Independent Task Scheduling by Hybrid Algorithm of Harmony Search and Variable Neighborhood Search

Hua Jiang, Liping Zheng, Yun Bao and Yanxiu Liu

College of Computer Science, Liaocheng University, Shandong Liaocheng, P.R.China 252059, China
Jianghua@lcu.edu.cn

Abstract

A new hybrid algorithm is proposed in this paper. An independent task scheduling algorithm is designed based on the hybrid algorithm. The hybrid algorithm adopt list scheduling method to code harmony solution and convert harmony vector to priority-based independent task scheduling model, and perform variable neighborhood search on harmony solutions to improve Harmony Search efficiency and solution quality. The simulation results demonstrate that the proposed algorithm can improve the global search abilities and convergence speed and can escape local minimizer to look for better solutions.

Keywords: *Harmony Search Algorithm; Variable Neighborhood Search; Multiprocessor; Independent Task Scheduling; Hybrid Policy*

1. Introduction

Optimization of task scheduling is an important factor to improve the parallel efficiency in the multiprocessor system [1]. Independent multiprocessor task scheduling is to find an optimal sequence which m independent tasks on n identical processors in order to minimize the makespan. The multiprocessors independent tasks scheduling problem is an optimization problem and is a NP-hard problem [2]. At present, many scholars to solve this problem by using intelligent algorithm and achieved good results [3-5].

Harmony search algorithm (HS) is a swarm intelligence optimization and heuristic global search algorithm. It simulates the phenomenon that musicians repeatedly adjust the pitch of each instrument in the orchestra while playing to get the wonderful harmony. In this algorithm, a new individual is generated by co-operation of all individuals and its local searching ability is enhanced by fine-tuning mechanism. So HS has advantages of simple, good robustness, fast convergence speed etc. Currently, it has been applied in many fields besides solving the optimization problem [6-8].

HS only updates the worst solution in each iteration process, so the individual information of population can not be fully utilized. In addition, the local search ability of this algorithm is limited. Variable neighborhood search algorithm (VNS) is introduced to HS algorithm to enhance the local search capability in this paper. Variable neighborhood search algorithm (VNS) through searching different neighborhood improve the solution quality, and by expanding and adjusting neighborhood range constantly to find the local optimum. VNS has some advantages such as less control parameters, good adaptability and strong robustness, and strong local search ability for continuous increasing neighbor set. Thus, The VNS

has wide range of applications. Combining with the characteristics of HS and VNS, this paper proposes an improved hybrid algorithm to solve the independent task scheduling problem. In hybrid algorithm, the VNS is used for locally searching harmony memory (HM) for its stronger local search ability to improve the search efficiency and the solution quality of HS.

2. The mathematical model of independent task scheduling

Multiprocessor independent task scheduling (MTS) problem is usually indicated by $P_m||C_{max}$. In distributed system, it can be briefly described as follow ：There are n independent tasks named$(\tau_1,\tau_2,...,\tau_n)$need to be processed by m processors with the same performance named$(P_1,P_2,...,P_m)$, and each task can be performed unrelated on any processor. Unfinished task is not allowed to be interrupted while executing. Task is not allowed to be spitted into sub-tasks. The assigned goal is to get a scheduling scheme to make n tasks run on m processors in the shortest time and minimize the total completion time. The n tasks running time is represented by an n-dimensional vector$(t_1,t_2,\cdots t_i \cdots t_n)$, Where t_i denotes the required time for finishing the task i. The processor execution time is the time to complete all tasks assigned on this processor. If k tasks $\tau_{i1},\tau_{i2},...\tau_{ik}$ $(1\le k\le n)$ is assigned on the processor P_j by a scheduling algorithm, the time of completing these tasks is $C_j = t_{j1} + t_{j2} + \cdots + t_{jk}(1 \le i \le k)$, $t_{ij}(1\le j\le k)$ is the time to complete the i-th task. $max(t_{ij})$ is schedule length. So the problem can be described as follow formula:

$$\begin{aligned} &\min \max_{1\le i\le m} C_i = \min \max_{1\le i\le m} \sum_{j=1}^{k} t_{ij} \cdot \\ &s.t.\ \ t_{ij} \ge 0 \cdot \\ &\sum_{i=1}^{m} \sum_{j=1}^{k} \tau_{ij} = n \cdot \end{aligned} \tag{1}$$

3. Design of hybrid VNS and HS algorithm for independent task scheduling

3.1 The basic principle of VNS and HS

3.1.1 HS algorithm: Instrument $i(1,2,...,n)$ is regarded as variable i of optimization, harmony instrument tone $Xi = \{x_{i0},x_{i1},...x_{in}\}$ is solution vector. $f(X_i)$ is evaluation function. The algorithm first produces m initial solutions and they are put in harmony memory *(HM)*. Then, a new solution is searched in the HM by harmony memory considering rate *(HMCR)* or searched in the whole variable range by *1-HMCR*.The new generated solution is locally disturbed by Pitch Adjusting Rate *(PAR)*.Finally, according to the objective function value to determine whether to update HM.

3.1.2 VNS algorithm: The algorithm starts with an initial solution, improves solution quality by searching different neighborhoods, gets the local optimum by constantly expanding the neighborhood range and then changes neighborhood structure based on the optimum until stop criterion is satisfied. VNS often use insert move and swap move to generate neighborhood. Insert move refers to the task of position i is moved to position j $(i\ne j)$ in task sequences of solution s to generate the new solution s', s' is a

neighbor of *s* and all the neighbors of *s* constitutes its insert neighborhood. Swap move refers to the two tasks which has different positions *i* and *j* are selected random in solution *s* to be exchanged and then the new solution *s'* is produced. For the n task, the range of insert move neighborhood is $(n-1)^2$ and swap move is $n(n-1)/2$. In this paper, Multimoves used to consist in performing several moves simultaneously are in a single iteration of algorithm [9].

3.2 The hybrid algorithm of VNS and HS for independent task scheduling

3.2.1 Task Sequence Determination: In this paper, VNS algorithm was used to local search the solutions of HM for its better local search ability. HS has continuous property, so the first step is to convert the tune value of harmony to scheduling sequence. Because the tone value of harmony can be used to express the priority of the task and the tone position can be corresponded to the scheduling task, the maximum position method based on the priority is adopted in the hybrid algorithm to construct task sequence. The tone values of harmony are sorted in decreasing order, the arrangement of corresponding position is a scheduling sequence. According to the scheduling sequence, the machine which has minimum completion time will be assigned to the corresponding task in turn. For example, the four tones value of a harmony shown as X={0.7,0.5,-1,0.3} are related to the four tasks 1, 2, 3, 4 respectively. Scheduling sequence φ of the four tasks is shown in table 1.

Table 1. The sorting process of maximum position

Position of each component in X	Value of each component in X	Scheduling sequence φ
1	0.7	1
2	0.5	2
3	-1	4
4	0.3	3

3.2.2 The Hybrid Algorithm Design: To research the performance of the hybrid algorithm, the 3 kinds of hybrid policies are designed as comparison experiment. First is to search the optimal solution of HS by VNS named VNHS1; second is to search the new solution of HS by VNS named VNHS2; third is to search random selection solution in HM after updating HM with new HS solutions by VNS called VNHS3. This paper VNHS2 is described in detail only, and the algorithm is presented as follows:

- Step1: Initialize the related parameters of HS. Harmony memory size HMS,HMCR, PAR and iteration number NI
- Step2: Initialize HM,HM=$\{X_1, X_2, \cdots \cdots, X_m\}$.The formula of generating initial solution is $x_i(j)=LB_j+r\times(UB_j-LB_j)$, where $x_i(j)$ is the jth solution vector of the ith design variable, r is a random number between (0,1), UB and LB are maximum and minimum value of solution respectively.
- Step3: Generate new solution X_{new} using HS.

```
for j=1, 2…, N do
 if U(0, 1)<HMCR  then  // Evaluate the HM and select
                           the component
 begin
   x_new^j=x_i^j; // i is a random number between[1,HMS]
     if U(0, 1)<PAR  then // Fine tuning disturbance
     begin
       x_new^j=x_new^j ±r×bw;
     Endif
    else // Randomly selected
      x_new^j = LB_j+r × (UB_j - LB_j );//r is a random number
between (0,1) UB and LB are maximum and minimum value of
solution respectively
 Endif
Endfor
```

- Step 4: Convert X_{new} to task sequence $\varphi = (\varphi_1, \varphi_2, \ldots \varphi_n)$ as initial solution of VNS.
- Step5: Perform VNS algorithm to get the optimal solution φ^{best} of neighborhood. Based on the multimoves neighborhood structure, the VNS search processes as follow：

```
i=0;φ^best=φ
While i<k_max do //k_max is the size of N_k(φ)(k=1..k_max)
and N_k is the kth neighborhood of solution φ
Randomly swap two different tasks of φ to get new
sequence φ′;
k=1;
While k≤k_max do
  Search for the optimal solution φ^best in N_k(φ);
  If f(φ^best)< f(φ′) then
     φ′=φ^best;  k=1;
   Else
     k=k+1;
  Endif;
Endwhile;
   i=i+1;
Endwhile;
```

- Step6: Convert φ^{best} to HS solution x^{best} using amendment strategies and update HM.
- Step7: If the iteration number reach the predetermined iterations *NI*, the algorithm is terminated else go to step3.

In VNHS2, only tasks sequence take part in VNS process, so the best sequence of tasks searched by VNS should be turned into HS solution to make HS algorithm run. A amendment algorithm should be designed for transforming task sequence into HS solution see as Figure 1.

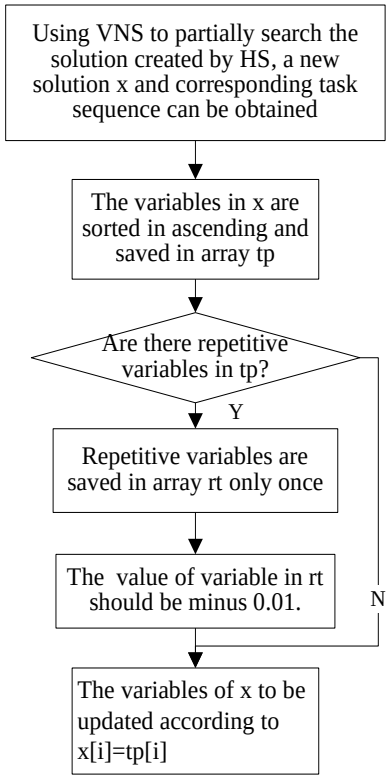

Figure 1. Amendment algorithm flow

3.2.3 HM initiation: This algorithm can be speeded up the convergence effectively depends on the better initial solution. There are two HM initialization methods often be used. One is random initialization, that is the HMS initial solutions of HM are all randomly generated and deposited in HM, another is NEH(Nawaz-Enscore-Ham) method, that is the first solution of HS is generated by NEH method and the remainder solutions are randomly generated in interval [-1,1] [10].Through the experiment, it is indicate that the algorithm convergence rate is dependent on the quality of the initial solution and NEH initialization method is much to be preferred because it can make initial HM not only has greater divergence but also has better solutions.

So in order to ensure the solution in initial HM has the relatively large divergence and has good quality, this paper uses the heuristic initialization method to initialize HM, The solution generated by NEH method is the task sequence, so it must be converted to continuous solution of HS in [-1,1] using formula(2).

$$x(Q(j)) = x^{\min j} + \frac{x^{\max j} - x^{\min j}}{m} \times (Q(J) - 1 + rand), j = 1,2,\cdots m \qquad (2)$$

Where $x(Q(j))$is variable value of the harmony solution in the j-position, $Q(j)$ is the task number of the j-position of the solution got by NEH method, $x^{max\,j}$ and $x^{min\,j}$ are the maximum and minimum values of the variables, rand is a uniform distribution random number between 0-1.

3.2.4 Related Parameters Design: For multiprocessor scheduling has different scale, the reasonable parameter adjustment can great influence on the efficiency and quality of the algorithm. The different parameters design as following.

HMS scale. HMS is a constant in basic HS algorithm, but in the combinatorial optimization problems, HMS is adopted according to the scale of the problem. In order to easily compare with other algorithms, this paper set HMS as 2 times of the number of processors.

HMCR parameters. *HMCR* decides the generate way of new solution. Each harmony vector are dependent on the *HMCR* as new solution is presented, so the value of HMCR is often relatively large.

PAR parameter. PAR is used to control the disturbance operation of harmony vector, and it determines the size of the search area. When the *PAR* value is large, the algorithm is better for searching in the global scope, otherwise, for searching in the local range. In the early part of the algorithm search, the harmony vector in *HM* distributes very separately, so HM search range is larger, and *PAR* value is adopted smaller. While with the algorithm running, harmony vector in HM will tend to be similar, and *HM* scale can become smaller, so *PAR* can be used larger value to prevent the algorithm into local minimum. So the value of *PAR* should be constant changed according to the number of iterations. This paper, the parameter *PAR* can be adjusted by formula (3).

$$PAR_t = PAR_{\min} + \frac{(P_{\max} - PAR_{\min})}{NI} \times t \tag{3}$$

Where PAR_t is the fine-tuning probability of the t-th cycles. PAR_{min} and PAR_{max} are the lower and upper bounds of the *PAR*.

bw parameter. P*arameter bw* is used to control of the disturbance amplitude of candidate solutions. Using *bw* to disturbance can prevent the candidate solution remains unchanged. In addition, the size of *bw* has bigger impact on the candidate solution, if *bw* is too small, the search is difficult to break the local optimums, instead it will cause random search, therefore, *bw parameter* need to be adjusted dynamically [11]. In this paper, the parameter *bw* is adjusted by formula (4).

$$bw^t = bw^{\max} \times e^{\left(\frac{\ln\left(\frac{bw^{\min}}{bw^{\max}}\right)}{NI} \times t\right)} \tag{4}$$

bw^{min} *and* bw^{max} are the upper and lower bounds of the *bw*, t is current iteration number.

By studying the influence of the parameter *HMCR* and *bw* on the performance of algorithm, the conclusion can be analyzed from the algorithm logic structure, it is that the larger *HMCR* has more opportunities disturb the solution of HM and the smaller *bw* can make the solution change in small perturbation and can change the corresponding task arrangement by fine adjustment. So in simulation experiment, the value of *HMCR* is larger and that of *bw* is relative smaller.

4. Simulation experiment

The algorithm is encoded by VC++6.0. The experiment data refer to reference [12]. With 3 processors and 9 tasks, the time of each task need is 81,40,26,4,65,98,53,71,15. The parameters include $HMCR=0.96, PAR_{max}=0.99, PAR_{min}=0.01, bw^{max}=0.05, bw^{min}=0.0001, NI=50, HMS=6$. The simulation number is 100. By recording the best solution, the worst solution, the best solution number and the average solution, the

performance of different algorithms are compared on independent task scheduling. In order to comparison, these algorithms including three improve algorithms with different hybrid policies in this paper, HS algorithm and DPSO-SA algorithm proposed in reference [5]. The statistics results as shown in Table 2. The VNIHS2 and DPSO-SA convergence rate as shown in Figure 2.

Table 2. Performance comparison of different algorithms

Algorithm name	Average value	The worst solution	The optimal solution	The number of the optimal solution
HS	151.53	155	151	30
VNHS1	151.20	155	151	77
VNHS2	151.13	152	151	87
VNHS3	151.18	153	151	82
DPSO-SA	151.15	152	151	84

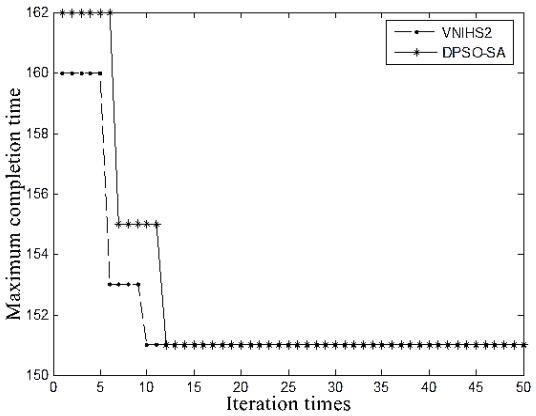

Figure 2. The convergence rate comparison of VNHS2 and DPSO-SA

The results of contrastive experiments results show that the performance of hybrid algorithm proposed this paper is better than others. VNS was embedded in HS to search locally can improve the search efficiency and solution quality of HS because along with constantly extending of the neighborhood scope, the capability of jumping out the local solution can be increased and search efficiency can be improved searching in different neighborhood. The hybrid policy which searches the new solution of HS by VNS has higher stability efficiency and better quality in solving independent task scheduling and the results of fig.1 show that the hybrid algorithm VNIHS2 is superior to DPSO-SA both on the quality of solution and on the convergence rate. And in VNSH2, the HS solution is converted to task sequence only at the search beginning and end, so its search efficiency is higher.

5. Conclusion

According to the existing problems of HS, this article puts forward the hybrid algorithm based on VNS for its stronger ability of local search. Several hybrid policies merge VNS into HS are analyzed and corresponding hybrid algorithms are designed.

The hybrid algorithm combines the advantages of the HS and VNS. By executing VNS search on task sequence of the solution generated by HS, the algorithm executing speed can be greatly speeded up and it can jump local optimum to look for better solutions. The performance of the algorithm is effectively improved. In this paper, the improved algorithm successfully applied in solving multiprocessor independent task scheduling problem. The simulation results and comparison with other algorithms verify the validity and superiority of the improved hybrid algorithm in independent task scheduling.

Acknowledgements

Liaocheng University Research Project Foundation. (X10018).

Shandong Province Higher Educational Science and Technology Project(J11LG02)

Key Laboratory of Intelligent Information Processing and Network Security in Liaoc heng University.

References

[1] O. Sinnen, A. Sousa and E. Sandnes, "Toward a realistic task scheduling model", IEEE Transactions on Parallel and Distributed Systems, vol. 17, **(2006)**, pp. 263-275.
[2] W. Ling, "Intelligent Optimization Algorithm and its Application", Tsinghua University Press, Beijing, **(2001)**.
[3] H. J. Sung, Y. K. Tae, K. K. Jae and S. L. Jong, "Study of Genetic Algorithm-based Task Scheduling for Cloud Computing", International Journal of Control and Automation, vol. 4, no. 5, **(2012)**, pp. 157-162.
[4] H. Hadis and C. Abdolah, "Scheduling in Multiprocessor System Using Genetic Algorithm", International Journal of Advanced Science and Technology, vol. 43, no. 6, **(2012)**, pp. 81-94.
[5] C. Jing and P. Quanke, "A Discrete Particle Swarm Algorithm for Independent Task Scheduling", Computer Engineering, vol. 34, no. 6, **(2008)**, pp. 214-218.
[6] Z. Geem, J. Kim and G. Loganathan, "A new heuristic optimization: application to pipe network design", International journal of model Simulation, vol. 22, no. 2, **(2002)**, pp. 125-133.
[7] M. P. Saka and O. Hasancebi, "Adaptive Harmony Search Algorithm for Design Code Optimization of Steel Structures", Studies in Computational Intelligence, vol. 239, **(2009)**, pp. 79-120.
[8] S. O. Degertekin, "Optimum design of steel frames using harmony search algorithm", Structural and Multidisciplinary Optimization, vol. 36, **(2008)**, pp. 393-401.
[9] P. Quanke and Z. Jianying, "A Variable Neighborhood Search for No-wait Flow Shop Scheduling", China Mechanical Engineering, vol. 17, no. 16, **(2006)**, pp. 1741-1743.
[10] B. Qian, L. Wang, D. X. Huang and X. Wang, "An effective hybrid DE-based algorithm for flow shop scheduling with limited buffers", International Journal of Production Research, vol. 47, no. 1, **(2009)**, pp. 1-24.
[11] M. Mahdavi, M. Fesanghary and E. Damangir, "An improved harmony search algorithm for solving optimization problems", Applied Mathematics and Computation, vol. 188, **(2007)**, pp. 1567-1579.
[12] G. Shang and Y. Jingyu, "Solving Multiprocessor Scheduling Problem by Particle Swarm Optimization Algorithm", Computer Engineering and Application, vol. 41, no. 27, **(2005)**, pp. 72-73.

Author

Hua Jiang, female, associate professor, was born in Liaocheng of Shandong Province in 1971. Her research areas include Intelligent Information Processing and semantic web.