Developer’s guide

1 — Last update: 2017/07/26

katarzynan

Table of Contents

1. Introduction

The basic features of this program include:
1. accepting user input
2. calculating response according to Power Control Algorithm v2 (PCAv2)
3. printing commands to the terminal
4. saving measurements and calculation into the database
5. creating a visualization graph

The PCAv2 algorithm tries to increase or decrease MS or BTS power to match target for signal strength. Hysteresis value is used to define range of signals when correction is not needed. If measured value is above target instruction to decrease power is sent.

The input file may be generated used applications built-in module (data_generator.py). The input data should be served in the following order:

DL |S0 | MS776 |-66 |1
directon of transmission|cell identity | MS identity |signal strength|signal quality

It is possible for some arguments to be missing in the input file. This is not a problem for the algorithm though.
- If measured value is below target instruction to increase power is sent.
- If measured value falls into expected range no action is required:
target-1dB <= signal <= target+1dB

The value of measurement matched with target is calculated of 1 to 8 last measurement reports.
The PCA2 does not send command when number of measurements collected is less than 4. (beginning of communication). It may happen that some measurements are missing – they should arrive every 500ms, if not, the system interpolates:
- first missing measurements – value of previous one
- number of allowed missing measurement recovered as previous value is configurable
from 1 to 3
- next CONSECUTIVE missing measurements are replaced with value -95dBm
- weight of each next measurement (first in calculations is last received) is dropping, if last measurement has weight 1, than one before last ½ , next ¼, etc.
- when received (average) signal is within hysteresis difference from target, algorithm does not stop, but maximum step is 1 dB, algorithm send NCH command when measurement to target is less than 1dB

In an input file there is a quality parameter in a last column. This input should have an impact on algorithm behavior. Quality, from 0 to 5, indicates the level of errors (Bit Error Rate), and this measurement may override decisions made on power measurements only.

Procedure: Quality is processed similar way as power MR:
• average is calculated over (1..8) window with weights.
• all missing Q MRs are considered = 5.

1) if Q is less than 2, algorithm works normally
2) if Q is in range of < 2,4>, algorithm is not allowed to decrease power. If that would be decision from power MR no change is used instead, increase is executed normally.
3) if Q is 4 or higher, decision is always to increase power, by minimum 2 dB or bigger if such calculated from power MR

Input file contains mixture of measurement, in columns:
— direction of transmission:
- DL, downlink, measurement comes from MS, concerns specific cell (S0, N1, N2…), commands are send for BTS of S0 cell only
- UL, uplink, measurements come from BTS, concern serving cell (S0) and mobile terminal (MS)
— cell identity: S0 (serving cell), Nx (neighbor cell, not used in PCA2)
— MS identity, e.g. MS222, MS667 etc
— Signal strength e.g. -78 [dBm], correct values -45..-95
— Signal quality (0..5) representing quality measured by error rate, measured only for serving cell (S0)

No access rights required for the program. The program is open source software.

1.1. Who should read this document

By reading this document you will get to know how to use **Python_BTS_Project** to calculate correct response (increase, decrease or do not change MS or BTS power to match target for signal strength) and save the calculated data, as well as the input into the database. You will get to know how to set up the project on different system, but the main objective of this document is to familiarize you with the project's structure, modules, functionality to allow you to develop this product.

1.2. Providing feedback

In the case you have any questions regarding the project and you can not find the answers in the project’s documentation please contact one of the developers:
Teresa.Bury[at]ust-global.com
Piotr.Bednarz[at]ust-global.com
Patrycja.Brzeska[at]ust-global.com
Katarzyna.Fedevych[at]ust-global.com

2. Configuration and setup

The basic requirements include:
1. Operating systems: Windows, Linux or OsX, with at least 2GB RAM,
2. Python 2.x and 3.x interpreters,
3. Gnuplot

Please select your platform and continue with the instructions.
Linux
Windows
OsX

2.1. Windows

Installation and setup instructions:

Download the program files from https://github.com/peterb91/Python_BTS_Project.git, and save them on your local drive.

Open cmd line:
Ensure you have installed python 2.x and 3.x interpreters on your station (command: *where python*).
If python 2.x or 3.x is missing, download required version of python from: https://www.python.org/getit/ and install it following instructions on the website.
Ensure you have Flask installed (follow this guide: http://flask.pocoo.org/docs/0.12/installation/#windows-easy-install)
Ensure you have requests installed (command: *python3 -m pip install requests*)

Install additional program Gnuplot for graph creation:
download gp426win32.zip from the following site:http://sourceforge.net/projects/gnuplot/files/

Extract the file into "Name of folder". Open "My Computer" and find the file wgnuplot. It should be somewhere like: C:/Documents and Settings/username/Name of folder/gnuplot/bin
click on the icon for gnuplot.exe and install the program. During the installation you should check the "Add application directory to your PATH environment variable" option.
now you can use gnuplot in cmd line using command: *gnuplot*

Go to the directory where all files were downloaded (command: cd directory_name_path eg. cd C:/Documents and Settings/username/Python_BTS_Project.

Before running the algorithm, you are able to modify basic configuration file – config_file.txt, where you can modify target value, hysteresis, maximum increase and max decrease values and other details, as well as you can set if the program should send the data to the database via http or not (it is much faster with http disabled).

This application enables saving data to the databes via http. To make it work:
1. use configuration file and set http to 1:
http = 1
2. start local server by running http_server.py (python3 http_server.py) in a separate terminal window (this script must be running for the whole time). Then, when you have server running start application in a normal way.
If you want to change the name of the database created via http please navigate to http_server.py and

change the path on the 6th line. If you are using directory outside tmp please make sure it exists first. Another thing is to restart the server every time you make some modifications in the file.

Run the program in cmd console using command: *type file.txt* | *python3 source.py* (where “file.txt” is file with data from terminals and stations and “source.py” is running file included in program pack.

Uninstallation

Remove all files from the folder where the program was downloaded/copied (right click and delete).

2.2. Linux

Installation and setup instructions:

Download the program files from https://github.com/peterb91/Python_BTS_Project.git, and save them on your local drive.

Open terminal:

Ensure you have installed python 2.x and 3.x interpreters on your station (command:_ whereis python_).
Ensure you have Flask installed globally (command:_ pip3 install flask_)
Ensure you have Requests installed (command: *pip3 install requests*)

If python 2.x or 3.x is missing, download required version of python from:" https://www.python.org/getit": https://www.python.org/getit/ and install it following instructions on the website. For linux – ubuntu installation use command:_ sudo apt-get install pythonx_.x (where x.x is version)

Install additional program for graph creation (command: *sudo apt-get install Gnuplot*)

Go to the directory where all program files are stored cd folder_name_path eg cd /usr/ Python_BTS_Project/

Before running the algorithm, you are able to modify basic configuration file – config_file.txt, where you can modify target value, hysteresis, maximum increase and max decrease values and other details, as well as you can set if the program should send the data to the database via http or not (it is much faster with http disabled).

This application enables saving data to the databes via http. To make it work:
1. use configuration file and set http to 1:
http = 1
2. start local server by running http_server.py (python3 http_server.py) in a separate terminal window (this script must be running for the whole time). Then, when you have server running start application in a normal way.
If you want to change the name of the database created via http please navigate to http_server.py and change the path on the 6th line. If you are using directory outside tmp please make sure it exists first. Another thing is to restart the server every time you make some modifications in the file.

Run the program using command: *cat file.txt* | *python3 source.py* (where "file.txt" is file with data from terminals and stations and "source.py" is running file included in program pack).

The most important is to make sure that file.txt exists and contains input data.at file.txt exists and contains input data.

Uninstallation

1. Remove all files from directory where the program was downloaded/copied (terminal command: *rm _) or remove directory if only program files are inside (command: _rm -r*)

2.3. OsX

Installation and setup instructions:

Clone the repository" https://github.com/peterb91/Python_BTS_Project.git": https://github.com/peterb91/Python_BTS_Project.git to any directory (command: git clone "https://github.com/peterb91/Python_BTS_Project.git":https://github.com/peterb91/Python_BTS_Project.git)
Open terminal:
Ensure you python 2.x and 3.x interpreters installed on your station (command: *whereis python*).
Ensure you have Flask installed globally (command:_ pip3 install flask_)
Ensure you have Requests installed (command: *pip3 install requests*)

If python 2.x or 3.x is missing, download required version of python fromhttps://www.python.org/getit/ and install it following instructions on the website. If you are using homebrew you can use commands_ brew install python_ and *brew install python3*

Install additional program for graph creation – Gnuplot – from binary" http://ricardo.ecn.wfu.edu/pub/gnuplot/": http://ricardo.ecn.wfu.edu/pub/gnuplot/ or using homebrew: brew install gnuplot

Go to the directory where all program files are stored *cd folder_name_path* eg cd /usr/Python_BTS_Project/

Before running the algorithm, you are able to modify basic configuration file – config_file.txt, where you can modify target value, hysteresis, maximum increase and max decrease values and other details, as well as you can set if the program should send the data to the database via http or not (it is much faster with http disabled).

This application enables saving data to the databes via http. To make it work:
1. use configuration file and set http to 1:
http = 1
2. start local server by running http_server.py (python3 http_server.py) in a separate terminal window (this script must be running for the whole time). Then, when you have server running start application in a normal way.
If you want to change the name of the database created via http please navigate to http_server.py and change the path on the 6th line. If you are using directory outside tmp please make sure it exists first. Another thing is to restart the server every time you make some modifications in the file.

Run the program using command: *cat file.txt* | *python3 source.py* (where "file.txt" is file with data from terminals and stations and "source.py" is running file included in program pack).
The most important is to make sure that file.txt exists and contains input data.

Uninstallation

Remove all files from directory where the program was downloaded/copied (terminal command: *rm _) or remove directory if only program files are inside (command: _rm -r*)

3. Application structure

This project is broken down a project into smaller modules that make the program more organized, thus make it easier to maintain by the developer.
Instead of putting everything into one big Python file, each specific task is put into its own module, and the result is assembled at the end. The description of each module can be found in the following subchapters.
algorithm.py
config.py
data_generator
database.py
graph_creation.py
input.py
input_charts.py
source.py
http_server.py

3.1. algorithm.py

This module performs fundamental calculation of the data given in the input file.

Functions:

avg_qual(numbers)
Calculates list of given n numbers into weighted average by following formula:

$$\text{value[0]} + \sum_{i=1}^{n-1} \text{value[i]} * \frac{1}{i * 2}$$

and returns the result.

avg(numbers)
It is returning difference between target signal strength and calculated average from previous function.

power_management(data)
Function is taking list of lists of five with given data and changes it into list of lists of five (or four if no change is applied) returning instructions for device if it should increase or decrease power consumption in order to improve performance. Also it is supporting handover function calculating if signal should change terminal according to last (max of 6) neighbors data and comparing it to given terminal (S0) signal strength with a margin of 3 points exclusively.

3.2. config.py

This module parses the config file, so it could be used by the algorithm.py module.

Functions:

read_config():
this function performs the main activity of the module by parsing the input data. The input file is is opened inside the function body, and then parsed and returned as a list.

3.3. data_generator.py

The main purpose of this function is to generate pseudo random input data. This module is not needed by the main algorithm to run, but it can be of a great help if you plan to extend the functionality of the module.

Functions:

random_data()
This function generates pseudo random data in a format:

DL	S0	MS776	-66	1
direction of transmission	cell identity	MS identity	signal strength	signal quality

It is possible that data will be missing in some of the fields – this is intended behavior as it helps with debugging the script and observing its behavior in different cases.

3.4. database.py

This module consists three Database classes:
BaseArchiver() – which is a base class including constants and saving functions
DatabaseArchiver() – which creates database and performs inserting into the database
HTTPArchiver() – which sends input and calculated output to the database via http server.

BaseArchiver functions:

save_measurement()
this function saves single measurement into the list, which is flushed in next steps

save_response
this function saves single response (calculated output), which is flushed in next steps

flush()
flushes remaining records into the database if the buffering is involved

DatabaseArchiver functions:

create_schema():
this function creates the schema if it was not created before.

create_schema_needed():
this function checks if the shema was created before. If not, it returns True so the create_shema() function could set up a new database.

flush_measurements():
performs actual database insert for measurements (saving without http sever)

flush_responses():
performs actual database insert for responses (saving without http sever)

DatabaseArchiver functions:

flush_measurements():
performs actual database insert for measurements (saving via http sever)

flush_responses():
performs actual database insert for responses (saving via http sever)

3.5. graph_creation.py

This module creates .png file with the graph and needs input_charts.py file to work correctly.

Functions:

graph_creation():
this function creates png files with the chart, taking given parameters.

3.6. input.py

The main purpose of this module is reading input file and prepare it for the calculation.

Functions:

read_file()
This function reads the data from the file and stores it in the list for further use of the algorithm.
Moreover input data are checked, additional data (columns) are removed, missed data (except of signal strength) are assigned to None value.
In case of signal strength if value for it is 'missed' then integer number 1000 is assigned. If no integer value or lack of data provided then function assigned for signal strength out of proper range value 9999.

3.7. input_charts.py

This module creates txt files so the graph charts could be created by the graph_creation.py module.

Functions:

input_charts():
the function creates text files for further use by the chart generating module.

3.8. save_output_txt.py

The main purpose of this module is to save calculated output in separated files, using MS names as distinguishers.

Functions:

write_to_text()
This functions takes calculated output, creates new txt files by MS name and saves calculations regarding given MS to this file. This is a helper module which can be useful while debugging and developing the application.

3.9. source.py

This module is responsible for handling the whole program. It consists of multiple imports of used modules and then creates instance of DatabaseArchiver class, creates database in given directory and writes data to the database. If the 'http' option is set to 1 in the config file, this module needs http_server.py to be running in order to save data via http server.

Created database consists of two tables: measurements and responses. The table measurements keeps the input data, and the responses table keeps calculated output. You create new database by creating new class instance (example: archiver = DatabaseArchiver('/tmp/BTS.db'). By default /tmp/ directory will be used. If you want to set the database in different directory make sure it exists first.

3.10. http_server.py

This module creates Flask local http server for receiving and saving data into the database.

Functions:

populate_measurements()
this function is responsible for saving received measurements to the database

populate_response()
this function is responsible for saving received responses to the database

The database path can be changed on the line 6. If you want to use path other than /tmp/ make sure it exists first.

If you want to make any change in this file please make sure to restart the server after the changes are made.

4. Conventions

This project provides a .editorconfig file to help you stick to programming conventions. Please follow the editorconfig documentation regarding install of plugin for the IDE you are using -http://editorconfig.org/#download. If you are using PyCharm, IntelliJ, Webstorm, BBedit there is no need to install additional plugins and you are good to go as soon as you set up the project locally.

If you don't want to use .editorconfig file or you don't want to be bothered with installing additional plugins please just follow PEP8 coding standards while developing this application. The description of the standard can be found on Python's website, but the very basic rules you should stick to are:

- indentation: 4 spaces (no tabs)
- naming: modules_and_packages, functions_and_methods, local_variables, GLOBALS, CONSTANTS, MultiWordClasses
- code line length should be limited to 120 characters
- Use UTF-8 encoding
- Always indent wrapped code for readablility
- Don't use wildcards
- Try to use absolute imports over relative ones
- When using relative imports, be explicit (with .)
- Don't import multiple packages per line2 blank lines before top-level function and class definitions
- 1 blank line before class method definitions
- Use blank lines in functions sparingly
- Avoid extraneous whitespace
- 2 blank lines before top-level function and class definitions
- 1 blank line before class method definitions
- Use blank lines in functions sparingly
- Avoid extraneous whitespace
- Keep comments up to date – incorrect comments are worse than no comments

5. Getting help

If you have problems or need help with this application we recommend to check for solution in project’s documentation first. The “Prerequisites and installation instruction”, “User guide” and “Developer guide” you are reading now were prepared for your convenience and should be the first source of information.
In the case you cannot find the solution in the documents please write to one of the developers:
Teresa.Bury[at]ust-global.com
Piotr.Bednarz[at]ust-global.com
Patrycja.Brzeska[at]ust-global.com
Katarzyna.Fedevych[at]ust-global.com
or create new GitHub issue.