分 类 号 _____ 密　　级_____

U　D　C _____ 单位代码 10151

大　连　海　事　大　学

硕士学位论文

基于 Hadoop 的遗传算法在 TSP 中的研究

曹立禄

指 导 教 师	薛大伸	职 称	教授
学位授予单位	大　连　海　事　大　学		
申请学位类别	工学硕士	学科（专业）	管理科学与工程
论文完成日期	2016 年 12 月	答辩日期	2017 年 3 月 4 日
		答辩委员会主席	

Research on Genetic Algorithm in TSP based on Hadoop

A thesis Submitted to

Dalian Maritime University

In partial fulfillment of the requirements for the degree of

Master of Engineering

By

Cao Lilu

(Management Science and Engineering)

Thesis Supervisor： Professor Xue Dashen

Dec. 2016

大连海事大学学位论文原创性声明和使用授权说明

原创性声明

本人郑重声明：本论文是在导师的指导下，独立进行研究工作所取得的成果，撰写成硕士学位论文“基于 Hadoop 的遗传算法在 TSP 中的研究”。除论文中已经注明引用的内容外，对论文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本论文中不包含任何未加明确注明的其他个人或集体已经公开发表或未公开发表的成果。本声明的法律责任由本人承担。

学位论文作者签名：_____

学位论文版权使用授权书

本学位论文作者及指导教师完全了解大连海事大学有关保留、使用研究生学位论文的规定，即：大连海事大学有权保留并向国家有关部门或机构送交学位论文的复印件和电子版，允许论文被查阅和借阅。本人授权大连海事大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，也可采用影印、缩印或扫描等复制手段保存和汇编学位论文。同意将本学位论文收录到《中国优秀博硕士学位论文全文数据库》（中国学术期刊（光盘版）电子杂志社）、《中国学位论文全文数据库》（中国科学技术信息研究所）等数据库中，并以电子出版物形式出版发行和提供信息服务。保密的论文在解密后遵守此规定。

本学位论文属于：　　保　密□ 在_____年解密后适用本授权书。

不保密□ （请在以上方框内打“√”）

论文作者签名：　　导师签名：

日期：　　年　月　日

摘 要

信息爆照时代，数据规模急剧增加，大数据分析计算平台应运而生。以遗传算法为代表的智能算法，具有并行运行的特点，在处理多条件、多约束及非线性等实际问题中所起的作用越来越明显。本文着重研究如何将遗传算法在大数据分析计算平台上进行有效的运行，并在 TSP（旅行商问题—Travelling Salesman Problem）问题中实例化文章提出的混合并行遗传算法（HPGA，Hybrid Parallel Genetic Algorithm）。

本文主要研究工作和成果如下：

（1）讨论遗传算法并行的可能性。由于遗传算法天生具有并行运行的属性。因此，如何提升其并行的效果是本文讨论的重点。学术界，已经有很多专家学者对遗传算法并行的可能性做了相关的研究。本文在前人研究的基础上，对研究成果进行充分的讨论，以确保优化遗传算法的并行特性。

（2）建立基于 Hadoop 的混合并行遗传算法模型。基于遗传算法并行可行性的讨论研究，本文构建了基于 Hadoop 的混合并行遗传算法模型。将不同的并行遗传算法模型进行合理的整合，使其具有更好的伸缩性，提高混合并行遗传算法模型在面对实际问题求解过程中的适应能力。在提高求解效率的同时，又能很好保持遗传算法的优秀特性。

（3）模型框架的实例化和方法研究。本文选取 TSP 旅行商问题实际背景，以混合并行遗传算法为基础，对提出的算法模型进行实例化验证，通过实验数据证明本文提出模型框架的可用性。

关键词：Hadoop；遗传算法；混合并行遗传算法；大数据；TSP 问题

ABSTRACT

Information explosion era, the rapid increase in the size of data, Big data analysis and computing platform came into being. The intelligent algorithm, which is represented by genetic algorithm, has the characteristics of parallel operation, and it plays a more and more important role in dealing with many practical problems, such as multi condition, multi constraint and nonlinear. This thesis focuses on how to use genetic algorithms analysis run on the big data computing platform effectively, and instantating hybrid parallel genetic algorithm(HPGA, Hybrid Parallel Genetic Algorithm) proposed by this thesis in the TSP (traveling salesman problem - Travelling Salesman Problem).

The main research work and achievements are as follows:

(1) Discuss the possibility of parallel genetic algorithm. Because the genetic algorithm has the property of parallel operation. Therefore, how to improve the parallel effect is the focus of this thesis. In academic circles, many experts and scholars have done some research on the possibility of parallel genetic algorithm. On the basis of previous studies, this thesis makes a full discussion of the research results to ensure the optimization of the parallel characteristics of genetic algorithm.

(2) To establish a hybrid parallel genetic algorithm model based on Hadoop. Based on the discussion of the feasibility of parallel genetic algorithm, a hybrid parallel genetic algorithm model based on Hadoop is constructed in this thesis. The parallel genetic algorithm model can be reasonably integrated to make it more flexible and improve the adaptability of the hybrid parallel genetic algorithm in solving the practical problems. At the same time, it can improve the efficiency of the genetic algorithm.

(3) Instantiation and method of model framework. In this thesis, based on the parallel genetic algorithm, the TSP problem is used to verify the proposed model, and the experimental data are used to prove the availability of the model framework.

Keywords：Hadoop; Genetic Algorithm; Hybrid Parallel Genetic Algorithm; Big Data; TSP

目 录

第 1 章 绪论

1.1 研究背景

近年来，计算机技术和互联网技术得到了极大的发展。国家总理李克强在国家创新技术大会上创新性的指出，要让互联网技术同传统行业相结合的发展，提出互联网加的战略。但同时，随着技术的高速发展，数据量也在急剧增加，从互联网刚起步的时候以 M 字节作为计算单位，到 G 字节作为计算单位，在数据量上有了质的提升。但是现如今要用 T 级别甚至 P 级别才能描述当今数据量的庞大[1]。因此，如何更加高效、更加快速的对这些数据中进行分析处理，从而挖掘出对我们的生产发展有价值，可利用的知识，成为了当今国家各类研究机构以及企业讨论的热门话题。为了解决这种局面，云计算技术的出现及兴起顺应了时代的需求和发展。在传统的数据处理框架中，一般海量的数据都是由大型机来处理的[2]。这种处理框架的缺点在于成本较高，维护比较复杂，并且很难扩展。而云计算一改传统的大型服务器处理的方式，所提倡的分布式计算框架和并行分析计算方式能够充分利用廉价的服务器，并且具有维护方便，极易扩展的优势。目前最具有代表性的大数据处理框架是 Apache 的 Hadoop 大数据分析计算平台，其特有的文件存储系统 hdfs 能将大数据按照一定机制分别存储在不同的服务器节点上，并且有多个备份。在数据存储速度以及容错性上有极大的优越性，并且其并行处理框架 MapReduce 可以充分利用各个廉价服务器节点的性能，使得对大数据的并行处理成为可能。同时也使得以前需要大量计算而不能推广的算法求解框架也变成了现实，比如遗传算法。

遗传算法[3]的编程思想来自于生物群体的遗传变异和进化规律。生物进化的目的在于保持种群的稳定和增加种群的适应性，以达到能与生存环境和谐发展。对于种群进化中的遗传变异属性则使种群具有优胜劣汰的属性，保证种群的繁荣发展。种群在进化的过程之中，每一次进化都将最适应环境的个体保留下来，而不适应环境的个体将遭到淘汰，这样保持了种群的最优解和适应度总是处于最优水平。因此根据遗传算法的特点，它非常适合于解决大规模求解问题，比如集群作业调度、非线性求解问题等。另外，由于遗传算法特有的并行计算的特性，使得它可以通过群体分组的方式进行计算求解。

从上述论述可知，遗传算法有着得天独厚的优势可以运行在目前最流行的并行分析计算框架 Hadoop 平台上。如今，在信息数据急剧增加的时代，很多运行良好的传统系

统也逐渐面临着数据量过多无法有效处理的瓶颈。另一方面，对于数据分析统计也成为当前的热门话题，比如用户行为分析，决策支持系统等等。因而大数据处理平台的出现结合传统的算法模型使得需要进行大量数据处理的大规模求解问题，成为可能。另外，随着我国社会经济的快速发展及其带来的数据量和约束条件的增多。将传统算法结合大数据处理平台进而解决当前传统求解框架无法应对的问题，也是适应时代进步的需要。

1.2 国内外研究现状

自从遗传算法被提出以后，其在应用也随着历史的发展越来越广泛，前人们也在对其算法架构进行越来越深入的研究及应用，使得遗传算法的内容变得越来越丰富。不仅如此，其学术研究也随着其关注度的提升变得越来越热门，国内外涌现出了一批又一批的研究算法研究成果。

1.2.1 国内研究现状

在 2006 年南洋理工大学计算机系的 Dudy Lim 及其研究团队在其发表的一篇论文中提出了 GE-HPGA(Grid-Enabled Hierarchical Parallel Genetic Algorithm)算法及其实现模型，此模型是针对网络计算所提出的，并基于标准的网络计算模型所实现。这篇论文中的模型有两个典型的特点，第一个是扩展了 GridRPC API 中复杂网络的计算环境，第二个则是无缝的资源发现和选择调度器。

在 2009 年伊利诺伊大学计算机系的 Abhishek Verma 等研究人员在其一篇发表的论文中基于 MapReduce 的实现机制提出了实现思想及其模型[4]。论文中主要介绍了如何将传统的遗传算法计算过程通过 MapReduce 来进行实现，并且通过采用编程试验的方式证明了遗传算法在 MapReduce 实现的可能性。其主要研究论点在于并行遗传算法模型的提出及其在大数据分析计算平台中的实现。

在 2012 年，来自四川大学的李剑峰和彭舰教授讲改进的遗传算法用在云计算的环境中并加上了很多任务调度[5]的任务。他们提出了一种新的双适应度的遗传算法的仿真实验，很好的证明了这种新的方法不仅可以在短时间内完成任务调度任务，并且可以在平均完成时间上也做了相应的优化。

在 2013 年 3 月份，中科院的两位院士提出来一种全新的编码策略，这种策略创新型的提出了如何将计算机资源作为遗传算法运算过程中的基本单元，从而在此基础上提

出了一种区域杂交算子和变异算子，并且基于 Hadoop 大数据分析计算平台进行研究在大数据平台上的实现也分析。根据大量的实验证明给予 Hadoop 大数据分析计算平台上的进行有效作业调度的可行性。

1.2.1 国外研究现状

早在 20 世纪 50 年代遗传算法被首次提出之后，经过将近十年的时间也就是 20 世纪 60 年代，美国的密歇根大学的 John Holland 教授可是着手从自然行为与人工系统的自适应性方面开始应用遗传算法。一开始，他想创造一种应用程序与机器的理论，想要使得程序和机器人对自然具有相当的适应能力，于是他意识到使用群体方式进行自然搜索、进化、选择和交换的重要性。后来，经过将近十多年的发展，也就是到 20 世纪 70 年代到 20 世纪 80 年代中期的时候，基于逻辑数学和语言智能的人工智能开始兴盛起来，从而很多著名大学的学者开始怀疑基于自然搜索进化的人工智能学说，但是 Holland 教授依旧坚持这一方面的研究。之后 Bagley 发明了“遗传算法”一词并且沿用至今，后来他创新型的提出了一种双倍体的编码[6]方式并对以后的遗传算法的各种算子比如交换算子、变异算子等的发展起到了很大的促进作用。同时，他还敏锐的观察到采用何种方式能方式整个算法早熟的原理，并且发展了自组织[7]的遗传算法概念。

在 1970 年 Cavicohio 教授研究了一种基于遗产算法实现的子程序选择和模式识别问题。在研究模式识别的时候，其采用了一种整数编码[8]的方式，这种编码的方式是可以极大的扩展整个算法的检索空间。并且，与选择测量层的提出使得在整个计算过程中极大的保持了物种的多样性，这种方式同时也是对遗传算法参数进行中心控制的方法。同一年，Weinberg 教授研究了生物体的计算仿真学说，他提出了运用多层遗传算法来进行遗传算法参数自动化的理论。1968-1971 年间 Holland 教授又提出了一种重要的模式理论，他的理论与前两位教授不同，其选择采用二进制编码的方式进行函数的研究及优化，并指出了运用 Gray 码的一些重要优点，他研究了从生物系统引申出的各种不同的选择点交换操作等。但是他的研究成果由于没有对 GA－deception 之类的非线性复杂类型问题进行有效研究，从而不具备充分的说服力。这一年，Holland 的模式理论逐渐发展成熟，但是在编码策略上却出现了延续至今的两个派别，一个是根据模式定理建议少用符号去进行种群编码，另一个则是是数值编码优先和设置适当的精度为准采用一个基金一个参数的方法，并相应的把基因操作改造成适合施术操作的形式，Bosworth，Zoo 和

Zeigler 是后者的开创者。

1957 年是遗传算法研究及发展的重要阶段，因为这一年竖立起了整个遗传算法的两块里程碑。一块是 Holland 出版了其经典的著作《Adaptation in Nature and Artificial System》[9]，这本书凝聚着作者在十几年间对遗传算法的许多思想及其实现，详尽的阐述了遗传算法的理论模式，并对其数学研究奠定了基础，并且发展了一整套模拟生物进行自适应的理论；第二块则是 De Jong 完成了一篇具有重要指导意义的论文《An Analysis of the Behavior of a Class of Genetic Adaptive System》[10]。这篇论文深入领会了模式定理的内涵并在此基础上做了大量严格的计算实验，并给出了非常明确的实验结论。同时，他还建立起了至今为止非常重要的 De Jong 五函数测试平台，定义了性能测评的指标，并在此基础上以函数的优化为例，针对遗传算法的性能和机理进行了详尽的实验研究和分析，因此，他为后来的后继者的工作和研究打下的坚实的基础。1981 年，为了克服 De Jong 的轮盘赌选择[11]中的误差为题，Brindle 教授带领其对六种复制测量又进行大量的研究工作。80 年代之后，以符号系统模拟人工智能的传统研究越来越力不从心，机器学习、神经网络和遗传算法这些从生物系统底层进行智能模拟的算法获得了重生。同年 Goldberg 在对遗传算法的研究中有了重大的发展，他在 1983 发表的一篇博士论文中开创性的把算法用于煤气管道的应用，这同时也是第一次将遗传算法应用于实际的生产问题求解。从此之后，越来越多的将遗传算法应用于实际问题的论文越来越广泛，并且其研究也越来越丰富。从 1985 年起，遗传算法及其应用从此设立了每两年召开一次的国际会议，并且越来越多的人工智能的会议上也逐渐丰富了遗传短发的身影。

九十年代之后，复杂问题作为对象的科学新范式在整个学术界得到来广泛的认可，其内涵主要包括不确定性、非线性、时间不可逆等理论。自从遗传算法提出并经过多年发展之后，其能有效的对 NPC 类型问题的组合优化问题以及多目标、非线性的函数优化问题进行有效的求解，从而得到了科学界的普遍认可和关注。一些数学方面的专家都认识到通过遗传算法的求解框架可以有效的将很多遗留问题进行有效的求解。换言之，生物的进化历程有效的证明了这种方法在很多场景下的能力是凌驾于传统的数学证明能力的，另外，遗传算法在吸收了遗传学、进化论以及生物学的各个最新成果之后在实验中的到了证明和伪证的同时也在进行自身的进化。

到目前为止，在经典的数学界、生物学、物理、化学等各个领域，遗传算法的出色

能力的到了科学家的极大认可和兴趣。基于遗传算法的特性，其能应用的领域也是非常广泛，通过对遗传算法进行有效的设计与分析并进行建模实现，可以对各个领域的发展起到非常大的促进作用。在以后，随着遗传算法的不断深入与发展，其必将取得更大的进步。

随着网络的飞速发展，促进了多约束规则的大数据信息的急剧增加，因此如何处理多约束的信息称为了当今研究的热门话题。这些信息数据的处理只有采用可增长式的处理方式才能适应时代的继续发展。遗传算法的兴起无疑给新时代的信息处理提供了一种高可靠，可并行处理的框架模型。但是大多数的计算算法的研究还处于试验研究阶段，缺乏充足的实践验证，在可增长数据的并行进化算法的研究还有待补充。

在这之后同一年马里兰大学计算机系的 Di-Wei Huang 和其带领的研究团队充分利用了遗传算法在 MapReduce 计算平台的实现求解了销售作业调度[12]问题。并且通过充分的试验证明了其在销售作业调度问题的优越性。

1.3 研究目的和意义

Hadoop 大数据处理平台在当今社会提供了一种新的资源处理方式，它将数据分布式存储在其各个服务器节点上，并就其架构来说具备高可用，高容错，存取效率高、扩展性强等特点。这使得当今计算机技术飞速发展下的数据量急剧增加不再成为阻碍生产力发展的瓶颈。相应的很多经典算法比如本文研究的遗传算法是很多领域问题求解的解决框架。但是很多时候，由于数据量的过大，使得算法求解显得力不从心。传统的做法在于增加计算服务器的性能（比如 CPU，内存等设备），但是这样做的缺点在于增加了维护成本，并且扩展性也很不灵活。

如今很多科学领域的经典求解算法慢慢的已经发展成为大数据的处理问题。例如娱乐圈的用户行为分析，互联网大文本的处理运算，天文学、地理学问题、基因重组，更高层面的军事卫星导航、军事侦察问题。

针对如上提出的各类问题，Hadoop 大数据平台与传统算法模型的结合有着深远意义。它不但可以帮助传统无法解决的问题提供一种有效的解决方案，同时可以为社会的发展和生产力的提高产生重要的推动作用。虽然理论和研究现在还处于技术前沿阶段，投入真实的实践环节还需要一段实践，但是这同时可以为后续的大力研究产生重要的指

导意义。综上所述，将传统的遗传算法与最新规定 Hadoop 大数据分析计算平台相结合进行讨论研究，不仅仅对社会与经济的发展具有理论指导意义，同时也具有相当的现实意义和应用价值。

本文以 Hadoop 大数据计算平台为技术，利用传统遗传算法的并行特性，将其与 Hadoop 大数据分析计算平台相结合，提出了新的遗传算法的求解框架。同时，本文选用传统的 TSP（旅行商）问题作为框架的验证案例，进一步说明两者结合正确性。并且比较传统算法，在遗传算法的全局与局部收敛速度，是否能保持种群物种的多样性，能得到全局最优解而不是局部最优解等方面进行系统的比较说明，证明框架的优越性。

1.4 研究内容和研究思路

本文在查阅大量国内外研究成果及文献之后，首先专注于研究 Hadoop 大数据分析计算平台的实现及其工作原理，并搭建简易的 Hadoop 大数据运行平台进行测试。紧接着对传统遗传算法在解决传统问题领域的应用做了相当的研究，之后对遗传算法在 Hadoop 大数据分析计算平台的上并行的可能性做了深入的论证，并根据传统问题解决方案的不足，提出了在最新的 Hadoop 大数据分析计算平台上搭建本文提出的混合并行遗传算法框架（HPGA）。之后通过求解 TSP 问题并在 E-MapReduce 进行测试运算，以证明了框架的可行性。主要的研究内容如下：

（1）对最新的 Hadoop 大数据分析计算平台及其数据库 Hbase 进行深入研究，并搭建简易的计算平台。

（2）对传统的遗传算法的求解过程做了深入的论述并详细分析了遗传算法并行运算的可能性。

（3）根据对 Hadoop 大数据分析计算平台和遗传算法的深入研究，提出了基于 Hadoop 大数据平台的混合并行遗传算法模型，并通过 TSP 问题进行了编程实现及论证分析。

（4）通过得出的求解数据与传统的求解框架进行对比分析，从而阐述改进后的计算框架的优点。

（5）对最新提出的基于 Hadoop 大数据分析计算平台的混合并行遗传算法求解框架进行总结并提出其中的不足和未来需要优化改进的方面。

本文将传统计算平台上的遗传算法求解问题进行分析研究并将其在最新的 Hadoop 大数据分析计算平台上实现。并通过实验平台数据充分证明了本文研究内容在处理持续增长数据的可行性。具体论文结构如下：

第一章提出本文的研究内容及其所需要用到的技术，并详细阐述了研究内容的国内外研究背景。最后给出了本论文的内容及研究思路安排。

第二章着重研究 Hadoop 大数据分析计算平台以及阿里云推出的基于 Hadoop 平台的大数据分析计算平台 E-MapReduce 服务。同时包括 Hadoop 平台的架构搭建及其 MapReduce 计算核心。

第三章主要是对传统的遗传算法如何进行求解运算的研究，以及对于 TSP 问题进行深入的介绍，并提出传统遗传算法求解思想和框架的不足之处。

第四章为本文的核心章节之一，核心内容主要是根据上述各个章节研究的成果，创造性的提出了基于 Hadoop 大数据分析计算平台的混合并行遗传算法求解框架。

第五章是本文的另一个核心章节，本章主要是将上一章节提出的并行遗传算法求解框架应用在 TSP 问题求解上，并通过 E-MapReduce 大数据平台验证其可行性。在经过反复实验之后得出实验数据，从而验证了本文研究内容的可行性。

第六章主要针对以上的研究内容和思路进行反思和总结，并提出研究过程中的不足以及将来要完成的改进。

第 2 章 Hadoop 大数据平台

2.1 Hadoop 大数据平台背景

Hadoop 大数据分析计算平台[13]的产生为科学计算与生产力的发展起到了极大的促进作用。它使得分布式计算的框架越来越成为主流，在极大的解放计算能力的基础上同时促进了智能搜索算法的发展。本模块主要从 Hadoop 大数据平台的产生背景以及生态圈两方面对 Hadoop 大数据分析计算平台进行介绍。

2.1.1 Hadoop 大数据平台的产生背景

随着互联网技术的迅速发展，数据量的爆发式增长。Google 的传统的搜索存储方式已经不能很好的适应这种环境。因此为了解决这个问题，Google 内部实现了一套针对自身运营和发展的大数据平台，这个平台主要由三部分组成，GFS、MAPREDUCE 和 BIGTABLE[14]。他们分别是分布式的文件存储系统、数据处理框架和基于分布式存储系统的列式存储数据库。

大约 2000 年，谷歌将其三项技术以论文的形式发表出来，Apache 的一位员工 Doug Cutting 基于这三篇论文产生了创建大数据平台的灵感。因此他基于 Apache 已有的项目 Lucune 和 Nutch，加上 Google 大数据平台的实现思想，创建了现在的大数据平台 Hadoop。

Hadoop 大数据平台和 Google 分布式大数据平台一样，也是由三个核心部分[15]组成，它们分别是分布式文件存储系统 hdfs、数据分析框架 MapReduce 和基于分布式文件存储系统的列式存储数据库 Hbase。

传统的数据存储及其分析方式一般是通过大型机进行批量存储，以及通过建立存储索引，链接分析等方式进行数据的处理。这种方式的缺点在于随着数据量的逐渐增长，扩展大型机的存储将越来越困难，维护成本也会越来越高，并且扩展性越来越差，并且对于中小型企业来说很难承受起昂贵的运行和维护费用，从而使得在大数据时代很难发挥出群体的智慧，在一定程度上阻碍了技术的发展。

自从 Hadoop 出现之后，一改传统的数据存储及其分析方式，数据将会以文件的形式分布存储在各个成本较低的服务器上，并且通过其特有的 MapReduce 分析计算框架来进行大规模的数据分析。这样就使得数据量的爆发式增长对于服务器来说将不再成为瓶颈，只需要增加价格低廉的服务器即可进行数据的无限扩展，从而也使得中小型企业

也能很好的参与进大数据时代的数据分析与计算工作。这将为新时代的技术发展提供得天独厚的优势。

2.1.2 Hadoop 大数据平台生态圈

随着 Hadoop 大数据平台的出现，基于 Hadoop 大数据平台的技术如雨后春笋般蓬勃发展，因此出现了基于 Hadoop 大数据平台的生态圈[16]。这些技术极大地促进了大数据平台的发展，同时也极大的增加了用户的使用方便度，使得基于 Hadoop 的大数据平台成为了当今主流的大数据存储和分析计算平台。以下列出集中主要的 Hadoop 生态圈技术。

（1）Pig

Hadoop 的整个运行过程是任务制的，而 pig [17]则是基于 Hadoop 而产生的一种编程语言，它的出现极大的简化了 Hadoop 的任务分发机制。另外，对于半结构化的数据而言，比如日志文件，pig 的出现使其变得更加具有现实意义。同时，pig 的扩展性极强，它可以在日常的编程中自定义添加 java 的数据类型并进行转换。

（2）Hive

对着 Hadoop 的文件式存储系统 hdfs 系统的逐步发展，基于 hdfs 的数据仓库也在慢慢兴起，比如 Hbase [18]。但是如何将传统的结构化比如传统的二维表数据直接映射到数据仓库中是 Hadoop 生态圈亟待解决的问题。Hive [19]的出现则解决了这个问题，Hive 作为一个数据仓库工具的一种实现，其提供了简单的 sql 查询功能，并且可以直接让 sql 语句在 MapReduce 中运行使用。其学习成本不高，可以十分方便的扩展数据仓库的功能。

（3）Mahout

Mahout [20]也是 Apache 的一个开源项目之一，从属于 Apache Software Foundation（ASF），它的核心领域为机器学习中的经典算法的实现。其出现的目的在于帮助开发人员更加方便的构建智能化的应用程序。常见的 Mahout 实现包括聚类分析，频繁子项挖掘，筛选过滤分析等。另外，Mahout 也在部署在云端进行使用，可以很好的同云计算进行集成应用。

（4）Zookeeper

Zookeeper[21]是一个源码开放的、分布式的应用协调服务。它以 Fast Paxos 算法为基础，实现 Hadoop 分布式系统的同步服务，配置维护和命名服务等分布式应用。

2.2 Hadoop 核心组件及其文件系统

Hadoop 大数据大数据分析计算平台并不是单单是一种软件架构，其内部包含非常广泛的概念，其生态圈的应用软件已经具备非常大的规模，但是就其狭义来讲，其主要包括它自己的文件系统、资源调度系统、数据库存储系统以及其特有的并行计算框架。本部分将从 Hadoop 的整体架构来剖析其核心组成部分。

2.2.1 核心组件

从狭义上来区分，Hadoop 的核心组件主要分为三个部分，HDFS（Hadoop Distributed File System）分布式文件系统、YARN[22]（Yet Another Resource Negotiator）资源管理调度系统和 MapReduce 分布式运算框架。其总体架构图如图 2.1[23]所示：

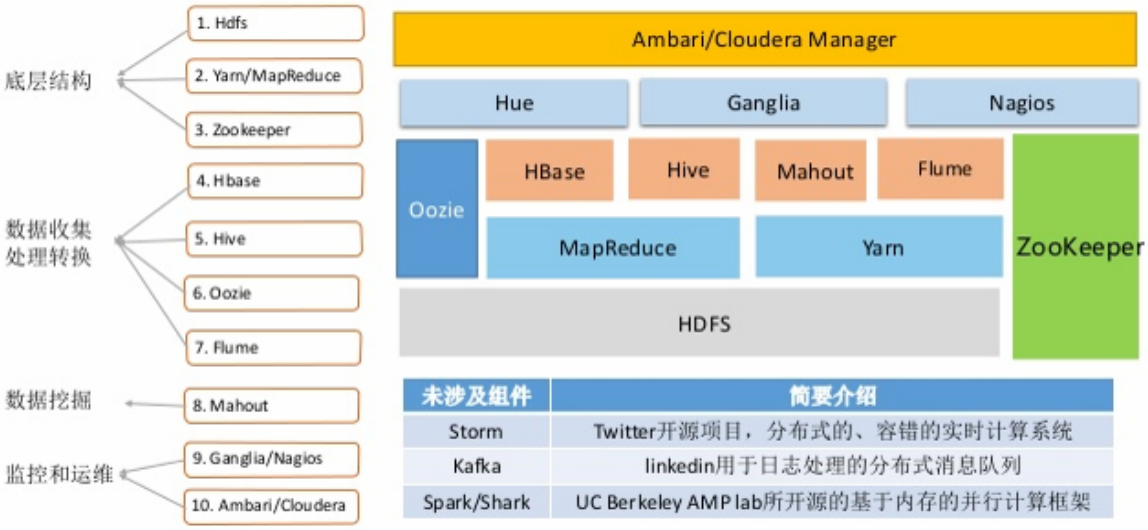

图 2.1 Hadoop 平台架构图

Fig. 2.1 The architecture chart of Hadoop Platform

HDFS：hdfs 是 Hadoop 大数据平台的分布式数据存储系统，这种存储系统与现在传统的存储系统有很大的相似点，但同时，它有自己特有的优势。比如，它有很强的容错性，这使得它可以部署在廉价的机器上。并且，hdfs 提供了可以承载高吞吐量的数据访问能力，这使得 hdfs 非常适合大规模数据的分析和计算。

YARN：yarn 是 Hadoop 大数据平台上的硬件资源协调者，负责平台的硬件资源分配、调控等功能。可以为 Hadoop 的上层应用提供可靠的资源统一调度和管理。这使得 Hadoop 大数据平台在集群使用效率上更加可靠、有效和统一。

MapReduce：是 Hadoop 大数据平台上特有的数据分析计算框架，主要用于 TB 级别的大数据分析计算。这种计算方式极大的简化的编程人员在并行编程方式下的编程和运行。

2.2.2 文件式存储系统 hdfs 及实现机制

Hadoop 的文件存储系统 hdfs 是一个分布式的数据存储系统，它为分布式数据分析提供了最底层的存储基础，也是 Hadoop 大数据平台的核心之一。hdfs 是基于本机文件存储系统的抽象层，如同 Windows 的 NTFS 文件存储格式屏蔽磁盘读写细节一样，hdfs 文件存储系统则屏蔽了各个存储节点的文件存储细节，使得分布式存储如同在一台机器上存储一样。通过分布式存储处理来解决大数据文件的存储问题，具有分布式管理，性能高，容错性高等特点。

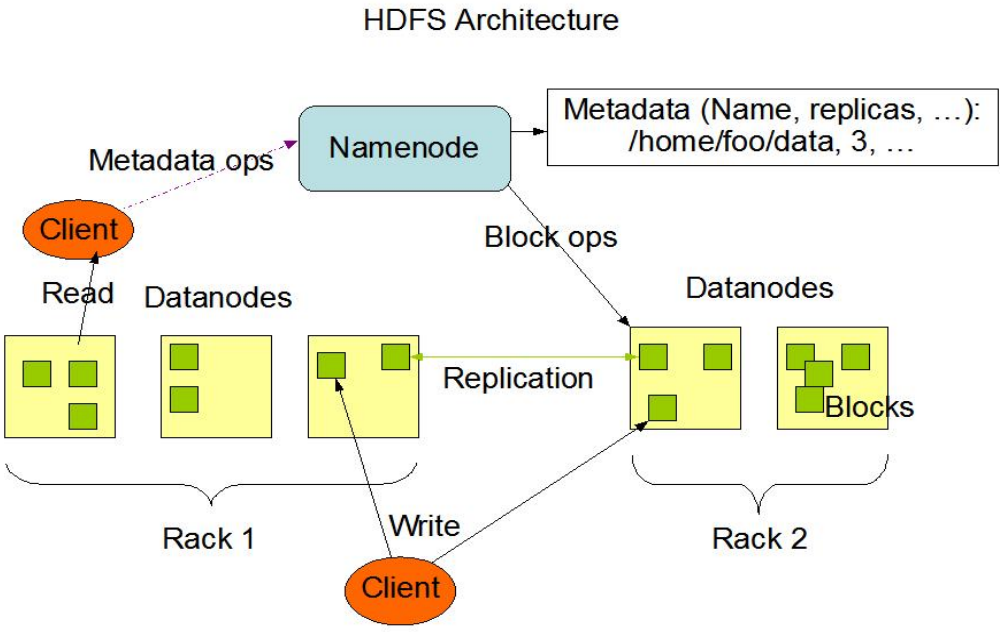

图 2.2 hdfs 存储架构图

Fig. 2.1 The architecture chart of hdfs storage

如上图 2.2[24]所示，整个 Hadoop 的文件存储系统主要是主从结构。具体分为主节点（namenode）和多个从节点（datanode）。其主要负责功能如下：

namenode 节点主要负责接收用户的发出的操作请求、对所存储的文件系统和目录结构索引进行相应的维护工作，另外，对文件系统和 block 之间的关系、block 和 datanode 之前的关系进行管理和维护。

而 datanode 的主要任务则是复制实际的文件存储，保证所存储的文件被切分为 block 存储在相应的磁盘上。此外，这些文件将会被复制成为多个副本，这极大的保证了数据的安全性。

2.2.3 MapReduce 分析计算框架

在传统的计算机的分析运算中，常规的思想是将数“数据”移向“计算”去进行数据的分析和统计，这样做的好处在于实现简单，符合常规的思考方式，从而也使得编程人员容易思考和掌握。但是这种方式的缺点在于，由于现今的 I/O 以及网络传输的瓶颈，这种方式不适合大规模数据的分析和计算，在当今大数据背景下显得力不从心。

而最新的 Hadoop 大数据计算平台一改传统的由“数据”移向“计算”的分析计算方式，演变成了既有传统的计算方式，又结合从“计算”移向“数据”的计算方式。正如其命名，它的计算步骤主要分为两个步骤，Map 数据统计计算和 Reduce 数据分析阶段。

Map 阶段正是从“计算”移向“数据”的分析计算，这种计算方式充分发挥了分布式系统的优越性，可以使得所有的分析计算节点一同进行数据的分析和统计，之后将分析统计的结果，通俗来说，其实就是过滤的结果一同交给接下来的 Reduce 框架。

Reduce 分析计算框架的主要任务是将 map 阶段的计算结果进行数据汇总和分析。如同 map 阶段一样，Reduce 分析计算框架也是由很多节点组成的，在实际的数据分析计算中，可以根据业务的具体需求和环境，将不同的 map 阶段完成的结果分发给不同的 Reduce 节点来进行数据的分析和统计。

MapReduce 的数据流图过程如下图 2.3[25]所示。

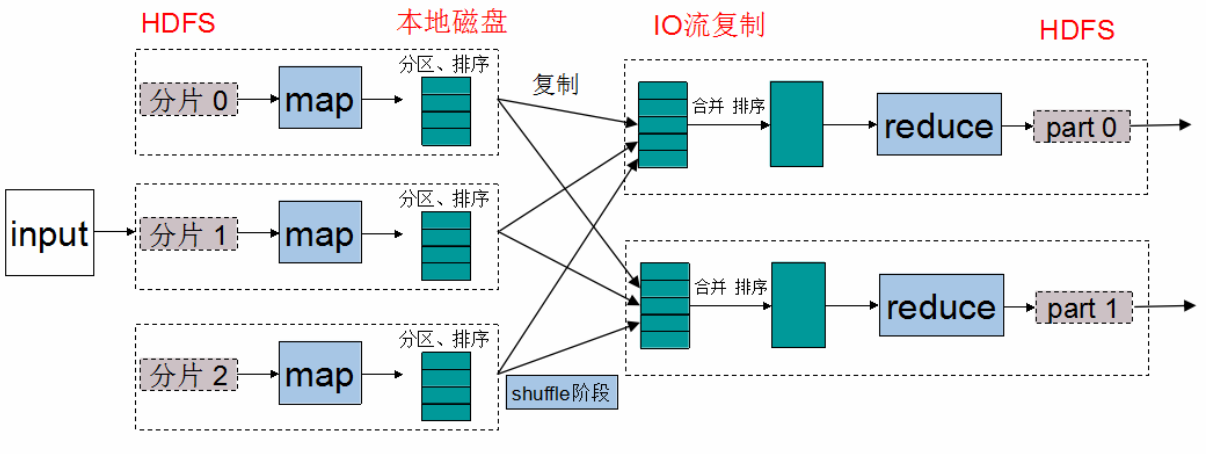

图 2.3 MapReduce 数据流图

Fig. 2.3 The data flow chart of MapReduce

综上所述，Hadoop 的 MapReduce 分析计算框架主要有以下优点：

第一，在实际的业务运行中，它将一般的业务在技术层面划分成了两个部分，Map 数据分析过滤阶段和 Reduce 数据计算统计阶段，这样使得在大数据量的环境下可以进行更加细致的数据处理，并且充分发挥的分布式计算的优点。

第二，MapReduce 分析计算框架将分布式计算中面临的公共问题，比如 jar 包的分发，任务的启动，任务的容错，节点之间的任务调度，中间节点的分组传递等封装成了计算框架。使得编程开发人员无需关注整个框架的底层是如何实现的，只需要在框架层面上专注于业务的开发即可，这样的使得整个开发过程变得更加可靠，节省了大量的数据处理成本和时间。最重要的是，编程开发人员无需过多的考虑此框架是否稳定可靠，因为框架的稳定性和可靠性已经有专门的人员去开发维护。

2.3 E-MapReduce 大数据分析计算平台

在云计算、大数据大力发展的今天，很多公司都退出了自己的云计算[26]和大数据分析处理品台，这无疑是一种将将平台商业化的手段，这同时促进了没有足够的财力物力的小公司在大数据云处理方面的发展，本部分主要介绍由阿里推出的大数据分析计算平台 E-MapReduce，它旨在为中小型企业或是科研单位提供一种更优雅的手段去使用 Hadoop 大数据分析计算平台去进行数据的分析和处理。

2.3.1 E-MapReduce 概述

E-MapReduce 是一项 Web 服务，其简化了大数据处理的流程，提供的大数据框架可以轻松、高效、经济、安全、稳定的处理大数据业务，满足如日志分析、数据仓库、商业智能、及其学习、科学模拟等业务的需求。

阿里云 Elastic MapReduce（E-MapReduce）是运行在阿里云平台上的一种大数据处理的系统解决方案。E-MapReduce 构建于阿里云云服务器 ECS 上，基于开源的 ApacheHadoop 和 Apache Spark，让用户可以方便地使用 Hadoop 和 Spark 生态系统中的其他周边系统（如 Apache Hive、Apache Pig、HBase 等）来分析和处理自己的数据。不仅如此，用户还可以通过 E-MapReduce 将数据非常方便的导入和导出到阿里云或其他的云数据存储系统和数据库系统中，如阿里云 OSS、阿里云 RDS 等。

在日常的大数据应用场景中，当用户想要使用 Hadoop、Spark 等分布式处理系统的时候，通常需要经历如下的步骤：

（1）评估业务特点；

（2）选择机器类型；

（3）采购机器；

（4）准备硬件环境；

（5）安装操作系统；

（6）部署 Hadoop 和 Spark 等 app；

（7）启动集群；

（8）编写应用程序；

（9）运行作业；

（10）获取数据等一系列的步骤。

在这些流程中，真正与用户的应用逻辑相关的是从第 8 步才开始，第 1-7 步的各项工作都是前期的准备工作，通常这个前期工作都非常冗长繁琐。而 E-MapReduce 提供了集群管理工具的集成解决方案，如主机选型、环境部署、集群搭建、集群配置、集群运行、作业配置、作业运行、集群管理、性能监控[27]等。

通过使用 E-MapReduce，用户可以从集群构建各种繁琐的采购、准备、运维等工作中解放出来，只关心自己应用程序的处理逻辑即可。此外，E-MapReduce 还给用户提供

了灵活的搭配组合方式，用户可以根据自己的业务特点选择不同的集群服务。例如，如果用户的需求是对数据进行日常统计和简单的批量运算，则可以只选择在 E-MapReduce 中运行 Hadoop 服务；而如果用户还需要流式计算和实时计算的需求，则可以在 Hadoop 服务基础上再加入 Spark 服务。

2.3.2 E-MapReduce 架构组成

E-MapReduce 最核心也是用户直接面对的组件是集群[28]。一个 E-MapReduce 集群是由一个或多个阿里云 ECS instance 组成的 Hadoop 和 Spark 集群。以 Hadoop 为例，在每一个 ECS 实例上，通常都运行了一些 daemon 进程（如 namenode、datanode、resoucemanager 和 nodemanager），这些 daemon 进程就组成了 Hadoop 集群。运行 namenode 和 resourcemanager 的节点被称为 master 节点，而运行 datanode 和 nodemanager 的节点被称为 slave 节点。

E-MapReduce 的组成实例图（包含 1 个 master 节点和 3 个 slave 节点的 E-MapReduce 集群）见下图 2.4 所示。

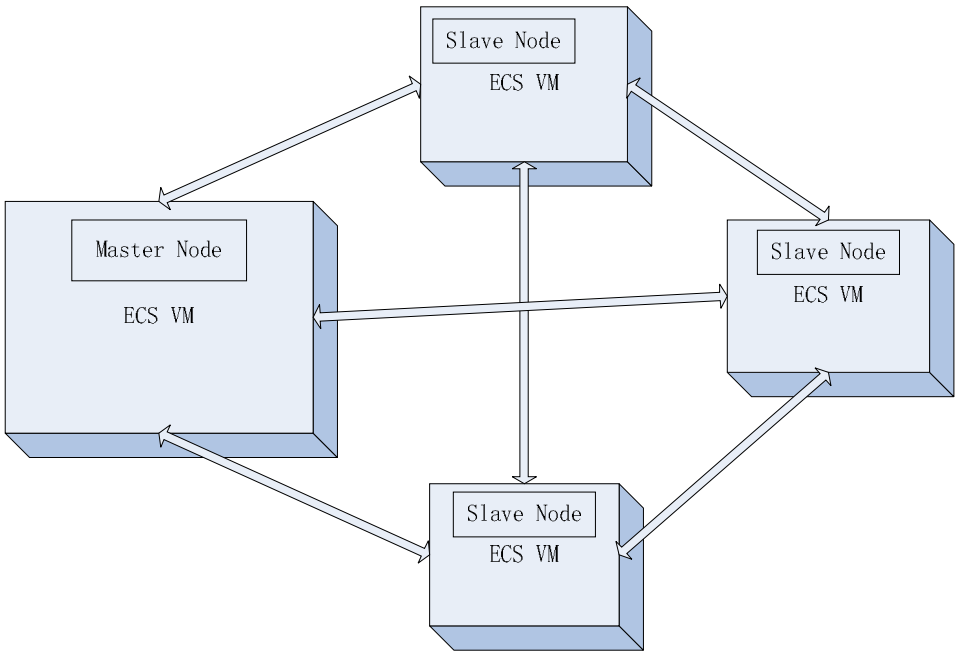

图 2.4 E-MapReduce 组成实例图

Fig. 2.4 The component chart of E-MapReduce

MapReduce 只是 E-MapReduce 整个架构计算框架中的一部分，E-MapReduce 自身是一个非常庞大的架构体系，位于最上层的 Web GUI 和 API 层充分的屏蔽了下层的架构体系，使得用户可以非常方便的使用整个架构，其总体架构体系如下图 2.5 所示。

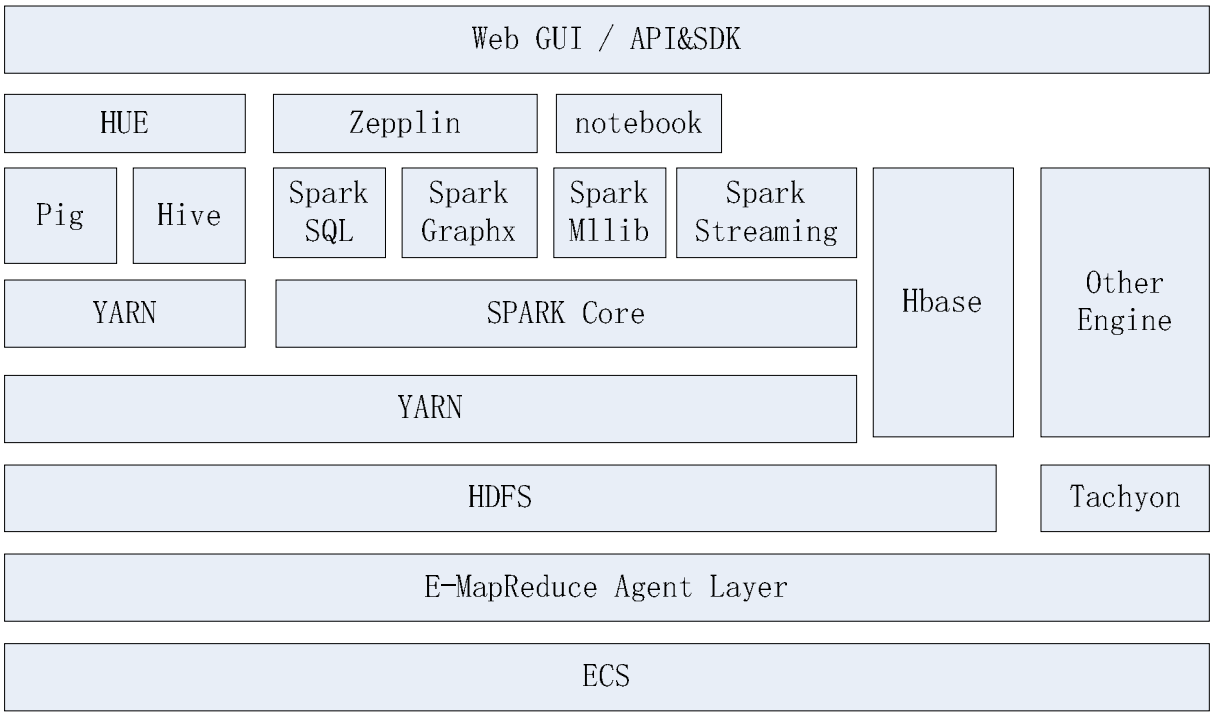

图 2.5 E-MapReduce 架构图

Fig. 2.5 The architecture chart of E-MapReduce

如果所示，我们只需要关注如何去使用整个架构体系所提供的 GUI 和 API 即可，无需关注下层的服务体系。

整个架构体系主要分为四层，最下面一层是阿里云提供的 ECS 大型虚拟机服务，这一层的主要任务是提供整个架构的硬件和系统支持，再上一层是 HDFS 层，这一层则是主要的 Hadoop 操作层，主要绝大多数文件存储服务是在这一层提供的。之后向上是 YARN 层，这一层的主要任务是为 Hadoop 的生态圈提供基础的资源调度服务。接下来的一层则是 Hadoop 的众多的生态圈软件及其服务。最顶层则是上文介绍的提供给用户的图形用户接口，极大地方便用户对其各项服务的使用。

2.3.3 E-MapReduce 优势

相对于自建集群，E-MapReduce 拥有一些比较实用的优势，例如，它能给用户提供

一些非常方便可控的手段从各方面去管理自己的集群。此外，它还具有以下优势：

（1）易用性。用户可简单选择所需 ECS 机型（CPU、内存）与磁盘，并选择所需的软件，进行自动化部署。

（2）用户可以根据自己或数据源所处的地理位置申请对应位置的集群资源。

（3）低价。用户可以按需创建集群，即离线作业运行结束就可以释放集群，还可以动态地在需要的时候增加节点。

（4）深度整合。与阿里云其它产品如 OSS、ONS、RDS、ODPS 等深度整合，使它们可以作为 E-MapReduce 产品中 Hadoop/Spark 计算引擎的输入源或者输出目的地。

（5）安全。E-MapReduce 整合了阿里云 RAM 资源权限管理系统，通过主子账号对服务权限进行隔离。

本文在核心论点在于如何将提出的并行遗传算法在 Hadoop 平台上进行运行，而不是着重与研究如何搭建 Hadoop 平台。同时，自搭建的 Hadoop 平台在高效性、易用性或者是否有其他因素影响整个算法的运算过程也是不可排除在外的。因此本文将选用阿里云推出的 E-MapReduce 大数据分析计算平台作为本文的测试实验品台。这样，可以不必过分关注映射设施和基础架构会对算法的运行带来的影响。只需要关注如何更好的改善本文所提出的算法架构。

2.4 本章小结

本章节先从 Hadoop 生态系统的宏观角度把握什么是最新的 Hadoop 大数据分析计算平台，之后分别从 Hadoop 大数据平台的背景，Hadoop 核心组件及其文件系统和基于 Hadoop 的产品级服务 E-MapReduce 来系统的介绍 Hadoop 分析计算平台的由来及其核心功能。

对于 Hadoop 大数据平台的背景，本章主要做了一个大概的介绍，旨在能对 Hadoop 大数据平台的实际应用能有一个宏观的把握，从而更好的在下文的应用中清晰流畅。

对于 Hadoop 大数据平台的核心组件及其文件系统是本章介绍的重点，原因在于它是一切文件及其数据存储的基础，也是整个架构最底层的部分。只有这一层搭建好，才能让基于其运行的架构软件更加出色的发挥功能。

本章的介绍的另外一个重点是阿里云基于 Hadoop 搭建的产品级服务

E-MapReduce，它充分的评论如何自搭建基于 Hadoop 的大数据分析计算平台的困难，使得用户可以直接将开发好的 MapReduce 应用部署到上面去运行，极大的节省了大量的人力物力财力。更重要的是，我们可以不必再过分关注底层架构给算法框架带来的影响，主要关注算法框架模型的实现。

第 3 章 传统遗传算法及 TSP 问题模型分析

3.1 遗传算法

遗传算法从提出到成熟不过仅仅 50 多年的历史，但是其对社会经济的贡献确实非常可观的，它是受到生物的遗传进化而提出并发展起来的群族算法[29]中的一种智能寻优算法，其整个算法操作过程充满了随机性，但是又能非常神奇的挖掘出问题的最有结果，并且自身带有的并行运算属性使得其在操作过程中能很好的扩展其运算效率，是一种即是从横向角度上的优秀的求解模型，又从纵向角度上的极易扩展的求解模型。

3.1.1 遗传算法简介

遗传算法（Genetic Algorithm）[30]是受生物进化学说和遗传学说的启发发展起来的。在生物进化的过程中主要有以下几个特点：不断繁殖，生存竞争，适者生存和遗传变异。而生物的遗传规律有分离规律和自由组合规律，生物变异的特点有基因重组、基因突变和染色体变异。

而遗传算法则是模仿了生物在整个遗传进化中的这些特点，从而模拟出的一种求解模型。具体来说，遗传算法是一种搜索寻优的技术或者说是一种优化的算法，其原理在于模仿生物遗传和进化，根据生存竞争和优胜劣汰的原则，复制、交换变异等操作，使得要解决的问题从初始解开始一步一步逼近最优解。

遗传算法的研究对象主要是各种非线性的、多变量的、多目标[31]的复杂的自适应系统问题。

3.1.2 遗传算法的发展历史

遗传算法的发展历史向前可以追溯到上个世纪 50 年代，大概有着 60 多年的历史，其具体的发展历史可以分为以下几个阶段：

萌芽期，50 年代后期到 70 年代中期。由一个数学家提出遗传算法。之后随着科学技术的不断发展，美国一位教授 Holland 和其学生对其做了比较大的贡献，在其基础上发表非常多的相关研究论文及专著。

成长期，70 年代中期到 80 年代末期。遗传算法产生了很多固定的算子和技术[32]。也就意味着遗传算法的发展趋向于成熟阶段。其每两年一次的学术交流会也成为其成熟

的标志。并且遗传算法不仅仅停留在数学方面，其在物理领域、生物领域、营销领域等有了长足的发展。对经济、社会产生了重要影响。

发展期，90 年代。遗传算法的发展处于成熟期。其每年的都有定量的论文及专著发表，学术交流也会定期举行。

今后的发展。随着遗传算法的不断发展，其渐渐衍生出来遗传规划的概念，遗传规划[33]在结构化设计、人工智能、复杂问题优化、复杂系统分析等方面有着比较优越的特性。

虽然遗传算法已经比较成熟，但是仍然有许多可以继续改进[34]的地方：

（1）编码技术和程序表达技术的改进。

（2）复制、交换、突变等遗传操作的改进。

（3）适应度的表达和适应度的改进。

（4）寻求其它有效遗传算子，防止近缘杂交、过早收敛。

（5）进一步探讨遗传算法和遗传规划的机理。

（6）开发遗传算法和遗传规划的商业软件。

3.1.3 遗传算法的特点

（1）智能式搜索。依据适应度（目标函数）进行的只能搜索。

（2）渐进式优化。利用复制、交换、突变等操作使下一代结果优于上一代。

（3）全局最优解。采用交换和突变操作产生新个体，使得搜索得到的优化结果逼近全局最优解。

（4）黑箱式结构。根据问题的特征进行编码（输入）和确定适应度（输出），具有只考虑输入与输出关系的黑箱式结构，并不深究输入与输出关系的原因。

（5）通用性强。不要求明确的数学表达式，只需要一些简单的原则要求，可应用于解决离散问题、函数关系不明确的复杂问题。

（6）并行式[35]运算。每次迭代计算都是对群体中的所有个体同时进行运算，是并行式运算方式，搜索速度快。

3.1.4 遗传算法的运算过程

遗传算法的运算过程主要是通过循环语句来控制 5 个主要的步骤。

第一步、编码、随机产生初始群体。这一步是遗传算法求解中非常关键的一步，是所有其他步骤的基础。编码的好坏直接决定遗传算法求解的速度[36]，效率。当编码完成之后紧接着产生初始群体。所有的初始群体是用码的方式表现出来的。

第二步、个体评价、选择、确定是否输出。在初始群体产生之后，要设定一个适应度函数对所有的群体进行评价。检测初始群体中的各个个体是否达到预想的要求。适应度函数是遗传算法中用来判定是否进行下一代循环的判别条件，也是判别是否得到全局最优解的一个判断指标。当要进入下一层循环迭代的时候，要像生物进化一样进行优秀个体的选择，对于优秀的个体还要进行相应的复制，使其在种群中的比例增大，相应的将比较差的个体筛选掉。

第三步、随机交叉运算。由于遗传变异的过程中是存在杂交现象，因此这一步则是模拟遗传变异种的现象。

第四步、随机变异运算。由于在生物的遗传变异过程中同时也存在个体变异的现象，因此在这一步则是模拟生物的进化的变异现象而做出的操作。上一步的交叉 和此步骤的变异运算是遗传算法中产生新解的重要操作，也是防止在计算过程中出现过早收敛，避免只得出局部最优解而不是全局最优解[37]的手段。

第五步、转向个体评价，开始新的循环。这一步是对上一代结果的评价，检查是否达到了预期的输出条件。若没有达到输出条件则进入下一代的循环迭代。

传统遗传算法的流程图如图 3.1 所示：

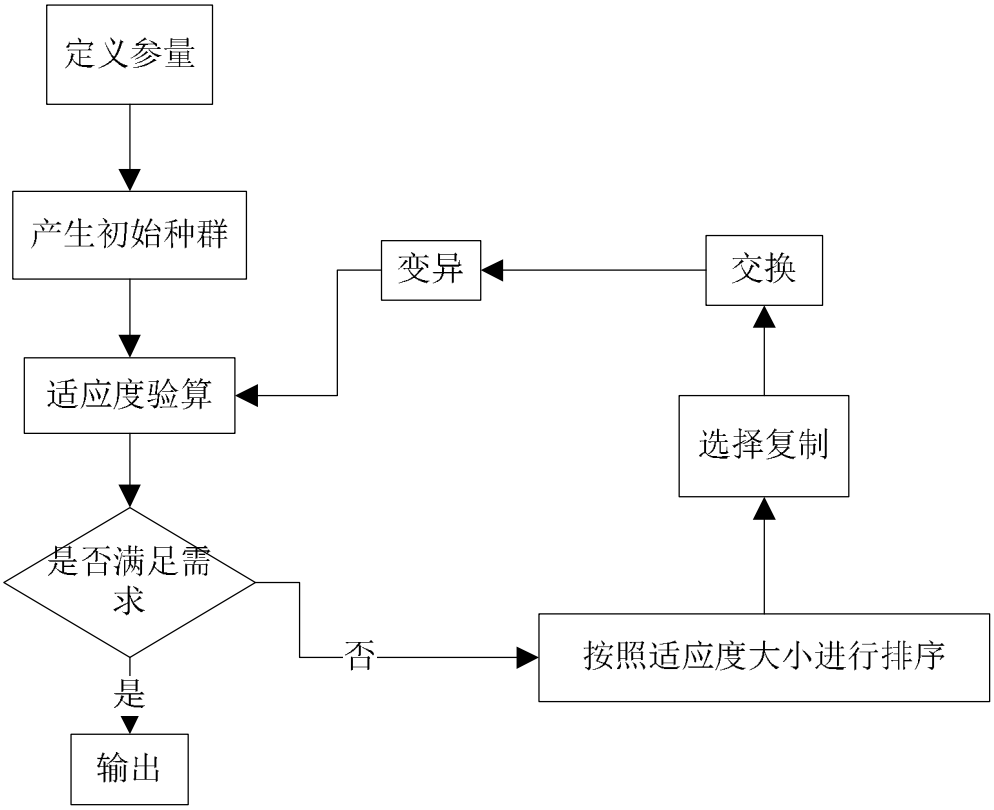

图 3.1 遗传算法流程图

Fig. 3.1 The flow chart of Genetic Algorithm

另外，在整个遗传算法运算的过程中，有许多关键点需要值得注意，比如如何进行编码，个体适应度的计算，以及各个遗传算子的选择等，具体如下：

编码过程。编码过程是整个遗传算法运行的基础，编码的好坏直接决定遗传算法运行的效率是是否能得到全局最优解而不是局部最优解。一般来说，遗传算法工作的对象是字符串和数字。因此对编码的要求要能反应所要研究问题的本质属性，并且所得到的初始编码是否能都便于计算机的运行和计算。

适应度计算。在整个遗传算法的计算过程中，第二步就是计算种群中个体的适应度，适应度的大小直接决定这一个个体是有多大几率会被遗传到下一代。因此适应度是衡量种群个体的优劣的重要尺度，遗传算法中的适应度同时也是驱动整个遗传算法运行的核心动力。

选择操作。选择操作是在种群个体在计算出适应度并根据个体适应度大小进行排序后的一个操作，目的在于选出能都最大程度适应环境的个体进入下一代个体。

复制算子。在进行个体选择之后，接下来就是对复制完毕的个体进行复制操作。这个体现了自然界的“适者生存”的原则。整个复制得数量也是根据种群个体在按照个体适应度进行排序后决定的。因此适应度越大的个体，其复制的数量也就越多，适应度比较小的个体相应的复制的个数也就越少，但是最终结果要保证整个个体的总数是不变的。J.H.Holland 教授提出了一种按照轮盘赌的方式进行个体的复制。这种方式很好的解决了复制所需要的个体数目。

交换算子。整个遗传算法在运算过程中的一个重要操作就是淘汰掉个体之后产生新的个体，而交换操作则是产生新个体的主要办法。这个算子的操作主要是两个个体的一部分字符进行两两交换。整个交换过程中个体的选择是完全随机的。这样极大的保证了整个过程的自然选择的特点。根据遗传算法多年求解的历史经验，一般来说交换的概率一般会定在 50%-80%[38]左右。

变异算子。变异是整个遗传算法运算过程中产生新个体的另一种方法，这个算子借鉴了整个生物在自然选择的过程中的突变现象。这个算子的重要作用在于尽量保持整个遗传算法的计算不会陷入局部最优解而是求得全局最优解。

终止条件。遗传算法的一个重要的特点就是通过反复的迭代和分析计算逐渐逼近整个求解域最优解的过程。因此整个算法必须有个能够进行终止的条件。在整个遗传算法应用的历史中，最常用的终止条件是规定迭代的代数。当整个算法迭代到一定代数的时候进行终止操作。另一种方式是采用控制变差的方式进行算法的终止，比如整个算法的最终的求解结果不会产生很大的变化，这时候我们可以对算法进行终止操作。第三种方式则是检查适应度变化来决定是否对算法进行终止操作。

3.2 TSP 问题及其传统求解模型分析

自从 18 世纪起 TSP[39]旅行商问题被首次提出之后，其一直受到学术界广泛的讨论，对其求解的方法也千差万别，如何能有效的在其给定条件的基础上有效的求解也成为了学术界广泛讨论的话题。原因在于其本质为优化组合问题[40]的同时，其初始解空间会成指数增长，这大大加大了其求解的难度和求解效率。本部分将系统的介绍 TSP 旅行商问题并对其传统求解思想做出相关论述和讨论。

3.2.1 TSP 问题简介

TSP 问题，又名为旅行商问题，是在数据求解模型中的一种著名模型，最早是由 Dantzig 在 1959 年提出的算法求解模型。其算法的基本思路是一位旅行商要向给定的一些城市去推销自己的货物，而城市与城市之间的路程不一样，因此，这位旅行商要选取一个最优的行走方案逐个城市去推销自己的货物。这同时也是一个最优组合问题的原型。

TSP 问题在实际的生产和生活中有着广泛的应用，比如我们日常生活中的物流配送系统[41]。如果在比较的小的国家中，此类问题可能不太突出，但是在我国这种地域范围广，城市数量较多的情况下，一种比较优秀的配送系统模型无疑能为整个物流系统的配送流程节约大量的人力物力成本。

旅行商问题最简单的理解方式为：有一个旅行商，他要到给定数量的城市取推销自己的商品，为了节约人力和物力成本，他要找出一个最短的行走路线去推销自己的商品。TSP 有着非常悠久的历史，最早的描述是在 1759 年欧拉研究一位骑士的周游问题，即对国际棋盘上的 64 个方格，每一个方格都走一次并且最终要返回到起点。后来，TSP 问题由美国的 RAND 公司在 1948 年引入，得益于该公司的声誉以及线性规划这一新方法的出现使得 TSP 成为一个知名且流行的问题。

3.2.2 TSP 问题的传统求解思想

对于旅行商的求解一直以来是研究的热门话题，原因在于随着城市数量的增多，其旅行商的求解组合将是成指数级数量增加，因此，如何构建一个优秀且效率出众的求解算法模型也成为了求解旅行商问题的关键。传统的旅行商求解问题在城市数量较少的情况下构建出了很多经典的模型，当然这些模型也可以用作城市数量较多的情况下，但是运算效率确也是值得深思的一个方面。

对于传统的旅行商求解问题思路主要有以下几种：

（1）分支界限法

基本求解思路：广度优先搜索或者是最小损耗的方式对解空间树的搜索方式是分支界限法求解问题的基本思路。整个问题的解空间树是整个问题解空间的一课有序树，最常见的是子集树和排列树两种。在搜索问题的解空间的时候，回溯法与分支定界法最根

本的区别在于他们对于当前节点的扩展所采用的扩展方式不同。在传统的分支定界法进行分支扩展时，每一个活节点只有一次机会成为扩展结点。当活节点一旦成为扩展结点后，就会一次性的将其子孙节点全部生成。在所有生成的子孙节点中，导致产生不可行解活着导致非最优解的子孙节点将会被舍弃，其余的儿子节点都将加入到活节点列表中。此后，从活节点列表中取出下一个节点作为当前节点并继续进行相应的扩展，重复上述步骤过程。根据上述步骤一直进行搜索并一直找到所需要的最优解或者直到活节点列表为空为止。以下是常用的两种分支定界的方法：

第一种，方式队列（FIFO）[42]分支定界法

这种方式是简单的将整个活节点列表组成一个队列，整个选择的原则是按照先进先出的原则进行选择作为当前的扩展结点。

第二种，优先队列式分支定界法

这种方法是将所有的活节点列表组织称一个具有优先权重的优先队列，并按照优先队列中规定的节点的权重进行选取，优先级较高的优先称为下一个节点的活节点，优先级低的反之。

此类求解算法在进行初步无向图规划模拟的时候计算量一般是比较一般的，但是如果随着城市数量的增多，其生成最小二叉树的过程的计算量将会非常巨大。

（2）动态规划法

对于动态规划[43]求解 TSP 问题的求解思想在于，对于此类问题，旅行商从哪个城市出发对于最终的求解结果都是一样的。因此我们可以假设从第一个城市出发。

（3）近似算法

对于近似算法求解，可以采用一个给定完全无向图 G=(V,E)，并且其假设每一条边 $(u,v)\in E$ 都有一组非负整数的费用。之后我们只需要找出图的最小哈密顿回路即可。其求解最小哈密顿回路的算法可以采用图 G 的最小生成树来求解。

这种近似算法有其自身的优越性，比如在初始节点不多的情况下求解是比较方便的。但是同样面临初始节点过多的情况下，求求解计算过程也是相当大庞大，并且其计算难度也是呈指数增长的。上述描述的采用图 G 的最小生成树的方式来进行求解，我们可以假设初始节点已经达到上百个，这样其求解框架虽然比较容易构建，但是求解的时间复杂度却不是一般的机器可以衡量的。因此此类求解框架在数据量较大的情况下是

有很大的局限性的。

3.2.3 TSP 问题的智能优化算法求解思想

随着社会生产力和科学技术的发展，智能优化算法[44]越来越成为很多经典问题求解的首选方法。智能优化算法有着可扩展性强，求解效率高效，符合人类生产和生活中的理解问题的思路。因此，在在以往很多不容易求解的问题中，智能优化算法发挥了其得天独厚的优势。对于本文的旅行商问题，同样有着非常多的智能优化算法来进行求解，比如常见的模拟退火算法，蚁群算法，蜂群算法以及本文将要提出的遗传算法。对于每种算法的求解思路及其介绍如下。

（1）模拟退火算法

模拟退火算法[45]是一种通用概率算法，其旨在一个给定的大的空间内进行最优解的搜索。该算法的原型是来自于对热力学中退火过程的描述，在某一给定的初始温度下，通过缓慢的降温过程，使得算法在多项式的空间内给出一个近似的最优解。对于 TSP 问题，模拟退火算法可以通过其贪心算法[46]在整个全局范围内进行近似求解。其算法思路如下：

第一步，编码选择。一般来说采用描述 TSP 问题常用的策略即路径编码策略。

第二步，对 SA 状态产生函数设计。由于编码选择策略选取的事路径编码策略，因此可将其设计为三种操作方式即，呼唤操作方式、逆序操作方式和插入操作方式。

第三步，SA 状态接受函数的设计。$\min\{1,\exp(-\triangle/t)\} > random[0,1]$ 准则是一种 SA 接受新状态的条件中最常用的设计方案，△为新旧状态的目标值差，t 为温度。

第四步，初始温度和初始状态设计。在 SA 设计中，最常用的一种初始温度设计方案是，首先随机产生一组状态，之后确定其中任意两个状态之间的最大的目标差值即 $|\triangle\max|$，然后进行公式 $t_0 = -\triangle\max/Inpr$ 的计算，其中 pr 为初始接受概率（理论上应接近 1，但是实际设计时可以取为 0.1。在初始状态可以采用启发式算法求解从而快速得到一个解，并将此作为 SA 的初始状态。

第五步，退温函数设计。对于常规的退火算法求解问题，指数退温函数是最为常规的退温策略（$t_k = \lambda t_{k-1}$，λ 为退温速率）。

第六步，温度修改法则和算法终止条件设计。这里可以采用阀值设计的温度修改和算法终止两个准则。

（2）蚁群算法

蚁群算法[47]的研究依据在于蚂蚁在寻找食物的过程中，会在自己走过的路上留下自己的一种分泌物，我们称之为信息素，这样在一定的范围内当其他的蚂蚁嗅到以前蚂蚁留下的信息素之后，会根据这些信息调整自己的觅食路径。这样，当一条路上的蚂蚁越来越多的时候，其他蚂蚁对这条路的选择概率也越来越大，同时这条路的信息激素也会越来越强。

运用蚁群算法求解旅行商问题就是根据蚂蚁的这种觅食行为而得出的模型。其求解思路主要分为以下几个步骤：

第一，初始化蚂蚁求解种群，即给定一定数量的蚂蚁作为初始求解种群，并且这个种群的蚂蚁数量应该远大于城市的数量。

第二，对其中的每一只蚂蚁选择到达下一个城市，对每一个蚂蚁计算下一个访问城市，访问城市不可重复，直至所有的蚂蚁走完所有的城市。

第三，计算各个蚂蚁经过的路径长度，记录当前问题的最优解，同时更新各个链接路径的信息素浓度。

第四，判断是否达到最大的迭代次数，是则终止，否则重复进行第二步。

从以上的描述中我们可以看出，蚁群算法的核心求解思路其实就是穷举法，这种方法对于小数据量的问题求解具有很高效的求解效率，但不适合于大规模的问题求解模型。

（3）遗传算法

遗传算法是一种模拟大自然自然选择的一种智能优化算法。这种算法的优势在于其模型了整个自然选择的过程，并且整个计算过程中遗传算法是可以并行计算的，因此在遗传算法求解过程中，可以进行大规模的并发计算，从而提高问题求解的效率。对于遗传算法求解 TSP[48]问题的详细过程在此不做过多的赘述。

3.3 本章小结

本章主要介绍了遗传算法和 TSP 问题两个问题，对于遗传算法的介绍，本章节主要

从遗传算法的发展历史作为开端，引出了在整个的遗传算法的发展流程中，遗传算法主要有哪些重大的改进。同时紧接着介绍了遗传算法的算法特点，并且根据这些总结出为什么在现如今科学技术和生产力急剧发展的时代，遗传算法成为了众多热门算法之一。然后介绍了整个遗传算法是如何仿真自然选择过程的，如何对传统的自然选择过程进行了有效的改进，使得遗传算法在求解问题的时候更加有效率，更加符合经典问题的求解模型。同时，对于遗传算法如何符合并行求解框架也做出了简要的介绍，为后文引出混合并行遗传算法求解做了铺垫。

对于本章节的后半部分主要介绍 TSP 旅行商求解问题，主要从 TSP 问题的简介，TSP 问题的传统求解思想及其 TSP 问题的智能优化算法求解模型两个方面进行介绍。详尽的列出了传统求解问题的框架及其思想，并对其各个求解模型的求解思想进行简单的概括说明并进行适当的改进评判。

第 4 章 基于 Hadoop 的混合并行遗传算法模型构建

4.1 基于 Hadoop 的并行遗传算法实现依据

遗传算法在社会的诸多领域有着非常广泛的应用，比如生产调度、图像处理、函数优化等。虽然随着社会的不断发展，数据量的增大，求解问题的复杂性增大，使得传统的算法模型在求解问题上力不从心。由于遗传算法的思想是从种群出发，因此自身有着并行的固有属性[49]，非常适合在并行的分布式的计算机集群上运行，随着并行计算平台的普及也同时为遗传算法的并行计算奠定了基础。比如遗传算法中个体适应度值的计算以及评价可以独立进行，互相之间没有任何依赖，并且计算过程中不需要任何通讯。

在并行遗传算法[50]（PGA）中，要做的不仅仅是要将并行遗传算法变成一种方案，更重要的是修改遗传算法的计算模型，使其能够更好的适应分布式并行计算平台。在并行遗传算法改进的过程中，一是要注意如何将大规模的初始种群进行划分，使得并行计算平台能够更有效率的运行；二是在其选择，交换和变异的过程中如何处理各个种群之间的通讯机制，使得遗传算法在整体运行中处于稳定状态。

以下是遗传算法可以进行大规模并行计算的理论依据。

4.1.1 个体适应度计算的并行性

在遗传算法的运算过程中，个体的适应度计算是非常关键的一步，适应度的计算决定了产生下一代种群个体的选择。但是群体中个体适应度的计算也是整个遗传算法中最消耗时间的一步操作。

但是对于个体的适应度来说，其计算只与所选取的适应度函数有关，和其他因素没有什么大的关系，并且，个体适应度的计算顺序也不会对遗传算法的最终结果产生影响。因此可以将个体适应度的计算通过一定方法进行分组执行，比如在分布式的环境中将群体分成不同的组，然后将各个组分别部署到不同的机器上，这样个体的适应度计算就可以分开进行了，通过这种方法可以提高遗传算法的计算效率。

4.1.2 整个群体中各个个体适应度评价的并行性

在遗传算法的求解过程中，当各个个体的适应度计算出来之后，紧接着就是对个体适应度进行科学的评价，而评价的优劣直接决定下一代群体初始群体的选择。但是对于

种群较多的情况下，对个体适应度进行逐一评价，毫无疑问将非常耗时并且浪费计算机资源。因此如果能通过一定的方法，比如借助最新的分布式平台策略无疑是提高遗传算法运算效率的途径之一。

因此我们可以首先通过一定的算法将待评价的种群分成不同的组，这样将各个组分布在不同的分布式计算服务器上，由各个服务器只关注处理自己机器上的个体评价，之后通过一定通讯机制进行汇总，这无疑也是提高遗传算法运行效率的一种优化方法。

4.1.3 子代群体产生过程的并行性

在遗传算法的求解过程中，产生子代的主要步骤是选择、交换和变异。这三种操作中，选择操作只与个体的适应度有关。而交换和变异操作只是和参与运算的编码有关，与之前是否有过选择、交换或是变异操作无关，因此在理论上是可以将其三种操作互相分开进行。

根据上述遗传算法的子代产生过程的并行特性，可以通过将其选择、交换和变异这三种操作分布在不同的主机上进行，使其独立运行，从而提高遗传算法的运算效率。

4.1.4 基于群体分组的并行性

以上几种并行求解理论都是整体算法的背景下进行的局部改进。从原理上来讲，遗传算法的总群体是有不同的群组组成的，根据生物界一个物种不同种群可以共同进化的特点，因此遗传算法的进化迭代同样也可以分成不同的物种种群进行，这就在理论上实现了遗传算法并行的基础。

在种群应用遗传算法计算的过程中，将一个种群规模较大的初始种群根据一定的方法将其分成不同的种群，之后对不同的种群应用一样的遗传算法分析求解框架。这样，在大数据分布式的环境下，不同的种群分析计算可以分布在不同的主机上[51]进行，从而可以提高遗传算法整体的求解效率。

4.2 传统并行遗传算法实现模型

根据上一子章节我们可以充分的体会到遗传算法具备得天独厚的并行计算条件。因此，基于其并行条件，前人们已经有了很多在其并行条件基础上的广泛研究实例，根据其并行模型的特征，一般传统的并行遗传算法模型[52]（PGA，Parallel Genetic Algorithm）

主要有四种，全局 PGA 模型、粗粒度 PGA 模型、细粒度 PGA 模型以及混合 PGA 模型，本部分主要基于这四种平行遗传算法模型进行详细的讨论和论证并指出其不足之处。

4.2.1 全局 PGA 模型

全局 PGA 模型是并行遗传算法模型中的一种主从结构算法模型，这种解决方案是保留初始种群的数量而不进行二次分组。在整个遗传算法运行的过程中，它只有一个种群，因此不管是种群的适应度函数还是适应度评价都只是一套方案。并且在种群的选择、交换和变异操作中，任何一个个体都有机会同全部的个体进行选择、交换和变异操作，因此这种模型是全局性的。其并行的核心在于主从服务器机构，主服务负责统一安排其他服务进行局部的操作运算，之后将运算的结果在整合到主服务器上进行分析汇总。这种实现模型主要有两种测量，第一种是对种群的个体适应度计算进行并行处理。第二种是对各个遗传算子进行从节点的并行处理。

这种模型的好处在于保留了传统遗传算法的全局搜索行为，是站在全局上的遗传算法的处理机制。并且这种算法对并行处理框架要求不高，实现比较简单。但是对于初始种群过大的群体，这种算法无疑又暴露出了传统遗传算法的缺点，种群数量过大又成为了影响计算效率[53]的瓶颈。

4.2.2 粗粒度 PGA 模型

这种算法模型又称之为孤岛遗传算法模型[54]，这种一种彻底的群体分组模型。在初始阶段就将种群分成很多组。每一个组独立运行一个遗传算法，从而全局内选择变成了区域内选择。因此在整个遗传算法的计算过程中，每一个组维护一套算法操作，在进行遗传算子操作的过程中，也是在自己组内进行选择，而不去在全局进行搜索。由于遗传算子不会跨组进行选择，所以在这种算法模型中引入了一种用于组之间通讯的算子，即“迁移”算子。

迁移算子最核心的作用就是维护组与组之间的信息交互，避免各个组在搜索过程中陷入局部最优解。迁移算子主要包含“迁移拓扑”、“ 迁移率”和“迁移周期”三个基本要素。迁移拓扑确定了这种模型在进行种群迁移时候的迁移路径，这与计算机的架构模型也有着一定的关系。而迁移率则是表示在进行种群之间迁移的时候应该迁移的数量，这从全局来看使得整个遗传算法不会陷入局部最优解，而可以在最终结果中找到全

局最优解。最后迁移周期则决定多长时间迁移一次，迁移周期的大小直接影响整个遗传算法的计算效率，迁移周期过多，则有损整个算法在计算过程中的时间消耗，如果过小，则达不到迁移算子的作用。一般来说，典型的迁移率控制在 10%到 20%位佳，而迁移次数则一般一代迁移一次或是几代迁移一次，因为这样比较符合自然界物种迁移的规律。通常来说，迁移率越高，则迁移周期越长。在迁移的过程中，有时候采用同步的方式进行迁移，有时候采用异步的方式进行迁移。另外，在种群迁移的过程中，有时候选择一个或多个最优的个体进行迁移，有时候则规定通过随机选择的方式进行迁移，但大多数情况下，可以采用适度比例或者排列比例进行选择，确定哪些个体将会被替换，以便利于整体算法的顺利运行。

正值国内经济技术飞速发展的今天，分布式遗传算法[55]模型称为了研究的主要方向。分布式 PGA 是 PGA 模型的一种形式，一般采用粗粒度的全局并行，各个子种群之间有着比较弱的依赖关系，其主要依靠串行 GA 的加速方法来增加整个算法的运行效率。一般来说，分布式 PGA 的求解步骤主要有以下几步：

第一步，模拟自然界的种群迁移的特性，将一个比较大的初始化种群分为各个独立的子种群。

第二步，对这些互相独立的子种群进行串行的遗传算法操作，并且选定响应的迭代周期对种群与种群之间进行种群迁移操作，从而保证搜索的全局性。

对于种群迁移有很多求解方法。第一种方法是直接进行染色体互换操作，这种方法在操作时要保证发出去的染色体数量要和接收到的染色体数量一样，从而保证整个种群个体的稳定性。第二种方法则是在进行迁移操作时，首先将所有的种群个体进行合并操作并形成一个大的种群，之后如同第一步进行种群分组操作一样，将这个大种群再次进行种群分组操作，这样，又回到各个子种群的正常状态。第三种方式则是基于中心种群的通讯机制，这种方法在于选定一个种群作为中心通讯种群，其负责与其他各个种群进行通讯操作，种群子种群要始终保持着在所有种群中的绝对最优位置，其他各个子种群通过和中心种群进行通讯的方法进行种群个体的迁移操作。同时，整个种群通过中心种群的最优个体加快整个遗传算法的收敛速度，同时改善各个个体的特性。

4.2.3 细粒度 PGA 模型

细粒度 PGA[56]模型又称之为领域模型，其核心思想是对传统遗传算法模型的修改，同样维持一个群体的进化，即对总群体不进行群体的拆分。但是这种模型在种群的平面网格上也就是在切面的模式下对种群进行的极细的小种群划分（理想情况下是每一个个体维护一个种群），并且种群之间有着极强的通讯能力。这种模型即维护了在一个种群的情况下的全局搜索能力，又使得可以并行计算的部分能以并行的方式在各个服务器上进行运行，极大的利用分布式环境下的服务器资源。细粒度模型要解决的主要问题是领域结构和选择测量。

这种领域模型是一种既能确定种群个体的空间位置，又能确定个体在所在种群中的传播路径的方法。领域模型与分布式的计算架构无关，其主要收到计算机内存结构和通讯结构的影响。一般来说，局部领域一般只有一个拓扑的直接领域，如果要把整个固定数内所有能够达到的个体都包含在内部，可以选择扩大领域半径的方法。在确定选择策略时，必须要考虑到选择压力的变化，并且选择压力与领域结构有很大的关系。其与全局的选择匹配相似，局部的匹配方式可以采用排列比例选择或是局部适应度比例的方式进行选择。局部生存选择确定局部临域中被替换的个体数量，如果子代自动替换领域中心中的个体，就可以直接使用代替换作为局部生存的策略。

4.2.4 混合 PGA 模型

混合 PGA 模型，顾名思义，是以上几种并行遗传算法求解模型的集合，它根据业务场景的不同，在实际情况下选择如何联合以上几种模型来进行求解运算。一般来说，混合 PGA 模型不是单一结构的，而是层次化的，比如上层采用粗粒度模型，下层则采用细粒度模型来进行求解。或者在求解过程中分阶段，前一个阶段采用一种模型结构，在最后进行总括分析的时候采用另一种求解模型。

4.3 基于 Hadoop 的并行遗传算法实现模型

Hadoop 大数据分析计算平台作为当今优秀的分布式并行处理平台，对处理大数据、分布式的算法模型具有非常大优势。基于上文对遗传算法并行可行性的分析以及对传统的并行遗传算法的不足进行论文的基础上，本文提出了一种混合并行遗传算法处理模型（HPGA, Hybrid Parallel Genetic Algorithm），这种处理模型主要是集合了各种处理模型

的优点并进行相应的改进，使得其可以更好的处理大数据并行问题，并在保证其计算效率的基础上，最大程度的保留了遗传算法的优秀特征。本部分内容将详细对新提出的 HPGA 进行论述并给出伪代码编程实现。

4.3.1 混合并行遗传算法求解模型

传统的并行遗传算法在进行问题求解的时候往往只能局限在某一类问题或是只是针对某几类问题进行重新设计，这大大减少了遗传算法的可用性。比如全局并行遗传算法虽然对算法的计算效率有了很大程度的优化，但是在面对初始种群过多的情况下也显得力不从心。而局部并行的遗传算法虽然解决了在初始种群过多的情况下的效率问题，但是其全局搜索能力却有了很大的限制，虽然在计算过程中引入了“迁移”算子的概念，但是其全局搜索能力的限制在很多问题上也是不可忽略的。实际情况下，既要有全局搜索能力好，并且效率极高[57]的遗传算法在并行环境下是不存在的，因此，如何能协调两者之间的关系，如何针对实际的生产环境进行有效的调节则成为了当前分布式环境下的并行遗传算的讨论热点。

针对上述环境，本文提出了混合并行遗传算在分布式大数据分析计算平台 Hadoop 上的实现，并且针对这种模型进行了扩展性方面的优化，使得既能兼顾全局搜索性能又能有着不错效率的并行遗传算法成为可能。具体实现思路如下：

本文模型是在传统遗传算法的求解框架基础上，对全局遗传算法和局部并行遗传算法进行结合，并引入相似度比较算子。由于局部并行遗传算法具有解决全局初始种群过大情况的能力，因此，可以将局部并行遗传算法放在整个算法模型的前端，对于初始种群过大的模型，可以先通过这种计算模型进行前 n 代的筛选，这样在经过 n 代种群迭代筛选之后，将边缘化的、进化偏离的种群进行筛选过滤，这样随着种群的提前迭代，会将整个种群的种群数目降低。当种群的整个种群数量降低到一定程度之后，则由全局并行的遗传算法模型进行下一轮的分析运算，对结果进行下一轮的迭代筛选，相似度比较算子引入的意义在于确定种群的前期迭代中将要淘汰的个体数量。

整个并行模型的流程图如下图 4.1 所示：

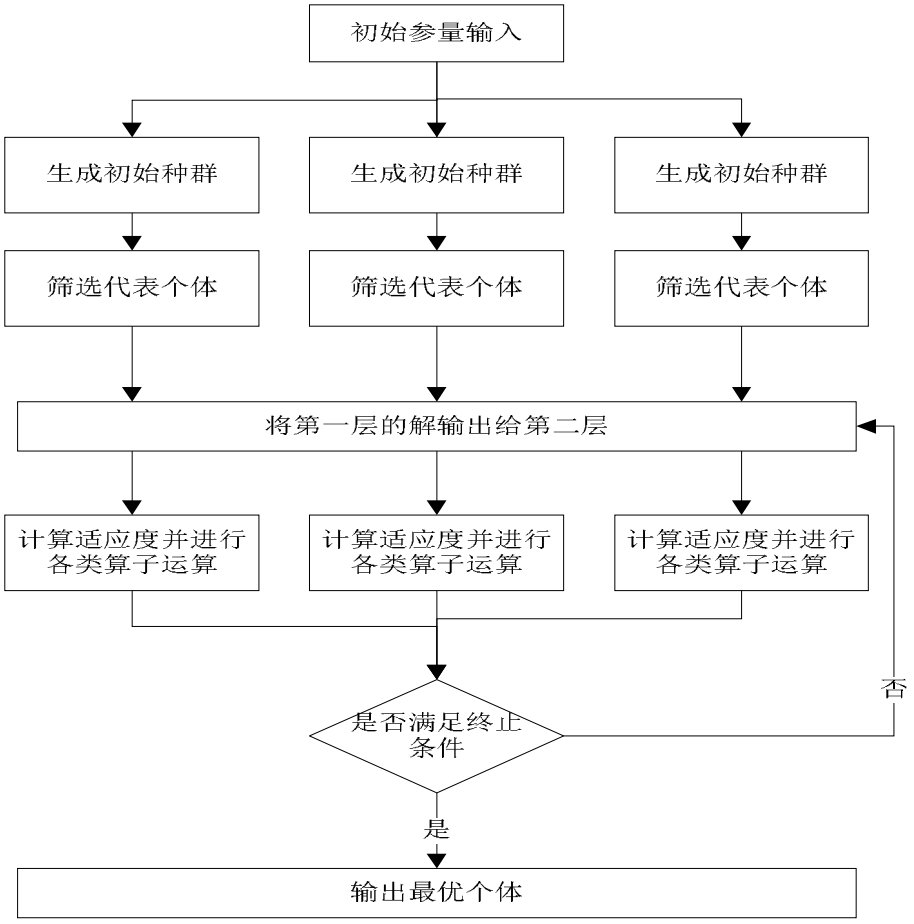

图 4.1 混合并行遗传算法整体流程图

Fig. 4.1 The global flow chart of Hybrid Parallel Genetic Algorithm

整个并行遗传算法的模型主要分为两个部分，上半部分为局部并行搜索部分，而下半部分为全局搜索部分。需要注意的一点是上半部分和下半部分不是绝对对等的，而是根据实际生产业务情况进行比重划分，有的时候可以让上半部分权重大一些，有的时候可以让下半部分权重大一些，当然在特殊情况下，其中一部分也可以舍弃掉。正如前面描述的，上部分主要解决初始种群过大的情况，这样在实际初始种群过大的情况下进行种群分割，让遗传算法的运行分布在不同的节点上进行运行，这样经过 n 代循环迭代之后有一些种群处于边缘化，成为无意义的数据，因此这个时候可以将其舍弃掉，从而整个初始种群数目会减少。从而紧接着进入下一步的全局遗传算法迭代，在全局迭代的过程中，仅仅对全局搜索没有影响的遗传算子操作，比如适应度计算等进行分节点的分布式计算，这样在局部分布式运行的情况下提高遗传算法的计算性能和效率。

4.3.2 混合并行遗传算法在 Hadoop 平台中的实现

Google 在以往经典的映射化简的函数式编程思想[58]中进行总结，从而产生了当今流行的 Hadoop 大数据分析计算平台，并且其 MapReduce 并行计算框架非常完美的同其文件系统 hdfs 结合，对大数据的分析效力产生了深远影响。

在整个 MapReduce 的运算过程中，所有的 Map 阶段操作都是并行操作的[59]，我们无需去关心一个 Map 是在一台机器节点上运行还是被多台机器节点同时调用，其框架中的并行机制能保证各个 Map 的调用是并行运算的。Map 接受参数为 key/Value 键值对，key 为每一行的行号，而 value 为每一行的内容。

当 Map 阶段运算完毕之后，也就是表明每一行的数据都被进行了一次全局的预处理，并且其整个处理过程是并行完成了，接下来 MapReduce 会将整个处理结果全部交付给 Map 阶段和 Reduce 阶段的中间链接环节 shuffle 来进行处理，这个阶段的处理主要是将 Map 阶段产生的数据进行排序和分组，这就是说会将整个数据按照 Map 的处理结果分成不同的域，之后将分完组的数据交由接下来的 Reduce 进行进一步的处理。

第三步则是由 Reduce 对分组数据进行处理，由于第三步 Reduce 得到的数据是已经在 shuffle 中排序好的数据，其实也就是在处理一个个的数据集合。因此整个过程的调用也是并行进行的。具体处理过程图如下图 4.2 所示：

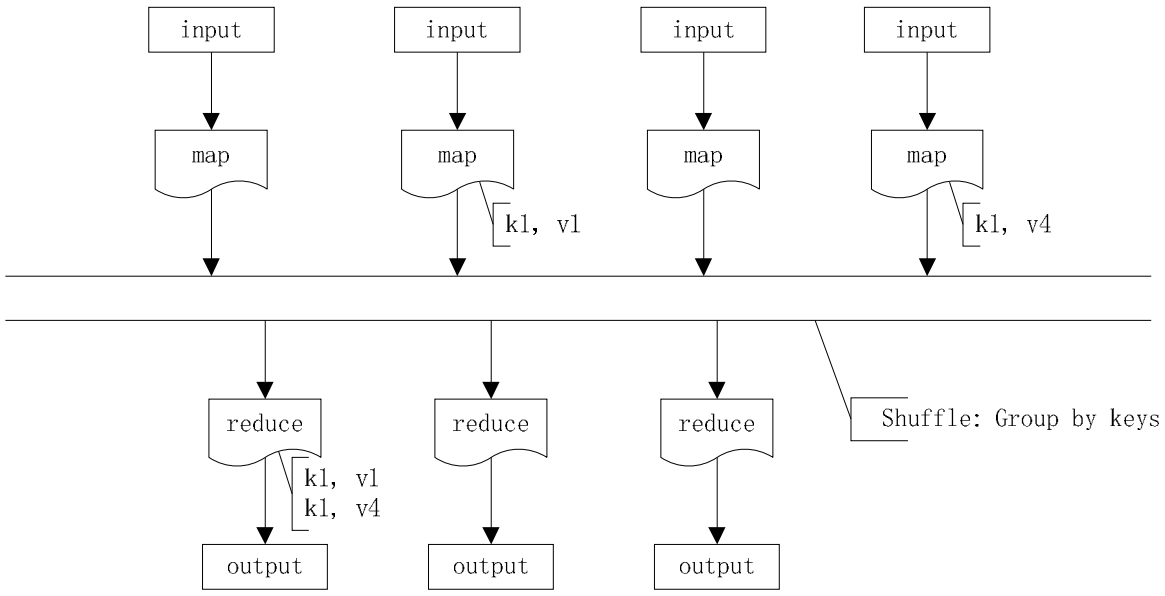

图 4.2 Hadoop 并行流程图

Fig. 4.2 The parallel flow chart of Hadoop

根据上文的描述以及遗传算法固有的并行属相，将其与 Hadoop 大数据分析计算平台具有天然的优势。由于上文提出的混合并行遗传算法的运算具有两个阶段，第一个阶段是局部并行计算阶段，而第二部分为全局并行阶段。具体阶段实现如下：

第一阶段，由于这个阶段的主要目的是对过大的初始种群进行并行筛选，得到具有强代表向的种族群体，为下一阶段的运算提供便利。因此这一步的主要任务是通过 MapReduce 进行初始种群的筛选，得到具有强代表性的种群群体，并作为第二阶段的主要输入。因此这一步是根据 MapReduce 的整个分析计算过程分成三个阶段，第一个阶段是 Map 阶段，Map 阶段将对整个种群群体进行适应度计算。紧接着 Map 阶段会将产生的结果输入到第二阶段的 shuffle 阶段，这一步由于是初始种群的筛选，因此我们采用传统的 shuffle 算法对整个种群进行排序和排序得出这一步的结果。随后进入第三个阶段即 Reduce 阶段，这个阶段我们将会对整个带有适应度值的群体进行两种操作，第一种是去重操作，由于这一步的输入结果是以分组的形式输入的，也就代表着这个组中的适应度值是一样的，因此，我们只需要取出这个组中的一个种群个体即可。另外，对相邻的个体进行适应度值的相似性比较。因此我们这一步的结果是筛选出具有代表性的个体，因此对于一些适应度值相似的个体我们只需要取出其中一个作为代表个体即可，不要全部使用，因此取出这些具有强代表性的个体作为下一阶段的的种群输入。因此第一层的流程图如下图 4.3 所示：

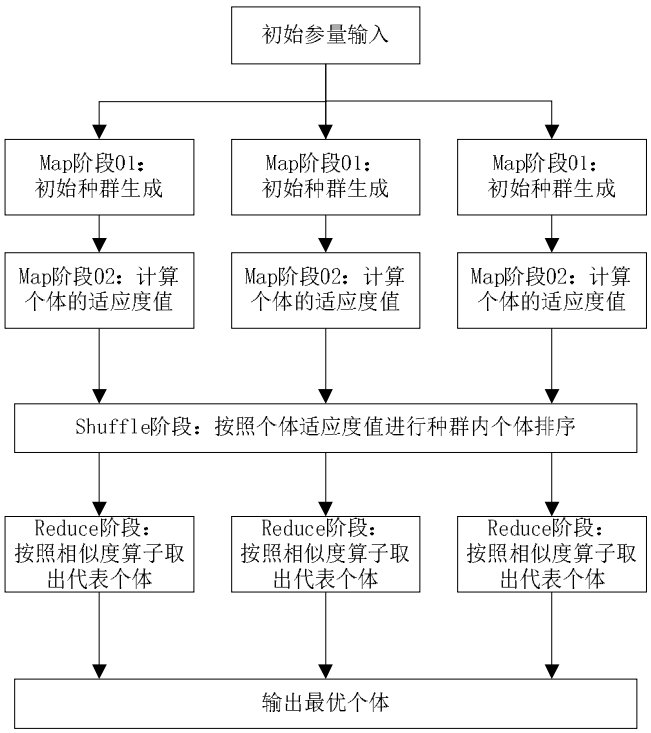

图 4.3 第一层混合并行遗传算法流程图

Fig. 4.3 The upper flow chart of Hybrid Parallel Genetic Algorithm

第二阶段，这一阶段主要是利用上一阶段的产生的结果，进行全局并行搜索，得出最终结果。因此，可以将整个遗传算法的整个阶段按照 MapReduce 算法分成三个阶段。Map 阶段主要做初始种群的输入和适应度的计算和终止条件判断。由于整个种群个体的适应度值具有无关性，因此完全可以在 Map 中进行种群是适应度值的并行计算。其次是终止条件的判断，在本文阐述的模型中终止条件一般选择迭代到一定次数或者最终搜索结果没有极大变化则去终止程序的运行。因此这一步可以刻在 Map 阶段进行判断。而 shuffle 阶段会对得出的结果进行分组和排序，在遗传算法中各个个体的适应度值计算完毕之后，很重要的一步就是对整个种群进行排序操作，因此在 shuffle 阶段将会完成整个种群的排序操作。而在 Reduce 阶段将对整个种群进行选择复制以及交换变异操作。这一步中的操作一共有三分部，第一部分是选择优秀的个体进行复制操作，并且淘汰适应度值不高的个体，之后将会随机选择一定比例的个体进行交换和变异操作，从而保证整个算法的最终结果不会陷入局部最优解，保证整个进化过程的完整性。

具体流程图如下图 4.4 所示：

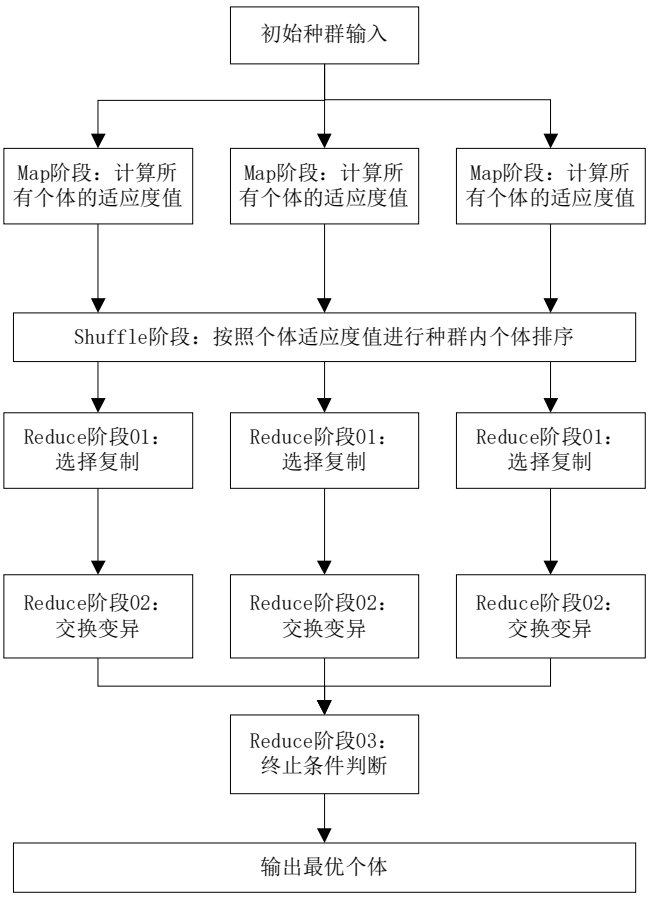

图 4.4 第二层混合并行遗传算法流程图

Fig. 4.4 The lower flow chart of Hybrid Parallel Genetic Algorithm

如上所述，我们可以明确地看出整个算法的模型主要是通过前期并行筛选结合后期全局搜索得出最终求解结果。值得我们注意的是，第二阶段的全局过程也是基于 Hadoop 大数据分析计算平台并行运算的，因此在一些情况下，我们应该动态的调控第一部分和第二部分的权重关系，甚至有些时候我们可以直接省掉第一部分直接使用第二部分进行遗传算法的并行求解。因此，整个算法模型的实现背景是基于在初始种群过大的群体之上进行的。比如我们将在下文提出的本模型在 TSP 问题的应用验证。我们知道 TSP 问题是著名的优化组合问题，但是当参与优化组合的个体过多的时候，其排列组合的种类将是会成指数增长的。例如我们的排列组合初始参选个体有 20 个，其排列组合的种类就是 20 的阶乘个数，也就是约 13516 亿亿个。在实际生活中，排列组合的初选个数是 20 个左右甚至更多的情况是很常见的，但是其排列组合的种类却是成指数增长的，因此一种优秀的算法解决模型会对这类问题的求解产生深远的意义和影响。本文提出的混合

并行遗传算法正是解决此类问题的算法模型，为这类问题的求解解决了传统算法无法达到的效率上的优势。

4.3.3 混合并行遗传算法的优越性

在以上论述中，本文根据传统的遗传算法求解框架模型，并结合其优点，提出了混合并行遗传算法求解模型，这种模型根据不同遗传算法模型的分层求解框架，使得并行遗传算法既保留的全局的搜索能力，又能极大的提高算法整体的计算性能和效率。具体优越性如下：

（1）保留了全局搜索能力。本文提出的遗传算法在第二层计算框架下主要采用全局并行的遗传算法计算模型，虽然在第一层进行了一次全局筛选，但是这种淘汰制的筛选只是将没有实际意义的种群个体进行淘汰，并不会影响最终算法全局搜索能力。同时，这样做的目的是最大化的保留了全局搜索能力，避免了算法模型进入局部最优解的危险。

在全局范围内提高了算法的搜索效率。本模型的两个模型部分都采用的分布式并行策略。虽然求解的模型不同，但是第一层求解模型在初始种群较多的情况下[60]采用分片并行策略提高了整个算法的求解性能。而第二层则是在于全局搜索无关的算子上进行了并行计算从而提高算法的计算性能和效率。这使得模型整体在全局范围内的搜索效率的了提升。

（2）充分利用 Hadoop 大数据平台的能力。本文提出的模型与最新的分布式大数据平台 Hadoop 大数据分析计算平台进行结合，借助 Hadoop 的文件系统 hdfs 和其计算框架 MapReduce。使得并行框架结合大数据计算平台，充分发挥了两者的性能。

（3）充分发挥了遗传算法的扩展性。本文提出的混合遗传算法求解框架，在原有算子的基础上提出了相似度比较算子，这个算子的主要核心目的在于确定第一层计算完毕后将会有多少种群参与第二层的遗传算法计算，并且可以根据实际问题对这个算子进行评估，从而对第一层框架和第二层框架的比重进行相应的衡量，从而确定两层计算框架的比重。

（4）提高具体问题求解的灵活性。在实际的问题求解中，往往不是拿一种固定的框架进行计算求解，而是根据实际问题确定计算框架的结构。而本文提出的混合并行模

型同样给出了很多的框架变形接口，使得在实际问题中可以根据实际的业务情况进行适当的框架变形。从而让求解框架更加灵活，适应度更好。

4.4 本章小结

本章节为整个论文的核心章节之一，主要的研究任务是先分析基于 Hadoop 的遗传算法并行的可行性，由于 Hadoop 是天生的并行计算平台，因此整个研究的核心是如何对遗传算法的计算过程进行相应的改进，在不失去遗传算法的核心计算分析计算优势的前提下对其算法的流程最早 Hadoop 进行改进。改进的着重点主要从遗传算的个体适应度的并行性，整个群体中各个个体适应度评价的并行性，子代群体产生过程的并行性以及基于群体分组的并行性这几个方面进行分析。最终的得出遗传算法在在 Hadoop 平台上并行的依据。

由于在遗传算法的并行模型前人已有了比较多的研究，因此第二部分的着重点在于分析前人们研究的成果，从全局 PGA 模型，粗粒度 PGA 模型以及细粒度的 PGA 模型这几方面进行相应的研究，并从中吸取有益的经验，结合自己的研究成果，提出了混合并行遗传算法在 Hadoop 大数据计算平台上的实现。

最后一部分主要是结合前面两部分的研究成果，基于前几部分的得出的混合并行遗传算法在 Hadoop 可以运行的依据，具体说明混合并行遗传算法在 Hadoop 大数据分析计算平台的实现模型，并对此模型进行了伪代码实现。最后通过分析论证对混合并行遗传算法在 Hadoop 大数据平台上实现的优越性进行总结概述。为下一章节的实际运用做了充分的铺垫。

第 5 章 TSP 问题下的模型验证

5.1 TSP 问题传统遗传算法求解模型

当旅行商问题中的城市数量不是很多时，采用传统的遗传算法进行求解无疑是一种比较高效，扩展性比较强的方法，整个过程的求解也会比较顺畅。针对具体的求解方法，我们先建立一个包含所有城市的数组，并且将这个数组的无序不重复组合的结果作为整个求解方法的初始种群数。之后对其整个种群进行选择复制、遗传变异操作，并且在整个进化过程中对每个种群个体进行个体适应度检查。当迭代到一定代数的时候进行终止操作，并从中选出最优个体。

通过遗传算法求解 TSP 问题流程图如下图 5.1 所示。

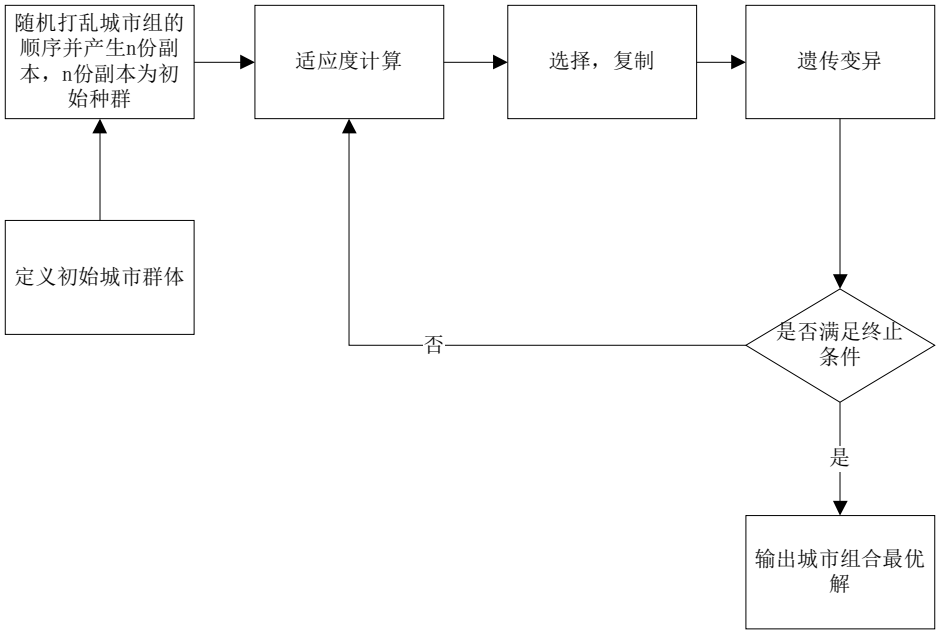

图 5.1 遗传算法求解 TSP 问题流程图

Fig. 5.1 The flow chart of TSP solution with Genetic Algorithm

如上图所示，TSP 问题的整个求解过程严格按照遗传算法的求解流程，具体的求解过程主要分为如下几步：

第一步，定义初始城市群体。这一步的主要任务是定义哪些城市要进入整个求解过程的候选队列中。在组合优化问题中，也就是有哪些元素进行参加组合优化问题。

第二步，初始化种群。由于遗传算法的计算搜索对象是一个给定的种群，并且当我

们取得已经定义好的城市之后，我们要将所有的城市放入一个数组，并且将这个数组中城市的排列顺序进行随机打乱，尽量做到不重复。最终得到各种组合类型的城市数组，这些数组的集合即整个算法的初始种群，其每一种组合形式即种群的一个个体。

第三步，对城市数组集合中的各个个体进行适应度计算。这一步中个体的适应度计算即城市的排列中依次走过路程的长短。我们将这个值作为每一个个体的适应度大小。

第四步，进行种群的选择复制操作。在这一步中我们将按照自然界中的淘汰比率 20 进行个体淘汰。由于上一步我们已经计算出了各个个体的适应度值，因此我们可以按照种群中个体的适应度值进行由小到大的排序，由于我们最终的结果求得是最小路径，因此，路径长的相应的适应度值也就是越小，我们按照这个规律对整个种群中的个体进行选择淘汰，我们先选取适应度值比较大的进行复制，并淘汰掉其中适应度值比较小的个体。但最终结果保持整个种群的数量不变。

第五步，进行遗传变异操作。这一步是前半阶段是将上一步中选中的个体进行复制之后遗传到下一阶段。之后对新的种群进行变异操作，变异操作我们可以对其中的选定个体中城市的顺序进行重新排序。这样做的目的也是为了避免最终的结果会陷入局部最优解而得不到全局最优解。

第六步，对是否满足终止条件进行判断。对于整个计算过程，我们选定的终止条件是看临近 10 代的结果有没有比较大的变化，如果变化较大，说明整个优化搜索过程还在继续，如果临近 10 代的结果没有较大的变化，我们可以判定整个搜索结果都在最优解的周围进行徘徊，因此我们可以终止整个迭代过程，并输出最优解。其求解模型的伪代码如下。

TSP 问题的中体调用类。

```
class TSP_GA {
    public static void main(String[] args) {
        1：调用城市个体类并进行城市个体的初始化并保存在数组中
        2：初始化整个种群
        3：调用选择复制、交换变异算法进行种群迭代
        4：输出最终结果
    }
}
```

整个遗传算法的城市个体类。

```
class City {
    1：定义城市类的具体属性
    double distanceTo(City city){
        2：计算到给定城市的距离
    }
}
```

保存的整个种群个体和进行种群个体适应度计算的类

```
class Population {
    1: 定义数组保存整个种群个体
    Tour[] tours;
    // Construct a population
    Population(int populationSize, boolean initialise) {
        2： 保存整个种群的信息
    }
    Tour getFittest() {
        3： 获取当前种群中最优的个体
    }
}
```

种群的每个个体具体信息类

```
public class Tour{
    1：保持所有城市的路径
    private ArrayList tour = new ArrayList<City>();
    public void generateIndividual() {
        2：创建一个随机个体并将内部的城市顺序打乱
    }
    public double getFitness() {
        3：计算每个个体的适应度值（距离小的适应值较大）
    }
    public int getDistance(){
       4：获取当前个体中路径的当前顺序下的路径总距离
    }
}
```

选择复制，交换变异调用过程。

```
public class GA {
    1：定义交换概率和突变概率
    double mutationRate = 0.015;
    int tournamentSize = 5;
    Population evolvePopulation(Population pop) {
        1：调用种群交换操作方法
        2：调用个体变异方法
    }
    Tour crossover(Tour parent1, Tour parent2) {
        2：对给定的个体进行交换操作
    }
    void mutate(Tour tour) {
        3：对于给定的个体，进行突变操作
    }
    Tour tournamentSelection(Population pop) {
        4：对于给定的个体，进行交换操作
    }
}
```

5.2 Hadoop 下 HPGA 求解 TSP 问题的数据平台设计

基于第二章介绍的 E-MapReduce 大数据分析计算平台，本章节将采用其作为模型验证的主要运行平台，阿里云推出的 E-MapReduce 大数据分析计算平台在运行方面有着诸多的优势，比如高扩展性，高稳定性等。同时，在应用 E-MapReduce 平台的同时，其自身预装的预计 Hadoop 大数据处理框架能够节省很多时间在整个框架基础设施的搭建上。由此看来，E-MapReduce 的应用对于本文的模型框架验证将是不二之选。

5.2.1 基础硬件设施设计

本文将采用小规模的 Hadoop 大数据平台作为算法的实验验证，因此硬件设施一共设置了 21 个节点，其中一个 Master 节点和 20 个 salver 节点的大数据品台，由于本文主要关注点为基于 Hadoop 大数据平台上数据的运行效率和可靠性，因此不再花费节点去做容灾处理和高可靠性处理。具体硬件平台清单如下表 5.1 所示。

表 5.1 Hadoop 大数据平台硬件架构

Tab. 5.1 The infrastructure of Hadoop platform

处理器型号	处理器	内存	节点容量	节点个数	节点功能
series II	2 核	4GB	40G	1 个	Master
series II	2 核	4GB	40G	20 个	Slaver

另外，E-MapReduce 大数据计算平台另外提供的一种 OSS 的数据存贮机制，因此，在使用的时候可以直接将数据上传到其 OSS 服务器[61]上。这样，上传的 MapReduce 程序可以直接在 OSS 服务器上读取将要处理的数据。

5.2.2 基于 E-MapReduce 的基础软件设施设计

本文的主要采用阿里云提供的 E-MapReduce 作为整个算法框架的验证平台。当硬件设置配置完毕后置，接下来为软件环境的构建，本文将选取目前 Hadoop 的稳定版本 2.4.1 作为环境框架和开发框架。对于本实例所依赖的 Hadoop 生态圈软件，我们将采用硬件架构的初始化安装的最小软件集即可满足需要。具体软件配置清单如下表 5.2 所示。

表 5.2 Hadoop 运行软件集

Tab. 5.2 The software compilation of Hadoop platform

名　　称	Hybrid-Parallel-Genetic-Algorithm
日志功能	是
日志路径	oss://hpga
产品版本	EMR-2.1.1
软件配置	Hadoop / Ganglia / Spark / Hive / Pig / Sqoop / Hue / Zeppelin / Phoenix / Tez
新建安全组	emr-default-securitygroup

5.3 Hadoop 下 HPGA 求解 TSP 问题模型设计

基于上一章提出的混合并行遗传算法模型，本部分将进行编码实验，整个编码过程将分为两个部分，第一部分为第一层的算法编码设计，第二部分将是第二层的编码设计，由于整个算法关注的 TSP 问题的求解效率，因此将针对 TSP 问题选取最适合的算子，包括编码设计、选择算子、交叉算子以及变异算子。并根据选取的算子进行 MapReduce 编码实现，并通过上文介绍的 E-MapReduce 进行实例验证，以证明整个算法框架提出的可行性。

5.3.1 初始总群与编码设计

遗传算法中，第一步也是非常关键的一步就是初始种群的生成以及编码，只有对初始种群及其编码进行合理的设计，才会让以后的步骤顺利的进行。很多时候，一个设计合理的初始种群和编码将会对整个算法的求解起到事半功倍的效果。

对于本文论述的遗传算法求解 TSP[62]问题。由于整个问题是实际问题，因此本文将不采用虚拟编码或者是二进制编码的方式，而采用仿真模拟编码[63]的方式进行初始种群的生成以及编码。由于在 TSP 问题的本质为排列组合优化问题，因此我们可以假设所有的城市都在一个二维坐标轴中（都在二维坐标轴的第一象限），因此每一个城市都有一个有序实数对作为其唯一标示，城市与城市之间的距离我们可以采用毕达哥拉斯定理来计算。城市与城市之间的有序组合将是整个种群中的一个个体。由于我们要搜索所有解的可能，因此，整个种群的生成我们将采用穷举法来做。种群初始种群的生成及其编码步骤如下：

（1）在坐标轴中初始化所有城市的位置，我们将所有城市按照初始点距离的远近按照 1 号到 n 编号，因此每一个城市的坐标点即 $L(x_i, y_i)$。

（2）根据第一步中每一个城市坐标点的选取，我们可以随机生成旅行商的各种可能出现的城市旅行路径。我们设 $T_j = \{L_1 ... L_n\}$ 为第一种可能的旅行路径，因此其它的旅行路径将是这个旅行路径的各种有序组合形式，这里我们将采用穷举法的方式全部生成。具体生成操作可由 Map 阶段来完成，由于 Map 也是并行进行的，因此我们可以得到比较客观的生成速度。

当第二步结束之后我们得到了旅行商全部可能的旅行路径，也就是各个备选排列组合个体的全部有序排列组合情况。因此我们也就得到了整个遗传算法的初始种群，我们设整个初始种群的为 S，从而 $S_j = \{T_1 ... T_n\}$。

5.3.2 求解 TSP 问题的遗传算子设计

（1）选择操作算子

一般来说，整个遗传算法在进行个体选择和复制得时候是根据个体的适应度来决定的，本文求解的问题将根据通过交叉、变异操作产生的个体与父类个体种群进行混合替

换的策略产生下一代种群个体，这也就是跨时代精英选择算法。在本次的实验中将采用轮盘赌的方式来产生新的种群个体。轮盘赌的方式优点具有以下几点：

1）择优选择。在下一个的个体选择中，适应度值高的个体将有更大的几率被遗传到下一代，而适应度值低的个体机会将会大大，这非常类似于生物物种的遗传进化原理。

2）保持物种的多样性。由于轮盘赌方式并没有完全否定适应度小的个体被选取的机会，因此能极大的保持物种的多样性。

3）排序测量解决了比例适应度计算的尺度问题。

（2）交叉操作算子

交叉算子又叫重组算子或配对算子，是指针对两个相互配对的染色体按照某种既定的方法进行部分基因的交换重组操作，从而生成两个全新的个体。交叉算子时整个遗传算法中最重要的算子，其算子的选择和有效性直接决定了整个遗传算法的全局搜索能力。

在遗传算法进化的过程之中，当染色体在进化的过程之中与上一代的染色体没有较大的差异时，可以通过染色体的两两之间的重组操作来产生新一代的染色体参与遗传进化操作。交叉算子的操作主要分为两个步骤：第一步首先在新复制的种群个体中随机选取出两个个体（每个个体有许多编码组成）。第二步在这个两个选出的个体中，按照对应的顺序随机选取出既定个数的位置进行交叉操作。因此，交叉算子的设计主要包含两个方面的内容，一是如何确定交叉点的位置，二是如何进行基因的互换操作。

本文的主要讨论点在于如何通过遗传算法在 Hadoop 大数据分析计算平台上解决 TSP 问题，因此本文在 TSP 问题的初始编码阶段选取的是实数编码方法，是用坐标的形式表示每个城市的距离，因此针对这个编码方式，本文将选择比较符合本文论述情况的单点交叉算子，即从种群中随机选取出两个交叉点，并对随机选取的个体进行基因互换操作。用表达式的方式表述即为：设有两个进行了实数编码的个体 A 和 B，长度为 $L=6$，$A=a_1a_2a_3a_4a_5a_6$，$B=b_1b_2b_3b_4b_5b_6$，随机选择一个整数 $k\in(1,L-1)$，设 $k=4$，经过交叉后的到 A' 为 $A=a_1a_2a_3b_4a_5a_6$，$B=b_1b_2b_3a_4b_5b_6$。

（3）变异操作算子

变异算子是在一次迭代中，选择算子和交叉算子在完成运算之后进行运算的一种算

子，变异算子在整个过程中处于比较小的比例，并不是关键算子，他是一种防止整个遗传算法早熟的算子。因此，变异算子是遗传算法中产生新个体的辅助方式而不是主要方式，但是它也是整个算法迭代中必不可少的一个环节。

本文由于采取的是实数编码的方式，用坐标点表示城市的位置，因此针对这种编码方式，采取自定义的变异操作。同时，种群个体变异的概率 P_m 直接影响到整个算法到收敛速度。因此这里我们采用针对不同到种群个体采取不同的变异策略来进行变异操作。这样做好的好处在于既能保持整个基因种群的多样性，也能防止进入局部最优的结果。同时还能避免破坏较优的个体并使其顺利进入下一代的迭代操作。基因个体的自适应变异函数 P_m 的公式计算为：

$$\begin{cases} round(lu\times(le+\dfrac{f_{\max}-f_1}{f_{\max}-f_{\min}})/1+\exp(\lg\times(t-ld))), f_1>0 \\ round(lu/(1+\exp(\lg\times(t-ld)))), f_1=0 \end{cases} \tag{5.1}$$

其中，$f_1=0$ 表示当前个体为不可行解；$f_1>0$ 表明当前个体是可行解并且为当前采取变异操作的个体，t 在这里表示当前进化的带数。lu 表示最终到达稳定个数状态情况下的变异数目，ld，le，lg 都是表述参数，其中，对 lg 的数值设置按照 $ld>1$，$ld<1$ 两种不同的情形进行处理 $(f_{\max}-f_1)/(f_{\max}-f_{\min})$ 代表在当前总群个体中待变异个体的优劣情况，值越小表示这个个体就更优秀，值越大表示这个个体越差。*round* 则表示对应的输入参数进行取整操作。

（4）相似度比较算子

相似度比较算子是本文为第一层初始种群筛选而引入的一个算子[65]，这个算子的目的旨在从一部分种群中选出具有强代表性的个体进行第二层计算过程。基于本案例的特点，其相似度的比较的实质其实就是文本的相似度比较。对于文本的相似度比较，目前比较流行且高效的算法模型无非是余弦定律，二是 JaccardSimilarity 方法，由于 JaccardSimilarity 方法在文本比较方面有更好的性能并且实现比较容易。因此本例将采用 JaccardSimilarity 方法进行不同旅行路径的相似性比较。

实际上就是两个集合的交集除以两个集合的并集，所得的就是两个集合的相似度，

具体比较步骤如下。

第一步，设两个集合 S 和 T；

第二步，将要比较的文本从前到后进行扫描，将一个文本的文字放入集合 S，并将另一个文本的内容放入文本 T；

第三步，计算文本的交集与文本并集相除的值作为文本比较的相似度指标，其数学表达式为|S ∩ T|/|S ∪ T|。

5.3.3 求解 TSP 问题第一层算法模型设计

根据上述描述，本部分将采用基于 Hadoop 的混合并行遗传算法针对 TSP 问题给出第一层的算法模型设计步骤，具体步骤如下：

第一步，对初始城市的个数及属性进行设定；

第二步，Map 阶段：生成大规模的初始种群，初始种群即为所有城市的不同的有序排列组合的集合。对种群中的所有个体进行适应度计算，得到带有适应度的全体种群个体；

第三步，Shuffle 阶段：对整个种群按照适应度的大小进行排序操作，得到具有既定顺序的种群；

第四步，Reduce 阶段：这个阶段主要是采用相似度比较算子进行种群中个体的相似度比较，从众多的个体中选取出具有代表性的个体，组成遗传算法问题求解的初始种群。

第一层算法模型流程图如下图 5.2 所示：

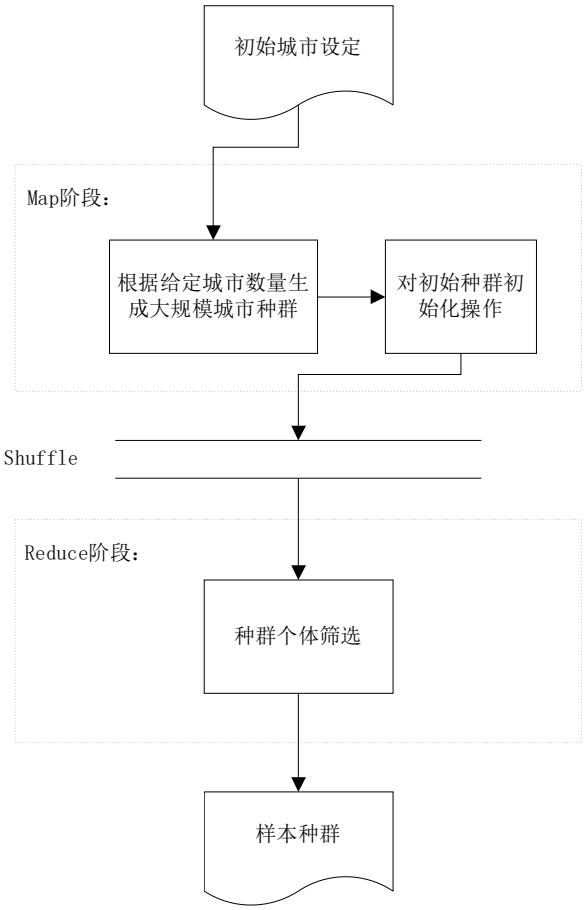

图 5.2 第一层遗传算法求解 TSP 问题流程图

Fig. 5.2 The flow chart of TSP solution with Genetic Algorithm at upper layer

根据上述流程图及流程描述，其伪代码实现如下。

Map 阶段伪代码如下。

Map {

1. 读取初始城市设定；
2. 根据给定城市生成所有可能的解空间；
3. 输出到 Shuffle 阶段。

}

Reduce 阶段伪代码如下。

Reduce {

1. 从 Shuffle 阶段的排序结果中读取数据。
2. 根据 JaccardSimilarity 算法比较文本的相似度值并进行种群规模筛选；
3. 将筛选结果输出到 hdfs 文件。

}

5.3.4 求解 TSP 问题第二层算法模型设计

第二层设计主要是应用第一层的计算结果，由于第一层的运算结果已经得出一个在数量上比较客观的初始种群数量。因此，我们可以种群带入第二层的 Hadoop 混合并行遗传算法进行求解运算，具体的求解步骤如下：

第一步，初始化第一层的计算结果作为这一层的初始种群；

第二步，Map 阶段：对所有的种群个体进行适应度计算，确定所有的种群个体的适应度；

第三步，Shuffle 阶段：对上一步得到个体适应度的个体进行分组排序，最终得到具有有序数列的群体。为下一步的选择复制操作提供基础数据；

第四步，Reduce 阶段：由于在上一步中我们已经得出具有有序数列的种群。这一步的主要任务有两个，一个按照上文定义的选择复制算子即轮盘赌的方式进行选择复制操作。第二个任务主要是按照上文定义的交叉算子和变异算子进行整个种群进行交叉和变异操作，从而使得整个搜索结果不会陷入局部最优解；

第五步，终止条件检查，本案例主要采用检查在一定代数内是否其最优解是否有大幅度的变化，如果没有，则输出最优解。

第二层算法模型流程图如下图 5.3 所示：

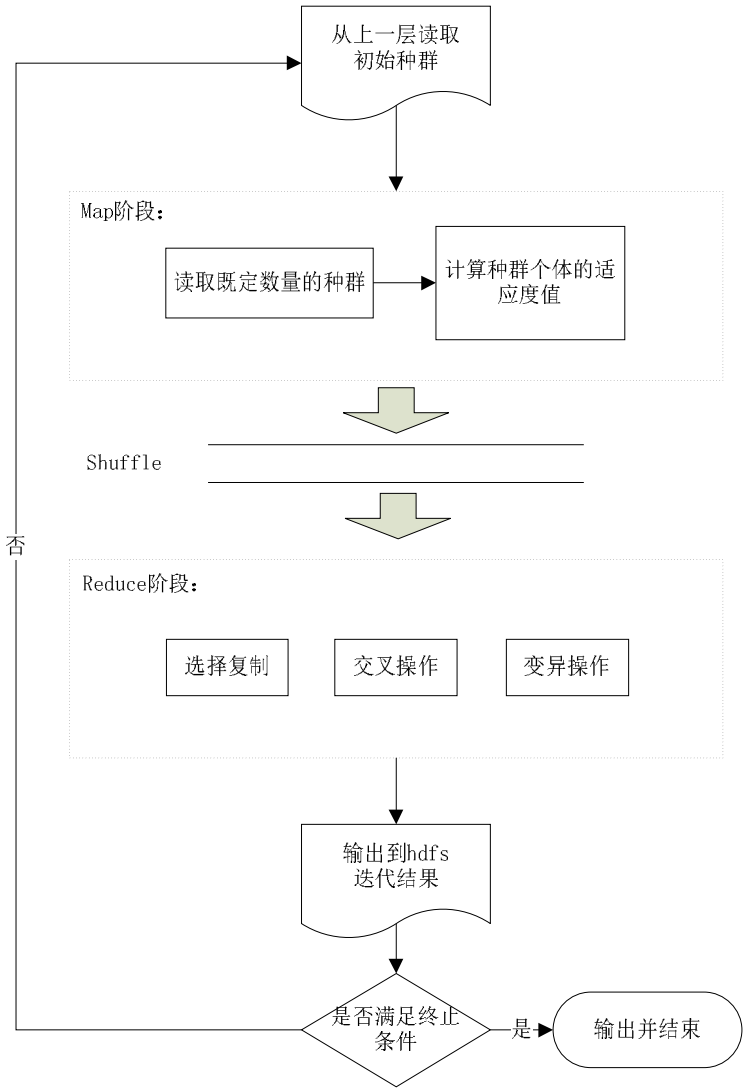

图 5.3 第二层遗传算法求解 TSP 问题流程图

Fig. 5.3 The flow chart of TSP solution with Genetic Algorithm at lower layer

根据上述流程图及流程描述，其伪代码实现如下。

（1）Map 阶段伪代码如下。

Map {

1. 读取上一层的的初始种群；
2. 计算初始种群中个体的适应度值；
3. 输出到 Shuffle 阶段。

}

（2）Reduce 阶段伪代码如下。

Reduce {

1. 从 Shuffle 阶段的排序结果中读取数据；
2. 根据轮盘赌方法进行选择复制操作；
3. 将种群进行交换变异操作；
4. 判断是否满足终止条件，若满足则输出，若不满足则返回到 Map 操作进行下一轮迭代。

}

5.4 算法测试与分析

本文不仅将本模型在 E-MapReduce 平台[66]上进行运行测试，同时选取传统的遗传算法求解模型和伪分布式的 Hadoop 大数据分析计算平台作为比较比较运行。整个实验环境主要关注于三种求解方式的求解效率问题。因此，在正式的试验中本案例将在初始城市为 2 个、5 个、10 个、15 个和 20 个的时候进行试验测试，由于初始种群的数量为城市数量的阶乘乘机，因此所对应的初始种群数量分比为 2 个、120 个、3628800 个、1307674368000 个和 2432902008176640000 个。由此我们可以充分的比较出在不同种群规模的大数据环境下不同的算法模型所运行的效率问题。

由于整个模型的分为两层进行，这两层在逻辑上具有无关性，只是第二层算法模型依赖于第一层算法模型的执行结果。因此，我们将分别在上文提到的不同初始城市数量下进行比较验证，以证明整个算法不仅仅在种群规模的生成下有较高的效率，并且在整个遗传算法的搜索过程中具备比较高的效率。

5.4.1 第一层 HPGA 算法模型试验结果

根据试验设计，第一层的主要任务是完成初始种群的生成操作，最终可以得到传统 GA、伪分布式 GA 模型和基于 Hadoop 的 GA 模型的计算结果如下表 5.1 所示。

表 5.1 第一层 HPGA 运行结果

Tab. 5.1 The data generated by upper layer of HPGA

Amount of cities	传统 GA	伪分布 GA	Hadoop 集群 GA
2	1s	54s	21s
5	10s	78s	37s
10	3250s	762s	256s
15	N/A	N/A	687s
20	N/A	N/A	1074s

注：N/A 表示不具备计算能力。

根据上述表格，可以得到算法运算效率的折线图如下图 5.4 所示。

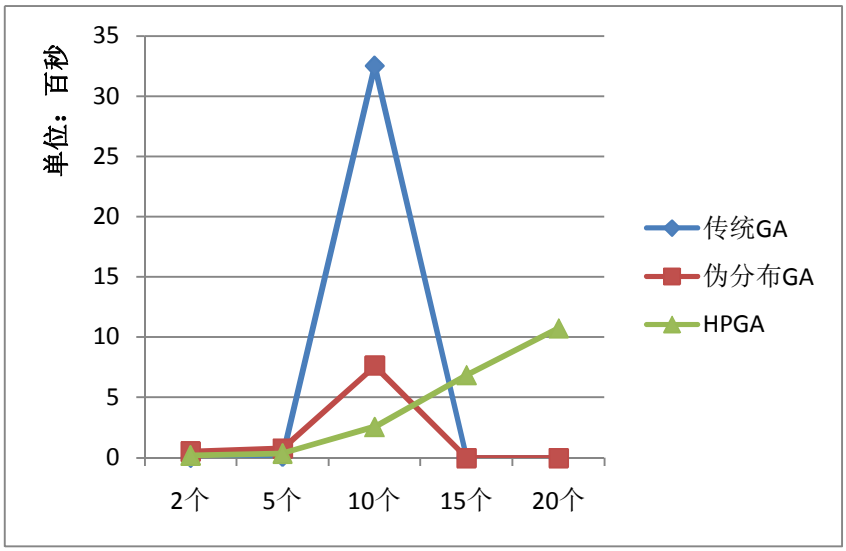

图 5.4 第一层遗传算法求解 TSP 问题流程图

Fig. 5.4 The flow chart of TSP solution with Genetic Algorithm at upper layer

5.4.2 第二层 HPGA 算法模型试验结果

根据试验设计，第二层的主要任务是完成种群的最优路径检索操作，最终可以得到传统 GA、伪分布式 GA 模型和基于 Hadoop 的 GA 模型的计算结果如下表 5.2 所示。

表 5.2 第二层 HPGA 运行结果

Tab. 5.2 The data generated by lower layer of HPGA

Amount of cities	传统 GA	伪分布 GA	Hadoop 集群 GA
2	7s	120s	68s
5	34s	782s	648s
10	23250s	N/As	1983s
15	N/A	N/A	7825s
20	N/A	N/A	12074s

注：N/A 表示不具备计算能力。

根据上述表格，可以得到算法运算效率的折线图如下图 5.5 所示。

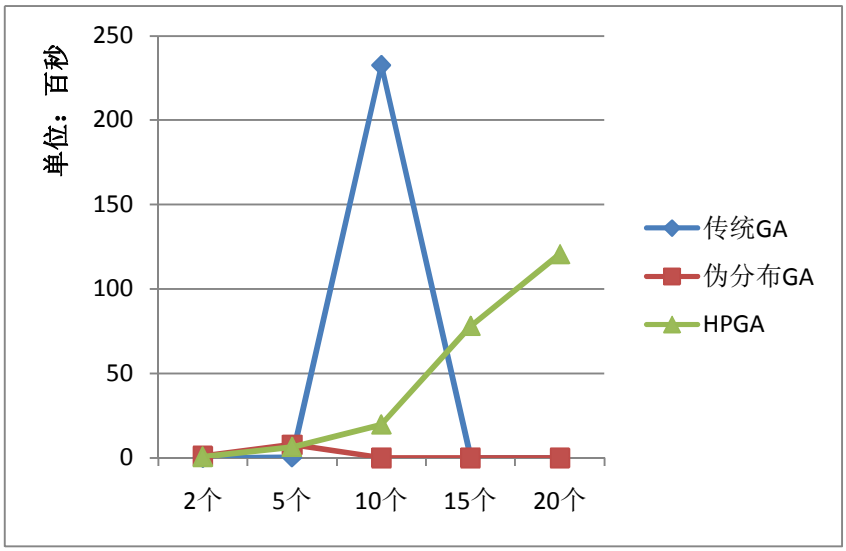

图 5.5 第二层遗传算法求解 TSP 问题流程图

Fig. 5.5 The flow chart of TSP solution with Genetic Algorithm at lower layer

5.4.3 基于 Hadoop 的 HPGA 试验结果分析

整个实验在选取了传统遗传算法、伪分布式的遗传算法和本文提出的基于 Hadoop 的混合并行遗传算法及其所选取的 5 组实验数据来证明本文提出的模型的可行性。

由上表和折线图我们可以清晰的看出，无论是在大规模种群生成阶段还是在种群个体进行遗传算法进化的阶段，传统的遗传算法框架在处理小规模群体数据的时候具有较

高的求解速率，远远超过本文提出的混合并行遗传算法模型框架。但是，当数据量一旦过大，传统的算法模型将难以承受住数据的大量分析，本文提出的混合并行遗传算法模型充分发挥了其分布式计算的优势。并且当数据提高到一定的数量的时候，传统的遗传算法和伪分布的遗传算法（其实质是在一台机器上部署大数据平台）由于计算时间过长而放弃计算。另外，其中出现的伪分布式的数据分析模型的作用只是作为传统遗传算法和本文提出的算法模型的参照，更加充分的说明了在不同的数据量下应该选择正确的算法模型进行求解。

另外，从上图还可以看出，在 E-MapReduce 大数据分析计算平台上，随着初始城市的逐渐增多（初始解空间的增多），其增长速率呈现出线线性增长的趋势。也就是说，虽然在本例中整个初始解空间是以随着初始城市数量多而呈现出指数增长的趋势，但是其求解速率确实呈线性增长的趋势。在这种情况下，如果初始解空间增长到本例中集群无法承受的程度，我们同样可以通过增加廉价的服务器节点的方式去进行整个算法硬件支持框架的扩展，并且能将整个算法的计算过程和承受能力置于可控范围之内。这同时也是本文提出的算法求解框架的优点之一。

基于上述实验结果和分析，可以得出整个算法框架的优越性如下。

（1）高灵活性。本框架在整个问题实例化实施的过程中，可以根据具体的情况对第一层架构和第二层架构进行权重比分配。同时由于遗传算法自带的灵活性，使得整个算法框架具有很高的灵活性。

（2）易扩展性。本文是用 Hadoop 大数据分析计算平台承载遗传算法的运行，并对遗传算法进行相应的并行性改进。这样做的好处在于在数据量增多的前提下可以对硬件环境进行充分的扩展，以适应整个算法的运行求解。

（3）高可靠和容灾性。Hadoop 天生的高可靠和容灾性为遗传算法的运行提供了非常稳定的运行条件。

当在面临实际的生产条件的时候，本文提出的模型可以根据实际的生产条件进行改变，并且要着重注意在第一层算法和第二层算法框架的衔接关系，第一层算法的框架在本文是用来生产大规模初始群体，但是实际情况下，很多大规模的数据群体已将存在，因此第一层框架可以用来对数据进行分析清晰，使得整个数据符合将要应用的算法框架。因此，合理使用两层框架的关系也是在实际应用中的权重比较高的。

5.5 本章小结

本章节为整个论文论述的核心章节之一，首先介绍 TSP 问题的传统求解模型，之后引出基于本论文论述的模型即基于 Hadoop 的混合并行遗传算法模型来求解 TSP 模型。详细介绍了新模型每个过程的设计方案，包括初始种群的编码设计，遗传算子的设计以及第一层求解算法和第二层求解算法的设计。从而验证在 TSP 问题下的基于 Hadoop 的混合并行遗传算法求解的可行性。在本章节论述的过程中，着重论述了本模型在初始种群过大的前提环境下是如何有效求解的，并给出了本模型的优点。

接着，对于本模型在 TSP 问题中的应用给出了编程实现，并给出了程序的伪代码算法。最后将编写的程序部署在 E-MapReduce 大数据分析计算平台上进行试验，在实验的过程中，本文选取了传统的 GA 模型，伪分布式的 GA 模型与本文提出的模型并在不同的初始总群规模下进行详尽的比较，得到数据表并对数据进行分析论证，最终证明了模型的可行性。最后给出了在实际的生产情况下，如果高效的应用本模型进行实际问题的求解。

第 6 章 结论与展望

6.1 总结

纵览整片文章，本文的核心思想主要基于当今信息数据爆发式增长的环境下，如何对大数据进行有效的处理。Hadoop 大数据分析计算平台的出现为当前面临的问题提供了一个优秀高效的平台。但同时，如何将传统的算法框架和求解方法同当今的大数据平台相结合也成为了当务之急。比如早在 50 年代初期提出的遗传算法，为社会的发展和经济的繁荣起到了不可磨灭的作用。但是，如何将遗传算法应用在大数据的平台上，从而产生一种稳定性高，效率优秀的框架却值得深入研究。本文通过上述背景论述，首先对 Hadoop 大数据分析计算平台进行了深入的研究，了解 Hadoop 的核心优势。比如 Hadoop 的文件存储系统是如何工作的，其分析计算框架 MapReduce 是如何进行并行运算的。此外，还对其生态系统进行深入的研究，比如依托其文件存储系统 hdfs 出现的 Hbase 数据库等。

接下来主要研究传统的遗传算法的原理以及前人们是如何利用遗传算法解决当今社会产生的一些接受的问题，比如在图像识别，天文学领域用传统的建模方式无法解决的问题等。由于遗传算法有着天然的并行计算优势，因此一开始就有很多智者提出了遗传算法并行的可能性。本文基于上述观点首先对遗传算法的整个运算流程做了深入的研究，以及其各种优秀的算子，比如选择复制算子，遗传变异算子等都有深入的讨论。然后针对其并行的优势做了比较深入的探讨和论述，主要在其各种并行方式之间做论证对比，并给出哪种并行结构更有利于适用于当今流行的 Hadoop 大数据分析计算平台。

结合上述两部分的学习和讨论，本文最终进入核心章节。将 Hadoop 大数据分析计算平台与传统的遗传算法相结合并提出了基于 Hadoop 的混合并行遗传算法求解模型。本文的核心章节主要分为四个部分，第一部分主要讨论基于 Hadoop 的并行遗传算法实现的依据，并从多个方面此模型的可行性。第二部分主要论述遗传算法并行的实现模型主要有几种，并对照这几种实现模型进行研究评价。结合上述部分，第三部分主要给出整个论文的核心模型即基于 Hadoop 的混合并行遗传算法，并对此模型进行了详尽的描述和可行性证明。紧接着对本模型的优越性进行分析论证。最后一分部主要是模型的验证阶段，本文采取了对当今经济发展具有重要指导意义的 TSP 即旅行商问题模型进行

证明，由于 TSP 问题设计大量的初始求解个体，并且其个体的数量随着城市数量的增多成指数增长。因此用本文提出的模型求解成为了不二之选。本文通过 TSP 问题对模型进行验证最终得到实验数据，通过实验数据证明了本模型的可行性和正确性。

6.2 展望

对于本文研究模型和实验结果，已经很好的验证了模型的可行性，但同时，本模型仍旧有许多可以进行深入优化的点具体优化点如下。

（1）在接下来的框架实验中，着重用来尝试解决实际生产环境下的问题，将更多的真实数据应用到本模型进行试验论证和积累经验。

（2）尝试将模型和其他的大数据计算模型相结合，比如 storm、spark 等大数据分析计算框架。

（3）通过基于本模型为原始模型，进行深入的研究，进行框架的深入优化。

6.3 不足之处

针对本文的研究内容，虽然最后顺利的在基于 Hadoop 的大数据分析计算平台上得到数据并证明了本框架的可行性，但是由于时间和其他因素，仍然有着许多可以改进之处，具体可以不足之处如下。

（1）第一层算法框架和第二层算法框架缺少既定的标准来衡量权重关系。虽然本文在着重论述了整个框架中应该如何分配第一层算法框架和第二层算法框架的关系，但是始终没有一个既定的可以衡量的标准来定量的确定其各自权重的大小。

（2）在整个框架中 MapReduce 计算框架中 shuffle 阶段没有一个既定的优化方案。MapReduce 是 Hadoop 大数据分析计算平台的核心计算框架。这个框架整体是并行运行的，但是再 MapReduce 计算框架之间还有一个阶段就是 shuffle 阶段，这个阶段的作用就是对 Map 阶段计算的值进行排序和分组。因此，能对这一阶段的计算框架进行有效的优化，将会极大的提高整个算法的计算效率。

（3）虽然本文提出的基于 Hadoop 的混合并行遗传算法在本文的实验数据下验证成功，但是由于缺少大量的实际生产环境下的数据验证。因此本模型在面临实际生产问题的时候仍然需要根据实际条件进行改进和测试。

参考文献

[1] 孟小峰, 慈祥. 大数据管理: 概念, 技术与挑战[J]. 计算机研究与发展, 2013, 50(1): 146-169.

[2] 郭艳霞, 颜军. 海量数据存储模式的研究[J]. 计算机与数字工程, 2008, 36(11): 162-165.

[3] 黄少荣. 遗传算法及其应用[J]. 电脑知识与技术. 2008(34).

[4] Jin C, Vecchiola C, Buyya R. MRPGA: an extension of MapReduce for parallelizing genetic algorithms[C]//eScience, 2008. eScience'08. IEEE Fourth International Conference on. IEEE, 2008: 214-221.

[5] 邓传华, 范通让, 高峰. 一种基于 Hadoop 的作业转移调度算法[J]. 河北省科学院学 报, 2012, 29(2).

[6] 陈章辉, 黄小晖, 任文艺, 等. 基于双倍体遗传算法求解大学排课问题[J]. 计算机应用, 2008, 28(12): 3074-3076.

[7] Kohonen T. Self-organized formation of topologically correct feature maps[J]. Biological cybernetics, 1982, 43(1): 59-69.

[8] 康立山, 胡能发. 一种采用整数编码的全局优化算法[J]. 湖北大学学报: 自然科学版, 2002, 24(2): 123-126.

[9] JH Holland. Adoption in Natural and Artificial System. MIT Press. 1975.6(2):126–137.

[10] DeJong K. An analysis of the behavior of a class of genetic adaptive systems[J]. Ph. D. Thesis, University of Michigan, 1975.

[11] 梁宇宏, 张欣. 对遗传算法的轮盘赌选择方式的改进[J]. 信息技术, 2009 (12): 127-129.

[12] 夏祎. Hadoop 平台下的作业调度算法研究与改进[D]. 华南理工大学, 2010.

[13] 王彦明, 奉国和, 薛云. 近年来 Hadoop 国外研究综述[J]. 计算机系统应用, 2013 (6): 1-5.

[14] Ghemawat S, Gobioff H, Leung S T. The Google file system[C]//ACM SIGOPS operating systems review. ACM, 2003, 37(5): 29-43.

[15] 郝树魁. Hadoop HDFS 和 MapReduce 架构浅析[J]. 邮电设计技术, 2012, 7: 37-42..

[16] 陈吉荣, 乐嘉锦. 基于 Hadoop 生态系统的大数据解决方案综述[J]. 计算机工程与科学, 2013, 35(10): 25-35.

[17] Pig. https://cwiki.apache.org/confluence/display/PIG/Index.

[18] Carstoiu D, Lepadatu E, Gaspar M. Hbase-non sql database, performances evaluation[C]//in Computer Science (1986), Master in Computer Science (1990), and PhD in Computer Science. 2010.

[19] 王悦. Hive 日志分析的大数据存储优化探讨[J]. 信息通信, 2015 (10): 130-131.

[20] S. Owen,R. Anil,T. Dunning,E. Friedman.Mahout in action. . 2011.

[21] Apache Software Foundation.Apache Zookeeper. http://Zookeeper.apache.org/ . 2013.

[22] YARN. http://hadoop.apache.org/docs/r2.2.0/hadoop-yarn/hadoop-yarn-site/ YARN.html.

[23] 朱颂. 分布式文件系统 HDFS 的分析[J]. 福建电脑, 2012, 28(4): 63-65.

[24] Zhang Y, Gao Q, Gao L, et al. imapreduce: A distributed computing framework for iterative computation[J]. Journal of Grid Computing, 2012, 10(1): 47-68.

[25] 易小华, 刘杰, 叶丹. 面向 MapReduce 的数据处理流程开发方法[J]. 计算机科学与探索, 2011, 5(2): 161-169.

[26] 刘琨, 李爱菊, 董龙江. 基于 Hadoop 的云存储的研究及实现[J]. 微计算机信息, 2011, 27(7): 220-221.

[27] 谌超, 强保华, 石龙. 基于 Hadoop MapReduce 的大规模数据索引构建与集群性能分析[J]. 桂林电子科技大学学报, 2012, 32(4): 307-312.

[28] 田秀霞, 周耀君, 毕忠勤, 等. 基于 Hadoop 架构的分布式计算和存储技术及其应用[J]. 上海電力學院學報, 2011, 27(1): 70-74.

[29] 陈皓, 崔杜武, 崔颖安, 等. 族群进化算法[J]. 软件学报, 2010, 21(5): 978-990.

[30] 陈国良等编著.遗传算法及其应用[M]. 人民邮电出版社, 1996.

[31] Coello C A C. Evolutionary multi-objective optimization: a historical view of the field[J]. IEEE computational intelligence magazine, 2006, 1(1): 28-36.

[32] da Fonseca Neto J V, Bottura C P. Parallel genetic algorithm fitness function team for eigenstructure assignment via LQR designs[C]//Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on. IEEE, 1999, 2.

[33] 林丹, 寇纪淞. 遗传规划研究与应用中的若干问题[J]. 管理科学学报, 1999, 2(4): 62-69.

[34] Holland J H, Goldberg D E. Genetic Algorithms in Search, Optimization and Machine Learning[J]. 1989.

[35] 侯广坤, 骆江鹏. 一种理想并行遗传算法模型[J]. 软件学报, 1999, 10(5): 557-560.

[36] 岳嵚, 冯珊. 遗传算法的计算性能的统计分析[J]. 计算机学报, 2009 (12): 2389-2392.

[37] 江中央, 蔡自兴, 王勇. 求解全局优化问题的混合自适应正交遗传算法[J]. 软件学报, 2010, 21(6): 1296-1307.

[38] Dorigo M. Optimization, learning and natural algorithms[J]. Ph. D. Thesis, Politecnico di Milano, Italy, 1992.

[39] 朱林杰. 基于 TSP 的遗传算法优化研究 [D][D]. 大连: 大连理工大学, 2007.

[40] 恩瑜, 数学教授, 仕平. 数学规划与组合优化[M]. 浙江大学出版社, 2001.

[41] 李雪飞, 刘经南, 傅佩红. 一种适用于物流配送系统的 TSP 算法[J]. 測繪通報, 2006, 2006(9): 53-56.

[42] 王英力, 庄奕琪. 一种新型异步 FIFO 的设计[J]. 电子设计应用, 2007 (9): 84-84.

[43] 肖青青, 赵娜. 对动态规划法和贪心法的分析和比较[J]. 现代商贸工业, 2009, 2: 130.

[44] 王凌. 智能优化算法及其应用[M]. 清华大学出版社, 2001.

[45] 江加和, 宋子善, 沈为群, 等. 模拟退火算法在连续变量全局优化问题中应用[J]. 北京航空航天大学学报, 2001, 27(5): 556-559.

[46] 邹哲讷. 贪心算法及其应用[J]. 计算机光盘软件与应用. 2015(03).

[47] 付欣,李静. 蚁群算法[J]. 硅谷. 2012(03).

[48] Thang N, Moon B R. A new genetic approach for the traveling salesman problem[C]//Proc. IEEE Int. Conf. On Evolutionary Computing: Orlando. 1994.

[49] 郭彤城, 慕春棣. 并行遗传算法的新进展[J]. 系统工程理论与实践, 2002, 22(2): 15-23.

[50] E.Paz.A survey of parallel genetic algorithms. Illigal Report, No.97003. 1997.

[51] 吉培荣, 赵青, 刘鹄, 等. 一种新的伪并行遗传算法[J]. 广西师范大学学报: 自然科学版, 2006, 24(4): 143-146..

[52] 侯广坤, 骆江鹏. 一种理想并行遗传算法模型[J]. 软件学报, 1999, 10(5): 557-560.

[53] 郭绚,石晓虹. 并行遗传算法的性能分析[J]. 航空计算技术. 1998(03).

[54] 李想, 魏加华, 傅旭东. 粗粒度并行遗传算法在水库调度问题中的应用[J]. 水力发电学报, 2012, 31(4): 28-33.

[55] 殷新春, 仇亮. 基于主从式并行遗传算法的 S 盒优化算法[J]. 计算机工程与应用, 2008, 44(24): 112-114.

[56] Kohlmorgen U, Schmeck H, Haase K. Experiences with fine‐grainedparallel genetic algorithms[J]. Annals of Operations Research, 1999, 90: 203-219.

[57] Muhlenbein H. Evolution in time and space-the parallel genetic algorithm[C]//Foundations of genetic algorithms. 1991.

[58] 王学瑞. 函数式编程语言发展及应用[J]. 计算机光盘软件与应用, 2012 (23): 181-182.

[59] Keco D, Subasi A. Parallelization of genetic algorithms using Hadoop Map/Reduce[J]. SouthEast Europe Journal of Soft Computing, 2012, 1(2).

[60] Qinghua L, Shida Y, Youlin Y. Improving optimization for genetic algorithms based on level set[J]. Journal of Computer Research and Development, 2006, 43(9): 1624-1629.

[61] 林伟伟. 一种改进的 Hadoop 数据放置策略[J]. 华南理工大学学报: 自然科学版, 2012, 40(1): 152-158.

[62] 穆艳玲. 遗传算法在 TSP 问题中的应用[D]. 天津师范大学, 2004.

[63] 吴少岩, 许卓群. 遗传算法中遗传算子的启发式构造策略[J]. 计算机学报, 1998, 21(11): 1003-1008.

[64] Konstantin Shvachko,Hairong Kuang,Sanjay Radia.The Hadoop Distributed File System. Proceedings of the26th IEEE Symposium onMass Storage Systems and Technologies (MSST’’10). 2010.

[65] Lee M C. A novel sentence similarity measure for semantic-based expert systems[J]. Expert Systems with Applications, 2011, 38(5): 6392-6399.

[66] 傅杰, 都志辉. 一种周期性 MapReduce 作业的负载均衡策略[J]. 计算机科学, 2013, 40(3): 38-40.

攻读研究生期间发表论文

[1] Lilu Cao, Dashen Xue. Research on Modified Artificial Bee Colony Clustering Algorithm, 2015 International Conference on Network and Information Systems for Computers. Hubei, Wuhan, China, 2015. 231-235. (EI 检索)

致谢

时光荏苒，研究生生活一晃而过。我深切的感受到，研究生生活将会给我即将要踏上的征途中增添无尽的财富。

在这里，我首先要感谢我的研究生导师薛大伸教授，是他在这三年的研究生生活中给予了我无私帮助。尤其在最后的论文指导中，他一丝不苟的治学精神、深邃的洞察力、远见卓识的战略思想以及强调学以致用的实事求是精神，无不感染着我时刻提醒自己要以一种慎独的态度对待我的毕业论文以及将来的事情。

另外，感谢我的导师在我读研期间给我创造的实习机会，还有给别人做指导的机会，这些经历无不在我的自我成长过程中起到了重要的作用。

我还要感谢在我读研期间给予各种指导的教授，老师们，感谢你们的时间付出，感谢你们的治学精神以及你们对我生活上的帮助和关怀。没有你们，我也许会走更多的弯路，也许会被前进道路上的困难阻碍不得前进。

还有，我的同门师兄，师姐，师弟，师妹们，感谢你们在我的研究生期间的各种支持和关怀。也感谢有你们陪伴的在研究生期间一块度过的快乐时光。在这里衷心祝福你们顺利完成研究生学业，事业有成。

感谢我的母校，大连海事大学，感谢你给了我一个发奋进取的平台。

感谢我的家人，尤其感谢我父母，感谢你们无时不刻的关怀与支持。

感谢导师以及答辩委员会的各位专家、教授、老师以及同我一起并肩作战的同学、朋友以及给予我宝贵意见的各位。