Front. Electr. Electron. Eng. China (2006) 3: 251–254
DOI 10.1007/s11460-006-0034-z

RESEARCH ARTICLE

QIANG Lei, XIAO Tian-yuan

Cooperated Bayesian algorithm for distributed scheduling problem

Abstract This paper presents a new distributed Bayesian optimization algorithm (BOA) to overcome the efficiency problem when solving NP scheduling problems. The proposed approach integrates BOA into the co-evolutionary schema, which builds up a concurrent computing environment. A new search strategy is also introduced for local optimization process. It integrates the reinforcement learning (RL) mechanism into the BOA search processes, and then uses the mixed probability information from BOA (post-probability) and RL (pre-probability) to enhance the cooperation between different local controllers, which improves the optimization ability of the algorithm. The experiment shows that the new algorithm does better in both optimization (2.2 %) and convergence (11.7 %), compared with classic BOA.

Keywords statistic optimization, distributed scheduling, Bayesian networks, data mining

1 Introduction

BOA belongs to the statistic evolutionary algorithm, which is a very interesting research topic of optimization methods during recent years. Unlike general evolutionary algorithm such as GA, BOA has no mutation operations. It explicitly models the statistic characters of the optimized solutions, and uses the constructed model to guide the further search. Thus it can avoid the break of high order building blocks and behave more efficiently during optimization search [1, 2].

Nowadays, BOA has been widely developed to solve different kinds of problems: mixed continuous-discrete optimization [3], multi-object optimization [4], etc.. But the complexity of constructing the probabilistic model prevents it from being widely used in real practice. Some algorithms apply simple Bayesian networks to reduce the modeling complexity (for example: UMDA [5]). But it also reduces the algorithm's optimization efficiency.

So, in this paper, a new cooperated BOA with reinforcement learning technique is presented. It uses the co-evolution scheme to decompose the global problem into a set of sub-problems, and combines the sub statistic decision models with reinforcement learning strategy, to enhance the algorithm's optimization performance. Experiment results validate the efficiency of the new algorithm.

Translated from *Journal of Tsinghua University* (*Science and Technology*), 2005, 45(10): 1328–1331 (in Chinese)

QIANG Lei, XIAO Tian-yuan (✉)
Department of Automation, Tsinghua University,
Beijing 100084, China
E-mail: xty-dau@tsinghua.edu.cn

2 Bayesian optimization algorithm

Bayesian optimization algorithm is an evolutionary method which uses Bayesian networks to estimate the solutions distribution. For each evolution generation, the BOA selects "good" solutions according to predefined criteria. Then it constructs the network from selected solution sets, using chosen metrics and constraints (techniques for learning Bayesian networks from data set can be found in Ref. [6]).

$$x^{(j)} = \{x_1, x_2, \cdots, x_N\}\ , j \in \{1, 2, \cdots, n\}$$

n is the total number of chromosomes in $x^{(j)}$, N is the number of decision variables in one solution chromosome. A_j represents variable $x^{(j)}$'s variety area, and it can be note that

$$x^{(j)} \in A = \prod\nolimits_{j \in J} A_j$$

Constructing a Bayesian net from such a solution aggregate is to build a directed acyclic graph G (Fig. 1). The nodes in graph G represent the variables, and the directed link from node A to node B represents the probability relation between variable A and B. Then A can be said as the parent variable of B. The directed link can be noted as

$$P\{x_B \mid x_A\} = P\{x_B, x_A\} / P\{x_A\}$$

The construct process of Bayesian network has two steps:

1) Decide the directed links between variable nodes according to minimal description length (MDL) methods [6].

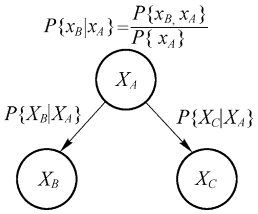

Fig. 1 Bayesian network

2) Decide the probability parameters of the direct links

$$P(X_i = x_i \mid G) = P(X_i = x_i \mid D_i = d_i) \tag{1}$$

Based on the data information of solution aggregate, where D_i is the parent node of variable X_i.

Thus, the Bayesian network models the probabilistic characters of solution variables, and new solutions are produced by sampling from the constructed models. The evolution process then repeat until the stop criteria is met.

Although BOA avoids the break of high order building blocks, the construct process of a global Bayesian net for all the variables is too complicate to get satisfactory results for large-scale optimization problems in the real world. So we introduced a cooperated Bayesian optimization algorithm based on co-evolutionary scheme, which decomposes the complicated problems into a set of more simple sub-problems, and uses a set of cooperated local decision controller to solve them.

3 Co-BOA based on reinforcement learning

3.1 Co-evolutionary scheme

Co-evolution refers to the simultaneous evolution of two or more species with coupled fitness. It includes two kinds of mutual mechanisms: compete and cooperate. Cooperative co-evolutionary algorithms involve a number of independently evolving species, which together form complex structures, well-suited to solving complicated problems. The fitness of an individual depends on its ability to collaborate with individuals from other species. In this way, the evolutionary pressure stemming from the difficulty of the problem favors the development of cooperative strategies and individuals.

Co-evolution scheme decomposes the global problem into a set of sub-problems, thus making the optimization process more simple. And also, it is suitable for problem solving under distributed and parallel environments. But for statistic evolution algorithm such as BOA, if local controllers independently produce the new chromosomes only with their own local models, the global information will be lost. So the independent local controllers need to cooperate with each other to enhance the global optimization performance.

3.2 Reinforcement learning based cooperate strategy for local decision controller

3.2.1 Reinforcement learning based decision algorithm

RL is one kind of unsupervised machine learning algorithm based on asynchronous dynamic planning mechanism.

Define a discrete state aggregate S, and a discrete decision action aggregate A. A state transfer function is: $T: S \times A \to P(S)$ and a reword function is $R: S \times A \to R$.

The decision controller selects action $a \in A$ at state $s \in S$, then the system state transfers to $s' \in S$, and the decision controller gets the immediate reword $r(s,a) \in R$.

Q-learning is a very important reinforcement learning method, which is proposed by Watkins [7]. It can be described as:

$$\begin{aligned} &V(s) = \max_a Q(s,a) \\ &Q(s,a) = r(s,a) + \gamma \sum T(s,a,s')V(s') \end{aligned} \tag{2}$$

$Q(s,a)$ is the long term reward function for action-reward pair (s,a), and γ is the discount factor for the delayed rewards. Watkins proves its convergence [7].

3.2.2 RL- BOA hybrid decision algorithm

For sequence optimization problem, when using the co-evolution scheme, each local decision controller provides sub-chromosome aggregates corresponding to certain sub-problems. Then, sub-chromosomes coming from different controllers combine with each other to form a global chromosome. Then, the global chromosome can be evaluated as a feasible solution for the global problem.

The RL model can make the controller able to learn the optimized strategy from the action-reward history. So, during the cooperated decision process, the RL model can help the controller to decide the optimized sequence actions.

Using RL decision model, suppose $X(j)$ is the feasible action aggregate for a local controller. $Q(s_j, x_i)$ is the long term expected reward for taking task x_i as the *j*th member to process. Then following the RL statistic decision rule, the probability for controller takes task x_i as the *j*th member to process is :

$$P_{\mathrm{RL}}(s_j, x_i) = \frac{Q(s_j, x_i)}{\sum_{x_k \in X(j)} Q(s_j, x_k)} \tag{3}$$

Here, the RL decision model is converted to statistic decision model. $P_{\mathrm{RL}}(s_j, x_i)$ is the decision probability which includes the history information. It can also be viewed as the prior probability for problem solving.

Let $P(x_i \mid G) = P(x_i \mid D_i)$ be the probability distribution for variable x_i and D_i be the parent variables of x_i,

$D_i = D_i^A \bigcup D_i^U$. Here, D_i^A represents the known variables (decision already made) and D_i^U represents unknown variables. Then at state s_j,

$$P_{\mathrm{BN}}(x_i, s_j) = P\{x_i \mid D_i^A\} = \sum_{D_i^U} P\{x_i \mid D_i\} \tag{4}$$

for computing the variable's probability distribution:

$$P(s_{,j}, x_i) = rP_{\mathrm{BN}}(s_{,j}, x_i) + (1-r)P_{\mathrm{RL}}(s_{,j}, x_i) \tag{5}$$

$0 < r < 1$ is the proportion factor.

Here, $P_{\mathrm{BN}}(x_i, s_j)$ can be viewed as the post-probability of problem solving. It describes certain action's fitness level according to its performance. $P_{\mathrm{RL}}(s_j, x_i)$ represents the problem's prior probability summarized from the history and it includes the environment information s_j. So, using the RL-BOA hybrid decision model will help to combine the prior and post probability for the optimization of problem's solution.

Within this hybrid model, let X_{train} be the training case aggregate, then the RL model's update function will be:

$$Q(s_j, x_i) = (1-\alpha)Q(s_j, x_i) + \alpha(r(s_j, x_i) + \gamma \max_{x'_{k'}} Q(s'_j, x'_i)),$$

where $x_i \in X_{\mathrm{train}}$, and r is the reward function.

Using the above update rule will make the RL model get the optimized decision strategy feasible to the problem. It will also help the decision controller to adapt to the dynamic environment.

The steps of the hybrid RL-BOA algorithm can then be described as:

For each local decision controller:

1) Randomly generate initial population and construct BN model using MDL methods.

2) When the new chromosome is prduce, execute Step 3 with probability *p*, and execute Step 4 with probability 1–*p*.

3) Use BN model to generate new chromosome.

4) Use RL-BN hybrid model to generate new chromosome.

5) If the number of new chromosome is enough to form the new chromosome population, execute Step 2, else execute Step 6.

6) For each controller, combine the local chromosome with the representative chromosome of other controllers to form the global solution for the problem, and evaluate its fitness. Then select the best l chromosomes to form local representative chromosome.

7) Update the RL model using the new chromosome and its fitness.

8) If the termination criteria are not met, go to Eq. (2).

4 Simulation and experiments

Consider a manufacturing system with 4 assembly lines in a 2-product planning and scheduling system. The number of workstations in each flow-shop style line is variable. The manufacturing route of product type 1 is (4->2->1), and that of product type 2 is(1,2,->3), as shown in Fig. 2. Process time in each workstation is randomly generated with uniform distribution (Table 1 and Table 2).

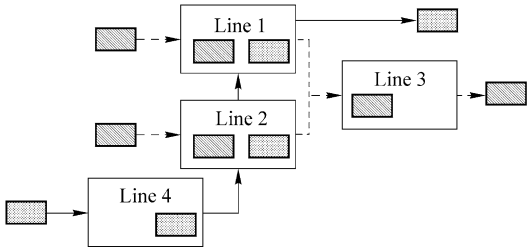

Fig. 2 Manufacturing process

Table 1 Task process time

Task I	Machine *k*			
	1	2	3	4
1	[8, 28]	[12, 38]	[13, 40]	--
2	[5, 23]	[6, 18]	--	[4, 15]

Table 2 Task DeadDate

Task I	DeadDate
1	[60, 180]
2	[40, 130]

Let the task ratio be 1:1, and a test problem is generated (parameters are shown in Table 3). The optimization results are shown in Fig. 3. It can be seen that the optimized output of algorithm Co-BOA is 249.4, and the evolution generated is 132. The optimized output of algorithm BOA is 266.8, and the evolution generated is 154.

Table 3 Illustration example

Task	Machine process time				DeadDate
	1	2	3	4	
1	16	31.5	18.4	--	122.5
2	13.6	29.0	27.1	--	97.2
3	22.5	24.8	28.2	--	155.4
4	17.9	20.9	19.4	--	89.0
5	23.5	18.2	28.5	--	138.7
6	22.8	17.5	--	7.6	109.4
7	7.6	9.5	--	13.9	54.8
8	6.6	15.8	--	4.7	76.8
9	13.8	7.9	--	8.6	77.9
10	8.5	15.6	--	6.6	126.4

Table 4 shows a set of test experiments where the task ratio η_1 varies from 0.1 to 0.9. O_1 is the optimized results and g_1 is the evolution generation when O_1 is first reached. The results show the BOA approach always performs better than GA in solving different task ratio problems.

With reinforcement learning model, Co-BOA can get optimization strategy adaptation to various environments. It can enhance the local decision controller's cooperation and optimization ability. Therefore Co-BOA has more optimization efficiency than general BOA with co-evolution scheme.

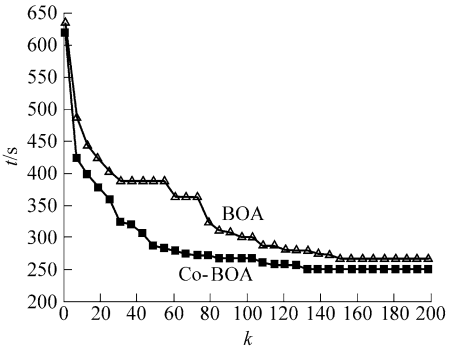

Fig. 3 Experiment results

Table 4 Optimization results

η_1	GA		BOA		Co-BOA	
	O_1	g_1	O_1	g_1	O_1	g_1
0.1	250.5	431	241.4	137	238.1	125
0.3	262.2	382	259.1	144	254.9	128
0.5	270.5	390	264.5	152	260.4	141
0.7	298.2	420	288.1	174	277.4	135
0.9	327.4	385	301.1	165	293.4	143

5 Conclusions

In this paper, a new distributed BOA is presented to overcome the efficiency problem when solving NP scheduling problems. The proposed approach integrates BOA into the co-evolutionary schema, which builds up a concurrent computing environment. A new search strategy is introduced for local optimization process. It integrates the RL mechanism into BOA search processes, and then uses the mixed probability information from BOA (post-probability) and RL (pre-probability) to enhance the cooperation between different local controllers, which improves the optimization ability of the algorithm. The experiment shows that the new algorithm does better in both optimization (2.2 %) and convergence (11.7 %), compared with classic BOA.

Acknowledgements This study was supported by the National Education Promotion Project (No. 081100601).

References

1. Martin P., David E. G., Erick C. P., BOA: the Bayesian optimization algorithm, Proceedings of the Genetic and Evolutionary Computation Conference GECCO-99, Orlando, Florida: Morgan Kaufmann, 1999
2. Zlochin M., Birattari M., Model-based Search for Combinatorial Optimization, IRIDIA/2001-15, Belgium,: Universite Libre de Bruxelles, 2001
3. Oèenášek J., Schwarz J., Estimation of distribution algorithm for mixed continuous-discrete optimization problems, 2nd Euro-International Symposium on Computational Intelligence. Kosice, Slovakia: IOS Press, 2002: 227–232
4. Schwarz J., Evolutionary multi-objective bayesian optimization algorithm: experimental study, Proceedings of the 35th Spring International Conference MOSIS'01. CZ, MARQ, Hradec nad Moravicí, 2001: 101–108
5. Paul T. K., Iba Hitoshi, Linear and combinatorial optimizations by estimation of distribution algorithms, 9th MPS symposium on Evolutionary Computation, Japan: IPSJ , 2002
6. Suzuki J., Learning Bayesian belief networks based on the MDL principle: an efficient algorithm using the branch and bound technique, IEICE Transactions on Information and Systems, 1999, E82-D(2): 356–367
7. Watkins C., Q-learning, Machine Learning, 1992, 8(3-4): 279–292