基于Spark全流程分布式遗传算法的贝叶斯网络结构学习

颜克斐，张炜健，方伟*

江南大学人工智能与计算机学院，无锡 214000

摘要： 贝叶斯网络(Bayesian network, BN)是一种概率图模型，学习贝叶斯网络结构已被证明是一个NP难问题。遗传算法(Genetic Algorithm, GA)已被广泛运用在BN结构学习方法中，可以产生比基于单解算法更准确的BN结构，但仍然需要较长的计算时间，特别在海量数据的情况下。针对该问题，本文提出了一种基于Spark的全流程分布式GA(Distributed Genetic Algorithm Based on Spark for BN Structure Learning, DGA-BN)来加速学习BN结构。DGA-BN设计了构建超结构、GA演化操作和评分计算三个流程上的并行化方法，通过引入Redis对中间数据进行存储使评分计算中可以复用超结构以及历史评分数据，减少冗余计算时间，加快计算效率。通过三种网络在海量数据情况下实验结果表明，本文所提DGA-BN算法有效提高了贝叶斯网络结构学习的算法效率和质量，并具有更好的扩展性。

关键词： 贝叶斯网络;结构学习;遗传算法;分布式并行;Spark

中图分类号： TP181

Distributed Genetic Algorithm Based on Spark for BN Structure Learning

YAN Kefei, ZHANG Weijian, FANG Wei*

Department of Computer, University of Jiangnan, Wuxi 214122

Abstract: Bayesian network (BN) is a probability graph model and learning BN structure has been proved to be a NP hard problem. Genetic algorithm (GA) has been widely used in BN structure learning methods, which can produce more accurate BN structure than single solution, but it still needs a long computation time, especially in the case of massive data. To solve this problem, this paper proposes a distributed genetic algorithm based on spark for BN structure learning (DGA-BN) to accelerate the learning of BN structure. In DGA-BN, three processes of superstructure construction, GA evolution operation, and scoring calculation have been designed to work in parallel. Redis is introduced to store the intermediate data, so that the superstructure and historical scoring data can be reused in scoring calculation, which help to reduce the redundant computing and accelerate the computing efficiency. The

基金项目： 国家重点研发计划(2017YFC1601800), 国家自然科学基金(62073155, 61673194,62106088),广东省重点实验室（2020B121201001）

作者简介： 颜克斐（1996-），男，研究生，主要研究方向：贝叶斯网络结构学习，智能优化算法。张炜健（1996-），男，研究生，主要研究方向：贝叶斯网络结构学习，智能优化算法。通信作者：方伟（1980-），男，教授，主要研究方向：计算智能，Email: fangwei@jiangnan.edu.cn。

experimental results show that the proposed DGA-BN algorithm effectively improves the efficiency and quality of BN structure learning and has better scalability.

Key words: Bayesian networks;Structure learning; Genetic algorithm; Distributed parallel;Spark

0 引言

贝叶斯网络(Bayesian network,BN)作为一种不确定性的推理方法，被认为是表示因果知识的最佳方法之一，在人工智能概率和不确定性领域中有较多的应用 [1]。BN结合了图与概率定理，从某种角度来看，它是一个有向无环图(DAG)，其中节点代表随机变量，弧代表变量之间的依赖关系。这些关系被一组条件概率分布进一步量化，每个条件概率分布对应于它的父变量。由于BN在推断、学习等方面的优势，它正在各个领域如生物 [2]、医药 [3]、图像 [4]等方向变得越来越流行。由专家提供BN结构耗时且不能保证正确率，因此，通过数据自动学习BN结构在过去20年被广泛关注。然而由于学习BN结构是一个NP难问题，对于有较多变量的数据来说，学习一个完全正确的模型较为困难 [5]。

学习BN拓扑结构的方法一般分为三种：基于约束的方法、基于评分的方法和混合的方法。

基于约束(CB)的方法首先通过统计学方法如χ^2检验 [6]得到变量间的条件独立性关系，通过这些条件独立性关系来构建一个与之拟合的BN。常用的方法如使用这些依赖项来限制每个节点的可能父集，并基于这些限制构建BN，这类方法是在BN结构学习问题中最早被提出的，最著名的如PC算法 [7]，GS算法 [8]等。CB方法的缺点主要在于其依赖于指数级的条件独立性测试，而且其中的一些测试结果可能不准确。

基于搜索评分(S&S)的方法通过评分函数评价候选网络结构的质量，将结构学习问题转变为组合优化问题，进而可以通过各类元启发式算法进行搜索以得到评分最高的结构。常用的如基于贪心的算法以及一些常用的元启发式算法如GA，PSO，ACO等。常见的基于GA的方法如Larranaga的K2GA算法 [9]、 Kabli提出的Chain-model GA [10]等。

混合方法将上述的两种方法进行了结合，其中一种流行的策略是通过CB的方法构建出结构的骨架，再通过S&S的方法搜索得到一个高评分的BN结构 [11]，最著名的方法就是MMHC算法 [11]。在搜索过程中也可以使用基于EA的算法，如Vafaee提出的HSL-GA算法 [12]。

一般情况下，S&S的方法结果准确，但需要太多的时间。而相比较常见的单解搜索算法如模拟退火和爬山算法 [13]等容易导致陷入局部最优，而基于GA的算法通常可以产生比基于单解算法更准确的BN结构，但通常需要更长的计算时间。为了解决在海量数据上的计算耗时问题，本文提出了一种基于Spark的分布式并行GA算法(Distributed Genetic Algorithm Based on Spark for BN Structure Learning, DGA-BN)以便加速学习BN结构。GA具有自然的并行性，同时是一种基于迭代的计算密集型算法，Spark是一种快速通用的集群计算平台，因此可以天然的适应Spark的架构。本文是较早提出使用Spark来并行化GA并解决BN结构学习问题的工作之一。在所提DGA-BN算法中，利用Spark实现了全流程的并行化，包括超结构(Super Structure, SS)构建的并行化，适应度评估的并行化和GA操作的并行化，同时引入Redis对中间数据进行

存储使评分计算中可以复用构建SS及以往迭代评分时的数据，减少了冗余计算时间加快计算效率。

1 研究背景

基于评分搜索的方法基本思想是从贝叶斯网络可能的搜索空间中找到最大化评分函数的BN结构。一般评分搜索方法分为两步：定义一个评分函数和采用一个搜索策略。评分函数用于度量样本数据集与BN结构之间的拟合程度，评分越高，代表数据集和BN结构的拟合效果越好。本文使用的BIC评分由两部分构成，网络结构的对数似然度和一个惩罚项。前者用来度量数据集与BN结构的拟合程度，后者用于避免由于结构模型过于复杂，参数过多，从而导致数据和结构过拟合。BIC公式如下所示：

$$BIC(G|D) = \sum_{i=1}^{n}\sum_{j=1}^{q_1}\sum_{k=1}^{r_i} m_{ijk}\log\frac{m_{ijk}}{m_{ij}} - \sum_{i=1}^{n}\frac{q_i(r_i-1)}{2}\log m \tag{1}$$

基于演化计算的搜索算法已被许多研究者用于BN结构学习算法，寻找正确的BN结构是学习过程的重要组成部分，同时也直接影响着BN参数的学习。基于EA的算法主要分为两类，一类是基于评分搜索的方法，另一类是基于混合的方法。对于评分搜索的方法来说，Larrañaga首先在1996年提出了通过GA算法学习节点序的方法。随后在此基础上出现了K2GA [14]，Chain-model GA [10]等算法。而后Myers通过同时在分离的种群中演化缺失数据和结构，将基于GA的BN结构学习算法扩展到了缺失数据集的BN结构学习问题中。Dijk设计了一种GA方法，其中的重组算子试图防止破坏到目前为止在种群中获得的良好的BN子结构。该算法使用MDL评分 [15]计算方法作为BN结构的适应度函数，并使用修复算子确保结构无环。 Hanzelka还提出了一种结合了拉马克进化与单解的局部搜索方法的GA。它使用一个卡方测试 [6]来确定哪些边缘应当被拆或修复。相比评分搜索方法，基于混合的方法由于减少了搜索空间导致在效率上一般优于基于评分搜索的方法。近来提出的基于EA的BN结构学习混合算法有如BNC-PSO [1],BEWCA-BN [16],HSL-GA [12]，AESL-GA [17]等。

1.1 进化算法的并行化

并行EA搜索算法是提高EA搜索算法性能的新技术。一般来说，实现的并行分布式架构可以分为以下三种模型:

- 主从模型: 由一个主节点来管理所有子种群，并将个体分布在从节点中。然后在相应的从节点上计算个体的适应度值。利用多个Mapper来评价每个染色体的适应度值。随后，通过单个的Reducer收集结果并进行其他的GA操作，包括选择，交叉和变异操作。一代意味着在MapReduce上执行一轮，整个计算过程就是一个执行序列。
- 岛屿模型: 一个种群被划分为几个分布在多个岛屿上的亚种群，遗传算法在每个岛屿上独立运行。由于每个岛屿只包含种群的部分个体，这些岛屿通过迁移一些个体定期交换信

息，向聚集的子种群中加入多样性。该模型能够在并行和分布式计算中执行遗传算法的所有操作，使得不同的岛屿可以周期性地探索搜索空间的不同部分。 Geronimo等 [18]设计了一个Partitioner来将每个岛分配一个Reducer。

- 网格模型: 将一个种群组织成社区，并将一部分个体放在一个节点(即网格)中，一般需要一个大规模的集群来处理这个模型。 Camacho [19]将聚类技术与遗传算法相结合，用于大规模并行计算环境下的大规模计算。Gong等 [20]在MapReduce上使用伪随机函数，对分布式模型的聚类技术稍加修改，提出了一种元胞模型。

1.2 BN结构学习的并行化

由于BN的计算复杂度较高导致算法执行时间过长，近年来研究人员提出了许多基于高性能计算平台和共享内存架构的BN结构学习并行算法。对于无共享计算集群，出现了一些使用基于Mapreduce范式进行可扩展的BN学习并行方法。对于基于约束的方法，Madsen [21]将基于约束的算法(如TPDA和PC)迁移到了MR平台，最近Arias [22]也提出了基于Spark计算属性多维列联表的贝叶斯分类器的并行版本。对于基于评分搜索和基于混合的方法，Fang首先提出了一种基于Mapreduce的从海量数据中学习BN的K2算法 [23]，扩展了传统的评分搜索算法，算法在评分计算过程中通过map和reduce计算需要的参数，然后在搜索过程中同样基于Mapreduce范式设计了并行的局部分数计算方法，最后将每个节点的局部最优结构被合并到全局最优结构中，从而得到最终的BN结构。基于K2的方法计算过程较为快速，但在BN结构准确率不高。而后Yue [24]根据MDL的评分计算方法，在Mapreduce上实现了基于评分搜索的分布式HC算法，通过Mapreduce的两布算法对MDL评分进行计算，并从样本数据中对候选BN结构进行评分，然后给出了相应的扩展经典爬山算法以获得最优结构的策略。由于混合算法必须对所有数据进行Map和Reduce操作才能得到分数，而不是像传统的集中式混合算法那样只对剪枝结构中的数据进行分数，Li [25]提出了基于Mapreduce的混合方法。当将混合算法应用于BN结构估计时，算法主要工作过程可分为基于约束的阶段和评分搜索阶段 [26]。因此我们提出的算法使用Spark实现全流程的并行化。

2 基于Spark和GA的全流程分布式贝叶斯网络结构学习方法

本节将详细介绍提出的BN结构学习算法DGA-BN，算法分为三个主要阶段，即基于Spark的构造SS并行化，GA演化操作并行化以及 BIC评分并行化。

2.1 DGA-BN算法流程

DGA-BN算法的设计思路如图1所示。首先将数据读取之后并行计算互信息并构建SS，然后基于广播后的SS并行的初始化种群，并行计算个体的BIC评分，通过并行演化算子和评分计算操作取得更优的种群和个体。重复这些步骤，直到满足终止条件，然后选择BIC评分最高的最终解作为识别出的BN结构。

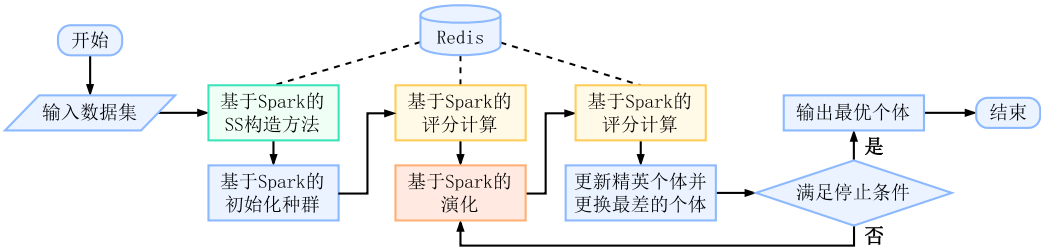


图 1: 基于Spark和GA的贝叶斯网络结构学习算法(DGA-BN)流程

- 基于Spark的SS构建方法：它使用互信息构造一个无向图结构，作为SS并依此减少搜索空间。依据互信息计算公式2实现并行化互信息计算并构建相关数据集的互信息矩阵M，并在过程中将计算得到的中间数据存入Redis中以便后续流程使用。若节点i和节点j之间的互信息$MI_{ij} \geqslant MMI_i * \alpha$，将节点$i$和节点$j$中的边加入SS。其中$MMI_i$代表互信息矩阵$M$中节点$i$和其余所有节点的互信息最大值，$\alpha$是一个0到1的预设值。

$$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log(\frac{p(x,y)}{p(x)p(y)}) \tag{2}$$

- 基于Spark的评分计算：首先通过公式1计算各个BN个体所需要的相关数据，并去除Redis中已存在的部分数据。通过Spark对数量进行计算后得到计算BN所需要的全部数据存入内存，最后对个体并行计算相应的BIC评分。

- 基于Spark的演化： GA的整个流程中包括了对初始化，选择，交叉，变异的并行化操作。我们使用锦标赛选择算子作为选择算子，交叉算子为均匀交叉，变异算子位单点变异。提出的演化方法将选择和交叉算子合并，减少了Shuffle的时间，使各算子适用于分布式计算中的并行化，更快的执行演化，高效的得到最终的BN个体。

2.2 基于Spark的超结构并行化构建

为了避免搜索空间过大而导致GA搜索时间较长，通过互信息矩阵构建SS减少不必要的搜索空间。由于数据量较大可能导致串行计算时间过长，因此基于Spark对其进行了并行化改造。由于流程中花费时间最多的步骤是对评分进行计算，为了减少冗余计算，引入Redis，将中间计算数据存入Redis中供后续步骤快速使用。

公式2中，$p(x,y)$代表$\frac{m_{xy}}{m}$，其中m_{xy}代表数据集D中满足$X_i = x$和$Y_i = y$的样本数目，m代表整个数据集的样本数量。$p(x)$代表$\frac{m_x}{m}$，$p(y)$代表$\frac{m_y}{m}$。因此离散情况下互信息的计算公式可以转换为公式3。

$$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} \frac{m_{xy}}{m} \log(\frac{m_{xy} * m}{m_x * m_x}) \tag{3}$$

基于Spark的算法步骤设计如下：

Algorithm 1 计算出现次数

Input: 数据集 RDD, D; 需要的情况变量集, S_s;

Output: 各情况出现的次数, M_s

1: 广播变量 S_s.

2: **MapPartitions(dataset** D**)** 对每个数据D中的分区，计算S_s中每个情况的出现次数。$< key, value >$中key为情况， $value$为在该分区中出现的次数

3: **reduceByKey** 对相同key的次数进行求和，得到各个key总出现的次数，并在该分区计算结束后将数据通过pipeline存入Redis中

4: 收集得k, v键值对M_s输出

1) 从HDFS中获取数据集为D

2) 到计算互信息所需的情况集合S_{mi}。假设对一个有三个节点ABC，每个节点两种取值的数据集分别为$a_1, a_2, b_1, b_2, c_1, c_2$，则该数据集所需要的集合为$A = a_1; B = b_1; C = c_1; A = a_1, B = b_1; B = b_1, C = c_1; A = a_1, C = c_1$共6种，情况集合代表了所有需要计算出现次数的情况。

3) 广播，保证各Executor快速得到集合，减少OOM出现的风险。

4) 通过算法1,使用D计算各情况出现的次数得到键值对，其中key是计算互信息所需的情况集合S_{mi}中的情况，value是该key在数据集D中出现的次数。MapPartitions阶段将每个Partition中每一行数据与S_s中的情况进行比较，计算每个情况在当前partition中出现的次数。相比通过map产生大量$< key, 1 >$的方式，对每个Partition中的数据以key-value形式的形式输出给Reduce可以减少Shuffle阶段的时间。Reduce阶段对相同key的数据进行求和计算，得到每个情况在整个数据集D中出现的次数。并将key-value键值对M_{mi}通过Pipeline存入Redis中，这些数据可以在后续BIC评分计算过程中进行复用避免重复计算。

5) 通过公式3 对每个节点两两间的互信息进行计算，计算中所需要的数据均能够从M_{mi}中得到，计算后构造数据集对应的互信息矩阵。

6) 对每个节点i，可以通过互信息矩阵得到节点i和其余节点的互信息最大值MMI_i，对每个其余节点j和节点i间的互信息MMI_{ij}和$\alpha * MMI_i$进行比较，若$MMI_{ij} > \alpha * MMI_i$，则将无向边$ij$加入$SS$构建无向边合集。

2.3 基于Spark的GA实现

构建完SS后需要对搜索空间进行搜索，我们使用改进的分布式并行GA算法进行搜索。由于在各岛屿中独立的计算评分会导致中间数据的重复计算，极大的浪费执行时间，同时若陷入局部最优会导致搜索时间进一步增长。本文采用的是主从模型。GA的要素主要有编码，初始种群设定，适应度计算，遗传算子等。由于串行计算效率过低，因此对初始种群设定，适应度

计算，遗传算子都进行了并行化工作加快了算法效率。改进的并行选择，交叉，变异算子伪代码如算法2所示。适应度计算，即评分计算将在下一小节中进行介绍。

Algorithm 2 并行的选择交叉变异算子算法

Input: 当前种群, P_a;当前种群数量, N_c;选择后的种群数量, N;

Output: 经过演化后的下一代种群, P_next

广播种群数组 P_a

构造大小为N的数组，其中每个元素是长度为T，数据为随机$[0, N_c)$的数组，作为A_{TSI}

构造大小为N的数组A_c，每个元素随机选取两个A_{TSI}中的元素放入

将数组A_c转换为种群RDDP_i

通过flatMap与算法3得到下一代种群，$P_{next} = P_i.flatMap(crossoverandmutation)$

如图可知，初始化流程中将数组转换为RDD后，每个个体基于SS并行且随机的生成结构，并在生成结构后通过去环算法删除产生的环，使结构合法化。由于随机生成结构和去环算法只需要SS的信息，因此先将SS广播以便后续可以在各个Executor上快速调用。而由于其余操作均不需要全部种群的信息，因此广播SS后其余所有操作均适用于分布式计算的并行化，减少不必要的落盘和Shuffle操作，加快效率。因此基于Spark的初始化算法步骤设计如下：

1) 将得到的SS进行广播以便使用。

2) 对每一个元素，通过Map方法并行的使用SS中的边集进行随机初始化。对于SS中的每个无向边，假定连接的两个顶点为A和B，随机选择决定三种可能的连接状态中的一种，如：$A \leftarrow B$，$A \rightarrow B$，$A \nleftrightarrow B$。由此得到了一个初始个体，组成了结构种群RDD$P_{invalid}$

3) 由于P_1中初始化产生的个体可能有环的存在，通过使用改进版本的GR去环算法进行去环操作，在出现环的情况下，优先考虑出度与入度的差较高的节点，由这些排序节点最小化反馈弧集，然后删除该集合，以此生成了一个DAG。并对此DAG限定了节点的最大父节点数目，当节点的父节点数大于设定的值时，随机删除父节点到此节点的边。以此最终得到的DAG作为输出的种群RDD。

4) 基于数据集RDDD，计算各情况出现的次数得到$< key, value >$对，其中key是计算互信息所需的情况集合S_{mi}中的情况，$value$是该key在数据集D中出现的次数。MapPartitions阶段将每个Partition中每一行数据与S_s中的情况进行比较，计算每个情况在当前partition中出现的次数。相比通过map产生大量$< key, 1 >$的方式，对每个Partition中的数据以键值对形式输出给Reduce可以减少Shuffle阶段的时间。Reduce阶段对相同key的数据进行求和计算，得到每个情况在整个数据集D中出现的次数。并将键值对键值对M_{mi}通过Pipeline存入Redis中，这些数据可以在后续BIC评分计算过程中进行复用避免重复计算。

5) 生成合法的种群RDDP

由于选择与交叉算子均需要其余个体信息才能运行，我们首先广播长度为N_c的种群数组以便后续可以快速查询个体。本文采用锦标赛选择，将T个范围在0到N_c的随机数放入数组中构

Algorithm 3 交叉变异算子算法

Input: 三个记载锦标赛信息的数组, TSI_1, TSI_2, TSI_3 ; 最大父节点个数r, MP;超结构, SS;

Output: 包含两个键值对的数组, A

初始化3个$< k, v >$对G_0,G_1,G_2，k为BN，v为评分，评分初始化负无穷

对TSI_1中的每个元素对应的BN结构评分做对比，选出评分最高的BN结构作为G_0

对TSI_2中的每个元素对应的BN结构评分做对比，选出评分最高的BN结构作为G_1

对TSI_3中的每个元素对应的BN结构评分做对比，选出评分最高的BN结构作为G_2

$G_{new} = UniformCorssover(G_1, G_2)$

对 G_{new} 去环并通过 MP 限制父节点.

$G_{new} = SinglePointMutation(G_{new}, SS)$

对 G_{new} 去环并通过 MP 限制父节点.

造为竞赛选择信息TSI。对每个个体，另外随机选取两个TSI组成一个个体，通过Spark生成为RDD。对每个个体而言，第一个TSI通过广播的种群数组取得对应的个体，选出评分最高的个体作为输出，后两个TSI分别得到两个$< BN, score >$，以均匀交叉方式操作，再通过去环算法等进行合法化。在选择与交叉结束后，由于算法采用的单点变异算子不需要使用全部种群的信息，因此天然适应并行化操作，可以在一个map过程中直接调用。

并行化后的选择算子、交叉算子、变异算子全部的工作都可以在一个Map方法中进行调用，最大化的减少了落盘和通信的时间开销，在减少GA算子运行时间，极大的提高了搜索效率。

2.4 基于Spark的BIC评分函数

对于每个个体，需要通过评分函数对其进行评价。通过Redis存储中间数据，以便将数据进行复用减少冗余计算时间。中间存储的数据和构造SS阶段的数据重复，可以使用构造数据SS阶段得到的数据，加大数据复用率。基于Spark并行BIC计算评分，增加并行度，总体加快了评分计算效率。基于Spark的评分计算方法设计如下：

1) 对种群中的每个贝叶斯网络计算需要的情况，将整个种群所有的需要情况的集合合并，将不存在Redis中的情况取出作为S_{needs}并广播。

2) 对数据RDD的每个分区，计算S_{needs}中各情况出现次数构造键值对$< key, value >$，其中key为情况，$value$为此分区出现次数。

3) 在reduce阶段将相同key的$value$求和并存入Redis中，构造出键值对后广播以便计算中快速使用。

4) 基于BIC计算公式1，对种群RDD中的每个个体计算评分。

5) 输出计算出评分后的由键值对组成的种群，key为BN结构，$value$为评分。

由公式1可知，一个BN的评分可以由一组条件出现次数计算和一些全局变量组成，如q_i,r_i,而m可以初始计算一次后多次使用，评分计算阶段主要的时间花费在计算条件出现次数中，因此我们将数据存入Redis中，以便数据再后续使用时可以快速取出复用减少冗余计算所花费的时间。而3中存储的数据也是m，故可以存入Redis而后在评分计算中使用以增加数据复用率。

3 实验与结果分析

3.1 实验数据集

本文采用3个不同的网络 [27]对算子效果和算法效果进行评估，包括Asia，Earthquake和Survey。各模型数据量分别为100000，300000，500000。为了验证提出算法的可行性和有效性，算法在每个数据集上分别独立运行了二十次，然后给出每个算法的平均性能度量。

3.2 实验环境及参数设置

基于GA的混合方法及DGA-BN的算法参数设置如下：种群个数为200，最大迭代次数250，锦标赛大小为2，SS构造过程中的阈值系数为0.05，最大父节点个数为4。使用的评分函数为BIC评分函数，并行化评分函数计算由Scala实现。算法对每个数据集独立执行20次，并记录最终结果的平均值。 K2算法的节点顺序随机生成，其余算法均使用标准参数设置。

为公平的分析算法性能，我们同时考虑了学习网络和原始网络的分数和结构差异，通过F1评分对算法得到的模型进行评估，F1评分通过准确率与召回率评估分类器的优劣 [28]。在本问题中，准确率代表的是得到的模型的正确有向边数量除以得到的模型中所有边的数目，召回率指的是得到的模型的正确有向边数量除以实际标准BN模型中所有边的数目，F1评分由准确率及召回率计算得到，如式(4)所示：

$$F_1 = 2 * \frac{precision * recall}{precision + recall} \tag{4}$$

算法由Scala和Java进行实现，由Matlab R2017b的Bayesian Network Toolbox-Structure Learning Package (BNT-SLP) 、Full BNT工具箱构造数据。实验使用五台计算机，其中一台作为Namenode，其余四台为Datanode，计算机配置均为1一个六核CPU，型号均为Intel(R) Core(TM) i5-8400 CPU@2.80GHz，8GB内存，操作系统均为64位Ubuntu16.04，java环境为jdk1.8，Hadoop版本为2.7.7，Spark版本为2.3.4，Redis版本为6.0.9。我们对提出的新算法进行了评价，我们在各个benchmark网络结构上进行测试提出的DGA-BN算法，首先我们对提出的各个并行方法进行实验，测试各方法的有效性，然后对算法的效果进行了详细的对比分析。

3.3 并行评分计算的有效性

首先我们对提出的并行BIC评分计算方法验证有效性。SGA算法作为Baseline，将变异算子后的串行评分计算方法替换为基于Spark的BIC评分计算方法。在Asia模型中，分别在10万，30万，50万样本数据量上独立进行了多次实验并取平均值,各数据集的实验中平均BIC评分相

同，并且平均迭代次数基本相同，这意味着它们都能够在相同迭代次数的情况下在大数据集中获得全局最优。在得到相同结果的条件下，算法运行时间的实验结果如图2(a)所示。分布式评分计算算法在五个节点的集群上运行，SGA算法串行运行。从算法执行时间的实验结果可知，当数据量较小时，分布式算法相比效果一般，随着数据量增大，算法相比效率更高。因此基于Spark的BIC评分计算方法在数据量较大的情况下效果较好，由于分布式算法将数据分割计算后还需通过通信的方式汇总，因此当数据量不大时分布式算法效果较差。

(a) 评分的有效性 (b) 构建SS的有效性

图 2: Asia算法执行时间对比

3.4 并行化构建 SS 的有效性

对于提出的基于Spark的并行化构建SS方法，我们将上节中的加入基于Spark的评分算法中加入串行的SS构建方法，以此作为作为Baseline，然后将串行的SS构建方法替换为基于Spark的并行化构建SS方法进行对比试验。算法在Asia的不同数据量的数据集上运行,各数据集的实验中平均BIC评分相同，并且平均迭代次数基本相同，这意味着它们都能够在相同迭代次数的情况下在大数据集中获得全局最优。在得到相同结果的条件下，算法运行时间的实验结果如图2(b)所示。

从算法执行时间的角度来看，随着数据量的增大，算法的执行时间明显降低。同时对于两个算法，最终得到的BN评分均相同，可知提出的基于Spark的并行化构建SS的方法的有效性与正确性。通过基于互信息的方法构建SS，中间计算结果可以在评分计算过程中复用，减少了冗余计算时间，加快算法效率。因此对Asia, Cancer, Survey, Earthquake数据集上的中间数据复用率进行了统计，分别为96.88%，100%，84.21%和100% 每个数据集中数据复用率均较高，Cancer和Earthquake中在并行构建SS过程中计算得到的所有数据均可以在并行计算BIC评分过程中被复用。较高的复用率代表了算法计算效率高，运行时间快。

3.5 并行 GA 的有效性

在Asia，Survey的数据集上对提出的基于Spark的GA算子进行实验验证并行GA算子的有效性。对每个数据集将算法分别在种群大小为100，200，300的情况下进行实验。同时由于GA算子运行时间较小，因此迭代次数设置为250轮以便效果更加明显。实验效果如表1所示。由表可知实验中，当种群大小小于等于100时，基于Spark的GA的运行时间和串行GA相当，当种群大小等于200时，基于Spark的GA的运行时间节省明显，当种群大小等于300时，基于Spark的GA的

表 1: Asia 数据集基于 Spark 的 GA 算子实验

数据量(万)	SGA			Spark-based GA		
	100	200	300	100	200	300
10	93.573	137.912	180.668	81.192	126.462	157.559
30	112.997	157.877	202.840	114.855	143.814	175.080
50	113.858	158.301	203.885	112.153	145.898	174.432
100	140.985	185.499	228.462	141.364	170.497	199.345

运行时间可显著。随着种群大小的增加，基于Spark的GA算法节省时间的更多。因此在种群规模较小的情况下，单机模式的GA算法相比基于Spark的GA算法节省更多的时间。因为算法在集群中运行时，起始集群节点和节点之间的消耗通信占整个运行时间的很大比例。随着种群大小的增加，一组并行计算将逐渐显示出相对于独立模式的优势。

3.6 对比实验分析

我们比较了其余算法和DGA-BN之间的性能。比较一些著名的传统方法包括PC, TAN, HC和基于GA的混合方法以及分布式混合算法。为了与传统方法进行全面比较，我们使用了BNT-SLP工具包并在集群的一个节点上运行所有的传统方法。

表 2: EKGA-BN 和 AESL-GA 的参数设置

方法	种群大小	最大迭代次数	锦标赛大小	条件独立测试阈值	精英集比例	最大父节点个数
DGA-BN	100	250	2	0.05	N/A	4
AESL-GA	100	100	N/A	0.01	0.9	12

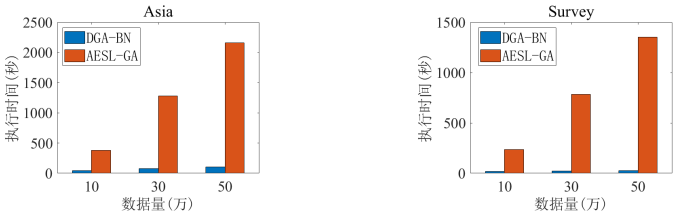
表3显示了单机算法在Asia, Survey, Cancer不同样本数据量下的实验结果，展示了各算法最终学习得到的BN结构的准确率。从结果可知传统的PC，TAN和HC，由于无法得到最优解，学习得到的BN结构准确率相比基于GA的结构学习算法较差。而DGA-BN相比AESL-GA相比在各数据集的不同样本上学习得到的BN结构准确率相差不多。本文所提DGA-BN相比单机版本的基于GA的BN结构学习算法在增加了数据扩展性的情况下并未降低得到的解的准确率，因此可以对更大规模的样本数据进行更快速地进行结构学习，若用于训练的样本数据量持续增加，也有利于提高学习得到的BN结构准确率。

由于基于GA的单机BN结构学习算法相比DGA-BN学习得到的BN结构准确率相差不多，我们将两种算法在单节点上运行，运行时间的实验结果如图3所示。

同时我们复现了分布式混合方法,并与我们提出的基于Spark的DGA-BN方法同样在4个Datanode的情况下进行比较，如表4显示了分布式混合算法在Asia, Survey, Cancer在样本数据量大小为一千万下的实验结果，表中可以看出在三个数据集中提出的算法最终学习到的BN结构都有更高的评分，在Asia数据集中两者学习得到的BN结构有相同的评分。基于混合的方法使用HC算法进行搜索，难以搜索到全局最优解，而基于GA的方法可以搜索到更优的BN结构。因此我们认为提出的算法在搜索能力上相比并行混合算法更好，能搜索到评分更高更符合数据的BN结

表 3: 对比实验结果

数据集	数据量(万)	PC	TAN	HC	AESL-GA	DGA-BN
Asia	10	0.444	0.571	0.706	0.853	0.938
	30	0.444	0.285	0.3	0.744	0.931
	50	0.444	0.190	0.105	0.762	0.906
Survey	10	0.571	0.133	0.615	1	1
	30	0.571	0.4	0.461	1	1
	50	0.571	0.4	0.461	1	1
Cancer	10	0.571	0.182	0.222	1	1
	30	0.571	0.363	0.222	1	1
	50	0.571	0.363	0.222	1	1



(a) ASIA (b) SURVEY

图 3: AESL-GA 和 DGA-BN 的执行时间比较

构。

表 4: DGA-BN和分布式混合算法在结构准确性上的对比

算法	Asia	Cancer	Survey
DGA-BN	-22376229.11	-20996675.74	-39502087.64
distributed hybrid algorithm	-22376229.11	-21007620.99	-39502101.39

由于DGA-BN算法在大部分模型中能够得到比混合算法评分更高的BN结构，因此为了公平比较，我们对DGA-BN达到混合方法的评分所需的时间进行了统计，如表5所示，在所有的数据集中我们提出的算法都比并行混合算法的时间更快达到此评分。基于Spark的算法更适应迭代式的计算同时减少了磁盘读写次数。对所有条件提前进行计算会导致计算的冗余，仅对搜索到的部分进行评分计算，并且将所有需要的结构进行汇总，统一进行计算可以极大的节省调度与计算时间。因此我们认为我们提出的算法在效率上高于并行混合算法。

4 总结与展望

本文提出了一种基于Spark的分布式BN结构学习算法（DGA-BN），用于在大数据情况下学习BN结构。DGA-BN设计了全流程的并行化工作，第一阶段是对构造SS的并行化，以便高效生成SS。第二阶段是遗传操作的并行化，将各算子进行改造使各算子适应并行化操作并减少

表 5: DGA-BN和分布式混合算法在时间消耗上的对比

算法	Asia	Cancer	Survey
DGA-BN	469.739	62.172	71.735
distributed hybrid algorithm	1534.073	135.838	324.468

不必要的落盘次数加快效率。第三阶段是对BIC评分计算的并行化，同时引入Redis，加大第一阶段和评分计算阶段中间计算数据的复用性减少冗余计算。提出的分布式并行化方法通过消融实验验证了有效性。相比串行GA算法，本文提出的DGA-BN方法在BN质量和算法效率上均有较好的表现，同时较好的扩展性。相比原有的并行混合算法，DGA-BN可以在效率更高的情况下得到更高的结构准确性。

参考文献（References）

[1] S. Gheisari and M. R. Meybodi, "Bnc-pso: structure learning of bayesian networks by particle swarm optimization," *Information Sciences*, vol. 348, pp. 272–289, 2016.

[2] J. Xuan, J. Lu, G. Zhang, R. Y. Da Xu, and X. Luo, "A bayesian nonparametric model for multi-label learning," *Machine Learning*, vol. 106, no. 11, pp. 1787–1815, 2017.

[3] F. L. Seixas, B. Zadrozny, J. Laks, A. Conci, and D. C. M. Saade, "A bayesian network decision model for supporting the diagnosis of dementia, alzheimer's disease and mild cognitive impairment," *Computers in biology and medicine*, vol. 51, pp. 140–158, 2014.

[4] S. Nikolopoulos, G. T. Papadopoulos, I. Kompatsiaris, and I. Patras, "Evidence-driven image interpretation by combining implicit and explicit knowledge in a bayesian network," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 41, no. 5, pp. 1366–1381, 2011.

[5] P. LarrañAga, H. Karshenas, C. Bielza, and R. Santana, "A review on evolutionary algorithms in bayesian network learning and inference tasks," *Information Sciences*, vol. 233, pp. 109–125, 2013.

[6] K. Pearson, "X. on the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 50, no. 302, pp. 157–175, 1900.

[7] P. Spirtes, C. N. Glymour, R. Scheines, and D. Heckerman, *Causation, prediction, and search*. MIT press, 2000.

[8] J.-P. Pellet and A. Elisseeff, "Using markov blankets for causal structure learning," *Journal of Machine Learning Research*, vol. 9, no. Jul, pp. 1295–1342, 2008.

[9] G. F. Cooper and E. Herskovits, "A bayesian method for the induction of probabilistic networks from data," *Machine learning*, vol. 9, no. 4, pp. 309–347, 1992.

[10] R. Kabli, F. Herrmann, and J. McCall, "A chain-model genetic algorithm for bayesian network structure learning," in *Proceedings of the 9th annual conference on Genetic and evolutionary computation*. ACM, 2007, pp. 1264–1271.

[11] I. Tsamardinos, L. E. Brown, and C. F. Aliferis, "The max-min hill-climbing bayesian network structure learning algorithm," *Machine learning*, vol. 65, no. 1, pp. 31–78, 2006.

[12] F. Vafaee, "Learning the structure of large-scale bayesian networks using genetic algorithm," in *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*. ACM, 2014, pp. 855–862.

[13] W. L. Buntine, "Operations for learning with graphical models," *Journal of artificial intelligence research*, vol. 2, pp. 159–225, 1994.

[14] P. Larrañaga, M. Poza, Y. Yurramendi, R. H. Murga, and C. M. H. Kuijpers, "Structure learning of bayesian networks by genetic algorithms: A performance analysis of control parameters," *IEEE transactions on pattern analysis and machine intelligence*, vol. 18, no. 9, pp. 912–926, 1996.

[15] J. Suzuki, "A construction of bayesian networks from databases based on an mdl principle," in *Uncertainty in Artificial Intelligence*. Elsevier, 1993, pp. 266–273.

[16] J. Wang and S. Liu, "Novel binary encoding water cycle algorithm for solving bayesian network structures learning problem," *Knowledge-Based Systems*, vol. 150, pp. 95–110, 2018.

[17] C. Contaldi, F. Vafaee, and P. C. Nelson, "Bayesian network hybrid learning using an elite-guided genetic algorithm," *Artificial Intelligence Review*, vol. 52, no. 1, pp. 245–272, 2019.

[18] L. Di Geronimo, F. Ferrucci, A. Murolo, and F. Sarro, "A parallel genetic algorithm based on hadoop mapreduce for the automatic generation of junit test suites," in *2012 IEEE Fifth International Conference on Software Testing, Verification and Validation*. IEEE, 2012, pp. 785–793.

[19] D. Camacho, "Bio-inspired clustering: basic features and future trends in the era of big data," in *2015 IEEE 2nd International Conference on Cybernetics (CYBCONF)*. IEEE, 2015, pp. 1–6.

[20] Y.-J. Gong, W.-N. Chen, Z.-H. Zhan, J. Zhang, Y. Li, Q. Zhang, and J.-J. Li, "Distributed evolutionary algorithms and their models: A survey of the state-of-the-art," *Applied Soft Computing*, vol. 34, pp. 286–300, 2015.

[21] A. L. Madsen, F. Jensen, A. Salmerón, H. Langseth, and T. D. Nielsen, "A parallel algorithm for bayesian network structure learning from large data sets," *Knowledge-Based Systems*, vol. 117, pp. 46–55, 2017.

[22] J. Arias, J. A. Gamez, and J. M. Puerta, "Learning distributed discrete bayesian network classifiers under mapreduce with apache spark," *Knowledge-Based Systems*, vol. 117, pp. 16–26, 2017.

[23] Q. Fang, K. Yue, X. Fu, H. Wu, and W. Liu, "A mapreduce-based method for learning bayesian network from massive data," in *Asia-Pacific Web Conference*. Springer, 2013, pp. 697–708.

[24] K. Yue, Q. Fang, X. Wang, J. Li, and W. Liu, "A parallel and incremental approach for data-intensive learning of bayesian networks," *IEEE transactions on cybernetics*, vol. 45, no. 12, pp. 2890–2904, 2015.

[25] S. Li and B. Wang, "Hybrid parrallel bayesian network structure learning from massive data using mapreduce," *Journal of Signal Processing Systems*, vol. 90, no. 8, pp. 1115–1121, 2018.

[26] J. Dai, J. Ren, W. Du, V. Shikhin, and J. Ma, "An improved evolutionary approach-based hybrid algorithm for bayesian network structure learning in dynamic constrained search space," *Neural Computing and Applications*, vol. 32, no. 5, pp. 1413–1434, 2020.

[27] M. Scutari, "Learning bayesian networks with the bnlearn r package," *arXiv preprint arXiv:0908.3817*, 2009.

[28] C. J. Van Rijsbergen, "Information retrieval," 1979.