Robust Multiobjective Optimization for Vehicle Routing Problem With Time Windows

摘要

本文研究的是不确定情况下的带有时间窗的车辆调度问题。为了模拟现实生活车辆调度问题上交通时间的不确定性，本文在solomon benchmark的基础上加上了时间扰动来模拟交通中各种发生的情况，加入的时间扰动的大小由一个最大扰动参数来决定。该问题是要在车辆调度问题中优化两个相互冲突的目标：(1)车辆旅行总距离最小化；(2)车辆使用数量最小化；基于上述基础，本文提出了一种鲁棒多目标粒子群算法，结合了先进的编码方案、新提出的鲁棒性度量指标以及局部搜索策略。算法通过观察粒子在决策空间中的变化来指导鲁棒优化，并且利用提出的鲁棒性度量指标来衡量粒子的鲁棒性，寻找鲁棒最优解。并且为了进一步提高解的性能，还采用了两种局部搜索策略来优化粒子。为了进行对比，本文通过对solomon benchmark上添加扰动构造了新的鲁棒优化问题，和提出的几种算法进行对比。实验结果表明，本文提出的算法能够生成足够的鲁棒解并且保持解的最优性。

关键词：多目标优化；粒子群算法；鲁棒优化；带时间窗的车辆调度问题；

1 引言

车辆调度问题(VRP)是现实生活中一个比较经典的优化问题。具体来说该问题是快递公司打算优化调度计划，包括了制定合适的送货路线和客户服务序列。优化的目标是最小化车辆行驶路线的总距离，并且要保证将商品送到客户手中。随着时代变化，人们对购物有了新的需求，比如客户要求在特定的时间窗口内收到货物，这就将车辆调度问题变化为了带有时间窗的车辆调度问题(VRPTW)，不仅要求优化行驶路线总距离最短，还要求货物在特定的时间窗内要保证送到客户手中。早期有关VRPTW的研究都是集中在单目标VRPTW上，即只要求优化车辆行驶距离最短。随着研究进展，多目标VRPTW的研究渐渐成为主流，该问题不仅需要优化车辆行驶距离，还要优化车辆数量，这些都是和快递公司的利益相关的，得到了关注。

然而，在实际应用中，VRPTW问题难免会遇到不确定性这一因素，这会直接影响优化性能。比如不可预测的交通问题(堵车、道路施工等)可能会造成延误；客户的需求动态波动，时间窗发生改变，需要及时制定新的调度计划等等。在不确定性条件下，由平稳优化方法得到的解不能很好地工作。最终，研究人员转向寻找对扰动不敏感并能在扰动下将其优化性能保持在可接受水平的鲁棒最优解。目前的一些研究都是集中在不确定条件下的单目标VRPTW，仅仅有一篇文献是研究多目标VRPTW的，并且该文献所描述的问题允许违反时间窗的约束，这在现实中是不可行的。为了克服现有MOVRPTW问题描述的缺陷，本文提出了一种新的鲁棒优化问题RMO-VRPTW，由于鲁棒性是指解在扰动下的不敏感程度，如果解的不敏感度高，则解具有鲁棒性。鲁棒优化的目标是充分利用搜索空间，在最小化不确定性影响的同时，确定可能的最优解。RMO-VRPTW的目标包括最小化车辆总数和总距离。为了提高鲁棒性，车辆数量和所有车辆行驶的总距离都必须适度增加，以适应不确定的行驶时

间。此外，不确定性明显加剧了两个目标之间的冲突程度。因此，在RMO-VRPTW中，鲁棒最优解既实现了两个冲突目标之间的权衡，又平衡了最优性和鲁棒性。

在解决优化问题的历史上，进化计算方法已经得到了广泛的应用。本文为了解决RMO-VRPTW问题，通过结合先进的编码解码方法、鲁棒性度量以及局部搜索策略，提出了一种新的粒子群算法(PSO)相关的算法RMOPSO。主要思想是粒子在飞行过程中不仅考虑最优性还考虑鲁棒性，探索鲁棒最优解。

2 相关工作

2.1 VRPTW问题建模

VRPTW问题描述如下，相关变量说明如表1所示。

$$\text{minimize} \quad \mathrm{F}(\mathrm{x}) = (f_1(x), f_2(x) \cdot f_M(x))^T \tag{1}$$

$$\text{subject to} \quad \sum_{\mathrm{j} \in V/D} \mathrm{x}_{0ja} = 1 \quad \forall \mathrm{a} \in B \tag{2}$$

$$\sum_{\mathrm{j} \in V/D} \mathrm{x}_{j(n+1)a} = 1 \quad \forall \mathrm{a} \in B \tag{3}$$

$$\sum_{a \in B} \sum_{\mathrm{j} \in V, j \neq i} \mathrm{x}_{ija} = 1 \quad \forall i \in V/D \tag{4}$$

$$\sum_{i \in V, i \neq \mathrm{h}} \mathrm{x}_{iha} - \sum_{j \in V, j \neq \mathrm{h}} \mathrm{x}_{hja} = 0 \qquad \forall \mathrm{h} \in V/D \tag{5}$$

$$\sum_{i \in V/D} d_i \sum_{j \in V, j \neq \mathrm{i}} \mathrm{x}_{ija} < cap \qquad \forall \mathrm{a} \in B \tag{6}$$

$$ser_i = \begin{cases} lb_i, & arr_i < lb_i \\ arr_i, & lb_i \leq \mathrm{ser}_i \leq ub_i \\ infeasible, & arr_i \geq ub_i \end{cases} \qquad \forall i,j \in V/D \tag{7}$$

$$ser_i + t_{ij} - K(1 - x_{ija}) \leq ser_j \qquad \forall i,j \in V/D \quad \forall \mathrm{a} \in \mathrm{B} \tag{8}$$

$$lb_i \leq ser_i \leq \mathrm{u}b_i \qquad \forall \mathrm{i} \in V/D \tag{9}$$

其中，x_{ija}表示边e_{ij}是否被车辆a通过，通过则为1，反之为0。M为目标个数。式(2)~(9)表示着该问题的约束，式(2)和(3)表示所有路线中车辆必须从车站出发，最终回到车站，并且一条路线只能通过一次车站。式(4)和(5)表示每个顾客必须并且仅被服务一次。式(6)表示了一条路线服务的客户需求总量不能超过车辆的容量。式(8)表示了一条路线上车辆服务的客户是有序的，服务时间不能颠倒。式(7)表示了车辆如果在时间窗下界前到达客户，需要等到时间窗下界再进行服务；如果在时间窗内到达则在到达时间进行服务；如果超出时间窗上限到达则不可行。式(8)结合式(9)说明车辆必须在各个客户的时间窗内为客户提供服务。

表1 VRPTW相关变量说明

V={0,1,2,…,N}	**客户以及车站的总集合**
D=(0)	车站
V/D	客户集合
E={e_{ij}|i,$j \in V$}	所有边的集合
D_i	第i个客户的需求
[lb_i,ub_i]	第i个客户的服务时间窗
DIS_{ij}	e_{ij}的长度
T_{ij}	e_{ij}的耗时
c_{ij}	e_{ij}的消耗
CAP	每辆车的容量
B	车辆集合
R={r_1,r_2…$r_{\|R\|}$}	路径集合

2.2 相关不确定性的添加方法

通过相关文献研究，收集了五种太耐时间扰动的方法如表2所示。其中，第四种不确定相关性的时间扰动添加方式很大程度依赖参数设计，这是比较困难的；第二种方法仅仅使用了有限个候选值来模拟时间扰动，很可能和现实场景不匹配。第一种扰动的不确定性和特定的路线有关，在不同的弧线上可能特征不同，泛用性不强。第三种的扰动则过于片面。由此我们采用了第五种扰动，在规定的最大扰动[- δ max, δ max]范围内随机添加时间扰动来模拟交通状况。

表2 五种时间扰动添加方式

不确定性种类	扰动添加方式
带预算的不确定性	t_{ij}= t_{ij}+β_{ij}*t'_{ij}
离散不确定性	t_{ij}={t_{ij}^1,t_{ij}^2…t_{ij}^N}
根据不同位置决定不同范围的不确定性	常规时$t_{ij} \in [5,10]$ 拥挤时$t_{ij} \in [10,15]$
相关性不确定性	$t_{ij} = t_{ij} * (1 + X)$
最大扰动范围不确定性	$t_{ij} = t_{ij} + \delta$

3 本文方法

本节中提出了针对RMO-VRPTW问题的算法RMOPSO，将在下面分为几个模块来描述。

3.1 粒子编码以及解码方法

算法在开始时候对粒子进行编码，粒子长度等于客户数量，每个维度对应一个客户。粒子的各个维度豆瓣韩一个客户的优先级值，代表客户被服务的优先级，其值越低，客户越早被服务。对于每个粒子，速度和位置初始化[0,1]范围内的随机数并且上下界为[0,1]。由此还需要一种解码策略。本文采用的是一种基于贪心策略的解码器，该贪心解码器根据客户优先级，选取未插入路径前的前K个客户作为待插入客户，如果插入客户满足约束则正常插入路径，否则遍历下一个客户。如果所有K个客户都不能插入当前路径中，说明需要新开一条路径。

贪心解码器伪代码[1]如下：

Algorithm 1 Greedy Decoder

```
Input:
  pop: the particle positions
  K: a predetermined threshold number
  n: the number of customers
  d = {d1, d2,... dn}: demand of customers
  t = {t1,2, t1,3,... tn,n−1}: travel time
1) Sort the customers by priority
   tovisite = {tovisite(1), tovisite(2),..., tovisite(n)} : the customer set sorted by priority
   value
  route : initialized as the empty set for recording route
2) Greedy-decoder
  Add 0 into route
  while tovisite is not empty
    Flag=false
    for i = 1:K //贪心地根据优先级来将客户加入路径
      if tovisite(i) satisfy the constrains
        Add tovisite(i) into route
        timenow = timenow + troute(i−1),route(i)
        loadnow = loadnow + droute(i)
        Remove tovisite(i) from tovisite
        Flag = true
        break
      end if
    end for
    if not Flag //若没有客户能够添加则新建一条路径
      A new route is constructed
      Add 0 into route
      timenow = 0
      loadnow = 0
    end if
  end while
  Add 0 to route
Output:
route: a group of feasible routes decoding from particle
```

3.2 鲁棒优化模块

鲁棒性评价指标Rob用来判断在不确定环境下哪种解具有较强的鲁棒性。在RMO-VRPTW中，当客户到达时间靠近时间窗口中心时，即定义为早期时间lb与最后时间ub之间的平均值，则解决方案更有可能逃离不可行的行程，在这种情况下，即使有干扰，到达时间也很难超过[lbi, ubi]范围。因此，我们可以用到达时间到时间窗口中心的距离来衡量鲁棒度。

$$\mathrm{Rob} = \sum_{i \in V/D} \left| arr_{\mathrm{i}} - \frac{\mathrm{ub}_i + lb_i}{2} \right|$$

在此基础上，设计了鲁棒优化模块，首先是对每个粒子利用粒子群算法进行优化，优化后评估各个粒子的鲁棒性，并更新粒子个体最优。值得一提的是，在更新粒子最优Pbest的时候，如果新粒子的最优性不如当前Pbest，但是鲁棒性比Pbest更好，那么有一定概率将Pbest更新为新粒子。并且在更新存档时如果解的数量大于存档容量，也是根据鲁棒性进行排序来选择留下来的个体。然后有概率地启用局部搜索策略来优化各个个体。这样的策略良好的考虑了鲁棒性。鲁棒优化[1]伪代码如下

Algorithm 2 Robust Multiobjective Optimization

```
Input:
  pop: the new initial population
  maxiter: maximum iterations
  N: the number of particle
  REP: the archive for restoring Pareto solutions
1) Robust multi-objective optimization
  for t = 1:maxiter
    for i = 1:N
      Update popti with PSO
      if popti is infeasible under uncertainty
        popti = popt−1 i
      end if
      evaluate the value of Rob
      update pbesti
      if rand < pl−s
        Apply problem-based local search
      end if
      if rand < pl−s
        Apply route-based local search
      end if
    end for
    Update the archive REP
  end for
Output:
  REP: Final robust optimal solution
```

3.3 局部搜索模块

算法还采用了两种局部搜索策略优化粒子，一种是对粒子的，对粒子中各个维度的优先级进行改变，时间窗上界靠后的客户应该是更加鲁棒的，可以靠后服务，所以要降低其优先级，基于该思想设计了第一种局部搜索策略。另外一种局部搜索策略是基于路径的，因为随机化，很容易产生冗余的路径(一辆车仅服务少数客户)，可以在一条总路径中预先选出需要去除的冗余路径，再对剩余路径进行有关鲁棒性的排序，然后将冗余路径的客户插入到鲁棒性较强的路径中，一旦可行则执行冗余路径的去除，并加入客户到可插入路径中，这样是为了优化车辆使用数量，尽量减少冗余路径达到减少车辆数量的效果。两种局部搜索模块伪代码[1]如下：

Algorithm 3 Localsearch1

```
Input:
  n: the number of customers
  m: the number of customers to be applied local search
  pop = {pop(1), pop(2),..., pop(n)}: a particle
1) Randomly select a set of customers
  indlist = {indlist(1), indlist(2), . . . , indlist(m)} : the customer set
2) Adjust the priority
  for i = 1:m
    Calculate the variation εindlist(i)
    pop(indlist(i)) = pop(indlist(i)) + εindlist(i)
  end for
Output:
  pop: new particle
```

Algorithm 4 Localsearch2

```
Input:
  K: pre-defined customers
1) Remove the route with few customers
  toremove: the selected route including less than K customers
  cus: customers in route toremove
2) Select route with strong robust degree
  toinsert: the route with strong robust degree
3) Insert customers into toinsert
  for all customers in cus:
    for all locations in toinsert:
      Try inset the customer in current location
      if current route is feasible:
        Record current route
      end if
    end for
    Update toinsert as best performance route
  end for
Output:
  toinsert: new route after local search
```

4 复现细节

4.1 与已有开源代码对比

本文并未参考任何相关源代码，在原文的伪代码基础上进行复现并且仅采用了原文提出的相关问题集进行测试对比。本文在原文的基础上，对的鲁棒优化模块中的PSO算法进行了改进，原文采用了基于支配的方法来决定PSO算法中的个体最优和全局最优如何取，本文并未使用支配的方法，而是设计了一个适应度函数，适应度函数综合考虑了两个目标，并且额外将鲁棒性纳入适应度函数之中，将多目标优化问题转化为单目标优化问题，并且该目标充分考虑了原来的两个目标以及个体鲁棒性，在问题集测试中，部分问题比原文更好，另外部分问题则不如原文。

4.2 创新点

原文中的代码使用的PSO模块中，采用的是基于支配的方法决定粒子的局部最优和种群的全局最优的更新方式，复现过程中我采用了一种适应度函数，组成如下：

```
fitness(i)=(f1(i))/customersize+(f2(i)-min(f2))/(max(f2)-min(f2))+(robust(i)-min(robust))/(max(robust)-min(robust));
```

其中f1是车辆个数，f2是旅行距离，最麻烦的情况下是一辆车服务一个顾客，所以第一项相当于一个对车辆个数的归一化。后面是对旅行距离的归一化以及和鲁棒性的归一化，经过调整发现三者比例1:1:1的表现还算可以，由此多目标问题转化为该适应度函数相关的单目标优化问题，并且以此为策略更新个体局部最优和种群全局最优，在实验中得到了良好的效果。

5 实验结果分析

5.1 实验问题

实验所用的测试问题使用的是原文提出的问题。原文在Solomon benchmark的基础上进行修改得到了RMO-VRPTW问题。Solomon benchmark根据问题特性进行了分类，有R1、R2、C1、C2、RC1、RC2类问题。R类问题的客户是随机生成的，C类客户的问题是聚类生成的，RC类问题是随机和聚类混合使用生成的客户位置。1类问题的调度范围比较短，一辆车仅能服务少数客户，时间窗的约束比较严格。2类问题的调度范围比较长，一辆车能服务多个客户。C、RC类问题由于客户是聚类生成的，可能需要同一辆车服务附近的多个客户，增加了调度的难度。文章采用的评估指标除了本身的两个目标函数，还有一个鲁棒性相关函数rob，rob的定义是在50次蒙特卡洛测试中解不可行的次数除以50。该指标越低越好。

5.2 实验参数设置

RMOPSO的客户大小设置为25，种群大小设置为50，最大迭代次数设置为100。PSO有关的参数中，惯性权重w设置为1，个体学习因子c1和群体学习因子c2都设置为2。基于鲁棒性更新个体最优pbest的概率pa为0.7，采用局部搜索的概率为0.9.蒙特卡洛测试的最大代数为50.贪心解码器中搜索范围K设置为5。

5.3 实验结果对比

将复现代码在RMO-VRPTW上运行统计数据后与原文对比，对比结果如表3所示：

在限制较为宽松的r101类问题中，复现代码基本都能表现出比源代码的性能，获得支配源代码的解，即使未支配，也和源代码中的解互不支配。并且随着扰动程度的提升，复现代码展现出更好的性能并且都是完全支配源代码获得的解的，并且还能获得更多数量的鲁棒解。加强时间窗限制的r104问题中，低扰动程度下源代码获得的鲁棒解多于复现代码，但是复现代码获得的解都完全支配源代码获得的解，并且随着扰动程度的提升性能差距越来越明显，说明复现代码能够更好的抗干扰。在r202问题中，调度范围变长，一辆车能够服务更多客户，调度路径的寻找需要更加准确。在这个问题上，复现代码获得的解和源代码获得的解之前互有支配，性能差不多，随着扰动程度的提高也是互相支配。在结合聚类生成策略的rc102问题中，由于聚类生成客户，增加了解生成的难度，两个方法获得的解都很少，仅有1个，但是复现代码获得的解都支配源代码获得的解，并且随着扰动程度的增加这一现象更加明显，即使在高扰动程度3之下，复现代码获得的车辆数更少的解行驶距离也和源代码获得的差不多，抗扰动能力刚好。并且结合所有的问题对比来看，复现代码获得的解都是完全鲁棒的，源代码获得的解有部分是不完全鲁棒的，综合来看，复现代码能更好的权衡鲁棒性和最优性，同时获得更好的解。

表3 实验结果对比

问题和扰动程度		原文			复现代码		
		车辆数	行驶距离	鲁棒性	车辆数	行驶距离	鲁棒性
r10125	δ=1	9	760.077	0	8	778.77	0
		11	758.4251	0.0041	9	726.07	0
	δ=2	9	805.107	0	8	769.72	0
					9	761.05	0
		10	778.6402	0.003	11	738.29	0
	δ=3	10	785.5696	0	8	736.74	0
		11	778.9224	0	10	727.85	0
r10425	δ=1	5	665.33	0	5	655.44	0
		6	658.38	0			
	δ=2	6	749.46	0	5	659.47	0
		7	699.04	0			
	δ=3	6	729.15	0	5	676.99	0
					6	665.72	0
r20225	δ=1	3	693.03	0	2	668.35	0
					3	661.31	0
		4	647.39	0	4	667.24	0
	δ=2	2	753.177	0	2	674.87	0
		3	713.09	0			
		4	697.77	0	3	668.25	0
	δ=3	2	780.13	0	2	734.89	0
		3	596.01	0	3	664.76	0
rc10225	δ=1	5	600.56	0	5	593.27	0
	δ=2	5	616.56	0	5	568.68	0
	δ=3	5	597.11	0	4	602.12	0

6 总结与展望

本文为了模拟现实生活中车辆路径调度问题的特点，制定了一个多目标鲁棒优化问题RMO-VRPTW，其中包括优化两个相互冲突的目标(行驶距离最小化、车辆数量最小化)，同时引入了一种新的不确定性时间扰动形式，更能模拟实际中的交通状况。为了应对RMO-VRPTW，提出了一种新的算法RMOPSO，冲分利用决策空间中的粒子特征来指导鲁棒优化，提出了新的鲁棒性度量指标来平衡鲁棒性和最优性，并且采用两种局部搜索策略优化粒子，提高性能。复现过程中对PSO部分进行了修改，得到了一定的效果提升，在多数情况下都能获得支配源代码的解的解，并且在高扰动范围下表现性能更好。另外，我认为RMOPSO对于鲁棒性的考虑可能不够全面，算法考虑了是到达时间和上下界中的平均值之差作为鲁棒性评估，但是车辆早到达时并不违反约束，所以可能可以考虑对鲁棒性的评价进行修改，改为max(0,arri-lbi)，以此求和来评估鲁棒性可能会更好。 不过这样可能会造成送货员有一定等待时间(lbi-arri)，如果将送货员的体验（等待时间短）也作为一个目标的话，那么还是尽量靠近中间值的鲁棒性评估指标可能较贴合一些。并且可能可以结合其他的算法模版(GA、DE)来观察是否能够改进算法性能。

参考文献

[1] DUAN J, HE Z, YEN G G. Robust Multiobjective Optimization for Vehicle Routing Problem With Time Windows [J]. IEEE Transactions on Cybernetics, 2021, PP(99): 1-15.