云南财经大学前沿研究丛书

数据密集型计算环境下贝叶斯网的学习、推理及应用

SHUJU MIJIXING JISUAN HUANJING XIA
BEIYESIWANG
DE XUEXI TUILI JI YINGYONG

马　冯◎著

大学出版社

▶云南财经大学前沿研究丛书
云南财经大学博士学术基金资助出版

数据密集型计算环境下贝叶斯网的学习、推理及应用

SHUJU MIJIXING JISUAN HUANJING XIA BEIYESIWANG DE XUEXI TUILI JI YINGYONG

马　冯◎著

西南交通大学出版社
·成　都·

图书在版编目（CIP）数据

数据密集型计算环境下贝叶斯网的学习、推理及应用 / 马冯著. —成都：西南交通大学出版社，2016.4
（云南财经大学前沿研究丛书）
ISBN 978-7-5643-4511-2

Ⅰ. ①数… Ⅱ. ①马… Ⅲ. ①贝叶斯推断－研究
Ⅳ. ①O212

中国版本图书馆 CIP 数据核字（2016）第 012207 号

云南财经大学前沿研究丛书

数据密集型计算环境下贝叶斯网的学习、推理及应用

马　冯　著

责任编辑	周　杨
特邀编辑	何　桥
封面设计	墨创文化
出版发行	西南交通大学出版社 （四川省成都市二环路北一段 111 号 西南交通大学创新大厦 21 楼）
发行部电话	028-87600564　028-87600533
邮政编码	610031
网　　址	http://www.xnjdcbs.com
印　　刷	成都蜀通印务有限责任公司
成品尺寸	170 mm × 230 mm
印　　张	6
字　　数	138 千
版　　次	2016 年 4 月第 1 版
印　　次	2016 年 4 月第 1 次
书　　号	ISBN 978-7-5643-4511-2
定　　价	42.00 元

序　言

随着网络应用的普及和信息采集技术的进步，人类生产和收集数据的能力迅速发展，人们需要面对的数据量也日益增长。这些数据通常呈现出数据量巨大，且分布在多个站点上的情况。数据密集型计算（Data-Intensive Computing）的出现，使得对这种新情况下的大数据进行有效处理成为可能。数据密集型计算指能推动前沿技术发展的对海量和高速变化的数据的获取、管理、分析和理解。目前它已经成为了数据研究和分析领域中的一个研究热点问题。

贝叶斯网络（Bayesian Network，BN），是概率理论和图论相结合的产物。它是一种帮助人们将概率、统计理论应用于复杂领域，进行不确定性推理和数据分析的有效工具。然而由于传统的贝叶斯网络在对数据进行处理时，默认是将所有的数据置于同一个站点之上，因此在数据密集型的环境之下，就很难将传统的贝叶斯网络的相关理论和方法直接运用于其上，因此，对传统的贝叶斯网络进行扩展，使得其相关的理论、方法和结论可以运用于数据密集型计算环境之下，就显得十分必要。

本书的主要工作和创新之处总结如下：

（1）对传统的贝叶斯学习方法进行扩展。在数据密集型的计算环境下，数据通常都呈现出大量且分布于多个站点上的情况，因此需要对传统的贝叶斯网学习方法进行适当的扩展，使得它能够应用于新的环境之下。由于贝叶斯网的构建通常分为参数学习和结构学习两个部分，而如果在已经确定了贝叶斯网结构的情况下，分布式的参数学习的扩展方法会变得相对比较容易，因此本书重点讨论分布式环境下的结构学习方法。

（2）对传统的贝叶斯推理方法进行扩展。传统的贝叶斯推理方法，也是默认所有的数据集在同一个站点上时进行的。在数据密集型计算环境下，如果在各个站点上分别利用传统的贝叶斯推理方法进行推理，那么推理出的结果仅仅适用于各个站点，而不同站点之间由于数据内容不尽相同，推理出的结果很可能不完全一致，甚至会出现部分冲突的现象。因此本书选取一种常用的贝叶斯推理方法进行扩展，这种方法选取了吉布斯抽样（Gibbs Sampling）为随机算法的核心组成内容，并最终获取一个适用于全体数据集合的推理结

果。此算法的有效性在书中后面的章节中给予了理论证明和实验验证。

（3）提出一种数据密集型计算环境下的贝叶斯网具体应用——社区发现。社区发现是近些年来的一个研究热点问题，书中提出了一种利用关联规则发现过程中的频繁项目集，来构建相应的网络，并最终进行社区发现的方法。这个方法有两个优点：一方面它可以直接应用于数据密集型计算环境之下，从而扩展了传统贝叶斯网的应用范围；另一方面，它又充分利用了关联规则发现过程中的频繁项目集信息，构建了一个能反映隐关系的网络，并在其上进行社区发现。

最后，感谢云南财经大学的支持。

编　者

2015 年 10 月

目 录

第1章 绪论

本章主要介绍整篇文章的相关研究背景及意义，并简要说明文章所做的主要研究工作，最后给出全书的组织结构。

1.1 研究背景及意义

近年来，随着信息技术的发展，计算机领域中涌现出一个新的研究热点，即数据密集型计算（Data-Intensive Computing）。美国能源部太平洋西北国家实验室[1]针对数据密集型计算给出了这样的定义："Data Intensive Computing is capturing, managing, analyzing, and understanding data at volumes and rates that push the frontiers of current technologies"。也就是说，数据密集型计算是能推动前沿技术发展的对海量和高速变化的数据的获取、管理、分析和理解。这个定义包含了三层含义[2, 3]：首先，数据密集型计算所处理的对象是数据，是一种围绕着数据而展开的计算。数据密集型计算需要处理的数据量非常巨大，且快速变化，这些数据往往是分布的、异构的。因此，传统的数据库管理系统已经不能满足它的需要了。其次，数据密集型计算中的"计算"，包括了从数据的获取到管理再到分析、理解的整个过程。因此，数据密集型计算既不同于数据检索和数据库查询，也不同于传统的高性能计算和科学计算。它是传统数据管理、数据分析和挖掘以及高性能计算的结合。第三，数据密集型计算的目的是推动技术前沿发展，推动那些依赖传统的单一数据源、准静态数据库所无法实现的应用。在数据密集型计算环境下，需要处理的大量数据，往往会分布在多个不同的站点之上。

贝叶斯网是一种表示变量之间依赖关系的有向图模型[4, 5]。在贝叶斯网中，结点表示问题域中的随机变量，有向边则表示这些随机变量之间的依赖关系，其具体依赖程度用条件概率进行定量表示[6, 7]。贝叶斯网巧妙地将图形理论和概率理论结合在了一起，其图形模型对于实体之间依赖关系提供了一种直观、紧凑、有效的表示方法；而概率理论为在贝叶斯网上进行推理提

供了坚实的数学基础，也给学习贝叶斯网的定量特征提供了必要的计算方法。在知识系统中，贝叶斯网不仅可以发现数据间潜在的关系，处理不完整的数据集，甚至可以在一定程度上解决数据间不一致的问题。贝叶斯方法以其特有的不确定性知识表示形式和丰富的概率表达能力，成为了不确定知识表示和推理的典型工具，并逐步应用于分类、预测和决策、信息回复、专家系统等领域[8-18]。

传统的贝叶斯网，其构建网络时，默认是将所有的数据存放在某一个站点上，然而，在数据密集型计算环境下，数据通常是分布在多个站点之上的。这种情况的出现通常有两个原因：一种是由于数据采集技术的发展，使得通过采集得到的需要处理的数据量相当大，远远超过了一台计算机所能够处理或存储的范围；另一种原因是，由于有些数据在采集的时候，其来源在地理上自身就是分布的，为了管理等方面的需要，也就进行了分布式的存储。总而言之，在这种数据密集型计算的环境之下，传统的贝叶斯网络，其原有的十分成熟的那套学习理论和推理方法，就很难直接应用于新的环境之下，如果希望在数据密集型计算环境之下，仍然使用贝叶斯网络进行推理或进行其他工作，则需要对传统的贝叶斯网理论进行一定程度的调整和扩展。

1.2 研究现状分析

1.2.1 数据密集型计算的研究现状

近年来，数据密集型计算已经成为了国内外研究领域中的一个热点问题，国内外很多公司根据数据密集型计算的特点，都相继提出并实现了处理海量数据的相关基础设施[19-24]，以应对新环境下所面临的数据管理和分析等相关问题。

国外，较为具有代表性的有谷歌，雅虎（连同 Apache 公司）和微软公司的相关产品。谷歌公司较早地提出了一种称为 MapReduce 的编程框架，该框架可以在数据密集型环境下，通过 Map 和 Reduce 两个主要关键步骤，并行地计算和处理大规模的数据。该技术在谷歌的搜索引擎中被广泛使用并已被事实证明为一个非常有效的数据处理工具[25]。雅虎公司连同 Apache 公司，后续推出了一种名为 Hadoop 的项目平台[26, 27]。该平台有两个比较明显的优点，一方面，它同样实现了谷歌提出的 MapReduce 编程思想，并且已经在 Facebook、Twitter 等众多大型的生产性系统中部署后，也是被现实证明了其有效性的一个编程平台。另一方面，它是一个开源的平台，因此使得感兴趣

的用户在其基础上研究或编写相关的分布式应用成为可能。因此，学术界也针对 Hadoop 项目平台进行了很多研究，并取得了不少有用的研究成果。微软公司研发了一种名为 Dryad[28]的数据并行处理通用执行引擎，它是一种在数据密集型计算环境下的具体应用并具有很强的扩展性。此外，太平洋西北国家实验室成立了专门的实验机构以便于研究数据密集型计算环境下的海量数据管理、分析和处理等问题。

国内，也相继出现了很对针对数据密集型计算的应用。如国内大型电子商务网站淘宝网的 OceanBase 是一个具有对海量数据进行实施分析功能的有效应用，腾讯公司开发了一种名为 Typhoon 的云平台，其主要目标是在数据密集型计算环境下，为腾讯搜搜等需要处理大规模数据的具体应用提供较好的平台。另外，华为、中兴等企业还进行了基于 NoSQL 的相关研究开发等[29, 30]。

综上所述，目前国内外的各种研究和具体的应用状况表明，数据密集型计算在理论研究和实践应用方面都有了较好的基础条件，因此在数据密集型计算环境下对传统的贝叶斯网进行扩展研究就成为了可能。

1.2.2 贝叶斯网的研究现状

贝叶斯网（Bayesian Network，BN）是一种概率理论和图论相结合的产物，通常可以用来进行数据分析和不确定性知识的推理[4, 5]。贝叶斯网兴起于 20 世纪 80 年代，是在人工智能领域中针对不确定性知识发现的研究过程中出现的，随着不断发展，它已经在很多领域都产生了较为重要的影响[6, 7]。

然而在近些年来，随着数据密集型计算方式的出现，贝叶斯网络的研究相对变得较为缓慢，很大程度上还停滞在单机环境的状态下。面对目前的大规模数据，传统的贝叶斯网络往往会显示出扫描、计算速度慢，学习数据不够灵活等问题。而其传统的构建、推理等方法更加很难直接适用于数据密集型计算环境下，呈现出海量分布和快速变化等特点的数据[31-35]。

1.2.3 研究内容的必要性

不确定性知识的发现一直以来都是数据分析等研究领域中的一个重要问题。贝叶斯网曾在单机状态的情况下，较好地表现出了它在这个方面的能力。虽然在数据密集型计算环境之下，传统贝叶斯网的学习、推理和应用等相关理论和方法无法直接应用于其上，但现有研究成果显示，已经有专家将传统的机器学习和进化算法扩展到基于 MapReduce 模型并予以了实现[36, 37]，这就

意味着利用 MapReduce 等相关技术，对贝叶斯网在数据密集型计算环境下进行扩展并用于表示不确定性知识是存在可能的。另外，由于目前仅有部分研究针对数据密集型计算环境下的贝叶斯网学习进行了讨论[38-40]，对于贝叶斯网的增量学习、推理和具体应用等工作，尚未见到有相关的报道，因此本书的研究工作，在一定程度上可以在此领域中为后续的研究提供一些相关的思路。

1.3 全书主要工作

本书主要研究数据密集型计算环境下贝叶斯网的构建，推理和应用问题，主要工作包括以下几个方面：

（1）数据密集型计算环境下的贝叶斯网学习方法。数据密集型计算环境下，贝叶斯网的学习方法大致可分为两类，静态构建和动态构建。静态构建，即指将所有站点上某一时刻需要处理的数据，其相关信息先统计后汇总，最后建立起一个适用于当前时刻数据的贝叶斯网。动态构建，是指在静态构建贝叶斯网后，若某一个或某几个站点上的数据有更新时，可以仅仅通过对更新数据信息的采集和处理，动态地调整之前建立好的贝叶斯网，使其适用于新的数据，而无需对全体数据集再重新处理一遍的学习方法。本书会详细介绍后一种，即动态学习方法。

（2）数据密集型计算环境下的贝叶斯网推理方法。贝叶斯的推理方法大致可以分为两类，精确推理和近似推理。一般来说，精确推理算法取决于贝叶斯网络的拓扑结构，而近似推理算法的性能往往取决于实际的概率分布。在近似推理中，随机算法推理，通常被公认为是一种十分有效的推理方法。本书提出了一种以改进后的吉布斯抽样方法为核心，适用于数据密集型计算环境的随机算法推理方法。

（3）数据密集型计算环境下贝叶斯网的应用——社区发现。社区发现也是近年来的研究热点之一。书中提出了一种以数据密集型计算环境下的贝叶斯网为基础，进行具有潜在关系的社区发现的方法，另外在社区发现的过程中，对于带有部分重叠社区的情况，进行了专门的讨论。

1.4 全书组织结构

全书共分为 5 章。第 1 章为绪论，介绍了研究背景及意义、全书的主要工作、组织结构；第 2 章介绍了传统贝叶斯网的概念，数据密集型计算的出

现，以及数据密集型计算对传统贝叶斯网的影响；第 3 章介绍数据密集型计算环境下贝叶斯网的学习方法，重点说明在数据密集型计算环境下如何进行贝叶斯网的动态构建；第 4 章介绍数据密集型计算环境下贝叶斯网的推理方法，重点讨论了在数据密集型计算环境下，使用改进后的随机算法进行近似推理的过程；第 5 章介绍数据密集型计算环境下贝叶斯网的应用——社区发现，讨论了如何在数据密集型计算环境下，利用贝叶斯网络发现具有潜在关联的社区，并针对具有重叠区域的社区发现问题进行了专门的讨论；结束语是对全书的总结。

第 2 章　数据密集型计算和贝叶斯网

本章主要介绍数据密集型计算的相关概念、面临的挑战和典型的应用，以及贝叶斯网的相关概念、学习、推理方法和应用，作为本书后续章节的背景知识。

2.1　数据密集型计算简介

数据密集型计算是近些年出现的一个比较热门的研究领域，下面将从它的概念、面临的挑战和典型应用三个方面对其进行介绍。

2.1.1　数据密集型计算概念

由于互联网的普及和信息采集技术的飞速发展，人们可以获得的数据量也在急速增长，信息集成技术可以使得来自于不同地域、不同时间、不同领域的数据聚集在一起。在传统的信息管理系统中，这些数据是一种“被管理者”的角色，而如今它们已经逐渐演变成了很多应用的核心内容之一，其地位在不断提升。由于这些数据通常呈现出数据量巨大且地理上往往是分布式存储的特点，因此采用传统的方法已经无法有效地处理它们了，如何有效地利用这些数据已经成为了当前人们面临的一个不可回避的挑战。在这种背景下，数据密集型计算应运而生。

关于数据密集型计算，美国能源部太平洋西北国家实验室[1]给出了这样的定义：“Data Intensive Computing is capturing，managing，analyzing，and understanding data at volumes and rates that push the frontiers of current technologies”。也就是说，数据密集型计算，是指能推动前沿技术发展的对海量和高速变化的数据的获取、管理、分析和理解。这个定义包含了三层含义[2]：

（1）数据密集型计算是一种围绕着数据而展开的计算，它所处理的对象

正是新环境下的数据。数据密集型计算需要处理的数据，其特点是数量巨大且地理上往往是分布式存储的。这些数据往往都是传统的数据处理方法很难以一种高效的方式进行处理的对象。

（2）数据密集型计算中的“计算”，包括了从数据的获取到管理再到分析、理解的整个过程。也就是说，数据密集型计算既不同于传统的高性能计算和科学计算，也不同于数据检索和数据库查询。它是一种将传统数据管理、高性能计算和数据分析挖掘相结合后的产物。

（3）数据密集型计算的目的是推动技术前沿发展，推动那些依赖传统的单一数据源所难以实现的应用。

2.1.2 数据密集型计算面临的挑战

目前数据密集型计算领域中所面临的挑战有很多，如数据的分布性问题[24,41]，对错误的检测和容忍能力问题[42]，用户进入的灵活性和高效性问题[43]，数据布局问题[44, 45]等。由于本书重点研究的是贝叶斯网在数据密集型环境下的学习、推理和应用等相关问题，因此本章仅仅选取了与后续内容相关的部分进行描述，其具体内容如下。

2.1.2.1 数据的分布性

数据密集型计算的一个显著特点就是：数据除了数量庞大外，还通常都是分布在多个不同的站点上的。为了应对这样的特殊环境，业内专家已经提出了通过文件系统、编程平台、存储和数据库系统以及数据仓库系统等多种方式来为这种特性提供必要的技术支撑。下面就来详细介绍其中的部分内容。

1）文件系统

文件系统的分布性决定了其存储和处理大数据的能力。分布式的文件系统，如谷歌文件系统（GFS）和 Hadoop 分布式文件系统（HDFS），为数据密集型计算环境下的数据分布式处理，提供了必要的数据存储基础。

GFS 和 HDFS 有许多相似之处，例如它们在存储大数据时，都是采用以 64 M 为基本大小来将数据划分成若干块。这样的划分，可以在查询时确保较低的查询时间，从而提高查询效率。另外在 HDFS 中，特定节点（称为数据节点）存储所有数据。元信息（Meta information）存储在名称节点中，以便于提供查找服务。每个数据块在默认情况下，在数据节点上复制三次。高度的复制可以提高数据的可用性和数据的本地性，即一种计算任务更加偏向于

在距离本地较近的位置执行的概念。这就在一定程度上缓解了网络瓶颈，并很好地支持了数据的分布性。GFS 和 HDFS 的主要区别在于，GFS 是针对谷歌平台的一种文件系统管理方式，而 HDFS 则是在 Hadoop 下的一种分布式文件系统。

2）编程平台

MapReduce[46-49]是目前数据密集型计算平台上最流行的程序设计语言之一。由于本书稍后的很多实验都用到了 MapReduce 的相关技术，因此，下面就对有关 MapReduce 的相关技术内容进行简要介绍。

谷歌公司最先提出了 MapReduce 的概念，并以 GFS 为基础将其具体实现以便于有效地处理大规模的数据。然而，谷歌并没有全部公开其整套系统的内部细节实现情况，这就使得除谷歌以外的公司或研究人员，很难利用 MapReduce 的技术进行相关研究和应用的开发。直到后来开源社区以 Hadoop[50]项目的形式实现了 MapReduce 功能，并以开源的方式提供给所有感兴趣的研究人员，这才使得学术界对于 MapReduce 的研究取得了较大的发展。MapReduce 是一种设计优良的处理大数据的软件架构，以它为基础写出的程序能够在具有上千个节点的大型集群上顺利地运行，并同时可以并行地处理数据量巨大的数据集。

在 MapReduce 架构中，一个输入的作业通常会被先切分成若干个较小的数据块，再由系统以并行的方式去处理这些数据块[51]。MapReduce 中的 Map 常常被翻译为映射，而 Reduce 则被翻译为化简。Map 的输入内容通常会先被按照某种策略进行排序后，再将结果输出给 Reduce 部分。这样的输入和输出，通常都需要被存储在适当的文件系统中，而 MapReduce 的框架结构则负责对任务进行整体的监控和调度，以及负责在发现任务失败后将其重新执行的工作。

在 MapReduce 的体系中，计算节点就是 MapReduce 的框架部分，而存储节点就是指相应的分布式文件系统。在实际运行时，这两者通常都是被置于同一个或同一组节点之上的。这样做的目的，是希望在已经存有数据的节点上，可以使得任务得以高效地执行，减少节点间的数据传输，降低整个集群内部网络的拥挤程度，从而提高执行效率。

MapReduce 框架通常是由若干个从机任务追踪器（Slave TaskTracker）和一个独立的主机作业追踪器（Master JobTracker）共同组成[52]。一个作业通常会被分成若干个任务，而主机作业追踪器主要负责将这些任务分配给不同的从机进行执行。同时，主机作业追踪器还要负责监督这些任务的执行过程，若发现有失败的任务，则需要发出命令，重新将其执行一次。从机任务追踪

器则只是负责执行主机作业追踪器分配给它的任务即可。

在实际编程的时候，程序需要实现相应的含有 Map 和 Reduce 函数的抽象类或者是接口，并指明输入输出所对应的具体路径。另外，还要根据具体情况，再添加一些相关的参数，来构建成具体应用的作业配置。然后，作业客户端会将这个配置信息连同作业一起发送给主机作业追踪器。主机作业追踪器则将它们再根据一定的策略分配给从机，然后再监控从机作业追踪器的执行过程。另外，必要的时候，主机作业追踪器还会向作业客户端反馈当前的执行状态。

MapReduce 框架在具体的输入输出内容上，是以<key，value>形式的键值对来处理的。换句话说，在 MapReduce 框架下，输入是一组<key，value>键值对，输出也是一组<key，value>键值对，但是这两组键值对的内容肯定是不同的。

下面举例来解释一个简单的 MapReduce 任务执行过程[53]，其主要流程如图 2-1 和图 2-2 所示。

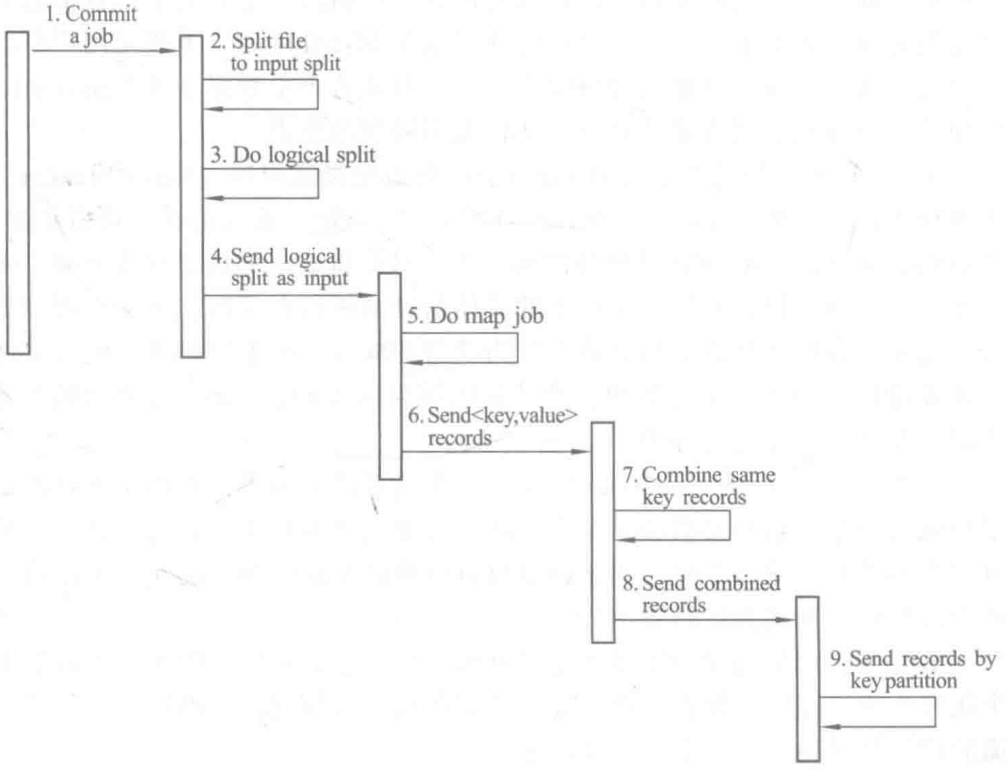

图 2-1　MapReduce 工作流程 1（第 1～9 步）

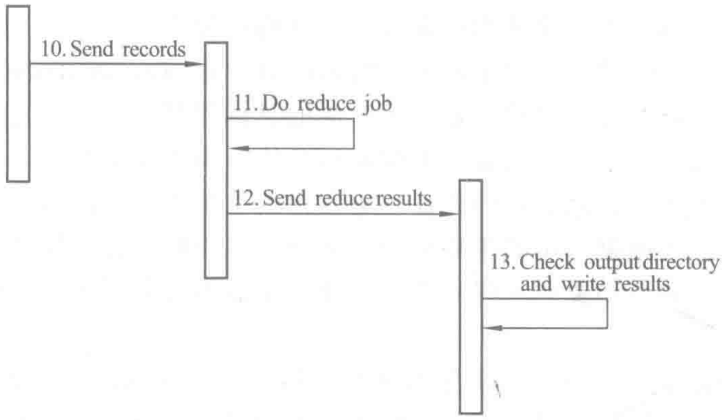

图 2-2　MapReduce 工作流程 2（第 10 ~ 13 步）

第 1 步，在分布式的环境中，作业追踪器（JobTracker）负责将客户端创建的任务进行提交进入系统。

第 2 步，映射（Map）前的预处理操作由输入格式化（InputFormat）模块负责完成。其中主要是指验证输入的内容，在格式上是否满足配置信息中事先设置好的相关定义要求。这种定义可以是 Writable 的子类或是用户自己专门的定义。另外，这里还会将输入的任务分成若干个数据小块 Inputsplit，以便于符合分布式文件系统对单个文件大小限制的要求。

第 3 ~ 6 步，经过划分后的 Inputsplit 交由记录读取器（RecordReader）来进行处理。处理的结果以一组 record 的形式，再作为输入传递给映射步骤。从逻辑上来分，Inputsplit 是划分的第一步，只是将输入的内容初步分解。而记录读取器则可以根据输入文件中的具体信息来进行在逻辑上更加合理的划分。记录读取器处理后的结果将会作为映射的输入，映射环境执行配置文件中预先设定好的策略进行处理，处理后的结果以<key，value>键值对的方式暂时存储在一个临时文件中。

第 7 ~ 8 步，这个 Combiner 模块是一个可选择的步骤。它的主要目的是根据映射执行完后的分析结果，来判断是否要先在本地执行化简工作。如果可以在本地执行化简工作，那么就可以有效地减少后续环节的化简工作量，从而提高整个算法的执行效率。

第 9 步，此处的 Partitioner 也是可选步骤。它的主要目的是，当有若干个化简步骤存在时，是否要指定那个特定的化简来接收映射的结果，并将化简后的结果单独放在一个输出文件中。

第 10 ~ 12 步，化简的具体执行过程。它会将数据处理成用户所需要的形式并输出到下一个步骤，即输出格式化（OutputFormat）。