基于 Backbone 的空间收缩与划分算法

高永超，钱　恒，刘丽梅，王云争，王　玎
（山东省标准化研究院，济南 250014）

［摘要］ 搜索空间的规模和复杂程度是决定问题求解难度的重要因素，而解空间的信息往往可以引导搜索找到最优解。在已知 JSP 空间结构的基础上，提出一种空间收缩与划分算法。算法利用搜索算法获得的较优解，结合组合优化问题解的 backbone 的概念，将搜索空间收缩并划分为一个或多个优解域，在优解域内再进行小规模问题的优化。该算法不必在求解前或求解过程中进行大量的统计分析工作，可以利用求解信息对解空间的地形进行估计，提高求解速度和解的质量。实验结果也证明了算法的有效性。

［关键词］ 车间作业调度问题；解空间；Backbone
［中图分类号］ TP278 ［文献标识码］ A ［文章编号］ 1009－1742(2009)09－0074－04

1　前言

优化问题的搜索空间庞大复杂是问题难解的重要原因，为了设计有效的算法，必须或明或暗地对搜索空间结构做某些假设。了解搜索空间结构有助于理解问题求解的难度，设计合适的搜索策略，提高求解效率。由于组合优化问题具有特殊的排列组合性质，可以考虑特性算法，减少求解的盲目性。

近年来，组合优化问题解的 Backbone 尺寸的研究[1]得到越来越多的关注，Backbone 是指所有组合优化问题的最优解中包含的相同模式、结构或部分，其信息能用于引导搜索找到最优解[2]。而 Kilby 等[3]证明计算 Backbone 是 NP 难的，用别的办法近似 Backbone 又难以保证性能。

车间作业调度问题（job－shop scheduling problem，JSP）是最难解的组合优化问题之一，由于有强烈的实际应用背景，该问题已被广泛研究，其模型有整数规划模型、线性规划模型、非连接图模型，求解方法主要有启发式方法、局部搜索算法以及遗传算法等随机搜索算法等。笔者利用搜索算法获得的近优解信息，结合解的 Backbone 的概念，估计最优解在解空间中的分布，判断 JSP 的解空间的结构类型，将收缩后的搜索空间自动地划分为一个或多个最优解域，再进行局部优化，以快速地获得高质量的解。

2　空间收缩与划分算法

在 JSP 问题中，以工序加工的先后顺序关系表示解，对某工序而言，另一工序排列在该工序之前则变量值为 1，在后则变量值为 0。JSP 的最优解的 Backbone 的定义为所有最优解中相同位置变量值都为 1 的变量，即最优解中包含的模式，定义模式的尺寸为相同位置变量值都为 1 的变量的个数占 1 的总数的比例。

定义 1　将所有机器上的工件工序排成一列，加工顺序相同的工件工序的排列段，称为解中包含的相同模式。

定义 2　模式尺寸 $\rho = \frac{n_{\text{same}}}{n}$，其中 n 为方格数值为 1 的总数，n_{same} 为相同位置方格为 1 的数目。

例如，下面是三个工件、两台机器的 JSP 问题的两个解，括号中的数字表示工序号，括号外的数字表示工件号，U1 表示机器 1，U2 表示机器 2。

解 1　U1：1(1)－3(1)－2(2)－1(3)
　　U2：2(1)－1(2)－3(2)

［收稿日期］ 2008－03－13；修回日期　2008－07－03
［基金项目］ 国家“八六三”计划资助项目（2006AA04A129）
［作者简介］ 高永超（1972－），女，山东平原县人，博士，山东省标准化研究院工程师，主要研究方向为现代优化算法；E－mail：gaoyc@sdis.cn

解 2　U1:3(1) -1(1) -2(2) -1(3)

U2:2(1) -1(2) -3(2)

表 1 中的 1 代表该行工件工序的加工顺序在该列工件工序之前,如第一行中,1(1)在 1(3),2(2)和 3(1)之前加工,数字 1 后的√符号表示这两个解的相对应方格(位置)中的变量值都为 1。

表 1　加工顺序变量表

Table 1　Variables of production order

U1	1(1)	1(3)	2(2)	3(1)	U2	1(2)	2(1)	3(2)
1(1)	0	1√	1√	1	1(2)	0	1√	1√
1(3)	0	0	0	0	2(1)	0	0	0
2(2)	0	1√	0	0	3(2)	0	1√	0
3(1)	0	1√	0					

U1	1(1)	1(3)	2(2)	3(1)	U2	1(2)	2(1)	3(2)
1(1)	0	1√	1√	0	1(2)	0	1√	1√
1(3)	0	0	0	0	2(1)	0	0	0
2(2)	0	1√	0	0	3(2)	0	1√	0
3(1)	1	1√	1√	0				

例如,上文的例子中,两个解中包含的相同模式为:* -2(2) -1(3),2(1) -1(2) -3(2) ,其尺寸为

$$\rho = \frac{n_{\text{same}}}{n} = \frac{8}{9} \times 100\ \% = 0.89$$

以加工顺序变量表格表征的解与在机器上按工件序号排列的解一一对应,工序号所在行的 1 的个数 $\sum x(i,j)$ 表征了该工序在机器上的加工顺序 s,它们之间存在如下关系:

$$s = m - \sum x(i,j)$$

其中, m 为该机器上加工的工件工序数量总和。

因此,利用智能搜索算法对大规模问题的求解能力,用不完全演化算法快速得到多个较优解。此处的不完全演化算法是指当遗传算法等演化算法运行到一定时间后,个体分散在各个峰值附近,此时停止演化,则形成了不完全演化算法。比较这些好解,得到其中包含的相同模式,对搜索空间的结构进行估计。根据 JSP 解空间的结构,存在两种情况,分别对应两种最优解的分布特征。

1)近优解的大部分工序排列是相同的,即模式的尺寸较大,则表明搜索空间只有一个最优解域,问题只有一个全局最优解,或者多个全局最优解聚集在很小的区域内。比较这些好解得到的模式,固定这些局部排列片段不变,与其他工序进行再排列,则问题规模减小,可以采用局部优化算法,最终得到最优解。

2) 若较优解可以分成多个子集,在子集内的解有较多相同的部分,但是不同子集的解之间相同的部分甚少,则表明搜索空间中存在多个分散的最优解,或多个分散的目标值很接近全局最优解的局部最优解。

这时,根据解之间相同模块部分的多少,将较优解分成多个子集,可以认为是多个最优解域,在每个子集内比较该集合内的好解,得到该局部区域内解的相同模式,将这些模式片断固定,采用局部搜索性能优越的优化算法(局部优化算法)求解规模减小的优化问题,得到多个全局最优解或近优解。

了解了 JSP 解空间的结构,就可以设计空间收缩与划分算法更有效地求解 JSP。

1) 在搜索空间均匀产生 N 个随机解;设定最优解模式的水平参数 ρ 的阈值 ħ , ρ 表示解的相同模式占问题规模的比例。

2) 用不完全演化算法搜索得到多个较优解。

3) 比较这些较优解,根据解的相同模式的尺寸估计搜索空间的地形;若绝大多数较优解的相同部分比例 $\rho \geq ħ$,即 backbone 尺寸较大,则空间可以看作一个最优解域,转步骤 4;否则,空间存在多个最优解域,转步骤 5。

4) 固定找到的解的模式,采用局部优化算法优化规模减小的 JSP 问题。

5) 将解按照相同部分的 Backbone 的水平参数 ρ 分成多个子集,归纳每个子集内解的 Backbone 并固定,进行再优化;转步骤 4。

3　分析与仿真

3.1　算法优势

新算法无需事先利用复杂的统计分析方法分析搜索空间的结构特征,算法可以在搜索得到多个较优解后,对搜索空间进行估计,判断解空间是有一个最优解域,还是有多个分散分布的目标值相近的多个优解。如果在比较好解的时候判断错误,即解空间有一个最优解域,被错误地判断为多个,在局部优化后解仍会集中到唯一的最优解域内。

利用近似算法的快速求得近优解的能力,多次运行得到好解,再通过比较好解的相同模式,可以减小再优化问题的规模至 (1 - ħ) 以下,对大规模问题有优越的性能。所以两者结合,可以减小问题复

杂度和计算量,提高解的质量,适用于较大规模问题。

3.2 解的最优性

提取最优解包含的模式进行再优化,不会丢失最优解,但是算法利用智能搜索算法得到的好解信息来得到解的模式,所以在理论上无法保证最优解的获得,但是在多次运行不完全演化算法后,集中了较多解的信息,较少次数的不完全演化就可以提供最优解的足够信息[4]。

3.3 参数影响

阈值 ħ 的设定决定了空间被划分成的最优解域的个数,若有经验可循,如用遗传算法求解 JSP 问题时,解能很快到达最优解的 90 %,则 ħ 可设为 0.9,否则可大致设定为[0.6,1.0]之间的某个数。一般来说,若阈值 ħ 设定的较大,本来可看作属于同一个最优解域的解可能会被划分进两个或多个更小的区域;若阈值 ħ 设定的较小,则可能分属于不同最优解域的解划归到一个最优解域内。在后续的局部细优化后,这些最优解域可以再进行合并或细分,这两种情况都不会影响在当前所得精英解的基础上求解的结果,只是后者相对来说计算量要大一些。

3.4 仿真实验

1)不完全演化算法。经验证明,遗传算法适合于解 TSP 和生产调度等组合优化问题,可以极快地达到最优解的 90 %,但要达到真正的最优解,却要花很长时间。在仿真中不完全演化算法采用基于工序号排序的遗传算法[5],当最好解连续 10 代无明显改善时停止演化。设定 Backbone 的水平参数 ρ 的阈值 ħ = 0.9。为了减少计算量,可以首先获得一个或几个较好解,保留目标函数更好的解,构成优解池,比较解的相同模块。

2)局部优化。在优解域内,将各机器上排列的相同位置的工件工序固定,排列其他位置的工件工序,进行局部细优化。对于实际问题来说,如果不严格要求求得所有最优解,可以只选择一个较好解集中的区域,进行局部优化。

3)仿真。首先对 $\frac{N}{M}$ 取值的几种不同情况,设定不同的工件数和机器数,每种情况随机产生 100 个 JSP 问题进行仿真实验,获得的结果如表 2 所示。因为仿真中只保留了好于阈值的解,所以最后确定的优解域数量较少,但这并不影响算法的有效性。

表 2 不同 N/M 取值的仿真结果比较

Table 2 Comparison of simulation results with different N/M

JSP	N	M	子空间个数	优解的 backbone 的最大差值比例/%
	3	15	1	0
$N/M < 1$	4	20	1	0
	5	25	1	0
	5	3	1	0
$N/M = 1$	10	10	2	0.01
	50	50	2	0.03
	15	3	2	0.97
$N/M > 1$	20	4	3	0.97
	100	10	3	0.99

Matthew & Stephen[3]研究了随机产生的作业车间调度问题 JSP 的 Backbone 尺寸、近优调度之间距离的期望值怎样随工件 N 与机器 M 的数量比 N/M 的变化而变化,表明 JSP 的搜索空间的结构是参数 N/M 的函数,因 N/M 不同而具有不同的空间结构。$N/M \leqslant 1$ 时,解空间可以被看成一个“大谷”,Backbone 尺寸达到 90 % 以上;$N/M > 1$ 时,解空间存在多个分散的最优解或目标值相近的多个局部优解,被看成多个分散的“大谷”,Backbone 尺寸较小,$N/M = 3$ 时,Backbone 尺寸已趋近于 0。很多实际 JSP 调度问题($N/M > 1$)的好解之间有很大不同,Backbone 尺寸很小。

下面对一个实际 JSP 问题进行求解,有 6 个工件,5 台机器,则 $N/M > 1$,问题的具体描述见文献[5]。文章算法找到了另外的 Makespan 等于 50 的解,表 3、表 4 列出了两个不同的最优解。

表 3 机器加工过程表

Table 3 Procedures on machines

机器 M		工件在机器上的排序					
1	工件	3	1	3	1	4	2
	工序	3	4	6	4	3	5
2	工件	3	6	5	1	2	4
	工序	1	1	1	2	3	2
3	工件	4	5	2	5		
	工序	1	2	4	4		
4	工件	3	6	3	2	1	5
	工序	2	2	5	2	3	3
5	工件	2	3	6			
	工序	1	4	3			
$C_{\max}$		50					

表 4　机器加工过程表

Table 4　Procedures on machines

机器 M		工件在机器上的排序					
1	工件	3	1	3	1	4	2
	工序	3	1	6	4	3	5
2	工件	3	5	2	1	4	6
	工序	1	1	3	2	2	1
3	工件	4	2	5	5		
	工序	1	4	2	4		
4	工件	3	2	3	1	6	5
	工序	2	2	5	3	2	3
5	工件	2	3	6			
	工序	1	4	3			
C_{max}		50					

4　结语

搜索空间的复杂程度对问题求解难度有很大影响,利用搜索空间的结构特征可以引导搜索找到最优解。利用搜索算法快速获得多个较优解,结合组合优化问题解的 Backbone 的概念,通过比较这些较优解,对收缩后的解空间的结构特征进行估计,划分为一个或多个优解域,再进行局部细优化。空间划分和收缩算法结合了搜索算法对大规模问题的快速求解能力和问题解的性质,不必在求解之前进行大量的统计工作,利用获得的好解的信息对空间进行估计,避免了搜索算法后期收敛速度慢及可能陷于局部最优的问题。收缩和划分空间后,局部优化问题的规模也减小,提高了求解的效率和解的质量。仿真实验也证明了该策略的有效性。

参考文献

[1] Slaney J, Walsh T. Backbones in optimization and approximation [A]. In Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI-2001) [C], 2001: 254-259

[2] Martin O C, Monasson R, Zecchina R. Statistical mechanics methods and phase transitions in combinatorial problems [J]. Theoretical Computer Science, 2001, 265(1-2): 3-6

[3] Matthew J S, Stephen F S. How the landscape of random job shop scheduling instances depends on the ratio of jobs to machines[R]. Pittsburgh: School of Computer Science, Carnegie Mellon University, CMU-CS-05-162, 2005

[4] Zeng Sanyou, Kang Lishan, Ding Lixin. A new method of evolutionary algorithm for mixed-integer nonlinear optimization problem [J]. Wuhan Univ (Nat Sci Ed), 2000, 46(5B): 554-558

[5] 谢胜利, 黄　强, 董金祥. 求解 JSP 的遗传算法中不可行调度的方案[J]. 计算机集成制造系统-CIMS, 2002, 8(11): 902-906

Space contracting and decomposing algorithm based on Backbone

Gao Yongchao, Qian Heng, Liu Limei, Wang Yunzheng, Wang Ding

(*Shandong Institute of Standardization, Jinan 250014, China*)

[**Abstract**] The scale and complexity of search space are important factors to decide the solving difficulty of an optimization problem. The information of solution space may lead searching to optimal solutions. Based on the known space structure of job shop scheduling problem, a space contracting and decomposing algorithm is proposed. This algorithm makes use of the good solutions found by searching algorithms, contracts the search space and decomposes it into one or several optimal regions combined with the concept of Backbone of combinatorial optimization solutions. And optimization of small-scale problems is carried out in optimal regions. Statistical analysis is not necessary before or through solving in this algorithm, and solution information is used to estimate the landscape of search space, which enhances the speed and solution quality. The experiments results testify the efficiency of this new algorithm.

[**Key words**] job shop scheduling problem; solution space; Backbone