Short description of simple Fuzzy-Logic example

The reason for that example is to demonstrate and explain how fuzzy logic works. I used therefore a very simple (and maybe not realistic) simulation of heating control. This simulation works not with fuzzy logic, but afterwards the fuzzy logic should handle this.

Here I explain how the very simple simulation works (no fuzzy stuff):

Outside temperature influents inside temperature (this means, that the inside temperature will be same like the outside temperature if no other influents will appear).
Radiator temperature influents also the inside temperature, but will be influents by inside temperature itself also (this means, if no other influents should appears, both temperatures will be come together).
The temperature of the radiator will be influents by the heating control. This means, depends on the setting, it will be more or less hot water get into the radiator, which also depending on the setting will influents more or less the radiator.

Compact means that:

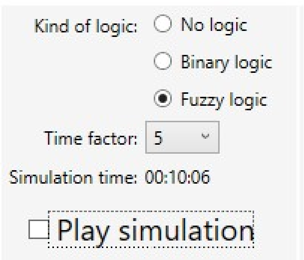
- **Inside** temperature will be influents by **outside** and **radiator** temperature.
- **Radiator** temperature will be influents by **heating control** and **inside** temperature.

The destination is now to find the best heating control setting to get a constant desired temperature.

In the simulation play there are 3 options:

1. If ‚No logic' selected, the user can try by his own to change the heating control.
2. In case of ‚Binary logic', the heating control will switch on to maximum (on) if it's colder than desired, otherwise switch to minimum (off).
3. And the last option is to select ‚Fuzzy logic' (and that is by the way the main reason for that example, isn't it?)

And for the impatient guys, there is a possibility to run the simulation a little bit faster (2, 5, 10, 20, 50 and 100 times). And don't forget to check „Play simulation" to run.



But now the Fuzzy Logik description:

To explain Fuzzy-Logic, I will split this in three parts:

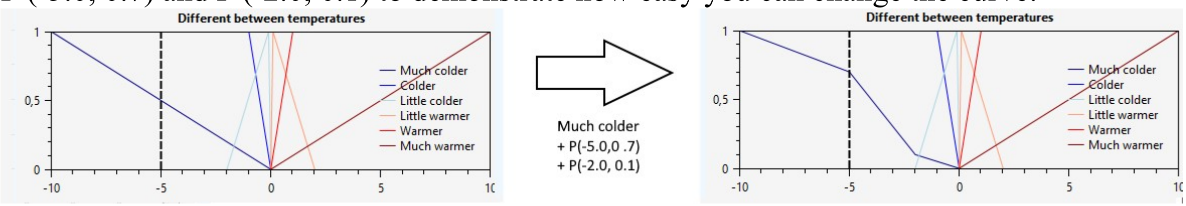
1. Fuzzification: Create by using the input (real) values all necessary fuzzy objects.
2. Implication: Handle the Fuzzy conditions.
3. Defuzzification: Transform the result of the conditions back to a (real) output value.

With **Fuzzification**, it is relatively easy to determine the degree of each corresponding fuzzy object, since there must be a clear function that always delivers exactly one fuzzy degree for each real value. In my example, for a temperature difference of -5°C you can determine that it

is 50% “Much colder” and 10% “Colder”, but it is neither “Warmer” or “Much warmer”. “Little warmer” and “Little colder” doesn’t fall into this area.
During fuzzification, several fuzzy objects can possibly arise, but these always have a clear degree between 0% and 100%.

To determine the specific functions for fuzzification, you have to think about how you want to interpret certain statements such as "Colder" or "Much warmer". I deliberately avoided using different mathematical functions, opted for a list of corner points and determined the value between two corner points using a linear relationship. So you can play around with the values at any time and (if you want or if it’s necessary) make the function more complex to correspond to the true feelings.

In the following screenshot I expanded for you the corner points for "Much colder" by adding P (-5.0, 0.7) and P (-2.0, 0.1) to demonstrate how easy you can change the curve:



Everyone has to determine what the best corner points are for themselves and get a good feeling for them over time. Ultimately, however, it has to match the conditions that I will address in implication now.

In the area of **Implication**, the fuzzy data determined are now processed using conditions. I decided on three different conditions, which are symmetrically mirrored in relation to "Warmer" and "Colder". But this is not absolutely necessary ... as I said: feel free to play around with the values and conditions.

The simplest condition is:
“If temperature (difference) is much colder then heating control open”
According to the fuzzification, I have determined that any temperature difference below 0°C can be regarded as a “Much colder”. Everything below -10.0°C is definitely "Much colder", between -10.0°C and 0.0°C you can say: Yes, it is already "Much colder" but not really 100% "Much colder", maybe only x%. The condition for a example results in -5.0°C and thus a fuzzy degree of 50% for "Much colder" also means that 50% fuzzy degree for "Open".

The other two conditions are somewhat more complex because they contain an AND. Let's take a look at the following condition:
“If temperature is colder and the temperature gets colder, then heating control more open”
Here two different fuzzy types are linked using AND. The fuzzy type for temperature differences such as I have already explained “Colder”, but the same applies to temperature changes as “Gets colder”, the corresponding input values are fuzzified using the function, so that a fuzzy degree of each fuzzy value is available. In case of AND, the smallest of the two fuzzy degree values is now taken as the result. Was e.g. determines 95% "Colder" and 40% "Gets colder", then the result is 40% "More open".
The second of that little be complex condition is:
“If temperature is little colder and the temperature gets fast warmer then heating control more close”
It works similar and also with AND but used “Little colder” and “Gets fast warmer” for the result of “More close”.

Even if I did not use OR and NOT in my program, it should be mentioned at this point: With OR you take the larger of the two values, with NOT you simply invert the value (e.g. 23% then becomes 77%). I do not want to go into further operators at this point.

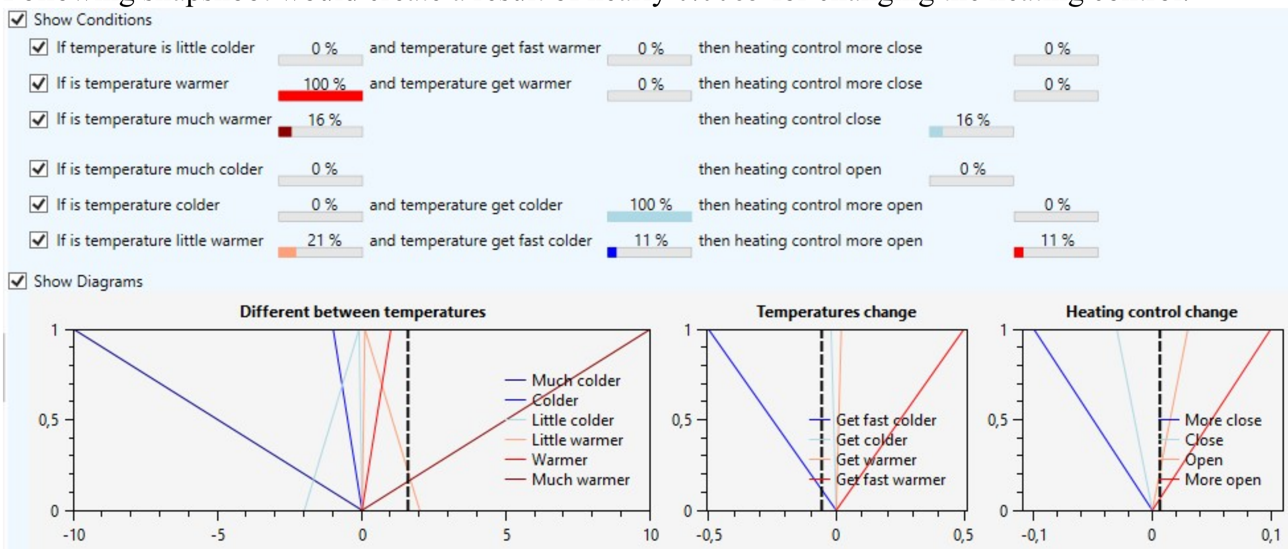
Now that all 6 conditions have been processed, in the best case a corresponding fuzzy degree has been determined for all 4 values of the fuzzy type of the heating control change. These results must now be transformed back to real values in the defuzzification area.

In **Defuzzification** there is now usually more than one fuzzy object (value and degree) for a real output, you first have to decide how to summarize it most sensibly. A few approaches are conceivable for this, in my example I decided to do the three following steps:

The ***first step*** in each case determines the corresponding real value for all fuzzy objects by looking at which sections in the function curve include the particular degree and for this the corresponding real value is calculated linearly. If there are several sections in the function curve with the desired degree value, the mean value is formed from all linearly calculated real values, but since my definition curve also delivers a clear real value for each degree value, the mean value is never formed.
In the ***second step***, I look at all the real values determined and only take the largest positive and the smallest negative real values. The strongest member for open and for close the heating control wins.
In the ***third step***, I summarize the largest positive and the smallest negative value by adding them. In most cases, one of the two values is zero then only the other value is taken into account accordingly. However, should the conditions delivered for both directions a change, than a kind of average of both values is taken (and therefore also the strongest member wins, but reduced by the other one).

And that's all, the result of that is the real value for heating control change. Easy, isn't it?

Following snapshoot would create a result of nearly 0.0065 for changing the heating control.



Degree of 16% for heating control "**Close**" is real value of change -0.0047678
Degree of 11% for heating control "**More open**" is real value of change 0.0112842
Output for heating control change is now 0.0065164!