计 算 机 研 究 与 发 展
Journal of Computer Research and Development
DOI: 10.7544/issn1000-1239.202330093
61(11): 2909−2922, 2024

基于归一分解的并行多目标 Dividing Rectangles 算法

李 晨[1] 陈逸东[1,2] 陆忠华[1] 杨雪莹[1,2] 王子田[3] 迟学斌[1]
[1](中国科学院计算机网络信息中心 北京 100083)
[2](中国科学院大学 北京 100049)
[3](北京智速科技有限公司 北京 100037)
(lichen@sccas.cn)

A Parallel Multi-Objective Dividing Rectangles Algorithm Based on Normalized Decomposition

Li Chen[1], Chen Yidong[1,2], Lu Zhonghua[1], Yang Xueying[1,2], Wang Zitian[3], and Chi Xuebin[1]
[1] (*Computer Network Information Center*, *Chinese Academy of Sciences*, *Beijing* 100083)
[2] (*University of Chinese Academy of Sciences*, *Beijing* 100049)
[3] (*Beijing AIHPC Co.*, *Ltd.*, *Beijing* 100037)

Abstract Multi-objective optimization problems are common in real-world applications. However, the high computational complexity severely hinders their solution. Currently, multi-objective evolutionary algorithms are mostly used for their solving. Nevertheless, these methods usually contain random operations in the process of population initialization and evolution to maintain diversity, which leads to non-reproducibility and lacking the theoretical guarantee of global convergence. Therefore, a novel multi-objective Dividing Rectangles(DIRECT) algorithm is introduced based on decomposition strategy. In this algorithm, the original problem is decomposed into a series of subproblems by a normalized decomposition method that can better capture the complex frontier to reduce the computational complexity of the problem. Meanwhile, the Dividing Rectangles algorithm is used to simultaneously optimize the decomposed subproblems, and the generated candidate solutions are assigned to the corresponding subproblems based on the global association mechanism in the optimization process to better retain the good candidate solutions and improve the algorithm search efficiency. Finally, the convergence of the algorithm is demonstrated. To further improve the computational efficiency, a multi-level and multi-granularity parallel scheme based on the adaptive associative migration strategy is proposed, based on which the proposed algorithm is parallelized. Furthermore, the proposed algorithm and its parallel version are applied to several benchmark optimization problems, and the experimental results show that the proposed algorithm can obtain a Pareto set with better convergence and diversity than NSGA-II, and the parallel version can further improve the accuracy of Pareto frontier approximation while massively reducing the problem-solving time.

Key words multi-objective optimization; objective space decomposition; Dividing Rectangles algorithm; parallel computing; global optimization

摘 要 多目标优化问题普遍存在且难以解决,目前多采用多目标进化算法进行求解.然而,这些方法通

收稿日期: 2023-02-16; **修回日期**: 2023-12-29
基金项目: 中国博士后科学基金项目(2021M693226); 国家自然科学基金项目(61873254)
This work was supported by the China Postdoctoral Science Foundation (2021M693226) and the National Natural Science Foundation of China (61873254).

常在种群初始化阶段和进化过程中包含随机操作以保持多样性,导致了其结果不可复现且缺乏全局收敛的理论保证.鉴于此,提出了一种基于归一分解的多目标Dividing Rectangles(DIRECT)算法,首先通过一种可较好捕捉复杂前沿的归一分解方法将原问题分解为一系列子问题,以降低问题计算复杂度;其次,采用Dividing Rectangles算法同时优化分解得到的子问题,并在优化过程中基于全局关联机制将生成的候选解分配给相应的子问题,以更好地保留优秀候选解并提高算法搜索效率;最后,证明了算法的收敛性.此外,为了进一步提高计算效率,提出了一种基于自适应关联迁移策略的多层次多粒度并行方案,并基于该方案对所提出的算法进行了并行化.将所提算法应用于多个基准优化问题,实验结果表明,相比于NSGA-II,所提串行算法能够产生收敛性、多样性更为优越的帕累托最优解集,并行算法可在大规模缩短问题求解时间的同时,进一步提升帕累托前沿近似精度.

关键词 多目标优化;目标空间分解;Dividing Rectangles算法;并行计算;全局优化

中图法分类号 TP301

多目标优化旨在同时优化2个或2个以上相互冲突的目标,在许多科学和工程问题中均有重要应用.例如,在投资组合选择中,投资者往往期望在较低风险水平下获得更高收益.然而收益的提升往往伴随着风险的增加,这意味着这2个目标在本质上是矛盾的.此时的投资组合选择问题便被称为一个多目标优化问题(multi-objective optimization problems, MOPs).在解决此类问题时,无论是否存在约束,我们通常只能获得一组权衡最优解(又称帕累托最优解),而无法获得单一最优解[1].

MOPs的目标函数通常具有多峰、不连续、不可导甚至离散等特点,无法通过传统的最优化方法来解决[2].因此,其求解方法成为了该领域的研究热点之一.在过去的几十年里,大量优秀的多目标优化算法被提出并成功应用,多目标进化算法(multi-objective evolutionary algorithms, MOEAs)便是目前比较流行的一类.该类算法基于种群搜索技术模拟自然进化过程,以得到具有良好收敛性和多样性的解集,非常适用于各种复杂的MOPs[3-4].NSGA-II[5]和MOEA/D[6]为其中最为典型的2种算法,为MOPs求解方法的后继研究[7-8]提供了坚实基础.

然而,MOEAs通常在种群初始化阶段和进化过程中包含随机操作以保持多样性,这导致了其结果不可复现且缺乏全局收敛的理论保证.在此背景下,一些研究人员开始基于确定性优化算法实现MOPs求解.Dividing Rectangles(DIRECT)算法是由Jones等人[9]在1993年提出的高效确定性全局优化算法,可在未进行“参数调优”的情况下平衡局部和全局搜索能力,目前已在各种单目标优化问题(single objective optimization problems, SOPs)中显示出了优越性能.近期,国内外研究人员开始探索其在MOPs上的适用性和改进方案[10].然而大多数基于非支配排序或HV(hypervolume)指标[11]构建.常用的快速非支配排序算法复杂度为$O(MN^2)$,其中M为目标个数,N为待排序点的数量[5];而计算d维n个非支配点的HV指标的算法复杂度下限为$O(n^{d/3}\text{poly}\log n)$[12].可以看出,随着非支配点和目标函数数量的增多,此类算法的计算开销将迅速增长,严重阻碍了其进一步发展及应用.

此外,随着问题规模的不断扩大和复杂程度的不断增加,人们对于多目标优化算法的求解质量和运行速度都提出了更高的要求[13].高性能计算(high performance computing, HPC)几乎在所有的科学学科中都发挥着重要的作用,是提高算法执行速度的有效途径.本文设计并实现了一种基于归一分解的并行多目标Dividing Rectangles算法,来求解复杂的MOPs.本文的主要贡献包括4个方面:

1)提出了一种归一分解方法,可更好地捕捉复杂前沿的同时使得算法维护较好的空间分布性和探索能力;此外,提出了一种全局关联机制,以更好地保留优秀候选解并提高算法搜索效率.

2)基于上述分解方法和关联机制,提出了一种多目标Dividing Rectangles算法并详细给出了收敛性分析.

3)采用多层次多粒度并行的思想,将所提出的算法进行了并行化,并提出了一种自适应关联迁移策略,以解决因单个计算节点内子问题数量较少而易陷入局部最优的问题.

4)将所提算法应用于不同的2-目标和3-目标基准优化问题,并与经典多目标优化算法进行对比,验证了算法的优越性.此外,实验结果表明,自适应关联迁移策略可提高并行算法的帕累托前沿近似精度.

1 相关工作

多目标优化算法的发展历程可以追溯到优化研究的早期，其实质是在多个目标之间实现最佳权衡. 随着进化计算的出现，基于进化和群体智能原则的多目标优化算法（即 MOEAs）开始涌现，直到今日依旧是该领域最热门的研究方向之一. 其中几种代表性算法包括：基于帕累托支配关系的 NSGA-II[5]，SPEA2[14]，NSGA-III[7]，CMPSO[15]，基于分解策略的 MOEA/D[6]，MOEA/D-M2M[16]，RVEA[17] 和基于评价指标的 IBEA[18]，HypE[19]，iR2-MOACO[20]. 特别是 Deb 等人[5] 提出的 NSGA-II，被认为是迄今为止最优秀的多目标优化算法之一. 该算法基于遗传算法进行解空间搜索并使用非支配排序机制对候选解进行排序，同时使用拥挤距离来维护解的多样性，避免过早收敛. 此外，2007 年 MOEA/D 的提出，是多目标优化领域的另一个重大创新. 该算法基于数学规划方法将原问题分解成多个单目标优化问题，并基于进化算法进行求解，显著降低了原问题的求解难度. 尽管 MOEAs 已被证明对解决大规模复杂 MOPs 非常有效，但是该类算法通常在种群初始化阶段和进化过程中包含随机操作，使其结果不可复现且缺乏全局收敛的理论保证.

1993 年，Jones 等人[9] 在 Lipschitzian 框架[21] 的基础上提出了 Dividing Rectangles 算法，该算法不需要 Lipschitz 常数先验知识，大大提升了算法通用性，同时在各种 SOPs 中显示出了优越性能. 近期，该算法开始受到多目标优化领域学者的关注，并取得了一些研究进展. Wang 等人[22] 基于非支配排序和拥挤度距离提出了一种改进的 Dividing Rectangles 算法，并在此基础上与多目标遗传算法（multi-objective genetic algorithm, MOGA）相结合，提出了一种名为 NSDIRECT-GA 的多目标优化算法，使用改进的 Dividing Rectangles 算法来生成 MOGA 中的种群个体. 实验表明，相较于 NSGA-II，NSDIRECT-GA 可获得分布性更优的帕累托解集. Al-Dujaili 等人[23] 在原始 Dividing Rectangles 算法的基础上进行扩展，提出了 MO-DIRECT. 此外，他们将 MO-DIRECT 与其他基于分解的多目标优化技术进行对比，验证了其优越性. Campana 等人[24] 通过将 Dividing Rectangles 算法与无梯度局部优化算法相耦合设计了一种多目标 Dividing Rectangles 算法，用于解决非线性约束 MOPs. Wong 等人[1] 将 HV 指标引入到 MO-DIRECT 算法中，提出了 MO-DIRECT-hv 算法，通过选择具有较高 HV 贡献度的潜在最优超矩形（potentially-optimal hyperrectangle）来增强其搜索的多样性，同时保持算法的局部和全局搜索能力. 然而，以上算法大多数基于非支配排序或 HV 指标构建，随着非支配点和目标函数数量的增多，此类算法的计算开销将迅速增长，严重地阻碍了其进一步发展及应用. 此外，日渐增大的问题规模和不断提升的复杂程度对求解算法的性能提出了更高的要求，但相关的并行算法及实现技术研究薄弱.

2 背　景

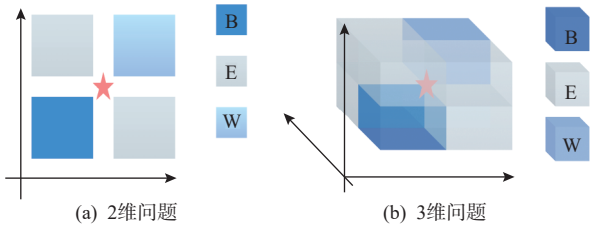
2.1 多目标优化

通常，对于有 n 个决策变量、m 个最小化目标的多目标优化问题 I，可以形式化描述为

$$\begin{aligned} &\min F(\boldsymbol{x})=(f_1(\boldsymbol{x}),f_2(\boldsymbol{x}),\cdots,f_m(\boldsymbol{x}))^{\mathrm{T}},\\ &\text{s.t.}\quad \boldsymbol{x}=(x_1,x_2,\cdots,x_n)\in\Omega. \end{aligned} \tag{1}$$

其中 $\boldsymbol{x}\in\mathbb{R}^n$ 称为决策向量，Ω 为可行决策空间（decision space），$F(\boldsymbol{x})\in\mathbb{R}^m$ 为目标向量.

与 SOPs 不同，没有一个单一的最优解可以同时优化 MOPs 中的所有目标[16]. 取而代之的是，通过“支配关系”来比较 MOPs 搜索域中的不同候选解[25]. 我们称候选解 $\boldsymbol{a}$ 支配（dominate）候选解 $\boldsymbol{b}$（$\boldsymbol{a}\prec\boldsymbol{b}$），当且仅当 $f_i(\boldsymbol{a})\leqslant f_i(\boldsymbol{b})\ \forall i\in\{1,2,\cdots,m\}$ 且 $F(\boldsymbol{a})\neq F(\boldsymbol{b})$. 候选解（如图 1 中五角星所示）在 2 维和 3 维问题中的支配关系如图 1 所示. 其中，E 表示该部分区域的解与候选解互不支配，B 表示该部分区域的解支配候选解，W 表示该部分区域的解被候选解支配. 如果候选解 $\boldsymbol{p}$ 在搜索域中不被任何其他候选解所支配，则称其为帕累托最优（Pareto optimal），称 $F(\boldsymbol{p})$ 为帕累托最优目标向量. 决策空间 Ω 上所有的帕累托最优解所构成的



(a) 2维问题　(b) 3维问题

Fig. 1 Dominance of the candidate solutions (pentagrams above) in the two-dimensional and three-dimensional MOPs

图 1 候选解（五角星）在 2 维和 3 维 MOPs 中的支配关系

集合称为帕累托解集(Pareto set, PS),所有的帕累托最优目标向量组成帕累托前沿(Pareto front, PF)[26]. 能否高效率地找到尽量多的帕累托最优解逼近 PF,是衡量多目标优化算法性能的标准.

2.2 Dividing Rectangles 算法

Dividing Rectangles 算法是一种高效的确定性单目标全局优化算法[9],在工程设计、科学研究等领域中应用广泛[27-28]. 该算法将整个决策空间看作一个超立方体,并不断地进行分割以得到新的子超矩形,直到某个限定的收敛条件(迭代的次数或分割矩形的数目等)被满足.

$n=2$时,算法的前 2 次迭代过程如图 2 所示,蓝色区域为潜在最优矩形,空心点表示矩形的中心点,实心点表示新采样得到的点. 算法在每一轮按照一定的规则选择潜在最优矩形,之后确定最长边方向(可能不止 1 个)并在$c\pm\delta e_i$处进行采样,其中,c为当前潜在最优矩形的中心点,e_i为第i个最长边方向,δ是最长边长度的 1/3. 随后,依次选择$w_i=\min(f(c+\delta e_i),f(c-\delta e_i))$最小的矩形的最长边方向进行 3 等分切割生成新矩形. 重复上述过程,直到迭代结束.

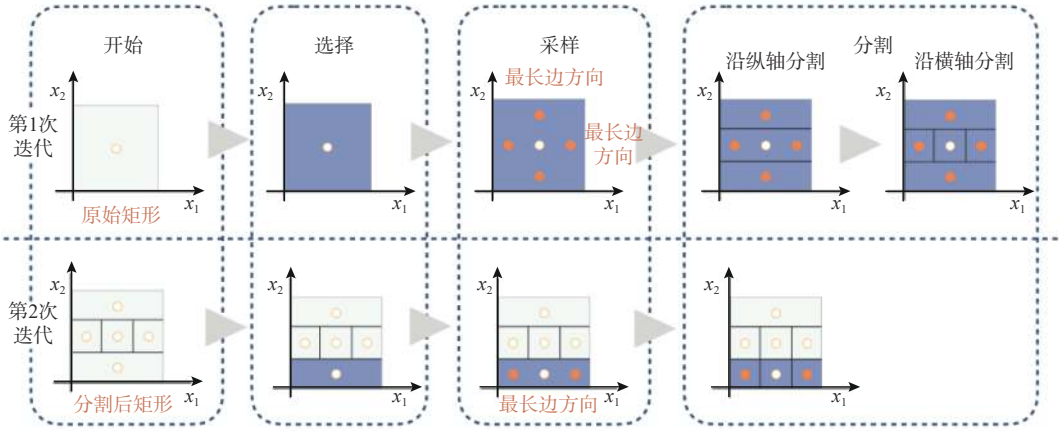


Fig. 2 First two iterations of the Dividing Rectangles algorithm (n=2)

图 2 Dividing Rectangles 算法的前 2 次迭代过程(n=2)

3 基于归一分解的多目标 Dividing Rectangles 算法

3.1 归一分解方法

多目标 Dividing Rectangles 算法在每轮迭代中需要对超矩形进行评价与比较,该过程将在很大程度上影响算法的多样性和收敛性,目前多基于非支配排序与 HV 指标实现. 然而,这类方法计算复杂度较高,使得多目标 Dividing Rectangles 算法收敛速度过慢,不利于其实际应用. 因此,如何解决超矩形评价过程中的计算困难问题成为多目标 Dividing Rectangles 算法设计的关键.

目标空间分解技术可根据均匀分布在目标空间中的权重向量,将原问题转换为一系列与特定权重向量相关联的单目标子问题,从而显著降低算法计算复杂度,目前已被广泛应用于多目标优化领域[29-30]. 常用的分解方法有加权和法(weighted sum approach, WS)、切比雪夫法(Tchebycheff approach, Tch)和边界交叉法(boundary intersection approach, BI)[31]. 然而,以上方法大多对前沿形状十分敏感,在捕捉凸形、凹形或陡峭等复杂前沿时,难以同时维护较好的空间分布性和探索能力[30]. 在本文中,我们提出了一种归一分解方法,使算法在保持多样性的同时可更快更好地收敛到 PF. 由该方法得到的子问题的数学表达式为:

$$\begin{aligned}&\min\ g^{\mathrm{sbi}}(\boldsymbol{x}|\boldsymbol{\lambda},\boldsymbol{z}_{\mathrm{ref}})=d_2,\\&\text{s.t. }d_1=\frac{\|(F(\boldsymbol{x})-\boldsymbol{z}_{\mathrm{ref}})^{\mathrm{T}}\boldsymbol{\lambda}\|}{\|\boldsymbol{\lambda}\|},\\&d_2=\frac{d_1-d_1^{\min}}{d_1^{\max}-d_1^{\min}}.\end{aligned}\tag{2}$$

其中:$\boldsymbol{x}$和$F(\boldsymbol{x})$分别为原问题的决策向量和目标向量;$\boldsymbol{\lambda}=(\lambda_1,\lambda_2,\cdots,\lambda_m)$为预先生成的权重向量,对于$\forall s\in\{1,2,\cdots,m\}$, $\lambda_s>0$且$\sum_{s=1}^{m}\lambda_s=1$, m为原问题目标数量;$\boldsymbol{z}_{\mathrm{ref}}=(z_1,z_2,\cdots,z_m)$为参考点,$z_k$是当前目标$k$方向上的最优值. $d_1^{\max}$和$d_1^{\min}$分别为当前$\boldsymbol{\lambda}$方向上投影最大值和最小值.

权重向量作为归一分解方法的前提,其生成方式至关重要. 不同的权重向量会得到不同的子问题集合,进而得到不同的帕累托最优解. 为了得到均匀

分布的帕累托最优解，需要产生 1 组在目标空间内均匀分布的权重向量$\boldsymbol{\lambda}^1,\boldsymbol{\lambda}^2,\cdots,\boldsymbol{\lambda}^N$($N$为权重向量数目)，具体方式描述为：

$$\sum_{s=1}^{m}\lambda_s^i=1,\qquad(3)$$

$$\lambda_s^i\in\left\{\frac{0}{H},\frac{1}{H},\cdots,\frac{H}{H}\right\},$$

其中H为预先设置的一个正整数. 该方法可保证N个权重向量都落在一个平面上. 此外，$N=\mathrm{C}_{H+m-1}^{m-1}$.

通过归一分解方法，问题I可被转换为数学形式为式(2)的N个单目标子问题：

$$\begin{cases}\min g^{\mathrm{sbi}}\left(\boldsymbol{x}\mid\boldsymbol{\lambda}^1,\boldsymbol{z}_{\mathrm{ref}}\right),\\ \min g^{\mathrm{sbi}}\left(\boldsymbol{x}\mid\boldsymbol{\lambda}^2,\boldsymbol{z}_{\mathrm{ref}}\right),\\ \quad\vdots\\ \min g^{\mathrm{sbi}}\left(\boldsymbol{x}\mid\boldsymbol{\lambda}^N,\boldsymbol{z}_{\mathrm{ref}}\right).\end{cases}\qquad(4)$$

通过对N个子问题进行优化，可得到问题I可达目标集左下边界(PF 是其一部分)与从参考点发出的一组直线的交点，形成对整个 PF 的良好近似.

3.2 全局关联机制

如何将产生的候选解与子问题关联起来同样是影响算法效率的关键环节. 不合理的关联机制会错误地淘汰优秀候选解，严重影响收敛速度.

为了解决上述问题，本文提出一种全局关联机制，将从所有子问题中全局地挑选一个与后代个体最匹配的子问题进行关联. 具体地，将参考点到候选解的向量在权重向量$\boldsymbol{\lambda}^i$上的投影值作为度量候选解与第i个子问题之间的关联度，投影值越小关联度越大. 选择与该候选解关联度最大的子问题进行关联. 如图 3 所示，相较于权重向量 2，参考点到候选解$\boldsymbol{x}^i$的向量在权重向量 1 上的投影值更小，因此候选解$\boldsymbol{x}^i$被关联到了子问题 1.

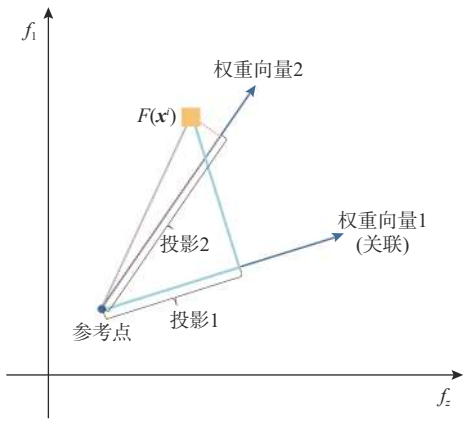


Fig. 3 Global association mechanism

图 3 全局关联机制

3.3 多目标 Dividing Rectangles 算法框架

基于以上分解方法与关联机制，本文提出了一种多目标 Dividing Rectangles 算法(简称 DEmo-DIRECT)，用于处理复杂 MOPs. 该算法基于归一分解方法将原始问题转换为N个子问题，采用 Dividing Rectangles 算法同时优化这N个子问题，并在优化过程中基于全局关联机制将生成的候选解分配给相应的子问题，最终形成对问题 PF 的有效近似，算法框架如算法 1 所示.

算法 1. DEmo-DIRECT 算法.

输入：最大迭代次数T，正数tol；

输出：帕累托集 PS.

① 产生N个权重向量$\boldsymbol{\lambda}^1,\boldsymbol{\lambda}^2,\cdots,\boldsymbol{\lambda}^N$；

② 设置参考点$\boldsymbol{z}_{\mathrm{ref}}$；

③ 将决策空间归一化为一个超立方体$\hbar$，设$\boldsymbol{c}_1$为超立方体的中心点并计算$g^{\mathrm{sbi}}(\boldsymbol{c}_1)$；

④ 令$S_0=\hbar$，$t=0$，$r=1$；

⑤ while $t<T$ do

⑥ 基于相应选择策略生成潜在最优超矩形集合I_t，其个数为$\tilde{r}$；

⑦ $S_t\leftarrow S_t-I_t$；

⑧ $r=r-\tilde{r}$；

⑨ for $i\leftarrow 1$ to $\tilde{r}$ do

⑩ 令潜在最优超矩形i的每个最长边$\boldsymbol{c}_i\pm\delta_i\boldsymbol{e}_k$处的点为新采样点；

⑪ 计算所有新采样点的目标向量；

⑫ end for

⑬ 更新$\boldsymbol{z}_{\mathrm{ref}}$；

⑭ while $I_t\neq\varnothing$ do

⑮ 选择一个潜在最优超矩形j；

⑯ 将其新采样点与子问题相关联；

⑰ 更新子问题的最大、最小投影值；

⑱ 计算所有新采样点的g^{sbi}；

⑲ 对超矩形j的每个最长边，计算$w_k=\min(g^{\mathrm{sbi}}(\boldsymbol{c}_j+\delta_j\boldsymbol{e}_k)g^{\mathrm{sbi}}(\boldsymbol{c}_j-\delta_j\boldsymbol{e}_k))$；

⑳ 依次选择w_k最小的方向对超矩形所有最长边进行 3 等分；

㉑ $I_t\leftarrow I_t-\{j\}$；

㉒ end while

㉓ $S_{t+1}\leftarrow S_t\cup\{\text{新生成超矩形}\}$；

㉔ $r=r+$新生成超矩形个数；

㉕ $t=t+1$；

㉖ end while

㉗ 选择$g^{\mathrm{sbi}}<tol$的超矩形以形成 PS；

㉘ return PS.

在初始化阶段, DEmo-DIRECT 算法将解空间规范化为一个超矩形, 并生成参考点 $\boldsymbol{z}_{\text{ref}}$ 与 N 个权重向量. 之后, 将迭代地识别和分割潜在最优超矩形, 直到达到最大迭代次数. 在这个过程中, 算法将根据如下策略选择潜在的最优超矩形.

选择策略: 假设当前已存在 r 个超矩形, 如果存在某个常数 $\widetilde{K}$, 使得第 j 个超矩形满足式(5), 则对应的超矩形被称为潜在最优超矩形.

$$\begin{aligned} &g^{\text{sbi}}\left(\boldsymbol{c}_j\right)-\tilde{K}d_j \leqslant g^{\text{sbi}}(\boldsymbol{c}_i)-\tilde{K}d_i,\ i=1,2,\cdots,r,\\ &g^{\text{sbi}}\left(\boldsymbol{c}_j\right)-\tilde{K}d_j \leqslant g^{\text{sbi}}_{\min}-\epsilon\left|g^{\text{sbi}}_{\min}\right|, \end{aligned} \tag{5}$$

其中 $\boldsymbol{c}_j$ 为第 j 个超矩形的中心, d_j 为第 j 个超矩形中心到其顶点的距离, ϵ 为任意正常数, $g^{\text{sbi}}_{\min}$ 是当前最优 g^{sbi} 值.

之后, 算法 1 将在所有潜在最优超矩形的所有最长边的 $\boldsymbol{c}_i \pm \delta_i \boldsymbol{e}_k$ 处进行采样, 并按照 w_k 从小到大的顺序对最长边进行 3 等分. 最终, 算法选择 $g^{\text{sbi}} < tol$ 的超矩形, 形成帕累托集 PS.

在 DEmo-DIRECT 中, 具备相同 g^{sbi} 值和对角线长度的超矩形将有同样的机会被选为潜在最优矩形, 即使它们相关联的子问题不同. 也就是说, 算法在实现潜在最优超矩形选择时, 可同时兼顾多个参考方向上的子问题. 即使当前已求解点集中于少数几个权重方向, 由于算法具有优秀的全局探索能力, 一旦发现新的权重方向上的解, 将能保证在新一轮迭代过程中对相关子问题实现进一步搜索.

3.4 收敛性分析

若 MOP 的目标函数是连续的, 或至少在其 PF 附近是连续的, 则 DEmo-DIRECT 算法可在理论上保证收敛到该问题的 PF, 具体证明如下.

引理 1. 当迭代次数 $t \to \infty$ 时, DEmo-DIRECT 算法生成的所有超矩形的最少分割次数 $D(t) \to \infty$.

证明. 如果 $\lim\limits_{t\to\infty} D(t) < \infty$, 那么一定存在一些迭代步 t', 使得该步之后 $D(t)$ 将不再增加, 即 $\lim\limits_{t\to\infty} D(t) = D(t')$. 假设在迭代步 t' 结束时, S_t 中有有限(P)个超矩形的分割次数为 $D(t')$, 那么这些超矩形具有最大的中心到顶点的距离. 设超矩形 j 的 g^{sbi} 值最小, 由于对 $\forall i = 1,2,\cdots,r$, 皆有 $\left(d_j - d_i\right) \geqslant 0$, 则根据 3.3 节中的选择策略可知, 一定存在 $\widetilde{K}$ 同时满足

$$\begin{aligned} &\tilde{K}\left(d_j-d_i\right) \geqslant g^{\text{sbi}}\left(\boldsymbol{c}_j\right)-g^{\text{sbi}}(\boldsymbol{c}_i),\ i=1,2,\cdots,r,\\ &\tilde{K} \geqslant \frac{g^{\text{sbi}}\left(\boldsymbol{c}_j\right)-g^{\text{sbi}}_{\min}+\epsilon\left|g^{\text{sbi}}_{\min}\right|}{d_j}, \end{aligned} \tag{6}$$

则在 $t'+1$ 次迭代时, 超矩形 j 将成为潜在最优超矩形, 继而被分割. 同理, 其余 $P-1$ 个超矩形也都将在 $t'+P$ 代之内被划分. 这意味着 $D(t'+P) \geqslant D(t')+1$, 与 $\lim\limits_{t\to\infty} D(t) = D(t')$ 假设相矛盾. 因此, $\lim\limits_{t\to\infty} D(t) = \infty$ 成立. 证毕.

定理 1. 若 MOP 的目标函数是连续的, 或至少在其 PF 附近是连续的, 当迭代次数 $t \to \infty$ 时, DEmo-DIRECT 算法将收敛于该问题的 PF.

证明. 由于新的超矩形是沿着最长边方向 3 等分的, 则超矩形边的长度皆为 3^{-y}, y 为非负整数, 且长度为 3^{-y} 的边被分割完之前, 长度为 $3^{-(y+1)}$ 的边无法被分割. 因此, 经过 $D(t)$ 次划分后, 超矩形将有 $D(t) \bmod n$ 条长度为 $3^{-(y+1)}$ 的边和 $n - D(t) \bmod n$ 条长度为 3^{-y} 的边, 其中 $y = (D(t) - D(t) \bmod n)/n$, n 为决策变量数. 则超矩形从中心到顶点的距离为

$$d = [(D(t) \bmod n)3^{-2(y+1)} + (n - D(t) \bmod n)3^{-2y}]^{0.5}/2. \tag{7}$$

从引理 1 中可知: $\lim\limits_{t\to\infty} D(t) = \infty$. 因此, 随着迭代次数 $t \to \infty$, 超矩形从中心到顶点的距离为 0. 那么, 对于任一 $\delta > 0$, 都存在一正整数 T, 使得在 $t > T$ 次迭代后存在的所有超矩形的中心到顶点的距离都小于 δ. 换句话说, 超立方体中的任一点 $\boldsymbol{x}$ 最终都将位于 DEmo-DIRECT 的某个采样点的 δ 距离内, 即 DEmo-DIRECT 在所有迭代过程中采样的点形成了初始超立方体的密集子集. 因此, 只要 MOP 的目标函数在其 PF 附近是连续的, 则 g^{sbi} 便在全局最小值的邻域内连续, DEmo-DIRECT 算法便可以收敛到所有子问题的全局最优函数值, 进而收敛到原始问题的 PF. 证毕.

4 并行 DEmo-DIRECT 算法

随着现代计算科学的进步, 现实科学与工程实践中存在的 MOPs 多以大量的决策变量、目标和昂贵的评估函数为特征, 使得其计算复杂度进一步提升, 计算时间将成为实际应用瓶颈. 为了满足实际应用需求, 且更好地利用当前多处理器节点集群体系结构, 本文基于 MPI+OpenMP 混合编程模型对 DEmo-DIRECT 算法实现多层次多粒度并行, 在缩短问题求解时间的同时进一步提升 PF 近似精度.

4.1 基于自适应关联迁移策略的多层次多粒度并行方案

由于 DEmo-DIRECT 算法是通过同时优化不同的子问题来形成对问题 PF 的有效近似, 且更新机制具有明显的局部性, 所以可在子问题间采用粗粒度并行, 将不同的子问题分配到不同的计算节点上进

行计算. 所有求出的解聚合到一起就构成了完整的 PF. 在节点内部, 为了进一步利用其多核或众核资源, 可将子问题优化过程中计算量较大的任务继续分解, 分配给不同的计算核心去执行.

4.1.1 自适应关联迁移策略

子问题间粗粒度并行虽然加快了算法的执行速度, 但相比所有子问题同时优化而言, 单个计算节点内子问题数量较少, 规避了不同的解引导转移搜索方向的情况, 易陷入局部最优. 为了解决上述问题, 本文提出了一种自适应关联迁移策略.

将该策略中对算法在通信开销、稳定性、可扩展性和准确性等方面具有较大影响的因素描述如下:

1)迁移频率

为了更好地保留原问题帕累托解集的多样性, 理想的迁移应每代进行, 但频繁迁移会带来过高的通信开销. 本文提出了一种多样性指标 *DN*(diversity in node), 并基于该指标对节点内个体的多样性进行评判, 进而确定是否进行迁移. *DN* 指标定义为

$$DN=\sum_{i=1}^{n}\left|g_i^{\mathrm{sbi}}-g_{\min}^{\mathrm{sbi}}-\sum_{j=1}^{n}\left(g_j^{\mathrm{sbi}}-g_{\min}^{\mathrm{sbi}}\right)/n\right|, \tag{8}$$

其中 n 为节点内个体(潜在最优超矩形)总数. *DN* 小于预定的阈值时, 节点内多样性不足, 需要进行个体替换, 由此实现自适应迁移, 以避免不必要的通信开销.

2)迁出个体选择策略

不同于一般迁移策略中选择节点内较优的个体(潜在最优超矩形), 本文基于全局关联机制进行迁出个体选择. 定义节点内个体 $\boldsymbol{x}$ 与其相邻节点的关联度为

$$Co(\boldsymbol{x},Nei)=\sum_{\lambda\in Nei}\frac{\|(F(\boldsymbol{x})-\boldsymbol{z}_{\mathrm{ref}})^{\mathrm{T}}\boldsymbol{\lambda}\|}{\|\boldsymbol{\lambda}\|}, \tag{9}$$

其中 *Nei* 为邻居节点中所有子问题的权重向量集合. 本文将选择 *Co* 值最大的若干个个体作为迁出个体.

4.1.2 算法并行方案

基于 MPI+OpenMP 混合编程模型, 我们给出了 DEmo-DIRECT 算法的基于自适应关联迁移策略的多层次多粒度并行方案, 具体如图 4 所示(以 3 节点为例).

现将算法并行方案说明如下. 启动 m 个 MPI 进程, 由 0 号进程初始化 N 个权重向量, 并基于欧氏距离对其进行排序. 随后, 使用函数 *MPI_Scatter* 将排序好的权重向量平均地划分给各个进程. 每个 MPI 进程同时优化与权重向量相关的 N/m 个子问题, 并采用点对点的方式与其邻居进程进行通信, 实现候选解迁移.

在进程内部, 将采样、关联子问题与潜在最优超矩形分割等计算量较大的核心计算模块进行线程级并行, 以充分利用节点内的多核/众核资源. 为了减少线程在核上动态迁移带来的额外开销, 我们采用了线程绑定的方式. 具体地, 在每个 MPI 进程内, 首先使用主线程来计算具有数据全局性特点的潜在最优超矩形选择过程. 之后, 在 OpenMP 并行域内, 每个线程为各自负责的潜在最优超矩形生成采样点并计算其目标向量, 将新采样点全局关联到进程内的相关子问题, 计算所负责的新采样点的 g^{sbi} 并进行潜在最优超矩形分割. 此过程中需要注意的是, 由于共享变量(子问题的最大和最小投影值等)的存在, 多线程同时访问可能引发竞争条件, 因此需要使用原子操作语句来确保对共享变量的安全访问; 此外, 需使用同步语句来确保正确的执行顺序和数据共享. 最后, 主线程计算 *DN* 指标, 并在满足预设条件时将目前 *Co* 值最大的若干个候选解迁出到其邻居进程.

4.2 并行算法流程

采用 4.1 节的并行方案, 则基于 MPI+OpenMP 的并行 DEmo-DIRECT 算法(pDEmo-DIRECT)流程如算法 2 所示.

算法 2. pDEmo-DIRECT 算法.

输入: 最大迭代次数 T;

输出: 帕累托集 PS.

① 启动 m 个 MPI 进程; /**MPI_INIT* ()*/

② if *rand_id* = 0 then

③ 初始化 N 个权重向量;

④ 权重向量排序;

⑤ 将权重向量平均分配给各个进程;

/**MPI_Scatter* ()*/

⑥ end if

⑦ for 每一个进程 do

/*每个进程负责处理 N/m 个子问题*/

⑧ $t\leftarrow 0$;

⑨ repeat

⑩ 潜在最优超矩形选择;

⑪ 设置 OpenMP 线程数;

/*omp_set_num_threads */

⑫ for 每一个线程 do

/*pragma omp parallel for*/

⑬ 在潜在最优超矩形的每个最长边处进行采样；
⑭ 计算新采样点处的目标函数值；
⑮ 新采样点与子问题关联；
⑯ 更新子问题的最大、最小投影；
⑰ 计算新采样点处的 g^{sbi}；
⑱ 潜在最优超矩形切割；
⑲ end for
⑳ 计算指标 DN；
㉑ if $DN \leqslant$ 预定阈值 then
㉒ 候选解迁移；
㉓ end if
㉔ $t = t + 1$；
㉕ until $t \geqslant T$
㉖ end for
㉗ return PS.

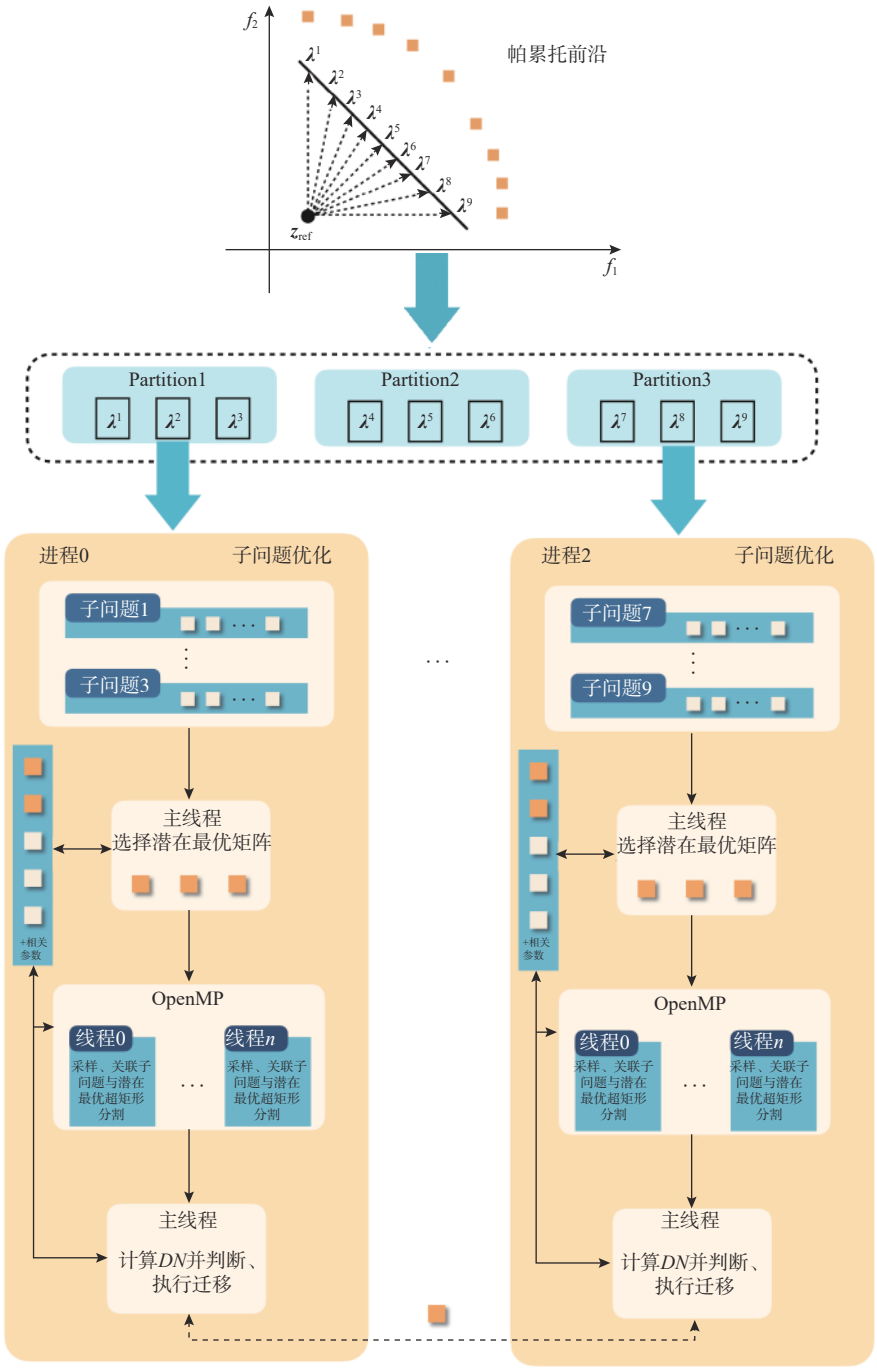


Fig. 4 Parallelism scheme for DEmo-DIRECT algorithm

图 4 DEmo-DIRECT 算法的并行方案

5 实验结果与分析

基于超级计算机“元”的高性能计算环境进行数值实验，从帕累托最优解集的收敛性和多样性 2 个维度将 DEmo-DIRECT 与目前主流的多目标优化算法进行了性能对比，以验证算法的有效性. 此外，分析了 pDEmo-DIRECT 算法的加速效果.

5.1 实验环境与参数设置

本文的所有实验都是在“元”超级计算机的刀片计算系统上进行的. 该系统包含 270 个计算节点(单节点配置 2 颗 Intel Xeon E5-2 680 V2 多核处理器，处理器时钟频率为 2.8 GHz，内存 64/128 GB)，节点间采用 56Gb FDR InfiniBand 网络全线速互连. 编译器和 MPI 库分别是 Intel Compiler 2013_sp1.0.080 和 Intel MPI 4.1.3.049.

ZDT 系列函数由 Zitzler 等人[32]在 2000 年提出，目前已成为多目标优化领域中经典的 2-目标测试用例[33]. 本文使用其中 4 个函数，分别为 ZDT1(凸面的)、ZDT2(凹面的)、ZDT3(非连续的)、ZDT6(凹面非均匀的)，用于验证算法在不同 PF 特征下的性能. 此外，选取 DTLZ 系列函数中的 3-目标 DTLZ5[34]作为另一个测试用例. 该测试用例为凹面退化的，用于考察算法在高维空间中收敛到低维 PF 的能力. 各测试用例皆为无约束最小化问题.

NSGA-II 是最优秀和最稳定的多目标优化算法之一，已经被验证在快速逼近真实 PF 和维护解集多样性这 2 个方面效果良好，目前已成为其他多目标优化算法性能的基准之一. 因此，我们选择该算法作为 DEmo-DIRECT 的对比算法. 在本文所有测试用例下，NSGA-II 的种群数量均设为 200，交叉和变异概率分别为 0.9 和 $1/nvar$($nvar$为测试用例变量数)，且保证其目标函数计算次数大于 DEmo-DIRECT. 在 DEmo-DIRECT 中，我们设置 $\epsilon = 10^{-4}$，$tol = 0.005$，而停止条件 T 则根据不同测试用例进行调整. 并行版本的参数与串行算法保持一致.

5.2 评价指标

在进行帕累托最优解集质量评价时，不仅要考虑获得的 PF 与真实 PF 的趋近程度，还需考虑前者在目标空间中的分布均匀程度和广泛程度等[35]. 在本文中，我们选择广泛使用的 GD(generational distance)[36]、IGD(inverted generational distance)[37]、HV 指标用于帕累托最优解集的质量对比. GD 和 IGD 值越小，HV 值越大，表示算法所得解集的性能越优[38].

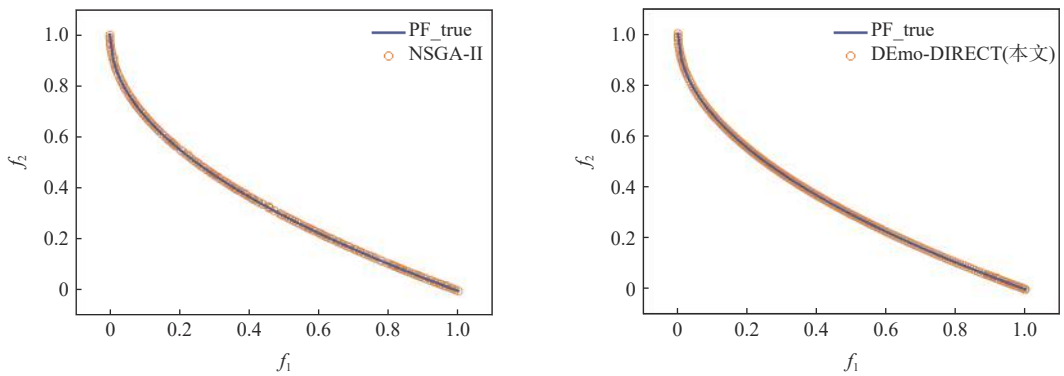
5.3 结果分析

5.3.1 DEmo-DIRECT 有效性分析

基于以上 5 个测试用例，我们从 HV, GD, IGD, PF 这 4 个方面对 NSGA-II 与 DEmo-DIRECT 进行了比较. 图 5~9 展示了上述 2 种算法在不同测试用例上的 PF 对比情况，其中曲线为测试用例的真实 PF.

从图 5~9 可以看出，在本文所有测试用例下，DEmo-DIRECT 都能够很好地近似到问题的真实 PF; 除了 ZDT3 测试用例外，DEmo-DIRECT 都可得到在均匀性和覆盖性上优于 NSGA-II 的 PF; 对于 ZDT3 测试用例，DEmo-DIRECT 与 NSGA-II 的表现不相上下. 也就是说，DEmo-DIRECT 可在保证收敛性的同时，获得分布均匀程度和广泛程度都较为出色的帕累托解集，且具有较为优秀的从高维空间中收敛到低维 PF 的能力.

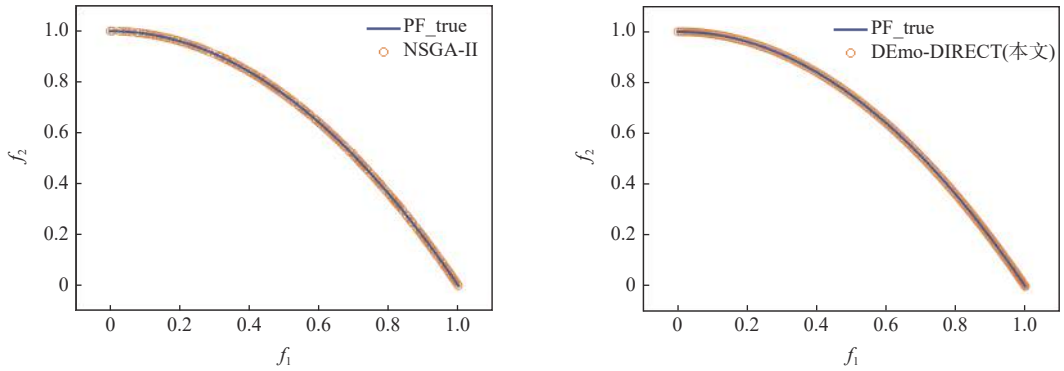
下面给出 DEmo-DIRECT 与 NSGA-II 的性能度量指标对比，定量地说明其优越性. 对于每一个测试用例，NSGA-II 独立运行 30 次以计算其 PF 的各项性



(a) NSGA-II (b) DEmo-DIRECT

Fig. 5 PF comparison between NSGA-II and DEmo-DIRECT on ZDT1

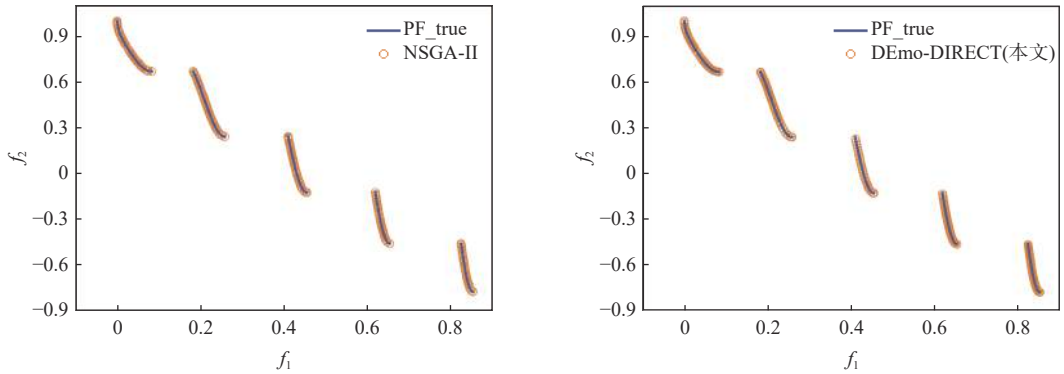
图 5 NSGA-II 与 DEmo-DIRECT 在 ZDT1 上的 PF 对比



(a) NSGA-II (b) DEmo-DIRECT

Fig. 6 PF comparison between NSGA-II and DEmo-DIRECT on ZDT2

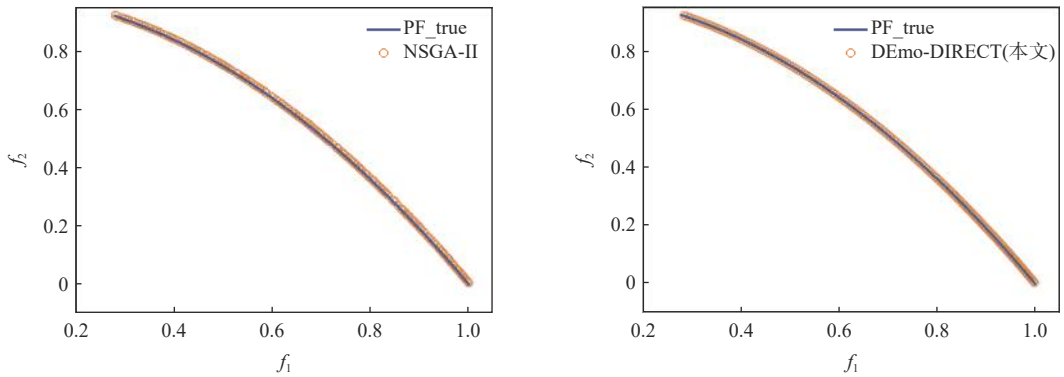
图 6 NSGA-II 与 DEmo-DIRECT 在 ZDT2 上的 PF 对比



(a) NSGA-II (b) DEmo-DIRECT

Fig. 7 PF comparison between NSGA-II and DEmo-DIRECT on ZDT3

图 7 NSGA-II 与 DEmo-DIRECT 在 ZDT3 上的 PF 对比



(a) NSGA-II (b) DEmo-DIRECT

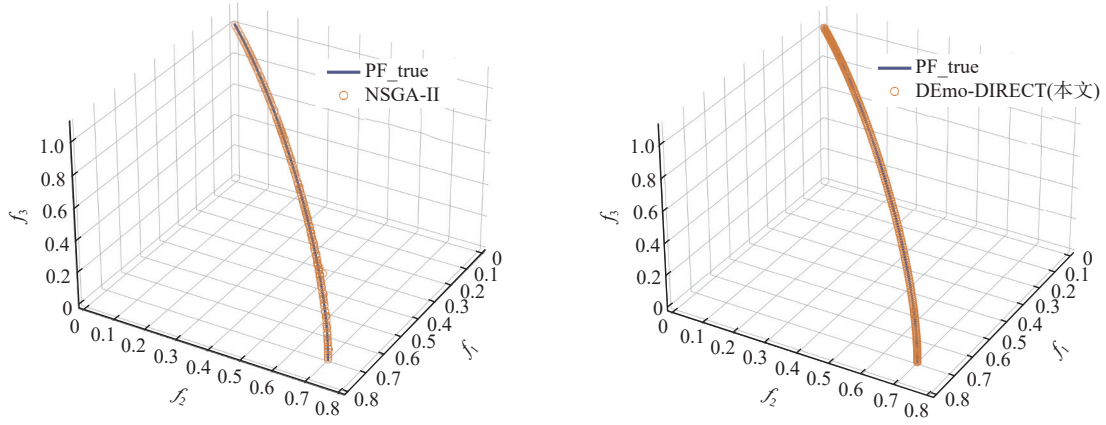
Fig. 8 PF comparison between NSGA-II and DEmo-DIRECT on ZDT6

图 8 NSGA-II 与 DEmo-DIRECT 在 ZDT6 上的 PF 对比

能指标，并统计它们的均值和方差. 表 1 展示了 2 种算法的性能指标对比情况.

由表 1 可以看出，对于除 ZDT3 外的所有优化问题，DEmo-DIRECT 的 GD，IGD，HV 指标均优于 NSGA-II；而在 ZDT3 问题中，DEmo-DIRECT 在 IGD 与 HV 指标上略逊一筹，在 GD 指标上具有显著优势，说明其与 NSGA-II 在此测试用例上表现不相上下，这也与 PF 对比实验结果一致.

综上所述，DEmo-DIRECT 在具有凸的、非凸的以及非均匀的 PF 特征的 2-目标优化问题上均有着较为优秀的表现，在求解 PF 不连续的 2-目标优化问题时与 NSGA-II 性能相当；同时，在 DTLZ5 测试用例上的实验结果表明了其可在高维空间中有效收敛到低维 PF.



(a) NSGA-II (b) DEmo-DIRECT

Fig. 9 PF comparison between NSGA-II and DEmo-DIRECT on DTLZ5

图 9 NSGA-II 与 DEmo-DIRECT 在 DTLZ5 上的 PF 对比

Table 1 Performance Metrics Comparison Between NSGA-II and DEmo-DIRECT (Ours)

表 1 NSGA-II 和 DEmo-DIRECT（本文）的性能指标对比

测试用例	GD		IGD		HV	
	NSGA-II	DEmo-DIRECT	NSGA-II	DEmo-DIRECT	NSGA-II	DEmo-DIRECT
ZDT1	5.147E−3(2.4E−8)	**3.829E−3(-)**	2.237E−3(2.8E−8)	**1.777E−3(-)**	5.940E−1(9.4E−9)	**6.642E−1(-)**
ZDT2	4.203E−3(1.6E−8)	**3.875E−3(-)**	2.515E−3(2.5E−8)	**1.744E−3(-)**	2.584E−1(2.5E−9)	**3.309E−1(-)**
ZDT3	6.198E−3(3.0E−8)	**3.847E−3(-)**	**2.823E−3(2.8E−8)**	3.894E−3(-)	**1.043E+0(1.7E−9)**	1.042E−0(-)
ZDT6	8.233E−3(3.7E−9)	**3.443E−3(-)**	2.777E−3(1.5E−8)	**1.095E−3(-)**	1.681E−1(9.6E−9)	**3.229E−1(-)**
DTLZ5	4.093E−3(3.4E−9)	**3.861E−3(-)**	2.726E−3(7.0E−9)	**2.126E−3(-)**	1.983E−1(1.2E−9)	**2.213E−1(-)**

注：括号外的数据表示均值，括号内的数据为方差，粗体显示的数据为 2 种算法得到的最优结果. 由于 DEmo-DIRECT 为确定性算法，各指标方差为（-）.

5.3.2 pDEmo-DIRECT 有效性分析

基于 5.1 节的测试用例，我们利用数值实验验证算法并行对 DEmo-DIRECT 的优化性能带来的影响. 由于子问题内并行化并没有对原算法进行本质的改动，只加快了潜在最优超矩形的分割速度，对解集的质量和 PF 并无影响，所以只需要分析不同子问题分组数量（即进程数）对算法优化能力的影响. 图 10 给出了不同并行规模下 pDEmo-DIRECT 的 HV 值，其中 n_sp表示进程数，$n_sp=1$时算法与串行算法的结果完全一致.

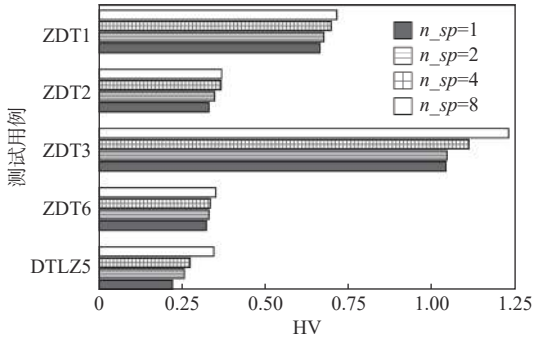


Fig. 10 HV metric values of pDEmo-DIRECT at different parallel scales

图 10 不同并行规模下 pDEmo-DIRECT 的 HV 指标值

由图 10 可以看出，在所有测试用例下，进程数的增多使得 HV 指标值存在不同程度的提升. 可能的原因是，pDEmo-DIRECT 在并行方案中设计了自适应关联迁移策略自适应地为相邻进程进行个体替换，更好地维护了各进程中解的多样性，使得 pDEmo-DIRECT 所得 PF 对真实 PF 的近似精度相对于串行算法有了一定程度的提升. 为了验证上述猜想，我们给出了未使用自适应关联迁移策略的 pDEmo-DIRECT 的 HV 值，如图 11 所示.

由图 11 可以看出，在所有测试用例下，相同并行规模的 pDEmo-DIRECT 的 HV 指标值皆存在一定程度的下降，甚至都小于串行算法所得的 HV 值. 可能的原因是，未加入自适应关联迁移策略的 pDEmo-DIRECT 将原问题产生的不同子问题分配到不同的计算节点上进行计算，相比于串行算法中所有子问题同时优化而言，单个计算节点内子问题数量较少，且仅以预先设定好的频率将一定数目的优秀个体（潜在最优超矩形）与相邻计算节点进行交换，搜索空间变窄且规避了不同的解引导转移搜索方向的情况，使得部分解空间无法得到充分的探索，易导致陷

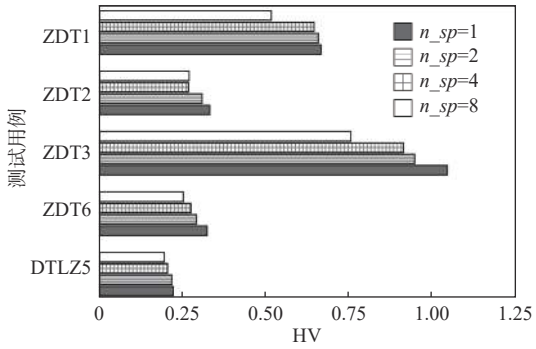


Fig. 11 HV metric values of pDEmo-DIRECT without adaptive association migration strategy

图 11 未使用自适应关联迁移策略的 pDEmo-DIRECT 的 HV 指标值

入局部最优、PF 断裂等问题, 从而降低了多样性和收敛性, HV 值较串行算法有所降低.

结合图 10 和图 11 的结果可知, 采用本文提出的自适应关联迁移策略, 并行算法得到的 PF 在分布、延展以及逼近性能上都有所提升, 未出现普遍存在于并行多目标优化算法中的帕累托解集分布不均匀、难以完整覆盖真实 PF 的情况.

5.3.3 pDEmo-DIRECT 加速效果分析

本节基于 ZDT1 和 ZDT2 测试用例, 计算 pDEmo-DIRECT 算法在不同并行规模下的加速比, 结果如图 12 所示, 其中每个进程创建 20 个线程.

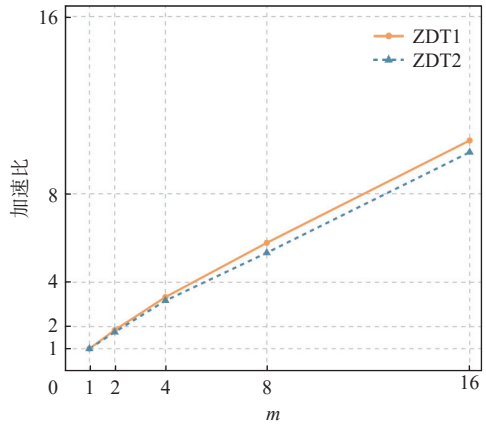


Fig. 12 Speedup of pDEmo-DIRECT under different parallel scales

图 12 pDEmo-DIRECT 在不同并行规模下的加速比

从图 12 可以看出, 随着进程数m的增加, pDEmo-DIRECT 在 2 个测试用例上的加速比均呈快速上升趋势, 但与理论加速比相比有一定差距. 一是因为每一个进程的计算任务都是与原问题性质相似的子问题组, 运行时间取决于耗时最长的子问题组的求解时间, 这种负载不均衡会在一定程度上造成加速比的下降. 二是因为, 为了保证进程内解的多样性, 进程间会基于 *DN* 指标计算结果不定时地进行个体迁移, 虽然相较每代迁移避免了不必要的通信开销, 但通信量仍然保持在相对较高的水平, 使得加速比进一步下降. 此外, 在相同并行规模时, 相比 ZDT2 测试用例, pDEmo-DIRECT 在 ZDT1 测试用例上的加速比更高. 这可能是因为 ZDT1 测试用例问题结构相对较简单, PF 较为平滑且更接近于凸形, 在相同并行规模下, 维持较好多样性时需要的通信开销通常更低.

总而言之, pDEmo-DIRECT 在测试用例上表现出了较为明显的加速效果, 但在可扩展性方面还存在一定提升空间, 特别是需要解决负载均衡和通信开销的挑战. 这些挑战对于进一步提升 pDEmo-DIRECT 在大规模 MOPs 求解中的性能至关重要, 因此需要在未来研究工作中重点关注.

6 结 论

本文基于目标分解技术提出一种全局确定性多目标优化算法, 并充分利用其并行化来求解复杂的 MOPs. 通过分解与目标函数聚合, 将一个 MOP 分解为若干子问题, 从而降低了问题求解的难度. 不同于常用的分解方法, 为了更好地捕捉凸形、凹形或陡峭等复杂前沿, 本文提出了一种归一分解方法, 使算法在保持多样性的同时可更快更好地收敛到 PF. 对于分解后的子问题, 采用 Dividing Rectangles 算法同时优化, 并在该过程中提出了一种全局关联机制, 以更好地保留优秀候选解, 提升算法搜索效率. 此外, 为了进一步缩短问题求解时间, 本文采用多层次多粒度并行的思想, 将所提出算法进行了并行化, 并提出了一种自适应关联迁移策略, 以解决因单个计算节点内子问题数量较少而易陷入局部最优的问题. 将所提方法应用于 ZDT 和 DTLZ 问题, 并与典型的进化算法 NSGA-II 比较. 实验结果表明, 在几乎所有测试用例上, 所提串行算法的性能皆优于 NSGA-II; 此外, 还验证了自适应关联迁移策略可有效提升算法的 PF 前沿近似精度, 并行算法可大规模缩短问题求解时间, 有望为大规模 MOPs 的快速求解提供计算支撑.

在本文的实验中, 仅通过典型基准优化问题来验证了本文方法的有效性, 还没有将其应用于实际的优化问题中. 因此, 在未来的工作中将进一步验证其在现实 MOPs 中的性能表现. 此外, 针对并行算法中存在的负载均衡问题, 可考虑基于子问题组新产

生的超矩形的数量等因素，提出新的负载均衡策略，从而进一步提升并行算法效率和可扩展性.

作者贡献声明：李晨提出了算法思路和实验方案，并负责程序实现、数据分析、论文撰写与修订；陈逸东负责实验结果可视化与论文修订；陆忠华负责研究指导、论文审阅与修订；杨雪莹负责实验结果可视化；王子田负责实验方案优化与改进；迟学斌负责研究指导、论文审阅.

参 考 文 献

[1] Wong C S Y, Al-Dujaili A, Sundaram S. Hypervolume-based direct for multi-objective optimization [C] //Proc of the 16th Genetic and Evolutionary Computation Conf Companion. New York: ACM, 2016: 1201−1208

[2] Tian Ye. Research on several key problems of many-objective optimization algorithms [D]. Hefei: Anhui University, 2015 (in Chinese)
(田野. 高维多目标优化算法的若干关键问题研究 [D]. 合肥: 安徽大学, 2015)

[3] Zelinka I. A survey on evolutionary algorithms dynamics and its complexity−Mutual relations, past, present and future[J]. Swarm and Evolutionary Computation, 2015, 25: 2−14

[4] Hernandez Gomez R, Coello Coello C A, Alba E. A parallel version of SMS-EMOA for many-objective optimization problems [C] // Proc of the 16th Genetic and Evolutionary Computation Conf Companion. New York: ACM, 2016: 568−577

[5] Deb K, Pratap A, Agarwal S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II[J]. IEEE Transactions on Evolutionary Computation, 2002, 6(2): 182−197

[6] Zhang Qingfu, Li Hui. MOEA/D: A multiobjective evolutionary algorithm based on decomposition[J]. IEEE Transactions on Evolutionary Computation, 2007, 11(6): 712−731

[7] Deb K, Jain H. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints[J]. IEEE Transactions on Evolutionary Computation, 2013, 18(4): 577−601

[8] Mashwani W K. MOEA/D with DE and PSO: MOEA/D-DE+ PSO [C] //Proc of the 31st SGAI Int Conf on Innovative Techniques and Applications of Artificial Intelligence. Berlin: Springer, 2011: 217−221

[9] Jones D R, Perttunen C D, Stuckman B E. Lipschitzian optimization without the Lipschitz constant[J]. Journal of Optimization Theory and Applications, 1993, 79(1): 157−181

[10] Jones D R, Martins J R. The DIRECT algorithm: 25 years later[J]. Journal of Global Optimization, 2021, 79((3):): 521−566

[11] Zitzler E, Thiele L. Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach[J]. IEEE Transactions on Evolutionary Computation, 1999, 3(4): 257−271

[12] Tang Weisen. Research on hypervolume based evolutionary multi-objective optimization algorithms [D]. Guangzhou: Guangdong University of Technology, 2018 (in Chinese)
(唐炜森. 基于 hypervolume 的多目标优化算法研究 [D]. 广州: 广东工业大学, 2018)

[13] Liu Yao, Zheng Lin, Zheng Kai, et al. Parallelization and optimization of NSGA-Ⅱ based on Sunway many-core processor[J]. Application Research of Computers, 2020, 37(1): 96−101 (in Chinese)
(刘垚, 郑琳, 郑凯, 等. 基于申威众核处理器的 NSGA-Ⅱ并行和优化方法[J]. 计算机应用研究, 2020, 37(1): 96−101)

[14] Zitzler E, Laumanns M, Thiele L. SPEA2: Improving the strength Pareto evolutionary algorithm[J]. Technical Report Gloriastrasse, 2001, 103(1): 1−21

[15] Zhan Zhihui, Li Jingjing, Cao Jiannong, et al. Multiple populations for multiple objectives: A coevolutionary technique for solving multiobjective optimization problems[J]. IEEE Transactions on Cybernetics, 2013, 43(2): 445−463

[16] Liu Hailin, Gu Fangqing, Zhang Qingfu. Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems[J]. IEEE Transactions on Evolutionary Computation, 2014, 18(3): 450−455

[17] Cheng Ran, Jin Yaochu, Olhofer M, et al. A reference vector guided evolutionary algorithm for many-objective optimization[J]. IEEE Transactions on Evolutionary Computation, 2016, 20(5): 773−791

[18] Zitzler E, Künzli S. Indicator-based selection in multiobjective search [C] //Proc of the 8th Int Conf on Parallel Problem Solving from Nature-PPSN VIII. Berlin: Springer, 2004: 832−842

[19] Bader J, Zitzler E. HypE: An algorithm for fast hypervolume-based many-objective optimization[J]. Evolutionary Computation, 2011, 19(1): 45−76

[20] Falcón-Cardona J G, Coello Coello A C. A new indicator-based many-objective ant colony optimizer for continuous search spaces[J]. Swarm Intelligence, 2017, 11(1): 71−100

[21] Shubert B O. A sequential method seeking the global maximum of a function [J]. SIAM Journal on Numerical Analysis, 1972, 9(3): 379−388

[22] Wang Luyi, Ishida H, Hiroyasu T, et al. Examination of multiobjective optimization method for global search using DIRECT and GA [C] //Proc of the 2008 IEEE Congress on Evolutionary Computation. Piscataway, NJ: IEEE, 2008: 2446−2451

[23] Al-Dujaili A, Suresh S. Dividing rectangles attack multi-objective optimization [C] //Proc of the 2016 IEEE Congress on Evolutionary Computation. Piscataway, NJ: IEEE, 2016: 3606−3613

[24] Campana E F, Diez M, Liuzzi G S, et al. A multi-objective direct algorithm for ship hull optimization[J]. Computational Optimization and Applications, 2018, 71(1): 53−72

[25] Houssein E H, Mahdy M A, Shebl D, et al. An efficient slime mould algorithm for solving multi-objective optimization problems[J]. Expert Systems with Application, 2022, 187: 115870

[26] Miettinen K. Nonlinear Multiobjective Optimization [M]. New York: Springer, 2012

[27] Wan Shasha, Pei Li, Wang Xikui, et al. Design of PMF based on FBG delay line using DIRECT algorithm[J]. Electrooptic Technology Application, 2008, 23(6): 41−44 (in Chinese)
(万沙沙, 裴丽, 王喜奎, 等. 利用 DIRECT 算法设计基于 FBG 延迟

线的微波光子滤波器[J]. 光电技术应用, 2008, 23(6): 41−44)

[28] He Hongwen, Huang Xianguang, Zhang Yaming, et al. Study on parameters optimization of hybrid electric vehicle based on DIRECT algorithm[J]. Chinese Journal of Automotive Engineering, 2011, 1(1): 35−41 (in Chinese)
(何洪文, 黄贤广, 张亚明, 等. 基于 DIRECT 算法的混合动力汽车参数优化研究[J]. 汽车工程学报, 2011, 1(1): 35−41)

[29] Mohammadi A, Omidvar M N, Li Xiaodong. Reference point based multi-objective optimization through decomposition [C] //Proc of the 2012 IEEE Congress on Evolutionary Computation. Piscataway, NJ: IEEE, 2012: 1150−1157

[30] Wei Jianle. Design and application of multi-objective evolutionary algorithm based on cooperative decomposition [D]. Guangzhou: Guangzhou University, 2022 (in Chinese)
(魏建乐. 基于协同分解的多目标进化算法设计与应用 [D]. 广州: 广州大学, 2022)

[31] Wang Liping, Wu Feng, Zhang Mengzi, et al. Decomposition multi-objective evolutionary algorithm based on differentiated neighborhood strategy[J]. Pattern Recognition and Artificial Intelligence, 2017, 30(12): 1069−1082 (in Chinese)
(王丽萍, 吴峰, 张梦紫, 等. 基于差异化邻域策略的分解多目标进化算法[J]. 模式识别与人工智能, 2017, 30(12): 1069−1082)

[32] Zitzler E, Deb K, Thiele L. Comparison of multiobjective evolutionary algorithms: Empirical results[J]. Evolutionary Computation, 2000, 8(2): 173−195

[33] Li Lu. Research on parallel multi-objective evolutionary algorithm based on bi-level decomposition [D]. Xi'an: Xidian University, 2020 (in Chinese)
(李璐. 基于双层分解的并行多目标演化算法研究 [D]. 西安: 西安电子科技大学, 2020)

[34] Deb K, Thiele L, Laumanns M, et al. Scalable Test Problems for Evolutionary Multiobjective Optimization [M]. Berlin: Springer, 2005

[35] Ge Mengyao. Application of artificial bee colony algorithm in multi-objective fuzzy portfolio optimization [D]. Beijing: Capital University of Economics and Business, 2016 (in Chinese)
(葛梦瑶. 人工蜂群算法在多目标模糊投资组合优化中的应用 [D]. 北京: 首都经济贸易大学, 2016)

[36] Van Veldhuizen D A. Multiobjective evolutionary algorithms: Classifications, analyses, and new innovations [D]. Dayton, OH: Air Force Institute of Technology, 1999

[37] Coello Coello C A, Reyes Sierra M. A study of the parallelization of a coevolutionary multi-objective evolutionary algorithm [C] //Proc of the 3rd Mexican Int Conf on Artificial Intelligence. Berlin: Springer, 2004: 688−697

[38] Li Chen, Wu Yulei, Lu Zhonghua, et al. A multiperiod multiobjective portfolio selection model with fuzzy random returns for large scale securities data[J]. IEEE Transactions on Fuzzy Systems, 2021, 29(1): 59−74



Li Chen, born in 1993. PhD, assistant professor. Member of CCF. Her main research interests include high performance computing, multi-objective optimization, and fuzzy computing.
李 晨, 1993 年生. 博士, 助理研究员. CCF 会员. 主要研究方向为高性能计算、多目标优化、模糊计算.



Chen Yidong, born in 1996. PhD candidate. His main research interest includes high performance computing technology in the computational finance. (chenyidong@cnic.cn)
陈逸东, 1996 年生. 博士研究生. 主要研究方向为高性能计算技术在计算金融中的应用.



Lu Zhonghua, born in 1965. PhD, professor, PhD supervisor. Senior member of CCF. Her main research interest includes high performance computing and its application. (zhlu@cnic.cn)
陆忠华, 1965 年生. 博士, 研究员, 博士生导师. CCF 高级会员. 主要研究方向为高性能计算及其应用.



Yang Xueying, born in 1997. PhD candidate. Her main research interest includes high performance computing technology in the computational finance. (yangxueying@cnic.cn)
杨雪莹, 1997 年生. 博士研究生. 主要研究方向为高性能计算技术在计算金融中的应用.



Wang Zitian, born in 1982. PhD. His main research interests include high performance computing and multi-objective optimization. (wzt@aihpc.cn)
王子田, 1982 年生. 博士. 主要研究方向为高性能计算、多目标优化.



Chi Xuebin, born in 1963. PhD, professor, PhD supervisor. Distinguished member of CCF. His main research interests include high performance computing and grid technology. (chi@sccas.cn)
迟学斌, 1963 年生. 博士, 研究员, 博士生导师. CCF 杰出会员. 主要研究方向为高性能计算、网格技术.