文章编号:1001 - 9081(2008)02 - 0355 - 05

# 网格环境下分层并行多群体协作 PSO 框架设计与实现

祁 超[1,2], 张 璟[1]
(1. 西安理工大学 计算机科学与工程学院,西安 710048; 2. 陕西师范大学 计算机科学学院,西安 710062)
(zhangjing@xaut.edu.cn; qichao@snnu.edu.cn)

**摘 要**:针对利用广域范围内的计算资源参与 PSO 执行,从而提高工程最优化问题计算效率并降低计算成本,提出一个网格环境下分层并行多群体协作 PSO(G-LPMCPSO)框架。首先给出一个适应负载不均衡和计算资源异构网格环境下的并行多群体协作 PSO(PMCPSO)算法;然后着重阐述了如何利用标准的网格技术和 PMCPSO 算法设计并实现 G-LPMCPSO 框架,该框架具有一个扩展的 GridRPC API 用于隐藏网格环境的复杂性和一个元任务调度器用于无缝的资源发现和选取;最后,根据理论分析及实验结果,证明利用网格技术及 PMCPSO 可以提供一个可靠的框架用于加速解决科学工程最优化问题。

**关键词**:粒子群优化;群体;网格;集群
**中图分类号**:TP393 **文献标志码**:A

# Design and realization of layered parallel multi-swarm cooperative PSO framework under grid environment

QI Chao[1, 2], ZHANG Jing[1]
(1. *School of Computer Science and Engineering, Xian University of Technology, Xi'an Shaanxi* 710048, *China*;
2. *College of Computer Science, Shannxi Normal University, Xi'an Shaanxi* 710062, *China*)

**Abstract:** Concerning how to utilize computational resources in large field to join the implementation of Particle Swarm Optimization(PSO), thereby the computational efficiency can be enhanced and the computational cost can be reduced, the Grid-based Layered Parallel Multi-swarm Cooperative PSO(G-LPMCPSO) Framework was presented in this paper, as well as a Parallel Multi-swarm Cooperative PSO (PMCPSO) algorithm adapted to grid environment of load imbalance and the heterogeneity of computational resources. And then, how to make use of standard Grid technologies and PMCPSO algorithm to design and realize G-LPMCPSO framework was discussed in detail. The framework has an extended GridRPC API to conceal the high complexity of the Grid environment, and a meta-scheduler for seamless resource discovery and selection. At last, the theoretical analysis and the result of experiment indicate that the proposed G-LPMCPSO using Grid can offer a credible framework for providing a significant speed-up to optimization in science and engineering.

**Key words:** Particle Swarm Optimization(PSO); swarm; grid; cluster

## 0 引言

粒子群优化(PSO)是一个随机优化方法,常被用来解决复杂工程最优化问题,如结构和生物力学的最优化[1,2],正在迅速得到越来越多工程研究团体的重视。在过去的几年里,出现了许多变异 PSO,如并行 PSO、多群体 PSO 等,目的是降低高昂计算成本,提高算法性能。但对它们的研究和应用几乎都是在同构计算节点上进行的,而且都局限在小规模区域内贡献出来的计算资源上。那么如何利用分布在广域范围内的实验室或组织的计算资源来进行 PSO 研究和应用? 随着网格[3]出现和发展,这个问题也有了切实可行的解决方案,因为网格就是要为分布式的计算资源和无处不在的服务构建一套开放的标准,网格的巨大发展促进了科学和工程领域中复杂设计问题移植到网格环境下解决的研究和开发[5]。本文将讨论如何利用网格技术和并行多群体协作 PSO 算法设计和实现一个分层并行多群体协作 PSO 框架,用于加速科学工程最优化问题的处理。

## 1 并行多群体协作 PSO

### 1.1 并行 PSO 和多群体协作 PSO

并行同步 PSO 适宜集群环境,而并行异步 PSO 适宜网格环境[6,7]。并行同步 PSO 使用静态负载平衡预先将工作量分配给已经确定的每个处理器,并行异步 PSO 则是将集中任务队列方法同动态负载平衡相结合以减少负载的不平衡,因此适宜在异构的网格中利用性能差异的计算资源来进行并行优化。将主从(master-slave)模式引入 PSO 衍生出多群体协作 PSO[4]。PSO 本质是一种粗颗粒度的优化算法,保证 Particle 的位置($n$ 维向量)作为一个整体被优化,向最优位置前进,但向量中的每维成分并非都在前进。因此,一种细颗粒度的多群体协作优化算法 CPSO-S[8]被提出用于解决这个问题,进一步提高 PSO 算法的性能。

### 1.2 并行多群体协作 PSO

考虑到执行 PSO 算法的计算环境不再是单一集群环境,而是由地域上跨越多个自治域的集群组成的网格计算环境,那么如何将并行机制充分且合理地引入细颗粒度的 PSO 算

**收稿日期**:2007 - 08 - 16;**修回日期**:2007 - 10 - 23。

**作者简介**:祁超(1975 - ),男(蒙古族),青海西宁人,讲师,博士研究生,主要研究方向:网格计算; 张璟(1952 - ),男,陕西宝鸡人,教授,博士生导师,主要研究方向:网格计算。

法成为提升算法性能的关键所在。为此,本文提出了并行多群体协作 PSO 算法(Parallel Multi-swarm Cooperative PSO, PMCPSO)旨在解决这个问题。

细颗粒度的 PSO 算法核心思想是:若一个种群包含 $S$ 个粒子,第 $i$ 个粒子的位置表示为 $n$ 维向量 $\boldsymbol{X}_i(1 \leqslant i \leqslant S)$,那么种群可用一个 $n \times S$ 矩阵表示:

$$\begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1S} \\ x_{21} & x_{22} & \cdots & x_{2S} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nS} \end{pmatrix}$$

将矩阵每一行看作一个子群,那么种群 $S$ 可被分割成为 $n$ 个由 $S$ 个一维向量组成的子群 $P_j(1 \leqslant j \leqslant n)$,其中,$P_j.x_i$ 表示第 $j$ 个子群中的第 $i$ 个粒子。相应地,搜索空间也被分割成为 $n$ 个子空间,子群 $P_j$ 的 $S$ 个粒子相互合作在其子空间内搜寻最优解的同时,还需要其他 $n-1$ 个子群相应粒子的协助,最终由 $n$ 个子群各自的群体最优解 $P_j.Y$ 组成的向量 $[P_1.Y, P_2.Y, \cdots, P_{j-1}.Y, P_j.Y, P_{j+1}.Y, \cdots, P_n.Y]^{-1}$ 即为在种群 $S$ 上搜寻的最优解。本文提出的 PMCPSO 算法是在细颗粒度的 PSO 算法基础上:

1)将并行同步机制引入子群自身优化过程。在网格计算环境中,子群被派遣到某个集群上进行优化,而单一集群提供的计算环境符合并行同步 PSO 达到最优性能的条件。

2)将并行异步机制引入子群间的协作优化过程,即相互提供本子群全局最优解 $P_j.Y$ 去协助其他子群优化。集群与集群间存在计算、负载等性能的差异,因此只有并行异步才能均衡负载,充分利用计算资源。

PMCPSO 算法伪代码如下:

```
master 程序:
Begin
  初始化 n 个 1 维 Swarm: P_j, j ∈ [1..n];
  将 n 个 Swarm 派遣到 n 个集群上开始并行异步优化;
  Do in Asynchronous parallel
    Repeat
      接收 n 个 Swarm 返回的子群当前最优解 P_j.Y;
      用此最优解 P_j.Y 更新种群 S 当前最优解向量[P_1.Y, P_2.Y, ..., P_{j-1}.Y, P_j.Y, P_{j+1}.Y, ..., P_n.Y]^{-1} 中的第 j 维,并将此更新过后最优解向量传送给子群 P_j;
    Until {终止条件满足}输出结果
  End Do in parallel
End
    Swarm 程序:
Begin
  Repeat
    从 master 获取种群 S 当前最优解向量,从中得到其他 n-1 个 Swarm 当前的全局最优解: P_1.Y, ..., P_{j-1}.Y, P_{j+1}.Y, ..., P_n.Y;
  Do in Synchronous parallel
    f(q(j, P_j.x_i))
    /*以并行同步方式在集群内的 S 个节点上分别调用目标函数 f 评价 S 个粒子,计算出它们的适应值*/
    Barrier synchronization                    //设置的同步点
    fitness_value[1..S]; //用数组 fitness_value 接收 S 个适应值
    for i = 1 to S
      if fitness_value[i] < f(q(j, P_j.y_i)) then
        P_j.y_i = fitness_value[i]
      if f(q(j, P_j.y_i)) < f(q(j, P_j.Y)) then
        P_j.Y = P_j.y_i
    end for
    /*更新 P_j.y_i, P_j.Y,其中 P_j.y_i 表示粒子 i 最优解,P_j.Y 表示子群 P_j 全局最优解*/
    在子群 P_j 上执行 PSO 更新;
    将 P_j.Y 返回给 master 程序;
  End Do in parallel
Until
End
```

master 程序负责创建子群,并在子群上开始 slave 程序进行群落优化,然后采用 blackboard 方式与 slave 交互。所有 slave 都将每次迭代优化信息写在 blackboard 上,并从 blackboard 上读其他 slave 程序写的信息,完成子群间的信息交换。注意,函数 $q(j, P_j.x_i)$ 首先构造一个由 $n$ 个子群全局最优解组合而成 $n$ 维向量,然后再用 $P_j.x_i$ 替换掉第 $j$ 维原始值,最后返回这个向量。

## 2 分层并行多种群协作 PSO 框架

在这一部分,将 1.2 节中提出的 PMCPSO 应用在网格环境中,设计并实现一个基于网格分层并行多种群协作 PSO 框架模型(Grid-based Layered Parallel Multi-swarm Cooperative PSO framework, G-LPMCPSO)。在此框架设计实现过程中,多种网格技术都得到了充分考虑。

### 2.1 支撑 G-LPMCPSO 框架的网格技术

Globus Toolkit[9]、Commodity Grid Kit (CogKit)、Ganglia monitoring tool[10] 和 NetSolve[11] 几种核心网格技术在 G-LPMCPSO 框架设计和实现中有所体现。现有的 GridRPC 标准缺少自动发现和选择网格资源机制,本文提出的 G-LPMCPSO 框架通过包含元任务调度器(meta-scheduler)扩展了现有的 GridRPC API,通过从网格中的计算集群和网格服务那里采集到的在线信息,meta-scheduler 负责执行发现、绑定和负载均衡,而这些在线信息在框架中是由 Globus MDS 和 Ganglia 负责提供,其中,MDS 维护着一个数据库,记录可用资源信息,同时作为一个集中目录服务,跟踪资源用以获取资源在网格中的位置和如何消费资源;而利用 Ganglia 监控和提供可用集群的工作量信息、计算节点和网格服务。一个软件组件必须作为网格服务存在于网格中。因此,为了在基于 CoGKit 下将目标函数作为网格服务,G-LPMCPSO 配备了扩展的 GridRPC API。这种方式可以对终端用户隐藏网格计算环境的复杂性,向上提供抽象向下简化执行。除此之外,还要利用 GSI 在网格中进行安全和授权的访问,利用 GridFTP 来管理各种格式数据的传输。

### 2.2 G-LPMCPSO 框架

G-LPMCPSO 框架主要包括三层:第一层为主程序层(Master-program layer);第二层为网格中间层(Mid-grid layer);第三层为集群内部层(Inner-cluster layer),如图 1 所示。

第一层的矩形框表示着 master 主程序的工作流程。

第二层的矩形框里包含着 $n$ 个集群(Cluster),每个集群中都部署了两个网格服务,子群优化服务(swarm evolution service)和粒子评价服务(particle evaluation service)。利用 Globus 任务提交协议可远程调用子群优化服务完成 $n$ 个子群的初始化和优化任务,在其执行优化的过程中要调用粒子评价服务完成子群 $S$ 个粒子的评价。注意,对于粒子的评价是利用局部集群调度策略(如 NetSolve, Condor)在集群内部的 $S$ 个节点上同步并行计算。

对应第二层中的每个集群,第三层的矩形框表示集群内

部的计算节点。

采用这样的分层方式有着十分显著的现实意义,网格中的集群可能隶属于不同的自治域,它们的硬件性能及软件配置都存在着较大的差异,因此在第一层和第二层之间采用并行异步机制,使得$n$个子群在网格中间层的$n$个集群上并行优化,并异步的与主程序层交互,达到子群之间的协作目的;相反,考虑到每个集群内部节点在硬件和软件配置上的一致性,满足并行同步 PSO 发挥最优性能的条件,因此在集群内部则采用并行同步机制在 $S$ 个节点上完成子群 $S$ 个粒子的评价计算。G-LPMCPSO 框架工作流如图 2 所示。

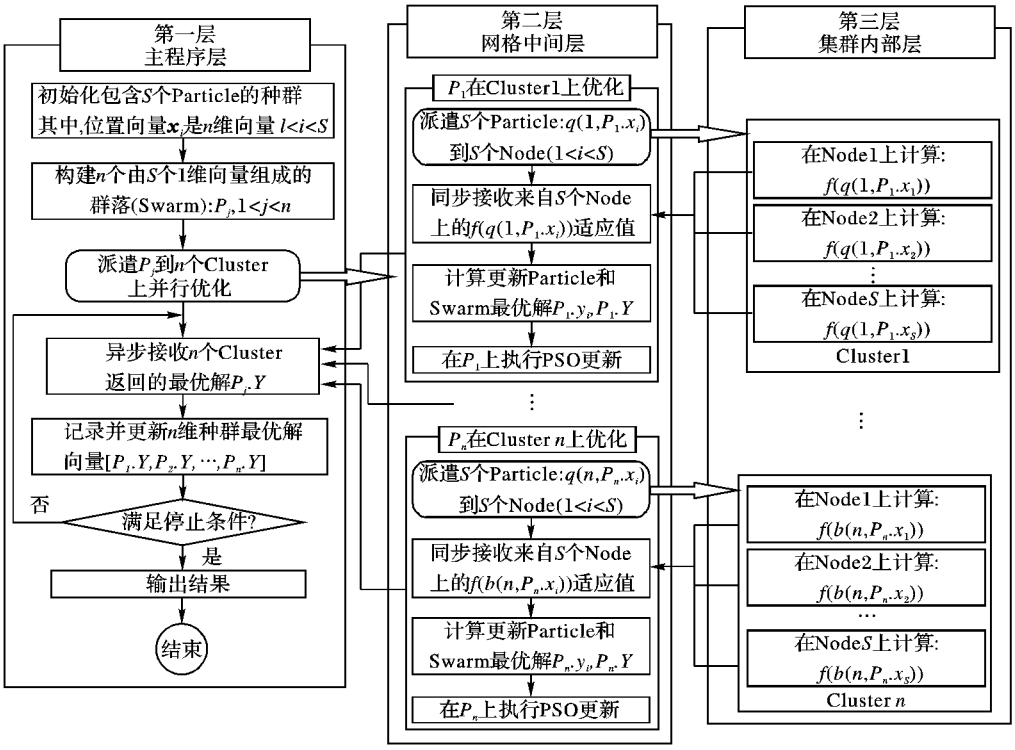


图 1　G-LPMCPSO 框架结构

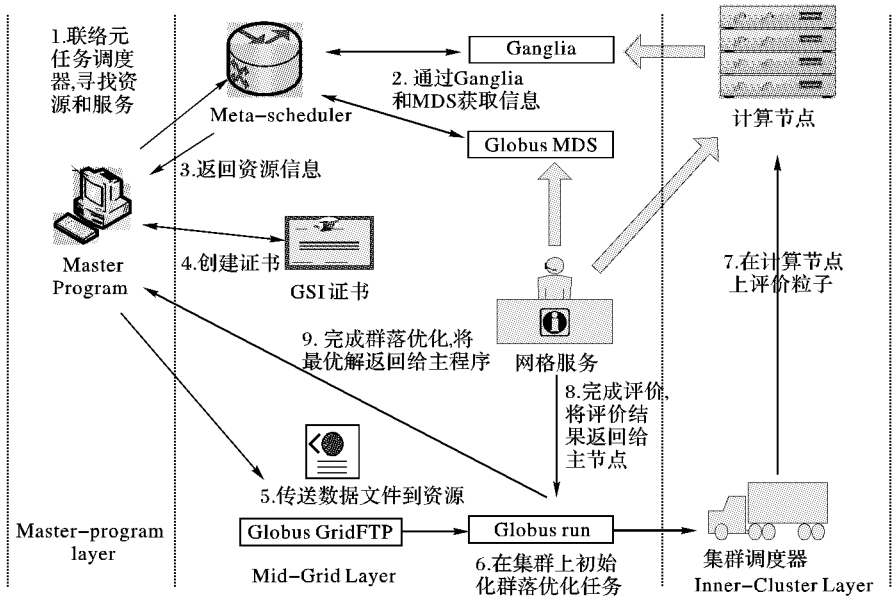


图 2　G-LPMCPSO 工作流

1)准备阶段

首先,主程序联系 meta-scheduler 申请合适的计算节点和请求子群优化服务及粒子评价服务;然后,meta-scheduler 通过 MDS 和 Ganglia 获取可用网格计算资源的列表及其状态信息,以及服务信息,并提供给主程序,用于执行其并行异步优化搜索;再次,产生 GSI 证书,验证并授权主程序使用网格中的计算资源;最后,将已经初始化好的 $n$ 个子群($P_j$,$1 < j < n$;每个子群有 $S$ 个粒子),以 XML 数据文件形式传送到确定的提供子群优化服务及粒子评价服务的远程计算 Cluster 上,并远程调用位于 Cluster 主节点上的子群优化服务,该服务的一个实例将被创建。此时,$n$ 个群体便可以并行开始各自的优化。

2)优化阶段

首先,子群优化服务在利用函数 $q(j,P_j.x_i)$ 装配好种群的 $S$ 个 Particle 的 $n$ 维位置向量后,为每个向量创建一个粒子评价服务实例;然后,粒子评价服务利用 NetSolve 在 Cluster 中

发现可用计算节点，将与之相应的 $n$ 维向量以 XML 数据文件形式传送到这个计算节点上并调用目标函数 $f$ 计算其适应值；等到所有的粒子评价服务结束，$S$ 个评价结果（适应值）返回给子群优化服务，接着子群优化服务更新粒子最优解 $P_j.y_i$ 以及子群的最优 $P_j.Y$；最后，在子群 $P_j$ 上执行 PSO 更新并将子群的最优解 $P_j.Y$ 返回给主程序，与此同时，主程序记录子群 $P_j$ 返回的子群最优解 $P_j.Y$，并更新当前群体最优解向量 $[P_1.Y, P_2.Y, \cdots, P_n.Y]^{-1}$。

## 3 G-LPMCPSO 性能研究

### 3.1 G-LPMCPSO 性能分析

在第一、二层间的通信采用并行异步点到点通信方案，通信时间涵盖在 Swarm 的优化时间中。因为主程序一次只能同一个 Cluster 通信，所以每个 Cluster 在完成一次子群优化后等待与主程序连接时，须等待一段很短的时间，这段时间相对于在 Cluster 上进行的子群优化的时间开销可以忽略不计，因此主要的通信时间是在初始化时将 $n$ 个 Swarm 派遣到 $n$ 个 *Cluster* 上所耗费的时间 $O_{\text{inner}}$。而在 G-LPMCPSO 的第二、三层间采用并行同步 PSO 机制，其通信时间 等于利用 Netsolve 将 $S$ 个粒子并行传送到 Cluster 内部的 $S$ 个节点上所花费的时间。考虑下述两种情况：

1）$n$ 个子群在同一个 Cluster 上串行优化，耗费的时间为：$T_s = nK(SO_{\text{inner}} + wF)$。其中，$w$ 是 Cluster 的并行因子（群体大小及 CPU 的性能指标等），$F$ 是完成一次粒子评价的所需时钟时间，$n$ 是 Cluster 的数目，$K$ 是优化次数。

2）$n$ 个子群在 $n$ 个 Cluster 上并行优化，耗费的时间为：$T_p = \sum_{i=1}^{n} O^i_{\text{cluster}} + K(SO_{\text{inner}} + wF)$。令 $t = SO_{\text{inner}} + wF$，则串/并行模式性能比值最大值：$P^{\max} = \frac{T_s^{\max}}{T_p^{\min}} \approx \frac{nKt^{\max}}{nO^{\min}_{\text{cluster}} + Kt^{\min}}$。其中，$T_s^{\max}$ 表示串行执行 PMCPSO 所需的最大时钟时间；$T_p^{\min}$ 表示并行执行 PMCPSO 所需的最小时钟时间；$t^{\max}$ 表示网格中参与优化最慢的 Cluster 完成一次粒子评价所需的最大时钟时间；$t^{\min}$ 表示最快的 Cluster 完成一次粒子评价所需的最小时钟时间。为了保证任何可能的加速，须 $P^{\min} > 1$，即：

$$P^{\min} = \frac{T_s^{\min}}{T_p^{\max}} > 1 \Rightarrow \frac{nKt^{\min}}{nO^{\max}_{\text{cluster}} + Kt^{\max}} > 1 \Rightarrow$$

$$t^{\min} > \frac{O^{\max}_{\text{cluster}}}{K} + \frac{t^{\max}}{n} \tag{3}$$

通过上述分析，若实际粒子评价计算时间 $t$、算法优化次数 $K$、集群数目 $n$ 和通信开销 $O_{\text{cluster}}$ 满足式（3），G-LPMCPSO 框架能够加速最优化。

### 3.2 模拟实验

**表1 网格计算实验环境**

| 集群(Cluster) | CPU 数目 | CPU 类型 | Memory |
|---|---|---|---|
| 本校教师实验机房 CCS_Lab | 30 | P4 782 MHz | 1 GB |
| 本校网络信息中心 NIC_Center | 12 | Xeon 3.0 GHz | 4 GB |
| 本校高性能实验中心 CHEM_Center | 16 | Xeon 2.6 GHz | 2 GB |

我们在网格环境下对 G-LPMCPSO 进行实验测试，实验环境如表 1 所示。物理上，CCS_Lab 和 NIC_Center 部署在长安校区，CHEM_Center 部署在雁塔校区，其中 CCS_Lab 是由 30 台 P4 计算机组成的集群，NIC_Center 和 CHEM_Center 分别是由 12、16 个计算节点组成的机架式集群服务器，它们的处理速度、处理器数目及负载强度都不同，因此可以实际地模拟分布式异构网格计算环境。

首先将 PMCPSO 中的异步机制更改为同步机制，然后分别在 1、4、8、16 和 32 个处理器上执行 PMCPSO 和更改后的算法，每种情况都各自进行 10 次独立实验，结果如图 3，PMCPSO 算法提供了更为理想的并行加速及并行效率。随着处理器的增加，其效率也在不断提高，符合并行异步 PSO 的性能趋势，而 PMCPSO 采用主从模式是提高并行效率的另一个原因，主处理器只负责与从处理器交互，而所有的优化计算都由从处理器完成，所以当处理器数目较少时，主处理器没有充分被利用，并且占总数的大部分处理器也没有满负载的工作，随着处理器数目增加，占总数的大部分处理器都被用来做计算，因此效率也就不断提高。

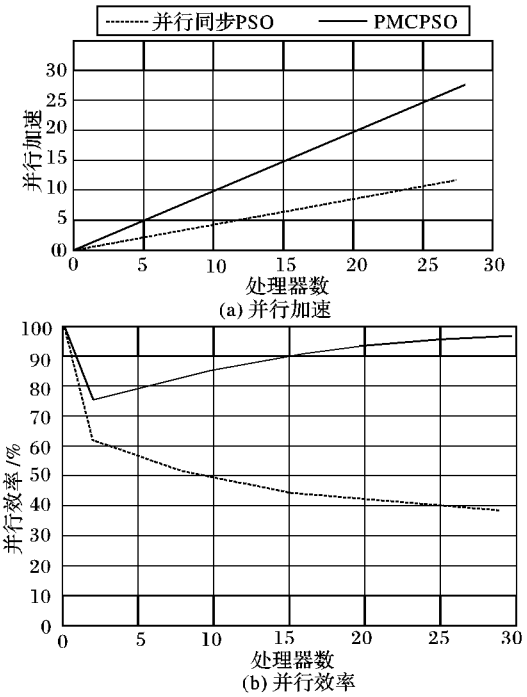


图3 PMCPSO 算法的并行效率和并行加速比较

借助于非线性基准函数 Rosenbrock：$f(x) = \sum^{n} 100 \times (x_{i+1} - x_i^2)^2 + (1 - x_i^2)$ 来衡量 G-LPMCPSO 在网格环境中的性能。函数全局极小值被设计出现在最开始：$x_i = 1$，$f(x) = 0$。初始化 G-LPMCPSO 参数：$c_1 = 1.49$，$c_2 = 1.49$，$w \in [0.9, 0.6]$ 随时间线性递减，$x^{\max}$ 与 $v^{\max}$ 都为 100。

**表2 $m$ 个子群在 $n$ 个集群上最优化 Rosenbrock 问题所需时钟时间 $\times 10^4$s**

| 运行情况 | Swarm($m$) | | | | |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 |
| A: | 0.35 | 0.40 | 0.50 | 0.73 | 0.95 |
| B: | 0.62 | 0.80 | 0.95 | 1.20 | 1.46 |
| C | 0.95 | 1.47 | 1.80 | 2.63 | 3.00 |
| D | 0.42 | 0.50 | 0.64 | 0.90 | 1.25 |

我们分别在：A—在采用高安全性通信协议的多集群网格环境中应用 G-LPMCPSO 并行执行 Rosenbrock；B—在采用低安全性通信协议的多集群网格环境中应用 G-LPMCPSO 并行执行 Rosenbrock；C—在单一集群网格计算环境中应用 G-LPMCPSO 串行执行 Rosenbrock；D—在单一集群网格计算环境中应用 G-LPMCPSO 并行执行 Rosenbrock；四种情况所需时间如表 2 所示。

如图 4 所示,可以看出无论采用高或低安全通信协议,G-LPMCPSO的执行效率都要比串行要高,同时可以看出采用高安全性通信协议所付出的通信代价导致情况 D 比情况 B 效率高,这同 3.1 节的理论分析结论是相符的,即随着集群数目的增加问题的优化不一定始终在加速,因为还有通信开销的因素。也正是这个原因,情况 A 始终比情况 B 效率高。

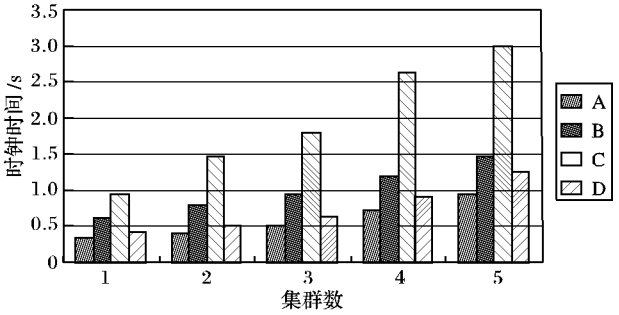


图 4　网格环境中应用 G-LPMCPSO 执行 Rosenbrock 实验结果

## 4　结语

基于网格技术本文设计并实现了一个网格环境下分层并行多种群协作 PSO 框架。通过扩展 GridRPC API 和元任务调度器实现动态资源发现,并对用户屏蔽复杂的网格环境,这个框架为并行优化问题的快速解决提供了一个广泛解决方案。最后通过运用基准函数对 G-LPMCPSO 进行实验分析,结论进一步确认该框架能够加快复杂工程最优化问题。

**参考文献:**

[1] REINBOLT J A, SCHUTTE J F, FREGLY B J, *et al*. Determination of patient-specific multi-joint kinematic models through two-level optimization[J]. Journal of Biomechanics, 2005, 38: 621 – 626.

[2] VENTER G, SOBIESZCZANSKI - SOBIESKI J. Multidisciplinary optimization of a transport aircraft wing using particle swarm optimization[J]. Structural and Multidisciplinary Optimization, 2004, 26(1/2): 121 – 131.

[3] FOSTER I, KESSELMAN C. The grid: blueprint for a new computing infrastructure[M]. San Francisco: Morgan Kaufman Publishers, 1999.

[4] NIU B, ZHU Y L, HE X X. Multi-population cooperative particle swarm optimization[C]// 8th European Conference, ECAL 2005. Canterbury: [s. n.], 2005: 874 – 883.

[5] ONG Y S, KEANE A J. A domain knowledge based search advisor for design problem solving environments[J]. Engineering Applications of Artificial Intelligence, 2002, 15(1): 105 – 116.

[6] SCHUTTE J F, FREGLY B J, HAFTKA R T, *et al*. A parallel particle swarm algorithm[C]// Proceedings of the 5th World Congress of Structural and Multidisciplinary Optimization. Milano: Italian Polytechnic Press, 2003.

[7] KOH B, GEORGE A D. Parallel asynchronous particle swarm optimization[J]. International Journal for Numerical Methods in Engineering, 2006, 67: 578 – 595

[8] VAN DEN BERGH F, ANDRIES P. Engelbrecht. A cooperative approach to particle swarm optimization[J]. IEEE Transaction On Evolutionary Computation, 2004, 8(3): 225 – 239.

[9] FOSTER I. The globus toolkit for grid computing[C]// Proceedings of the 1st International Symposium on Cluster Computing and the Grid. Brisbane: IEEE Computer Society Press, 2001: 2.

[10] MASSIE M, CHUN B, CULLER D. The ganglia distributed monitoring system: design, implementation, and experience[R]. Berkeley: University of California, 2003.

[11] AGRAWAL S, DONGARRA J, SEYMOUR K. NetSolve: past, present, and future; a look at a grid enabled server[M]. New York: Wiley Publishing, 2002.

(上接第 338 页)

算法能够较为准确地估计标签数目:在识别少量标签时,识别时间明显缩短,能够给标签数据的读取、写入操作提供较宽松的时间,当标签数增加时识别时间线性增加;系统吞吐率能很快达到 30%,经过小幅抖动后稳定在理论最大值下方。同时,识别时间成分分析表明,缩短读写器命令传输时间能进一步减少识别时间。

作为一种无线通信中的目标数量估计算法,BDE 算法也可用于 Ad Hoc 网络中的 MAC 协议,如 E-TDMS、RR-ALOHA、TBMAC 及 HRMA。

**参考文献:**

[1] FINKENZELLER K. RFID handbook: radio-frequncy identification fundamentals and applications in contactless smart cards and identification [M]. 2nd ed. New York: Jhon Wiley, 2003: 198 – 202.

[2] CAPETANAKIS, J I. Tree algorithms for packet broadcast channels [C]// IEEE Transactions on Information Theory. Munich: IEEE Inc. 1979, IT – 25(5): 505 – 515.

[3] LAW C, LEE K, SIU K Y. Efficient memoryless protocol for tag identification [C]// Proceedings of the 4th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications. Lyons: [s. n.], 2000, 4: 75 – 84.

[4] JACOMET M, EHRSAM A, GEHRIG U. Contactless identification device with anti-collision algorithm [C]// IEEE Conference on Circuits, Systems, Computers and Communications. Athens: IEEE Press, 1999, 2: 87 – 91.

[5] International Organization for Standardization. ISO/IEC CD 18000-6, Information technology automatic identification and data capture techniques-Radio frequency identification for item management-Part 6: Parameters for air interface communications at 860 MHz to 960 MHz[S], 2002.

[6] VOGT H. Multiple object identification with passive RFID tags[C]// IEEE International Conference on Systems, Man and Cybernetics. Milan: IEEE Press, 2002, 3: 6 – 9.

[7] ANDREW S. Tanenbaum computer networks [M]. San Francisco: PrenticeHall, 1996: 78 – 79.

[8] VOGT H. Efficient object identification with passive RFID tags [C]// Proceedings International Conference on Pervasive Computing. Amsterdam: Springer-Verlag, 2002, 2: 98 – 113.

[9] 天津大学概率统计教研室. 应用概率统计[M]. 天津: 天津大学出版社, 1997: 129 – 130.