control outputs (decision variables) influence the physical environment through the actuators. Input parameters include indoor and outdoor temperature, CO_2, light, humidity, utility prices, weather forecast data etc. The outputs represent set-point values for controlling heating, lighting, CO_2, ventilation, screen and irrigation subsystems [8]. The greenhouse climate control subsystems are described in Table I.

A real control scenario is used based on the input values, control subsystems and the greenhouse control strategy. The case study includes three objectives and two constraints, formulated by domain experts, to optimize a light plan for a greenhouse. The description of the objectives and the constraints is presented in Table II. The MOEAs are responsible for finding solutions that fulfills the objectives and constraints.

For each objective or constraint, input values are transferred from the sensors into input variables, that are used in the formulations.

III. APPROACH

The CONTROLEUM-GA algorithm has two arguments: 1) a time-stamp *time* for when the algorithm is executed, and 2) the population *oldPop* from previous executions. The time-stamp *time* is used for dynamic optimization problems that use the start time of the optimization. The function III describes the overall pseudo-code logic of the algorithm. In Line 1 to 5 the population *pop* is initialized. A population consist of a number of non-dominated Pareto optimal solutions. Each solution is represented by a data-structure, consisting of objective values $solution.objectives$ and decision variables $solution.variables$. A solution can have multiple different types of domain specific variables; e.g., temperature, CO_2 and light plan. Line 1 checks if the previous population *oldPop* is empty. The population *oldPop* is empty, the first time the algorithm is executed. If the population *oldPop* exists from previous executions, it is copied into the new non-dominated population *pop* (Line 3). The population *pop* is evolved in Line 11-31 also called a generation. A number of generations, terminated by a termination criteria, is called an epoch. The different phases of the evolution is described in subsection III-A to III-E.

```
CONTROLEUM-GA(time, oldPop)
 1  if oldPop.isNotEmpty
 2      for each oldSolution ∈ oldPop
 3          ADD-NONDOMSOLUTION(COPY(time, solution))
 4  for i = 0 to POPSIZE
 5      ADD-NONDOMSOLUTION(D-INIT(time))
11  while isNotTerminated
12      for i = 0 to i ≤ POPSIZE
13          if RANDOM-DOUBLE() < MUTATIONRATE
14              child = S-MUTATE(RANDOM(pop))
17          else
18              child =
19              S-CROSSOVER(RANDOM(pop),
20                          RANDOM(pop))
31          ADD-NONDOMSOLUTION(child)
```

A. Initialization

In Line 5, the initial population *pop* is generated based on domain specific initialization operators D-INIT. Each implementation of the domain specific initialization operators is called from D-INIT. That is, a domain specific initialization operator is implemented for each type of decision variable and each of the operators is called in D-INIT (Line 6). For example, the supplemental light plan is initialized by the following domain initialization operator D-INIT-LIGHT that is called from D-INIT.

```
D-INIT-LIGHT(time)
 6  lp = NEW-LIGHTPLAN(time, END-OF-LIGHTPLAN(time),
 7       TIMESLOTINTERVAL)
 8  for i = 0 to i < lp.size
 9      lp[i] = RANDOM-BOOLEAN()
10  return lp
```

In Line 6, a new light plan is generated based on the time-stamp *time* and the interval TIMESLOTINTERVAL, as the light plan changes dynamically for each epoch. The variable

TIMESLOTINTERVAL can be configured by the user to change the on/off intervals of the light plan. The variable END-OF-LIGHTPLAN determines the end of the light plan. A light plan is expressed as an array of ON/OFF light states over a time period, indexed by the time slots. Line 8-10 initialize a random light plan to represent the ON/OFF lamp property. The frequency of which a lamp may be turned on and off, within a given time period, depends on the type of lamp. For example, a LED lamp can be lit within short time intervals. On the contrary, a metal halide lamp requires a cooling phase before it can be lit again, if its life expectancy is not to be reduced. These domain specific properties are incorporated into the LIGHTPLAN-INIT operator to ensure initialization of variable candidate solutions.

B. Ranking

Line 3, 5 and 31 in CONTROLEUM-GA, calls the function ADD-NONDOMSOLUTION. The function ADD-NONDOMSOLUTION sorts all solutions in the population *pop* according to the Pareto dominance relation (Line 34). That is, the objectives are ranked given the proposed decision variables *solution.variables*. The results of the evaluations are assigned to the objective values *solution.objectives* for each proposed solution. Only non-dominated solutions are added to the population *pop*.

```
ADD-NONDOMSOLUTION(newSolutionA)
32  for each oldSolutionB ∈ pop
33      flag =
34          PARETO-COMPARE(newSolutionA, oldSolutionB)
35      if flag == ADOMINATESB
36          REMOVE(oldSolutionB, pop)
37      elseif flag == BDOMINATESA
38          return false
39      elseif DISTANCE(newSolutionA, oldSolutionB) < EPS
40          return false
41  ADD(newSolutionA, pop)
42  return true
```

Line 34 compares if a solution *newSolutionA* dominates a solution *oldSolutionB* or visa verse. If solution *newSolutionA* dominates solution *oldSolutionB* then solution *oldSolutionB* is removed from the population *pop* (Line 36). Contrary, if solution *oldSolutionB* dominates solution *newSolutionA* then it is not added to the population *pop* (Line 38). Two solutions are defined as the same, if the Euclidean DISTANCE between two solutions, in the objective space, is less than the level of significance defined by constant EPS. In case *newSolutionA* is the same as *oldSolutionB* then it is not added to the population *pop* (Line 40).

C. Mutation

For each generation, solutions are randomly selected a number of times for mutation. The number of mutations is determined by the constants MUTATIONRATE and POPSIZE. For example, if POPSIZE is 100 and MUTATIONRATE is 50% then 50 randomly selected solutions are mutated. If a mutation results in a non-dominated solution then it is added to the population *pop*.

```
S-MUTATE(solution)
15  i = RANDOM-INT()
16  solution.variables[i] = D-MUTATE(solution.variables[i])
```

Mutation is applied at solution level and domain variable level. At solution level a random decision variable is selected for mutation in Line 15 using a generic uniform mutation (UM) operator. Each decision variable has its own domain specific mutation operator D-MUTATE. The D-MUTATE operator is applied on the randomly selected variable in Line 16. D-MUTATE-LIGHT shows the implementation of D-MUTATE for a light plan variable. The selected light plan variable is copied and a randomly selected index in the plan is negated. That is, if the light state for the selected index was ON, then after mutation it will be set to OFF.

```
D-MUTATE-LIGHT(lightPlan)
17  lp = COPY(lightPlan)
18  i = RANDOM-INT(lp.size)
19  lp[i] = ¬lp[i]
20  return lp
```

Implementations of D-MUTATION operators incorporate domain knowledge to ensure that the values of the decision variables are always viable. For example, temperature and CO_2 variables can operate only within well-defined ranges as they are constrained by a certain amount of inertia per unit of time. In case of the light plan variable, the D-MUTATION incorporate knowledge about which index of the light plan that is viable for change (Line 18). For example, if a light interval only can change once for a given period of time, or if a light state is always fixed, then it is implemented in the implementation of the D-MUTATE function for the given variable. In case of greenhouse climate control, several domain mutation operators are formulated based on the specification sheets for each actuator. Each domain mutation operator defines a range for a specific type of decision variable (temperature, energy, CO_2, etc.). The intersection of these ranges defines the viability space of the decision variable.

D. Crossover

Solutions are randomly selected for crossover for a number of iterations. The number of crossover iterations is determined by the constant POPSIZE. Crossover is applied at solution and decision variable level. The solution level crossover function S-CROSSOVER is called in Line 20, see CONTROLEUM-GA. Random variables from two solutions *solutionA* and *solutionB* are selected for crossover at decision variable level using a generic one-point crossover operator (Line 21-23). At the other variables from the selected index $i+1$ till last index from *solutionB* is copied to *solutionA*.

```
S-CROSSOVER(solutionA, solutionB)
21  i = RANDOM-INT(solutionA.variables.size)
22  solutionA.variables[i] =
23  D-CROSSOVER(solutionA.variables[i],
                solutionB.variables[i])
28  for j = i+1 to j < solutionA.variables.size − 1
29      solutionA.variables[j] = solutionB.variables[j]
30  return solutionA
```

The domain specific D-CROSSOVER operator is applied on the selected decision variables in Line 23. The function D-CROSSOVER-LIGHT is the domain specific crossover operator for two light plans *lightPlanA* and *lightPlanB*. The two light plans are crossed by a one-point crossover operator but can also include knowledge about the light plan domain. For example, if the domain property of the lamps demands that light has to be lit for a continues number of time-slots when first lit, then this knowledge is implemented in the provided domain specific operator.

D-CROSSOVER-LIGHT($lightPlanA, lightPlanB$)
24 $child = \text{COPY}(lightPlanA)$
25 **for** $i = \text{RANDOM-INT}(child.size)$ **to** $i < child.size$
26 $child[i] = lightPlanB[i]$
27 **return** *child*

E. Termination

In CONTROLEUM-GA, Line 11 test if the evolution should terminate. Evolution is terminated after a specified time limit, after a number of generations or when the population is stable.

IV. BASELINE ALGORITHMS

The following paragraphs highlight the most significant differences between the baseline algorithms.

NSGAII: NSGAII is a second generation classic example of a Pareto-based MOEA, introduced by Deb et al., that incorporates improvements compared to the original algorithm NSGA [9]. NSGAII is historically used as a benchmark MOEA together with its ancestor NSGA. NSGAII can perform at least as well as or better than other second generation MOEAs when solving difficult multi-objective optimization problems [10], [11], [12]. It is the first MOEA to use the Pareto dominance relation to search for the Pareto front in a single run [13]. NSGAII has four significant differences compared to the original NSGA: 1) less computational complexity, 2) eliminate the need of parameter sharing, 3) support elitism by using a selection operator that is able to combine the parent and offspring populations and select the best N solutions and, 4) support real and binary decision variables for output representation.

ε-NSGAII: The ε-NSGAII is an extension of NSGAII that is improved to make the original NSGAII more efficient, reliable and easy-to-use. ε-dominance archiving, dynamic population sizing, randomized restart and automatic termination are the most significant changes in the ε-NSGAII [13].

ε-MOEA: The ε-MOEA uses ε-dominance parameterization. This property enables the user to define desired objective precision and find more diverse solutions and maintain a fixed-size population [13]. It is worthy to mention that, the dynamic size for the ε-dominance archive allows the algorithm to grow and shrink as needed. Furthermore, the ε-MOEA exploits efficient parent and archive update strategies. Laumanns et al. claimed that ε-MOEA is able to achieve well-distributed solution sets, in appropriate time, which support both diversity and convergence [14]. In other research done by Deb et al., it is indicated that ε-MOEA performs as well as, or better than other second generation MOEAs in terms of convergence, diversity and computational time [12].

V. EXPERIMENTAL SETUP

To measure the performance of the MOEAs, the experiments are based on the standard procedure introduced by Coello. The indicators are calculated based on Pareto-optimal solutions obtain by running each algorithm. A brief description of each performance indicator is explained in the following sections. For more details and mathematical formulation of each indicator see [15].

A. Performance Indicators

Hypervolume: The hypervolume indicator measures the area of the solution set, found by the MOEA, dominated by the reference set. It evaluates proximity and diversity.

Generational Distance: The generational distance indicator calculates the average distance between the MOEA solution set and the nearest solution from the reference set. This indicator evaluates proximity and coverage of the found solution set.

Additive ε-indicator: The additive ε-indicator calculates the additive minimum distance in which the solution set cover every solution in the reference set. This indicator measures the consistency of the found solutions.

Maximum Pareto Front Error: The maximum Pareto front error indicator measures the maximum error band between found solutions and every solution in the reference set. This indicator evaluates the coverage of the solutions set found by the MOEA.

Spacing: The spacing indicator calculates the spread of the solutions. This indicator is not related to the reference set and considers the solutions in the found solution set independently. The spacing indicator measures the diversity of the solutions.

Contribution: The contribution indicator calculates the percentage of solutions from solution set that is part of the reference set. The contribution indicator is used to evaluate whether the MOEA supports optimality or not.

Convergence Time: Convergence time is the time needed for the MOEA to converge to the reference set. In an ideal situation, convergence can happen with maximum contribution to the reference set. That is, all solutions found by the MOEA are included in the reference set. This indicator is important when evaluating MOEAs that perform equally well.

B. The Greenhouse Climate Control Strategy Setup

The greenhouse climate control system can have different settings to achieve different goals. For example, a solution may has of a set of decision variables (outputs) like periodic light plan, CO_2 plan or temperature plan. To achieve an optimal value for a decision variable, the MOEA has evaluate the objectives and constraints for each generated solution. For each epoch, a number of input values need to be read by sensors or extracted from external databases or data services. In this work, a real control scenario is used for the experiments. The list of objectives and constraints used in the experiments are presented in Table II in Section II. The input parameters, objectives and constraints are presented in Table III.

TABLE III: The greenhouse climate control input parameters, objectives and constraints.

Name	Variables	Formulation
PARSumBalance	**TAP:** Total achieved photosynthetically active radiation (PAR) sum **FNP:** Forecasted natural and supplemental light for a specific period **TPL:** Total PAR sum gained by the proposed light plan **DLI:** The PAR day light integral	$PARSumBal = min((TAP+FNP+TPL)-DLI)$
PosPARSum	PARSumBal	$PosPARSum = \begin{cases} true & if\ PARSumBal \geq 0 \\ false & if\ PARSumBal < 0 \end{cases}$
FixedLightPlan	Proposed light plan Fixed light plan	$mismatchRate = min(\frac{NoMismatchedSwitchStatus}{NoTotalRequiredSwitchedON})$
CheapLightPlan	Proposed light plan **EP:** Electricity price **TLL:** Total lamp load	$CheapLightPlan = min(\sum_{i=1}^{n} Switch_i * EP_i * TLL * t_i)$
LightInterval	Light switch status	Checks if the light switch status remain in the same state during a control cycle

TABLE IV: Parameters used for the baseline MOEAs.

Parameter	Description	Possible Values
Max. Evaluations	Maximum number of evaluation	2000 - 6000
Population Size	The size of the population	10 - 100
sbx.Rate	The crossover rate for simulated binary crossover	0.0 - 1.0
sbx.distributionIndex	The distribution index for simulated binary crossover	0.0 - 500
1x.rate	The crossover rate for single-point crossover (for binary encoding)	0.0 - 1.0
bf.rate	The mutation rate for bit-flip mutation (for binary encoding)	0.0 - 500
epsilon	The values used by the ε-dominance archive	0.0 - 1.0
injectionRate	Controls the percentage of the population injects for restarting the evaluations	0.1 - 1.0

C. MOEAs Parameters Setup

Since the MOEAs are initialized with randomly generated populations and the operators are probabilistic, it is crucial to evaluate the algorithms with different configuration settings. Configuration settings for MOEAs can be different for each algorithm and are based on the characteristics of the algorithm. In this work, all MOEAs are genetic-based and have common configuration settings. For simplicity, we provide the list of parameters used for evaluating the MOEAs in Table IV. The parameters are common in all baseline MOEAs, except from the epsilon parameter that is used for the ε-based MOEAs ε-NSGAII and ε-MOEA, Among all these parameters, maximum evaluations, population size, crossover and mutation rate are used in CONTROLEUM-GA.

The parameters specified for the algorithms are described in Table IV. Each algorithm is evaluated with 50 different set of parameter samples and 50 different seeds. These combinations lead to a around 50 to 150 million executions of the algorithms. It is important to mention that, our experiments are not restricted to any specific assumption and the parameters

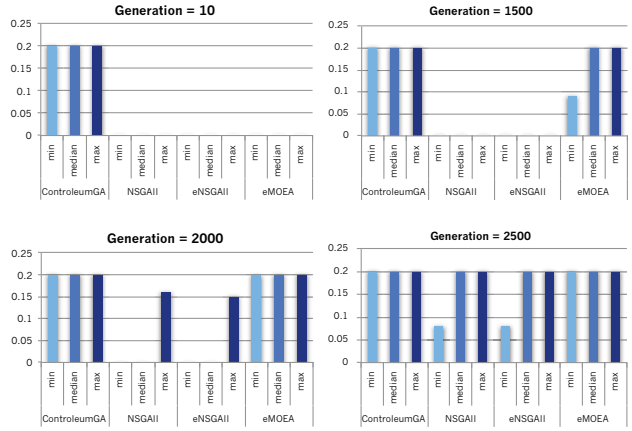

Fig. 1: Comparison of algorithms based on Hypervolume indicator for different generations.

used to evaluate the algorithms are not fixed or tuned. This large number of evaluations with different randomized values for algorithm parameters, is needed to perform a statistical analysis without bias from the specific parameter settings.

D. Application Setup

In order to perform the experiments, we used the MOEAFramework version 2.3 together with a greenhouse climate control application DynGrow [7], [8]. Integrating the MOEAFramework with the climate control application makes it possible to apply the baseline MOEAs algorithms to the multi-objective optimization problem defined in DynGrow. DynGrow is an application that provides all the necessary components for measuring inputs, providing links to external data sources like weather and electricity price forecast. In addition, all objectives, constraints and domain specific decision variables are formulated in DynGrow.

VI. EXPERIMENTAL RESULTS

The reference set is needed to process the results and is generated by evaluating and merging the final approximation sets obtained by executing all the state-of-the-art MOEAs together with CONTROLEUM-GA. To achieve the results, the five performance indicators hypervolume, generational distance, additive ε-indicator, maximum Pareto front error and contribution are evaluated. The experiments are preformed for a different number of generations, to better understand the runtime and the end-of-run behaviour of the algorithms . The maximum number of generations considered is 10, 1500, 2000 and 2500. For each indicator minimum, median and maximum values are measured to evaluate each algorithm. The median is considered instead of the average value, to avoid bias from minimum and maximum values when performing the statistical analysis.

Fig. 1 illustrates the results for the hypervolume indicator for a different number of generations. The figure shows that within first 10 generations, CONTROLEUM-GA is able to gain a large value for hypervolume compared to other baseline algorithms. The approximation sets generated by NSGAII and ε-NSGAII are far from the reference set and has a low

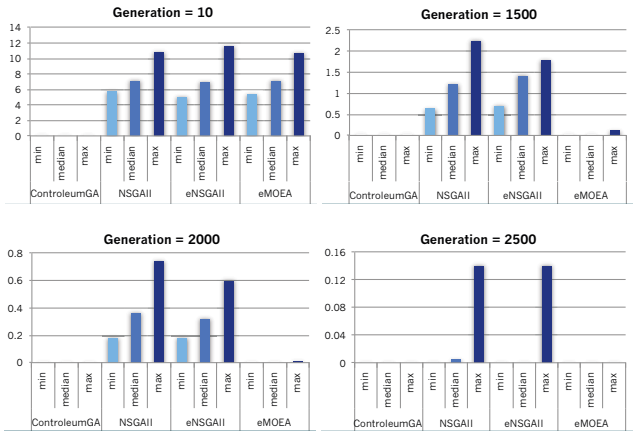

Fig. 2: Comparison of algorithms based on Generational Distance indicator for different generations.

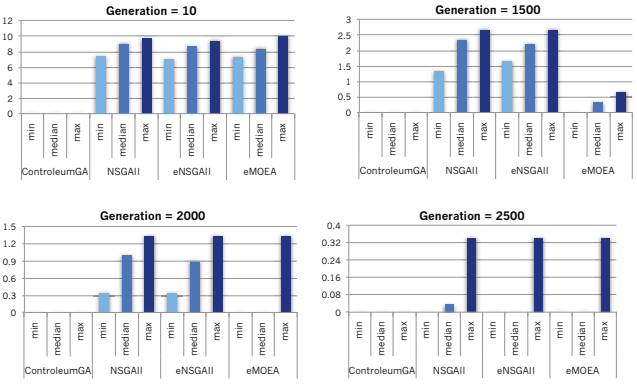

Fig. 3: Comparison of algorithms based on Additive ε-indicator for different generations.

diversity for generations below 2000. ε-MOEA outperforms NSGAII and ε-NSGAII. The approximation sets generated by CONTROLEUM-GA have larger proximity and diversity and results in a large hypervolume in first 10 generations. The baseline algorithms are not able to generate optimal or near-optimal approximation sets in few generations.

The generational distance indicator is shown in Fig. 2. The generational distance indicator measures the proximity of the approximation set towards the reference set. In the first 10 generations, approximation sets are generated by CONTROLEUM-GA that have large proximity while the other three algorithms are generating approximation sets far from the reference set. The three baseline algorithms are able to generate optimal and near-optimal approximation sets after 2500 generations. This means, that the baseline algorithms need to evaluate more populations to obtain approximation sets near to the reference set.

The additive ε-indicator measures the gap between solutions in the approximation set compared to the reference set. Fig. 3 shows that CONTROLEUM-GA outperforms the other three baseline algorithms in 10 generations. Furthermore, the low additive ε-indicator value for CONTROLEUM-GA indicates

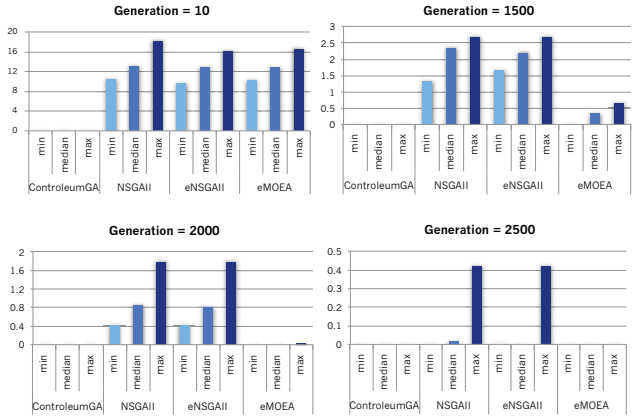

Fig. 4: Comparison of algorithms based on Maximum Pareto Front Error indicator for different generations.

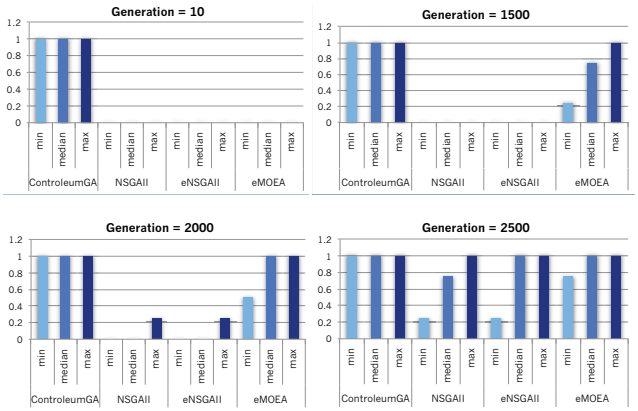

Fig. 5: Comparison of algorithms based on Contribution for different generations.

the consistency of algorithm.

The maximum Pareto front error indicator calculates the maximum error band in which the solutions in the approximation set encompasses every solution in the reference set. For that reason, a low value for maximum Pareto front error indicator is desirable for the algorithms. The results obtained from the statistical analysis, show that the approximation set generated by CONTROLEUM-GA has the minimum value for this indicator, see Fig. 4. The maximum error band between solutions in the generated approximation set towards the reference set is quite small and it becomes zero after 10 generations for CONTROLEUM-GA. The three baseline algorithms underperform compared to our proposed algorithm and have larger maximum Pareto front error after 10 generations.

The Contribution indicator calculates the percentage of solutions from the approximation set that appears in the reference set. That is, the Contribution indicator indicates whether solutions found by a MOEA supports optimality or not. The statistical analysis of the Contribution indicator for different number of generations and for all algorithms is illustrated in Fig. 5. CONTROLEUM-GA is able to contribute to the

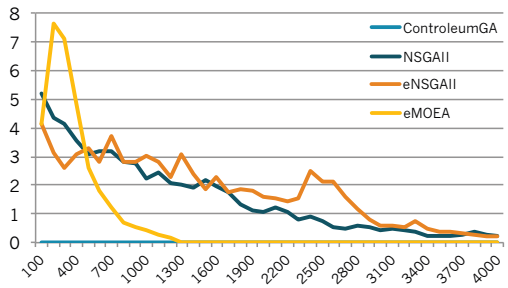

Fig. 6: Generational Distance of MOEAs.

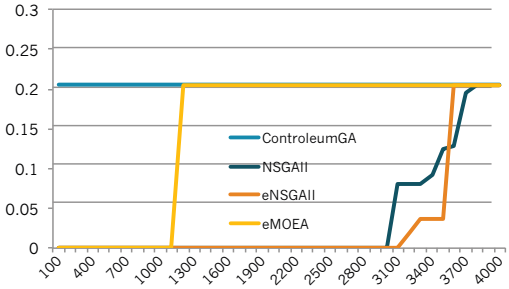

Fig. 7: Hypervolume of MOEAs.

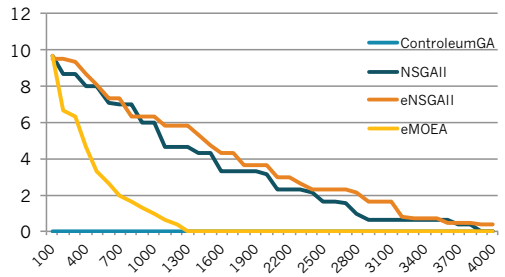

Fig. 8: Additive ε of MOEAs.

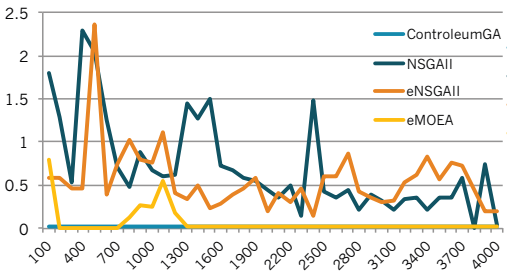

Fig. 9: Spacing of MOEAs.

reference set from very early generations while the other baseline algorithms contribute in around 2500 generations. That is, the proposed algorithm can outperform all the baseline algorithms in regard to finding exact solutions that matches the reference set.

A statistical analysis, considering different maximum number of generations, is performed to observe progress of convergence and run-time behaviour of each algorithm. The observed run-time behaviour of each algorithm is presented and analyzed in Fig. 6 - 10. Five indicators are selected to explain the run-time behaviour of the MOEAs: hypervolume, generational distance, additive ε-indicator, spacing and convergence time. It is possible to justify the quality of the approximation sets generated by the algorithms using a set of indicators instead of only one indicator [15]. Each algorithm was executed with maximum 4000 generations to be sure that all of the algorithms would converge.

Fig. 6 indicates that after a few generations, CONTROLEUM-GA is able to generate an approximation set close to the reference set with a minimum value for generational distance indicator. Next after CONTROLEUM-GA, ε-MOEA can outperform the other two baseline algorithms and obtain an approximation set close to the reference set after 1300 generations.

Fig. 7 shows that CONTROLEUM-GA can reach the largest value for the hypervolume indicator before 100 generations compared to ε-MOEA that reaches this value after 1200 generations. The results indicate that the proposed CONTROLEUM-GA and ε-MOEA outperform NSGAII and ε-NSGAII.

The run-time results of executing algorithms for using the additive ε-indicator is illustrated in Fig. 8. The additive ε-indicator measures the gap between solutions in the approximation set compared to the reference set. It is clear from the data in this figure, that CONTROLEUM-GA outperform the other three baseline algorithms i less than 100 generations. Furthermore, ε-MOEA reaches the minimum value for the additive ε-indicator after 1300 generations while NSGAII and ε-NSGAII obtain the minimum value after 4000 generations.

Fig. 9 indicates that after few generations, CONTROLEUM-GA is able to generate an approximation set with large diversity that remains until the end of generations. The other three baseline algorithms generate very diverse approximation set in different generations and the spread of the solutions vary until the end of generations. Only ε-MOEA reaches a stable unified diversity after 1300 generations.

Fig. 10 illustrates the convergence time for CONTROLEUM-GA and the three baseline MOEAs. Convergence time is the time required to obtain a stable approximation set that is as close as possible to the reference set. Fig. 10 illustrates that CONTROLEUM-GA can perform about 10 generation within 17 seconds to converge to the reference set. NSGAII, ε-NSGAII and ε-MOEA converge after about 3000 generations. ε-NSGAII and ε-MOEA converge in approximately 21 and 22 seconds, respectively. NSGAII is able to converge after 28 seconds.

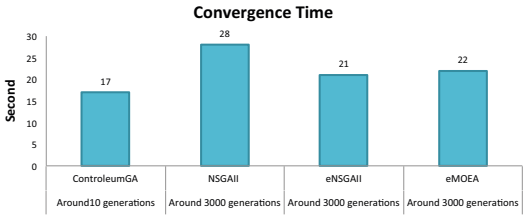

Fig. 10: Convergence time for all of the algorithms.

VII. DISCUSSIONS

The effects of using domain specific operators in a MOEA to solve a dynamic multi-objective optimization problem was investigated. Fast convergence is an important property when solving dynamic optimization problems. Applying domain specific operators and decision variables, reduced the convergence time. CONTROLEUM-GA limits the search space to a domain viable search space by using domain specific operators and the effect is faster convergence. An important finding is that the quality of the solutions found by CONTROLEUM-GA is better or equally as good compared to NSGAII, ε-NSGAII and ε-MOEA. State of the art indicators have been evaluated to support this claim and to justify the quality of the algorithms [15]. According to the results, hypervolume and generational distance indicators show that CONTROLEUM-GA outperforms baseline algorithms in terms of proximity in less than 10 generations. Also, generational distance and maximum Pareto front error indicators indicate that the solutions generated by CONTROLEUM-GA has better coverage compared to NSGAII, ε-NSGAII and ε-MOEA. Furthermore, the additive ε-indicator shows that CONTROLEUM-GA is consistent in generating approximation sets close to the reference set.

Based on the evaluated indicators, CONTROLEUM-GA combined with domain specific operators is able to find high quality solutions in less time. ε-MOEA outperforms the ε-NSGAII and NSGAII respectively. The reason for this is that ε-MOEA is able to achieve a well-distributed approximation set within appropriate computation time and guarantee both diversity and convergence [13]. This is consistent with what Laumanns et al. claimed in [14] about ε-MOEA.

The experimental results indicate that ε-NSGAII outperforms NSGAII in terms of convergence time. This is based on the improvement done in ε-NSGAII compared to NSGAII, to eliminate unnecessary algorithm executions by terminating the search process automatically [10]. Although CONTROLEUM-GA is able to generate high quality solutions in appropriate computation time, the results regarding other baseline algorithms are consistent with earlier studies [6], [10], [13], [14]. Hadka and Reed observed, in their experiments, that ε-dominance archiving used in ε-NSGAII and ε-MOEA improves the search process [7]. This research will serve as a base for future studies to improve MOEAs with more domain specific variables and operators for solving complex dynmaic control problems.

VIII. CONCLUSIONS

It is fair to state that domain specific variables and operators enable the proposed algorithm to converge fast enough to solve the dynamic greenhouse climate control problem. The most significant finding from this research is that the proposed enhanced MOEA CONTROLEUM-GA outperforms the other baseline MOEAs in terms of convergence time without compromising the quality of the solutions in the approximation set. Also, the findings have significant implications for the understanding of how to apply domain specific operators to enhance state-of-the-art MOEAs. The presented study confirms previous findings and contributes with a new MOEA CONTROLEUM-GA, that can solve dynamic multi-optimization problems based on domain specific operators and variables.

More research, based on experiments, is required to assess the efficacy of different domain specific variables and operators. Future experiments should explore: the effect of a large number of objective functions, more complex dynamic control problems, different representations of decision variables and variations of domain specific operators.

REFERENCES

[1] I. Marsaa-Maestre, E. de la Hoz, and M. Klein, "Negotiation techniques for complex system optimization with self-interested parties: the challenges of ever-increasing scale and complexity," *European Network for Social Intelligence*, 2013.

[2] M. Mavrovouniotis, F. Neri, and S. Yang, "An adaptive local search algorithm for real-valued dynamic optimization," *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2015)*, 2015.

[3] G. P. Wagner and L. Altenberg, "Perspective: Complex adaptations and the evolution of evolvability," *Evolution*, vol. 50, no. 3, pp. 967–976, 1996.

[4] B. N. Jørgensen, M. L. R. Jensen, J. C. Sørensen, and O. Korner, "Advanced model-based greenhouse climate control using multi-objective optimization," *Proceedings of the International Symposium on Models for Plant Growth, Environmental Control and Farm Management in Protected Cultivation (HORTIMODEL)*, 2012.

[5] P. Reed, D. Hadka, J. Herman, J. Kasprzyk, and J. Kollat, "Evolutionary multiobjective optimization in water resources: The past, present, and future," *Advances in Water Resources*, vol. 51, pp. 438–456, Jan. 2013.

[6] S. N. Ghoreishi, J. C. Sørensen, and B. N. Jørgensen, "Comparative study of evolutionary multi-objective optimization algorithms for a non-linear greenhouse climate control problem," 2015.

[7] D. Hadka and P. Reed, "Diagnostic assessment of search controls and failure modes in many-objective evolutionary optimization," *Evolutionary Computation*, vol. 20, no. 3, pp. 423–452, 2012.

[8] J. C. Sørensen, B. N. Jørgensen, M. Klein, and Y. Demazeau, "An agent-based extensible climate control system for sustainable greenhouse production," *Agents in Principle, Agents in Practice, 14th International Conference, PRIMA 2011, Wollongong, Australia*, vol. 7047, pp. 218–233, November 2011.

[9] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multi-objective genetic algorithm: Nsga-ii," *IEEE Transaction on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2000.

[10] J. B. Kollat and P. M. Reed, "Comparing state-of-the-art evolutionary multi-objective algorithms for long-term groundwater monitoring design," *Advances in Water Resources*, vol. 29, no. 6, pp. 792–807, 2006.

[11] E. Zitzler, M. Laumanns, and L. Thiele, "Spea2: Improving the strength pareto evolutionary algorithm," *In: Evolutionary methods for design, optimization, and control*, pp. 95–100, 2002.

[12] K. Deb, L. Thiele, E. Zitzler, and M. Laumanns, "Scalable test problems for evolutionary multi-objective optimization," Indian Institute of Technology Kanpur, Tech Rep KanGAL 2001001, 2001.

[13] K. Deb, M. Mohan, and S. Mishra, "A fast multi-objective evolutionary algorithm for finding well-spread pareto-optimal solutions," *KanGAL Report 2003002*, January 2003.

[14] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, "Combining convergence and diversity in evolutionary multiobjective optimization," *Evolutionary Computation*, vol. 10, no. 3, pp. 263–282, 2002.

[15] C. A. Coello, G. B. Lamont, and V. Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer, 2007.