文章编号:1671-4598(2017)03-0227-04 DOI:10.16526/j.cnki.11—4762/tp.2017.03.061 中图分类号:TP393 文献标识码:A

基于接纳控制和 ILP 资源调度模型的 BDAaaS 系统架构

郑志翔[1], 罗文华[2]
(1. 新疆警察学院 在职教育培训处, 乌鲁木齐 830013; 2. 中国刑警学院 网络犯罪侦查系, 沈阳 110035)

摘要: 为了使云计算平台为大数据分析提供有效支持, 提出一种大数据分析即服务 (BDAaaS) 的系统架构; 首先, 当用户向系统提交大数据分析应用 (BDAA) 时, 通过接纳控制器评估任务的执行时间和成本并作出接纳决策; 然后, 通过服务等级协议 (SLA) 管理器根据任务的服务质量 (QoS) 需求制定 SLA; 最后, 利用提出的整数线性规划 (ILP) 资源调度模型, 以最小化执行成本为目标, 在满足 SLA 下合理调度资源来执行任务; 仿真结果表明, 提出的方案能够有效降低任务执行时间, 具有有效性和可行性。

关键词: 云计算; 大数据分析即服务; 接纳控制; 资源调度; 整数线性规划

A BDAaaS System Architecture Based on Admission Control and ILP Resource Scheduling Model

Zheng Zhixiang[1], Luo Wenhua[2]
(1. Division of In-service Education and Training, Xinjiang Police College, Wulumuqi 830013, China;
2. Department of Cybercrime Investigation, Criminal Police University of China, Shenyang 110035, China)

Abstract: In order to make the cloud computing platform provide effective support for large data analysis, a big data analytics as a service (BDAaaS) system architecture is proposed. First, when a user submits a big data analysis application (BDAA) to the system, the admission controller is used to evaluate the execution time and cost of the task and make an admission decision. Then, the SLA is built by the service level agreement (SLA) manager according to the Quality of Service (QoS) requirements of the tasks. Finally, the resource scheduling model based on ILP with the goal of minimize the execution cost is proposed, and used to schedule resources reasonably under satisfying the SLA. Simulation results show that the proposed scheme can effectively reduce the task execution time, which is effective and feasible.

Keywords: cloud computing; big data analysis as a service; admission control; resource scheduling; integer linear programming

0 引言

大数据分析需要大量的计算和存储资源来分析和存储数据, 且需要特定的大数据分析应用资源[1]。随着云计算的发展, 很大数据分析任务通过云计算来实现。但由于分析任务需求的多样化, 使得大数据分析应用 (Big Data Analytic Application, BDAA) 的资源合理调度变的困难[2]。因此, 需要开发一种云计算平台上的大数据分析即服务 (Big Data Analytics as a Service, BDAaaS)[3]系统架构, 即以大数据分析作为服务对象构建的系统平台。实现以易于使用的方式和较低的价格, 在满足 BDAA 服务等级协议 (service level agreement, SLA)[4]下为 BDAA 用户提供分析服务。

目前, 有学者提出了一些通用的分析框架来处理多用户 BDAA, 并分配所需的分析资源。例如, 文献 [5] 提出了用于大数据分析的 SLA 与成本感知的云资源调度方案 (SLAA)。然而, 其没有对用户应用进行接纳控制, 即不管用户应用有何要求都一并接纳, 这就导致其会违背 SLA, 例如任务截止期限等[6]。SLA 违规会严重影响服务提供商的信用等级和声誉。

为此, 提出了一种用于支持 BDAA 的 BDAaaS 架构, 在满足用户 SLA 下, 合理调度计算资源以最小化服务供应商的执行成本。首先, 通过接纳控制器来接收有能力完成的 BDAA。然后, 利用整数线性规划 (Integer Linear Program, ILP)[7]模型构建一个资源调度系统, 根据 SLA 和执行成本来调度 BDAA 所需的计算资源, 以此达到节约成本的目的。

1 提出的 BDAaaS 平台架构

提出了 BDAaaS 平台的体系结构如图 1 所示, 为不同领域的用户提供分析服务。该体系结构主要由接纳控制器、SLA 管理器和资源调度器组成。

接纳控制器: 用来决定是否接受用户提交的任务。当有用户 BDAA 任务提交时, 接纳控制器首先在 BDAA 注册表中进行搜索, 检查该用户所请求的 BDAA 是否已经存在。如果存在, 则接纳控制器获得 BDAA 的信息。基于该信息, 计算在不同资源配置下的预期执行时间和任务成本。如果存在一种资源配置下, 其执行的时间和成本都满足任务的服务质量 (Quality of Service, QoS) 要求[8], 则接纳控制器做出接收决定, 并将该任务提交给 SLA 管理器。

SLA 管理器: 根据任务 QoS 要求, 为接收的任务建立 SLA。SLA 违规不仅会降低用户满意度, 还会产生额外的惩

收稿日期:2016-01-04; **修回日期**:2016-07-26。
基金项目:公安部技术研究计划项目(2015JSYJC04);辽宁省社科规划基金项目(L16BFX011)。
作者简介:郑志翔(1970-),男,河北秦皇岛人,讲师,硕士,主要从事计算机应用等方向的研究。
通讯作者:罗文华(1977-),男,辽宁沈阳人,教授,硕士,主要从事计算机应用、电子数据取证方向的研究。

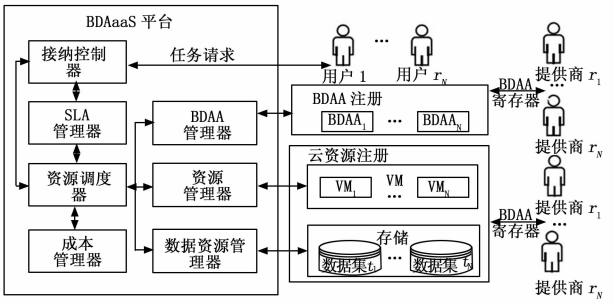

图 1 BDAaaS 平台的体系结构

罚成本[9]，所以需要避免违规。

资源调度器：是 BDAaaS 平台的核心组成部分，用来为 BDAA 做出资源调度决策并协调其他组件。其中，在云计算中，资源通常以虚拟机（Virtual Machine，VM）形成呈现[10]。调度调度器中主要包含：(a) 资源配置，即调度器决定使用哪种类型的 VM，以及该类型 VM 的使用量；(b) VM 控制，即调度器发送 VM 控制命令到资源管理器来控制 VM，例如创建 VM、终止 VM 和移动 VM；(c) BDAA 选择，即调度器选择被请求的 BDAA 来执行任务；(d) 任务执行序列，即将具有相同 BDAA 请求的任务添加到正在执行 BDAA 的 VM 上的等待队列中，由于任务具有截止期限的限制，所以执行它们需避免期限违规；(e) 任务管理，即调度器监控任务的状态，任务状态包含提交、接受、拒绝、等待执行、正在执行、执行成功和失败。同时执行相应的措施，例如，发送执行失败的任务到成本管理器来产生惩罚成本。

成本管理器：管理 BDAaaS 平台生成的所有成本，同时提供价格策略，以此吸引更多的用户来扩大市场份额，产生更高的利润。

BDAA 管理器：管理由不同用户提供的 BDAA，并动态更新 BDAA 信息。

数据源管理器：管理需要处理的数据集。由于大数据具有很高的容量，所以需要对数据进行并行计算，以节省数据传输时间和网络成本。

资源管理器：保持来自不同供应商的所有云资源的目录，并监控 VM 的状态以支持调度器做出调度，如在计费周期结束时终止空闲的 VM 以节约成本。

2 基于 ILP 的资源调度

在提出的 BDAaaS 平台体系结构中，资源调度是关键部分，决定着执行用户任务的有效性和经济性。资源调度的目标为：在满足任务 SLA 下，合理调度云平台中的资源来执行 BDAA 任务，并最小化执行成本。本研究基于 ILP 模型对资源调度进行公式化并建模，在满足任务的 SLA 下，最大限度地减少资源成本和最大化 VM 使用率。

2.1 ILP 模型的构建

资源调度问题可转化为一个多目标 ILP 问题[11-12]，其优化目标为最小化资源成本。资源成本由三个单独的目标（A、B、C）组成，分别描述如下。

目标 A：将任务分配到现有 VM，并使其以最大能力执行，以此提高资源使用率。目标 A 如公式（1）所示，其中 n 表示创建的 VM 的数量；m 为需要调度的任务的数量；L_i 为任务所请求的资源；x_{ij} 表示是否将 $Query_i$ 分配到 VM_j（如果 $Query_i$ 分配到 VM_j，则 $x_{ij}=1$；否则 $x_{ij}=0$。

$$A=\max\left(\sum_{i=1}^{n}\sum_{i=1}^{m}(L_i*x_{ij})\right) \tag{1}$$

目标 B：优秀在成本较低的 VM 上执行任务，以此来降低或移除成本较高 VM 上的荷载，致使其停止运行，从而节约资源成本。目标 B 如公式（2）所示，其中，C_j 为 VM_j 的成本（按小时建模），y_j 表示 VM_j 是否被终止（如果 VM_j 被终止，则 $y_j=0$；否则 $y_j=1$）。

$$B=\max\left(-\sum_{j=1}^{n}(C_j*y_j)\right) \tag{2}$$

目标 C：尽可能快速地执行任务，以此降低 VM 的运行时间，从而节约成本并降低任务响应时间[13]。目标 C 如公式（3）所示，其中，s_i 为任务 $Query_i$ 的起始时间。

$$C=\max\left(-\sum_{i=1}^{m}s_i\right) \tag{3}$$

另外，以上 3 个目标的重要性顺序为 $A>B>C$。结合这 3 个独立目标，综合形成了目标 D，如公式（4）所示，并服从约束条件（5）～（13）。ILP 模型定义如下：

目标：

$$D=\max\left(F_0*\sum_{j=1}^{n}\sum_{i=1}^{m}(L_i*x_{ij})-F_1*\sum_{j=1}^{n}(C_j*y_i)-\sum_{i=1}^{m}s_i\right) \tag{4}$$

服从约束：

$$\sum_{i=1}^{m}(L_i*x_{ij})\leqslant CP_j,\forall\, i\in m,j\in n \tag{5}$$

$$b_{ik}+b_{ki}\leqslant 1,\forall\, i,k\in m \tag{6}$$

$$b_{ik}+b_{ki}-x_{ij}-x_{kj}\geqslant -1,\forall\, i,k\in m;j\in n \tag{7}$$

$$s_i-s_k+F_2*b_{ik}\leqslant F_2-e_{ij},\forall\, i,k\in m;j\in n \tag{8}$$

$$s_i+F_3*x_{ij}\leqslant F_3+D_i-e_{ij},\forall\, i\in m,j\in n \tag{9}$$

$$C_{ij}*x_{ij}\leqslant B_i,\forall\, i\in m,j\in n \tag{10}$$

$$\sum_{j=1}^{n}x_{ij}\leqslant 1,\forall\, i\in m,j\in n \tag{11}$$

$$y_j\geqslant x_{ij},\forall\, i\in m,j\in n \tag{12}$$

$$y_j\geqslant y_{j+1},\forall\, j\in n \tag{13}$$

在目标函数（4）中，系数 F_0 和 F_1 用来对目标 A 和目标 B 的值进行规范化，使其目标函数的最大值与初始问题一致[14]，如下式所示。

$$F_0=\max(B+C)-\min(B+C)+1 \tag{17}$$

$$F_1=\max(C)-\min(C)+1 \tag{18}$$

上述 ILP 模型中的符号定义如表 1 所示。

2.2 约束定义

VM 容量约束：用来确保任务所需的总资源不超过 VM_j 的可用资源，如公式（5）所示。其中，二元变量 x_{ij} 表示是否将 $Query_i$ 调度给 VM_j；CP_j 表示在 m 个任务的最大截止期限之前，VM_j 的可用容量。

任务截止期限约束：如公式（6）～（9）所示，其中，二元变量 b_{ik} 表示是否在 $Query_k$ 之前执行 $Query_i$，如果是，则 $b_{ik}=1$；否则 $b_{ik}=0$。约束（6）通过设置 b_{ik} 和 b_{ki} 中只存在一个为 1，来确保 $Query_i$ 和 $Query_k$ 的特定执行顺序。当在相同 VM_j 上执行 $Query_i$ 和 $Query_k$ 时，约束（7）用来限制这些任务不同时执行，即要么在 $Query_k$ 之前要么在 $Query_k$ 之后执行 $Query_i$。建模表示为，若 $Query_i$ 在 $Query_k$ 之前执行，则 $b_{ik}=1,b_{ki}=0$；若 $Query_i$ 在 $Query_k$ 之后执行；则 $b_{ik}=0,b_{ki}=1$。约束（8）用来限制如果 $b_{ik}=1$，则 $Query_i$ 应该在 $Query_k$ 开始执行之前完成，其中，e_{ij} 表示 VM_j 上 $Query_i$ 的执行时间，

表 1　ILP 模型中的符号定义

符号	定义
i	特定任务 $Query_i$
k	特定任务 $Query_k$
j	特定 VM，VM_j
m	任务集
n	VM 集
L_i	任务所需要的资源
x_{ij}/x_{jk}	它表示是否将 $Query_i/Query_k$ 分配给 VM_j
C_j	VM_j 的成本
y_j	它表示是否终止 VM_j
s_i/s_k	$Query_i/Query_k$ 的开始执行时间
CP_j	VM_j 的可用容量
b_{ik}/b_{ki}	它表示 $Query_i$ 是否在 $Query_k$ 之前执行
e_{ij}	VM_j 上 $Query_i$ 的执行时间
D_i	$Query_i$ 的截止期限
B_i	$Query_i$ 的预算
C_{ij}	VM_j 上 $Query_i$ 的执行成本
z_j	它表示是否创建 VM_j

F_2 为满足 $F_2 \geqslant \max(s_i + e_{ij} - D_i) + 1$ 的足够大常数。约束 (9) 用以确保如果 $Query_i$ 于 s_i 时刻在 VM_j 上开始执行，则它应该在其截止期限 D_i 之前完成，F_3 为满足 $F_3 \geqslant \max(s_i + e_{ij} - D_i) + 1$ 的足够大常数。

任务预算约束：用以保证 VM_j 上 $Query_i$ 的执行成本不超过它的预算 B_i，如公式 (10) 所示。其中，C_{ij} 为 VM_j 上执行 $Query_i$ 的成本。

任务调度次数约束：如公式 (11) 所示，它表示 $Query_i$ 只能调度到一个资源上。

VM 终止约束：如公式 (12) 到 (13)，二元变量 y_j 表示是否终止 VM_j。约束 (12) 用以保证当 $y_j = 0$ 时，x_{ij} 必须为 0 (即不能将 $Query_i$ 调度到 VM_j)；当 $y_j = 1$ 时，x_{ij} 可以为 0 或 1 (即可以将 $Query_i$ 调度到 VM_j，也可以不调度到 VM_j)。约束 (13) 用来提供 VM 使用的优先顺序。将创建的 VM 根据其成本添加到列表中，因此该约束可以使调度器优先使用成本较低的 VM。进而降低高成本 VM 上的负载，当高成本 VM 变的空闲时，则终止它。

3　仿真及分析

3.1　资源设置

在 CloudSim 仿真器[15]中构建实验平台，数据中心包含 20 个物理节点，每个节点配备 50 个 CPU 内核、100 GB 内存、10 TB 存储和 10 GB/s 网络带宽。基于亚马逊 EC2 云[16]中的 VM 模型，仿真了 4 类 VM，分别为 r3.large、r3.xlarge、r3.2xlarge 和 r3.4xlarge，各类 VM 配置如表 2 所示。

表 2　VM 配置表

VM 类型	CPU/个	内存/GB	存储空间/GB	成本/($/h)
r3.large	2	1	32	0.175
r3.xlarge	4	3	80	0.350
r3.2xlarge	8	6	160	0.700
r3.4xlarge	16	12	320	1.400

3.2　任务设置

仿真中，对不同 BDAA 任务的资源需求进行建模。每个任务请求包含以下信息：(1) 任务提交时间，使用均值为 1 分钟的泊松分布来构建任务到达时间间隔。(2) BDAA 任务类，在本实验中考虑了基于 4 类 SQL 数据库查询系统的 BDAA，分别为是基于 Impala (BDAA1)、基于 Shark (BDAA2)、基于 Hive (BDAA3) 和基于 Tez (BDAA4)。(3) 任务量，其基于任务的资源需求而建模。由于各任务的性能不同，所以通过引入变差系数来建模具有 10% 变化量的任务性能，其服从 [0.9，1.1] 范围内的均匀分布。(4) 用户号，实验中设置了 50 名提交 BDAA 的用户。(5) 任务截止期限，考虑了两类截止期限，严格截止期限和宽松截止期限。使用正态分布 (3，1.4) 来生成严格截止期限，其中，3 表示任务的平均截止期限为其处理时间的 3 倍，1.4 表示标准差。类似地，使用正态分布 (8，3) 来生成宽松截止期限。(6) 任务预算，包含紧缩预算和宽松预算。同样用正态分布 (3，1.4) 生成紧缩预算，使用正态分布 (8，3) 生成宽松预算。

3.3　结果分析

实验生成了大约 6 个小时的任务工作，共包含 400 个任务。接纳控制器对任务进行筛选，接收在截止期限和预算上可以满足 QoS 需求的任务。然后，由 BDAA 管理器利用 ILP 模型求解调度方案。为了评估接纳控制器和调度算法的有效性，进行了以下研究。

3.3.1　接纳控制器性能分析

为了评估接纳控制算法的有效性，在不同任务启动时间 ST 下进行接纳实验，其启动时间范围为 0 到 60 (单位：分钟)，即将任务推迟 0 到 60 分钟再执行接纳和调度。结果如表 3 所示，其中 STN 为提交的任务数量，ATN 为接受的任务数量，SEN 为成功执行的任务数量。

表 3　任务接纳和执行数量

任务数量	任务启动时间 ST/分钟						
	0	10	20	30	40	50	60
STN	400	400	400	400	400	400	400
ATN	336	317	299	287	274	261	252
SEN	336	317	299	287	274	261	252

表 3 可以看出，对于实时任务 (起始时间为 0)，接受率为 84.0%，而对于滞后 10 到 60 分钟的任务，由于截止期限的约束，滞后会导致一些任务无法完成，所以接受率降低。滞后时间越长，接受率越低。另外，通过接纳控制器后的任务都得到了成功执行。这证明了接纳控制器的有效性，能够确保所接受的任务都具备 SLA 保证，这有助于提高 BDAaaS 提供商的声誉。

3.3.2　执行成本分析

将提出的 ILP 调度模型与文献 [5] 提出的 SLAA 调度方法在成本节约方面进行比较，其中，为了公平起见，都采用了任务接纳控制。所获得的结果如表 4 所示。可以看出，在不同任务启动时间 ST 下，各种调度方案的资源成本不同，但提出方案的整体成本明显小于 SLAA 方案，降低了约 13%。

表 4　调度方案的资源成本　　($)

方案	ST=0	ST=10	ST=20	ST=30	ST=40	ST=50	ST=60	平均
ILP	125.4	123.2	129.6	119.4	136.2	124.1	120.7	125.5
SLAA	140.2	135.3	139.7	145.7	150.1	143.3	159.2	144.8

另外，在接纳控制实验中，我们发现，当启动时间 ST 增加时，所接受的任务量是变小的。然而，这些任务的执行总成

本并没有随着任务量的减小而明显减小。这是因为，当 ST 增加时，满足截止时间约束边缘的任务较多。由于这些任务所允许的执行时间较紧迫，所以调度器被迫安排一些高价格的能力较强的资源来执行它，所以总体成本不会明显降低。

表 5 给出了 2 种调度方案所分配的资源类型和数量。正如上述分析，ST 越大，调度器所调用的高级资源越多。总体来说，提出 ILP 模型所调用的普通和高级资源数量都要小于 SLAA 方案，所以具有较小的执行成本。

表 5 资源配置

调度场景		SLAA	ILP
启动时间	ST=0	28 * r3. large	24 * r3. large
	ST=10	27 * r3. large	23 * r3. large
	ST=20	28 * r3. large+ 1 * r3. xlarge	21 * r3. large+ 1 * r3. xlarge
	ST=30	22 * r3. large+ 3 * r3. xlarge	17 * r3. large+ 2 * r3. xlarge
	ST=40	18 * r3. large+2 * r3. xlarge+2 * r3. 2xlarge	16 * r3. large+ 3 * r3. xlarge
	ST=50	14 * r3. large+5 * r3. xlarge+1 * r3. 4xlarge	11 * r3. large+3 * r3. xlarge+1 * r3. 2xlarge
	ST=60	21 * r3. large+4 * r3. xlarge+ 2 * r3. 2xlarge+1 * r3. 4xlarge	16 * r3. large+2 * r3. 2 xlarge+1 * r3. 4xlarge

4 结束语

为了提高云平台对大数据分析应用的执行效率，提出了一种 BDAaaS 架构，通过接纳控制器筛选出可执行的 BDAA 任务，并建立相应的 SLA。然后，通过 ILP 资源调度模型在满足 SLA 保证下为 BDAA 分配资源，以此最小化任务执行成本。在不同提交时间的任务申请下进行调度仿真，结果证明了提出方法能够有效降低执行成本，具有有效性和可行性。

参考文献：

[1] 李晓飞. 基于云计算技术的大数据处理系统的研究 [J]. 长春工程学院学报（自然科学版），2014，15（1）：116－118.

[2] 徐 聪. 大数据应用在云计算平台的优化部署与调度策略研究 [D]. 北京：清华大学，2015.

[3] Arun J，Hazaruthin M M，Karthik M. Analytics as a service delivery model for the cloud [A]. IEEE International Conference on Engineering and Technology [C]. IEEE，2015：1－5.

[4] Wu L，Garg S K，Buyya R. Service Level Agreement（SLA）Based SaaS Cloud Management System [A]. IEEE，International Conference on Parallel and Distributed Systems [C]. IEEE，2015：440－447.

[5] Alrokayan M，Vahid Dastjerdi A，Buyya R. SLA－Aware Provisioning and Scheduling of Cloud Resources for Big Data Analytics [A]. IEEE International Conference on Cloud Computing in Emerging Markets [C]. IEEE，2014：1－8.

[6] 王德文，刘晓萌. 基于改进粒子群算法的云计算平台资源调度 [J]. 计算机应用研究，2015，32（11）：3230－3234.

[7] 刘 曦，张潇璐，张学杰. 异构云系统中基于智能优化算法的多维资源公平分配 [J]. 计算机应用，2016，36（8）：2128－2133.

[8] 周芸韬. 基于 MQAAR 的移动自组织网络路由方案 [J]. 湘潭大学自然科学学报，2016，38（3）：69－73.

[9] 林清滢，陆锡聪，徐 林. 云计算中面向 SLA 的作业分层优先级调度策略 [J]. 计算机科学，2014，41（1）：316－317.

[10] Garg S K，Toosi A N，Gopalaiyengar S K，et al. SLA－based virtual machine management for heterogeneous workloads in a cloud datacenter [J]. Journal of Network & Computer Applications，2014，45（4）：108－120.

[11] Manzini R，Accorsi R，Cennerazzo T，et al. The scheduling of maintenance. A resource－constraints mixed integer linear programming model [J]. Computers & Industrial Engineering，2015，8（7）：561－568.

[12] 谢丽霞，严焱心. 云计算环境下的服务调度和资源调度研究 [J]. 计算机应用研究，2015，35（2）：528－531.

[13] Zhu L，Li Q，He L. Study on Cloud Computing Resource Scheduling Strategy Based on the Ant Colony Optimization Algorithm [J]. International Journal of Computer Science Issues，2012，9（5）：131－138.

[14] 张希翔，李陶深. 云计算下适应用户任务动态变更的调度算法 [J]. 华中科技大学学报自然科学版，2012，40（1）：165－169.

[15] Genez T A L，Bittencourt L F，Madeira E R M. Workflow scheduling for SaaS / PaaS cloud providers considering two SLA levels [J]. Network Operations & Management Symposium IEEE，2012，104（5）：906－912.

[16] Poola D，Ramamohanarao K，Buyya R. Enhancing Reliability of Workflow Execution Using Task Replication and Spot Instances [J]. Acm Transactions on Autonomous & Adaptive Systems，2016，10（4）：1－21.

（上接第 196 页）

相比上述实验结果，在计算最短路径时，结点相同的情况下，MapReduce 和 GPU 双重并行条件下最短路径的计算比 MapReduce 计算或普通的并行计算速度更快，所用的时间明显减少，GPU 加速的 MapReduce 模型，充分发挥优势，在结点增加时，平均加速比也有所增加，即双重并行条件下的最短路径的计算，提高了大规模数据并行处理的优势。

4 结论

本文利用 GPU 来加速 MapReduce 构成双并行模型，说明将 GPU 和 MapReduce 二者相结合的原因，将 GPU 应用到 MapReduce 过程中，实现多层次并行，研究了双重并行环境下最短路径的实现，加入了预处理和数据动态处理器。最后通过实验进行验证，结果表明，双重并行环境下最短路径的计算具有双重并行的效果。

参考文献：

[1] 张凌洁，赵英. 基于 GPU 的并行 APSP 问题的研究 [J]. 电子设计工程，2012，20（17）：15－18.

[2] 钮 亮，张宝友. MapReduce 求解物流配送单源最短路径研究 [J]. 电子技术应用，2014，40（3）：123－125.

[3] 杨 玲，李仁发，唐 卓. 基于 MapReduce 的单源最短路径算法研究 [J]. 微计算机信息，2011（12）：97－99.

[4] 王晓东. 算法设计与分析 [M]. 北京：清华大学出版社，2003.

[5] 郭钇汝. 基于 GPU 集群系统的 MapReduce 编程模型研究 [D]. 济南：山东大学，2014.

[6] 曾青华，袁家斌. 基于 MapReduce 和 GPU 双重并行计算的云计算模型 [J]. 计算机与数字工程，2013，41（3）：333－336.

[7] 瞿李峰. 基于 GPGPU 的 MapReduce 高性能并行计算模型研究与应用 [D]. 桂林：桂林理工大学，2009.

[8] 张 凯，秦 勃，刘其成. 基于 GPU－Hadoop 的并行计算框架研究与实现 [J]. 计算机应用研究，2014，31（8）：2548－2550.