Robotics and Computer-Integrated Manufacturing 25 (2009) 926–931

Contents lists available at ScienceDirect

Robotics and Computer-Integrated Manufacturing

journal homepage: www.elsevier.com/locate/rcim

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

Anna Syberfeldt [a,*], Amos Ng [a], Robert I. John [b], Philip Moore [c]

[a] *Center for Intelligent Automation, University of Skövde, Skövde, Sweden*
[b] *Centre for Computational Intelligence, De Montfort University, Leicester, United Kingdom*
[c] *Computing Sciences and Engineering, De Montfort University, Leicester, United Kingdom*

ARTICLE INFO

Article history:
Received 27 November 2008
Received in revised form
24 February 2009
Accepted 9 April 2009

Keywords:
Simulation
Multi-objective optimisation
Evolutionary algorithms

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.

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. Either a complete population is bred at once (generational approach), or one individual at a time is bred 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

* Corresponding author. Tel.: +46 500448577; fax: +46 500448598.
E-mail address: anna.syberfeldt@his.se (A. Syberfeldt).

0736-5845/$ - see front matter © 2009 Published by Elsevier Ltd.
doi:10.1016/j.rcim.2009.04.013

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 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% 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 has precedence. In this function, a critical ratio (CR) value of each component is calculated to determine how well 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 the following equation:

$$\mathrm{CR} = \begin{cases} due \geqslant now : \dfrac{1 + due - now}{1 + TRPT} \\ due < now : \dfrac{1}{(1 + now - due) \ast (1 + TRPT)} \end{cases} \tag{1}$$

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 hours 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 utilisation 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 using the SIMUL8 11.0 software package[1] 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.

[1] www.simul8.com