Getting started with LEADSTO

This document contains a stepwise introduction to the LEADSTO software. It is shown for a specific toy example (in the domain of animal behaviour) how a simulation model can be made. For this example, the user is guided through the process of generating the simulation model (Section 1), running the model (Section 2), and experimenting with the model (Section 3).

1. Creating a simulation model in LEADSTO

The goal of this exercise is to build a simulation model for the example described in Box 1.

> Consider an experimental setting, involving two positions (say, p1 and p2), an animal, a piece of food, and a transparent screen (e.g., a window). Suppose the animal is placed at position p1, the food is placed at p2, and the screen is placed in between, separating the animal from the food. Multiple trials are performed in which, at some variable moment, the screen is raised, and the animal is free to go to any position. After a number of trials, it turns out that a regularity can be observed in the behaviour of the animal. This regularity can be expressed informally by the following property:
>
> *Every time that the agent observes that there is food at p2,*
> *and no screen is present, it will go to p2.*
>
> In semi-formal form, this property can be written as follows:
>
> **LP1 (Local Property 1)**
> at any point in time,
> if the agent observes that food is present at position p2
> and it observes that no screen is present,
> then it will go to position p2

Box 1. Example domain.

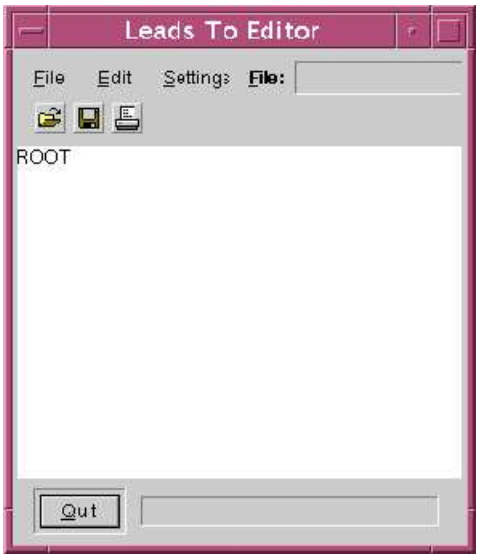
In particular, we will formalise dynamic property **LP1**. To express this property, the following three *state properties* will be used: **observes_food_at_p2**, **observes_no_screen**, and **goes_to_p2**. Notice that the first two are *input state properties*, and the last one is an *output state property*. In order to generate a working simulation model, we will have to do three important things:

- Specify the dynamic property in the LEADSTO editor (see Section 1.1)
- Initialise the input state properties (see Section 1.2)
- Specify the total duration of the simulation (see Section 1.3)

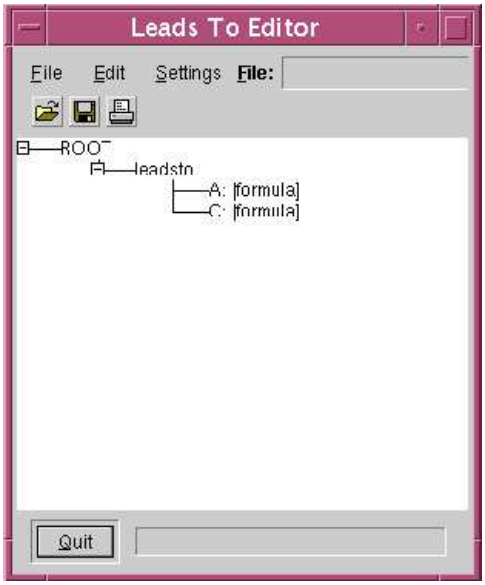
1.1. Specifying the dynamic property

Let us start by introducing **LP1** to the editor. In order to do this, execute the following steps:

a) Download and install the software from http://www.few.vu.nl/~wai/TTL. A detailed installation instruction can be found at the same web site. Then start the LEADSTO editor (`lteditor`). The initial window will look like this:

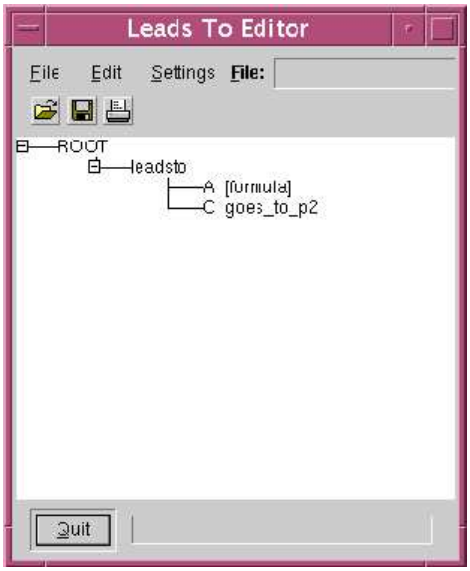


b) Mouse over **ROOT** and click the right mouse button. A menu with nine entries will appear.
c) Move the cursor to the entry **Leadsto**. Click the left mouse button. The following subtree will appear, representing a LEADSTO rule:



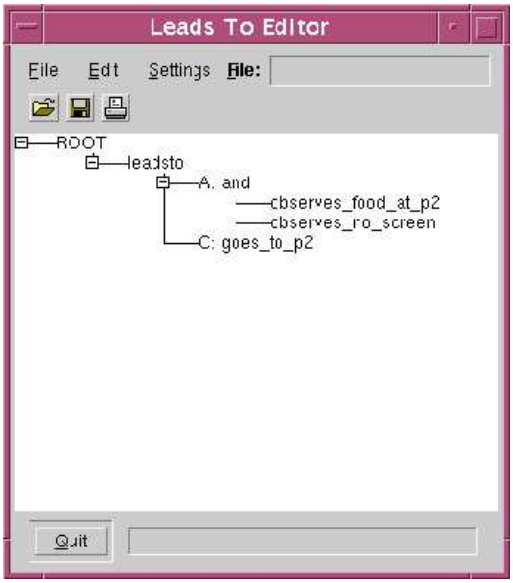
d) The **Leadsto** node has two children **A:[formula]** (A for antecedent) and **C:[formula]** (C for consequent). First, we will change the **C:** node into an **atom** node: click the right mouse button on the node **C:[formula]**. Point to **Change atom** and click the left mouse button. The node will now change into **C:[atom]**.
e) Next, we are going to edit the atom. Right-click on this node and select **Edit**. A window with name "**Atom editor**" will appear:

f) Within this editor, we can replace the line "**[atom]**" by a specific identifier. In our case, replace it by the text **goes_to_p2**. Press button **Ok**. As a result, the main canvas now looks like this:



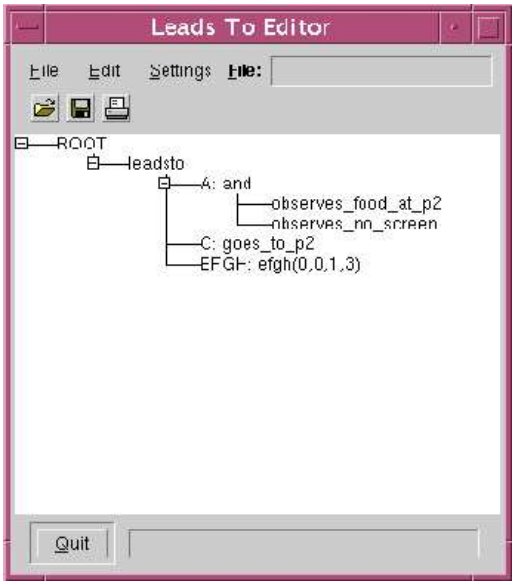
g) Now, we have to change the antecedent. The antecedent consists of the conjunction of two state properties (see Box 1: the property has the format "if X1 *and* X2 then Y"). Therefore, change the **A:** node into an **and** node (by right-clicking on it and selecting **Change and**). A new subtree with two formulae appears.

h) Edit both formulae. First change them to atoms, and then change their names into resp. **observes_food_at_p2** and **observes_no_screen**, in the same way as you did in step e) and f). The result looks like this:



i) Next, we add the time parameters e, f, g and h: Right click on the **leadsto** node and select **add delay**. The node **EFGH: efgh([time],[time],[time],[time])** appears.

j) Edit this node (right click on it and select **Edit**). Replace the **[time]** strings in such a way that the result is **efgh(0,0,1,3)**. By doing this, we express that if the antecedent is true with duration 1.0, then the consequent will be true with duration 3.0, with a delay of duration 0.0. Finally, the main canvas will look as follows:



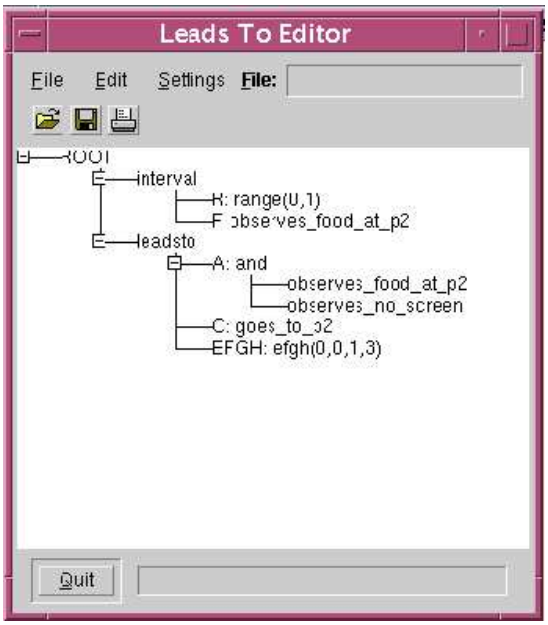
1.2. Initialising the input state properties

We have now successfully entered **LP1** into the editor. However, if we would start the simulation at this point, nothing would happen. The reason is that our property can only have some effect if the input state properties are true at certain time points. Thus, we have to initialise the input state properties (i.e. **observes_food_at_p2** and **observes_no_screen** in our example). In order to do this, execute the following steps:

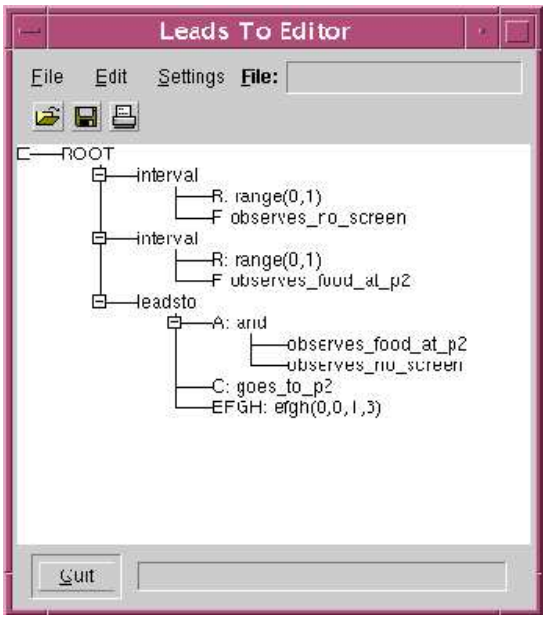
a) Mouse again over **ROOT** and click the right mouse button. This time, select the entry **Interval**. A subtree will appear, representing an interval rule with entries **R** for Range and **F** for Formula.
b) To fill in the range, click right mouse button on the node "**R: range([time],[time])**". Select **Edit**. A window with name "**Range Editor**" will appear:

c) Edit the line "**range([time],[time])**". For the moment, let us assume that the input state properties are only true during the first time unit of the simulation. Thus, replace both times so that the result is "**range(0,1)**". Press button **Ok**. In the canvas you will see the change.
d) Next edit the entry representing the formula, marked **F [formula]** by placing the mouse over that text, right click and select entry **Change atom**. The node will be changed into **F [atom]**

e) Edit this atom by clicking the right mouse button on **F [atom]**, and selecting entry **Edit**. Change the value in the "**Atom Editor**" window into **observes_food_at_p2**. Press button **Ok**. The result in the main canvas will look like this:



f) To include the second input state property, repeat steps a) to e). Again, choose range(0,1), but this time, change the value in the text area into **observes_no_screen**. The result in the main canvas will look like this:

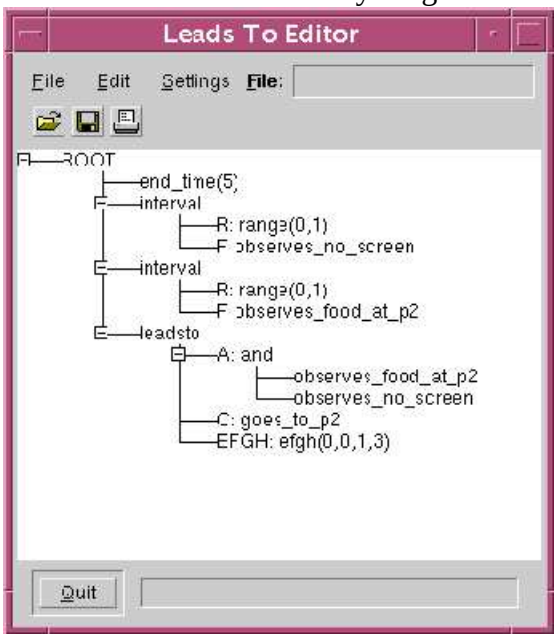


1.3. Specifying the total duration of the simulation

The last thing we need to do is specifying the total duration of the simulation. For example, let us assume that the simulation will last five time units. Execute the following steps:

a) Mouse again over **ROOT** and click the right mouse button. A menu will appear.
b) Within this menu, point to the entry **End time**. Click the left mouse button. A node labeled "**end_time([value])**" will appear.
c) At this place we can fill in the range: click the right mouse button on this node and select **Edit**. A window with name **"Constant Editor for end_time"** will appear:

d) Replace the line "**[value]**" by "**5**". Press button **Ok**. You have now successfully specified your simulation model. If everything went well, the main canvas now looks like this:

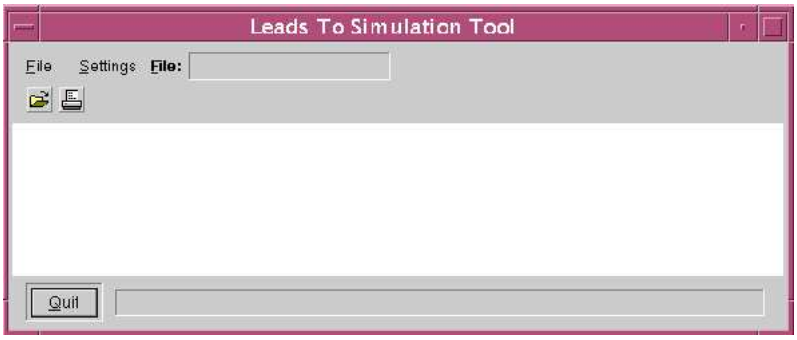


e) To save the specification entered, click on the second icon below the **File** entry (looks like a floppy) and specify a file name (without extension), say **`spec1`**.

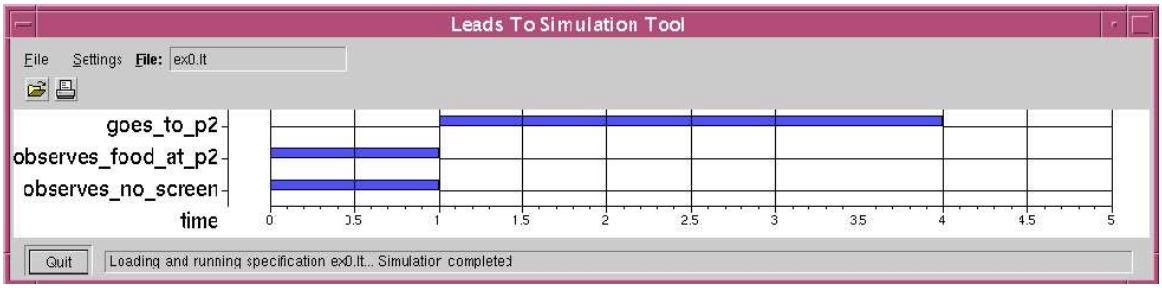
2. Running your LEADSTO specification with the Simulation Tool

In the second exercise you are going to run the specification you just created. In order to do this, execute the following steps:

a) Start the LEADSTO Simulation Tool (**`leadsto`**). The initial window will look like this:



b) To load (and immediately run) the file you just saved, click on the first icon below File (folder icon).
c) A file chooser window will appear. Select the file you saved in the last step of the previous exercise, say **`spec1.lt`**.
d) If you did not make any mistakes, your specification will now be executed. The resulting trace will look like this:



Note that the software automatically stores this trace as file **`trace.tr`** in your working directory.

e) After quitting the LEADSTO Simulation Tool, you can change the name of the trace. As mentioned above, a generated trace is stored by default as **`trace.tr`**. If you want to make sure that your trace is not overwritten by the next simulation, rename it to some other file, such as **`spec1.tr`**.

3. Making experimental modifications to your LEADSTO specification

Now that you have a working LEADSTO specification, try to experiment with the software by making some small modifications to your specification. In particular, execute the following steps:

a) In the first place, see what happens if you choose different delay parameters. In order to do this, execute the **`lteditor`** again, and load your file by clicking on the folder icon. Now, mouse over the **EFGH: efgh(0,0,1,3)** node and click right mouse button. Edit the parameters by giving them some new values, e.g. **efgh(1,1,1,3)** or **efgh(0,0,2,3)**. Do not forget to save your changes! Then go back to the **`leadsto`** simulation software, and re-load your modified file by clicking on the folder icon. Do you see any changes in behaviour? Can you explain these changes?

b) Besides modifying the time parameters of the "leadsto" formula, you can also modify the time parameters of the input state properties (you initialised them in exercise 1.2, remember?). In order to do this, you have to give the "**range(0,1)**" nodes a new value, e.g. "**range(0.5,2.5)**". Another possibility is to give both input state properties different time parameters. Experiment a bit with these time parameters, also in combination with the different efgh-parameters you used in a). Make sure you understand the exact meaning of e, f, g, and h.