<http://www.diva-portal.org>

Postprint

This is the accepted version of a paper published in *Robotics and Computer-Integrated Manufacturing*. This paper has been peer-reviewed but does not include the final publisher proof-corrections or journal pagination.

Citation for the original published paper (version of record):

Syberfeldt, A., Ng, A., John, R., Moore, P. (2009)

Multi-objective evolutionary simulation-optimisation of a real-world manufacturing problem.

Robotics and Computer-Integrated Manufacturing, 25(6): 926-931

<http://dx.doi.org/10.1016/j.rcim.2009.04.013>

Access to the published version may require subscription.

N.B. When citing this work, cite the original published paper.

Permanent link to this version:

<http://urn.kb.se/resolve?urn=urn:nbn:se:his:diva-3419>

ELSEVIER

Multi-objective evolutionary simulation-optimisation of a real-world manufacturing problem

Anna Syberfeldt^{a*}, Amos Ng^a, Robert I. John^b, Philip Moore^c

^aCenter for Intelligent Automation, University of Skövde, Skövde, Sweden

^bCentre for Computational Intelligence, De Montfort University, Leicester, United Kingdom

^cComputing Sciences and Engineering, De Montfort University, Leicester, United Kingdom

Abstract

Many real-world manufacturing problems are too complex to be modelled analytically. For these problems, simulation can be a powerful tool for system analysis and optimisation. While traditional optimisation methods have been unable to cope with the complexities of many problems approached by simulation, evolutionary algorithms have proven to be highly useful. This paper describes how simulation and evolutionary algorithms have been combined to improve a manufacturing cell at Volvo Aero in Sweden. This cell produces high-technology engine components for civilian and military airplanes, and also for space rockets. Results from the study show that by using simulation and evolutionary algorithms, it is possible to increase the overall utilisation of the cell and at the same time decrease the number of overdue components.

Keywords: Simulation; Multi-objective optimisation; Evolutionary algorithms.

1. Introduction

Real-world manufacturing problems often contain nonlinearities, combinatorial relationships and uncertainties that are too complex to be modelled analytically. In these scenarios, simulation-based optimisation is a powerful tool to determine optimal system settings [1]. Simulation-based optimisation (SO) is the process of finding the best values of some parameters for a system, where the performance of the system is evaluated based on the output from a simulation model. In a manufacturing system, for example, one might be interested in finding the optimal buffer settings with respect to throughput of the system and average cycle time of products. Finding the optimal parameter values is an iterative simulation-optimisation process. An optimisation procedure generates a set of parameter values and feeds them to a simulation that

estimates the performance of the system. Based on evaluation feedback from the simulation, the optimisation procedure generates a new set of parameter values and the generation-evaluation process continues until a user-defined stopping criterion is satisfied.

While traditional optimisation methods have been unable to cope with the complexities of many real-world problems approached by simulation, evolutionary algorithms have proven to be highly useful in SO of complex problems. Evolutionary algorithms (EAs) are generally recognised by a genetic representation of solutions, a population-based solution approach, an iterative evolutionary process, and a guided random search. In evolving a population of solutions, EAs apply biologically inspired operations for selection, crossover and mutation. The solutions in the initial population are usually generated randomly, covering the entire search space. During each generation, some solutions are selected to breed offspring for the next generation of the population.

* Corresponding author. Tel.: +46(0)500448577; fax: +46(0)500448598; e-mail: anna.syberfeldt@his.se.

Either a complete population is breed at once (generational approach), or one individual at a time is breed and inserted into the population (steady-state approach). The process of evolving generations continues until a user-defined termination criterion has been fulfilled. Such a criterion may for example be that a certain amount of time has passed, or that specific objective values have been achieved.

Although EAs have achieved great success in many applications, these algorithms have also encountered some technical hurdles. Among these, efficiency is a major challenge. Real-world optimisation problems often involve an immense number of possible solutions, and an EA needs a large number of simulation evaluations before an acceptable solution can be found [3]-[4]. Even with improvements in computer processing speed, one single simulation evaluation may take a couple of minutes to hours or days of computing time [4]-[5]. This poses a serious hindrance to the practical application of EAs in real-world scenarios. To tackle this problem, the incorporation of computationally efficient surrogates (also called metamodels) has been suggested. A surrogate is an approximation of the simulation; if the simulation is represented as $y=f(x)$, then a surrogate is represented as $y'=f'(x)$, such that $f'(x)=f(x)+e(x)$, where $e(x)$ is the approximation error. For constructing surrogates, a variety of different techniques have been proposed. Among the most popular in evolutionary optimisation is Artificial Neural Networks (ANNs). In general terms, ANN is a non-linear statistical data modelling method used to model complex relationships between inputs and outputs [6]. Originally, the inspiration for the technique was from the area of neuroscience and the study of neurons as information processing elements in the central nervous system. ANNs have universal approximation characteristics and the ability to adapt to changes through training. Instead of only following a set of rules, ANNs are able to learn underlying relationships between inputs and outputs from a collection of training examples, and to generalise these relationships to previously unseen data. These attributes make ANNs very suitable to be used as surrogates for computationally expensive simulation models. However, due to lack of data and high complexity of many problems, surrogates often suffer from false optima [7]. A surrogate can therefore not completely replace the simulation, but the simulation and the surrogate must be used in conjunction during the optimisation.

This paper describes the use of ANNs and EAs in optimising a manufacturing cell at a Volvo Aero in Sweden. Similar to other manufacturing companies, Volvo Aero continuously strives for competitiveness and cost-reduction, and it is therefore important that the cell is operated as efficient as possible. To allow testing and analysis of different production set-ups in the cell, a simulation model has been developed. Experimenting on

the real system is not possible; it is not only impractical but also extremely costly.

In the next section, both the cell and the simulation model are further described. In Section 2, the approach used to tackle the multiple objectives involved in optimising the cell is presented. Section 3 describes the evolutionary algorithm used in the optimisation, followed by a presentation of the results in Section 5. A summary of the paper and future work are presented in Section 6.

2. Manufacturing cell at Volvo Aero

Volvo Aero develops and manufactures high-technology components for aircraft- and gas turbine engines. Today, more than 80 percent of all new commercial aircrafts with more than 100 passengers are equipped with engine components from Volvo Aero. Components manufactured at Volvo Aero can be found in military fighter aircrafts too, such as the F/A-18 E/F Super Hornet. As a partner of the European space program, Volvo Aero is also the primary supplier of nozzles and fuel pump turbines for the Vulcain rocket engine.

Volvo Aero's facilities are located both in Scandinavia and in the US, and in this study the focus is on a factory located at the headquarters of the Volvo Aero Corporation in Sweden. In this factory, a new automated manufacturing cell has recently been introduced that processes a wide range of different engine components. The cell comprises five multi-task machines and five burring stations. The operations to be performed in a machine or at a station vary with different components. Instructions and tools are automatically set-up in a machine for an arriving component, which means that several different components can easily be processed in the cell at the same time. In case two or several components arrive simultaneously to a machine or station, a priority function is used to determine which component that has precedence. In this function, a Critical Ratio (CR) value of each component is calculated to determine how much it is on schedule. The CR value is derived by dividing the time to due date (i.e. scheduled completion) by the time expected to finish the component, according to Equation 1

$$CR = \begin{cases} due \geq now: & \frac{1 + due - now}{1 + TRPT} \\ due < now: & \frac{1}{(1 + now - due) * (1 + TRPT)} \end{cases} \quad (1)$$

where *due* is the due date of the component (i.e. deadline), *now* is the current time, and *TRPT* is the theoretical total remaining processing time (the active operation time including set-up times and movements between machines/stations, and excluding possible delays in the cell). A component with a CR value of 1.0 is "on

schedule”, while it is behind if the value is less than 1.0 and ahead if the value is larger than 1.0. In case of a race condition, the component with the lowest CR value has precedence.

The inflow of the cell is controlled by using fixed inter-arrival times of components. The inter-arrival time does not only specify when a component should enter in the system, but it also determines the component’s due date since an overall production strategy is to process no more than one component of a specific type in the cell at a time. This means that if, for example, the inter-arrival time of component *X* is set to two hours, a new component of type *X* is introduced in the cell every two hour with a due date corresponding to two hours from the point in time it was introduced. For an efficient production, the inter-arrival times should be specified in a way that maximises the utilization of the cell and simultaneously minimises overdue components (i.e. tardiness). For a high utilisation, short inter-arrival times are needed in order to obtain a high load of the cell and thereby avoid machine starvation. However, avoiding overdue components requires generous due dates; that is long inter-arrival times. This means that the two objectives of maximal utilisation and minimal tardiness are conflicting with each other, which makes finding a good configuration of inter-arrival times a difficult task. To aid the cell operators in this task, a discrete-event simulation model of the cell was developed in parallel with its physical build-up. The simulation model is created in the SIMUL8 11.0 software package¹ and a front-end interface is developed in Excel, which facilitates the user in entering input parameters to the model without the need to learn the simulation language. Initial validity tests indicate that the simulation model is a good representation of reality and that it captures the stochastic variations of the cell due to unpredictable machine breakdowns.

Although the simulation model provides a convenient tool for evaluating alternative settings and performing “what-if” analysis of different production scenarios, it supports neither generation nor optimisation of inter-arrival times. Manually optimising inter-arrival times is practically impossible and complementing the simulation model with an optimisation procedure is therefore highly motivated.

In this study, the scenario considered in the optimisation represents two weeks of medium-intensive production and includes 62 individual components of eleven different types.

3. Multi-objective optimisation approach

One difficulty with optimising the manufacturing cell at Volvo Aero is that it requires the simultaneous optimisation

of two conflicting objectives (utilisation and tardiness). No single optimal solution with respect to both objectives exists, as improving performance on one objective deteriorates performance of the other objective.

One way to handle this problem is to derive a set of alternative trade-offs, so called Pareto-optimal solutions. Fig. 1 illustrates the Pareto concept for a minimisation problem with two objectives f_1 and f_2 . In this example, solution *A-D* are non-dominated, i.e. Pareto optimal, since for each of these solutions there exist no other solution that is superior in one objective without being worse in another objective. Solution *E* is dominated by *B* and *C* (but not by *A* or *D*, since *E* is better than these two in f_1 and f_2 , respectively).

Although it is important to find as many (trade-off) optimal solutions as possible in multi-objective optimisation, the user needs only one solution regardless of the number of objectives [2]. Which of the optimal solutions to choose is up to the user based on previous experiences and qualitative information (for example, ergonomic conditions or set-up of factory workers for the day).

Fig. 1 Illustration of Pareto concepts

EAs are very well suited for multi-objective optimisation [2]. Since these algorithms maintain a population of solutions, it is possible to find multiple Pareto-optimal solutions in a single optimisation run. In comparison, classical optimisation algorithms are considerably less efficient since they can only find one Pareto-optimal solution in each run.

4. Evolutionary algorithm

In the optimisation of the manufacturing cell, a multi-objective EA called “Multi-Objective Parallel Surrogate-Assisted EA” (MOPSA-EA) is used. MOPSA-EA (described in [8]) is designed for complex and time-consuming real-world SO problems and therefore suits well for the problem under consideration.

The overall procedure of MOPSA-EA is shown in fig. 2. First, an initial population of random solutions is created

¹ www.simul8.com

and evaluated using the simulation. An ANN surrogate is then initiated using the simulated samples from the first population. Through crossover between solutions in the population, a pool of offspring solutions is created. The offspring are evaluated using the ANN, and their objective values are modified to consider the imprecision of the ANN. The offspring with the best objective values is identified, mutated, and evaluated using the simulation. The simulation sample obtained from evaluation is used to update the ANN. The selected offspring replaces the worst solution in the population, and the procedure continues with generating a new pool of offspring.

Fig. 2 Overall procedure of MOPSA-EA.

The ANN adopted has a feed-forward architecture with one hidden layer (fig. 3). It has been shown (e.g. [10]) that one hidden layer is sufficient for nearly all problems. The number of hidden nodes of the ANN is dynamically adapted to the number of samples available. For a good performance of an ANN, it is recommended that the number of weights of the network is proportional to the size of the training data set [6]. Since the number of samples continuously increase during the optimisation, a static number of hidden nodes is not appropriate. Therefore, the optimisation starts with an ANN of one single node, and additional hidden nodes are successively being added. When the number of samples available exceeds five times the number of weights in the network, a new hidden node is added (according to the weight-sample ratio suggested in [6]). The ANN is trained using back-propagation with a learning rate of 0.5. For each 10th simulation, the ANN is re-trained with the most recent samples (at most 50). The idea of regularly re-training the ANN with the most recent samples is to have a local surrogate defined over the current search region. Local ANNs have been preferred over global ANNs in optimisations [11], mainly because they reduce the time-consumption of the training process [12].

Fig. 3 Artificial neural network.

In fig. 4, the general implementation of MOPSA-EA is outlined. A detailed description of the algorithm can be found in [8].

```

function Main ( )
    population ← ∅
    iter ← 0
    while (not Stop_Optimization ( )) do
        while (ProcessingNodeAvailable ( )) do
            if (| population | = μ) then
                BeginSimulation (GenerateNewSolution (population))
            else
                BeginSimulation (GenerateRandomSolution ( ))
            endif
        endwhile
        p ← WaitForFinishedSimulation ( )
        if (Mod (iter, surrogateUpdateFrequency) = 0) then
            surrogate.Update ( )
        endif
        if (| population | = μ) then
            population.Remove (Select Worst (population))
        endif
        population.Add (p)
        iter ← iter + 1
    endwhile

function GenerateNewSolution (population)
    O ← ∅
    repeat λ times
        parent1 ← SelectForReproduction (population)
        parent2 ← SelectForReproduction (population)
        offspring ← Crossover (parent1, parent2)
        Mutate (offspring)
        SurrogateEvaluation (o)
        O.Add (offspring)
    end
    return Select Best (O)
  
```

Fig. 4. Multi-Objective Parallel Surrogate-Assisted Evolutionary Algorithm (MOPSA-EA).

The following parameter settings of the algorithm have been used in the study:

- Population size: 20
- Number of offspring candidates: 40
- Mutation rate: 0.1
- Crossover operator: Single-point
- Crossover probability: 0.8

5. Results

In fig. 5, the Pareto-optimal solutions found by MOPSA-EA in the optimisation are presented (black points). For the integrity of the company, the axes of the diagram have not been numbered. The optimisation has been performed for 400 simulations, which is the maximum number of simulations that can be performed within the optimisation time-budget defined by the company. For comparison, a solution created manually by domain experts is also shown in the figure (grey rectangle). As can be noticed, several solutions found by MOPSA-EA dominate this solution.

As can be seen in fig. 5, there is an obvious conflict between utilisation and tardiness; an improvement in one of the objectives corresponds to a deterioration of the other objective.

Fig. 5 Optimisation results.

An interesting aspect of the optimisation results is how far the obtained Pareto front is from the “optimal” front. Although the true optimal Pareto front is unknown for the problem (as with most real-world problems), it is at least possible to get an idea of how much further the front can move by performing a large number of additional simulations. The optimisation was therefore run for 2500 simulations, of which the last 100 simulations resulted only in small changes of the front, which indicates that the optimisation has converged after this number of simulations. In fig. 6 the Pareto-front found after 2500

simulation evaluations is shown (grey points). As can be seen in the figure, the front obtained from the optimisation with 400 simulations (black points) is quite far from this front, which indicates that further improvements of the optimisation results are possible if the optimisation budget is made more generous.

Fig. 6 Optimisation results after 400 simulations (black points) and 2500 simulations (grey points).

Another aspect of the optimisation process that is interesting to consider is how much the ANN actually improves the results. To evaluate the impact of the ANN, a test was made of applying MOPSA-EA without using the ANN. The obtained Pareto front from this optimisation is shown in fig. 7 in comparison with the results when using the ANN. Although the algorithm is still able to find a quite good front without using the ANN, the results when using it is somewhat better. An explanation of the relatively small gain of using the ANN might be that the problem is complex, hence making it hard to obtain an ANN of high precision. Although the precision of the ANN might not be very high, it can probably aid the algorithm in filtering out really poor solutions and thereby preventing efforts to be wasted on inferior solutions.

Fig. 7 Optimisation results with ANN (black points) and without ANN (white points).

To analyse the accuracy of the ANN over time, its Mean Square Error is measured every time the ANN is re-trained (i.e. every 10th simulation). Mean Square Error (MSE) is the summed square of the difference between the ANN output value and the simulation output value for all samples, according to equation 2.

$$\frac{1}{n} \sum_{j=1}^n (m_j - s_j)^2 \quad (2)$$

In this equation, n is the number of samples, m is the output from the ANN for sample j , and s_j is the output from the simulation for sample j . The lower the MSE value, the better the accuracy of the ANN. The MSE of the ANN is shown in fig. 8. As can be noticed in the figure, the MSE value is high in the beginning of the optimisation and quickly decreases when the training set is filled with more solutions. When the training set is full after about 50 simulations, the MSE value starts to become stable and it then stays relatively constant during the optimisation. The stable MSE value might be explained by two factors: (a) the size of the training data set is constant, and (b) the training data is normalised, which means that although the actual error of the ANN decrease as the optimisation converge towards a narrow area of the search space (assuming that it is easier for an ANN with a limited number of training samples to learn a smaller part of the fitness landscape), the MSE stays unchanged.

Fig. 8 Mean Squared Error of ANN over time.

6. Summary and future work

This paper describes the application of an evolutionary algorithm in simulation-based optimisation of a manufacturing cell at Volvo Aero in Sweden. In the cell, components for aircraft- and gas turbine engines are processed, and the aim of the study is improve the processing by increasing the utilisation of the cell while at

the same time decrease the number of overdue components. The two objectives of increased utilisation and decreased tardiness are conflicting; improving performance on one of them deteriorates performance of the other. To handle this problem, the concept of multi-objective Pareto-optimisation is adopted in the evolutionary algorithm.

For reduced time consumption, the evolutionary algorithm incorporates a computationally efficient artificial neural network to screen candidate solutions. Results from the optimisation show that the evolutionary algorithm in combination with the artificial neural network is successful in optimising the manufacturing cell. This indicates that the use of Pareto-optimisation and neural networks may be efficient in simulation-based optimisation of complex manufacturing problems.

The next step of this work will be to improve the optimisation to consider simulation randomness. To capture the stochastic behaviour of most real-world complex systems, simulations contain randomness. Instead of modelling only a deterministic path of how the system evolves in the process of time, a stochastic simulation deals with several possible paths based on random variables in the model. To tackle the problem of randomness of output samples is crucial because the normal path of the algorithm would be severely disturbed if estimates of the objective function come from only one simulation replication. The common technique to handle randomness is to send the algorithm with the average values of output samples obtained from a large number of replications. Although this technique is easy to implement, the large number of replications needed to obtain statistically confident estimates from simulation of a complex system that requires long computation time can easily render the approach to be totally impractical. There are methods that guarantee to choose the “best” among a set of solutions with a certain statistically significance level, which require fewer replications in comparison to the others (e.g. Kim-Nelson ranking and selection method [13]). However, combining statistically meaningful procedures that require relatively light computational burden with evolutionary algorithms is still an important topic for further research [12].

Acknowledgments

This work is done within the OPTIMIST project which is partially financed by the Knowledge Foundation (KK Stiftelsen), Sweden. The authors gratefully acknowledge the Knowledge Foundation for the provision of research funding and the industrial partner company Volvo Aero in Sweden for their support in this study.

References

- [1] April J, Better M, Glover F, Kelly J. New advances for marrying simulation and optimization. Proceedings of the 2004 Winter Simulation Conference, Washington, DC, 2004, p. 80-86.
- [2] Deb K. Multi-Objective Optimization using Evolutionary Algorithms. John Wiley & Sons Ltd; 2004.
- [3] Ulmer H, Streichert F, Zell A. Evolution strategies assisted by gaussian processes with improved pre-selection criterion. Proceedings of the IEEE Congress on Evolutionary Computation, Canberra, Australia, December 8-12, 2003, p. 692-699.
- [4] Boesel J, Bowden RO, Glover F, Kelly JP, Westwig E. Future of simulation optimization. Proceedings of the Winter Simulation Conference, 2001, p. 1466-1470.
- [5] Chafeka D, Xuan J, Rasheed K. Constrained multi-objective optimization using steady state genetic algorithms. Proceedings of the Genetic and Evolutionary Computation Conference, Springer-Verlag Germany, 2003, p. 813-824.
- [6] Mehrotra K, Mohan CK, Ranka S. Elements of Artificial Neural Networks. MIT Press; 1996.
- [7] Jin Y, Olhofer M, Sendhoof B. A Framework for evolutionary optimization with approximate fitness functions. IEEE Transactions on Evolutionary Computation, 2002;6:5, p. 481-494.
- [8] Syberfeldt A, Grimm H, Ng A, and John RI. A Parallel Surrogate-Assisted Multi-Objective Evolutionary Algorithm for Computationally Expensive Optimization Problems. Proceedings of the 2008 IEEE Congress on Evolutionary Computation, Hong Kong, June 1-6, 2008.
- [9] Lacksonen T. Empirical comparison of search algorithms for discrete event simulation. Computers & Industrial Engineering 2001;40:1-2, p. 133-148.
- [10] Chen T, Chen H, Liu R. Approximation capability in $\{C(Rn)\}$ by multilayer feedforward networks and related problems. IEEE Transactions on Neural Networks 1995;6:1, p. 25-30.
- [11] Giotis AP, Giannakoglou KC, Periaux J. A reduced-cost multi-objective optimization method based on the pareto front technique, neural networks and PVM. Proceedings of the European Congress on Computational Methods in Applied Sciences and Engineering, Barcelona, Spain, September 15-17, 2000.
- [12] Jin Y, Branke J. Evolutionary optimization in uncertain environments – a survey. IEEE Transactions on Evolutionary Computation 2005;9:3, p. 303-317.
- [13] Gosavi A. Simulation-based optimization: parametric optimization techniques and reinforcement learning. Boston, Mass., Kluwer Academic; 2003.