

基于 Spark 和 IPPSO_LSSVM 的短期 分布式电力负荷预测算法

王保义,王冬阳,张少敏

(华北电力大学 控制与计算机工程学院,河北 保定 071003)

摘要: 为了提高电力负荷预测的精度,应对单机运算资源不足的挑战,提出一种改进并行化粒子群算法优化的最小二乘支持向量机短期负荷预测模型。通过引入 Spark on YARN 内存计算平台,将改进并行粒子群优化(IPPSO)算法部署在平台上,对最小二乘支持向量机(LSSVM)的不确定参数进行算法优化,利用优化后的参数进行负荷预测。通过引入并行化和分布式的思想,提高算法预测准确率和处理海量高维数据的能力。采用 EUNITE 提供的真实负荷数据,在 8 节点的云计算集群上进行实验和分析,结果表明所提分布式电力负荷预测算法精度优于传统的泛化神经网络算法,在执行效率上优于基于 MapReduce 的分布式在线序列优化学习机算法,且提出的算法具有较好的并行能力。

关键词: Spark; IPPSO; LSSVM; 负荷预测; 短期预测; 支持向量机; 并行处理; 优化

中图分类号: TM 715

文献标识码: A

DOI: 10.16081/j.issn.1006-6047.2016.01.017

0 引言

电力系统负荷预测是进行电力系统决策的关键操作和规划途径。精确的负荷预测可以经济合理地安排发电机组的启停,从而确保电网运行的安全稳定,保持社会的正常生产和生活^[1-3]。随着智能电网的不断发展,电力数据的采集规模也不断膨胀,一年的数据存储规模从 GB 级增长到 TB 级,甚至 PB 级。同时为了精细化过滤,数据的维度也不断加大。

负荷预测算法的实现一直是国内外专家学者研究的热门课题。其中最小二乘支持向量机(LSSVM)以其算法简单、易实现的特点被广泛应用于负荷预测当中,但 LSSVM 中惩罚因子 c 和核函数参数 δ 需要事先指定,二者很大程度上决定 LSSVM 模型复杂程度、泛化能力以及预测精度^[4]。因此为了提高预测精度,诸多文献将粒子群优化(PSO)算法、蚁群算法等智能优化算法引入 LSSVM 算法的参数寻优中。文献[5]对 PSO 算法进行改进,加入扩展记忆因子,提高收敛速度和优化精度。文献[6]采用模拟退火(SA)算法对 LSSVM 进行参数优化,提高了 LSSVM 算法的预测精度。文献[7]利用改进的蚁群算法对 LSSVM 算法的参数进行优化,对比 BP 神经网络优化,提高了预测精度。文献[8]采用人工免疫算法对 LSSVM 算法进行优化,提高了算法的预测精度。虽然通过引入智能优化算法在不同程度上提高了 LSSVM 参数

的精度,从而提高了预测精度,但这类优化算法都以大量的迭代运算为基础,消耗大量计算资源。同时,面对电力数据量的增大和维数的提高,单机运算已很难满足负荷预测的精度和效率要求。在这种情况下,各类文献出现了并行化^[9]的解决方案,文献[10]采用粗粒度并行化设计的思想,设计实现了“卫星模型”PSO 算法,在目标函数高复杂度、数据高维度的情况下,大幅提高了 PSO 算法的执行效率和准确度。文献[11]采用 MapReduce 思想设计实现了分布式的支持向量机算法,在大数据、高维度下提高了算法效率。

Hadoop 云计算技术从 2003 年发展至今,已成为一种新兴的商业计算模型,MapReduce^[12]编程模型得到广泛推广。但 MapReduce 计算过程的数据存储依赖本地磁盘和 HDFS,面对诸如 PSO 等智能优化算法的迭代运算时,磁盘存取将占用大量时间开销,大幅降低算法整体执行效率。虽然 Google 后来开发了 Hadoop 的迭代版本 Haloop^[13-14],但 Haloop 使用局限性较大,如文献[9]中便对 Haloop 进行了改进优化,以适应文献中提出的并行化算法。Spark 发源于美国加州大学伯克利分校 AMPLab 实验室的集群计算平台,克服了 MapReduce 在迭代式计算的不足,能够很好地解决 MapReduce 不易解决的问题^[12]。

本文进行短期负荷预测的核心思想是运用分布式计算框架运算并行化算法。通过引入 Spark on YARN 内存计算平台,同时对 PSO 算法进行并行化改进,提出了一种基于 Spark 的分布式电力负荷预测算法,应用改进的并行 PSO 算法优化 LSSVM 算法进行负荷预测。在实验室搭建了 Hadoop 2.2.0 云平台,结合真实负荷数据测试算法并行性能和预测准确率。

收稿日期:2014-12-15;修回日期:2015-10-09

基金项目:国家自然科学基金资助项目(61300040);河北省高等学校科学研究计划资助项目(Z2012077)

Project supported by the National Natural Science Foundation of China(61300040) and Hebei Higher Education Research Program(Z2012077)

1 内存计算框架 Spark

1.1 数据表示模型

Spark 能够很好地解决迭代运算和交互式运算, 主要是因为 Spark 引入了弹性分布式数据集 RDD (Resilient Distributed Datasets), 它是一个有容错机制、可以被并行操作的数据集合, 能够被缓存到内存中, 而不必像 MapReduce 那样每次都从 HDFS 上重新加载数据。

在负荷预测中, Spark 将原始负荷数据集创建成 RDD, 缓存到内存, 进而被多个并行执行的任务重用。且当一个节点宕掉后丢失的 RDD 可被重构。

1.2 RDD 数据集操作

Spark 通过转换和动作对 RDD 数据集进行操作, 从而得出所需 RDD 数据集。运行机制与 MapReduce 类似, 如图 1 所示。关键操作具体描述如下。

a. input. 首先从 HDFS 或其他文件系统读取原始数据集, 并转换为 RDD 数据集。

b. flatMap. 将输入原始 RDD 数据集映射成 0 到多个输出 RDD 数据集。按照用户编写映射程序逻辑, 映射成 (key, value) 键值对。

c. Map. 按照用户编写 Map 映射函数程序逻辑, 对步骤 b 形成的 (key, value) 键值对重新进行映射, 形成新的 (key, value) 键值对。该步骤主要为步骤 d 的 Reduce 阶段确定合适的 key 字段。

d. Reduce. 类似于 MapReduce 的 Reduce 阶段, 将数据按照 key 分组后, 调用用户编写函数进行处理, 每组返回一个键值对。

e. Join. 根据 key 连接步骤 c RDD 数据集和步骤 d 结果键值对, 产生新 RDD 数据集。

f. Cache. 将 RDD 数据集缓存到内存中。

g. 判断迭代是否结束。 结束则通过 SaveAsTextFile 方法将结果保存到 HDFS 或其他文件系统; 否则返回步骤 c, 进行下一轮操作。

2 基于 Spark 的 IPPSO_LSSVM 算法

2.1 改进的并行化粒子群优化算法

PSO 算法的并行化分为粗粒度和细粒度 2 种不同的方式, 文献[10]采用粗粒度并行化思想, 设计实现“卫星”模型并行 PSO 算法, 本文在此基础上进一步改进优化, 提出改进的并行化粒子群 IPPSO (Improved

Parallel PSO) 算法, 进一步加快粒子群收敛速度和提高精确度。

PSO 算法的速度更新和位置更新公式如下^[12]:

$$\mathbf{v}_i(t+1) = \mathbf{v}_i(t) + c_1 r_1 (\mathbf{P}_i - \mathbf{x}_i(t)) + c_2 r_2 (\mathbf{P}_g - \mathbf{x}_i(t)) \quad (1)$$

$$\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{v}_i(t+1) \quad (2)$$

其中, t 为迭代次数; $\mathbf{v}_i$ 和 $\mathbf{x}_i$ 分别为粒子速度和粒子位置; $\mathbf{P}_i$ 和 $\mathbf{P}_g$ 分别为粒子经历的最佳位置向量和粒子群体经历的最佳位置向量; c_1 和 c_2 为加速常数; r_1 和 r_2 为 0~1 之间的随机数。

粗粒度并行化 PSO 算法的核心思想是将粒子群划分为若干个子群, 各子群分别采用 PSO 算法, 为了防止产生局部收敛, 设置整个种群经历最佳位置作为各个子群的中心, 粒子的“运动”围绕这个中心进行, 从而最终达到全局收敛, 同时, 子群的 PSO 算法中引入收敛因子, 子群的收敛速度也得到进一步提高。具体而言, 子群进行带收敛因子的 PSO 算法, 粒子速度的更新公式如下:

$$\mathbf{v}_i(t+1) = \chi [\mathbf{v}_i(t) + c_1 r_1 (\mathbf{P}_i - \mathbf{x}_i(t)) + c_2 r_2 (\mathbf{P}_g - \mathbf{x}_i(t))] \quad (3)$$

$$\chi = \frac{2}{2 - \varphi - \sqrt{\varphi^2 - 4\varphi}}, \quad \varphi = c_1 + c_2, \quad \varphi > 4 \quad (4)$$

其中, χ 为子群收敛因子。为了防止子群的局部收敛, 另加入 $\mathbf{P}_u$ 表示整个粒子群经历的最佳位置。将式 (3) 进行简化, 即:

$$\mathbf{v}_i(t+1) = \mathbf{G}_1 \quad (5)$$

$$\mathbf{G}_1 = \chi [\mathbf{v}_i(t) + c_1 r_1 (\mathbf{P}_i - \mathbf{x}_i(t)) + c_2 r_2 (\mathbf{P}_g - \mathbf{x}_i(t))] \quad (6)$$

则 IPPSO 算法可描述为:

$$\mathbf{v}_i(t+1) = \tau_{nf} \mathbf{G}_1 + \tau_{pf} \mathbf{G}_2 \quad (7)$$

$$\mathbf{G}_2 = c_3 r_3 (\mathbf{P}_u - \mathbf{x}_i(t)) \quad (8)$$

$$\tau_{nf} = \arctan\left(\frac{\lambda_n}{t}\right) \frac{2}{\pi}, \quad \tau_{pf} = \arctan\left(\frac{t}{\lambda_p}\right) \frac{2}{\pi} \quad (9)$$

其中, c_3 为整个粒子群加速因子; r_3 为 0~1 之间随机数; τ_{nf} 和 τ_{pf} 分别为负反馈和正反馈系数; λ_n 和 λ_p 分别为负反馈和正反馈因子。由式 (7) 可以看出在算法迭代前期, τ_{nf} 较大, 而 τ_{pf} 较小, 粒子子群对速度更新产生的影响较大, 让粒子尽可能地扩大运动范围; 在迭代后期, τ_{nf} 较小, 而 τ_{pf} 较大, 中心对速度更新产生影响更大, 让粒子群加速收敛。

2.2 IPPSO 算法优化的 LSSVM 算法

LSSVM^[15] 通过线性方程组, 即式 (10), 利用训练样本数据集求得 b 和 α , 其中 $\mathbf{w} = [w_1, w_2, \dots, w_n]$ 表示训练样本输出, $\mathbf{E} = [1, 1, \dots, 1]^T$, $\mathbf{I}$ 为 $n \times n$ 阶单位矩

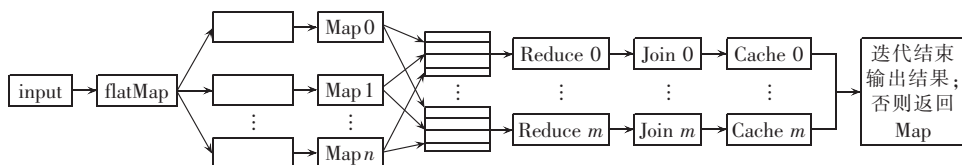


图 1 RDD 数据集操作运行机制

Fig.1 Mechanism of RDD dataset operation

阵, $\alpha = [\alpha_1, \alpha_2, \dots, \alpha_n]^T$ 表示拉格朗日乘子, Ω 是核函数, 本文采用高斯径向基核函数, 即式(11)。再通过式(12), 利用 $\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n]$ 得到预测结果 $\mathbf{w}$ 。

$$\begin{bmatrix} 0 & E^T \\ E & \Omega + c^{-1}I \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{w} \end{bmatrix} \quad (10)$$

$$k(\mathbf{x}, \mathbf{w}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{w}\|^2}{2\delta^2}\right) \quad (11)$$

$$f(\mathbf{x}) = \sum_{i=1}^n \alpha_i k(\mathbf{x}, \mathbf{x}_i) + b \quad (12)$$

本文通过运用分布式内存计算框架 Spark, 利用并行化算法对 LSSVM 的参数 c 和 δ 进行优化, 即 IPPSO 算法的输入为一个二维向量 $[c, \delta]$ 。算法的执行步骤如下。

a. 初始化粒子群, 在取值范围内随机生成一组粒子向量 $[c, \delta]$ 。训练一次 LSSVM 通过模型输出预测结果, 再按式(13)计算适应度值:

$$f_{\text{fit}} = \frac{1}{l} \sum_{i=1}^l |y_i - y'_i| \quad (13)$$

其中, y_i 为真实结果; y'_i 为预测结果; l 为结果个数。

b. 对 IPPSO 算法参数初始化, 步骤 **a** 中初始值作为粒子个体最佳位置。

c. 各子群搜索自身的最小适应度值作为子群最佳适应度值, 对应粒子作为子群最优解。根据各子群最佳适应度值搜索最小值作为整个粒子群最佳适应度值, 对应粒子作为整体最优解。

d. 根据 IPPSO 算法更新粒子速度和位置。将更新后粒子重新代入 LSSVM 训练, 计算并更新各个粒子适应度值。

e. 分别更新子群最佳适应度值、子群最优解、整体最佳适应度值、整体最优解。

f. 达到迭代次数则终止, 否则返回步骤 **d**。

g. 输出整体最优解, 得到最优参数 c 和 δ 。

h. 利用 c, δ , 根据预测样本进行负荷预测。

2.3 Spark 框架下 IPPSO_LSSVM 算法具体步骤

本文的并行化优化算法在分布式内存计算框架 Spark 上进行运算, 其中, 实现该算法的 RDD 数据集数据格式如下:

(ID, ColonyID, $\mathbf{x}, \mathbf{v}, f_{\text{fit}}, (P_i, f_{\text{fit}}), (P_g, f_{\text{fit}}), (P_u, f_{\text{fit}}))$

其中, ID 为粒子群编号, 文中一致编号为 1; ColonyID 为子群编号; $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$ 为粒子的当前位置向量; $\mathbf{v} = (\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n)$ 为粒子的当前速度向量; f_{fit} 为粒子的当前适应度值, 通过式(13)计算得出; (P_i, f_{fit}) 为个体经历的最佳位置信息向量, $P_i = (x_{i1}, x_{i2}, \dots, x_{in})$ 为个体经历的最佳位置, f_{fit} 为个体经历的最佳适应度值; 同理 (P_g, f_{fit}) 和 (P_u, f_{fit}) 分别对应子群经历的最佳位置信息向量和整体粒子群经历的最佳位置信息向量。

分布式内存计算框架 Spark 下通过对 RDD 数

据集操作实现本文并行化算法的具体步骤如下:

a. 初始化粒子群, 在取值范围内随机生成粒子群, 包括初始位置和初始速度, 并按照预先计划子群数量确定粒子 ColonyID;

b. Map 操作, 训练 LSSVM, 通过并行化雅可比迭代法求解式(10)获得预测值, 从而获得粒子的个体适应度值 f_{fit} , 并更新 (P_i, f_{fit}) , 同时以 ColonyID 为 key 值, 其他数据为 value, 构成 (key, value) 键值对;

c. Reduce 操作, 获取各个子群的最佳适应度值和对应最优解, 产生 (key, value) 键值对, 其中 key 是 ColonyID, value 是 (P_g, f_{fit}) 向量;

d. Join 操作, 将步骤 **c** 获取键值对与步骤 **b** 中键值对连接, 再进行一次 Map, 更新 (P_g, f_{fit}) ;

e. Map 操作, 以 ID 为 key, 其他数据为 value 重新构造 (key, value) 键值对;

f. 再一次 Reduce, 获得整体粒子群最佳适应度值和对应最优解, 产生 (key, value) 键值对, 其中 key 是 ID, value 是 (P_u, f_{fit}) 向量;

g. Join 操作, 将步骤 **f** 获得键值对与步骤 **e** 中键值对连接, 再进行一次 Map, 更新 (P_u, f_{fit}) ;

h. 迭代结束则输出 P_u 获得最优解, 否则返回步骤 **b** 进行下一轮迭代;

i. Map 操作, 将预测样本数据集与最优解集合映射成新 RDD 数据集, 再进行 Reduce 操作, 根据式(12)得出预测结果。

3 实验与分析

文中所有的实验都是在实验室搭建的 Hadoop 平台上运行的。平台由 16 个节点组成, 物理机配置为 i5 处理器, 主频 2.30 Hz, 内存 4 G, 硬盘 100 G, 网络带宽 100 Mbit/s 以太网。Hadoop 版本为 2.2.0, Spark 版本为 1.0.2。

3.1 原始数据集

原始数据集采用 2001 年欧洲智能技术网络 EUNITE (EUropean Network on Intelligent TEchnologies) 组织的中期电力负荷预测竞赛提供的某地区 1997、1998 年真实负荷数据^[17]。EUNITE 提供的数据样本为从 1997、1998 年每 0.5 h 采集的电力负荷; 1995—1998 年的平均日气温、1997—1999 年节假日日期。负荷预测的目标是通过以上数据样本预测出 1999 年 1 月 1 日到 31 日最高电力负荷值。

输入样本包括 3 个特征向量, 即 $[D, T, L]$, 其中 D 表示日期向量, T 表示温度向量, L 表示当天最大负荷向量, 其中 T 和 L 均用式(14)进行归一化处理。

$$G'_i = \frac{G_i - G_{\min}}{G_{\max} - G_{\min}} \quad (14)$$

其中, G_i 为原始数据; $G_{\min}$ 为原始数据中的最小值; $G_{\max}$ 为原始数据中的最大值; G'_i 为处理后对应数据。

为了提高预测准确度,训练样本由 1997 年 11 月至 1998 年 4 月的数据组成,训练集输出为每日的最高负荷值。

3.2 算例分析

3.2.1 预测准确率分析

本文采用平均相对误差(MAPE)作为评价预测算法的指标,如式(15)所示。

MAPE=1/n * sum_{i=1}^n |(y'_i - y_i)/y_i| * 100% (15)

其中,y_i为真实值;y'_i为预测值;n为预测值个数,本文中n=31。MAPE 越小,说明预测结果越精确。

本文选取泛化神经网络的泛函网络(functional networks)算法^[16]、基于 MapReudce 的分布式在线序列优化学习机 MR-OSELM-WA (MapReduce-Online Sequential Extreme Learning Machine-Weighted Average)算法^[18]进行算法比较,测试本文算法负荷预测性能。

本文 IPPSO 算法初始化粒子群大小为 40,在取值范围内随机产生,分为 8 个子群,每个子群 5 个粒子,迭代次数为 2000 次,λ_n=50,λ_p=100;取 φ 为 4.1,则子群收敛因子 χ=0.729;c₁=c₂=1.49445,c₃=2。为保证实验数据的客观性,执行 50 次预测计算 MAPE 平均值作为最终结果,如表 1 所示。从表 1 可以看出本文算法明显优于泛函神经网络预测算法,略优于 MR-OSELM-WA 算法。

表 1 实验结果 MAPE 值对比

Table 1 Comparison of experimental MAPE value among different algorithms

预测算法	MAPE/%
IPPSO_LSSVM	1.7302
MR-OSELM-WA	1.9498
functional networks	3.4300

另外,泛函神经网络预测算法执行时,需要更多的内存空间,一旦内存空间超过一定限制,算法执行效率将大幅下降^[19],而 MR-OSELM-WA 算法建立并提交 MapReduce 任务再到运行任务,会耗费大量时间,但基于分布式内存计算框架 Spark 的 IPPSO_LSSVM 并行化算法采用 memory to memory 的数据流转模式,大幅降低了数据周转耗时,同时利用整个云计算集群存储运算,内存空间大小足够,算法执行效率高。

图 2、图 3 为 1999 年 1 月电力负荷真实值与 IPPSO_LSSVM、MR-OSELM-WA 和泛函神经网络预测算法的负荷预测值的比较。

3.2.2 算法并行性能分析

本文采用加速比 S_{peclup} 和扩展率 S_{caleup} 来测试 IPPSO_LSSVM 算法的并行性能。

加速比是衡量并行系统或程序并行化的性能和效果的指标^[19],如式(16)所示,其中 T_s为算法的单

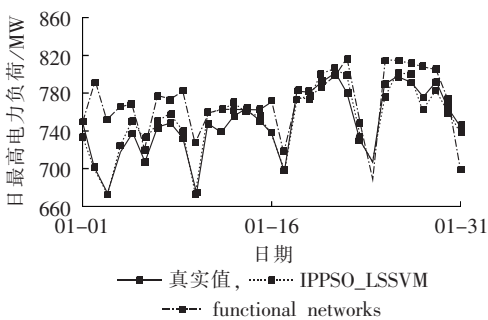


图 2 负荷真实值与 IPPSO_LSSVM、泛函神经网络预测算法预测值比较

Fig.2 Comparison among real load,load forecasted by IPPSO_LSSVM and load forecasted by functional neural network algorithm

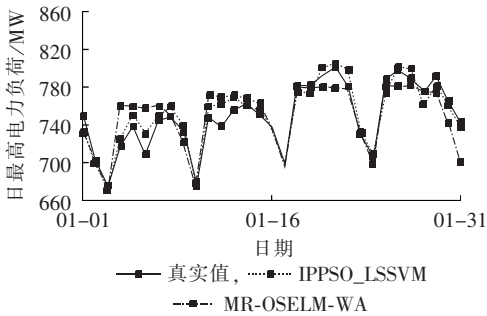


图 3 负荷真实值与 IPPSO_LSSVM、MR-OSELM-WA 算法预测值比较

Fig.3 Comparison among real load,load forecasted by IPPSO_LSSVM and load forecasted by MR-OSELM-WA

机执行耗时,T_c为算法的云集群执行耗时。

S_{peclup}=T_s/T_c (16)

扩展率是用来比较增大 m 倍的集群执行增大 m 倍的任务数据量与运行原数据集的算法执行时间,如式(17)所示,其中 T_m为增大 m 倍的集群执行增大 m 倍的任务数据量的算法执行时间,T_{origin}为增大 m 倍的集群执行原数据集的算法执行时间。

S_{caleup}=T_m/T_{origin} (17)

为了分析 IPPSO_LSSVM 算法的并行运算能力,本文将原始负荷数据样本扩大 1000 倍、2000 倍和 4000 倍,分别在集群节点个数为 4 个、8 个和 16 个的云平台上运行,以计算 S_{peclup} 和 S_{caleup}。文献[17]所提出的 MR-OSELM-WA 算法具有较好的加速比和扩展率,本文算法与 MR-OSELM-WA 算法的加速比指标比较如表 2—4 所示。在实际应用中,并行系统的算法加速比很难达到线性增长。从表中知:相较

表 2 数据集扩大 1000 倍时算法加速比比较

Table 2 Comparison of speedup rate for dataset magnified 1000 times

预测算法	加速比		
	4 节点	8 节点	16 节点
IPPSO_LSSVM	1	1.65	2.81
MR-OSELM-WA	1	约 1.5	约 2.5

表 3 数据集扩大 2000 倍时算法加速比比较

Table 3 Comparison of speedup rate for dataset magnified 2000 times

预测算法	加速比		
	4 节点	8 节点	16 节点
IPPSO_LSSVM	1	1.76	3.04
MR-OSELM-WA	1	约 1.5	约 2.4

表 4 数据集扩大 4000 倍时算法加速比比较

Table 4 Comparison of speedup rate for dataset magnified 4000 times

预测算法	加速比		
	4 节点	8 节点	16 节点
IPPSO_LSSVM	1	1.98	3.56
MR-OSELM-WA	1	约 1.8	约 2.9

MR-OSELM-WA 算法,本文算法加速比更接近线性,且随着数据量增大,接近程度更大。因此 IPPSO_LSSVM 算法能满足海量高维电力数据的负荷预测的性能要求。

本文并行算法的扩展率与 MR-OSELM-WA 算法的比较如表 5 所示。在实际应用中,扩展率不可能达到恒定为 1 的理论指标,随着数据集增大,扩展率会逐渐减小。本文算法与 MR-OSELM-WA 算法扩展率基本相当,指标表现较好。

表 5 算法扩展率比较

Table 5 Comparison of expansion rate between algorithms

预测算法	扩展率		
	4 节点	8 节点	16 节点
IPPSO_LSSVM	1	0.88	0.82
MR-OSELM-WA	1	约 0.84	约 0.78

4 结语

我国的电网智能化快速发展,这带来了数据量激增和数据维度不断提高的问题,单机运算和存储能力已不能满足电力负荷预测对效率和精度要求。近年来国内外都将解决方案集中到了算法并行化和分布式求解方面。本文提出的基于分布式内存计算框架 Spark 的并行化电力负荷预测算法不仅缩短了训练建模时间,而且提高了预测的精度,同时有效利用了大量闲置主机的运算能力。

参考文献:

[1] 万昆,柳瑞禹. 区间时间序列向量自回归模型在短期电力负荷预测中的应用[J]. 电网技术,2012,36(11):77-81.
WAN Kun,LIU Ruiyu. Application of interval time-series vector autoregressive model in short-term load forecasting[J]. Power System Technology,2012,36(11):77-81.

[2] 黄元生,方伟. 基于灰色傅里叶变换残差修正的电力负荷预测模型[J]. 电力自动化设备,2013,33(9):105-107,112.
HUANG Yuansheng,FANG Wei. Power load forecasting model with residual error correction based on gray Fourier transform[J]. Electric Power Automation Equipment,2013,33 (9):105-

107,112.

[3] 赵强,景罗,赵光俊,等. 顾及空间异质性的多尺度空间负荷预测[J]. 电力自动化设备,2014,34(2):91-96.
ZHAO Qiang,JING Luo,ZHAO Guangjun,et al. Multi-scale spatial load forecasting considering spatial heterogeneity[J]. Electric Power Automation Equipment,2014,34(2):91-96.

[4] 周华鑫. 基于 PSOEM-LSSVM 的中长期电力负荷预测及其应用研究[D]. 重庆:重庆大学,2013.
ZHOU Huaxin. Research on medium and long term power load forecasting based on PSOEM-LSSVM and its application [D]. Chongqing:Chongqing University,2013.

[5] 段其昌,周华鑫,曾勇,等. 带扩展记忆的粒子群优化最小二乘支持向量机在中长期电力负荷预测中的应用[J]. 计算机科学,2013,40(6A):41-43.
DUAN Qichang,ZHOU Huaxin,ZENG Yong,et al. Application of PSOEM-LSSVM in medium and long term power load forecasting [J]. Computer Science,2013,40(6A):41-43.

[6] 朱兴统. 基于 SA-LSSVM 的电力短期负荷预测[J]. 科学技术与工程,2012,12(24):6171-6174.
ZHU Xingtong. Power short-term load forecasting based on SA-LSSVM[J]. Science Technology and Engineering,2012,12(24): 6171-6174.

[7] 龙文,梁昔明,龙祖强,等. 基于改进蚁群算法优化参数的 LSSVM 短期负荷预测[J]. 中南大学学报:自然科学版,2011,42(11): 3408-3414.
LONG Wen,LIANG Ximing,LONG Zuqiang,et al. Parameters selection for LSSVM based on modified ant colony optimization in short-term load forecasting[J]. Journal of Central South University:Science and Technology,2011,42(11):3408-3414.

[8] 杨洁. 基于人工免疫算法优化 LSSVM 的短期电力负荷预测[J]. 山西电力,2014,2(4):4-7.
YANG Jie. Short-term power load forecasting based on LSSVM of artificial immune algorithm[J]. Shanxi Electric Power,2014,2 (4):4-7.

[9] ISARD M,BUDI M,YU Y,et al. Dryad:distributed data-parallel programs from sequential building blocks[C]//Proceedings of the 2nd AcmSIGOPS/EuroSys European Conference on Computer Systems 2007. New York,NY,USA:ACM,2007:59-72.

[10] 程兴国. 仿生算法的动态反馈机制及其并行化实现方法研究 [D]. 广州:华南理工大学,2013.
CHENG Xingguo. Research on dynamic feedback mechanism for the bionical algorithms and parallel implementation [D]. Guangzhou:South China University of Technology,2013.

[11] BHATOTIA P,WIEDER A,RODRIGUES R,et al. Incoop: MapReduce for incremental computations[C]//Proceedings of the 2nd ACM Symposium on Cloud Computing. New York,NY, USA:ACM,2011:1-14.

[12] 雷学智. 云计算平台下分布式支持向量机在煤炭行业分类预测应用[J]. 煤炭技术,2013,32(11):248-250.
LEI Xuezhi. Application of distributed support vector machine basedon cloud platform in coal system[J]. Coal Technology, 2013,32(11):248-250.

[13] BU Y,HOWE B,BALAZINSKA M,et al. HaLoop:efficient iterative data processing on large clusters[J]. Proceedings of the VLDB Endowment,2010,3(1-2):285-296.

[14] 王岗,姜杰,唐昆明,等. 基于自适应双向加权最小二乘支持向量机的超短期负荷预测[J]. 电力系统保护与控制,2010,38(19): 142-146.

- WANG Gang,JIANG Jie,TANG Kunming,et al. Ultra-short-term load forecasting based on adaptive bidirectional weighted least squares support vector machines[J]. Power System Protection and Control,2010,38(19):142-146.
- [15] 董西成. Hadoop 技术内幕:深入解析 YARN 架构设计与实现原理[M]. 北京:机械工业出版社,2014:60-120.
- [16] Europe Network on Intelligent Technologies for Smart Adaptive Systems. World-wide competition within the EUNITE network [EB/OL]. (2001-12-14)[2013-09-05]. <http://neuron.tuke.sk/competition/>.
- [17] CASTILLO E,GUIJARROB,ALONSO A. Electricity load forecast using functional networks[R]. Puerto de la Cruz,Spain:[s.n.], 2001.
- [18] 王保义,赵硕,张少敏. 基于云计算和极限学习机的分布式电力负荷预测算法[J]. 电网技术,2014,38(2):526-531.
- WANG Baoyi,ZHAO Shuo,ZHANG Shaomin. A distributed load forecasting algorithm based on cloud computing and extreme learning machine[J]. Power System Technology,2014,38(2):526-531.

- [19] 曾飞,苗世洪,尚亚男,等. 分解协调式节点阻抗矩阵生成算法的并行实现[J]. 电网技术,2011,35(9):99-104.
- ZENG Fei,MIAO Shihong,SHANG Yanan,et al. Parallel implementation of node-impedance matrix generation algorithm based on decomposition and coordination[J]. Power System Technology, 2011,35(9):99-104.

作者简介:



王保义

王保义(1964—),男,山东栖霞人,教授,博士,主要研究方向为电力信息化与信息安全(E-mail:wangbaoyi@126.com);

王冬阳(1988—),男,河南开封人,硕士研究生,主要研究方向为电力信息化与信息安全(E-mail:wdy_newdays@163.com);

张少敏(1965—),女,河北保定人,教授,博士,主要研究方向为电力信息化与信息安全(E-mail:zhangshaomin@126.com)。

Distributed short-term load forecasting algorithm based on Spark and IPPSO_LSSVM

WANG Baoyi,WANG Dongyang,ZHANG Shaomin

(School of Control and Computer Engineering,North China Electric Power University,Baoding 071003,China)

Abstract: Aiming at the insufficient resource of single computer,a short-term load forecasting model based on LSSVM(Least Squares Support Vector Machine) optimized by IPPSO(Improved Parallel Particle Swarm Optimization) algorithm is proposed to improve the accuracy of load forecasting. A Spark-on-YARN memory computing platform is introduced and the IPPSO is operated there to optimize the uncertain parameters of LSSVM,which are then applied in the load forecasting. The parallel and distributed computation is adopted to improve the accuracy of forecasting algorithm and the capability of massive high-dimensional data processing. Experiment and analysis are carried out with the actual load data provided by EUNITE on an 8-bus cloud computing platform and results show that,the proposed algorithm has better accuracy than the generalized traditional neural network algorithm and better efficiency than the MR-OSELM-WA(MapReduce-Online Sequential Extreme Learning Machine-Weighted Averaged) algorithm.

Key words: Spark; IPPSO; LSSVM; electric load forecasting; short-term forecasting; support vector machines; parallel processing; optimization

(上接第 116 页 continued from page 116)

Optimal reconfiguration of distribution network with DG and STATCOM

WU Zhongqiang,ZHAO Liru,JIA Wenjing,WU Changhan

(Key Lab of Industrial Computer Control Engineering of Hebei Province,College of Electrical Engineering, Yanshan University,Qinhuangdao 066004,China)

Abstract: An improved GA(Genetic Algorithm) is applied to realize the DNR(Distribution Network Reconfiguration),which introduces DG(Distributed Generation) and STATCOM into the distribution network model to include its recent development. The influences of DG and STATCOM on DNR and power quality improvement are discussed. The hybrid binary and decimal coding method and the special crossover and mutation operation are adopted to avoid the infeasible solutions of GA. The cloud algorithm is applied to improve the crossover rate and mutation rate for enhancing the convergency of GA. The simulative results for IEEE 33-bus distribution network prove that the proposed method realizes the DNR effectively and the application of DG and STATCOM enhances the power quality and reliability successfully.

Key words: network reconfiguration; distribution network; genetic algorithms; cloud model; distributed power generation; STATCOM