Research on robotic disassembly sequence planning based on genetic programming

Wangzhao Qin[1], and *Liang* Jin[2,*]

[1]School of Information Engineering, Wuhan University of Technology, Wuhan, China
[2]School of Mechanical and Electronic Engineering, Wuhan University of Technology, Wuhan, China

Abstract. In the context of green recycling and remanufacturing, in order to achieve economic and environmental benefits, the problem of robotic disassembly sequence planning (RDSP) has received more and more attention from researchers. Considering the structure and kinematic characteristics of industrial robots, this paper uses spatial interference matrix to describe the physical constraint relationship between End-of-Life (EoL) product parts, and a multi-objective robot disassembly sequence planning model with the goal of minimizing the switching time of disassembly tools and maximizing the disassembly profit is created. Meanwhile, a multi-objective robot disassembly sequence planning algorithm based on genetic programming (GP) algorithm is proposed to solve the mathematical model. Finally, a case studies based on a gear pump verify the effectiveness of proposed method.

Keywords: Remanufacturing, Robotic disassembly sequence planning, Genetic programming, Multi-objective optimization.

1 Introduction

Disassembly is the first and most difficult step of remanufacturing[1]. Robot automatic disassembly overcomes the shortcomings of traditional disassembly design and researchers have proposed a robot-driven real-time disassembly unit, which can generate the optimal disassembly sequence for EoL products[2].

Robot disassembly sequence planning is a combination of disassembly sequence planning and robot disassembly, the total disassembly time of the robot and the disassembly profit of the robot disassembly sequence planning are two important optimization goals. The moving time of the robot between different disassembly points has some degree of Influence on the above two goals, and it is a part that cannot be ignored. Liu et al. considered the moving path of the robot end effector around the product contour into the total disassembly time[3]. At the same time, many researchers use genetic algorithm RDSP problems, but rarely use genetic programming to solve RDSP problems.In this paper, the improved spatial interference matrix is used to represent the constraint relationship between the various parts of the EoL product to be disassembled.The genetic programming algorithm is used to solve the multi-objective

* Corresponding author: jinl@zfsycf.com.cn

robot disassembly sequence planning problem, and the optimal disassembly sequence is generated.

2 Problem statement

Multi-objective disassembly sequence planning based on industrial robots is a problem of how to find and select the optimal disassembly scheme with specific optimization goals before the disassembly unit of the industrial robot performs the disassembly task.

2.1 Product disassembly constraint model

This paper adopts a matrix-based method to describe the physical constraint relationship between the various parts of the product: obtain the corresponding spatial interference matrix according to the physical constraint relationship between the parts of the product[3], and then use the feasibility analysis method to analyze the feasibility of the spatial interference matrix, and finally obtain a feasible disassembly sequence.

2.2 Mathematical model

Based on the above description, the multi-objective industrial robot disassembly sequence planning problem model studied in this paper is to maximize the profit of robot disassembly and minimize the total switching time of the industrial robot disassembly tool. The mathematical expressions of these two objective functions are as follows:

$$max\,TP = \sum_{i=1}^{N} R_i - p\left[\sum_{i=1}^{N} bt(x_i) + \sum_{i=1}^{N-1} dt(x_i, x_{i+1}) + \sum_{i=1}^{N-1} tt(x_i, x_{i+1}) + \sum_{i=1}^{N-1} mt(x_i, x_{i+1})\right] \quad (1)$$

$$min\,DT = \sum_{i=1}^{N-1} dt(x_i, x_{i+1}) \quad (2)$$

In the formula，x_i means the i-th component is in the disassembly sequence; p means the operating cost of the industrial robot during the disassembly operation of parts per unit time; R_i means recycling/reuse value of component i, which is constant; $bt(x_i)$ means basic disassembly time of component x_i; $dt(x_i, x_{i+1})$ means the switching time of the disassembly direction of the industrial robot from component x_i to component x_{i+1};$tt(x_i, x_{i+1})$ means the switching time of the industrial robot disassembly tool from component x_i to component x_{i+1};$mt(x_i, x_{i+1})$ means the moving time of the robot's obstacle avoidance when disassembling components x_i and x_{i+1}.

3 Solution method

RDSP problem is a multi-objective optimization problem. This paper uses the GP algorithm based on non-dominated sorting to optimize the multi-objective robot disassembly sequence. The corresponding coding method and the genetic selection method are given, the fitness function is defined, the crossover and mutation operator are designed.

3.1 Genetic programming algorithm based on non-dominated sorting

GP algorithm is a kind of evolutionary algorithm (EA), which inherits the basic idea of genetic algorithm (GA) to select and breed children from parents. According to the actual disassembly requirements, this paper combines the NSGA-II algorithm with the GP

algorithm, and proposes the NSGA-II-GP algorithm to search for the optimal disassembly sequence. The main steps of GP are summarized as follows:

Table 1. NSGA-II-GP algorithm.

Algorithm1 : NSGA-II-GP	
Step 1:	Set the number of population to *npop*, the maximum number of iterations is *maxIt*, set the function set ***F*** and the termination set ***T***, the crossover probability is *pCrossover*, the mutation probability is *pMutation*, the initial population generates *npop* feasible solutions, and the initial population is recorded as *popu*.
Step 2:	Non-dominated quicksort and crowding degree calculation for initialized population.
Step 3	According to the results obtained in Step2, the competition is selected in the population, and *n* parents suitable for breeding are obtained.
Step 4	Randomly select *pCrossover* * *n* individuals from the *n* parents selected in Step3 for pairwise crossover operation to obtain the child population *popc*.
Step 5:	Randomly select *pMutation* * *n* individuals from the *n* parents selected in Step3 for mutation operation, and obtain the child population *popm*.
Step 6:	The descendant populations *popc* and *popm* are merged into the population *popu* for non-dominated quick sorting and crowding degree calculation, and the truncate operation is performed on *popu* to keep the population number as *npop*.
Step 7:	If the number of iterations reaches the set maximum number of iterations *maxIt*, the algorithm terminates and outputs the best solution, otherwise returns to Step3.

3.2 Genetic programming coding

The GP algorithm used in this paper is different from the traditional encoding (fixed-length gene) of GA, the individual of GP is a computer program with various forms of expression, and this paper adopts the tree-based genetic programming which is the earliest form and mainstream method of genetic programming.

In the GP algorithm in this paper, as shown in Figure 1, the program is represented in the form of a syntax tree.

In this paper, taking a simple model as an example, a EoL product has six parts, corresponding to the terminator ***x*** which is set in the GP algorithm. The function set is set to ***F***, as shown in Figure 2, the algorithm randomly generates a syntax tree through the terminator and function set and converts it to an expression of string type: (***x***).*exp ((***x***).*sin(***x***)), and substitute the value of ***x*** into the calculation to get the result matrix ***r*** = [2.23, 12.33, 4.58, 0.19, 0.04, 1.12]. Finally, the matrix ***r*** is arranged in descending order to obtain the corresponding index matrix, that is, the disassembly sequence corresponding to the EoL product: 2-3-1-6-4-5, thus, the correspondence and conversion from the syntax tree to the disassembly sequence is realized. At the same time, when generating the disassembly sequence, the generation and feasibility analysis of the disassembly direction will be completed according to the spatial interference matrix. If the resulting sequence is infeasible, repair or regenerate a new syntax tree (as do the crossover and mutation operations below).

3.3 Crossover

The crossover operation includes single-point crossover and two-point crossover. The commonly used crossover is single-point crossover, and the crossover operation used in this paper is also single-point crossover. Since the terminal in this article has only one matrix ***x***,

the cross point is selected in the function node. The process of the crossover operation is shown in Figure 3.

3.4 Mutation

The mutation operation used in this paper is the subtree mutation commonly used in the GP algorithm, the process of mutation operation is shown in Figure 4. The mutation point is randomly selected in the parent syntax tree, and then the randomly generated subtree or node to replace the node and generate a new syntax tree, the expression is (***x***).*exp((***x***).^sin(***x***)).

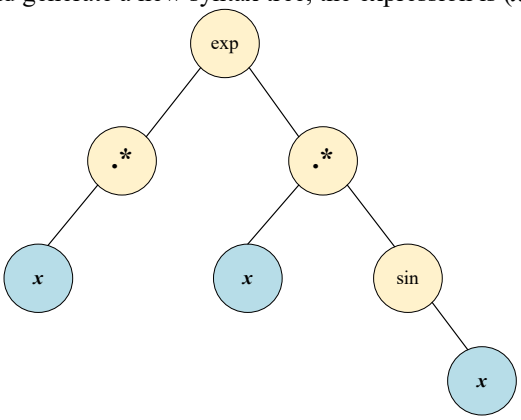

Figure 1. Syntax tree.

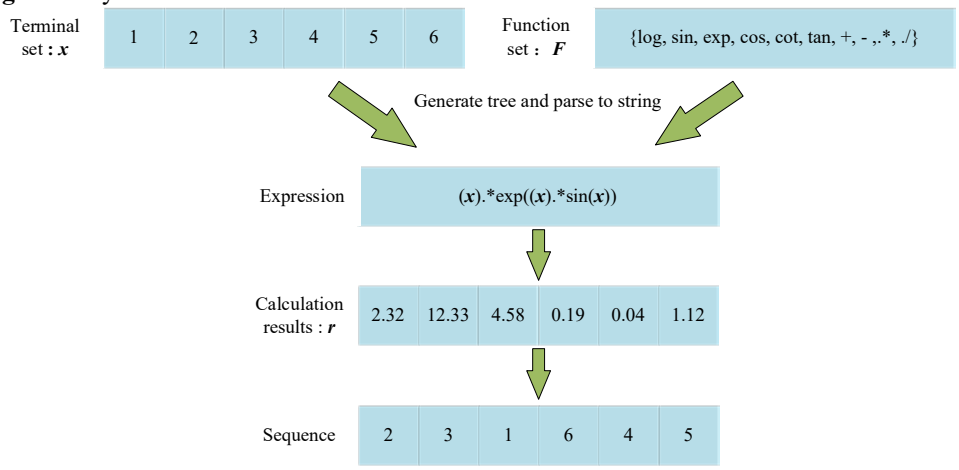

Fig. 2. Encoding process.

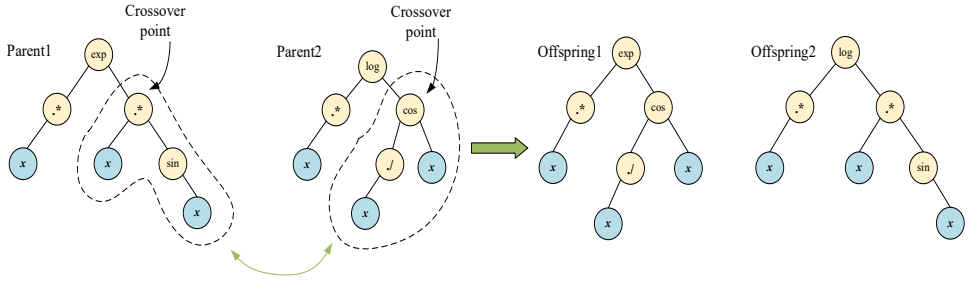

Fig. 3. Crossover operator.

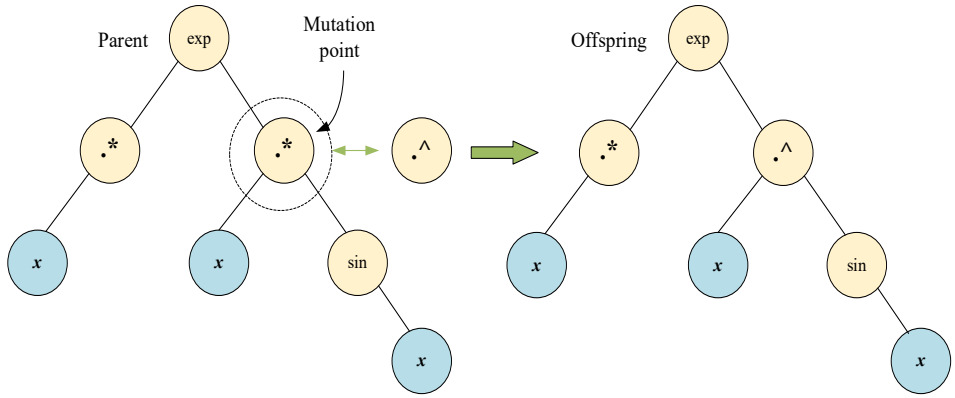

Fig. 4. Mutation operator.

4 Experiment and discussion

4.1 Experimental settings

In this study, we verify the algorithm based on the calculation example of gear pump. At the same time, we choose the NSGA-II and MOABC algorithms to compare with our proposed NSGA-II-GP algorithm, and use the two indicators of IGD and HV to measure the effectiveness of the algorithm. There are many public parameters involved in the operation of the above algorithm, including population number, evolutionary algebra, crossover probability and mutation probability. How to set appropriate public parameters to make the algorithm achieve the best running state is a problem worth considering.

We generate nine different parameter settings through the taguchi orthogonal table. For each parameter setting, we will run the three algorithms ten times to obtain their respective mean IGD (MIGD) and mean HV (MHV), smaller IGD means that the obtained solution set is closer to the true Pareto front, and larger HV means that the solution set has better performance in terms of convergence and uniformity.

4.2 Experimental results and discussion

Table 2. Comparison of MIGD values of three algorithms with different parameters.

NO	Parameter settings	NSGA-II-GP	MOABC	NSGA-II
1	[80 80 0.9 0.1]	**0.02**	0.06	0.85
2	[80 50 0.95 0.05]	**0.06**	0.1	0.89
3	[80 30 0.98 0.02]	0.12	**0.1**	0.64
4	[50 80 0.95 0.02]	**0.77**	0.79	1.06
5	[50 50 0.98 0.1]	**0.08**	0.14	0.96
6	[50 30 0.9 0.05]	1.70	1.62	**1.53**
7	[30 80 0.98 0.05]	**0.67**	0.70	1.10
8	[30 50 0.9 0.02]	**0.12**	0.24	1.22
9	[30 30 0.95 0.1]	0.18	**0.16**	1.08

Table 3. Comparison of MHV values of three algorithms with different parameters.

NO	Parameter settings	NSGA-II-GP	MOABC	NSGA-II
1	[80 80 0.9 0.1]	0.49	**0.73**	0.39
2	[80 50 0.95 0.05]	**0.47**	0.45	0.31
3	[80 30 0.98 0.02]	0.44	0.45	**0.55**
4	[50 80 0.95 0.02]	**0.48**	0.46	0.28
5	[50 50 0.98 0.1]	**0.55**	0.52	0.49
6	[50 30 0.9 0.05]	0.66	1.11	**1.26**
7	[30 80 0.98 0.05]	**0.47**	0.39	0.25
8	[30 50 0.9 0.02]	0.53	**0.72**	0.29
9	[30 30 0.95 0.1]	0.40	**0.41**	0.23

We can observe in table 2 that NSGA-II-GP performs best most of the time, MOABC performs best with the third and ninth parameter configuration, and NSGA-II performs the best with the sixth parameter configuration. We speculate that the number of iterations has the greatest impact on the solving ability of the algorithm. However, the MHV index of NSGA-II-GP in table 3 is average, we speculate that it may be related to the coding structure, and the coding associated with the tree is not prominent enough in the diversity of generated solutions. In the future, we will continue to study the GP algorithm, and further improve the performance of the algorithm by optimizing the selection of function sets, terminatal sets and the coding structure.

5 Conclusion

In this paper, a mathematical model of robotic disassembly is created with the goal of maximizing the profit of robotic disassembly and minimizing the total switching time of industrial robotic disassembly tools. Furthermore, this study is the first application of a genetic programming algorithm to solve RDSP. The experimental results show that the genetic programming algorithm is feasible and effective in solving the RDSP problem.

When using GP to solve RDSP problems, the settings of GP's function set, terminatal set and the coding structure have an impact on the performance of the algorithm. Therefore, how to set appropriate function set, terminatal set and the coding structure for different problems to further improve the problem-solving efficiency of the GP algorithm is our future work.

This work was supported by the National Natural Science Foundation Committee of China under Grant No.52075402.

References

1. Disassembly sequence planning: recent developments and future trends [J]. Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, 2019, 233(5): 1450-1471.
2. Elsayed A, Kongar E, Gupta S M, et al. A Robotic-Driven Disassembly Sequence Generator for End-Of-Life Electronic Products [J]. Journal of Intelligent & Robotic Systems, 2012, 68(1): 43-52.
3. Liu J, Zhou Z, Pham D T, et al. Robotic disassembly sequence planning using enhanced discrete bees algorithm in remanufacturing[J]. International Journal of Production Research, 2018, 56(9-10):3134-3151.