基于改进智能水滴算法的多目标供应链最优模型

方 青[1,2] 邵 媛[2]
(华中科技大学管理学院 武汉 430074)[1] (武汉科技大学管理学院 武汉 430065)[2]

摘 要 为了最大限度地降低制造型供应链的销售成本并缩短供货时间,提出了一种基于改进智能水滴算法的多目标供应链优化模型。该模型通过在选项选择期间同时考虑成本和时间来提高供应链效率,并能够将制造型供应链中的销售成本和交货时间最小化。通过使用帕累托最优准则对传统的智能水滴算法进行修改,从而得到一个帕累托集,以实现两个目标的最小化。通过3个实例对所提算法进行了测试,并采用世代距离和超区域比指标将其与蚁群优化算法进行了比较。实验结果显示,所提方法的性能更优,生成的解集更接近真实帕累托集,能够覆盖更大的解区域面积,且计算效率较高。

关键词 制造型供应链,多目标供应链,智能水滴算法,帕累托最优
中图法分类号 TP183 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2018.08.035

Optimal Model of Multi-objective Supply Chain Based on Improved IWD Algorithm

FANG Qing[1,2] SHAO Yuan[2]
(School of Management, Huazhong University of Science and Technology, Wuhan 430074, China)[1]
(School of Management, Wuhan University of Science and Technology, Wuhan 430065, China)[2]

Abstract In order to minimize the selling cost and delivery time of manufacturing supply chain, a multi-objective supply chain optimization model based on improved intelligent water drop algorithm was proposed. The model improves the efficiency of the supply chain by considering both cost and time during option selection, and minimizes the sales cost and leads time in the manufacturing supply chain simultaneously. By using the Pareto optimality criterion, the traditional intelligent water drop algorithm is modified to obtain a Pareto set to minimize the two objectives. The algorithm was tested by three examples and compared with the ant colony optimization algorithm using the generation distance and hyperarea ratio index. The results show that the performance of the proposed method is more excellent and the generated set is closer to the real Pareto set to cover a larger area of solution region, with the calculation efficiency being high.

Keywords Manufacturing supply chain, Multi-objective supply chain, Intelligent water drop algorithm, Pareto optimality

1 引言

当今,全球市场日益激烈的竞争迫使企业对其供应链(Supply Chain,SC)进行配置和评估,许多物流供应商已经认识到,最佳供应链设计(Supply Chain Design,SCD)是所有业务战略的重要组成部分。设计SC时,最重要的目标之一是以尽可能低的成本在适当的时间向客户交付产品[1-3]。这一设计目标很重要,因为最佳的SCD会使成本降低10%,且使服务时间减少40%。

但是众所周知,由于一些因素,例如,市场扩张、供应商范围广泛、客户等待时间和竞争对手,最佳SCD的设计过程变得很困难。虽然这些因素很重要,但销售成本和交货时间(或市场投放时间)已被认为是最重要的优化目标[4-5]。传统上,SC被建模为网络,其中的节点分别代表供应商、制造商、仓库、零售商和客户等设施。SCD问题仅限于选择设施数量,并确定其中流通的单位数量。

在过去十年中,研究人员专注于开发和应用启发式和基于群体的算法来解决复杂供应链配置和物流的问题。例如,文献[6-7]提出了已汇总基于模糊控制理论的多目标供应链网路优化方案,但是其收敛性较弱,且得到的最优解的数量也较少。为了得到较多的可行解,文献[8]提出了一种基于蚁群优化的多目标启发式供应链优化方法,以期在供货时间和总成本两者之间得到一种优化的结果。文献[9]提出了一种基于增强蜂群算法的农产品多目标供应链优化方法,其能够获

到稿日期:2017-05-11 返修日期:2017-08-11 本文受国家自然科学基金青年资助项目(71501147),湖北省大学生创新创业训练计划项目(201510488028)资助。

方 青(1974-),男,博士,讲师,主要研究方向为物流和供应链建模与优化决策、算法,E-mail:fangqing@wust.edu.cn;邵 媛(1984-),女,博士,讲师,主要研究方向为系统工程与优化决策、算法,E-mail:68228155@qq.com(通信作者)。

得较多的帕累托最优(供货时间和总成本)。

为实现物流网络的复杂多目标优化，不同于上述方法，本文提供了一种基于智能水滴(Intelligent Water Drop，IWD)算法的有效方法，该方法能够使两个目标最小化，并且使用帕累托最优准则进行评估。本文的主要工作有两个方面：1)不同于现有基于蚁群的多目标优化方法，提出采用智能水滴算法来解决双目标 SCD 问题；2)对传统 IWD 算法进行修改，以解决双目标问题。本文通过 3 个实例对所提方法进行了测试，并采用世代距离、间距指标和超区域指标将其与蚁群优化算法进行了比较。结果显示，所提方法的性能更优，生成的解集更接近真实的帕累托集，能够得到更多的间隔解集，即能够覆盖解区域的更大面积，且计算效率较高。

2　智能水滴的原理

IWD 启发式方法是一种基于群体的新颖算法，可对水滴与河床之间的自然过程进行模拟。这种启发式方法是为了解决业务员出差问题被首次提出的，且取得了预期结果，因为它能够很快收敛到最佳解[10]。嵌入智能水滴启发式方法的基本思想是：由于重力，水滴按照理想的直线从原点流向目的地(如湖泊、大海或更大的河流)。在现实生活中这是不可能的，因为障碍会迫使水滴去寻找另外一条畅通的路径。

在 IWD 算法中：1)每个水滴都有两个属性，即速度和土壤(在其到达目的地的行程中改变)；2)环境或河流表现为一组充满土壤的路径；3)假设水滴以离散的步骤流动，问题由图 $G=\{V,E\}$ 表示，其中 V 为顶点集合，E 为边缘集合，因此水滴从顶点到顶点来回流动，直到找到目的地为止。

当水滴从一个顶点流到另一个顶点时，其速度有一定的提升，该数量与边缘上(连接两个顶点)土壤的倒数呈非线性比例关系。另外，连接两个顶点的边缘(路径)上的一定量的土壤被去除，且水滴将去除的土壤聚集起来。土壤的数量与水滴从一个顶点流到另一顶点所需时间的倒数呈非线性比例关系。时间与水滴的速度成正比，与两个顶点之间的距离成反比。

水滴根据概率决策规则选择下一个顶点。该规则规定，水滴选择顶点的概率与连接两个顶点的边缘上的土壤量成反比，因此水滴更有可能选择具有较少土壤量的边缘。

如果水滴已选择了下一个顶点，则使用更新参数来更新顶点上的土壤和水滴中的土壤，该更新参数是小于 1 的较小正数。水滴达到其终止条件后停止。

水滴用这种方式构建了返回总目标值(如成本、距离等)的顶点序列。通过使用所有水滴的目标值，找到迭代最佳解，并且通过在[0，1]之间选择的全局更新参数来更新形成解的边缘土壤。

传统的 IWD 启发式方法基本上分为两部分：1)解的构建，每个水滴访问所有顶点，其速度和土壤通过局部更新参数进行更新，然后从边缘去除一定量的土壤；2)整体最佳方案，比较所有水滴产生的解，以找到迭代最佳解。如果某个方案比最佳解“更好”，那么它将被迭代最佳解所取代。使用全局更新参数对整体最佳解中边缘上一定量的土壤进行更新。

3　智能水滴算法的改进

本文中所涉及的参数及其意义如表 1 所列。

表 1　参数说明

Table 1　Parameter description

参数名	参数意义
i	阶段
I	阶段总数
j	选项
J_i	可执行阶段 i 的选项 j 的总数
V	阶段 i 的集
E	阶段关系(i 和 i')的边缘集
$DS(\subseteq V)$	交付阶段的子集
c_{ij}，t_{ij}	选项 j 执行阶段 i 所需的成本和时间
C_i，T_i	所选选项到执行阶段 i 所需的成本和时间
y_{ij}	二进制变量(如 j 执行 i，则等于 1，反之等于 0)
ξ	利息间隔时间
μ_i	阶段 i 的每单位时间平均需求
a_v，b_v，c_v	速率更新参数
a_s，b_s，c_s	土壤更新参数
HV	启发式值
r	河流
d	水滴
R	河流总数
D	水滴总数
S_d	d 选择的选项 j 的子集，$S_d=\{j,j',\cdots\}$
s_d	用 S_d 创建的解，$s_d=\{LT$，销货成本$\}$
SS_r	通过 r 计算的解集，$SS_r=\{s_1,\cdots,s_d,\cdots\}$
p_{ij}	选择 j 来执行 i 的概率
φ_{ij}	$j\in i$ 的土壤数量
φ_d	d 的土壤数量
v_d	d 的速率
θ	小常数以避免零分割
$\Delta\varphi_{ij}$	j 中土壤的变化量
ρ_n	局部更新因数
ρ_w	全局更新因数
τ_{ij}	水滴穿过选项 j 所花费的时间
CGS	销售成本
LT	交货时间

本文提出的算法将两个目标(CGS 和 LT)最小化，并应用帕累托最优准则来确定较好的解。这些解形成解集，并称为非支配解。这种解是指：只有在其中至少一个目标恶化的情况下才能对目标进行改进。

3.1　数学表达式

为了用数学方法表示双目标优化问题，将第二个目标函数添加到 Graves 和 Willems[11] 提出的单目标模型中。将 SC 表示为图 $G=\{V,E\}$，其中顶点集分别表示供应、制造和交付阶段(i)，因此 $V=\{1,\cdots,i,\cdots,I\}$，$I$ 表示阶段总数。边缘集表示两个阶段之间的关系，这些关系可能存在于供应阶段和制造阶段之间、两个制造阶段之间或制造和交付阶段之间，因此 $E=\{(1,2),(1,i),\cdots,(i,i')\}$。交付阶段的子集被定义为 $DS(DS\subseteq V)$。

每个阶段 i 有不同的选项 $j=\{1,\cdots,J_i\}$，各选项可执行

各个阶段。各选项的成本和时间分别为 c_{ij} 和 t_{ij}。使用二进制变量选择执行阶段的选项。如果选项 j 执行阶段 i，则定义该变量为 $y_{ij}=1$；否则 $y_{ij}=0$。

利用式(1)对 CGS 进行建模。文献[11]将其定义为在公司利息间隔 ξ 期间出售的商品价值。

$$CGS=\xi\sum_{i=1}^{I}\mu_i C_i \tag{1}$$

$$\mu_i=\xi\sum_{i';(i,i')\in E}\mu_{i'} \tag{2}$$

其中，μ_i 表示阶段 i 的需求，C_i 表示阶段 i 所选选项的成本。

阶段 i 的前置时间(LT_i)被定义为所选选项的阶段执行时间(T_i)加上其先前阶段 i' 的最大前置时间。形式上，前置时间被定义为 $LT_i=T_i+\max\{LT_{i'}\}$。交付阶段的前置时间被称为市场投放时间，如式(3)所示，将其添加到单目标模型中。

$$LT=\max_{i\in DS}\{LT_i\} \tag{3}$$

按照式(4)－式(8)的形式将式(1)和式(3)最简化。通过求解式(4)、式(5)和式(7)，来对 SC 进行设计。例如，当所有二进制变量(y_{ij})的值已知且阶段的时间(T_i)和成本(C_i)设定时，式(6)可计算出所有阶段的前置时间，式(8)可保证 y_{ij} 只能取 0 或 1。

$$\sum_{j=1}^{J_i}c_{ij}y_{ij}-C_i=0,i=1,\cdots,I \tag{4}$$

$$\sum_{j=1}^{J_i}t_{ij}y_{ij}-T_i=0,i=1,\cdots,I \tag{5}$$

$$T_i+\max_{i';(i',i)\in E}\{LT_{i'}\}-LT_i=0,i=1,\cdots,I \tag{6}$$

$$\sum_{j=1}^{J_i}y_{ij\,i}=1,i=1,\cdots,I \tag{7}$$

$$y_{ij}=\{0,1\},i=1,\cdots,I;j=1,\cdots,J_i \tag{8}$$

3.2 基于 IWD 的算法

为了解决 SCD 问题，IWD 算法创建了 R 河流(表示迭代次数)，$r=\{1,\cdots,R\}$，每次迭代都会有 D 个水滴 d，$d=\{1,\cdots,D\}$。水滴解表示执行阶段的选项的子集 S_d。$s_d=(LT,CGS)$表示式(1)和式(3)的值，由水滴 d 所选选项的子集产生。每次迭代(每条河一次)时，每条河 r 创建一个解集 $SS_r=\{s_1,\cdots,s_d,\cdots\}$，其中包含所有非支配解。为了将 s_d 添加到 SS_r，必须对河流 r 产生的各个 s_d 的最后一个条件(称为帕累托最优准则)进行验证。算法最终输出最后一个解集 $SS=SS_R$。

在所提算法的第一部分中，每个水滴都创建一个 s_d。为此，将 d 置于阶段 i，利用式(9)对 d 选择 j 的概率($j\in i$)进行计算，其中 θ 为非常小的常数，可避免零分割。p_{ij} 的值取决于 $j(\varphi_{ij})$的土壤量，因此 φ_{ij} 的值越大，选择 j 的机会就越小。

$$p_{ij}=\frac{\frac{1}{\theta+g_{ij}}}{\sum_{j';j'\in i}\frac{1}{\theta+g_{ij'}}} \tag{9}$$

其中：

$$g_{ij}=\begin{cases}\varphi_{ij}, & \text{if } \min_{j\in i}\{\varphi_{ij}\}\geqslant 0\\ \varphi_{ij}-\min_{j\in i}\{\varphi_{ij}\}, & \text{其他}\end{cases} \tag{10}$$

φ_{ij} 的值可能等于或大于零。另一方面，当 $\varphi_{ij}<0$ 时，对 $\varphi_{ij}-\min_{j\in i}\{\varphi_{ij}\}$ 进行计算以获取 $\varphi_{ij}\geqslant 0$ 的所有值。计算出各个 p_{ij} 之后，根据概率决策法则选择执行阶段 i 的选项 j。使用该法则使水滴寻找新的路径(或尝试新的路径)，从而避免停滞，搜寻更多选项。因此，不经常选择具有最高概率 p_{ij} 的选项 j。一般来说，p_{ij} 的值越大，选择 j 来执行 i 的机会就越大，而具有较低 p_{ij} 的选项仍有可能被选择并探索新的解。

将所选选项 j 的二进制变量 y_{ij} 的值设为 1，并将其存储在所选选项列表 S_d 中。之后，使用式(11)更新水滴速度；使用式(12)更新选项土壤的增量；使用式(13)更新选项 $j(\varphi_{ij})$的土壤以及水滴土壤 φ_d，其中 ρ_n 为局部更新参数。

$$v_d=v_d+\frac{a_v}{b_v+c_v(\varphi_{ij})^2} \tag{11}$$

$$\Delta\varphi_{ij}=\frac{a_s}{b_s+c_s(\tau_{ij})^2},\tau_{ij}=\frac{e^{1/t_{ij}}+e^{1/c_{ij}}}{v_d} \tag{12}$$

$$\varphi_{ij}=(1-\rho_n)\varphi_{ij}-\rho_n\Delta\varphi_{ij},\varphi_d=\varphi_d+\Delta\varphi_{ij} \tag{13}$$

注意：水滴在选项 j 上花费的时间为 $\tau_{ij}=HV/v_d$(式(12))，用 $HV=e^{1/t_{ij}}+e^{1/c_{ij}}$ 来估计选项的时间和成本。

当水滴 d 流经所有阶段 i 时，根据所选选项 S_d 对式(1)和式(3)进行计算。在算法的第二部分中，河流 r 创建一个将帕累托最优准则应用于所有 $s_d=(LT,CGS)$的解集 SS_r，从而将非支配 s_d 添加到 SS_r。最后，使用全局更新参数(ρ_w)对属于非支配 s_d 的选项 j 的土壤进行更新，如式(14)所示。

$$\varphi_{ij}=(1-\rho_n)\varphi_{ij}-\rho_w\left(\frac{1}{l-1}\right)\varphi_d,j\in s_d,s_d\in SS_r \tag{14}$$

算法 1 给出了改进智能水滴算法的步骤。在第一部分中，每个水滴在每个阶段选择一个选项，创建 S_d。为了创建一个解，使每个水滴 d 流经各个阶段来选择一个选项 j。当水滴 d 在阶段 i 上时，可通过 d 计算出各个 j 的 p_{ij}。进入下一阶段之前，应用概率决策规则选择 j 来执行 i，即将变量 y_{ij} 设为 1；计算和更新土壤和速度的增量；将所选选项 j 存储在水滴的解 S_d 中。水滴选择各个阶段的选项之后，可计算出各个阶段的成本(C_i)和时间(T_i)，也可以计算出前置时间(LT_i)。最后，可通过 $s_d=(LT,CGS)$计算出水滴的解。

计算出各个水滴的 $s_d=(LT,CGS)$之后，通过非支配 s_d 建立河流 $r(SS_r)$的解集。算法的输出为最后一条河流生成的解集。

算法 1 改进智能水滴算法

输入：V={1,…,i,…,I}；j={1,…,J_i}，∀i；E={(1,i),…,(i,i′),…}

输出：帕累托最优解集 SS

1. 初始化静态参数：D，R，ρ_n，ρ_w，a_v，b_v，c_v，a_s，b_s，c_s；
2. 初始化可变参数：φ_{ij}；
3. 设 r=1；
4. while(r≤R)
5. 设 $SS_r=\{\}$；
6. for d=1 to d=D
7. 设 $S_d=\{\}$；
   从 V 选择一个阶段 i；
8. for j=1 to j=J_i
9. 计算 p_{ij}(式(10))；
10. end for

11. 根据概率决策法则选择 j 来执行 i，即设 $y_{ij}=1$（式(7)已解出）；
12. 更新 v_d（式(11)）；
13. 计算土壤增量 $\Delta\varphi_{ij}$（式(12)）；
14. 更新 φ_{ij} 和 φ_d 的土壤（式(13)）；
15. 存储所选选项，$S_d \leftarrow j$；
16. 从 V 中删除 i；
17. for i=1 to i=I
18. 用 S_d 计算 C_i 和 T_i（式(4)和式(5)）；
19. 计算 LT_i（式(6)）；
20. end for
21. 设 $s_d=(LT,CGS)$（式(1)和式(3)）；
22. end for
23. 验证每一个 s_d 是否受其他 s_d' 的支配；
24. 如果 $\neg\exists s'_d | s_d' \leqslant s_d$，则
$SS_r \leftarrow s_d$
25. 如果 r=R，则
26. 停止 $SS=SS_R$；
27. 否则
28. 更新各个选项 $j|j\in s_d$ 且 $s_d \in SS_r$（式(14)）；
29. r=r+1；
30. end while

4　实验结果与分析

4.1　实例的选取和参数的设置

为了测试算法的性能，利用其对 3 个实例进行求解。实例 1 和实例 2 取自文献[11]，它们代表笔记本的 SC。笔记本共享电路板组件和几个组装的主组件（称为笔记本电脑组件）。实例 3 表示配备 3 种装载机 SC[8]。这 3 款产品共享由传动装置、发动机和底盘组成的主组件。当铲子和带轮的悬架连接到主组件上时，产生轮式装载机。当叶片、履带轮框架和悬架连接到主组件上时，产生履带式牵引机。履带式装载机由叶片和铲子连接到主组件上产生。这 3 个实例的具体参数如表 2 所列。

表 2　3 个实例的详细参数

Table 2　Detailed parameters of three instances

实例	需求			产品	阶段 $i(V)$	选项(j)	边缘(E)	解	ξ
	产品	顾客	μ						
1	1	1	125	1	1	26	12	3072	250
2	1	1	200	2	3	33	16	24576	250
	1	2	75						
	2	1	125						
3	1	1	20	3	9	105	37	1.28×10^{16}	250
	1	3	12						
	1	4	23						
	2	2	10						
	2	4	32						
	3	1	21						
	3	2	9						
	3	3	17						
	3	4	6						

在理论研究和类似研究文献[12]的基础上，通过不同的组合调整静态参数的值。用于解决车间调度问题的多目标方法设置$(a_v,b_v,c_v)=(a_s,b_s,c_s)=(1,0.01,1)$，$\rho_n=0$，以获得最佳解[14]。

可变参数由用户进行选择，并通过实验进行调整[11]。根据各个实例中 CGS 和 LT 的值，本文设定 $\varphi_{ij}=10000$，$\varphi_d=10000$，$v_d=4$。根据各个阶段的时间和成本值对启发式参数 HV 进行调整。根据实例的 CGS 和 LT 值对时间和成本参数的值进行调整。

本文算法使用 10 条河流（$R=10$），每条河流取 450 个水滴（$D=450$）。设置 $R=30$ 和 $D=450$ 以便将输出结果与 Moncayo-Martínez 和 Zhang 提出的蚁群优化算法[8]产生的结果进行对比，其中创建了 30 个蚁群。

本实验仿真环境为：Core i7 CPU ，4 GB 内存，2.90 GHz 主频。将本文算法与基本蚁群算法[14]、蚁群优化算法[8]进行对比实验。

4.2　与其他算法的比较

为了对算法的性能进行量化分析，采用 3 个多目标优化指标的平均值[15]进行对比分析，具体如表 3、表 4 所列。

1）世代距离（Generational Difference，GD）。这是一种用于测试真实帕累托集与解集 SS 之间的距离的方法，GD 值越小，SS 越接近真实帕累托集；2）超区域比（Hyperarea Ratio，HR），用来测量 SS 覆盖区域与真实帕累托集目标空间的比值，因此 HR 的最优值为 1。

对于 3 种不同的优化算法，计算上述指标的平均值。当 $R=10$，$D=450$ 时，运行算法 30 次，求出各个实例的解，蚁群优化算法的参数为 $P=10$，$Q=450$，结果如表 3 所列。然后设置 $R=30$，$D=450$，$Q=450$，再运行 30 次，蚁群优化算法的参数为 $P=10$，$Q=450$，结果如表 4 所列。

表 3　不同优化算法的性能比较（$R=P=10$，$D=Q=450$）

Table 3　Performance comparison of different optimization algorithms（$R=P=10$，$D=Q=450$）

实例	基本蚁群算法		蚁群优化算法		本文方法	
	GD	HR	GD	HR	GD	HR
1	78317	1.2117	61031	1.0041	11067	0.9987
2	283447	1.1216	213554	1.0042	203919	1.0051
3	401392	1.2813	315272	1.1573	281036	1.0422

表 4 不同优化算法的性能比较($R=P=30, D=Q=450$)

Table 4 Performance comparison of different optimization algorithms($R=P=30, D=Q=450$)

实例	基本蚁群算法		蚁群优化算法		本文方法	
	GD	*HR*	*GD*	*HR*	*GD*	*HR*
1	79624	1.2031	61040	1.0043	10298	1.0208
2	288314	1.2123	220943	1.0051	159347	1.0031
3	420163	1.3118	320065	1.1685	231412	1.0122

首先,使用基本蚁群算法和蚁群优化算法计算 3 个实例的真实帕累托集,再将指标的值与本文所提算法产生的值进行对比。对于应用本文算法的相同案例,当 $R=30$ 时的指标值比 $R=10$ 时的指标值更好,因为 *GD* 值更小。但是对于应用基本蚁群和文献[8]的蚁群优化算法的相同案例,当 $P=30$ 时的 *GD* 数值比当 $P=10$ 时的 *GD* 数值更高,即性能并没有提高。同样地,对于应用本文算法的相同案例,$R=30$ 时的 *HR* 数值比 $R=10$ 的 *HR* 数值更小,即性能更优。反之,对于应用基本蚁群和文献[8]的蚁群优化算法的相同案例,随着 *P* 的增加,*HR* 的结果没有降低。

从表 3 和表 4 的结果可以看出,在相同的案例和参数情况下,本文提出的智能水滴算法的 *GD* 数值更小,*HR* 数值更接近于 1,说明本文所提方法的解集覆盖区域与真实帕累托集覆盖区域的相似度最高。

4.3 计算时间成本的比较分析

当 $R=P=10, D=Q=450$ 和 $R=P=30, D=Q=450$ 时,迭代 30 次,对本文算法和其他两种算法的平均 CPU 时间进行统计,结果分别如图 1 和图 2 所示。

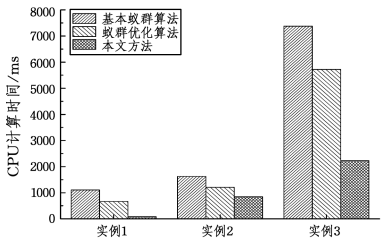

图 1 不同优化算法的 CPU 计算时间的比较($R=P=10, D=Q=450$)

Fig. 1 Comparison of CPU calculation time of different optimization algorithms($R=P=10, D=Q=450$)

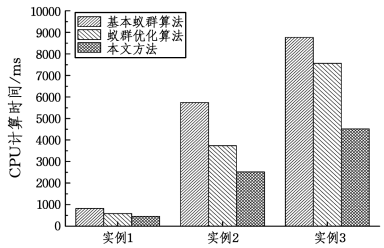

图 2 不同优化算法的 CPU 计算时间的比较($R=P=30, D=Q=450$)

Fig. 2 Comparison of CPU calculation time of different optimization algorithms($R=P=30, D=Q=450$)

如图 1 和图 2 所示,基本蚁群算法计算出帕累托集所需的 CPU 时间最长。使用蚁群优化算法解出相同实例时,$P=30$ 时的 CPU 时间比 $P=10$ 时的时间长。同样,对于本文算法,当 $R=30$ 时的 CPU 时间长于 $R=10$ 时的时间。然而,需要注意的是,使用 $R=30$ 时指标的数值比 $R=10$ 时的数值好,即 *GD* 值更小。

最后,当 $R=P=30$ 时,对利用所提智能水滴算法和蚁群优化算法求出的解的数量进行比较,结果如表 5 所列。从表 5 可以看出,在所有 3 个实例中,使用蚁群优化算法得到的 *SS* 解的数量均少于使用智能水滴算法得出的解的数量。例如,对于实例 3,两者的数量分别为 13 和 16。此外,用蚁群优化算法计算的解的 *LT* 在 9～12 天之间,而智能水滴中解的 *LT* 在 16～52 天之间,因此 *SS* 解在解区域中的分布范围更大,即智能水滴产生了一个能覆盖大部分解区域的 *SS*。

表 5 不同优化算法的解的数量比较($R=P=30, D=Q=450$)

Table 5 Comparison of number of solutions of different optimization algorithms($R=P=30, D=Q=450$)

实例	基本蚁群算法	文献[15]方法	本文方法
1	8	9	10
2	10	12	13
3	11	13	16

还应注意,对于实例 3,使用蚁群优化算法获得的 *SS* 解不会均匀地扩散到解区域中。因此,综合上述 *GD* 和 *HR* 结果,通过本文算法生成的解集更接近真实的帕累托集,且生成的 *SS* 可覆盖的面积比蚁群优化算法生成的集的解区域更大,计算效率更高。

结束语 本文提出了一种基于改进 IWD 的多目标优化模型,以同时最小化制造供应链中的 *CGS* 和 *LT*。该模型通过在执行阶段的选项选择期间同时考虑成本和时间来提高 SC 效率。对于第二个目标即最小化 *CGS*,对原来的智能水滴算法进行修改,以得到一个帕累托集,该集可胜过传统方法生成的帕累托集。将提出的多目标 IWD 算法应用到多目标供应链研究中被广泛使用的 3 个实例中,计算真实帕累托集,并将该算法的结果与基于蚁群优化的算法以及基本蚁群算法的结果进行比较。实验结果显示,本文所提算法的性能优于其他两种算法,其解集覆盖区域与真实帕累托集覆盖区域的相似度最高,更接近真实帕累托集。该算法可获得更多的间隔解集,即能够覆盖更大面积的解区域。

未来一方面计划将提出的方法与其他启发式算法进行对比,另一方面尝试用提出的智能水滴算法来解决更复杂的供应链设计问题。

参 考 文 献

[1] ALIJLA B O, WONG L P, LIM C P, et al. A modified Intelligent Water Drops algorithm and its application to optimization problems[J]. Expert Systems with Applications, 2014, 41(15): 6555-6569.

[2] TRISNA T, MARIMIN M, ARKEMAN Y, et al. Multi-objective optimization for supply chain management problem: A literature review[J]. Decision Science Letters, 2016, 5(2): 283-316.

[3] BOOYAVI Z, TEYMOURIAN E, KOMAKI G M, et al. An improved optimization method based on the intelligent water drops algorithm for the vehicle routing problem[C]// Proceedings of IEEE Conference on Computational Intelligence in Production & Logistics Systems. New York: IEEE Press, 2015: 59-66.

(下转第 212 页)

25(5):61-64. (in Chinese)
白烁,周晴. 嵌入式软件资源冲突自动检测系统设计[J]. 电子设计工程,2017,25(5):61-64.

[4] HUA S Z,DING A L,GUO D W,et al. Nash game power control algorithm for D2D communication underlaying cellular networks[J]. Application Research of Computers,2016,33(4):1187-1190. (in Chinese)
滑思忠,丁爱玲,郭达伟,等. 基于纳什均衡的 D2D 通信功率控制博弈算法[J]. 计算机应用研究,2016,33(4):1187-1190.

[5] YANG G L,WANG J,ZHU S W,et al. Multi-label Classification Based on the Relevance of K-Nearest Neighbor[J]. Science Technology and Engineering, 2016,16(34):222-226. (in Chinese)
杨国亮,王建,朱松伟,等. 基于 k-邻域相关性的多标签分类[J]. 科学技术与工程,2016,16(34):222-226.

[6] ZHANG C G,SONG J Z,JIANG J Q,et al. Imbalanced data classification algorithm of improved de-noising auto-encoder neural network[J]. Application Research of Computers,2017,34(5):1329-1332. (in Chinese)
张成刚,宋佳智,姜静清,等. 一种改进的降噪自编码神经网络不平衡数据分类算法[J]. 计算机应用研究,2017,34(5):1329-1332.

[7] ZHANG C,GUO M L. Research and realization of improved native Bayes classification algorithm under big data environment [J]. Journal of Beijing Jiaotong University,2015,39(2):35-41. (in Chinese)
张春,郭明亮. 大数据环境下朴素贝叶斯分类算法的改进与实现[J]. 北京交通大学学报,2015,39(2):35-41.

[8] DU H L,ZHANG Y. A classification algorithm based on mixed sampling for imbalanced dataset[J]. Journal of Yanshan University,2015,39(2):158-164. (in Chinese)
杜红乐,张燕. 不均衡数据混合取样分类算法[J]. 燕山大学学报,2015,39(2):158-164.

[9] LI L,QIU F. A Classification. Optimization Scheduling Method of Massive Data under Cloud Environment[J]. Computer Simulation,2016,33(5):315-317. (in Chinese)
李玲,邱芬. 云环境下海量数据的分类优化调度方法研究[J]. 计算机仿真,2016,33(5):315-317.

[10] LI Z H. Classification Query of Huge Amounts of Data in Cloud Computing Environment Based on Genetic Optimization[J]. Bulletin of Science and Technology,2015,31(6):34-36. (in Chinese)
李志虹. 基于遗传迭代优化的云计算下海量数据分类查询[J]. 科技通报,2015,31(6):34-36.

[11] TAO X M,HAO S Y,ZHANG D X,et al. Overview of classification algorithms for unbalanced data[J]. Journal of Chongqing University of Posts and Telecommunications (Natural Science Edition),2013,25(1):101-110. (in Chinese)
陶新民,郝思媛,张冬雪,等. 不均衡数据分类算法的综述[J]. 重庆邮电大学学报(自然科学版),2013,25(1):101-110.

(上接第 202 页)

[4] YOU F,GROSSMANN I E. Balancing responsiveness and economics in process supply chain design with multi-echelon stochastic inventory[J]. Aiche Journal,2010,57(1):178-192.

[5] SHEN Z J M. Integrated supply chain design models:a survey and future research directions [J]. Journal of Industrial & Management Optimization,2007,3(1):1-27.

[6] KRISTIANTO Y,GUNASEKARAN A,HELO P,et al. A model of resilient supply chain network design:A two-stage programming with fuzzy shortest path[J]. Expert Systems with Applications,2014,41(1):39-49.

[7] BAI X,LIU Y. Robust optimization of supply chain network design in fuzzy decision system [J]. Journal of Intelligent Manufacturing,2016,27(6):1131-1149.

[8] MONCAYO-MARTÍNEZ L A,ZHANG D Z. Multi-objective ant colony optimisation:a meta-heuristic approach to supply chain design [J]. International Journal of Production Economics,2011,131(1):407- 420.

[9] MASTROCINQUE E,YUCE B,LAMBIASE A,et al. A multi-objective optimization for supply chain network using the Bees Algorithm [J]. International Journal of Engineering Business Management,2013,38(1):1-11.

[10] MOKHTARI H. A nature inspired intelligent water drops evolutionary algorithm for parallel processor scheduling with rejection [J]. Applied Soft Computing Journal,2015,26(26):166-179.

[11] GRAVES S C,WILLEMS S P. Optimizing the Supply Chain Configuration for New Products[J]. Management Science,2005,51(8):1165-1180.

[12] SHAH-HOSSEINI H. The intelligent water drops algorithm:a nature-inspired swarm-based optimization algorithm [J]. International Journal of Bio-Inspired Computation,2009,1(2):71-79.

[13] NIU S H,ONG S K,NEE A Y C. An improved Intelligent Water Drops algorithm for achieving optimal job-shop scheduling solutions[J]. International Journal of Production Research,2012,50(15):4192-4205.

[14] PENG J. Optimization of multi-objective supply chain of agricultural products based on enhanced bee colony algorithm[J]. Control Engineering,2016,23(7):1123-1128. (in Chinese)
彭剑. 基于增强蜂群算法的农产品多目标供应链优化[J]. 控制工程,2016,23(7):1123-1128.

[15] COELLO C A C,LAMONT G B,VELDHUIZEN D A V. Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation) [M]. New Jersey:Springer-Verlag New York,Inc. 2006.

[16] HE K L,LI W,CHENG C Y. Application of SOM Neural Network in Performance Evaluation of Green Supply Chain for Pig Industry[J]. Journal of Chongqing University of Technology (Natural Science),2014,28(9):92-97. (in Chinese)
何开伦,李伟,程创业. SOM 神经网络在生猪绿色供应链绩效评价中的应用[J]. 重庆理工大学学报(自然科学),2014,28(9):92-97.