

PARAMICS Plugin Document – On-ramp queue override control

Lianyu Chu

PATH ATMS Center
University of California, Irvine

Plugin Compatibility: V4
Release date: 3/20/2003

522 Social Science Tower
Irvine, CA 92697-3600
URL: <http://www.its.uci.edu/>



Table of Contents

Table of Contents	2
1. Introduction.....	3
2 Plugin implementation.....	4
2.1 Development framework.....	4
2.2 Control logic	4
3 Step-by-step user manual.....	5
3.1 Adding queue detector	5
3.2 Preparation of the “queue_control” file	5
3.3 Loading plugin	7
3.4 Output file	7
3.5 Error checking.....	8
4 Technical supports	9
4.1 Future development	9
4.2 Contact information	9

1. Introduction

The effect of ramp metering is that not all the vehicles can enter freeway from a meter and thus a waiting queue will be formed at the entrance ramp. If the queue exceeds a certain length, it will interfere with the adjacent street traffic. The queue override strategy is often used to solve this problem through the placement of a queue detector at the ramp entrance (usually, at the end of the entrance ramp).

An example of the queue override policy is that if the occupancy value of the queue detector exceeds a threshold (e.g. 50%), a high metering rate will be applied to the corresponding meter in order to release more vehicles to freeway.

This plugin is a complementary module of the ramp metering control plugin, which implements pre-timed metering control, but does not include any queue override strategy. The purpose of this plugin is to implement a typical queue override strategy that uses the queue detector. This plugin can be used together with the ramp metering control plugin. Also, this plugin can be used together with other advanced ramp metering control algorithms, such as ALINEA, which do not integrate a queue override strategy with them.

2 Plugin implementation

2.1 Development framework

Figure 1 illustrates the hierarchical development framework of advanced ramp-metering algorithm plugins in PARAMICS. The advanced ramp-metering algorithm plugin is built on top of two basic plugin modules, i.e., ramp metering controller and loop data aggregator. The on-ramp signals in the simulation network are controlled by the ramp metering plugin, through which metering rates can be queried and set by other plugin modules. The loop data aggregator emulates the real-world loop data collection, typically with a thirty-second polling interval, and broadcast the latest loop data to the dynamic memory. At each time increment, the advanced ramp-metering algorithm plugin can access the dynamic memory and obtains the required loop data through interface functions provided by the loop data aggregation plugin. Then the metering rate for the next control interval is calculated based on the advanced ramp-metering algorithm. The new metering rate is finally sent back to the ramp controller plugin for implementation.

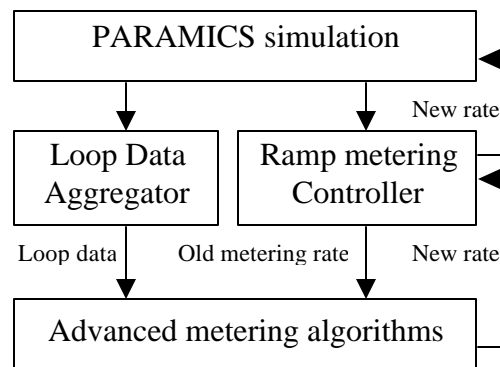


Figure 1 The hierarchical approach for the development of advanced metering algorithms

2.2 Control logic

This plugin implements a queue override strategy depending on a queue detector placed on the entrance ramp. A typical location of the queue detector is located at the upstream end of the entrance ramp.

The control logic is as follows:

If the percent occupancy of mainline detector is higher than the pre-specified “override occupancy threshold”, an override control plan will be applied to the corresponding meter in order to avoid interference with the arterial traffic.

The override control plan can be a specified metering rate (such as maximum metering rate), or all green.

3 Step-by-step user manual

3.1 Adding queue detector

The queue override strategy implemented in this plugin is based on a queue detector, which is generally located at the end of the entrance ramp in the real world. The ramp meter design manual of Caltrans has the following descriptions of the location of queue detectors: “One loop per entrance ramp lane should be installed for queue detection near the connection of the surface street”.

This location of queue detector can be modified in order to improve the control performance. A study shows that when the queue detector is located at 75% of the physical ramp length of the on-ramp, the performance of queue control is better than the cases when the queue detector is located at 100% and 62.5% length of the on-ramp (Hasan, M., 1999).

3.2 Preparation of the “queue_control” file

The “queue_control” file includes the queuing control information, which is the input of this plugin. The file should be located at the same directory as any other network files. An example of “queue_control” is shown below.

```

total number of queuing-controlled on-ramps is      7
checking control file                             yes
control cycle                                     30
algorithm activation time                         06:00:00
algorithm deactivation time                       09:00:00
report queuing condition                         yes

on-ramp signal                                    33
queue detector                                    405n0.93orspill
override occupancy threshold                      0.5
override control plan                             METER_ON with 1 veh per 5 sec

...

on-ramp signal                                    36
queue detector                                    405n1.11orspill
override occupancy threshold                      0.5
override control plan                             METER_OFF

```

There are two parts in this “queue_control” file. The first part is the basic information of this plugin.

1. The first line defines the number of entrance ramps to be controlled by this API.
2. The option of “checking control file” is used for checking if there are any mistakes in the control file. If “yes”, this API will print out the information obtained from “queue_control” file during the starting process of simulation.
3. The third line defines the control cycle of the ramp metering control. Basically, this control cycle is the loop data aggregation cycle, typically 30 seconds. If the queue detector detects the excessive queue length on entrance ramp, the metering rate controlled by the queuing strategy will be effective for the time length equal to this control cycle.
4. During the time period between “algorithm activation time” and “algorithm deactivation time”, the ramp-metering algorithm is activated.
5. If “report queuing condition” is “yes”, the metering rates of queuing control condition of all controlled ramps will be output (every update cycle) to a file named “rampQueue.txt”. This file can be found in the sub-directory “Log” under the network directory .

The second part of the “queue_control” file is the input information of each entrance ramp, which always begins from a blank line.

1. “*on-ramp signal*” is the global node number of the on-ramp signal in the road network.
2. “*queue detector*” is the name of the queue detector for the entrance ramp. If there is no queue detector or no queuing strategy is involved in ramp metering, please specify as “N/A”. If the specified detector can not be found in the “detectors” file, a warning message will be shown and the entrance ramp will be operated without any queuing strategy (even a strategy has been input in the “queue_control” file).
3. “override occupancy threshold” is required to be specified if the queue detector has been specified earlier in the file. If the percent occupancy of the queue detector station (if there is more than one queue detector at this location, the maximal percent occupancy of this location will be used) exceeds this threshold, the queuing strategy will be applied to avoid interference with the traffic on the surface street. If there is no queue detector, nothing needs to be filled in this line.
4. “override control plan” is actually the timing plan to handle the excessive queue length, which is required to be specified if the queue detector has been specified earlier in the file. The format is one of the following

```
METER_ON with BB veh per CC sec
METER_OFF
RAMP_CLOSURE
```

where ‘**BB**’ is the type of metering operation, single-entry metering (**BB** = 1) or platoon metering (**BB** = 2). ‘**CC**’ is the cycle of metering control. If the status is **METER_ON**, **BB** and **CC** must be specified. If the queue detector is specified but the override control plan timing plan of the corresponding queuing strategy is not specified, ‘**METER_OFF**’ is applied. If there is no queue detector, nothing needs to be filled in this line;

3.3 Loading plugin

This plugin needs the support of loop data aggregator plugin and ramp metering plugin. They should be specified earlier than this plugin in the ‘‘plugins’’ or ‘‘programming’’ file, i.e.:

```
loop_agg.dll
ramp_controller.dll
queue_controller.dll
```

In order to load and run this plugin correctly, please satisfy the following requirements:

- (1) The queue detectors in the ‘‘queue_control’’ file should be specified in ‘‘loop_control’’ file.
- (2) The ‘‘report cycle’’ in the ‘‘loop_control’’ file should be the same as the control cycle specified in the ‘‘queue_control’’ file.

3.4 Output file

If ‘‘report queuing condition’’ in the ‘‘queue_control’’ file is specified as ‘‘yes’’, the queuing information of all controlled ramps will be output (every update cycle) to a file named ‘‘moe-rampQueue.txt’’. It can be found in the subdirectory:

```
network/Log/run-xxx
```

where network is the name of the current working directory, and xxx is a three-digit sequence number.

For each on-ramp, queuing condition of last control cycle will be output. If the queuing control is enabled due to a long queue (spillover) on an entrance ramp, the output value is 1. Otherwise, it is 0. At the end of this file, a summary of the percentage of time that the queue override strategy is activated is provided. An example of this file is shown in Figure 2.

RAMP	#33	#36	#65	#73	#76	#95	#92
07:00:30	0	0	0	0	0	0	0
07:01:00	0	0	0	0	0	0	0
07:01:30	0	0	0	0	0	0	0
07:02:00	0	0	0	0	0	0	0
07:02:30	0	0	0	0	0	0	0
07:03:00	0	0	0	0	0	0	0
07:03:30	0	0	0	0	0	0	0
07:04:00	0	0	0	0	0	0	0
07:04:30	0	0	0	0	0	0	0
07:05:00	0	0	0	0	0	0	0
07:05:30	0	0	0	0	0	0	0
07:06:00	0	0	0	0	0	0	0
07:06:30	0	0	0	0	0	0	0
...							
08:58:00	0	0	0	0	0	0	0
08:58:30	0	0	0	0	0	0	0
08:59:00	0	0	0	1	0	0	0
08:59:30	0	0	0	1	0	0	0
09:00:00	0	0	0	1	0	0	0
SUMMARY:	0.42	0.00	0.00	23.33	0.00	0.83	1.67
AVERAGE:	3.75						

Figure 2 An example of the “moe-rampQueue.txt” file

3.5 Error checking

If any mistakes happened in the “queue_control” file or the other two supporting plugins, i.e. loop data aggregator and ramp metering control, this plugin will be disabled. The report window of PARAMICS will show whether this plugin is working.

Through enabling the option: “checking control file” in the “queue_control” file, you can check if there is any error in the “queue_control” file.

4 Technical supports

4.1 Future development

This plugin implements a rather rough queue control logic, which could lead to an oscillatory override behavior and under-utilization of the ramp storage space. Some better queue override strategies will be developed and evaluated in the future.

4.2 Contact information

Any comments and suggestions are welcome. Please contact us at the email address: lchu@translab.its.uci.edu.