api

API Documentation

September 6, 2019

Contents

1 Package command_ap

1.1 Modules

- **cmd** *(Section 2, p. 3)*
  - **command_ap** *(Section 3, p. 4)*
  - **ifconfig**: converts the output of ifconfig into a dictionary *(Section 4, p. 9)*
  - **iwconfig**: convert the output of iwconfig into a dictionary *(Section 5, p. 10)*
  - **scan**: convert the output of iw dev station dump into a dictionary *(Section 6, p. 11)*
  - **station**: convert the output of iw dev station dump into a dictionary *(Section 7, p. 12)*
  - **survey**: convert the output of iw dev station dump into a dictionary *(Section 8, p. 14)*
  - **xmit**: Module xmit *(Section 9, p. 15)*
- **get_set** *(Section 10, p. 16)*
  - **client**: the server accepts requests from an http client. *(Section 11, p. 17)*
  - **server**: server that accepts requests from an http client used to send commands to the AP *(Section 12, p. 18)*
  - **server_ffox**: The client (firefox) sends the following json data: *(Section 13, p. 24)*
  - **teste**: Test to get the data to compute: MOS client, hybrid and AP *(Section 14, p. 26)*
- **publisher_subscriber** *(Section 15, p. 27)*
  - **publisher** *(Section 16, p. 28)*
  - **subscriber** *(Section 17, p. 29)*
- **rl** *(Section 18, p. 30)*
  - **agent**: runs the agent: python3 agent.py *(Section 19, p. 31)*
  - **app1** *(Section 20, p. 35)*
  - **basic** *(Section 21, p. 36)*
    - **environment** *(Section 22, p. 37)*
  - **mab**: This module define three abstract MAB agents: * RandomAbstract: select random actions * EpsilonGreedyAbstract: select action using an epsilon-greedy policy * UCBAbstract: selects actions based on the UCB policy *(Section 23, p. 38)*
  - **model**: This module calculates the QoS based on the features *(Section 24, p. 45)*
  - **reward**: runs the agent: python3 agent.py *(Section 25, p. 46)*

1.2 Variables

Name	Description
__package__	**Value:** `None`

2 Package command_ap.cmd

2.1 Modules

- **command_ap** *(Section 3, p. 4)*
- **ifconfig**: converts the output of ifconfig into a dictionary
  (Section 4, p. 9)
- **iwconfig**: convert the output of iwconfig into a dictionary
  (Section 5, p. 10)
- **scan**: convert the output of iw dev station dump into a dictionary
  (Section 6, p. 11)
- **station**: convert the output of iw dev station dump into a dictionary
  (Section 7, p. 12)
- **survey**: convert the output of iw dev station dump into a dictionary
  (Section 8, p. 14)
- **xmit**: Module xmit
  (Section 9, p. 15)

2.2 Variables

Name	Description
__package__	**Value:** `None`

3 Module command_ap.cmd.command_ap

3.1 Functions

get_xmit(*phy_iface*=`'phy0'`)

get data from the xmit file. looks for it in /sys/kernel/debug/ieee80211/ath*/xmit

Return Value

the xmit fields

(type=dict)

get_ifconfig(*interface*, *path_ifconfig*=`__PATH_IFCONFIG`)

get data from ifconfig <interface>.

Parameters

`interface`: the wireless interface name, e.g. wlan0

`path_ifconfig`: path to ifconfig

Return Value

the ifconfig fields

(type=dict)

get_iw_stations(*interface*, *path_iw*=`__DEFAULT_IW_PATH`)

executes "iw station dump"

Parameters

`interface`: the wireless interface name, e.g. wlan0

`path_iw`: path to iw

Return Value

the command fields

(type=dict)

get_status(*path_hostapd_cli*=`__DEFAULT_HOSTAPD_CLI_PATH`)

get information from "hostapd_cli status" TODO: what if the interface has multiple SSIDs ???

Parameters

`path_hostapd_cli`: path to hostapd_cli

Return Value

the returned command fields

(type=dict)

change_channel(*interface*, *new_channel*, *count*=1, *ht_type*=`None`, *path_hostapd_cli*=`__DEFAULT_HOSTAPD_CLI_PATH`)

```
set the AP's channel using "hostapd_cli chan_switch" command.
TODO: add other optional parameters
      [sec_channel_offset=] [center_freq1=] [center_freq2=] [bandwidth=] [blocktx]

@param interface: the wireless interface name, e.g. wlan0
@param new_channel: the new channel number. Trying to change to the current channel returns an erro
@param ht_type: Valid values are ['', 'ht', 'vht']. Defines the type of channel. Invalid type retu
@param path_hostapd_cli: path to hostapd_cli

@return: the ifconfig fields
@rtype: dict
```

get_stations(*path_hostapd_cli*=`__DEFAULT_HOSTAPD_CLI_PATH`)

returns information about all connected stations

Parameters

`path_hostapd_cli`: path to hostapd_cli

Return Value

dictionary of dictionary

get_iw_info(*interface*, *path_iw*=`__DEFAULT_IW_PATH`)

executes "iw dev info"

Parameters

`interface`: the wireless interface name, e.g. wlan0

`path_iw`: path to iw

Return Value

the command fields

(type=dict)

get_iwconfig_info(*interface*, *path_iwconfig*=`__DEFAULT_IWCONFIG_PATH`)

get the return from "iwconfig <interface>" NOTE: this method only supports (tested) two modes = Managed and Master

Parameters

`interface`: interface to change

`path_iwconfig`: path to iwconfig

Return Value

the command fields

(type=dict)

get_power(*interface*, *path_iw*=`__DEFAULT_IW_PATH`, *path_iwconfig*=`__DEFAULT_IWCONFIG_PATH`)

get the power in the interface (from a station or AP)

Parameters

`interface`: interface to change

`path_iw`: path to iw

Return Value

the command fields

(type=dict)

set_iw_power(*interface*, *new_power*, *path_iw*=`__DEFAULT_IW_PATH`)

command dev <devname> set txpower <auto|fixed|limit> [<tx power in mBm>] NOTE: this module needs to run as superuser to set the power

Parameters

`interface`: interface to change

`new_power`: can be a string 'auto', or a number (int or float) that represents the new power in dBm

`path_iw`: path to iw

Return Value

if the command succeded

disassociate_sta(*mac_sta*, *path_hostapd_cli*=`__DEFAULT_HOSTAPD_CLI_PATH`)

sends the command to disassociate a station

Parameters

`mac_sta`: the MAC address of the station we want to disassociate

Return Value

if the command succeded

(type=bool)

get_config(*path_hostapd_cli*=`__DEFAULT_HOSTAPD_CLI_PATH`)

executes "hostapd_cli get_config"

Parameters

`path_hostapd_cli`: path to hostapd_cli

Return Value

dictionary {'ssid': 'ethanolQL1', 'bssid': 'b0:aa:ab:ab:ac:11', 'rsn_pairwise_cipher': 'CCMP', 'group_cipher': 'CCMP', 'key_mgmt': 'WPA-PSK', 'wpa': '2', 'wps_state': 'disabled'}

get_iw_survey(*interface*, *path_iw*=`__DEFAULT_IW_PATH`)

executes command "iw dev <interface> survey dump"

Parameters

`interface:` interface to change

`path_iw:` path to iw

Return Value

decoded information from survey

get_iw_scan_full(*interface*, *path_iw*=`__DEFAULT_IW_PATH`)

execute command "iw dev <interface> scan dump"

Parameters

`interface:` interface to change

`path_iw:` path to iw

Return Value

decoded information from scan dump

get_iw_scan_mac(*interface*, *path_iw*=`__DEFAULT_IW_PATH`)

executes the command "iw dev <interface> scan dump"

Parameters

`interface:` interface to scan

`path_iw:` path to iw

Return Value

decoded information from scan dump, only the detected MACs

get_iw_scan(*interface*, *path_iw*=`__DEFAULT_IW_PATH`)

command dev <interface> scan dump

Parameters

`interface:` interface to scan

`path_iw:` path to iw

Return Value

decoded information from scan dump, only the detected MACs

trigger_scan(*interface*, *path_iw*=`__DEFAULT_IW_PATH`)

command dev <interface> scan trigger it is necessary to call this method before call any method with 'scan', it forces the AP to scan all valid channels, and populate the statistics

Parameters

`interface:` interface to scan

`path_iw:` path to iw

Return Value

nothing

get_phy_with_wlan(*interface*, *path_iw*=`__DEFAULT_IW_PATH`)

Parameters

`interface`: the name of the interface, e.g. 'wlan0'

Return Value

a string with the phy interface name

3.2 Variables

Name	Description
valid_frequencies	**Value:** `[2412+ i* 5 for i in range(13)]`

4 Module command_ap.cmd.ifconfig

converts the output of ifconfig into a dictionary

4.1 Functions

decode_ifconfig(*data*)

read ifconfig's output and returns a dictionary with the data

Parameters

`data`: is the captured screen from ifconfig output

Return Value

dictionary with decoded ifconfig output

4.2 Variables

Name	Description
__package__	**Value:** `'command_ap.cmd'`

5 Module command_ap.cmd.iwconfig

convert the output of iwconfig into a dictionary

5.1 Functions

grab_first(*x*, *k*, *type*=`None`)

helper function to decode iwconfig. grabs the first element of the split given by key k

Parameters

`x`: string to be splitted by 'espaces'

`k`: position of the splitted result to be returned

`type`: valid values are [int, float, None]. If None, return the str, else try to convert to the specified type

Return Value

the element 'k'

(type=type)

decode_iwconfig(*data*)

get the output of iwconfig and convert it into a dictionary

Parameters

`data`: output of iwconfig captured by the system

Return Value

a dictionary with iwconfig fields

5.2 Variables

Name	Description
cmds_iwconfig	**Value:** `{'AP': <__builtin__.function object>, 'Bit Rate': <__buil...`
___package___	**Value:** `None`

6 Module command_ap.cmd.scan

convert the output of iw dev station dump into a dictionary

6.1 Functions

find_in_cmd(*line*)

searches the line against the text in ‘cmds‘ returns the data in a simple dictionary

get_subitems(*_l*, *lines*)

decode_scan(*data*)

decodes all the information returned by ‘scan dump‘ TODO: finish all fields

Parameters

`data`: the output of scan dump

Return Value

dictionary containing the data

decode_scan_mac(*data*)

get the list of APs in range

Parameters

`data`: the output of scan dump

Return Value

list with the macs detected

decode_scan_basic(*data*)

get the list of APs in range

Parameters

`data`: the output of scan dump

Return Value

list with the macs detected

6.2 Variables

Name	Description
cmds	**Value:** `[’TSF’, ’freq’, ’beacon interval’, ’capability’, ’signal’...`
cmds_sub	**Value:** `[’RSN’, ’WMM’, ’BSS Load’, ’HT operation’, ’Overlapping B...`
__package__	**Value:** `’command_ap.cmd’`

7 Module command_ap.cmd.station

convert the output of iw dev station dump into a dictionary

7.1 Functions

decode_iw_station(*data*)

return the data from "iw dev station dump"

Parameters

`data`: output from "iw dev station dump"

Return Value

decode_hostapd_status(*data*)

```
decodes "hostapd_cli status"’s output

@param data: output from hostapd_cli status
@return: dictionary containing
    {olbc_ht : 1
     cac_time_left_seconds : N/A
     num_sta_no_short_slot_time : 0
     olbc : 0
     num_sta_non_erp : 0
     ht_op_mode : 0x15
     state : ENABLED
     num_sta_ht40_intolerant : 0
     channel : 6
     bssid[0] : b0:aa:ab:ab:ac:11
     ieee80211n : 1
     cac_time_seconds : 0
     num_sta[0] : 2
     ieee80211ac : 0
     phy : phy0
     num_sta_ht_no_gf : 1
     freq : 2437
     num_sta_ht_20_mhz : 2
     num_sta_no_short_preamble : 0
     secondary_channel : 0
     ssid[0] : ethanolQL1
     num_sta_no_ht : 0
     bss[0] : wlan0
    }
```

is_mac(*s*)

verifies if ’s’ contains a MAC address

Return Value

the mac address found or None

(type=str)

decode_hostapd_station(*data*)

```
@param data: output from hostapd_cli all_sta
@return: dictionary of dictionary
     {station1_mac: {’dot11RSNAStatsSelectedPairwiseCipher’: ’00-0f-ac-4’,
                     ’rx_packets’: ’164’,
                     ’dot11RSNAStatsTKIPLocalMICFailures’: ’0’,
                     ’rx_bytes’: ’5420’,
                     ’inactive_msec’: ’11828’,
                     ’connected_time’: ’3402’,
                     ’hostapdWPAPTKState’: ’11’,
                     ’tx_bytes’: ’1340’,
                     ’dot11RSNAStatsVersion’: ’1’,
                     ’tx_packets’: ’10’,
                     ’hostapdWPAPTKGroupState’: ’0’,
                     ’dot11RSNAStatsTKIPRemoteMICFailures’: ’0’},
     }
```

7.2 Variables

Name	Description
__package__	**Value:** `’command_ap.cmd’`

8 Module command_ap.cmd.survey

convert the output of iw dev station dump into a dictionary

8.1 Functions

decode_survey(*data*)

```
decodes the data provided by "iw survey dump"

@param data: output from iw dev survey dump
@return: dictionary of dictionary
        {2432: {'noise': '-95 dBm',
                'in use': True,
                'channel transmit time': '713 ms',
                'channel busy time': '9479 ms',
                'channel active time': '54259 ms',
                'channel receive time': '8279 ms'},
         2467: {},
         }
```

8.2 Variables

Name	Description
__package__	**Value:** `'command_ap.cmd'`

9 Module command_ap.cmd.xmit

Module xmit

This module decodes the "xmit" file. Returns a dictionary with all decoded fields.

9.1 Functions

check(*line*, *items*)

helper function: test if one of the items in items exists in line

Parameters

`line`: the line to check

`items`: list of items

Return Value

true if the item in items exists in line

decode_xmit(*filename*)

reads the ath*k/xmit file, if file not found returns an empty dictionary otherwise decodes the file and returns a dictionary with its contents

Parameters

`filename`: full path to xmit

Return Value

a dictionary with xmit's content

9.2 Variables

Name	Description
lines_with_queue_data	**Value:** `['MPDUs Queued', 'MPDUs Completed', 'MPDUs XRetried', 'Ag...`
___package___	**Value:** `'command_ap.cmd'`

10 Package command_ap.get_set

10.1 Modules

- **client**: the server accepts requests from an http client.
  (Section 11, p. 17)
- **server**: server that accepts requests from an http client used to send commands to the AP
  (Section 12, p. 18)
- **server_ffox**: The client (firefox) sends the following json data:
  (Section 13, p. 24)
- **teste**: Test to get the data to compute: MOS client, hybrid and AP
  (Section 14, p. 26)

10.2 Variables

Name	Description
___package___	**Value:** `None`

11 Module command_ap.get_set.client

the server accepts requests from an http client. this module is uses to send commands to the AP, for testing purposes.

Usage: python3 server.py [–port 8080]

11.1 Variables

Name	Description
valid_urls	**Value:** `['/', '/test', '/info', '/get_power', '/set_power', '/iwc...`

12 Module command_ap.get_set.server

```
server that accepts requests from an http client
used to send commands to the AP


Usage from command line:
-------------------

python3 -m get_set.server.py [--port 8080]


Usage from program:
-------------------

import get_set.server
server.run(port)


Requirements
------------
iw 4.9+  (https://git.kernel.org/pub/scm/linux/kernel/git/jberg/iw.git/snapshot/iw-4.9.tar.gz)
iwconfig version 30
```

12.1 Functions

run(*port*=8080)

collect(*port*)

creates an HTTP server that receives POST requests from the client save the BODY as JSON in a file

Parameters

`port`: number of the server port. Required.

12.2 Variables

Name	Description
LOG	**Value:** `logging.getLogger('REST_SERVER')`
httpd	**Value:** `None`
last_rt	**Value:** `dict()`
last_tx_bytes	**Value:** `None`
last_ampdu	**Value:** `None`
MAX_REPORTED_BITRATE	**Value:** `20000.0`
MAXIMUM_TX_BITRATE	**Value:** `54.0`
MAX_TX_BYTES_WIFI	**Value:** `MAXIMUM_TX_BITRATE* 1024* 1024`

12.3 Class myHandler

http.server.BaseHTTPRequestHandler
└ **command_ap.get_set.server.myHandler**

"This class will handles any incoming request from the browser

12.3.1 Methods

__init__(*self*, *request*, *client_address*, *server*)

query(*self*)

parses the HTML query field

send_error(*self*)

returns to the web client a 404 error

send_dictionary(*self*, *d*)

returns to the web client a dictionary containing the data. the client should use pickle.loads() to reconvert the data to a python object

info(*self*)

```
process /get_info

@return: dictionary
    {’wiphy’: ’0’, ’Interface’: ’wlan0’, ’addr’: ’b0:aa:ab:ab:ac:11’,
     ’width’: ’20MHz,’, ’channel’: ’6’,
     ’txpower’: ’1.00 dBm’, ’ssid’: ’ethanolQL1’, ’type’: ’AP’,
     ’ifindex’: ’3’, ’frequency’: ’2437MHz,’,
     ’wdev’: ’0x1’, ’center1’: ’2437MHz’}
@rtype: dict
```

iwconfig(*self*)

```
process /get_iwconfig

@return: dictionary
{’Power Management’: ’off’, ’RTS thr’: ’off’, ’IEEE’: ’802.11bgn’,
 ’Mode’: ’Master’, ’Retry short limit’: 7, ’Fragment thr’: ’off’,
 ’interface’: ’wlan0’}
```

ifconfig(*self*)

```

process /get_ifconfig

@return:
    {’iface’: ’wlan0’,
     ’rx_bytes’: ’2986426585’, ’rx_overruns’: ’0’, ’rx_dropped’: ’0’,
     ’rx_packets’: ’30257063’, ’rx_scale_bytes’: ’2.9’, ’rx_errors’: ’0’
     ’tx_scale_bytes’: ’53.9’, ’tx_bytes’: ’53923422941’, ’tx_dropped’: ’0’,
     ’tx_packets’: ’43083207’, ’tx_overruns’: ’0’, ’tx_errors’: ’0’,
     ’collisions’: ’0’, ’frame’: ’0’,
     ’txqueuelen’: ’1000’,
     ’carrier’: ’0’,
     }
```

get_power(*self*)

process /get_power

Return Value

the tx power of iface

set_power(*self*)

process /set_power

Return Value

set the tx power of iface to new_power

set_channel(*self*)

process /set_channel

Return Value

new channel in a dictionary format {’channel’: new_channel}

(type=dict)

xmit(*self*)

```

process /get_xmit

@return: dictionary
{’TXOP Exceeded_VO’: ’0’, ’TX-Pkts-All_VO’: ’4441336’, ’FIFO Underrun_BK’: ’0’,
 ’HW-put-tx-buf_BK’: ’0’, ’DELIM Underrun_VI’: ’0’, ’MPDUs Queued_BE’: ’866’,
 ’DESC CFG Error_VO’: ’0’, ’Aggregates_BK’: ’0’, ’FIFO Underrun_VO’: ’0’,
 ’DESC CFG Error_VI’: ’0’, ’AMPDUs Queued HW_VI’: ’0’, ’TX-Pkts-All_BE’: ’42978693’, ’TX-Pkts-All_
@rtype: dict
```

get_stations(*self*)

```
process /num_stations

@return:
{'54:e6:fc:da:ff:34': {'short slot time': 'yes', 'DTIM period': 2.0,
                       'authorized': 'yes',
                       'tx bitrate': 1.0,
                       'tx bytes': 322.0, 'tx packets': 2.0, 'tx failed': 0.0,
                       'rx bitrate': 1.0
                       'rx bytes': 288.0, 'rx drop misc': 1.0, 'rx packets': 2.0,
                       'preamble': 'short',
                       'WMM/WME': 'yes',
                       'signal avg': 58.0, 'MFP': 'no',
                       'beacon interval': 100.0, 'signal': 57.0,
                       'tx retries': 1.0,
                       'authenticated': 'yes', 'TDLS peer': 'no',
                       'connected time': 0.0, 'inactive time': 4.0, 'associated': 'yes',
                       }
 }
@rtype: dict
```

get_num_stations(*self*)

process /get_num_stations

Return Value

number of stations

(type=int)

get_survey(*self*)

```
@return:
    {2432: {'channel busy time': 394.0, 'channel receive time': 285.0, 'channel transmit time': 81
     2437: {'in use': True, 'channel receive time': 1073537372.0, 'noise': 80.0, 'channel busy tim
     2442: {'channel busy time': 682.0, 'channel receive time': 336.0, 'channel transmit time': 31
     2467: {},
     2472: {},
@rtype: dict
```

get_scan(*self*)

```
returns the partial results from iw scan dump

{'50:c7:bf:3b:db:37': {'channel': '1',
                       'SSID': 'LAC',
                       'TSF': '0d, 05:19:27',
                       'last seen': 104,
                       'freq': 2412,
                       'signal': -54.0,
                       'beacon interval': 100},
 '84:b8:02:44:07:d2': {'channel': '1',
                       'SSID': 'DCC-usuarios',
                       'TSF': '27d, 03:24:26',
                       'last seen': 1024,
                       'freq': 2412,
                       'signal': -58.0,
                       'beacon interval': 102}
 }
```

get_scan_mac(*self*)

return the result from iw scan dump

Return Value

list[str] each entry is a detected mac

get_config(*self*)

return the result from hostapd_cli get_config

Return Value

{'group_cipher': 'CCMP', 'key_mgmt': 'WPA-PSK ', 'rsn_pairwise_cipher': 'CCMP', 'ssid': 'ethanolQL1', 'bssid': 'b0:aa:ab:ab:ac:11', 'wps_state': 'disabled'}

(type=dict)

hello(*self*)

standard hello response. white page with 200 code

do_GET(*self*)

self.path is the command the client wants to execute

function_handler is a dictionary that contains {url : function responds to the command}

fill_feature_results(*self*, *survey*, *station*, *k*, *stations*, *iface*)

function that returns the features of a station.

Parameters

`survey`: data from iw survey dump

`station`: the station data selected from the result of "iw station dump"

`k`: the k-th value of the survey

`stations`: data from iw station dump

`iface`: wireless interface name

get_features(*self*)

```
process /get_features

here we collect all features necessary to train the QoS predictor

@return: dictionary
    {'54:e6:fc:da:ff:34': {'tx_bitrate': 1.0, 'rx_bitrate': 1.0,
                           'tx_power': 1.0, 'avg_signal': 54.0,
                           'rxdrop': 16.0, 'rxb': 1232.0, 'rxp': 32.0,
                           'txr': 0.0, 'txp': 3.0, 'txf': 0.0, 'txb': 487.0,
                           'crt': 1073085286.0, 'cbt': 1163082876.0,
                           'ctt': 60749755.0, 'cat': 3626867638.0,
                           'num_stations': 1
                           }
    }
```

get_mos_hybrid(*self*)

Return Value

[[timestamp, FR, frame_loss, SBR, PLR], ...]

get_mos_ap(*self*)

Return Value

[num_stations, BER, AMPDU, traffic_load] needed to compute the MOS_AP

get_mos_client(*self*)

```
read from local memory is filled using an node.js server
this server receives connections from the clients, and then stores
the values in a local json file

- r[t] = reportedBitrate in time [t] / max_bitrate
- srt = not_running_time / (not_running_time + execution_time)
- r[t-1] is obtained from a saved variable: self.last_rt[client_ip]

@ return: [rt, rt_1, srt, sta]
```

13 Module command_ap.get_set.server_ffox

The client (firefox) sends the following json data:

{'chunkData[resolution][]': '768', 'chunkData[start]': '32', 'chunkData[filename]': '7-16.video', 'chunkData[index]': '16', 'chunkData[quality]': '6', 'chunkData[endFragment]': 'true', 'chunkData[bandwidth]': '976342', 'chunkData[segmentType]': 'MediaSegment', 'playing[quality]': '6', 'playing[time]': '31.607175', 'playing[paused]': 'false', 'chunkData[representationId]': '7', 'chunkData[end]': '34', 'chunkData[codec]': 'video/mp4;codecs="avc3.64000C"'}

'index': 6, 'latency': {'avg': 0.04, 'low': 0.08, 'high': 0.06}, 'droppedFPS': 15, 'maxIndex': 19, 'reportedBitrate': 976, 'calculatedBitrate': 810, 'video_ratio': {'avg': 11.63, 'low': 17.24, 'high': 13.63}, 'bufferLevel': 2.4, 'download': {'avg': 0.12, 'low': 0.17, 'high': 0.15},

13.1 Functions

decode3field(*x*)

Parameters

`x`: a string formatted as "float | float | float" to be converted to a list

Return Value

a list with 3 floats

(type=list(float))

decodeInt(*x*)

Parameters

`x`: a string to be converted to int

Return Value

an int, or np.nan

(type=int)

13.2 Variables

Name	Description
LOG	**Value:** `logging.getLogger('SERVER_FFOX')`
funcs	**Value:** `{'droppedFPS': lambda x: decodeInt(x), 'index': lambda x:...`
map_ip_to_sta	**Value:** `{'192.168.0.11': 'cloud', '192.168.0.12': 'storm', '150.1...`
ffox_memory	**Value:** `FirefoxDataMemory()`

13.3 Class FirefoxDataMemory

object

command_ap.get_set.server_ffox.FirefoxDataMemory

13.3.1 Methods

___init___(*self*)

x.___init___(...) initializes x; see help(type(x)) for signature

Overrides: object.___init___ extit(inherited documentation)

push(*self*, *data*)

pop(*self*)

Inherited from object

___delattr___(), ___format___(), ___getattribute___(), ___hash___(), ___new___(), ___reduce___(), ___reduce_ex___(), ___repr___(), ___setattr___(), ___sizeof___(), ___str___(), ___subclasshook___()

13.3.2 Properties

Name	Description
Inherited from object	
___class___	

13.4 Class SrvPosts

http.server.BaseHTTPRequestHandler

command__ap.get__set.server__ffox.SrvPosts

receives posts from the client (firefox), and saves the data into a json file

13.4.1 Methods

do__POST(*self*)

get the data, and save it into memory (a global variable called ffox_memory)

14 Module command_ap.get_set.teste

Test to get the data to compute: MOS client, hybrid and AP

14.1 Functions

call(*cmd*, *ap_name*=`'gnu-nb3.winet.dcc.ufmg.br'`)

calls the AP

Parameters

`cmd`: valid values are ['/get_mos_hybrid', '/get_mos_ap', '/get_mos_client']

call_h(*cmd*=`'/get_mos_hybrid'`)

get MOS hybrid data

call_a(*cmd*=`'/get_mos_ap'`)

get MOS AP data

call_c(*cmd*=`'/get_mos_client'`)

get MOS client data

15 Package command_ap.publisher_subscriber

15.1 Modules

- **publisher** *(Section 16, p. 28)*
- **subscriber** *(Section 17, p. 29)*

15.2 Variables

Name	Description
___package___	**Value:** `None`

16 Module command__ap.publisher__subscriber.publisher

16.1 Variables

Name	Description
port	**Value:** `sys.argv [1]`
context	**Value:** `zmq.Context()`
socket	**Value:** `context.socket(zmq.PUB)`

17 Module command_ap.publisher_subscriber.subscriber

17.1 Variables

Name	Description
port	**Value:** `sys.argv [1]`
port1	**Value:** `sys.argv [2]`
context	**Value:** `zmq.Context()`
socket	**Value:** `context.socket(zmq.SUB)`
topicfilter	**Value:** `"10001"`
total_value	**Value:** `0`

18 Package command_ap.rl

18.1 Modules

- **agent**: runs the agent: python3 agent.py
  (Section 19, p. 31)
- **app1** *(Section 20, p. 35)*
- **basic** *(Section 21, p. 36)*
  - **environment** *(Section 22, p. 37)*
- **mab**: This module define three abstract MAB agents: * RandomAbstract: select random actions * EpsilonGreedyAbstract: select action using an epsilon-greedy policy * UCBAbstract: selects actions based on the UCB policy
  (Section 23, p. 38)
- **model**: This module calculates the QoS based on the features
  (Section 24, p. 45)
- **reward**: runs the agent: python3 agent.py
  (Section 25, p. 46)

18.2 Variables

Name	Description
curr	**Value:** `os.getcwd()`

19 Module command_ap.rl.agent

runs the agent: python3 agent.py

the –double-trick parameter uses the trick suggested by xxx, since MAB was not meant to run forever. If it is active, time periods of T iterations will be considered, and for each T iteractions this period is increased to 2T. –T define the initial period.

Version: 2.0

Author: Henrique Moura

Copyright: Copyright 2018, h3dema

License: GPL

19.1 Functions

send_command(*server*, *port*, *interface*, *cmd*)

send a command to the AP, using the REST API

Parameters

`server`: server name or IP

`port`: socket port

`interface`: name of the wireless interface, e.g. 'wlan0'

`cmd`: the /command[?query]

set_power(*server*, *port*, *interface*, *new_power*)

set the AP's transmission power

Parameters

`server`: server name or IP

`port`: socket port

`interface`: name of the wireless interface, e.g. 'wlan0'

`new_power`: the new transmission power in dBm [1, 15]
(type=int)

get_power(*server*, *port*, *interface*)

get the AP’s transmission power

Parameters

`server`: server name or IP

`port`: socket port

`interface`: name of the wireless interface, e.g. ’wlan0’

Return Value

the transmission power in dBm [1, 15]

(type=int)

get_features(*server*, *port*, *interface*)

get the AP’s features necessary to calculate the QoS

Parameters

`server`: server name or IP

`port`: socket port

`interface`: name of the wireless interface, e.g. ’wlan0’

Return Value

the features

(type=dict)

19.2 Variables

Name	Description
__credits__	**Value:** `["Henrique Moura"]`
__maintainer__	**Value:** `"Henrique Moura"`
__email__	**Value:** `"h3dema@gmail.com"`
__status__	**Value:** `"Production"`
LOG	**Value:** `logging.getLogger(’AGENT’)`
f_handler	**Value:** `logging.FileHandler(’Log_Qos.log’)`
f_format	**Value:** `logging.Formatter(’%(message)s’)`

19.3 Class MABAgent

object
command_ap.rl.mab.MAB
command_ap.rl.mab.UCBAbstract
command_ap.rl.agent.MABAgent

this is the real class. it implements the abstract methods from UCBAbstract you should implement only the run_action method this method interacts with the environment, performing the action and collection the reward it returns if the agent was able to perform the action

19.3.1 Methods

__init__(*self*, *n_actions*, *server*, *port*, *interface*)

the defaults of C and b define a UCB1 policy

Parameters

`n_actions`: number of actions the agent can perform from [0, n_actions - 1]

(type=int)

`server`: server name or IP

`port`: socket port

`interface`: name of the wireless interface, e.g. 'wlan0'

Overrides: object.__init__

run_action(*self*, *action*)

:return r: the reward of the action taken :return success: boolean value indicating if the agent could perform the action or not

Overrides: command_ap.rl.mab.MAB.run_action

Inherited from command_ap.rl.mab.UCBAbstract(Section 23.6)

get_action(), get_prob(), w()

Inherited from command_ap.rl.mab.MAB(Section 23.3)

name(), reset_pulls(), update()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

19.3.2 Properties

Name	Description
Inherited from object __class__	

20 Module command_ap.rl.app1

21 Package command_ap.rl.basic

21.1 Modules

- **environment** *(Section 22, p. 37)*

22 Module command_ap.rl.basic.environment

22.1 Class environment

object ─┐
command_ap.rl.basic.environment.environment

22.1.1 Methods

__init__(*self*, **kwargs)

x.__init__(...) initializes x; see help(type(x)) for signature

Overrides: object.__init__ extit(inherited documentation)

get_reward(*self*, **kwargs)

take_action(*self*, **kwargs)

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

22.1.2 Properties

Name	Description
Inherited from object	
__class__	

23 Module command_ap.rl.mab

This module define three abstract MAB agents: * RandomAbstract: select random actions * EpsilonGreedyAbstract: select action using an epsilon-greedy policy * UCBAbstract: selects actions based on the UCB policy

Version: 2.0

Author: Henrique Moura

Copyright: Copyright 2018, h3dema

License: GPL

23.1 Functions

softmax(*x*)

returns the softmax function (probabilities) given an array x

Parameters

`x`: float

Return Value

softmax(x)

(type=float)

23.2 Variables

Name	Description
__credits__	**Value:** `["Henrique Moura"]`
__maintainer__	**Value:** `"Henrique Moura"`
__email__	**Value:** `"h3dema@gmail.com"`
__status__	**Value:** `"Production"`
LOG	**Value:** `logging.getLogger('MAB')`

23.3 Class MAB

object ─┐

command_ap.rl.mab.MAB

23.3.1 Methods

__init__(*self*, *n_actions*)

creates the agent handler

Parameters

`n_actions`: number of actions the agent can perform from [0, n_actions - 1]

(type=int)

Overrides: object.__init__

get_action(*self*)

Get current best action

Return Value

the selected action according to the policy

(type=int)

run_action(*self*, *action*)

```
@return: r, success
       'r' is the reward of the action taken, and 'success' is boolean value indica
@rtype float, bool
```

reset_pulls(*self*)

reset the number of executions to zero.

update(*self*, *action*, *reward*)

observe the reward from action and update agent's internal parameters

Parameters

`action`: the action 'a' taken

`reward`: the reward obtained by taking action 'a'

name(*self*)

name of the class

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

23.3.2 Properties

Name	Description
Inherited from object __class__	

23.4 Class RandomAbstract

object ┐

command_ap.rl.mab.MAB ┐

command_ap.rl.mab.RandomAbstract

this class implements the random policy

23.4.1 Methods

get_action(*self*)

returns a random action

Return Value

the selected action

(type=int)

Overrides: command_ap.rl.mab.MAB.get_action

Inherited from command_ap.rl.mab.MAB(Section 23.3)

__init__(), name(), reset_pulls(), run_action(), update()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

23.4.2 Properties

Name	Description
Inherited from object	

continued on next page

Name	Description
__class__	

23.5 Class EpsilonGreedyAbstract

object

command_ap.rl.mab.MAB

command_ap.rl.mab.EpsilonGreedyAbstract

this class implements the epsilon-greedy policy

23.5.1 Methods

__init__(*self*, *n_actions*, *epsilon*=`0.01`)

creates the agent handler

Parameters

`n_actions`: number of actions the agent can perform from [0, n_actions - 1]

Overrides: object.__init__ extit(inherited documentation)

get_action(*self*)

selects the action to be taken based on the epsilon-greedy policy

Return Value

the selected action

(type=int)

Overrides: command_ap.rl.mab.MAB.get_action

Inherited from command_ap.rl.mab.MAB(Section 23.3)

name(), reset_pulls(), run_action(), update()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

23.5.2 Properties

Name	Description
Inherited from object	
__class__	

23.6 Class UCBAbstract

object ─┐

command_ap.rl.mab.MAB ─┐

command_ap.rl.mab.UCBAbstract

23.6.1 Methods

__init__(*self*, *n_actions*, *C*=1, *b*=2)

the defaults of C and b define a UCB1 policy

Parameters

`n_actions`: number of actions the agent can perform from [0, n_actions - 1]

Overrides: object.__init__

w(*self*)

get_prob(*self*)

returns the probability of each action using SOFTMAX

get_action(*self*)

selects the action to be taken based on the UCB policy

Return Value

the selected action

(type=int)

Overrides: command_ap.rl.mab.MAB.get_action

Inherited from command_ap.rl.mab.MAB(Section 23.3)

name(), reset_pulls(), run_action(), update()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

23.6.2 Properties

Name	Description
Inherited from object __class__	

23.7 Class Boltzmann

object

command_ap.rl.mab.MAB

command_ap.rl.mab.UCBAbstract

command_ap.rl.mab.Boltzmann

23.7.1 Methods

get_action(*self*)

selects the action to be taken based on the softmax policy

Return Value

the selected action

(type=int)

Overrides: command_ap.rl.mab.MAB.get_action

Inherited from command_ap.rl.mab.UCBAbstract(Section 23.6)

__init__(), get_prob(), w()

Inherited from command_ap.rl.mab.MAB(Section 23.3)

name(), reset_pulls(), run_action(), update()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

23.7.2 Properties

Name	Description
Inherited from object __class__	

24 Module command_ap.rl.model

This module calculates the QoS based on the features

24.1 Functions

get_QoS(*model*, *features*)

TODO: implement this, using the model and the features use keras model to predict the QoS based on the features

create_window(*data_values*, *timesteps*)

convert the data_values into the format needed by the keras model

Parameters

`data_values:`

`timesteps:` number of time steps

25 Module command_ap.rl.reward

runs the agent: python3 agent.py

the –double-trick parameter uses the trick suggested by xxx, since MAB was not meant to run forever. If it is active, time periods of T iterations will be considered, and for each T iteractions this period is increased to 2T. –T define the initial period.

Version: 2.0

Author: Henrique Moura

Copyright: Copyright 2018, h3dema

License: GPL

25.1 Functions

calc_reward(*qos*, *power*)

this function goes to the agent it receives two scaled parameters (between 0 and 1), and returns the reward between 0 and 1

25.2 Variables

Name	Description
___credits___	**Value:** `["Henrique Moura"]`
___maintainer___	**Value:** `"Henrique Moura"`
___email___	**Value:** `"h3dema@gmail.com"`
___status___	**Value:** `"Production"`

26 Script script-hostapd_conf

26.1 Variables

Name	Description
interface	**Value:** `wlan0`
bssid	**Value:** `aa:`
ssid	**Value:** `my_wifi`
driver	**Value:** `nl80211`
ignore_broadcast_ssid	**Value:** `0`
channel	**Value:** `6`
hw_mode	**Value:** `g`
wmm_enabled	**Value:** `1`
ieee80211n	**Value:** `1`
wpa	**Value:** `2`
wpa_passphrase	**Value:** `password`
wpa_pairwise	**Value:** `TKIP`
rsn_pairwise	**Value:** `CCMP`
auth_algs	**Value:** `1`
macaddr_acl	**Value:** `0`
ctrl_interface	**Value:** `/ var/ run/ hostapd`
logger_syslog	**Value:** `-1`
logger_syslog_level	**Value:** `0`
logger_stdout	**Value:** `-1`
logger_stdout_level	**Value:** `0`

27 Script script-setup_cfg

27.1 Functions

platform(*iterable*)

Return True if bool(x) is True for any x in the iterable. If the iterable is empty, return False.

Return Value

bool

27.2 Variables

Name	Description
name	**Value:** `Command-the-ap`
version	**Value:** `1.0.0`
author	**Value:** `Henrique Moura`
description	**Value:** `This group of python modules allows to send commands from...`
license	**Value:** `GNU`
keywords	**Value:** `wireless`
classifiers	**Value:** `Development Status:`
zip_safe	**Value:** `false`
python_requires	**Value:** `>= 3.0`

Index