A Better Expression of Knowledge To Reduce Spurious Behaviors in Qualitative Simulation

Pascal Jézéquel[†], Laurent Zimmer

Dassault Aviation (DGT/DEA/DIA2)
78, Quai Marcel Dassault - 92214 Saint-Cloud - France
phone (33 1) 47 11 52 95 fax (33 1) 47 11 52 83
e-mail jezequel@dassault-avion.fr or zimmer@dassault-avion.fr

† Université de Savoie, Chambéry, France

Abstract: Qualitative Reasoning has been a field of great interest in various areas of research for the last ten years. Its ability to deal with incomplete knowledge can be very helpful when only a general description of a system or phenomena is available. Obviously, any lack of information in a model may lead to indeterminacy when using this model to make behavioral predictions. Moreover, a calculus based upon qualitative algebras properties induces additional loss of information. Non-transitivity of qualitative equality and indetermination generated when adding negative and positive values are the main causes of ambiguities in qualitative calculus.

Our purpose, in this paper, is to show how it is possible to increase the expression of knowledge in order to avoid ambiguities and generation of irrelevant behavior during qualitative simulation. We will first show that classical mappings of the real[1] operators onto qualitative ones are not sufficient to express all the knowledge embedded in the definitions of the former. We will then describe how a formal calculus can be used to draw additional conclusions using transitivity of the real operators in order to dynamically refine the resolution process and to reject many irrelevant states. We will also present a general method for building functional relations which are very useful to describe characteristics of physical components. We will, finally, illustrate our method by modelling and simulating an example using the SQUALE[2] qualitative simulator.

1 Introduction

Qualitative Reasoning has been used in numerous fields of applications during the last ten years. Qualitative models have been applied to different tasks such as designing, diagnosing or supervising systems that cannot be described by a precise set of equations but about which general laws are known. Qualitative Reasoning is particularly interesting when modeling complex systems composed of numerous interconnected components and for which numerical models are hard to develop because of the difficulty of giving compatible values to all parameters and constants.

We will focus here especially on such kinds of complex systems because they typically match the industrial problems we are daily faced with. Aircraft circuits (hydraulic, pneumatic, electric...) have become more and more complex and it is worth designing them and controlling their design with the help of new CAD tools and simulators. We have developed a new kind of tool to help designers in the early phases of their job. This tool, SQUALE, is based on symbolic and qualitative reasoning. It provides a knowledge representation language that supports qualitative modeling and an inference engine able to predict global behaviors of ancillary circuits.

Qualitative modeling and simulation appear especially interesting when we want to know if the global architecture of an ancillary circuit works as it is expected to. If a contradiction is pointed out during qualitative simulation, it will be not worth continuing with this architecture. Qualitative Reasoning can thus be a powerful tool in design control for comparing the behaviors predicted by the simulation with the behaviors described in the specification without having to go through a complete and costly phase of numerical simulation. However, both qualitative and numerical simulations are then complementary, as a numerical phase may still be necessary to test precise models once qualitative simulation has pruned the incorrect alternatives.

We shall present here the different particularities of SQUALE. We will first describe the basic real operators we are able to handle and the way they are generally expressed in their qualitative form. We will then show that, using these general expressions, we lose a great part of the knowledge embedded in the real operators. In order to recover some of this knowledge, we define a new mapping of the real operators and we extend the inference engine by using formal calculus to compute the transitive closure lost by the qualitative calculus.

1. *In this paper we will call "real operators" the operators and relations with a common mathematical semantics. "Qualitative operators" result from a mapping of the "real operators" onto the sign algebra.*

2. *Simulateur QUALitatif Etendu (Extended Qualitative Simulator).*

Then, we will focus on functional relations giving a general method to represent them, and finally, we will give an example of an ancillary circuit, modeling it using the introduced operators and relations, and give the results of the simulation.

2 Basic operators and relations

Qualitative calculus is based on sign algebra properties. Sign algebra is the smallest algebra we can build from $\mathbb{R}$. Values can be either negative (−), positive (+) or null (0). We will call these values "signs".

SQUALE uses two basic operators and two types of basic relation:

2.1 addition and multiplication

These operators are defined by the following tables:

$\oplus$	−	0	+
−	−	−	?
0	−	0	+
+	?	+	+

$\otimes$	−	0	+
−	+	0	−
0	0	0	0
+	−	0	+

2.2 equality

This relation is defined as:

$[x]^3 \approx [y] \Leftrightarrow [x] = [y]$

2.3 inequality

Few qualitative approaches consider inequality as a basic operator. However, in the case of ancillary circuits, we are daily faced with models expressing comparisons between values or variables. For example, we can be told that *"pressure inside the tank is higher than external pressure"*. It is obviously possible to handle this type of information using the addition operator by defining a positive variable that, added to the lesser, will give us the sign of the higher. The relation $x < y$ can then be mapped to:

$$[pos] = +,\ [x] \oplus [pos] \approx [y]$$

However, this introduces a useless variable.

In our formalism, we consider qualitative inequality as a basic relation. In fact, we give two distinct qualitative relations to express strict inequality ($\prec$) and large inequality ($\preceq$) between two signs. These relations are defined as:

$[x] \preceq [y] \Leftrightarrow [x] < [y]$ or $[x] = [y]$

$[x] \prec [y] \Leftrightarrow [x] < [y]$ or $\{[x],[y]\} = \{-,-\}$ or $\{[x],[y]\} = \{+,+\}$

3. *[x] stands for "sign of the real value x".*

given the general order between signs:

$$- < 0 < +.$$

These relations can be viewed as "poor" and therefore useless in a qualitative formalism because, in terms of signs only, we are not able to distinguish between two positive or two negative values. However, we will show hereafter that, with good expression of knowledge, these operators can help to make very important deductions.

3 The problem of qualitative mapping

If we want to transpose the real operators into their qualitative counterparts, we have to replace values with signs. A basic level of mapping could then be:

$$\begin{array}{ll} x + y = z & \Rightarrow [x] \oplus [y] \approx [z] \\ x.y = w & \Rightarrow [x] \otimes [y] = [z] \\ x = y & \Rightarrow [x] = [y] \\ x < y & \Rightarrow [x] \prec [y] \\ x \leq y & \Rightarrow [x] \preceq [y] \end{array}$$

However, expressing only the signs of qualitative parameters gives us very poor information. We are not able to say, looking at all the parameters, what is the current evolution of the system. We do not capture any dynamics.

We would thus like to represent knowledge about trends of variables. To handle this knowledge, two new qualitative constraints are given:

3.1 derivative

The derivative constraint is considered with respect to time. It is then possible to express dynamics in the model. If we have $X = x(t)$ then the derivative of X will be noted dX/dt. In terms of signs, if we know that dX/dt is positive (resp. negative) then we know that X is increasing (resp. decreasing). If dX/dt is null, then X is possibly reaching an extremum or possibly constant.

3.2 constant

Saying $X = x(t)$ is constant means:

$$\forall t,\ dX/dt = 0$$

3.3 expressing order(s) of derivation

Classical qualitative approaches [DEK 84], [KUI 86], consider qualitative variables as <value,trend> pairs. To deal with trends, it is necessary to assume that all the variables are continuous functions of time and that their derivative does exist. Added to this, the first order derivatives are themselves considered as continuous functions of time. Mathematically speaking, these approaches consider that any qualitative variable is a C^1 function of time (continuous and continuously derivable).

The basic operators can then give informations about the derivatives. If we have X = x(t), a function of time, and X' = dX/dt the derivative of X. We then have:

$$X + Y = Z \Rightarrow X' + Y' = Z'$$

$$X.Y = Z \Rightarrow X'.Y + X.Y' = Z'$$

$$X = Y \Rightarrow X' = Y'$$

The qualitative mapping of the real operators becomes then more precise than the one given before:

$$\begin{array}{ll}
x + y = z & \Rightarrow [x] \oplus [y] \approx [z] \\
 & [x'] \oplus [y'] \approx [z'] \\
x.y = w & \Rightarrow [x] \otimes [y] = [z] \\
 & [x'] \otimes [y] \oplus [x] \otimes [y'] \approx [z'] \\
x = y & \Rightarrow [x] = [y] \\
 & [x'] = [y'] \\
x < y & \Rightarrow [x] \prec [y] \\
x \leq y & \Rightarrow [x] \preceq [y] \\
y = dx/dt & \Rightarrow [x'] = [y] \\
\text{constant}(x) & \Rightarrow [x'] = 0
\end{array}$$

Expressing higher orders of derivation is necessary to capture more information from the real operators. In Kuipers' approach, it is possible to deal with second order derivatives [KUI 87] making some "smoothness" assumptions. Missier [MIS 91] also gives a mapping for the qualitative operators at the second level of derivation and explains how to avoid this smoothness assumption. Yannou [YAN 93] gives the ability to formally derive the operators at even higher orders according to the higher order derivatives expressed in the system.

Finally, the choice of the qualitative mapping is a question of representation depending on the level of information we want to capture and depending, of course, on the task we want to achieve. To compare different qualitative calculators, it is necessary to know the different mappings they use and the level of dynamics they take into account.

In SQUALE, we consider that a qualitative variable is a C^1 function of time. Each variable is represented by a pair <value,trend>. We then have to map operators onto at least one level of derivation.

We do not have to take into account higher orders of derivation since the physical systems we are modeling do not necessitate a deeper representation. If we were dealing with systems with higher dynamics, it would of course be possible to expand our mapping to deal with second or even third order derivatives. However, along with Makarovic and Mars [MAK 89], we believe that qualitative calculus is not well suited for solving highly dynamic systems. Solving such problems at a given level of derivation only passes ambiguities to the immediately superior level. We have then decided, in our case, to stop at the first level. This corresponds, in fact, to the common reasoning level of the engineers we are working with. They are interested in representing ancillary circuits which are mostly quasi-static systems with few dynamic features. Stopping at the first order of derivation is then sufficient to manage this type of circuit.

Because of this restriction, SQUALE is not adapted to solve QDEs expressing dynamic behaviors. For example, if the differential equation X + X'' = 0 is given, SQUALE will only discover a unique oscillating behavior. It will be unable to distinguish between amortized and non-amortized cases.

In any case, even if we choose to express higher orders of derivation to precise the mapping of the real operators, loss of information already occurs at the basic level (concerning only values) if we use the "classical" mappings described above.

3.4 Loss of knowledge

In the first versions of our qualitative simulator, using Kuipers' mapping for QSIM, we were quickly faced with well known phenomena of spurious behavior generation even on rather simple circuits. However, these spurious behaviors were not only due to occurrence branching or indetermination on Higher Order Derivatives. Many of them occurred because the mapping of real operators was insufficient. We describe here some of the problems encountered and explain how we have solved them.

EXAMPLE 1: Let us consider a simple system:

$$x > 0,\ x = z,\ x + y = z$$

Mapping this system at the basic level gives:

$$[x] = +,\ [x] = [z],\ [x] \oplus [y] \approx [z]$$

We expect to find the unique solution

$$[x] = +,\ [z] = +,\ [y] = 0.$$

However, qualitative calculus gives three compatible solutions:

$$\begin{array}{l}
[x] = +,\ [z] = +,\ [y] = -, \\
[x] = +,\ [z] = +,\ [y] = 0, \\
[x] = +,\ [z] = +,\ [y] = +
\end{array}$$

The impossibility of deciding between the three cases is due to the definition of qualitative addition (see the above table). Any qualitative value added with a positive value may give another positive value.

EXAMPLE 2: Let us consider another system:

$$x > 0, x < z, x + y = z$$

Mapping this system into qualitative equations gives :

$$[x] = +, [x] \prec [z], [x] \oplus [y] \approx [z]$$

We expect to find the unique solution

$$[x] = +, [z] = +, [y] = +.$$

However, the qualitative calculus again yields three compatible solutions :

$$[x] = +, [z] = +, [y] = -,$$
$$[x] = +, [z] = +, [y] = 0,$$
$$[x] = +, [z] = +, [y] = +$$

In this case, again, the resolution gives two incoherent qualitative states.

When both variables have the same sign (x and z are both positives in the above case) we lose all the knowledge contained inside the inequality semantics.

Each example clearly leads to two incoherent qualitative states. We must thus enhance our mapping onto qualitative operators and relations in order to avoid such irrelevancies.

3.5 Expressing signs of differences

In the above examples we know that the solutions are unique by 'commonsense' reasoning. We mentally computed the simple system to find out that y had to be null or positive. One way to formalize this computation is to take into account the differences between variables.

Equality $x = z$ also means $z - x = 0$ and $x - z = 0$.

Inequality $x < z$ also means $z - x > 0$ and $x - z < 0$.

Addition $x + y = z$ also means $x = z - y$ and $y = z - x$.

Expressing this knowledge when transposing usual equations into the qualitative formalism is useful to avoid ambiguities. The translation of the above example then becomes:

EXAMPLE 1

$$[x] = +, [x] = [z], [z-x] = 0, [x-z] = 0$$
$$[x] \oplus [y] \approx [z], [y] = [z-x], [x] = [z-y]$$

With this data set, qualitative calculus succeeds in finding the unique solution:

$$[x] = +, [z] = +, [y] = 0$$

EXAMPLE 2

$$[x] = +, [x] \prec [z], [z-x] = +, [x-z] = -$$
$$[x] \oplus [y] \approx [z], [y] = [z-x], [x] = [z-y]$$

With this data set, qualitative resolution gives us the unique compatible solution:

$$[x] = +, [z] = +, [y] = +$$

3.6 Expressing constancy

Other problems occur when we use the constant constraint. Mapping the constancy operator into its qualitative counterpart might be easy. It would be natural to say :

$$constant(X) \Rightarrow [dX/dt] = 0$$

However, things are not so simple !

EXAMPLE 3: Let us consider the system :

$$y = dx/dt, x = dw/dt, constant(w)$$

Transposing this system into its qualitative expression then gives:

$$[y] = [x'], [x] = [w'], [w'] = 0$$

We expect to find the unique solution

$x = [0]$ and $y = [0]$.

However, the qualitative calculus yields three compatible solutions :

$$[x] = 0, [y] = -,$$
$$[x] = 0, [y] = 0,$$
$$[x] = 0, [y] = +$$

EXAMPLE 4: if we consider the system:

$$z = dy/dt, y = x, x = dw/dt, constant(w)$$

Mapping this system gives us:

$$[z] = [y'], [y] = [x], [y'] = [x'],$$
$$[x] = [w'], [w'] = 0$$

We expect resolution to find that x is null but also that both y and z are null. However, the calculus is unable to constrain z enough. Therefore, three compatible solutions are found:

$$[x] = 0, [y] = 0, [z] = -,$$
$$[x] = 0, [y] = 0, [z] = 0,$$
$$[x] = 0, [y] = 0, [z] = +$$

Both of the above problems occur because only part of the definition of the derivative has been expressed. Our translation only considers one level of derivation while the usual definition implicitly considers all levels:

$$\forall t,\ dx/dt = 0 \Rightarrow \quad \forall t, \forall n \in \mathrm{N}^*,\ d^n x/dt^n = 0$$

We must thus take into account all the derivatives of constant parameters expressed in the model.

Moreover, it is possible (as in EXAMPLE 4) that the information of constancy is transmitted indirectly via other basic links (addition, multiplication or equality).

We expose below a general method able to propagate the constancy information to all the variables via the operators.

3.7 Constancy propagation

To be sure that we capture all the information, we associate each parameter of the system with a boolean information concerning its constancy. This information is computed by each operator in order to propagate it throughout the resolution.

- **x + y = z and x.y = z**

c(y) \ c(x)	yes	no
yes	yes	no
no	no	?

(values: c(z))

Table *1*

Propagation of constancy with the addition and multiplication operators

- **x = y**

$$c(x) = c(y)$$

- **y = dx/dt**

c(x)	c(y)
yes	yes
no	no
no	yes

Table *2*

Propagation of constancy by a derivation link

- **constant(x)**

$$c(x) = \text{'yes'}$$

In addition to this, we can define a general relation between the derivative of a variable and the information of constancy. This relation is given by the following table :

c(x)	[x']
yes	0
no	?

Table *3*

General relation between the derivative of a variable and its constancy

Applying these propagators to our examples gives us :

EXAMPLE 3

$$y = dx/dt,\ x = dw/dt,\ constant(w)$$

we can add the following information:

$$c(w) = \text{'yes'},\ c(x) = \text{'yes'},\ c(y) = \text{'yes'}$$
$$[w'] = 0,\ [x'] = 0,\ [y'] = 0$$

We finally find the expected result:

$$[x] = 0,\ [y] = 0.$$

EXAMPLE 4

$$z = dy/dt,\ y = x,\ x = dw/dt,\ constant(w)$$

the resolution tells us that

$$c(w) = \text{'yes'},\ c(x) = \text{'yes'},\ c(y) = \text{'yes'},\ c(z) = \text{'yes'}$$
$$[w'] = 0,\ [x'] = 0,\ [y'] = 0 \text{ and } [z'] = 0$$

We finally get the unique correct result:

$$[x] = 0,\ [y] = 0,\ [z] = 0.$$

3.8 The SQUALE mapping

As we have shown above, a "naive" mapping onto the qualitative formalism leads to a significant loss of knowledge and to the generation of many irrelevant solutions. Each of these irrelevant solutions produces at least one instance of spurious behavior in the qualitative simulation tree.

To reduce these spurious behaviors, we have extended our mapping to capture more knowledge in our qualitative description. In the SQUALE formalism, each real parameter x(t) of the modelled circuit is associated with three additional variables:

- one representing the sign of the parameter: [x],
- one representing the sign of its derivative: [x'],
- one representing the boolean information on constancy of the parameter: c(x).

These three variables are used by each qualitative operator and relation.

Moreover, addition, equality and inequalities apply to other internal variables expressing sign of differences between the involved parameters.

- **x + y = z**
  $[x] \oplus [y] \approx [z]$.
  $[x'] \oplus [y'] \approx [z']$,
  c(x), c(y) and c(z) are linked according to table 1,
  [x'] and c(x) are linked according to table 3
  [y'] and c(y) are linked according to table 3
  [z'] and c(z) are linked according to table 3
  $[x] = [z-y]$,
  $[y] = [z-x]$.
- **x.y = z**
  $[x] \otimes [y] = [z]$
  $[x'] \otimes [y] \oplus [x] \otimes [y'] \approx [z']$
  c(x), c(y) and c(z) are linked according to table 1
  [x'] and c(x) are linked according to table 3
  [y'] and c(y) are linked according to table 3
  [z'] and c(z) are linked according to table 3.
- **x = y**
  $[x] = [y]$
  $[x'] = [y']$
  $c(x) = c(y)$,
  [x'] and c(x) are linked according to table 3
  [y'] and c(y) are linked according to table 3
  $[x-y] = 0$,
  $[y-x] = 0$.

- **x < y**
  $[x] \prec [y]$
  $[x-y] = -$,
  $[y-x] = +$.
- **x ≤ y**
  $[x] \preceq [y]$
  $[x-y] \neq +$,
  $[y-x] \neq -$.
- **y = dx/dt**
  $[y] = [x']$
  c(x) and c(y) are linked according to table 2
  [x'] and c(x) are linked according to table 3
  [y'] and c(y) are linked according to table 3
  $[x-y] = 0$,
  $[y-x] = 0$.
- **constant(x)**
  $[x'] = 0$,
  $c(x) =$ *'yes'*.

It is important to notice here that the qualitative mapping is automatically done by the SQUALE modelling interface. The designer has only to describe the high-level model of the circuit (i.e. using only real operators). The qualitative model is then generated from this high-level description. The designer does not have to "think qualitatively".

4 The problem of transitivity

We have previously given more precise definitions of the basic operators and primitives necessary to build enriched qualitative models. These definitions were directly drawn from the mathematical properties of the operators. In all the above cases, qualitative calculus and constancy propagation were sufficient to drive resolution to good real results. However, transposing knowledge to its qualitative counterpart can still hardly be done without losing some of this knowledge.

The qualitative equality introduced above is a typical example. Let us consider the system :

$$x > 0, x = y, y = z, x + w = z$$

If we map this system into the qualitative formalism described above, we get[4]:

$$[x] = +,$$
$$[x] = [y], [y-x] = 0, [x-y] = 0$$
$$[y] = [z], [z-y] = 0, [y-z] = 0$$
$$[x] \oplus [w] \approx [z], [x] = [z-w], [w] = [z-x]$$

This set of qualitative relations is not able to drive the resolution to the expected solution (x > 0, y > 0, z > 0 and w = 0). Since w is not constrained enough, the three qualitative values –, 0 and + remain compatible.

4. *The mapping of trends and constancy information do not appear here since they have no effect in the reasoning.*

This loss of knowledge is due to the fact that we are not able to directly express that

$$x = y \text{ and } y = z \text{ implies } x = z.$$

The transposition into sign algebra is not conservative with respect to transitivity. This is an inherent property of sign algebra. It is thus illusory to search for a better formalization of qualitative operators. If we want to find a solution to this problem, we have to exit the qualitative formalism and reason at the formal level.

4.1 Qualitative and Formal Calculus for Deduction

The SQUALE qualitative solver is based on the CQFD[5] algorithm. This algorithm deals with two sets of relations : high-level relations expressed in their usual form and their automatically mapped qualitative expressions. The set of qualitative relations is viewed as a constraint satisfaction problem (CSP). The set of high-level relations is kept in the resolution process in order to make formal deductions.

We are able to make obvious deductions by applying algebraic inference rules such as:

$x = y$ *and* $y = z$ *implies* $x = z$
$x = y$ *and* $y < z$ *implies* $x < z$
$x < y$ *and* $y < z$ *implies* $x < z$
$x + y = z$ *and* $x > 0$ *implies* $y < z$
...

These deductions are then mapped into the qualitative formalism to expand the global CSP and give additional information about some variables.

CQFD processes as follow:

1. Solve the constraint satisfaction problem (CSP)
2. Apply inference rules to make formal deductions.
3. Map the deductions into qualitative expressions.
4. Expand the old CSP and solve the new one.
5. Get all the compatible solutions.

This algorithm uses two passes of the constraint solver because the first pass can find local information (as $x > 0$ above) which can lead to the firing of useful rules during the step of formal deduction.

4.2 Adding new algebraic inference rules

The combination of qualitative calculus and algebraic inference rules necessitate a compromise between expressive power and complexity. It is obviously of great interest to add new inference rules in order to make more algebraic deductions. However, this increases the complexity in step 2 of the CQFD algorithm. We thus prefer to limit the set of inference rules and to refine as far as pos-

5. *Calcul Qualitatif et Formel pour la Déduction (Qualitative and Formal Calculus for Deduction).*

sible the expressions of the qualitative operators to draw more conclusions from the calculus.

4.3 Incompleteness of Qualitative Simulation

Qualitative simulation has been proved sound but incomplete [KUI 86]. This means that qualitative simulation is able to find all the real behaviors of the modelled systems but can also generate spurious behaviors which cannot occur practically. The formal step of our algorithm eliminate many spurious behaviors generally due to non-transitivity of the qualitative operators. We still cannot assure completeness because we cannot make all possible inference and we are not able to capture all the knowledge of the real operators in their qualitative counterparts as we must limit the mapping at one given level of derivation. However, in the case of quasi-static systems such as ancillary circuits, we are able to prune most of the spurious behaviors by mapping only onto a first level of derivation.

4.4 Extended qualitative calculus in other work

Some other approaches also add a step of formal or semi-quantitative calculus to refine the results of the simulation. Berleant [BER 88] and Tréteault [TRE 92] extend the mapping of the operators to manage some quantitative information to reduce ambiguities. Simmons' Quantity Lattice [SIM 86] and Williams' MINIMA calculator [WIL 88] also merge the algebras of signs and reals to make formal deductions. SQUALE differs from these other "enhanced" qualitative systems in using only sign algebra. Formal deductions are made at a purely symbolic level and no quantitative information is required to reduce ambiguities. SQUALE is then closer to Farquhar's QPC, [FAR 94], in which algebraic rules are also used to add new qualitative constraints into QSIM's network.

5 Building functional relations

If the only manageable qualitative operators and relations were those described above, qualitative calculus would not be justified since models would be better suited to numerical simulators. The main advantage of qualitative reasoning is its ability to deal with partial or approximate knowledge.

In the Kuipers' formalism [KUI 86], such approximate knowledge is expressed by using relations named M+ (resp. M−), signifying that two variables are increasing (resp. decreasing) functions of each other.

Functional relations are necessary and well suited to deal with physical parameters. However, monotonic functional relations do not represent all classes of functional relations. Many physical behaviors are expressed by polynomial relations or combinations of monotonic behaviors. Kuipers gives, for example, the ability to express U+, U−, S+ and S− behaviors but these specific functional relations do not cover the whole range of possible functional relations needed to model real behaviors (see fig. 1). It is also possible, in QSIM, to compose monotonic functions in order to build any specific function but it is then necessary to define operating regions and a way of switching from one to another.

In the following, we propose a way of defining functional relations as global qualitative constraints by assembling three types of monotonic behaviors M+, M− and Mo (constant).

These relations are mapped, at a first order of derivation, as:

M+(x,y) $\Rightarrow$ *[x'] = [y']*
M−(x,y) $\Rightarrow$ *[x'] = −[y']*
Mo(x,y) $\Rightarrow$ *constant(y)*

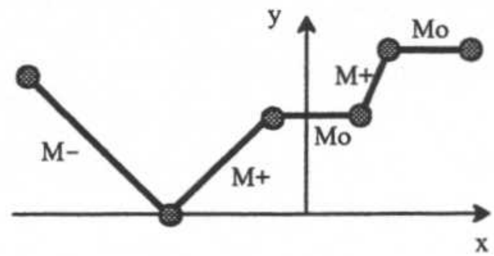


Figure 1
Any functional relation can be defined by assembling together three types of monotonic segments.

To define a functional relation, we have to describe all the corresponding points at which the general behavior is possibly changing. In order to respect the continuity properties of the C^1 variables, we cannot change from a M+ segment to a M− segment without the derivative dy/dx becoming null. Each corresponding point separating two segments of different behaviors is then an extremum where [dy/dt] = 0.

The modeler is also given the ability to define free extrema on the extremities of the relation and between two segments having identical behaviors.

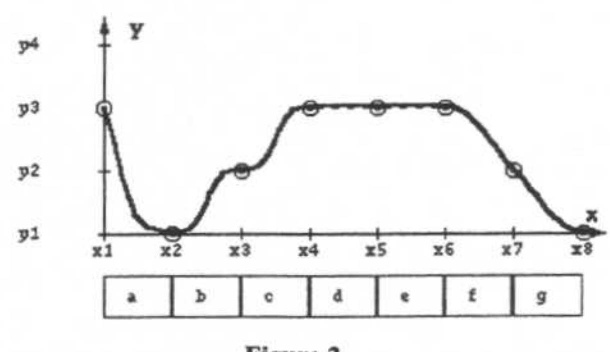


Figure 2
Example of a point to point constructed functional relation in SQUALE.. Each 'region' located between two points can be labelled.

The point to point construction of a functional relation is then computed by SQUALE as a set of qualitative constraints mapped from the M+, M− and Mo relations.

6 Modelling a circuit with SQUALE

The different operators and constraints described above are generally sufficient to model many physical systems. We will deal here with an example representing a basic hydraulic circuit that can be used to generate pressure and flow to activate brakes, flaps or flight controls in an air-craft.

This circuit is composed of:

- a self–regulated hydraulic pump used to generate flow and pressure in the circuit,
- a pressure–reducing valve used to evacuate part of the fluid in case of surpressure,
- a tank used as fluid supply,
- an actuator used to activate the required parts of the aircraft (flaps, traps...),
- a four–port, three–position, hand–operated directional control valve dispatching or blocking the flow to the actuator.

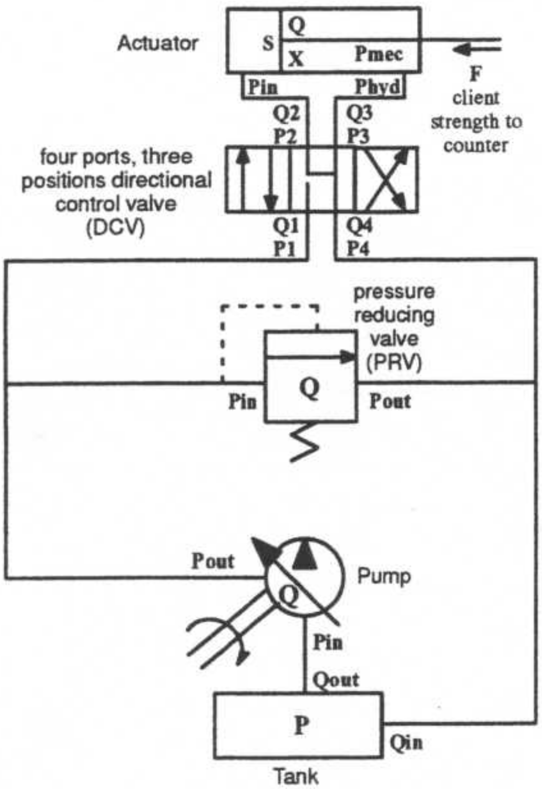


Figure 3
The hydraulic generation circuit

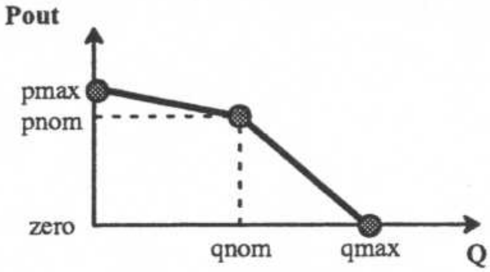
The tank is modelled as follows:

vmin < V < vmax
DV = dV/dt
DV = Qin − Qout

We assume moreover that the internal tank pressure is negligible compared to other pressures:

P = zero

The pump is modelled by a functional relation:



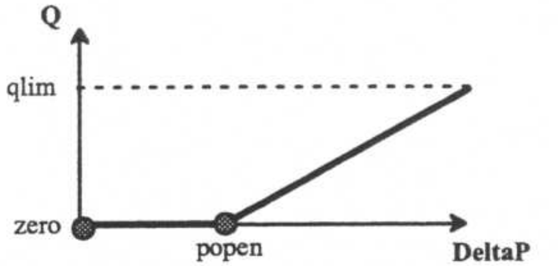
The directional control valve is modelled as follows:

left position:
Q1 = Q2, Q3 = Q4, P1 = P2, P3 = P4

middle position:
Q1 = 0, Q2 + Q3 = Q4, P2 = P3 = P4

right position:
Q1 = Q3, Q2 = Q4, P1 = P3, P2 = P4

The pressure reducing valve is modelled by a functional relation:



with *DeltaP = Pin − Pout*

The actuator is modelled as follows:

the client strength F is assumed to be a constant equal to the symbolic value "f"
the mechanical pressure Pmec applied by the client is an increasing function of F
Phyd + Pmec = Pout
DeltaP = Pin −Pout
Acceleration is an increasing function of DeltaP
Acceleration = dSpeed/dt
Speed = dX/dt
the section S of the actuator is a constant equal to the symbolic value "section"
S.Speed = Q

6.1 Positioning symbolic values

The model refers to symbolic values which have to be positioned relative to each other to describe the circuit. In our example, we have chosen the following positioning:

$zero < pnom < pmax < popen$
$zero < qnom < qmax < qlim$

We also have to mention corresponding values in the relation between F and Pmec in the actuator:

$Pmec = pf$ when $F = f$

We consider that the nominal pressure of the pump is sufficient to counter this pressure:

$pf < pnom$

6.2 Connecting equations

We must then connect the components together in order to propagate pressures and flows all along the circuit. We consider here ideal connections without any loss of charge or energy. We are then able to express two general laws relating flow and pressure at the connecting points:

Pressures are all equal.
The algebraic sum of all flows is null.

We can notice that these two laws are equivalent to Kirschoff's laws transposed from the electric field to the hydraulic field. Here we deal with flows and pressures instead of intensities and tensions.

Considering the above circuit, we get:

$$Tank\text{-}Qout + Pump\text{-}Q = 0$$
$$Pump\text{-}Pin = Tank\text{-}P$$
$$Pump\text{-}Pout = PRV\text{-}Pin = DCV\text{-}P1$$
$$Pump\text{-}Q + PRV\text{-}Q + DCV\text{-}Q1 = 0$$
$$DCV\text{-}Q2 + Actuator\text{-}Q = 0$$
$$DCV\text{-}P2 = Actuator\text{-}Pin$$
$$DCV\text{-}Q3 + Actuator\text{-}Q = 0$$
$$DCV\text{-}P3 = Actuator\text{-}Phyd$$
$$Tank\text{-}Qin + PRV\text{-}Q + DCV\text{-}Q4 = 0$$
$$Tank\text{-}P = PRV\text{-}Pout = DCV\text{-}P4$$

6.3 Simulating

Once the global model is built, it is translated into a qualitative one. To do this, we have developed a general tool able to compute general information given by the designer and draw additional conclusions concerning, for example, partial orders between symbols. The connecting equations are also generated by this tool.

The complete model is then compiled into the qualitative formalism. General laws are mapped, as described previously, into qualitative constraints. Parameters are associated with quantity spaces containing only the required symbols. The set of all parameters, operators and relations is viewed as a Constraint Satisfaction Problem which can be solved by SQUALE using the CQFD algorithm stated above. Applied to the previous example, the simulation is able to predict the correct behaviors corresponding to the chosen configuration of the control valve. The actuator can then move to the right (DCV in left position) or to the left (DCV in right or middle position). In all cases of configurations, we get only possible states (one <value, trend> pair for each parameter) and no spurious behavior is generated. We give hereafter some graphical outputs from SQUALE corresponding to the first configuration (DCV in left position).

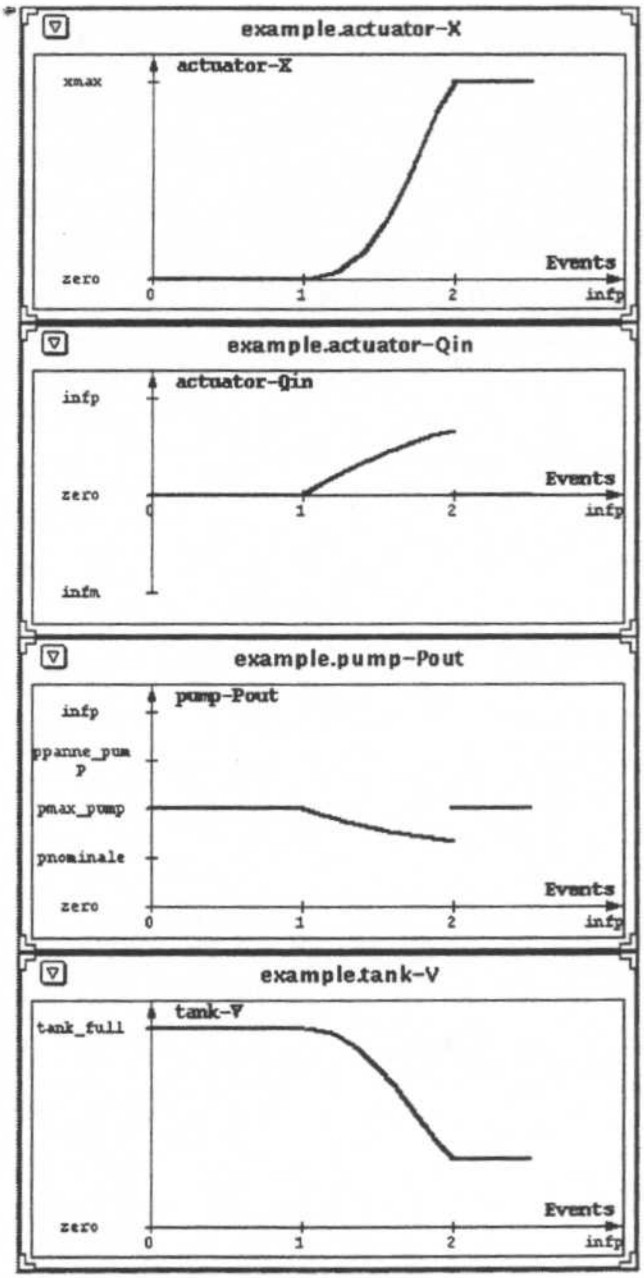


Figure 4

Some outputs from SQUALE applied on the given example.
The control valve, initially closed, is opened at left position. The pump pressure is then able to push the actuator which moves to its extreme right position. When moving, the actuator consumes flow and volume and generates a pressure drop of the pump and a diminution of volume in the tank.

7 Conclusion

Qualitative calculus can be used to draw conclusions from physical systems even if some information is missing. This can obviously lead to several different predictions concerning the behaviors of those systems. We would ideally like to constrain branching of behaviors only to the lack of knowledge coming from the user's description and to avoid indeterminacy due to the qualitative calculus "limitations".

We have shown that it is possible to improve qualitative calculus by a better expression of the knowledge embedded into the different real operators. To solve ambiguities due to the intrinsic properties of sign algebra, we have added a set of formal inference rules able to make transitivity propagation between the parameters using the "real" properties of the operators.

All these improvements aim at drawing a maximum of conclusions at the basic level as formal inference rules do not deal with trends or higher orders of derivations. Our goal is not here to build a general tool able to solve highly dynamic systems or sets of complex QDEs. We are trying to make a tool able to take into account ancillary circuits with relatively low dynamics.

The discussed work has been implemented into the SQUALE simulator which is used to assist designers and to predict general behaviors of aircraft circuits in the early stage of design. Using this tool, engineers are then able to compare general alternatives and to eliminate those which do not fit the specifications.

We are now intending to extend the functionalities of our tool to new qualitative operators in order to express more knowledge about the modelled systems. Designers often use simplifications in their systems by considering that some parameters are much greater or evolve much faster than others. We would then like to express this kind of knowledge in our models. Works are going on to improve SQUALE in order to make model based diagnosis [MIS 93]. We would like also that specific questions such as "can this parameter reach this value?" or "show me all the behaviors fitting this specification" could be answered as it is done, for example, in Yannou's QDES [YAN 93] by intersecting a functional specification representing the desired behavior and the behaviors found during the simulation.

References

[BER 90]: D. Berleant and B. Kuipers. *Qualitative-Quantitative simulation with Q3.* Proceedings of the fourth International Workshop on Qualitative Physics, QP-90, pp 140-152, 1990.

[DEK 84]: J. De Kleer, J.S. Brown. *A Qualitative Physics based on Confluences.* Artificial Intelligence, 24, pp 7-83, 1984.

[FAR 94]: A. Farquhar. *A Qualitative Physic Compiler.* Proceedings of AAAI 94, pp 1168-1174, 1994.

[KUI 86]: B.J. Kuipers. *Qualitative Simulation.* Artificial Intelligence, 29, pp 289-338, 1986.

[KUI 87]: B.J. Kuipers and C. Chiu. *Taming Intractable Branching in Qualitative Simulation.* Proceedings of IJCAI 87, pp 1079-1085, 1987.

[MAK 89]: A. Makarovic and N. Mars. *Fundamental Limitations of Qualitative Simulation.* Internal report of the University of Twente, Netherlands, 1989.

[MIS 91]: A. Missier. *Structures Mathématiques pour le Calcul Qualitatif, Contribution à la Simulation Qualitative.* PhD Dissertation, INSA de Toulouse, France, 1991.

[MIS 93]: A. Missier, L. Zimmer, P. Jézéquel, L. Travé-Massuyès. *The SQUALE qualitative simulator, potentialities for diagnosis.* Proceedings of TOOLDIAG'93, pp 1054-1062, Toulouse, 1993.

[SIM 86]: R. Simmons. *'Commonsense' Arithmetic Reasoning.* Proceedings of the AAAI 86, pp 118-124, 1986.

[TRE 92]: M. Tréteault, B. Marcos, J. Lapointe. Temporal Duration *Reasoning in Qualitative simulation.* Artificial Intelligence in Engineering, 7, pp 185-197, 1992.

[WIL 88]: B. Williams. *MINIMA: a Symbolic Approach to Qualitative Algebraic Reasoning.* Proceedings of the AAAI 88, pp 264-269, 1988.

[YAN 93]: B. Yannou. *Aide à la Conception Qualitative de Systèmes Dynamiques.* PhD Dissertation, Ecole Normale Supérieure de Cachan, France, pp 91-173, 1993.