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

Hua Jiang1, Liping Zheng1, Yun Bao1, and Yanxiu Liu1,

1 College of Computer Science, Liao-cheng University, Shandong Liaocheng, P.R.China 252059, China

Abstract. A new hybrid algorithm combining with characteristics of Harmony Search and Variable Neighborhood Search is proposed. Based on the hybrid algorithm, the independent task scheduling problem of multiprocessors is researched based on the hybrid algorithm. The variable neighborhood search is performed on harmony solutions to improve harmony search efficiency and solution quality. The simulation results demonstrate that the hybrid 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]. 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. A new individual is generated by co-operation of all individuals and its local searching ability is enhanced by fine-tuning mechanism in HS. HS has advantages of simple, good robustness and fast convergence speed etc. HS only updates the worst solution in each iteration process, so the individual information of population can not be fully utilized and the local search ability of this algorithm is limited. 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

1 Liao-cheng University Research Project Foundation. (X10018). Shandong Province Higher Educational Science and Technology Project(J11LG02) Key Laboratory of Intelligent Information Processing and Network Security in Liao-cheng University.
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 $P_m||C_{\text{max}}$. 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 $(t_1, t_2, ..., t_n)$, 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 $\tau_{ij}, \tau_{ij2}, ..., \tau_{ijk}$ $(1 \leq k \leq n)$ is assigned on the processor $P_j$ by a scheduling algorithm, the time of completing these tasks is $C_j = t_{ij1} + t_{ij2} + ... + t_{ijk}$ $(1 \leq i \leq k)$, $t_{ij} (1 \leq j \leq k)$ is the time to complete the $i$-th task. $\text{max}(t_{ij})$ is schedule length. So the problem can be described as follow formula:

$$\min \ max \ C_i = \min \ max \ \sum_{j=1}^{k} t_{ij}. $$

$s.t.$ $t_{ij} \geq 0.$

$$\sum_{i=1}^{m} \sum_{j=1}^{k} \tau_{ij} = n. $$

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

To research the performance of the hybrid algorithm, the 3 kinds of hybrid policies are designed as comparison experiment. VNHS1 is to search the optimal solution of HS by VNS; VNHS2 is to search the new solution of HS by VNS; VNHS3 is to search random selection solution in HM after updating HM with new HS solutions by VNS. 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=$(X_1, X_2, ..., X_m)$. The formula of generating initial solution is $x(j)=LB_j+r \times (UB_j-LB_j)$, where $x(j)$ is the jth solution vector of the
Independent Task Scheduling by Hybrid Algorithm of Harmony Search and Variable Neighborhood Search

ith design variable, \( r \) is a random number between (0,1), UB and LB are maximum and minimum value of solution respectively[4].

- **Step3:** Generate new solution \( X_{\text{new}} \) using HS.
  
  for \( j=1, 2..., N \) do
  
  if \( U(0, 1) < \text{HMCR} \) then  // Evaluate the HM and select the component
  
  begin
  
  \( x_{\text{new}}^j = x_i^j; \)  // \( i \) is a random number between[1,HMS]
  
  if \( U(0, 1) < \text{PAR} \) then  // Fine tuning disturbance
  
  begin
  
  \( x_{\text{new}}^j = x_{\text{new}}^j \pm r \times \text{bw}; \)
  
  Endif
  
  else  // Randomly selected
  
  \( x_{\text{new}}^j = \text{LB}_j + r \times (\text{UB}_j - \text{LB}_j); \) // \( r \) is a random number between (0,1) UB and LB are maximum and minimum value of solution respectively
  
  Endif
  
  Endfor

- **Step 4:** Convert \( X_{\text{new}} \) to task sequence \( \varphi= (\varphi_1, \varphi_2, ..., \varphi_n) \) as initial solution of VNS.

- **Step5:** Perform VNS algorithm to get the optimal solution \( \varphi^{\text{best}} \) of neighborhood. Based on the multimoves neighborhood structure, the VNS search processes as follow:
  
  \( i=0; \varphi^{\text{best}}=\varphi \)

  While \( i<k_{\text{max}} \) do  // \( k_{\text{max}} \) is the size of \( N_{k}(\varphi)(k=1..k_{\text{max}}) \)
  
  and \( N_{k} \) is the kth neighborhood of solution \( \varphi \)

  Randomly swap two different tasks of \( \varphi \) to get new sequence \( \varphi' \);

  \( k=1; \)

  While \( k\leq k_{\text{max}} \) do

  Search for the optimal solution \( \varphi^{\text{best}} \) in \( N_{k}(\varphi) \);

  If \( f(\varphi^{\text{best}}) < f(\varphi') \) then

  \( \varphi' = \varphi^{\text{best}}; \ k=1; \)

  Else

  \( k=k+1; \)

  Endif;

  Endwhile;

  \( i=i+1; \)

  Endwhile;

- **Step6:** Convert \( \varphi^{\text{best}} \) to HS solution \( x^{\text{best}} \) 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 VNSH2, the HS solution is converted to task sequence only at the search beginning and end, so its search efficiency is higher.
4 Simulation Experiment

The algorithm is encoded by VC++6.0. The experiment data refer to reference [6]. 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 fig.1.

Table 2. Performance comparison of different algorithms.

<table>
<thead>
<tr>
<th>Algorithm name</th>
<th>Average value</th>
<th>The worst solution</th>
<th>The optimal solution</th>
<th>The number of the optimal solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>HS</td>
<td>151.53</td>
<td>155</td>
<td>151</td>
<td>30</td>
</tr>
<tr>
<td>VNHS1</td>
<td>151.20</td>
<td>155</td>
<td>151</td>
<td>77</td>
</tr>
<tr>
<td>VNHS2</td>
<td>151.13</td>
<td>152</td>
<td>151</td>
<td>87</td>
</tr>
<tr>
<td>VNHS3</td>
<td>151.18</td>
<td>153</td>
<td>151</td>
<td>82</td>
</tr>
<tr>
<td>DPSO-SA</td>
<td>151.15</td>
<td>152</td>
<td>151</td>
<td>84</td>
</tr>
</tbody>
</table>

Fig. 1. 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...
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.

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.

References