# THE USE OF PARTIAL QUANTITATIVE INFORMATION WITH QUALITATIVE REASONING

APPROVED BY

SUPERVISORY COMMITTEE:

_____

_____

_____

_____

_____

_____

© 1991 by Daniel Berleant

To my wife, daughter

and

all those who have

been at my side

during my graduate school years

# THE USE OF PARTIAL QUANTITATIVE INFORMATION WITH QUALITATIVE REASONING

by

JARED DANIEL BERLEANT, B.S., M.S.

**DISSERTATION**

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

**DOCTOR OF PHILOSOPHY**

THE UNIVERSITY OF TEXAS AT AUSTIN

AUGUST, 1991

# Acknowledgments

The author wishes to acknowledge his friends and associates during graduate school, and in addition Jo Decuyper and Alex Yeh, for comments which perhaps seemed minor at the time but ended up being of consequence.

JARED DANIEL BERLEANT

*The University of Texas at Austin*
*AUGUST, 1991*

# THE USE OF PARTIAL QUANTITATIVE INFORMATION WITH QUALITATIVE REASONING

Publication No._____

JARED DANIEL BERLEANT, Ph.D.
The University of Texas at Austin, 1995

Supervising Professor: Benjamin J. Kuipers

There is a need for combining qualitative and quantitative simulations, to do simulation tasks that would be difficult using either alone. This task is made more difficult by the fact that available quantitative information may be incomplete, bounding values with intervals or describing them with probability distribution functions. This research demonstrates the combination of qualitative and quantitative simulation in an implemented system, Q3. Q3 utilizes partial or complete quantitative information, to gradually refine a qualitative simulation into a simulation that has properties and advantages of both qualitative simulations and quantitative ones.

The technique exemplified by Q3 is shown to possess properties often used in analyzing both qualitative and quantitative simulators. Qualitative and quantitative inferences are correct. Theoretical convergence to the true solution and stability in the presence of partial model inputs are also shown.

Q3 has been applied to the problem of finding probabilities of qualitative behaviors, an important problem. Partial quantitative characterization of model inputs, in the form of intervals and probability distributions, may be used to bound the probabilities of different behaviors. This is demonstrated for simple models including one in the dependability analysis application domain.

# Table of Contents

**Acknowledgments** iv

**Abstract** v

**Table of Contents** vii

**List of Tables** xii

**List of Figures** xiii

**1. Introduction and Overview** 1

1.1 Motivating Overview . . . . . 1

1.2 Specific Benefits of Qualitative-Quantitative Simulation . . . . . 4

1.3 Previous work . . . . . 7

1.3.1 Qualitative-quantitative work in VLSI . . . . . 7

1.3.2 Qualitative-quantitative spatial reasoning . . . . . 8

1.3.3 Qualitative-Quantitative work in temporal ordering . . . 8

1.3.4 Interval and inequality reasoning . . . . . 9

1.3.5 Qualitative-quantitative simulation work . . . . . 10

1.4 Outline of Remaining Chapters . . . . . 14

**2. Q2: Adding Interval Information to Qualitative Simulation** 15

2.1 Overview . . . . . 15

2.2 Propagation of Incomplete Quantitative Information . . . . . 16

2.2.1 Types of quantitative propagation constraints . . . . . 22

2.2.2 The propagation algorithm . . . . . 27
2.3 Previous Work . . . . . 29
2.4 Discussion . . . . . 30
2.4.1 Constraint based quantitative reasoning and other qualitative simulation approaches . . . . . 31
2.4.2 Quantitative processing: during versus after qualitative simulation . . . . . 32

**3. Supplemental Algorithms for Interval Inference: Target Interval Partitioning and Behavior Splitting 33**

3.1 Target Interval Partitioning . . . . . 33
3.1.1 Variations and extensions . . . . . 35
3.2 Behavior Splitting . . . . . 37
3.2.1 Behavior splitting in simulation . . . . . 39
3.3 Conclusion . . . . . 39

**4. Time Point Interpolation: Bringing Qualitative and Quantitative Simulation Closer Together 40**

4.1 Introduction . . . . . 40
4.2 Brief Overview of Previous work . . . . . 40
4.3 A Representation for Combined Quantitative and Qualitative Simulations . . . . . 41
4.4 Increasing the Power of Inference . . . . . 45
4.4.1 The knowledge representation . . . . . 45
4.4.2 Extracting better results . . . . . 47
4.4.3 A larger example . . . . . 47
4.5 The Benefits to Quantitative Simulation . . . . . 49
4.6 Summary . . . . . 49

**5. Progressive Simulation Refinement and *Q3*** **51**

5.1 Step Size Refinement . . . . . 51

5.1.1 Qualitative simulation . . . . . 52

5.1.2 Propagating quantitative information . . . . . 52

5.1.3 Locating a gap . . . . . 54

5.1.4 Interpolating a state . . . . . 56

5.1.5 Refining the quantitative intervals . . . . . 56

5.1.6 Iteration . . . . . 57

5.2 Example: The Returning Rocket . . . . . 57

5.3 Correctness, Convergence, and Stability . . . . . 62

5.3.1 Correctness . . . . . 62

5.3.2 Convergence . . . . . 64

5.3.3 Stability . . . . . 67

5.4 Discussion . . . . . 68

5.4.1 Alternatives to the Mean Value Constraint . . . . . 69

5.4.2 Step Size Refinement on Other Variables . . . . . 70

**6. Applications of Q3** **71**

6.1 Diagnosis . . . . . 71

6.1.1 Reasoning with models, behaviors, and values . . . . . 71

6.1.2 The pressure regulator: differential diagnosis based on quantitative predictions . . . . . 72

6.1.3 The proportional thermostat: a more complex example . 77

6.2 Measurement interpretation . . . . . 80

6.2.1 The rocket revisited . . . . . 80

6.2.2 Previous work on measurement interpretation . . . . . 81

6.3 Some other applications . . . . . 82
6.3.1 Tolerance analysis . . . . . 82
6.3.2 Probability distribution functions (pdfs) . . . . . 82
6.3.3 Other potential applications . . . . . 83

**7. Application: Probabilities of Qualitative Behaviors 84**

7.1 Overview . . . . . 84
7.2 Histogram Discretization . . . . . 84
7.3 Using Histogram Discretization to Find Probabilities of Qualitative Behaviors . . . . . 90
7.4 The Partially Stopped Bathtub . . . . . 91
7.5 Deriving the Probabilities of Behaviors from a Pdf . . . . . 94
7.6 Limitations of the Algorithm . . . . . 98
7.7 Discussion . . . . . 99

**8. Probabilities of Qualitative Behaviors for Dependability Analysis 102**

8.1 Overview . . . . . 102
8.2 Introduction . . . . . 103
8.3 A Dependability Model Example . . . . . 104
8.3.1 Finding the consistency of a behavior . . . . . 107
8.4 Discussion . . . . . 110
8.4.1 Excess width issues . . . . . 110
8.4.2 Mixing intervals and pdfs . . . . . 111
8.4.3 Ordering probability intervals . . . . . 111
8.5 Conclusions . . . . . 111

**9. Concluding Summary** **113**

9.1 Future Work . . . . . 114

**A. The Lorenz attractor** **116**

**B. Proof of convergence and stability for step size refinement.** **117**

**BIBLIOGRAPHY** **123**

Vita

# List of Tables

2.1 Use of an ADD constraint in the bathtub simulation of figure 2.5. 25

5.1 A qualitative model describing the gravity experienced by a rocket or other object fired upward. Gravity decreases with distance according to the standard nonlinear 2nd order differential equation $\frac{d^2r}{dt^2} = \frac{-GM}{r^2}$. . . . . 59

5.2 The returning rocket simulation at various stages in its refinement. Each technique available to Q3 has been used at some point in the simulation, as dictated by events occuring during the simulation. Model variable values are shown at times $time = 153$ (which happens to be the first time point interpolated by the step size refinement algorithm), `T1` when the rocket is at its highest point, and/or `T2` when it hits the ground on the way down. . . . . 61

6.1 Effects of two different measurements. The figures in the top line are from figure 1.1 and reflect no measurement, the weakest possible measurement condition. The middle line shows the effects of refining the simulation with an incompletely specified measurement. The last line reflects a strong measurement. Note how the predicted intervals tend to get narrower as the simulation is refined with stronger and stronger measurement conditions. . . 81

8.1 Some dependability results for the 1-tank dependability model of figure 8.1. Two pdf shapes were tested. Results for an isosceles shape, $\wedge$, are shown in parentheses. . . . . 109

# List of Figures


1.1 A rocket is fired upward with initial velocity R* $\in [10, 20] km/sec$. Since escape velocity is 11 $km/sec$., The rocket may possibly fall back to Earth (behavior 1, shown) although it will most likely escape. . . . . 5

1.2 Combined simulation of an air conditioned dwelling. The time points in this simulation come from three sources. The qualitative simulation generated time points `T0, T1, T2, T3` and `T4`. New states representing the times `100, 1000`, and `10000` were interpolated into the gap between `T0` and `T1`. New states were then created by interpolating landmarks representing 79.5° and 81.5° fahrenheit into the quantity space of `Inside Temperature`, resulting in the time points `K, K0, K1`, and `K2`. . . . . 11

1.3 A fuzzy interval. The membership $\mu$ of numbers on the $x$-axis is less certain, the lower the value of $\mu(x)$. . . . . 13

2.1 Qualitative model schematic for a 1-tank "Bathtub" system. The model is valid when there is a definite level in the bathtub. (At very low rates of inflow, water forms a rivulet running across the surface of the tub and level is undefined, although amount is positive. Alternatively, where the faucet is directly over the drain so that inflow never touches the tub, there is no level as such even though a positive amount of water is within the confines of the tub before passing out the drain.) . . . . . 17

2.2 The QSIM code describing the bathtub model. The system was initialized to empty (`amount`=0) with `inflow=if*`. . . . . 18

2.3 A qualitative simulation of the bathtub system, showing overflow and equilibrium behaviors. . . . . 19

2.4 Quantitative information provided Q2 for the bathtub model. Envelopes partially specify monotonic constraints (figure 2.7). . 20

2.5 Q2 applied to one member of the 1-tank class of models, the venerable bathtub. Of three behaviors, two equilibrium behaviors were ruled out by Q2, leaving only the overflow behavior shown. 21

2.6 Waltz filtering using interval labels is done on an addition constraint. Waltz filtering is correct, in that possible values won't be filtered out [Waltz 75; Davis 87]. . . . . 23

2.7 Propagation through an $M^+$ (monotonically positive) function is shown graphically. Shaded regions of the axes represent numerical intervals, and the upper and lower envelopes represent numerical envelope functions. Propagation shown is from Y to X; propagation from X to Y is also attempted by the constraint processing code. . . . . 24

3.1 **Target Interval Partitioning** . . . . . 36

4.1 Qualitative simulation of an object dropped from about 8 feet. . 42

4.2 Time 0.45 is present implicitly between `T0`$= [0, 0]$ and `T1`$\geq 0.489$ (see figure 4.1). . . . . 43

4.3 Simulating the oscillation of a frictionless spring using time point interpolation several times. . . . . 48

5.1 A rocket is fired upward at $3000 m/sec.$, less than escape velocity. 17 time points were interpolated, and the TIP and behavior splitting algorithms were also used when necessary (see text). Table 5.2 summarizes some of the key intermediate stages in the simulation. . . . . 53

5.2 Convergence occurs despite excess width in interval calculations. A simulation for $\frac{dx}{dt} = \frac{x+1}{x}$ reveals a slightly concave down curve at three different amounts of step size refinement. When just two new time points are inserted (at $t = .5$ and $t = 1$) the uncertainty in $x$ increases rapidly, as shown by the two tall ranges. After refinement with ten time points, the trajectory is significantly more constrained, as shown by the ten range indicators of intermediate height. Refinement with 100 time points leads to much more constrained results as shown by the 100 small range indicators. . . . . 66

6.1 In addition to reasoning about quantitative values, Q3 can reason about behaviors and models from the available quantitative information. . . . . 73

6.2 **Pressure regulator device.** Fluid enters at left with input pressure $P_{in}$, passes through the valve with area $V_{area}$, and exits at right with output pressure $P_{out}$. $P_{out}$ is transmitted to the chamber at the top, where it pushes down on the valve, whose spring pushes back with force $F_s$. The higher $P_{out}$ is, the greater the equilibrium value of $F_s$ and consequently the more restrictive the valve. This feedback tends to moderate the transmission of variations in $P_{in}$ to $P_{out}$. . . . . 74

6.3 A diagram depicting the pressure regulator model of figure 6.4. Underlined model variables are held constant (Rload is the resistance of some load attached between Pout and the sump, defined as having $pressure = 0$). . . . . 75

6.4 Code defining the qualitative differential equation (QDE) for the pressure regulator model. . . . . 76

6.5 Qualitative simulations of working and stuck pressure regulators. The correspondences of the model constraints imply that, given RLOAD* (constant, not shown), then PIN* implies (indirectly) an output pressure of POUT*. Subjected to higher input pressure of PIN+, both regulators produce output pressure higher than POUT*. Thus it would be hard to use the qualitative predictions to match a real regulator with the working or stuck model without observing transient behavior, which may be difficult in practice. . . . . 78

6.6 The simulations of figure 6.5 augmented with quantitative processing. The final steady state for each regulator provides a clear, easily observable distinction between the two models in their predictions for final output pressure. Thus there is no need to measure transient properties of system behavior to make a differential diagnosis. (Predictions for PO, QO and Q* are slightly narrower in (a) because the behavior contains more constraints, resulting in better, but overlapping and thus consistent intervals.) . . . . . 79

7.1 Multiplying histogram representations of pdfs. The cartesian product of histogram discretizations for the inputs is used. The probability masses of the bars are indicated in their interiors. . . 86

7.2 A qualitative simulation of the bathtub, with partial quantitative information added. Both behaviors are consistent with the initializing intervals. Note that while initial inflow is specified as [0, 1.6], each behavior turns out to be consistent with different subsets of that interval. . . . . 93

7.3 A pdf for inflow and behavior probabilities based on it. . . . . 95

7.4 Inflow has a 0.25 chance of being in [0, 0.57]. If it is in [0, 0.57] then only the equilibrium behavior is plausible, shown here. One branch of the behavior tree in the upper right corner is marked with an "X" to indicate that the overflow behavior has been ruled out. . . . . 97

8.1 Simple dependability model for a larger system. If elements fail faster than they are repaired, eventually the failed element tolerance is exceeded and the system crashes. If this occurs after some time $t$ of interest, the system is dependable. (This model is compared with a standard Markov dependability model in section 8.5.) . . . . . 105

8.2 An isosceles pdf is used to derive probability ranges for behaviors of a system. . . . . 106

8.3 Constraint network representing a behavior of the model of figure 8.1. . . . . 108

# Chapter 1

# Introduction and Overview

This chapter starts with a motivating overview of qualitative-quantitative simulation, proceeds to a list of those specific reasons often cited for combining qualitative and quantitative reasoning, and then provides a review of the literature. It closes with an outline of the remaining chapters, each of which is intended to be largely self-contained and independently readable.

## 1.1 Motivating Overview

The often noted Lorenz attractor showed the impossibility of long term weather forecasting and provided the first milestone in the discovery of the theory of *chaos*, or nonlinear dynamics [Gleick 87]. The path Lorenz' set of 3 simple equations takes through its 3-D parameter space has 2 *qualitatively* distinct "wings". Each wing contains infinitely many paths, yet the space containing those paths remains bounded in extent (figure, appendix A). Systems like the Lorenz attractor help illustrate the potential value of combining qualitative reasoning — in this case to distinguish each wing — with incomplete quantitative information, to characterize the numerical bounds of a particular wing.[1] Such systems help make the future of qualitative-quantitative simulation interesting.

Much work has been done on quantitative simulation. This includes both numerical simulation work, well known and in general use, as well as interval simulation [Eijgenraam 81]. More recently, artificial intelligence has given rise to the area of qualitative simulation [Bobrow 85; Weld and de Kleer 90]. However, neither qualitative nor quantitative simulation do everything the

[1] The existence of phase space attractors makes the idea of simulation with partial information much more plausible. Without attractors, incomplete specifications would lead to rapid divergence in the possible paths and hence little predictability. With them we can hope to predict, to a significant degree, the future.

other can. Thus we would like to combine qualitative and quantitative simulations, to do information processing tasks that would be difficult or impossible using either alone.

A rocket, fired upward, helps exemplify the shortcomings of either qualitative or quantitative simulation used alone, and the value of combining them into an integrated simulation method. Qualitatively, the rocket either falls back to Earth with a final height of 0, or escapes the Earth's gravitational field and goes on forever to a final height of $\infty$. Clearly no intermediate final height is possible for a completely vertical trajectory.

If the initial velocity at blast-off is precisely specified, a numerical simulation will be adequate, perhaps concluding unambiguously that the rocket falls back down. Unfortunately an ordinary numerical simulator can't infer a final height of $\infty$, because that would require simulating forever. Worse, if the initial velocity is close to escape velocity, error inherent in the numerical simulation algorithm could incorrectly lead to the conclusion that the rocket escapes when it doesn't, or *vice versa*.

Suppose instead that the initial velocity is partially known to within an interval. Then depending on the interval, either or both of the qualitative behaviors could occur. However most existing simulation techniques are deficient in simulating this simple system. An ordinary numerical simulator will be unable to represent the interval or conclude that qualitatively different behaviors can occur; an interval simulator will not automatically find and represent qualitively distinct behaviors, thus correctly — but crudely — predicting a final height in the interval $[0, \infty]$ if run forever or $[0, f(t)]$ if run for finite time; a Monte Carlo-based simulation analysis [Kalos and Whitlock 86] will not automatically classify the results of its many runs into qualitative categories and may not even find an example of an unlikely but possible behavior; and finally, qualitative simulation will not use the interval but rather return all behaviors even when the interval is consistent with only one. An alternative to these simulation techniques is intelligent matching of specialized numerical analysis methods to the particular problem at hand. These two methodologies, intelligent numerical analysis and simulation, are both active research areas [1].

We will describe a method that can successfully simulate, given initial or other boundary values described anywhere on a spectrum from numerical values, through interval values, to qualitative values, depending on the specifications available. This capability is important, because it is often the case that completely precise numerical information is unavailable, while it is still important to obtain reliable behavioral predictions from what information is available.

This ability to simulate given partial quantitative specifications is provided by utilizing strengths from three simulation methods:

1. *Qualitative simulation*, which can represent incomplete, qualitative specifications of a model and its initial conditions; can represent the limits of applicability of a particular model and switch to new models when limits are reached; and can provide a guarantee that all possible behaviors are predicted.

2. *Interval simulation*, which can express an important type of incomplete quantitative model specifications; and can guarantee that the correctness of quantitative inference is preserved.

3. *Numerical simulation*, which can express precise quantitative model specifications when available; can provide precise predictions when they are implied by precise specifications; and can be viewed as a limiting case of interval simulation — as an interval is narrowed it contains a smaller part of the number line, until when the endpoints are equal it contains a single number.

In qualitative-quantitative simulation, qualitative and quantitative simulations are combined to provide the strengths of both. The implementation described in this work is called Q3. For the rocket example (page 2), Q3's simulations include the following characteristics:

1. The rocket can be simulated even with no quantitative initial conditions, unlike with a numerical simulation.

2. The rocket can be simulated with interval-valued initial conditions, unlike with qualitative or numerical simulation.

3. The rocket can be simulated with precise initial conditions, unlike with qualitative simulation.

4. The rocket simulation can end automatically when the rocket hits the ground, unlike with many numerical simulations.

5. A final height of $\infty$ can be predicted for the escape behavior, unlike with interval or numerical simulations.

6. All behaviors are guaranteed to be found when they could occur, unlike with interval or numerical (or Monte Carlo) simulations.

7. Predicted intervals describing the trajectory of the model are guaranteed to contain the actual trajectories of systems conforming to the model, like interval simulation but unlike numerical (and Monte Carlo) simulation.

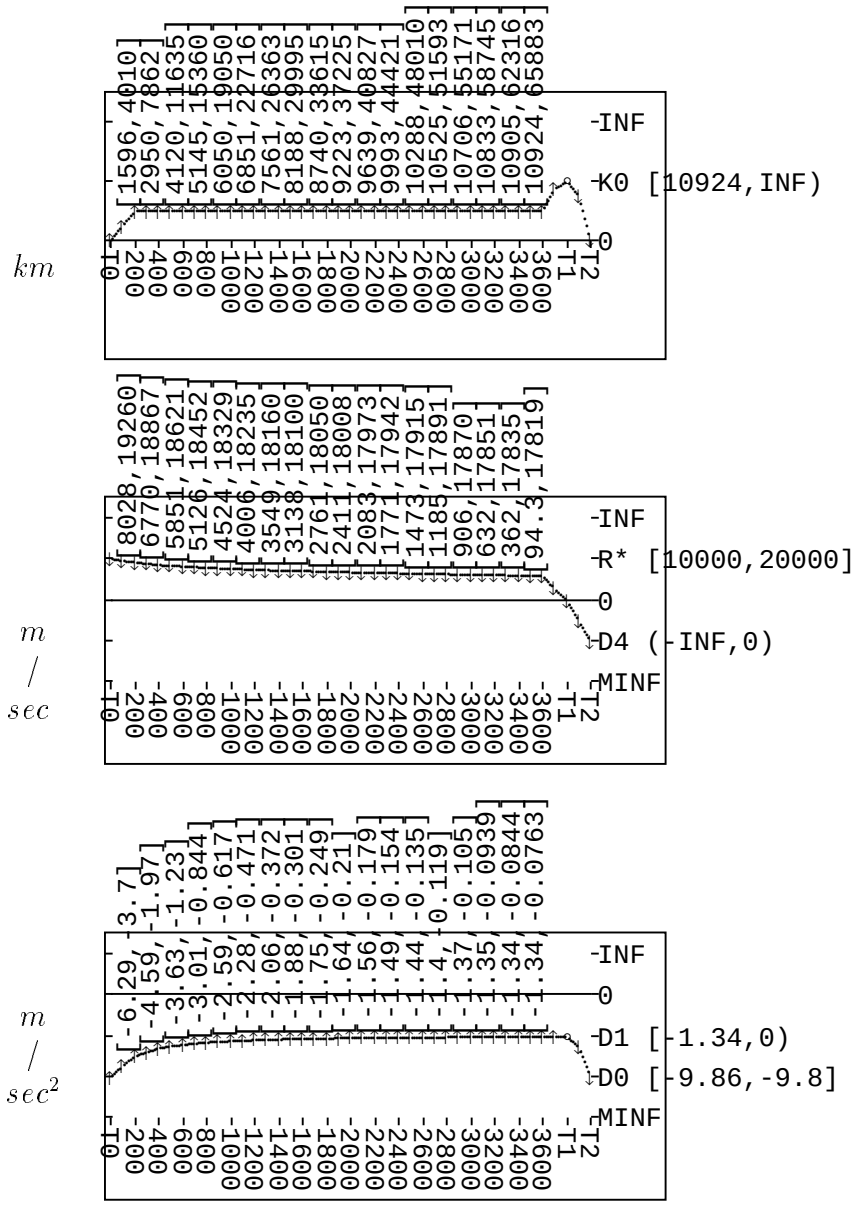
Figure 1.1 shows the results of running Q3 on one of the behaviors of the rocket example, the one in which the rocket falls back to earth. In the other two behaviors the rocket reaches a final distance of $\infty$, with velocity decreasing asymptotically to zero in one case, and to some positive value in the other.

Q3 uses what quantitative information is available to make predictions unavailable from other kinds of simulation, combining qualitative and quantitative simulation in a fine-grained way. Intervals are utilized as a representation for partial numeric information because intervals can represent a continuous spectrum of partial information from actual numbers, such as $[7, 7]$, through finite width ranges like $[7, 9]$, open-ended ranges like $[7, \infty)$ and finally to qualitative values like $[+]$, which is the interval $(0, \infty)$. Because intervals can represent a continuous spectrum of partial information they can be called a *full-spectrum* representation, meaning they can express at any point along a continuum of information specificity. Intervals are in the qualitative-quantitative spirit of working with however much or little information about value is available.

## 1.2 Specific Benefits of Qualitative-Quantitative Simulation

We now list in more detail the expected advantages of combining qualitative and quantitative simulation into qualitative-quantitative simulation.

1. *Reasoning with incomplete information.* We would like to be able to simulate, even if there is no quantitative information available or not enough available to specify a quantitative simulation. Qualitative simulation does not use quantitative information even when available. Qualitative-quantitative simulation can use numerical information when available but can still simulate without it, in which case the result is equivalent to a qualitative simulation. In a qualitative-quantitative simulation technique like that of the Q3 program, partial quantitative information in the form of intervals can be used. Intervals, which are intermediate in specificity between numbers and qualitative values, can be used to draw conclusions of intermediate power so that the strength of the conclusions reflects the strength of the problem specifications.




$T1 \in [3671, \infty]$, was $[1014, \infty]$
$T2 \in [3671, \infty]$, was $[1014, \infty]$

Figure 1.1: A rocket is fired upward with initial velocity $\texttt{R*} \in [10, 20] km/sec.$ Since escape velocity is 11 $km/sec.$, The rocket may possibly fall back to Earth (behavior 1, shown) although it will most likely escape.

2. *Reasoning about classes of systems.* Qualitative simulation models represent a *class* of numerical simulation models, because differing numerical models can correspond to the same more abstract qualitative model. This is in contrast to the complete problem specification typical of a numerical model. Qualitative-quantitative specifications therefore can represent either a particular problem or a class. When the qualitative-quantitative modeling language allows use of partial numerical information (of which intervals are one example) then there can be a great deal of flexibility in the specification of the class of problem, as represented by the quantitative model specifications.

3. *Finding all behaviors automatically.* A numerical simulation describes one path for the future of the system. In contrast, a qualitative simulation can find all possible qualitatively distinct futures. However, the weak specifications inherent in a qualitative model can cause a simulation to be unable to predict very far before encountering more than one and perhaps intractably many possible futures. Also, the predictions of a qualitative simulator are weak compared to numerical simulations. With qualitative-quantitative simulation, some behaviors could be ruled out and predictions strengthened beyond the capabilities of pure qualitative simulation. For example, quite rough information about the velocity of a baseball player's bat and the Earth's gravitational field strength can be used to infer that the ball will not escape into space.

4. *Performance considerations.* The relative computational complexities of qualitative and quantitative simulations are as yet unclear. Some factors to consider are these: 1) The primitive state of development of qualitative simulation methods means less attention has been paid to performance than in numerical simulation, which is a highly developed field in which considerable emphasis is given to performance considerations; 2) Qualitative simulations may predict more than one future, unlike numerical simulation, and performance must clearly be slower the more futures that are to be processed; yet 3) The simpler state descriptions of qualitative simulation may be in principle faster to manipulate than the more complete state descriptions of numerical simulations.

   In combining qualitative and numerical simulation, Forbus and Falkenhainer [1990:386] argue for performance comparable to pure numerical simulation in their SIMGEN system.

5. *Handling discontinuous model changes automatically.* Quantitative simulations often ignore the limits of the model, and can end up providing meaningless (and wrong) predictions, such as extrapolating decreasing

level in a tank to predict negative levels [Forbus and Falkenhainer 90]. Even when model limits are handled by a simulation language, these limits are specified as precise numbers. Qualitative simulation can detect the limits of a model symbolically, describing such limits with qualitative values like "0", "$\infty$" and "*top*". In qualitative-quantitative modeling, limits can be described more precisely, such as by assigning intervals or numbers to qualitative values such as "*top*".

6. *Supporting reasoning about causality.* Qualitative models and simulations have been used to help generate descriptions of the causality behind a system's operation (e.g. Iwasaki and Simon [1986], Iwasaki [1988]). Qualitative-quantitative simulation could be useful here, because the future of a system — and hence the cause and effect relationships among its states — can depend critically on even partially constrained quantitative values.

7. *Supporting user friendly explanations about the system and simulation.* Qualitative simulation has been shown to facilitate explanation [Forbus and Falkenhainer 90]. This can be useful to engineers, operators and for teaching purposes. Qualitative-quantitative simulation could additionally utilize the fact that arbitrary reasonable numbers are often used in engineering practice to help illustrate qualitative concepts of system operation.

These ideas can be used to help motivate potential applications based on utilizing the advantages of combining qualitative and quantitative simulations.

## 1.3 Previous work

Considerable work in the area of qualitative-quantitative reasoning has been reported, such as in VLSI, the spatial domain, temporal ordering, general purpose interval reasoning, and of course simulation work in the artifical intelligence community. The following survey of previous work cannot be comprehensive but rather intends to indicate the range of different approaches that have been taken, often in apparent disregard of each other.

### 1.3.1 Qualitative-quantitative work in VLSI

At the qualitative end of the spectrum, when the transistor is viewed as a primitive unit its logic state can be represented qualitatively as being in

$\{0, 1, X\}$, where $X$ means "unknown" [Bryant 81]. Given input vectors, the stable state of the circuit is derived by relaxation. At the numerical end of the spectrum are simulators such as SPICE [Tuinenga 88] which are numerical.

There is a variety of systems in between that do qualitative-quantitative reasoning [Hon 87]. For example, timing simulators [Terman 83; Chawla et al. 75] combine numerical circuit information with qualitative information about logic state, and *multilevel* simulators simulate at different levels of abstraction within a single integrated circuit design [Insinga 84].

Qualitative-quantitative work in the VLSI domain is specialized and lacks generality, because simulators take advantage of special properties of the domain. For example, component states may be modeled with a quantity space of discrete values with simulation done discretely [VLSI 86]. These are called multiple-state simulators. Some VLSI simulation systems can simulate large systems and are quite valuable commercially.

### 1.3.2 Qualitative-quantitative spatial reasoning

Spatial domain work in artificial intelligence has also utilized qualitative-quantitative reasoning. For example, Brooks' ACRONYM system combined qualitative information about the geometric properties of objects with approximate quantitative information to do qualitative-quantitative reasoning for model based processing in the computer vision area [Brooks 81; Cohen and Feigenbaum 82].

In another area of the spatial domain, McDermott and Davis' [1984] navigation program SPAM did qualitative-quantitative reasoning, combining qualitative features with partial numeric information in the form of intervals to produce qualitative-quantitative maps of the terrain. Inference on the partial numeric information was done using such traditional methods of numerical optimization as linear programming, as well as a Monte Carlo based method that provided faster (but unguaranteed) results.

### 1.3.3 Qualitative-Quantitative work in temporal ordering

Temporal ordering has also been an active domain for qualitative-quantitative work in artificial intelligence. For example, Dean and McDermott [1987] describe methods for consistency maintenance of a temporal database. Dean and Boddy [1988] abstracted collections of events, each known within a temporal interval, into non-numeric partial orders. Inference was done within this new frame of reference, and conclusions could (if desired) be mapped back

to numerical intervals via the original mapping from intervals to the abstracted partial order.

### 1.3.4 Interval and inequality reasoning

General purpose work with partial quantitative information as it relates to our present concerns includes mainly interval and inequality reasoning. Considerable work has been done in interval arithmetic and its applications to simulation including Moore [1979], Eijgenraam [1981], Alefeld and Herzberger [1983] and many papers, collections and reports.

One of the sub-problems in applying interval mathematics techniques to applications such as qualitative-quantitative simulation is finding the variation in the value of a function given known variation in one or more of its parameters [Kolev et al. 88; Asaithambi et al. 82]. This can be difficult because the RHS of a function is an expression, and *excess width* is a well-known problem in evaluating interval expressions [Moore 79; Abelson and Sussman, 1985:84-88]. As a trivial example, we know that $X - X = 0$, but the straightforward interval-based evaluation method gives a weaker answer for non-numerical $X$'s. For example, $[1,2] - [1,2] = [-1,1]$. The excess width problem occurs whenever dependencies exist among inputs or intermediate values when evaluating an interval expression.

In artificial intelligence, interval inference has been reviewed by Davis [1987], whose discussion of interval inference by constraint propagation describes the Waltz filtering algorithm used by Q3. Williams [1988] describes a hybrid of qualitative and quantitative algebras called *Q1* (no relation to Q3). Simmons' [1986] quantity lattices also do both interval and inequality inference. These results allow more powerful qualitative-quantitative constraint reasoning in some respects than Q3 currently does because they support simple symbolic algebraic reasoning. On the other hand, since Q3 is a simulator, it supports integration and differentiation unlike *Q1* and quantity lattices. Powerful qualitative-quantitative constraint reasoning modules might profitably be employed in Q3 and in future qualitative-quantitative simulators.

Inequality reasoning, such as by linear and nonlinear programming, is a large area in its own right [Luenberger 73]. In artificial intelligence it has been applied in a number of areas including some previously mentioned. Domain independent work in artificial intelligence includes Sacks' [1987] work on hierarchical inequality reasoning.

There are also other representations for partial numeric information such as probability (e.g. Pearl [1988], Wellman [1990] and chapter 7) and Zadeh's fuzzy mathematics [Dubois and Prade 80, Shen and Leitch 90].

### 1.3.5 Qualitative-quantitative simulation work

Simulation of models to predict behavior of real systems has a long history and much practical use. *Discrete event* simulation, used e.g. in computer engineering, simulates models whose state variables change discretely. Simulation of *continuous* systems also has a history of practical use. Such simulation has traditionally come under the heading of *numerical methods*. *Combined* simulators handle both discrete and continuous changes in state. In the mainstream computer simulation field, simulators such as SLAM [Pritsker 84] are combined simulators. Q3 is a combined simulator, as illustrated by the simulation of an air conditioned dwelling in figure 1.2. Artificial Intelligence has relatively recently entered the simulation field with qualitative simulation [de Kleer 77, 84; Forbus 84; Kuipers 84, 86; and collections Bobrow 85; Weld and de Kleer 90].

Work on combining qualitative and quantitative simulation into *qualitative-quantitative* simulation may be broadly classified in two categories:

1. Augmenting qualitative simulation with quantitative inference.
2. Augmenting numerical simulation with AI techniques.

Work on augmenting qualitative simulation with quantitative inference (item (1) above) is reported by de Kleer [1975, 1977], Simmons [1983], Williams [1986], Forbus [1986], Sacks [1987], Skorstad and Forbus [1989], Forbus and Falkenhainer [1990], De Coste [1990], Berleant [1991] and elsewhere.

Simmons' *imagining* technique [1983], later used in the GORDIUS program [Simmons and Davis 87; Simmons 88] combined qualitative and quantitative simulation in a *discrete* simulator using most notably the geological domain, showing that using both types of simulation together can produce results neither alone can produce in a discrete simulation. In that work and in most qualitative-quantitative simulators in the (piece-wise) continuous area, qualitative and quantitative knowledge have been maintained in distinct representations, integrated by mediating between the two representations. Among the exceptions, Williams [1986] integrated qualitative and numeric information about *time* in his qualitative-quantitative simulator.

Domain independent work on augmenting numerical simulation with AI techniques (item (2) above) has emphasized phase-space analysis [Yip 88, Sacks 90a, Nishida and Doshita 91, Zhao 91]. An overview of some of that work appears in Abelson [1989]. By starting with numerical simulation and adding artificial intelligence techniques, existing proven methods of numerical reasoning can be embedded in larger systems that have capabilities beyond

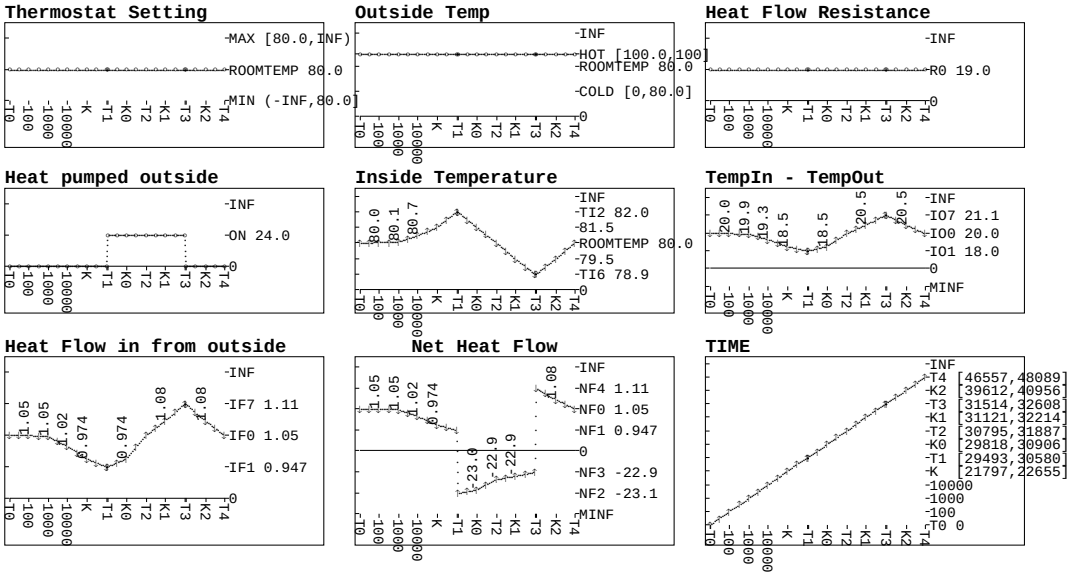



Figure 1.2: Combined simulation of an air conditioned dwelling. The time points in this simulation come from three sources. The qualitative simulation generated time points `T0, T1, T2, T3` and `T4`. New states representing the times `100, 1000`, and `10000` were interpolated into the gap between `T0` and `T1`. New states were then created by interpolating landmarks representing 79.5° and 81.5° fahrenheit into the quantity space of `Inside Temperature`, resulting in the time points `K, K0, K1`, and `K2`.

stand-alone numerical simulators. Nishida and Doshita report results for two-dimensional systems of ODEs. Sacks claims his PLR program works at the expert level with any combination of qualitative and numeric information about a 2nd-order system. Unlike Q3, PLR is limited to separable systems. For example, PLR does not handle the nonseparable system $x''(x')^2 = 0$ [Sacks 90a:320] whereas Q3 can simulate it. Also, because it relies on approximations, PLR's results are not guaranteed correct.

Techniques that add AI to numerical simulation will tend to retain some shortcomings of numerical simulation, such as the possibility of conclusions which are uncertain due to the approximating character of many numerical methods. Conversely, adding numerical inference to qualitative simulation tends to retain a different set of shortcomings, those found in qualitative simulation. Thus these two major categories of research in qualitative-quantitative simulation are complementary.

Q3 utilizes partial numerical information in the form of intervals, and provides *correctness* guarantees, both qualitative [Kuipers 86, 88] and quantitative. The quantitative correctness of Q3 derives from the correctness of Waltz filtering, which is the underlying basis for quantitative inference in Q3. The potential shortcomings of qualitative simulation, such as a profusion of behaviors, are alleviated substantially by the numerical component of Q3.

**Monte Carlo simulation** Inferring intervals for the possible ranges of model variables can be done with Monte Carlo simulation. However, Monte Carlo simulation [Kalos and Whitlock 86] lacks some desirable characteristics:

1. Monte Carlo simulation relies on random sampling of the input space, so portions of the input space could be missed by chance, leading to predictions that do not include possible values.

2. Missing portions of the input space (item 1) could lead to leaving out possible qualitative behaviors from predictions. This problem would be especially acute in the case of behaviors which are unlikely but important — perhaps critically important, such as the failure behaviors of a nuclear power plant or other usually reliable yet critical system.

3. Because Monte Carlo simulation relies on numerical simulations, predictions about behavior at infinite time are not made. Thus for example, prediction that a rocket escapes to infinity would not be possible without the use of supplementary analysis.

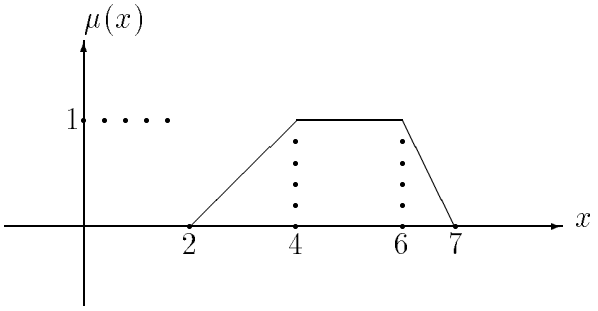



Figure 1.3: A fuzzy interval. The membership $\mu$ of numbers on the $x$-axis is less certain, the lower the value of $\mu(x)$.

4. Monte Carlo simulation does not categorize the results of simulation runs into qualitative behaviors, whereas this is a natural result of using a qualitative simulation-based method such as Q3.

**The fuzzy mathematics approach** Much of the qualitative-quantitative simulation work has utilized intervals as a representation for partial numeric knowledge. Intervals are not the only representation for partial numeric information. Another recent approach is the use of *fuzzy* values and intervals [Zadeh 65; Dubois and Prade 80] for simulation. While fuzzy reasoning has been criticized [Cheeseman 86], it has also been used in qualitative-quantitative simulation by Shen and Leitch [1990]. They generalize intervals, which have sharp boundaries, to have fuzzy boundaries (figure 1.3).

Usually numbers in $[4, 6]$ of the figure would be considered full members of the fuzzy interval shown, and numbers in the sloping areas considered possible members. However this semantics does not allow for the excess width problem resulting from certain arithmetic operations on non-independent arguments. For example evaluating $X - X$ in the straightforward manner when $X$ is a fuzzy interval yields a fuzzy interval wider than 0 but containing it. An alternative semantics that says that numbers in $[4, 6]$ of figure 1.3 have a *maximum possible* membership of 1, numbers in the sloping area have a maximum possible membership less than 1, and numbers outside the fuzzy interval have no membership is correct, in that it corresponds to reality.

Shen and Leitch do not claim to guarantee generation of all possible behaviors in their simulations. Their models may also require trial-and-error tuning of transition tables. Their approach is one possible way to do qualitative-quantitative simulation using fuzzy values, and this may be an area

with potential for further work.

Simulation using probability distributions to represent partial numeric information about values has also been approached (chapters 7 and 8).

---

This concludes the survey of previous work. Brief overviews of particular relevance to individual topics appear later in the relevant chapters.

## 1.4 Outline of Remaining Chapters

In the next chapter we describe how partial quantitative information, in the form of intervals, may be added to the framework of a qualitative description of behavior. The implementation is called Q2. Chapter 3 describes inference algorithms for interval information used to supplement the ordinary Waltz filtering used in Q2. Chapter 4 explains in detail how step size refinement — interpolating states with quantitative but not qualitative significance into an existing simulation — facilitates further improvements in inference. Chapter 5 further develops step size refinement into an algorithm with the desirable properties of quantitative convergence and stability. This was implemented in a simulator, Q3, with characteristics of both qualitative and quantitative simulation, thus bridging a conceptual gap between qualitative and quantitative simulation. Following this, Q3 is applied to diagnosis and measurement interpretation in chapter 6. Then Q3 is applied to the utilization of another form of partial quantitative information, probability distributions. Chapter 7 describes one algorithm for doing this, and chapter 8 describes another algorithm in the context of a dependability analysis application.

---

Each chapter is intended to be largely self-contained and independently readable.

# Chapter 2

# Q2: Adding Interval Information to Qualitative Simulation

## 2.1 Overview

Incomplete knowledge of the structure of mechanisms is an important fact of life in reasoning, commonsense or expert, about the physical world. Qualitative simulation captures an important kind of incomplete, ordinal knowledge, and predicts the set of qualitatively possible behaviors of a mechanism given a qualitative description of its structure and initial state. A qualitative model of a device or system is an abstraction of a set of real systems. The behavior of these systems can vary greatly, yet purely qualitative descriptions of these behaviors are identical.

However, one frequently has *quantitative* knowledge as well as qualitative, though not enough to specify a numerical simulation. This chapter presents a method for exploiting incomplete quantitative knowledge, by using it to refine the predictions of a qualitative reasoner. Incomplete quantitative descriptions, in the form of intervals within which unknown values are assumed to lie, are asserted about some landmark values in the quantity spaces of qualitative model variables. Unknown monotonic functions may be bounded by numerically computable envelope functions. Quantitative inferences are derived by local propagation across the constraints in each behavior.

Adding quantitative information to qualitative models allows more precise characterization of systems and their behaviors. This increased precision can help in diagnosis and in prediction of behavior, by pruning away some behaviors and making quantitative predictions about others, even when there is much too little quantitative information for numerical simulation.

## 2.2 Propagation of Incomplete Quantitative Information

We will explain our quantitative propagation method in the context of a simple one-tank "bathtub" system, in this case one with a partially blocked drain, so that outflow increases only slowly with pressure. The model is shown in figure 2.1, and the QSIM code for the model in figure 2.2.

There are three qualitative behaviors for a bathtub which is being filled from empty with the drain incompletely closed:

1. Equilibrium between inflow and outflow before amount reaches FULL;
2. Overflow while inflow is greater than outflow; and
3. Equilibrium between inflow and outflow exactly when amount reaches FULL. This one has infinitesimal probability and is ignored henceforth.

The important behaviors are shown in figure 2.3.

In the implementation Q2, two types of quantitative information are provided as part of the initial description of the system:

- Quantitative intervals describing what is known about the values of certain landmark values, in this case the landmark IF* of the model variable $inflow(t)$, and the landmark TOP of model variable $level(t)$.
- Numerically computable envelopes that bound the unknown and possibly nonlinear monotonic functions, such as
$$outflow = M^{+}(pressure).$$

Figure 2.4 shows the quantitative initialization used by Q2.

Figure 2.5 shows the only quantitatively consistent behavior out of the three qualitative possibilities, given initial quantitative assertions about TOP, IF*, and envelopes constraining the relations between amount and level, between level and pressure, and between pressure and outflow. The two equilibrium behaviors were found to be inconsistent with the quantitative information given.

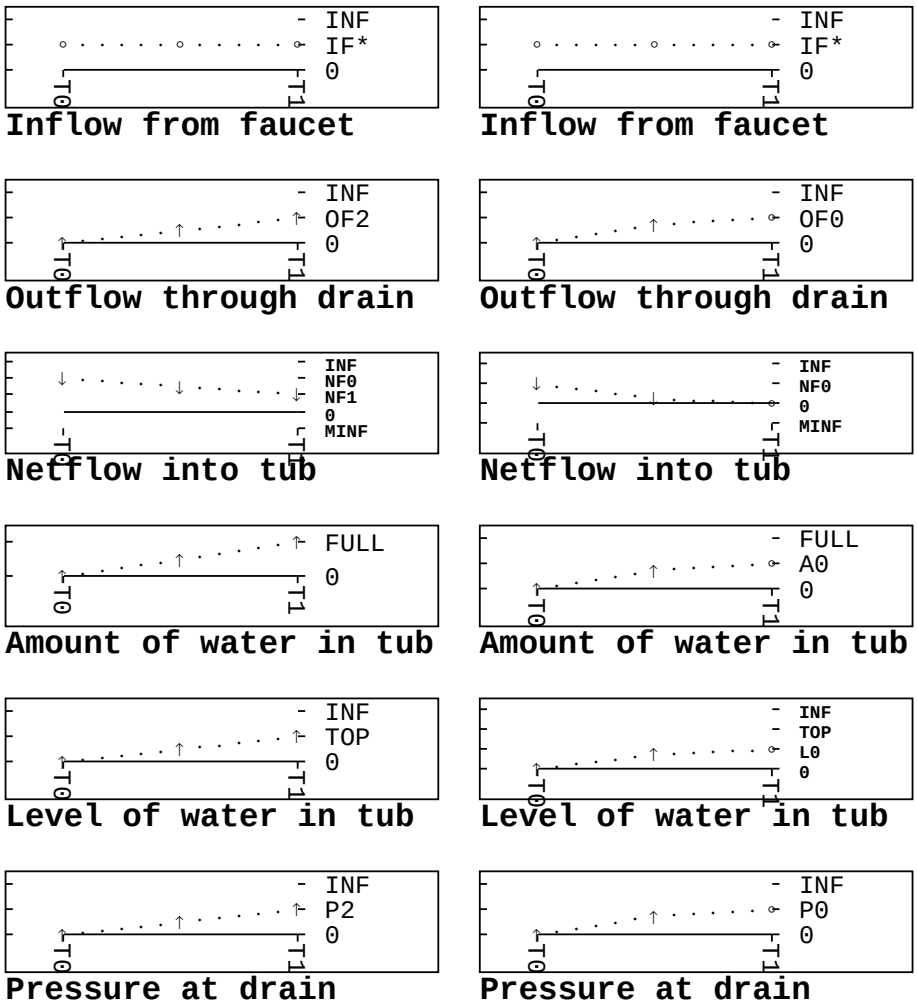
Don't forget to underline inflow & drain

Figure 2.1: Qualitative model schematic for a 1-tank "Bathtub" system. The model is valid when there is a definite level in the bathtub. (At very low rates of inflow, water forms a rivulet running across the surface of the tub and level is undefined, although amount is positive. Alternatively, where the faucet is directly over the drain so that inflow never touches the tub, there is no level as such even though a positive amount of water is within the confines of the tub before passing out the drain.)

```
; ============================================================
; A model of a ``bathtub'' 1-tank system.
; ============================================================

(define-QDE bathtub
  (text "Standard bathtub with open drain.")
  ; Define model variables and their qualitative values.
  (quantity-spaces
    (amount    (0    FULL     ))  ; Quantity of fluid in tank.
    (level     (0    TOP  inf ))  ; Height   of fluid in tank.
    (pressure  (0    inf      ))  ; Pressure at tank drain.
    (outflow   (0    inf      ))  ; Rate fluid exits drain.
    (inflow    (0    if*  inf ))  ; Rate of faucet flow.
    (netflow   (minf 0    inf ))) ; Net rate of fluid in.
  (independent inflow) ; Held constant during simulation.
  (constraints
    ((M+   amount   level         )   (0 0) (full top))
    ((M+   level    pressure      )   (0 0) (inf  inf))
    ((M+   pressure outflow       )   (0 0) (inf  inf))
    ((ADD  netflow  outflow inflow)                   )
    ((d/dt amount   netflow       )                   ))
  (print-names (amount   "Amount of water in tub"  A )
       (level    "Level of water in tub"   L )
       (pressure "Pressure at drain"       P )
       (outflow  "Outflow through drain"   OF)
       (inflow   "Inflow from faucet"      IF)
       (netflow  "Netflow into tub"        NF))
  (layout ( inflow outflow)
          ( netflow amount)
  ( level pressure)
  ( time )))
```

Figure 2.2: The QSIM code describing the bathtub model. The system was initialized to empty (`amount`=0) with `inflow=if*`.




(a) overflow

(b) equilibrium

Figure 2.3: A qualitative simulation of the bathtub system, showing overflow and equilibrium behaviors.

```
; ============================================================
; Partial quantitative specifications for the bathtub model.
; ============================================================

(def-quantitative-info
  (name true-bathtub-inflow=[1_1.01]) ; Descriptive name.
  (quantitative-initializations
    (level  (top (9/10  91/100)))  ; height of tank.
    (inflow (if* (1    101/100)))) ; inflow rate from faucet.
  (envelopes
    (lo-envelope (M+ amount        ; first  arg returns amount.
                     level)        ; second arg returns level.
                 (M+ (times level  100/98)   ; calculates amount.
                     (times amount 98/100))) ; calculates level.
    (hi-envelope (M+ amount
                     level)
                 (M+ (times level  98/100)
                     (times amount 100/98)))
    (lo-envelope (M+ level
                     pressure)
                 (M+ (times pressure 100/98)
                     (times level    98/100)))
    (hi-envelope (M+ level
                     pressure)
                 (M+ (times pressure 98/100)
                     (times level    100/98)))
    (lo-envelope (M+ pressure
                     outflow)
                 (M+ (if (not (neg outflow))
                         (times (square-root outflow) 100/98))
                     (times pressure pressure 98/100)))
    (hi-envelope (M+ pressure
                     outflow)
                 (M+ (if (not (neg outflow))
                         (times (square-root outflow) 98/100))
                     (times pressure pressure 100/98)))))
```

Figure 2.4: Quantitative information provided Q2 for the bathtub model. Envelopes partially specify monotonic constraints (figure 2.7).

Structure: Standard bathtub with open drain..
Initialization: Filling at constant rate, starting from empty (S0)
Behavior 1 of 1: (S0 S1 S2).
Final state: (TRANSITION FINAL GF COMPLETE), (TRANSITION-IDENTITY), T<INF.
Time point bounds: T1=[0.873,17.7]

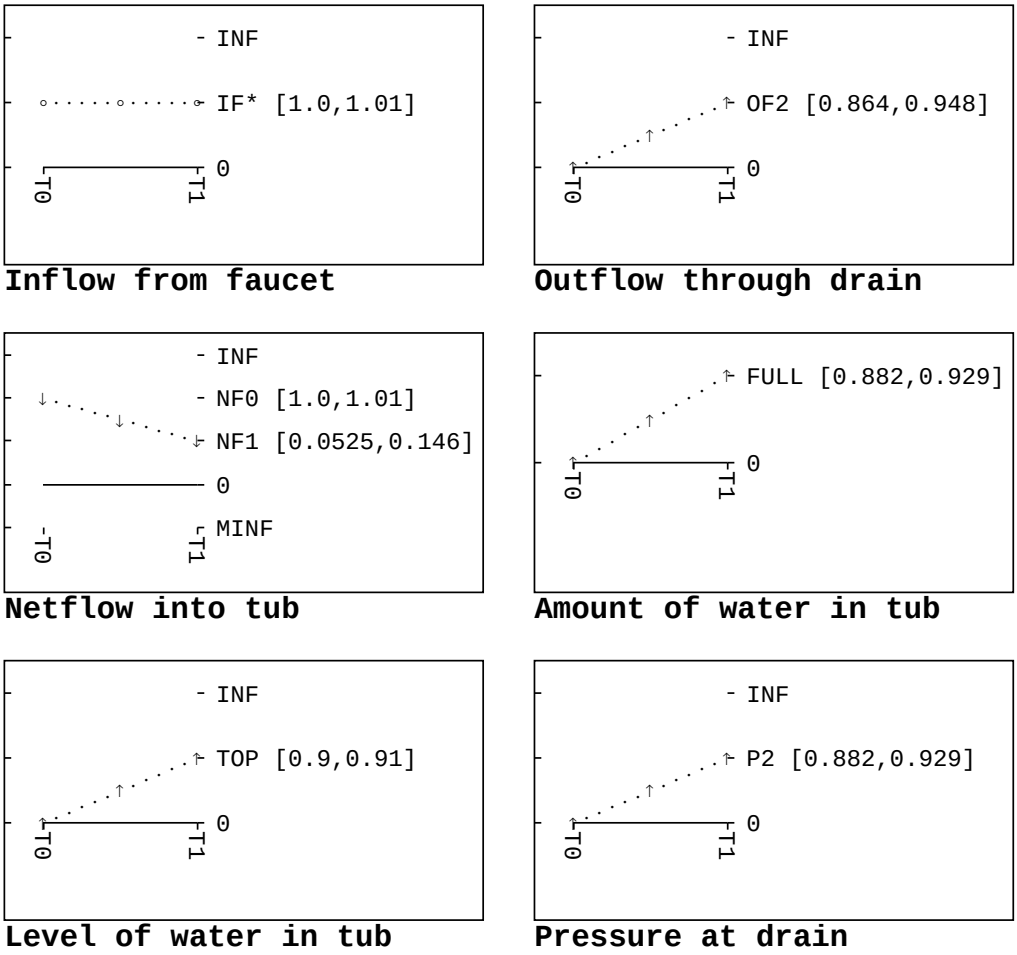



Figure 2.5: Q2 applied to one member of the 1-tank class of models, the venerable bathtub. Of three behaviors, two equilibrium behaviors were ruled out by Q2, leaving only the overflow behavior shown.

### 2.2.1 Types of quantitative propagation constraints

Quantitative propagation occurs in different ways for the various qualitative constraint types. As a notational convention, if the qualitative behavior has $modelVariable(t) = L$ for a landmark $L$ at a particular time point $t$, we may use either $modelVariable(t) \in [lo, hi]$ or $L \in [lo, hi]$, to indicate that the quantitative interval $[lo, hi]$ must contain the (unknown) numerical value of $L$.

In Q2, each type of qualitative constraint has a routine for propagating partial quantitative information among its arguments. These routines define a quantitative semantics for the constraint that must of course be consistent with the semantics already defined by the qualitative simulator. The four main types of qualitative constraints and their propagation methods for quantitative information are:

1. **Propagation across arithmetic constraints: ADD, MULT, MINUS.**

   This is exemplified by an ADD constraint in a model of a bathtub, as shown in table 2.1. See also figure 2.6. Note that division and subtraction constraints are trivially implemented with ADD and MULT. In the case of division by an interval containing 0, no inference would be made because the result is either undefined or $(-\infty, \infty)$ depending on how interval division is defined.

2. **Propagation across monotonic functions: $\mathbf{M^+}, \mathbf{M^-}$.**

   This is typified in the bathtub model by an "M+" monotonic function between *amount* of water and *level* in the tub, indicating that a change in either model variable implies a change in the other in the same direction. A qualitative monotonic function is a generalization of a large space of possible quantitative functions — indeed, all monotonic quantitative functions for which the monotonicity has the same sign as that of the corresponding qualitative function. There is a middle ground between purely qualitative and fully specified quantitative monotonic functions. We implement this middle ground by using upper and lower ENVELOPES (figure 2.7).

   ENVELOPES are quantitative functions which bound the space of quantitative functions that could apply to a monotonic function to a greater extent than the sign of the monotonicity. For the bathtub example, a particular tub may be consistent with a bathtub model that is partly quantified by envelopes constraining the relation between *amount* and *level* if its function relating amount and level falls within those envelopes.

Waltz Filtering an Addition Constraint

- Uses intervals as labels
- Correct — won't rule out possible values
- Example:

- $Z := (X + Y) \cap Z = [1, \infty] + [3, 4] \cap [-\infty, 5] = [4, 5]$.
- $X := (Z - Y) \cap X = [4, 5] - [3, 4] \cap [1, \infty] = [1, 2]$.
- $Y := (Z - X) \cap Y = [4, 5] - [1, 2] \cap [3, 4] = [3, 4]$.

Figure 2.6: Waltz filtering using interval labels is done on an addition constraint. Waltz filtering is correct, in that possible values won't be filtered out [Waltz 75; Davis 87].

———————

Pasteup here: Range propagation across an M+ function.

Figure 2.7: Propagation through an $M^+$ (monotonically positive) function is shown graphically. Shaded regions of the axes represent numerical intervals, and the upper and lower envelopes represent numerical envelope functions. Propagation shown is from Y to X; propagation from X to Y is also attempted by the constraint processing code.

———————————

(a) An ADD constraint:
$$netflow = inflow - outflow$$
(b) Landmark values at time $t_1$:
$$netflow(t_1) = inflow(t_1) - outflow(t_1)$$
i.e., $NF1 = IF^* - OF2$
(c) in terms of known ranges:
$$[0.0525, 0.146] = [1, 1.01] - [0, 9999]$$
(d) The ADD narrows the range for $OF2 = outflow(t_1)$:
$$\underline{[0.0525, 0.146] = [1, 1.01] - [0.864, 0.948]}$$

Propagation narrows the bounds for $OF2 = outflow(t_1)$.

Table 2.1: Use of an ADD constraint in the bathtub simulation of figure 2.5.

---

Otherwise it is definitely not consistent (perhaps it is very oddly shaped tub, or perhaps not a tub at all but a sink or swimming pool). Propagation through a partially quantified M+ function occurs as described in figure 2.7.

3. **Propagation within a quantity space.**

   Consider the $netflow$ of water into the tub. At time T1, the value of $netflow$ is whatever quantitative value corresponds to the landmark named "NF1" (figure 2.5). This unknown value must be less than the value of NF0, which may be as high as 1.01, but is greater than 0. Thus from the ordinal position of NF1 and the quantitative information associated with its neighbors, we infer that $netflow$(T1) = NF1 $\in [0, 1.01]$. (Using constraints of other types present in the behavior representation, Waltz filtering eventually narrows NF1 all the way to $[0.0525, 0.146]$.)

4. **Propagation across time points: D/DT.**

   Finally, there is information flow between states in the behavior of a model. This occurs via D/DT constraint templates, such as D/DT(amount)=netflow. By looking at quantitative information about the values at adjacent time points of the integral, derivative and time model variables, propagation can potentially constrain the intervals associated with each.

The mean value theorem states that

$$\exists t^* : rate(t^*) = \frac{level(t_{n+1}) - level(t_n)}{t_{n+1} - t_n} \tag{2.1}$$

where $t_n \leq t^* \leq t_{n+1}$, and *rate* and *level* are respectively the derivative and integral quantities in a derivative constraint template

```
(d/dt level rate)
```

where "`level`" and "`rate`" are after the terminology of Forrester [1961]. When quantities are interval valued,[1] as in *Q3*, we have

$$\begin{aligned} RATE(t^*) &\in RATE(T_n) \sqcup RATE(T_{n+1}) \\ &\supseteq \frac{LEVEL(T_{n+1}) - LEVEL(T_n)}{T_{n+1} - T_n} \end{aligned} \tag{2.2}$$

where $A \sqcup B$ is defined as

$$\begin{aligned} & [min(loBound(A), loBound(B)), max(hiBound(A), hiBound(B))] \\ \equiv\ & [min(\underline{A}, \underline{B}), max(\overline{A}, \overline{B})] \end{aligned}$$

i.e., everything in or between either interval. Equation 2.2 constrains the quantities on the RHS and can be used to try to narrow the interval for each of those quantities by rearranging the equation to solve for it, then evaluating the resultant interval expression (chapter 2). This is the *mean value constraint*. Note that the bounding expression $RATE(T_n) \sqcup RATE(T_{n+1})$ is *correct*, in that it contains the rate at any time between $T_n$ and $T_{n+1}$. However, intervals can be supersets of the actual range of values they describe, due to the excess width property of interval calculations (section 1.3).

For the bathtub, then,

$$netflow(t^*) = \frac{amount_{T1} - amount_{T0}}{T1 - T0} \tag{2.3}$$

[1] Let $f(x_1, ..., x_n)$ be a rational function of real variables. Then we can straightforwardly generalize $f$ to a function $F$ with interval arguments. The real variables $x_i$ are replaced by interval variables $X_i$ and the arithmetic operations in $f$ are changed to their corresponding interval arithmetic operations. Then $F(X_1, ..., X_n)$ is the **natural interval extension** of $f(x_1, ..., x_n)$ [Moore 1979:21].

From figure 2.5 we can see that $netflow(t^*) = [0.0525, 1.01]$. We also see that $amount(t)$ started out at 0 and climbed to some value between 0.882 and 0.929. T0 is known to have the value 0. Thus,

$$T1 = 0 + \frac{[0.882, 0.929]}{[0.0525, 1.01]} = [.873, 17.7]$$

...as indicated in the banner of figure 2.5.

These four constraint types propagate the quantitative information provided, to attempt narrowing of the intervals associated with landmarks for each model variable at each time point, until either no further narrowing is possible or an inconsistency is flagged. Inconsistency, of course, is relative to a behavior and means that the behavior is not compatible with the available quantitative information. If all behaviors of a model are inconsistent then an additional inference is possible: the model itself is incompatible with the quantitative information, whether that information is known *a priori* or from observations.

### 2.2.2 The propagation algorithm

There are a number of potential approaches to inference given partial quantitative information. These include sophisticated interval analysis techniques, linear and non-linear programming, and inequality reasoning (section 1.3.4). For this work, the well known AI technique of *Waltz filtering* [Davis 87; Waltz 75] has been applied, supplemented with other methods to be described in later chapters.

The interval propagator starts by setting each landmark of each model variable to an initial interval of $(0, \infty)$, $(-\infty, 0)$ or $[0, 0]$, depending on whether the landmark is above zero, below zero, or is the "0" landmark. Then any quantitative information provided by the user is used to narrow the appropriate landmarks. For the bathtub, (inflow . IF*), the IF* landmark of the inflow of water from the faucet, is initialized to the rate (in e.g. gallons/min.) of $[1.0, 1.01]$. In addition, (level . TOP) is initialized to $[0.9, 0.91]$, meaning that we are dealing with bathtubs whose height lies between 0.9 and 0.91.

Narrowed landmarks can potentially enable narrowing of other landmarks via constraints. A constraint is *attached* to a landmark $L$ if its interval can be used to try to narrow other landmarks. This is the case if $L$ is an argument of the constraint, which may be inferred from the behavior representation. The behavior representation is a network of constraints instantiated for

specific landmarks during qualitative simulation from the *constraint templates* (also called model constraints or qsim constraints) defined by the model. All constraints attached to the narrowed landmarks are added to a queue. The propagation algorithm now takes a constraint off the queue and uses it to try to narrow the landmarks associated with it. If it fails it goes back to the queue for another constraint. If it succeeds it adds to the queue all constraints attached to any landmarks it succeeded in narrowing, and returns to the queue for a new constraint to service.

There are a variety of methods for deciding which constraint to service:

1. *Remove and process the most recently arrived constraint.* This method treats the queue as a stack. It is a depth first approach in that constraints attached to the most recently processed constraint are processed before those attached to less recently processed constraints. This method had been found to often lead to loops, in which the same sequence of constraints are serviced and reserviced many times, before settling with respect to each other and thus allowing other constraints to be serviced. Hence this technique is susceptible to starvation of some constraints.

2. *Process all the constraints in the network describing the behavior in an arbitrary predetermined order.* Repeat until no constraint produces a narrowing of any of its parameters. This method avoids starvation because all constraints will eventually be serviced. It is also quite simple to implement because no code is required for finding constraints attached to the current one. The queue is simply formed of all constraints, gradually depleted, and when empty simply re-initialized the same way as before. However it is slow because it may contain many constraints that do not narrow their parameters, yet are tested anyway.

3. *Remove and process the least recently arriving constraint.* This is the first-in-first-out queue. Starvation will not occur, as every constraint on the queue is guaranteed to eventually be served. It is faster than the method of item 2 because the only constraints on the queue are ones likely to narrow their arguments, since they are on the queue in the first place because one of their arguments has been narrowed by another constraint.

4. *Process the queue in some way to decide which constraint to process next.* The most obvious way to do this is to remove any copies elsewhere on the queue of the constraint just removed. This speeds things up by not bothering to process the same constraint more than once (unless it arrives back at the queue later).

Another approach is to sort the constraints into categories, processing those in some categories in preference to those in others (priority queueing). I have found that arithmetic and monotonic constraints tend to be more powerful than $D/DT$ and quantity space constraints, and hence should be processed in preference to them as long as there are any available on the queue. Quantity space constraints are weak because they are inequalities and can only constrain one of the two bounds of an argument. For example, if $X_1 \in [2,3]$ and $X_1 < X_2$ then we can infer information only about the low bound of $X_2$, namely $X_2 \in [2,\infty)$. $D/DT$ constraints are weak because they utilize several arithmetic operations (see equation 2.3) and each such operation tends to increase the width of a calculated interval and hence reduce the specificity of the result. A priority queuing discipline in which weak constraints are invoked after stronger ones is a heuristic way of improving average performance, while sacrificing an absolute guarantee of avoiding starvation. The danger of starvation can be minimized by processing MULT constraints before ADDs, as loops characterized by MULTs settle exponentially while those characterized by ADDs settle only linearly.

It is not difficult to imagine arbitrarily complex schemes for calculating preference orders among the constraints. The guiding principle, applicable to any Waltz filtering problem, is to choose more powerful constraints before less powerful ones.

## 2.3 Previous Work

The measurement interpretation methods developed by Forbus [1983, 1986] are related to this work in that we, like Forbus, are attempting to interpret quantitative observations by matching them against the predictions of a model. However, the present approach is quite different.

The present method differs from Forbus' approach in the handling of quantitative information. In the more complete formulation [Forbus 86], a continuous stream of quantitative data is mapped into a stream of qualitative descriptions; in his example directions of change, or Ds values $\{+1, 0, -1\}$. In an example involving heating a container of mixed alcohol and water, the stream of temperature measurements is described qualitatively as $[+1, 0, +1, 0, +1]$. Notice that a significant amount of quantitative information is lost when fine-grained quantitative measurements are mapped to coarser-grained qualitative representations (in this case the Ds values).

In Q2, by contrast, quantitative information is used to *augment* the qualitative descriptions. One advantage of this approach is that quantitative information may be propagated across constraints, providing information about landmarks of model variables whose values have not even been measured. Another advantage is that uncertain quantitative knowledge can be expressed (in the form of intervals), and used effectively. Third, multiple behaviors that satisfy the known quantitative constraints now carry quantitative predictions, easing the problem of differential diagnosis (chapter 6).

Q2 does not reason with a continuous stream of quantitative measurements. Instead, it takes descriptions of values at the endpoints of time-periods of monotonic change. However, measurements at other time values can be handled as described later (see chapter 6).

Karp and Friedland [1987] also share the goal of integrating qualitative and quantitative constraints in reasoning about mechanisms. They create a frame for each model variable at each instant, capable of representing a rich variety of algebraic equations and inequalities involving that value, plus frames for interactions between constraints. While the expressive and inferential power of their approach is potentially very large, so is the potential for combinatorial explosion, since there is no clear structure on the types of constraints and the circumstances under which different types of constraints are applied. This may be part of the reason why Karp and Friedland disallow multiple behaviors, instead requiring possible divergent futures to be disambiguated at each potential branch point. It is not clear how this can be done reliably.

Simmons' [1986] quantity lattice, and Sacks' [1987] hierarchical inequality reasoner are more powerful methods of arithmetic reasoning than the package currently in Q2. Extensions along these lines are possible.

## 2.4 Discussion

After assimilating a set of quantitative observations, the refined quantitative descriptions of surviving behaviors are precisely what is needed for differential diagnosis, for example by selecting a quantity whose intervals in two different behaviors are non-overlapping, and testing for its value. Work on diagnosis from first principles by Davis [1984], Genesereth [1984] and Reiter [1987] provides methods for optimizing the selection of new tests. An example is given in chapter 6.

It should also be possible to perform a sensitivity analysis [Raiffa 70] on the results of the propagation, to assess the sensitivity of Q2's conclusions

to variations in the quantitative observations. This may be a fruitful direction for future work.

Let us now look briefly at how the Q2 approach can apply to other qualitative simulation methods.

### 2.4.1 Constraint based quantitative reasoning and other qualitative simulation approaches

As described, Q2 works with constraints and their landmark arguments. However, every model variable value at every distinct time point can be associated with an interval constraining its quantitative value, even if neither the quantity space defined by the model nor the qualitative simulation has assigned it a symbolic name (a "landmark"). Since ignoring constraints that have argument(s) without distinct symbolic names is unnecessary, and since processing such constraints can help the Waltz filtering process, the current implementation of Q2 as a subset of Q3 extends the original Q2 described in this chapter by also using those constraints with one or more non-landmark arguments. Implementationally, this is a simple extension: if a value is not named as a landmark, Q2 simply generates a name automatically for internal use. Thus, the value of model variable VARX at time point Tn, if not already a landmark, is given the name "VARX@Tn" which is invisible to the user but used by Q2.

Liberating Q2 from the requirement that a quantity space be an ordered list of landmarks makes the quantitative inference approach of Q2 much more applicable to qualitative simulation using the qualitative category representation, in which a quantity space is defined with purposely "vague" values such as $\{low, medium, high\}$. Qualitative simulation using the $\{-, 0, +\}$ sign based approach of de Kleer and Brown [1985] is also now consistent with quantitative inference using the methods of Q2. One could even consider mixing the landmark, qualitative category and sign based approaches to qualitative simulation, resulting in quantity space descriptions like

$$\{-, 0, low, medium, high, TOP\}$$

where "$-$" is a sign, *medium* a qualitative category and $TOP$ a landmark. Quantitative inference could proceed without regard to the qualitative representation of a value, and only later categorized appropriately.

Viewing quantitative inference outside the context of the type of underlying qualitative simulator, Q2 can be more easily compared with Shen and Leitch's recent work on FuSim [1991]. FuSim uses a generalization of intervals

called fuzzy intervals, which can have "fuzzy" boundaries if desired instead of sharp boundaries as required by Q2. However, the meaning of these fuzzy intervals is not clear when excess width occurs in interval calculation. This semantic ambiguity may result in lack of correctness guarantees in the qualitative simulation component of FuSim as fuzzy intervals with excess width and unclear semantics are used to help determine qualitative behaviors.

### 2.4.2 Quantitative processing: during versus after qualitative simulation

Q2 originally applied quantitative knowledge to individual, complete qualitative behaviors from the output of QSIM. A subsequent implementation interleaved quantitative and qualitative processing, so that quantitative inferences could be applied to partially complete qualitative behaviors. Where a quantitative inconsistency can be identified at an early stage, an entire subtree of qualitative behaviors can be eliminated, greatly increasing the efficiency of the overall simulation. When a subtree cannot be eliminated, the efficiency may be decreased because every node in the subtree will have been subjected to Q2 instead of just the leaves. This potential decrease is mitigated by retaining the results of quantitative inference from the previous state in the simulation to initialize further inference based on new constraints instantiated by the current state.

Experience has shown that interleaving is usually but not always faster. Interleaving also has the advantage of conceptual consistency, as quantitative inference is treated as a filter applied to each new state created by the simulation just like other filters present in QSIM.

---

In this chapter have discussed adding incomplete quantitative knowledge to a qualitative simulation framework. Quantitative inference by constraint propagation is done using Waltz filtering.

While this chapter has discussed the bathtub, numerous other systems have also been modeled, some of which are discussed in later chapters. In the next chapters techniques are developed for extracting better quantitative inferences than those obtained by Waltz filtering, finally arriving at an integrated form of simulation called *qualitative-quantitative simulation* and its implementation Q3, in chapter 5.

# Chapter 3

# Supplemental Algorithms for Interval Inference: Target Interval Partitioning and Behavior Splitting

So far, results have depended upon Waltz filtering with interval labels [Davis 87] (see figure 2.6). However, the need for additional methods for deriving better inferences has motivated the development of more powerful techniques. These are called *target interval partitioning* (TIP) and *behavior splitting* and are explained in this chapter.

Excess width is a fact of life in interval calculations, occurring in many cases when a variable appears more than once in an expression. The amount of excess width can often be reduced (or eliminated) by algebraic rearrangement of an expression to be calculated. For the "natural interval extension" form of an expression [Moore 79], excess width can increase as $o(w(X))$, where $X$ is an operand in the expression and $w(X)$ is its width. For the "centered normal form" and "mean value form" [Moore 79] excess width is as high as $o(w(X)^2)$. This means that for wide $X$, the natural interval extension tends to give better results, and for narrow $X$, the $o(w(X)^2)$ forms are better.

A detailed discussion of algebraic rearrangement is beyond the scope of this work. However, the fact that excess width tends to increase with increasing widths of arguments leads to the observation that breaking an argument into narrower pieces — interval *subdivision* — and analyzing each piece separately will enable results containing less excess width. This idea underlies both TIP and behavior splitting. These techniques use subdivision to produce better inferences than those obtained by Waltz filtering alone. Their development was motivated by the need for better inferences and they have been found useful for that purpose. We explain each of them in the following sections.

## 3.1 Target Interval Partitioning

Models containing point specifications will not, in general, reflect the reality that they model with complete truthfulness. We have been dealing with this problem by using intervals instead of numbers. This is a way of

generalizing a model to help it approach a correct description of an imprecisely known reality. Inferences based on the model will then entail mathematical operations on intervals instead of on numbers. While individual arithmetic operations such as those in $\{+, -, \times, /\}$ are defined on intervals with little difficulty [Moore 66; 79], solving for an interval valued variable in an equation so as to achieve a result that does not have excess width is, as pointed out above, not straightforward. As a trivial example, $Y = X - X = 0$ yet, if $X \in [0, 1]$, a straightforward calculation of $[0, 1] - [0, 1]$ gives $Y \in [-1, 1]$ which is true but contains excess width and so is not an optimal answer.

Existing algorithms such as Asaithambi et al. [1982] and Kolev et al. [1988] rely on partitioning the intervals associated with the various arguments of a function as a stepping stone to finding a good or optimal interval for the range of the function. These algorithms are based upon the framework developed by Moore [1966]. The new algorithm is based on partitioning of the target interval only and is therefore called *target interval partitioning*, or TIP. It has the advantage of simplicity, and although not guaranteed to produce optimal results has been found useful.

TIP tests sub-intervals of the interval that is to be optimized, iteratively trying smaller and smaller sub-intervals as it attempts to rule out a sub-interval to narrow the target interval. Sub-intervals that lead to an inconsistency may be eliminated from the target interval, thereby narrowing it. For example, if $Y = X^2 - X$ and $X$ is given an initial interval of $[0, 1]$, ordinary Waltz filtering concludes $Y \in [-1, 1]$ (see figure 3.1, top). However if $Y$, the target interval, has its subinterval $[-1, -.5]$ tested by propagating through the constraint network representing $Y = X^2 - X$ using ordinary Waltz filtering, it turns out that there is no consistent way to label the network. Thus $[-1, -.5]$ can be removed from consideration for possible values of $Y$. We can do the same partitioning procedure on $[-.5, 0]$. Eventually TIP will conclude that the low bound of $Y$ is $-.25$. The same procedure, oriented this time to reducing the high end of $[-.25, 1]$ finally concludes that $Y \in (-.25, 0)$.

The algorithm may be described as follows, exemplified by figure 3.1.

1. Represent the equation as a network of constraints. For example $\{+, -, \times, \div$,exponentiation$\}$ are constraints among three values, whereas $\{=, <, >\}$ exemplify binary constraints. Once a set of equations is represented as a constraint network (see also appendix B note 2), it can be processed by the TIP algorithm.

2. Repeat until no further significant change occurs:

(a) Set the target to a sub-interval (the test interval) of its current interval, as illustrated in figure 3.1.

(b) Propagate the effects of this change throughout the network [Davis 87], using appropriate interval operations to process each constraint. For example, processing $Z = X + Y$ is shown in figure 2.6. If there is no consistent labeling for the network given the test interval, the test interval may be removed from the current interval. If there *is* a consistent labelling, test a sub-interval of the test interval the same way.

In its most basic form, TIP takes a relatively unsophisticated approach to deciding what sub-interval to test. It starts by testing the upper (or lower) half of the target interval, then testing half of the previously tested sub-interval, iteratively shrinking the tested sub-interval until one is found that can be eliminated. The sub-intervals tested always share an endpoint with the target interval, for the pragmatic reason that if a sub-interval in the interior of the target were to be ruled out, the target would no longer be an interval but rather 2 disjoint intervals. Disjoint intervals cannot be represented as a single interval and hence would not fit easily into our interval representation for partial quantitative information. The representation could be extended to describing partial quantitative information as *sets* of intervals to support a version of TIP that tested interior sub-intervals. This would make TIP more computationally complex. In addition there would be the overhead of using sets of intervals instead of intervals everywhere as a representation for partial quantitative information. Yet such an expanded capability would be useful only very occasionally, as when a constraint such as a square root "splits" an interval at one terminal into two at another: $\sqrt{[1,4]} = \{[-2,-1],[1,2]\}$. Even then, the sign will usually have been determined by the qualitative component of the simulation, ruling out one of the two intervals in the set.

As illustrated, when TIP partitions a target interval or one of its partitions, or any node in a tree such as that in figure 3.1, it breaks the interval into two sub-intervals of equal width. However other partitions may be implemented instead. The current implementation used in this research partitions an interval by testing 1/3 of it instead of 1/2. TIP works slightly faster with a 1/3 and 2/3 partition than with a 1/2 and 1/2 partition, however we do not claim the current method is the best one possible. A more sophisticated analysis of possible TIP variations might enable a slight additional speed-up.

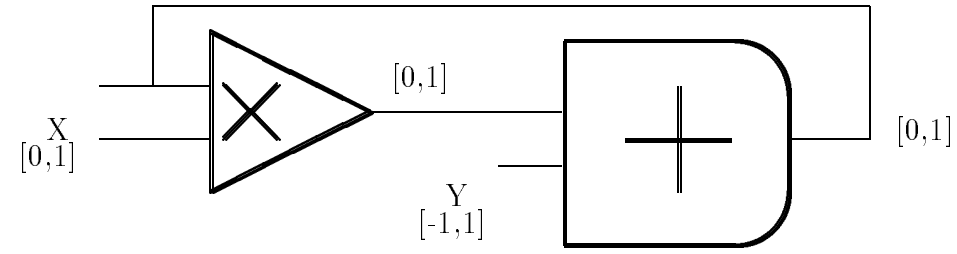
### 3.1.1 Variations and extensions

The TIP algorithm can be modified in various ways:

Target Interval Partitioning (TIP)

- Try to rule out pieces of a target interval.

EXAMPLE: $Y = X^2 - X$



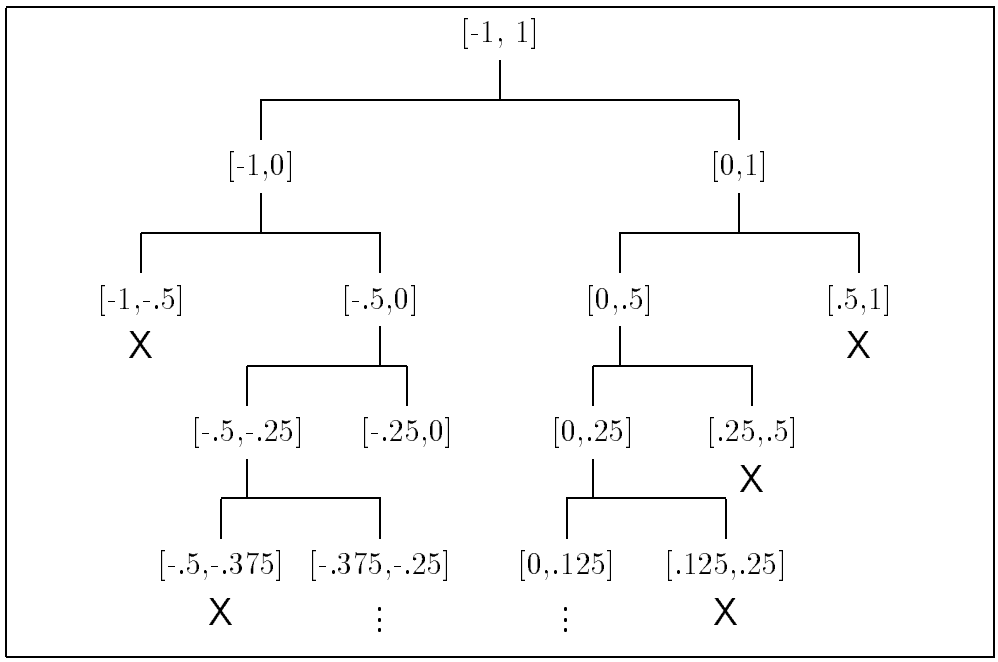




Figure 3.1: **Target Interval Partitioning**

Test the lower half of a target interval for consistency, using Waltz filtering. If it is consistent, then test the lower quarter, eighth, etc., until an inconsistent sub-interval is detected. Do the same but for the higher half, quarter, etc. Inconsistent sub-intervals are marked with an "X" in the figure; the algorithm then continues with the sibling node. For the example, $Y = X^2 - X$, TIP gradually narrows $Y$ toward, yet never quite reaching, $[-.25, 0]$. Termination may be insured by testing a sub-interval only if its width exceeds some $\epsilon$.

1. As described, TIP proceeds by halving the intervals at parent nodes (figure 3.1. However other partition methods can easily fit into the same framework. A parent could be partitioned into sub-intervals of 1/3 and 2/3 the width of the parent, with the sub-interval to be tested having the 1/3 width. This has been observed to be slightly faster and is the method currently employed by Q3, although no claim about optimality is made. Clearly, however, a very skewed partitioning will tend to be slow because only small sub-intervals could be ruled out at each iteration.

2. Halving a target interval into two sub-intervals implicitly assumes the number line is linearly scaled — that, for example, the midpoint of the interval $[1, 100]$ is 50.5, both arithmetically and in principle. Logarithmic scaling however would dictate a midpoint of 10. Pragmatically, choosing a midpoint of 50.5 is preferable if sub-interval $[1, 10]$ is more likely to be ruled out than sub-interval $[50.5, 100]$, and *vice versa*. I have not established whether linear or the logarithmic scaling provides a better ontology in general, and regardless of any general tendency different problems will have different characteristics. One can — perhaps should — take the ontology issue one step further by arguing that since neither ontology is clearly superior, the best approach is to hedge by using an average of the partition points dictated by the two approaches.

3. Sometimes TIP will be faced with a target or other parent interval with an infinite endpoint. Then, neither the linear nor the logarithmic ontology provides guidance in partitioning into sub-intervals. Q3 simply uses a predefined constant to add to the finite bound of a parent interval $[l, \infty]$ (or subtract from the finite bound of $[-\infty, h]$) to get a partition point. Due to insufficient ontological guidance, the value of this constant is necessarily somewhat arbitrary.

4. One case where the linear ontology is clearly superior to the logarithmic, is when a bound of a parent interval is zero. The logarithmic ontology provides no non-zero midpoint whereas the linear one does. (When qualitative reasoning is also taking place, as in Q3, an interval rarely contains zero in its interior.)

## 3.2 Behavior Splitting

Another method that has been found useful for handling problems involving excess width in interval calculations we call *behavior splitting*. Because excess width tends to become more severe as the widths of arguments in interval calculations increase, it can often help to convert a calculation involving

an interval with large width into separate calculations involving parts of the interval.

For example, consider the familiar simple case of calculating $X - X$, which we know to be 0. Given $X \in [0, 1]$, we could calculate

$$X - X \in [0, 1] - [0, 1] = [-1, 1]$$

which is true but too wide. If instead we split $X = [0, 1]$ into two cases $X_1 = [0, \frac{1}{2}]$ and $X_2 = [\frac{1}{2}, 1]$ we can calculate

$$\begin{aligned} X - X &= (X_1 - X_1) \cup (X_2 - X_2) \\ &\in ([0, \frac{1}{2}] - [0, \frac{1}{2}]) \cup ([\frac{1}{2}, 1] - [\frac{1}{2}, 1]) \\ &= [-\frac{1}{2}, \frac{1}{2}] \cup [-\frac{1}{2}, \frac{1}{2}] \\ &= [-\frac{1}{2}, \frac{1}{2}] \end{aligned}$$

thus inferring $X \in [-\frac{1}{2}, \frac{1}{2}]$ instead of $X \in [-1, 1]$.

Taking this further, we could split $X = [0, 1]$ into smaller, more numerous cases. For example, suppose we split $X$ into 10 cases each of width $\frac{1}{10}$. For each one,

$$X_n - X_n \in [l, l + \frac{1}{10}] - [l, l + \frac{1}{10}] = [-\frac{1}{10}, \frac{1}{10}]$$

This technique is based on the fact that calculating with narrower intervals reduces excess width in results. In the limit, $X_n$ approaches zero width — a number such as 0.3 — and then $X_n - X_n = 0$ with no excess width.

Since $X - X$ does not vary with time, a qualitative simulation of it contains only the first time point, $T0$, whose value is known *a priori* to be 0. For a simulation of a time varying model, behaviors will be represented by more complex constraint networks containing instantiations of the mean value constraint (chapter 2). Behavior splitting is used in a more complex example in section 5.2.

We have shown behavior splitting for $X - X$ because of its clarity. In this case of course a symbolic, algebraic approach would eliminate the problem entirely as we know that $X - X = 0$. However the comparable expression $X - X^{0.99999}$ is not reducible symbolically. Thus in many cases purely symbolic approaches are insufficient, yet behavior splitting, like many numerical techniques, may still be used.

### 3.2.1 Behavior splitting in simulation

Behavior splitting has been found useful for helping to rule out bounds of $\infty$. A behavior with landmark $L \in [lo, \infty]$ can be split into two behaviors, one with $L \in [lo, hi]$ and one with $L \in [hi, \infty]$. The behaviors would otherwise be identical. Further processing would address each behavior individually, with the result that one of them could end up being ruled out. An example of this appears in section 5.2.

There are times when ambiguity exists between the ordering of a number and an interval. If variable T1 $\in (0, 100)$ and variable $t = 30$, for example, T1 could be $<, =$ or $> t$. Suppose qualitative time point T1 $\in (0, 100)$ and we have a measurement that took place at $t = 30$. The time $t = 30$ could occur before T1 if T1 $\in (30, 100)$, at T1 if T1 $= 30$, or after T1 if T1 $\in (0, 30)$. By splitting the behavior into these three cases for T1, the measurement can be assigned a location on the time axis of each of the three behaviors generated by behavior splitting, such that its ordering relationship with T1 is unambiguous. As another example, one might wish to insert extra time points into a simulation in order to have a simulation with more time points, therefore a smaller time step size, and consequently (as in ordinary numerical simulation), more accurate simulation results (chapters 4 and 5). Behavior splitting can allow such extra time points by providing unambiguous orderings between them and existing qualitative time points described by intervals, hence determining where to insert them into the simulation given the behavior.

## 3.3 Conclusion

In this chapter we have described target interval partitioning and behavior splitting, two algorithms extending constraint propagation using interval labels. These techniques will be shown to usefully supplement standard Waltz filtering in chapter 5, particularly when it is desired to narrow particular intervals (such as to produce a "gap" between adjacent landmarks). Like Waltz filtering, TIP and behavior splitting are generally applicable to constraint propagation problems involving interval labels. In the next chapter we present another technique whose use is specific to *simulation* problems.

# Chapter 4

# Time Point Interpolation: Bringing Qualitative and Quantitative Simulation Closer Together

## 4.1 Introduction

One aspect of model-based reasoning is simulation. A model is defined and its behavior(s) inferred through qualitative or quantitative simulation. The simulated behaviors then facilitate tasks that one expects model-based reasoning to aid in, such as forecasting, monitoring, diagnosis, and design.

When qualitative and quantitative simulations are closely integrated, the benefits fall into 3 categories. One benefit is in improving qualitative simulation results. A second benefit is in improving quantitative simulation results. A third benefit is conceptual. People have thought of quantitative simulation and qualitative simulation as different and distinct, and an integrated representation establishes their compatibility at a basic level.

## 4.2 Brief Overview of Previous work

Combining quantitative and qualitative information in simulation has been a goal of a number of researchers (chapter 1). In chapter 2 we showed how partial quantitative knowledge can be added to qualitative simulations of continuous systems, and the benefits to qualitative reasoning that ensue. Quantitative knowledge was expressed as intervals within which the values of model variables in a qualitative simulation are known to lie at given qualitative time points. Inference about the possible ranges of values of other model variables at various qualitatively significant times is done by applying interval arithmetic [Moore 79] to evaluate expressions. The work described in chapter 2 has been applied to several domains including reasoning about process control in the chemical engineering domain [Dalle-Molle 89].

Previous simulation work on combining quantitative and qualitative information has treated quantitative and qualitative information as distinct types of knowledge. Chapter 2 for example used qualitative simulation to

create sets of constraints such as additions, multiplications, and mean value theorem instantiations. Each qualitative behavior was represented with such a set of constraints. These constraints among symbols were used to propagate partial quantitative information in the form of intervals, to other symbols for which *a priori* quantitative information was not given. Thus a qualitative simulation was annotated with quantitative information. The result was unlike a conventional quantitative simulation in that each time point signifies an event of qualitative significance, and no other time points can appear. In contrast, a quantitative simulation may have as many time points as desired.

The present chapter, in contrast, treats qualitative and quantitative simulation within a single representational framework. This means that values explicitly represented in the simulation no longer need to be qualitatively significant. They may instead describe quantitative properties of a simulation trajectory for times and model variable values that have no qualitative distinction. Let us look at how this is achieved.

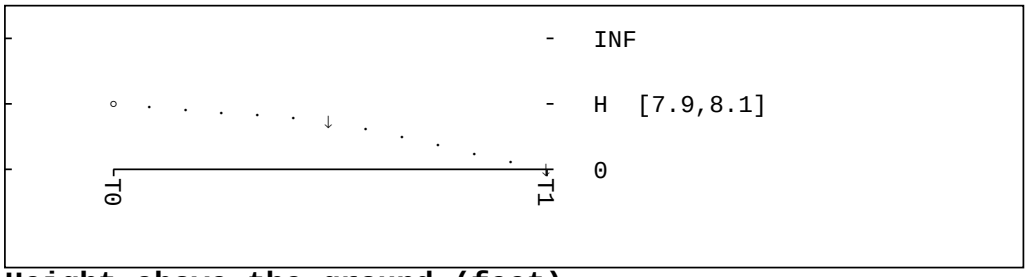
## 4.3 A Representation for Combined Quantitative and Qualitative Simulations

Consider the qualitative simulation of a dropped object shown in figure 4.1. Given gravitational acceleration of $[-33, -31] feet/sec^2$, and an initial height of $[7.9, 8.1]$ feet, inferences about other values in the trajectory of this system are rather weak, despite use of Waltz filtering (chapter 2) and further processing using TIP (chapter 3).

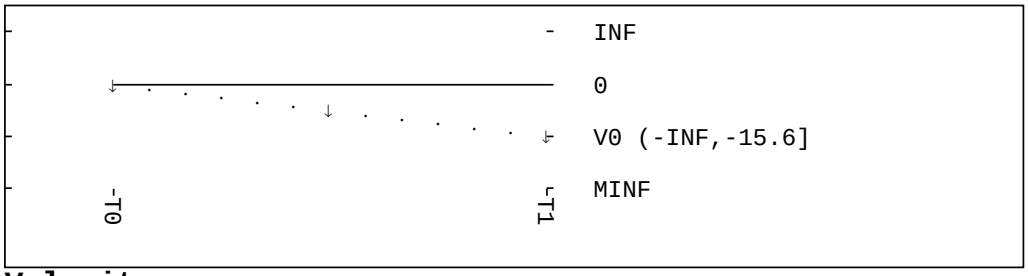
Even the best methods for dealing with explicitly represented knowledge may not be enough, when considerable information is not explicit but rather implicit. Since `T0` $= 0$ and `T1` $\geq 0.489$ (see banner, figure 4.1), clearly `T0` $< t = 0.45 <$ `T1`. In figure 4.2, the numerically defined time step 0.45 was inserted between pre-existing qualitative time points `T0` and `T1`. Note that no new quantitative information or assumptions have been made, since the information that 0.45 is between `T0`$= 0$ and `T1` $\geq 0.489$ is already implicit in figure 4.1.

Much narrower intervals could then be inferred for the symbols in figure 4.2 compared to the same symbols in figure 4.1. In the case of figure 4.1, there is no upper bound on how long it takes for the object to hit the ground. Yet figure 4.2, which explicitly (instead of only implicitly as in figure 4.1) represents the fact that time 0.45 occurs before time `T1`, does provide a definite upper bound on how long the object takes to fall (banner, figure 4.2).

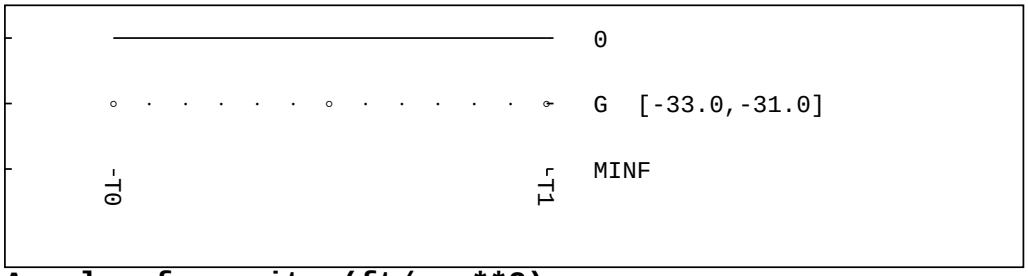
```
Structure: Constant gravity (no friction)..
Initialization: Drop an object downward  (S0)
Behavior 1 of 1:    (S0 S1 S2).
Final state: (TRANSITION FINAL GF COMPLETE), (TRANSITION-IDENTITY), T<INF.
 Time point bounds:  T1=[0.489,INF)
```




**Height above the ground (feet)**




**Velocity**

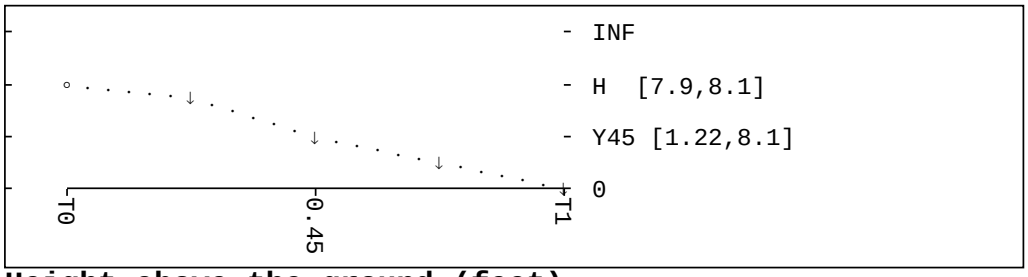



**Accel. of gravity (ft/sec**2)**

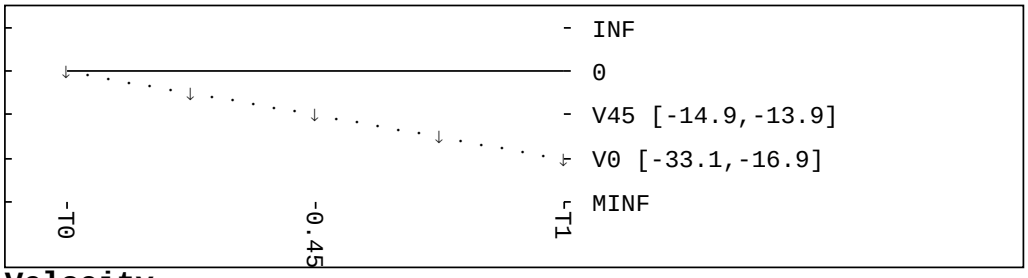
Figure 4.1: Qualitative simulation of an object dropped from about 8 feet. Intervals for `H, G` and `T0` are given. Target interval partitioning improved results for `V0` from $(-\texttt{INF}, 0)$ to $(-\texttt{INF}, -15.6)$, and for `T1` (shown in banner) from $(0, \texttt{INF})$ to $(0.489, \texttt{INF})$. However, these results are still rather weak and will be improved later (see figure 4.2). Note that all symbolic values (such as `V0` and `T1`) have qualitative significance: they indicate starting or ending values, or in the case of INF and MINF, extreme values for the domain of a model variable.

———————————

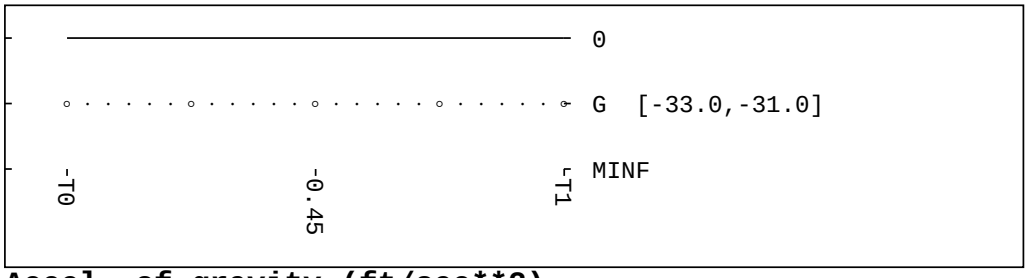
Structure: Constant gravity (no friction)..
Initialization: Drop an object downward (S0)
Behavior 1 of 1: (S0 S6 S5 S7 S2).
Final state: (TRANSITION FINAL GF COMPLETE), (TRANSITION-IDENTITY), T<INF.
Time point bounds: T1=[0.521,1.03]




**Height above the ground (feet)**




**Velocity**




**Accel. of gravity (ft/sec**2)**

Figure 4.2: Time 0.45 is present implicitly between T0$= [0, 0]$ and T1$\geq 0.489$ (see figure 4.1).

Putting it there explicitly reduces the time step size, thereby reducing the discretization error and resulting in inferences about values that are much better than in figure 4.1. The simulation trace now has characteristics of both qualitative and quantitative simulation, in that the qualitative trace is present, and the system state at $T = 0.45$ is also represented, although that state would never be present in a purely qualitative simulation.

———————————

Figure 4.2 includes all of the qualitatively significant symbolic values of figure 4.1, plus some new ones which have no qualitative significance. `Y45`, for example, represents one of many heights which are between maximum height `H` and minimum height 0, and at which the object's height is decreasing.

Inferring narrower intervals can help rule out qualitatively plausible behaviors and qualitatively plausible models (e.g. fault models, chapter 6). Of course, narrower intervals also constitute better quantitative predictions about uncertain symbol values.

Let us look at how the representation illustrated by figure 4.2 can be viewed as either a quantitative simulation or a qualitative one.

- Figure 4.2 is a *quantitative* simulation. There are three time points. `T0` has the default value of 0, which is a precise number. The second time point, 0.45, is also a number, giving us a step size of $0.45 - 0 = 0.45$. The final time point, `T1`, is quantitatively known only to within $[0.521, 1.03]$. Other values present in the simulation, such as `H`, `G`, and `V0` (which also happen to have qualitative significance) are also imprecisely known numbers. Imprecise numbers are the best that can be obtained because the quantitative model specifications include starting height `H` $= [7.9, 8.1]$ and gravitational acceleration `G` $= [-33, -31]$ which are imprecise. In chapter 5 we will see how some of this imprecision can be removed by interspersing more time points (reducing the average step size of the simulation), thereby instantiating the model constraints at more time points resulting in more explicitly represented values and constraints among them. This process is similar to reducing the step size in a numerical simulation and tends to result in better inferences about the path of the simulation (notice the improved quantitative inferences about symbols in figure 4.2 compared to the same symbols in figure 4.1). Because the representation in figure 4.2 contains a quantitative description of the path of the system that is not limited to qualitatively significant values, it constitutes a quantitative simulation.

- Figure 4.2 is also a *qualitative* simulation: all qualitatively distinct values in the simulation [Kuipers 86] are present. For example, `T0` is when the object is dropped, `T1` is when it hits the ground, `H` is the initial height from which it is dropped, `G` is gravitational acceleration, and `V0` is the final, maximum velocity. Of course, symbolic values such as these need not be present in a purely quantitative simulation, where reasoning is based on numerical rather than symbolic values.

  The simulation shown in figure 4.2 illustrates the unified representation which constitutes the objective of this chapter. A comparison of the

weak quantitative conclusions shown in figure 4.1 with the much stronger ones of figure 4.2 shows that this representation leads to improvement in inferential capability.

We now take a closer look at how the improved inferences in figure 4.2 are obtained.

## 4.4 Increasing the Power of Inference

There are at least two general approaches to achieving better inferences. One is extracting more from the current representation (chapter 3). The second is by improving the representation so that an inferencer has more to work with. The model trajectory represented in figure 4.2 has more states and hence more explicitly represented values than the representation in figure 4.1. We now explain the representation in more detail, and how this allowed better quantitative inference about the trajectory.

### 4.4.1 The knowledge representation

Uncertainty in value is represented using intervals. *Constraints* among values are also represented. Most constraints are instantiations of *constraint templates* (sometimes called "model constraints") defined by the model, which are instantiated at each time point in a simulation trajectory. For example, we might choose to model the momentum of a falling object by using as a template the general equation

$$momentum = mass \times velocity$$

Each state of a qualitative simulation would instantiate this template as constraints among values. For example, one of these constraints would relate values for finalMass, finalVelocity, and finalMomentum by the equation

$$finalMomentum = finalMass \times finalVelocity$$

The model whose behavior is shown in figure 4.1 does not describe momentum, but it does contain the derivative constraint

$$velocity = d(height)/dt$$

This constraint template is instantiated by the program for the values V0, 0 and T1 of figure 4.1 (as well as for 0, H and T0 and for two other tuples containing G). Instantiation is achieved using the mean value theorem (section 2.2.1). The form of the mean value theorem used here is

$$\Delta height = mean(velocity) \times \Delta time \tag{4.1}$$

which is instantiated as:

$$\begin{aligned} \mathtt{H} - 0 &\in [minimum(\mathtt{V0}), 0] \times (\mathtt{T1} - 0) \\ \mathtt{H} &\in [minimum(\mathtt{V0}), 0] \times \mathtt{T1} \end{aligned} \tag{4.2}$$

where $mean(velocity) \in [minimum(\mathtt{V0}), 0]$. This constraint among the qualitative values H, V0, and T1 can be used to try to narrow the intervals for each of them by Waltz filtering as in chapter 2 (figure 4.1 supplemented Waltz filtering with target interval partitioning, chapter 3). "0" is also an important qualitative value in the constraint, but its quantitative value is unambiguous and cannot be improved.

So far, the representation is that of figure 4.1. Let us look at how it can be improved, eventually enabling the improved results of figure 4.2.

- From figure 4.1, T1 $\geq 0.489$.
- Therefore time = 0.45 occurs between the qualitatively significant time points T0= 0 and T1.
- Time point 0.45 is explicitly added to the representation of the behavior that was originally inferred from the falling object model in figure 4.1. This is shown in figure 4.2.
- Name the velocity at time 0.45 "V45" and the height "Y45".

We now show how this representation of the problem enables better inferences.

### 4.4.2 Extracting better results

We can instantiate the constraint template of equation 4.1 as equations

$$\begin{aligned} \texttt{Y45} - \texttt{H} &\in [minimum(\texttt{V45}), 0] \times (0.45 - 0) \\ 0 - \texttt{Y45} &\in [minimum(\texttt{V0}), maximum(\texttt{V45})] \times (\texttt{T1} - 0.45) \end{aligned} \tag{4.3}$$

We also get new instantiations of the model constraint $\Delta velocity = mean(gravity) \times \Delta time$:

$$\begin{aligned} \texttt{V45} - 0 &\in [minimum(\texttt{G}), maximum(\texttt{G})] \times (0.45 - 0) \\ \texttt{V0} - \texttt{V45} &\in [minimum(\texttt{G}), maximum(\texttt{G})] \times (\texttt{T1} - 0.45) \end{aligned} \tag{4.4}$$

from which the inference

$$\texttt{V45} \in [-14.9, -13.9]$$

can be calculated.

Equations 4.3, 4.4 and intervals given for H and G (and T0 = 0) enable inference of much narrower intervals for the values of T1 and V0. Compare the wide intervals inferred for T1 and V0 in figure 4.1 to the much narrower ones in figure 4.2.

Note that further refinement of the simulation could be done by interpolating additional time points, leading to instantiation of more constraints and further improvement to the inferred simulation trajectory.

### 4.4.3 A larger example

A standard example used in the qualitative reasoning field is the oscillating mass on a spring. Interpolating a number of new states in this example resulted in figure 4.3.

Figure 4.3: Simulating the oscillation of a frictionless spring using time point interpolation several times.
The simulation was refined by interpolating states whose times are numbers, between preexisting qualitatively significant states whose times are symbolically labeled `T0, T1, T2, T3` and `T4` and were found by qualitative simulation using QSIM [Kuipers 86] (the particular values of time inserted were determined by hand in an *ad hoc* way, however an algorithm for determining them is the subject of the next chapter). As the number of interpolated states increases, inferred intervals for model variable values in other states tend to become narrower. An additional benefit is that information about the state of the system is represented explicitly at more points in its evolution.

## 4.5 The Benefits to Quantitative Simulation

In addition to benefiting qualitative simulation by being able to represent and use partial (or full) quantitative information when available, enabling better model and behavioral description, there are also benefits to quantitative simulation.

All the distinct qualitative behaviors that are consistent with the available quantitative information can be automatically generated. For example, water flowing into a tank with a partially clogged drain could result in an equilibrium water level at which the pressure at the drain is high enough to produce an outflow equal to the inflow. Or, it may result in overflow. The simulation method of this chapter will automatically find both behaviors, if both are possible in light of available quantitative information, and make distinct quantitative predictions for each behavior. Quantitative simulation alone cannot do this easily or automatically.

Also, the intervals inferred are correct (although they may be weak as in figure 4.1), in that the trajectory of an actual system conforming to the model will exhibit values within those intervals. This is so because quantitative inference is based on Waltz filtering, which is known to be safe [Davis 87; Berleant 90]). Numerical simulation is in general incapable of correctness guarantees, although interval simulation can provide them [Eijgenraam 81].

Finally, this technique is well suited to working with non-linear systems since no distinction is made between linear and non-linear systems. In contrast, the linear/non-linear distinction is important in the differential equations field because linear systems are much more likely to be solvable analytically than non-linear systems.

## 4.6 Summary

This chapter has addressed simulation of continuous systems, using a representation that supports both qualitative and quantitative descriptions of model behavior. I have described this representation and how it may be used to produce better descriptions of model behavior than are obtained using qualitative simulation, even when augmented with partial quantitative information as in previous chapters. The simulation results also embody the advantages of qualitative simulation, such as working with partial information, finding all qualitatively distinctive behaviors a model could exhibit, finding model limits (such as the point at which a dropped object hits the ground) and describing the model state at qualitatively significant time points. The next chapter will build on the state interpolation technique by embedding it in a technique for

producing simulations whose predictions improve as the computation expended increases, resulting in convergence and stability properties for the qualitative-quantitative simulation technique implemented by the program Q3.

# Chapter 5

# Progressive Simulation Refinement and *Q3*

In the previous chapter we showed in detail how interpolating a new time point into an existing simulation can result in better inferences. In this chapter, we develop interpolation of time points into a method termed *step size refinement.*

Step size refinement is an *adaptive discretization* method developed to suit our interval based representation for uncertain values. Adaptive discretization methods seek to reduce numerical error due to representing a continuous problem at a finite number of points, while maintaining acceptable efficiency, by varying the density of the represented points depending on the local characteristics of the problem. Two adaptive discretization methods that have been previously reported are *adaptive stepsize control* [Press et al. 86; Gear 71], and *multigrid* methods [Brandt 77; Briggs 87].

## 5.1 Step Size Refinement

First we overview the stages of the step size refinement algorithm. Each stage is discussed in more detail in the following subsections, which are each titled with the name of the stage given in this overview:

1. **Qualitative simulation:** Simulate the model with the QSIM qualitative simulator [Kuipers 86] or any simulator that produces qualitatively distinct behaviors encoded as networks of constraints on the values of model variables at the qualitative time points.

2. **Propagating quantitative information:** Assert quantitative information in the form of intervals, about values of the model variables in any state(s) of the simulation, such as the initial state. Propagate this information to other model variable values in the states of the simulated behaviors. This is done in the Q2 program (chapter 2).

3. **Locating a gap:** Find a gap between the quantitative values of TIME in adjacent states. For example, in figure 5.1, the left-hand column shows a gap between `T0`=0 (simulations start at time 0) and `T1`=$[305, \infty]$, of width=305.

4. **Interpolating a state:** Reduce the average step size of the simulation: interpolate a new state $S_i$ between the pair of adjacent existing states $S_1$ and $S_2$ that has the gap. Since the system trajectory must pass through the times in the gap on its way from $S_1$ to $S_2$, we can set the numerical value of time for $S_i$ to any number in the gap.

5. **Refining the quantitative intervals:** The newly interpolated state instantiates the various model constraint templates, thereby augmenting the constraint network representing the behavior with new constraints. Apply Waltz filtering as implemented in Q2 to propagate intervals and settle the network, as in "propagating quantitative information", step 2 above.

6. **Iteration:** Return to step 3, **locating a gap**, continuing to iterate until the desired precision is reached. This was done several times to figure 5.1(a) in the process of refining it into figure 5.1(b).

Let us now explain each of these steps in more detail.

### 5.1.1 Qualitative simulation

QSIM predicts a set of behaviors guaranteed to include a description of the actual behavior of any system described by the given QDE [Kuipers 86]. The qualitative simulation produces a representation of each qualitatively distinct behavior that a system conforming to the model might exhibit. Each behavior is represented as a constraint network, which will be used to propagate partial quantitative information in the form of intervals. The progressive refinement algorithm presented here is suitable for use with any qualitative simulation method that represents behaviors as networks of constraints among the values of the state variables in the various states of the simulation.

### 5.1.2 Propagating quantitative information

Each qualitative behavior defines a set of constraints, consisting of instantiations of arithmetic constraints from the constraint templates in the model definition, ordinal constraints among "landmark" symbols defined by the model for each model variable, and definite integrals over time intervals

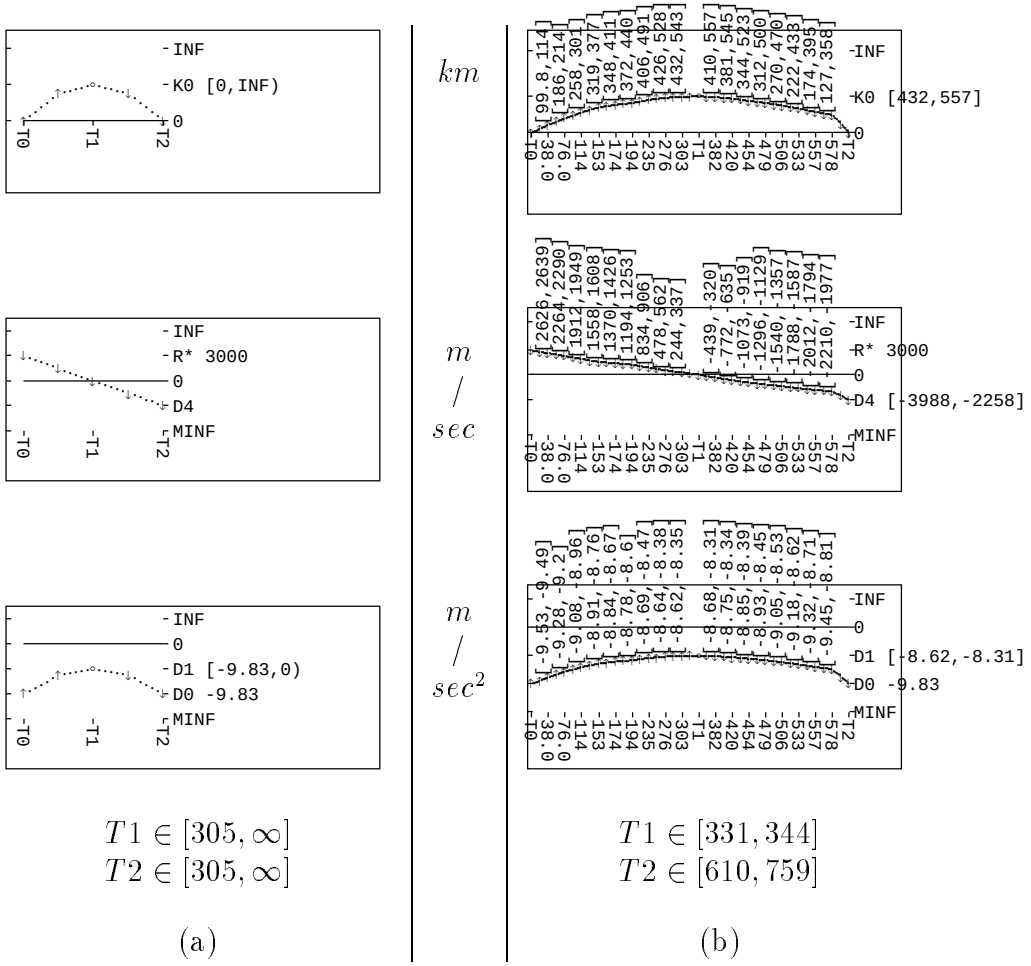



Figure 5.1: A rocket is fired upward at $3000 m/sec.$, less than escape velocity. 17 time points were interpolated, and the TIP and behavior splitting algorithms were also used when necessary (see text). Table 5.2 summarizes some of the key intermediate stages in the simulation.

appearing in that behavior (which are currently instantiated with the mean value constraint). Intervals are asserted as labels for these symbols and propagated over the constraints by Waltz filtering, chapter 2. Quantitative inference is rather weak over time intervals because of the coarse discretization defined by the qualitative behavior. For example, in figure 5.1(a), prior to step size refinement the only states in the simulation are those at time points `T0`, `T1` and `T2`. The inferences at that stage are far fewer and mostly weaker, compared to figure 5.1(b).

### 5.1.3 Locating a gap

First, find the largest step size between two adjacent time points. This is where a new time point will be interpolated, reducing the largest step size in the simulation. Since time point values can be intervals a step size may also be an interval, in which case the maximum width of that interval is conservatively used to estimate step size. By reducing the maximum step size with each iteration of the algorithm such that in the limit it approaches 0, we will satisfy the main requirement for convergence, which is the property that the predictions of the simulation will gradually approach the actual trajectory of a system conforming to the model (section 5.3.2).

Once a time step has been chosen the algorithm looks for a *gap*, or region where the time points on either side of the time step do not overlap, to interpolate a new time point. Consider the step from time `3600` to time `T1`$= [3671, \infty]$ in figure 1.1. While the step size is potentially as great as $w([3600, \infty]) = \infty$, there is a gap where 3600 and $[3671, \infty]$ do not overlap. This gap is $[3600, 3671]$.

The midpoint of the chosen gap is preferred as the time value for a new interpolated state[1], as long as it reduces the enclosing step size by a prior arbitrarily chosen percentage $P$ (if the step size is finite). Otherwise some other point in the gap is chosen that will provide $P\%$ reduction. This ensures that the maximum finite step size of the simulation will decrease toward 0.

If a finite time point is interpolated into a potentially infinite step size width, then obviously the step size width cannot be reduced by any percentage. However, the boundary of the region in which convergence can occur moves

[1]In Q3, code for finding a point in a gap now modifies the point initially chosen in an attempt to find one of the roundest numbers in the gap. This helps reduce clutter in the display in order to better communicate with the reader [Tufte 83]. Empirically, this makes simulations more readable without interfering with convergence.

toward infinity. An example of this occurs in figure 1.1, for which the region of convergence starts out as $[0, 1014]$ but ends up $[0, 3671]$, as shown by the beginning and ending values of `T1`.

In a few cases both the maximum step size and the gap it encloses have infinite width. For example, the system of figure 5.2 has the qualitative time point `T1`=$\infty$, making both a gap and a step size between `T0` and `T1` of $[0, \infty]$. In such a case, the first interpolated time point must be somewhat arbitrary. This time point is 1.0 in figure 5.2. The result of this interpolation leaves a finite step size and an infinite one (naturally one cannot partition an interval containing an infinite bound into two finite width intervals). Subsequent interpolated time points can be inserted either in a finite time step or in the infinite step. The simulation of figure 5.2 avoided the infinite step, however a keyword argument to the step size refinement routine will cause it to choose the infinite time step when it observes the finite step size widths to be decreasing satisfactorily. In such cases step size refinement can occasionally "foray" toward the infinite time point while still converging in the finite region.

Under some circumstances such as excessively weak initial conditions there may be no gap, or no gap permitting some desired $P\%$ reduction of the enclosing step size. For example, in figure 5.1(a) there is no gap between times `T1` and `T2`. There are several possible approaches to creating or enlarging a gap.

- Attempt to narrow the interval values for the two time points demarcating the time step by using a specialized interval inference algorithm like TIP, which Q3 contains, or that of Kolev et al. [1988]. TIP was used in producing the simulation of figure 5.1.

- Use *behavior splitting* to force a place to interpolate a new time point. This was done once in the example of figure 5.1. Its use there will be explained more later. Behavior splitting can always be used to force a gap (chapter 3), but because it converts one behavior into two, doubling the future computational expense, resorting to it unnecessarily can lead to excessive computation.

- Represent and use *differences* between values in adjacent states explicitly. Thus even if $T1$ and $T2$ have overlapping intervals (and thus no gap), if their difference can be inferred to be at least $x$ then there is clearly a gap of width $x$ or more between them. Even though we may not know enough about the gap's location on the number line to be able to choose a number in it, a state could still be inserted and its distance in time from $T1$ and $T2$ specified as (positive) numbers or intervals.

The mean value rule uses difference operations explicitly, and could easily be used with difference values if Q3 were extended to represent them.

- Instead of interpolating a new state specified by a time point, specify its value for another model variable that does have a gap (section 5.4.2). Step size refinement of `Inside Temperature` was used in the air conditioner example of figure 1.2.

- Look for another time step that does have a gap. Or, if the gap exists but does not permit a desired $P\%$ reduction in the step size, use it anyway. This is the method currently invoked automatically by *Q3*.

- Additional information may be requested from the user. By identifying where overlapping intervals are preventing further analysis, Q3 can direct the user's attention to measurements that would particularly benefit the simulation.

A small expert system module would be useful in choosing and invoking method(s) for dealing with lack of a gap. This would be consistent with the current emphasis in the computer simulation community on seeking AI methods for increased autonomy in simulation [1].

### 5.1.4 Interpolating a state

This is what reduces the average step size of the simulation. By interpolating a new state between a pair of adjacent states we instantiate the constraint templates of the model definition, thereby creating more constraints. These new constraints enable the quantitative reasoning algorithms to infer more about the trajectory. This additional knowledge takes the form of both narrower intervals, and of explicit intervals describing system states that were not explicit in the simulation before.

State interpolation is the subject of chapter 4 and is described in detail there.

### 5.1.5 Refining the quantitative intervals

The new state is used to refine the simulation inferences. Q3 calls Q2 which propagates quantitative knowledge using the new constraints created by the new state. We use the *mean value constraint*, explained in chapter 2, to constrain states based on their temporal neighbors.

Interpolating a new state and applying the mean value rule tends to result in some intervals being narrowed, thereby improving numerical predictions, because of the convergence property (section 5.3.2 and appendix B) that results from smaller step sizes as in ordinary numerical simulation. When an interval is narrowed, the improvement tends to ripple outward to other states. The overall settling of the values describing the states of the simulation is a *Waltz filtering* process [Waltz 75] over interval values [Davis 87] and hence is guaranteed [Davis 87; Cooper 89; Berleant 90] not to narrow any interval too much.[2]

### 5.1.6 Iteration

The algorithm iterates by continuing to divide gaps by interpolating new states. While the user currently specifies the number of iterations, it will be straight-forward to apply goal-oriented termination criteria. The iteration could terminate when a desired level of quantitative precision is achieved on a model variable, or when a desired level of qualitative precision is achieved by refuting or guaranteeing a particular qualitative behavior. When gaps need to be created or widened, it would be desirable to decide automatically and intelligently when to apply the behavior splitting and TIP methods to create new gaps, as these methods are relatively expensive to apply. Criteria for deciding have been developed, described in the context of the example (section 5.2). A small expert system may be built to apply these criteria automatically in the future.

## 5.2 Example: The Returning Rocket

Let us now step through an example in some detail, one which requires different techniques available to Q3 at different points in the simulation refinement process. The example is based on a rocket in a gravitational field

[2]This correctness guarantee does not account for machine arithmetic error. However, Q3 handles that problem by subtracting a small percentage from the low bound of every interval calculation, and adding a small percentage to the high bound. This makes interval calculations safe as long as the amount by which the interval bounds are changed is greater than the error introduced by the machine in the calculation. For a Symbolics computer using single precision floating point numbers, a proportion of $10^{-6}$ = has been empirically found safe; for double precision numbers a proportion of $10^{-13}$ has usually been used although $10^{-15}$ or $10^{-16}$ should be safe. A switch in Q3 also allows use of rational instead of floating point numbers, with any desired round-off proportion. Use of rationals slows processing by a factor in $[1, 10]$, depending on the problem. Rounding can also be turned off entirely.

which decreases with distance. The qualitative constraints in the model appear in table 5.1. Table 5.2 and figure 5.1 provide simulation results.

The returning rocket model was given an initial velocity of $3000 m/sec.$ (less than escape velocity, so it must return to Earth) and parameters describing known data about the Earth.[3] As with all quantitative processing by Q3, it was first processed by Q2 (chapter 2) which provided a qualitative simulation annotated with some rough bounds (table 5.2, row *a*). Note that there is no gap between $T1$ and $T2$ because, although qualitatively $T2 - T1 > 0$, nevertheless

$$lowBound(T2) = 305 < highBound(T1) = \infty$$

These two facts are mutually consistent. Unfortunately, Q2 cannot rule out the behaviors in which the rocket escapes to $\infty$, even though they are quantitatively impossible.

The next stage in a Q3 simulation is step size refinement when this is possible (in cases where there is no gap in which to do step size refinement it is necessary to create one first). For this example, Q3 interpolated two new time points. After interpolating one new time point in each behavior ($t = 153$), not much improvement occurred (table 5.2, row *b*); still no qualitative behavior has been ruled out. However after interpolating two time points, row *c*, the only qualitative behavior to survive is the rocket falling back to Earth, which is the behavior shown in figure 5.1 and summarized by table 5.2.

Having achieved the useful subgoal of pruning behaviors, a user (or a future expert system for managing the techniques available to Q3) should reevaluate the status of the simulation and decide which technique is best to use next. In order to infer narrow bounds for the system during the fall back to Earth, we need a gap between $T1$ and $T2$ in which to do step size refinement. To get this gap it is necessary to infer an increase in $lowBound(T2)$ and a decrease in $highBound(T1)$ sufficient to produce a gap. The two techniques Q3 uses for creating gaps are TIP and behavior splitting. TIP is always tried first because behavior splitting creates new branches in the behavior tree and hence may lead to computational complexity problems especially if many new branches are being created (section 3.2.1). Thus, the TIP algorithm (chapter 3) is tried next.

Q3's implementation of TIP assumed $T2 \in [318, 449]$, let the simulation settle via Waltz filtering as usual, and discovered that settling eventually

[3]The model ignores the Earth's rotation and air resistance. Note that many gravitational bodies have no atmosphere, and that rotational effects are small near the rotational axes.

```
; ============================================================
; A model of gravity that decreases with distance.
; ============================================================

(define-QDE escape-velocity
  (text "Gravity decreases with height as r''=-GM/r^2")
  ; Define model variables and their qualitative values.
  (quantity-spaces
    (r       (0    sea-level inf)  "meters from Earth's core")
    (r^2     (0    inf          )  "distance squared"        )
    (h       (0    inf          )  "meters above surface"    )
    (km      (0    inf          )  "kilometers above surface")
    (surface (0    s*           )  "depth of Earth"          )
    (dr/dt   (minf 0 r*      inf)  "velocity, m/sec"         )
    (d2r/dt2 (minf 0         inf)  "acceleration, m/sec^2"   )
    (G       (0    G*           )  "Gravitational constant G")
    (earth-M (0    M*           )  "Mass of Earth M"         )
    (K       (0    K*           )  "K (=G*M)"                )
    (-K      (-K*  0            )  "-K"                      )
    (=1000   (0    1000         )  "Thousand"                ))
  ; The model defines these constraint templates
  (constraints
    ((mult   km      =1000   h ))
    ((add    surface h       r  ) (s* 0 sea-level))
    ((mult   r       r       r^2)              )
    ((mult   G       earth-M K  ) (g* m* k*)   )
    ((minus  k       -k         ) (k* -k*)     )
    ((mult   d2r/dt2 r^2    -K  )              )
    ((d/dt   r       dr/dt      ))
    ((d/dt   dr/dt   d2r/dt2    ))
    ((constant surface))
    ((constant G))
    ((constant earth-M))
    ((constant k))
    ((constant -k))
    ((constant =1000))) )
```

Table 5.1: A qualitative model describing the gravity experienced by a rocket or other object fired upward. Gravity decreases with distance according to the standard nonlinear 2nd order differential equation $\frac{d^2r}{dt^2} = \frac{-GM}{r^2}$.

led to an inconsistency. Before successfully ruling out $[318, 449]$, Q3 unsuccessfully tested $[318, 32102]$, $[318, 10912]$, $[318, 3849]$, $[318, 1495]$, and finally $[318, 710]$, choosing smaller and smaller sub-intervals to test in accordance with Q3's version of TIP (see section 3.1.1). Since TIP has inferred T2$\notin [318, 449]$, therefore T2$\in [449, \infty]$. TIP then tried unsuccessfully to rule out $[449, 633]$ and $[449, 510]$, successfully ruled out $[449, 469]$, etc., trying smaller and smaller pieces, eventually inferring $T2 \in [489, \infty]$. The final results are shown in table 5.2, row *d*. Note that TIP did not succeed in reducing $highBound(T1)$ from $\infty$ although it tried ruling out some sub-intervals $[x, \infty]$ with large $x$. Since TIP did not succeed in reducing $highBound(T1)$ from $\infty$ there is still no gap between $T1$ and $T2$.

Since TIP produced no gap, the other technique for creating a gap, *behavior splitting*, is used. Behavior splitting (chapter 3) involves "splitting" a behavior into two qualitatively identical, numerically different, independently represented behaviors, and processing each of them (section 3.2.1). In this example, $T1 = [318, \infty]$ was split into the two cases $[318, 10^6]$ and $[10^6, \infty]$ (where $10^6$ is some arbitrary high number), because infinite interval bounds often have a deleterious effect on quantitative inference. Propagating the effects of the new interval for $T1$ in each of the two behaviors resulted in row *e* for the $[318, 10^6]$ behavior.

There is still no gap between *T1* and *T2*, and there are three possible choices at this point: Try TIP again, behavior split on *T1* or on *T2* so as to force a gap, or continue time step refinement. TIP had been done previously and behavior splitting would create yet more behaviors. On the other hand, step size refinement is the mainstay of Q3 and should take priority when in doubt. Further, if step size refinement doesn't help then TIP or behavior splitting can always be done later. However, once behavior splitting has been done the extra behaviors make further step size refinement expensive since it must be done on all of them. Thus, the user chose further step size refinement and an expert simulation manager should do the same.

After two new time points were interpolated into the $[10^6, \infty]$ behavior (not shown) it was found inconsistent. So we can conclude that $T1$ must be in $[318, 10^6]$, row *e*.[4]

Having reached a simulation subgoal of ruling out a behavior, the simulation needs to be reevaluated to decide what to do next. Since the high bound of $T1$, which had been $10^6$, was dropping, there was no reason not to continue with step size refinement for a while. After two more interpolations

[4]Actually, we only conclude that *if* the behavior is possible, then $T1 \in [318, 10^6]$.

| $Variable$ | Acceleration | | Velocity | | Height | | Time | |
|---|---|---|---|---|---|---|---|---|
| $time \rightarrow$ | 153 | T1 | 153 | T2 | 153 | T1 | T1 | T2 |
| $Stage \downarrow$ | | | | | | | | |
| [a] After Q2 | | [−9.83, 0) | | (−∞, 0) | | (0, ∞) | [305, ∞) | [305, ∞) |
| [b] After 1 interp. | [−9.16, −8.55] | [−9.16, 0) | [1496, 1691] | ,, | [229, 459] | [229, ∞) | [316, ∞) | [316, ∞) |
| [c] After 2 interp. | ,, | [−8.99, 0) | ,, | ,, | ,, | [290, ∞) | [318, ∞) | [318, ∞) |
| [d] After TIP | ,, | ,, | ,, | ,, | ,, | ,, | ,, | [489, ∞) |
| [e] After beh. split | ,, | [−8.99, −.00038] | ,, | ,, | ,, | [290, 1.02e6] | [318, 1e6] | ,, |
| [f] After 4 interp. | [−8.99, −8.69] | [−8.77, −8.16] | [1535, 1635] | (−∞, −1129] | [289, 405] | [373, 623] | [326, 351] | ,, |
| [g] After 6 interp. | ,, | [−8.73, −8.19] | ,, | [−11142, −1147] | ,, | [389, 607] | [327, 349] | [489, 1469] |
| [h] After 17 interp. | [−8.91, −8.76] | [−8.62, −8.31] | [1558, 1608] | [−3993, −2255] | [319, 377] | [432, 557] | [331, 344] | [610, 759] |

Table 5.2: The returning rocket simulation at various stages in its refinement. Each technique available to Q3 has been used at some point in the simulation, as dictated by events occuring during the simulation. Model variable values are shown at times $time = 153$ (which happens to be the first time point interpolated by the step size refinement algorithm), `T1` when the rocket is at its highest point, and/or `T2` when it hits the ground on the way down.

---

between $T0$ and $T1$ (making a total of four), marked improvement occurred, row *f*. Now $highBound(T1) < lowBound(T2)$, and there is a gap between $T1$ and $T2$ where more time points can go. In fact this is now the largest gap, so the next time point, $t = 420$, was put into that gap. With more interpolations, row *g*, inferences continued to improve.

Q3 was terminated by the user after a total of 17 interpolations in row *h* while still a relatively coarse simulation. At that point the distribution of gap sizes ranged from 20 to 41 (figure 5.1). When to terminate is not predetermined by Q3 because step size refinement can continue forever, gradually producing better and better results (figure 5.2). However, a user or expert system can easily impose running time constraints. Another approach would be to continue until machine arithmetic round-off error causes further step size refinement to fail to lead to improvement, but this could take awhile.

The simulation at this stage is shown in figure 5.1.

## 5.3 Correctness, Convergence, and Stability

Intuitively, *correctness* means that interval inferences are guaranteed to contain the values that would be exhibited by any system conforming to the model. *Convergence* means that with continued refinement of the simulation, the inferences are guaranteed to become narrower until they approach fully specified points instead of intervals, given that model parameters and initial conditions are fully specified. When the modeled system is not fully specified we are interested in *stability*, which says that interval inferences will not blow up immediately in width but will degrade gracefully as the system specifications become weaker. Let us look at correctness, convergence and stability in more detail.

### 5.3.1 Correctness

The Q3 program uses *Waltz filtering* [Waltz 75] on interval labels, as in chapter 2 for interval inference, a method discussed by Davis [1987]. Waltz filtering has been proven correct [Cooper 89; Berleant 90]. Therefore the results provided by *Q3* are correct, meaning:

> *An inferred interval for any model variable at any point in the simulation contains all possible values that could be exhibited by a system conforming to the model and its quantitative specifications.*

Of course, correctness of Waltz filtering in any but the most formal sense requires that the individual constraints be semantically correct. The arithmetic constraints used by Q3 are taken directly from the standard interval extensions of common operations like $+$, $-$, $\times$ and $\div$ [Moore 79; Alefeld and Herzberger 83]. The careful reader may, however, wonder about the semantic correctness of the mean value constraint (equation 2.2). While the RHS is the obvious interval extension of the mean value theorem (equation 2.1), the LHS requires establishing that $rate(t^*) \in RATE(T_n) \sqcup RATE(T_{n+1})$, for $t_n \leq t^* \leq t_{n+1}$. This holds because:

1. It is given that model variables can change only monotonically (if at all) between adjacent time points in a QSIM qualitative simulation.

2. Therefore monotonicity continues to hold between adjacent time points even when one or both were interpolated into the monotonic region by step size refinement.

3. It is given that mean value constraint is applied only to adjacent time points in a simulation.

4. Therefore
   $min(rate(t_n), rate(t_{n+1})) \leq rate(t^*) \leq max(rate(t_n), rate(t_{n+1}))$.

5. Extending to intervals,

$$\begin{aligned} & min(loBound(RATE(T_n)), loBound(RATE(T_{n+1}))) \\ \leq \quad & rate(t^*) \\ \leq \quad & max(hiBound(RATE(T_n)), hiBound(RATE(T_{n+1}))) \end{aligned}$$

6. Therefore $rate(t^*) \in RATE(T_n) \sqcup RATE(T_{n+1})$ (recall the definition of $\sqcup$ from equation 2.2) and the mean value constraint therefore makes semantic sense.

Correctness is important for numerical methods, such as interval analysis, whose results are intended to *contain* the actual answers rather than *estimate* them as in the more common numerical methods [Eijgenraam 1981:65]. In general, inferred intervals may also contain other values. In other words, the inferences are likely to be weaker than the theoretical ideal. Finding the theoretically narrowest intervals is non-trivial, although expensive and rather convoluted algorithms such as those of Kolev et al. [1988] exist which claim to approach the theoretical ideal.

This correctness claim assumes that arithmetic calculations are completely accurate. However, machine arithmetic in general introduces small inaccuracies. Q3 deals with this by incrementing the high bound of calculated intervals by a small percentage, and decrementing a small percentage from the low bound. Provided this amount is greater than the inaccuracies introduced by the machine calculations (footnote 2), correctness is maintained, although theoretical convergence is eliminated as for any numerical simulation method when subject to round-off. The simplicity of this method for maintaining correctness derives from the fact that interval arithmetic became an active area of mathematical research precisely because it was observed to be well suited to dealing with machine arithmetic error [Moore 66].

**Termination.** Settling of a Waltz filtered network is guaranteed when the label sets containing candidate values have a finite number of elements. However, in the case of intervals or other label sets containing an infinite number of elements, termination may not occur. Settling of such a network may approach a steady state asymptotically, or worse, one or a few nodes may settle asymptotically while others starve, i.e. are queued for filtering after the asymptotically settling ones and so are never serviced. Thus termination may in theory never

occur [Davis 87]. Theoretically, termination is ensured by the use of machine arithmetic because the graininess means only a finite number of values are representable. Use of rational numbers, supported by Common Lisp [Steele 90], has been found occasionally useful for testing but does not ensure termination. The policy of narrowing intervals only if they can be narrowed by a percentage greater than some suitable $\epsilon$ can ensure termination in reasonable time [Davis 87] in either machine or rational arithmetic[5].

### 5.3.2 Convergence

There are two aspects to convergence. One is theoretical and assumes machine arithmetic is perfect. The other deals with the round-off error inherent in real computers.

**Theoretical convergence** in the interval domain is analogous to the convergence concept familiar in discussions of numerical simulation [Gear 71]. In the interval domain, convergence means fully precise predictions, that is, intervals of zero width, in the limit as the maximum step size approaches zero, given precise initial conditions and model parameters and ignoring round-off effects. While real machines are subject to round-off which prevents complete convergence, convergence is nevertheless an important tool in analyzing simulation techniques.

The analysis here is based on traditional proofs of convergence for numerical simulation methods such as Euler's method [Gear 71], also found in basic texts on ODEs such as Derrick and Grossman [1978]. The theorem of appendix B tells us that

$$w(Y_n) \leq K_1 w(Y_0) + K_2 h \tag{5.1}$$

where $w(Y_0)$ represents the *width* of the interval vector describing the initial conditions, defined as the width of the widest interval component of the vector [Moore 79]. $w(Y_n)$ is the width of the interval representing the uncertainty in $y$ after $n$ steps of simulation. $K_1$ and $K_2$ are constants and the maximum step size is $h$. The system undergoing simulation must satisfy a Lipschitz condition, which holds for almost all continuous systems in the

[5] When a high bound is decreasing asymptotically to zero or a low bound increasing to $\infty$, termination will occur due to arithmetic over- or under-flow. This has only happened in models specifically designed to create the situation.

real world and is a normal assumption in convergence proofs for numerical simulation.

Given precise initial conditions so that $w(Y_0) = 0$, for any fixed time $t = nh$ we have that $w(Y_n) \to 0$ as $h \to 0$. Thus we have *convergence*, assuming that step size refinement can reduce the maximum step size arbitrarily close to 0. In Q3, satisfying this assumption means finding an appropriate gap, or creating one using supplemental methods from section 5.1.3. In particular, convergence can occur without use of supplemental methods to help create gaps in the following situations:

- Whenever the behavior is described with a single qualitative time interval bounded by points $T0 = 0$ and $T1 = \infty$, as in figure 5.2. Then, convergence is possible for every finite time;
- Whenever $loBound(T1) = t_l > 0$. Then, convergence is possible at least up to $T_l$, and further if $loBound(T1)$ increases as the simulation is refined.

An example of a converging simulation is shown in figure 5.2. The figure plots $x$ for the equation

$$\frac{dx}{dt} = \frac{x+1}{x}$$

The predictions become much narrower as additional time points are interpolated, illustrating the process that would result in full convergence to precise values given infinitely small step size and perfect machine arithmetic. This particular system suffers from excess width problems. For example,

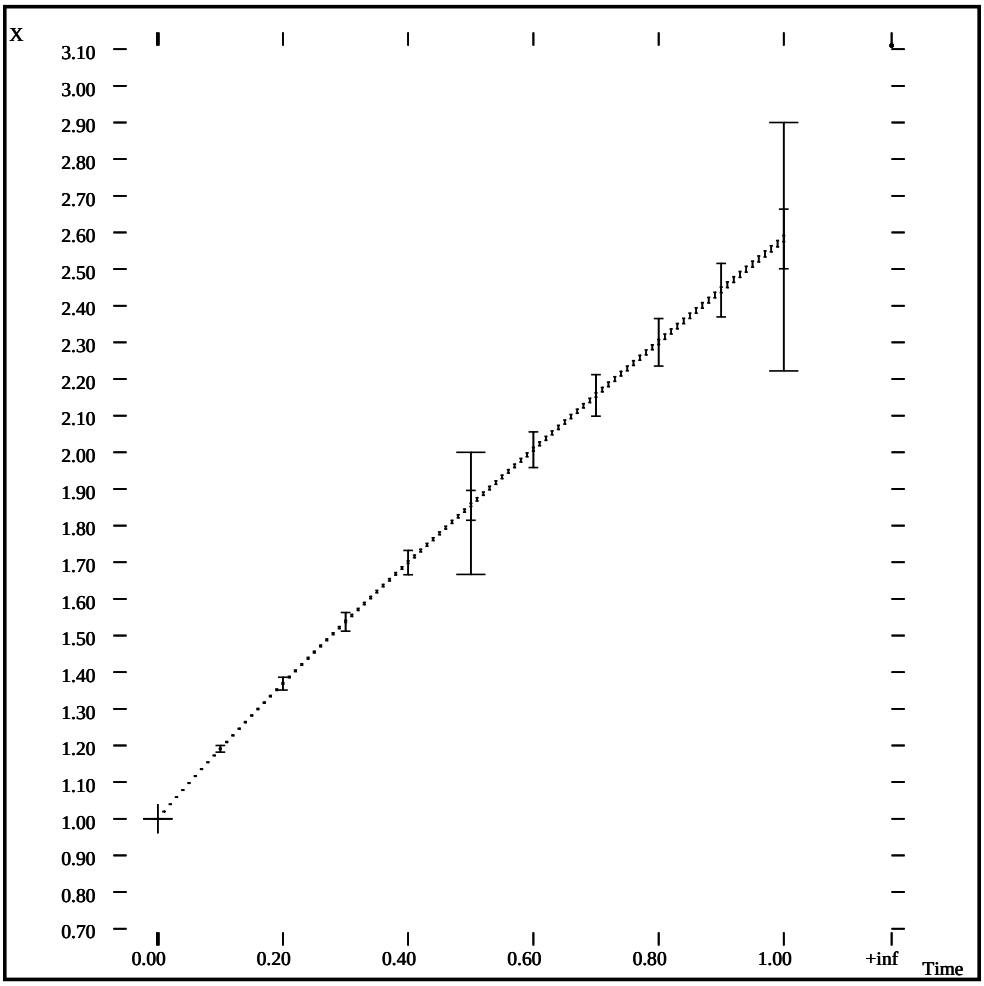
$$\textbf{if } x \in [1,2] \textbf{ then } \frac{x+1}{x} = [\frac{3}{2}, 2]$$

but straightforward calculation, doing the division operation by hand or in Q3, gives

$$\frac{[1,2]+1}{[1,2]} = [1,3]$$

Thus this system exemplifies the fact that convergence occurs even in systems suffering from excess width in individual calcuations (see appendix B).

**Round-off error.** In simulation, round-off error can build up from one time step to the next leading to accumulated error called *discretization error*. This error is critically dependent on step size in numerical and interval simulators, where it is minimized by minimizing the number of arithmetic operations, which




$$\frac{dx}{dt} = \frac{x+1}{x},\ x(0) = 1$$

Figure 5.2: Convergence occurs despite excess width in interval calculations. A simulation for $\frac{dx}{dt} = \frac{x+1}{x}$ reveals a slightly concave down curve at three different amounts of step size refinement. When just two new time points are inserted (at $t = .5$ and $t = 1$) the uncertainty in $x$ increases rapidly, as shown by the two tall ranges. After refinement with ten time points, the trajectory is significantly more constrained, as shown by the ten range indicators of intermediate height. Refinement with 100 time points leads to much more constrained results as shown by the 100 small range indicators.

means fewer (hence larger) steps in a simulation. Too small a step size, in the world of real computers, often means poorer results. In interval simulation poorer results means wider intervals [Eijgenraam 1981:136] if correctness is maintained. Thus there is a tradeoff between the convergence requirement of decreasing step size and the resultant increased discretization error. In step size refinement, discretization error cannot lead to results getting worse with continued refinement, because inferences can narrow existing intervals but never widen them.

Thus while too fine a discretization in Q3 will be a waste of effort, unlike for most simulators the results will not become worse. Indeed, when Q3's results are no longer improved by interpolating a state, this indicates that further refinement should not be attempted, an easily automated termination criterion.

The foregoing discussion applies when initial conditions and model specifications are precise. But what if they are not? Then *stability* becomes an issue.

### 5.3.3 Stability

For numerical simulation, stability refers to the desired property that small variations in initial conditions lead to bounded variations in the predictions of the simulation method [Gear 71:9]. This extends naturally to the interval domain by a concept of stability that requires initial conditions described by bounds to lead to predictions that do not "blow up" in width due to the properties of the simulation method. We now make a more formal argument for a reasonable notion of stability in interval simulation.

Gear [1971:56] defines a large class of simulation method as *stable* if results satisfy

$$\| \mathbf{y}_n - \tilde{\mathbf{y}}_n \| \leq K \| \mathbf{y}_0 - \tilde{\mathbf{y}}_0 \| \tag{5.2}$$

where $\mathbf{y}_0$ and $\tilde{\mathbf{y}}_0$ are two sets of initial conditions, $\mathbf{y}_n$ and $\tilde{\mathbf{y}}_n$ are the corresponding results of numerical simulation after $n$ steps, $\|$ is the *norm* operator which here is a vector generalization of absolute value, and the numerical simulator is governed by a set of Lipschitz differential equations containing the variables in $\mathbf{y}$. This must hold for all positive step sizes less than some $h_0$ and for some constant $K$.

This notion of stability is adapted to the domain of interval simulators by replacing the idea of the *difference between two solutions* with the idea of a *single interval valued solution with a width.* This may seem at first like little more than a change in notation, but keep in mind that the interval is correct,

i.e. guarantees that the behavior of any system satisfying the initial interval conditions will be within the intervals predicted at later times. The difference of two solutions stability concept provides no such correctness guarantee[6].

Thus, a reasonable stability criterion is $\mathbf{w(Y_n)} \leq \mathbf{K_1 w(Y_0)}$. This stability property becomes true as $h \rightarrow 0$ (theorem 1, appendix B), generalized to interval vectors as outlined in note 1. Therefore, we utilize the concept of $h \rightarrow 0$ *stability*, described by Henrici [1962] and Young and Gregory [1972], and named this by Young [1978]. Thus, given imprecise initial conditions, Q3 has $h \rightarrow 0$ stability.[7]

The pragmatic result of this stability property is two fold.

1. Simulation results benefit from step size refinement even when initial conditions and model parameters are imprecisely specified as intervals. For example figure 1.1 illustrates how reasonable inferences may be made after step size refinement reduces $h$ sufficiently, despite incompletely specified initial conditions.

2. More precise initial conditions lead to better inferences of values at later times.

## 5.4 Discussion

In developing and demonstrating Q3, the mean value constraint has been used to propagate quantitative information between temporally adjacent states. This was done because of its relative clarity and simplicity. However there are other methods that could be used. Methods more efficient than the mean value constraint are planned for when Q3 is used in an application that demands greater efficiency, and some are reviewed below.

Another area for further development is generalizing step size refinement to handle variables other than time. In fact it is as easy to interpolate a

[6]For example, both numerical solutions could underestimate the value of some model variables leading to a genuine solution $\mathbf{y}(t_n)$ not between $\mathbf{y}_n$ and $\tilde{\mathbf{y}}_n$.

[7]This may seem like a weak stability criterion, but is reasonable because the $K_2 h$ term of theorem 1 is due to the predicted intervals containing the *discretization error* of the simulation algorithm. The discretization error of the usual methods of one-step numerical simulation, such as Runge-Kutta, can be accounted for explicitly by representing the nominal predictions $\pm$ calculated error bounds, yielding intervals. Then such methods in the general case possess no better than $h \rightarrow 0$ stability even when, ignoring error terms, their nominal solutions have stronger stability properties.

state specifying a value for some other model variable as it is for time, however the convergence properties for step size refinement of a non-time variable, and for step size refinements that interpolate some states for time and some for other model variables in the same simulation, are not well understood.

We discuss these two issues below.

### 5.4.1 Alternatives to the Mean Value Constraint

The mean value constraint used in Q3 is a generalization of Euler's method [Derrick and Grossman 78] in numerical simulation. While Euler's method is geared toward finding the next state in a simulation of a system of differential equations, the mean value constraint also works backwards from a state to the adjacent previous state. The mean value constraint also is useful for interval arguments, whereas Euler's method is not immediately useful for interval calculations (even if a slope used to extrapolate to a value $\Delta t$ later is correctly bounded by an interval, it could change to a value outside the interval before $\Delta t$ has elapsed).

There are of course many ways in standard numerical simulation for inferring the next state from previous states. This is called the integration step. Euler's method is a rather poor method from a computational efficiency standpoint, and there are better methods. One of the most commonly used is the Runge-Kutta method, and there are others. It would be useful to generalize the Runge-Kutta method for the integration step to deal with intervals and to work backwards in time as well as forward. Then this generalized Runge-Kutta method could be plugged into Q3, replacing the mean value constraint and providing a substantially more efficient qualitative-quantitative simulator.

In the interval simulation area, there are a number of approaches to inferring the next state in a simulation from the existing states. In the "classic" interval arithmetic field, Moore describes a Taylor series method [1979:94-97] more enthusiastically than other methods. An existence and convergence theorem for interval operator equations [Moore 1979:theorem 5.7] could also be used to develop an extrapolation method, and a largely unelaborated and rather cryptic reference to "any of a variety of methods" for extrapolation [Moore 1979:94] may also be of interest. Such methods are likely to be fairly easily generalizable to inference backwards in time, making them candidates for use in Q3 as a substitute for the mean value constraint.

The mean value constraint was used in the development of Q3 because it is easily understandable and works, thereby supporting the research objective of developing a qualitative-quantitative simulator capable of utilizing whatever partial quantitative information is available and satisfying desirable properties

of correctness, convergence and stability. Further research on the integration step currently performed by the mean value constraint would almost certainly lead to a more efficient version of Q3. This would be of benefit in the context of a larger application that demands it.

### 5.4.2 Step Size Refinement on Other Variables

Q3's algorithm has been explained in the context of *time*. This is the model variable usually used in simulation of ODEs. However, step sizes for other model variables may readily be refined as well. For example, the air conditioner simulation (figure 1.2) was refined with `Inside Temperature`s of 79.5 and 81.5, at times symbolically labeled `K, K0, K1` and `K2`. These times were then assigned interval values as propagation of quantitative information progressed. The reason two refinements to `Inside Temperature` led to four new times is that each new temperature could be interpolated into the simulation in two different places.

Refinements to different model variables can also occur in the same simulation: in addition to refinements to `Inside Temperature`, `TIME` was also refined with the values 100, 1000, and 10000 (figure 1.2). Multi-variable discretization is rare in simulation outside of Q3, but this is unfortunate because it can be useful in cases where a value of some non-time model variable is conceptually meaningful, or where the user prefers to have precision in the value of another model variable at the cost of having an interval value for the corresponding time. In Q3, it can also be used when there is no gap in which to interpolate a new time point but there is a gap in some other model variable.

# Chapter 6

# Applications of Q3

Simulation is most often useful as a means to an end rather than an end in itself. For tasks ranging from diagnosis and understanding to planning and design, simulation has been found to be a useful tool. This chapter illustrates some application areas in which Q3 has been tested. We show how Q3 can be used in diagnosis first, then measurement interpretation, and finally glance briefly at probabilistic analysis (presented in more detail in chapters 7 and 8), tolerance analysis and other applications.

## 6.1 Diagnosis

Q3 can approach the diagnosis tasks by reasoning at three granularities, described below. Then two examples, one utilizing a pressure regulator model, and the other a working model and three fault models of a proportional thermostat, illustrate the three types of diagnostic reasoning supported by Q3.

### 6.1.1 Reasoning with models, behaviors, and values

Here are the three levels of granularity that Q3 works with, in order from finest to coarsest, given known partial quantitative information:

1. Which *quantitative values* are plausible and which are not;
2. Which *qualitative behaviors* are plausible and which are not;
3. Which *models* are plausible and which are not.

The finest level of granularity deals with quantitative values of model variables. For the bathtub model, quantitative data in the form of envelopes, and intervals that constrain uncertainty about the tub height and the faucet flow, imply quantitative predictions (figure 2.5) about the values of the various model variables at the various time points in the simulation, resulting in a mixed quantitative/qualitative description of behavior.

When this constraint propagation process can find no consistent assignments of quantitative values to qualitative landmarks in a behavior, a qualitatively plausible behavior is shown to conflict with the quantitative knowledge. The behavior can thus be ruled out on the basis of partial quantitative knowledge, bringing us to the Q3 capability of reasoning about behaviors.

Behaviors of a model are consistent or not with the incomplete quantitative knowledge a user has provided. We have previously shown how Q3 and its subset Q2 can infer, for a bathtub with partially blocked drain that satisfies the specifications of certain quantitative intervals, that the equilibrium behaviors are inconsistent and only the overflow behavior could occur (chapter 2). When all predicted behaviors of a given model are contradicted, the model is refuted, bringing us to how Q3 can reason with models.

Reasoning with models is the level with the greatest granularity. For the bathtub system, different models might include bathtubs with completely blocked drains, ones with rusted out bottoms, ones with partially blocked drains, and upside-down bathtubs. A model is ruled out if all of its behaviors can be ruled out. Ruling out models such as fault models can be useful in differential diagnosis. Figure 6.1 illustrates reasoning about models and behaviors in Q3.

### 6.1.2 The pressure regulator: differential diagnosis based on quantitative predictions

We will use the pressure regulator to exemplify diagnosis based on different quantitative predictions for a working vs. a stuck regulator.

A working pressure regulator of the type described by de Kleer and Brown [1985] (figure 6.2) has a fluid input of varying pressure. The regulator has an internal, continuously variable valve which regulates the pressure at the output port so that its variation is considerably smaller than the variation at the input port. It regulates by sensing the output pressure and closing the valve to restrict the flow as the output pressure rises, or opening the valve when the output pressure falls. A graphical description of the model is shown in figure 6.3, with the model definition code in figure 6.4.

The pressure regulator may malfunction by having the internal valve stuck in one position, so that it cannot control the output pressure. For both the working and stuck pressure regulators, an increase in input pressure leads to an increase in output pressure, but for the working regulator the increase is significantly less. Qualitative simulation indicates correctly that an increased input pressure implies an increased output pressure, for both the working and stuck regulators (figure 6.5). Thus, the qualitative description alone cannot

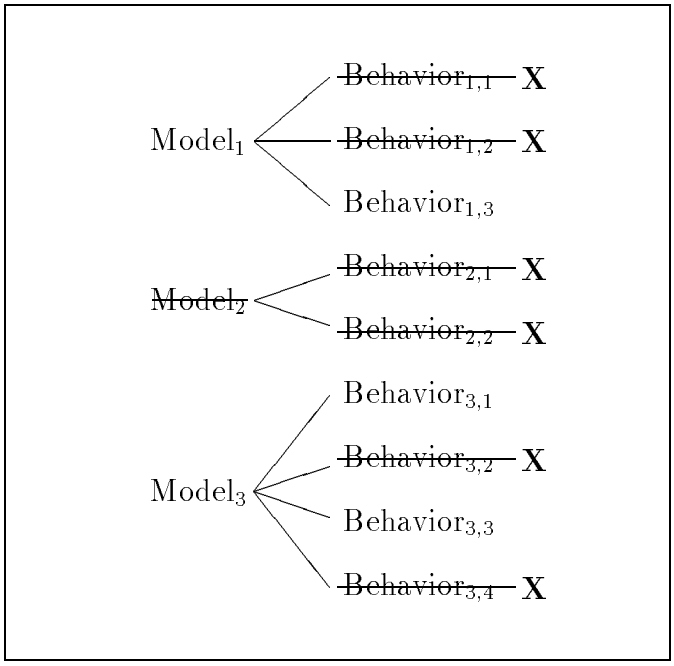



Figure 6.1: In addition to reasoning about quantitative values, Q3 can reason about behaviors and models from the available quantitative information.

Figure 6.2: **Pressure regulator device.** Fluid enters at left with input pressure $P_{in}$, passes through the valve with area $V_{area}$, and exits at right with output pressure $P_{out}$. $P_{out}$ is transmitted to the chamber at the top, where it pushes down on the valve, whose spring pushes back with force $F_s$. The higher $P_{out}$ is, the greater the equilibrium value of $F_s$ and consequently the more restrictive the valve. This feedback tends to moderate the transmission of variations in $P_{in}$ to $P_{out}$.

---

| | | |
|---|---|---|
| Pin | = | Pressure in |
| Pout | = | Pressure out |
| R | = | Resistance of regulator |
| Pdrop | = | Pressure drop across regulator |
| Q | = | Flow through regulator |
| Fs | = | Force of spring |
| Rtotal | = | R + Rload |
| Varea | = | Area of valve opening |
| Rload | = | Resistance outside regulator |
| dx | = | Speed of valve movement |

Figure 6.3: A diagram depicting the pressure regulator model of figure 6.4. Underlined model variables are held constant (Rload is the resistance of some load attached between Pout and the sump, defined as having $pressure = 0$).

```
;;;===========A MODEL OF A PRESSURE REGULATOR==========
(define-QDE Working-Pressure-Regulator
  (text "Pressure Regulator -- working correctly")
  (quantity-spaces   (Pin     (0    Pin-    Pin*  Pin+ inf))
                     (Rload   (0    Rload*  inf          ))
                     (Rtotal  (0    Rtotal* inf          ))
                     (Pout    (0    Pout-   Pout* inf    ))
                     (Pout    (0    Pout*   inf          ))
                     (Pdrop   (0    Pdrop*  inf          ))
                     (Q       (minf 0       Q*    inf    ))
                     (R       (0    Rmin    R*    inf    ))
                     (Varea   (0    0+      V*    Vm     ))
                     (Fs      (0    Fs*     Fmax         ))
                     (dX      (minf 0       inf          )))
  (constraints
     ((add  R      Rload Rtotal)  (R* Rload* Rtotal*)        )
     ((mult Rload  Q     Pout  )  (Rload*  Q* Pout*)         )
     ((mult Rtotal Q     Pin   )  (Rtotal* Q* Pin*)          )
     ((add  Pdrop  Pout  Pin   )  (Pdrop* Pout* Pin*) (0 0 0))
     ((mult R      Q     Pdrop )  (R* Q* Pdrop*)             )
     ((m-   Varea  R           )  (V* R*)  (Vm Rmin) (0+ inf))
     ((m-   Varea  Fs          )  (0 Fmax) (V* Fs*)  (Vm 0)  )
     ((add  dX     Pout  Fs    )  (0 Pout* Fs*)              )
     ((d/dt Varea  dX          )                             ))
  (independent Pin Rload)
  (print-names       (Pin    "Input pressure"       )
                     (Rload  "Resistance of load"   )
                     (Pout   "Output pressure"      )
                     (Pdrop  "Pressure drop"        )
                     (Q      "Flow rate"            )
                     (R      "Regulator resistance" )
                     (Rtotal "Total resistance"     )
                     (Varea  "Area of valve opening")
                     (Fs     "Force from spring"    )
                     (dX     "Valve motion"         ))
  (layout (Pin    Pout Pdrop)   ; default format of
          (Rload  dx   Q    )   ; simulation output
          (Varea  Fs   R    )   ; display.
          (Rtotal time nil  )))
```

Figure 6.4: Code defining the qualitative differential equation (QDE) for the pressure regulator model.

easily distinguish between the working and stuck regulators on the basis of quiescent output pressure (or any other easily measured model variable). This type of problem may arise with any proportionally controlled, negative feedback system, an important class of real mechanisms.

Quantitative information is necessary to resolve this ambiguity, but we wish to preserve our ability to reason reliably with incomplete knowledge of the quantitative specifications characterizing the physical system. Qualitative-quantitative simulation using Q3 makes it possible to assert incomplete quantitative knowledge in the form of intervals[1] about landmark values in Kuipers' [1986] QSIM behavioral descriptions, and infer their consequences. The method is applicable to other qualitative reasoning systems with limitations discussed in section 2.3.

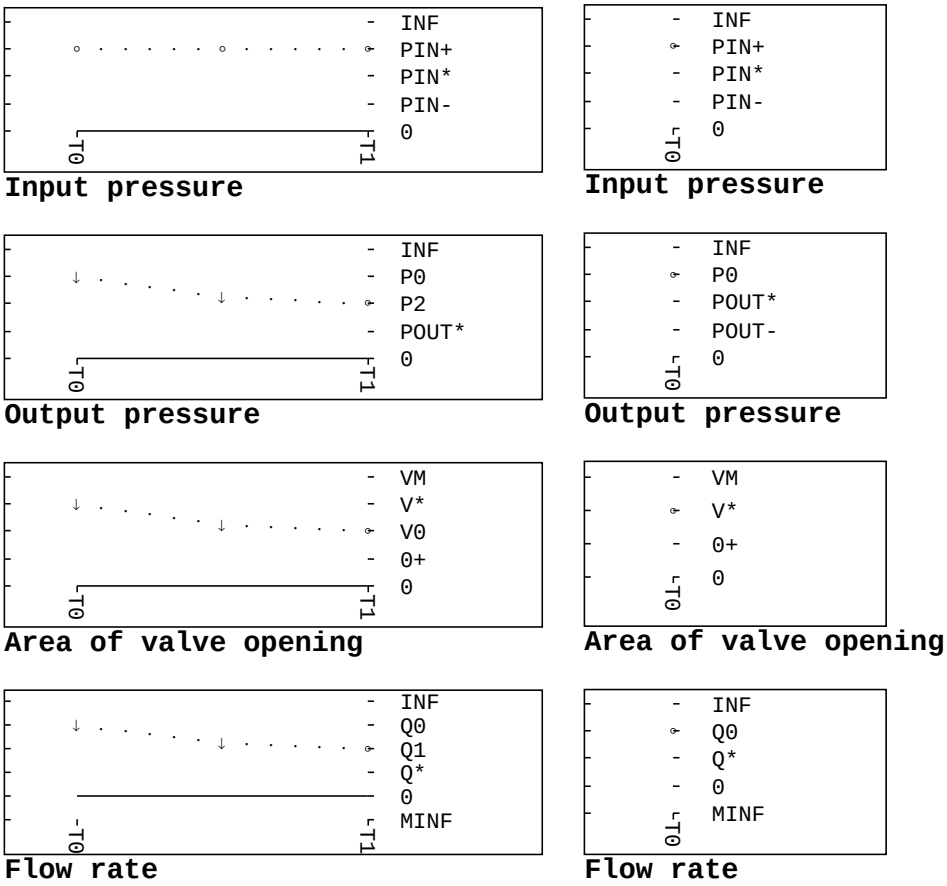
In the case of the pressure regulator, we assumed plausible intervals representing the possible values for resistance and flow capacity, and simulated the response of the regulator to a doubling of the input pressure from $[5, 5.1]$ to $[10, 10.2]$. Each of the two models (working and stuck) predicted a single qualitative behavior: output pressure increased. However, the working model predicted the final value of the output pressure to be in $[1.91, 2.98]$, while the stuck model predicted an output pressure in $[3.83, 6.24]$ (figure 6.6). This is precisely what is required for differential diagnosis between the two models.

### 6.1.3 The proportional thermostat: a more complex example

The next diagnosis problem illustrates Q3's ability to reason with both behaviors and models.

Consider the more complex example of a one-tank equilibrium system with a proportional controller attempting to keep an abstract *amount* quantity near a desired point by modifying an abstract $inflow$. One example of such a system is a heating system where the rate of heat inflow from the heater is proportional to the difference between the actual and desired temperature of the heated vessel. (The usual household thermostat does ON-OFF control, not proportional control.) Linear proportionality is only a special case of such a controller: in general, restoring force may be a monotonic function of the measured error. Proportionally controlled systems are very common in the world, including physiological mechanisms, chemical systems, automobile cruise control, etc.

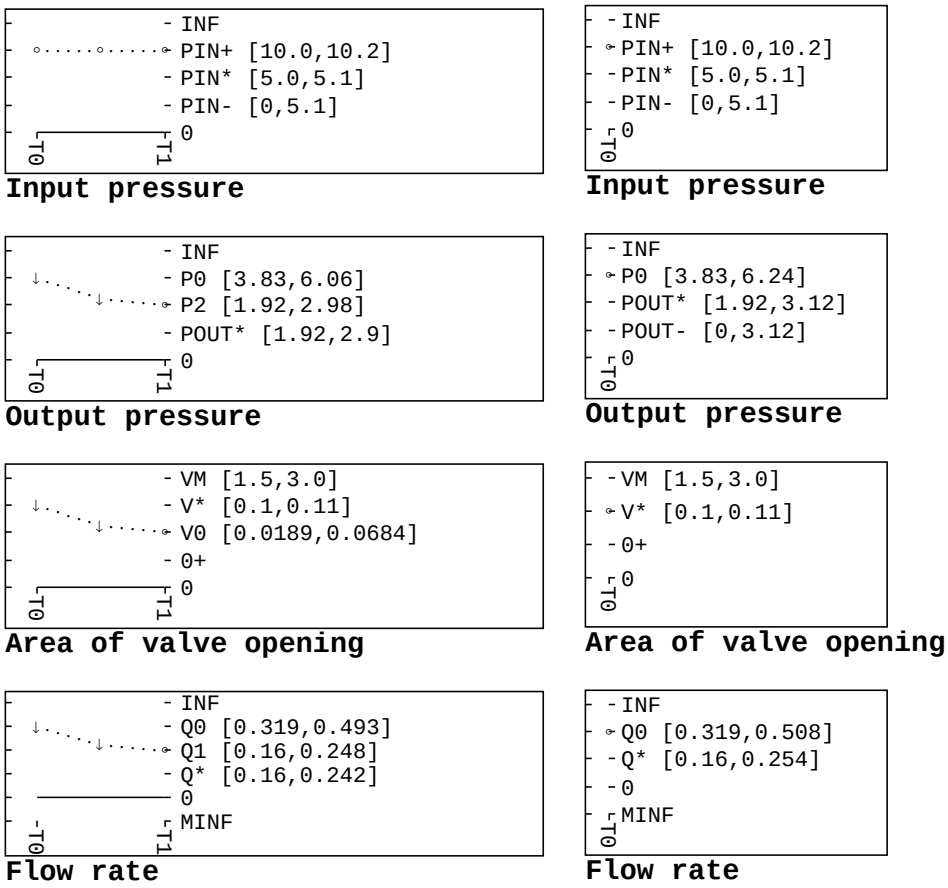
[1] Fully specified quantitative values are expressed as intervals whose endpoints are identical.




(a) working regulator

(b) stuck regulator

Figure 6.5: Qualitative simulations of working and stuck pressure regulators. The correspondences of the model constraints imply that, given RLOAD* (constant, not shown), then PIN* implies (indirectly) an output pressure of POUT*. Subjected to higher input pressure of PIN+, both regulators produce output pressure higher than POUT*. Thus it would be hard to use the qualitative predictions to match a real regulator with the working or stuck model without observing transient behavior, which may be difficult in practice.




(a) working regulator

(b) stuck regulator

Figure 6.6: The simulations of figure 6.5 augmented with quantitative processing. The final steady state for each regulator provides a clear, easily observable distinction between the two models in their predictions for final output pressure. Thus there is no need to measure transient properties of system behavior to make a differential diagnosis. (Predictions for `P0`, `Q0` and `Q*` are slightly narrower in (a) because the behavior contains more constraints, resulting in better, but overlapping and thus consistent intervals.)

We created four distinct models for a hypothetical proportionally controlled heating system:

1. The properly working system;
2. Continuous maximum heating, regardless of temperature (thermostatic control inoperative);
3. No heating at all, regardless of temperature (furnace broken);
4. Thermostat with faulty calibration, which acts as though the temperature is higher or lower than it really is and therefore causes an equilibrium temperature different from the thermostat setting.

Most models have more than one possible qualitative behavior. For example, a properly working temperature controlling system may respond successfully to a demand for increased heating, or it may "max out" by delivering heat steadily at its maximum capacity despite increasing demand. When given a particular set of *a priori* quantitative knowledge and quantitative observations, Q3 generated eighteen qualitatively possible behaviors from these four models, and used the quantitative knowledge to eliminate all but two of them. Examination of the two remaining behaviors revealed that they make identical predictions, because the fault model accounting for the uncalibrated thermostat includes the behaviors of the properly working thermostat as special cases.

## 6.2 Measurement interpretation

Time point interpolation (chapters 4 and 5) extends naturally to support measurement interpretation. We show how using the rocket example of figure 1.1.

### 6.2.1 The rocket revisited

Suppose we measured the height of the rocket of figure 1.1 to be $[12000, 12500]$ $km$ at time 3375. Clearly a state could be interpolated between the states for times `3200` and `3400` using the state interpolation method of chapters 4 and 5. This state could be further specified with the measured value for height. This would constrain the heights of the neighboring states. For example, at $t = 3200$ the height is narrowed from $[10833, 58745]km$ (figure 1.1) to $[10833, 12500]$, since clearly the height at $t = 3200$ must be less than the high bound of the measured height at $t = 3375$. The effects of the measurement

| $Variable \rightarrow$ <br> $time \rightarrow$ <br> $Measurement \downarrow$ | Acceleration | | |
|---|---|---|---|
| | 400 | 3000 | T1 |
| none | $[-4.59, -1.97]$ | $[-1.37, -0.105]$ | $[-1.34, 0)$ |
| height $\in$[12000, 12500] at time=3800 | $[-4.59, -3.34]$ | $[-1.37, -1.14]$ | $[-1.18, 0)$ |
| height=12000 at time=3800 | $[-4.59, -3.67]$ | $[-1.37, -1.2]$ | $[-1.18, 0)$ |

| $Variable \rightarrow$ <br> $time \rightarrow$ <br> $Measurement \downarrow$ | Velocity | | Height | |
|---|---|---|---|---|
| | 400 | 3000 | 400 | T1 |
| none | [6770, 18867] | [906, 17870] | [2950, 7862] | [10924, $\infty$) |
| height $\in$[12000, 12500] at time=3800 | [6770, 18463] | [906, 14295] | [2950, 4558] | [12025, $\infty$) |
| height=12000 at time=3800 | [6770, 18293] | [906, 13844] | [2950, 4058] | [12025, $\infty$) |

Table 6.1: Effects of two different measurements. The figures in the top line are from figure 1.1 and reflect no measurement, the weakest possible measurement condition. The middle line shows the effects of refining the simulation with an incompletely specified measurement. The last line reflects a strong measurement. Note how the predicted intervals tend to get narrower as the simulation is refined with stronger and stronger measurement conditions.

---

are propagated by Waltz filtering, leading to better predictions for other model variables and other time points. The results of an experiment are summarized in table 6.1.

### 6.2.2 Previous work on measurement interpretation

The total envisionment of a given situation can be regarded as a finite-state transition graph, which is used to "parse" the stream of Ds values from

an acceptable initial state to an acceptable final state. The path successfully taken through the envisionment describes the sequence of process structures the system goes through. Failure to parse presumably refutes the model.

Previous work on measurement interpretation in qualitative reasoning has been reported most notably by DeCoste [1990] and Forbus [1986]. They abstract measurements into qualitative values, thus expressing measurements in the same coarse qualitative terms as those used in the model and thus throwing away some of the numerical information. They then use those qualitative values to infer the state of a qualitative simulation. In contrast, the method described here utilizes the actual measured numbers, but requires the location in the simulation where the measured state will be inserted to be unambbiguous. Thus the two methods are complementary.

## 6.3 Some other applications

Tolerance analysis, sensitivity analysis and numerical simulation applications are likely directions for future Q3 application domains. The use of partial quantitative information in the form of probability distributions is an application area investigated in later chapters.

### 6.3.1 Tolerance analysis

In engineering, components are often specified using tolerances, a form of partial quantitative, often interval, information. A model in which component tolerances are encoded as intervals could be simulated to see what behaviors and numeric values might occur. Simply simulating numerically many times with different combinations of plausible component values, as in Monte Carlo analysis, is likely to be tedious and could miss unlikely but potentially serious misbehaviors: frequently, failure modes have low probability, yet high importance.

### 6.3.2 Probability distribution functions (pdfs)

Pdfs are a way of describing partial information that is related to, but more informative, than intervals. Any pdf whose value is zero outside the endpoints of some interval may be characterized abstractly by that interval. The interval is a generalization of the pdf, and infinitely many pdfs correspond to any one interval of non-zero width (an interval of zero width, i.e. a number, would correspond to the single pdf described by a unit impulse, or Dirac $\delta$

function, which is non-zero only at a single point yet has a definite integral of precisely one over $(-\infty, \infty)$ [Derrick and Grossman 78]).

A representation method utilizing pdfs instead of intervals has been shown to enable bounding the probabilities of different qualitative behaviors in chapters 7, 8, Berleant [1990] and Berleant and Chandra [1991]. Unfortunately, pdfs are not full-spectrum: while numbers can be expressed using pdfs, qualitative values cannot. One way to use pdfs and retain the desired full-spectrum property is to create a representation that can express pdfs, intervals, and knowledge that is intermediate between pdfs and intervals. An approach to this utilizes the concept of *first order stochastic dominance*, described by Whitmore and Findlay [1978] and used by Wellman [1990].

### 6.3.3 Other potential applications

Any application of ordinary differential equation simulation can be examined as a possible area for utilizing qualitative-quantitative simulation. Existing simulation models could be translated, perhaps automatically, into the Q3 input language, their numerical specifications "blurred" by replacing each with an enclosing interval (again automatically) and then simulated by Q3. Then the benefits of qualitative-quantitative simulation such as correct interval enclosures, and guaranteed coverage of all behaviors, could be obtained.

A common problem in finance is to determine the feasibility of a goal given certain initial assumptions such as interest rates. The exact values of such input parameters often cannot be known ahead of time with desired precision, so qualitative-quantitative simulation could be beneficial.

A subset of Q3 called Q2 has been applied in the field of chemical engineering [Dalle Molle 89].

# Chapter 7

# Application: Probabilities of Qualitative Behaviors

## 7.1 Overview

Research on finding the different behaviors that an incompletely specified model can exhibit has concentrated on finding the plausible behaviors, and ruling out implausible ones. However, it would also be useful to estimate the *probabilities* of different behaviors. That is the goal of the present chapter.

We show how to make inferences about the probabilities of the various qualitative behaviors a model could exhibit, when partial quantitative information in the form of probability distributions is given about values (such as initial values) of model variables.

**Probability distribution functions (pdfs)** are an important form of partial quantitative information about values. Pdfs are more informative than intervals: an interval is an abstraction of any pdf which has the value of zero outside the bounds of the interval. Since it is beneficial to be able to use what quantitative information is available, there is a strong motivation for finding methods of using pdfs in qualitative-quantitative simulation, such as to find the probabilities of different behaviors.

The algorithm described does not make the assumptions that a model is linear or that a given probability distribution function (pdf) is of a particular class. It will be demonstrated on a 1-tank model. We start by describing the basic histogram discretization algorithm, then modify it for the purpose of finding probabilities of behaviors in a way that satisfies desirable safety properties. The result is illustrated with a 1-tank model, after which limitations and previous work will be described.

## 7.2 Histogram Discretization

Simulation of continuous systems involves arithmetic operations to infer new system states. These operations are defined by the functional dependencies of the model variables on each other. This is true for qualitative

simulation as well as for numerical simulation, and for qualitative-quantitative simulation using intervals — or pdfs, as in the current chapter. The integration step that generates a new state from previously inferred states is also implemented using arithmetic operations (even the inference of qualitative values for a new state can be viewed that way). Thus, to use pdfs we need to be able to do arithmetic operations on them.

One approach to doing operations on pdfs is analytical. However this approach becomes intractable or impossible for problems involving more than a few operations [Jackson et al. 81]. Another approach is the numerical combination of random variables using histogram approximations to their pdfs[1], what we will call the *histogram discretization method*. Each pdf is discretized using a histogram with bars defined by intervals locating them on the $x$-axis, and heights which approximate the height of the pdf over the corresponding intervals such that the area of a histogram bar equals the definite integral of the pdf over its corresponding interval. An arithmetic operation may be made on pdfs represented this way by doing the following:

1. For each term in the cartesian product of the bars of the histograms, produce a *result collection* this way:

   (a) execute the corresponding interval arithmetic operation on the elements of the term, then associate the resulting interval with:

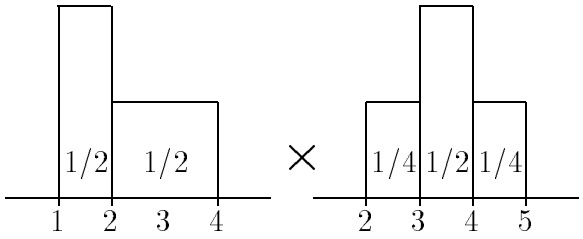
   (b) the product of the probability masses of the elements in the term (the probability mass of a histogram bar is its area).

2. The results collection (exemplified by figure 7.1) for the terms of the cartesian product must then be combined to get an overall result. One way is this:

   (a) Decide on a set of disjoint intervals partitioning the domain of the probability distribution of the result variable. This partitioning determines the location and widths of the bars of the histogram approximating the distribution function. The particular partition is not defined by the algorithm.

   (b) Now calculate the areas of the histogram bars delimited by the partition:

[1]The histogram may also represent the probabilistic knowledge in itself instead of being merely an approximation to it.




| | operand1 | operand2 | result collection |
|---|---|---|---|
| interval | $[1,2]$ | $[2,3]$ | $[2,6]$ |
| probability | $1/2$ | $1/4$ | $1/8$ |
| interval | $[1,2]$ | $[3,4]$ | $[3,8]$ |
| probability | $1/2$ | $1/2$ | $1/4$ |
| interval | $[1,2]$ | $[4,5]$ | $[4,10]$ |
| probability | $1/2$ | $1/4$ | $1/8$ |
| interval | $[2,4]$ | $[2,3]$ | $[4,12]$ |
| probability | $1/2$ | $1/4$ | $1/8$ |
| interval | $[2,4]$ | $[3,4]$ | $[6,16]$ |
| probability | $1/2$ | $1/2$ | $1/4$ |
| interval | $[2,4]$ | $[4,5]$ | $[8,20]$ |
| probability | $1/2$ | $1/4$ | $1/8$ |

Figure 7.1: Multiplying histogram representations of pdfs. The cartesian product of histogram discretizations for the inputs is used. The probability masses of the bars are indicated in their interiors.

———————

i. Any result collection interval (see column 3 of table, figure 7.1) that falls completely within one member of the partition has its entire probability assigned to that member.

ii. A result collection interval that overlaps more than one member of the partition has its probability divided among them, in the proportion of how much of it overlaps each partition member. For example, the first result collection interval in the table of figure 7.1 is $[2, 6]$ with probability $1/8$. If the partition contains the intervals $[2, 5]$ and $[5, 9]$ then $[2, 5]$ is assigned $3/4$ of the $1/8$ probability, or $3/24$, and $[5, 9]$ is assigned $1/4$ of it. This will be termed *proportional assignment.*

iii. All of the probabilities assigned to each member of the partition are summed to get the final estimate of the probability of each partition member.

(c) The heights of the histogram bars of the result pdf can be calculated from their widths and areas (and the histogram could then be drawn if desired).

While the example happens to be of multiplying two distributions, almost any operation or function of two variables can be used to get a result collection like that in column three of the table (figure 7.1).

So far, the algorithm is as described in Ingram et al. [1968] and Colombo and Jaarsma [1980]. A modified version approximates the bars with their midpoints and probability masses [Kaplan 81].

A number of observations may be made on the histogram discretization technique. These include:

1. The histogram discretization of a pdf could easily be described the same way as a result collection, since a result collection is simply a list of intervals and probability masses. Thus, a histogram discretized pdf is a result collection, although of course not every result collection can be described using a pdf. Most can't, and that is why there exist methods like proportional assignment to force a result collection into an approximating pdf.

2. Each result collection interval constrains a probability mass to be within some interval but does not define how it is distributed within that interval. Thus, while the result collection is sound, the conversion of it into a pdf will likely introduce some distortion of the curve due to the proportional assignment of probability masses to overlapped members of the partition

of the result pdf domain. This will be called *proportional assignment error.*

3. Even worse, result collection intervals may contain excess width, common in interval operations. The proportional assignment of a result collection interval probability mass will almost always lead to distortion if the result collection interval has excess width.

4. However, proportional assignment is not necessary. The result collection could be used as the input to another operation without first forcing it into a possibly distorted pdf.

5. Therefore, using result collections as the representation of value, without converting them into pdfs before doing further operations, eliminates proportional assignment error.

Unfortunately, another source of error can affect the soundness of the result collections themselves. For example, consider the equation $X - X = Y = 0$, with $X$ described by some pdf with probability mass $p_1$ in $[2,3]$ and probability mass $p_2$ in $[4,5]$. Doing the subtraction operation using the histogram discretization method would yield a result collection containing

$$Y \in [4,5] - [2,3] = [1,3], p = p_1 \cdot p_2$$

which is not right, since $X - X$ must in fact be 0 with $p = 1$. This source of error also applies to Kaplan [1981]. Thus:

- Observe that this source of error is due to dependence (or partial dependence) between operands. We will therefore call it *dependence error.* To avoid it, inputs to a program that uses histogram discretization should be independent — that is, uncorrelated.

- Worse, dependence error also occurs when doing operations on result collections. This becomes clearer when one considers that dependence error applies to histogram discretized pdfs, and histogram discretized pdfs are formally just a variety of result collection. Compounding this problem, variables used to store intermediate results in expression evaluation can be dependent on one another even when input variables are independent, leading to dependence error. For example consider the equation

$$\frac{dX}{dt} = Y + \frac{X}{X+1} \tag{7.1}$$

In evaluating it we combine $X$ and an intermediate variable $V = X + 1$, where $X$ and $V$ are dependent. If result collections representing $X$ and $V$ are combined, the resultant result collection for $dX/dt$ will contain this dependence distortion, even if input variables $X$ and $Y$ are independent. For example, suppose the histogram discretization for $X$ contains $[1,2], p = .5$ and $[8,9], p = .5$. Storing the result collection for $X + 1$ in a variable $V$ and then calculating $\frac{X}{V}$ from $X$ and $V$ would result in the probability estimate $p(X \in \frac{[1,2]}{[8,9]+1}) = 0.25$, clearly wrong since $X$ can't be in $[1,2]$ and $[8,9]$ simultaneously. Hence dependence distortion applies even though inputs $X$ and $Y$ may be independent, because of dependence between $X$ and $V$.

Dependence distortion may be handled by avoiding doing operations on result collections for intermediate variables. Thus for equation 7.1 a result collection for $dX/dt$ would be evaluated from the histogram discretizations of input values $X$ and $Y$ without ever representing $V$ with a result collection: each interval $i_{jk}$ in the result collection for $dX/dt$ would be calculated directly from intervals $i_j$ and $i_k$ in the discretizations of $X$ and $Y$, respectively, by evaluating

$$\forall j, k : i_{jk} = i_k + \frac{i_j}{i_j + 1} \tag{7.2}$$

In conclusion, we can use the histogram discretization method safely, without risk of dependence error or proportional assignment error by:

1. Not forcing a result collection into an approximating pdf (thus avoiding proportional assignment error);

2. Avoiding using result collections for temporary variables to calculate other result collections (because such temporary variables often introduce dependencies); and

3. Only dealing with problems for which the inputs are independent (luckily this is often the case).

   We now describe a way to use histogram discretization that satisfies these requirements.

## 7.3 Using Histogram Discretization to Find Probabilities of Qualitative Behaviors

We modified the standard histogram discretization method, as previously described, to suit the goal of finding probabilities of qualitative behaviors. We will see that proportional assignment error is eliminated Also eliminated is dependence error arising from partial dependence among variables used to store the intermediate results arising in expression evaluation — as long as inputs are independent.

The first modification is in using constraint networks to relate variables instead of functions. The use of constraint networks instead of functions is only a representational matter, because equations (and sets of equations) can be represented as constraint networks by expressing all binary operations as ternary constraints and all unary operations as binary constraints. Thus the equation

| | | | |
|---|---|---|---|
| $y'$ | = | $x'(y + x')$ | becomes the constraints |
| (D/DT | y | dydt), | |
| (D/DT | x | dxdt), | |
| (PLUS | y | dxdt | w), |
| (MULT | dxdt | w | dydt). |

Note that sometimes it is necessary to create and use new variables, such as w, whose values are the results of intermediate expressions within the equation(s) (appendix B, notes, item 2). The constraint networks are built automatically from qualitative simulations.

The second modification is in the goal. The goal is not to estimate the pdf for an output value, but to estimate the probability of an output value being within some region — a region in this case corresponding to a qualitative behavior. We will see that with this goal, we can *bound* probabilities instead of requiring point estimates. Bounding probabilities of behaviors has the important property that distortion in output pdfs due to proportional assignment can be sidestepped since we don't infer such pdfs. Instead excess width in interval calculations ends up as excess width in the probability bounds, resulting in weaker but still valid conclusions (see also section 8.4.1 on excess width). Dependence distortion is avoided by not doing operations on the result collections of intermediate variables. Instead, the whole constraint network is settled for each member of the cartesian product of the input histograms individually (see equation 7.2).

Here are three possible methods for finding probabilities of behaviors. (We will employ the last in the example.)

1. We could partition the domain of an output variable into the qualitative categories "high", "medium" and "low", and assign the probability mass of each member of the result collection to the appropriate category, proportionally assigning those that overlap more than one category. Then we get probability estimates for the qualitative behaviors "high", "medium" and "low". This simplistic method clearly suffers from proportional assignment error.

2. We could modify the previous method of item 1 to bound probabilities with *intervals* instead of point estimates prone to proportional assignment error. In this method, a result collection interval such as one of those in figure 7.1 that straddles one or more boundaries between qualitative categories (e.g. "high", "medium" and "low") does not have its probability mass $p$ pro-rated among the qualitative categories. Instead, each overlapped category is incremented by the safe probability mass interval $[0, p]$.

3. We could use qualitative behaviors generated by a qualitative simulator. We used QSIM [Kuipers 84], which encodes each such behavior as a constraint network by instantiating model constraints for each qualitatively distinct time point in the simulation. This constraint network is augmented with whatever quantitative information is available and processed as described in chapter 5. Each element of the cartesian product of the input histograms is tested for consistency with the constraint network representing each qualitative behavior. If an element is consistent with just one qualitative behavior, its probability mass is assigned to that behavior. Otherwise, the probability mass of each qualitative behavior it is consistent with is incremented by $[0, p]$ where $p$ is the probability of the cartesian product element.

Method 3 will be used to satisfy the goal of finding probabilities of qualitative behaviors while avoiding proportional assignment and dependency errors. We now explain this technique more fully in the context of an example typifying the large and often useful class of "1-tank" models.

## 7.4 The Partially Stopped Bathtub

Let us consider a bathtub with a partially blocked drain. When the faucet is turned on, water starts flowing into the bathtub. Since the drain is partially blocked, not much flows out, so the amount of water increases from its original value of zero. As the amount increases, the level of the water in the

tub increases. This causes the pressure at the drain to increase, which causes the amount of water flowing out to increase. This means that the level rises at a slower rate, since the increased pressure is causing more of the inflow from the faucet to go out the drain. Thus the level continues to rise, although at an ever-decreasing rate. Eventually something must happen. The two possible behaviors are either, 1) the bathtub overflows, or 2) the bathtub reaches some equilibrium level. If it reaches equilibrium, this means the level is such as to produce a pressure at the drain which forces water out at exactly the same rate at which it enters the bathtub from the faucet. Note that the bathtub system is qualitatively identical to many members of the larger class of 1-tank systems such as sinks, swimming pools, and industrial tanks, as well as to systems whose models are isomorphic to the bathtub but which do not involve literal fluids (e.g. leaky capacitors).

A schematic of a one-tank "bathtub" model appears in figure 2.1. It is basically self-explanatory, except for the M+ constraints. M+ designates a monotonically increasing relationship between its arguments. Thus when amount is increasing, level must also be increasing. Conversely, when level is increasing so is amount. The same is true between level and pressure. Note that *drain* and *inflow* are underlined, indicating that their values are constant throughout the simulation.

A purely qualitative simulation of the bathtub appears in figure 2.3. Both the overflow and the equilibrium behaviors were shown. Partial quantitative information was then added, in the form of intervals for inflow and drain aperture, and in the form of a space of possible functions for the relation between amount and level, and between level and pressure. These function spaces were defined by upper and lower envelope functions, so that

$$f_{upper}(x) \geq \ f(x) \geq \ f_{lower}(x)$$

for all functions $f$ in the space, and by their monotonicity, determined by the M+ specification as increasing. A qualitative simulation in which this partial quantitative information was utilized appears in figure 7.2. While predictions are made about quantitative values for various model variables via Waltz filtering with interval labels (chapter 2), both behaviors are still plausible, and there is no obvious way to judge their probabilities of occurrence.

But suppose we have a bit more knowledge about the initial conditions: perhaps we can do better than a plain interval for the inflow, instead expressing it as a probability distribution function (pdf). One approach to this would be to make the heuristic assumption that the probability density is the same everywhere inside the interval, and zero outside it. Another reasonable heuristic would be to replace the interval with a bell curve whose ends

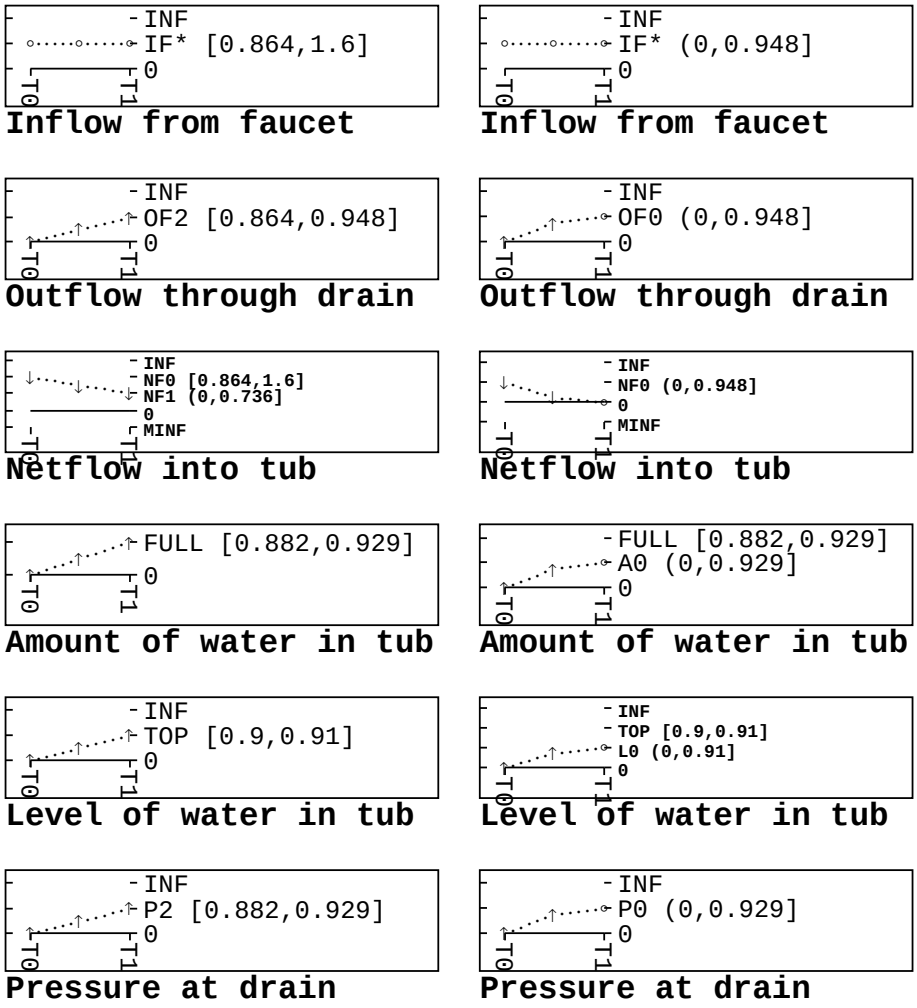



Figure 7.2: A qualitative simulation of the bathtub, with partial quantitative information added. Both behaviors are consistent with the initializing intervals. Note that while initial inflow is specified as [0, 1.6], each behavior turns out to be consistent with different subsets of that interval.

are truncated. An even better approach can be used when the pdf is actually known — just use it. All 3 of the pdfs just mentioned can be consistent with the original interval, because an interval is an abstraction of a pdf: an interval matches every pdf whose value is zero everywhere outside the interval bounds. Figure 7.3 shows the pdf used here, a triangular shape that is zero for negative inflows and inflows greater than 1.6 gallons/min., which we take as the inflow when the faucet is fully opened. We chose this pdf arbitrarily; an irregularly shaped pdf could have been used instead. We now explain how to find the probabilities of behaviors of this model from the pdf.

## 7.5 Deriving the Probabilities of Behaviors from a Pdf

The algorithm for deriving bounds of behavior probabilities, using pdfs for independent model variables is quite simple in essence. This section describes it in four stages. The next section discusses limits and directions for further research.

1. Divide the domain of the pdf into parts, each expressed as an interval and a height. Approximate the pdf with a histogram of variable width bars, each described by a height and an interval expressing the width and location of the bar on the horizontal axis.

   In figure 7.3 we modeled the **inflow** with a triangular pdf (a different pdf could have been used had we wished). We approximated this pdf using a histogram of 4 bars, whose intervals and heights were chosen such that the area under the pdf for the portion of the domain bounded by each interval is 0.25, reflected by the heights of the bars which each also have areas of 0.25. The resulting intervals are [0, 0.57], [0.57, 0.8], [0.8, 1.03], and [1.03, 1.6]. These areas were made equal for simplicity; they could have been different.

2. Initialize the model value described by the pdf with each of the intervals in turn from the previous step. For each such interval, infer its implications for the other model variable values. For this we used the *Q2* program, chapter 2. Do this for each behavior of the model.

3. We might expect that for some initializing intervals, some behaviors are plausible and some are not. For example, an inflow of 0 will be inconsistent with the overflow behavior. It is reasonable, therefore, to wonder if overflow is impossible for the lowest interval for inflow, [0, 0.57]. In fact the Q2 program finds this to be the case. Figure 7.3 shows the results. Equilibrium is plausible for inflow intervals [0, 0.57], [0.57, 0.8],

Figure 7.3: A pdf for inflow and behavior probabilities based on it.

and [0.8, 1.03]. Overflow is plausible for inflow intervals [0.8, 1.03] and [1.03, 1.6].

4. We now assign probability ranges to each behavior, based on which intervals of the histogram-expressed pdf can plausibly result in which behaviors.

   For the bathtub model and the triangular pdf for inflow, this means that the 0.25 probability of an inflow in [0, 0.57] is assigned to the equilibrium behavior, because that is the only behavior that can occur with such a low inflow. This is shown in figure 7.4. Therefore equilibrium has at least 0.25 probability, whereas p(overflow) has a *maximum* of 0.75 because overflow is impossible if the inflow exhibits its 0.25 probability of being in [0, 0.57].

   The same situation is true for the inflow interval [0.57, 0.8]: equilibrium is plausible and overflow is impossible, so the 0.25 probability of inflow being in [0.57, 0.8] goes to the equilibrium behavior, giving it a probability of at least 0.5 (and overflow a probability of no more than 0.5).

   Let us skip for now to the highest interval for inflow, [1.03, 1.6]. This interval is consistent with overflow but not with equilibrium; in other words, if the faucet is turned all the way on or almost all the way on, the bathtub will definitely overflow. Thus, $p(overflow) \geq 0.25$ and our estimate of $p(equilibrium) \geq 0.5$ remains unchanged.

   Finally we turn our attention to the inflow interval [0.8, 1.03]. From the table in figure 7.3 we see that either overflow or equilibrium could occur, so it is not clear what to do with the 0.25 probability associated with inflow being in that interval. There is more than one option available at this point.

   (a) Divide the interval [0.8, 1.03] with its probability of 0.25 into smaller intervals each with a lower probability, and see if these smaller intervals are consistent with just one behavior. We can then treat these smaller intervals as we treated the inflow intervals [0, 0.57], [0.57, 0.8], and [1.03, 1.6], assigning their probabilities to the behavior each is consistent with.

   Given only partial quantitative information about the model, however, there may always be some values for inflow that are plausible for both behaviors. This is due for example to the fact that the blockage of the drain is not fully specified but is only known within a certain interval.

Structure: Standard bathtub with open drain..
Initialization: Filling at constant rate, starting from empty (S0)
Behavior 2 of 2: (S0 S1 S3).
Final state: (GF QUIESCENT COMPLETE), NIL, NIL.

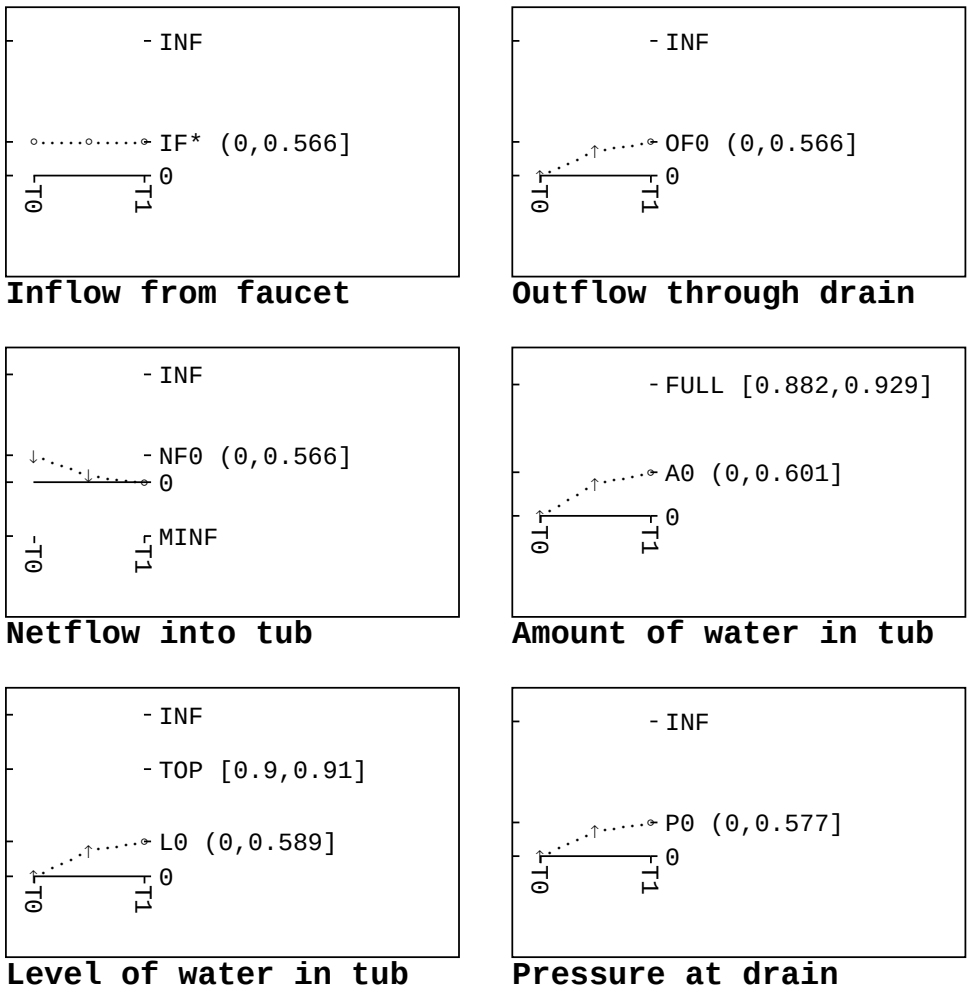



Figure 7.4: Inflow has a 0.25 chance of being in [0, 0.57]. If it is in [0, 0.57] then only the equilibrium behavior is plausible, shown here. One branch of the behavior tree in the upper right corner is marked with an "X" to indicate that the overflow behavior has been ruled out.

(b) Use the heuristic of simply dividing the probability of an interval that is plausible for more than one behavior among those plausible behaviors. We could do that for inflow in [0.8, 1.03], apportioning its 0.25 probability equally and thus contributing probabilities of 0.125 to each behavior. Thus we would arrive at an estimated probability of .625 for equilibrium and 0.375 for overflow. Such estimates can have practical value. In this case for example it would be prudent to do something to clear the drain, in order to avoid the substantial 0.375 chance of water damage from the bathtub overflowing.

(c) Add interval probability estimates of $[0, 0.25]$ to both the equilibrium and the overflow behaviors, giving final behavior probability estimates of $[0.5, 0.75]$ for equilibrium and $[0.25, 0.5]$ for overflow. This is what was done here (see figure 7.3). More generally, add $[0, p]$ to each behavior which is plausible for the input pdf being in a particular interval for which other behaviors are also plausible, where $p$ is the area of the histogram bar under consideration. The results may be used for ranking the relative likelihood of the different behaviors; e.g. for the bathtub we expect equilibrium more than we expect overflow. We see applications in engineering or economics as a good domain for applying this technique.

## 7.6 Limitations of the Algorithm

There are currently some limitations to the capabilities of the technique discussed here:

1. There may be more than one independent value described by a pdf. This could be handled by dividing each such pdf into intervals, taking the Cartesian product of the different possibilities for each pdf, and treating each member of the Cartesian product as we have treated each histogram bar in the bathtub example. In effect, what we used there was the Cartesian product of one value, the constant inflow. Combining pdfs from different independent values by expressing each as a set of histogram bars, and using their Cartesian product has been shown to be practical [Colombo and Jaarsma 80]. Unfortunately the number of elements in the Cartesian product increases exponentially with the number of pdf-described values, so there is a complexity problem that occurs in the case of models with many such values.

   One approach to this complexity problem is simply to reduce the number of intervals into which a pdf is divided. In a strong sense, the bathtub example utilizes this approach by "dividing" the drain blockage pdf,

whatever it is, into just 1 interval. Handling more than one pdf is done in chapter 8.

2. When variables are partially dependent, they can in principle be described as functions of independent variables, which could then be handled as described. Finding these functions is often non-trivial, however.

3. There may be more than one behavior plausible for each (or most of) the histogram bars. Then the probability intervals for the behaviors will be rather wide and uninformative (see section 7.5, item 4c). In such a case it would still be possible to use the midpoints of the probability intervals to order, or partially order, the behaviors by their estimated probabilities. This problem could also be solved by discretizing the pdf finely enough that most of the intervals for the input produced by the discretization are only plausible for one behavior. The drawback would be the computational complexity of dealing with each of a large number of possible input intervals.

4. While the technique works for arbitrary discretizations of a given pdf, some discretizations will give better results with less effort than others. For example, if the discretization resulted in histogram bars within which the input always produced a single qualitative behavior, there would be no behavioral ambiguity to motivate subdividing the intervals further (as explained in section 7.5, item 4a). A preliminary experiment tested an algorithm that discretizes intelligently (appendix of this chapter). That algorithm was then developed in the context of a dependability analysis example in the next chapter.

We expect the technique described in this paper to work for any situation in which a pdf can be assigned to some independent input(s) of a system which exhibits more than one behavior. The requirement is that there must be some way of determining if a given behavior is plausible for intervals on the independent input pdfs. The domain of applicability therefore is not limited to continuous qualitative models; discrete simulation models and operations research are other directions for applications development.

## 7.7 Discussion

Some previous work on estimating probabilities of behaviors has appeared in Yeh [1990], Doyle and Sacks [1989], and DeCoste [1990].

Yeh [1990] estimated probabilities of behaviors given envelopes assumed to contain all plausible probability distribution functions (pdfs) of an input. Doyle and Sacks do not discuss pdfs in their technique.

Yeh concentrates on final system *state*. In contrast, the present work deals with probabilities of qualitative *behaviors*. Behaviors may — or may not — end in limiting equilibrium states.

The approach of Doyle and Sacks [1989] is quite different from the present work. They describe the behavior of a system using a Markov chain, deriving conclusions by analyzing the chain. Their method works for systems whose dynamics can be adequately modeled as Markov chains. However, a system with values characterized by intervals or a mixture of intervals and pdfs does not lend itself easily to finding the required transition probabilities — except perhaps by applying the present work. They also have not automated the finding of transition probabilities, leaving that task to expert estimation or expert application of Monte Carlo simulation.

DeCoste [1990] utilizes the Doyle and Sacks approach, arguing that it could provide useful, although not guaranteed, estimates of behavior likelihoods given measurement data. DeCoste also utilizes probability estimates of the category of uncertain measurements. For example, a nominal measurement of 0.501 might actually be in the $low = [0, 0.5]$ range with $p = 0.1$, and $medium = (0.5, 0.8]$ with $p = 0.9$.

A discussion of Monte Carlo methods in the context of artificial intelligence is given in Yeh [1990]. Monte Carlo simulation is an important component of the Doyle and Sacks method. This means Doyle and Sacks' method will be susceptible to error due to stochastic noise, as well as the potential cost of multiple simulation runs. The probabilistic analysis approach described here, however, does not use Monte Carlo methods and is free of stochastic noise. Our technique can also significantly reduce the number of simulation runs, compared with Monte Carlo simulation, by avoiding further analysis of input value regions that have been found consistent with just one behavior. The technique does however require that external model inputs be independent of one another, which need not be a problem in Monte Carlo-based methods. Fortunately, in many problems the external inputs are in fact independent. Also, since partially dependent variables are implicitly functions of independent variables, our technique maintains at least theoretical capabilities of application with partially dependent variables as well. Finally, Monte Carlo simulation cannot handle a mixture of pdfs and intervals describing model inputs. Yet this was the situation for the 1-tank "bathtub" model used as an example, as

the inflow was a pdf but the drain aperture was described as an interval[2].

Stochastic differential equations is a topic that can arise in connection with models with uncertain inputs [Lasota and Mackey 85]. Two major differences between that and the current work are:

1. inputs to stochastic differential equations vary randomly, often due to a random "noise" term, whereas the present work assumes no noise. Uncertainty in a value description here rather reflects lack of knowledge about a value which nonetheless is definite and unvarying.

2. Outputs of stochastic differential equations are distributions, whereas in this work we are concerned with probability bounds on qualitatively distinct behaviors.

## Appendix, Chapter 7

Observe that arithmetic constraints are usually reversible, in that knowledge about any $n-1$ arguments can be used to make inferences about the $n$th. Thus if $x + y = z$, knowledge about any 2 symbols constrains the third[3].

Thus, this reversibility of constraints means we can work backwards from a constraint-represented behavior to the input intervals for which it is consistent. An input pdf can then be integrated over such a interval to estimate the probability of the behavior. By working backwards in this way, the partitioning of an input pdf is done intelligently, instead of blindly as in method (3) above.

This modification was found to work better in an experiment. A similar but more complex model than that of this report which included feedback and interval valued monotonic functions could either overflow or reach an equilibrium. Using the method of item (3), in chapter 7 we found
$p(overflow) = [.25, .5]$ and $p(equilibrium) = [.5, .75]$. Applying the modified method yielded $p(overflow) = [.33, .42]$ and $p(equilibrium) = [.58, .67]$, a definite improvement.

[2]More precisely, the drain was described by envelope functions relating outflow with pressure at the drain. This implicitly defines the aperture as an interval.

[3]However, some constraints are not completely reversible: A `square` constraint allows inferring that $2^2 = 4$, but only that $\sqrt{4} \in \{-2, 2\}$; A multiplication constraint $x \cdot y = z$ does not constrain $y$ when $x = z = 0$.

# Chapter 8

# Probabilities of Qualitative Behaviors for Dependability Analysis

The previous chapter described one method for bounding probabilities of qualitative behaviors, given partial quantitative information. In this chapter we use another method. The present method discretizes pdfs intelligently, in contrast to the brute force approach of the previous chapter. The reader should understand figure 7.1 from the previous chapter, at least, before reading this chapter.

## 8.1 Overview

The quantitative aspects of model specifications are often only partially known. Thus there is a need for ways to analyze models whose quantitative specifications are incomplete. One important goal for model analysis is finding the probabilities of the different qualitatively distinct behaviors a model can exhibit. We show how a model with partial quantitative specifications may be analyzed to bound the probabilities of its behaviors. Some specifications may be characterized by intervals and others by probability distributions, making the technique quite flexible in its ability to use what quantitative information is available.

The technique takes knowledge in the form of a *structural model* with partial quantitative specifications. This model is used to automatically construct *behavioral models* by simulating it. The behavioral models together with the given partial quantitative information are then analyzed probabilistically by a modification of the histogram discretization method, resulting in intervals bounding the probabilities of occurrence for the behaviors. (The histogram discretization method is discussed in Ingram et al. [1968], Colombo and Jaarsma [1980] and chapter 7.)

We use dependability analysis as an example domain. The choice of dependability analysis as a domain was motivated by the fact that most dependability analysis tools face some significant problems that our approach

addresses. In our example domain, the technique is applied to a simple dependability model, where the goal is to bound the probability that a modeled system will work dependably for a given period of time.

## 8.2 Introduction

We describe a technique for finding probabilities of qualitative behaviors given partial quantitative information about model specifications in the form of intervals and probability distributions, and apply it to a dependability analysis problem. By quantitative model specifications we mean quantitative aspects of the a model such as parameters, inputs, and initial conditions. Our technique for bounding the probabilities of qualitatively meaningful behaviors given partial quantitative model specifications uses a modification of the histogram discretization method [Ingram et al. 68; Colombo and Jaarsma 80].

Critical applications like the Space Shuttle Program have contributed to the computer industry's quest for highly dependable computers. Such applications have increased the importance of dependability analysis in computer system development. However, even modern, state of the art dependability analysis tools rely on accurate quantitative model specifications, which in real situations are not available. Such specifications are obtainable only roughly at best, via expert knowledge or limited testing data. Therefore there is a need to do dependability analysis given uncertain quantitative information, thus avoiding unsupported assumptions.

We explain a dependability analysis technique which overcomes some problems facing present dependability analysis tools. This dependability modeling technique uses models whose quantitative specifications may be described with intervals, probability distributions, or numbers. In this application of Q3 [Berleant and Kuipers 91] we find the *probabilities of qualitative behaviors*. In this dependability example, the behaviors are system *crash* by some time point of interest, and system *dependable* until (at least) that time. Dependability is simply the probability of the dependable behavior.

There are currently four major dependability analysis techniques, three of them analytical. The simplest analytical techniques use combinatorics while the complex analytical evaluation techniques make use of Markov models. Petri net modeling is a popular simulation dependability analysis technique. Another simulation technique for dependability analysis is Monte Carlo simulation. Many dependability analysis tools like HARP [Bavuso 87], ARIES [Makam 82] and SHARPE [Sahner and Trivedi 86] have been developed and are available. All of these tools support one or more of the major dependability

analysis techniques. See Johnson [1988] and Mullazini [1986] for a thorough comparison of the available dependability analysis tools.

The state of the art dependability analysis tools face some significant problems. These include:

1. Numerical specifications are assumed correct. Yet accurate numerical inputs are often difficult or impossible to obtain, so tools return incorrect (unrealistic) results.

2. Present evaluation techniques do not easily handle uncertainty in the quantitative model specifications to convert it to uncertainty at the output (that is, the dependability).

3. With the notable exception of SHARPE, tools usually do not support hierarchical modeling.

Our Q3-based [Berleant and Kuipers 91] approach gives us the ability to describe quantitative model specifications as probability distribution functions or intervals (or numbers which are degenerate intervals). This alleviates problem (1) by providing more flexibility in describing the quantitative knowledge available, instead of requiring modellers to choose arbitrary "reasonable" numbers. The ability to express quantitative information using pdfs and intervals also alleviates problem (2). Given model inputs described as intervals or pdfs, output is an interval bounding the dependability, that is, the probability of system failure.

## 8.3 A Dependability Model Example

We model dependability using a simple model for a large system that can tolerate some limited number of failed elements. The elements fail at some rate, are repaired at some rate, and if the total number of failed elements exceeds the tolerance of the system it crashes. The rates and tolerance are to be described using partial quantitative information in the form of intervals or probability distribution functions (pdfs). Figure 8.1 shows the model.

Suppose we want to specify one of the model inputs, for example the *element failure rate* of figure 8.1, with a pdf. Figure 8.2 shows an iscosceles pdf representing a failure rate such as might be obtained subjectively from an expert.

Triangular and trapezoidal shapes are often assumed, e.g. by work in fuzzy reasoning [Shen and Leitch 91]. Gaussian (bell) curves with clipped

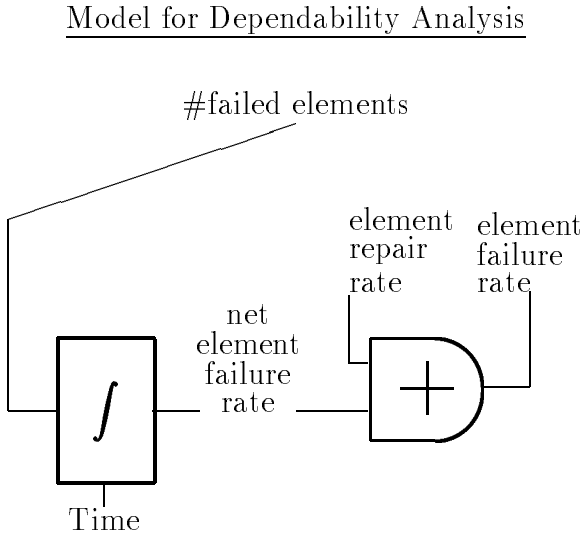



Figure 8.1: Simple dependability model for a larger system. If elements fail faster than they are repaired, eventually the failed element tolerance is exceeded and the system crashes. If this occurs after some time $t$ of interest, the system is dependable. (This model is compared with a standard Markov dependability model in section 8.5.)

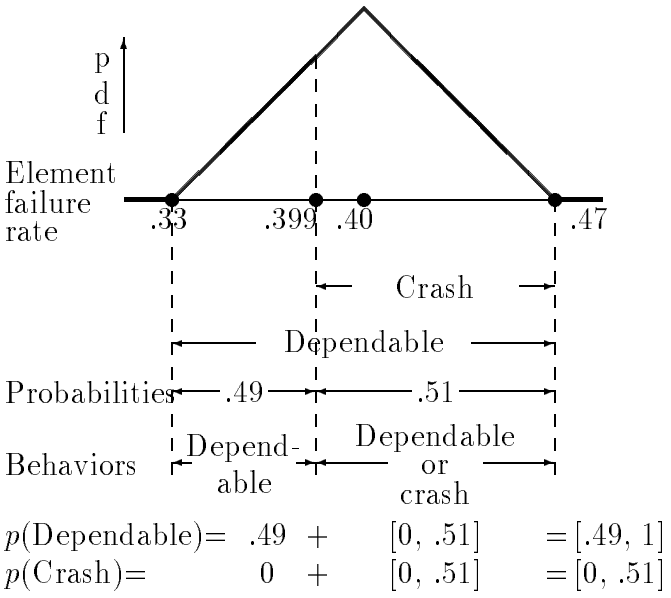



Figure 8.2: An isosceles pdf is used to derive probability ranges for behaviors of a system.

The domain of possible element failure rates is partitioned into two intervals which are each consistent with one or both of the qualitative behaviors called "dependable" and "crash". "Crash" is inconsistent with low element failure rates in the left portion of the pdf; "dependable" appears plausible for any possible element failure rate. This allows inferring intervals bounding the probabilities of each behavior, as illustrated by the figure: the left portion of the pdf has a probability mass of 0.49 which is assigned to the dependable behavior; the rest with a probability of 0.51 cannot be unambiguously assigned to a behavior, so each behavior is assigned its share, known only to be within the interval $[0, 0.51]$.

———————————

tails are another commonly used pdf. The method of this paper is capable of handling arbitrary pdfs, given code for their associated integrals (i.e. their cumulative distribution functions). Figure 8.2 exemplifies how probabilities of behaviors are bounded.

### 8.3.1 Finding the consistency of a behavior

It is necessary to determine, for each interval in the partition of the pdf domain (or in the case of more than one pdf-described model inputs, for each element in the *cartesian product* of the partitions), which of the qualitative behaviors the interval may be consistent with. Partitioning may be done unintelligently as in Berleant [1990] and chapter 7, but it is useful to make the partition wisely, so that the partition boundaries have some correspondence to boundaries of regions of the pdf domain space such that values within a region are all plausible for the same subset of behaviors.

The method for achieving these objectives is as follows:

1. *Represent the behaviors as constraint networks.* This is done by simulating the model to automatically produce behavior models, using the *Q3* program [Berleant and Kuipers 91][1]. The behavior models are represented as constraint networks consisting of the dependencies of the various model variables upon each other, instantiated for each time point in the simulation, as well as integration constraints that use values at a time point to constrain values in neighboring time points. A constraint network describing one behavior of the model of figure 8.1 is shown in figure 8.3.

2. Label the constraint network nodes with intervals, which are 1) the quantitative model specifications when these specifications are intervals, 2) the pdf domains when the specifications are pdfs, and 3) intervals describing the boundary conditions imposed by the behavior itself. The boundary conditions in figure 8.3 are, $T1 = K_1$ and #failed elements$_{T1} = MaxTolerance$; sample values for $K_1$ and MaxTolerance as well as other model inputs are given in table 8.1. (Note that an interval may be quite weak, e.g. $[0, \infty]$.)

3. Propagate the quantitative information throughout the constraint network, allowing it to settle using Waltz filtering [chapter 2; Davis 87].

[1] *Q3* derives from a previous program *Q2* [Kuipers and Berleant 88] which in turn derives from QSIM [Kuipers 84].

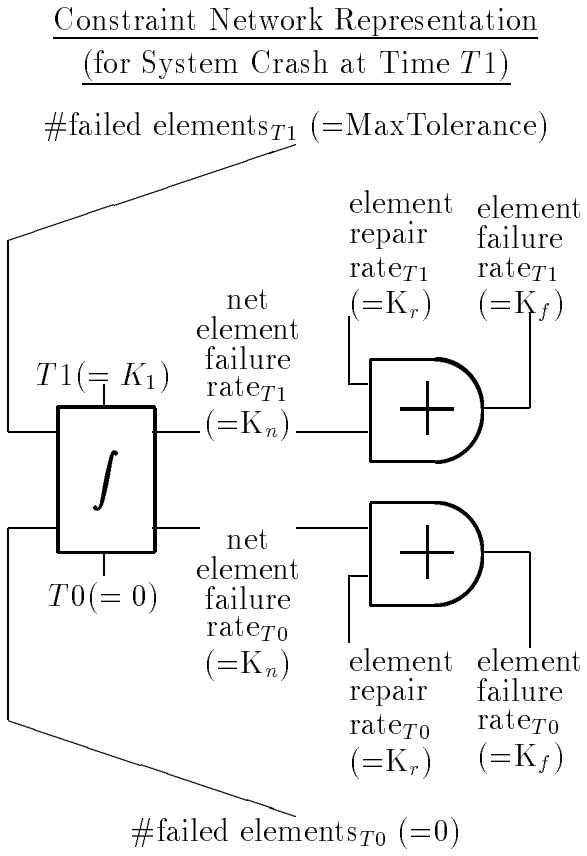



Figure 8.3: Constraint network representing a behavior of the model of figure 8.1.

Dependability analysis results

| element failure rate $\in$ [.033,.047] | element failures tolerated $\in$ [60,80] | element repair rate $\in$ [.007,.012] | $p(working)$ (dependable behavior), time = 1800 |
|---|---|---|---|
| $\frown$ ($\wedge$) | † | † | $> 0.49$ (0.48) |
| † | $\frown$ ($\wedge$) | † | $> 0.34$ (0.32) |
| $\frown$ ($\wedge$) | $\frown$ ($\wedge$) | † | $> 0.66$ (0.65) |

Table 8.1: Some dependability results for the 1-tank dependability model of figure 8.1. Two pdf shapes were tested. Results for an isosceles shape, $\wedge$, are shown in parentheses.
**Key:**
$\wedge$designates an isosceles pdf with base defined by the associated interval.
$\frown$designates a trimmed Gaussian pdf with 4 standard deviations within the associated interval, the tails outside trimmed off, and the area normalized to 1.
† indicates no pdf, just the interval.
Note that the shape of the pdf makes a difference in the results. With repair rate also described by a pdf (not shown) there was no further improvement in inference due to excess width in interval calculations [Moore 79]. However, application of the previous algorithm of chapter 7 to the output of the current algorithm, in conjunction if necessary with step size refinement, is expected to provide arbitrarily good results — given arbitrary amounts of computation.

———————————

4. Examine (automatically) the settled constraint network to see if, for the behavior represented by the network, the inferred plausible values for the pdf are a subinterval of the domain of the pdf. If so, values outside the subinterval are inconsistent with the behavior represented by the network, and the boundaries of the subinterval should be used in partitioning the domain.

5. The probability of each behavior is bounded as illustrated in figure 8.2.

## 8.4 Discussion

### 8.4.1 Excess width issues

Excess width in interval calculations causes distortion when the output of probabilistic analysis is a pdf, because excess width almost always leads to proportional assignment error (chapter 7). That the presence of excess widths in intervals used to build the pdf causes the inferred pdf to be invalid was recognized previously by Colombo and Jaarsma [1980].

Fortunately, in contrast to when the output of probabilistic inference is a pdf, when the output is probability bounds of behaviors excess width merely weakens those bounds. In other words, behavior probability estimates end up having excess width. This is due to the way a result collection interval is used (section 7.2): its probability mass $p$ is "shared" among the behaviors it is consistent with by assigning to each the mass $[0, p]$, as in figure 8.2, and a result collection interval with excess width may appear to be consistent with too many behaviors.

This weakening of results is true for the current algorithm in which we work backward from the behaviors to the input values consistent with them. Excess width causes more of the input domain to be ambiguous with respect to which behaviors a given input value seems consistent with, which results in more of the probability mass of the input domain being ambiguous, thus causing inferred probability intervals for the behaviors to be wider. See figure 8.2. In that figure, system crash is consistent with a low-level element failure rate in $[0.399, 0.47]$. But suppose failure rate was instead in $[0.35, 0.47]$. Then, working through the figure it is clear how this would lead to weaker inferences for $p(Dependable)$ and $p(Crash)$.

The excess width problem might also be approached by using known methods for eliminating or minimizing it. Such methods include target interval partitioning and behavior splitting (chapter 3), as well as step size refinement (chapters 4 and 5) and subdivision algorithms such as that of Kolev et al. [1988].

There is another source of uncertainty in the calculation of behavior probabilities, which is intrinsic to the model definition. A model may be defined using intervals to specify any given value. Such an interval is an abstraction of *any* pdf which is zero outside the endpoints of the interval. Thus the probability distributions of values specified only as intervals are not known, and this is exhibited as inability in principle to extract precise numerical (as opposed to interval) probabilities from the model. The advantage of this, of course, is that probabilistic analysis is possible even when some quantitative model specifications may be known only as an interval.

### 8.4.2 Mixing intervals and pdfs

We have not found any method other than histogram discretization which is flexible in this way. The reason for the flexibility of histogram discretization becomes clearer when one observes that each interval defining the location and width of a histogram bar constrains a certain probability mass — the area of the bar — to be within the bounds of the interval, yet says nothing about how it is distributed within that interval. Thus, describing a value as a single interval is in effect discretizing its pdf with a histogram of one bar which is consistent with the (unknown) pdf.

### 8.4.3 Ordering probability intervals

There may be more than one qualitatively meaningful behavior plausible for large regions of the input space. Probabilistic analysis then results in probability intervals which are wide and overlap a great deal. In such a case it will often still be possible to order them. For example, a behavior with probability $p_1 = [0.3, 0.7]$ *might* turn out in a real case or through more sophisticated analysis to be more likely than one with $p_2 = [0.4, 0.8]$. However, a rational person would guess $p_2 > p_1$. Thus, using the *measure of rational belief* semantics[2] we can order $p_1$ and $p_2$. More sophisticated conclusions can also be drawn from a collection of probability bounds, based on the fact that if one represents the probability of each behavior as a separate dimension, a space is formed in which the probability bounds delimit a polytope. This polytope may may be conveniently operated on with Bayes' theorem to produce useful conclusions [Snow 86].

## 8.5 Conclusions

This work has described a method for bounding probabilities of qualitative behaviors, given partial quantitative model specifications in the form of intervals and pdfs. Previous work on finding probabilities of behaviors has appeared in Yeh [1990], Doyle and Sacks [1989], and DeCoste [1990]. Monte Carlo simulation can also be used for this purpose. However, these techniques do not handle partial quantitative model specifications as flexibly as the method reported here: previously reported methods cannot handle quantitative specifications for which some specifications are intervals and others are pdfs. A more

[2]Cheeseman [1986] argues for the measure of rational belief semantics. However, the semantics of probability remains a topic of some controversy.

detailed discussion of previous work appears in the longer version of this paper [Berleant and Chandra 91].

The need for dependability analysis has become necessary for the development of next generation, high-performance computer systems. We have described a technique for bounding the probabilities of qualitatively meaningful behaviors. This allows using what knowledge is actually available to describe values of model inputs, whether it be in the form of numbers, intervals, probability distributions or a some of each. The technique was demonstrated on a simple model, and provided dependability bounds not easily obtainable otherwise, as other methods cannot work as flexibly with partial quantitative model specifications.

The idealized dependability model used here was checked by comparing the results with a considerably more complex Markov model requiring precise inputs, analyzed using the HARP tool. The results were similar to those predicted by the model used here, given precise numerical model specifications. Results for a Markov model given quantitative model specifications in the form of intervals and pdfs are not available; we plan to investigate the use of such partial quantitative information in Markov models in future research.

The technique appears to have utility and to merit further investigation in other domains requiring probabilistic bounds on qualitative alternatives given limited information about value, such as risk and decision analysis.

# Chapter 9

# Concluding Summary

In this report I have motivated qualitative-quantitative simulation, explained how it is done using the Q3 program, and extended it to bounding probabilities of qualitative behaviors.

Chapter 1 uses the rocket example to help motivate qualitative-quantitative simulation. A fired rocket can either fall back to Earth or escape to $\infty$. Quantitative simulation methods like numerical simulation and interval simulation do not find the separate distinct qualitative behaviors. Monte Carlo simulation does not classify its results into behaviors. Pure qualitative simulation cannot use quantitative information to refine predictions when it is available. This is why us we need to combine both quantitative and qualitative simulation.

Chapter 2 shows how adding even weak quantitative information to qualitative simulation can enable ruling out behaviors and models, and make quantitative predictions. For example, quite weak information about the speed of a baseball is sufficient to rule out escape from the Earth's gravity and make a "ballpark" estimate of when it will return to the ground. If all behaviors of a model are ruled out, then the model may be ruled out. Chapter 3 describes algorithms for providing improved quantitative inference over the simple Waltz filtering of chapter 2.

Chapter 4 describes how a state implicitly in the trajectory of a simulation may be inserted explicitly, leading to better inferences about a simulation trajectory. Inserting a new state makes the step size of the simulation smaller, leading to better results as is also the case in the usual methods of numerical simulation. Chapter 5 describes the *step size refinement* algorithm, which inserts few or many states into a simulation in a well defined way. This results in simulations that have important properties of both qualitative simulations and quantitative simulations. These properties include:

- Correctness, like good interval and qualitative simulators;
- Convergence, like good numerical and interval simulators;

- Stability, like good numerical and interval simulators.

Chapter 6 illustrates some applications of Q3. These include differential diagnosis and measurement interpretation. The suitability for future work of some other applications are also discussed.

Chapters 7 and 8 apply Q3 to working with partial quantitative information in the form of probability distribution functions (pdfs). Since we wish to flexibly deal with whatever quantitative information is available this is important. Chapter 7 gives one algorithm for bounding the probabilities of qualitative behaviors given inputs some of which may be described as pdfs and others as intervals. Chapter 8 provides another algorithm which uses boundary conditions imposed by the behavior and quantitative specifications to intelligently discretize pdfs that were discretized by brute force in chapter 7. The algorithm was applied to dependability analysis, addressing some limitations of traditional dependability analysis methods.

These probabilistic analysis techniques have the following desirable properties:

- Probability bounds are calculated safely, so while they may have excess width, they are correct in that they contain the actual behavior probability.
- Model inputs may be specified such that some are intervals and others are pdfs. This flexibility in working with intervals and pdfs is facilitated by the interval-based discretization of the pdfs, and is an advantage over other techniques, which usually require partial quantitative information to be either intervals or pdfs but not both.

Comparing with Monte Carlo simulation, Monte Carlo can handle either intervals or pdfs for model inputs, but not both in the same problem. In contrast, the method described can handle mixtures of pdfs and intervals in the same problem. Furthermore, bounds found using Monte Carlo are not safe, but may be too narrow because the random sampling Monte Carlo uses may miss the extreme cases. However, Monte Carlo can handle inputs which are partially dependent, whereas the present work requires independent inputs.

## 9.1 Future Work

There is much more to be done in qualitative-quantitative simulation research. Some directions include (1) Improvements to Q3, (2) Applications

using Q3, and (3) Alternatives to Q3. Each is discussed in the following paragraphs.

The most promising improvement to Q3, from a pragmatic standpoint, is in the integration step of the simulations. Q3 currently uses the mean value theorem to propagate quantitative information between temporally adjacent states in a simulation (chapter 2). This is simple, but weak. One improved approach would be to extend the Runge-Kutta method of numerical simulation to deal with intervals instead of numbers, and to propagate quantitative information either backward or forward in time, as appropriate, instead of only forward. A second improvement over the use of the mean value theorem would utilize Moore's Taylor Series method of interval simulation [Moore 79].

Applications of simulation are the major reason for doing simulation. Thus applying Q3 or other forms of qualitative-quantitative simulation is a relevant direction for further work. Some potential applications are discussed in chapter 6, which also describes how Q3 may be applied to measurement interpretation and diagnosis. But more needs to be done. Further development of algorithms for probabilistic analysis of qualitative behaviors (chapters 7 and 8) is also a promising direction for further work.

A natural next step in the development of Q3 would be to simulate a large, useful system. The scaling problems that would certainly arise could then drive further developments to the simulation technique. One application for simulating complex systems is in monitoring [Doyle et al. 89]. This could form the context for using Q3 to simulate a complex system.

Methods for qualitative-quantitative simulation other than that of Q3 are plausible. It seems certain that Monte Carlo simulation could be used as a front end for a qualitative-quantitative simulator. The Monte Carlo runs would be used for extraction of characteristics of qualitative behaviors. These behaviors would be used as input to Q3 just as QSIM behaviors currently are used. Besides Monte Carlo simulation, interval simulation may be useful as a front end for further qualitative-oriented processing. These approaches have not been tried, yet are quite appropriate directions to be pursued.

# Appendix A

## The Lorenz attractor

The Lorenz attractor: a 3-D phase space portrait of the nonlinear chaotic system

$$\begin{aligned} dx/dt &= -10x + 10y \\ dy/dt &= 28x - y - xz \\ dz/dt &= -8z/3 + xy. \end{aligned}$$

The numerical simulation path shown here is incorrect in its precise details because the system is chaotic, but the qualitative aspects are clear. An interesting problem for qualitative-quantitative simulation?

# Appendix B

## Proof of convergence and stability for step size refinement.

We want to provide a bound on the uncertainty of the result of a simulation as a function of the step size and the uncertainty in initial conditions. First, we provide the bound for a first-order differential equation, with fixed step size. Then we extend the result to a first-order *system* of equations, and variable step size.

**Lemma 1 (Bounded Uncertainty)** *Let $y' = f(y)$ be a first-order qualitative differential equation, where $f$ represents an expression composed of arithmetic constraints and monotonic function constraints with numerical envelopes. We assume that $f$ satisfies the Lipschitz condition.*[1] *We also consider only a bounded subset of the domain, $|y| \leq M$.*[2]

*Let $h$ be the step size for the simulation, and let $w(Y_n)$ be the width of the interval representing the uncertainty in $y$ after $n$ steps of simulation. $w(Y_0)$ thus represents the uncertainty in the initial conditions. Then, for any fixed time $t = nh$, there are constants $K_1$ and $K_2$ such that*

$$w(Y_n) \leq K_1 w(Y_0) + K_2 h. \tag{B.1}$$

**Proof**: The proof follows the standard proofs for convergence and stability for interval simulation [Moore 79].

---

[1] A function $f$ satisfies the Lipschitz condition if there is a constant $L$ such that $|f(y_1) - f(y_2)| \leq L|y_1 - y_2|$ for all $y_1$, $y_2$ in the domain of $f$. This property extends straight-forwardly to interval extensions of functions: "An interval extension $F$ is *Lipschitz* in $X_0$ if there is a constant $L$ such that $w(F(X)) \leq Lw(X)$ for every $X \subseteq X_0$." [Moore 1979:34].

[2] The bound $|y| \leq M$ is not a major restriction because physical variables in real systems remain finite. However, in case of a model variable that diverges to infinity, the theorem only applies within a bounded region.

1. The inference method used by Q3 to propagate information from one time step to the next may be represented by the **mean value constraint** which is applied iteratively, $Y_{n+1}$ becoming narrower on successive iterations.

$$Y_{n+1} := [Y_n + h(F(Y_n) \sqcup F(Y_{n+1}))] \cap Y_{n+1}$$

   where $F(Y)$ is the natural interval extension of the real function $f(y)$. $F$ is specified by the constraint model of the system of interest.

2. The mean value constraint is iterated until a fixed point is reached with no further changes to $Y_{n+1}$, or until $Y_{n+1}$ has no legal interval. The iteration will eventually terminate (section 5.3.1). If a fixed point exists, then

$$Y_{n+1} = Y_n + h[F(Y_n) \sqcup F(Y_{n+1})] \tag{B.2}$$

   If there is no fixed point the behavior is pruned.

   If a solution exists, it will be consistent with some qualitative behavior generated by the QSIM algorithm because QSIM provides that guarantee [Kuipers 86], and the solution will be within the interval bounds specified by equation B.2 because Waltz filtering with interval labels is correct (section 5.3.1).

3. Since $F$ is a natural interval extension, $F$ is **inclusion monotonic**, meaning $F(A) \subseteq F(B)$ if $A \subseteq B$ [Moore 79]. Hence $F(Y_n) \subseteq F(Y_n \sqcup Y_{n+1})$ and $F(Y_{n+1}) \subseteq F(Y_n \sqcup Y_{n+1})$. Consequently,

$$F(Y_n) \sqcup F(Y_{n+1}) \subseteq F(Y_n \sqcup Y_{n+1}).$$

   So from (B.2) we conclude

$$Y_{n+1} \subseteq Y_n + hF(Y_n \sqcup Y_{n+1})$$

4. We now change our interest from intervals to interval *widths*.

$$w(Y_{n+1}) \leq w(Y_n) + hw(F(Y_n \sqcup Y_{n+1})) \tag{B.3}$$

5. Because $F$ is Lipschitz, we know that

$$w(F(Y_n \sqcup Y_{n+1})) \leq Pw(Y_n \sqcup Y_{n+1})$$

   where $P$ is the Lipschitz constant for $F$. Therefore,

$$w(Y_{n+1}) \leq w(Y_n) + hPw(Y_n \sqcup Y_{n+1})$$

6. We can infer $Y_{n+1}$ is bounded by $Y_n$ plus a function of the step size $h$ from (B.2). Then we can infer that $w(Y_{n+1})$ is bounded using (B.3). Since $\mid y \mid \leq M$,

$$w(Y_{n+1}) \leq w(Y_n) + hPw(Y_n \sqcup (Y_n + hF([-M, M])))$$

7. Since $f$ is Lipschitz and $F$ is a natural interval extension, $F([-M, M])$ is bounded, by Moore's theorem 4.1 with $N = 1$ [Moore 79]. (In cases of division by zero see note 1.) Let $M_F = \mid F([-M, M]) \mid$, where the absolute value of an interval is the maximum of the absolute values of its endpoints. Then

$$\begin{aligned} w(Y_{n+1}) &\leq w(Y_n) + hPw(Y_n \pm hM_F) \\ &= w(Y_n) + hP[w(Y_n) + 2hM_F] \\ &= w(Y_n) + hPw(Y_n) + 2h^2PM_F \\ &= (1 + hP)w(Y_n) + 2h^2PM_F \end{aligned}$$

Thus for the "worst" case, that is, the widest possible $Y$s,

$$w(Y_{n+1}) = (1 + hP)w(Y_n) + 2h^2PM_F$$

8. We now have a first order linear difference equation which can be solved by applying a standard formula [Derrick and Grossman 76], giving

$$\begin{aligned} w(Y_{n+1}) &= w(Y_0)\left(\prod_{k=0}^{n}(1 + hP)\right) + 2h^2PM_F\sum_{k=0}^{n}\left(\prod_{j=k+1}^{n}(1 + hP)\right) \\ &= w(Y_0)(1 + hP)^{n+1} \\ &\quad +2h^2PM_F[(1 + hP)^n + (1 + hP)^{n-1} + ... + (1 + hP)^0] \end{aligned}$$

We can eliminate the intermediate powers of $(1 + hP)$ by multiplying both sides by $(1 + hP)$ and then subtracting:

$$\begin{aligned}
(1+hP)w(Y_{n+1}) &= w(Y_0)(1+hP)^{n+2} \\
&\quad +2h^2PM_F[(1+hP)^{n+1}+...+(1+hP)^1] \\
-w(Y_{n+1}) &= -w(Y_0)(1+hP)^{n+1} \\
&\quad +2h^2PM_F[-(1+hP)^n-...-(1+hP)^0] \\
\hline
hPw(Y_{n+1}) &= hP(1+hP)^{n+1}w(Y_0)+2h^2PM_F((1+hP)^{n+1}-1)
\end{aligned}$$

9. We know $e^{hP} = 1 + hP + \frac{(hP)^2}{2!} + \frac{(hP)^3}{3!} + ...$ (see many elementary calculus texts). Hence, $1 + hP \leq e^{hP}$, and

$$\begin{aligned}
hPw(Y_{n+1}) &\leq hP(e^{hP})^{n+1}w(Y_0)+2h^2PM_F((e^{hP})^{n+1}-1) \\
&= hPe^{hP(n+1)}w(Y_0)+2h^2PM_F(e^{hP(n+1)}-1)
\end{aligned}$$

Substituting $n$ for $n+1$ for brevity,

$$hPw(Y_n) \leq hPe^{hnP}w(Y_0)+2h^2PM_F(e^{hnP}-1)$$

10. Recall that $Y_n$ is the interval calculated to contain $y$ at time $b = nh$. Then

$$\begin{aligned}
hPw(Y_n) &\leq hPe^{bP}w(Y_0)+2h^2PM_F(e^{bP}-1) \\
w(Y_n) &\leq e^{bP}w(Y_0)+2hM_F(e^{bP}-1)
\end{aligned}$$

11. For any fixed time $t = nh$, $K_1 = e^{tP}$ and $K_2 = 2M_F(e^{tP}-1)$ are constant. Therefore, at time $t = nh$,

$$w(Y_n) \leq K_1w(Y_0)+K_2h$$

Recall $w(Y_0)$ represents the uncertainty in the initial condition of the state variable $Y$. As this uncertainty approaches zero, $w(Y_0) \rightarrow 0$. When in addition $h \rightarrow 0$, the uncertainty in $Y_n$, $w(Y_n)$, also approaches 0. Thus we have shown *convergence*. This result also demonstrates *stability*: the uncertainty in the predictions is bounded by the uncertainty in the initial conditions times a constant $K_1$, as $h \rightarrow 0$ (see section 5.3.3).

This proves the lemma. We now generalize it to produce the following theorem.

**Theorem 1** *Let* $\boldsymbol{y}'=\boldsymbol{f}(\boldsymbol{y})$ *be a system of first-order qualitative differential equations, where* $\boldsymbol{y}$ *is a vector of state variables, and* $\boldsymbol{f}$ *represents a vector of expressions composed of arithmetic constraints and monotonic function constraints with numerical envelopes. We assume that* $\boldsymbol{f}$ *satisfies the Lipschitz condition. We also consider only a bounded subset of the domain,* $||\boldsymbol{y}|| \leq M$.

*Let* $h$ *be the maximum step size for the simulation, and let* $w(Y_n)$ *be the maximum width of uncertainty intervals for variables in* $\boldsymbol{y}$ *after* $n$ *steps of simulation. Then, for any fixed time* $t = nh$, *there are constants* $K_1$ *and* $K_2$ *such that*

$$w(Y_n) \leq K_1 w(Y_0) + K_2 h. \tag{B.4}$$

**Proof**:

1. The proof of the Lemma extends to higher order systems as follows. First, assume the system is described as a system of 1st order equations. Gear [1971] describes how higher order systems can be transformed into 1st order systems. We then take the various $Y_m$ to be interval vectors instead of individual intervals, and the $F$'s to be a vector of functions.

   Step 2 therefore involves one mean value constraint for *each* member of vector $F$, and termination becomes a question. The algorithm needs to prevent starvation of any individual constraint for a fixed point to be reached. Avoiding starvation is discussed in section 5.3.1.

   The third step, correct for a single $F$, generalizes to the function vector case by application to each component. Then we discuss widths, step 4, which must be redefined for vectors. The width of an interval vector is the width of its widest component [Moore 79]. Since Moore proves his theorem 4.1 and lemma 4.2 for interval vectors, and widths are single numbers whether their arguments are numbers or vectors, the remaining steps work for vectors as stated.

2. The proof as stated takes all steps as having the same size $h$. It extends easily to variable step sizes (recall Q3 utilizes variable step sizes). The intuitively reasonable key is to take $h_m$ as the maximum step size instead of the size of each step. Then, most of the intermediate conclusions become conservative, because most of the steps they apply to use $h < h_m$. The reader may find it instructive to step through the proof with this in mind.

**Notes**

1. In cases where $F$ specifies division, there is the possibility of $F$ being undefined in cases where the divisor is an interval having 0 as a member or endpoint. In such cases $f$ does not satisfy the Lipschitz property over intervals containing zero, putting it outside the scope of this proof. Qualitative constraints deriving from QSIM quantity spaces will disambiguate the sign of a quantity so that division by intervals containing zero does not occur in practice. However, an interval divisor can still have 0 as an endpoint, in which case open-ended intervals can result, $F$ is not Lipschitz as required by the proof and convergence is not guaranteed to occur. Fortunately, such convergence problems can often be handled by behavior splitting, as exemplified by the example of section 5.2, because a behavior split can always put the $\infty$ in one case, usually making the other case Lipschitz.

2. The proof assumes a system description in the form of a set of first order differential equations, the standard form for convergence proofs [Gear 71]. Q3, however, utilizes a constraint network. Fortunately any set of first order differential equations can be easily converted into the constraint network representation, by expressing all binary operations as ternary constraints and all unary operations as binary constraints. Thus the equation $y' = x'(y + x')$ becomes the constraints `(D/DT y dydt)`, `(D/DT x dxdt)`, `(PLUS y dxdt w)` and `(MULT dxdt w dydt)`.

   Conversely, a constraint network can be transformed into a set of equations by expressing each ternary constraint `(OP a b c)` as an equation $a$ `op` $b = c$ and each binary constraint `(OP a b)` as an equation `op`$(a) = b$. This set of equations can then be converted into a set of first order differential equations by standard means [Gear 71].

3. The Lemma shows bounded uncertainty for each fixed time $t = nh$, but the constants $K_1$ and $K_2$ are exponential functions of $t$. This reflects the "uncertainty explosion" in traditional interval simulators. This explosion is controlled in Q3 because new states are only created when bounded by pairs of existing states, and because interval values are updated by intersection, so that uncertainty can only decrease, and never increase. The initial weak quantitative bounds derived by Q2 provide a bound on the uncertainty, which Q3 can only strengthen.

# BIBLIOGRAPHY


[1] *Proceedings, Second Annual Conference on AI, Simulation and Planning in High Autonomy Systems*, IEEE Computer Society Press, 1991.

[Abelson et al. 90] Abelson, H., M. Eisenberg, M. Halfant, J. Katzenelson, E. Sacks, G. Sussman, J. Wisdom, and K. Yip, Intelligence in Scientific Computing, *Communications of the ACM* **32**: 546-562, May 1989. Reprinted in Weld and de Kleer [1990].

[Abelson and Sussman 85] Abelson, H. and G. Sussman, *Structure and Interpretation of Computer Programs*, MIT Press, 1985.

[Alefeld and Herzberger 83] Alefeld, G. and Jurgen Herzberger, *Introduction to Interval Computations*, Academic Press, 1983.

[Asaithambi et al. 82] Asaithambi, N.S., Shen Zuhe and R.E. Moore, On Computing the Range of Values, *Computing*, **28**:225-237, 1982.

[Bavuso 87] S. J. Bavuso, B. Dugan, K. S. Trivedi, E. M. Rothmann, and W. E. Smith, Analysis of Typical Fault-Tolerant Architectures using HARP, *IEEE Transactions on Reliability* **R-26**(2):176-185, 1987.

[Berleant 90] Daniel Berleant, Constraint Propagation by Waltz Filtering is Sound, in *AAAI-90 Constraint Directed Reasoning Workshop Notes*, pp. 2-10.

[Berleant 91] Daniel Berleant, Combining Qualitative and Quantitative Simualation: in Brief, in *Proceedings of the Second Annual Conference on AI, Simulation and Planning in High Autonomy Systems*, IEEE Computer Society Press, 1991.

[Berleant 90a] Daniel Berleant, Probabilities of Qualitative Behaviors from Probability Distributions on Inputs. Technical report AI90-136, Artificial Intelligence Laboratory, Dept. of Computer Science, University of Texas at Austin, August 1990.

[Berleant and Chandra 91] Daniel Berleant and Arun Chandra, Probabilities of Qualitative Behaviors for Dependability Analysis, in *AAAI-91 Model Based Reasoning Workshop Notes*.

[Berleant and Kuipers 91a] Daniel Berleant and Benjamin Kuipers, Bridging the Gap from Qualitative to Numerical Simulation. Technical report, forthcoming, Artificial Intelligence Laboratory, Dept. of Computer Science, University of Texas at Austin, 1991.

[Berleant and Kuipers 91b] Daniel Berleant and Benjamin Kuipers, Qualitative-Quantitative Simulation with Q3, to appear in B. Faltings and P. Struss, eds., *Recent Advances in Qualitative Physics*, MIT Press, 1991.

[Bobrow 85] Bobrow, D. G., ed., *Qualitative Reasoning about Physical Systems*, MIT Press, 1985. Reprinted from *Artificial Intelligence* **24** (1984) special volume.

[Brandt 77] Brandt, Achi, Multi-Level Adaptive Solutions to Boundary-Value Problems, *Mathematics of Computation* **31**(138):333-390, April 1977.

[Briggs 87] W. L. Briggs, *A Multigrid Tutorial*. Society for Industrial and Applied Mathematics, 1987.

[Brooks 81] Rodney A. Brooks, Symbolic Reasoning Among 3-D Models and 2-D images, *Artificial Intelligence* **17** (1981) 285-348.

[Bryant 81] R. E. Bryant, MOSSIM: A Switch-Level Simulator for MOS LSI, in *Proceedings of the 18th Design Automation Conference* (July 1981) 786-790.

[Chawla et al. 75] B. R. Chawla, H. K. Gummel and P. Kozak, MOTIS — An MOS Timing Simulator, *IEEE Transactions on Circuits and Systems* **CAS-22** (12) (Dec. 1975) 901-910.

[Cheeseman 86] Peter Cheeseman, Probabilistic Versus Fuzzy Reasoning. In *Uncertainty in Artificial Intelligence*, L. N. Kanal and J. F. Lemmer, eds., Elsevier Science Publishers B. V. (North-Holland), 1986, pp. 85-102.

[Cherkassky 84] V. Cherkassky, E. Opper, and M. Malek, Reliability and Fault Diagnosis Analysis of Fault-Tolerant Multistage Interconnection Networks, in *Proceedings IEEE 14th Fault-Tolerant Computing Symposium*, 1984, pp. 246-251.

[Cohen and Feigenbaum 82] Paul R. Cohen and Edward A. Feigenbaum, eds., *Handbook of Artificial Intelligence, vol. 3*, William Kaufmann Inc., 1982, pp. 313-321.

[Colombo and Jaarsma 80] A.G. Colombo and R.J. Jaarsma, A Powerful Numerical Method to Combine Random Variables, *IEEE Transactions of Reliability*, **R-29**(2):126-129, June 1980.

[Cooper 89] M. C. Cooper, An Optimal $k$-consistency Algorithm, *Artificial Intelligence* **41**(1) (November 1989) 89-95.

[Dalle Molle 89] D. Dalle Molle, Qualitative Simulation of Dynamic Chemical Processes, Chemical Engineering Dept. dissertation, University of Texas Artificial Intelligence Laboratory report AI89-107, May 1989.

[Das et al. 90] C. R. Das, J. Kreulen, and M. Thazhuthaveetil, Dependability Modeling for Multiprocessors, *IEEE Computer*, **23**(10):7-19, October 1990.

[Davis 84] Randall Davis, Diagnostic Reasoning Based on Structure and Behavior, *Artificial Intelligence*, 1984, **24**: pages 347-410.

[Davis 87] Ernest Davis, Constraint Propagation with Interval Labels, *Artificial Intelligence* **32** (July 1987) 281-331.

[Dean and Boddy 88] Tom Dean and Mark Boddy, Reasoning About Partially Ordered Events, *Artificial Intelligence* **36** (3) (1988) 375-399.

[Dean and McDermott 87] Tom Dean and Drew McDermott, Temporal Database Management, *Artifical Intelligence* **32** (1) (1987) 1-55.

[DeCoste 90] Dennis DeCoste, Dynamic Across-Time Measurement Interpretation, in *Proceedings of the Eight National Conference on Artificial Intelligence (AAAI-90)*, pp. 373-379, 1990.

[de Kleer 75] Johan de Kleer, Qualitative and Quantitative Knowledge in Classical Mechanics, technical report AI-TR-352, MIT Artificial Intelligence Laboratory, 1975.

[de Kleer 90] Johan de Kleer, Multiple Representations of Knowledge in a Mechanics Problem-Solver, in *Proceedings IJCAI-77*, also reprinted in Weld and de Kleer [1990].

[de Kleer and Brown 85] Johan de Kleer and John Seely Brown, A Qualitative Physics Based on Confluences, in *Artificial Intelligence* **24**, reprinted as Bobrow [1985].

[Derrick and Grossman 78] W. Derrick and S. Grossman, *Elementary Differential Equations with Applications*, Addison-Wesley, 1978.

[Doyle et al. 89] R. Doyle, D. Berleant, L. Falcone, and U. Fayyad, Selective Simulation and Selective Sensor Interpretation in Monitoring, in *Proceedings of AIAA Computers in Aerospace VII*, American Institute of Aeronautics and Astronautics, 370 L'Enfant Promenade, S.W., Washington, D.C., October 1989.

[Doyle and Sacks 89] Doyle, J. and E. P. Sacks, Stochastic Analysis of Qualitative Dynamics, in *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pp. 1187-1192, 1989.

[Dubois and Prade 80] D. Dubois and H. Prade, *Fuzzy Sets and Systems: Theory and Applications*, Academic Press, 1980.

[Eijgeraam 81] P. Eijgenraam, *The solution of Initial Value Problems Using Interval Arithmetic: Formulation and Analysis of an Algorithm*, Mathematisch Centrum, Amsterdam, 1981.

[Feng 86] T. Feng, Development of a Fault-Tolerant Computer System for Ballistic Missile Defense, Technical Report to RADC, The Pennsylvania State University, 1986.

[Finizio and Ladas 82] N. Finizio and G. Ladas, *Ordinary Differential Equations with Modern Applications*, Wadsworth Publishing Co., 1982.

[Fishwick and Zeigler 90] Paul A. Fishwick and Bernard P. Zeigler, Qualitative Physics: Towards the Automation of Systems Problem Solving, in *Proceedings of the First AI, Simulation and Planning Conference*, IEEE Computer Society Press, Washington, DC, 1990, pp. 118-134.

[Forbus 83] Kenneth D. Forbus, Measurement Interpretation in Qualitative Process Theory, in *Proceedings IJCAI-8*, 1983, pp. 315-320.

[Forbus 85] Kenneth Forbus, Qualitative Process Theory, in *Artificial Intelligence* **24**, reprinted in Bobrow [1985].

[Forbus 86] Kenneth D. Forbus, Interpreting Measurements of Physical Systems, in *Proceedings AAAI-86*, 1986, pp. 113-117.

[Forbus 90] Kenneth Forbus and Brian Falkenhainer, Self-Explanatory Simulations: an Integration of Qualitative and Quantitative Knowledge, *Proceedings AAAI-90* (380-387).

[Forrester 61] Jay W. Forrester, *Industrial Dynamics*, M.I.T. Press, 1961.

[Gear 71] William Gear, *Numerical Initial Value Problems in Ordinary Differential Equations*, Prentice-Hall, 1971.

[Genesereth 84] Michael R. Genesereth, The Use of Design Descriptions in Automated Diagnosis, in *Artificial Intelligence*, 1984, **24**: pp. 411-436.

[Gleick 87] James Gleick, *Chaos: Making a New Science*, Viking Penguin Inc., 1987.

[Henrici 62] P. Henrici, *Discrete Variable Methods for Ordinary Differential Equations*, John Wiley and Sons, 1962.

[Hon 87] Robert W. Hon, Dynamic Analysis Tools, chapter 6 of Steven M. Rubin, *Computer Aids for VLSI Design*, Addison-Wesley, 1987.

[Ingram et al. 68] G. E. Ingram, E. L. Welker, and C. R. Herrmann, Designing for Reliability based on Probabilistic Modeling using Remote Access Computer Systems, in *Proceedings ASME 7th Reliability and Maintainability Conference*, 1968, pp. 492-500.

[Insinga 84] Aron K. Insinga, Behavioral Modeling in a Structural Logic Simulator, *ICCAD '84* (November 1984) 42-44.

[Iwasaki 88] Yumi Iwasaki, Causal Ordering in a Mixed Structure, *Proceedings AAAI-88* (313-318).

[Iwasaki and Simon 86] Yumi Iwasaki and Herbert Simon, Causality in Device Behavior, *Artificial Intelligence* **29** (July 1986).

[Jackson 82] Jackson, P. S., A Second-Order Moments Method for Uncertainty Analysis, *IEEE Transactions on Reliability*, **R-31**(4)382-384, 1982.

[Jackson et al. 81] P. Jackson, R. Hockenbury and M. Yeater, Uncertainty Analysis of System Reliability and Availability Assessment, *Nuclear Engineering and Design* **68**:5-29, 1981.

[Johnson 89] B. Johnson, *Design and Analysis of Fault Tolerant Digital Systems*, Addison-Wesley, Reading, MA, 1989.

[Johnson and Malek 88] A. Johnson, and M. Malek, Survey of Software Tools for Evaluating Reliability, Availability, and Serviceability, IBM Technical Report, TR 51.0404-1, 1988.

[Kalos and Whitlock 86] M. H. Kalos and P. Whitlock, *Monte Carlo Methods*, J. Wiley and sons, 1986.

[Kaplan 81] Stanley Kaplan, On the Method of Discrete Probability Distributions in Risk and Reliability Calculations — Application to Seismic Risk Assessment, Risk Analysis, **1**(3), 1981.

[Karp and Friedland 87] Peter D. Karp and Peter Friedland. Coordinating the Use of Qualitative and Quantitative Knowledge in Declarative Device Modeling, Knowledge Systems Lab. Report KSL 87-09, Stanford Computer Science Dept., 1987.

[Kolev et al. 88] Lubomir V. Kolev, Valeri M. Mladenov, and Simeon S. Vladov, Interval Mathematics Algorithms for Tolerance Analysis, *IEEE Transactions on Circuits and Systems*, **35**(8):967-975, August 1988.

[Kuipers 85] Benjamin Kuipers, Commonsense Reasoning about Causality: Deriving Behavior from Structure, in *Artificial Intelligence* **24**, reprinted in Bobrow [1985].

[Kuipers 85] Benjamin Kuipers, Qualitative Simulation, in *Artificial Intelligence* **32** (1986) 57-95.

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

[Kuipers 88] Benjamin Kuipers, The Qualitative Calculus is Sound but Incomplete: a Reply to Peter Struss, *Int. J. Artificial Intelligence in Engineering* **3**(3):170-173, 1988.

[Kuipers and Berleant 88] Benjamin J. Kuipers and Daniel Berleant, Using Incomplete Quantitative Knowledge in Qualitative Reasoning, In *Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI-88)*, 1988.

[Kuipers and Berleant 90] Benjamin Kuipers and Daniel Berleant, A Smooth Integration of Incomplete Quantitative Knowledge into Qualitative Simulation, Artificial Intelligence Laboratory tech. report AI90-122, University of Texas at Austin, Jan. 1990.

[Laprie 85] J. C. Laprie, Dependable Computing and Fault-Tolerance: Concepts and Terminology, *Proceedings IEEE 15th Fault-Tolerant Computing Symposium*, pp. 18-21, 1985.

[Lasota and Mackey 85] A. Lasota, and M. C. Mackey, *Probabilistic Properties of Deterministic Systems*, Cambridge University Press, 1985.

[Luenberger 73] D. G. Luenberger, *Introduction to Linear and Nonlinear Programming*, Addison-Wesley, 1973.

[Makam 82] S. V. Makam and A. Avizienis, ARIES 81: A Reliability and Life-Cycle Evaluation Tool for Fault-Tolerant Systems, in *Proceedings IEEE 12th Fault-Tolerant Symposium*, pp. 267-274, 1982.

[McDermott and Davis 84] Drew McDermott and Ernest Davis, Planning Routes Through Uncertain Territory, *Artificial Intelligence* **22** (March 1984) 107-156.

[Moore 66] Ramon E. Moore, *Interval Analysis*, Prentice-Hall, NJ, 1966.

[Moore 79] Ramon E. Moore, *Methods and Applications of Interval Arithmetic*, SIAM Studies in Applied Mathematics, SIAM, Philadelphia, 1979.

[Mullazini 86] M. Mullazini, Dependability Prediction: Comparisons of Tools and Techniques, Technical Report CS-1986-20, Duke University, 1986.

[Nishida and Doshita 91] T. Nishida and S. Doshita, Reasoning about Global Behavior of Ordianry Differential Equations by Combining Qualitative and Quantitative Analysis, in *Proceedings, Second Annual Conference on AI, Simulation and Planning in High Autonomy Systems*, IEEE Computer Society Press, 1991.

[Pearl 88] Judea Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann Publishers, Inc., 1988.

[Press et al. 86] W. Press, B. Flannery, S. Teukolsky, W. Vetterling, *Numerical Recipes*, Cambridge University Press, 1986.

[Pritsker 84] A. Pritsker. *Introduction to Simulation and SLAM II*, Halsted Press, Div. of John Wiley & Sons, Inc., NY, 1984.

[Raiffa 70] H. Raiffa, *Decision Analysis: Introductory Lectures on Choices under Uncertainty*, Addison-Wesley, Reading MA, 1970.

[Reiter 87] Raymond Reiter, A Theory of Diagnosis from First Principles, In *Artificial Intelligence*, 1987 **32**:57-95.

[Sacks 87] Elisha Sacks, Hierarchical Reasoning About Inequalities, in *Proceedings AAAI-87* (649-654).

[Sacks 90a] Elisha Sacks, Automatic Qualitative Analysis of Dynamic Systems Using Piecewise Linear Approximations, *Artificial Intelligence* **41** (3) (Jan. 1990) 313-364.

[Sacks 90b] Elisha Sacks, A Dynamic Systems Perspective on Qualitative Simulation, *Artificial Intelligence* **42** (2-3) (March 1990) 349-362.

[Sahner and Trivedi 86] R. A. Sahner and K. S. Trivedi, Reliability Modeling Using SHARPE, *IEEE Transactions on Reliability* **R-36**(2):186-193, 1986.

[Shen and Leitch 91] Qiang Shen and Roy Leitch, Integrating Common-Sense and Qualitative Simulation by the Use of Fuzzy Sets, To appear in *Current Work in Qualitative Physics*, MIT Press, 1991. Also in *Proceedings of the 4th International Workshop on Qualitative Physics*, Lugano, Switzerland, July 1990, 220-232.

[Simmons 83] Reid Simmons, The Use of Qualitative and Quantitative Simulations, in *Proceedings AAAI-83* (364-368).

[Simmons 86] Reid Simmons, "Commonsense" Arithmetic Reasoning, in *Proceedings AAAI-86* (118-124).

[Simmons 88] Reid Simmons, A Theory of Debugging Plans and Interpretations, in *Proceedings AAAI-88* (94-99).

[Simmons and Davis 87] Reid Simmons and Randall Davis, Generate, Test and Debug: Combining Associational Rules and Causal Models, in *Proceedings IJCAI-87* (1071-1078).

[Skorstad and Forbus 89] G. Skorstad and K. Forbus, Qualitative and Quantitative Reasoning about Thermodynamics, in proceedings of the 3rd Qualitative Physics Workshop, 1989.

[Snow 86] Paul Snow, Bayesian Inference Without Point Estimates, in *Proceedings, AAAI-86*, pp.233-237.

[Steele 90] Guy L. Steele, Jr. *Common Lisp: The Language*, Digital Press, 1984 and 1990.

[Terman 83] C. J. Terman, RSIM — A Logic-Level Timing Simulator, in *Proceedings IEEE International Conference on Computer Design* (October 1983) 437-440.

[Tufte 83] Edward R. Tufte, *The Visual Display of Quantitative Information*, Graphics Press, Box 430, Cheshire CT 06410, 1983.

[Tuinenga 88] P. W. Tuinenga, *SPICE: A Guide to Circuit Simulations and Analysis Using PSpice*, Prentice-Hall, 1988.

[VLSI 86] VLSI Systems Design Staff, 1986 Survey of Logic Simulators, *VLSI Systems Design*, **VII**(2):32-40, Feb. 1986.

[Waltz 75] David Waltz, Understanding Line Drawings of Scenes With Shadows, in P. Winston, ed., *The Psychology of Computer Vision*, McGraw-Hill, 1975.

[Weld and de Kleer 90] Daniel Weld and Johan de Kleer, eds., *Readings in Qualitative Reasoning about Physical Systems*, Morgan Kaufmann Inc., 1990.

[Wellman 90] Michael Wellman, Foundations of Qualitative Probabilistic Networks, *Artificial Intelligence* (1990).

[Whitmore and Findlay 78] G. A. Whitmore and M. C. Findlay, *Stochastic Dominance: an Approach to Decision-making Under Risk*, Lexington Books, 1978.

[Williams 86] Brian Williams, Doing Time: Putting Qualitative Reasoning on Firmer Ground, in *Proceedings of the National conference on Artificial Intelligence (AAAI-86)*, pp. 105-112.

[Williams 88] Brian Williams, MINIMA: a Symbolic Approach to Qualitative Algebraic Reasoning, in *Proceedings AAAI-88* (264-269).

[Yeh 90] Yeh, A., Finding the Likely Behaviors of Static Continuous Nonlinear Systems, *Annals of Mathematics and Artificial Intelligence*, Vol. **2**, 1990.

[Yip 88] Kenneth M. Yip, Generating Global Behaviors Using Deep Knowledge of Local Dynamics, In *Proceedings AAAI-88*.

[Young 78] David M. Young, Stability ($h \rightarrow 0$ and $h > 0$) for Systems, Course notes supplement 78.6 superseding pp. 566-573 of Young and Gregory [1972], D. Young, Center for Numerical Analysis, University of Texas at Austin, 1978.

[Young and Gregory 72] D. M. Young and R. T. Gregory, *A Survey of Numerical Mathematics, vol. II*, Addison-Wesley Pub. Co., 1972.

[Zhao 91] Feng Zhao, Extracting and Representing Qualitative Behaviors of Complex Systems in Phase Spaces, in *Working Papers, Fifth International Workshop on Qualitative Reasoning about Physical Systems*, 1991.

# VITA

Jared Daniel Berleant was born in Buffalo, New York on March 6, 1959, the son of Arnold Jerome Berleant and Riva Berleant-Schiller. After graduating from Glen Cove High School in 1977, he attended the Massachusetts Institute of Technology, receiving a Bachelor of Science degree from there in February 1982. He worked as a programmer in Santa Clara California, then entered graduate school at the University of Texas in 1983.

Permanent address: Dept. of Computer
Sciences, University of
Texas, Austin, Texas 78712

This dissertation was typeset[3] with LaTeX by the author.

[3] LaTeX document preparation system was developed by Leslie Lamport as a special version of Donald Knuth's TeX program for computer typesetting. TeX is a trademark of the American Mathematical Society. The LaTeX macro package for The University of Texas at Austin dissertation format was written by Khe-Sing The.