大规模合作进化中的高效资源分配全球优化

杨明，Mohammad Nabi Omidvar，*IEEE会员*，李长河，*IEEE会员*，李晓东，*IEEE高级会员*，蔡志华，Borhan Kazimipour，*IEEE学生会员*，姚欣，*IEEE院士*

Abstrncr-Cooperative co-evolution（CC）是多种群进化算法中解决大规模优化问题的一种明确的问题分解手段。对于CC来说，代表大规模优化问题的子种群共同进化，并且可能对问题的最佳整体解决方案的改进有不同的贡献。因此，将更多的计算资源分配给有更大贡献的子种群是有意义的。在本文中，我们研究了如何在这种情况下分配计算资源，并随后提出了一个新的CC框架，即CCFR，以根据子群体对改善最佳整体解决方案的目标值的动态贡献，在子群体中有效地分配计算资源。我们的实验结果表明，CCFR能够有效地利用计算资源，是解决大规模优化问题的一种极具竞争力的CCFR。

Index Terms-Cooperative co-**evolution** (CC), large-wide optimization, problem decomposition, resource allocation.

2015年9月28日收到稿件；2016年1月5日修订、2016年4月11日、2016年8月14日和2016年10月26日；11月接受2, 2016.发表日期2016年12月15日；当前版本日期2017年7月27日。该工作部分得到国家自然科学基金61305086、61673355、61673354、61329302和61305079的支持，部分得到EPSRC的EP/K001523/1资助和智能地理信息处理湖北省重点实验室开放研究项目KLIGIP201602的资助。*(通讯作者：姚欣)*

M.Yang和Z. Car在中国地质大学计算机科学学院，武汉，430074；也在中国地质大学智能地理信息处理湖北重点实验室，武汉，430074（电子邮件：yangming0702 Hgmai1.com；zhcai Hcug.edu.en）。

M.N. Omidvar是伯明翰大学计算机科学学院计算智能和应用卓越研究中心，伯明翰B15 2TT，英国（电子邮件：m.omidvarHcs.bham.ac.uk）。

C.Li在中国地质大学自动化学院，武汉430074（电子邮件：changhe.lw Hgmail.com）。

X.Li和B. Kazimipour在澳大利亚墨尔本皇家理工大学计算机科学和信息技术学院（电子邮件：xiaodong.1i Hrmit.edu.au; borhan.kazimipourHrmit.edu.au）。

X.姚明是英国伯明翰大学计算机科学学院计算智能与应用卓越研究中心，伯明翰B15 2TT，同时也是南方科技大学计算机科学与工程系，深圳518055，中国（电子邮件：x.yao Pcs.bham.ac.uk）。

本文有补充的可下载的多媒体材料在http://ieeexplore.ieee.org，由作者提供。

本文中一个或多个数字的彩色版本可在网上查询：http://ieeexplore.ieee.org。

数字对象标识符10.1109/TEVC.2016.2627581

I. 简介

在解决许多优化问题上，自由算法（EA）已经取得了巨大的成功[1]。然而，随着问题维度的增加，它们往往会失去其功效[2]。许多现实世界的问题涉及大量的决策变量，例如，在机翼的设计中，需要成千上万的变量来代表飞机机翼的复杂形状[3]。这类大规模的优化问题给我们带来了严重的挑战。现有EA。

解决高维优化问题的一个自然方法是采用*分而治之*的策略[4]-[6]，该策略将一个大规模的优化问题分解为一组较小和较简单的子问题。这些子问题可以被单独解决。完全可分离的大规模优化问题，在决策变量之间没有相互依赖，可以通过独立优化每个变量来解决[7]。在光谱的另一端，完全不可分离的大规模优化问题，即任何一对变量之间存在相互依赖，需要通过对所有变量进行优化来解决。然而，大多数现实世界的问题介于这两个极端之间，即只有一些变量是独立的或相互依赖的[8]。对于这种泛在可分离的问题，通常有几个相互依赖的变量群。合作协同进化（CC）[7] 是EA 中问题分解的一种明确手段。对于CC，有一组子种群，每个子种群负责优化一个变量子集（即一个子组件）。

在一个固定的计算预算下，CC 可能受到计算资源在子种群之间分配方式的影响[9]。对于CC 来说，不同的子种群可能对改善最佳总体目标值（即由这些子种群的最佳个体组成的最佳总体解决方案的目标值）做出不同程度的贡献。为了提高计算效率，应将更多的计算资源分配给贡献较大的子种群。在[9] 中显示，对于不平衡的问题，不同的子种群对总体目标值的贡献是不平等的，基于贡献的

合作共进化（CBCC）的表现优于传统的
CC。然而，对于CBCC来说，贡献信息是从进化过程的一开始就被累积的。CBCC在进化过程的早期阶段非常依赖贡献信息，因此它可能对总体目标值的局部变化反应太慢甚至不正确。由于子种群的贡献可能会随着时间的推移而变化，所以资源分配应该实时地适应性地完成是有意义的。

在本文中，我们研究了如何在子种群之间分配计算资源，并提出了一个新的CC框架（CCFR），它可以根据每个子种群对提高最佳总体目标值的动态贡献，自适应地分配计算资源。这个新的CCFR在以下两个方面与现有的CCFR不同。

1) 这个新的CCFR可以检查一个子种群是否停滞不前。为了节省计算资源，停滞的子种群被排除在进化之外（见第三部分A）。
2) 在这个新的CCFR中，一个亚种群的贡献是动态更新的。在每个周期中，只有贡献最大的亚群被选中进行进化（见第三部分B）。

本文的其余部分组织如下。第二节介绍了CC的概况。第三节介绍了我们新的CCFR。第四节介绍了实验研究。最后，第五节提供了结论意见。

II. 相关的工作

在进化计算的文献中，一个问题的决策变量之间的相互依存关系被称为*联系*[10]或*外显关系*[11]。CC算法的性能在很大程度上受到变量之间相互依赖的影响[7], [12]。变量分组方法旨在将相互依赖的变量分组到同一被优化的子组件中，在克服这种问题方面发挥了关键作用[13]。在[14]中表明，如果所有的子组件都是可分离的，那么原始问题的整体解决方案就是所有子问题的各自解决方案的组合。在这里，我们主要在大规模优化的背景下回顾CC。

在Potter和Jong提出的原始CC遗传算法（CCGA）中[7]，一个D维问题被分解成D个1维的子问题。然后，CCGA使用进化优化器以轮流的方式解决这些子问题。[7]中的实验结果表明，原始的CC在不可分离的函数，即有相互依赖的变量的函数，如*Griewank*和*Rosenbrock*上不能表现良好。Liu *等人*[2]将CC应用于快速进化编程，以解决高达1000D的大规模优化问题。van den Bergh和Engelbrecht[15]将CC应用于粒子群优化（PSO）[16]，并提出一种CC PSO算法，即CPSO，它将一个D维问题分成k个5维的子问题，其中s 4/ D。Shi *等人*[17]将微分进化（DE）[18]引入CC，将决策变量分成两个

等大小的子组件。显然，这种分解策略在维度很高的问题上表现不佳。

Yang等人[13]提出了一种随机变量分组方法并将其应用于CC。与CPSO依靠从优化开始到结束的固定变量分组不同，Yang等人[13]提出的随机分组方法在每个协同进化循环中随机地将所有决策变量洗成k个s维的子组件。在[13]中显示，这种随机分组策略能有效地将两个相互依赖的变量分组到一个子组件中，并持续几个周期。采用这种随机分组策略的DE算法，即DECC-G，在一组多达1000D的大规模优化问题上表现良好[13]。

前面提到的分组策略使用预先指定的、固定的子组件大小进行分解。因此，在使用这些分解策略之前，用户需要指定一个k或s的值，这在实践中可能很困难。此外，CC的性能可能高度依赖于这些指定值。

调整子组件的大小可以潜在地提高CC的性能[19]。Yang 等人[20]提出了一种多级CC（MLCC）算法。MLCC使用一组可能的s值进行分解，而不是使用固定的子组件大小。优化过程中使用的每个子组件大小的性能是根据最佳总体目标值的改善来衡量的。具有更好性能的子组件大小将在下一个共同进化周期中以更高的概率被选择。Li和Yao[22]用改进的随机变量分组策略进一步加强了CCPSO算法[21]，提出了CCPSO2来解决一组多达2000D的大规模优化问题。

当相互依赖的变量数量超过5个时，随机分组是无效的[19]。文献[23]表明，在CEC2010的大多数基准函数上，一种非随机方法，即delta分组，优于随机分组[24]。德尔塔分组法使用优化过程中某个变量的平均差值来检测相互依赖的变量。具有相似差值的变量被认为是可能的相互依赖的变量。然而，这个假设可能并不总是成立。例如，当有一个以上的子组件时，delta分组方法不能很好地执行[23]。

如[25]所建议的，一个给定的问题可以以自动的方式进行分解，而不必事先知道其基本结构。在共同进化过程的开始阶段，所有的变量都由不同的子群体单独优化。在[25]中使用了一个计数器来计算将两个变量组合在一起的可能性。如果随机选择的个体中的两个变量能够进一步改善最佳个体，计数器就会增加。在每个共同进化周期结束时，具有最大计数器的两个变量被组合在一起。与这两个变量相对应的子种群被合并为一个子种群。Chen ei 等人[26]提出的CC与变量交互学习（CCVIL）算法采用了一个两阶段的方法。

在第一阶段，CCVIL按照[25]的做法检测变量之间的相互作用，以完成分解。在第二阶段，CCVIL以传统CC[7]的方式对这些分解的组进行优化。

Tezuka等人[27]提出了通过非线性检查的联动识别，用于实际编码的GA（LINC-R）。如果关于一个变量的函数值的差异独立于关于另一个变量的函数值的差异，那么这两个变量是可分离的。Omidvar等人[28]对LINC-R进行了理论研究，并提出了一种检测相互依赖变量的新方法，即差分分组（DG）。DG能够以较高的精度识别相互依赖的变量。它表明
在[28]中，带有DG的CC在一组多达1000D的大规模优化问题上表现良好。

对于可分离的决策变量，在[29]中显示，单独优化每个变量可能不是解决大规模优化问题的最佳方式。一个更有效的方法是将可分离的变量分成几个
组。然而，要确定最佳的小组规模可能很困难。

在处理部分可分离的问题时，不同的子群体对总体目标值的提高的贡献可能是不平衡的。经典CC中的轮流策略在处理这类问题时不再有效，因为它为每个子群体分配了等量的计算资源，而没有考虑子群体的不平等贡献。为了克服这个问题，在[9]中提出了一个CBCC，根据各子种群对提高最佳总体目标值的贡献，在各子种群之间分配计算资源。CBCC强调在进化过程的早期阶段的贡献。因此，它可能将大部分计算资源分配给最初贡献较大但在几代之后就会下降的子种群。对于CBCC的两个变体（CBCC1和CBCC2），[30]中的实验结果表明，CBCC1对分解精度和子种群贡献之间的不平衡的敏感性比CBCC2低得多。CBCC1和CBCC2无法在优化过程中对子种群的动态贡献作出适应性反应。

III. 拟议的《公约》框架

本节将介绍一种新的CCFR。CCFR旨在根据子种群对提高最佳总体目标值的动态贡献，在子种群之间智能地分配计算资源。请注意，CCFR采用了与DECC-DG[28]相似的两阶段方法。在第一阶段，使用分解方法形成分解；在第二阶段，在分解保持固定的情况下，分别对结果组进行优化。

假设*i*表示解体后的第i个子种群。对于第G代的Ci所对应的亚种群，为了检查该亚种群是否停滞不前，可以计算出个体在维度j（j e *Ci）的*基因值的平均值和标准差：

$$m_{j,G} = \frac{1}{N}\sum_{t=1}^{N} x_{t,j,G} \tag{1}$$

$$\text{标准；} \quad \sqrt{\frac{1}{N}\sum_{t=1}^{N}(x_{t,j,G} - m_{j,G})^2} \tag{2}$$

A. 节省对停滞的子群的计算

CC使子种群以循环的方式使用进化运算器进行进化。对于那些属于以下情况的子组件

其中*N*是子种群的大小，x',;,G是个体-*t*的第y个基因值，*G* -',G - -*t, I,G* 。, *x ,D,GB* 如果一个种群的分布，即个体基因值在维度j中的平均值和标准差，在连续几代中保持不变，那么这个种群就被认为在这个维度上是停滞的[31]。基于这一策略，我们提出以下方法来检查一个子种群是否在所有维度上都是停滞的：

$$Qj = \begin{cases} 1，如果 m_{;,G} - z\text{-} 和 \\ stdj,G \text{ -- } stdj\ p\text{-}t & (3a) \\ 0 \quad 否则 & (3b) \end{cases}$$

其中8，G表示$m_{;,G}$和std;,G的值是否与上一代在维度y中保持不变，注意$,o = 0。yq表示维度数，其中Qj = 1

$$\gamma_G = \sum_{j \in C_i} \beta_{j,G}. \tag{4}$$

如果子种群没有变化（即没有更好的个体产生），=*D-i*，其中*Di*是子成分*C-i*的维度，*VG*表示连续几代的数量，其中*rG-Di*

$$\begin{cases} rG\text{-}1 + 1 \text{ 如果} 3y\text{-}Di & (5a) \\ 0 \quad 否则 & (5b) \end{cases}$$

并注意9=0。 *pz*是一个标志，表示子种群在第G代是否停滞，*PG*的值计算如下：

$$\begin{cases} 1 \quad 如果 i;p > U & (6a) \\ 否则为0 & (6b) \end{cases}$$

其中*U*是一个整数，其值等于*D*；。我们的经验-心理结果表明，子成分的大小越大，其相应的子种群进入停滞状态所需的代数就越多。根据对*U*的敏感性研究（在附录中所列的补充材料的第一节中提供），我们使用*U--Di*--如果一个子种群的分布

子种群在几个连续的遗传中保持不变（即*eG " U*），*PG* 被设置为1，表示
子种群很可能会停止进化。

一些现有的方法认为一个种群是滞后的。如果最佳健身值[32], [33]或个体之间的差异[34], [35]的改进非常小，即使种群仍然缓慢地收敛到一个最佳状态，那么就会出现停滞。Guo *等人*[36],[37]认为，当个体的适配度在连续几代中不能得到改善时，该个体就是停滞的。这种方法对于具有高原健身景观的问题（如*阶梯*函数[38]）是无效的，即个体的健身值不发生变化，而个体的决策变量值发生变化。Yang *等人*[39]认为，当个体之间的平均距离在连续几代中保持不变时，种群是滞后的。然而，整个种群的分布有可能发生变化（例如，所有的个体都以相同的位移变化）。在这种情况下，杨氏方法可能会错误地将种群归类为停滞的种群。与上述停滞检测方法相比，我们提出的方法根据个体基因值的平均值和标准差更准确地识别- ing一个停滞的群体。

对于*po* 1的亚种群，我们将其排除在共同进化周期之外，这意味着停滞的亚种群不会在随后的共同进化周期中发生进化。

B. 基于贡献的资源分配

概率匹配算法[40]和自适应追求算法[40]学习运营商之间的最佳资源分配。这些基于概率的方法会以最小的概率将资源分配给无效的运营商。基于置信度上限算法[41]，Li ei *al.* [42]提出了一种在操作者之间分配资源的方法，其中具有最大相对适配度改进的操作者被选择参加进化过程[43], [44]。这些基于相对适应性改进的方法将资源分配给绝对适应性改进非常小但相对适应性改进相对较大的项目（如收敛项目）。在[45]中，平均绝对适配度的提高被用于确定资源分配。De Rainville *等人*[46]提出了一种基于二元奖励的CC资源分配。如果总体目标值变得更好，一个子群被分配到1的奖励，否则就是0。然而，二进制奖励不能反映出目标值改善的真实幅度。在本节中，我们提出了一种基于最佳总体目标值的绝对改进的CC的资源分配策略。与[45]中的平均绝对改善最佳总体目标值

$$Fi\cdot \frac{OF、\ \mathtt{f(bes)}\ \mathtt{(*bes)}}{2} \tag{7}$$

改进不同，我们提出的方法在资源分配上更多地考虑整体目标值的最近改进。

对于一个子种群i *)'*，当*i*在一个周期内完成演化时，我们根据以下方法计算其贡献

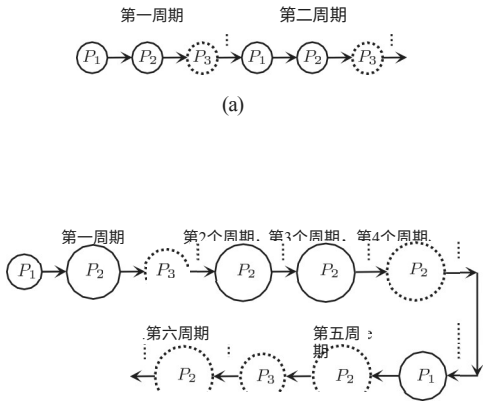
其中，f(*"best)和/(best)轮胎分别是*Pi*在本周期内进行进化之前和之后的最佳总体目标值，*fYFi*是-*i*的最后贡献，*kFi*的初始值为零。方程（7）通过平均最后的贡献（即 *i*）和当前的贡献[即 |/(best) (best)]来平滑地更新*i*，以提高最佳总体目标值。较近的贡献|*f*(best) f(Xbest)|是，影响越大的是!/(best) / (<best) |对A *i*的值是。随着共同进化的进行，早期贡献对A *i*的影响变得越来越小。

在第一个共同进化周期中，各子种群逐一进行进化。所有子种群的*OFI*值在第一个周期结束时被计算出来。在随后的共同进化循环中，我们只选择具有最大的*2^YFi*值的子种群进行进化。在每个共同进化周期结束时，*OF*的值根据（7）进行更新。*OFi*的值越大，*Pi*在未来经历进化的机会就越大。如果根据（6），一个子种群是停滞的，我们把它的贡献 *(2^YFi)* 设为零。因此，这个停滞的子种群将被排除在随后的共同进化循环之外。当*2^YFi*的值对所有子种群都相同时，我们从第一个共同进化周期重新开始这个过程。这样做的好处是，被误认为是停滞的子种群可以恢复其进化。上述过程重复进行，直到出现终止
符合标准。

CBCC[9]也可以根据子种群对提高最佳总体目标值的贡献，在子种群之间分配计算资源。CCFR和CBCC之间的重要区别是，CCFR比CBCC更快地响应整体目标值的最新变化。对于CCFR来说，贡献是通过对最近和当前的贡献进行平均而顺利更新的，而对于CBCC来说，贡献是从进化过程的一开始就积累的。此外，CBCC不考虑**停滞的**子种群。

图1说明了传统CC[7]、CBCC2[9]（CBCC的一个变体）和CCFR中的计算资源分配。传统CC中的循环方式在所有子种群之间平均分配计算资源，而不考虑子种群的不同贡献[见图1(a)]。传统的CC总是将计算资源分配给停滞不前的子种群[如图1(a)中的P3]，这显然是一种浪费。对于CBCC，每个子种群的贡献从进化过程的一开始就被积累起来，如图1（b）所示、

其中，不同的圆圈大小意味着不同的子群贡献量。CBCC2将大部分计算资源分配给累积贡献最大的子种群。在第二和第三周期，CBCC2选择具有最大累积贡献的子种群Pz来进行进化。从第二个周期开始、



(a)

(b)

(c)

图1.CC、CBCC2和CCFR中的计算资源分配，其中圆圈大小表示算法计算的贡献量，虚线圆圈表示子种群停滞不前。 (a) 传统CC。(b) CBCC2。(c) CCFR。

2在一个周期内的贡献（[illegible]的大小变化）是很小的。即使在Pz一直停滞不前的[illegible]下，CBCC2仍然认为P2的贡献最大，并将计算资源分配给P2[例如，图1（b）中的第六个周期]。CBCC2将计算资源分配给停滞的子种群2和P3。CCFR通过平均每个周期结束时的最后和当前贡献来计算贡献。在图1(c)中，可以看到对于P2来说，随着进化的进行，圆圈的大小变得越来越小。P2在第三个周期的贡献相对较小。CCFR将在*P*和P3之间选择一个亚群，在下一个周期进行进化。虽然3的最后贡献大于
P，CCFR选择*P*在第四周期进行进化。
这是因为P3是停滞的，被排除在循环之外。该图表明，在计算资源相等的情况下，CCFR有可能获得比传统CC和CBCC2更好的解决方案。

C. 获得最佳的整体解决方案

在共同进化循环中，许多CC算法[9]、[13]、[20]、[28]在综合种群水平上更新原始问题的最佳整体解决方案（见算法1的第7步）。以下面的2-D *Sphere*函数为例：

$$f(\mathbf{x}) = x_1 +$$

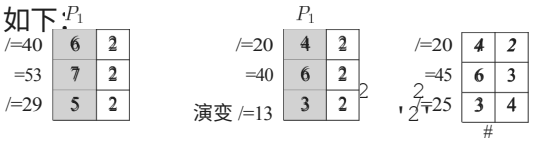
算法1 DECC[28]

/*假设C = *(C* , ... , Cy) 是一个分解，*P* -- {xi 、 , ^*N* I 是一个种群。*/

:**b* - argmin /(x)；

2: 对于*k* =- 1到*循环*，做
3: for i +- 1 to *M* do
4: *P* =-*x j* (*j* *P, t* -- 1, ..., *N,* j e *C,*) -、
5: *Pi* +-Optimizer(xt",t, *Pi, GEs)*、
6: *je , t*-1, ...,*N, j C ;)*+- *P;',*
7? *胸部* argmin /(x)；
8: 结束
9: 结束

其中，当前最好的整体solutiOR >best' (6, 2) iS用黑体和斜体字表示。假设子群i的进化过程（算法1的第4-7步）如下：



这个函数是可加可分的[47]。它的理想解构浓度为*C-C*，C2) (毫升 (<2))

假设某一代的人口*P*如下：

/=40
/=58
/=41

f'

这产生了一个更新的>最好的'（4，2）。假设子种群的进化过程如下：

这同样会产生一个更新的<最佳>（4，1）。当这个共同进化周期结束时，来自不同子种群的个体的所有可能组合是（4，1），（4，0）、(4, 3), (6, 1).(6，0），（6，3），（3，1），（3，0），和（3，3）。每个组合是问题的整体解决方案。在所有的组合，最佳的整体解决方案是（3，0）。对于一个拥有N个种群和M个子种群的种群来说，组合的数量是N^M。我们改进了CCFR中的算法1通过以下方式更新^best。在对应于M个子种群的子组件相互之间可以分离的情况下，通过改进的CCFR获得的>best是N个^M 组合中的最佳整体解决方案。

根据可分离性的定义[24],[47]。

$$\text{argmin } /(\mathbf{x}) = \left(\text{argmin } f(\mathbf{x}_1), \ldots, \text{argmin } /(\ldots \mathbf{x}_M)\right) \quad (8)$$

对于一个具有M个独立子项的可分离函数/(x)，以下方程成立：

$$\text{argmin } f(\mathbf{x}) = \left(\text{argmin } f(\mathbf{x}_1), \ldots, \text{argmin } /(\ldots \mathbf{x}_M)\right) \quad (9)$$

其中Z是来自$P, \ldots$的所有可能组合的集合。方程（9）简单地说明，如果子组件是可分离的，每个子群体的最佳解决方案的组合一定是最佳的整体解决方案。

从Z到原始问题。当分解形成后，我们设定最佳整体解决方案Xbest aS follOWS：

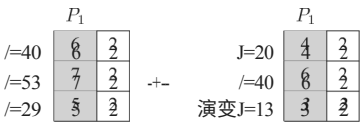
$$\text{Xbest} = \left(\begin{matrix} \text{最小值} \\ \text{-i e i} \end{matrix} f\left(\mathbf{x}_1, \begin{matrix} P_1 \\ \text{,最好} \end{matrix} \right. \right. \left. \left. \begin{matrix} \\ \text{的} \end{matrix} \right) \text{ . , } \underset{*M}{\operatorname{argmin}} f\left(\mathbf{x}_M, \begin{matrix} P_M \\ \text{,最好的} \end{matrix}\right) \right) \tag{10}$$

其中,最佳 <J< 最好的'。*P;*，其中包括最佳移除*Pi*的尺寸。*最好的是一个串联。

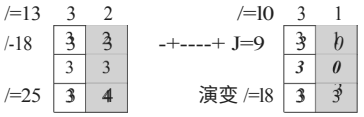
的*M*个子种群中的所有最佳解 （$P_{1'\ldots\ ,\ M)}$。，如[15]中构建的那样。在算法1中，步骤5被改变如下：

$$(Pi' \text{ best}) \text{ +- Optimizer(*best' (GES)}$$

其中Xbest在每个子种群的共同进化过程结束时被更新，而步骤7被删除。上述共同进化的例子变化如下：



这产生了一个更新的Xbest' (39 2)d和



这同样会产生一个更新的Xbest ' (3, 0)。从上面的进化过程可以看出，在进化过程中，Xbest iS总是被更新为最佳的整体解决方案。注意

如果子组件之间存在相互依存关系，通过上述改变后的进化过程获得的*佳方案可能不是最佳的整体解决方案。

D. CCFR

算法2概述了拟议的CCFR。步骤8至17计算每个子种群的贡献（即*Fi*的值）。步骤18至28选择具有最大贡献的子种群进行进化，并在进化结束时更新其贡献。当所有子种群的贡献相等时，CCFR进入步骤8，重新设定每个子种群的贡献。上述过程重复进行，直到满足终止标准。步骤11和22调用子种群的进化过程，如算法3所示。

在算法3中，一个子种群经历了预先指定的代数的进化，即GEs。步骤15至18检查一个子种群是否停滞不前。如果该子种群被确定为停滞不前，CCFR将停止该子种群的进化。在算法3中，当发现一个更好的解决方案时，最佳

算法2 CCFR

1：生成一个分解*C*--{C'j, ..., Cy）；
2：生成*个*均匀的随机种群*P_i*、 , *N I*、
3: 计算*胸部* argmin f(x);
4：用公式（10）设定*Chest*的值；5：
Ali=- 0，*G*；=- 0，i=1，2，...。, *M;*
6：在不符合终止标准的情况下，做
7: 对于每个子群，将p（见公式（5））重置为0
；8： 对于*i* +- 1到*M*做
9- *"最佳 *最好的"*
10: $P_i \leftarrow \{x_{t,j} \mid x_{t,j} \in P, t = 1, \ldots, N, j \in C_i\}$;
lll *I i' *best- PG_i , G;)*+- Optimizer(*heel *P GES G;)*
12: $\{x_{t,j} \mid x_{t,j} \in P, t = 1, \ldots, N, j \in C_i\} \leftarrow P_i$;
13: *斐济* $p F_i + |f(\hat{\mathbf{x}}_{best}) - f(\mathbf{x}_{best})|)/2$;
14: 如果*PZ*.等于1，那么
15: *kF ¡* +- 0；
16: 结束 如果
17: 结束
18: 虽然min(AFili -- 1, ... , *M)* max(AFili -- 1, ... , M) do 19:
i +- 最大*bF*的索引、
20: $\hat{\mathbf{x}}_{best} \leftarrow \mathbf{x}_{best}$;
21: $P_i \leftarrow \{x_{t,j} \mid x_{t,j} \in P, t = 1, \ldots, N, j \in C_i\}$;
221 (i' *best- PG_i - *i) -=- Optimizer(*best' j t *GES G;)*
23: *!x j x c P, t* -- 1, ...，*N*，*j U Ci* $\leftarrow P_i$;
24: *bF. +- \bF. -|- * $|f(\hat{\mathbf{x}}_{best}) - f(\mathbf{x}_{best})|)/2$;
25: 如果*PZ*，等于1，那么
26: *4 F_i* =- 0;
27: 结束 如果
28: 结束时间
29: 结束时

算法 *(i'* 最好的' *PG' +!* 优化器(*best') *i'* 创业板，围棋)

1: *G* +- 去：
2：对于x e P_i，评估（x，*xbestt"*
3: while G < Go + <Es do
4: for x e P_i do
5: s+-Reproduction(x); /*进化过程*/
6: 评估 (â -*b!)*
7: 如果 $\mathbf{x}(\hat{s}, \begin{matrix} P_i \\ BEST \end{matrix})$ 比（X、$\mathbf{x}_{best}^{P_i}$）**then**
8: x +- x;
9: 结束 如果
10: 如果(x, *s$)*是更好的 thdfl **best* 则
12: *"最好的" ""b' st)*
结束 如果

的整体解决方案*best被更新。最后，*best被返回到算法2。

与传统的CC相比，CCFR需要额外的运算来在协同进化循环开始前初始化最佳整体解决方案（算法2中的第4步），其计算复杂度为*O(M - N)*。 CCFR还需要

13: 结束
14: $G = -G + 1$；
15: 使用公式（6）计算po；
16: 如果PTY等于1，那么
17: 终止算法并返回; 18: 结
束 如果
19: 结束时

额外的计算是检查一个子群在每一代是否是滞后的（算法3的第15步），计算复杂度为$O(Di^{'}N)$。

IV. 实验性研究

在IEEE CEC'2010和CEC'2013关于大规模全球优化的特别会议上，提出了一套35个1000-D的测试实例，用来研究CCFR的性能。这些测试实例的详细描述在[24]和[47]中给出。与CEC'2010的函数相比，CEC'2013的函数有四个新特征：1）不均匀的子组件大小；2）子组件贡献的不平衡性；3）函数具有

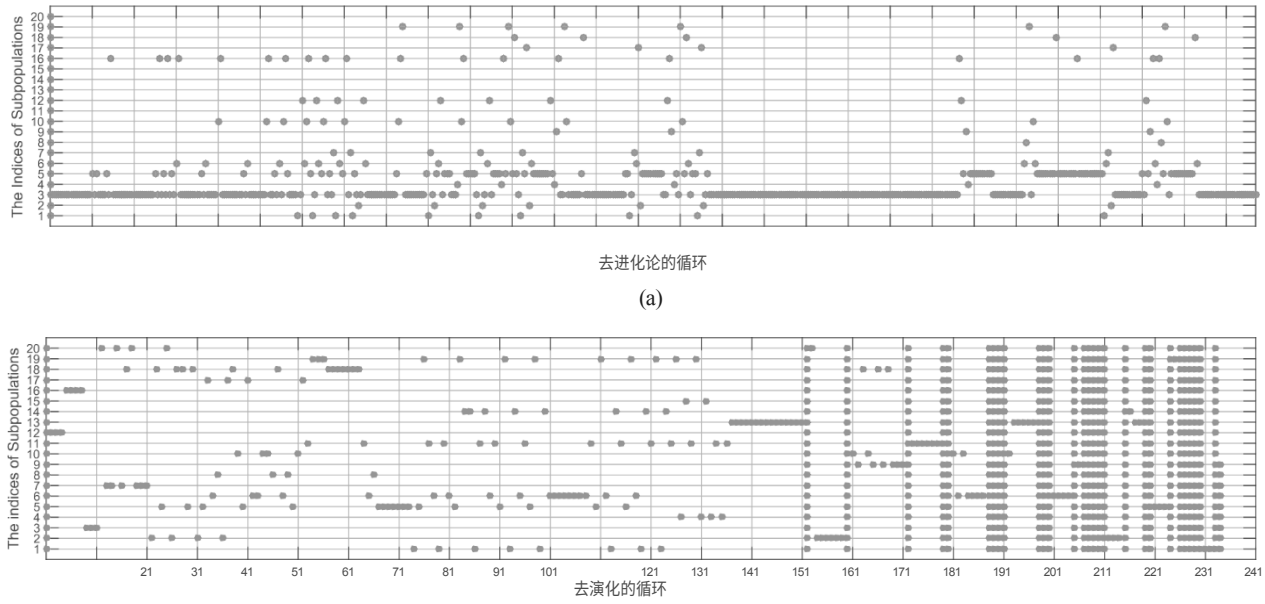


图2. 在一次运行中，CCFR-I对两个选定的CEC'2013函数的子种群的激活，其中填充的圆点表示该子种群在相应的共同进化周期中经历了进化。(a) fg。(b) *ft* o

重叠的子组件；以及4）对基础函数的新转换。

在实验研究中，CCFR与七个CC算法（DECC-G [13], MLCC [20], DECC-D [23]）进行了比较、DECC-DML [23], DECC [28], CBCC1 [9], 和CBCC2 [9]) 和两种记忆算法（MA-SW-Chains[48] 和 MOS-CEC2013[49]）。这两种记忆算法分别在IEEE CEC'2010和CEC'2013的大规模全球优化竞赛中排名第一。按照[24]中的建议，我们将最大适配度评估数设定为MaxFEs = 3 x 10^6 作为终止标准。对于CCFR的竞争对手，参数被设置为他们出版物中使用的值。为了进行公平的比较，CCFR和其他被比较的CC算法采用了以下相同的参数设置。

1) 子组件优化器是SaNSDE[50]，是DE[18]的一个变种。SaNSDE的群体大小被设定为50。
2) 预先指定的进化代数，即算法3中的GEs，被设定为100。

A.CCFR的行为

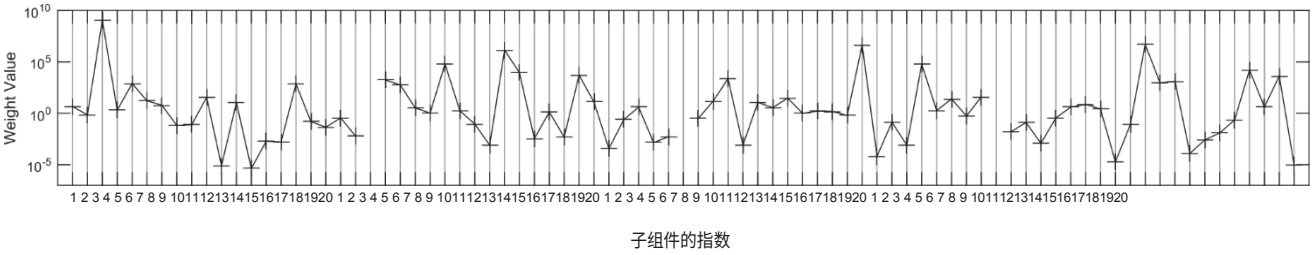
在本节中，研究了CCFR的行为。变量的分组是一个理想的分解，它是利用基准函数的先验知识手工完成的。

图2显示了在两个CEC'2013函数（/8和/io）上单独运行的子种群的激活情况，其中有20个可分离的子成分。所有子种群的贡献都是在第一个共同进化周期中计算出来的。对于fg，由于第三个子种群具有最大的权重值[47]，相应的子种群（即3）对改善最佳整体的贡献最大。

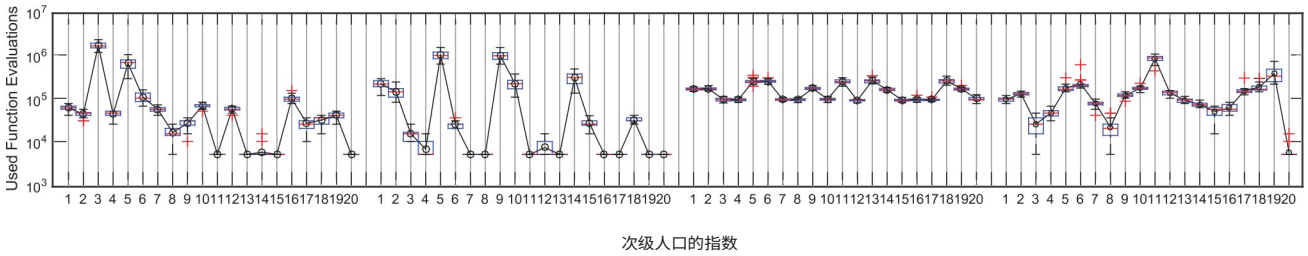
目标值。在图2(a)中，可以看出，在第一个周期之后，P3在随后的连续周期中经历了演变。随着演化的进行，P3的贡献越来越小。在第21个周期中，Ps的相关子成分具有第二大权重值，经历了演变。从图2(a)中，可以看出两点：1）子种群交替进行演化；2）大部分计算资源花在P3和P5上，其对应的子成分分别具有最大和第二大权重值。对于Mfg，根据子种群的动态贡献，CCFR可以在子种群之间自适应地分配计算资源。

对于/；，从图2(b)中可以看出，P12.其响应的子组件具有最大的权重值[47]，在几个连续的周期中经历了演变。由于CCFR使用的**优化器**SaNSDE不能解决这个函数有效地，Pi2是停滞不前的。rl2的分布在连续的几代中保持不变。在第四个周期，CCFR根据(6)认为Pi2是停滞的，并将其排除在后续周期之外。在第l52个周期中，所有的亚种群都停滞不前。共同进化从第一个周期重新开始。所有的亚种群都逐一进行了进化。

图3显示了CCFR-I在四个CEC'2013函数（/g-/i i）上的资源分配情况。这些函数有20个可分离的子成分。各子组件的权重值明显不同[见图3(a)]，这导致各子种群对提高最佳总体目标值的贡献明显不同。从图3中可以看出，对于jfg-/ t除了/io.，一个子组件的权重值越大，其相应的子种群用于进化的资源就越多。如前所述，CCFR中使用的优化器无法解决



(a)



(b)

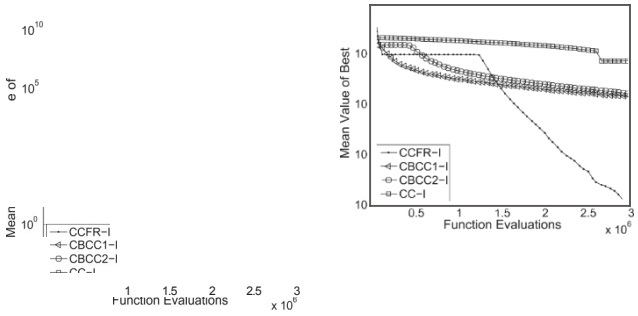
图3.在CCFR-I中，对四个选定的CEC'2013函数的子群进行计算资源分配。(a)子群的权重值。(b) 每个子群在25次独立运行中用于优化其相应子组件的函数评价的箱形图，其中圆点表示每个子群在25次独立运行中使用的函数评价的平均数。

/实验结果表明，所有的亚种群都是有效的，所以所有的亚种群在一些周期后都处于停滞状态。然后所有的子种群逐一进行进化。因此，对于/i0.来说，分配给不同子种群的计算资源并没有很大差别
[见图3(b)]。

*B.**CCFR**与其他CC框架的比较*

在本节中，CCFR与CBCC的两个变种（CBCC1和CBCC2）[9]和传统的CC[7]进行了比较。CCFR-I、CBCC-I和CC-I的变量分组是一种理想的分解，它是利用函数的先验知识手工完成的。所有的函数评价都用于优化。对于可分离的变量，CCFR和CC分别对变量进行优化，而CBCC对变量进行优化[28]。CCFR-I、CBCC-I和CC-I之间的唯一区别是它们采用的CCFR。表一总结了CCFR-I、CBCC1-I、CBCC2-I和CC-I的结果。

*1) 对**IEEE CEC'2010**函数的比较。*结果显示，在20个函数中的13个上，CCFR-I的表现明显优于其他同行的算法。CCFR-I在所有可分离函数（/i-/3）和大部分部分可分离函数（/4-/ig）上的表现优于其他同行算法。对于CCFR-I表现较差的部分可分离函数，CCFR-I和其他同行算法的结果差异不大。对于CCFR-I表现较好的函数，其差异是显著的，特别是对于/7./对于不可分离



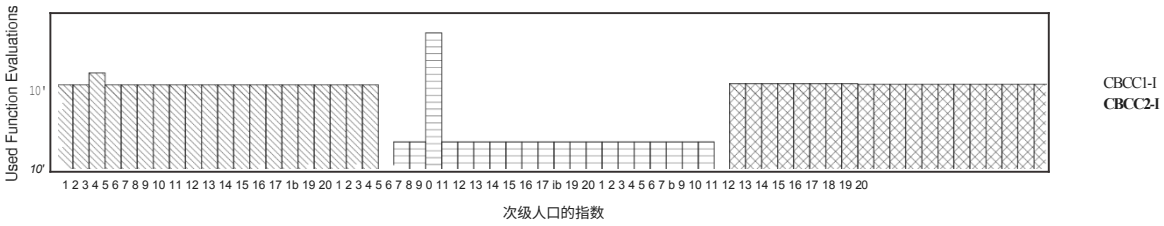
(a) (b)

的函数（/t9和/20），所有的变量都被归入一个子组件并一起优化，因此，不存在计算资源分配的问题。CCFR-I、CBCC-I和CC-I在不可分离函数上有类似的性能。

图4.在25次独立运行中，两个选定的CEC'2010函数的平均收敛性。(a) /t . (b) *ft* z

图4显示了四种CC算法的收敛情况。/是一个完全可分离的函数，其中每个变量都有一个权重值。这些权重值随着变量指数的增加而增长。/i2是一个部分可分离的函数，有10个不可分离的子成分和500个可分离的变量。

CC不能在停滞的子种群上节省计算资源。从图4中可以看出，CC-I的收敛速度非常慢。相比之下，CCFR可以阻止停滞的子种群的演化。因此，CCFR在可分离变量上花费的计算资源要少得多，收敛速度比CC-I快。在进化过程的开始阶段，CCFR-I收敛得很慢。这是因为CCFR-I在第一个共同进化周期中逐一优化了所有的子组件，包括可分离变量。当第一个周期结束时（对/来说，大约2.5 x 10^6 函数评估；对/12来说，大约1.3 x 10^6 函数评估），CCFR-I开始选择具有最大贡献的子种群来进行进化，因此CCFR-I的收敛速度增加。CBCC将所有的可分离变量归入一个子组，所有的可分离变量一起被优化[28]，这就失去了CC的分而治之策略的威力。在图4(a)中，它可以



图S.在　25次独立运行中，每个子群用于优化其相应子组件的CEC2013函数（/g）的平均函数评价。

表一

CEC'2010和CEC'2013函数在25次独立运行中的平均F值-i标准偏差。在B OLD FONT(WILC OXON RANK S UM TEST WITH HOLM *p*- VALUE C ORR ECTI ON)中的结果较好、

= 0.05).A+、A和p值是通过WILC OXON的多指标模拟分析获得的。

在CC F-I和其合作伙伴之间进行测试

	CEC'2010 Functions			
	CCFR-I	CBCC-I	CBCC2-I	CC-I
/j /2 /3	1.2e-05Ad.9e-06 2.7e+（ llA5.2e+00 4.6e*(l0A4.6e-01	9.9e*06A1.3e+07s 4.7e*03A4.8e+02s- 1.2e*01-L3.7e-01{	9.9e*06A1.3e* +1 4.7e*03A4.8e*021 1.2e*01+3.7e-011	3.5e+11A2.0e*10d- 9.4e+03A2.1e*02d- 2.0e+01+4.4e-021
/ /5 /# /y J$	8.3e*10-I-6.2e*l0 7.2e*07Al.3e*07 7.7e+(l5A7.1e+05 1.5e-(l3A2.5e-04 3.2e+05A1.1e+06	6.0e*l0-L4.4e+10 6.8e*07A1.0e+07 1.3e*06A6.4e+05s- 5.9e*04-L 3°* 31 8.6e*05A1.6e*06j' 。	9.9e*10-I-2.7e*101 6.7e*07A9.1e*06 1.3e*06A6.8e*05d- 8.4e*04+1.9e*04t 1.0e*06-i-1.7e*061	3.4e+14+7.5e*13 - 4.9e+08A2.4e*07d- 1.1e+07A7.5e*05d 7.7e+10+9.6e*09t * 8e+14-j-9.3e*13d-
J9 我 ƒi i 2 ÃÃÃ	9.4e*06-i-1.2e*06 1.de+03d1.0e+02 1.0e*ll1A2.7e+00 1.2e+00Ad.6e+00 3.2e+Il2A9.9e+01	1.7e*07A2.LE*07 3.0e*03d1.7e-F02j- 2.2e*01A3.2e+ 1 1.8e*04A6.5e*03j' 。 1.9e*04A6.3e+03j	2.8e*09-i-1.8e*0 1 4.5e*0346.6e*021 2.4e*01-1-2.7e*001 2.5E*04-J-7.3E*0+1 2.8e*0445.4e*031	.4e+08-i-7.1e*073 4 8e+0346.7e*01d- 4.1e+01-i-I.5e*001 4.9N05-i-3.4e*04d'。 1.5e+0744.1e*06d-
姬4 ƒz ƒz /j/ ƒ jg	2.5e+(l7A2.9e+06 2.8e+(l3Al.3e+02 2.0E*01-J-2.6E*00 9.8e+(l0Al.1e+01 1.1e*03A1.8e*02	2.8e*07A2.Ie+06s- 4.0e*03A1.5e+02s- 1.9e*01A3.2e+00 3.5e*01A4.9e+OU- 1.1e*03A1.8e+02	9.5e*09A5.2e* 1 4.2e*03A1.6e*021 2.0E*01-J-3.4E*00 1.4e*02A4.4e*011 1.4e*03A1.9e*021	2.7e+07A2.1e*06d- 4.0e+03Al.6e*02d- 2.0M01-J-4.0E*00 2.2e+01A3.7e*01d- 1.0e+03Al.7e*02
ƒi 9 ƒag	1.2e*06A9.5e*04 1.0e*09-1-9.0e*08	1.2e*06A9.5e+04 1.0e*09A9.0e+08	1.2e*06A9.5e*04 I.0e*09-i-9.0e*08	1.2e+06A9.5e*04 1.0e+09-i-9.0e*08
A" A* P值		167.0 43.0 2.06e-02	194.0 16.0 8.92e-04	204.0 6.0 2.19e-04

CEC'2013的职能

F	CCFR-I	CBCC I -I	CBCC2-I	CC-I
i	1.3e-05A3.2e-06	1.4e*07A3.6e*071	I.4e*07-1-3.6e*071	3.7e+11-i-I.5e*10+ J9
	5.5e-01A1.5e+00	2.1e*04A9.9e*02j-	2.1e*04-j-9.9e*021	8.5m04-j-5.1e*03d'
/32	.0e+(llA3.1e-07	2.1e*01A1.ie-021	2 .1e*01A1.ie-02s-	2 .Ie+01A9.ie-03j-
/4	.5e+(l7Al.7e+07	1 .6e*08ñ6.0e+07s	6 .6e*10A5.6e* 1	1 .7e+12A4.8e*11d-
/5	2.5e*06A2.7e*05	2.5e*06A4.2e+05	2.4e*06A4.5e*05	1.2e+07A6.9e*05d- Jb
	1.1e*06A1.2e*03	1.1e*06A1.9e+03. [	1.1e*06A1.7e*03.b	1.1e+06Al.6e*03d-
//8	.6e+(l6Al.9e+07	1.9e*07A2.4e+07s-	9.6e*07A3.7e* 1	4.2e+09Al.1e*09d-
/ø9	.6e+（I9A1.6e+10	2.0e*13A2.8e+13j-	1.0e*12A1.3e*111	4.7e+13A2.8e*13d-
/91	.9e*(l8A2.8e+07	2.5e*08A3.8e+		071
2.2e*08A	2.8e*071	2.9e+08A5.2e*07d- *ig*		9.5e*07-1-1.9e*05
9.4e*07A	2.8e*05. [	9.4e*07-1-2.3e*05.b	9.4e+07-i-2.9e*05.b	*f i i*
	3.3e+08A3.2e+08	3.0e*09A1.0e+10]	4.9e*10-1.8e*10	2.2e+09-i-8.4e*09d'
f_{12}	3.6e+08±7.1e+08	6.7e+08±7.1e+08	6.1e+08±7.1e+08	6.7e+08±7.1e+08
f_{13}	9.3e+08±5.3e+08	9.5e+08±5.4e+08	9.5e+08±5.4e+08	9.5e+08±5.4e+08
f_{14}	2.1e+09±2.1e+09	2.2e+09±2.1e+09	2.2e+09±2.1e+09	2.2e+09±2.1e+09
i z	8.2e*06-3.3e*06	8.3e*06A3.3e*06	8.3e*06-I-3.3e*06	8.3e+06-i-3.3e*06
A" fi* P值		109.0 11.0 3.36e-03	107.0 13.0 5.37e-03	115.0 5.0 6.10e-04

符号d'和.b分别表示，通过Wilcoxon秩和检验，在0.05的显著性水平下，CCFR-I算法的表现明显优于和差于该算法。

可以看出，CBCC1-I和CBCC2-I在*f*上收敛得很慢。当CCFR-I完成第一个协同进化循环时，*f*的最佳总体目标值比CBCC更好的亚种群。在/12上单次运行的实验结果表明，对于第三个亚群，CBCC1-I和CBCC2-I使用了大约5 x 10^5 和1 x ^{106} func-

在这一过程中，CCFR-I使用了大约1.9 x 10次函数评估，使最佳总体目标值提高了6.9 x 10^5 。CCFR-I使用了大约1.9 x 10次函数评估来实现6.9 x ^{105}的改进。当

第三个子群的实时贡献相对较小，CBCC仍然将计算资源分配给该子群，而CCFR则将资源分配给其他一些实时贡献相对较大的子群。

2) *在IEEE CEC'2013函数上的比较。*为了进一步研究不平衡的影响，在CEC'2013函数上测试了CCFR-I、CBCC-I和CC-I。结果显示，在15个函数中的8个上，CCFR-I明显优于其他同行的算法。CCFR-I在所有函数上的表现明显优于其他同行算法。

属于可分离的函数i-(/3和大多数部分可分离的

急剧下降。对于/12，CCFR-I比CBCC1-I和CBCC2-I收敛得更快[见图4（b）]。CBCC根据累积的贡献在子种群之间分配计算资源。在强调近期贡献的同时，CCFR将计算资源的分配适应于以下的实时贡献

函数（/4-/i i ）。CCFR-I、CBCC-I和CC-I在非分离函数/12-/is上有相似的表现。 对于CCFR-I表现较差的部分分离函数，CCFR-I和其他同行算法的结果差异不大。对于CCFR-I表现较好的函数，其差异是显著的，尤其是对于/4./7./g和/ ，其中CCFR-I的表现优于其他算法。
其他同行算法的几个数量级。

图5显示了在25次独立运行中，每个子群用于优化其相应子组件的CBCC1-I、CBCC2-I和CC-I对fg的平均函数评价。对于fg，各子构件的权重值明显不同[见图3（a）]。从图5中可以看出，CC-I对所有子种群分配了相等的计算资源。CBCC-I和CBCC2-I对所有的子种群分配了相等的计算资源，除了第三个子种群（3）在进化过程的开始，3的贡献最大。因此，CBCC1-I和CBCC2-I将更多的计算资源分配给P3。在随后的共同进化循环中，P3在一个循环中的贡献下降了，但CBCC-I和CBCC2-I仍然认为P3的贡献最大，并将资源分配给P3，而不是其他实时贡献最大的子种群。相反，当P3的实时贡献很小时，CCFR-I将计算资源分配给具有最大实时贡献的P5。将更多的计算资源分配给具有最大贡献的子群体，可以增加做出更大贡献的概率。

最佳总体目标值的改进。简而言之，对于

表二

CEC'2010年和CEC'20 13年职能的平均排名情况

(弗里德曼测试) 。最重要的结果是以旧字体显示的。

	CCFR-I CCFR-I	CBCC1-I CBCC1-I	CBCC2-I CBCC2-I	CC-I CC-I	*p*-value P值
Average Ranking 平均排名	1.4000 **1.4000**	2.3714 2.3714	2.8286 2.8286	3.4000 3.4000	1.15e-10 1.15e-10

表三

平均健身值　　四个方面的标准差

I 部分分离的CEC'2013函数 *(fg -f* j) 超过25
。
独立运行，显著更好的结果是在
B 旧字体（WILCOXON RANK S UM TEST WITH HOLM p-VALUE
修正，h=0.05）。A+ , A*和p值有
与表一中的含义相似

I	CCFR-I	ICBCCI-I	ICBCC2-I	ICC-I
/g	9.6e+(l9A1.6e+10	1.9e*13A2.7e+13s-	9.9e*11A1.3e*111	4.7e+13A2.6e*13d-
/9	1.9e+（l8A2.8e+07	2 .5e*08A3.8e*071	2 .2e*08A2.9e*071	2 .8e+08-i-5.4e*07t
中	9.5e*07-i-1.9e*05	9 .5e*07A2.8e*05. [	9 .5e*07A3.1e*05. [	9 .5m07-i-2.8e*05
Ji i	3.3e*08A3.2e*08	5.2e*08A4.6e+087	.9e*09A1.2e	
1	.8e+09A6.1 e*09d-		/	
A'		9.0	9.0	9.0
A-	-	1.0	1.0	1.0
pvMu		230601	2.30>01	2.50e-01

符号I和.b的含义与表I中相似。

表四

平均值--标准偏差

CEC'20 10和CEC'20 13在25次独立运行中的功能。在25次独立运行中，S IG NIFI CANTLY B ETT ER RES ULTS处于B OLD FO NT（W ILCOX O N RANK S UM TEST WITH HOLM p-VALUE CORRECTION, O = 0.05）。A+、A*和*p*-值的含义如表一所示，为IMILA R。

*CJC'20? Ofun*行

F	CCFR-IDG2	**DECC-G**	**MLCC**	DECC-D	DECC-DML
/j	2-5-i-7-6	4e-7-1-le-7}。	8e-7-l-4e-7}	Ie-22-j-9e-21}	3e-7-1-9e-7}的情况下
/2	l.7e2:+:9e0	1.3e3:+:3eld-	3e-3:+:5e-3}。	6.5EL-I-4E1}。	1.0E1:+:2E3}
/3	l.2el:+:4e-I	1.1e0:+:4e-IQ	i e - 2 **:+:3e-2. [**	2.3e0:+:2e-1.b	3e-1:+:7e-IJ
/	1e11A8e10	2el3A5el2}	lel4A4el3{	3el2A9elld'。	1el4A2e14}
/5	9.2e7-i-2e7	2.2e8-i-7E7}。	3.0e8A1e81	2.9e8-j-1e8}。	3.0e8-i-1e8}。
/p	6.8e5:+:7eS	5.3e6:+:1e6d-	I.9e7A2e6t	5.9e6-i-5e6t	1.7e7:+:6e6d-
/2	2e-3A3e-4	8.le8A5e8}	5el0A2el0{	1.5e5A2e5d'。	3el0A5el0]。
/9	1.3e7+2e6	9.5e8+5e7d'。	1.7e9+-2e8{	i.0e8+9e6J	i.0e9+-ieg'
/bp	3.2e5-i-1e6	6.8e7-J-3e7t	8.2e8-E2e8I	1.3e8-j-1e8}。	3e10-l-7e 10t
/	1.8e3-j-1e2	1.1e4-j-4e2d'。	5.2e3A2e3{	4.1E3-J-1E3's}	4.3E3-J-
/ y					
/ 3	2E3d' 。				
** 1	2.0e1A3e0	2.6e1A1e0'{	2.0e2A2e0{	1.0e2A1e2d)	1.9e243e1}。
	2.0e1+2e1	9.9e4+1e4d')	8.7e5-+le5{	9.1e3+le3J	4.8es+res'
/ 5	5.3e2-j-1e2	53e3i3e3j	3.2E4F3O4}的情况。		5.4E3-J-3E3}。
/ ¢		8.6E4-J-2E5D'。			
/	3.1e7:+:3e6	9.8e8i8e7j	3.6e9i3e8{	3.0e8+2e7J	2.2e9+2e9J'。
/ g	**3.2e3+2e2**	1.2e4-I-7e2j')	1.2e4A2e3{	1.3e4-i-2e2}。	1.LE4-I-3E3J'
	。				
/ig	2.0e1+3e0	6.9e1+5e0]'	4.0e2+-3e0{	2.0E2+2E2}'。	3.6e2+1e2j'
/g	**6.7el:+:9el**	3.1e5:+:2e4d	I.8e6A2e5t	7.5e4-i-5e3}的情况。	
R⁻	—	9.7e5:+:1e6j'			
	1.4e3+2e2	3.5e4-i-1e4d'。	1.1e5A3e4{	1.4e4-i-1e4}。	7.8e4-i-2e5d'。
	I.3e6-J-te5	1.1e6i6e4}。		3.0E6C4E5}。	I.6e6-i-le6
	2.0e9-J-2e9	4.5e3i8e2}的情况。		1.8e5C2e5}	**2.3e3:+:2e2}**
	。	5.4e3:+:1e4.[			
-		176.0	187.0	184.0	188.0
		34.0	23.0	26.0	22.0
P值		8.03e-03	2.20e-03	3.19e-03	1.94e-03

CBCC1-I、CBCC2-I和CC-I（见表一）。

在CEC'2010和CEC'2013函数上，CCFR-I的平均排名是四个CC算法中最好的（见表二）。本节的结果表明，在CEC'2010和CEC'2013函数上，CCFR比CBCC和CC能更好地利用计算资源。

为了显示基于贡献的资源分配（见III-B节）对CCFR整体性能的影响，我们将CCFR-I与ICBCC2-I、ICBCC1-I和ICC-I进行了比较，它们分别是CBCC1-I、CBCC2-I和CC-I的改进版。ICBCC2-I、ICBCCl-I和ICC-I采用了CCFR的组成部分（见III-A-III-C节），除了基于贡献的资源分配。表三总结了部分分离的CEC'2013函数/g-/| t的结果。表一和表三的结果比较表明，在III-A和III-C节中提出的CCFR组件改善了CBCC1-I、CBCC2-I和CC-I在四个CEC'2013函数（/g-/}t）上的性能。然而，由于CCFR-I具有更好的基于贡献的资源分配，它在这四个功能中的大部分上仍然优于其他CC算法。

CCFR-I对块状旋转椭圆函数[51]的可扩展性研究在附录中所列的第二节补充材料中提供。结果表明，对于CCFR-I来说，随着函数维度和子组件数量的增加，函数求值的数量也会直线上升。CBCC1-I、CBCC2-I和CC-I的性能与CCFR-I相似，但对于CCFR-I来说，随着函数维度和子组件数量的增加，函数评估的数量增加的速度低于其他三种CC算法。

C.CCFR污垢IDG2

采用两种分组方法的CCFR的实验结果（在补充材料的第三节中提供）。

CEC'2013的职能

F	CCFR-IDG2	DECC-G	MLCC	DECC-D	DECC-DML
/t	2-5-+5-6	3e-6-i-2e-6}	ie-6:+:6e-7}	ie-17-1- ie-17}	7-8-i-3-7}
/2	3.6e2:+:2e I	1.3e3:+:3eld	2e-2:+:4e-2。[	7.1e I:+:3el}。	4.9e0:+:2el.[
/3	2.1e1-i- Ie-2	2.0e1-l-7e-3{	2.0e I A9e-4{	2.0e i+2 -3	2.0ei+2e-2
/	9.6e7-i-4e7	2e1l -l- lel I}。	2e12-j-8e1*1	3e10-l-2e10t	Ie12-1-Ie12}的情况下
/$	2.8e6:+:3eS	8.6e6:+:1e6d-	I.9e7A5e6t	6.1e6-i-2e6t	1.9e7:+:8e6d-
/b	1. Ie6:+: Ie3	1.Ie6:+:1e3.[	I.1e6A3e3}	I.1e6:+:2e3}。	**l.0e6:+:5e3.[**
//	2.0e7-J-3e7	!0e9-J-5e8t	8.4e9-E4e91	9.0e7-I-4e7}。	3.7e9-i-5e9t
/ø	7e10-1-Iett	9el5-1-4e15}的内容	8el6-i-4e16d-	2e14-l-9e13}	5e16-l-8e16}
/9	1.9e8-j-3e7	6.Ie8-j-1e8d'。	l.2e9A3e8{	5.1e8z ie8J	i.2e9z4e8'。
/ ø	9.5e7A2e5	9.3e7A5e5}。	9.3e7A5e5{	9.3e7A6e5}。	9.3e7A6e5}。
/j	4e8-t-3e8	2ell-1-9e10t	Ll2-J-5e1ld-	9e8-i-5e81	6el I -l-7e1 I}。
/ty	I.6e9+2e9	4.4e3+7e2 }	8.8e4-+-3e4{	2.3e3+2e2}.	5.2e3+1e4}。
/ 3	1.2e9-j-6e8	9.6e9-j-3e9]'	5elO-j- Ie10{	1.7e9-j-5e8}的情况。	2el 0-1-2e lO'{
/	3.4e9A3e9	2el I A5e10]	9ellA4e HQ	7.4e9A9e9	2el I A5e1 I}
/t5	9.8e6-i-4e6	1.2e7-l- 1e6j'	3.7e8A3e8{	**6.9e6+7e5}。**	3eI 0-i- Ie l ld-
A+		**98.0**	**96.0**	87.0	97.0
H-		**22.0**	**24.0**	33.0	23.0
yvdue	-	3.02e02	4.13e02	1.35e-01	3.53e-02

符号d-和的含义与表一中相似。

附录中列出的数据表明，高分组准确率可以提高CCFR的性能，特别是对于不可分离的变量。

在这一节中，将预先介绍CCFR-IDG2的性能。IDG2[52]是DG[28]的改进版，能够以非常高的精度将相互依赖的变量组合在一起，并正确识别决策变量之间的间接互动。CCFR-IDG2与七种CC算法（DECC-G [13], MLCC [20], DECC-D [23], DECC-DML [23], DECC [28], CBCC1 [9]）进行了比较、

和CBCC2[9]）和两种记忆算法（MA-SW- Chains[48]和MOS-CEC2013[49]）。在[53]中显示，这两种记忆算法在解决大规模优化问题方面具有竞争力。请注意，对于有IDG2的算法，花在分组上的函数评价被算作计算预算的一部分。

表四总结了CCFR-IDG2、DECC-G、MLCC、DECC-D和DECC-DML的结果。CCFR-IDG2的表现

在CEC 2010年所有的部分可分离的算法上，明显比其他同行的算法好几个数量级。

表五

CEC'2010年和CEC'20 13年职能的平均排名情况

(弗里德曼测试) 。最重要的结果是以旧字体显示的。

	CCFR-IDG2	DECC-G	MLCC	DECC-D	DECC-DML	p-value
Average Ranking	**2.1429**	3.0286	4.0000	2.3143	3.5143	5.66e-07

CCFR-IDG2 DECC-G MLCC DECC-D DECC-DML p-value 平均排名
2.1429 3.0286 4.0000 2.3143 3.5143 5.66e-07

函数(/4-/ig)和CEC2013年的大部分部分分离的 ble函数 $(f_4–f_{11}$ 。这表明高效的分组方法和高效的资源分配策略可以帮助CCFR获得有竞争力的性能。在CEC'2010和CEC'2013函数上，CCFR-IDG2的平均排名是五个CC算法中最好的（见表五）。

CCFR与CBCC1、CBCC2和DECC进行了比较，后者采用两种分组方法（即DG[28]和IDG2[52]）。详细结果见附录中所列补充材料的第三节。对于IDG2，比较结果与IV-B节中CCFR-I与其竞争对手（CBCC1-I、CBCC2-I和CC-I）的比较结果相似。结果显示，在大多数完全分离和部分分离的函数上，CCFR-IDG2的表现明显优于CBCC I-IDG2、CBCC2-IDG2和DECC-IDG2。在CEC'2010和CEC'2013的函数上，CCFR-DG的整体性能也优于CBCC1-DG、CBCC2-DG和DECC-DG。采用IDG2的算法比采用DG的算法表现更好。这是因为IDG2能够以更高的精度识别变量之间的相互依赖关系。

CCFR-IDG2和两种记忆算法（MA-SW-Chains和MOS-CEC2013）之间的比较在附录中所列的补充材料的第四节中提供。实验结果显示，在CEC'2013的函数上，CCFR-IDG2的整体性能比MA-SW-Chains和MOS-CEC2013差。然而，当我们用另一个优化器（即CMAES[54]）取代SaNSDE时，CCFR-IDG2的性能得到了改善。总的来说，在CEC'2010和CEC'2013函数上，带有CMAES的CCFR-IDG2的表现比MA-SW-Chains和MOS-CEC2013更好。

V. C ONC LUS ION

在本文中，我们提出了一个新的CCFR，名为CCFR，用于解决大规模的全球优化问题。CCFR的目的是在子群体中有效利用计算资源。与传统的计算资源在子种群间平均分配的CC和计算资源根据进化过程开始时各子种群的累积贡献分配的CBCC不同，CCFR根据各子种群以前和现在的贡献来分配资源。CEC'2010和CEC'2013大型基准函数被用来评估CCFR的性能。从我们的实验结果中，可以得出几个结论。

首先，CCFR可以检测到停滞的子种群并节省停滞子种群的计算成本。第二，根据子种群以前和现在对提高最佳总体目标值的贡献、

CCFR可以在子种群之间进行更有效的计算资源分配，并获得比其他CCFR更好的解决方案。最后，CCFR的性能取决于分组方法的性能。将相互关联的决策变量分组并具有较高的精度可以提高CCFR的性能。采用改进的DG方法的CCFR是解决大规模优化问题的一种极具竞争力的CC算法。

在未来，我们计划研究使用竞赛算法[55]、强化学习[56]和自适应选择操作者[57]中采用的技术在子群体中分配计算资源的潜力。

附录

在网上提供的补充材料

补充材料中的实验由以下部分组成：

1) 对CCFR的参数U进行敏感性研究；
2) 气候变化框架的可扩展性研究；
3) 含有DG和IDG2的CCFR的性能；
4) 实验结果显示，CCFR-IDG2算法与非CC算法之间的比较。

掌握了这些信息后，我们就可以对其进行管理了。

作者要感谢墨尔本大学计算机和信息系统系的B.Salehi博士对本文的认真校对。

参考文献

[1] R.A. Sarker, M. Mohammadian, and X. Yao, Eds., *Evolutionary Optimization.*New York, NY, USA: Springer, 2002.

[2] Y.Liu, X. Yao, Q. Zhao, and T. Higuchi, "Scaling up fast evolutionary programming with cooperative coevolution," in *Proc. IEEE Congr.Evol. Comput.,* Seoul, South Korea, 2001, pp.1101-1108.

[3] A.Vicini和D.Quagliarella，"通过混合优化策略的机翼和机身设计"，*AIAA J.，第*37卷，第。5, pp. 634-641, 1999.

[4] G. B. Dantzig 和 P. Wolfe, "Decomposition principle for linear programs," *Oper Res.,* vol. 8, no. 1, pp.101-111, 1960.

[5] A.Griewank 和 P. L. Toint, "Partitioned variable metric updates for large structured optimization problems," *Numerische Mathematik,* vol. 39, no. 1, pp.119-137, 1982.

[6] M.Z. Ali, N. H. Awad, and P. N. Suganthan, "Multi-population differen- tial evolution with balanced ensemble of mutation strategies for large- scale global optimization," *Appl. Soft Comput.,* vol. 33, pp.304-327, Aug.

[7] M.A. Potter和K. A. D. Jong, "A cooperative coevolutionary approach to function optimization," in *Parallel Problem Solving from Nature-PPSN III.*海德堡，德国：Springer，1994，第249-257页。

[8] P.L. Toint, "Test problems for partially separable optimization and results for the routine PSPMIN," Dept. Math., Univ. Namur, Namur, Belgium, Tech.Rep. 83/4, 1983.

[9] M.N. Omidvar, X. Li, and X. Yao, "Smart use of computational resources based on contribution for cooperative co-evolutionary algorithms, " in *Proc.Genet.Evol. Comput.Conf.,* Dublin, Ireland, 2011、pp.1115-1122.

[10] Y.-P.Chen, T.-L. Yu, K. Sastry, and D. E. Goldberg, "A survey of linkage learning techniques in genetic and evolutionary algorithms," Illinois Genet.Algorithms Lab., Univ. Illinois at Urbana-Champaign, Champaign, IL, USA, Tech.Rep. 2007014, 2007.

[11] T.Weise, R. Chiong, and K. Tang, "进化优化：陷阱和诱杀陷阱"，*J. Comput.Sci. Technol.,* vol. 27, no.5, pp. 907-936, 2012.

[12] R.Salomon, "Re-evaluating genetic algorithm performance under coordinate rotation of benchmark lunctions.A survey o1 some theoretical and practical aspects of genetic algorithms," *Bins5.stem.s,* vol. 39, no.3, pp.263-278, 1996.

[13] Z.Yang, K. Tang, and X. Yao, "Large scale evolutionary opti- mization using cooperative coevolution, " /n/：*Sci.,* vol. 175, no. 15、

14] pp.2985-2999, 2008.
Y.Mei, M. N. Omidvar, X. Li, and X. Yao, "A ct'mpetitive division-and-conquer algorithm for unconstrained large-scale black-box optimiza- tion, " *ACM Trans.Math.Sofrtv.,* vo1.42， no.2， pp.1-24，2016。

[15] F. van den Bergh和A. P. Engelbrecht, "A cooperative approach to particle *swarm* optimization," *IEEE Trans.Evol. Conifiut.,* vol. 8, no.3, pp.225-239, Jun. 2004.

[I 6] J.肯尼迪和R.埃伯哈特，"粒子群优化"，在 *Proc. IEEE Int.Con(. Neural Netw.),* Piscataway, NJ, USA, 1995、pp.1942-1945.

[17] Y.-J. Shi, H.-F.Teng, and Z.-Q.Li, "Cooperative co-evolutionary differential evolution for functir'n optimization," in *Ad ances in Natural Comytita limit.*Heidelberg, Germany：Springer, 2005, pp.1080-1088.

[18] R.S torn 和 K. Price, "Differential evolution-A simple and effi- cient heuristic for global optimization over continuous spaces," 7.*Glob.*Optic., vol. 11, no.4, pp. 341-359, 1997.

[19] M.N. Omidvar, X. Li, Z. Yang, and X. Yao, "Cooperative co-evolution for large scale optimization through more frequent random grouping, " in *Proc. IEEE Congr.Evol. Comput.,* 201(l, pp. 1-8.

[20] Z.Yang, K. Tang, and X. Yao, "Multilevel cooperative coevolution for large scale optimization," in *Prc'c'.IEEE Congr.Evol. Cc'mput.,* 2008、

[21] pp.1663-1670.
X.Li and X. Yao, "Tackling high dimensional nonseparable optimization problems by cooperatively coevolving particle swarms," in

[22] #rrc.***IEEE*** Con or *Evol. Comput.,* Trondheim, Ntirway, 2009, pp.1J46-1J53.
X.Li an'l X. Yao, "Cooperatively ctievolving particle swarms for large

[23] scale optimization," *IEEE Trans.Evol. Comput.,* vol. 16, no. 2、
pp.210-224, Apr. 2012.

[24] M.N. Omidvar, X. Li, and X. Yao, "Cooperative co-evolution with delta grouping fior large scale non-separable function optimization, " in *Proc. IEEE Cc'ngr.Evc'l.Cc'input.,* Barcelona, Spain, 2010, pp.I-S.
K.Tang, X. Li, P. N. Suganthan, Z. Yang, and T. Weise, "Benchmark functions for the CEC'2010 special session and competition on massive

[25] global optimization," Nature Inspired Comput.Appl.实验室，中国科学技术大学，合肥，2010年lOnline]。Available: http://goanna.cs.rmit.edu.au/ xiaodong/publications/lsgo-cec10.pd1

K.Weicker 和 N. Weicker, "On the improvement of coevolutionary optimizers by learning variable interdependencies," in *Proc. IEEE Congr. Evol. Comput.,* Washington, DC, USA, 1999, pp.1627-1632.

[26] W.Chen, T. Weise, Z. Yang, and K. Tang, "Large-scale global optimiza- tion using cooperative coevolution with variable interaction learning, " in *Parallel Problem Solving from Nature-PPSN XI,* Heidelberg, Germany：
Springer, 2010, pp.300-309.

[27] M. Tezuka, M. Munetomo, and K. Akama, "Linkage identification by ntanlinearity check for real-coded genetic algorithms, " in Proc.Can *Genet.Evol. Comf'ut.,* Seattle, WA, USA, 2004, pp.222-233.

[2x1 M. N. Omidvar, X. Li, Y. Mei, and X. Yao, "Cooperative co-evolution with differential grouping for large scale optimization, " /££ *Trans. Evol. Comput.,* vol. lS, no.3, pp. 378-393, Jun. 2t114.

[16] S.-M.Guta, C.-C.Yang, P.-H.Hsu, and J.-H.Tsai, "Improving differ- ential evolution with a successful-parent-selecting Framework, " *IEEE Tran.s. Evol. Com{iut.,* vol. 19, no.5, pp. 717-730, Oct. 2015.

[37] S.-M.Guo, J. S.-H.Tsai, C.-C.Yang, and P.-H.Hsu, "A self L-SHADE的优化方法与基于特征向量的交叉和成功父母选择框架在CEC 2015基准集上的结合"，在*Proc. IEEE Con gr. Evol. Comput.,* Sendai, Japan, 2(l15, pp. 1fl(l3-101(1.

[38] X.Yao, Y. Liu, and G. Lin, "Evolutionary Programming made laster," 姚明、刘宇、和G. *IEEE Truns.Evol. Comput.*，第3卷，第2期，第82-1 fl2页，1999年7月。

[39] M.Yang, Z. Cai, C. Li, and J. Guan, "An improved adaptive dillerential 带种群适应的进化算法"，在*Proc.Genet.Evol. Com;'ut.*Con/, Amsterdam, The Netherlands, 2013, pp.145-152.

[40] D.Thierens, "Adoptive strategies for operator allocation," in *Pcircinleter Setting in Evc'lutit'ilury Al gc'rithms* (Studies in Computational Intelligence), F. G. Lobo, C. F. Lima, and Z. Michalewicz, Eds. , vol. 54.Heidelberg, Germany：Springer, 2007, pp.77-90.

41] P.Auer, N. Cesa-B ianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Mach.Lecirn.*，第47卷，第2-3期、

[42] pp.235-256, 2002.
K.Li, A. Fialho, S. Kwong, and Q. Zhang, "Adoptive operator selec- tion with bandits for a multiobjective evolutional algorithm based on

43] decomposition," *IEEE Trans.Evol. Comput.*，第18卷，第1期，第114-130页，2014年2月。

[44] Q.Zhang, W. Liu, and H. Li, "The performance of a new version of MOEA/D on CEC(I9 unconstrained mop test instances," in *Proc. IEEE Congr.Evol. Comprit.,* Trondheim, Norway, 20fl9, pp.203-20S.

[45] A.Zhou and Q. Zhang, "Are all the subproblems equally important'!Resource allocation in decomposition-based multiobjective evolution- ary algorithms," *IEEE Trans.*£vo/。*Comput.*，第20卷，第1期，第52-64页，2016年2月。

[4f'] A.Fialho, M. Schoenauer, and M. Sebag, "Ana1y.sis of adaptive operator selection techniques on the royal road and long k-path problems," in *Près'.Grnet.Evol. Coinyut.Conf.,* Montreal, QC, Canada, 2009、
pp.779-786.

[47] F.-M.De Rainville, M. Sebag, C. Gagné, M. Schoenauer, and D.Laurendeau, "Sustainable cooperative coevolution with a multi-armed bandit," in *Proc.Genet.E vol. Comput.Conf.,* Amsterdam, The Netherlands, 2013, pp.1517-1524.
X.Li, K. Tang, M. N. Omidvar, Z. Yang, and K. Qin, "Benchmark tiinctions for the CEC' 2013 special session and competition on large scale global optimization, " EvoL Comput.Mach.Learn.Group, RMIT Univ., Melbourne, VIC, Australia, 2013.[在线]。Available: http://goanna.es.rmit.edu.au/ xiaodong/cec 13-Isgo/competition/cec2013-lsgo-benchmark-tech-report.pdf

[4B] D. Molina, M. Lozano, and F. Herrera, "MA-SW-Chains：基于局部搜索链的记忆算法，用于大规模的连续全局优化，"在*Proc. IEEE Congr.Evol. Coniput.,* Barcelona, Spain, 2010, pp.1-S.

[49] A.LaTorre, S. Muelas, and J.-M. Peña, "Large scale global optimization: a large scale global optimization.Peña, "大规模的全球优

[50] 化：基于MOS的混合算法的实验结果，"在*Proc. IEEE Congr* £vo/.Camper., Cancun, Mexico, 2013, pp.2742-2749.

[51] Z.Yang, K. Tang, and X. Yao, "Self-adaptive difterential evolution with neighborhood search, " in *Prc'c'.IEEE Congr.Evol. Cc'mput., 香港*, 20(l8, pp. HU-1116.

[29] [32] [33] [30] [34] [31] [35] R.Ros和N. Hansen, "CMA-ES中的一个简单修改实现了 M.N. Omidvar, Y. Mei, and X. Li, "Eitective deco mposition of large- scale separable continuous functions for cooperative co-evolutionary algorithms," in *Proc. IEEE Congr.Evol. Comyut.,* 中国北京, 2014、
pp.1305-1312.

B.Kazimipour, M. N. Omidvar, X. Li, and A. K. Qin, "A sensitivity analysis of contribution-based cotaperative co-evolutionary algorithms," in *Proc. IEEE Congr.Evol. Comf'ut.,* Sendai, Japan, 2015, pp.417W24.

M.Yang, C. Li, Z. Cai, and J. Guan, "Differential evolution with auto-

enhanced population diversity," /££Traits.*C' beri1.*, vol. 45, no. 2、pp.3(12-315, Feb. 2015.

F.Peng, K. Tang, G. Chen, and X. Yao, "Multi-start JADE with knowl- edge transfer for numerical optimization," in *Proc. IEEE Congr.Evol. Comput.,* Trondheim, Norway, 2009, pp.1889-1895.

Y.-L. Li and J. Zhang, "A new differential evolution algorithm with dynamic population partition and local restart," in *Prc'c.* level. Owl. *Comput.Conf.,* Dublin, Ireland, 2011, pp.1083-1092.

M.Vasile, E. Minisci, and M. Locatelli, "An inflationary dilferential evolution algorithm *for* space trajectory optimization," *IEEE Trans.Evol. Comput.,* vo1.15, no. 2, pp. 267-281, Apr. 2011 .

M.Zhabitsky and E. Zhabitskaya, "Asynchronous differential evolution with adaptive correlation matrix," in *Proc.Genet.Evol. Comput.Conf:,* Amsterdam, The Netherlands, 2013, pp.45J-462.

线性时间和空间的复杂度，"在*并行问题解决的前线*

[52] *Nature-PPSN X.* Heidelberg, Germany：Springer, 2008, pp.296-305.

M.N. Omidvar, M. Yang, Y. Mei, X. Li, and X. Yao, "IDG: A fiaster and more accurate ditferential grouping algo- rithm, " School Comput.Sci., Univ. Birmingham, Birmingham, U.K., Tech.Rep. CSR-15-04, Sep. 2015. lOnline].Available: ftp://ftp.es.bham.ac.uk/pub/tech-reports/2015/CSR- 15-04.pdf

53]

A.LaTorre, S. Muelas, and J.-M. Peña, "A comprehensive compari-son of large scale global optimizers," lu/s.Peña, "A comprehensive compari-

[54] son of large scale global optimizers," Lu/.*Sci.,* vol. 316, pp. 517-549, Sep. 2015.

N.Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time com-

[55] plexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES), " Avr/.Couper.，第11卷，第1期，第l-1 8页，2003

[56] 年。

M.Birattari, T. Stützle, L. Paquete, and K. Varrentrapp, "A racing algo-

[57] rithm for confi gurim metaheuri stics, " in *Proc.Genet.Evol. Comprit.*CHU., 2(102, pp. 11-18.

J.Schmidhuber, "A general method for multi-agent reinforcement Learn- ing in unrestricted environments," in *Evolutionary Computation：*

Thero y and Appli'atir'ns, X. Yao, Ed.新加坡：World Sci., 1999, pp.S1-123.

P.A. Consoli, Y. Mei, L. L. Minku, and X. Yao, "Dynamic selection of evolutionary operators based on online learning and fitness landscape analysis," *Soft Comput.,* vol. 20, no. 10, pp. 38S9-3914, 201f*.

杨明分别于2005年、2008年和2012年在中国武汉的中国地质大学获得计算机科学学士、硕士和博士学位。

2014年至2015年，他在英国伯明翰大学计算机科学学院担任博士后研究人员。目前，他是中国地质大学计算机科学学院（武汉）的副教授。他是湖北省智能化重点实验室的成员。

地理信息处理，武汉。他目前的研究兴趣包括蜂群智能、大规模优化、多目标优化及其应用。

穆罕默德-纳比-奥米德瓦尔（S'09-M'15）于2010年获得计算机科学学士学位（一等荣誉），2015年获得计算机科学博士学位，并在澳大利亚维多利亚州墨尔本市的RMIT大学获得应用数学学士学位。

2008年至2015年，他是RMIT大学进化计算和机器学习小组的成员。他目前是进化计算的研究员，也是计算的卓越研究中心的研究员。

他目前的研究兴趣包括大规模的全局优化和多目标优化，他是英国伯明翰大学计算机科学学院的智能和应用专家。

奥米德瓦尔博士是2010年澳大利亚研究生奖和皇家墨尔本理工大学计算机科学与信息技术学院的最佳计算机科学荣誉论文奖的获得者。

李长河（M'12）分别于2005年和2008年在中国武汉的中国地质大学获得计算机科学学士和硕士学位，并于2011年在英国莱斯特的莱斯特大学获得计算机科学博士学位。

自2012年以来，他一直是中国地质大学的副教授。他目前的研究兴趣包括机器精益的进化算法，群集智能，多模态优化，以及动态优化。

观念

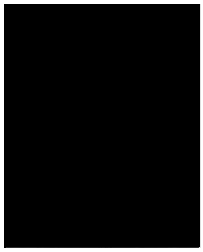
。

李博士是动态和不确定环境下的进化计算工作组的副主席。

李晓东（M'03-SM'07）在中国西安的西安大学获得理学士学位，并在新西兰达尼丁的奥塔哥大学获得信息科学博士学位。

'他目前是澳大利亚维多利亚州墨尔本市RMIT大学理学院（计算机科学和软件工程）的副教授。他目前的研究兴趣包括进化计算、神经网络、数据分析、多目标优化、多模态优化和蜂群智能。

李博士是2013年ACM SIGEVO影响力奖的获得者。他担任**IEEE TRANSACTIONS ON EVOLUTIO NA RY COMPUTATION**、*Swami Intelligence*（Springer）和*International Journal of Swarm Intelligence Research*的副编辑。



蔡志华于1986年在中国武汉大学获得理学士学位，1992年在中国北京理工大学获得硕士学位，2003年在中国武汉地质大学获得博士学位。

他目前是中国地质大学计算机科学学院的教授。他已在期刊和国际会议上发表了100多篇研究论文。他目前的研究兴趣包括数据挖掘、机器学习和进化计算及其应用。

Borhan Kazimipour (S'12)分别于2007年和2010年获得伊朗设拉子大学的软件工程学士学位和人工智能硕士学位。他目前正在澳大利亚维多利亚州墨尔本市RMIT大学的进化计算和机器学习研究小组攻读进化优化博士学位。

他目前是莫纳什大学信息技术学院的一名讲师

、

墨尔本。他目前的研究兴趣包括进化计算，机器学习，和大数据分析。

卡齐米普尔先生自2012年以来一直是墨尔本皇家理工大学进化计算和机器学习研究小组的成员。

姚欣（M'91-SM'96-F'03）1982年在中国合肥的中国科技大学获得学士学位，1985年在中国北京的华北计算技术研究所获得硕士学位。

1990年获得中国科技大学的博士学位。

1985年至1990年，他是中国科技大学的副讲师和讲师，澳大利亚堪培拉的澳大利亚国立大学和墨尔本的联邦科学和工业研究组织的博士后研究员、

1990年至1992年，他在澳大利亚维多利亚州的新南威尔士大学担任讲师、高级讲师和副教授，1992年至1999年，他在澳大利亚首都堪培拉的澳大利亚国防军学院担任讲师。自1999年以来，他一直是英国伯明翰大学的计算机科学教授（主席），目前是计算智能和应用卓越研究中心的主任。他也是中国深圳南方科技大学计算机科学与工程系的教授。

姚博士是2001年IEEE唐纳德-G-芬克奖论文的获得者。

奖，2010年**IEEE进化计算大会**奖。

2011年**IEEE TRANSACTIONS ON NEURAL NETWORKS**杰出论文奖，以及其他一些最佳论文奖。他是IEEE CIS杰出讲师，也是IEEE **TRANSACTIONS ON EvolUTIONARY COMPUTATION**的前主编。