A Nonrelational Data Warehouse for the Analysis of Field and Laboratory Data From Multiple Heterogeneous Photovoltaic Test Sites

Yang Hu, *Member, IEEE*, Venkat Yashwanth Gunapati, Pei Zhao, Devin Gordon, Nicholas R. Wheeler, Mohammad A. Hossain, *Member, IEEE*, Timothy J. Peshek, *Member, IEEE*, Laura S. Bruckman, Guo-Qiang Zhang, *Member, IEEE*, and Roger H. French, *Member, IEEE*

***Abstract*—A nonrelational, distributed computing, data warehouse, and analytics environment (Energy-CRADLE) was developed for the analysis of field and laboratory data from multiple heterogeneous photovoltaic (PV) test sites. This data informatics and analytics infrastructure was designed to process diverse formats of PV performance data and climatic telemetry time-series data collected from a PV outdoor test network, i.e., the Solar Durability and Lifetime Extension global SunFarm network, as well as point-in-time laboratory spectral and image measurements of PV material samples. Using Hadoop/HBase for the distributed data warehouse, Energy-CRADLE does not have a predefined data table schema, which enables ingestion of data in diverse and changing formats. For easy data ingestion and data retrieval, Energy-CRADLE utilizes Hadoop streaming to enable Python MapReduce and provides a graphical user interface, i.e., py-CRADLE. By developing the Hadoop distributed computing platform and the HBase NoSQL database schema for solar energy, Energy-CRADLE exemplifies an integrated, scalable, secure, and user-friendly data informatics and analytics system for PV researchers. An example of Energy-CRADLE enabled scalable, data-driven, analytics is presented, where machine learning is used for anomaly detection across 2.2 million real-world current-voltage *(I–V)* curves of PV modules in three distinct Köppen–Geiger climatic zones.**

***Index Terms*—Data science, degradation science, Hadoop, HBase, informatics, photovoltaics (PVs).**

I. Introduction and Background

MOST utility-scaled photovoltaic (PV) systems are instrumented with meteorological and irradiance sensors to monitor the system performance and weather conditions according to PV system monitoring guidelines [1], [2]. These information-rich, temporally continuous or semicontinuous datasets are often complemented by high-resolution solar data and other climate data from nearby regional weather stations measurements, and TMY3 data in the National Solar Radiation Database [3]. In addition, irradiance data can be retrieved from geographic information system (GIS) satellite images; an example is the commercial datasets from SolarGIS [4]. PV system monitoring has proven useful over many years, and more specific methods for PV energy data analysis are needed to improve the performance of new installations and improve the management of existing power plants [5]. Recent durability studies of PV materials show that a more scalable data warehouse and analytics infrastructure is enabling vast new insights [6]–[8], and the new approach to degradation science of power systems raises the demand for large-volume high-temporal-resolution real-world data [6].

At the Solar Durability and Lifetime Extension (SDLE) Research Center, a “Global SunFarm Network” (GSFN) was established with 16 outdoor test facilities in six countries [9]. As shown in Table I, these sites are heterogeneous in terms of plant topologies and hardware. In order to process, store, and analyze about 120 GB of time-series data that accumulate from the GSFN each year and integrate them with laboratory-based data, we need a data informatics infrastructure that is dedicated to solar energy research. Existing solar energy databases are based conventionally on relational database management systems (RDBMS), for example, several databases supported by the National Renewable Energy Laboratory (NREL) in the Open PV Project [10], and PV Data Acquisition project [11], as well as the International Energy Agency’s Photovoltaic Power Systems Program performance database [12]. The initial data infrastructure design of the SDLE GSFN originates from NREL’s proposed regional test center (RTC) data warehouse proposal [13], which uses a typical RDBMS MySQL database.

Several drawbacks of the initial design influenced the performance and efficiency of the system. First, MySQL databases require a static database schema, which limits the scalability of the database structure and increases the preprocessing overhead of ingestion. Moreover, the proposed RTC schema required having individual tables for each data type, which is costly to construct and to maintain. As GSFN expanded to more test facilities and commercial PV power plants, the poor scalability of the RDBMS data model and accumulation of a large number of tables would be an obstacle in the near future. Furthermore, sequential data processing would be delayed resulting in longer database write times, due to the atomicity, consistency, isolation, and durability properties of RDBMS databases [14].

Manuscript received July 18, 2016; revised September 29, 2016; accepted October 31, 2016. Date of publication December 1, 2016; date of current version December 20, 2016. The SDLE Research Center was established through funding through the Ohio Third Frontier, Wright Project Program Award Tech 12-004. The Energy-CRADLE project was supported by the Bay Area Photovoltaic Consortium Prime Award DE-EE0004946 under Subaward Agreement 60220829-51077-T.

The authors are with the Solar Durability and Lifetime Extension Research Center, Case Western Reserve University, Cleveland, OH 44106 USA (e-mail: yang.hu@case.edu; vxg120@case.edu; pxz83@case.edu; dag109@case.edu; nrw16@case.edu; mohammad.a.hossain@case.edu; tjp3@case.edu; lsh41@case.edu; gqatcase@gmail.com; roger.french@case.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JPHOTOV.2016.2626919

TABLE I
DESCRIPTION OF TIME-SERIES DATA SOURCES

Data Source	SF	Location	Instruments	DataType
SDLE	6	Cleveland OH, USA	String Inverter, Micro Inverter, DayStar Multitracer, Campbell Datalogger	XML-IV Curve, JSON, Split-CSV, DAT
Replex	1	Mt Vernon OH, USA	DayStar Multitracer, Campbell Datalogger	XML-IV Curve
AEP	1	Dolan OH, USA	String Inverter	Split CSV
Q-Lab	2	AZ&FL, USA	Datalogger	CSV
UL Taiwan	2	Taiwan, China	Power Meter, datalogger	SQL
IITGN	1	GN, India	Micro Inverter, Datalogger	CSV, DAT
Fraunhofer ISE	3	Mount Zugspitze, Gran Canaria, Negev Desert	Power Meter, Datalogger, *I–V* tracer	CSV

Besides the time-series data from GSFN, spectral and image data collected using nondestructive spectroscopic and imaging techniques are also critical in studies of degradation mechanisms of PV components, such as monitoring backsheet chemical changes, encapsulant browning, and encapsulant delamination. Spectral and image data are measured periodically in the laboratory on PV component samples that have undergone outdoor exposure or indoor accelerated exposure. These data types are naturally different from time-series data. It is not suitable to ingest spectral data into the original database schema. However, in practice, cross-correlation of different measurements on the same sample is often necessary for studies spanning real-world and lab-based data and for the development of mesoscale evolution models for PV systems [15], [16].

This scalability issue can be addressed by using a distributed computing platform and nonrelational databases such as NoSQL or NewSQL databases [17]. Several recent applications have shown the advantage of using nonrelational database for high-dimensional and large-scale data in chemistry [18], biology [19], and climate research [20]. The marriage of big data and distributed computing with high-performance computing will result in scalable data analytics [21].

We have developed a stable, scalable, fast processing, and querying system based on Hadoop to replace our initial MySQL RDBMS design. We will introduce a common research analytics and data lifecycle environment for energy (Energy-CRADLE), using the distributed computing platform Hadoop and its database tool, HBase, to solve the existing data management and processing metrics (see Section II). We will illustrate the data processing and data extraction performance. To demonstrate that Energy-CRADLE enables scalable PV research, the result of using a machine learning algorithm [22] to classify 2.2 million current-voltage *(I–V)* curves is also presented (see Section III). Finally, we will discuss what challenges and new directions lie ahead (see Section IV).

II. Energy-CRADLE Development

Energy-CRADLE serves as a platform of data acquisition, data processing, data storage, data sharing, and analytics that is built to meet the requirements of PV system performance data [23].

A. Hadoop and HBase

A widely used parallel distributed computing suite Apache Hadoop 1.1.2 [24] and its nonrelational database HBase 0.94 [25] were utilized in our distributed computing model. The Hadoop architecture is based upon a master–slave concept, where the namenode controls datanodes by allocating tasks and storage. Apache Hadoop 1.1.2 software comes with the default Hadoop distributed computing framework MapReduce [26], and the distributed file system Hadoop Distributed File System (HDFS), which is based upon the Google File System [27].

The nonrelational database, i.e., Apache HBase, is a NoSQL database based upon Google’s Bigtable [28], which was developed as part of the Apache Hadoop project. HBase is paired with Apache Zookeeper, which enables highly reliable distributed coordination [29]. Apart from the Zookeeper server, HBase comes with service modules such as Thrift [30] and Stargate, which are based on remote procedure call and representational state transfer (REST).

Our Hadoop system was installed as a service on our existing cloud architecture, using virtual machines (VMs) on a VSphere server for the namenode and datanodes of the distributed computing cluster.

Energy-CRADLE1.1 uses a single master node and eight slave nodes in our Hadoop cluster, and HBase is configured on top of HDFS. Each VM has a two-core processor with 2.2 GHz per Core, 10 GB of RAM, 100 GB secondary storage, and access to 5-Tb network drive. The operation system of the VMs is Kubuntu 12.04 LTS.

Across the GSFN, the data were originally collected or generated by many different instruments. Each instrument reports data in different formats, including conventional CSVs, split CSVs, XML, JSON, and DAT file formats. For some SunFarm locations, direct access to the internal network was not available for security concerns, so the data were first wrapped up into an SQL file and then pushed to a MariaDB server. Table I lists the instruments used in each SunFarm location and the corresponding data types that are collected.

Besides time-series data, spectral data were generated by the instruments listed in Table II. All of the instruments were located in the laboratories of the SDLE Research Center.

TABLE II
DESCRIPTION OF SPECTRAL DATA SOURCES

Instruments	Spectra Type	DataType
Agilent Cary 6000i UV-Vis-NIR Spectrophotometer	Optical absorbance spectra	.spc
Agilent Cary 630 Spectrophotometer	FTIR spectra	.spc
Cary Eclipse Fluorescence Spectrophotometer	Fluorescence spectra	.spc
SDLE EL camera	Electroluminescence	.jpg

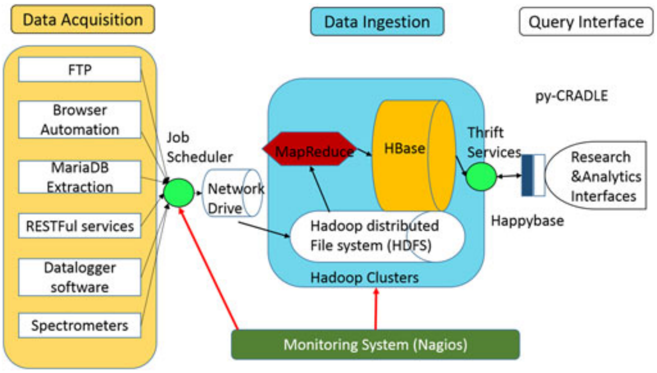

Fig. 1. Flow chart of data ETL in Energy-CRADLE. Data are first collected via different data acquisition methods and temporarily stored on a local network drive; then, data are pushed on to HDFS, where multiple MapReduce jobs run to register data onto HBase; users can extract data from HBase via py-CRADLE interfaces. Nagios monitors the data acquisition process, as well as all the MapReduce jobs run on Hadoop clusters.

B. Data Acquisition

Data acquisition is the initial step of the data Extract, Transform, and Load (ETL) of Energy-CRADLE [31]. As shown in Fig. 1, data need to be collected from different data sources and stored on a local network drive. Due to the variety and diversity of data sources, six different data acquisition methods have been deployed.

1) *FTP server:* Automatically pull data daily from instruments provided by their FTP servers. An central FTP server is also provided by us for collaborators to push data to CRADLE.
2) *Web browser automation:* The Selenium Python client [32], [33] enables online data acquisition using webscraping from online data dashboards of instruments.
3) *MariaDB extraction:* A MariaDB server was hosted locally to enable data to be pushed to our MySQL database.
4) *RESTful services:* Instruments send HTTP requests to the REST server on a minute-by-minute basis and store the retrieved data as a CSV file.
5) *Datalogger software:* Campbell's LoggerNet Server, which runs on a PC in the SDLE research center, remotely controls multiple dataloggers on each SunFarm and collects data from each datalogger across the GSFN, generating a CSV file every two hours for each datalogger.
6) *Spectrophotometers and EL camera:* Spectral data files (.spc files) are generated by spectrophotometers as a default interchange format. The electroluminescence (EL) images are taken from a customized EL camera setup developed in the SDLE laboratory [34]. These .spc and .jpg files, along with other lab-based evaluation files, are saved to the network drive.

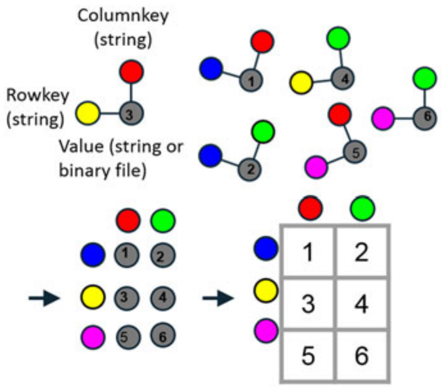

Fig. 2. Registration of triples into Hbase data table. HBase triple consists of a row key, a column key, and the input value. Triples share the same row key, e.g., value 1 and value 2 share the same row key in blue, to form a row in the HBase table. Triples share the same column key, e.g., values 1, 3, and 5 share the same column key in red to form a column in the Hbase table.

C. Data Ingestion

There are two steps in the data ingestion process. The first step is to periodically transfer new data from our network drive into HDFS. This transferring action is achieved by shell scripts run as Cron jobs on Linux hosts. The second step is to run MapReduce programs to read new data files and register them into HBase. In Energy-CRADLE 1.1 ingestion, the map-reduce jobs only need to perform mapping of the input data into our HBase data model; there is no associated reduce function needed.

In HBase, each input value is saved as triple [35]. As shown in Fig. 2, each triple consists of a row key, a column key, and the input value. Unlike relational databases, where columns and column names are predefined and fixed in HBase, columns are created dynamically and closely related data are stored under the same column family. After input data tables are split into individual triples, they can be reconstructed from triples having the same row key or the same column key that are assembled into a data table by Hbase. For time-series data, it is intuitive to use timestamps as a unique identification key for each value. Thus, in an HBase table, timestamps are used as the row keys. The column key consists of column family and column qualifier. Each SunFarm's location is used as column family name, which facilitates faster data lookup and reduces the need for data aggregation. Column qualifiers are created dynamically according to the metadata of each data source. The full column key consists of "column family: instrument type-instrument serial number-attributes." During data extraction, this naming convention can help selectively filter the data. All the time-series data files are naturally separated by time; as we use the timestamps as row keys, we are able to map each row in the original data file into Hbase independently. This separation makes it possible for multiple nodes to process the same data file in parallel, dramatically reducing the processing time. It is not cost effective to save an *I–V* curve containing 200 points as individual triples. Thus, we split each current and voltage pair, reassemble them into two strings of current and voltage values, and save the strings as list objects in Hbase.

For spectral data, as the data file is small enough (10–20 kB), each .spc file is saved as an entry in HBase. In contrast with time-series data, the spectral data are not collected in a continuous time manner; therefore, it is not suitable to use timestamps as row keys. Instead, the row keys of spectral data consist of four parts: sample id, exposure step, measurement step, and measurement number.

D. Data Extraction and Query (py-CRADLE)

In order to extract data from Energy-CRADLE, we developed a command line and graphical user interface with Python called py-CRADLE. In HBase, the scan command is used to read across the table and then apply a filter to extract the relevant records. Filters in HBase can be applied to row keys, column qualifiers, or data values. Hbase scan commands are naturally written in Java. For non-Java users, a Thrift interface provides a client API to communicate with HBase using various programming languages such as Ruby, Python, PHP, etc. [30]. Py-CRADLE was developed using the Happybase module in Python to interface with the Thrift server in HBase to perform data extraction [36].

The underlying operation of py-CRADLE is to first retrieve data from HBase; then either creating a CSV file to store the data or downloading the data as binary files onto the local hard drive. In order to query data more efficiently, a web server serving as a databook was implemented to query all available column qualifier names in HBase every night. Users can copy and paste the column qualifier to the query interface to extract a particular attribute or variable from HBase. An advantage of the HBase scan method is that applying multiple filters at the same time will not add extra computational load because the number of rows being scanned does not change. By contrast, in relational databases, multiple constraints added to a query command will slow down the data retrieval process as a result of implicit or explicit table joins.

E. Monitoring System (Nagios)

Nagios core 3.5.1 and Nagios Service Check Acceptor (NSCA) [37] are implemented to monitor the whole Energy-CRADLE project. Nagios core is an industry standard for network monitoring hosted on Linux systems. It is used for monitoring the status of the eight data nodes of the Hadoop cluster, HDFS, HBase, and the Thrift server. NSCA, which is a Linux/Unix daemon, allows passive alerts and checks from remote machines and applications with Nagios. It makes it possible for Nagios to accept log messages submitted by data acquisition scripts.

III. Results

A. Data Processing Performance

In Energy-CRADLE 1.1, we implemented MapReduce jobs in Python using the Hadoop streaming method. In comparison with the Java language, Python MapReduce code is simpler to develop and easier to understand. In order to test whether using a higher level language, such as Python MapReduce, has introduced too much overhead, an experiment was performed on the running time of Python MapReduce compared to native Java MapReduce.

Both Python MapReduce and Java MapReduce scripts share the same algorithm and process the same amount of data, and the experiment is performed on the same Hadoop cluster. The data file being processed is 85 MB in size, with 13 columns and 1 400 000 rows. It is a typical sized CSV file that we would collect and process from one data source every day.

The Python MapReduce job runs 5 minutes to complete, and the Java MapReduce job runs 3.5 minutes to complete. The experiment proved that using Python MapReduce does not tremendously increase the data processing time while broadening the researcher base that is able to use Energy-CRADLE.

B. Data Extraction Performance

As mentioned in the previous section, py-CRADLE, a Python GUI, was developed for data extraction. In order to test the data extraction speed, another experiment was conducted that extracted time-series data and spectral data from HBase. In this experiment, 20 MB of each data type, time-series data and spectral data, are extracted using py-CRADLE. Time-series data extraction took 75 s, while the spectral data extraction took only 0.498 s. Noting that the time-series data are stored as a collection of single values, while the spectral data are stored as a binary file, the huge difference can be understood by simply considering the data types being extracted. For time-series data being extracted, it is a single CSV file containing 400 000 rows and 13 columns, which are 5 200 000 data points in HBase. The spectral data file contains 120 spc (binary) files. It takes much less time for HBase to extract 120 data points than 5 200 000 data points, even though each of the 120 data points are thousands of times larger than each entry of time-series data.

Both time-series data and spectral data do not show errors or missing values introduced by data processing and data extraction through Energy-CRADLE. In comparison with conventional solar database schema, for example, the NREL's RTC schema, Energy-CRADLE is able to process more data types, such as *I–V* curve data and spectral data.

C. Energy-CRADLE Analytics Example: I–V Curve Classification

As an example of scalable PV research studies enabled by Energy-CRADLE, a recent case study was conducted on classifying 2.2 million real-world current–voltage measurements (*I–V* curves) [38]. These *I–V* curves were measured from three Fraunhofer ISE SunFarms, which are Gran Canaria, Spain (GC), Mount Zugspitze, Germany (UFS), and Negev Desert, Israel

(NEG). These three sites are located in three distinct climate zones, Bsk, ET, and Bwh respectively, according to Köppen–Geiger climate classification [39]. *I–V* curves of two module samples are measured every 5 minutes. The two modules on each site are from two distinct brands, and these two brands are identical across three sites. Sites GC and UFS have been recording data since 2010, while the NEG site started recording in 2012. There are 0.75 million *I–V* curves collected from Site GC, 0.85 million from Site UFS, and 0.55 million from Site NEG. Besides *I–V* curve data, the maximum power output of each module is recorded by a power meter approximately every minute between two *I–V* measurements. All the data were first ingested into Energy-CRADLE and then retrieved for analysis in an identical format.

From previous research, we found that "step" *I–V* curves, or *I–V* curves showing bypass diodes becoming forward bypassed and turning on during the *I–V* measurement, can provide insights to any heterogeneity inside the PV module and over time [22], [38]. These "step" *I–V* curves are typically ignored in PV module studies because they cannot be described using a simple diode model and were believed to arise due to random measurement errors. However, using a machine-learning anomaly detection algorithm, we are able to classify massive real-world *I–V* curves measurements. We classify *I–V* curves into five categories: Type I curves show only V_{oc} and no additional change point, Type II show V_{oc} and one change point, and Type III show V_{oc} and two change points; Types II and III are "step" curves. There are curves measured at low irradiance (for example during night time) or interrupted; these curves are classified as "small.amps" or "few.points" and will be eliminated from further analysis. The classification algorithm is based on local regression and residual thresholding; the details of the algorithm are out of the scope of this paper and have been published elsewhere [22].

By carefully examining the classification results, we found that one module on NEG site had high short-circuit current (I_{sc}), which exceeded the *I–V* tracer's measurement range. As a result, a number of *I–V* curves show clipping near I_{sc} ,which is not necessarily caused by bypassing. Thus, we excluded the result from that module; the classification results of the rest of the modules are shown in Fig. 3. Fig. 3 shows the percentage of Type II and III *I–V* curves, which are the "step" curves, in each year. Five PV module samples are differentiated by shape; the modules on the same site have the same color.

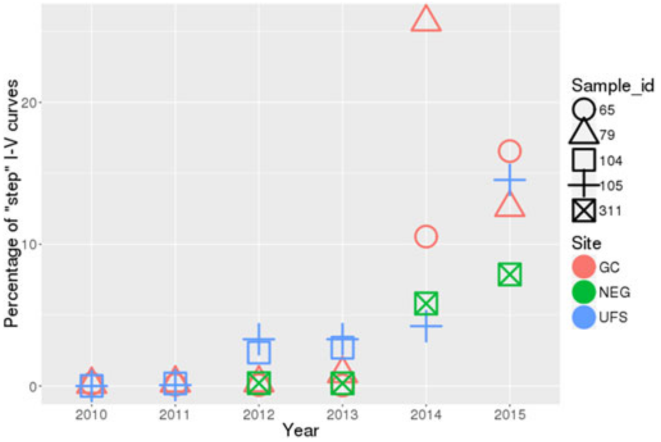

Fig. 3. Percentage of Type II and Type III *I–V* curves ("step" *I–V* curves) collected from five modules. Modules 65 and 79 are on Site GC, modules 104 and 105 are on Site UFS, and module 311 is on Site NEG. A clear increasing trend of the proportion of "step" curves in each year is observed from all five module samples.

IV. Discussion

The distributed computing platform, i.e., Hadoop, and the NoSQL big data storage engine, i.e., HBase, are well suited for scalable PV data analytics. Energy-CRADLE features integrated data acquisition, data storage, and data retrieval. It provides a one-stop data informatics solution for PV researchers. Energy-CRADLE utilizes HBase's key value pair schema; it makes it possible to ingest time-series data, *I–V* curve data, and spectral data in the same database, which is convenient for conducting analyses across different data types. It also enables adding new power plants' data that contains new variables to existing data tables, which is not possible using RDBMS predefined table schema. Energy-CRADLE handles data that is commonly found in both commercialized PV power plants and outdoor test facilities for research purposes, such as data from inverters, *I–V* tracers, and data previously stored in conventional databases. It is beneficial to all researchers who are trying to migrate from conventional RDBMS to a more scalable data warehouse.

The current version of Energy-CRADLE, which is based on Hadoop 1.1.2, has been used for several pilot case studies [40], [41]. With py-CRADLE, researchers can download the data needed and perform analytics on their local computers using analytical programs such as R or Python. However, the process of downloading the desired dataset becomes intractable, as time-series PV performance and climatic data grow to giga-, tera-, and peta-byte scale over several years of data acquisition. This massive data transfer is expensive in terms of Internet bandwidth and computing resources. In order to avoid massive data transfer, we are migrating to Hadoop 2.5.0, which enables analytic programming languages such as "R" to run directly on HDFS, eliminating the need for data download to a local computer [42]. This in-place analytics will dramatically improve the analytic efficiency, especially when dealing with large datasets, such as lifetime performance data from hundreds of PV power plants.

As mentioned in Section I, in order to develop the mesoscopic evolution models for PV energy systems, laboratory-based "point-in-time" data are critical to the larger study. Laboratory-based analytical instruments only generate data whenever a measurement is made by the researcher, i.e., there is no temporal regularity of data. The typical type of data could be single value (e.g., maximum power), value strings (e.g., spectroscopy), image (2-D) (e.g., electroluminescence, *I–V* curve), or even volumetric data (3-D combinations of multidimensional datasets such as hyperspectral images). High-dimensional datasets may require more advanced preprocessing prior to ingestion into Energy-CRADLE.

A nonrelational, Hadoop-based, data warehouse and analytics environment enables the combination of different data types, makes data analytics scalable, and opens up new frontiers of PV research. From the *I–V* curve classification result, we observe that step *I–V* curves represent a small percentage of the first two years of outdoor exposure from all five modules. This indicate that there is only a negligible amount of "step" *I–V* curves caused by transient partial shading or instantaneous irradiance changing at these sites. As years of outdoor exposure increase, the percentage of "step" *I–V* curve measurements increases from all the modules. This gradual increase could be related to the heterogeneous changing of each PV module. Furthermore, as "step" *I–V* curves have lower fill factor and maximum power than Type I curves, the increasing of the percentage of "step" curves directly leads to power degradation of PV modules. This classification result enables further quantitative study of "step" *I–V* curves and PV modules' degradation, and it also enables studies of correlating *I–V* curve shape to real-world degradation mechanisms.

V. Conclusion

A nonrelational data warehouse and analytics environment, Energy-CRADLE 1.1, was developed based on Hadoop and Hbase, which enables storage of multimodal stream data from diverse data formats and dynamically formats it on-demand for the needs of PV researchers. Without any data reduction or data aggregation to fit a predefined database schema, researchers are able to study the data stored in Energy-CRADLE in an unbiased way. Python MapReduce, based on Hadoop streaming, reduces the obstacle of writing Java MapReduce code and broadens the user base. It is shown that Python MapReduce did not significantly increase the processing overhead. Py-CRADLE gives the end user a graphical interface to access data in Energy-CRADLE. Furthermore, Energy-CRADLE is highly flexible, with the ability to handle multiple data types and is supportive of the long-term growth of the PV industry. Energy-CRADLE provides a reliable, scalable, and accessible environment for researchers in the solar energy field. An example of Energy-CRADLE-enabled scalable data-driven analytics is the anomaly detection across 2.2 million real-world *I–V* curves. The percentage of "step" *I–V* curves, which show bypass diodes turn on, increases with outdoor operation time at three distinct climate zones from all five PV modules. This phenomena could be a sign of heterogeneous changing inside the modules, and it directly leads to power degradation.

Acknowledgment

The GSFN is developed in cooperation with The Medical Center Company, Cleveland, OH, USA; Replex Plastics; American Electric Power; Q-Lab Corporation; Underwriters Laboratories Taiwan; Indian Institute of Technology Gandhinagar; and Fraunhofer Institute for Solar Energy Systems. The authors would like to thank A. Loach for their help with manuscript preparation.

References

[1] *Photovoltaic System Performance Monitoring—Guidelines for Measurements, Data Exchange and Analysis*, Int. Std. IEC 61724, 1998.

[2] G. Blaesser *et al.*, *Guidelines for the Assessment of Photovoltaic Plants*. Luxembourg City, Luxembourg: Office Official Publ. Eur. Communities, 1995.

[3] S. Wilcox *et al.*, *Users Manual for TMY3 Data Sets*. Golden, CO, USA: Nat. Renew. Energy Lab., 2008.

[4] M. Šúri *et al.*, "SolarGIS: Solar data and online applications for PV planning and performance assessment," in *Proc. 26th Eur. Photovoltaics Sol. Energy Conf.*, 2011, pp. 3930–3934.

[5] A. Woyte *et al.*, "Monitoring of photovoltaic systems: Good practices and systematic analysis," in *Proc. 28th Eur. Photovoltaic Sol. Energy Conf.*, 2013, pp. 3686–3694.

[6] R. H. French *et al.*, "Degradation science: Mesoscopic evolution and temporal analytics of photovoltaic energy materials," *Current Opin. Solid State Mater. Sci.*, vol. 19, pp. 212–226, 2015.

[7] L. S. Bruckman *et al.*, "Statistical and domain analytics applied to PV module lifetime and degradation science," *IEEE Access*, vol. 1, pp. 384–403, 2013.

[8] D. Kraus *et al.*, "Automatic spectral database and archive system for optical spectroscopy," *Appl. Spectrosc.*, vol. 44, no. 7, pp. 1221–1226, 1990.

[9] Y. Hu *et al.*, "Global SunFarm data acquisition network, energy cradle, and time series analysis," in *Proc. IEEE Energytech*, 2013, pp. 1–5.

[10] "Open PV project," Aug. 2011. [Online]. Available: https://openpv.nrel.gov

[11] "Photovoltaic data acquisition," Aug. 2010. [Online]. Available: http://maps.nrel.gov/pvdaq

[12] T. Nordmann *et al.*, "Analysis of long-term performance of PV systems," International Energy Agency, 2015.

[13] "RTC about the US DOE regional test centers," [Online]. Available: https://rtc.sandia.gov/about-pvrtc/. Accessed on: Feb. 20, 2015.

[14] N. Leavitt, "Will nosql databases live up to their promise?" *Computer*, vol. 43, no. 2, pp. 12–14, 2010.

[15] J. Hemminger, G. Crabtree, and J. Sarrao, "From quanta to the continuum: Opportunities for mesoscale science," Tech. Rep. 1183982, U.S. Dept. of Energy Basic Energy Sciences Advisory Committee, Sep. 2012.

[16] J. C. Hemminger, "Challenges at the frontiers of matter and energy: Transformative opportunities for discovery science," U.S. Dept. Energy Office Sci., Washington, DC, USA. Tech. Rep., Nov. 2015. [Online]. Available: http://science.energy.Gov/∼/media/bes/besac/pdf/Reports/Challenges_at_the_Frontiers_of_Matter_and_Energy_rpt.pdf. Accessed on: Nov. 22, 2016.

[17] K. Grolinger, *et al.*, "Data management in cloud environments: NoSQL and NewSQL data stores," *J. Cloud Comput.: Adv., Syst. Appl.*, vol. 2, no. 1, 2013, Art. no. 22.

[18] Y. Zhang *et al.*, "A Hadoop-based massive molecular data storage solution for virtual screening," in *Proc. IEEE 7th China Grid Annu. Conf.*, 2012, pp. 142–147.

[19] S. Wang, M. A. Mares, and Y.-K. Guo. "CGDM: Collaborative genomic data model for molecular profiling data using NoSQL." *Bioinformatics*, 2016, btw531.S..

[20] P. Ameri *et al.*, "On the application and performance of MongoDB for climate satellite data," in *Proc. IEEE 13th Int. Conf. Trust, Security Privacy Comput. Commun.*, 2014, pp. 652–659.

[21] B. Obama, "Executive Order—Creating a National Strategic Computing Initiative," Jul. 2015. [Online]. Available: https://www.whitehouse.gov/the-press-office/2015/07/29/executive-order-creating-national-strategic-computing-initiative

[22] T. J. Peshek *et al.*, "Insights into metastability of photovoltaic materials at the mesoscale through massive I–V analytics," *J. Vac. Sci. Technol. B*, vol. 34, no. 5, 2016, Art. no. 050801.

[23] G. Zhang *et al.*, "Multi-modality, multi-resource, information integration environment," U.S. Patent 8 856 169 Oct. 7, 2014. [Online]. Available: https://www.google.com/patents/US8856169

[24] "Apache Hadoop," [Online]. Available: http://hadoop.apache.org/. Accessed Jan. 8, 2015.

[25] "Apache HBase," [Online]. Available: https://hbase.apache.org/. Accessed Jan. 8, 2015.

[26] J. Dean *et al.*, "MapReduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, 2008. [Online]. Available: http://dl.acm.org/citation.cfm?id=1327492

[27] S. Ghemawat *et al.*, "The Google file system," *ACM SIGOPS Oper. Syst. Rev.*, vol. 37, no. 5, pp. 29–43, 2003.

[28] F. Chang *et al.*, "Bigtable: A distributed storage system for structured data," *ACM Trans. Comput. Syst.*, vol. 26, no. 2, 2008, Art. no. 4.

[29] P. Hunt *et al.*, "ZooKeeper: Wait-free coordination for internet-scale systems," in *Proc. USENIX Annu. Tech. Conf.*, 2010, pp. 11–11.

[30] M. Slee *et al.*, "Thrift: Scalable cross-language services implementation," *Facebook White Paper*, vol. 5, no. 8, 2007.

[31] S. Sagiroglu *et al.*, "Big data: A review," in *Proc. IEEE Int. Conf. Collaboration Technol. Syst.*, 2013, pp. 42–47.

[32] D. Burns *et al.*, *Selenium 2 Testing Tools: Beginner's Guide*. Birmingham, U.K.: Packt, 2012.

[33] "Python Software Foundation, Python Language Reference, version 2.6.6." 2010. [Online]. Available: http://www.python.org

[34] J. S. Fada *et al.*, "Democratizing an electroluminescence imaging apparatus and analytics project for widespread data acquisition in photovoltaic materials," *Rev. Sci. Instrum.*, vol. 87, no. 8, 2016, Art. no. 085109.

[35] M. Adhikari *et al.*, "NoSQL databases," in *Handbook of Research on Securing Cloud-Based Databases with Biometric Applications*. Hershey, PA, USA: IGI Global, 2014, p. 109.

[36] "Open source python project Happybase," [Online]. Available: http://happybase.readthedocs.io/en/stable/index.html. Accessed Feb. 2, 2016.

[37] W. Barth, *Nagios: System and Network Monitoring*. San Francisco, CA, USA: No Starch, 2008.

[38] Y. Hu *et al.*, "Detecting heterogeneity in pv modules from massive real-world "step" I-V curves: A machine learning approach," in *Proc. 42th IEEE Photovoltaic Spec. Conf.*, 2016, pp. 2079–2084.

[39] F. Rubel *et al.*, "Observed and projected climate shifts 19012100 depicted by world maps of the Köppen-Geiger climate classification," *Meteorologische Zeitschrift*, vol. 19, no. 2, pp. 135–141, Apr. 2010.

[40] Y. Hu *et al.*, "Comparison of multi-crystalline silicon PV modules' performance under augmented solar irradiation," *MRS Proc.*, vol. 1493, pp. 3–9, 2013.

[41] M. A. Hossain *et al.*, "Microinverter thermal performance in the real-world: Measurements and modeling," *PloS One*, vol. 10, no. 7, 2015, Art. no. e0131279.

[42] R Core Team, *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Found. Statist. Comput., 2016. [Online]. Available: https://www.R-project.org/

Authors' photographs and biographies not available at the time of publication.