RGV dynamic scheduling model based on kruskal algorithm

Hai Xiao*
School of North China Electric Power University, Baoding 071000, China
*Corresponding author e-mail:1525893608@qq.com

Abstract. In order to solve the problems in traditional processing industry and meet the increasingly urgent demand for product processing performance, intelligent processing has begun to be widely used in industrial production. Today, the Rail Guided Vehicle (RGV) has become one of the main equipment for logistics distribution tasks in intelligent processing systems. It automatically controls the direction and distance of movement according to the command. By planning the dynamic scheduling model of RGV, the output efficiency of industrial production can be greatly improved. Based on the Kruskal algorithm in the greedy algorithm, we design a set of material processing RGV dynamic scheduling simulation model through C++ programming. According to the theory of “local variable optimization and global variable optimization” in the Kruskal algorithm, the shortest path is selected every time the next target of RGV is judged, so that the whole distance traveled is the shortest, wasting shortest time and achieving the highest output.

1. Introduction
Assume that an intelligent machining system consists of 8 CNC machine tools, 1 track-type automatic guided vehicle, 1 track-type automatic guided vehicle linear track, 1 loading conveyor, 1 blank conveyor and other auxiliary equipment. We need to establish its RGV dynamic scheduling model for the material processing operations where there is only one procedure. Dynamic scheduling refers to scheduling performed in the event of unpredictable disturbances in scheduled tasks. Compared with static scheduling, dynamic scheduling can produce a more operational decision-making scheme for the actual situation at the production site. The essence of the RGV dynamic scheduling mode is to decide the shortest path between the RGV and the CNC with the loading and unloading requirements at any time of the system operation. The Kruskal algorithm in the greedy algorithm is applied to this process. According to the principle of the algorithm, we design a set of RGV scheduling simulation program. After inputting relevant working parameters, the RGV optimal scheduling sequence for material processing can be directly generated, thereby improving the processing efficiency.

The Kruskal algorithm is the most commonly used algorithm for implementing the minimum spanning tree of graphs. The basic idea is to always choose the smallest weighted edge that is currently available. When there are v vertices and e edges in the graph, the new graph Graphnew is created, and Graphnew has the same e vertices in the original image, but no edges. Sort all e edges in the original Graph by weight from small to large. Then traverse each edge from the edge with the smallest weight until all nodes in the graph are in the same connected component. If the two nodes connected by this edge are not in the same connected component in the graph Graphnew, add this edge to the graph...
Graph new.[2] The sum of the shortest distances of each segment makes the shortest total distance traversing all the nodes, which is the local optimal solution and the global optimal solution.

According to the conclusion that the local optimal solution in the Kruskal algorithm is the global optimal solution, we can easily infer that if the RGV traverses all eight CNCs according to the principle of the shortest relative distance, the total distance is also the shortest, and the corresponding running time is the shortest, so the number of parts that can be processed in a fixed 8-hour working time increases, and the efficiency of the whole system is also improved. According to the Kruskal algorithm, we use C++ programming to design a set of RGV scheduling simulation programs.

Define the main function `main(t)` with time `t` as the argument. Let $0 \leq t \leq 28800$. Define functions `CNC1()`, `CNC2()`, `CNC3()`, `CNC4()`, `CNC5()`, `CNC6()`, `CNC7()`, `CNC8()` for eight CNCs to indicate whether a CNC sends a demand signal at a certain time. Define the function `move()` and function `dis()` to indicate the number of moving units of RGV at a certain time and the relative distance to a CNC.

After defining the necessary variables and assigning the initial values to them, the CNC() function can be invoked to show the status of all CNC demand signals at any time during one shift of eight hours of work. For all CNCs that send demand signals, invoke the dis() function to determine the distance between RGV and each CNC at that moment and compare with the shortest distance. Then invoke the move() function to let RGV go to the CNC for loading, unloading. Given the relevant parameters of the process, the data results of a shift operation can be output. [3] According to the loading start time of the CNC corresponding to the machining number in the machining process and the cyclic nature of the machining CNC number, the corresponding cutting start time can be derived. [4]

In addition, we used the software Tecnomatix Plant Simulation to simulate the process according to known conditions and parameters. The difference between the simulation results obtained, which are the machining material serial number, the machining CNC number and the loading and unloading start time, and the result of the program is within the error tolerance, which can prove the correctness of this program.

As for the material processing problem of two processes, we found that the RGV scheduling method is consistent with the principle of one process. Therefore, we add the conditions and limitations of the RGV response to the dynamic scheduling model of one process based on the Kruskal algorithm. Then get the RGV optimal dynamic scheduling mode of material processing in two processes.

The idea of establishing the model is consistent with the idea of establishing a model for the processing of one process. It is necessary to decide the shortest path between the RGV and the CNC that sends the demand signal at any point in the system, which is also based on the Kruskal algorithm. However, the processing of two processes should be designed to have different RGV response modes compared to the processing of one process. The reasons are as follows. Due to the different processing time of two processes and the fixed function of each CNC, the number of CNC machines for the first and second processes is different. When the RGV recognizes the signal and judges the next destination, it is necessary to consider not only the two signal states of the CNC (requires material/finish) but also the state of the RGV itself (empty/received the material that has completed the first process). Since the material must be processed in the first process before the second process can be processed, when the CNC responsible for the second process issues a demand signal, the RGV won’t reach it unless it obtains a semi-finished product that has completed the first process. Go to the CNC. Otherwise, it will stay in the same place or go to the CNC responsible for the first process which simultaneously sends the demand signal.

According to the above principles and Kruskal algorithm, we design a simulation program for RGV dynamic scheduling of material processing for two processes. Define the main function `main(t)` with time `t` as the argument. Let $0 \leq t \leq 28800$. Define functions `CNC1()`, `CNC2()`, `CNC3()`, `CNC4()`, `CNC5()`, `CNC6()`, `CNC7()`, `CNC8()` for eight CNCs to indicate whether a CNC sends a demand signal at a certain time. Define the function `move()` and function `dis()` to indicate the number of moving units of RGV at a certain time and the relative distance to a CNC.
During the running of the program, the RGV will accept the demand information of all the CNCs that send the demand signal at any time. At this time, the RGV will discard a part of the signal according to its own situation (the state variable in the program). Specifically, when the RGV state variable is 1, it means that the RGV has obtained the material processed through the first process. At this time, the RGV judges to discard the demand signal sent by the CNC responsible for the first process, and only considers the demand signal sent by the CNC of the second process; when the RGV state variable is 0, it means that the RGV does not obtain the material processed by the first process, then the RGV will discard by judgment. The demand signal sent by the CNC responsible for the second process only considers the demand signal sent by the CNC responsible for the first process, and then judges the next target position according to the Kruskal principle. [5]After the relevant parameters of the work operation are given, the data result of the machining operation of one shift can be output, that is, the machining material serial number, the CNC number of the process 1, the loading start time, the CNC number of the process 2, and the loading start time. According to the loading start time of the CNC corresponding to the machining number in the machining process and the cyclic nature of the machining CNC number, the corresponding blanking start time can be derived. Finally, using the exhaustive method, the yields of all possible CNC distributions are compared, and the distribution pattern of the most processed materials is the optimal distribution.

Also, we used the software Tecnomatix Plant Simulation to simulate the process according to known conditions and parameters. The difference between the simulation results obtained, which are the machining material serial number, the machining CNC number and the loading and unloading start time, and the result of the program is within the error tolerance, which can prove the correctness of this program.

References
[1] https://baike.baidu.com/item/RGV/877626?fr=aladdin
[2] https://blog.csdn.net/zwz2011303359/article/details/63254575
[3] Yan Weimin, Wu Weimin. Data Structure (C Language Edition). Tsinghua University Press. 2007
[4] Chen Weixin, Lin Xiaocha. C++ object-oriented programming tutorial. Beijing: Tsinghua University Press, 2004
[5] Su Shihua and so on. Data structure course design. Beijing: Mechanical Industry Press, 2005