基于积极集策略的最小闭包球问题算法研究

丛伟杰[1] 刘红卫[2]
(西安邮电大学理学院 西安 710121)[1] (西安电子科技大学理学院 西安 710071)[2]

摘 要 首先,基于每次迭代计算距离当前球心最远的两个点,提出一种求解 n 维空间中 m 个点的最小闭包球问题的$(1+\varepsilon)$-近似算法。对于$\varepsilon\in(0,1)$,建立了该算法的核心集大小和计算复杂度,分别为$O(1/\varepsilon)$和$O(mn/\varepsilon)$。然后,给出一种积极集策略,每次迭代计算距离当前球心最远的 N 个点。将该策略结合到提出的算法中,得到一个基于积极集策略的算法。最后,实验结果表明基于积极集策略的算法能够快速、有效地求解 $m\gg n$ 的大规模数据集的近似最小闭包球。

关键词 最小闭包球,核心集,积极集策略,大规模数据集
中图法分类号 TP301.6 **文献标识码** A

Study on Algorithm of Minimum Enclosing Ball Problem Based on Active Set Strategy

CONG Wei-jie[1] LIU Hong-wei[2]
(School of Science, Xi'an University of Posts and Telecommunications, Xi'an 710121, China)[1]
(School of Science, Xidian University, Xi'an 710071, China)[2]

Abstract Firstly, based on computing two furthest points from the current center at each iteration, a $(1+\varepsilon)$- approximation algorithm was proposed for solving the minimum enclosing ball problem of m points in n dimensions. The algorithm returns a core set of size $O(1/\varepsilon)$ and achieves an $O(mn/\varepsilon)$ time complexity for a given $\varepsilon\in(0,1)$. Secondly, an active set strategy was presented, which computes N furthest points from the current center at each iteration. By incorporating this strategy into the proposed algorithm, an algorithm based on active set strategy was obtained. Finally, the experiment results show that the algorithm based on active set strategy can quickly and effectively solve approximate minimum enclosing ball of the large-scale date sets with $m\geqslant n$.

Keywords Minimum enclosing ball, Core set, Active set strategy, Large-scale date sets

1 引言

给定点集 $S=\{p^1,p^2,\cdots,p^m\}\subset R^n$,最小闭包球(minimum enclosing ball,简记为 MEB)问题就是寻找一个半径最小、包含 S 中所有的点的球。点集 S 的最小闭包球可以定义为

$$\mathrm{MEB}(S):=B_{c^*,r^*}=\{x\in R^n:\|x-c^*\|\leqslant r^*\}$$

式中,c^* 为最优球心,r^* 为最优半径,$\|\cdot\|$ 表示 Euclidean 范数。

MEB 问题在机器学习[1]、模式识别[2]和支持向量机[3,4]等领域有着广泛的应用,并且这个问题在计算几何[5-13]中也是一个重要的问题。因此,设计有效求解 MEB 问题的算法是一项有意义的工作。

给定 $\varepsilon>0$,如果 $S\subseteq B_{c,(1+\varepsilon)r}$ 并且 $r\leqslant r^*$,则称球 $B_{c,(1+\varepsilon)r}$ 为 MEB(S)的一个$(1+\varepsilon)$-近似。此外,如果点集 S 的子集 X 满足 $\mathrm{MEB}(X):=B_{c,r}$ 并且 $S\subseteq B_{c,(1+\varepsilon)r}$,则称 X 为 S 的一个核心集。

文献[5]构造了一个大小为 $O(1/\varepsilon^2)$ 的核心集,并在此基础上建立了计算 MEB(S)的一个$(1+\varepsilon)$-近似算法,计算复杂度为 $O(mn/\varepsilon^2+(1/\varepsilon^{10})\log(1/\varepsilon))$。文献[6]将核心集大小改进为 $O(1/\varepsilon)$,并依此改进文献[5]中算法的计算复杂度为 $O(mn/\varepsilon+1/\varepsilon^5)$。文献[7]中在给定 $n\geqslant\lfloor 1/\varepsilon\rfloor$ 的假设条件下,建立了核心集大小的一个紧上界$\lceil 1/\varepsilon\rceil$。与文献[6]同样,文献[8]也获得了大小为 $O(1/\varepsilon)$ 的核心集,通过使用二阶锥规划结合割平面法得到改进的算法,计算复杂度为 $O(mn/\varepsilon+(1/\varepsilon^{4.5})\log(1/\varepsilon))$。此外,文献[9-11]分别使用组合结构及光滑近似技术处理高维数据集的 MEB 问题,由于未结合核心集思想来处理高维大规模数据,因此效率不是很高。最近,文献[12]根据 Frank-Wolfe 算法[14]并结合"远离步"策略[15]给出了求解 MEB(S)的$(1+\varepsilon)$-近似的两个算法。对于 $\varepsilon\in(0,1)$,两算法均获得大小为 $O(1/\varepsilon)$ 的核心集,算法的计算复杂度为 $O(mn/\varepsilon)$。文献[13]基于支持向量机中序列最小最优化(sequential minimal optimization-SMO)算法[16]的思想给出了求解 MEB 问题的 SMO-型近似算法,并获得大小为 $O(1/\varepsilon)$

到稿日期:2012-11-06 返修日期:2013-03-23 本文受国家自然科学基金项目(61072144,61075117),陕西省教育厅专项科研基金项目(12JK0735),西安邮电大学博士科研启动基金项目(1051203)资助。
丛伟杰(1981-),男,博士,讲师,主要研究方向为计算几何、机器学习中的最优化理论与算法,E-mail:wjcong@xupt.edu.cn;刘红卫(1967-),男,教授,博士生导师,主要研究方向为最优化理论与算法。

的核心集，算法的计算复杂度为 $O(mn/\varepsilon)$。这也是目前近似计算 MEB 问题最好的核心集和计算复杂度结果。

本文首先在文献[12]中算法初始化的基础上，通过每次计算距离当前球心距离最远的两个点来构造 S 的一个核心集，进而提出计算 MEB(S)的一个$(1+\varepsilon)$-近似算法。该算法获得近似计算 MEB 问题目前最好核心集大小 $O(1/\varepsilon)$和计算复杂度 $O(mn/\varepsilon)$的理论结果。然后，基于文献[8]中割平面法的思想，给出一个积极集策略，并将此策略结合到提出的算法中得到一个基于积极集策略的算法。实际上，该算法是割平面法的一个推广，割平面法每次迭代只计算一个距离当前球心最远的点加入到积极集中，而推广的算法将每次迭代计算的 $N(N\geqslant 2)$个距离当前球心最远的点加入到积极集中来提高计算效率。最后，数值试验结果表明给出的积极集策略的确能够大大降低计算量，提高算法的运行速度。

2 算法及分析

根据文献[12]，MEB(S)问题可以转化为如下二次规划问题：

$$\begin{aligned}&\min_{c,r} r^2\\ &\text{s.t. } (p^i)^{\mathrm{T}}p^i-2(p^i)^{\mathrm{T}}c+c^{\mathrm{T}}c\leqslant r^2,i=1,2,\cdots,m\end{aligned}\tag{1}$$

其中，$c=(c_1,c_2\cdots,c_n)^{\mathrm{T}}\in R^n$ 和 $r\in R$ 为原始变量。问题(1)的 Lagrangian 对偶问题[12]为：

$$\begin{aligned}&\max_u \sum_{i=1}^m u_i(p^i)^{\mathrm{T}}p^i-(\sum_{i=1}^m u_ip^i)^{\mathrm{T}}(\sum_{i=1}^m u_ip^i)\\ &\text{s.t. } \sum_{i=1}^m u_i=1,u_i\geqslant 0,i=1,2,\cdots,m\end{aligned}\tag{2}$$

其中，$u=(u_1,u_2\cdots,u_m)^T\in R^m$ 为对偶变量。

将对偶问题(2)的目标函数记为 $\Phi(u)$，由文献[12]的引理 1 可得：

$$c^*=\sum_{i=1}^m u_i^*p^i,r^*=\sqrt{\Phi(u^*)}$$

其中，$(c^*,r^*)\in R^n\times R$ 和 $u^*\in R^m$ 分别为问题(1)和(2)的最优解。因此，MEB(S)问题能转化为求解对偶问题(2)，这也是本文算法设计的基础。

2.1 基于计算两个最远点的算法

算法 1 给定点集 $S=\{p^1,p^2,\cdots,p^m\}\subset R^n$ 和精度参数 $\varepsilon\in(0,1)$。具体算法步骤如下：

1. 选取两个初始点指标：$\alpha=\arg\max_{i=1,2,\cdots,m}\|p^i-p^1\|^2$，$\beta=\arg\max_{i=1,2,\cdots,m}\|p^i-p^\alpha\|^2$；令初始核心集 $X_0=\{p^\alpha,p^\beta\}$；定义初始可行解：$u^0\in R^m$，其中 $u_\alpha^0=u_\beta=1/2$，其余分量取 0；定义初始球心和半径分别为：$c^0=\sum_{i=1}^m u_i^0p^i$，$r_0=\sqrt{\Phi(u^0)}$。置 k=0。
2. 计算两个最远点指标：$i_+=\arg\max_{i=1,2,\cdots,m}\|p^i-c^k\|^2$，$j_+=\arg\max_{i,i\neq i_+}\|p^i-c^k\|^2$；令 $I_+=\|p^{i_+}-c^k\|^2$，$J_+=\|p^{j_+}-c^k\|^2$，$\delta_k=\frac{I_+}{r_k^2}-1$；如果 $\delta_k\leqslant(1+\varepsilon)^2-1$，则算法终止，并输出 X_k，u^k，c^k，r_k。否则，转第 3 步。
3. 更新核心集：$X_{k+1}=X_k\cup\{p^{i_+},p^{j_+}\}$；令 $K=\|p^{i_+}-p^{j_+}\|^2$，$\lambda_k=\frac{I_++J_+-2r_k^2}{4(I_++J_+)-2K}$；更新可行解：$u^{k+1}=(1-2\lambda_k)u^k+\lambda_k(e^{i_+}+e^{j_+})$；更新球心和半径：$c^{k+1}=(1-2\lambda_k)c^k+\lambda_k(p^{i_+}+p^{j_+})$，$r_{k+1}=\sqrt{\Phi(u^{k+1})}$。置 k=k+1，转第 2 步。

算法 1 的第 1 步是初始化过程，与文献[12]中算法 3.1 的初始化过程相同。两个算法主要的区别在于：每次迭代算法 3.1[12]仅计算距离当前球心最远的一个点，将其加入到核心集中，然后使当前球心向该点移动；而算法 1 计算距离当前球心最远的两个点，将它们加入到核心集中，然后使当前球心同时向这两个点靠近。当算法 1 终止时，由第 2 步可得 $I_+=\|p^{i_+}-c^k\|^2=(1+\delta_k)r_k^2\leqslant(1+\varepsilon)^2r_k^2$，则有

$$\|p^{i_+}-c^k\|=\sqrt{1+\delta_k}r_k\leqslant(1+\varepsilon)r_k\tag{3}$$

这表明当算法 1 终止时能得到 MEB(S)的一个$(1+\varepsilon)$-近似 $B_{c^k,(1+\varepsilon)r_k}$。

下面描述在每次迭代中如何更新可行解 u^{k+1}，即如何得到算法 1 第 3 步中的 λ_k。由对偶问题(2)可得 $\Phi(u^k)=\sum_{i=1}^m u_i(p^i)^{\mathrm{T}}p^i-\|c^k\|^2$。令

$$\begin{aligned}\Phi_k(\lambda):&=\Phi((1-2\lambda)u^k+\lambda(e^{i_+}+e^{j_+}))\\ &=(1-2\lambda)\sum_{i=1}^m u_i(p^i)^{\mathrm{T}}p^i+\lambda(\|p^{i_+}\|^2+\|p^{j_+}\|^2)\\ &\quad-\|(1-2\lambda)c^k+\lambda(p^{i_+}+p^{j_+})\|^2\\ &=(1-2\lambda)\Phi(u^k)+2\lambda(1-2\lambda)\|c^k\|^2-2\lambda(1-2\lambda)\\ &\quad(c^k)^{\mathrm{T}}(p^{i_+}+p^{j_+})+\lambda(1-\lambda)(\|p^{i_+}\|^2+\|p^{j_+}\|^2)-2\lambda^2(p^{i_+})^{\mathrm{T}}p^{i_+}\\ &=(1-2\lambda)r_k^2+\lambda(1-2\lambda)(\|p^{i_+}-c^k\|^2+\|p^{j_+}-\\ &\quad c^k\|^2)+\lambda^2\|p^{i_+}-p^{j_+}\|^2\\ &=(K-2(I_++J_+))\lambda^2+(I_++J_+-2r_k^2)\lambda+r_k^2\end{aligned}$$

其中，I_+，J_+，K 和 r_k 分别定义在算法 1 的第 2、3 步中。因此，算法 1 中的 λ_k 由下式给出：

$$\lambda_k=\arg\max_{\lambda\in[0,1/2]}\Phi_k(\lambda)=\frac{I_++J_+-2r_k^2}{4(I_++J_+)-2K}$$

于是，可得 $\Phi_k(\lambda_k)=\Phi(u^{k+1})$。此外，容易得到

$$\begin{aligned}c^{k+1}&=\sum_{i=1}^m u_i^{k+1}p^i=(1-2\lambda_k)c^k+\lambda_k(p^{i_+}+p^{j_+})\\ r_{k+1}&=\sqrt{\Phi(u^{k+1})}=\sqrt{r_k^2+\frac{(I_++J_+-2r_k^2)^2}{8(I_++J_+)-4K}}\end{aligned}\tag{4}$$

2.2 算法 1 的复杂度分析

在每次迭代中，由第 2、3 步可得

$I_+=(1+\delta_k)r_k^2\geqslant J_+\geqslant r_k^2=\Phi(u^k)$

结合式(4)得到

$$\begin{aligned}\Phi(u^{k+1})&=\Phi(u^k)+\frac{(I_++J_+-2r_k^2)^2}{8(I_++J_+)-4K}\\ &\geqslant\Phi(u^k)+\frac{(I_++r_k^2-2r_k^2)^2}{8(I_++I_+)}\\ &=\Phi(u^k)(1+\frac{\delta_k^2}{16(1+\delta_k)})\end{aligned}$$

注意到这一结果与引理 3[12]的结果在形式上相同。因此，结合文献[12]中引理 3.3、3.4 和定理 3.1 的结论，得到算法 1 的的迭代复杂度结果如下。

定理 1 给定点集 S 和参数 $\varepsilon\in(0,1)$，算法 1 计算 MEB(S)的一个$(1+\varepsilon)$-近似需要 $O(1/\varepsilon)$次迭代。

下面建立算法 1 的核心集和计算复杂度结果。

定理 2 给定点集 $S=\{p^1,p^2,\cdots,p^m\}\subset R^n$ 和参数 $\varepsilon\in(0,1)$，算法 1 终止时构造了一个核心集 $X_k\subseteq S$，其满足 $|X_k|=O(1/\varepsilon)$，并且算法计算 MEB(S)的一个$(1+\varepsilon)$-近似的时间复杂度为 $O(mn/\varepsilon)$。

证明：当算法 1 终止时，由式(3)可得

$$r_k\leqslant r_{MEB(X_k)}\leqslant r^*\leqslant(1+\varepsilon)r_k\leqslant(1+\varepsilon)r_{MEB(X_k)}$$

其中，$r_{MEB(X_k)}:=\sqrt{1+\delta_k}r_k$。因此，$X_k\subseteq S$ 为 S 的一个核心集。由于每次迭代只添加两个点到当前核心集中并且考虑到

初始核心集中也只有两个点，因此结合定理 1 的结论可得最终构造的核心集大小为 $|X_k|=O(1/\varepsilon)$。

时间复杂度方面，初始化过程需要 $O(mn)$ 次基本运算。此外，每次迭代中最主要的时间花费也是最远点的计算，需要 $O(mn)$ 次基本运算。因此，结合定理 1 的结论即可完成定理 2 的证明。

2.3 基于积极集策略的算法

算法 2 给定点集 $S=\{p^1,p^2,\cdots,p^m\}\subset R^n$ 和精度参数 $\varepsilon\in(0,1)$。具体算法步骤如下：

1. 初始化过程与算法 1 的第 1 步相同，并定义初始积极集为 $Y_0=\{p^\alpha,p^\beta\}$。
2. 计算距离当前球心距离最远的 N 个点的集合，记为 $Y_{tem}=\{p^{i_1},p^{i_2},\cdots,p^{i_N}\}$；更新积极集：$Y_{k+1}=Y_k\cup Y_{tem}$。置 k=k+1。
3. 调用算法 1 计算 $MEB(Y_k)$ 的一个 $(1+\varepsilon)$-近似，记为 $B_{c^k,(1+\varepsilon)r_k}$。如果 $S\subseteq B_{c^k,(1+\varepsilon)r_k}$，则算法 2 终止，并输出 X_k,u^k,c^k,r_k。否则，转第 2 步。

算法 2 中给出积极集策略的动机是：在计算点集 S 的近似最小闭包球问题中并不是 S 中每个点都起到积极作用，实际上只要包围住 S 边界上的一些点，内部点自然就包住了。尤其，对于 $m\gg n$ 的大规模数据集，很多内部点并没有起到积极作用，反而增加了大量计算量。例如，文献[12]中算法 3.1 和本文算法 1 每次迭代都要计算 S 中每个点(包括很多 MEB(S)的内部点)到当前球心的距离来计算最远点。

因此，在算法 2 中选取一些起积极作用(尽量远离当前球心)的点组成积极集。具体地，每次迭代计算距离当前球心最远的 $N(N\geqslant 2)$ 个点并将其加入积极集。然后，通过算法 1 计算当前积极集的近似最小闭包球，最终得到点集 S 的近似最小闭包球。在数值试验中，基于一些简单的测试我们选取 $N=n$。类似的策略在文献[8]的割平面法中也得到体现。不同之处是该方法每次迭代只计算一个距离当前球心最远的点，将其加入到这里所说的积极集中，然后通过使用二阶锥规划来计算该积极集的近似最小闭包球。实际上，算法 2 可以看作该方法的一个推广或改进，显然将每次迭代计算所得的最远的 $N(N\geqslant 2)$ 个点加入到积极集中比只加入一个最远点能够减少算法的迭代次数。理论上，给出的积极集策略能很大程度地减少计算量，从而提高算法的运行速度，尤其是对于 $m\gg n$ 的大规模数据集。

3 实验结果及分析

为验证提出算法的有效性，在精度 $\varepsilon=10^{-3}$ 下对于相同的数据集，在 Matlab 中同时执行了文献[12]中的两个算法和本文的两个算法。计算机配置为 Pentium IV 2.3GHz 处理器和 2G 内存。用到的数据集是由函数 randn(n,m)随机产生的正态分布数据，且对于每对固定的(n,m)，随机产生 10 组不同的数据来执行算法，得到的结果以其算术平均值的形式报告。

表 1 给出本文算法 1 和文献[12]中算法 3.1 的实际核心集大小和 CPU 时间的比较。由表 1 结果可见，核心集方面，两个算法得到的实际核心集大小几乎相等，且都比其理论结果 $O(1/\varepsilon)$ 小得多。CPU 时间方面，本文算法 1 比文献[12]算法 3 整体上减少了近一半，即速度提高了近一倍。数值实验结果与算法 1 每次迭代加入两个点到核心集中而文献[12]算法 3.1 只加入一个点到核心集中导致算法 1 理论上要快一倍的理论结果相一致。

表 1 本文算法 1 和文献[12]中算法 3 的比较

数据集大小		实际核心集大小		CPU 时间(秒)	
n	m	算法 3.1[12]	算法 1	算法 3.1[12]	算法 1
10	10000	10.9	11.2	0.49	0.29
20	20000	17.2	17.4	4.32	2.45
30	30000	20.9	20.8	11.83	6.04
50	50000	30.2	30.2	44.56	23.06
100	100000	39.4	39.6	154.24	79.98

表 2 给出本文算法 2 和文献[12]中算法 4.1 的比较，其中算法 2 将“积极集”策略结合到算法 1，而算法 4.1[12]将“远离步”策略[15]结合到算法 3.1[12]。由于“远离步”策略允许从核心集中删除点，因此正如表 2 所列，算法 4.1[12]得到的实际核心集大小要比算法 2 略少。CPU 时间方面，表 2 中的两个算法的运行速度都比在表 1 中各自对应的原始算法有很大的提高，这也表明各自算法所用策略的有效性。另一方面，尽管算法 4.1[12]在实际的核心集大小上比算法 2 要小，但算法 2 所需的时间总是比算法 4.1[12]要少，尤其是对于大规模的数据集，算法 2 的运行速度要比算法 4.1[12]快 10 倍以上。这也充分表明对于 $m\gg n$ 的大规模数据集，积极集策略能够更有效地提高运行速度。

表 2 本文算法 2 和文献[12]中算法 4.1 的比较

数据集大小		实际核心集大小		CPU 时间(秒)	
n	m	算法 4.1[12]	算法 2	算法 4.1[12]	算法 2
10	10000	7.9	11.2	0.08	0.04
20	20000	13.2	16.8	0.58	0.12
30	30000	18.4	21.4	1.82	0.21
50	50000	25.1	28.2	5.15	0.49
100	100000	38.4	42.6	27.69	2.31

结束语 本文首先提出一种基于计算两个最远点来构造核心集的近似最小闭包球问题的算法，建立其核心集大小 $O(1/\varepsilon)$ 和计算复杂度 $O(mn/\varepsilon)$，它们是当前计算该问题最好的理论结果。然后，给出一个简单的积极集策略，并得到一个基于该策略的算法。实验结果表明，对于 $m\gg n$ 的大规模数据集，该积极集策略能够很大程度提高算法的运行速度。

此外，针对实际问题中的具体数据集，积极集策略中的 N 究竟取多大合适(取得太小增加主循环的迭代次数，取得太大迭代次数虽减少但求解子问题又要花费过多代价)，以及是否有更好的选点标准构造合适的积极集？这些都是需要进一步研究的问题。

参 考 文 献

[1] Ben-Hur A, Horn D, Siegelmann H T, et al. Support vector clustering [J]. Journal of Machine Learning Research, 2001, 2(12): 125-137

[2] 来疆亮，王守觉. 最小球覆盖几何算法及其在模式识别中的应用[J]. 模式识别与人工智能，2006，19(2)：271-276

[3] Tsang I W, Kwok J T, Cheung P-M. Core Vector machines: Fast SVM training on very large date sets [J]. Journal of Machine Learning Research, 2005, 6(4): 363-392

[4] 艾青，赵骥，秦玉平. 基于最大间隔最小体积超球支持向量机的多主题分类算法[J]. 计算机科学，2012，39(8)：237-238，267

(下转第 253 页)

算法在22代左右就可以达到最优解。结果表明，虽然都是独立运行10次，但IWD算法效率明显要高于基本PSO算法。

采用两种算法对每一个算例均独立运行10次进行测试，并对结果进行比较。

算法的参数设置如下：表2为标准粒子群算法(PSO)参数；表3为智能水滴算法(IWD)参数。

表2 PSO算法参数表

参数名	ω	c1	c2	Num	MaxGen
参数值	1.0	2.0	2.0	30	100

表3 IWD算法参数表

参数名	ρ	a_v	b_v	c_v	a_s	b_s	c_s	N	MaxGen
参数值	0.9	0.1	1	1	0.1	1	1	30	100

表4中列出了两种算法对算例独立运行10次的仿真统计结果，其中，C^*表示算例问题的最优值，RE表示应用算法计算的结果相对于C^*的误差百分比$(C-C^*)/C^* \times 100\%$，BRE表示应用算法计算的最优结果对于C^*的误差百分比，ARE表示应用算法计算的平均结果对于C^*的误差百分比，WRE表示应用算法计算的最差结果对于C^*的误差百分比。

表4 两种算法测试结果比较

问题	C^*	PSO			IWD		
		BRE	ARE	WRE	BRE	ARE	WRE
Car1	7038	0	1.98	9.19	0	0	0
Car2	7166	0	4.97	10.76	0	0	0
Car3	7312	0	4.38	7.76	0	1.32	2.30
Car4	8003	0	3.68	7.41	0	0	0
Car5	7720	0	2.91	6.5	0.16	0.71	1.32
Car6	8505	0	2.13	6.34	0.76	0.93	2.47
Rec01	1247	2.83	8.25	12.01	1.44	3.90	8.42
Rec13	1930	4.30	9.63	12.07	4.04	5.70	7.05

通过表4，可以得到8个不同的实例，其中有6个(Car1—Car6)能取到已知的最优解，2个实例(Rec01、Rec13)不能取到已知最优解，只能得到局部最优。通过与标准PSO算法对比可以看出，智能水滴算法到达最优解的迭代次数远远小于标准PSO算法，即使对于有些算例不能获得已知最优解，但误差还是比标准PSO算法小。由此，智能水滴算法可以很好地求解置换流水线调度问题，通过与标准PSO算法求解结果的对比，验证了智能水滴算法的有效性。

结束语 研究智能算法一直是生产调度领域的热点问题，并取得了很多成果。本文通过对几个典型函数、经典的Car1—Car6问题以及Rec01，Rec13问题进行仿真测试，得出智能水滴算法在离散空间和连续空间优化的可行性与有效性，其具有很好的应用前景。目前，对于智能水滴算法应用到生产调度问题中的研究还很少，本文对其做了初步的研究，还存在很多问题，有待于做进一步的探讨。

参考文献

[1] 叶春明，陈子皓，寇明顺. 应用新型量子微粒群优化算法求解PFSP问题[J]. 技术与创新管理，2012，33(2)：162-165

[3] 黄华，肖菁，张军. 改进并行蚁群算法求解置换流水线调度问题[J]. 计算机工程与设计，2010，31(3)：582-585

[4] 张松艳. 基于遗传算法的大型Flow-shop生产调度[J]. 浙江科技学院学报，2010，22(2)：102-106

[5] 宋存利，时维国. 基于启发式信息的蚁群算法在车间作业调度中的应用研究[J]. 科学技术与工程，2008，8(12)：3359-3361

[6] 刘长平，叶春明. 一种新颖的仿生群智能优化算法：萤火虫算法[J]. 计算机应用研究，2011(9)

[7] 盛晓华，叶春明. 基于蝙蝠算法的PFSP调度干扰管理研究[J]. 计算机工程与应用，2012(8)

[8] Shah-Hosseini H. Problem solving by intelligent water drops[C]// IEEE Congress on Evolutionary Computation，2007，CEC 2007. Sept. 2007：3226-3231

[9] Shah-Hosseini H. The intelligent water drops algorithm：A nature-inspired swarm-based optimization algorithm[J]. International Journal of Bio-Inspired Computation，2009，1(1/2)：71-79

[10] Duan H，Liu S，Lei X. Air robot path planning based on intelligent water drops optimization[C]// Institute of Electrical and Electronics Engineers. Hong Kong，China，2008：1397-1401

[11] Duan H，Liu S，Wu J. Novel intelligent water drops optimization approach to single ucav smooth trajectory planning[J]. Aerospace Science and Technology，2009，13(8)：442-449

[12] Shah-Hosseini H. Optimization with the nature-inspired intelligent water drops algorithm[C]// Santos W P D，ed. Evolutionary Computation. Vienna：Tech，2009：297-320

[13] Carlier J. Ordonnancements a contrainses disjonctives [J]. R. A. I. R. O. Recherche Operationelle / Oper. Res.，1978，12：333-315

[14] Reeves C R. A genetic algorithm for flowshop sequencing [J]. Comput. Oper. Res.，1995，22：5-13

(上接第236页)

[5] Badoiu M，Har-Peled S，Indyk P. Approximate clustering via core-sets [C]// Proceedings of the 34th Annual ACM Symposium on Theory of Computing. 2002：250-257

[6] Badoiu M，Clarkson K L. Smaller core-sets for balls [C]// Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms. 2003：801-802

[7] Badoiu M，Clarkson K L. Optimal core-sets for balls [J]. Computational Geometry：Theory and Applications，2008，40(1)：14-22

[8] Kumar P，Mitchell J S B，Yildirim E A. Approximate minimum enclosing balls in high dimensions using core- sets [J]. The ACM Journal of Experimental Algorithmics，2003，8(1)

[9] Fischer K，Gartner B. The smallest enclosing ball of balls：Combinatorial structure and algorithms [J]. International Journal of Computational Geometry and Applications，2004，14：341-378

[10] Zhou G，Toh K C，Sun J. Efficient algorithms for the smallest enclosing ball problem [J]. Computational Optimization and Applications，2005，30：147-160

[11] Pan S H，Li X S. An efficient algorithm for the smallest enclosing ball problem in high dimensions [J]. Applied Mathematics and Computation，2006，172：49-61

[12] Yildirim E A. Two algorithms for the minimum enclosing ball problem [J]. SIAM Journal on Optimization，2008，19(3)：1368-1391

[13] 丛伟杰，刘红卫. 求解最小闭包球问题的一种SMO-型方法[J]. 西北大学学报：自然科学版，2010，40(6)：965-969

[14] Frank M，Wolfe P. An algorithm for quadratic programming [J]. Naval Research Logistics Quarterly，1956，3(1/2)：95-110

[15] Guelat J，Marcotte P. Some comments on Wolfe's away steps [J]. Mathematical Programming，1986，35(1)：110-119

[16] Chen P H，Fan R E，Lin C J. A study on SMO-type decomposition methods for support vector machines [J]. IEEE Transactions on Neural Networks，2006，17：893-908