Received September 27, 2021, accepted October 23, 2021, date of publication October 26, 2021, date of current version November 1, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3123211

Online Knowledge Extraction and Preference Guided Multi-Objective Optimization in Manufacturing

INGEMAR KARLSSON[1], SUNITH BANDARU[1], (Senior Member, IEEE), AND AMOS H. C. NG[1,2]
[1]School of Engineering Science, University of Skövde, 541 28 Skövde, Sweden
[2]Department of Civil and Industrial Engineering, Uppsala University, 752 37 Uppsala, Sweden

Corresponding author: Sunith Bandaru (sunith.bandaru@his.se)

This work was supported by The Knowledge Foundation (KK-stiftelsen, Stockholm, Sweden) under project Virtual Factories with Knowledge-Driven Optimization (VF-KDO) (www.virtualfactories.se).

ABSTRACT The integration of simulation-based optimization and data mining is an emerging approach to support decision-making in the design and improvement of manufacturing systems. In such an approach, knowledge extracted from the optimal solutions generated by the simulation-based optimization process can provide important information to decision makers, such as the importance of the decision variables and their influence on the design objectives, which cannot easily be obtained by other means. However, can the extracted knowledge be directly used during the optimization process to further enhance the quality of the solutions? This paper proposes such an online knowledge extraction approach that is used together with a preference-guided multi-objective optimization algorithm on simulation models of manufacturing systems. Specifically, it introduces a combination of the multi-objective evolutionary optimization algorithm, NSGA-II, and a customized data mining algorithm, called Flexible Pattern Mining (FPM), which can extract knowledge in the form of rules in an online and automatic manner, in order to guide the optimization to converge towards a decision maker's preferred region in the objective space. Through a set of application problems, this paper demonstrates how the proposed FPM-NSGA-II can be used to support higher quality decision-making in manufacturing.

INDEX TERMS Manufacturing, decision making, simulation-based optimization, data mining, evolutionary algorithms.

I. INTRODUCTION

It is widely acknowledged that manufacturing is the engine of the modern economy, due to its leading contribution to overall productivity and its multiple effects on growth in the rest of the economy [1]. For manufacturing companies to stay competitive, efficient manufacturing operations are required, both in their design phase and during continuous improvements that necessitate changes and investments to improve their efficiency. Central to the decision-making process in designing or improving manufacturing operations is the concept of knowledge – the more a decision maker understands the relationship between how to design and run a system and how it performs, the better are the decisions that can be made [2]. However, knowing what actions to perform to maintain and improve efficiency can be hard, due to the complex nature of manufacturing processes. One of the tools for analyzing manufacturing processes is simulation, where a model is built to depict a real-world process or system. Simulation has historically been used for prediction and performance analysis of different aspects of manufacturing, for example, workforce planning, supply chain management and capacity planning [3], as well as improving resource efficiency [4]. Some recent comprehensive reviews of the use of simulation in manufacturing can be found in [5] and [6]. Although simulation is often used as an evaluative tool, it cannot generate any optimal solution when used on its own. Consequently, researchers have shown a significant interest in merging optimization and simulation in order to find the optimal or near-optimal design and/or operating policies for manufacturing systems [5]. In such a simulation-based optimization (SBO) procedure, an optimization algorithm is used

The associate editor coordinating the review of this manuscript and approving it for publication was Yongming Li.

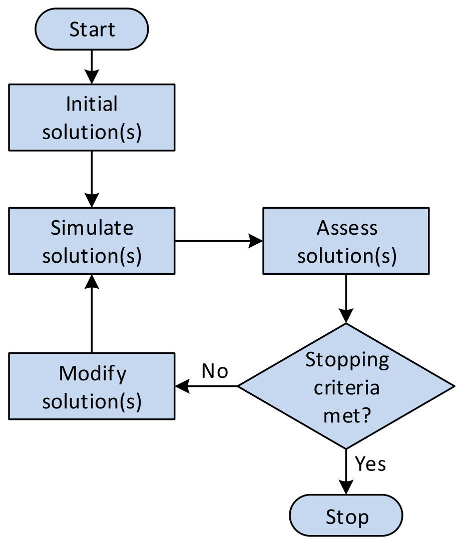

FIGURE 1. Simulation-based optimization process.

together with a simulation model for evaluating candidate solutions in an iterative manner as shown in Figure 1. It has huge potential to generate optimal or near-optimal solutions that can be used to assist decision-making in designing and improving manufacturing systems [7], [8].

Optimization problems within manufacturing as well as many other domains often involve multiple conflicting objectives [9]. A multi-objective optimization problem can be defined as:

$$\begin{aligned} &\text{Minimize } \mathbf{F}(\mathbf{x}) = \{f_1(\mathbf{x}), \ldots, f_M(\mathbf{x})\} \\ &\text{subject to } g_j(\mathbf{x}) \geq 0 \quad \forall j = 1, 2, \ldots, J \\ &\qquad h_k(\mathbf{x}) = 0 \quad \forall k = 1, 2, \ldots, K \\ &\qquad \mathbf{x}^{(\mathbf{L})} \leq \mathbf{x} \leq \mathbf{x}^{(\mathbf{U})} \end{aligned} \tag{1}$$

where f_i: $\mathbb{R}^n \to \mathbb{R}$ are two or more objectives (M) that are conflicting and need to be minimized. The vector of variables $\mathbf{x} = [x_1, x_2, \ldots, x_n]^T$ belongs to the non-empty feasible region formed by the inequality constraints $g_j(\mathbf{x})$, equality constraints $h_k(\mathbf{x})$ and variable bounds. The feasible region is defined by the constraints of the problem and bounds on the variables. An important concept within multi-objective optimization is that of *dominance* which can be used to compare two solutions. A variable vector $\mathbf{x_1}$ is said to dominate another variable vector $\mathbf{x_2}$ ($\mathbf{x_1} \preceq \mathbf{x_2}$) if and only if:

1) $f_i(\mathbf{x_1}) \leq f_i(\mathbf{x_2})$, $\forall i \in \{1, 2, \ldots, M\}$,
2) $\exists j \in \{1, 2, \ldots, M\}$ such that $f_j(\mathbf{x_1}) < f_j(\mathbf{x_2})$.

If neither $\mathbf{x_1} \preceq \mathbf{x_2}$ nor $\mathbf{x_2} \preceq \mathbf{x_1}$, then they are said to be non-dominated with respect to each other and denoted as $\mathbf{x_1} \parallel \mathbf{x_2}$. A feasible solution $\mathbf{x}^*$ is said to be Pareto-optimal if there does not exist any other feasible solution dominating $\mathbf{x}^*$. The manifold containing the Pareto-optimal solutions is called the Pareto-optimal front.

The n-dimensional space formed by the variables is called the *decision space*. Each solution in the decision space maps to a point in the M-dimensional space formed by the objectives, which is called the *objective space*. Figure 2 shows the mapping between the decision space and objective space in a problem with two variables and two conflicting minimization objectives. The Pareto-optimal solutions are shown in dark.

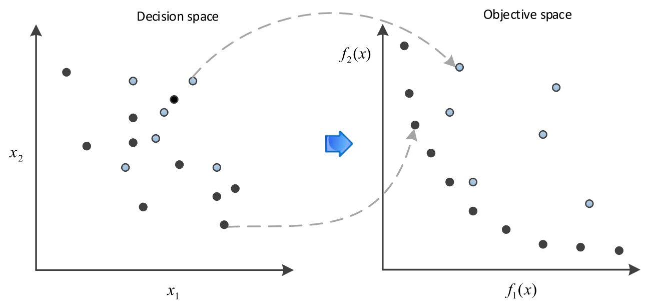

FIGURE 2. Decision and objective space.

Simulation-based multi-objective optimization problems are difficult to solve using classical optimization methods due to various reasons. Firstly, as explained above, multiple objectives lead to multiple solutions. Most classical optimization methods can only generate one solution at a time, therefore, in order to find the Pareto-optimal set, several optimization runs have to be carried out. Secondly, by definition, a simulation model has non-analytical objective functions. Classical optimization techniques often rely on objective and constraint function gradients which are either unavailable or difficult to calculate for simulated outputs. Thirdly, due to the stochastic nature of objective and constraint function, the resulting multi-objective optimization problem can have several locally optimal solutions and classical optimization techniques are prone to such solutions.

The three main issues described above can be overcome using evolutionary algorithms. Evolutionary algorithms aim to mimic principles of natural evolution observed in the nature by using a population of individuals (solutions) that *evolve* over several generations (iterations) to generate the optimal solutions [10]. Evolutionary algorithms do not require gradient information and, due to their stochastic nature, they also have the ability to recover from local optima. Evolutionary algorithms are especially popular in solving multi-objective optimization problems due to their population-based approach which allows obtaining multiple optimal solutions simultaneously. Evolutionary algorithms are also better suited for parallelization.

A decision maker's expert knowledge can be advantageous when solving complex multi-objective optimization problems, if his/her preferences can somehow be embedded into an evolutionary multi-objective (EMO) algorithm [11]–[13]. There are many different ways of incorporating decision maker's preferences. For example, by using different weights for each objective, the decision maker can define the relative importance of one objective over another, and then minimize the weighted sum of the objectives, essentially converting the multi-objective optimization problem into a single objective optimization problem. However, such scalarization methods have some drawbacks and do not always work as expected. For example, the simple weighted sum approach described here is only applicable to problems where the Pareto-optimal front is convex [14].

Combining SBO with evolutionary algorithms and data mining can be advantageous as a tool for extracting knowledge about a manufacturing system. The manual use of these techniques can be a complex undertaking and, therefore, they should be integrated into a decision support system. However, to our best knowledge, there are hitherto no decision support systems that combine these techniques [15].

The aim of this article is therefore to study how knowledge extracted from combining data mining methods with SBO can be used to better understand the implications of optimal solutions in manufacturing system problems for the purpose of decision-making. The study also explores how online knowledge extraction can be used to guide the SBO process based on the decision maker's preference. The paper is structured as follows: Section II reviews previous related research studies. Section III describes a technique that uses data mining to extract knowledge during and after an optimization run. Section IV proposes an extension of a multi-objective evolutionary algorithm that uses knowledge extracted through data mining to automatically guide an optimization towards the preferences indicated by a decision maker. Then, in Section V, the proposed methods are demonstrated with applications from the manufacturing industry. The obtained results are discussed in Section VI, and finally, the paper concludes in Section VII.

II. RELATED WORK

In this section, we present a review of previous works related to the three methodological aspects of this work. Section II-A discusses data mining methods that can generate knowledge in a form suitable for integration with an optimization algorithm. Section II-B covers existing preference-based evolutionary algorithms and their corresponding preference articulation methods. And finally, Section II-C explores existing approaches to combining data mining methods with evolutionary algorithms.

A. DATA MINING METHODS FOR KNOWLEDGE DISCOVERY

Several data mining approaches have been used in the literature for knowledge discovery from the solutions obtained through multi-objective optimization. A recent survey of such methods [16] classifies them based on the representation of the obtained knowledge: (i) *descriptive statistics*, which are simple numerical measures that summarize data, leads to explicit knowledge, (ii) *visual data mining techniques* represent knowledge through various graphical means such as clusters or heatmaps, thus they have an implicit representation, (iii) *machine learning methods* can represent knowledge in both implicit and explicit forms.

Implicit knowledge is knowledge without a formal notation and therefore it is often presented visually, which not only leads to subjective interpretations, but also makes it difficult for an algorithm to use the knowledge automatically. Explicit knowledge, on the other hand, uses mathematical notation which is well-defined and complete so that no information is lost [17]. Explicit forms of knowledge have a fixed mathematical notation and are therefore easy to store, transfer and process automatically within a computer program. Hence, here we only focus on data mining methods that can discover explicit knowledge.

The most common type of explicit knowledge takes the form of *if A then B* rules involving the variables in the antecedent *A* and objectives in the consequent *B*. The earliest application of rule extraction in multi-objective optimization can be found in [18], where each objective is discretized into different levels and the rules involving the variables are obtained using classification trees. A similar method involving regression trees is used in [19] and [20] to generate rules describing a preferred region in the objective space. Both works demonstrate how the obtained rules can provide a better understanding of complex SBO problems to decision makers. Rough set theory and association rule mining have also been used to obtain *if-then* rules. The former has been applied to multi-objective design optimization of a centrifugal impeller in [18], [21], while the latter has been shown to be effective in [22] on the same problem. Another approach called Multi-Objective Rule Extraction, specifically designed for obtaining rules for SBO problems in production systems, is proposed in [23]. The method extracts rules by finding critical values for each variable that give significant jumps in objective function values. This gives the decision maker information about which configuration is best suited for production system given the desired level of investment. The authors show the application of the method in real-world production system optimization.

Another common type of explicit knowledge takes the form of mathematical relationships involving variables, objective functions and constraint functions. The simplest form of mathematical relationships are linear correlations and these can easily be obtained through methods like Principal Components Analysis (PCA) and Linear Discriminant Analysis (LDA). These methods have traditionally been used for graphical representation of solutions [24], [25], but can also be used for knowledge extraction. For example, a closely related method called Proper Orthogonal Decomposition has been employed in [26] and [27] to extract design knowledge from the Pareto-optimal solutions of an airfoil shape optimization problem.

While linear correlations are indeed insightful when present in optimization data, often the relationships between any given set of variables are nonlinear. Moreover, the nature of the relationships may be different in different regions of the search space. *Innovization* or (innovation through optimization) proposed in [28] can handle such nonlinearities through a manual approach of analyzing scatter plots of different variable combinations and performing regression with appropriate models on the correlated parts of the data. Later, an *automated innovization* approach was developed [29], [30] which uses unsupervised machine learning to extract multiple nonlinear relationships simultaneously without human intervention [31]. This approach was further extended in [32] to obtain mathematical relationships of arbitrary kinds using genetic programming. Automated innovization has been used for knowledge discovery in several real-world multi-objective optimization problems [33].

B. PREFERENCE-BASED EVOLUTIONARY ALGORITHMS

Preference-based evolutionary algorithms use information about the decision maker's preferences to obtain a preferred set of solutions on the Pareto-optimal front. These algorithms can be classified into three categories based on when the user input is needed, *a priori*, *a posteriori* and interactive methods. *A priori* methods take the decision maker's preferences into account right from the start of an optimization run. It is assumed that the decision maker is able to elicit the preference information accurately at the outset. On the other hand, *a posteriori* methods require preferences only after obtaining a representative set of trade-off solutions. While this helps the decision maker to realistically set his/her expectations, generating the entire trade-off front in the first place can itself be a challenging task, particularly when many objectives are involved. Interactive methods try to combine aspects of *a priori* and *a posteriori* methods by involving the decision maker during the optimization run in an iterative manner to guide search towards his/her preferences [34]. While this generally reduces the computational effort required to obtain preferred solutions, it drastically increases the number of times the decision maker's inputs are needed, thus risking cognitive overload.

Preference-based evolutionary algorithms can also be classified based on how the user specifies his/her preferences. One of the most common ways of articulating preferences is to use one or more *reference points*. A reference point is a point in the objective space which specifies the aspirations of the decision maker for each objective function. Depending on its location, the reference point can either be attainable (achievable) or unattainable (unachievable). An attainable point is one that lies in the feasible space and it is therefore possible for an optimization algorithm to reach the point. An unattainable point is one that is specified in the infeasible region, in which case the optimization algorithm can approach it only as close as the feasible space allows. Preferences can also be obtained through pairwise comparisons between representative solutions. However, in practice, this approach can be time-consuming and too demanding of the decision maker. A more recent method of preference articulation involves specifying a region of the objective space through the action of *brushing* using mouse input as presented in [35]. This is a visual and interactive method and as in the case of reference points, the decision maker does not need to have prior knowledge of the range of values in the objective space. Other common ways of articulating preferences is through reference directions, light beams and value functions [36]. Here, we specifically review preference-based evolutionary algorithms that use reference point(s), since the algorithm proposed in this paper is of a similar nature.

The reference point-based NSGA-II or R-NSGA-II introduced in [37], uses the weighted Euclidean distance of each solution from the reference point as a metric to promote diversity close to the reference point. It also employs a clearing strategy to avoid overcrowding of solutions at a particular reference point by controlling the extent and distribution of obtained solutions. While R-NSGA-II uses *a priori* elicitation of preferences, an interactive version of the algorithm proposed in [38] relies on pairwise comparisons between selected solutions at regular intervals. Based on the inputs from a decision maker, the algorithm creates a utility function which in turn guides search.

The Preference-Based Evolutionary Algorithm (PBEA), introduced in [39], incorporates preferences into an evolutionary algorithm using modified quality indicators. The decision maker's preference is expressed as reference points which are used in the selection operator of the algorithm together with an achievement scalarizing function to guide the search towards the region of interest. A weighted form of achievement scalarizing function (WASF-GA) [40] has also been developed where the weights are chosen so as to project the reference point onto a region of interest on the Pareto-optimal front.

The concept of g-dominance proposed in [41] works by redefining Pareto-dominance such that solutions that dominate or are dominated by the reference point are considered to be preferable to solutions that non-dominated with the reference point. The proposed g-NSGA-II algorithm requires minimal changes to the base algorithm and can be utilized in both *a priori* and interactive manner. However, the reference point needs to be fairly close to the Pareto-optimal front in order to avoid obtaining a large region of interest.

The concept of r-dominance introduced in [42] also redefines Pareto-dominance such that a strict partial order can be obtained among non-dominated solutions. This allows solutions closer to the reference point to be preferred over other solutions of the same rank. The authors demonstrate that the approach is effective in both *a priori* and interactive forms.

The recently proposed RMEAD2 [43] and MOEA/D-NUMS [44] integrate user-preferences into the decomposition-based evolutionary algorithm MOEA/D by biasing the weight vectors towards the given reference point.

It is noteworthy that most of the preference-based evolutionary algorithms in the literature use NSGA-II as the underlying multi-objective optimizer. Since this is also true for the algorithm proposed in this paper, for completeness we briefly describe NSGA-II later in Section III-A.

C. COMBINING EVOLUTIONARY ALGORITHMS WITH DATA MINING

The main purpose of combining evolutionary algorithms with data mining methods is to improve the performance of the algorithm through the knowledge extracted during the optimization process. Such algorithms may be referred to as online knowledge-driven optimization algorithms as proposed in [16]. Since the knowledge needs to be processed automatically by an algorithm, only explicit representations can be used.

One of the early works in this area is the Learnable Evolution Model (LEM) proposed in [45]. It uses machine learning to obtain rules that differentiate between high-performing and low-performing solutions. These rules are used in turn to generate new solutions. While LEM was only applied to single-objective optimization problems, it showed the potential of embedding extracted knowledge into the optimization algorithm. Later, the method was extended in [46] to handle multi-objective optimization problems.

Linear correlations obtained by PCA can also be utilized to enhance search. For example, the approach proposed in [47] detects correlations in the decision space and builds a probability model that is used to generate solutions in subsequent iterations. Identification of linear and nonlinear correlations in the objective space can also help the search process when redundant objectives are removed online as demonstrated in [48]. This approach is especially useful for solving many-objective optimization problems.

The interactive method presented in [49] uses Dominance-based Rough Set approach to build a preference model based on the decision maker's ranking of certain representative solutions. The preference model consists of if-then-else rules which are in turn added as constraints to the original optimization problem, thus guiding the algorithm towards preferred regions of the search space.

III. ONLINE KNOWLEDGE EXTRACTION

In this section, we first describe the NSGA-II algorithm, a well-known multi-objective evolutionary algorithm, to facilitate our discussion later on how it can be extended with online knowledge extraction based on the decision maker's preference. Next, we present the Flexible Pattern Mining (FPM) approach that is used in this work for extracting knowledge. Thereafter, we first describe how FPM is used in a decision support system to extract knowledge in an offline manner, i.e. after the optimization run is complete. And finally, we present the extension of FPM-based decision support system to facilitate online knowledge extraction, i.e. during the optimization run.

A. THE NSGA-II ALGORITHM

Genetic algorithms are inspired by the process of evolution seen in nature. A typical genetic algorithm starts with *initialization* of a set of random solutions called the *parent population*. Based on their fitness with respect to the objective function, a subset of these solutions is selected to form the so called *mating pool*. A *child population* is generated from the mating pool by applying the genetic operators, *crossover* and *mutation* on the solutions. Based on a crossover probability, each crossover operation recombines two parent solutions to form two recombinant solutions. Similarly, based on the mutation probability, the mutation operator mutates each recombinant to form a child solution. The parent and the child population are then merged and the new parent population is formed by selecting the best fitness solutions while maintaining the population size. This process is repeated for a fixed number of iterations or generations.

There are many multi-objective optimization techniques based on genetic algorithms. Non-dominated Sorting Genetic Algorithm II (NSGA-II) [50] is once such technique that has been used as a template for many other multi-objective optimization algorithms in several different application areas [9]. NSGA-II has been chosen in this work because it is well-known and has relatively good performance on multi-objective test problems as well as real world optimization problems [50]. Like genetic algorithms, NSGA-II is also a population based technique, which allows it to converge to multiple Pareto-optimal solutions in a single algorithmic run. The solutions in the population are evolved with crossover and mutation and each new set of individuals is called a generation. The individuals in a generation is evaluated with, in our case, a test function for a test problem or a simulation model. To create a set of individuals for a new generation the current generation is sorted based on the concept of dominance and crowding distance. Based on this sorting a new population is selected and the process is repeated until a stopping criterion is met, for example, maximum number of generations or running time. This process of selecting a new generation is shown in Figure 3 [50]. The child population C_t is created from the population P_t. These two populations is combined (R_t) and are sorted with non-dominated sorting into fronts ($F_1, F_2, F_3, \ldots, F_l$) where the individuals from the best fronts are used to create the new generation. If a front does not fit into the new generation it is sorted with crowding distance and the individuals with the highest crowding distance is kept for the new generation (P_{t+1}). The algorithm is described in detail in [50] and pseudo-code for the algorithm can be seen in Algorithm 1.

Algorithm 1 Pseudo-Code for NSGA-II

```
population ← GenerateRandomSolutions
while termination criteria not met do
    EvaluateObjectiveValues(population)
    parents ← Select(population)
    recombinants ← Crossover(parents)
    children ← Mutate(recombinants)
    combinedPopulation ← population + children
    NonDominatedSorting(combinedPopulation)
    population ← SelectBest(combinedPopulation)
end while
```

B. FLEXIBLE PATTERN MINING

Flexible Pattern Mining or FPM is a recent extension [51] to the popular sequential pattern mining (SPM) algorithm proposed in [52]. The original SPM algorithm was designed to extract frequent customer sequences (also known as

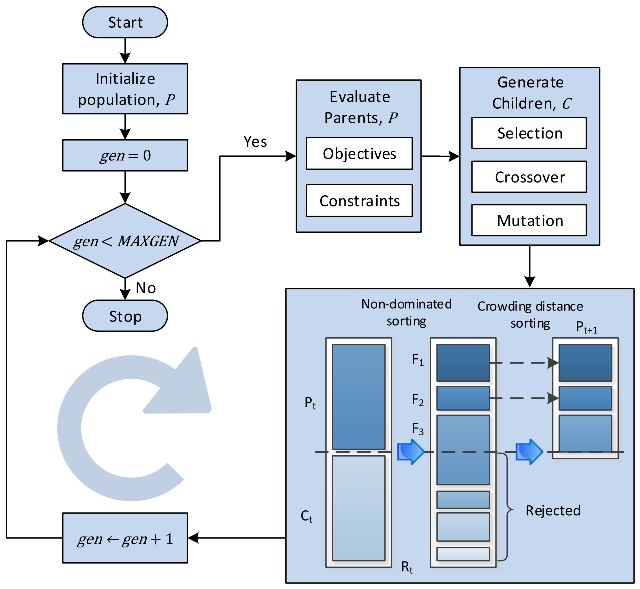

FIGURE 3. NSGA-II algorithm as shown in Algorithm 1.

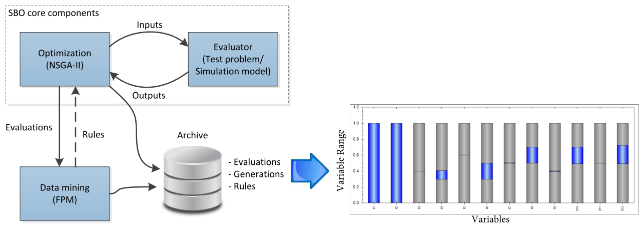

FIGURE 4. Integration of SBO with online and offline knowledge discovery through FPM.

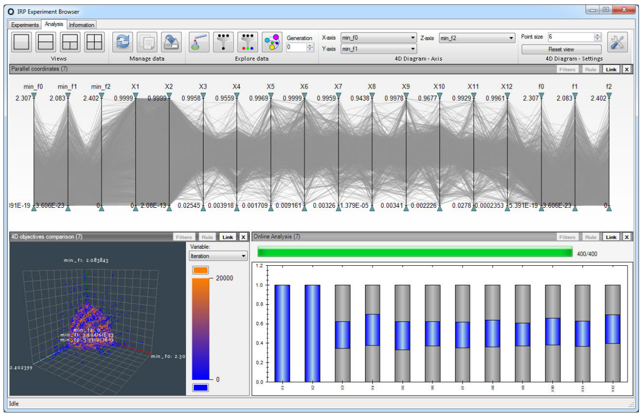

FIGURE 5. Experiment Browser showing visualization of solutions on parallel coordinate and scatter plots. The rules obtained by FPM at the last generation are visualized as colored vertical bars. For example, here $x_3 > 0.35 \wedge x_3 < 0.625$.

sequential patterns) from market basket data. Given a dataset of transactions of different customers, called the transaction database, the SPM algorithm first generates a database of customer sequences. A customer sequence is essentially a chronologically ordered list of *itemsets*, each itemset itself being a group of items bought together by the customer. Frequent customer sequences are those that meet a minimum support threshold. FPM uses a similar approach, by considering the solutions generated during optimization as transactions and the variables as items. Thus, each transaction will have the same number of items, which is not the case in SPM. When the variable values are continuous or ordinal in nature, FPM can also generate *sequential rules* instead of sequential patterns by transforming the optimization data into a truth matrix. For each possible value d_{ij} of a variable x_i present in the solution set, FPM adds three binary columns in the truth matrix corresponding to the logical expressions $x_i < d_{ij}$, $x_i == d_{ij}$ and $x_i > d_{ij}$. Frequent patterns can then be found within the truth matrix. For example, if $x_1 > d_{12}$ is `true` for 100% of the solutions and $x_2 < d_{24}$ is `true` for 80% of the solutions, then the sequential rule $x_1 > d_{12} \wedge x_2 < d_{24}$ has a support of 80%.

C. DECISION SUPPORT SYSTEM FOR KNOWLEDGE DISCOVERY

To support decision makers with the task of SBO in manufacturing applications, the authors have previously proposed a Decision Support System (DSS) in [15]. This DSS, implemented in C++, has the ability to perform multi-objective optimization using NSGA-II on discrete event simulation models of real-world production systems and thereafter explore the obtained Pareto-optimal solutions. Here, we describe how FPM was integrated into this DSS to enable both offline and online knowledge discovery.

The FPM algorithm is added as the third component of the DSS as shown in Figure 4. It uses the traditional SBO process to feed optimization data to the FPM algorithm in real-time through an SQL database containing an archive of solutions from all generations. The archive also contains the rules generated by the FPM algorithm at the end of each generation of NSGA-II in Algorithm 1. The solutions and rules from any generation can be visualized by the decision maker through another interface called the Experiment Browser.

The Experiment Browser is simply a piece of software, implemented in C#, that connects to the internal storage of the DSS and provides the user with different tools to visualize and analyze the results. The Experiment Browser has scatter plots for two and three dimensional optimization data, parallel coordinates, as well as a table that displays the raw data from experiments. Figure 5 shows a screen-shot of the Experiment Browser with three open diagrams: parallel coordinates, scatter plot, and FPM rule plot. The FPM rule plot (shown as 'Online Analysis') is used to visualize the rules obtained by the FPM algorithm at the last generation. Each vertical bar represents the bounds of a variable in the optimization problem. Since FPM rules are of the form $x_i < d_{ij}$ or $x_i > d_{ij}$, they can be interpreted as subsets of the original variable bounds and shown in color on the vertical bars.

IV. FPM-NSGA-II: A PREFERENCE GUIDED EVOLUTIONARY ALGORITHM

The FPM algorithm described in Section III-B is capable of generating knowledge in the form of explicit rules from

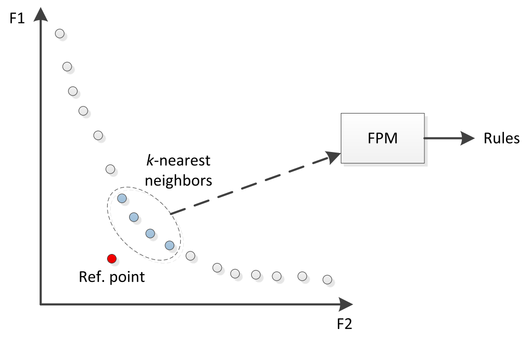

FIGURE 6. FPM-NSGA-II uses FPM to generate rules that describe solutions close to the reference point. The k nearest Euclidean neighbors in the normalized [0, 1] objective space are passed to FPM as *selectedSols*.

intermediate generations of an evolutionary multi-objective optimization algorithm. As mentioned before, a unique advantage of explicit knowledge is that it can be processed automatically by an algorithm. Our proposed preference guided evolutionary algorithm is called of FPM-NSGA-II due to the integration of FPM into NSGA-II. Like most other *a priori* and interactive methods, FPM-NSGA-II uses a reference point for articulating user preferences. After every T generations, where T is set by the user, FPM-NSGA-II calls a function called `RunFPM()` shown in Algorithm 2. This function first retrieves the current non-dominated from, *currentNDF* from the archive A of evaluated solutions. After normalizing the objectives to [0, 1], the k nearest Euclidean neighbors of the reference point, *refPoint*, are selected from the *currentNDF* as shown in Figure 6 to form the set *selectedSols*. This set of solutions is assumed to represent the current preferences of the decision maker. Any significant rules extracted from this subset of non-dominated solutions will thus capture the decision maker's preferences in the decision space. The solutions that do not represent the decision maker's preferences, *unselectedSols*, are obtained by taking the set different between *currentNDF* and *selectedSols*. Next, FPM is used to obtain rules corresponding to *selectedSols*. The support for each rule in both *selectedSols* and *unselectedSols* is also calculated.

Algorithm 2 Pseudo-Code for `RunFPM(A, k)`

```
$currentNDF \leftarrow$ GetNonDominated(A)
$selectedSols \leftarrow$ kNN($currentNDF$, $refPoint$)
$unselectedSols \leftarrow currentNDF \setminus selectedSols$
$rules \leftarrow$ FPM($selectedSols$, $unselectedSols$)
```

In order to guide the optimization towards the preferred regions (in this case, towards *refPoint*), the rules obtained by FPM are added as *probabilistic constraints* to the current generation. In subsequent generations, the probabilistic constraints penalize solutions that do not conform to the decision maker's preference, while promoting solutions close to *refPoint*. Probabilistic constraints act like regular inequality constraints, with the only difference that not all solutions are affected by the same constraints. Each rule from FPM

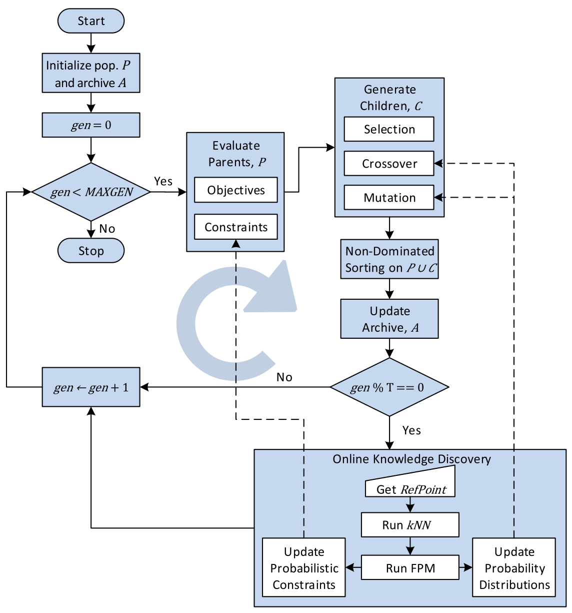

FIGURE 7. A schematic of the FPM-NSGA-II algorithm.

that is added as a probabilistic constraint has an associated probability equal to its support. Thus, a support value of 95% means that the rule acts as a constraint with a probability of 0.95 on each solution in the current generation. A solution that adheres to the rule is not affected by this constraint. However, on the average 95% of the solutions that do not follow this rule will be penalized. Such probabilistic constraints help in maintaining some diversity among the solutions, that avoiding premature convergence.

The crossover and mutation operators are also modified using the rules to promote creation of children that adhere to the rules obtained from FPM. Simulated binary crossover (SBX) [50] and polynomial mutation operators are used in this work. Both the operators are modified by altering the bounds of the variable being recombined or mutated based on the FPM rule corresponding to that variable. For example, if a variable x_i with original bounds $\left[x_i^{(L)}, x_i^{(U)}\right]$ is associated with the FPM rule $x_i > p_i$, then the bounds are modified as $\left[p_i, x_i^{(U)}\right]$ in SBX and polynomial mutation. The support of the rule is again used as the probability with which this modification is applied. The net effect is a change in the probability distributions of the offspring in the current and subsequent generations.

Together, the probabilistic constraints and modified SBX and polynomial mutation operators guide the solutions towards the preferred region as specified by the reference point. The complete FPM-NSGA-II algorithm is shown in Figure 7.

In this work, we set the number of generations between FPM calls to $T = 10$ generations. After every T generations, existing probabilistic constraints are purged and all modifications to the crossover and mutation operators are reversed. Thus, theoretically the algorithm has the ability to recover from premature convergence towards *refPoint*.

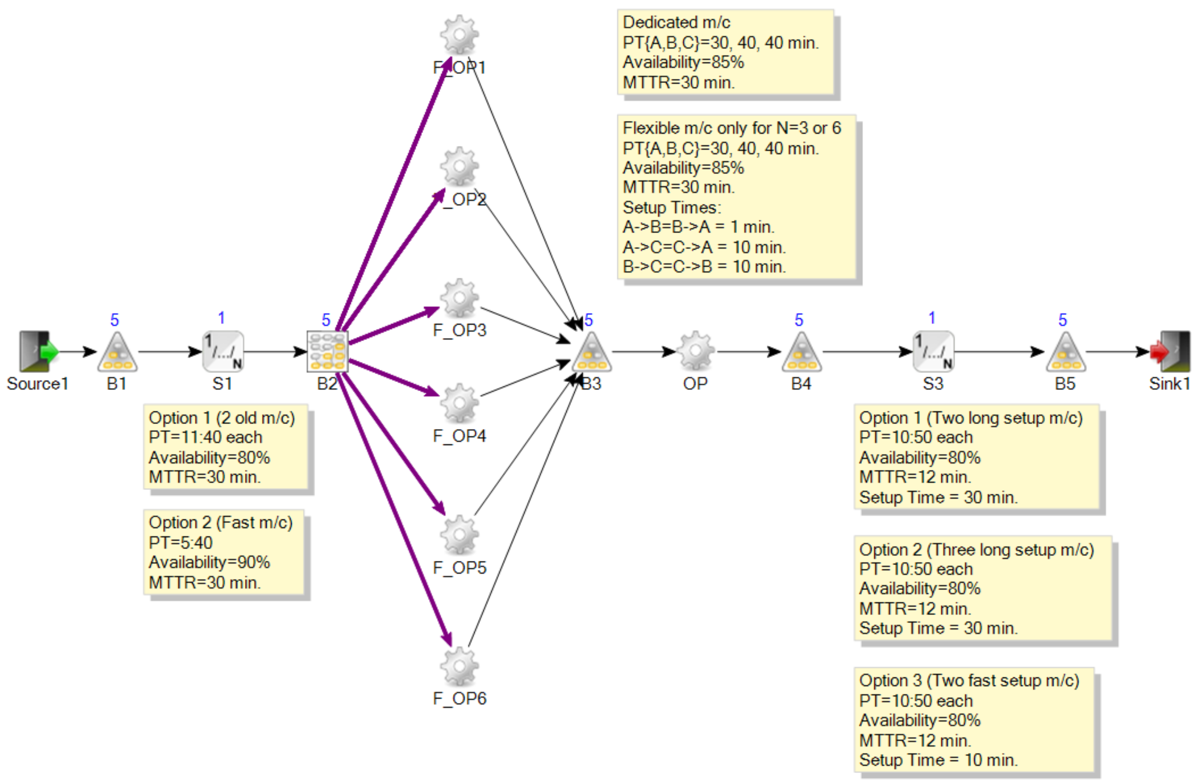

FIGURE 8. Simulation model depicting the PSU2 production system.

Note that since the online knowledge extraction procedure is a module attached to the regular NSGA-II algorithm, the latter can therefore easily be replaced with any other state-of-the-art multi-objective optimization algorithm.

FPM-NSGA-II is fundamentally different from existing preference-based evolutionary algorithms in two main ways. Firstly, as discussed in Section II-B, algorithms such R-NSGA-II, PBEA and g-NSGA-II work via the selection mechanism by modifying the dominance operator to emphasize solutions closer to *refPoint*. On the other hand, FPM-NSGA-II works via the variation mechanisms by modifying crossover and mutation operators as described above. It also affects the selection process indirectly through probabilistic constraints that change every T generations. A more important distinction however is the fact that unlike existing preference-based evolutionary algorithms, FPM-NSGA-II also generates a set of rules at the end that capture decision maker's preferences in the variable space.

A. COMPUTATIONAL COMPLEXITY OF FPM-NSGA-II

The time complexity of NSGA-II algorithm is determined by the non-dominated sorting procedure depicted in Figure 3, which is known to be $\mathcal{O}(MN^2)$ [50], where M is the number of objectives and N is the population size. The computational runtime of FPM is determined by the underlying Apriori algorithm [53] and depends on four factors [54], namely (i) support threshold, (ii) number of transactions (N), (iii) number of unique items (d) and (iv) average transaction width (w). As mentioned before, in FPM all transactions (solutions) have the same number of unique items (variables). Therefore, the average transaction width is $w = d = n$, n being the number of variables. The worst case complexity of $\mathcal{O}(2^n)$ occurs when support values for all possible variable combinations need to be calculated. However, this upper bound is practically only possible when support threshold is zero. In FPM-NSGA-II, only unary rules of the forms $x_i < d_{ij}$, $x_i == d_{ij}$ and $x_i > d_{ij}$ are sought, which only requires $\mathcal{O}(Nn)$ operations. Thus, the overall computational complexity of FPM-NSGA-II is either $\mathcal{O}(MN^2)$ or $\mathcal{O}(Nn)$, whichever is greater. From this we can conclude that the FPM part of FPM-NSGA-II scales linearly with the number of solutions and number of variables. In the application problems used in this paper, we have $N > n$, so the runtime of FPM-NSGA-II is limited by the non-dominated sorting procedure itself.

V. APPLICATION STUDIES

In this section, we describe two different production system design problems that are used in this work to demonstrate (i) online knowledge extraction through the developed DSS, and (ii) the proposed FPM-NSGA-II algorithm.

The first problem is based on a machining cell model extended from the one presented in [55] and is called 'PSU2'. Although the model is hypothetical, it is realistically designed based on the authors' experience in working with real-world complex models and decision-making situations in industry. This model makes use of three so-called *selection objects*

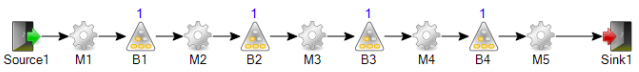

FIGURE 9. The 5S scalable simulation model.

in the simulation software FACTS Analyzer (or FACTS hereafter) [56] that can enable several design alternatives to co-exist in a model. By changing the values of the corresponding decision variables, the optimization module in FACTS can then alter the selection objects into various combinations. The analysis of such a model can then be conducted efficiently, by letting the optimization algorithm seek the optimal combination of design alternatives with their optimal settings, together with the optimal values of other decision variables (e.g. buffer spaces), in a single optimization run. The model is shown in Figure 8. The optimization problem formulation can be found in [15] with a complete description of the objectives and variables. Here it suffices to say that the model involves three objectives, minimize payoff time for investments (calculated as the ratio of total investment to profit per year), minimize work-in-process (WIP) and maximize throughput. The variables, their possible values and investments costs associated with them are shown in Table 1.

TABLE 1. Decision variables, their ranges and associated cost for calculating Payoff time.

Variable	Notation	Possible Values	Investment Cost ($)
Buffer	B_i	$\{1, 2, 3, \ldots, 50\}$	1000 / unit
Selection 1	S_1	$\{1, 2\}$	{500000, 900000}
No. of machines between B_2 and B_3	NM	$\{1, 2, 3, 4, 5, 6\}$	-
Selection 2 when $NM = 3$	M_3	$\{1, 2\}$	{500000, 250000}
Selection 2 when $NM = 6$	M_6	$\{1, 2\}$	{500000, 250000}
Selection 3	S_3	$\{1, 2, 3\}$	{100000, 150000, 120000}

The second problem under consideration is based on a scalable simulation model depicting simple flow lines that manufacture a single product type. Although the simulation model is simple, it has the interesting property of being scalable to any desired number of stations (operations). This allows the model to be used for benchmarking SBO algorithms and for understanding the behavior of other complex models under scaling. In this paper, we consider simulation models with four different number of stations: 5, 10, 15 and 20, hereafter referred to as 5S, 10S, 15S and 20S respectively. Between every two stations, there is one buffer whose capacity can be adjusted. Thus, for s stations, the total number of buffers is $s - 1$. Figure 9 illustrates the 5S scalable model which starts and ends with a source and a sink, and with the operations and buffers alternating in between.

An optimization problem is set-up for each of the scaled models with the same type of configuration. The variables for each station are the processing time, availability and mean time to repair. The buffer capacities are also included as variables. Table 2 shows the notation and possible values for each variable. The subscript i in the notation refers to the operation or buffer number, starting from $i = 1$ immediately after the source. The total number of variables is $n = 3s + (s - 1)$. Thus, 5S, 10S, 15S and 20S respectively involve 19, 39, 59 and 79 variables to be optimized. The optimization problem

TABLE 2. Variables for optimization of scalable models.

Object Type	Variable	Notation (Units)	Possible Values
Operation	Processing time	T_i (seconds)	$\{60, 65, 70, 75, 80\}$
Operation	Availability	A_i (percentage)	$\{90, 92, 94, 96, 98\}$
Operation	Mean Time To Repair	R_i (seconds)	$\{180, 210, 240, 270, 300\}$
Buffer	Capacity	B_i (numbers)	$\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$

has three objectives: (i) maximizing the throughput of the line, (ii) minimizing the total number of improvements in processing time, availability and mean time to repair from the base values of $T_i = 80$, $A_i = 90$ and $R_i = 300\ \forall\ i$, and (iii) minimizing the total number of buffer spaces required. An optimization of this problem should achieve the optimal trade-off between the throughput, number of improvement actions, and total buffer capacity.

VI. RESULTS AND DISCUSSION

Each experiment is composed of a simulation model, the experiment type and the parameter settings. To account for the stochastic nature of evolutionary algorithms, each experiment is replicated 10 times. The evaluations and the FPM rules from each generation of all optimization runs are stored in the database, which can be retrieved and plotted either during or after the optimization run. Both the original NSGA-II algorithm and the proposed FPM-NSGA-II algorithm use the parameter settings shown in Table 3. For the mutation probability, the $1/n$ rule-of-thumb, where n is the number of variables, is used.

In addition to the above mentioned NSGA-II parameters, FPM-NSGA-II involves two other parameters, namely, (i) the number of nearest neighbors k from the given reference point *refPoint* to be selected from the current non-dominated front *currentNDF*, and (ii) the support threshold ST for FPM rule generation. The value of k determines how well the decision maker's preferences are captured in the objective space. When too many solutions are selected, the preferences may not be adequately emphasized to guide search towards the reference point. On the other hand, selecting too few solutions may mean that no rules are found that meet the support threshold. Ideally, the two parameters should be adaptively controlled during runtime, with a high k and low ST at the beginning when relatively fewer solutions are non-dominated, and a low k and high ST towards the end when a majority of the population is expected to be non-dominated. This way the emphasis on the preferences is gradually increased with each generation. However, the control algorithm for an optimal balance between the two parameters is a matter for future research. In this paper, we keep both k and ST static thoruhgout the runtime. In our experiments, we found that using a low k and moderate ST allows preferences to be emphasized right from the start while still allowing rules satisfying the support threshold to be found. The following results were generated with $k = 20\%$ the size of *currentNDF* and $ST = 50\%$. Small changes to $k \in [20\%, 30\%]$ and $ST \in [50\%, 60\%]$ have shown to have negligible effect on the region and rate of convergence for all problems considered.

TABLE 3. Parameter settings for optimization experiments.

Parameter	Value	Parameter	Value
Number of replications	10	Max generations ($MaxGen$)	200
Population size (N)	100	Total number of evaluations	$20,000$
Polynomial mutation prob. (p_m)	$1/n$	Mutation distrib. index (η_m)	5
SBX Crossover probability (p_c)	0.9	Crossover distrib. index (η_c)	20
No. of kNN neighbors (k)	20%	Support threshold (ST)	50%

The results are generated as follows:

1) Set up experiment
   a) Set simulation model (5S, 10S, 15S, 20S, PSU2)
   b) Set experiment type (Online knowledge extraction / FPM-NSGA-II)
   c) Set optimization algorithm parameter settings
2) Run optimization with replications
3) Retrieve evaluations and FPM rules to generate plots

Each optimization run requires 200,000 evaluations (Population size × Max generations × Number of replications). The DSS developed in this work has an architecture that enables parallelization to distribute the evaluations of solutions on several computers, which reduces the total runtime. The implementation uses 100 computation nodes to parallely evaluate all solutions in a given generation, thus achieving a speed-up of nearly 100x. The generation of rules with FPM cannot be efficiently parallelized, and is hence performed on the local machine.

In Section VI-A, we demonstrate the online knowledge extraction process through the developed DSS from the solutions obtained by solving the above problems with the original NSGA-II algorithm. In this first scenario, the obtained rules apply to the whole non-dominated front and therefore can be interpreted as `IF 'non-dominated', THEN 'rule'`. Section VI-B presents how the FPM-NSGA-II algorithm uses the same DSS for online knowledge extraction and incorporates that knowledge to converge towards a specified reference point, thus meeting the decision maker's preferences. Thus, the second scenario allows finding rules that apply to a preferred region of the Pareto-optimal front, without having to generate a representation for the whole front. These rules can be interpreted as `IF 'preferred', THEN 'rule'`.

A. VISUALIZATION OF FPM RULES

When the experiment type is set to 'Online knowledge extraction', the non-dominated fronts from all generations of a complete optimization are analyzed. The final non-dominated front contains the trade-off solutions obtained by the original NSGA-II algorithm. By applying FPM to these solutions, knowledge about what makes a solution perform well can be discovered.

In order to visualize the FPM rules obtained from multiple optimization runs, a visualization method called the *rule map* has been developed. Figure 10 shows the rule map for the final non-dominated front from all ten replications of the 5S model. The diagram shows, for each input parameter, the range of variable values described by the FPM rules.

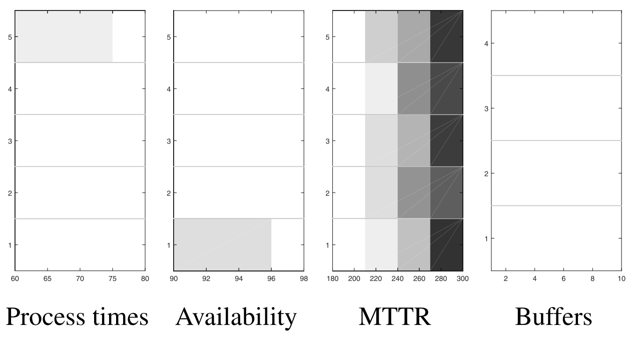

FIGURE 10. Rule map for final generation of 5S model.

The figure is divided into four parts representing the four groups of input parameters: processing time, availability, mean time to repair (MTTR), and buffer capacity. The y-axis represents the station/buffer number. For each rule in a replication, a translucent rectangle representing the value range of the rule and parameter is added to the diagram. Due to the stochastic nature of optimization, the rules from the last non-dominated front may vary from one replication to another. However, when rules from multiple replications are added on top of each other, the overlapping areas become darker indicating the most reliable values for each variable.

In Figure 10, it can be seen that MTTR variables follow several rules which makes their map darker. For process times, availability, and buffer capacities, there are very few rules, which means that FPM could not find many significant rules for these variables when considering the whole trade-off front.

Figure 11 shows the same analysis for the 10S variant of the scalable models. The behavior of the MTTR parameters is similar to the 5S model, but it has more significant rules for the other parameters. Rules obtained from the 15S model, presented in Figure 12, shows even more rules for process times, availability, and buffer capacities. Most processing time values are centered around 70 seconds. For the 20 station model, Figure 13 shows that the rules are approximately the same as for the 15S model. Such rule maps can help decision makers easily visualize the most frequent optimal values when dealing with a large number of variables. For example, the rule map of PSU2 in Figure 14 shows that B_1 and B_2 usually take values between 1 and 10, while B_3 and B_5 mostly take values between 10 and 30.

B. RESULTS FROM FPM-NSGA-II

The proposed preference guided evolutionary algorithm, FPM-NSGA-II, is now tested on the five application problems. The experimental setup is the same as described before, except that FPM-NSGA-II also requires a reference point for each simulation model. These reference points, which are all unattainable, are shown in Table 5. Our DSS can currently only handle one reference point as this is the most common use-case. The algorithm itself is not restricted with respect to the number of reference points.

TABLE 4. Top 20 significant rules from final generations of all five models.

Rule	5S		10S		15S		20S		PSU2	
	Expression	Count	Expression	Count	Expression	Count	Expression	Count	Expression	Count
1	$R_1 > 270$	6	$R_1 > 210$	8	$T_{11} < 75$	7	$R_{14} > 210$	6	$B_4 < 45$	3
2	$R_3 > 270$	5	$R_6 > 210$	6	$R_{10} > 210$	6	$R_{12} > 210$	5	$B_5 < 29$	2
3	$R_4 > 240$	5	$R_{10} > 210$	5	$R_{13} > 240$	5	$R_5 > 210$	5	$B_5 < 28$	2
4	$R_5 > 270$	5	$R_8 > 240$	4	$R_{14} > 210$	5	$B_{19} < 9$	5	$B_5 < 26$	2
5	$R_2 > 240$	4	$R_3 > 210$	4	$R_3 > 210$	5	$R_{11} > 210$	5	$B_1 < 8$	2
6	$R_5 > 210$	3	$R_{10} > 240$	4	$R_4 > 240$	5	$R_{20} > 210$	5	$S_1 < 4$	2
7	$R_4 > 270$	3	$R_6 > 240$	4	$T_6 < 75$	4	$R_{17} > 210$	5	$B_1 < 12$	2
8	$A_1 < 96$	2	$B_9 < 9$	3	$B_4 < 9$	4	$R_9 > 210$	5	$B_1 < 4$	1
9	$R_2 > 210$	2	$R_2 > 240$	3	$R_{13} > 210$	4	$T_{10} > 65$	4	$B_4 < 42$	1
10	$R_1 > 240$	2	$R_7 > 210$	3	$R_9 > 210$	4	$T_{12} > 65$	4	$B_1 < 9$	1
11	$R_3 > 210$	2	$R_8 > 210$	3	$R_1 > 210$	4	$R_{16} > 210$	4	$B_3 < 46$	1
12	$R_2 > 270$	2	$R_9 > 240$	3	$B_{12} < 9$	3	$R_{17} > 240$	4	$B_5 < 39$	1
13	$R_5 > 240$	2	$R_2 > 210$	3	$T_{10} < 75$	3	$A_{20} < 96$	4	$S_1 < 13$	1
14	$R_3 > 240$	2	$B_1 < 9$	3	$R_{11} > 210$	3	$R_4 > 210$	4	$B_1 < 14$	1
15	$T_5 < 75$	1	$A_1 < 96$	3	$R_{12} > 210$	3	$B_8 < 9$	4	$B_3 < 35$	1
16	$R_4 > 210$	1	$A_8 > 92$	2	$R_2 > 210$	3	$R_1 > 210$	4	$B_4 < 43$	1
17	$R_1 > 210$	1	$T_3 > 65$	2	$R_9 > 240$	3	$R_{15} > 210$	4	$S_1 < 5$	1
18			$R_4 > 240$	2	$B_3 < 9$	3	$T_{20} > 65$	4	$B_1 < 20$	1
19			$R_5 > 240$	2	$A_{10} > 92$	3	$T_3 < 75$	4	$B_3 < 48$	1
20			$T_5 < 75$	2	$R_{12} > 240$	3	$B_5 < 9$	4	$B_4 < 47$	1

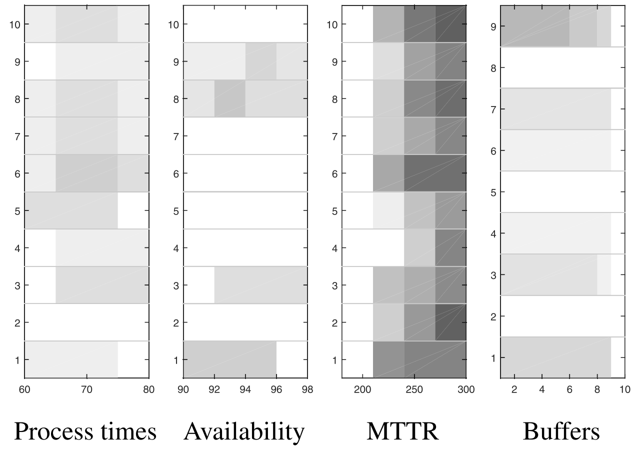

FIGURE 11. Rule map for final generation of 10S model.

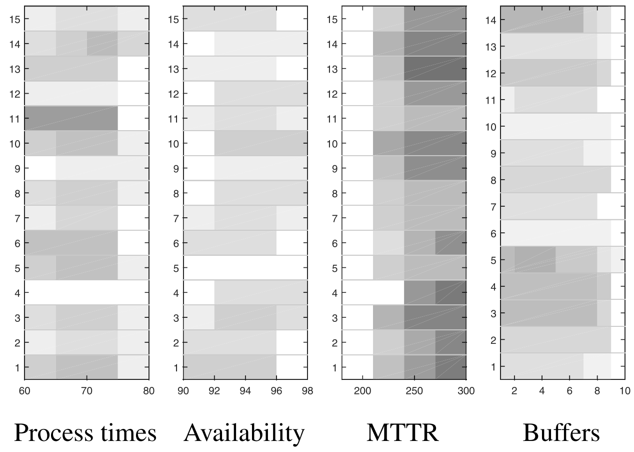

FIGURE 12. Rule map for final generation of 15S model.

The top 20 significant rules obtained from the last generation of FPM-NSGA-II are shown in Table 6. The table also shows the number of times the rules appear in 10 replications of FPM-NSGA-II. Note that due to the specification of a

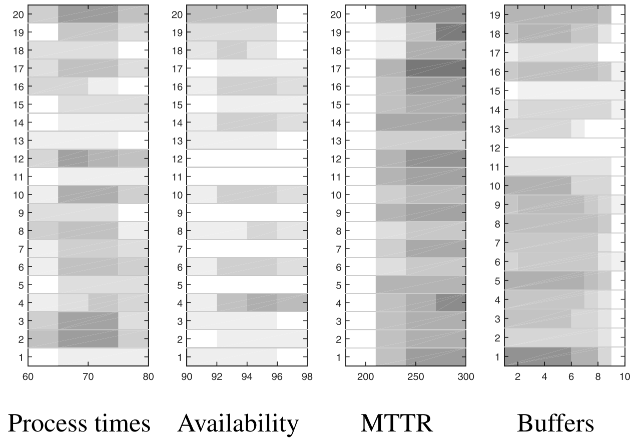

FIGURE 13. Rule map for final generation of 20S model.

TABLE 5. Reference points (unattainable) of all five models used in FPM-NSGA-II.

Model	Objective 1	Objective 2	Objective 3
	Total improvements	Total buffers	Throughput
5S	15	28	50
10S	20	60	40
15S	35	100	40
20S	25	120	40
	Work-in-process	Payoff Time	Throughput
PSU2	15	3	5.5

preferred region (through reference points), the obtained rules are very different from those in Table 4.

Similar to the previous sub-section, the rule maps can be generated for all five models. These are shown in Figures 15-18. Note that there are more distinct and darker regions in these figures than in Figures 10-14. This is a graphical manifestation of the difference in rules observed in Table 6. The visual nature of rule maps helps decision makers to easily make sense of multiple variables at a single glance. For example, Figure 17 shows that processing times

TABLE 6. Top 20 significant rules from final generations of all five models using FPM-NSGA-II.

Rule	5S		10S		15S		20S		PSU2	
	Expression	Count	Expression	Count	Expression	Count	Expression	Count	Expression	Count
1	$T_4 < 70$	6	$T_2 > 70$	5	$T_{15} > 70$	5	$T_9 > 75$	3	$S_1 < 2$	5
2	$T_2 < 70$	5	$T_8 > 70$	4	$T_6 > 70$	4	$T_3 > 75$	3	$N > 5$	4
3	$B_4 > 5$	5	$B_2 > 4$	4	$B_2 > 5$	3	$T_{16} > 75$	3	$B_5 < 3$	3
4	$B_1 < 9$	4	$B_1 < 5$	3	$T_8 > 70$	3	$B_3 < 6$	3	$B_3 < 9$	2
5	$T_3 < 70$	4	$B_5 > 8$	3	$B_7 < 9$	3	$B_{14} < 9$	2	$B_4 < 10$	2
6	$T_1 < 70$	4	$T_3 > 65$	3	$B_{10} > 7$	3	$B_{12} > 6$	2	$B_5 < 2$	2
7	$T_5 < 70$	4	$T_5 > 70$	3	$B_4 > 8$	3	$B_{17} > 3$	2	$B_4 < 13$	2
8	$B_4 < 8$	3	$B_1 < 7$	3	$T_{10} > 70$	3	$B_9 > 3$	2	$B_3 < 7$	2
9	$B_1 < 7$	3	$T_6 > 70$	3	$B_{11} > 6$	2	$B_9 < 7$	2	$B_3 < 22$	1
10	$B_4 < 9$	3	$T_9 > 70$	3	$B_{12} < 8$	2	$B_{12} > 5$	2	$N > 4$	1
11	$B_2 > 6$	3	$B_1 > 4$	3	$B_{14} > 5$	2	$B_{14} > 3$	2	$B_3 < 13$	1
12	$A_3 < 92$	3	$B_6 > 5$	3	$B_2 < 8$	2	$B_7 < 9$	2	$B_5 < 7$	1
13	$A_2 < 92$	2	$T_4 > 70$	3	$B_7 > 6$	2	$A_1 < 96$	2	$B_4 < 19$	1
14	$B_3 > 7$	2	$T_5 > 75$	3	$T_3 > 70$	2	$T_6 > 75$	2	$B_1 < 2$	1
15	$B_3 < 9$	2	$T_7 > 70$	3	$T_7 > 75$	2	$T_8 > 75$	2	$B_3 > 7$	1
16	$B_4 > 6$	2	$B_2 < 7$	2	$B_3 < 4$	2	$B_1 > 2$	2	$B_4 > 5$	1
17	$B_3 > 8$	2	$B_6 > 7$	2	$B_4 < 8$	2	$B_5 < 3$	2	$B_4 < 11$	1
18	$B_3 > 6$	2	$T_1 > 70$	2	$T_{13} > 70$	2	$T_{11} > 75$	2	$B_1 < 3$	1
19	$A_1 < 92$	2	$T_{10} > 70$	2	$A_4 > 94$	2	$T_{12} > 70$	2	$B_3 < 10$	1
20	$A_2 < 94$	2	$B_3 > 5$	2	$T_9 > 70$	2	$B_5 > 6$	2	$S_1 < 14$	1

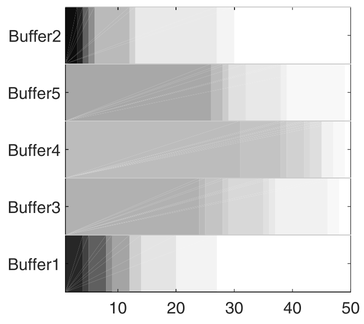

FIGURE 14. Rule map for final generation of PSU2 model.

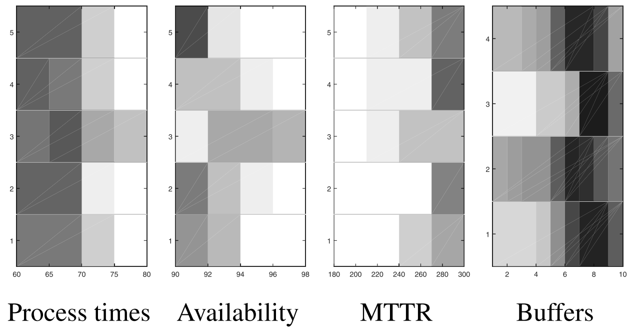

FIGURE 15. Rule map for final generation of 5S model using FPM-NSGA-II.

usually have to be high while availability values should be relatively low for solutions close to the reference point of 15S model. More specifically, the decision maker can also see that B_4, B_7 and B_{13} should be close to their upper bounds (10). The relative darkness of the regions tell how important the variables and their values are.

Next, we show the non-dominated solutions obtained using FPM-NSGA-II. In Figures 19-23, '$\ast$' represents the reference point and the solutions obtained by FPM-NSGA-II are shown in blue. For the sake of comparison, the blue solutions are overlaid on the trade-off solutions obtained from original NSGA-II. The figures show that the FPM-NSGA-II solutions have converged close to the Pareto-optimal front and are also

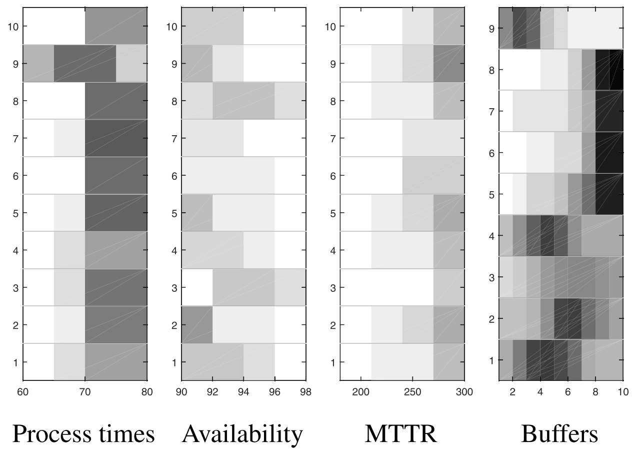

FIGURE 16. Rule map for final generation of 10S model using FPM-NSGA-II.

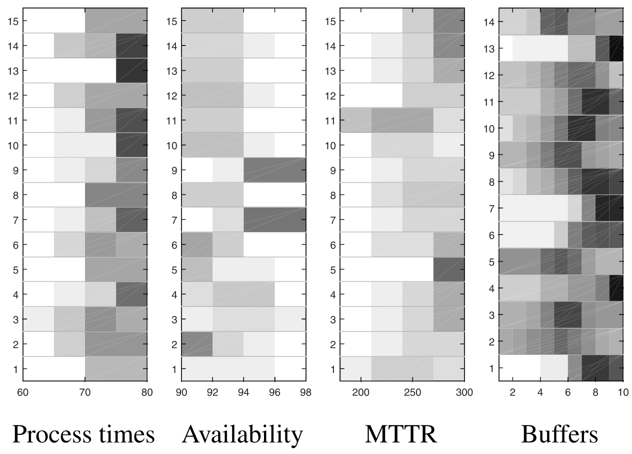

FIGURE 17. Rule map for final generation of 15S model using FPM-NSGA-II.

clustered close to the reference point in each case. While there are other reference point based algorithms in the literature, the unique feature of FPM-NSGA-II is that in addition to solutions close to the preferred reference point, the algorithm

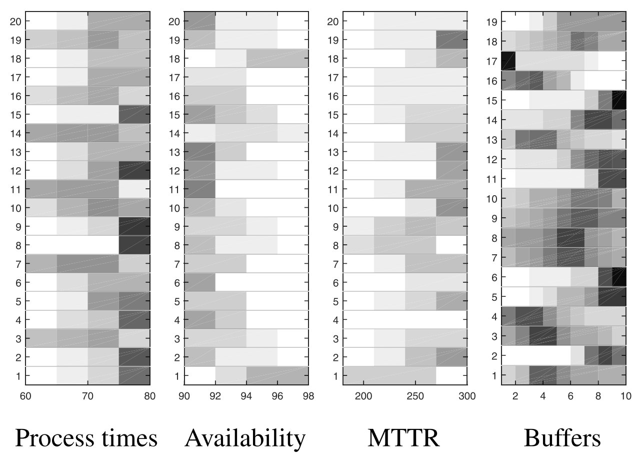

FIGURE 18. Rule map for final generation of 20S model using FPM-NSGA-II.

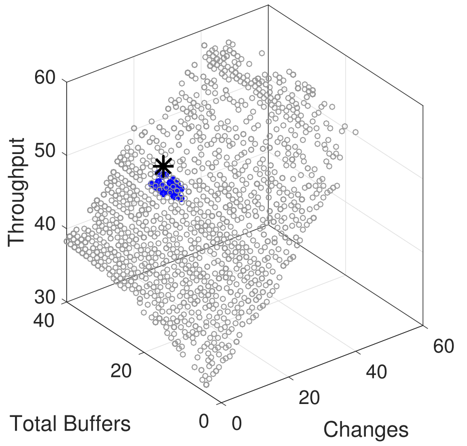

FIGURE 19. Non-dominated front (blue solutions) for 5S model obtained using FPM-NSGA-II.

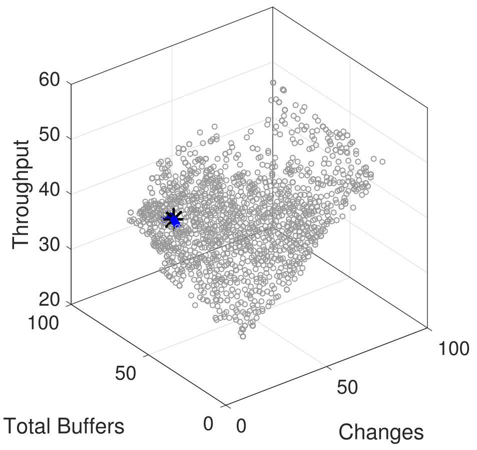

FIGURE 20. Non-dominated front (blue solutions) for 10S model obtained using FPM-NSGA-II.

can also generate rules that describe these solutions. In the classical sense, decision making is almost always performed in the objective space. FPM-NSGA-II allows decision makers to simultaneously understand the implications of their decisions in the decision space. This is not possible with any of the reference-point based evolutionary algorithms that exist in the literature.

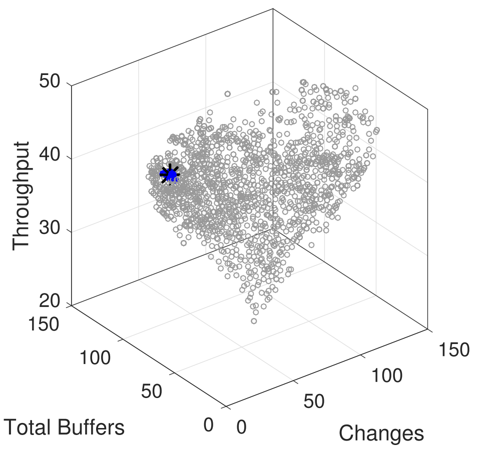

FIGURE 21. Non-dominated front (blue solutions) for 15S model obtained using FPM-NSGA-II.

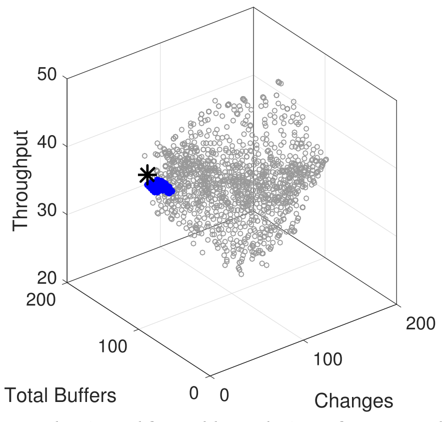

FIGURE 22. Non-dominated front (blue solutions) for 20S model obtained using FPM-NSGA-II.

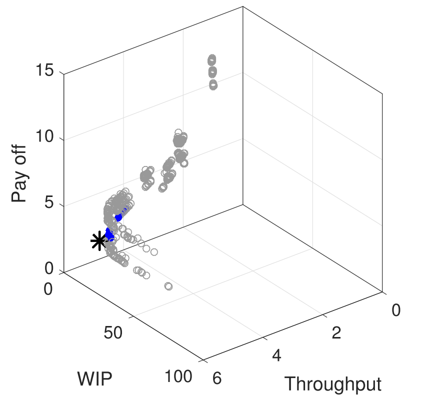

FIGURE 23. Non-dominated front (blue solutions) for PSU2 model obtained using FPM-NSGA-II.

VII. CONCLUSIONS

Although a combination of simulation and multi-objective optimization has previously been shown to be effective for supporting decision making in manufacturing, higher quality decisions can be assured when the optimal solutions are supplemented with the knowledge about what makes them optimal in terms of the variable values. Such knowledge can be automatically extracted using data mining techniques. In this paper, we first extended an existing DSS, that combines a discrete event simulation engine and an optimization module, with a data mining module that implements the recently proposed Flexible Pattern Mining algorithm. The DSS is capable of replicating optimization runs on a parallel architecture and online knowledge extraction. Also part of the DSS is an Experiment Browser for visualizing the solutions and the obtained knowledge. In order to visualize the rules generated by FPM, we developed the concept of rule map, which allows FPM rules from multiple replications of the experiments to be shown on a single plot. The decision maker can obtain an overview of knowledge pertaining to the trade-off solutions at a glance.

The paper also proposes a new preference guided multi-objective optimization algorithm called FPM-NSGA-II. The new algorithm incorporates the decision maker's preferences by (i) selecting solutions that are close to the specified reference point in intermediate generations, (ii) using the DSS to obtain FPM rules specific to the selected solutions, and (iii) adding probabilistic constraints and modifying the crossover/mutation operators of NSGA-II to promote the generation of solutions that satisfy the rules. In this way, the algorithm is able to converge towards a preferred region on the Pareto-optimal front. While there are several preference-based evolutionary algorithms in the literature, the advantage of FPM-NSGA-II is that in addition to the set of preferred solutions, the algorithm also generates knowledge that describe those solutions in the form of explicit and easily interpretable rules.

The DSS for online knowledge extraction and FPM-NSGA-II, are applied to SBO problems involving discrete event simulation models of realistic production systems. We observed that knowledge extracted from the complete trade-off front can be different from that pertaining to a specific region of the trade-off front. In either case, the rule maps were found to be an effective way of communicating the extracted knowledge to the decision maker. The results shows that FPM-NSGA-II is able to extract relevant knowledge in an automatic and iterative manner to guide the evolutionary algorithm towards the decision maker's preferences. In each application study, we found that the rules generated by FPM-NSGA-II throw new light on understanding the implications of choosing a solution from the preferred optimal set. Such an approach to decision making is believed to provide the decision maker with additional information that cannot be obtained by any existing industrial practice today.

REFERENCES

[1] European Commission. (2013). *Towards Knowledge Driven Re-Industrialisation*. [Online]. Available: http://ec.europa.eu/growth/tools-databases/newsroom/cf/itemdetail.cfm?item_id=7507&lang=en

[2] N. Slack and M. Lewis, *Operations Strategy*. London, U.K.: Pearson, 2002.

[3] M. Jahangirian, T. Eldabi, A. Naseer, L. K. Stergioulas, and T. Young, "Simulation in manufacturing and business: A review," *Eur. J. Oper. Res.*, vol. 203, no. 1, pp. 1–13, 2010.

[4] S. Thiede, Y. Seow, J. Andersson, and B. Johansson, "Environmental aspects in manufacturing system modelling and simulation—State of the art and research perspectives," *CIRP J. Manuf. Sci. Technol.*, vol. 6, no. 1, pp. 78–87, Jan. 2013.

[5] A. Negahban and J. S. Smith, "Simulation for manufacturing system design and operation: Literature review and analysis," *J. Manuf. Syst.*, vol. 33, no. 2, pp. 241–261, Apr. 2014.

[6] D. Mourtzis, N. Papakostas, D. Mavrikios, S. Makris, and K. Alexopoulos, "The role of simulation in digital manufacturing: Applications and outlook," *Int. J. Comput. Integr. Manuf.*, vol. 28, no. 1, pp. 3–24, Jan. 2015.

[7] L. Pehrsson, "Manufacturing management and decision support using simulation-based multi-objective optimisation," Ph.D. dissertation, Volvo Car Corp., De Montfort Univ., Sweden Univ. Skövde, Skövde, Sweden, Nov. 2013. [Online]. Available: https://www.dora.dmu.ac.uk/xmlui/handle/2086/9697

[8] S. Amaran, N. V. Sahinidis, B. Sharda, and S. J. Bury, "Simulation optimization: A review of algorithms and applications," *4OR*, vol. 12, no. 4, pp. 301–333, Nov. 2014.

[9] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P. N. Suganthan, and Q. Zhang, "Multiobjective evolutionary algorithms: A survey of the state of the art," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 32–49, Mar. 2011.

[10] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*, 2nd ed. Chichester, U.K.: Wiley, 2004.

[11] F. Goulart and F. Campelo, "Preference-guided evolutionary algorithms for many-objective optimization," *Inf. Sci.*, vol. 329, pp. 236–255, Feb. 2016.

[12] Y. Li, J. Wang, D. Zhao, G. Li, and C. Chen, "A two-stage approach for combined heat and power economic emission dispatch: Combining multi-objective optimization with integrated decision making," *Energy*, vol. 162, pp. 237–254, Nov. 2018.

[13] M. Zhang and Y. Li, "Multi-objective optimal reactive power dispatch of power systems by combining classification-based multi-objective evolutionary algorithm and integrated decision making," *IEEE Access*, vol. 8, pp. 38198–38209, 2020.

[14] A. Wierzbicki, M. Makowski, and J. Wessels, *Model-Based Decision Support Methodology With Environmental Applications*. Dordrecht, The Netherlands: Kluwer, 2000.

[15] I. Karlsson, A. H. C. Ng, A. Syberfeldt, and S. Bandaru, "An interactive decision support system using simulation-based optimization and data mining," in *Proc. Winter Simulation Conf. (WSC)*, Piscataway, NJ, USA, Dec. 2015, pp. 2112–2123.

[16] S. Bandaru, A. H. C. Ng, and K. Deb, "Data mining methods for knowledge discovery in multi-objective optimization: Part A—Survey," *Expert Syst. Appl.*, vol. 70, pp. 139–159, Mar. 2017.

[17] J. P. L. Faucher, A. M. Everett, and R. Lawson, "Reconstituting knowledge management," *J. Knowl. Manage.*, vol. 12, no. 3, pp. 3–16, May 2008.

[18] K. Sugimura, S. Obayashi, and S. Jeong, "Multi-objective optimization and design rule mining for an aerodynamically efficient and stable centrifugal impeller with a vaned diffuser," *Eng. Optim.*, vol. 42, no. 3, pp. 271–293, Mar. 2010.

[19] C. Dudas, A. H. C. Ng, L. Pehrsson, and H. Boström, "Integration of data mining and multi-objective optimisation for decision support in production systems development," *Int. J. Comput. Integr. Manuf.*, vol. 27, no. 9, pp. 824–839, Sep. 2014.

[20] C. Dudas, A. H. C. Ng, and H. Boström, "Post-analysis of multi-objective optimization solutions using decision trees," *Intell. Data Anal.*, vol. 19, no. 2, pp. 259–278, Apr. 2015.

[21] K. Sugimura, S. Obayashi, and S. Jeong, "Multi-objective design exploration of a centrifugal impeller accompanied with a vaned diffuser," in *Proc. Fluids Eng. Division Summer Meeting*, vol. 42894, 2007, pp. 939–946.

[22] K. Sugimura, S. Jeong, S. Obayashi, and T. Kimura, "Kriging-model-based multi-objective robust optimization and trade-off-rule mining using association rule with aspiration vector," in *Proc. IEEE Congr. Evol. Comput.*, May 2009, pp. 522–529.

[23] L. Pehrsson, I. Karlsson, and A. H. C. Ng, "Towards automated multi-objective rule extraction," in *Proc. 30th Eur. Simulation Modelling Conf.* Las Palmas, Spain: Univ. Las Palmas, Oct. 2016, pp. 64–68.

[24] P. Korhonen, J. Wallenius, and S. Zionts, "A bargaining model for solving the multiple criteria problem," in *Proc. Multiple Criteria Decis. Making Theory Appl.* Berlin, Germany: Springer, 1980, pp. 178–188.

[25] B. Mareschal and J.-P. Brans, "Geometrical representations for MCDA," *Eur. J. Oper. Res.*, vol. 34, no. 1, pp. 69–77, Feb. 1988.

[26] A. Oyama, T. Nonomura, and K. Fujii, "Data mining of Pareto-optimal transonic airfoil shapes using proper orthogonal decomposition," *J. Aircr.*, vol. 47, no. 5, pp. 1756–1762, Sep. 2010.

[27] A. Oyama, P. Verburg, T. Nonomura, H. Hoeijmakers, and K. Fujii, "Flow field data mining of Pareto-optimal airfoils using proper orthogonal decomposition," in *Proc. 48th AIAA Aerosp. Sci. Meeting Including New Horizons Forum Aerosp. Expo.*, Jan. 2010, p. 1140.

[28] K. Deb and A. Srinivasan, "Innovization: Innovating design principles through optimization," in *Proc. 8th Annu. Conf. Genet. Evol. Comput. (GECCO)*, New York, NY, USA, 2006, pp. 1629–1636, doi: 10.1145/1143997.1144266.

[29] S. Bandaru and K. Deb, "Automated discovery of vital knowledge from Pareto-optimal solutions: First results from engineering design," in *Proc. IEEE Congr. Evol. Comput.*, Jul. 2010, pp. 1–8.

[30] S. Bandaru and K. Deb, "Towards automating the discovery of certain innovative design principles through a clustering-based optimization technique," *Eng. Optim.*, vol. 43, no. 9, pp. 911–941, Sep. 2011.

[31] S. Bandaru and K. Deb, "Automated innovization for simultaneous discovery of multiple rules in bi-objective problems," in *Proc. Int. Conf. Evol. Multi-Criterion Optim.* Berlin, Germany: Springer, 2011, pp. 1–15.

[32] S. Bandaru and K. Deb, "A dimensionally-aware genetic programming architecture for automated innovization," in *Evolutionary Multi-Criterion Optimization* (Lecture Notes in Computer Science), R. C. Purshouse, P. J. Fleming, C. M. Fonseca, S. Greco, and J. Shaw, Eds. Berlin, Germany: Springer, Mar. 2013, pp. 513–527.

[33] K. Deb, S. Bandaru, D. Greiner, A. Gaspar-Cunha, and C. C. Tutum, "An integrated approach to automated innovization for discovering useful design principles: Case studies from engineering," *Appl. Soft Comput.*, vol. 15, pp. 42–56, Feb. 2014.

[34] B. Xin, L. Chen, J. Chen, H. Ishibuchi, K. Hirota, and B. Liu, "Interactive multiobjective optimization: A review of the state-of-the-art," *IEEE Access*, vol. 6, pp. 41256–41279, 2018.

[35] R. Wang, R. C. Purshouse, I. Giagkiozis, and P. J. Fleming, "The iPICEA-g: A new hybrid evolutionary multi-criteria decision making approach using the brushing technique," *Eur. J. Oper. Res.*, vol. 243, no. 2, pp. 442–453, Jun. 2015.

[36] K. Li, M. Liao, K. Deb, G. Min, and X. Yao, "Does preference always help? A holistic study on preference-based evolutionary multiobjective optimization using reference points," *IEEE Trans. Evol. Comput.*, vol. 24, no. 6, pp. 1078–1096, Dec. 2020.

[37] K. Deb and J. Sundar, "Reference point based multi-objective optimization using evolutionary algorithms," in *Proc. 8th Annu. Conf. Genet. Evol. Comput. (GECCO)*, New York, NY, USA, 2006, pp. 635–642.

[38] K. Deb, A. Sinha, P. J. Korhonen, and J. Wallenius, "An interactive evolutionary multiobjective optimization method based on progressively approximated value functions," *IEEE Trans. Evol. Comput.*, vol. 14, no. 5, pp. 723–739, Oct. 2010.

[39] L. Thiele, K. Miettinen, P. Korhonen, and J. Molina, "A preference-based evolutionary algorithm for multi-objective optimization," *Evol. Comput.*, vol. 17, no. 3, pp. 411–436, Sep. 2009.

[40] A. B. Ruiz, R. Saborido, and M. Luque, "A preference-based evolutionary algorithm for multiobjective optimization: The weighting achievement scalarizing function genetic algorithm," *J. Global Optim.*, vol. 62, no. 1, pp. 101–129, 2015.

[41] J. Molina, L. V. Santana, A. G. Hernández-Díaz, C. A. C. Coello, and R. Caballero, "g-dominance: Reference point based dominance for multiobjective metaheuristics," *Eur. J. Oper. Res.*, vol. 197, no. 2, pp. 685–692, Sep. 2009.

[42] L. B. Said, S. Bechikh, and K. Ghédira, "The r-dominance: A new dominance relation for interactive evolutionary multicriteria decision making," *IEEE Trans. Evol. Comput.*, vol. 14, no. 5, pp. 801–818, Oct. 2010.

[43] A. Mohammadi, M. N. Omidvar, X. Li, and K. Deb, "Integrating user preferences and decomposition methods for many-objective optimization," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2014, pp. 421–428.

[44] K. Li, R. Chen, G. Min, and X. Yao, "Integration of preferences in decomposition multiobjective optimization," *IEEE Trans. Cybern.*, vol. 48, no. 12, pp. 3359–3370, Dec. 2018.

[45] R. S. Michalski, "Learnable evolution model: Evolutionary processes guided by machine learning," *Mach. Learn.*, vol. 38, nos. 1–2, pp. 9–40, Jan. 2000.

[46] L. Jourdan, D. Corne, D. Savic, and G. Walters, "Preliminary investigation of the 'learnable evolution model' for faster/better multiobjective water systems design," in *Evolutionary Multi-Criterion Optimization*. Berlin, Germany: Springer, Mar. 2005, pp. 841–855.

[47] A. Zhou, Q. Zhang, Y. Jin, E. Tsang, and T. Okabe, "A model-based evolutionary algorithm for bi-objective optimization," in *Proc. IEEE Congr. Evol. Comput.*, vol. 3, Sep. 2005, pp. 2568–2575.

[48] D. K. Saxena, J. A. Duro, A. Tiwari, K. Debm, and Q. Zhang, "Objective reduction in many-objective optimization: Linear and nonlinear algorithms," *IEEE Trans. Evol. Comput.*, vol. 17, no. 1, pp. 77–99, Feb. 2013.

[49] S. Greco, B. Matarazzo, and R. Słowiński, "Dominance-based rough set approach to interactive multiobjective optimization," in *Multiobjective Optimization* (Lecture Notes in Computer Science), vol. 5252, J. Branke, K. Deb, K. Miettinen, and R. Słowiński, Eds. Berlin, Germany: Springer, 2008, pp. 121–155.

[50] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[51] S. Bandaru, A. H. C. Ng, and K. Deb, "Data mining methods for knowledge discovery in multi-objective optimization: Part B—New developments and applications," *Expert Syst. Appl.*, vol. 70, pp. 119–138, Mar. 2017.

[52] R. Agrawal and R. Srikant, "Mining sequential patterns," in *Proc. 11th Int. Conf. Data Eng.*, 1995, pp. 3–14.

[53] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proc. 20th Conf. Very Large Databases (VLDB)*, vol. 1215, 1994, pp. 487–499.

[54] P.-N. Tan, M. Steinbach, A. Karpatne, and V. Kumar, *Introduction to Data Mining*. London, U.K.: Pearson, 2019.

[55] A. H. C. Ng, J. Bernedixen, M. U. Moris, and M. Jägstam, "Factory flow design and analysis using internet-enabled simulation-based optimization and automatic model generation," in *Proc. Winter Simulation Conf. (WSC)*, Dec. 2011, pp. 2176–2188.

[56] A. H. C. Ng, H. Grimm, T. Lezama, A. Persson, M. Andersson, and M. Jägstam, "Web services for metamodel-assisted parallel simulation optimization," in *Proc. IAENG Int. Conf. Internet Comput. Web Services (ICICWS)*, Hong Kong, Mar. 2007, pp. 879–885.

INGEMAR KARLSSON received the B.Sc. degree in computer science, the M.Sc. degree in automation engineering, and the Ph.D. degree from the University of Skövde, in 2006, 2011, and 2018, respectively. He is a Senior Software Developer with the University of Skövde. His research interests include simulation-based optimization, data mining, augmented reality, and cloud technologies.

SUNITH BANDARU (Senior Member, IEEE) received the bachelor's degree in mechanical engineering from Jawaharlal Nehru Technological University, India, in 2006, and the master's and Ph.D. degrees from IIT Kanpur, India, in 2008 and 2013, respectively. He is an Associate Professor of production and automation engineering with the University of Skövde, Sweden. He is also the Chair of the Committee for Research Education in Informatics. His research interests include the intersection of evolutionary computation, machine learning, and data mining, where he develops methods for knowledge discovery in multi-objective optimization.

AMOS H. C. NG received the Ph.D. degree in computing sciences and engineering from De Montfort University, Leicester, U.K. He is a Professor of production and automation engineering with the University of Skövde, Sweden. He is also a Visiting Professor with the Department of Civil and Industrial Engineering, Uppsala University, Sweden, and the CEO of Evoma AB. His main research interests include applying simulation, multi-objective optimization, and prescriptive analytics for manufacturing/service/health-care systems design, analysis, and improvement.