Applying fuzzy logic
to a radio system with location dependent errors

Nandor Csereoka, 4th Year - CTI ENG - FLA, 2020-2021

Problem statement

Consider a radio system consisting of K users (e.g. K=3) which transfer data.

Between the moments t1 and t2, one of the users has a very bad radio link, with so many errors that the scheduler will not allocate radio resources (radio channels) to that user.

Compared to an ideal system (i.e., without errors), the user affected by (location dependent) errors will be lagging, which means that the amount of data transfered by that user is with LAG smaller than the data transfered by its correspondent in the ideal system.

After the moment t2 the user will have again a good radio link, and its weight in the scheduling algorithm has to be increased, with the purpose to reduce the value of LAG to zero, or very close to zero.

LAG is defined as the difference between the amount of data transdered by a user in the ideal system, and the amount of data transfered by the same user, in the real system.

Like in the previous problem, you can start from the *fifo* model from *samples*. We can consider either a discrete scheduling algorithm (e.g. WRR - Weighted Round Robin), or you can consider that each user has a transfer rate equal with 1/K of the capacity C of the network (e.g. C= 1 Mega bits per second).

The fuzzy algorithm will adjust the weight of the user affected by errors (respectively its transfer rate) on the expense of the other users.

Process

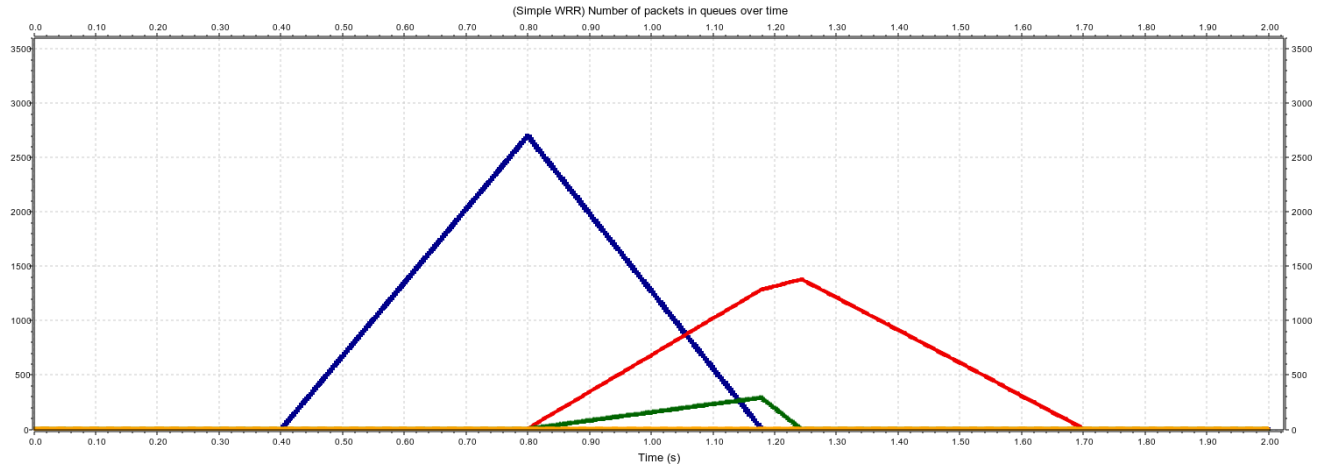
We will use the existing components from our implementation: the WRR scheduler, the simulated user modules and the sink. We'll also keep in mind the previously stated information regarding the structure of the network and the different interactions between its modules.

But now, we also introduce the fuzzy logic controller. This will intelligently adjust the weight of the unlucky user which loses their connection.
In terms of OMNeT modules, this means the addition of a controller which is triggered by a generator. This starts sending messages at a predefined time and at predefined intervals. In each such interval, the controller recalculates the optimal weight for the user which is lagging behind using fuzzy inference.

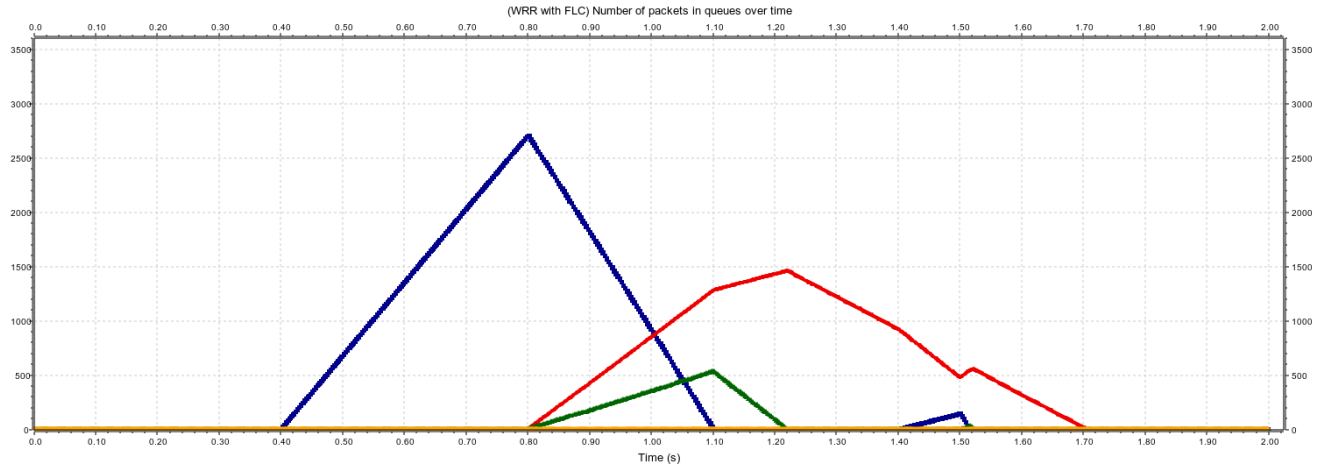
The generator module will be configured to start when the user regains their connection. After this, it will start triggering the controller at 0.1-second intervals.

The general fuzzy logic controller needs to be configured for this particular scenario. The wanted delay should be zero, just like we defined in the problem statement. The actual delay or LAG will be measured as the number of packets that are sitting in the user's queue. The mapping and scaling will be done according to the number of available channels. At each recalculation, the starting weight of the unlucky user will be considered to be null, in order to apply the optimal weight.
After everything is set up, we let the fuzzy inference do its magic.

The scenario for 4 users



This is the number of packets for the WRR scheduler **without** fuzzy inference. After the user regains their connection, their weight becomes the same as the user's with the highest priority.

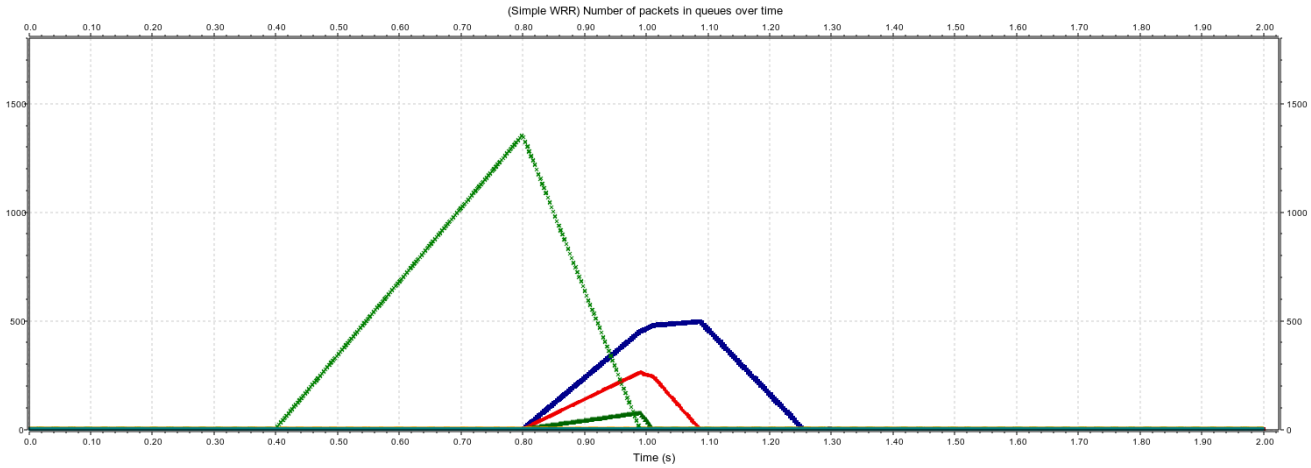


When the fuzzy logic controller is introduced, we can see the gradual adjustments which take place. The user is assigned a higher priority than previously - causing the user with the second highest priority (green line) to accumulate more LAG than before.

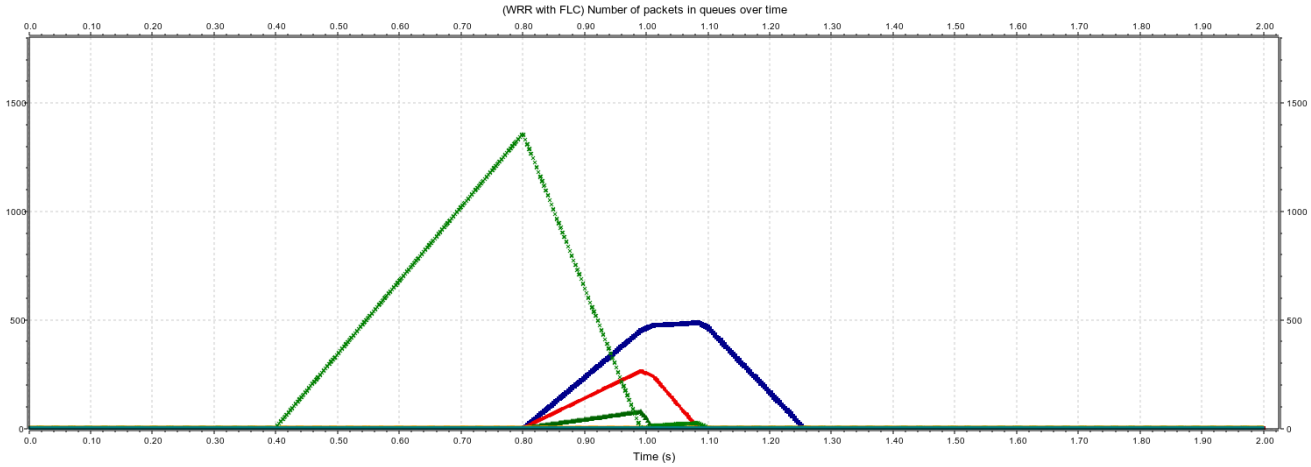
INFO:Calculez nou HP
INFO:Unlucky user: 0
INFO:Unlucky user lag: 1832
INFO:Unlucky user weight: 0
INFO: Dif nescalat = -1832
INFO: Dif scalat = 0
INFO:Fuzzify input
INFO:Fuzzify input
INFO: Result = 49
INFO:Res= 9
INFO:Pondere noua: 9

INFO:Calculez nou HP
INFO:Unlucky user: 0
INFO:Unlucky user lag: 7
INFO:Unlucky user weight: 0
INFO: Dif nescalat = -7
INFO: Dif scalat = 9
INFO:Fuzzify input
INFO:Fuzzify input
INFO: Result = 39
INFO:Res= 4
INFO:Pondere noua: 4

The scenario for 8 users



In the scenario without fuzzy logic, the user's weight is increased to the weight of the user with the highest priority.



When making use of the fuzzy logic controller, we can see an almost identical graph. However, we can see a small difference at the 1-minute mark, where due to the assigned weight, the user starts to accumulate LAG again. At the consecutive computation, the assigned weight is satisfactory and the user loses the LAG. The other users are not impacted significantly, just like when not using any fuzzy logic controller at all.

INFO:Calculez nou HP
INFO:Unlucky user: 4
INFO:Unlucky user lag: 4
INFO:Unlucky user weight: 0
INFO: Dif nescalat = -4
INFO: Dif scalat = 18
INFO:Fuzzify input
INFO:Fuzzify input
INFO: Result = 32
INFO:Res= 0
INFO:Pondere noua: 1

INFO:Calculez nou HP
INFO:Unlucky user: 4
INFO:Unlucky user lag: 3
INFO:Unlucky user weight: 0
INFO: Dif nescalat = -3
INFO: Dif scalat = 21
INFO:Fuzzify input
INFO:Fuzzify input
INFO: Result = 32
INFO:Res= 0
INFO:Pondere noua: 1