用于求解混合车辆路径问题的混合进化算法

孙 启[1,3] 金 燕[1] 何 琨[1,2] 徐凌轩[1]

(华中科技大学计算机科学与技术学院 武汉 430074)[1] (深圳华中科技大学研究院 广东 深圳 518057)[2]

(莱斯大学计算机科学系 休斯敦 77005)[3]

摘 要 文中研究了具有NP难度的混合车辆路径问题(Mixed Capacitated General Routing Problem,MCGRP),其是在基本车辆路径问题(Vehicle Routing Problem,VRP)的基础上通过添加限载容量约束及弧上的用户需求而衍生的。给定一列车辆数不限的车队,使车辆从站点出发向用户提供服务,服务完用户需求后仍返回站点;规定每辆车的总载重不能超过其载重量,且每个需求只能被一辆车服务且仅服务一次。MCGRP旨在求解每辆车的服务路线,使得在满足以上约束条件的情况下所有车辆的旅行消耗之和最小。混合车辆路径问题具有较高的理论价值和实际应用价值,针对该问题提出了一种高效的混合进化算法。该算法采用基于5种邻域算符的变邻域禁忌搜索来提高解的质量,并通过一种基于路径的交叉算符来继承解的优异性,从而有效地加速算法的收敛。在一组共计23个经典算例上的实验结果表明,该混合进化算法在求解混合车辆路径问题时是非常高效的。

关键词 元启发式,车辆路径问题,禁忌搜索,混合进化算法

中图法分类号 TP183 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2018.04.011

Hybrid Evolutionary Algorithm for Solving Mixed Capacitated General Routing Problem

SUN Qi[1,3] JIN Yan[1] HE Kun[1,2] XU Ling-xuan[1]

(School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China)[1]

(Shenzhen Research Institute of Huazhong University of Science and Technology, Shenzhen, Guangdong 518057, China)[2]

(Department of Computer Science, Rice University, Houston 77005, USA)[3]

Abstract This paper studied an NP-hard mixed capacitated general routing problem(MCGRP), which is derived from the classical vehicle routing problem(VRP) by adding the capacitated constraints and the demands on the arcs. Given a vehicle team with unlimited number, the vehicles serve the customers from the depot and need to return to the depot after completing service. The constraints are that the total load of each vehicle cannot exceed its capacity and each demand can only be served by one vehicle once. This paper aimed to provide a plan of the service route for each vehicle, so that the total travel expenses of all vehicles are minimized when the constraints are satisfied. The MCGRP has a relatively high theoretical value as well as application value. An efficient hybrid evolutionary algorithm was proposed for MCGRP. It applies a variable neighborhood tabu search with five neighborhood operators to improve the quality of solutions, and designs a route-based crossover operator to inherit the high-quality solutions so that the search efficiency can be improved. Experimental results on 23 well-known benchmark instances show that the proposed algorithm is effective for solving the MCGRP.

Keywords Metaheuristic, Vehicle routing problem, Tabu search, Hybrid evolutionary algorithm

1 引言

车辆路径问题在现实生活中的应用十分广泛,针对不同的实际情况,其约束条件和目标也有所不同。研究者应当根据需求对车辆路径问题进行变形,其中最常见的变行便是限载车辆路径问题(Capacitated Vehicle Routing Problem,CVRP)和限载弧路径问题(Capacitated Arc Routing Problem,CARP),目前学术界对这两个问题进行了大量的研究。Golden等人[1]对CVRP问题做了很详细的阐释。在CVRP中,有一列车队对提出需求的顾客提供服务。车队中每辆车的载重上限都相同,并且都从同一个仓库出发,最后回到该仓库。提出需求的顾客可以抽象为一系列的节点,它们分布于由道路连接起来的连通图模型的交叉路口上。该问题的目标是为车队规划服务路线,使其在满足载重条件且完成对所有

到稿日期:2017-05-21 返修日期:2017-07-06 本文受国家自然科学基金项目(61602196,61472147,61772219,61270183),深圳市科技计划项目(JCYJ20170307154749425)资助。

孙 启(1995-),男,硕士,主要研究方向为大规模组合优化问题的智能计算,E-mail:francis_sun@hust.edu.cn;金 燕(1986-),女,博士,讲师,CCF会员,主要研究方向为大规模组合优化问题的智能计算,E-mail:jinyan@mail.hust.edu.cn(通信作者);何 琨(1972-),女,博士,教授,主要研究方向为NP难问题的现实求解、数据挖掘和机器学习;徐凌轩(1994-),男,硕士,主要研究方向为大规模组合优化问题的智能计算。

顾客的服务需求的情况下产生的旅行消耗最小。关于 CARP 问题，Corberan 和 Prins 给出了一个详细的综述[2]。两者的区别在于：CVRP 将需求抽象在节点上，而 CARP 将需求抽象在弧上。

然而，某些特殊约束条件的存在，导致原本适用于 CVRP 或 CARP 的方法无法有效地求解这些问题。例如，Prins 和 Bouchenoua[3] 指出，在许多城市垃圾回收问题中，可以将大多数的垃圾抽象定位到图模型的边上，但从学校和医院产生的垃圾却只能抽象分布到图的节点上。混合图限载车辆路径问题（MCGRP）受到了学术界和企业界越来越多的关注。MCGRP 最初由 Pandi 和 Muralidharan 于 1995 年提出[4]，具有较高的理论研究和应用价值。

1.1　问题描述

混合车辆路径问题的描述如下。

已知：图 $G=(N,E,A)$，其中 N 为节点集合，E 为边集合，A 为弧（即有向边）集合。现有集合 $Nr\subseteq N, Er\subseteq E, Ar\subseteq A$，这些集合中的每个元素都提出了一定量的需要被服务的需求，令 $\tau=|Nr|+|Er|+|Ar|$ 为总需求个数，d_i 表示节点 $i\in Nr$ 上的需求量，d_{ij} 表示边（弧）$\{i,j\}\in Er\cup Ar$ 上的需求量。一列车辆数为 m 的车队从仓库 D 出发，服务若干需求后又回到 D，它们所经过的路径构成了 m 条回路。其中，每辆车的载重上限都是 Q，且 Q 大于任一节点、边和弧上的单个需求，车辆经过边 $\{i,j\}$ 产生的旅行消耗为 c_{ij}。

约束：车辆数 m 不限，但每辆车的总载重不得超过 Q，且每个需求都必须且只被服务一次。

目标：为每辆车规划服务路线，使得所有车辆的旅行消耗之和 z 最小。

1.2　研究现状

由于 MCGRP 是一个 NP 难问题，精确算法只适用于小规模的算例[5]，而现实中的问题规模一般非常庞大，因此现有研究大多采用元启发式算法对该问题进行求解。

Kokubugata 等人采用模拟退火算法框架来求解 MCGRP[6]。他们设计出算法采用特殊的字符串数据结构来表示路径，在许多算例上的计算结果相比之前的工作有很大的提升。Brandao 和 Eglese 采用禁忌搜索来解决 CARP[7]，其允许非法解并为其附加惩罚值的设计思想对后续研究具有启发意义。

Lacomme 等人采用文化基因算法的框架来求解 MCGRP[8]，将车辆路径表示成一条染色体，基于顺序交叉，并采用局部搜索代替变异。该算法在其设计的 23 个算例上得到了很好的结果，但是对于较大规模的算例，还需要更高效的局部搜索和交叉操作来提高算法的搜索效率。

Hasle 等人在其提出的适应性迭代局部搜索算法[5] 中构造了一系列的“destructor”和“constructor”。在每次迭代中，首先使用它们对当前解施加扰动，以达到疏散的目的。接着对解进行局部搜索，以达到增强搜索的目的。值得一提的是，他们每次选择“destructor”和“constructor”时都引入了加强学习的思想，每次迭代之后增加能找到符合预期可行解的“destructor”和“constructor”的权值，从而使得在下次迭代中选择它们的概率变得更大。他们的算法在许多算例上都计算出了新的目标值下界，但通过观察可以发现，对于规模较大的算例，计算结果与求解问题上界的算法计算出的结果之间仍然有很大的差距，个别计算结果与上界的差距甚至超过了上界的 60%。由此可见，针对 MCGRP 的求解算法仍然有着非常大的改进空间。

2　混合进化算法

本文提出了一种混合进化算法，它以基因算法为基础框架，使用禁忌搜索算法代替变异过程。混合进化算法首先使用贪心策略得到规模为 PN 的初始种群，循环执行算法的交叉、局部搜索和种群更新。在循环的每次迭代中，从种群中选择出质量最好的两个解作为父代解，然后使用它们进行交叉得到子代解。接着，对子代解进行局部搜索，局部搜索主要采用禁忌搜索来改进子代解。最后，将改进的子代解与种群中的各个解进行比较，对种群进行更新。整个算法的伪代码如算法 1 所示。

算法 1　混合进化算法

```
输入：G =(N,E,A)
输出：混合进化算法得到的最优解 BEST(solutionMatrix)
1. function HEA()
2. solutionMatrix← GREEDY(Instance,PN),geneIter←0 /* 产生初始种群，初始化种群代数 geneIter */
3. repeat
4.    geneIter←geneIter+1
5.    P1← BEST(solutionMatrix),P2←SECOND_BEST(solutionMatrix) /* 从种群中选出两个父代解 */
6.    C ← CROSSOVER(P1,P2)/* 两个父代解经过交叉操作生成一个子代解 */
7.    C′← LS(C) /* 通过迭代局部搜索提高子代解的质量 */
8.    UPDATE_POPULATION(C′,solutionMatrix) /* 更新种群 */
9. until TIME_OUT() or gene=MAX_GENE
10. return BEST(solutionMatrix)
11. end function
```

2.1　初始种群的构造

基因算法一般会从具有一定规模的初始解种群出发开始整个进化过程。这里将种群的规模设置为 PN（即种群中一共有 PN 个不同的初始解）。由于贪心策略中带有一定的随机因子，因此直接使用 PN 次贪心策略即可得到 PN 个不同的可行解。而且，贪心策略计算出的可行解一般都具有不错的质量。

2.1.1　贪心策略构造初始解

每个解都由一系列的路径构成，其中每条路径都从仓库出发，在选择当前路径上的下一节点时主要依据以下两条规则：

1）若当前节点周围存在未被服务且不会使车辆超载的需求，则优先选择利润值 p 最大的节点作为下一节点，p 的具体定义如下：

$$p_{ij}=\alpha*(d_{ij}+d_j)/c_{ij}$$

其中，i 为当前节点，j 为下一节点，p_{ij} 表示选择 j 作为下一节点获得的利润，α 为 (0,1) 内的随机数，d_{ij} 表示边 $\{i,j\}$ 上未被服务过的需求量，d_j 表示节点 j 上未被服务过的需求量，c_{ij} 表示车辆经过边 $\{i,j\}$ 产生的旅行消耗。

2）如果当前节点周围没有满足条件的需求，并且若车辆

当前载重小于载重上限的一半，则选择当前图中未被服务且不会使车辆超载的需求所在的边或点作为目标区域；否则，直接将仓库作为目标节点。选定目标之后，再在图中搜索出离目标最近的点，即为当前路径的下一节点。需要注意的是，寻找离目标最近的节点时，还需要将当前节点与下一节点之间的距离考虑在内，否则可能造成循环选点。

重复上述两个步骤，直到当前节点为仓库时结束当前路径，开始寻找下一条路径，直到所有需求都被服务，就得到了由一组路径构成的初始解。此外，在选点时加入一定的随机因子可以使贪心策略生成的解带有随机性，这就使得每次执行该过程得到的解具有一定的差异性。

2.1.2 解的表示

由于初始解是通过遍历图模型生成的，因此使用单向链表即可对其进行表示。但在后续搜索过程中由于边和点上都有可能存在需求，若使用单向链表，则需增加额外的属性才可表示。例如，假设边$\{u,v\}$上存在需求，点u上也存在需求，当仅仅移动u上的任务时，需要更新该路径上节点u所代表的意义，即将任务变为单纯的节点；同时将u点加入禁忌列表时，还要标注只对点任务u施加禁忌，而边任务$\{u,v\}$仍可自由移动。

因此，我们采用了 Kokubugata 等人[6]提出的三维结构编码数组来表示解，其编码过程如下：

1)按顺序提取出初始解中的每个任务，并将它们依次编号为 1～τ，将路径起点(也就是仓库)编号为 0；

2)创建一个四维的数据结构来记录单个任务，其第一维记录任务起点，第二维记录任务终点，第三维记录任务是否为弧任务，第四维记录任务量；

3)将解表示为只由任务编号构成的数字序列；

4)新建一个长度与步骤 3)中序列相同的数组，将步骤 3)中生成的解的下标记录到数组中，并作为最终解。

与 Kokubugata 等人使用的数据结构不同的是，为了计算方便，我们还加入了第四维。此外，他们的解也仅仅使用步骤 3)中得到的序列来表示，但是为了后续计算的方便，我们对其进行了第二次编码，将步骤 3)中所得序列的下标作为最终解的表示。

2.2 交叉算符

为了将父代解的优良特征遗传给子代，必须设计出与问题特征密切联系的交叉算符。考虑到 MCGRP 的解由一系列路径构成，很自然能够联想到在对 MCGRP 的解进行交叉时可以以路径为单位进行交叉，而不是像其他交叉算符那样以任务为单位进行交叉。Potvin 和 Bengio[10]在针对 VRP 设计出的算法中首次提出了这种想法，他们将其交叉算符称为“基于路径的交叉(Route-Based Crossover，RBX)”，此后，Chen 等人[11]将其移植到了 CARP 中。他们的实验证明了这种交叉算符对于车辆路径问题的求解效果确实十分良好。因此，我们将其移植到 MCGRP 中，作为本文提出的 HEA 中的交叉算符。

本文采用的 RBX 的具体操作步骤与 Chen 等人提出的交叉算符[11]相似。首先从种群中选择出最优解和次优解作为父代P_1和P_2，然后分别随机从P_1和P_2中挑选出两条路径R_1和R_2，并用R_2代替P_1中的R_1，将得到的结果作为子代C。这时C中可能会出现某些任务被服务了两次和某些任务没有被服务的情况，于是需要对C进行可行性完善。首先通过比较移除后对解的目标值造成的改变来移除解中被服务了两次的任务中使目标值恶化较大的一个；移除完成后，再以随机顺序将未被服务的任务插回解中，插入的位置以插入后解的目标值更优为依据。经过可行性完善之后，便得到了已经成为合法解的子代C。

在本文提出的算法中，为了避免将父代的缺陷集中表现到子代，当P_1和P_2的目标值之差在某一范围内时，我们便认为它们具有一定程度的相似性，并避免使用它们进行交叉。解决的方法是，放弃使用种群中质量次优的解作为P_2，而使用质量最差的解W作为P_2。其原因在于，虽然W的目标值不够理想，但是由于它与当前的最优解P_1之间的目标值之差相对较大，因此在某种程度上它们之间的结构差异也较大，从而使得交配产生的子代能更好地继承父代双方的优良性状。

2.3 迭代局部搜索算法

迭代局部搜索算法(Iterated Local Search Algorithm，ILSA)的核心在于对解空间搜索的“集中”与“疏散”这一过程的设计。“集中”一般由邻域搜索来实现，即对当前解使用某一邻域算符进行局部搜索，以求得更优解。而“疏散”则可以理解为对解使用扰动操作，以使其跳出局部最优陷阱。本文采用以禁忌搜索算法为框架的迭代局部搜索算法。

ILSA 采用禁忌搜索对解进行局部搜索，禁忌搜索使用的邻域算子由随机数决定，若搜索得到更优的解，则更新当前的全局最优解。当搜索发生停滞时，执行 *RAND_STEP* 步随机操作来对解进行扰动操作。当然，随机地对解进行操作时仍然会受到禁忌列表的限制，同时也需要将这些操作录入禁忌列表中，以保证搜索不会陷入循环。扰动完成后，若停滞的次数达到了 *RAND_STAGE*，则采用 Inverse 算符对解进行操作，然后继续进行搜索过程，否则直接返回到局部搜索。按此循环，直到达到停机时间或迭代次数上限为止。整个算法框架如算法 2 所示。

算法 2 迭代局部搜索算法

输入：两个父代解经过交叉操作得到的一个子代解 x

输出：改进后的子代解 x_B

```
1. function ILS(Instance)
2. xB← x,fB← f;
3. iter←1;/* 迭代数初始化为 1 */
4. repeat
5. if randCount / RAND_STEP<RAND_STAGE then
6. if stagCount< STOP then
7. moveType ← RANDOM_CHOOSE(OP_NUM);/* 随机选择一个邻域算符 */
8.    (bestMove,f)← OPT(moveType,x);/* 进行局部搜索，并返回局部最好解 */
9. If(f−fB)>= 0 then stagCount←stagCount+1;
10. else stagCount← 0 endif
11. else
12. randCount ← randCount+1,randMove← randMove+1;
13. if randMove=RAND_STEP then
14. stagCount ← 0,randMove ← 0;
15. endif
```

```
16. bestMove ← RAND_OPT(x);/* 进行扰动操作，以期跳出局部最优陷阱 */
17. endif
18. UPDATE(x,bestMove,moveType);/* 更新局部最优解 */
19. else
20. randCount ← 0;
21.  INVERSE(x);/* 使用 Inverse 操作进行扰动 */
22. endif
23. if z(x) <z(x_B) then x_B← x endif
24. iter ← iter+1;
25. until TIMEOUT() or iter=ITER_MAX;
26. return x_B
27. end function
```

2.3.1　邻域算符

对于序列状的解，目前学术界一般采用 insert，or-opt，2-opt，3-opt，swap 以及 flip(也称作 inverse)等邻域算符对解进行变换。我们采用了以下 5 种邻域算符。

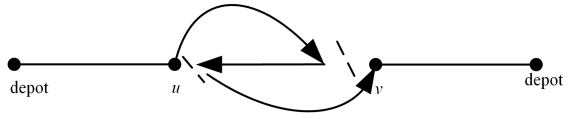
设 u 和 v 为当前解 S 中的任意两个任务(即客户提出的需求)，x 和 y 分别是 u 和 v 的后继任务，$rt(u)$为 u 所在的路径。

1)单个插入(Single Insert，SI)：将任务 u 插入到 v 之后。

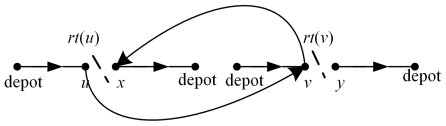
2)双插入(Double Insert，DI)：将任务(u，x)同时插入到 v 之后。

3)交换(Swap)：交换任务 u 和 v 的位置。

4)2-opt：该算符分两种情况，若 $rt(u)=rt(v)$，则翻转解中从 x 开始到 v 的前一个任务为止的所有任务的排列顺序；否则，切断任务 u 和 x 以及任务 v 和 y 之间的连接，并在 u 和 y、v 和 x 之间建立连接，图 1 给出了 2-opt 算符操作的图示。



(a)路径内的 2-opt 算符



(b)路径间的 2-opt 算符

图 1　2-opt 算符示意图

Fig. 1　Illustration of 2-opt operator

5)翻转(Inverse)：翻转某一带有任务的无向边。

可以发现，虽然可以由多次 SI 操作叠加得到与 DI，Swap 和 2-opt 这 3 种操作一样的效果，但是对于单次邻域操作而言，它们对解造成改变的强度是不一样的。因此，在每次迭代中，都随机选用前 4 种算符中的一种作为邻域算符对解进行操作。

至于 Inverse 算符，我们在算法框架成型前通过单个邻域算子进行大量局部搜索测试，发现 Inverse 算子的局部搜索功能非常有限，而且 Inverse 算子的操作目标为带有需求的无向边，然而许多算例中都没有这样的无向边。此外，除了 Inverse 算子之外，其他算子都是双目操作，只有 Inverse 是单目操作，这会对编码实现时的模块化过程造成不必要的麻烦。综合上述原因，本文将 Inverse 算子单独剥离出来作为扰动操作的一部分，没有将其加入到邻域搜索操作中。

2.3.2　扰动策略

首先对解施加固定次数的 SI 算符操作，算符的操作目标为任意、合法(即运算得到的结果满足约束条件)的两个任务。此外，我们还设置了阈值来判定扰动得到的解是否可接受，该阈值在整个搜索过程中并不是一成不变的，而是由当前解的目标值与固定参数 beta 相乘得到的。若新解的目标值与当前解的目标值之差小于阈值，则判定为可接受；否则放弃该解，继续寻找下一对随机任务并进行邻域操作。由于阈值的存在，每次对解的扰动强度不会太大，从而保证了扰动后的解仍然具有良好的性质。

当搜索的停滞次数达到 *RAND_STAGE* 时，在对解施加随机 SI 扰动后，接着执行 Inverse 操作，以弥补局部搜索中对无向边任务进行服务时遍历方向的固定性。另外，由于禁忌列表的存在，扰动结束后的局部搜索不会使解重新回到原来的状态，排除了循环搜索的可能性。

2.3.3　禁忌列表与藐视准则

禁忌对象的选取必须十分严格，才能避免搜索回到原来的解状态而构成循环搜索。因此，在每次邻域搜索完成后，都将这一步中作为邻域算符操作目标的任务计入禁忌列表，由于这些任务无法被选中，因此解不会立即回到原来的状态。另外，我们对禁忌列表中的每个对象的禁忌时长采用随机的策略，并设置参数 *TABU_RANGE*，每当有任务加入禁忌列表时，都将其禁忌时长设置为 0～*TABU_RANGE* 之间的随机数。本文使用的藐视准则也十分简单，当某个新解能使目标值更优时，即可忽略禁忌。

2.3.4　邻域解及其质量的表示

当算例的规模非常大时，如果每次邻域搜索都将邻域中每个合法解的目标值重新计算一遍，将会消耗大量的时间。但通过观察可以发现，一次邻域操作运算之后，解中发生变化的部分是非常有限的，仅仅局限在某几个地方或某几条路径，大部分邻域操作对解的目标值的影响并没有改变。

例如，对于 SI 算符，假设在第一次迭代中将任务 u 插入到 v 之后(以下用 SI(u，v)表示，对解的目标值造成的变化记为 *delta*-SI(u，v))，那么在第二次迭代时，若 SI 的操作对象为 $rt(u)$和 $rt(v)$之外的任务 x 和 y，则 *delta*-SI(x，y)的值并未发生变化。其原因在于：

$$c_{orig}=d(x_p,x)+d(x,x_n)+d(y_p,y)+d(y,y_n)$$

$$c_{opted}=d(x_p,y)+d(y,x_n)+d(y_p,x)+d(x,y_n)$$

$$delta\text{-SI}(x,y)=c_{opted}-c_{orig}$$

其中，x_p 和 y_p 分别为解中 x 和 y 的前驱任务，x_n 和 y_n 分别为解中 x 和 y 的后继任务，$d(x,y)$表示车辆由 x 的尾节点到 y 的头节点产生的旅行消耗。c_{orig} 为 SI 操作之前 x 和 y 处的局部旅行消耗，c_{opted} 为操作之后的局部旅行消耗，由于车辆经过 x 和 y 时的旅行消耗总是不变的，因此可将其从局部旅行消耗中忽略。由上述计算式可知，当 x_p，x_n，y_p 和 y_n 不发生变化时，*delta*-SI(x，y)的值总是不变的，但考虑到车辆载重上限的约束，当 $rt(x)$或 $rt(y)$发生变化时，使用 SI(x，y)对解进行操作后，得到的解有可能从合法解变为非法解(反之亦然)，即使其对解目标值造成的变化仍然不变。

因此，我们通过以下策略使用 SI 算符来进行局部搜索（假设局部搜索只使用 SI 算符）：

1）建立一个 $L\times L$ 的矩阵 *siDeltaMatrix*（以下简称 *delta* 矩阵），并针对初始解，初始化 *siDeltaMatrix* 矩阵；

2）选择 *delta* 矩阵中最小值对应的邻域操作 SI(u,v)，并将其施加到当前解上，将得到的新解作为解的下一状态；

3）对 *delta* 矩阵中 u 附近的任务和 $rt(v)$ 上的任务对应的 *delta*-SI 进行更新；

4）回到步骤 2），开始下一次迭代。

上述步骤中，L 表示最终解 S 的长度，*delta* 矩阵用来记录当前解的 SI 算符邻域中每个解相对当前解的目标值之差，即，用 *siDeltaMatrix*[i][j] 表示将 $S[i]$ 对应的任务插入到 $S[j]$ 之后对目标值造成的变化。因此，每次迭代时对当前解施加矩阵中的最小值对应的邻域操作，即可得到邻域中的最优解；接着对 *delta* 矩阵中部分行和列进行更新，即可开始下一次迭代，使得邻域搜索的计算量大大减少。

类似地，对其他邻域操作时也可以建立相同的 *delta* 矩阵，以此来减少计算时间。当算法交替使用多种邻域操作时，每次对各个 *delta* 矩阵中更新的目标稍有不同，但仍然更新部分值即可。

在设计解的数据结构时，若解序列中存在多个仓库且其编号都为 0，那么对它们进行邻域操作时可能无法区分。例如，将解中第一个 0 插入到任务 t 之后和将解中第二个 0 插入到 t 之后都可以表示为 SI(0,t)，但它们所得到的新解是不同的。因此，以初始解为依据，为解中的每个元素（包括仓库）都加以编号，这样便排除了混淆的可能性。

2.4 种群更新

本文提出的混合进化算法中使用的种群更新策略非常简单，即当对子代解进行局部搜索后得到的改进结果比当前种群中质量最差的解更优时，则使用该子代解替代最差解并保持种群的规模不变，否则丢弃当前子代解。

3 实验

3.1 实验环境及数据集介绍

本算法的测试环境如表 1 所列。

表 1 实验环境

Table 1 Experimental environment

硬件环境	软件环境
主机型号：Dell PowerEdge T620	编程语言：C 语言
处理器：Intel(R) Xeon(R) E5-2640 v2，2GHz，32 core	编译器：gcc4.8.3
内存：64GB	运行环境：Linux3.10.0

车辆路径问题 MCGRP 的算例主要有 Prins 等人[8]提出的 CBMix 和 Bach 等人[12]提出的 DI-NEARP。

CBMix 是随机生成的混合图，它模拟了现实世界中的街道网络，一共由 23 个算例组成。这 23 个算例中，节点的数目在 11～150 之间，边的数目在 29～332 之间，需求数在 20～212 之间。平均来讲，算例中有 50%的边和点上存在需求。

DI-NEARP 共由 24 个算例组成，这些算例来自挪威当地的 6 个报纸派送路径规划问题。算例包含的节点数在 563～1120 之间，边数在 815～1450 之间（不包含弧），需求的数目在 240～833 之间，大概有 1/3 的节点和边上存在需求。

由于 DI-NEARP 中的算例规模都非常大，而我们能够使用的计算资源有限，因此本文只使用 CBMix 作为实验测试算例。因为算法具有很强的随机性，所以将每个算例都计算了 10 次，以使结果更为客观，计算结果的平均值和最优值将在 3.3 节的表 4 中进行列举。

3.2 算法参数设置

局部搜索算法中存在的参数及其设置如表 2 所列。*STOP* 为局部搜索停滞步数，若局部搜索在 *STOP* 次迭代中没有找到更优解，则将其视为停滞；*RAND_STEP* 为随机步长，停滞后使用 *RAND_STEP* 次随机 SI 操作对解进行扰动；*RAND_STAGE* 为 Inverse 的使用频率参数，当搜索停滞 *RAND_STAGE* 次时，对解施加 Inverse 操作；*TABU_RANGE* 为禁忌时长上限，每次对禁忌目标设置的禁忌时长为 0～*TABU_RANGE* 之间的一个随机数；*beta* 为扰动解的接受阈值，若扰动后解的目标值改变量小于当前解的目标值的 *beta* 倍，则认为该解可接受；*ITER_MAX* 为局部搜索迭代次数上限，由于局部搜索是进化算法的一个过程，因此将其停止条件设置为迭代次数达到上限 *ITER_MAX*。

表 2 局部搜索算法的参数设置

Table 2 Setting of parameters in local search algorithm

参数	值
STOP	2
RAND_STEP	2
RAND_STAGE	2
TABU_RANGE	5
beta	0.5
ITER_MAX	10000

除了局部搜索算法中的参数，混合进化算法中还有其他两个参数，其设置如表 3 所列。*MAXGENE* 为最大进化代数，当进化算法迭代 *MAXGENE* 次时算法停止；*TIME_OUT* 为停机时间，参考 Dell'Amico 等人[5]的实验，我们将其设置为 1h。

表 3 混合进化算法的参数设置

Table 3 Setting of parameters in hybrid evolutionary algorithm

参数	值
MAXGENE	1000
TIME_OUT	3600

3.3 实验结果对比与总结

将本文算法与已有文献中最好的 3 种算法进行了对比，对比算法分别是：

1）Prins 和 Bouchenoua[3]设计的文化基因算法 MA，实验使用的 CPU 为主频 1Hz 的 Pentium III；

2）Kokubugata 等人[6]设计的模拟退火算法 SA，实验使用的 CPU 为主频 1.8GHz 的 Pentium IV；

3）Dell'Amico 等人[5]设计的适应性迭代局部搜索算法 AILS，实验使用的 CPU 为 2.40 GHz 的 Intel Xeon(R) E5530。

MA[3]以基因算法为框架，采用了顺序交叉，并对交叉结果施加了纯下降局部搜索，以达到优化种群的目的；SA[6]属

于典型的模拟退火算法，在其中提出了一种针对 MCGRP 的高效数据结构，使得算法具有优良的性质；AILS[5] 是一种带有一定适应性的迭代局部搜索算法，使用了多达 27 种邻域算符对解空间进行有效的搜索。

表 4 列出了 MA，SA 和 AILS 的最优解 z^* 和计算时间 sec_{best}，以及所提出的迭代局部搜索算法与混合进化算法计算出的最优解 z^*、平均解 z_{avg} 和计算时间 sec_{best}。分析表中的数据可以得到以下 5 个结论：

1）本文提出的 HEA 算法能很好地解决混合图上的车辆路径问题，在 4 个算例上已经计算出了目前已知的最优解，且其中之一已经被证明为该算例的全局最优解；

2）混合进化算法的性能比单纯地进行迭代局部搜索的性能更好，其中的交叉算符功不可没，它起到了继承父代解优良特征以及对父代解进行扰动的功能；

3）相对于 Prins 和 Bouchenoua 设计的文化基因算法，本文提出的混合进化算法在绝大多数算例上都计算出了更优的解，其中父代的选取策略、基于路径的交叉算符、类型较多的邻域算符以及禁忌列表的引入起到了很大作用；

4）相对于 Kokubugata 等人设计的模拟退火算法，本文算法具有更强的后续搜索能力，且在大多数算例上都能计算出更优解，这说明基于禁忌搜索的混合进化算法是一种高效的求解车辆路径问题的算法；

5）相对于 Dell'Amico 等人设计的适应性迭代局部搜索算法 AILS，本文算法与其之间的差距都在 5% 以内。AILS 算法采用了 27 种邻域搜索算符，而本文算法只采用了 5 种邻域算符，若采用更多的邻域算符，则有望对目前的最优解进行改进。

图 2 直观地给出了本文提出的 HEA 算法与已有文献中最优的 3 种算法 AILS，MA 和 SA 的计算结果差距百分比的对比情况。

表 4　CBMix 数据集上的测试结果比较

Table 4　Comparison of test results on CBMix dataset

算例名	MA[3]		SA[6]		AILS[5]		ILS			HEA		
	z	sec_{best}	z	sec_{best}	z	sec_{best}	$z*$	sec_{best}	z_{avg}	$z*$	sec_{best}	z_{avg}
CBMix1	2632	108.3	2595	15.1	**2585**	26.4	**2585**	930.8	2588.2	**2585**	1120.0	2586.8
CBMix2	12336	1078.5	12220	661.4	**11794**	1869.2	12170	562.5	12217.7	12079	3296.2	12158.6
CBMix3	3702	157.0	3660	56.0	**3614**	418.3	3666	1589.4	3694.4	3654	1496.4	3676.2
CBMix4	7583	548.1	7641	76.1	**7483**	3384.7	7735	3495.5	7787.3	7719	2080.0	7737.3
CBMix5	4562	100.0	4531	41.5	**4459**	593.1	4587	942.6	4637.6	4531	709.5	4588.9
CBMix6	7087	204.5	7078	98.0	**6969**	1619.0	7109	2188.3	7139.7	7087	2344.7	7106.5
CBMix7	9974	662.6	9615	351.7	**9428**	2516.7	9890	1652.9	9976.6	9775	1797.8	9889.8
CBMix8	10714	767.6	10524	263.8	**10338**	2143.9	10799	1488.7	10859.9	10766	529.8	10858.7
CBMix9	4041	140.8	4103	12.5	**3991**	430.7	4045	355.4	4066.9	4025	3070.4	4042.9
CBMix10	7755	843.2	7687	108.3	**7525**	1399.5	7703	1775.0	7830.8	7669	3552.1	7761.2
CBMix11	4503	414.7	4506	49.8	**4484**	543.8	4535	719.0	4556.6	4503	3240.6	4526.0
CBMix12	3235	71.3	3235	21.4	* **3138**	178.9	3235	0.2	3235	3235	42.4	3235.0
CBMix13	9339	550.6	9133	312.8	**9037**	2840.4	9189	2050.8	9293.1	9227	2868.6	9255.6
CBMix14	8615	357.2	8608	65.3	**8473**	608.7	8606	991.6	8654.2	8553	3090.4	8589.2
CBMix15	8359	390.2	8280	97.3	**8221**	2962.2	8355	194.2	8426.2	8333	3445.8	8401.4
CBMix16	9389	536.1	8886	445.5	**8742**	844.6	9033	1788.3	9180.8	8968	3062.2	9079.5
CBMix17	4165	116.1	4037	43.0	**4034**	62.2	**4034**	1195.2	4047.7	**4034**	674.5	**4034.0**
CBMix18	7411	475.7	7098	278.4	**7052**	2556.7	7247	2367.4	7348.2	7287	3471.5	7350.2
CBMix19	17036	1273.4	16347	469.8	**16155**	451.8	16832	128.0	16961.7	16719	2871.0	16872.7
CBMix20	4918	164.6	4846	50.7	**4738**	577.9	4843	2473.7	4886.7	4839	3434.4	4869.6
CBMix21	18509	1370.6	18069	530.4	**17875**	1012.9	18544	3062.8	18705.6	18415	3371.3	18656.7
CBMix22	1941	65.8	1941	9.5	**1941**	8.8	**1941**	1.6	**1941**	**1941**	23.1	**1941**
CBMix23	* **780**	20.4	* **780**	2.7	* **780**	0	* **780**	0	* **780**	* **780**	0	* **780**
总和/平均	167806	452.9	167689.0	176.6	162811	1176.1	167463	1302.3	168815.9	166724	2156.2	167997.8

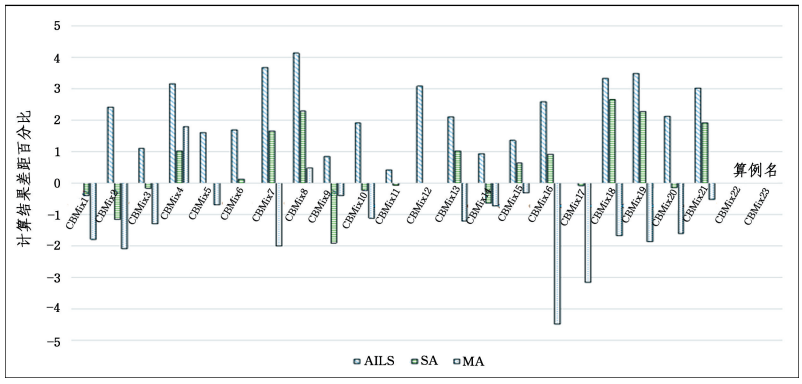


图 2　HEA 算法与其他 3 种算法的比较结果

Fig. 2　Comparative results between HEA and three other algorithms

结束语 本文研究了混合车辆调度问题，该问题是具有NP难的限载车辆路径问题和限载弧路径问题的综合模型。基于该问题的高复杂性，提出了一种混合进化算法。该算法采用了一种基于5种邻域结构的禁忌搜索和随机扰动来提高解的质量，设计了一个基于路径的交叉算符以尽可能保留高质量解的特性。在经典的车辆调度算例上对所提出的算法进行了测试和对比，实验结果表明该混合进化算法可以在较短的时间内得到满意的结果，尤其在4个算例上可以达到最优解。

参 考 文 献

[1] GOLDEN B, RAGHAVAN S, WASIL E. The Vehicle Routing Problem: Latest Advances and New Challenges [M]. Springer US, 2008.

[2] CORBERÁN A, PRINS C. Recent results on arc routing problems: An annotated bibliography[J]. Networks, 2010, 56(1): 50-69.

[3] PRINS C, BOUCHENOUA S. A memetic algorithm solving the vrp, the carp and general routing problems with nodes, edges and arcs[M] // Recent Advances in Memetic Algorithms. Springer Berlin Heidelberg, 2005, 166: 65-85.

[4] PANDI R, MURALIDHARAN B. A capacitated general routing problem on mixed networks [J]. Computers & Operations Research, 1995, 22: 465-478.

[5] DELL'AMICO M, HASLE G, CARLOS J, et al. An Adaptive Iterated Local Search for the Mixed Capacitated General Routing Problem [J]. Transportation Science, 2014, 50(4): 1223-1238.

[6] KOKUBUGATA H, MORIYAMA A, KAWASHIMA H. A practical solution using simulated annealing for general routing problems with nodes, edges, and arcs[M] // Engineering Stochastic Local Search Algorithms: Designing, Implementing and Analyzing Effective Heuristics. Springer Berlin Heidelberg, 2007.

[7] BRANDIO J, EGLESE R. A deterministic tabu search algorithm for the capacitated arc routing problem [J]. Computers & Operations Research, 2008, 35(4): 1112-1126.

[8] PRINS C, BOUCHENOUA S. A memetic algorithm solving the vrp, the carp and general routing problems with nodes, edges and arcs[M] // Recent Advances in Memetic Algorithms. Springer Berlin Heidelberg, 2005.

[9] BOSCO A, LAGANA D, MUSMANNO R, et al. Modeling and solving the mixed capacitated general routing problem [J]. Optimization Letters, 2013, 7(7): 1451-1469.

[10] POTVIN J Y, BENGIO S. The vehicle routing problem with time windows part II: genetic search [J]. INFORMS Journal on Computing, 1996, 8(2): 165-172.

[11] CHEN Y, HAO J K, GLOVER F. A hybrid metaheuristic approach for the capacitated arc routing problem[J]. European Journal of Operational Research, 2016, 253(1): 25-39.

[12] BACH L, HASLE G, WØHLK S. A lower bound for the node, edge, and arc routing problem [J]. Computers & Operations Research, 2013, 40(4): 943-952.

(上接第70页)

于江德，葛彦强，余正涛. 基于条件随机场的汉语词性标[J]. 微电子学与计算机，2011，28(10)：63-66.

[7] COLLOBERT R, WESTON J, BOTTOU L, et al. Natural Language Processing(Almost) from Scratch[J]. Journal of Machine Learning Research, 2011, 12(1): 2493-2537.

[8] ZHENG X, CHEN H, XU T. Deep learning for Chinese word segmentation and POS tagging[C] // Conference on Empirical Methods in Natural Language Processing. 2013.

[9] ZHOU Q, WEN L, WANG X, et al. A Hierarchical LSTM Model for Joint Tasks[M] // Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data. Springer International Publishing, 2016.

[10] HUANG Z, XU W, YU K. Bidirectional LSTM-CRF Models for Sequence Tagging [J]. arXiv Preprint. arXiv: 1508.01991.

[11] BAHDANAU D, CHO K, BENGIO Y. Neural Machine Translation by Jointly Learning to Align and Translate[C] // Proceeding of International Conference on Learning Representations. 2015.

[12] CHENG H, FANG H, HE X, et al. Bi-directional Attention with Agreement for Dependency Parsing[C] // Conference on Empirical Methods in Natural Language Processing. 2016.

[13] RUSH A M, CHOPRA S, WESTON J. A Neural Attention Model for Abstractive Sentence Summarization [C] // Conference on Empirical Methods in Natural Language Processing. 2015.

[14] 宗成庆. 统计自然语言处理[M]. 北京：清华大学出版社，2008.

[15] COTTER A, SHAMIR O, SREBRO N, et al. Better Mini-Batch Algorithms via Accelerated Gradient Methods[C] // Advances in Neural Information Processing Systems. 2011: 1647-1655.

[16] HINTON G E, SRIVASTAVA N, KRIZHEVSKY A, et al. Improving neural networks by preventing co-adaptation of feature detectors[J]. Computer Science, 2012, 3(4): 212-223.

[17] BASTIEN F, LAMBLIN P, PASCANU R, et al. Theano: new features and speed improvements[C] // Deep Learning and Unsupervised Feature Learning, IPS 2012 Workshop. 2012.

[18] ZHU C H, ZHAO T J, ZHENG D Q. Joint Chinese word segmentation and pos tagging system with undirected graphical models [J]. Journal of Electronics & Information Technology, 2010, 32(3): 700-704. (in Chinese)
朱聪慧，赵铁军，郑德权. 基于无向图序列标注模型的中文分词词性标注一体化系统[J]. 电子与信息学报，2010，32(3)：700-704.

[19] WANG Z, XUE N. Joint POS Tagging and Transition-based Constituent Parsing in Chinese with Non-local Features[C] // Meeting of the Association for Computational Linguistics. 2014: 733-742.

[20] YANG L, ZHANG M, LIU Y, et al. Joint POS Tagging and Dependency Parsing with Transition-based Neural Networks[J]. arXiv Preprint. arXiv: 1704.07616.