Australian National University

Department of Engineering

A thesis submitted in part fulfilment of the degree of

Bachelor of Engineering

Automated PV Data Extraction from the Web

By: Ahmad Al-Kurdi

ID: u4887713

Supervisor: Associate Lecturer Nicholas Engerer

This thesis contains no material which has been accepted for the award of any other degree or diploma in any university. To the best of the author's knowledge, it contains no material previously published or written by another person, except where due reference is made in text.

Ahmad Al-Kurdi

2 June 2015

This thesis is dedicated to my family – my Dad, Dr. Mahmoud who has always been a symbol of success and strength in my life, my Mum, Mariam, who has endured and gave everything to me and my brothers that I will never be able to repay, and my brothers, Laith, who is the smartest person in the family and always makes everything looks easy, Luai, who is the chillest coolest 3-D architect out there, and Haitham, the most reliable and genuine person ever. Thank you all.

Acknowledgements

This thesis could not have been possible without the following people:

Nicholas Engerer - Associate Lecturer, Fenner School of Environment and Society, ANU

Thank you for proposing such an important engineering project, and the belief in my abilities and the continuous support that you have provided throughout the year. You pushed me and gave me motivation to work hard. I honestly could not have asked for a better and cooler supervisor. This project would not have been successful without your contribution. I wish to get the chance to work with you again in the near future.

Nate Wiley – Web Developer, Michigan, USA

Thank you for inspiring me to use PHP to accomplish this project. Your contributions to PHP and Xpath queries have been very useful to programmers.

Prof. Fatih Porikli & Dr. Mehrtash Harandi – ENGN4200 Course Coordinators, Research School of Engineering, ANU

Thank you both for your efforts coordinating the honours Individual Project course and being very helpful with students throughout the project's duration.

Abstract

The purpose of this thesis is to develop a system that allows for the automatic extraction of data about photovoltaic systems installed in Australia from “pvoutput.org. The primary aim underlying the development of this system was to create an extensive database in order to be used for research purposes in the photovoltaics field. Currently, data that characterize photovoltaic systems on the web is not available for rapid high-frequency extraction nor is it available in an efficient database-like format that allows for meaningful statistical analysis and visualization. The outcome of the software simulation is the compilation of an efficient and extensive Excel database. The database is to be used for research purposes in the photovoltaics field. A secondary aim of this project is to provide a framework for the development of a user-customizable automated data extraction system. A future goal is to include several more web sources in the data extraction process, so as to include as many photovoltaic systems in Australia reporting to such web sources as possible. Moreover, a secondary future goal is to develop a user-interface for the software, such that a user can specify the target state or territory for the data extraction as well as the number of systems to be targeted etc… The data extracted is also to be used in future research projects to analyse photovoltaic systems installed in Australia.

Table of Contents

Table of Figures

List of Tables

Glossary of Terms

Abbreviation	Description
PV	Photovoltaic
AC	Alternating Current
DC	Direct Current
ACT	Australian Capital Territory
NSW	New South Wales
VIC	Victoria
QLD	Queensland
SA	South Australia
WA	Western Australia
TAS	Tasmania
NT	Northern Territory
CBD	Central Business District
W	Watts
kW	Kilo-Watts
MW	Mega-Watts
kWh	Kilo-watt Hour
API	Application Program Interface
PHP	PHP Hypertext Pre-processor
XML	Extensible Markup Language
KPV	Clear-Sky Index
HTML	Hyper-Text Markup Language
CGI	Common Gateway Interface
DOM	Document Object Model
URL	Uniform Resource Locater
SID	Site Identification
REGEX	Regular Expression
WAMP	“Windows, Apache, MySQL and PHP”

This page intentionally left blank

Chapter 1 Introduction

1.2 Background & Introduction

Large amounts of data about photovoltaic (PV) systems in Australia is available on the worldwide-web. Installers as well as owners use monitoring products (e.g. 3G PV inverter data loggers) to provide real-time monitoring on these systems. These monitoring devices are connected to the inverters by Bluetooth, serial, WiFi or ethernet connections and regularly transmit data (such as PV module AC/DC power, open circuit voltage and short circuit current) to an on-site router or switch which in turn sends the data to an online database. This data can then be accessed from a computer where the user can inspect and analyze it.

Such an online database is "pvoutput.org". This database has recorded more than 8.1 million measurements from 794,471 panels (1) up until the 6th of February 2015. Within Australia, data for more than 300 systems in the ACT, 1000 systems in NSW and 500 systems in VIC is continuously being recorded. This data includes two types: first is the PV system power output, or so-called 'performance data'. The second is the 'metadata' or data about the given PV system. This includes the angle of module tilt, latitude of system, longitude of system, inverter size incorporated within the module, manufacturer…etc.

The monitoring of such data is essential, as it can improve the operation and reliability as well as the energetic and economic yield of PV power systems. For example, over the last 20 years, the statistical average performance ratio of new photovoltaic installation has improved from 0.65 to approximately 0.85 (2). This continuous improvement in the field would not have been possible without operational monitoring and the continued analysis of the monitored data. The data obtained is valuable as it can be used for different research purposes. For example, the actual electricity produced by the PV system can be compared to the estimated power generation output expected prior to the system installation. The continuous monitoring can be used to follow up on the energy yield, to evaluate the PV system performance and to timely identify design flaws or malfunctions (2). The data can also be

used to model and visualize PV systems in Australia. For example, a map showing the live kWh output for every state in Australia as well as how much power PV systems are providing that state with can be modelled.

Figure 1: shows an example of how data gathered about PV systems installed around the world is used in research. The visualization shows the monitored Gigawatts added in 2012 by different vendors vs. the average size of new sites monitored in 2012.

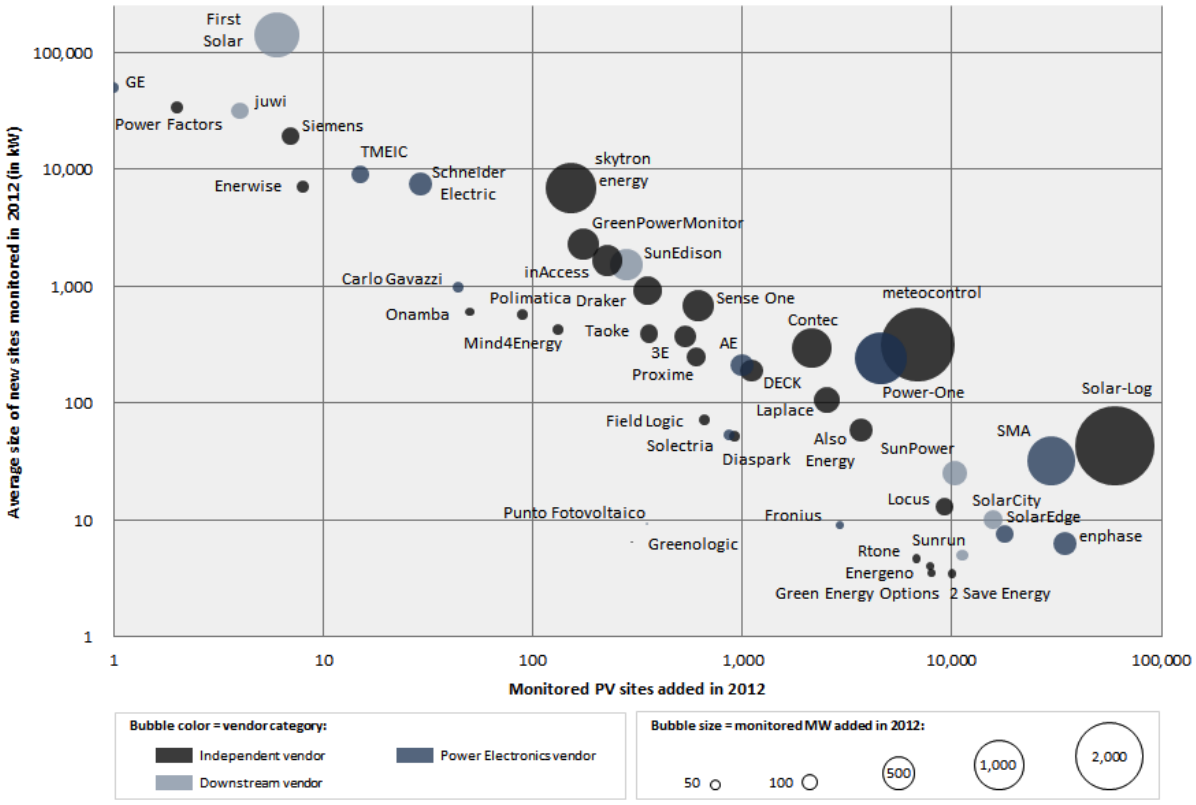


Figure 1: Global PV Monitoring Competitive Landscape (3)

Currently, the data on the web is not available for rapid high frequency extraction nor is it available in an efficient database-like format that allows for meaningful statistical analysis and visualization. Furthermore, the application programming interface (API) for "pvoutput.org" has a limit of 60 requests per hour for downloading data. With a request of several thousands of datasets, this process could take up to several hours and is not an efficient way of collecting data.

The unavailability of such data as well as the difficulty of collecting them manually, limits researchers and hinders the progress of research being done on PV systems in Australia. Given the limitations on the availability of such data and the lack of a source that provides this data in an efficient structure. The following thesis aims to design and develop a software system that will enable users to perform an automated extraction of data from the website "pvoutput.org" and store it in a database-like structure to be used for research and statistical analysis purposes on PV systems within Australia.

Automating the tedious process of collecting data such that a software accumulates this large database; allows for extracting data at a higher rate and opens the door for researchers to perform advanced analysis and provides them with a large sea of data to look into.

The system was created to allow the user to extract data from the website "pvoutput.org" with a high level of customization. This includes the extractions of two types of data: first is the PV system power output, or so-called 'performance data'. The second is the 'metadata' or data about the given PV system. This includes the angle of module tilt, latitude of system, longitude of system, inverter size incorporated within the module, manufacturer…etc.

The software can be regularly ran and will be used in the future to provide continuous monitoring of PV systems in Australia. It will be also used for in-depth research on the power output and performance of PV systems and to perform statistical analysis on such systems.

The programming language used to develop the software system is PHP. The web-page that needs to be crawled is specified, then different objects are accessed using either their XML description or their XPath location. Data from several "pvoutput.org" webpages are accessed and aggregated then stored in an XLSX (Excel) database. Recursion is used to obtain data from all available pages until no entries are found.

The development process incorporated the use of error detection, where the errors returned were displayed on the output screen. This is used to help troubleshoot errors during the

development stage as well as in its maintenance stage in the future. Testing was undertaken over the entire period of the project. The code ran on different platforms to verify whether it runs smoothly and to detect any incompatibilities. The code also ran on different webpages on "pvoutput.org" representing several states in Australia to verify if it can perform the same task regardless of the "pvoutput.org" web-address introduced.

The thesis is divided into five chapters. Chapter 2 provides an in-depth literature review of how PV data is used in research. Followed by an introduction to APIs and how a web-crawler is used to extract information from the web. Finally, a discussion of how software automation techniques are coupled with web-crawlers to obtain automated data extraction is introduced. Chapter 3 discusses the algorithm created and how it works, with a mention of the major challenges faced as well as solutions implemented during its construction. Chapter 4 presents the data gathered as a result of running the software and displays the usefulness of the data by providing statistical analysis of the data. Finally, chapter 5 provides a conclusion to the thesis highlighting the importance of the work at hand and discussing future recommendations.

1.2 Research Question & Scope

The research question addressed in this paper is: how can we create an automation software for the extraction of PV systems data from the web?

The scope of the project is to develop a software that is capable of automatically extracting performance data as well as metadata for PV systems reporting to "pvoutput.org". The systems that will be targeted are those installed in the six Australian states (NSW, VIC, QLD, SA, WA and TAS) as well as the two territories (ACT and NT).

In a later stage of the project, a basic analysis on the data obtained will be performed to show its usefulness. Due to the time restrictions governing the ENGN4200 course, this research paper will not go into an advanced analysis of the data.

Chapter 2 Literature Review

2.1 The Use of Photovoltaic Data in Research

There are a few instances of the use of PV system databases in the literature. The KPV index formulation methodology of Engerer and Mills (4) in which a clear-sky index was used to predict the performance of an unknown PV system using the data from a PV system that is nearby. Data was collected for systems in Canberra, Australian Capital Territory, from "pvoutput.org". Five of the highest quality stations were hand-selected from the available systems in Canberra to undertake this study. The systems reported data from early 2011 through 2012, providing data such as the module rating, make and number of modules and inverters, as well as the estimated tilt, orientation and shade conditions. The data obtained manually from this website was used to calculate the transposed clear-sky radiation available to the PV system, and eventually for the simulation of the PV system given the available radiation resource.

Another example was the usage of large amounts of test-beds of PV systems to perform analysis in Austin, Texas. The main objective of this research project was to describe the experimental and data collection methods for a large-scale smart grid deployment (5). As a secondary goal, the project aimed to provide results based on the collected data. The test-bed comprised of 250 homes concentrated in a single neighbourhood at 15-second resolution, and 160 homes distributed throughout Austin with device ages ranging from 10 to 92 years old. Direct measurements include quantitative data that has been collected on-site with installed system monitors as well as data pulled from other sources. Similarly, the Pal town neighbourhood of Ota city in Japan, was used as a unique test-bed of high penetration PV deployment with a total of 553 rooftop installations in the neighbourhood (approximately 80% of the homes). Each site was collecting PV power output data once per second in 2006 and 2007. The analysis of the collected data allowed for detailed characterization of PV output variability and how it can be enhanced (6).

Cameron et al (7) performed a comparison of performance-model calculations to actual measured PV system performance in order to evaluate the ability of models to accurately predict PV system energy production. The models used measured meteorological and irradiance data as input. The data was collected at two-minute intervals then averaged over one-hour intervals. Three grid-tied PV Systems consisting of 24 PV modules were installed and operated for a year, output data was continuously being logged. To calculate the system output, the solar radiation incident on the module, the module DC output and the inverter AC output were modelled. The model outputs were then compared to actual measured data. The results showed that module performance models, including radiation model errors fell within 4-11% of actual measured data. However, modelling of modules using non-crystalline technologies showed significant disagreement between models.

The researchers needed to install the PV system and operate it for a year to obtain a data-set that can be used for the research paper. Moreover, the number of modules used to undertake this research was limited to 24 due to the high cost of installing a large number of modules as well as the unavailability of large data sets publicly to use in research.

Providing access to system output data as well as the metadata for a large number of PV systems would allow researchers to access and use this information without spending time or money to obtain a data-set of their own.

2.2 Application Program Interface (API)

A service that allows the gathering of "pvoutput.org" data exists in the form of an application program interface (API) dedicated to pvoutput.org. An API is a set of programming rules and protocols for accessing a web-based application or service. Companies release APIs so that web developers could access data on their websites in a controlled and structured way. For example, 'Amazon' allows users to post direct links to Amazon products with updated prices and "buy now" option using their API. 'Twitter' allows developers to access core Twitter data as well as interact with Twitter search and trends data using their dedicated API (8). The dedicated API for pvoutput.org allows for a registered user to send and receive

"pvoutput.org" data without using the web user interface. The most common use of the API is to automate the live energy output for a PV system every 5 to 15 minutes (9). The API has a limit of 60 requests per hour with a recommended 10 seconds time period between each request. However, the API only allows access to the user's system. Thus, the user can't access the data for other PV systems in the database. In order to get a large dataset, for example 5000 data points, this will take the user 83.33 hours with access only to the user's system. This is very limited and inefficient. This is why the software developed in this paper is important. It can access data for all the systems on "pvoutput.org" and can accomplish that in a significantly less time period.

2.3 Web Crawling

A web crawler is a software system, which systematically finds and retrieves web pages from the web documents (10). Web crawlers are used for a variety of purposes. Most notably, they are the main components of web search engines; such as Google, Yahoo and Bing. The web crawler takes the user's query and "crawls" the entire web for pages relevant to it, then presents them as indexed search results for the user to access. Another use is web archiving, where large sets of web pages are periodically collected and archived for posterity. A third use is web data mining, where web pages are analysed for statistical properties, or where data analytics is performed on them (such as Attributor, a company that monitors the web for copyright and trademark infringements) (11). Finally, a user-customized web crawler can be used to scrape information from web pages and store the findings for a user-defined purpose (such as display the results or save them in an excel database). The system that will be developed in this work can be considered as a user customized web crawler.

2.4 Software Automation

Software automation is the use of a software program to perform a series of actions. It is also behind a large amount of web applications. For example, counting how many visitors a website has had at the end of the day. Or, updating the Facebook 'feed' whenever a new post has been written. Software automation coupled with the use of web crawlers allow the user to crawl several web-pages without the need to specify which destination the crawler should investigate after each task is completed. This concept is very important for the functionality of the software constructed in this paper. As the software system will be run with one click, and the results will be a consequence of multiple web pages being "crawled" as data will be gathered and stored in a database.

APVI (12), has developed a software that extracts data from "pvoutput.org". However, this data is inaccessible and cannot be downloaded. Moreover, there are several similar software that extract information from some popular websites or webpages. "import.io" allows the user to extract data from different webpages. For example, it could extract a 'Twitter' user's profile including all the posts on the profile page, including 'tweets', 'retweets' and favourites. Another web scraper is "webscraper.io" which offers an internet browser extension that operates similarly to import.io. These software scrape single pages and do not allow the user to scrape from several webpages of the same website at once and collate the findings in a single database. The user has to manually select the webpage that contains the data needed to be scraped at each instance. Also, they do not offer the ability to scrape data embedded in the HTML of the webpage. The software developed in this paper addresses all these issues and allows for a user-customizable extraction from several webpages at once.

Chapter 3 Analysis and Requirements

3.1 Introduction to Protocols Used

3.1.1 PHP

PHP (PHP: Hypertext Preprocessor) is a scripting language generally used for web development and HTML design (13). The PHP script can be embedded in the HTML code, which allows the user to create dynamic and interactive web pages. PHP is also a general purpose programming language and can be used for several other purposes such as command line scripting and application development. PHP code is usually interpreted by a web-server or a Common Gateway Interface (CGI). After the PHP code is interpreted, the web-server sends the output to its client in the form of a web page, graphical interface or other kinds of data depending on the code.

PHP employs several useful extensions, each serving a specific application purpose. The 'DOM' extension is vital for the operation of the algorithm explained in this paper. It contains several classes, two of which are used extensively in the algorithm, the DOMDocument class (14) and the DOMXpath class (15). The use of these classes in PHP is explained further in the following sections.

3.1.2 DOMDocument Class

When an HTML document is loaded into a browser, it becomes a document object. The HTML document object is represented by several nodes. These nodes are used to represent elements, attributes as well as text content of the HTML page.

DOM or Document Object Model, is an interface that allows programs and codes to access and update the content of HTML document objects. DOM is used as a convention for interacting with and representing HTML objects (16). The DOMDocument class, used in PHP, treats the webpage targeted as a DOM, therefore allowing access to its different node contents.

Figure 2 shows a typical HTML DOM tree. Each of the rectangular boxes represents a node. There are several types of nodes shown, namely: 'Element', 'Attribute' and 'Text' nodes.

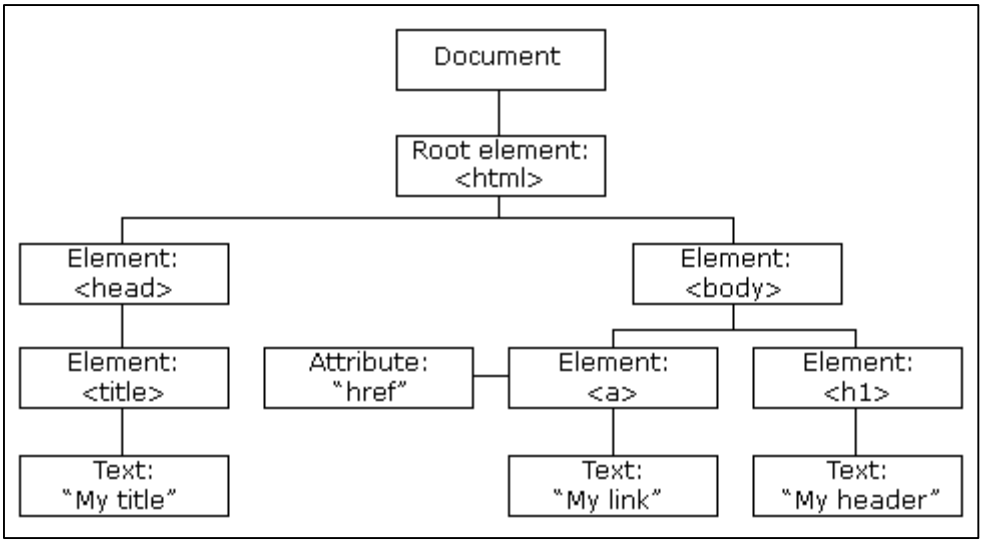


Figure 2: HTML DOM tree (17).

Several instances are used by the DOMDocument class to access the nodes, depending on the type of node and its contents. The algorithm utilizes the 'getElementsByTagName' instance to access content from 'Element' nodes depending on the tag-name ("<a>" or "<h1>") for example. Additionally, the 'getAttribute' instance is used to access content of 'Attribute' nodes. The DOMDocument class cannot access all the content of the HTML page. For instance, it cannot access HTML content that uses 'Class' tag-names, nor HTML content that is located outside nodes. Some data that needs to be extracted from the "pvoutput.org" HTML webpages includes this inaccessible content, thus, an alternative approach using the DOMXpath class was exploited.

3.1.3 DOMXpath Class

A webpage can be represented by an HTML code. The HTML code defines the type and location of the content and how it will be displayed on the webpage. The code snippet below shows an example of an HTML code, showing the head and the body of a webpage. The content of the tag-names (<h1>, , <h2> and <p>) each have a path location that describes them, this location can be represented as an XPath. An XPath is the destination path that describes the location of content within HTML and XML documents. This allows

the use of the XPath instance of the DOMXPath class to perform queries to access content of the HTML code. DOMXPath is a class used in PHP that takes a DOMDocument as an input and performs XPath queries on it using the path location described earlier.

```
<html>
	<head>
		<title> Thesis Example </title>
	</head>
	<body>
		<h1 class="Chapter 3"> How the Code Works </h1>
		<div> class= "Introduction" >
			<p> A very short intro to the algorithm </p>
		</div>
		<h2> class="Chapter 4"> Results </h2>
		<p> This demonstrates the results </p>
	</body>
</html>
```

The algorithm discussed in this paper uses XPath queries to access content that cannot be accessed using DOMDocument instances. An example is the content nested within a 'Class' tag-name used in the HTML code.

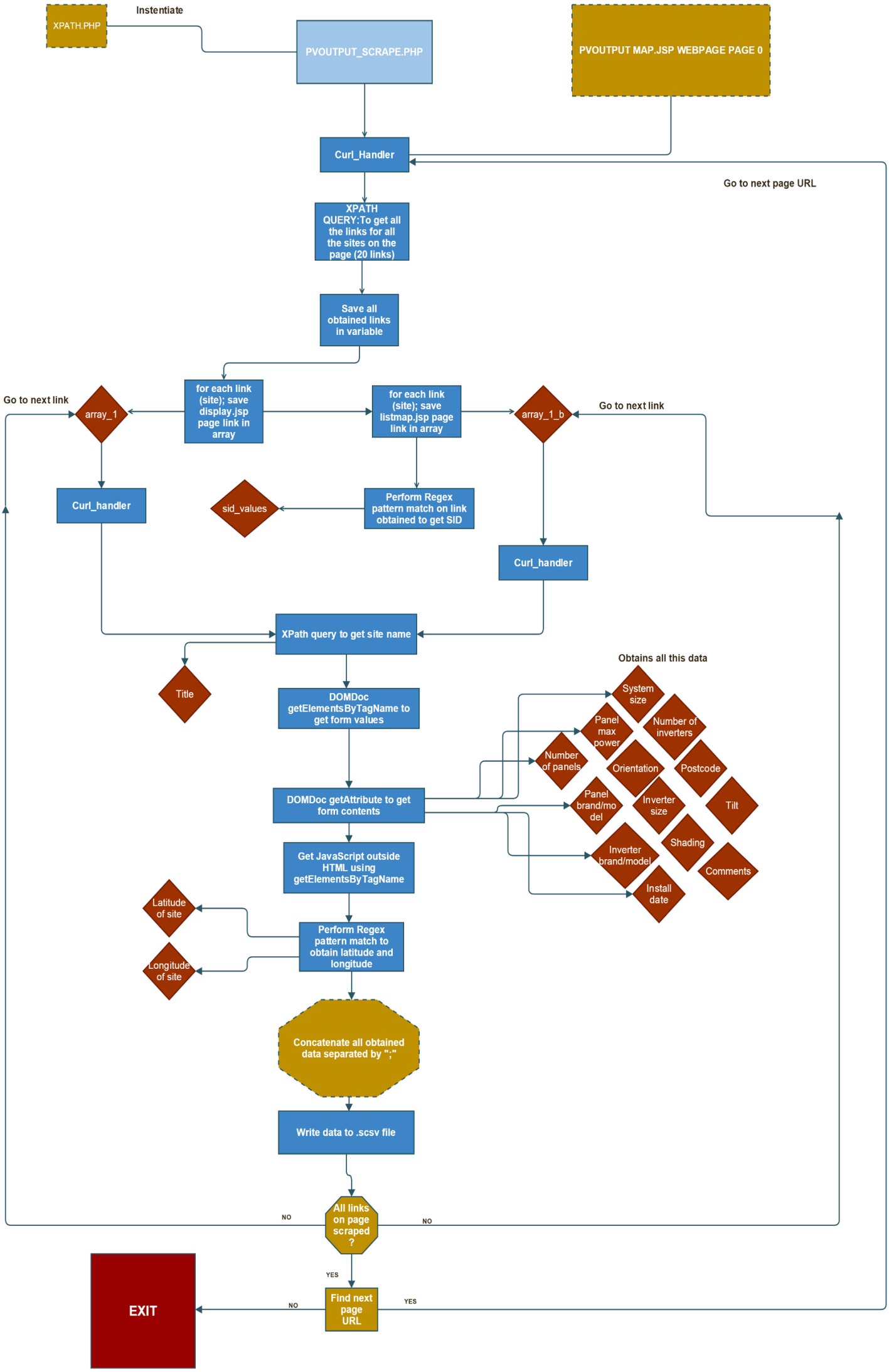
3.1.4 Regex

Regex (Regular Expression) is a syntax that describes a pattern of characters. Regex syntax is used to perform search queries in PHP using the 'preg_match' function. It is used to perform pattern matching and find-and-replace actions on the HTML document. Some content on "pvoutput.org" cannot be retrieved using the aforementioned DOMDocument nor DOMXPath instances, as this content exists outside both the HTML DOM nodes and the HTML tag-name code. Therefore, Regex was used to access this content. The pattern matching occurs when a user-defined Regex capturing group is found within the HTML webpage. A number of characters can be captured and stored in a variable.

3.2 Overall System Architecture

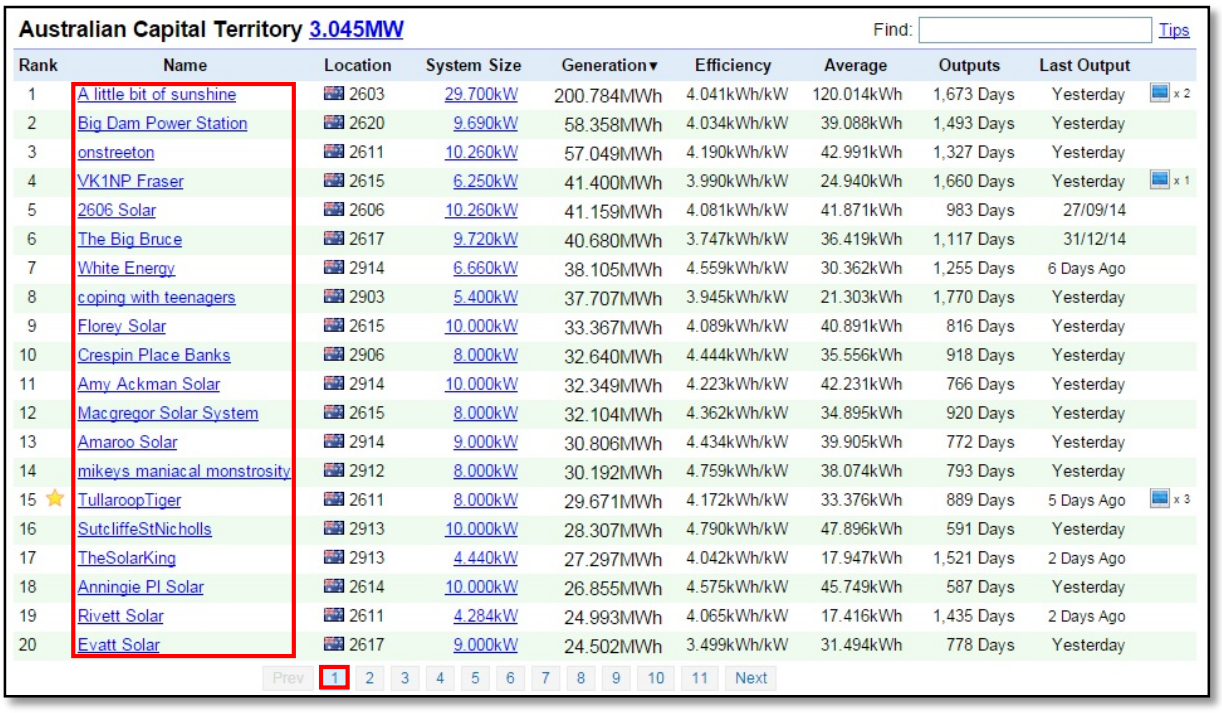
3.2.1 System Architecture Diagram

Figure 3 below, shows the scraping algorithm system architecture. Fundamentals of how the system operates is explained in detail in section 3.2.2.


XPATH.PHP
Instentiate
PVOUTPUT_SCRAPE.PHP
PVOUTPUT MAP.JSP WEBPAGE PAGE 0
Curl_Handler
Go to next page URL
XPATH QUERY:To get all the links for all the sites on the page (20 links)
Save all obtained links in variable
for each link (site); save display.jsp page link in array
for each link (site); save listmap.jsp page link in array
Go to next link
array_1
array_1_b
Go to next link
Curl_handler
sid_values
Perform Regex pattern match on link obtained to get SID
Curl_handler
XPath query to get site name
Title
DOMDoc getElementsByTagName to get form values
Obtains all this data
System size
Number of inverters
Panel max power
Number of panels
Orientation
Postcode
Panel brand/model
Inverter size
Tilt
Inverter brand/model
Shading
Comments
Install date
DOMDoc getAttribute to get form contents
Get JavaScript outside HTML using getElementsByTagName
Latitude of site
Longitude of site
Perform Regex pattern match to obtain latitude and longitude
Concatenate all obtained data separated by ";"
Write data to .scsv file
All links on page scraped ?
NO
NO
YES
Find next page URL
NO
YES
EXIT

3.2.2 PVOutput.org Webpage HTML Structure

The properties of the webpages accessed to extract the information needed to build the intended database have to be investigated before the construction of the algorithm. Figure 4 below shows the content of the webpage *"pvoutput.org/map.jsp?p=0&state=ACT"*. The webpage contains the names as well as other power generation information of PV sites in ACT, Australia. The valuable information that need to be extracted from this page is the URL address of each of the 20 sites listed, the 'SID', or site identification number, of each site, as well as the next-page URL address (pagination). The URL address of each of the sites and the 'SID' are contained in the HTML code of the hyperlink of the names of the sites (highlighted in red on Figure 4). The page-URL address of the next webpage to be extracted is contained in the HTML code of the hyperlink of the page numeration on the bottom of the webpage (highlighted in red at the bottom of the page in Figure 4).



Australian Capital Territory 3.045MW Find: Tips

Rank	Name	Location	System Size	Generation	Efficiency	Average	Outputs	Last Output	
1	A little bit of sunshine	2603	29.700kW	200.784MWh	4.041kWh/kW	120.014kWh	1,673 Days	Yesterday	x 2
2	Big Dam Power Station	2620	9.690kW	58.358MWh	4.034kWh/kW	39.088kWh	1,493 Days	Yesterday	
3	onstreeton	2611	10.260kW	57.049MWh	4.190kWh/kW	42.991kWh	1,327 Days	Yesterday	
4	VK1NP Fraser	2615	6.250kW	41.400MWh	3.990kWh/kW	24.940kWh	1,660 Days	Yesterday	x 1
5	2606 Solar	2606	10.260kW	41.159MWh	4.081kWh/kW	41.871kWh	983 Days	27/09/14	
6	The Big Bruce	2617	9.720kW	40.680MWh	3.747kWh/kW	36.419kWh	1,117 Days	31/12/14	
7	White Energy	2914	6.660kW	38.105MWh	4.559kWh/kW	30.362kWh	1,255 Days	6 Days Ago	
8	coping with teenagers	2903	5.400kW	37.707MWh	3.945kWh/kW	21.303kWh	1,770 Days	Yesterday	
9	Florey Solar	2615	10.000kW	33.367MWh	4.089kWh/kW	40.891kWh	816 Days	Yesterday	
10	Crespin Place Banks	2906	8.000kW	32.640MWh	4.444kWh/kW	35.556kWh	918 Days	Yesterday	
11	Amy Ackman Solar	2914	10.000kW	32.349MWh	4.223kWh/kW	42.231kWh	766 Days	Yesterday	
12	Macgregor Solar System	2615	8.000kW	32.104MWh	4.362kWh/kW	34.895kWh	920 Days	Yesterday	
13	Amaroo Solar	2914	9.000kW	30.806MWh	4.434kWh/kW	39.905kWh	772 Days	Yesterday	
14	mikeys maniacal monstrosity	2912	8.000kW	30.192MWh	4.759kWh/kW	38.074kWh	793 Days	Yesterday	
15	TullaroopTiger	2611	8.000kW	29.671MWh	4.172kWh/kW	33.376kWh	889 Days	5 Days Ago	x 3
16	SutcliffeStNicholls	2913	10.000kW	28.307MWh	4.790kWh/kW	47.896kWh	591 Days	Yesterday	
17	TheSolarKing	2913	4.440kW	27.297MWh	4.042kWh/kW	17.947kWh	1,521 Days	2 Days Ago	
18	Anningie Pl Solar	2614	10.000kW	26.855MWh	4.575kWh/kW	45.749kWh	587 Days	Yesterday	
19	Rivett Solar	2611	4.284kW	24.993MWh	4.065kWh/kW	17.416kWh	1,435 Days	2 Days Ago	
20	Evatt Solar	2617	9.000kW	24.502MWh	3.499kWh/kW	31.494kWh	778 Days	Yesterday	

Prev 1 2 3 4 5 6 7 8 9 10 11 Next

Figure 4: "pvoutput.org/map.jsp?p=0&state=ACT" webpage content. Highlighted in red is the data to be extracted.

Figure 5 shows the HTML content of the same webpage displayed in Figure 4. The HTML content of any webpage can be accessed and examined using internet browsers such as Google Chrome, Mozilla Firefox and Internet Explorer. The marked HTML line (in red) shows the URL address for the first site: "A little bit of sunshine". The XPath query function

discussed earlier is used to access this HTML content and save its 'href' value into an array. 'SID' values are extracted from the 'href' value using Regex and saved into a separate array of their own. This process is repeated for all the PV sites listed on the page until all the URL addresses as well as 'SID' values are saved in their relevant arrays.

Figure 5: HTML content of "pvoutput.org/map.jsp?p=0&state=ACT" webpage

The second webpage of interest is *"pvoutput.org/listmap.jsp?sid=312"*, shown in Figure 6. Using the 'SID' array obtained from the previous discussion, the URL address of this webpage and similarly webpages for other PV sites can be identified and accessed by simply adding the value of the 'SID' to the end of the following webpage address: "*pvoutput.org/listmap.jsp?sid=".* Figure 6 contains the map showing the location of "A little bit of sunshine" PV site. The map is generated using 'JavaScript' code shown in Figure 7. The longitude and latitude values can be extracted from this code (highlighted in red on Figure 7). Obtaining the longitude and latitude values is essential since they can be used for several applications such as calculating the solar irradiation values.

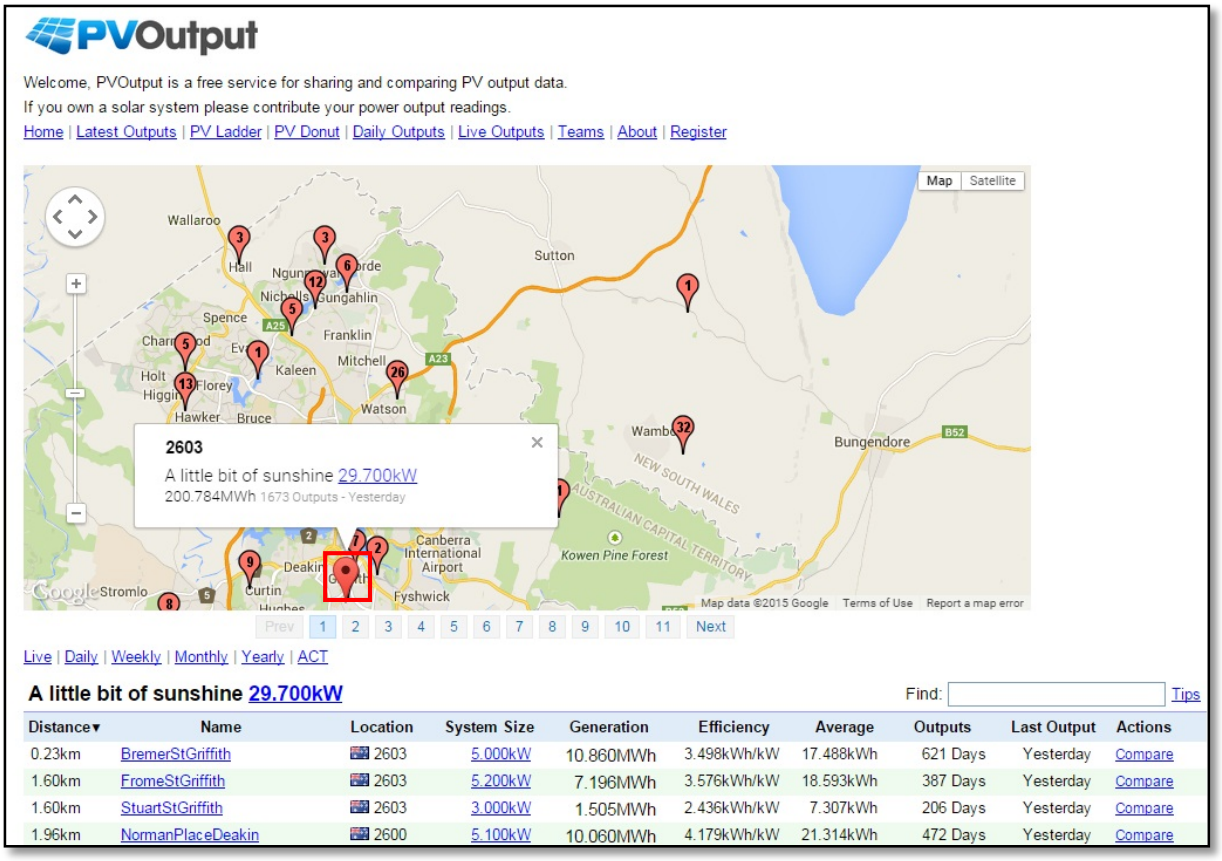


Figure 6: "pvoutput.org/listmap.jsp?sid=312" webpage content

The latitude and longitude values are extracted from the 'JavaScript' code using Regex. As the equivalent webpage for every PV site listed is accessed, the retrieved latitude and longitude values are saved in two separate arrays until the recursion through all the PV sites has finished.

Figure 7: Javascript content of "pvoutput.org/listmap.jsp?sid=312" webpage

Finally, the last page of interest is identified and accessed using the same process of using 'SID' values of each site discussed earlier. Figure 8 shows the webpage: *"pvoutput.org/display.jsp?sid=312".* The information highlighted in red is included in the 'JavaScript' code shown in the HTML code in Figure 9. The site name values as well as the form values are extracted and saved in separate arrays using DOMDocument instances.

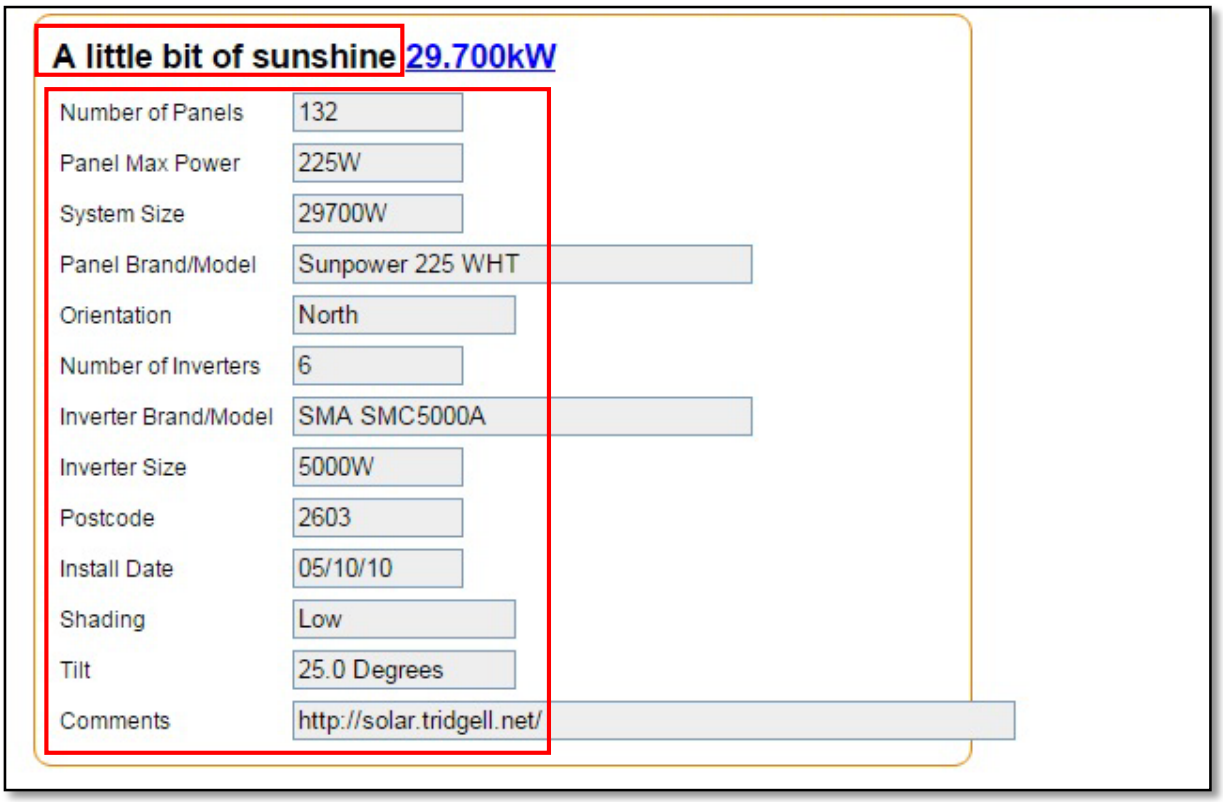


Figure 8: "pvoutput.org/display.jsp?sid=312" webpage content

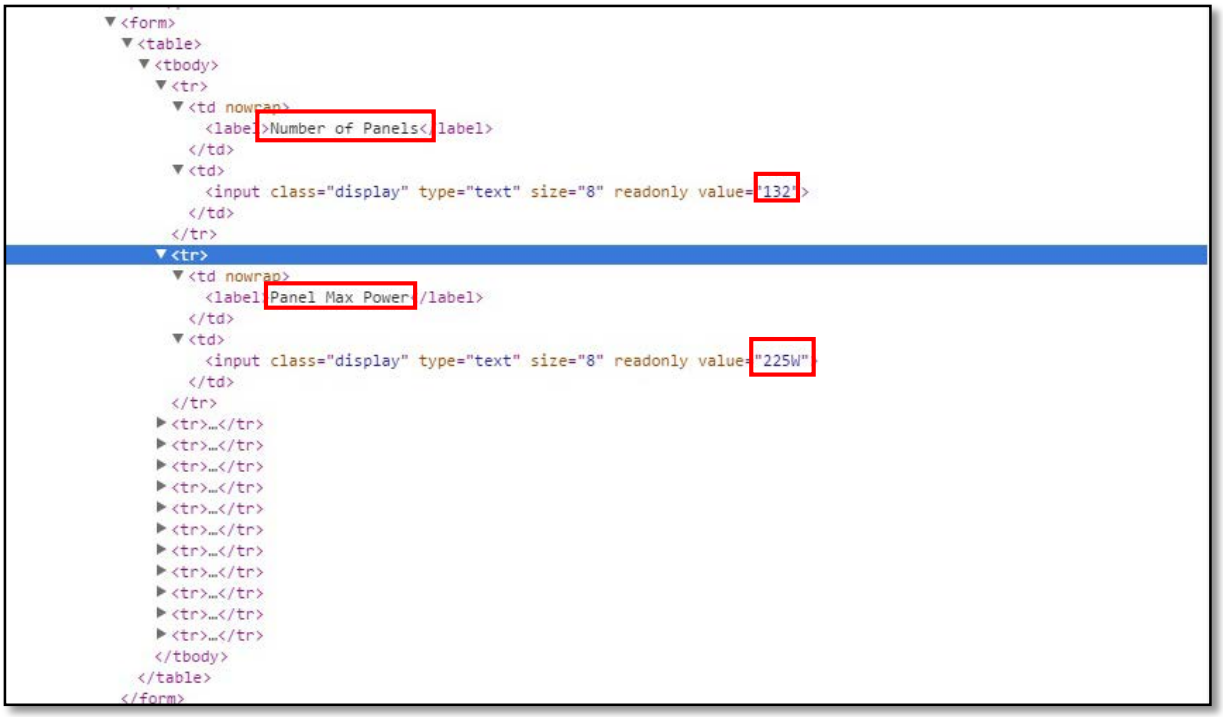


Figure 9: Javascript content of "pvoutput.org/display.jsp?sid=312" webpage

3.2.3 Fundamentals of System Algorithm Operation

There are three main approaches used in the development of the algorithm used to extract data from the desired webpages. First, the use of DOMDocument instances to obtain data from the HTML content of the webpage. DOMDocument instances allow the access to JavaScript read-only values, 'TagName' elements, as well as the retrieval of attributes. The instances used in the algorithm are: 'getElementsByTagName' and 'getAttribute'.

Second, the use of DOMXpath class and the XPath query function to access data stored inside 'class' tags, which are not accessible using the DOMDocument instances. Finally, the use of Regex to extract data that is neither displayed on the webpage nor is accessible using DOMDocument instances or XPath queries but is however included in the HTML content of the webpage, such as the latitudinal and longitudinal coordinates of a geographical location displayed on a map on a webpage.

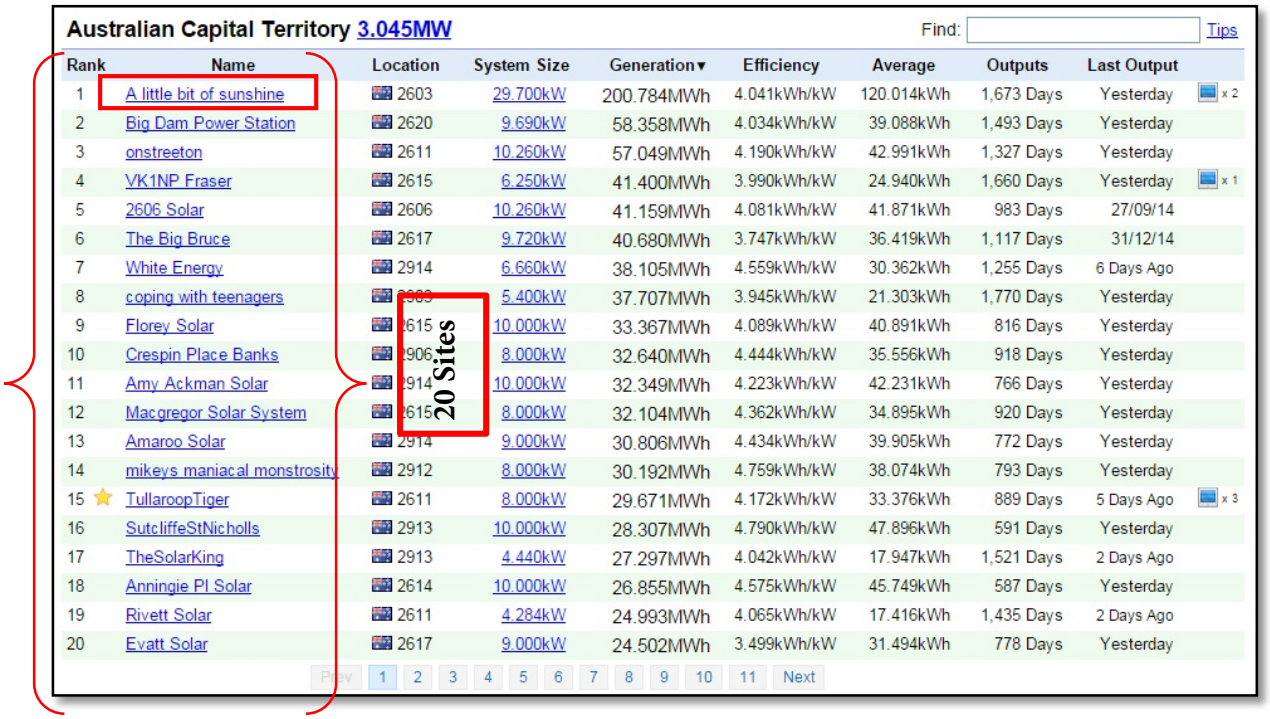
Refer to Figure 3, the system architecture overview, throughout this section to gain a visual understanding of the algorithm's operation. The algorithm begins by instantiating the

'XPath.php' class (line 2, Appendix A). The XPath class (Appendix B) was created by Nate Wiley (18) using PHP. This class allows the user to use XPath queries by instantiating the 'DOMXpath' function. Thus, whenever an XPath query is performed, it can be used without instantiating the 'DOMXpath' function at each time. The URL of the webpage to be scraped is specified and is fed to PHP's 'Curl-handler' as follows:

```
$ch = curl_init();

curl_setopt($ch, CURLOPT_URL, $url);
```

The initial webpage in this case study is taken as the first webpage (having a page value of p=0) displaying PV sites in ACT (state=ACT), Australia: *"pvoutput.org/map.jsp?p=0&state=ACT"*. However, PV sites in other states in Australia can be accessed and scraped using the target state's initials (VIC, NSW, WA, etc…) in the URL address, (state=**VIC**, state=**NSW**, state=**WA**, etc…). This webpage displays properties of 20 PV sites as shown in Figure 10.



Rank	Name	Location	System Size	Generation	Efficiency	Average	Outputs	Last Output
1	A little bit of sunshine	2603	29.700kW	200.784MWh	4.041kWh/kW	120.014kWh	1,673 Days	Yesterday
2	Big Dam Power Station	2620	9.690kW	58.358MWh	4.034kWh/kW	39.088kWh	1,493 Days	Yesterday
3	onstreeton	2611	10.260kW	57.049MWh	4.190kWh/kW	42.991kWh	1,327 Days	Yesterday
4	VK1NP Fraser	2615	6.250kW	41.400MWh	3.990kWh/kW	24.940kWh	1,660 Days	Yesterday
5	2606 Solar	2606	10.260kW	41.159MWh	4.081kWh/kW	41.871kWh	983 Days	27/09/14
6	The Big Bruce	2617	9.720kW	40.680MWh	3.747kWh/kW	36.419kWh	1,117 Days	31/12/14
7	White Energy	2914	6.660kW	38.105MWh	4.559kWh/kW	30.362kWh	1,255 Days	6 Days Ago
8	coping with teenagers	[illegible]	5.400kW	37.707MWh	3.945kWh/kW	21.303kWh	1,770 Days	Yesterday
9	Florey Solar	2615	10.000kW	33.367MWh	4.089kWh/kW	40.891kWh	816 Days	Yesterday
10	Crespin Place Banks	2906	8.000kW	32.640MWh	4.444kWh/kW	35.556kWh	918 Days	Yesterday
11	Amy Ackman Solar	2914	10.000kW	32.349MWh	4.223kWh/kW	42.231kWh	766 Days	Yesterday
12	Macgregor Solar System	2615	8.000kW	32.104MWh	4.362kWh/kW	34.895kWh	920 Days	Yesterday
13	Amaroo Solar	2914	9.000kW	30.806MWh	4.434kWh/kW	39.905kWh	772 Days	Yesterday
14	mikeys maniacal monstrosity	2912	8.000kW	30.192MWh	4.759kWh/kW	38.074kWh	793 Days	Yesterday
15	TullaroopTiger	2611	8.000kW	29.671MWh	4.172kWh/kW	33.376kWh	889 Days	5 Days Ago
16	SutcliffeStNicholls	2913	10.000kW	28.307MWh	4.790kWh/kW	47.896kWh	591 Days	Yesterday
17	TheSolarKing	2913	4.440kW	27.297MWh	4.042kWh/kW	17.947kWh	1,521 Days	2 Days Ago
18	Anningie Pl Solar	2614	10.000kW	26.855MWh	4.575kWh/kW	45.749kWh	587 Days	Yesterday
19	Rivett Solar	2611	4.284kW	24.993MWh	4.065kWh/kW	17.416kWh	1,435 Days	2 Days Ago
20	Evatt Solar	2617	9.000kW	24.502MWh	3.499kWh/kW	31.494kWh	778 Days	Yesterday

Prev 1 2 3 4 5 6 7 8 9 10 11 Next

Figure 10: "pvoutput.org/map.jsp?p=0&state=ACT" webpage content

The HTML content of this webpage is then accessed and stored in a variable. An XPath query is performed on this variable to extract the URL addresses for all the sites on the webpage and the results are saved in the "linkHrefQuery" array as follows:

```
$startUrl =   "http://pvoutput.org/map.jsp?p=0&state=ACT]";
$xpath = new XPATH($startUrl);
$linkHrefQuery = $xpath -> query("//tr/td[3]/a/@href");
```

The content of the first 4 elements in the array are shown in Table 1 below.

Table 1: Content of 'LinkHrefQuery' array

Array Index Number	'linkHrefQuery'	Content
0	linkHrefQuery[0]	pvoutput.org/listmap.jsp?sid=**312**
1	linkHrefQuery[1]	pvoutput.org/listmap.jsp?sid=**793**
2	linkHrefQuery[2]	pvoutput.org/listmap.jsp?sid=**2135**
3	linkHrefQuery[3]	pvoutput.org/listmap.jsp?sid=**407**

The 'SID' values for the different sites are extracted from the "linkHrefQuery" array using Regex and then stored in an array of their own "sid_values" as follows:

```
$SID = $linkHrefQuery ->item($x)->nodeValue;
$sid_regex= "/\bsid=\b(\S*)/";
preg_match($sid_regex, $SID, $sid_values);
```

The contents of the first 4 elements in the "sid_values" array is shown in Table 2 below.

Table 2: Content of 'sid_values' array

Array Index Number	'sid_values'	Content
0	sid_values[0]	**312**
1	sid_values[1]	**793**
2	sid_values[2]	**2135**
3	sid_values[3]	**407**

The 'SID' value for each PV site, representing the site identification number, is then used to build the next-target webpage URL link array "array_1_b". The next-target webpage for the first PV site is shown in Figure 11.

The 4 first contents of the "array_1_b" array are shown in Table 3 below.

Table 3: Content of 'array_1_b" array

Array Index Number	'array_1_b'	Content
0	array_1_b[0]	pvoutput.org/**display.jsp?**sid=312
1	array_1_b[1]	pvoutput.org/**display.jsp?**sid=793
2	array_1_b[2]	pvoutput.org/**display.jsp?**sid=2135
3	array_1_b[3]	pvoutput.org/**display.jsp?**sid=407

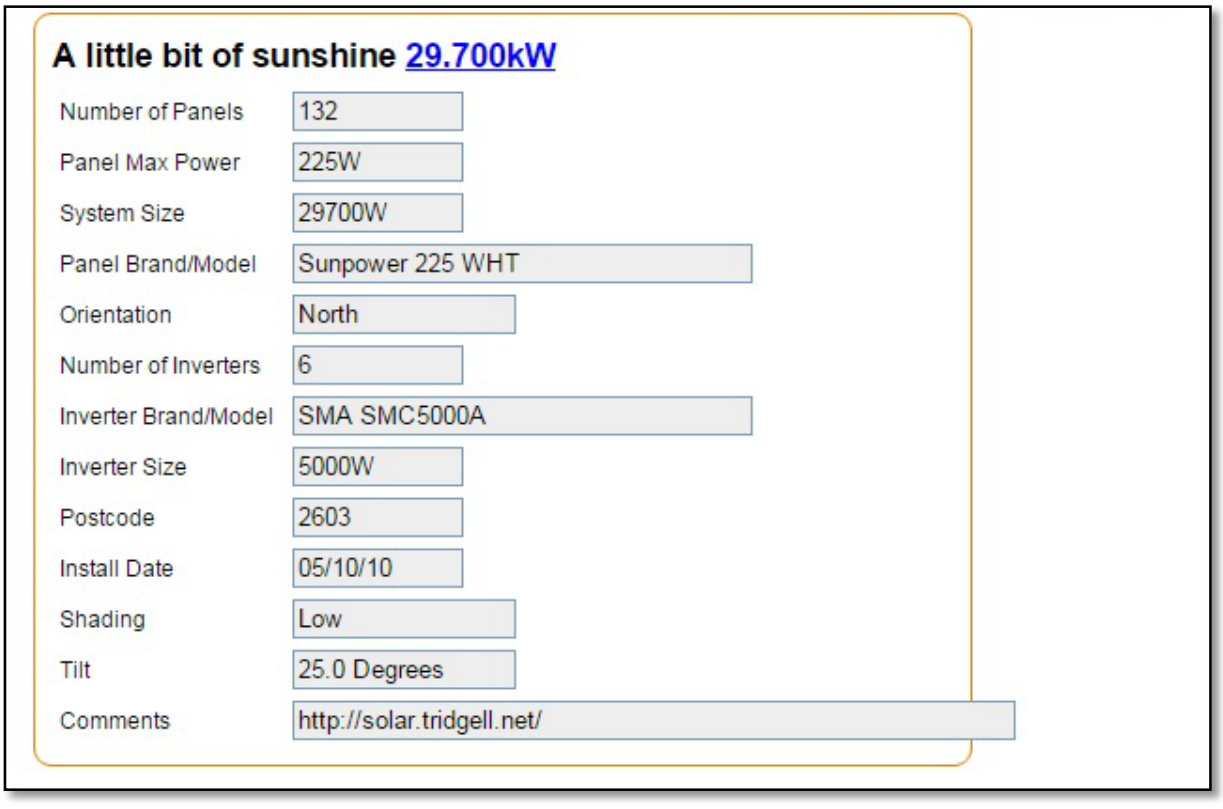


Figure 11: "pvoutput.org/display.jsp?sid=312" webpage content

The contents of the aforementioned arrays ('linkHrefQuery' and 'array_1_b') contain URL addresses for webpages that contain target data needed for the construction of the output database, thus need to be scraped. These URL addresses represent the site-specific webpages

for the 20 PV sites displayed on the first webpage (Figure 10) fed to the 'Curl-handler' in the first step.

The first URL address in the 'linkHrefQuery' array is fed to the 'Curl-handler' and its HTML content is saved into a variable. The latitude and longitude values are then extracted from this HTML content using DomDocument instances and Regex as follows:

```
$Lat_data = $dom_b -> getElementsByTagName("script");

$content =$Lat_data -> item(10) -> textContent;

$regex = "/latitude: (\S*)/";

preg_match( $regex , $content , $values );

$latitude = $values[1];   //this just gives the latitude value : "-35.33"

$regex2 = "/longitude: (\S*)/";

preg_match( $regex2 , $content , $values2 );

$longitude = $values2[1];
```

The HTML content of the first URL address in the 'array_1_b' array is then similarly accessed and the 'getAttribute' instance is used to store all the values of the 'JavaScript' form that exists on that webpage, and then saved in 'form_out' array.

```
$formVal = $dom -> getElementsByTagName("input");
foreach($formVal as $link) {
      $form_out .= $link->getAttribute("value") . ";";
      }
```

The result of this action is shown below. The first element in 'form_out' contains all the data scraped from the webpage targeted.

$form_out[0] = "**132** ; **225W** ; **29700W** ; **Sunpower 225 WHT** ; **North** ; **6** ; **SMA SMC5000A** ; **5000W** ; **2603** ; **05/10/10** ; **Low** ; **25.0 Degrees** ; **http://solar.tridgell.net/**"

The data was separated by semicolons instead of colons in the array because some of the data scraped contains commas within it. Thus, to avoid treating a single data entry as multiple when separated, semicolons are used.

The content of the element above represents the number of panels, panel maximum power, system size, panel brand, orientation, number of inverters, inverter brand, inverter size, postcode, install date, shading, tilt and comments. Finally, the data scraped from the different webpages discussed earlier are concatenated and stored in an array as displayed below:

$form_out_nl[0] = "**A little bit of sunshine**; **312**; **-35.33**; **149.13**; **132** ; **225W** ; **29700W** ; **Sunpower 225 WHT** ; **North** ; **6** ; **SMA SMC5000A** ; **5000W** ; **2603** ; **05/10/10** ; **Low** ; **25.0 Degrees** ; **http://solar.tridgell.net/**"

This variable contains the complete set of data scraped from the target webpages. A new element is created each time a PV site is completely scraped. This process is repeated recursively for each of the 20 PV sites included on the target page. The next-page URL address is found for the next page that includes the following set of 20 PV sites using XPath queries as follows:

```
$nextPageQuery = $xpath -> query("(//div[@id='tnt_pagination']/a/@href)[24]");

$contentQuery=$xpath -> query("(//td/a[@class='system1']/text())[1]");

if (($contentQuery->length)!=0){

$lastPageQuery = $xpath -> query("//tr/td[3]/a/@href"); //the number of hrefs in a
page, the last page will have none

      if (($lastPageQuery->length) > 18) {

      $nextUrl = "http://pvoutput.org/map.jsp" . $nextPageQuery->
      item(0)->nodeValue;

                  $array_2 =  array_merge($array_2, scrapePV($nextUrl));
      }
}
```

The pagination link is sent to the curl-handler. The same processes discussed earlier are repeated recursively until there are no more PV sites found in the HTML code of the next-page URL address investigated.

3.3 System Properties

3.3.1 Database Check Prior To Data Extraction

The output of the simulation is a Microsoft Excel database file containing all the PV sites as well as the scraped data relating to the sites. New PV sites are added to the 'pvoutput.org' server constantly, therefore, in order to keep the database up to date with these additions, the simulation is to be run on a regular basis. A problem faced when running subsequent simulations, was that the algorithm scrapes the data for all the PV sites on 'pvoutput.org' specific to the chosen state in Australia, including those that already exist in the database. This is inefficient, a more resource effective approach is to only scrape the updated data that is missing in the database.

The latter approach was implemented into the algorithm by importing the existent Excel database at the beginning of the simulation. Then, the PV site identification numbers were imported from the database and saved into an array. This array was used as a reference array, such that the site identification number of each PV site extracted during the simulation is compared with every element of the array. If a match is found, this means that the PV site already exists in the database, therefore the algorithm skips it to the next recursive cycle. If, however, a match is not found, that means that a new PV site has been detected. The remaining extraction steps are performed resulting in the addition of the PV site to the existing database.

```
$row =1;
if (($handle = fopen("sites_info_CBR.csv", "r")) !== FALSE) {
       while (($data = fgetcsv($handle,1000,",")) !==FALSE) {
       $row++;
       $sid_import[$row]= $data[0];     }
       fclose($handle);
}//end of importing sid values for pre-existing database
if (in_array($sid_values[1], $sid_import)){
              continue;    }
       else
//continue with code
```

3.3.2 Error detection

One of the most important tools used throughout the development of the algorithm was the error detection functionality. The algorithm incorporates the use of error detection to report coding mistakes, type mismatches and server problems. If an error occurs during the simulation of the algorithm, the error is reported and displayed on the output screen for the user to see. If the error is related to the coding, the line of the code is specified as well as the type of error encountered. The following code was incorporated to achieve this functionality:

```
        echo "<br />cURL error number:" .curl_errno($ch);
        echo "<br />cURL error:" .curl_error($ch) . "on URL - " . $url;
```

3.3.3 Stealth Web Crawling

The use of the XPath incorporates several settings, if the page accessed is not responding or redirecting the algorithm to another server, the algorithm is created such that it will follow all the redirects until it is redirected back to the target webpage.

```
curl_setopt($ch, CURLOPT_AUTOREFERER, true);
curl_setopt($ch, CURLOPT_FOLLOWLOCATION, true);
```

Websites monitor the amount of incoming and outgoing traffic they experience. Each IP address is therefore limited to a number of requests in a timeframe to access a webpage. If the limited is exceeded, the user is blocked from accessing the webpage content. This problem was faced in the initial stages of developing the algorithm. Due to the large number of page content requested, the server was blocked and could no longer access 'pvoutput.org' to complete the scraping process. This problem was overcome by utilizing a 'Googlebot' user-agent (19), which allows unlimited access to webpage content. 'Googlebot' disguises the requests sent by the software as being sent by Google servers. Requests from Google are not blocked by websites since they are essential for archiving the webpages and displaying them on search engines. The following code was used to enable the 'Googlebot':

```
curl_setopt($ch, CURLOPT_USERAGENT, "Mozilla/5.0 (compatible; Googlebot/2.1;
+http://www.google.com/bot.html)");
```

3.3.4 Data Storing of Extracted Data

Data obtained during the simulation is saved in real-time to an Excel database. This eliminates the potential problem of resource exhaustion. After the PV data for a site is saved onto the database, the same variables can be used again. Each time data for a new PV site is being extracted, the same arrays and variables are used and overwritten to save that data. Thus, a limited number of variables and arrays are used.

3.3.5 Executing the Code

In order to run a simulation, the following steps need to be taken:

1. Install and run a webserver software (eg: WAMP).
2. Ensure both PHP documents (Xpath.php and pvScrape.php) are in the webserver's directory folder.
3. Open a web-browser.
4. Specify the file directory address of the 'pvScrape.php' file in the address bar (eg: "C:\wamp\www\pvScrape.php").
5. Click Go.
6. The simulation will now run. Once the simulation has finished, an excel file with the name of the state scraped will be created in the same webserver's directory folder.

Chapter 4 Results & Using Data for Analysis

The following chapter demonstrates the validity of the algorithm developed, as well as the achievement of the research question of this paper. First, the results obtained as an output of simulating the algorithm will be displayed and discussed. Then, the usefulness of the obtained data will be highlighted by conducting a simple statistical analysis on the data. Finally, the use of the data to scrape other relevant PV performance-data and metadata will be discussed.

4.1 Results Obtained

Table 4 below, shows the number of PV systems reporting to 'pvoutput.org' in different states in Australia, namely: ACT, NSW, VIC, SA, QLD, WA, TAS and NT. These systems have been scraped and saved into their corresponding databases.

Table 4: Number of PV systems reporting in each state/territory in Australia

State	Number of PV systems scraped
ACT	348
NSW	1001
VIC	682
SA	1031
QLD	794
WA	817
TAS	217
NT	11
Total	**4902**

Once the simulation has been completed, an Excel file with extension (.scsv) is saved in the directory folder. Each database is saved under a name corresponding to the state name which the simulation ran on.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	sid	st	ar	tl	nm	mr	mt	sh	ni	ir	it	pc	lat	lon	name	url	notes				
2	5663	20110812	30000	27.5	120	250	Ningbo Qixin	1	3	9000	SMA Sunny Mini Central 9000TL	6164	-32.1328	115.8606	S & P ~ Lower Roof	http://pvo	Tin roof - Aspect 352Â° - 10cm clearance				
3	11850	20120920	21280	0	112	190	Hanwa SolarOne	1	5	5000	4xSMA 5000TL	6090	-31.8526	115.8859	Solargain Malaga PV	http://pvo	Commercial Installation 140 x 190W Panels on a flat r				
4	5463	20110422	17500	15	70	250	Ningbo Qixin	1	3	5000	SMA Sunny Boy 5000TL	6167	-32.2277	115.8705	Shed-27	http://pvo	utput.org/list.jsp?sid=5463				
5	5661	20110812	20000	27.5	80	250	Ningbo QIXIN	1	2	9000	SMA Sunny Mini Central 9000TL	6164	-32.1328	115.8606	S & P ~ High Roof	http://pvo	Tin roof - Aspect 352Â° - 10cm clearance				
6	1007	20110314	6460	17.5	18	190	Suntech STP190S	2	1	5000	SMA Sunny Boy 5000TL	6062	-31.8738	115.9087	Noranda Power	http://pvo	18 panels NW / 16 panels NE. Some morning shading.				
7	15794	20130124	12400	NA	62	200	FVG 72-175	2	1	10000	SMA TRIPOWER 10000TL	6150	-32.0596	115.8444	Icarus Maximus	http://pvo	Normal house, over 5KW so no FIT or REBS.				
8	1324	20101117	5460	26	28	195	Solarfun SF195	2	1	4600	Aurora PVI-5000	6208	-32.619	115.9734	SpringGrove6208	http://pvo	Had a Sunny Roo Inverter that failed				
9	2971	20100903	5040	26	24	210	Sanyo HIP210NKH	1	1	5040	SMA SB-5000TL-20	6109	-32.0599	116.0049	MoreSolar	http://pvo	utput.org/list.jsp?sid=2971				
10	1545	20110407	6460	23	34	190	Suntech STP190S-	3	1	5000	SMA 5000TL-20	6170	-32.2597	115.8056	Pat's PVA	http://pvo	Upgrade to 34 (from 30) panels 3/10/12 - terrible afte				
11	1280	20110308	5850	17	30	195	Suntech STP195S-	1	1	5000	SMA Sunny Boy 5000TL-20	6062	-31.8968	115.9166	Beats heating the p	http://pvo	utput.org/list.jsp?sid=1280				
12	1268	20110428	6080	25.5	32	190	Suntech STP190S:	2	1	5000	SMA 5000TL	6063	-31.859	115.934	Masda's Bennett Sp	http://pvo	upgraded system by 5 panels on august 30 2012.				
13	1339	20110619	5800	16	20	290	HHV Solar 290W M	1	1	5000	SMA SB 5000TL-20	6053	-31.9185	115.9128	Manhattan Project	http://pvo	Arrid racking system				
14	3525	20110915	5640	27	24	235	Sanyo HIT-N235SI	1	1	5060	SMA SB5000TL-20	6008	-31.9584	115.8179	PVgen Shenton104	http://pvo	2 PV inputs comprising arrays of (1 x 8) and (2 x 8) par				
15	160	20100825	5160	18	24	215	Hyundai HiS-215S	2	1	4400	Sunteams 4000	6066	-31.8404	115.8949	kaju2	http://pvo	no early direct sun, late winter shade				
16	984	20110203	4995	20	27	185	Suntellite ZDNY-1	1	1	5000	Suntellite ZDNY5000	6026	-31.8091	115.7889	yorky Home Solar	http://pvo	350deg orientation.				
17	1417	20110325	5700	25	30	190	Suntech Mono (S	1	1	5000	SMA Sunny Boy (SB 5000TL-20)	6155	-32.058	115.9181	T-Storm	http://pvo	2/3 NW, 1/3 NE				
18	74	20100601	4800	25	35	137	Sanyo 210/Solar F	1	4	5000	3 x SMA 1100 & 1 x SMA 1700	6065	-31.7342	115.7924	Sleepy Cat	http://pvo	Output figures as per Sunny Beam wireless monitor				
19	1451	20100106	4800	10	80	60	Kaneka K60	1	2	2500	SMA	6317	-33.6759	117.5558	Smiths Domain	http://pvo	40 panels east, 40 panels west				
20	865	20110114	5130	NA	27	190	Suntec	1	1	5300	SMA Sunny Boy 5000	6154	-32.0342	115.8086	solar-on-siddons	http://pvo	utput.org/list.jsp?sid=865				
21	395	20091214	5220	21.5	29	180	Conergy 18 x P17	2	1	5000	SMA SunnyBoy 5000TL	6105	-31.9678	115.9401	In Clover(dale)	http://pvo	9 panels face North East, actually 5220 Watts				
22	1736	20110526	5400	20	30	180	BP Solar BP4180T	1	1	5000	SMA SB5000TL-20	6149	-32.0649	115.8587	MadParrot PV Powe	http://pvo	20 panels facing North and 10 panels facing East				
23	2748	20110706	5320	27	28	190	Suntech STP190S-	2	1	5000	SMA Sunny Boy 5000TL	6065	-31.7161	115.7737	MRC PV Plant	http://pvo	System from UNLTD Solar (now NEXT POWER)				
24	972	20110202	5220	20	18	290	HHV HSTAF24290	1	1	5000	SMA Sunny Boy 5000TL-20	6109	-32.0502	115.9764	Richo's Solar by Ene	http://pvo	2 x 9 NE/NW Split Array				
25	983	20110316	5850	22	30	195	SUNTECH MONO-	1	1	5300	SMA SUNNY BOY SB5000TL-20	6155	-32.0882	115.9249	GSI - Canningvale al	http://pvo	utput.org/list.jsp?sid=983				
26	1511	20110621	5800	18	20	290	HHV HSTAF24290	1	1	5000	Sunny Boy 5000-TL	6026	-31.8091	115.7889	Pieface's Photonic I	http://pvo	utput.org/list.jsp?sid=1511				
27	220	20100818	4180	1	22	190	CEEG Monocrysta	1	1	5000	SMA Sunny Boy 5000	6016	-31.9208	115.8142	NTC	http://pvo	utput.org/list.jsp?sid=220				
28	2217	20110819	5700	17	20	190	Suntech STP190S-	2	1	5000	SMA Sunny Boy SB5000TL-20	6169	-32.3023	115.7423	Speedy Dave's	http://pvo	GSI, 20 North and 10 East				
29	2315	20110602	5700	18	30	190	Suntech STP190S	2	1	5000	SMA Sunny Boy 5000TL	6053	-31.9185	115.9128	Bayswater Sun Fact	http://pvo	16 panels facing east, 14 panels facing north. early m				
30	7846	20111220	8280	25	36	230	Conergy P-Plus 9-	2	1	5000	SMA SB5000TL-20	6101	-31.9751	115.9149	Boomer's PV	http://pvo	1x9 NE and 1x9 ESE Polystring, 2x9 NW parallel				
31	1349	20110316	5850	30	30	195	Suntech STP-195S	1	1	5000	SMA Sunny Boy 5000TL-20	6065	-31.7342	115.7924	Ed's Solar Farm	http://pvo	Array is split 12 east and 18 west				
32	9086	20120511	7680	23	32	240	32 x Q-Cells ProG	1	1	5000	SMA Sunny Boy 5000TL-20	6163	-32.0779	115.7763	Champagne Supern	http://pvo	Started as 5.17kw upgraded to 7.68kw				
33	3077	20110730	5700	25	30	190	Suntech STP190S-	1	1	5000	SMA SB5000TL	6065	-31.7342	115.7924	Apollonia	http://pvo	Let There Be Sun!!!!				
34	3325	20110817	5320	1	8	190	Suntech STP190S-	2	1	5000	SMA Sunny Boy 5000TL	6018	-31.863	115.7995	Aachen Power Grid	http://pvo	8 Panels North 20 East				
35	2109	20110519	5390	30	49	110	Kaneka U-DB110	1	1	5000	SMA SMC-5000A	6149	-32.0649	115.8587	Leeming Hybrid	http://pvo	System upgraded to 5.39kW, offline until 2pm				
36	839	20110113	5460	5	28	195	Suntech ST195S-2	1	1	5000	SMA Sunnyboy 5000TL	6057	-31.9394	116.0133	Roodogs power pla	http://pvo	utput.org/list.jsp?sid=839				
37	6234	20120223	6270	NA	33	190	Solartech Sunrise	1	1	5000	SMA 5000 TL	6167	-32.2277	115.8705	Bertram_Yelyab	http://pvo	http://greensunsolar.com.au/				
38	629	20101201	5510	31	29	190	STP190-18/Ud	1	1	5000	SMA SunnyBoy 5000TL-20	6065	-31.817	115.8516	MnM's SolarPower	http://pvo	utput.org/list.jsp?sid=629				
39	2462	20091204	3360	18	16	210	Sunpower 210 bll	1	1	2650	Fronius IG 30 indoor 2.5 kW	6104	-31.9526	115.9364	Signal Hill	http://pvo	2 string of 8 as of Oct/2012 was 3 string of 5				

Figure 12: First 38 entries of raw Excel database for WA

Figure 12 shows the first 38 entries out of 817 of the Excel database obtained for WA. The columns of the database correspond to (from left to right): system identification number (sid), starting reporting date (st), array size (ar), module tilt (tl), number of modules (nm), module rating (mr), module type (mt), shading (sh), number of inverters (ni), inverter rating (ir), inverter type (it), postcode (pc), latitude (lat), longitude (lon), name of system (name), the URL of the system (url) and notes about the system (notes). These abbreviations have been chosen so that each column in the database could be used as an object when used in conjunction with the "NickEngerer" R package (20).

4.2 Analysing the Data

Overall, eight databases corresponding to Australia's states and territories have been compiled. In this section, data obtained for each state and territory will be analysed. First, the PV systems scraped are displayed on a map to show the distribution of the sites within each state according to the system size. Then, the system size distribution will be graphed and displayed on a histogram. Finally, a statistical analysis on each data-set will be performed and displayed in a table.

4.2.1 Analysing the Australian Capital Territory (ACT)

Figure 13 below shows a bubble map of the distribution of PV systems within the ACT according to their system size. A bubble map is a map that displays the distribution of a certain variable over a geographical region using bubbles, each bubble has a size proportional to the associated data, in this case, the system size. It is worth noting that there is a similar distribution between the number of PV sites reporting from the Northern and Southern parts of the ACT. Moreover, there is a significantly less number of PV sites reporting from the Eastern rural parts of the ACT. The largest system 'A Little Bit of Sunshine', rated at 29.7 kW, is reporting from the 'Red Hill' district in the ACT.

Figure 14 shows a scatter plot of the array size for all the sites scraped compared to the average system size in the ACT. The average system size in the ACT is 4795.22 W, while the total size of installed PV systems is 1.67 MW. Figure 15 shows a histogram of the distribution of the system sizes. Only 11 sites have a system rating equal to or larger than 10kW, while 70.40% of the systems range between 4-6kW. The most installed system size ranges between 4000W – 5000W, with a frequency of 102 systems, accounting for 29.30% of the total number of systems reporting.

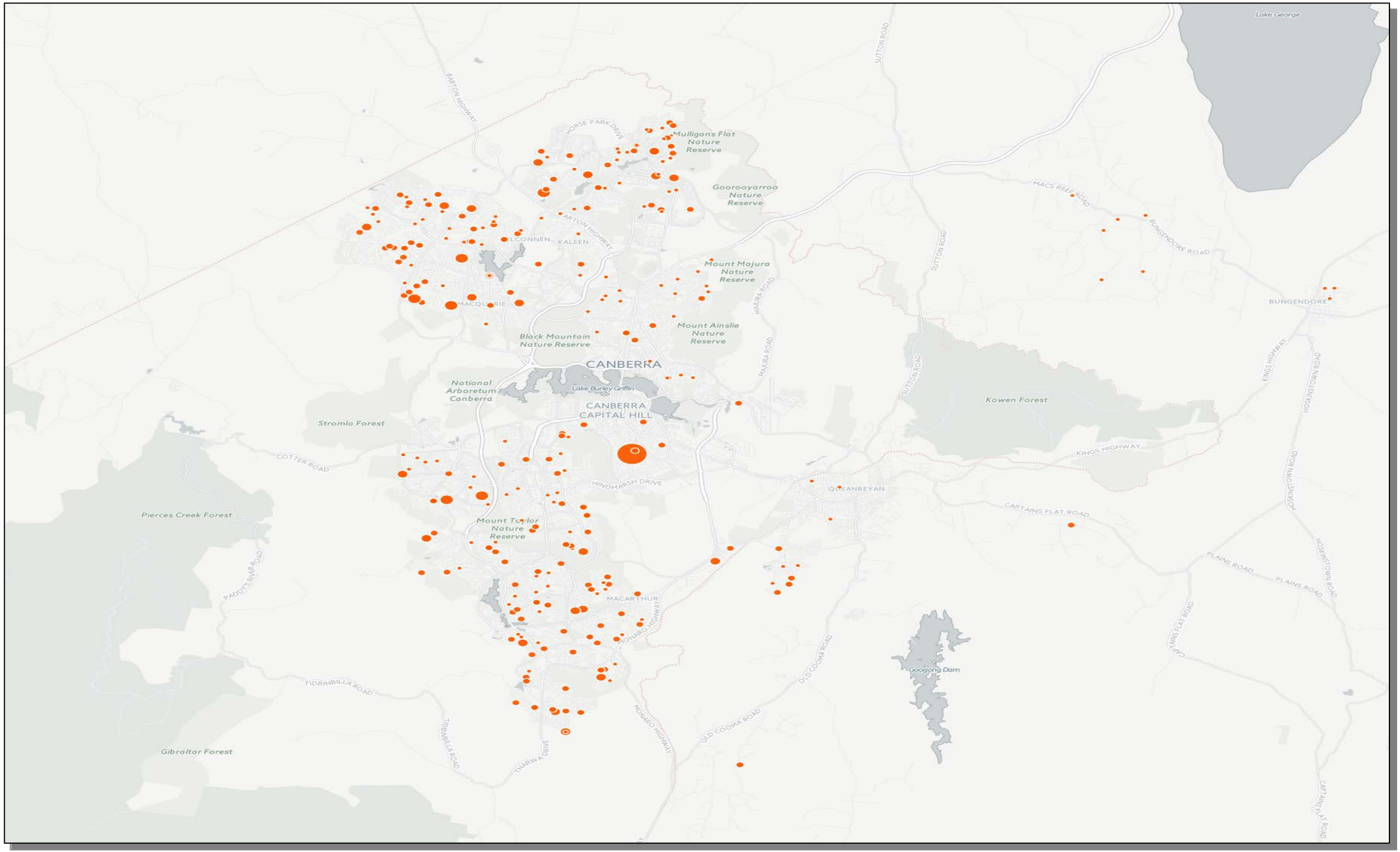


Figure 13: PV sites distribution in the ACT © OpenStreetMap contributors ©CartoDB CartoDB attribution

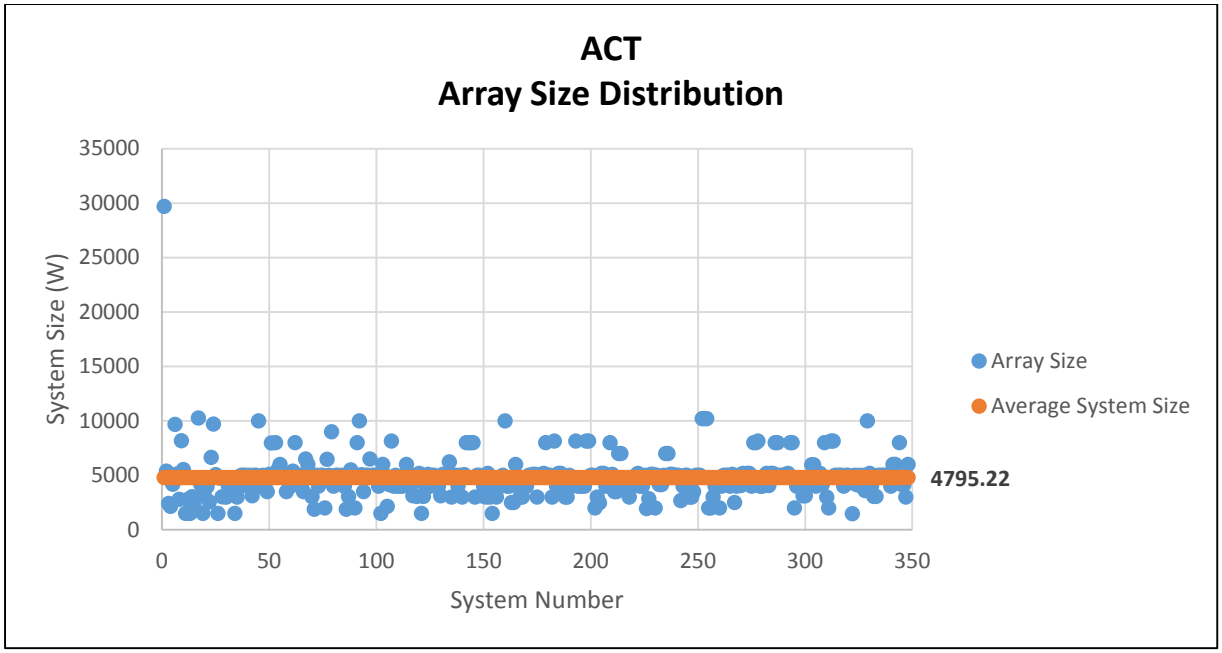


Figure 14: ACT array size distribution

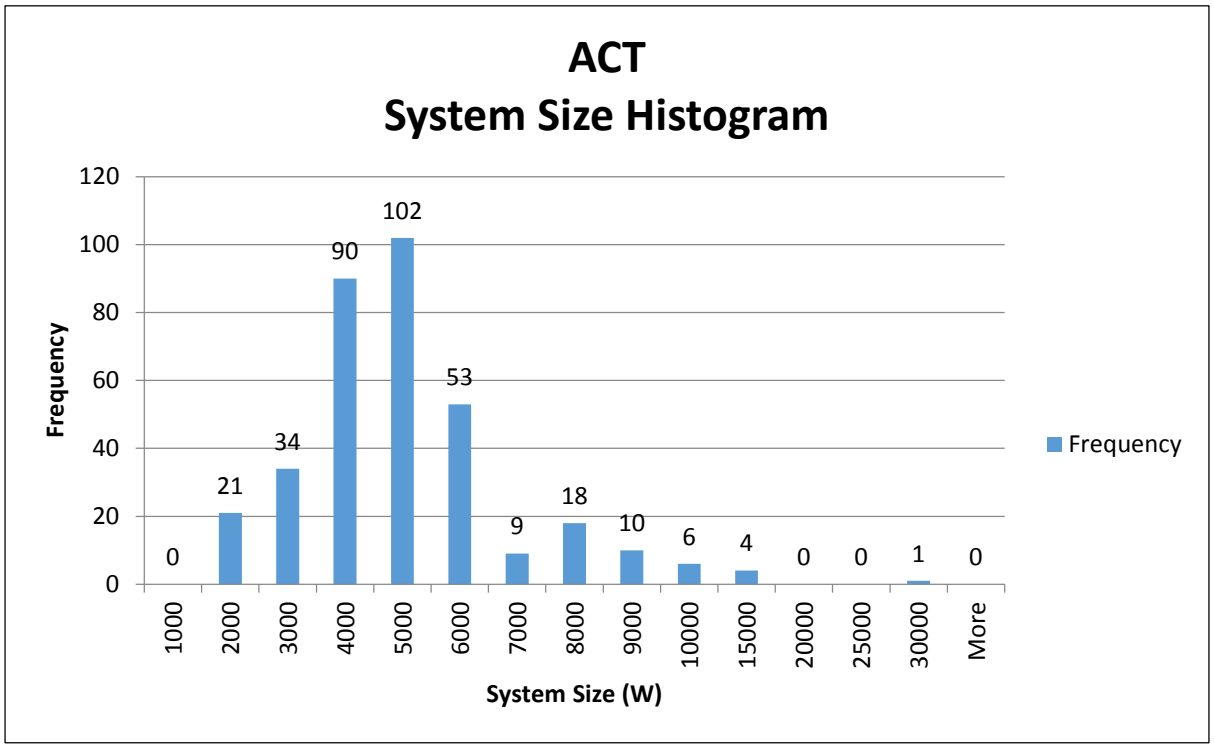


Figure 15: ACT system size histogram

Table 5 displays a summary of findings as a result of analysing the ACT database. The highest reporting postcode in the ACT was 2615, with 44 systems reporting. In addition, the most used module type is the 'TSM-250PC05A (60 Poly cells)' module, while the most used inverter type is the 'SB5000TL'. The average module tilt in the ACT is 18.39°.

Table 5: Summary of ACT database analysis

ACT							
Number of Systems Scraped	**Average System Size (W)**	**Largest System Sizes (W)**	**Number of Occurrences**	**Lowest System Sizes (W)**	**Number of Occurrences**		
348	4795.22	29700	1	1962	1		
		10260	1	1900	2		
		10200	3	1520	6		
		10000	4	1500	2		
		9720	1	1480	1		
Most Used Inverter Size (W)	**Number of Occurrences**	**Most Used Inverter Type**	**Number of Occurrences**	**Highest Five Reporting Postcodes**	**Number of Occurrences**	**Most Used Module Rating (W)**	**Number of Occurrences**
5000	154	SB5000TL	44	2615	44	250	83
4000	98	SB4000TL	28	2914	30		
3000	44	SB5000TL-20	23	2905	24		
2000	12	SB4000TL-21	22	2617	22		
6000	8	SB5000TL-21	20	2913	21		
Most Used Module Type	**Number of Occurrences**	**% of Systems Equal or Larger than 5 kW**	**Average Module Tilt**	**Total State PV System Size Installed (MW)**			
TSM-250PC05A (60 Poly cells)	75	53.16	18.39	1.67			
250PC05A	40						
CS6P-250P	35						
6P250P	23						
LG260S1C-G3	22						

4.2.2 Analysing Queensland (QLD)

Figure 16 below shows a bubble map of the distribution of PV systems within the CBD area of Brisbane according to their system size. There is a large density of PV systems reporting within proximity from the CBD area, compared to areas further away from the CBD. The largest system has a rating of 100.80kW and is reporting from the 'Eight Mile Plain' suburb in Brisbane.

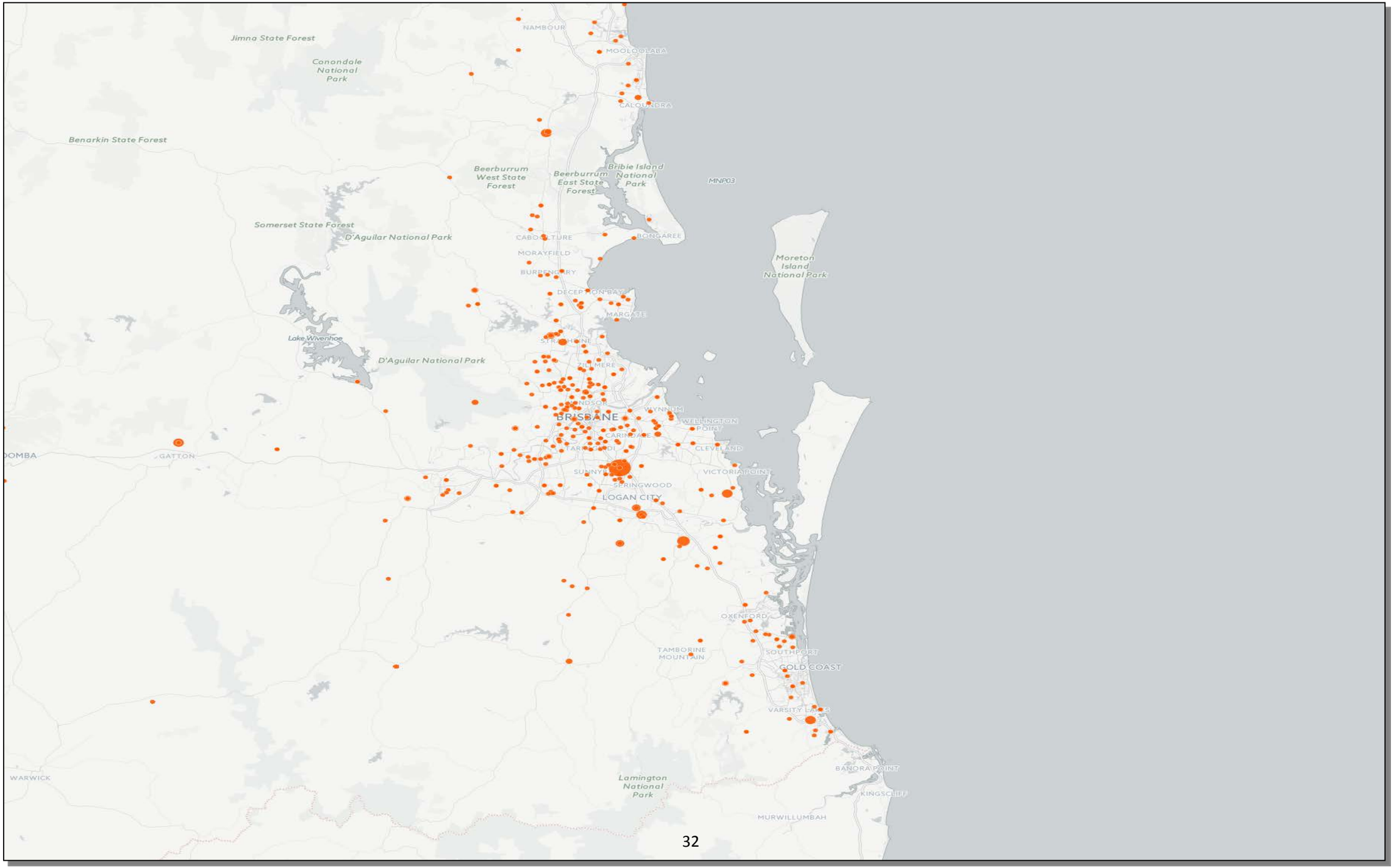


Figure 16: PV system distribution in QLD © OpenStreetMap contributors ©CartoDB CartoDB attribution

Figure 17 shows a scatter plot of the array size for all the sites scraped compared to the average system size in QLD, while Figure 18 shows a histogram of the distribution of the system sizes.

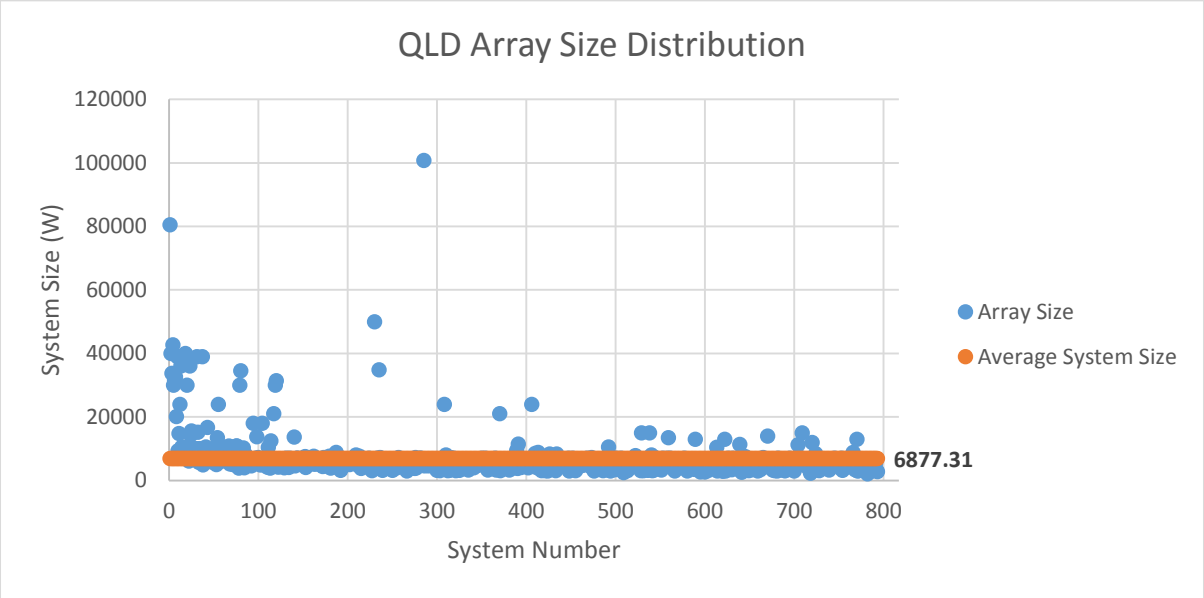


Figure 17: QLD array size distribution

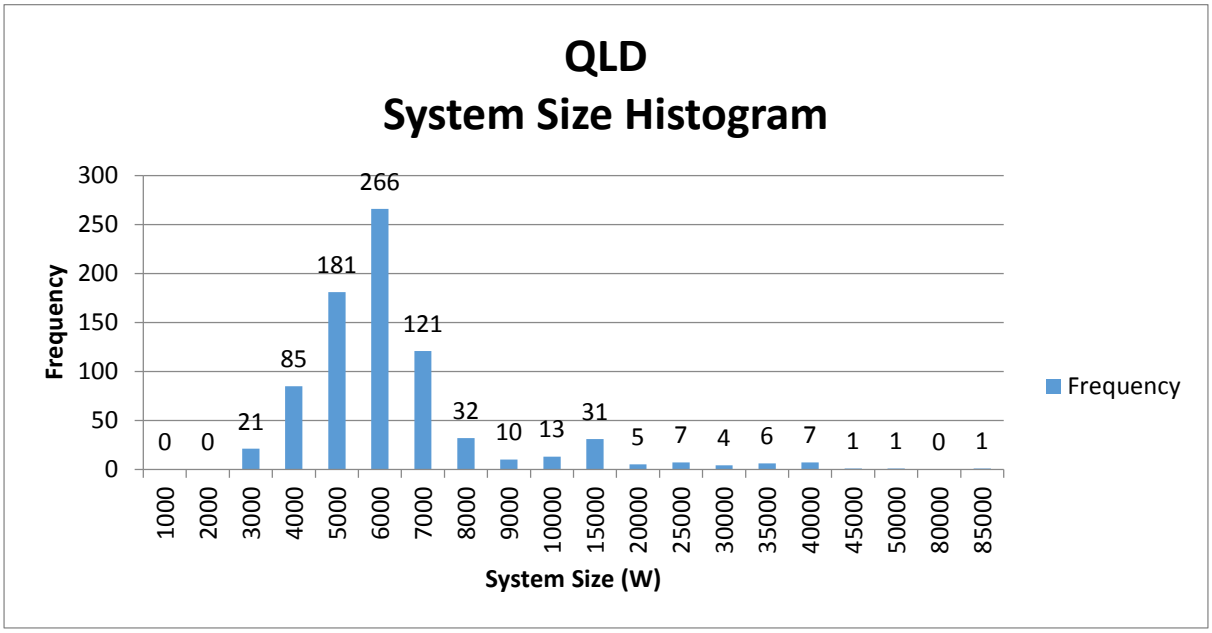


Figure 18: System Size Histogram

The majority of the systems installed have a rating between 4000W - 6000W, accounting for 33.50% of the total number of systems installed, followed by system sizes ranging from 4000W – 5000W accounting for 22.79% of the total number of systems installed.

Table 6: Summary of QLD database analysis

QLD							
Number of Systems Scraped		**Average System Size (W)**		**Largest System Sizes (W)**	**Number of Occurrences**	**Lowest System Sizes (W)**	**Number of Occurrences**
793		6877.31		100800	1	2660	1
				80500	1	2590	1
				50000	1	2450	1
				42735	1	2300	1
				40000	2	2100	1
Most Used Inverter Size (W)	**Number of Occurrences**	**Most Used Inverter Type**	**Number of Occurrences**	**Highest Five Reporting Postcodes**	**Number of Occurrences**	**Most Used Module Rating (W)**	**Number of Occurrences**
5000	309	Trannergy PVI5400TL	221	4053	22	250	320
4600	240	SMA Sunny Boy 5000	23	4500	21	190	136
4000	35	SMA Sunny Boy 5000TL	20	4017	18	200	120
2800	34	Aurora	16	4152	17	235	37
10000	29	Aurora PVI-5000	15	4814	15	185	24
Most Used Module Type	**Number of Occurrences**	**% of Systems Larger than 5 kW**	**Average Module Tilt (o)**	**Total State PV System Size Installed (MW)**			
ET-M572200	79	75.40	17.35	5.45			
ET-M660250	56						
ET Solar ET-M660250	38						
ET	31						
ET Solar ET-M572200	27						

Table 6 displays a summary of findings as a result of analysing the QLD database. The highest reporting postcode in QLD was 4053, with 22 systems reporting. In addition, the most used module type is the 'ET-M572200' module, while the most used inverter type is the 'Trannergy PVI5400TL'. The average module tilt in QLD is 17.35° and the total installed system size is 5.45 MW.

4.2.3 Analysing Victoria (VIC)

Figure 19 below shows a bubble map of the distribution of PV systems within the CBD area of Melbourne according to their system size.

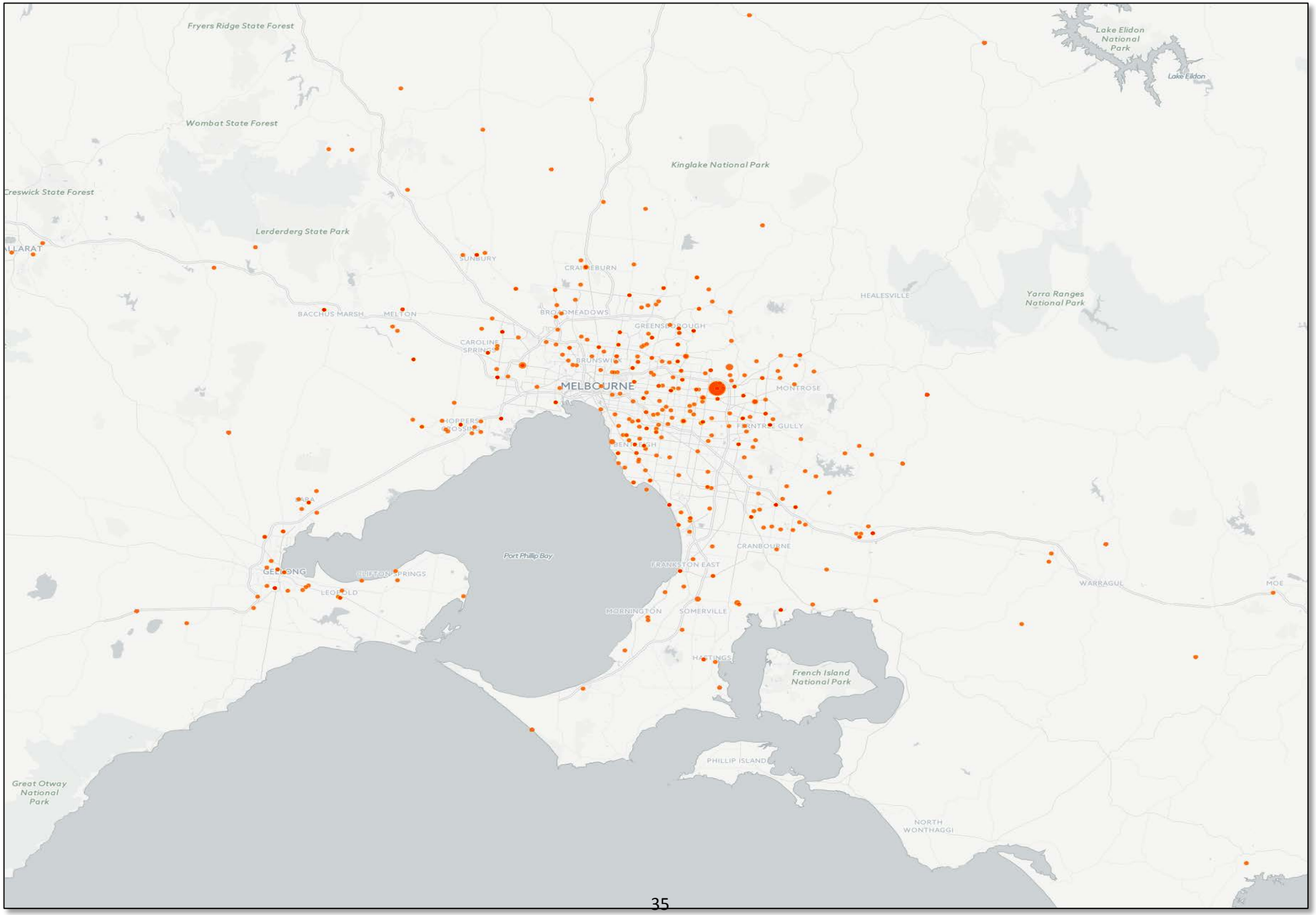


Figure 19: PV system distribution in VIC © OpenStreetMap contributors ©CartoDB CartoDB attribution

Figure 20 shows a scatter plot of the array size for all the sites scraped compared to the average system size in VIC, while Figure 21 shows a histogram of the distribution of the system sizes.

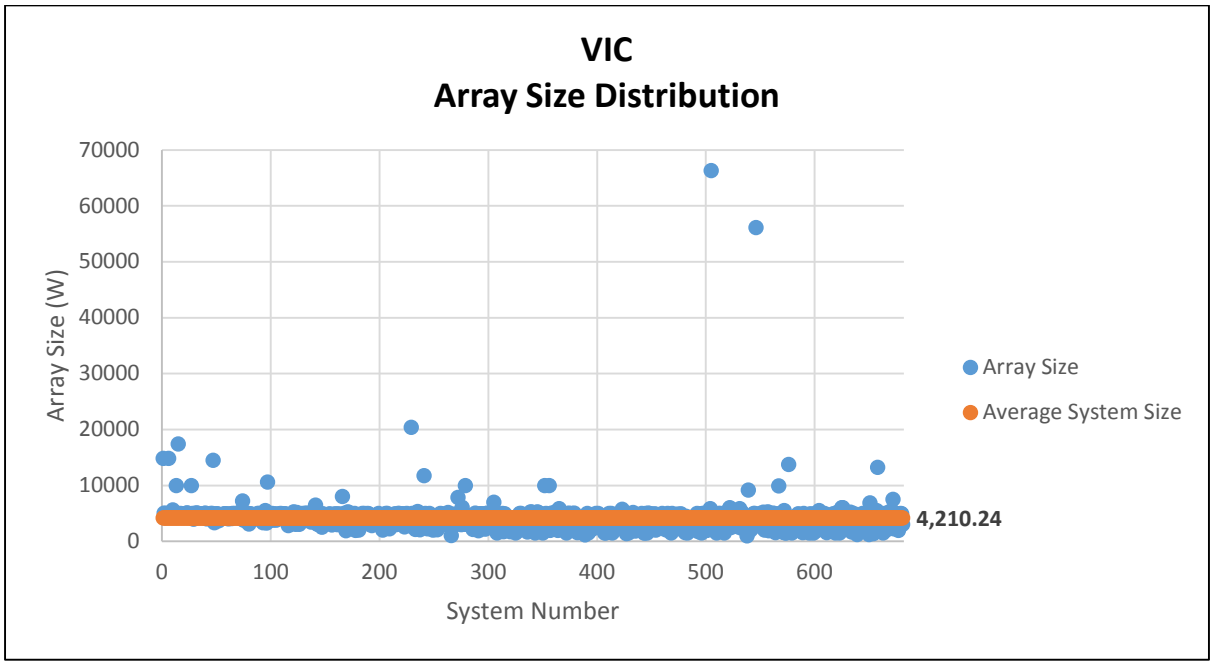


Figure 20: Array size distribution in VIC

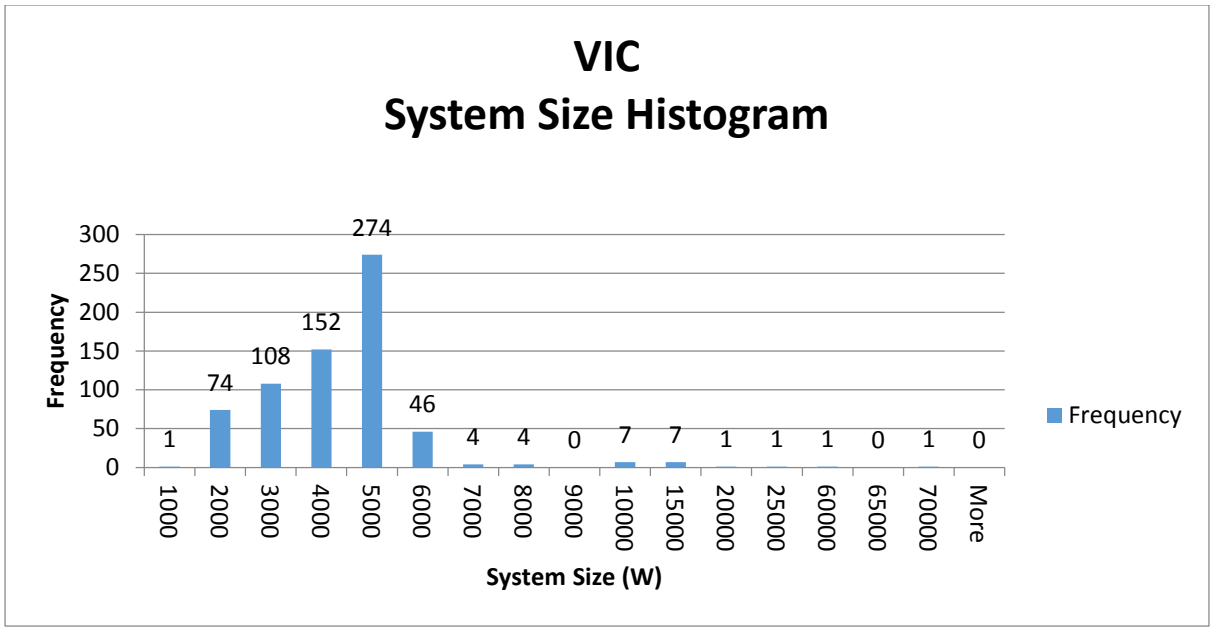


Figure 21: System size histogram for VIC

Table 7 displays a summary of findings as a result of analysing the VIC database. The highest reporting postcode in VIC was 3030, with 18 systems reporting. In addition, the most used module type is the 'Suntech' module, while the most used inverter type is the 'SMA Sunny

Boy 5000TL' inverter. The average module tilt in VIC is 21.24° and the total installed system size is 2.87 MW.

Table 7: Summary of VIC database analysis

VIC							
Number of Systems Scraped		**Average System Size (W)**		**Largest System Sizes (W)**	**Number of Occurrences**	**Lowest System Sizes (W)**	**Number of Occurrences**
681		4210.24		66300	1	1239	1
				56100	1	1200	1
				20400	1	1140	1
				17400	1	1050	1
				14820	2	990	1
Most Used Inverter Size (W)	**Number of Occurrences**	**Most Used Inverter Type**	**Number of Occurrences**	**Highest Five Reporting Postcodes**	**Number of Occurrences**	**Most Used Module Rating (W)**	**Number of Occurrences**
5000	244	SMA Sunny Boy 5000TL	28	3030	18	250	208
3000	82	SMA	20	3805	12	190	174
4000	55	SMA Sunny Boy 5000	18	3029	12	185	38
2000	45	Trannergy PVI5400TL	13	3340	11	235	33
2800	23	SMA Sunny Boy 3000	12	3156	11	180	19
Most Used Module Type	**Number of Occurrences**	**% of Systems Larger than 5 kW**	**Average Module Tilt (o)**	**Total State PV System Size Installed (MW)**			
Suntech	24	22.17	21.24	2.87			
REC	23						
ET-M660250WW	17						
NESL	13						
Canadian Solar	12						

4.2.4 Analysing South Australia (SA)

Figure 22 below shows a bubble map of the distribution of PV systems within the CBD area of Adelaide according to their system size. The largest system reporting is rated at 98kW and is located in the 'Park Holme' suburb in Adelaide, SA.

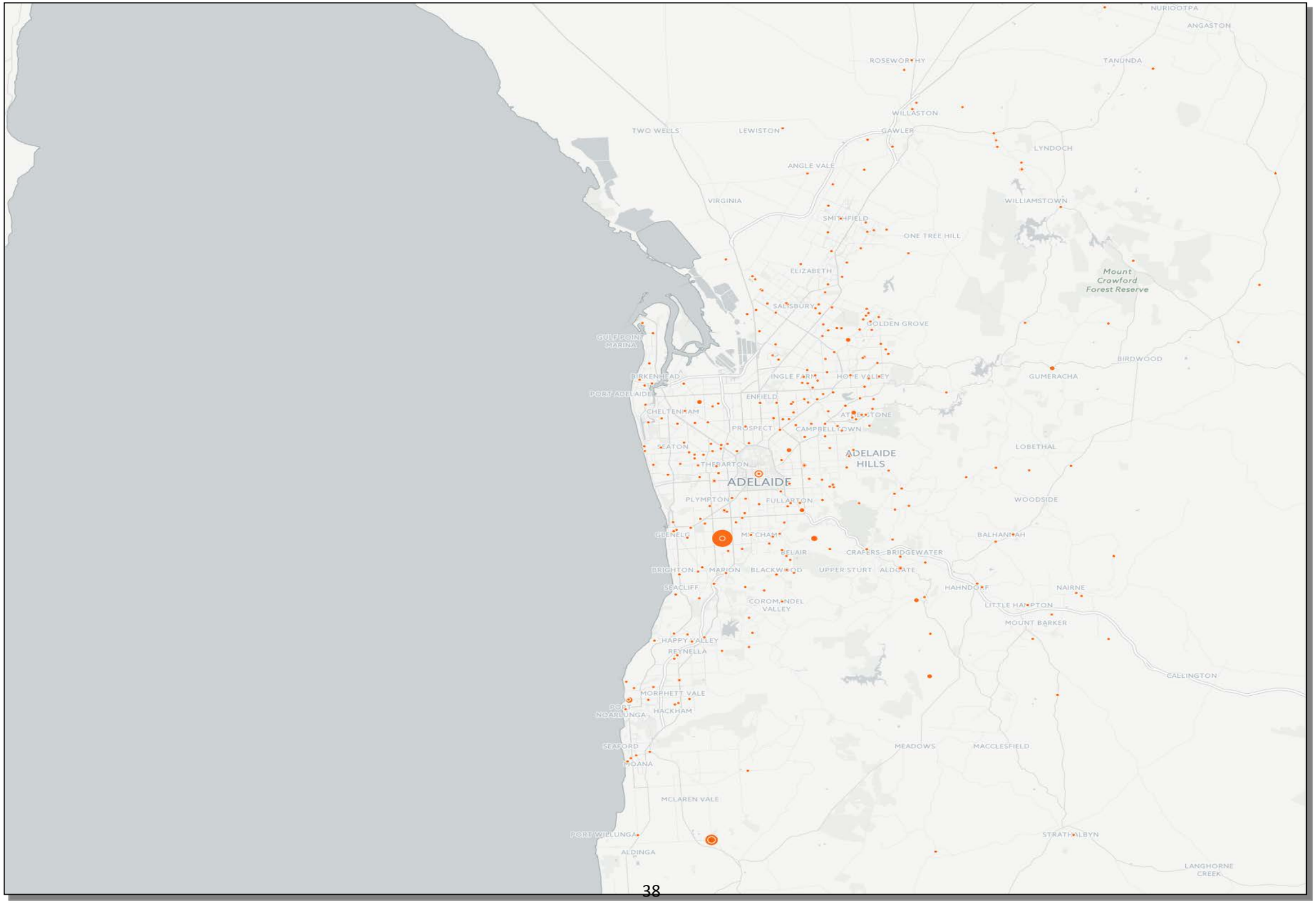


Figure 22: PV system distribution in SA © OpenStreetMap contributors ©CartoDB CartoDB attribution

Figure 23 shows a scatter plot of the array size for all the sites scraped compared to the average system size in SA, while Figure 24 shows a histogram of the distribution of the system sizes.

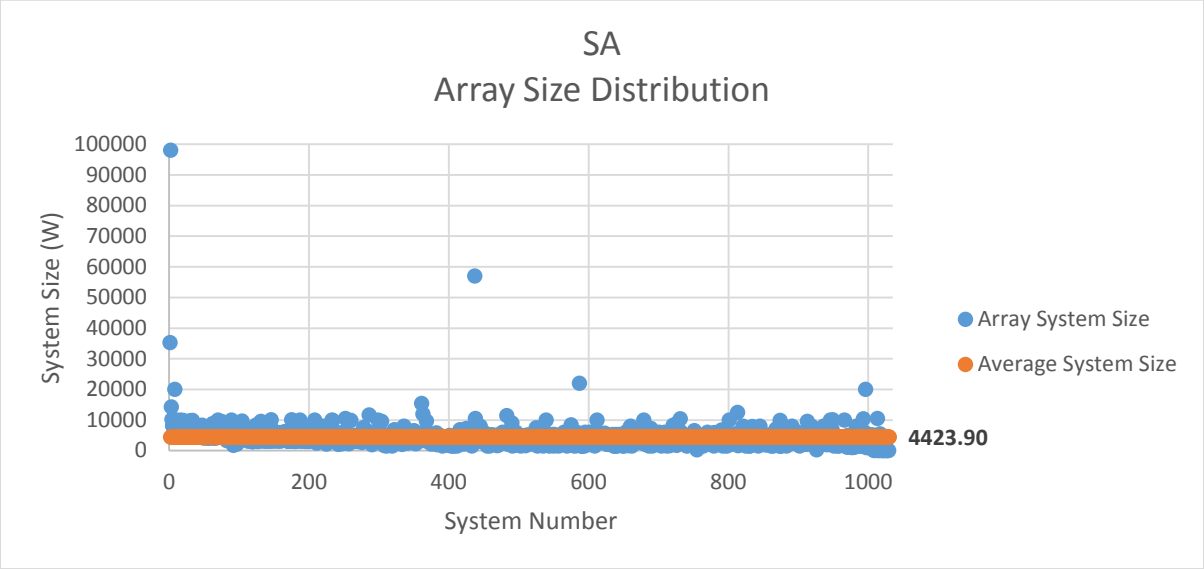


Figure 23: Array size distribution in SA

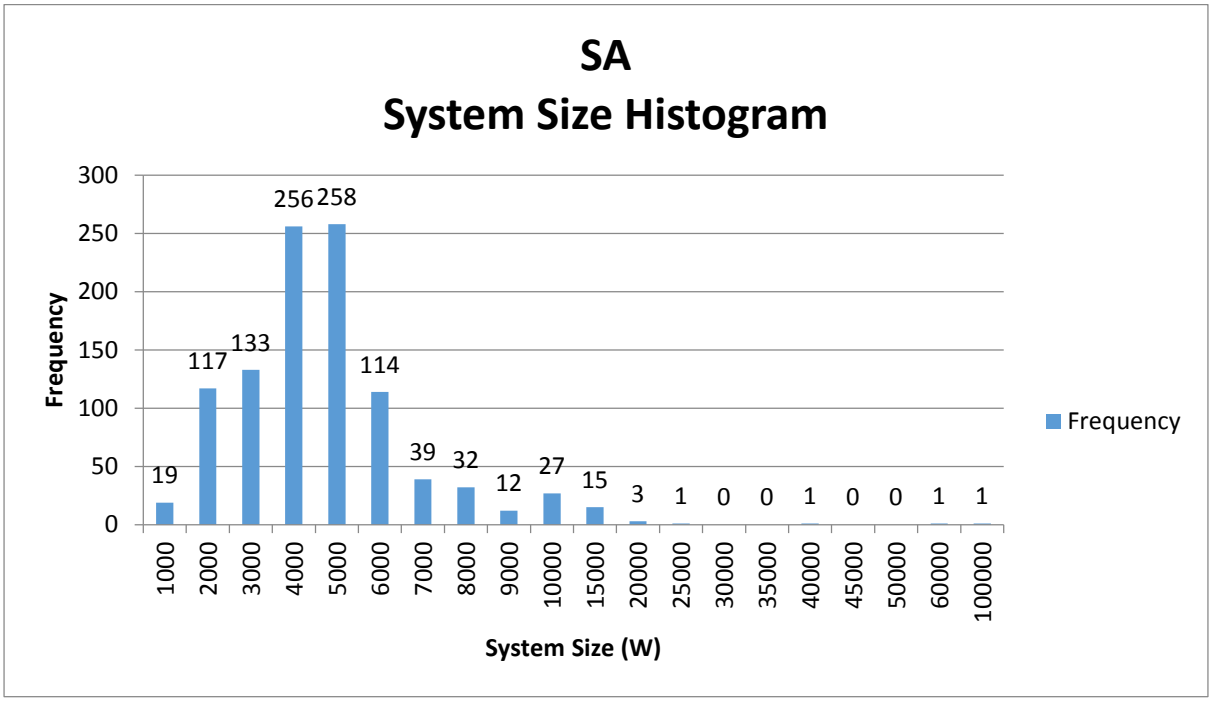


Figure 24: System size histogram in SA

The majority of systems reporting in SA have a system size ranging from 3000W-5000W, accounting for 49.85% of the systems reporting in SA. While 4.75% of the systems have a rating of 10,000W or higher.

Table 8 displays a summary of findings as a result of analysing the SA database. The highest reporting postcode in SA is 5158, with 31 systems reporting. In addition, the most used module type is the 'Suntech' module, while the most used inverter type is the 'SMA Sunny Boy 5000TL' inverter. The average module tilt in SA is 20.25° and the total installed system size is 4.56 MW.

Table 8: Summary of SA database analysis

SA							
Number of Systems Scraped		**Average System Size (W)**		**Largest System Sizes (W)**	**Number of Occurrences**	**Lowest System Sizes (W)**	**Number of Occurrences**
1029		4423.9		98000	1	690	1
				57000	1	250	1
				35250	1	240	1
				22000	1	55	1
				20000	2	50	14
Most Used Inverter Size (W)	**Number of Occurrences**	**Most Used Inverter Type**	**Number of Occurrences**	**Highest Five Reporting Postcodes**	**Number of Occurrences**	**Most Used Module Rating (W)**	**Number of Occurrences**
5000	286	SMA Sunny Boy 5000TL	38	5158	31	190	312
3000	160	sma	34	5159	27	250	265
4000	131	SMA Sunny Boy 5000	25	5125	26	235	78
6000	47	aurora	21	5162	23	200	55
2000	37	SMA Sunny Boy 4000TL	20	5086	21	260	36
Most Used Module Type	**Number of Occurrences**	**% of Systems Larger than 5 kW**	**Average Module Tilt (o)**	**Total State PV System Size Installed (MW)**			
Suntech	86	32.56	20.25	4.56			
REC	38						
N/A	37						
ET	19						
Simax	18						

4.2.5 Analysing New South Wales (NSW)

Figure 25 below shows a bubble map of the distribution of PV systems within NSW according to their system size. There is a large distribution of PV systems reporting throughout the state, however, the majority of the systems are reporting

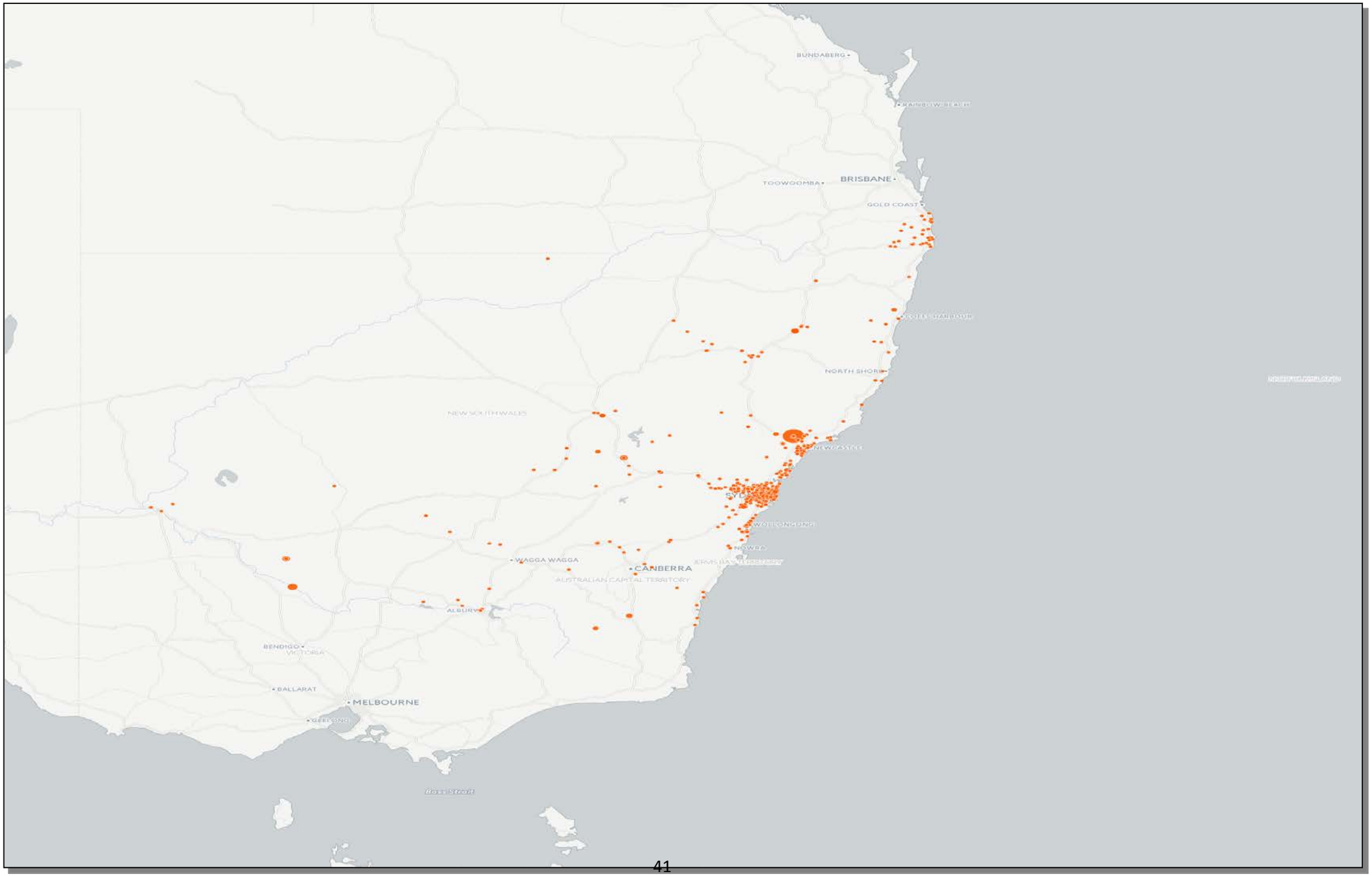


Figure 25: PV system distribution in NSW © OpenStreetMap contributors ©CartoDB CartoDB attribution

within close proximity to the Sydney CBD area. The largest system is rated at 97.9kW and is located in the 'Maitland' suburb. Figure 26 shows a scatter plot of the array size for all the sites scraped compared to the average system size in NSW, while Figure 27 shows a histogram of the distribution of the system sizes. It is worth noting that 61.60% of the systems are rated between 2000W-5000W.

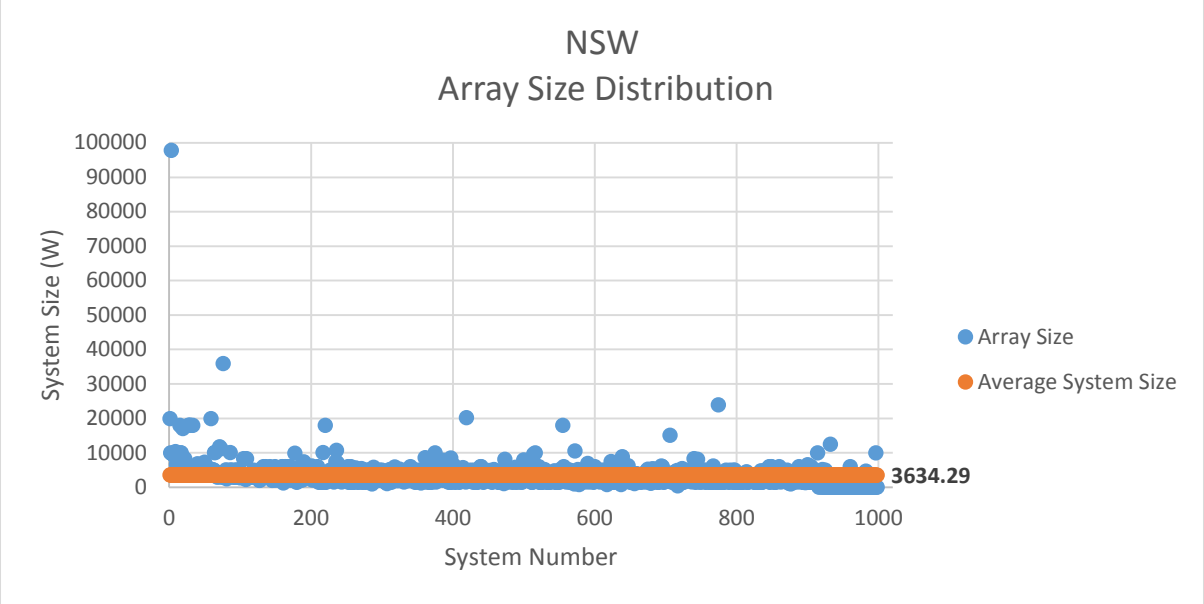


Figure 26: NSW array size distribution

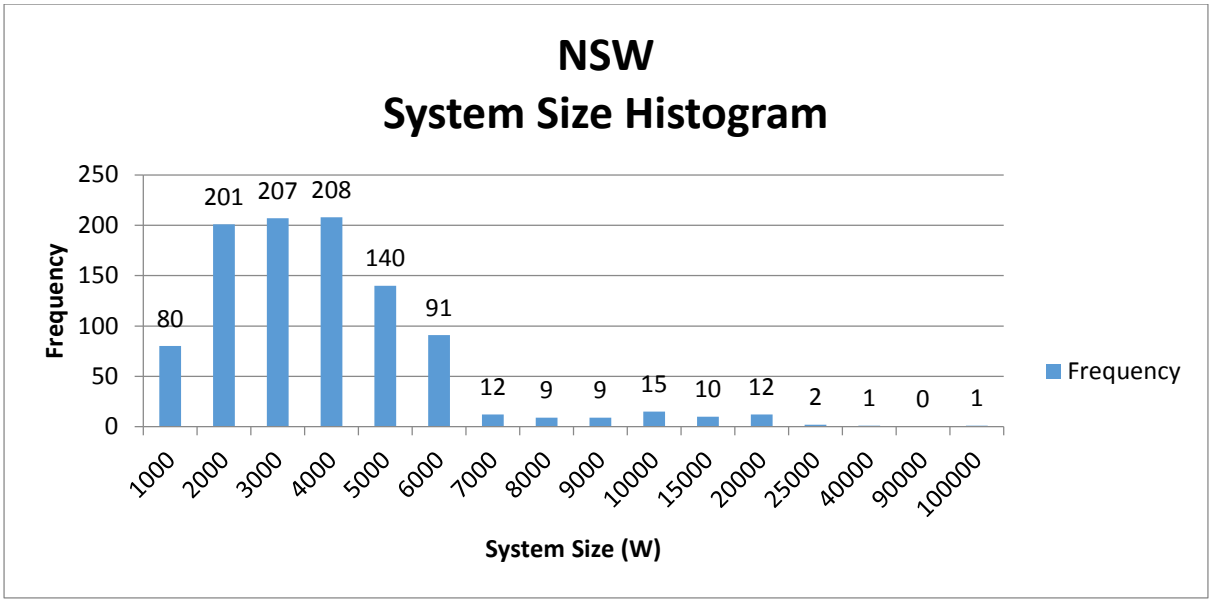


Figure 27: NSW system size histogram

Table 9 displays a summary of findings as a result of analysing the NWS database. The highest reporting postcode in NSW is 2733, with 37 systems reporting. In addition, the most used

module type is the 'ET Solar ET-M660250' module, while the most used inverter type is the 'Energy Monitor' inverter. The average module tilt in NSW is 19.20° and the total installed system size is 3.61 MW.

Table 9: Summary of NSW database analysis

NSW							
Number of Systems Scraped		**Average System Size (W)**		**Largest System Sizes (W)**	**Number of Occurrences**	**Lowest System Sizes (W)**	**Number of Occurrences**
998		3624.29		20000	2	50	71
				20210	1	150	1
				24000	1	500	1
				36000	1	780	1
				97900	1	800	1
Most Used Inverter Size (W)	**Number of Occurrences**	**Most Used Inverter Type**	**Number of Occurrences**	**Highest Five Reporting Postcodes**	**Number of Occurrences**	**Most Used Module Rating (W)**	**Number of Occurrences**
5000	158	Energy Monitor	69	2733	37	2335	1
3000	120	Trannergy PVI5400TL	41	2619	16	2042	1
2000	97	Aurora	26	2153	16	2090	1
50	79	SMA	19	2155	15	2445	1
4000	54	CMS2000	18	2259	15	2867	1
Most Used Module Type	**Number of Occurrences**	**% of Systems Larger than 5 kW**	**Average Module Tilt (o)**	**Total State PV System Size Installed (GW)**			
ET Solar ET-M660250	37	21.14	19.2	3.61			
Suntech	31						
Trina	24						
Upsolar	16						
Trina Honey	14						
Conergy	14						

4.2.6 Analysing Western Australia (WA)

Figure 28 below shows a bubble map of the distribution of PV systems within the CBD area of Perth according to their system size.

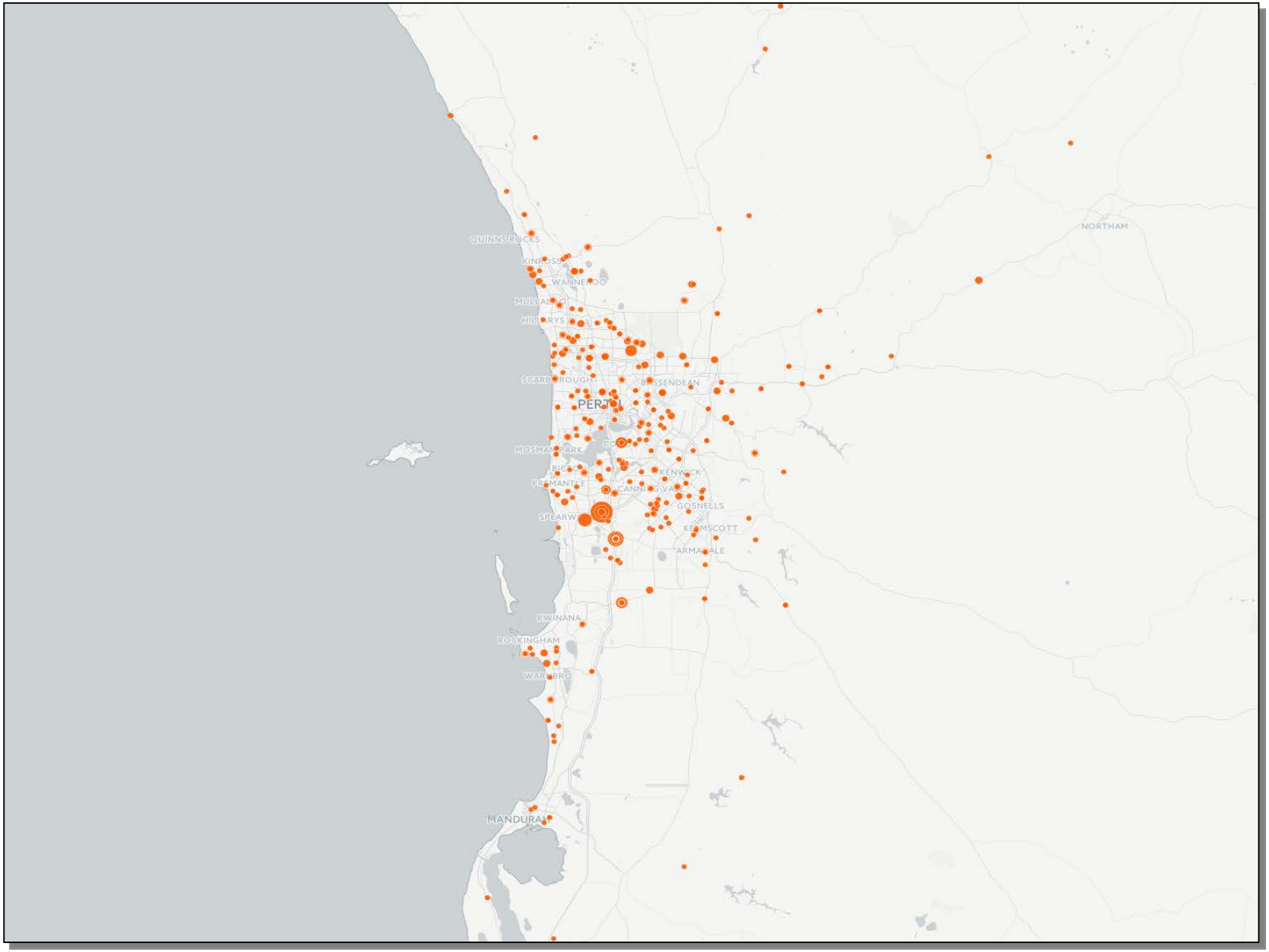


Figure 28: PV system distribution in WA © OpenStreetMap contributors ©CartoDB CartoDB attribution

Figure 29 shows a scatter plot of the array size for all the sites scraped compared to the average system size in WA, while Figure 30 shows a histogram of the distribution of the system sizes. It is worth noting that 87.15% of the systems are rated between 2000W-6000W.

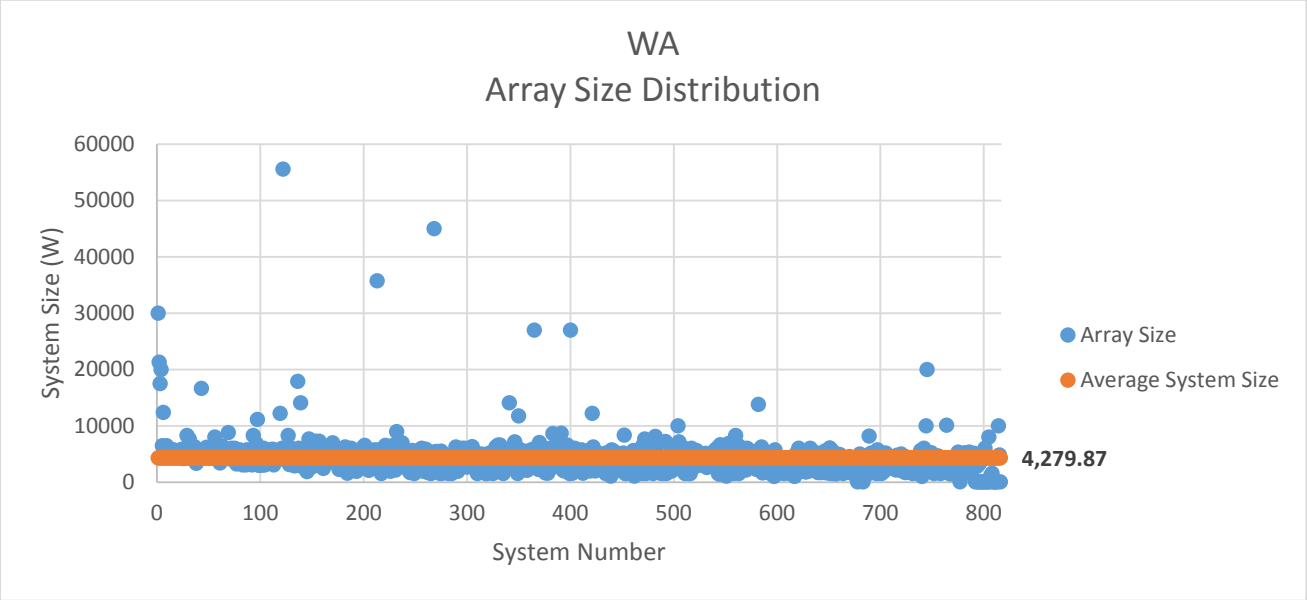


Figure 29: WA array size distribution

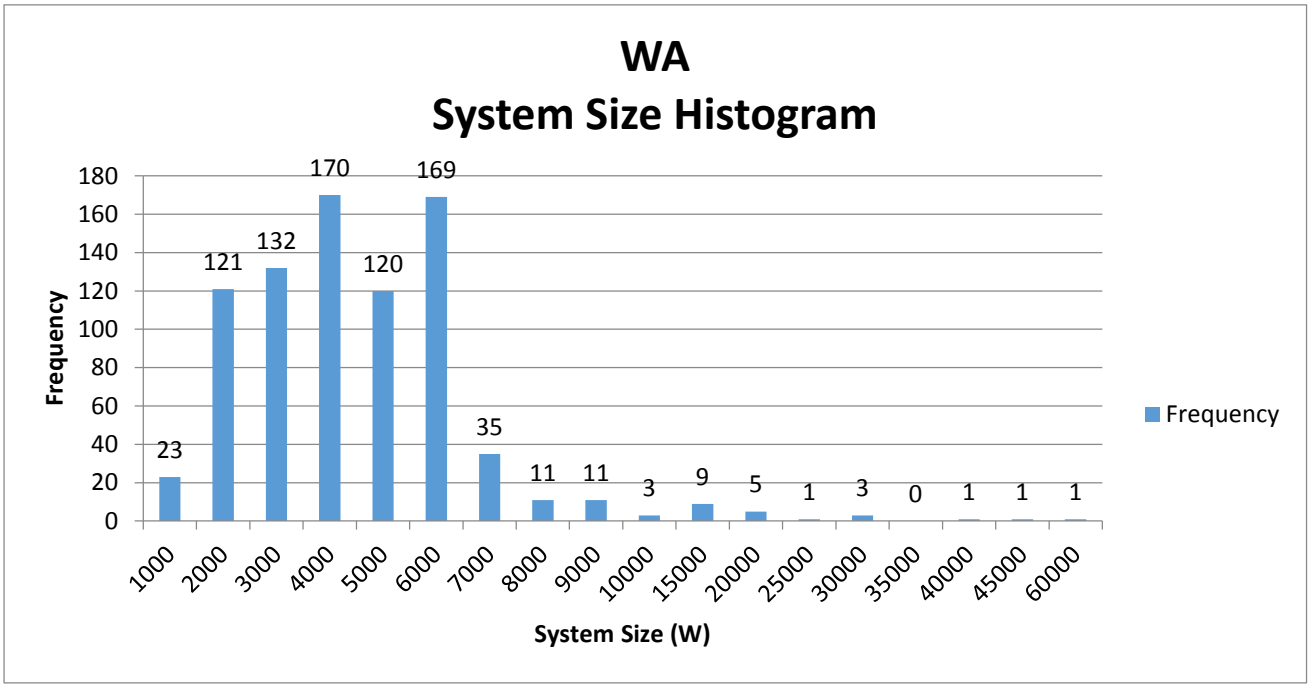


Figure 30: WA system size histogram

Table 10 displays a summary of findings as a result of analysing the WA database. The highest reporting postcode in WA is 6065, with 46 systems reporting. In addition, the most used

module type is the 'Suntech' module, while the most used inverter type is the 'SMA Sunny Boy 5000' inverter. The average module tilt in WA is 20.31° and the total installed system size is 3.49 MW.

Table 10: Summary of WA database analysis

WA							
Number of Systems Scraped		**Average System Size (W)**		**Largest System Sizes (W)**	**Number of Occurrences**	**Lowest System Sizes (W)**	**Number of Occurrences**
817		4279.87		55594	1	1075	1
				45000	1	1050	1
				35720	1	1020	1
				30000	1	1000	3
				27000	2	50	20
Most Used Inverter Size (W)	**Number of Occurrences**	**Most Used Inverter Type**	**Number of Occurrences**	**Highest Five Reporting Postcodes**	**Number of Occurrences**	**Most Used Module Rating (W)**	**Number of Occurrences**
5000	250	SMA Sunny Boy 5000	26	6065	46	190	209
3000	146	SMA Sunny Boy 5000TL	24	6164	38	250	204
4000	55	Energy Monitor	22	6155	36	200	37
2500	55	SMA Sunny Boy 3000	20	6112	19	195	34
2000	43	SMA	18	6169	18	235	32
Most Used Module Type	**Number of Occurrences**	**% of Systems Larger than 5 kW**	**Average Module Tilt (o)**	**Total State PV System Size Installed (MW)**			
Suntech	66	34.63	20.31	3.49			
REC	30						
Suntech STP190S-24/AD+	14						
Sunpower	13						
Suntellite	12						
REC	30						

4.2.6 Analysing Tasmania (TAS)

Figure 31 below shows a bubble map of the distribution of PV systems within TAS according to their system size.

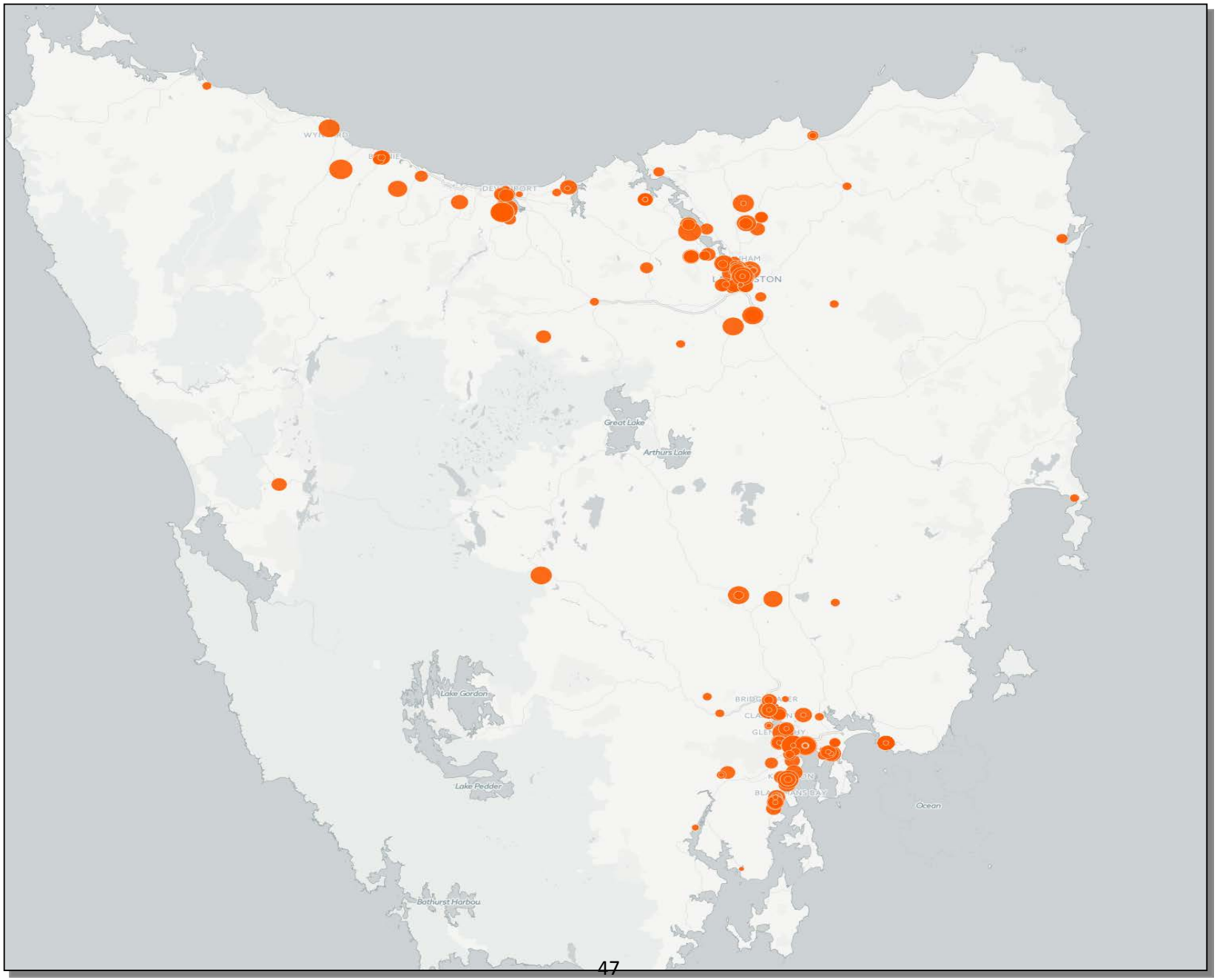


Figure 30: PV system distribution in TAS © OpenStreetMap contributors ©CartoDB CartoDB attribution

Figure 33 shows a scatter plot of the array size for all the sites scraped compared to the average system size in TAS, while Figure 34 shows a histogram of the distribution of the system sizes. It is worth noting that 53.54% of the systems are rated between 4000W-6000W.

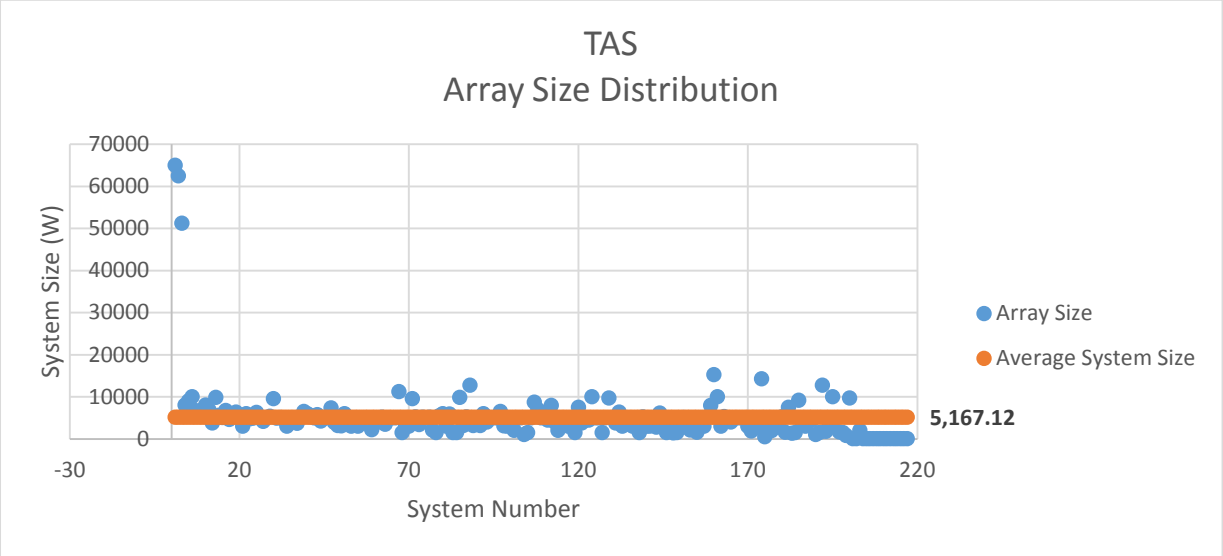


Figure 31: TAS system size distribution

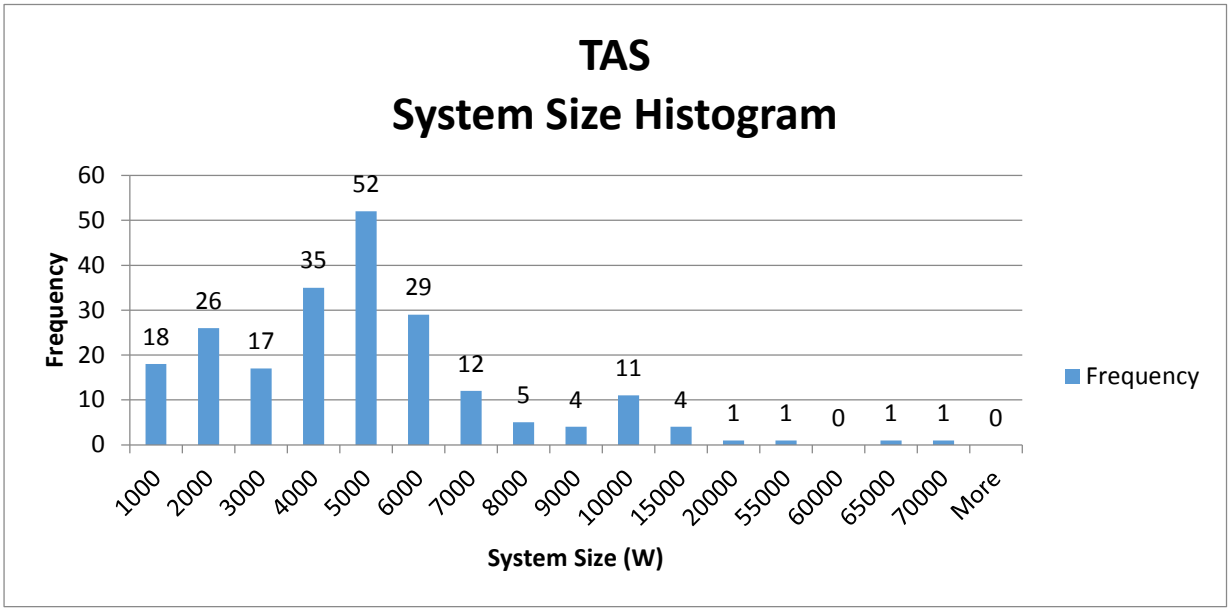


Figure 32: TAS System Size Histogram

Table 11 displays a summary of findings as a result of analysing the TAS database. The highest reporting postcode in TAS is 7250, with 32 systems reporting. In addition, the most used module type is the 'Rec' module, while the most used inverter type is the 'Growatt' inverter. The average module tilt in WA is 19.63° and the total installed system size is 1.12 MW.

Table 11: Summary of TAS database analysis

TAS							
Number of Systems Scraped		**Average System Size (W)**		**Largest System Sizes (W)**	**Number of Occurrences**	**Lowest System Sizes (W)**	**Number of Occurrences**
217		5167.12		65025	1	1040	1
				62475	1	1020	1
				51255	1	800	1
				15300	1	500	1
				14280	1	50	16
Most Used Inverter Size (W)	**Number of Occurrences**	**Most Used Inverter Type**	**Number of Occurrences**	**Highest Five Reporting Postcodes**	**Number of Occurrences**	**Most Used Module Rating (W)**	**Number of Occurrences**
5000	74	Growatt	10	7250	32	250	77
3000	23	SMA	8	7018	17	190	24
50	18	aurora	7	7310	14	200	17
4000	9	Energy Monitor	7	7050	11	50	16
6000	7	sma sunny boy 5000	6	7054 7011 7000	10	245	15
Most Used Module Type	**Number of Occurrences**	**% of Systems Larger than 5 kW**	**Average Module Tilt (o)**	**Total State PV System Size Installed (MW)**			
rec	24	46.54	19.63	1.12			
Suntech	7						
Cetc Astronergy Renesola Trina Honey CETC255W Daqo	4						
ASP-60-6M250 Sunways	3						

4.2.6 Analysing Northern Territory (NT)

Figure 34 below shows a bubble map of the distribution of PV systems within NT according to their system size.

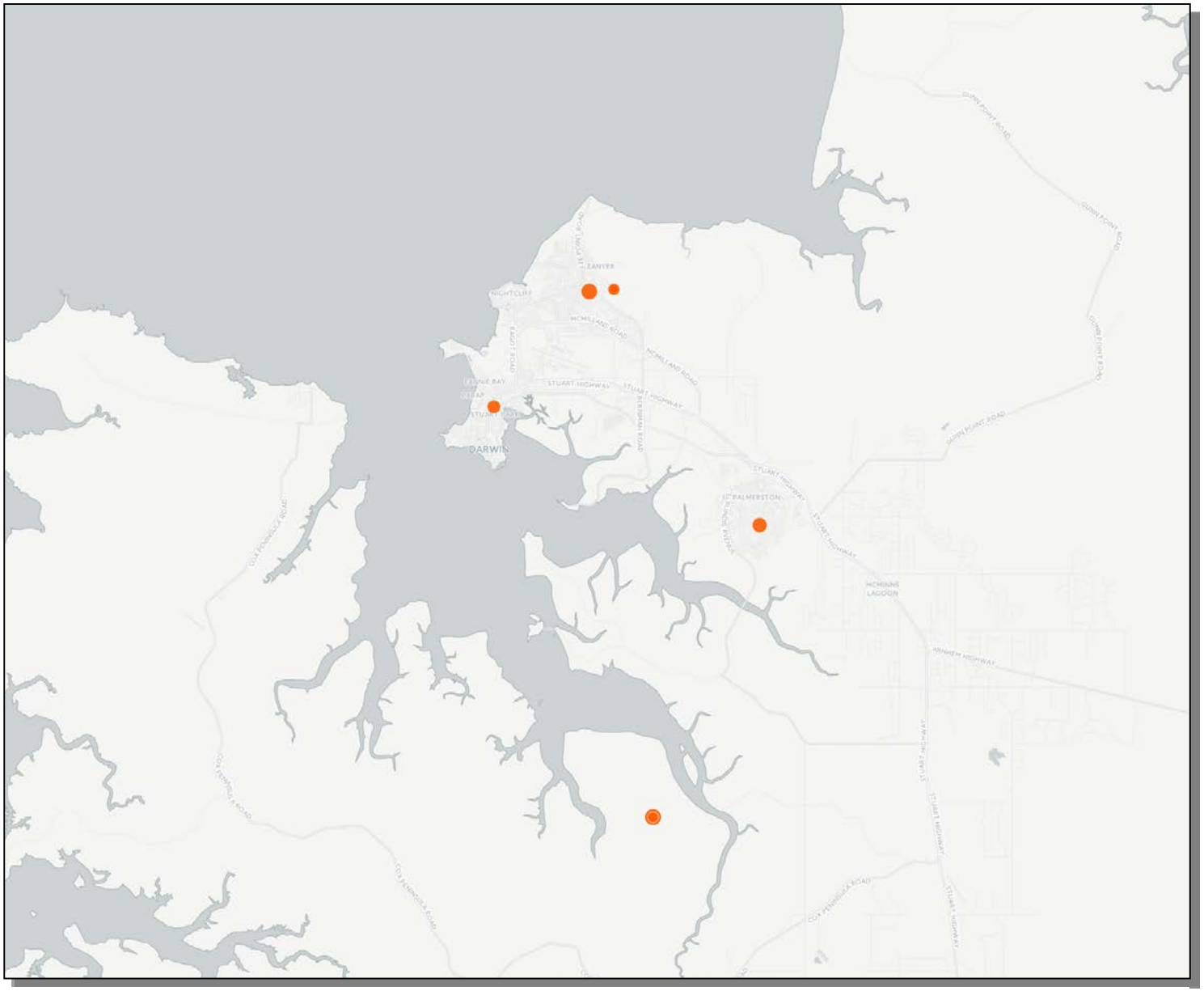


Figure 33: PV system distribution in NT © OpenStreetMap contributors ©CartoDB CartoDB attribution

Figure 35 shows a scatter plot of the array size for all the sites scraped compared to the average system size in NT, while Figure 36 shows a histogram of the distribution of the system sizes. The average system size is 4263.27 W, and the largest system size is 6120 W. Due to the small number of systems reporting from NT, a further detailed analysis was not conducted.

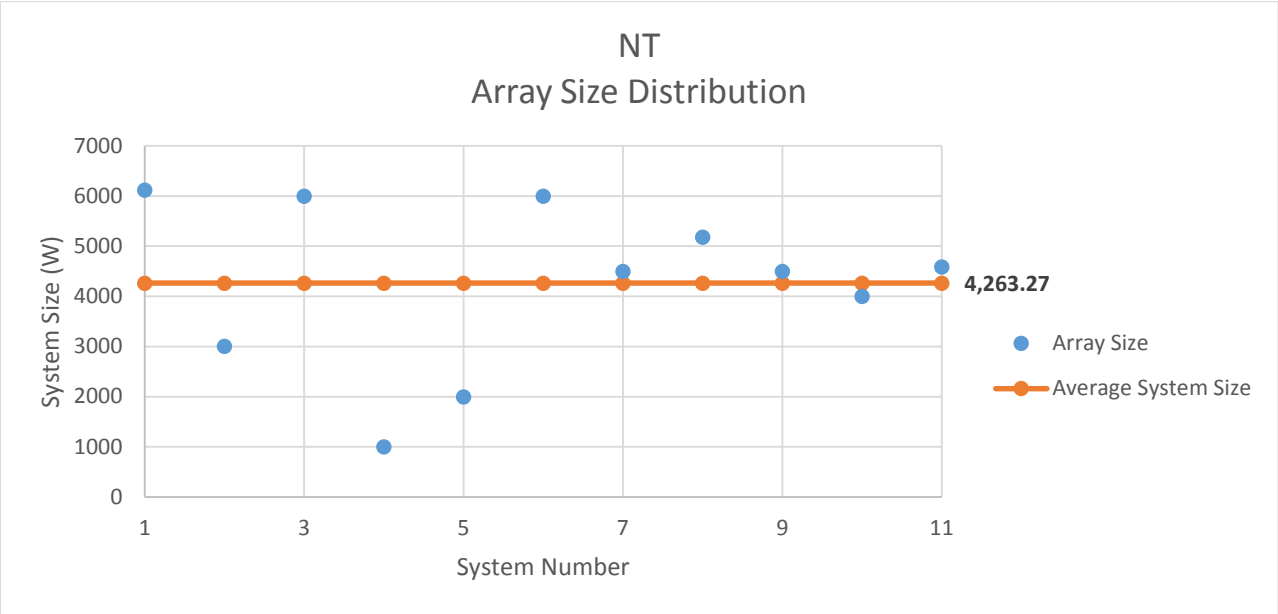


Figure 34: NT array size distribution

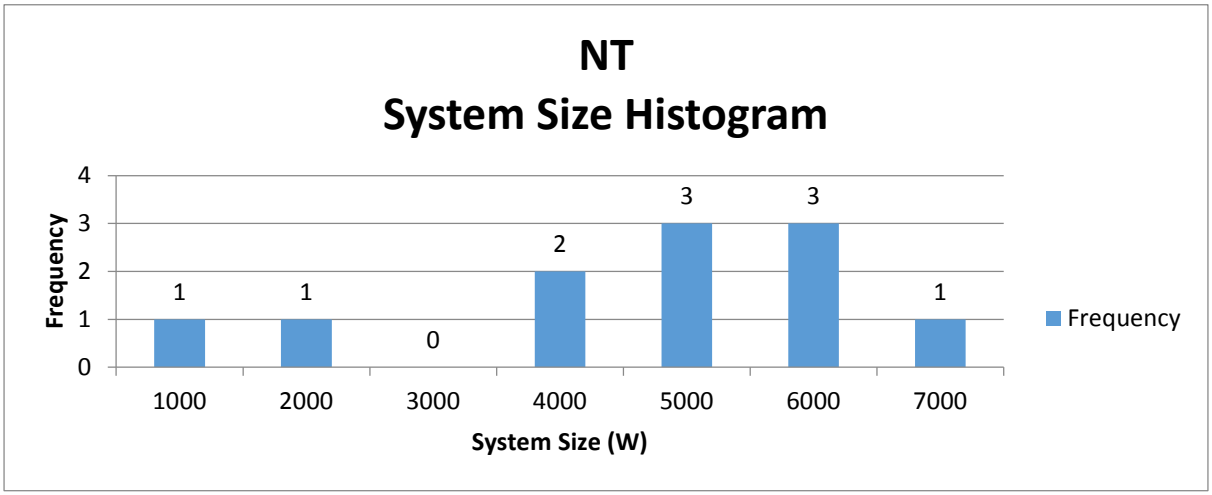


Figure 35: NT system size histogram

4.2.6 Analysing Australia

Figure 37 below shows a bubble map of the distribution of all the PV systems within Australia according to the system size.

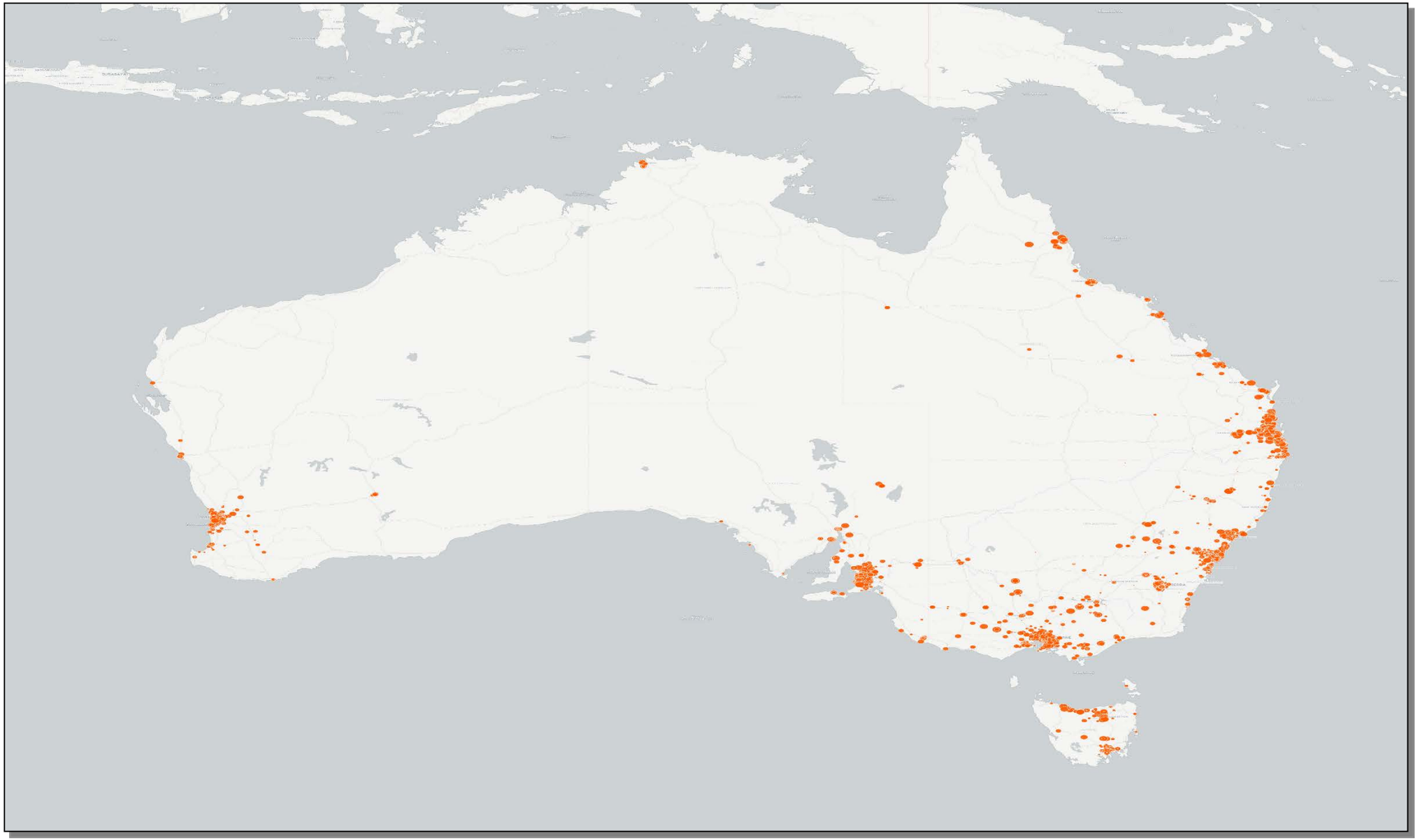


Figure 36: PV system distribution within Australia © OpenStreetMap contributors ©CartoDB CartoDB attribution

Table 12: Total system size installed for each state/territory

State	Total System Size Installed (MW)
ACT	1.67
NSW	3.61
VIC	2.87
SA	4.56
QLD	5.45
WA	3.49
TAS	1.12
NT	0.046
Total	**21.14**

Table 12 shows the total system sizes in each of states and territories that have been reported to 'pvoutput.org.

It is also found that the most used module types are the 'Suntech' module (reported 204 times), followed by the 'REC' module (reported 145 times), and the 'ET-M572200' module (reported 79 times). In addition, the most used inverter types were 'Trannergy PVI5400TL' (reported 275 times), followed by the 'SMA Sunny Boy 5000TL' (reported 110 times) and both the 'SMA sunny boy 5000' and 'Energy Monitor' (reported 98 times). Moreover, the postcode reporting the most PV sites is 6065, in WA (reporting 46 sites). Finally, the largest system reporting was rated at 100.8kW, located in QLD.

4.3 Using the data to scrape other relevant PV-performance and Metadata

The data obtained, especially the 'SID' for each PV site can be used in conjunction with the 'NickEngerer' R package to obtain ten minute interval data or hourly data of power generation of the systems throughout the dates the systems have been reporting to "pvoutput.org".

Chapter 5 Conclusion & Future Recommendations

5.1 Achievements

The work undertaken in this thesis has resulted in the successful development of a system that allows for the automatic extraction of data about PV systems installed in Australia from "pvoutput.org". In addition, the software system creates a database that contains the extracted data and will allow for further analysis to be conducted on PV systems installed in Australia.

The main challenges faced during the development process were attributed to the unavailability of a uniform method to extract all data types from the specified webpages. This is because each webpage has a different HTML structure, and the data contained within the HTML script can exist in various forms. To overcome this challenge, several approaches were integrated in order to create a comprehensive system that can extract data available in different formats. The combination of using DOMDocument instances, DOMXpath instances and Regex were found to be the most effective in obtaining the data required in this paper.

Another challenge faced was overcoming the repetitive data extraction for systems that have already been scraped using the software. This was overcome by importing data about systems that have been extracted previously and comparing the real-time extraction results with this data. The result of implementing this technique was a drastic reduction in simulation time as well as resources used. Finally, a Googlebot was implemented in order to overcome the limitation of the number of times the software is allowed to access "pvoutput.org" web-servers to extract data.

The project also made it possible for future engineering 'honours' projects to take place at the Australian National University. Currently, three 'honours' students from the department of Engineering at the Australian National University are undertaking research projects as a continuation of this project. The students will use the databases obtained by the automation

software, the KPV methodology developed by Engerer and Mills (4), and full city scale PV simulations to characterise collective PV ramp events in Melbourne, Sydney and Brisbane.

5.2 Future Recommendations

A future goal of this project is to include several web sources in the data extraction process, so as to include as many photovoltaic systems in Australia reporting to such web sources as possible. Moreover, a secondary future goal is to develop a user-interface for the software, such that a user can specify the target state or territory for the data extraction as well as the number of systems to be targeted etc… The data extracted could also be used in conjunction with the 'NickEngerer' R package to obtain 10 minute interval or hourly system power generation data for each of the systems scraped. The results could be used in future research projects to analyse the operation of photovoltaic systems installed in Australia.

The next step in analysing the data obtained, would be automating the process of interpreting the data itself, in order to obtain meaningful conclusions and visualizations that describe the data obtained.

This project could also be as a framework to create similar systems that can automate data extraction from the web. Whether the data is related to PV systems or other related scientific data-sets. In the future, it is envisaged that this software can be made more generic to include data extraction from several web sources that contain data about PV systems rather than just "pvoutput.org".

Bibliography

1. **PVOutput. [Online] 2015. [Cited: 2 6 2015.] pvoutput.org.**

2. *Monitoring of Photovoltaic Systems: Good Practrices and Systematic Analysis.* Achim Woyte, Mauricio Richter, David Moser, Stefan Mau, Nils Reich, Ulrike Jahn. Paris : PV SOLAR ENERGY CONFERENCE AND EXHIBITION, 2013.

3. Global PV Monitoring: Technologies, Markets and Leading Players, 2013-2017. *Greentech Media.* [Online] [Cited: 28 5 2015.] http://www.greentechmedia.com/research/report/global-pv-monitoring-2013-2017.

4. *KPV: A Clear Sky Index for Photovoltaics.* Nicholas Engerer, F. P. M. 2014.

5. *Experimental and data collection methods for a large-scale smart grid deployment: Method and first results.* Rhodes, J. D. U. C. R. H. C. B. M. C. M. W. D. A. N. P. A. …. W. M. E. s.l. : Energy, 2014, Vol. 65, pp. 462-471.

6. *Ota City: Characterizing Output Variability from 553 Homes with Residential PV Systems on a Distribution Feeder.* M. Lave, J. S. A. E. C. H. E. N. a. Y. M. Albuquerque : Sandia National Laboratories, 2011.

7. *Comparison of PV system performance-model predictions with measured pv system performance.* Christopher P. Cameron, Willian E. Boyson, Daniel M. Riley. ALbuquerque : Sandia National Laboratories, 2008.

8. Beal, Vangie. API - application program interface. [Online] [Cited: 4 5 2015.] http://www.webopedia.com/TERM/A/API.html.

9. PVOutput - Rate Limits. *PVOutput.* [Online] 2015. [Cited: 12 2 2015.] http://pvoutput.org/help.html#api-ratelimit.

10. *A study on competent crawling algorithm (CCA) for web search to enhance efficiency of information retrieva.* Saranya, S. ,. Z. B. ,. V. P. P. s.l. : Advances in Intelligent Systems and Computing, 2014, Vol. 325, pp. 9-16.

11. *Web Crawling.* Christopher Olston, M. N. s.l. : Foundations and Trends, 2010, Vol. 4(3), pp. 175-246.

12. Institute, Australian PV. Mapping Australian Photovoltaic installations. *apvi.org.au.* [Online] [Cited: 23 5 2015.] http://pv-map.apvi.org.au/historical#12/-35.3146/149.1545.

13. Mehdi Achour, Friedhelm Betz, Antony Dovgal, Nuno Lopes, Hannes Magnusson, Georg Richter, Damien Seguy, Jakub Vrana. *PHP Manual.* 2015.

14. The DOMDocument class. *php.* [Online] 2015. [Cited: 9 5 2015.] http://php.net/manual/en/class.domdocument.php.

15. The DOMXPath class. *php.* [Online] 2015. [Cited: 9 5 2015.] http://php.net/manual/en/class.domxpath.php.

16. Document Object Model (DOM). *World Wide Web Consortium (W3C).* [Online] 2015. [Cited: 9 5 2015.] http://www.w3.org/DOM/.

17. *HTML Tree.* http://www.w3schools.com/js/pic_htmltree.gif, s.l. : 2015.

18. Wiley, Nate. [Online] https://github.com/shmulim/XPATH/blob/master/Xpath.php.

19. Google. Google Crawlers. [Online] [Cited: 12 5 2015.] https://support.google.com/webmasters/answer/1061943?hl=en.

20. Engerer, Nicholas. NickEngerer_Rpkg_Manual_v1_Circa_June2014. *Github.* [Online] [Cited: 10 2 2015.] https://www.assembla.com/code/anusolar/git/nodes/master/man/nickengerer_Rpkg_manual_v1_circa_june2014.pdf.

Appendices

APPENDIX A – Scrape Code

```
1.      <?php
2.      require_once 'Xpath.php';
3.      set_time_limit(0);

4.      //Specify URL to scrape
5.      $startUrl =   "http://pvoutput.org/map.jsp?p=0&state=ACT";
6.      $startUrl_2 = "http://pvoutput.org/map.jsp?p=1&state=ACT";

7.      //create column headers
8.      $headers = "Title ; Sid; Latitude ; Longitude ; Number of
        Panels ; Panel Max Power ; System Size ; Panel Brand/Model ;
        Orientation ; Number of Inverters ; Inverter Brand/Model ;
        Inverter Size ; Postcode ; Install Date ; Shading ; Tilt ;
        Comments ; " . "\r\n";
9.      $fh = fopen("ACT_database.scsv", "a+");
10.     fwrite($fh, $headers);

11.     //to scrape page=0
12.     $xpath = new XPATH($startUrl);
13.     $linkHrefQuery = $xpath -> query("//tr/td[3]/a/@href");

14.     for ($x=0; $x<$linkHrefQuery->length; $x++)
        {////////////////start of for
15.     $array_1[$x]['linkHref']  = "www.pvoutput.org/display.jsp?".
        $linkHrefQuery ->item($x)->nodeValue;
16.     $array_1_b[$x]['linkHref']  = "www.pvoutput.org/listmap.jsp?".
        $linkHrefQuery ->item($x)->nodeValue;

17.     ////////////////////////////////////////////////////////////////
        ///
18.     $SID = $linkHrefQuery ->item($x)->nodeValue;
19.     $sid_regex= "/\bsid=\b(\S*)/";
20.     preg_match($sid_regex, $SID, $sid_values);
21.     //echo "<pre>";
22.     //echo $sid_values[1];


23.     //to check if sites already exist in the database
24.     //specify existing database file name in fopen("FILE NAME
        HERE", "r")
25.     //file has to be in CSV and in same directory of localhost
26.     $row =1;
27.     if (($handle = fopen("sites_info_CBR.csv", "r")) !== FALSE) {
28.     while (($data = fgetcsv($handle,1000,",")) !==FALSE) {
29.     $row++;
30.     $sid_import[$row]= $data[0];
31.     //$sid_values[row]= $data[1];
32.     //echo $data[1] . "<br />\n";
33.     }
34.     fclose($handle);
35.     }//end of importing sid values for pre-existing databse
36.     //echo $sid_values . "<br /> \n";
37.     //print_r($sid_import)  . "<br /> \n";
38.     //echo $sid_import[4];
39.     //$sid_values = "312";
```

```
40.       if (in_array($sid_values[1], $sid_import)){
41.       //echo "MATCH!";
42.       continue;
43.       }
44.       else

45.       //rest of code goes here
46.       ////////////////////////////////////////////////////////////
          ///
47.       $ch = curl_init($array_1[$x]['linkHref']);
48.       $ch_b = curl_init($array_1_b[$x]['linkHref']);

49.       curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
50.       $cl = curl_exec($ch);
51.       curl_setopt($ch_b, CURLOPT_RETURNTRANSFER, true);
52.       $cl_b = curl_exec($ch_b);

53.       $dom = new DOMDocument();
54.       $dom_b = new DOMDocument();
55.       @$dom ->loadHTML($cl);
56.       @$dom_b ->loadHTML($cl_b);

57.       $xpath = new DOMXpath($dom);=

58.       $formTitle = $xpath -> query("//b/text()");
59.       $form_out = $formTitle -> item(0) -> nodeValue . ";";
60.       $form_out_nl= '';

61.       $formVal = $dom -> getElementsByTagName("input");

62.       $fh = fopen("ACT_database.scsv", "a+");

63.       $Lat_data = $dom_b -> getElementsByTagName("script");
64.       $content =$Lat_data -> item(10) -> textContent;

65.       $regex = "/latitude: (\S*)/";
66.       preg_match( $regex , $content , $values );
67.       $latitude = $values[1]; //this just gives the latitude value :
          "-35.33"

68.       $regex2 = "/longitude: (\S*)/";
69.       preg_match( $regex2 , $content , $values2 );
70.       $longitude = $values2[1];

71.       $form_out =  $form_out . $sid_values[1] . ";" . $values[1] .
          ";" . $values2[1]. ";";

72.       foreach($formVal as $link) {
73.       //echo $link->getAttribute("value") . "<br>";
74.       $form_out .= $link->getAttribute("value") . ";";
75.       }

76.       $form_out_nl .= $form_out . "\r\n";

77.       fwrite($fh, $form_out_nl);
78.       echo "<br>";
79.       echo $form_out_nl;
80.       ////////////////////////////////////////////////////////
81.       }//end of for
82.       //_____________________________________________________________
          _____________________________//
```

```
83.     //to scrape all the other pages recursively until there is no
        input in the page
84.     function scrapePV($url){

85.     $array_2 = array();
86.     $xpath = new XPATH($url);

87.     //for the 20 sites on a page, get all the Href links
88.     $linkHrefQuery = $xpath -> query("//tr/td[3]/a/@href");
89.     for ($x=0; $x<$linkHrefQuery->length; $x++) {
90.     $array_2[$x]['linkHref']  = "www.pvoutput.org/display.jsp?".
        $linkHrefQuery ->item($x)->nodeValue;
91.     $array_2_b[$x]['linkHref']  = "www.pvoutput.org/listmap.jsp?".
        $linkHrefQuery ->item($x)->nodeValue;
92.     ////////////////////////////////////////////////////////
93.     $SID_2 = $linkHrefQuery ->item($x)->nodeValue;
94.     $sid_2_regex= "/\bsid=\b(\S*)/";
95.     preg_match($sid_2_regex, $SID_2, $sid_2_values);



96.     //to check if sites already exist in the database
97.     //specify existing database file name in fopen("FILE NAME
        HERE", "r")
98.     $row =1;
99.     if (($handle = fopen("sites_info_CBR.csv", "r")) !== FALSE) {
100.    while (($data = fgetcsv($handle,1000,",")) !==FALSE) {
101.    $row++;
102.    $sid_import[$row]= $data[0];
103.    //$sid_values[row]= $data[1];
104.    //echo $data[1] . "<br />\n";
105.    }
106.    fclose($handle);
107.    }//end of importing sid values for pre-existing databse
108.    //echo $sid_values . "<br /> \n";
109.    //print_r($sid_import)  . "<br /> \n";
110.    //echo $sid_import[4];
111.    //$sid_values = "312";
112.    if (in_array($sid_2_values[1], $sid_import)){
113.    //echo "MATCH!";
114.    continue;
115.    }
116.    else

117.    ////////////////////////////////////////////////////////
118.    $ch_2 = curl_init($array_2[$x]['linkHref']);
119.    $ch_2_b = curl_init($array_2_b[$x]['linkHref']);
120.    curl_setopt($ch_2, CURLOPT_RETURNTRANSFER, true);
121.    $cl_2 = curl_exec($ch_2);
122.    curl_setopt($ch_2_b, CURLOPT_RETURNTRANSFER, true);
123.    $cl_2_b = curl_exec($ch_2_b);

124.    $dom_2 = new DOMDocument();
125.    @$dom_2 ->loadHTML($cl_2);
126.    $dom_2_b = new DOMDocument();
127.    @$dom_2_b ->loadHTML($cl_2_b);

128.    $xpath2 = new DOMXpath($dom_2);
```

```
129.     $formTitle_2 = $xpath2 -> query("//b/text()");//to add the
         title to the beginning of the form
130.     $form_out_2 = $formTitle_2 -> item(0) -> nodeValue . ";";
131.     $form_out_2_nl= '';

132.     $formVal_2 = $dom_2 -> getElementsByTagName("input");
133.     $fh = fopen("ACT_database.scsv", "a+");

134.     $Lat_data_2 = $dom_2_b -> getElementsByTagName("script");
135.     $content_2 =$Lat_data_2 -> item(10) -> textContent;

136.     $regex_2 = "/latitude: (\S*)/";
137.     preg_match( $regex_2 , $content_2 , $values_2 );
138.     $latitude_2 = $values_2[1]; //this just gives the latitude
         value : "-35.33"

139.     $regex2_2 = "/longitude: (\S*)/";
140.     preg_match( $regex2_2 , $content_2 , $values2_2 );
141.     $longitude_2 = $values2_2[1];

142.     $form_out_2 =  $form_out_2 . $sid_2_values[1] . ";" .
         $values_2[1] . ";" . $values2_2[1]. ";";

143.     foreach($formVal_2 as $link) {
144.     //echo $link->getAttribute("value") . "<br>";
145.     $form_out_2 .= $link->getAttribute("value") . ";";
146.     }
147.     $form_out_2_nl .= $form_out_2 . "\r\n";
148.     fwrite($fh, $form_out_2_nl);
149.     echo "<br>";
150.     echo $form_out_2_nl;
151.     ////////////////////////////////////////////////////////

152.     }

153.     $nextPageQuery = $xpath ->
         query("(//div[@id='tnt_pagination']/a/@href)[24]"); //goes to
         the next page/index is 24
154.     $contentQuery=$xpath ->
         query("(//td/a[@class='system1']/text())[1]");
155.     //echo $nextPageQuery->item (0)->nodeValue;
156.     //print_r($nextPageQuery);
157.     // print_r($nextPageQuery->item(0)->nodeValue);
158.     if (($contentQuery->length)!=0){

159.     $lastPageQuery = $xpath -> query("//tr/td[3]/a/@href"); //the
         number of hrefs in a page, the last page will have none
160.     if (($lastPageQuery->length) > 18) {
161.     $nextUrl = "http://pvoutput.org/map.jsp" . $nextPageQuery-
         >item(0)->nodeValue;
162.     $array_2 =  array_merge($array_2, scrapePV($nextUrl));
             }
163.     }
164.     return $array_2;
165.     }
166.     // //display the data
167.     $data = scrapePV("$startUrl_2");
168.     //echo"<pre>";
169.     //$array_all = array_merge($array_1,$data);
170.     //print_r($array_all);
```

APPENDIX B – XPATH.PHP

```
1.      <?php
2.      //Copyright © 2013 Nate Wiley

3.      class XPATH {


4.      public $dom, $xpath;

5.      public function __construct($url){
6.      $html = $this ->_curl($url);//use the function created below
        (_curl) to save the curled url into the variable html
7.      $this->dom = new DOMDocument();//create  a new domdocument
8.      @$this->dom ->loadHTML($html);//load the html of the obtained
        url into the dom document
9.      $this->xpath = new DOMXpath($this->dom);

10.     }
11.     public function query($q){
12.     $result = $this->xpath->query($q);
13.     return $result;
14.     }

15.     public function preview($results){
16.     echo"<pre>";
17.     print_r($results);
18.     echo "<br>NOde Values <br>";
19.     foreach ($results as $result) {
20.     echo trim($result->nodeValue) . '<br>';
21.     $array[] = $result;
22.     }
23.     echo "<br><br>";
        print_r($array);
24.     }

25.     private function _curl($url){
26.     $ch = curl_init();
27.     curl_setopt($ch, CURLOPT_URL, $url);
28.     curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);//store url
        fetched by curl into a variable
29.     curl_setopt($ch, CURLOPT_USERAGENT, "Mozilla/5.0 (compatible;
        Googlebot/2.1; +http://www.google.com/bot.html)");
30.     curl_setopt($ch, CURLOPT_AUTOREFERER, true);
31.     curl_setopt($ch, CURLOPT_FOLLOWLOCATION, true);

32.     $result = curl_exec($ch);
33.     if(!$result){
34.     echo "<br />cURL error number:" .curl_errno($ch);
35.     echo "<br />cURL error:" .curl_error($ch) . "on URL - " . $url;
36.     var_dump(curl_getinfo($ch));
37.     var_dump(curl_error($ch));
38.     exit;
39.     }
40.     return $result;
41.     }
42.     }
```