An Improved MOEA/D Algorithm for the Carbon Black Production Line Static and Dynamic Multiobjective Scheduling Problem

Yao Wang
Key Laboratory of Data Analytics and Optimization for Smart Industry (Northeastern University), Ministry of Education, China
WYfromNEU@163.com

Zhiming Dong
Liaoning Engineering Laboratory of operation Analytics and Optimization for Smart Industry Northeastern University
Shenyang, China
dongzm@stumail.neu.edu.cn

Tenghui Hu
Liaoning Key Laboratory of Manufacturing System and Logistics Northeastern University
Shenyang, China
450332679@qq.com

Xianpeng Wang
Institute of Industrial & Systems Engineering Northeastern University
Shenyang, China
wangxianpeng@ise.neu.edu.cn

***Abstract*—The make-to-order (MTO) manufacturers generally make production plans based on orders, which can help enterprises effectively avoid market risks, reduce market pressure and improve competitiveness. However, due to the characteristics of MTO production mode, the order static scheduling problem and rush order dynamic rescheduling problem have become more and more important for these MTO manufacturers. Therefore, in this paper, we take the packaging production line of a typical carbon black production enterprise as the research background to study the carbon black production line static and dynamic multiobjective scheduling problem. Firstly, multiobjective optimization models of both order static scheduling and rush order dynamic rescheduling are established. Then the improved MOEA/D algorithm combined the heuristic algorithm based on heuristic rules and discrete dynamic local search is developed to solve these two models. Based on the actual production data, eight instances of order static scheduling problems of different scales and four instances of rush order dynamic rescheduling problems of different scales are constructed respectively. Experimental results illustrate that the improved MOEA/D is effective and superior in solving these two problems.**

Keywords—MTO, order static scheduling, rush order dynamic rescheduling, multiobjective optimization, MOEA/D

I. INTRODUCTION

Make-to-order (MTO) production mode refers to the arrangement of enterprise production plans and organization of enterprise production activities according to the order requirements put forward by customers. Due to the advantages of reducing the enterprise inventory overhang and avoiding market risks, MTO production mode has become one of the important ways of enterprise production. However, the characteristics of MTO production mode, such as the variety of products, large fluctuations of order demand, large number of rush orders and complex production process, also make the MTO manufacturers encounter many management difficulties, which increases the operational risk of the enterprises. Therefore, in this paper, we take the carbon black packaging line of a carbon black manufacturing enterprise with MTO production mode as the research object to study order static scheduling problem (OSSP) and rush order dynamic rescheduling problem (RODRP).

In recent years, there have been many researches focused on static scheduling problem and dynamic rescheduling problem. Brucker *et al.* [1] proposed a fast branch and bound algorithm to solve the workshop scheduling problem, and verified the effectiveness of the proposed algorithm on 10 test problems. Dell' Amico and Trubian [2] applied tabu search to the job-shop scheduling problem. Pan *et al.* [3] proposed a new discrete differential evolution algorithm to solve the flow shop production scheduling with makespan criterion scheduling problem. For dynamic rescheduling problem, Vieira *et al*. [4] defined and classified the strategies, policies and methods in rescheduling manufacturing system, and gave a detailed overview of rescheduling manufacturing system. Moratori *et al.* [5] considered how to insert new rush orders into the current scheduling plan while ensuring the reasonable performance level and workshop production stability of the rescheduling plan, and designed an effective matching strategy. Wang *et al.* [6] established a multiobjective rescheduling optimization model to ensure a single machine rescheduling problem with preventive maintenance in the planned period under the condition of single insertion disturbance.

In this paper, we take the packaging production line of a typical carbon black production enterprise as the research background, to study the OSSP and the RODRP. The main works of this paper are as follows.

- Based on the actual investigation and analysis, the multiobjective static scheduling optimization model (MOSSOM) and multiobjective dynamic rescheduling optimization model (MODROM) are established respectively.

This research was supported by the National Key Research and Development Program of China (2018YFB1700400), the Fund for the National Natural Science Foundation of China (61573086), the Major Program of National Natural Science Foundation of China (71790614), the Major International Joint Research Project of the National Natural Science Foundation of China (71520107004), and the 111 Project (B16009).

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

- Two heuristic algorithms based on heuristic rules and discrete dynamic local search (DDLS) are designed according to the characteristics of the two problems respectively, which are used to improve the population initialization method of the original MOEAD.
- An improved MOEA/D is proposed to solve the two problems. Based on the actual production data, eight instances of the OSSPs of different scales and four instances of the RODRPs of different scales are separately constructed. With comparison to other algorithms, the superiority of the improved MOEA/D are verified in solving these two problems.

The rest of the paper is organized as follows. In section II, the MOSSOM and MODROM are established respectively. The heuristic algorithms based on heuristic rules and DDLS for OSSP and RODRP are present in Section III. The Section IV introduces the improved MOEA/D algorithm in detail. Numerical experiments are presented and discussed in Section V. Section VI concludes the paper.

II. Problem Statement

A. Order Static Multiobjective Scheduling Problem

The OSSP can be described as follows. *N* orders are allocated to *M* manufacturing units for production. The production speed of each unit is different for different planned products. In the production process, if the planned products of two adjacent orders are different in the same unit, the unit needs to switch from the previous planned product to the later one. The problem to be solved is how to reasonably allocate *N* orders to *M* manufacturing units and determine the production sequence of all orders, so as to achieve the optimal target.

TABLE I. The Parameters of the Order Static Scheduling Problem Model

Parameter	Description
M	Manufacturing unit set
N	Order set
K	Planned product set
t_{ik}	Per unit production time of planned product k on unit i, $k\in K,i\in M$
r_j	Planned product of order j, $r_j\in K,j\in N$
o_j	Weight of order j, $j\in N$
h_{imn}	Switching time of planned product m and n on unit i, $m,n\in K,i\in M$
S_i	Order set on unit i, $i\in M$
g_i	Switching times on unit i, $i\in M$
L_j	Delay time of order j, $j\in N$
q_j	Unit to which order j is assigned, $j\in N$
d_j	Due time of order j, $j\in N$
c_i	Production time of unit i, $i\in M$
w_1,w_2	Weight coefficients of average production time of all units and total delay time of all orders
p_1	Penalty coefficient of delay time of all orders

Based on the actual investigation and analysis of the OSSP, the MOSSOM is established. The parameters used in the model are shown in Table I. The decision variables in the model are described as follows.

- $x_{ij}=\begin{cases}1, \text{ if order } i \text{ is assigned to unit } j\\ 0, \text{ otherwise} \qquad j\in N, i\in M\end{cases}$
- $z_{ijh}=\begin{cases}1, \text{ if two adjacent orders } j \text{ and } h \text{ on unit } i\\ \quad \text{have the same planned product}\\ 0, \text{ otherwise} \qquad j,h\in S_i, i\in M\end{cases}$
- $v_{ijh}=\begin{cases}1, \text{ if order } h \text{ is produced before the}\\ \quad \text{adjacent order } j \text{ on unit } i\\ 0, \text{ otherwise} \qquad j,h\in S_i, i\in M\end{cases}$
- b_{ij} is the starting time of order j on unit i, $j\in S_i, i\in M$
- e_{ij} is the end time of order j on unit i, $j\in S_i, i\in M$

Based on the above parameters and decision variables, the mathematic model of the OSSP can be established as follows.

$$\min\ f_1=\sum\nolimits_{i\in M} g_i \tag{1}$$

$$\min\ f_2=w_1\left(\sum\nolimits_{i\in M} c_i\right)/\left|M\right|+w_2 p_1\sum\nolimits_{j\in N} L_j \tag{2}$$

$$\min\ f_3=\sqrt{\left(\sum\nolimits_{i\in M}\left(c_i-\left(\sum\nolimits_{l\in M} c_l\right)/\left|M\right|\right)^2\right)/\left|M\right|} \tag{3}$$

s.t.

$$\sum\nolimits_{i\in M}\sum\nolimits_{j\in N} x_{ij}=\left|N\right| \tag{4}$$

$$\sum\nolimits_{i\in M} x_{ij}=1,\forall j\in N \tag{5}$$

$$\sum\nolimits_{j\in N} x_{ij}=\left|S_i\right|,\forall i\in M \tag{6}$$

$$q_j=\sum\nolimits_{i\in M}\left(x_{ij}\cdot i\right),\forall j\in N \tag{7}$$

$$\begin{aligned}b_{ij}&=x_{ij}\times\left(\sum\nolimits_{h\in N} v_{ijh}\cdot e_{ih}+\sum\nolimits_{h\in N}\left(1-z_{ijh}\right)\cdot h_{ir_jr_h}\right),\\ &\forall i\in M,\forall j\in N\setminus\{1\}, b_{i1}=0\end{aligned} \tag{8}$$

$$e_{ij}=x_{ij}\times\left(b_{ij}+o_j/t_{ir_j}\right),\forall j\in N,\forall i\in M \tag{9}$$

$$g_i=\sum\nolimits_{j\in S_i}\sum\nolimits_{h\in S_i}\left(1-z_{ijh}\right),\forall i\in M \tag{10}$$

$$c_i=max\left\{e_{ij}\right\},\forall j\in S_i,\forall i\in M \tag{11}$$

$$L_j=max\left\{e_{ij}-d_j,0\right\},\forall j\in S_i,\forall i\in M \tag{12}$$

$$\left|N\right|=\sum\nolimits_{i\in M}\left|S_i\right| \tag{13}$$

$$b_{ij}\geq 0,\forall j\in N,\forall i\in M \tag{14}$$

$$x_{ij}\in\{0,1\},\forall j\in N,\forall i\in M \tag{15}$$

$$z_{ijh}\in\{0,1\},\forall j,h\in S_i,\forall i\in M \tag{16}$$

$$v_{ijh}\in\{0,1\},\forall j,h\in S_i,\forall i\in M \tag{17}$$

In the above model, there are three objectives to be simultaneously optimized. The first objective in (1) is expressed by the minimization of the total switching times. The second objection in (2) is used to minimize the weighted sum of the average production time of each unit and the total delay time of all orders. To ensure capacity balance, the third objective in (3) is expressed by the minimization of the standard deviation of the production time of all units. Constraint (4) indicates that there are *N* and only *N* orders assigned to M units. Constraint (5) indicates that order j can only be assigned to a unit once. Constraint (6) gives the relationship between the order set on each unit and the decision variables x_{ij}. Constraint (7) represents the unit to which order j is assigned. Equation (8) and equation (9) indicate the starting and end production time of order j. Equation (10) gives the switching times on unit i. Equation (11) gives the total production time of unit i. Equation (12) gives the delay time of order j. Equation (14) to (17) show the range of each decision variable. In the above mathematical model, the objectives are conflicting with each other. So, the order static multiobjective scheduling problem is a multiobjective optimization problem.

B. Rush Order Dynamic Multiobjective Recheduling Problem

Due to the frequent changes in customer demand, enterprises need to add rush orders to the executed scheduling plan (i.e. the OSSP) to generate a rescheduling plan (i.e. the RODRP).

The RODRP studied in this paper can be described as follows. In the executed scheduling plan, N orders are allocated to M manufacturing units, and each unit carries out production operations according to the order sequence. R rush orders need to be inserted into the order sequence in the executed scheduling plan at some point. The problem to be solved is how to reasonably insert these rush orders into the arranged order sequence and determine the allocated unit and insertion position of each rush order, so as to ensure the less impact on the executed scheduling plan and achieve the optimal target.

TABLE II. THE NEW PARAMETERS OF THE RUSH ORDER DYNAMIC RESCHEDULING PROBLEM MODEL

Parameter	Description
R	Rush order set
N'	Scheduled and not yet produced order set
P_i	Scheduled and not yet produced order set on unit i, $i \in M$
S_i'	Rush order set on unit i, $i \in M$
q_j'	Unit to which rush order j is assigned, $j \in R$
w_1, w_2, w_3	Weight coefficients of average production time of all units, total delay time of all orders and standard deviation of the production time of all units
p_1, p_2	Penalty coefficients of delay time of all orders and the standard deviation of the production time of all units

Based on the actual investigation and analysis of the RODRP, the MODROM is established. Except for the parameters listed in Table I, some new parameters used in the MODROM are shown in Table II. The decision variables are described as follows.

- $z_{ijh} = \begin{cases} 1, \text{ if planned products for order } j \text{ and adjacent} \\ \quad \text{order } h \text{ on unit } i \text{ are same} \\ 0, \text{ otherwise} \qquad j,h \in S_i' \cup P_i, i \in M \end{cases}$

- $v_{ijh} = \begin{cases} 1, \text{ if order } h \text{ is produced before the} \\ \quad \text{adjacent order } j \text{ on unit } i \\ 0, \text{ otherwise} \qquad j,h \in S_i' \cup P_i, i \in M \end{cases}$

- $x_{ij} = \begin{cases} 1, \text{if rush order } j \text{ is assigned to unit } i \\ 0, \text{otherwise} \qquad j \in R, i \in M \end{cases}$

- b_{ij} is the starting time of order j on unit i, $j \in S_i' \cup P_i, i \in M$

- e_{ij} is the end time of order j on unit i, $j \in S_i' \cup P_i, i \in M$

Based on the above parameters and decision variables, the mathematic model of the RODRP can be established as follows.

$$\min \ f_1 = \sum_{i\in M} g_i \tag{18}$$

$$\min \ f_2 = \sum_{j\in R} L_j \tag{19}$$

$$\begin{aligned} \min \ f_3 = & w_1\left(\sum_{i\in M} c_i\right)/|M| + w_2 p_1 \sum_{j\in N'} L_j \\ & + w_3 p_2 \sqrt{\left(\sum_{i\in M}\left(c_i - \left(\sum_{l\in M} c_l\right)/|M|\right)^2\right)/|M|} \end{aligned} \tag{20}$$

s.t.

$$\sum_{i\in M}\sum_{j\in R} x_{ij} = |R| \tag{21}$$

$$\sum_{i\in M} x_{ij} = 1, \forall j \in R \tag{22}$$

$$\sum_{j\in N'} x_{ij} = |S_i'|, \forall i \in M \tag{23}$$

$$q_j' = \sum_{i\in M}\left(x_{ij}\cdot i\right), \forall j \in R \tag{24}$$

$$\begin{aligned} b_{ij} = & x_{ij}\times\left(\sum_{h\in N'\cup R} v_{ijh}\cdot e_{ih} + \sum_{h\in N'\cup R}\left(1 - z_{ijh}\right)\cdot h_{ir_j r_h}\right) \\ & , \forall i \in M, \forall j \in N' \cup R \setminus \{1\}, b_{i1} = 0 \end{aligned} \tag{25}$$

$$e_{ij} = x_{ij}\times\left(b_{ij} + o_j / t_{ir_j}\right), \forall j \in N' \cup R, \forall i \in M \tag{26}$$

$$g_i = \sum_{j\in S_i'\cup P_i}\sum_{h\in S_i'\cup P_i}\left(1 - z_{ijh}\right), \forall i \in M \tag{27}$$

$$c_i = max\left\{e_{ij}\right\}, \forall j \in S_i' \cup P_i, \forall i \in M \tag{28}$$

$$L_j = max\left\{e_{ij} - d_j, 0\right\}, \forall j \in N' \cup R, \forall i \in M \tag{29}$$

$$\sum_{i\in M}|S_i'| = |R|, \sum_{i\in M}|P_i| = |N'| \tag{30}$$

$$b_{ij} \ge 0, \forall j \in N' \cup R, \forall i \in M \tag{31}$$

$$x_{ij} \in \{0,1\}, \forall j \in N' \cup R, \forall i \in M \tag{32}$$

$$z_{ijh} \in \{0,1\}, \forall j,h \in S_i' \cup P_i, \forall i \in M \tag{33}$$

$$v_{ijh} \in \{0,1\}, \forall j,h \in S_i' \cup P_i, \forall i \in M \tag{34}$$

In the above model, three objectives are optimized simultaneously. The first objective in (18) is expressed by the minimization of the total switching times after inserting rush orders. The second objective in (19) is expressed by the minimization of the total delay time of all rush orders. The third objective in (20) is used to minimize the weighted sum of the average production time of all units, the total delay time of all scheduled but not yet produced orders and the standard deviation of the production time of all units. Constraint (21) indicates that that there are $|R|$ and only $|R|$ orders assigned to M units. Constraint (22) indicates rush order j can only be assigned to a unit once. Constraint (23) gives the relationship between the rush order set on each unit and the decision variables x_{ij}. Constraint (24) represents the unit to which rush order j is assigned. Equation (25) and equation (26) indicate the start and end production time of the order j. Equation (27) gives the switching times on the manufacturing unit i. Equation (28) gives the total processing time of the manufacturing unit i. Equation (29) gives the delay time of order j. Equation (31) to (34) show the value range of each decision variable. In the above mathematical model, the objectives are conflicting with each other. So the rush order dynamic rescheduling problem is also a multiobjective optimization problem. In this paper, we proposed an improved MOEA/D (details in Section IV) to solve these two multiobjective optimization problems.

III. INTORDUCTION TO THE HEURISTIC AGLORITHM

In this paper, two heuristic algorithms based on heuristic rules and DDLS are proposed for the OSSP and the RODRP respectively. These heuristic algorithms can directly solve two problems and are utilized to improve the population initialization method of the traditional MOEA/D.

A. Heuristic Algorithm for Order Static Scheduling Problem

1) Heuristic rules: According to the characteristics of the OSSP, we employ some heuristic rules which are suitable for this problem.

a) Order classification: In order to reduce product switching times and improve equipment production efficiency, all orders to be scheduled are classified according to planned product.

b) Earliest due time: In order to reduce total delay time, the orders in each order class are sorted according to the due time. The order with early due time has prority in production.

c) Order calss allocation rule: The specific rule is described as follows. M order classes are randomly assigned to M manufacturing units. After that, a certain order class randomly selected from the remaining order classes is assigned to the unit with the shortest total production time. This process is repeated until all order classes are assigned to the appropriate unit. This rule ensures that the production capacity of each unit is as balanced as possible.

2) Discrete dynamic local search: After all order classes are allocated to each manufacturing unit, the DDLS algorithm is employed to search each unit locally to obtain the better sequence of all order classes on each manufacturing unit. The procedure of the DDLS algorithm is as follows.

Algorithm 1 ***Procedure of Discrete Dynamic Local Search***

Input: Iteration times M, Local search times N, Initial search step K, Reduced step length k, the initial solution $X_{current}$.

Output: the best solution X_{best}.

Initialization: Set iteration counter $epoch = 0$, $i = 0$. set $X_{best} = X_{current}$, $f_{best} = f(X_{best})$ and $f_{current} = f(X_{current})$.

1. **while** $epoch < M$ **do**
2. Randomly generate search step $dk \in [1, K]$. Set $epoch = epoch + 1$, $X_{current} = X_{best}$ and $f_{current} = f_{best}$.
3. **for** $i = 1, 2, \ldots, N$ **do**
4. Randomly generate exchange point $P_1 \in [1, L]$, and then get exchange point $P_2 = (P_1 + dk)\%L$, where L is the length of the decision vector of Current Solution $X_{current}$. Exchange variables between exchange point P_1 and P_2 to obtain new solution X_{new}, $f_{new} = f(X_{new})$.
5. If $f_{new} < f_{best}$, set $f_{best} = f_{new}$, $X_{best} = X_{new}$, and skip to step 3.
6. If $f_{new} < f_{current}$, set $f_{current} = f_{new}$, $X_{current} = X_{new}$, and skip to step 3.
7. Randomly generate exchange point $P_1 \in [1, L]$, and then get exchange point $P_2 = (P_1 + L - dk)\%L$. Exchange variables between exchange point P_1 and P_2 to obtain new solution X_{new}, $f_{new} = f(X_{new})$.
8. If $f_{new} < f_{best}$, set $f_{best} = f_{new}$, $X_{best} = X_{new}$, and skip to step 3.
9. If $f_{new} < f_{current}$, set $f_{current} = f_{new}$, $X_{current} = X_{new}$, and skip to step 3.
10. **end for**
11. If $K - k < 0$, set $K = 1$, otherwise, $K = K - k$.
12. **end while**

3) Heuristic algorithm for the OSSP: The heuristic algorithm designed for the OSSP is described in detail in Algorithm 2. In the DDLS, the decision vector is the sequence of all order classes on each manufacturing unit, and the fitness function is the weighted sum of switching times, production time of all manufacturing units and total delay time of all orders.

Algorithm 2 ***Procedure of the Heuristic Algorithm for the OSSP***

Input: All orders to be scheduled

Output: The scheduling plan

1. Classify all orders by planned product according to the *order classification* rule to get all order classes $C = \{c_1, c_2, \ldots, c_k\}$, k is the number of order class.
2. Sort orders in order class $c \in C$ based on the *earliest due time* rule.
3. Allocate all order classes to each manufacturing unit according to the *order class allocation rule*.
4. For each manufacturing unit, utilize the DDLS algorithm to obtain the better sequence of all order classes on the unit.
5. Get the manufacturing unit to which each order is assigned and the production sequence of the orders on each unit, output the scheduling plan.

B. Heuristic Algorithm of Rush Order Dynamic Rescheduling Problem

1) Heuristic rules: Aiming at the characteristics of the RODRP, some heuristic rules suitable for the RODRP are used to solve the problem.

a) Earliest due time: In order to reduce the total delay time of rush orders, the rush orders to be inserted are sorted according to the due time.

b) Neighboring-region search: Aiming at minimizing switching times, production time, delay time of all scheduled but not yet produced orders and delay time of rush orders on all units, the neighboring-region search is applied to archive the current optimal insertion location of each rush order.

2) Discrete dynamic local search: The DDLS algorithm is performed for the RODRP. The decision varibles of the DDLS algorithm are the insertion position of rush orders on each manufacturing unit, and the fitness function is the weighted sum of switching times, production time, delay time of all scheduled but not yet produced orders and delay time of all rush order on the manufacturing unit. The procedure of the DDLS algorithm is described in detail in Algorithm 1.

3) Heuristic algorithm for the RODRP: The procedure of the heuristic algorithm for the RODRP is introduced in detail in Algorithm 3.

Algorithm 3 ***Procedure of the Heuristic Algorithm for the RODRP***

Input: All rush orders to be inserted

Output: The rescheduling plan

1. Determine the minimum insertable position of each manufacturing unit according to the start rescheduling time and the scheduling plan.
2. Sort rush orders based on the *earliest due time* rule.
3. Use *neighboring-region search* to find the current optimal insertion position of each rush order.
4. For each manufacturing unit, employ the DDLS algorithm to get the better inserting sequence and position of all rush orders on the unit.
5. Get the manufacturing unit to which each rush order is assigned, the insertion sequence and the location where each rush order is inserted, output the rescheduling plan

IV. An Improved MOEA/D for Multiobjective Static Scheduling Optimization Model and Multiobjective Dynamic Rescheduling Optimization Model

In recent years, many kinds multiobjective evolutionary algorithms (MOEAs) has been proposed in the literature [7]-[9]. Among these MOEAs, MOEA/D [7] is one of the most widely used algorithms for scheduling problems [10]-[12] due to its characteristics of fast convergence, good diversity and low computational complexity. Therefore, in this paper, we prefer to adopt MOEA/D as the solved algorithm and improve the traditional MOEA/D combined with the MOSSOM and the MODROM.

A. Solution Representation

Aiming at the characteristics of the solutions of the MOSSOM and the MODROM, two encoding and decoding methods suitable for these two problems are designed respectively.

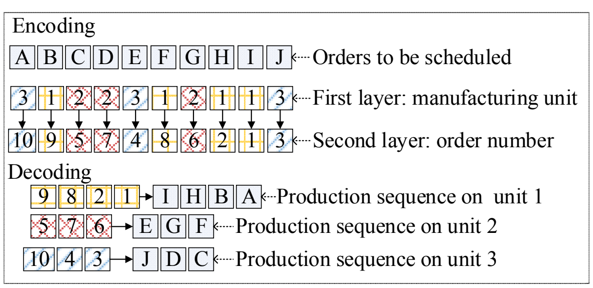

Fig. 1. An example of encoding and decoding for the MOSSOM.

For the OSSP, we need to determine the unit to which each order is assigned and the production sequence of the orders on each unit. Therefore, we adopt the two-layer encoding method. An example of encoding and decoding for the MOSSOM is shown in Fig. 1. Each element in the first layer represents the unit to which the order is assigned. The order-permutation-based presentation is employed in the second layer and each element represents the order number. In the decoding process, the orders assigned to each unit is determined by the first layer, and then the second layer determines the production sequence of the orders on each unit.

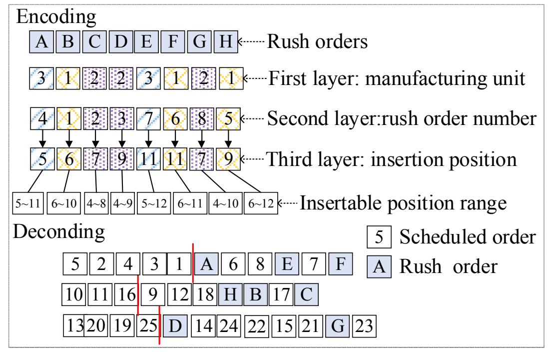

Fig. 2. An example of encoding and decoding for the MODROM.

For the RODRP, we employ the three-layer encoding method, and each layer is used to determine the manufacturing unit to which each rush order is assigned, the insertion sequence and the location where each rush order is inserted. Fig. 2 shows an example of encoding and decoding for the MODROM. Each element in the first layer represents the unit to which rush order is assigned. In the second layer, the order-permutation-based presentation is used to indicate the insertion sequence of rush orders. Each element in the third layer is the insertion position of the corresponding rush order. It should be noted that the insertable position of each rush order has a range and changes dynamically. In other words, the insertable position range of the latter rush order is increased by 1 compared with that of the previous rush order. In the decoding process, the first layer determines the manufacturing unit to which each rush order is assigned, the order insertion sequence on each manufacturing unit is determined by the second layer, and then the third layer determines the insertion position of each rush order on each manufacturing unit. In the Fig. 2, the orders on the left side of the red line are the produced orders, which means the red line is the minimum insertable position of each manufacturing unit.

B. Crossover and Mutation Strategy

In this paper, we design three crossover operators and one mutation operator for the MOSSOM and the MODROM, which are two-point crossover operator, similar HUX crossover operator, PMX crossover operator and swap mutation crossover.

In view of the two-layer encoding method of the OSSP, we design the crossover and mutation strategy used in the improved MOEA/D described as follows.

Algorithm 4 ***Procedure of Crossover and Mutation Strategy for the MOSSOM***

Input: Two parent solutions x_1 and x_2

Output: Two offspring solutions x_1'' and x_2''

1. Randomly generate the crossover probability $p_c \in [0,1]$.
2. **if** $p_c < 0.5$ **do**
3. Apply two-point crossover operator on the first layer of solution encoding of the two parents solutions x_1 and x_2 to generate two new crossed solutions x_1' and x_2'.
4. **else do**
5. Utilize similar HUX crossover operator on the first layer of solution encoding of the two parents solutions x_1 and x_2 to generate two new crossed solutions x_1' and x_2'.
6. **end if**

7. Randomly generate mutation probability $p_m \in [0,1]$
8. **if** $p_m < 0.5$ **do**
9. Use swap mutation operator on the first layer of solution encoding of the two parents solutions x_1' and x_2' to generate two new mutated solutions x_1'' and x_2''.
10. **else do**
11. Apply swap mutation operator on the second layer of solution encoding of the two parents solutions x_1' and x_2' to generate two new mutated solutions x_1'' and x_2''.
12. **end if**

Aiming at the three-layer encoding method for the RODRP, we design a crossover and mutation strategy described as follows, which is employed in the improved MOEA/D to solve the MODROM.

Algorithm 5 ***Procedure of Crossover and Mutation Strategy for the MODROM***

Input: Two parent solutions x_1 and x_2
Output: Two offspring solutions x_1'' and x_2''

1. *Randomly* generate crossover probability $p_c \in [0,1]$
2. **if** $p_c < 0.4$ **do**
3. Apply two-point crossover operator on the first layer of solution encoding of the two parents solutions x_1 and x_2 to generate two new crossed solutions x_1' and x_2'
4. **else if** $0.4 <= p_c <= 0.7$ **do**
5. Utilize similar HUX crossover operator on the first layer of solution encoding of the two parents solutions x_1 and x_2 to generate two new crossed solutions x_1' and x_2'.
6. **else do**
7. Use PMX crossover operator on the first layer of solution encoding of the two parents solutions x_1 and x_2 to generate two new crossed solutions x_1' and x_2'.
8. **end if**
9. Randomly generate mutation probability $p_m \in [0,1]$
10 **if** $p_m < 0.5$ **do**
11. Use swap mutation operator on the first layer of solution encoding of the two parents solutions x_1' and x_2' to generate two new mutated solutions x_1'' and x_2''.
12 **else do**
13. Apply swap mutation operator on the second layer of solution encoding of the two parents solutions x_1' and x_2' to generate two new mutated solutions x_1'' and x_2''.
14 **end if**

C. The Improved MOEA/D

In this paper, in the view of the MOSSOM and the MODROM, we propose a improved MOEA/D. The main improvement straegies are as follows. We utilize the heuristic algorithms, which are introduced in detail in Section III, to improve the population initialization method of the traditional MOEA/D. The specific improvement is that some solutions in the initial population are obtained by the heuristic algorithms, and the remaining solutions are randomly generated in the decision space. The strategy not only ensures the initial population has a certain quality and diversity but also improves the convergence speed of the algorithm. In order to further improve the speed of the algorithm, we do not set the external population, but use the fast nondominated sorting method of NSGA-II [9] to sort the last population after the end of the algorithm iteration, and then use the solutions corresponding to the first Pareto Front as the last output solutions. The whole procedure of the improved MOEA/D is detailed in Algorithm 6.

Algorithm 6 ***Whole Procedure of the Improved MOEA/D***

Input:
Emax: Maximum evaluation times
N: Population size
$\lambda_1, \lambda_2, \cdots, \lambda_N$: Uniformly distributed weight vectors
T: Neighborhood size
Output: The first Pareto Front PF_1 and the nondominated solution set

1. **Parameter Initialization:** Calculate Euclidean distance between λ_i and λ_j, $\forall i,j \in \{1,2,\cdots,N\}$. For each weight vector λ_i, $i = 1, 2,\ldots, N$, its closest T weights vectors $\lambda_i^{i_1}, \lambda_i^{i_2}, \cdots \lambda_i^{i_T}$ are found to form its neighborhood $B(i) = \{i_1, i_2,\ldots, i_T\}$. Initialize the reference point $z = (z^1, z^2, \ldots, z^m)^T$ where m is number of objectives . Set current evaluation $E_{cur} = 0$ and probability $p_1 = 0.5$.
2. **Population initialization:** Apply the heuristic algorithm (details in Algorithm 2 and algorithm 3) to obtain U initial solutions $x_1, x_2, \ldots, x_U$ and randomly generate the remaining initial solutions $x_{U+1}, x_{U+2}, \ldots, x_N$ in decision space. Set initial population $P_{init} = \{x_1, x_2, \ldots, x_U, x_{U+1}, x_{U+2}, \ldots, x_N\}$ and current population $P = P_{init}$.
3. **Evaluation:** Evaluate each solution x in current population P to get objectives vectors $F(x) = (f_1(x), f_2(x),\ldots, f_m(x))$ where f_i, $i = 1,2,\ldots,m$ is the i-th objective，Set $E_{cur} = E_{cur} + N$.
4. **do**
5. **for** $i = 1, 2, \ldots, N$ **do**
6. **Parent solution selection:** Randomly generate probability $p \in [0,1]$. If $p < p_1$, randomly select two indices k and l from $B(i)$. If $p >= p_1$, then Randomly select two indexes k and l from the whole indexes $\{1, 2, \ldots, N\}$.
7. **Evolution:** Apply crossover and mutation strategy (details in Algorithm 4 and algorithm 5) on parent solutions x_k, x_l to obtain two offspring solutions y_1 and y_2. Calculate objectives vectors $F(y_1)$ and $F(y_2)$. Set $E_{cur} = E_{cur} + 2$.
8. **Updating:** Utilize offspring solutions y_1 and y_2 to update reference point $z = (z^1, z^2, \ldots, z^m)^T$. For $j = 1,2, \ldots, m$, if $z^j < f_j(y')$, then set $z^j = f_j(y')$, where $y' \in \{y_1, y_2\}$.
9. **Updating:** If $p < p_1$, for $j \in B(i)$, if $g^{et}(y'|\lambda_i, z) < g^{et}(x_j|\lambda_i, z)$, where $g^{et}()$ is the *Tchebycheff Approach* , then replace x_j with y', where $y' \in \{y_1, y_2\}$. If $p >= p_1$, for $j = 1, 2, \ldots, N$, if $g^{et}(y'|\lambda_i, z) < g^{et}(x_j|\lambda_i, z)$, replace x_j with y', $y' \in \{y_1, y_2\}$.
10. **end for**
11 **while** $E_{cur} < E_{max}$
12 Sort the final population P by the fast no-dominated sorting method to obtain h Pareto Front $PF_1, PF_2, \ldots, PF_h$.

V. Numerical Experiments

A. Experiment Setting

In this paper, we compare the improved MOEA/D with MOEA/D [7] and NSGA-II [9]. The algorithms involved in this paper are all implemented in C++. All experiments are carried out on a personal computer with CPU of Intel Core i7-7700k and memory of 16.0GB. we choose the generational distance (GD) [13] and the C-metric [14] as our performance metrics. GD can consider the convergence property of the nondominated solution set. The calculation expression is shown in (35), where P is the solution set obtained by the algorithm, P^* is the reference set, and $dis(x, y)$ represents the Euclidean distance between point y in P and point x in P^*. The smaller the GD value, the higher the quality and convergence of the nondominated solution set will have. C-metric is used to evaluate the dominance relationship between the nondominated solution sets obtained by each algorithm. The calculation expression is shown in (36).

$$GD = \frac{\sqrt{\sum_{y \in P} \min_{x \in p^*} dis(x, y)^2}}{|P|} \tag{35}$$

$$C(X, Y) = \frac{|y \in Y \mid \exists x \in X, x \prec y|}{|Y|} \tag{36}$$

where X and Y represent nondominated solution set obtained by two algorithms respectively. $C(X,Y)>C(Y,X)$ indicates that X from the algorithm A is better than Y from the algorithm B.

For each problem, each algorithm runs 20 independent times. The mean value of the 20 runs and the corresponding standard deviation are used for evaluating algorithm performance. The *Wilcoxon rank sum test* with a significance level of 5% is employed to compare the improved MOEA/D with the other algorithms. The best result for each problem is highlighted in gray background in the table The symbol '+' and '-' represent that the improved MOEA/D is significantly better than and significantly worse than its rival respectively, the symbol '=' indicates that there is no significant difference between the two algorithms. A, B and C are used to represent the nondominated solution set obtained by the improved MOEA/D, MOEA/D and NSGA-II respectively.

The parameter settings of the heuristic algorithm for OSSP are as follows. The number of iterations is set to 300. The local search times is set to 20, and the initial search step $K=ceil(V/2)$, where V is the number of variables and ceil() is the integral function. The reduction step is set to k=1. The weights of the three items in the fitness function is 0.4, 0.4 and 0.2 respectively, and the penalty coefficient of the first item is set to 10. The parameter settings of the improved MOEA/D algorithm for order static scheduling problem are as follows. The number of initial solutions obtained by the heuristic algorithm is set to 20. The population size is set to 800 and the maximum evaluations is set to 10^6. The crossover probability is 0.95 and the mutation probability is set to 0.7. The neighborhood size is set to 20, the weight coefficient and penalty coefficient of the second objective are set to 0.3, 0.7 and $100/O$ respectively, O is the number of orders. The parameter settings and crossover and mutation strategy of MOEA/D are the same as those of the improved MOEA/D. The population size of NSGA-II is 100, and the maximum evaluations is set to 10^6. The crossover and mutation strategy is the same with the improved MOEA/D. The parameter settings of heuristic algorithm for RODRP are as follows. The initial search step is set to $K = ceil(V/3)$. The weights of the four items in the fitness function are set to 0.2, 0.2, 0.2 and 0.4 respectively. The penalty coefficients of the first, third and fourth items are set as 10, 10 and 500 respectively. Other parameter settings are the same with heuristic algorithm for OSSP. The parameter settings of the improved MOEA/D for RODRP are as follows. The weight coefficients of the third objective are respectively set as 0.4, 0.4, 0.2, and the penalty coefficients p_1 and p_2 are respectively set as $50/R$ and $5M$, where R is the number of rush orders, and M is the number of manufacturing units.

B. Performance Analysis of the Improved MOEA/D on the MOSSOM

In this paper, based on the actual data, eight instances of order static scheduling problems of different scales (different number of order and manufacturing unit) are designed. The improved MOEA/D, MOEA/D and NSGA-II are used to solve the OSSPs of different scales. The comparison results obtained by the three algorithms via C-metric and GD are presented in Table III and Table IV respectively.

TABLE III. COMPARISON RESULTS FOR THE IMPORVED MOEA/D AND TWO OTHER ALGORITHMS ON MOSSOM VIA C-METRIC

Scale	C-metric	Mean	Std.	C-metric	Mean	Std.
40×3	C(A,B)	**3.78e-01**	3.57e-01	C(B,A)	2.27e-01$^{=}$	3.21e-01
	C(A,C)	3.03e-01	2.67e-01	C(C,A)	**4.45e-1**$^{=}$	4.02e-01
40×6	C(A,B)	**5.17e-01**	2.69e-01	C(B,A)	2.40e-01^{+}	2.78e-01
	C(A,C)	1.52e-01	1.36e-01	C(C,A)	**2.71e-01**$^{=}$	2.15e-01
80×3	C(A,B)	**4.31e-01**	4.79e-01	C(B,A)	2.10e-01^{+}	3.30e-01
	C(A,C)	2.77e-01	3.59e-01	C(C,A)	**4.12e-01**$^{=}$	4.37e-01
40× 6	C(A,B)	**3.65e-01**	3.94e-01	C(B,A)	3.48e-01$^{=}$	3.71e-01
	C(A,C)	6.2e-02	8.17e-02	C(C,A)	**5.53e-01**$^{-}$	2.75e-01
120×3	C(A,B)	**6.11e-01**	4.29e-01	C(B,A)	2.03e-01^{+}	3.66e-01
	C(A,C)	**5.97e-01**	2.67e-01	C(C,A)	2.18e-03^{+}	4.74e-03
120×6	C(A,B)	**5.21e-01**	3.61e-01	C(B,A)	1.56e-01^{+}	3.04e-01
	C(A,C)	**2.97e-01**	1.02e-01	C(C,A)	1.97e-01^{+}	1.62e-01
158×3	C(A,B)	**4.75e-01**	4.30e-01	C(B,A)	2.06e-01^{+}	3.40e-01
	C(A,C)	**5.02e-01**	1.59e-01	C(C,A)	5.62e-03^{+}	9.15e-03
158×6	C(A,B)	**9.51e-01**	1.46e-01	C(B,A)	2.89e-03^{+}	1.23e-02
	C(A,C)	**8.73e-01**	8.06e-02	C(C,A)	4.38e-03^{+}	1.96e-02

The comparison results between the three algorithms via C-metric are presented in Table III. From the table, it is clear that the improved MOEA/D gets the better results for all eight instances compared with the traditional MOEA/D. More specifically, the improved MOEA/D is significantly better than MOEA/D for six out of eight instances. With comparison to NSGA-II, the improved MOEA/D can obtain significantly better results for the last four large-scale instances. Besides, the improved MOEA/D is significantly inferior to NSGA-II for the OSSPs with problem scale 80×6, for the first three OSSPs with problem scale 40×3, 40×6 and 80×3 there is no significant performance difference between the two algorithms. Based on these results, it can be conducted that the improved MOEA/D may be inferior to NSGA-II algorithm in small-scale instances, but for large-scale instances, the improved MOEA/D can be significantly superior to MOEA/D and NSGA-II.

TABLE IV. COMPARISON RESULTS FOR THE IMPORVED MOEA/D AND TWO OTHER ALGORITHMS ON MOSSOM VIA GD METRIC

Scale	MOEA/D		NSGA-II		The improved MOEA/D	
	Mean	Std.	Mean	Std.	Mean	Std.
40×3	**1.50e-02**$^{=}$	1.4e-02	4.34e-02^{+}	3.6e-02	1.69e-02	1.3e-02
40×6	3.09e-03^{+}	1.4e-03	5.22e-03^{+}	1.9e-03	**2.34e-03**	9.9e-04
80×3	**4.35e-01**$^{-}$	7.2e-01	1.15e-00$^{=}$	1.9e+00	1.83e+00	2.0e+00
80× 6	3.52e-03$^{=}$	1.2e-03	8.82e-03^{+}	6.0e-03	**3.48e-03**	8.4e-04
120×3	4.23e-03$^{=}$	2.7e-03	6.02e-02^{+}	3.1e-02	**2.55e-03**	1.6e-03
120×6	1.72e-03$^{=}$	6.0e-04	1.05e-02^{+}	3.6e-02	**1.58e-03**	5.8e-04
158×3	3.20e-03$^{=}$	1.4e-03	3.27e-02^{+}	2.2e-02	**2.93e-03**	1.3e-03
158×6	2.99e-03^{+}	1.0e-03	1.82e-02^{+}	4.9e-03	**7.36e-04**	3.2e-04

The comparison results via GD metric are listed in Table IV. It can be seen in Table IV that the improved MOEA/D can obtain the significantly better results than NSGA-II for seven out of eight instances. With comparison with MOEA/D, the improved MOEA/D gets the better results for six out of the eight instances. For the two small-scale instances with problem scale 40×3 and 80×3 ,the traditional MOEA/D is better than the improved MOEA/D. Based on these results, we can conduct the improved MOEA/D has better convergence than MOEA/D for the large-scale OSSPs and is superior to NSGA-II for all OSSPs in this paper.

C. *Performance Analysis of the Improved MOEA/D on the MODROM*

In this paper, we design four instances of rush order dynamic rescheduling problems of different scales (different number of rush order and manufacturing unit) base on the actual production data. The improved MOEA/D, MOEA/D and NSGA-II are employed to solve these RODRPs.

TABLE V. COMPARISON RESULTS OF C-METRIC FOR THE IMPORVED MOEA/D AND TWO OTHER ALGORITHMS ON MODROM

Scale	C-metric	*Mean*	*Std.*	C-metric	*Mean*	*Std.*
20×6	C(A,B)	**6.40e-01**	3.14e-01	C(B,A)	$1.56e\text{-}01^+$	1.88e-01
	C(A,C)	**6.45e-01**	8.54e-02	C(C,A)	$7.47e\text{-}03^+$	1.37e-02
20×3	C(A,B)	**9.99e-01**	5.95e-04	C(B,A)	$0e\text{-}00^+$	0e-00
	C(A,C)	**4.78e-01**	1.31e-01	C(C,A)	$5.46e\text{-}03^+$	2.44e-01
40×6	C(A,B)	2.37e-01	3.36e-01	C(B,A)	**$6.46e\text{-}01^-$**	3.94e-01
	C(A,C)	**4.62e-01**	1.16e-01	C(C,A)	$5.67e\text{-}03^+$	1.02e-02
40×3	C(A,B)	**9.93e-01**	1.08e-02	C(B,A)	$0e\text{-}00^+$	0e-00
	C(A,C)	**3.78e-01**	1.15e-01	C(C,A)	$0e\text{-}00^+$	0e-00

TABLE VI. COMPARISON RESULTS OF GD METRIC FOR THE IMPORVED MOEA/D AND TWO OTHER ALGORITHMS ON MODROM

Scale	MOEA/D		NSGA-II		Improved MOEA/D	
	Mean	**Std.**	**Mean**	**Std.**	**Mean**	**Std.**
20×3	$7.52e\text{-}03^+$	1.8e-03	$1.97e\text{-}02^+$	8.3e-03	**1.36e-03**	2.4e-04
20×6	$1.36e\text{-}03^+$	9.5e-04	$1.68e\text{-}02^+$	2.4e-03	**7.96e-04**	2.2e-04
40×3	$1.25e\text{-}02^+$	2.6e-03	$2.28e\text{-}02^+$	7.4e-03	**2.22e-03**	1.2e-03
40×6	$3.71e\text{-}03^=$	1.8e-03	$1.31e\text{-}02^+$	2.7e-03	**3.18e-03**	6.6e-04

The comparison results between the three algorithms via C-metric are presented in Table V. From the table, it is clear that the improved MOEA/D is significantly better than NSGA-II for all the four instances. Besides, the improved MOEA/D is significantly inferior to MOEA/D for the RODRP with problem scale 40×6 and for the other three instances the improved MOEA/D obtains the significantly better results than MOEA/D. Table VI shows the comparison results between the three algorithm via GD metric. From the comparison results, it can be seen that the improved MOEA/D can get the significantly better results than NSGA-II for all the four instances. The improved MOEA/D is significantly better than the traditional MOEA/D for three out of the four instances. Based on these comparison results, we can conduct that the improved MOEA/D is more effective and superior than the other two algorithms for the RODRPs in this paper.

VI. CONCLUSION

MTO manufacturers make production plans according to customer orders, which can help enterprises avoid market risk, reduce inventory pressure, and meet customer demand to the maximum extent. However, due to the characteristics of MTO production mode, the OSSP and the RODRP have become the urgent problems to be solved by MTO manufacturers. Therefore, in this paper we study the OSSP and the RODRP for a carbon black packaging production line of an order oriented manufacturing enterprise. Firstly, the MOSSOM and MODROM are established respectively. In view of the characteristics of the OSSP and the RODRP, two heuristic algorithms based on heuristic rules and DDLS are designed respectively which are employed to improve the population initialization method of MOEA/D to get the improved MOEA/D. Based on the actual production data, eight instances of the OSSPs of different scales and four instances of the RODRPs of different scales are separately constructed. The experimental results indicate that the improved MOEA/D is effective and superior in solving these two problems.

REFERENCES

[1] P. Brucker, B. Jurisch, and B. Sievers, “A branch and bound algorithm for the job-shop scheduling problem,” Discrete Appl. Math., vol. 49, no. 1-3, pp. 107-127, March. 1994.

[2] M. Dell’Amico, and M. Trubian, “Applying tabu search to the job-shop scheduling problem,” Ann. Oper. Res., vol. 41, no. 3, pp. 231-252, 1993.

[3] Q. K. Pan, M. F. Tasgetiren, and Y. C. Liang, “A discrete differential evolution algorithm for the permutation flowshop scheduling problem,” Comput. Ind. Eng., vol. 55, no. 4, pp. 795-816, 2008.

[4] G. E. Vieira, J. W. Herrmann, and E. Lin, “Rescheduling manufacturing systems: a framework of strategies, policies, and methods,” J. Scheduling, vol. 6, no. 1, pp. 39-62, 2003.

[5] P. Moratori, S. Petrovic, and J. A. Vázquez-Rodríguez, “Integrating rush orders into existent schedules for a complex job shop problem,” Appl. Intell., vol. 32, no. 2, pp. 205-215, 2010.

[6] D. J. Wang, F. Liu, J. J. Wang, and Y. Z. Wang, “Integrated rescheduling and preventive maintenance for arrival new jobs through evolutionary multi-objective optimization,” Soft Comput., vol. 20, no. 4, pp. 1635-1652, 2016.

[7] Q. Zhang, and H. Li, “MOEA/D: A multiobjective evolutionary algorithm based on decomposition,” IEEE Trans. Evolut. Comput., vol. 11, no. 6, pp. 712-731, 2007.

[8] L. X. Tang, X. P. Wang and Z. M. Dong, “Adaptive multiobjective differential evolution with reference axis vicinity mechanism,” IEEE Trans. Cybern., vol. 49, no. 9, pp. 3571-3585, 2018.

[9] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” IEEE Trans. Evolut. Comput., vol. 6, no. 2, pp. 182-197, 2002.

[10] P. C. Chang, S. H. Chen, Q. Zhang, and J. L. Lin, “MOEA/D for flowshop scheduling problems,” IEEE Congress on Evolutionary Computation, CEC 2008, 2008, pp. 1433-1438, Hong Kong, china.

[11] A. Alhindi, and Q. Zhang, “MOEA/D with Tabu Search for multiobjective permutation flow shop scheduling problems,” IEEE Congress on Evolutionary Computation, CEC 2014, 2014, pp. 1155-1164, Beijing, China.

[12] F. Q. Zhao, Z. Chen, J. B. Wang, and C. Zhang, “An improved MOEA/D for multi-objective job shop scheduling problem,” Int. J. Comput. Integ. M., vol.30, no. 6, pp. 616-640, 2017.

[13] O. Schutze, X. Esquivel, A. Lara, and C. A. C. Coello, “Using the averaged Hausdorff distance as a performance measure in evolutionary multiobjective optimization,” IEEE Trans. Evolut. Comput., vol. 16, no. 4, pp. 504-522, 2012.

[14] E. Zitzler, “Evolutionary algorithms for multiobjective optimization: methods and applications,” PhD thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, November 1999.