Evolutionary Algorithms in Control Optimization: The Greenhouse Problem

Thiemo Krink
EVALife Group
Dept. of Computer Science
University of Aarhus, Denmark
krink@daimi.au.dk

Rasmus K. Ursem
EVALife Group
Dept. of Computer Science
University of Aarhus, Denmark
ursem@daimi.au.dk

Bogdan Filipič
Dept. of Intelligent Systems
Jožef Stefan Institute
Ljubljana, Slovenia
bogdan.filipic@ijs.si

Abstract

In recent years, evolutionary algorithms (EAs) have been applied to a variety of dynamic optimization problems. In control optimization, dynamic problems are characteristically dominated by the feedback between the controller and the controlled system. Most studies in this field are rather pragmatic and many principal issues in control optimization have not been addressed yet. In this paper, we studied the performance of various control strategies regarding the investment of computation time in number of generations versus population size. Further, we investigated the evaluation of candidate solutions in respect to their changing fitness over time. Our experiments showed that both aspects were significant factors for the optimization performance. As a benchmark control problem, we implemented a simplified model of a crop producing greenhouse, where the objective was to maximize the profit. The fitness landscapes generated with this simulator showed that previously suggested test case generators cannot model realistic control problems.

1 INTRODUCTION

The ultimate goal of an optimization technique is its application to real-world problems. Typically, real-world optimization problems are characterized by constraints, multiple objectives, and dynamic properties. In particular control problems are notoriously dynamic due to the feedback between the controller and the controlled system. Therefore, the applied optimization technique should be able to continuously search for the best solution. Evolutionary algorithms (EAs) and other adaptive search techniques, such as simulated annealing, fulfill this requirement. However, EAs have the additional advantage that they maintain a population of solutions throughout the run rather than just a single solution. Thus, the search for new solutions can take advantage of the diversity of the population and the competition between the individuals. Each available candidate solution offers a starting point for discovering new ways to treat the problem given whatever change has occurred. Therefore, the optimization process does not have to rely on a single starting point only, and new solutions do not have to be recomputed from scratch. For reasonably small changes of the problem, some individuals in the present population are likely to be in close vicinity of optima in the next generation.

There are four main issues to consider regarding optimization in control problems:

First, computation time is a critical factor. In most cases, the optimization technique is applied to a simulator before its solutions are transferred to the real system. Realistic simulators of systems, such as industrial production control or vehicle steering control, are usually complex and evolutionary optimization may take days on state-of-the-art computers. An example is the complex simulation of the temperature field in a slab of continuously cast steel, which has to be computed to evaluate the fitness of each candidate solution for the controller (Filipič and Šarler, 1998). This limits the experimental feasibility regarding population size and number of generations. Furthermore, the time needed for the fitness evaluation of a candidate solution can be a serious limitation regarding real-time applications. In so-called direct optimal control (Fogarty et al., 1995), an EA continuously evolves the settings of the control parameters, i.e., the EA is not tuning or evolving other controllers such as a PID or a fuzzy controller. Such an online-evolution process faces the problem that the longer it takes to compute new solutions the more the fitness landscape can change in real time, which makes the problem increasingly harder. Therefore it is of critical importance to achieve an optimal balance between the number of evaluations and the required computation time as well as between the population size and the number of generations per time-step.

Second, in contrast to dynamic observation problems, in control problems there is feedback between the controller and the system. Each control action consequently affects the shape of the fitness landscape. In other words, the search for the optimal control affects the problem.

Third, control has to be robust. A typical problem in hardware control is the drift of material parameters due to machinery wear-out and sensitivity to environmental changes, such as temperature, light, and humidity (Filipič and Juričić, 1993).

Finally, the pure performance output of the system is often inappropriate to serve as a fitness evaluation, but rather the design of the fitness function itself is of critical importance for the success of the optimization (Filipič et al., 1999).

We suggest three main classes of control problems: (i) state stabilization (e.g. constant electricity supply of a power plant), (ii) system-to-system interaction (e.g. 'arms-race' with another system, such as pest control versus a crop pest), and (iii) profit maximization (e.g. market-oriented production optimization in a factory).

In this paper, we investigated control optimization strategies in profit maximization by the example of a simulated crop producing greenhouse. In this setup, we experimented with different control strategies, which were based on the evaluation of a limited amount of candidate solutions per simulated system time-step.

Our motivation for the design of the greenhouse simulator was that there are hardly any appropriate benchmark tests or test case generators for control problems until today. An exception in state stabilization is the pole-balancing problem (Karr, 1991), which is a well-known standard problem that is relatively easy to implement. Classic benchmark tests in numerical optimization, such as the Rastrigin or De Jong functions, are usually simple and static. In control problems, the demands on the optimization process are very different and performance results from experiments with simple static functions are of little use due to the issues outlined above. Also the recently suggested test case generators (TCGs) for dynamic environments (e.g. (Branke, 1999; Morrison and Jong, 1999; Grefenstette, 1999)) are insufficient to mimic control problems. Their main weakness is that they arbitrarily distort the landscape over time by moving or bouncing peaks. Not surprisingly, it remains unclear how much and in which way these landscape distortions would resemble characteristics of real-world problems. Another major limitation of these TCGs regarding control problems is that they cannot simulate the feedback between the control and the system, i.e., that the search affects the change of the landscape. Moreover, the frequency and degree in which fitness landscapes are changed over time is specified by independent parameters, whereas in real systems the speed of changes between two successive control actions is mainly affected by the time it takes to evaluate the candidate solutions. For reasonable test cases, a realistic interaction between the speed of changes and the evaluation of individuals is essential, since the speed ultimately determines the success of the optimization process, see e.g. (Ursem, 2000).

These limitations of previously suggested TCGs have been recently addressed by a study of two of the authors and other collaborators (Ursem et al., subm). The result of this study was a new test-case simulator language that supports the modeling of dynamic test case problems by simulation of real-world systems. The main idea was to create a tool that allows to model instances of time-varying fitness landscapes by the performance output of a modeled system. As an example, the control problem of a crop producing greenhouse was implemented, which we adopted and refined in this study. In contrast to the earlier paper, our main intention with the present paper was to focus on the principle issue of the design of the control strategy.

The remaining sections of the paper are structured as follows: In Section 2, we present a general design of benchmark control problems. Afterwards, in Section 3, we introduce the greenhouse model with all its components and parameters. Section 4 contains the results of our preliminary experiments with a set of control strategies for production control optimization, and finally, in Section 5, we discuss the results of this study.

2 DESIGN OF BENCHMARK CONTROL PROBLEMS

Control problems in engineering are often represented by the interaction between the controller and the controlled system (Figure 1). Here $\mathbf{x}(t)$ represents the internal state of the system at time t, $\mathbf{u}(t)$ is the control signal, and $\mathbf{y}(t)$ is the output from the system.

Further, it is often necessary to model the environ-

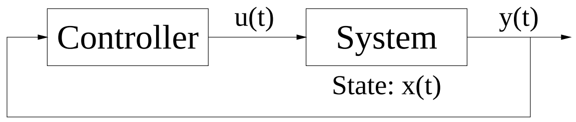

Figure 1: Controller and the system being controlled.

ment that surrounds the system if it, in addition to the controller, affects the system (see Figure 2).

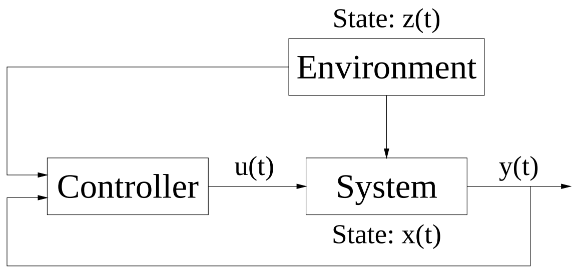

Figure 2: Model for controller, system, and environment.

The representation of a system requires to identify all variables that are either a part of the system or influence it indirectly. Further, the type and range of each variable (control, system, or environment) needs to be determined. It is not always clear where to draw the line between control, system, and environment; however, the following simple rules can be used to categorize variables into one of the three classes.

1. Control variables are variables that can be set by the controller, e.g., outlet of a valve.
2. System variables are directly affected by the control variables, but also through interactions with other variables. For instance, the level in a rainwater tank.
3. Environment variables represent components that influence the system, but are not directly affected by the control variables, e.g., amount of rainfall.

The time-varying system and environment states can often be modeled by difference equations of the form

$$x_i(t+1) = x_i(t) + \Delta x_i(t) \qquad (1)$$

where $\Delta x_i(t)$ usually depends on other variables of the controller, the system, and the environment.

2.1 TECHNICAL ASPECTS

As stated in the introduction, the application of EAs to control problems has the side-effect that the search changes the problem. Hence, the candidate solutions in a population have to be evaluated from the same starting state. Consequently, the entire state of the simulator has to be stored and restored between evaluations. After all candidate solutions within one time-step have been evaluated, one solution, usually the best, controls the system for a number of simulation steps. From this state on the process is repeated. Figure 3 illustrates an abstract scenario where four control settings are evaluated for three time-steps. The best setting is then used to control the system one time-step.

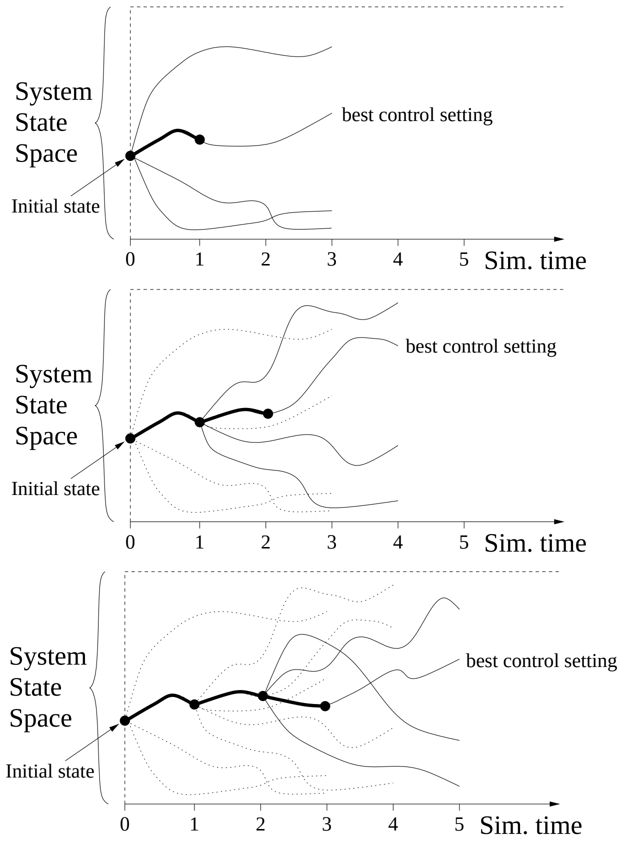

Figure 3: Example of state space exploration at simulation time $t = 0$, $t = 1$, and $t = 2$. Thin lines represent control settings exploration of the current time-step, thin dotted lines are previous explored control strategies, and thick lines are actual control as it was performed by the selected control setting.

3 THE GREENHOUSE MODEL

As a benchmark test for our study, we implemented a simple simulator for a crop producing greenhouse. The production is controlled by heating, injection of CO_2, ventilation, and optional use of artificial light. All produced crops are sold at a time-varying market price.

The objective is to maximize the profit, i.e., to maximize the production while minimizing the expenses of heating, CO_2, and electricity.

The simulator consists of three types of interacting variables and can store and restore the state before the fitness evaluations. The implemented model only represents a simplified subset of the components found in real greenhouses; however, it still illustrates interesting characteristics of greenhouse control.

The model consists of the following variables:

Control variables:

- u_{heat} – Heating. $u_{heat} \in [0, 1]$
- u_{vent} – Ventilation. $u_{vent} \in [0, 1]$
- u_{CO_2} – CO_2 injected. $u_{CO_2} \in [0, 1]$
- u_{light} – Artificial light. $u_{light} \in [0, 1]$

System variables:

- x_{itemp} – Indoor temperature. $x_{itemp} \in [-20, 60]$
- x_{CO_2} – CO_2 level in the greenhouse. $x_{CO_2} \in [0, 5]$
- x_{crop} – Amount of harvested crop. $x_{crop} \in [0, \infty]$

Environment variables:

- z_{otemp} – Outdoor temperature. $z_{otemp} \in [-20, 60]$
- z_{sun} – Sunlight intensity. $z_{sun} \in [0, 1]$
- z_{pcrop} – Price for crops. $z_{pcrop} \in [6.75, 7.25]$
- z_{pheat} – Price for heating. $z_{pheat} \in [1.25, 1.75]$
- z_{pCO_2} – Price for CO_2 gas. $z_{pCO_2} \in [1.75, 2.25]$
- z_{pelec} – Price for electricity. $z_{pelec} \in [0.75, 1.25]$

During the simulation, each system variable is updated using Equation 1 (constants are listed in Table 1). A step in the simulator corresponds to 15 minutes, i.e., a day consists of 96 steps.

The indoor temperature is changed by

$$\begin{aligned}\Delta x_{itemp} &= k_1 \cdot u_{heat} + k_2 \cdot z_{sun} + \\ &\quad (k_3 + k_4 \cdot u_{vent})(z_{otemp} - x_{itemp})\end{aligned}$$

where k_1 is the temperature increase due to heating, k_2 is the increase from sunlight radiation, k_3 is the minimal heat exchange with the environment, and k_4 is the exchange rate when ventilation is used.

The change in indoor CO_2 level is modeled by

$$\begin{aligned}\Delta x_{CO_2} &= -k_5 \cdot \Delta x_{crop} + k_6 \cdot u_{CO_2} + \\ &\quad (k_7 + k_8 \cdot u_{vent})(k_9 - x_{CO_2})\end{aligned}$$

where k_5 is the CO_2 consumption by the plants, k_6 is the increase due to injected CO_2, k_7 is the minimal CO_2 exchange with the environment, k_8 is the exchange by ventilation, and k_9 is the atmospheric CO_2 level.

The crop production per time-step is somewhat more complex. The actual growth per time-step is modeled as a percentage of the optimal growth, i.e., the growth under optimal conditions of temperature, light, and CO_2 level. The change in crop growth is

$$\Delta x_{crop} = k_{10} \cdot \min(G_{temp}, G_{light}, G_{CO_2})$$

where k_{10} is the maximal amount of produced crops. The min-function models that plant growth is limited by the smallest "growth-percentage". For instance, if $G_{temp} = 0.73$, $G_{light} = 0.35$, and $G_{CO_2} = 0.98$, then the current production is at 35%. The impact of these variables on growth is described by growth transfer functions. They are illustrated in Figure 4. The transfer function for temperature models an optimal growth temperature of 30 degrees Celsius with a near optimal range of 25 to 35 degrees. The intervals from -20 to 0 degrees and 45 to 60 degrees do not allow any growth. In fact, the plants die and have to be replanted if the indoor temperature is not between 0 to 45 degrees. In this case, no production is possible for 30 days ($\Delta x_{crop} = 0$). The transfer function for the CO_2-level models a saturation effect. The light transfer function maps both sunlight and artificial light to a production percentage, which is also modeled as a saturation relationship.

The profit in a time-step is calculated as follows:

$$\begin{aligned}p_{profit} &= z_{pcrop} \cdot \Delta x_{crop} - (z_{pheat} \cdot u_{heat} + \\ &\quad z_{pCO_2} \cdot u_{CO_2} + z_{pelec} \cdot u_{light})\end{aligned}$$

Real weather data were available for the environment variables z_{otemp} and z_{sun} (see Figure 5). The data represent a typical year in Denmark. The subset corresponding to March was used in the simulations. March in Denmark has days with both freezing and nonfreezing temperatures. The prices were updated every 7-14 days (randomly determined) and set to a random value in the variable's interval (rounded to steps of 0.05).

Table 1: Constants for Δ-functions.

$k_1 = 0.5$	$k_2 = 0.3$	$k_3 = 0.005$	$k_4 = 0.1$
$k_5 = 0.15$	$k_6 = 0.5$	$k_7 = 0.05$	$k_8 = 1$
$k_9 = 3.0$	$k_{10} = 3.0$		

4 EXPERIMENTS AND RESULTS

The objective in the experiments was to test trade-offs between population size and number of generations using the previously described direct control strategy.

We implemented a classic GA to evolve the four control settings of the greenhouse simulator. For this,

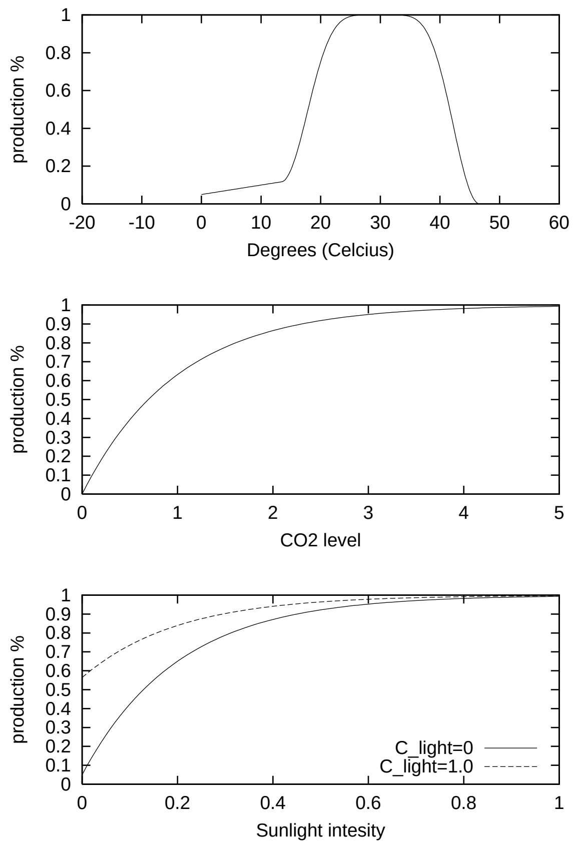

Figure 4: Growth transfer functions for Δx_{crop}. Upper graph: G_{temp}, middle graph: G_{CO_2}, and lower graph: G_{light} with and without artificial light.

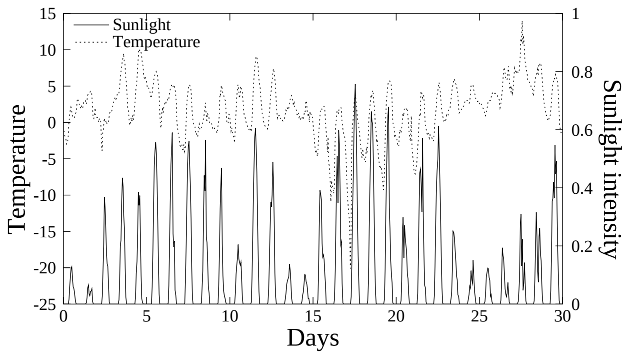

Figure 5: Temperature and sunlight intensity for March in Denmark.

we used real-valued vectors, arithmetic crossover, and Gaussian mutation to evolve the four control settings. Probability of crossover $p_c = 0.9$, probability of mutation $p_m = 0.5$, and variance $\sigma = 0.5$. The search space was discretized, such that u_{heat}, u_{vent}, and u_{CO_2} had steps of size 0.01, whereas the artificial light u_{light} had three settings 0, 0.5, and 1.

Each solution was evaluated by simulating eight steps (2 hours) using the control setting encoded in the genome. The profit achieved in each time-step was recorded and used to calculate the fitness. Two fitness calculation functions were tested, i) sum of profit

$$Fit_{sum}(I) = \sum_{i=1}^{8} p_{profit}[i] \quad (2)$$

and ii) weighted sum of profit

$$Fit_{wei}(I) = \sum_{i=1}^{8} w[i] \cdot p_{profit}[i] \quad (3)$$

where $w = [1.0, 0.875, 0.75, ..., 0.125]$ and $p_{profit}[i]$ denotes the recorded profit in the i-th measurement in the simulation. The second evaluation methods "rates" near future profit higher than profits of a more distant future.

The total number of evaluations provides a basis for comparison. To examine the trade-off between population size and number of generations, we kept the total number of evaluations constant. Two series of experiments were performed; one with 200 evaluations and one with 400. The tested trade-off settings are listed in Table 2.

Table 2: Test cases for trade-offs between population size and generations before update.

# Evals	Pop size	# Generations
200	200	1
	100	2
	50	4
	25	8
400	400	1
	200	2
	100	4
	50	8

The results from the experiments are listed in Table 3. Figure 6 shows the average daily profit for 200 evaluations (popsize=100, # generations=2) for both fitness methods.

Figure 7 illustrates two typical fitness landscapes where u_{heat} is plotted vs. u_{CO_2}. The variables u_{vent} and u_{light} were set to constant values according to the currently best individual to allow a 3D visualization of the fitness landscape.

Table 3: Mean and standard error of total profit (average of 100 runs).

		Pop size	# Gen	Profit	Std. error
200 Evaluations	Sum	200	1	39943.73	± 63.79
		100	2	40097.91	± 63.20
		50	4	40149.43	± 70.40
		25	8	40115.95	± 56.60
	Weight	200	1	39778.90	± 65.29
		100	2	39846.94	± 69.85
		50	4	39908.78	± 68.19
		25	8	39748.82	± 57.88
400 Evaluations	Sum	400	1	39912.91	± 60.25
		200	2	40203.79	± 58.29
		100	4	40028.13	± 60.55
		50	8	40068.30	± 60.86
	Weight	400	1	39706.74	± 62.06
		200	2	39778.89	± 60.19
		100	4	39881.62	± 58.54
		50	8	39790.53	± 60.86

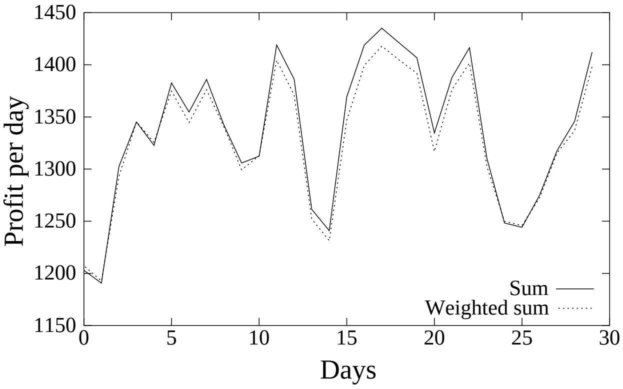

Figure 6: Average of daily profit in March with 200 evaluations (popsize=100, # generations=2) for sum and weighted sum fitness. Average of 100 runs.

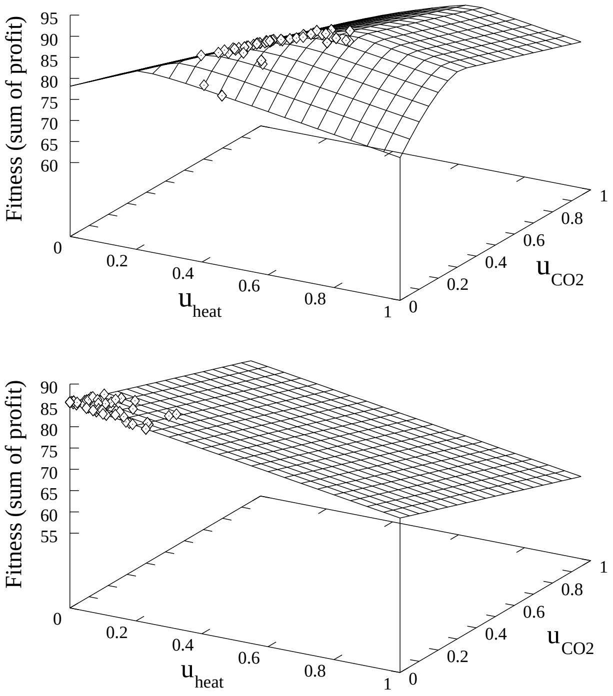

Figure 7: Two typical fitness landscapes. Plots illustrate the control variables u_{heat} versus u_{CO_2}. The variables u_{vent} and u_{light} were set to constant values according to the currently best individual to allow a 3D visualization of the fitness landscape. Diamonds represent the individuals (i.e. the genotypes) of the GA population.

5 DISCUSSION

In this paper, we investigated different strategies for control optimization by the simulation example of profit maximization in a crop producing greenhouse. One of the issues that we examined was how to design the fitness function, such that it estimates the quality of a solution in respect to its impact in the near future. The control strategy calculating the fitness as the simple sum of all profits over the look-ahead time yielded better results than the strategy using a weighted sum to give greater priority to profits in the near than the distant future. This was particularly the case when the profit was high because of beneficial weather conditions. Overall, weather conditions tightly constrained the achievable profit, i.e., cold weather with almost no sunlight required an inevitably higher investment in heating and electricity for the artificial light.

Regarding the time critical problem of fitness evaluations, our results show that the performance of the investigated strategies with different investments in population size versus generations (see Table 2) were rather small. The investment strategies with more than one generation evaluation per time-step turned out to be superior compared with strategies based on only one generation and a maximum number of individuals. However, there was no conclusive ranking pattern among strategies that were evaluated more than one generation. Furthermore, doubling the number of evaluations from 200 to 400 did not yield any significant improvements.

Regarding the greenhouse benchmark test, we found that the generated fitness landscapes resembled a fundamentally different dynamic behavior than what can be simulated with previously suggested dynamic TCGs (e.g. (Branke, 1999; Morrison and Jong, 1999; Grefenstette, 1999)). This result confirms our very recent study on a new test case generator language for dynamic problems (Ursem et al., subm). The greenhouse fitness landscapes turned out to be simple tilted planes when the optimization process was far from the optimum (see Figure 7, bottom). Indeed, this result is not surprising if one considers that a controller can only set control variables within a certain limited range and the target value may only be reached after some time. For instance, to heat up a freezing cold room with a radiator, the best possible setting is to turn up the heat regulator to its maximum until the temperature approaches a pleasant level. Only at this point, the radiator control needs to be more sophisticated. In general, control problems require simple means when the current system state is far from the target state. In multiple dimensions this means that if one or more parameters are poorly controlled then the entire landscape will be tilted in these parameter dimensions. Further, fitness landscapes at more interesting system states turned out to be unimodal with characteristic ridges (see Figure 7, top) rather than multimodal, cone-shaped peaks as suggested by earlier dynamic TCGs. Characteristically for control problems, the search for optimal control affected the transformations of the fitness landscape, which is another property that cannot not be simulated by earlier dynamic TCGs. Although our current implementation of the greenhouse simulator is very simple and preliminary, the principle characteristics of the resulting landscapes reflect what we would expect from realistic dynamic landscapes in control optimization. However, so far the simulator does not pose a very challenging control optimization task. Delay factors, such as a gradual warming-up of the heating system or the rather slow diffusion of the CO_2 gas, would make the control problem harder and more realistic.

Apart from the greenhouse model as a representative for profit maximization problems, it would be important to investigate instances of the other two suggested control problem domains (state stabilization and system to system interaction) in the future. Furthermore, it would be interesting to make an elaborate performance comparison between a classic GA and more advanced techniques, such as Multinational GAs (Ursem, 1999), Religion-based EAs (Thomsen et al., 2000), or mass extinction techniques (Krink et al., 2000; Greenwood et al., 1999), which clearly outcompete classic GAs in numerical benchmark tests. The ability of Religion-based EAs and so-called SOC mass extinction EAs to maintain a much higher population diversity than the classic GA (Thomsen and Rickers, 2001) might turn out to be of particular benefit in control optimization. Moreover, it remains an open question how well other search heuristics, such as hill climbers, like next ascent, steepest ascent, and simulated annealing (Davis, 1987), would cope with control optimization compared to EAs. Also regarding the control strategy there are still open questions related to the design of the fitness function in respect to the quality evaluation of a candidate solution over time. One interesting possibility would be the implementation of self-adaptive look-ahead strategies. Clearly, the present paper is only a first and preliminary attempt to tackle these questions and further research in this area will be required.

Acknowledgements

The authors would like to thank the Danish Research Council, the Ministry of Education, Science and Sport of the Republic of Slovenia, and the Institute for Advanced Study, Berlin, for financial support.

References

[Branke, 1999] Branke, J. (1999). Memory enhanced evolutionary algorithms for changing optimization problems. In Angeline, P. J., Michalewicz, Z., Schoenauer, M., Yao, X., and Zalzala, A., editors, *Proceedings of the Congress of Evolutionary Computation*, volume 3, pages 1875–1882, Washington D.C., USA. IEEE Press.

[Davis, 1987] Davis, L., editor (1987). *Genetic Algorithms and Simulated Annealing*. Research Notes in Artificial Intelligence. Pitman Publishing, London.

[Filipič and Juričić, 1993] Filipič, B. and Juričić, D. (1993). An interactive genetic algorithm for controller parameter optimization. In Albrecht, R. F., Reeves, C. R., and Steele, N. C., editors, *Proceedings of the International Conference on Artificial Neural Networks and Genetic Algorithms ANNGA '93*, pages 458–462, Innsbruck, Austria. Springer-Verlag.

[Filipič et al., 1999] Filipič, B., Urbančič, T., and Križman, V. (1999). A combined machine learning and genetic algorithm approach to controller design. *Engineering Applications of Artificial Intelligence*, 12(4):401–409.

[Filipič and Šarler, 1998] Filipič, B. and Šarler, B. (1998). Evolving parameter settings for continuous casting of steel. In *Proceedings of the 6th European Congress on Intelligent Techniques and Soft Computing EUFIT'98*, volume 1, pages 717–721, Aachen, Germany. Verlag Mainz.

[Fogarty et al., 1995] Fogarty, T. C., Vavak, F., and Cheng, P. (1995). Use of the genetic algorithm for load balancing of sugar beet presses. In Eshelman, L., editor, *Proceedings of the Sixth International Conference on Genetic Algorithms*, pages 617–624, San Francisco, CA. Morgan Kaufmann.

[Greenwood et al., 1999] Greenwood, G. W., Fogel, G. B., and Ciobanu, M. (1999). Emphasizing extinction in evolutionary programming. In Angeline, P. J., Michalewicz, Z., Schoenauer, M., Yao, X., and Zalzala, A., editors, *Proceedings of the Congress of Evolutionary Computation*, volume 1, pages 666–671, Washington D.C., USA. IEEE Press.

[Grefenstette, 1999] Grefenstette, J. J. (1999). Evolvability in dynamic fitness landscapes: A genetic algorithm approach. In Angeline, P. J., Michalewicz, Z., Schoenauer, M., Yao, X., and Zalzala, A., editors, *Proceedings of the Congress of Evolutionary Computation*, volume 3, pages 2031–2038, Washington D.C., USA. IEEE Press.

[Karr, 1991] Karr, C. L. (1991). Design of an adaptive fuzzy logic controller using a genetic algorithm. In Belew, R. and Booker, L., editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 450–457, San Mateo, CA. Morgan Kaufman.

[Krink et al., 2000] Krink, T., Thomsen, R., and Rickers, P. (2000). Applying self-organised criticality to evolutionary algorithms. In Schoenauer, M., Deb, K., Rudolph, G., Yao, X., Lutton, E., Merelo, J. J., and Schwefel, H.-P., editors, *Parallel Problem Solving from Nature – PPSN VI*, volume 1, pages 375–384, Paris, France. Springer.

[Morrison and Jong, 1999] Morrison, R. W. and Jong, K. A. D. (1999). A test problem generator for non-stationary environments. In Angeline, P. J., Michalewicz, Z., Schoenauer, M., Yao, X., and Zalzala, A., editors, *Proceedings of the Congress of Evolutionary Computation*, volume 3, pages 2047–2053, Washington D.C., USA. IEEE Press.

[Thomsen and Rickers, 2001] Thomsen, R. and Rickers, P. (2001). Introducing spatial agent-based models and self-organised criticality to evolutionary algorithms. Master's thesis, University of Aarhus, Denmark.

[Thomsen et al., 2000] Thomsen, R., Rickers, P., and Krink, T. (2000). A religion-based spatial model for evolutionary algorithms. In Schoenauer, M., Deb, K., Rudolph, G., Yao, X., Lutton, E., Merelo, J. J., and Schwefel, H.-P., editors, *Parallel Problem Solving from Nature – PPSN VI*, volume 1, pages 817–826, Paris, France. Springer.

[Ursem, 1999] Ursem, R. K. (1999). Multinational evolutionary algorithms. In Angeline, P. J., Michalewicz, Z., Schoenauer, M., Yao, X., and Zalzala, A., editors, *Proceedings of the Congress of Evolutionary Computation (CEC-99)*, volume 3, pages 1633–1640, Washington D.C., USA. IEEE Press.

[Ursem, 2000] Ursem, R. K. (2000). Multinational GAs: Multimodal optimization techniques in dynamic environments. In *Proceedings of the Second Genetic and Evolutionary Computation Conference (GECCO-2000)*, volume 1, pages 19–26, Las Vegas, USA. Morgan Kauffmann Publishers.

[Ursem et al., subm] Ursem, R. K., Krink, T., Jensen, M. T., and Michalewicz, Z. (subm). Analysis and modeling of control tasks in dynamic systems. *IEEE Transactions on Evolutionary Computation.*