Augmenting Qualitative Simulation with Global Filtering

Hyun-Kyung Kim
Qualitative Reasoning Group
Beckman Institute, University of Illinois
405 North Mathews Avenue, Urbana, Illinois 61801
hkim@cs.uiuc.edu

Abstract

Capturing correct changes both locally and globally is crucial to predicting the behavior of physical systems. However, due to the nature of qualitative simulation techniques, they cannot avoid losing some information which is useful for finding precise global behavior. This paper describes how global constraints are represented and manipulated in current simulation systems, using a model of an internal combustion engine. The basic idea of our approach is to automatically generate additional information for maintaining global constraints during simulation so that simulation techniques can filter global behaviors with the sufficient information. This is done by automatically introducing variables and controlling their values to guide correct transitions between the behaviors. We express this idea within the framework of *Qualitative Process* (QP) theory. This technique has been implemented and integrated into an existing qualitative simulation program QPE.

Introduction

Understanding the behaviors of physical systems is an important part of commonsense physics. Given a system description, a qualitative simulator predicts the behavior by finding possible states and the transitions between them.

Conceptually, we view determining possible behavior as a process of filtering illegal states and illegal state transitions (Struss, 1988). Transition filtering should be done at two different levels: *local* and *global* filtering. Local filtering focuses on whether a transition between two states is legal or not, based on the relation between the two without considering the other states. In qualitative simulation, this is determined by the changes of state variables and continuity (this process is called limit analysis). For example, for state variable a and b, if a < b and a is increasing while b is not changing, then next state may be a = b. On the other hand, global filtering concerns finding correct behaviors, i.e, correct sequences of transitions.

Simulation process, whether it is numerical or qualitative, finds behaviors only by local filtering. In spite of the lack of global filtering, numerical simulation can find correct behaviors since it uses precise metric information. However, in the case of qualitative simulation, the localized nature of simulation combined with qualitative description is not sufficient to infer precise global behaviors. Thus, understanding how to automate global filtering and how to integrate it with local filtering is crucial to designing an intelligent reasoning system.

Recent work in global constraints has focused on applying the idea of qualitative theory of dynamic systems to qualitative simulations (Lee & Kuiper, 1988; Struss, 1988). After states and transitions are computed by a qualitative simulator, each state is converted to a point in a phase space[1] and each transition to the segment which connects its predecessor and successor state. Once the behavior of a system is expressed as a trajectory in a phase space, then some geometric constraints are applied to this trajectory for filtering behaviors. The trajectory should be checked in every phase space due to the localized nature of simulations. This approach was useful for some cases, such as the stability of a cycle, even though this approach can be applied only to a limited class of systems. Basically, this method is not adequate for reasoning task where threshold plays an important role (Lee & Kuiper, 1988). Furthermore, understanding this approach is not easy for the people who do not have mathematical background. Obviously, people do not seem to use this phase space for filtering behaviors.

This papers presents an approach to extend qualitative simulation to include global filtering. Instead of checking global behavior after getting locally correct behavior, the global constraints are also filtered during limit analysis. Given the constraints about the behavior after a particular state,

[1] A phase space for a system is a Cartesian product of state variables.

the transitions to the illegal behavior from the state are pruned. Since this filtering is done by limit analysis, our approach can capture the global constraints which are sensitive to the changes of variables. Our approach is illustrated using an implemented model of an internal combustion engine. We describe how subtly different expansion periods with and without combustion in the cylinder are captured.

Problem

Envisioning is a process of deriving all possible behavior of a system given the qualitative descriptions of the system. The behavior is represented by a set of qualitative states and the transitions between them. When these states and transitions are expressed as a graph, the graph is called the envisionment for the system.

Like the state of a finite automaton, each state in an envisionment has the summary of information about its past path (Hopcroft & Ullman, 1979). The next transition from a state is determined only by the constraints between the information in the state. This is the locality nature of simulation. The past behavior needs not be traced for this since the current state has all information about these. This technique can predict accurate behavior on the assumption that each state carries enough information so that local filtering is sufficient to get globally correct behavior. Unlike numerical simulation, which uses precise metric information, qualitative simulation sometimes cannot avoid losing some useful information during suppression of details. We cannot always expect the local filtering to guarantee the global filtering in qualitative simulation. In this section, we illustrate one of the examples using a model of an internal combustion engine.

In an internal combustion engine (Ferguson, 1976), a piston is connected to a crankshaft through a connecting rod (Figure 1). It goes through four phases (i.e., intake, compression, power, and exhaust) to complete one cycle. The rapid heat rise by combustion is translated into pressure which acts on the piston to force it down. Then, by the geometry, positive torque is transmitted to the crankshaft by the connecting rod. The flywheel, which is connected to the crankshaft, also gets positive torque during this power stroke. The heavy flywheel gives back to the crankshaft, during the three other strokes, the surplus energy it took during the power stroke. The interaction of the motions of each part, and pressure changes plays a key role in understanding this system.

In building our model, we have focused on the interaction, ignoring features irrelevant to showing our motivating example, such as intake and exhaust flows. For this, our model is built based

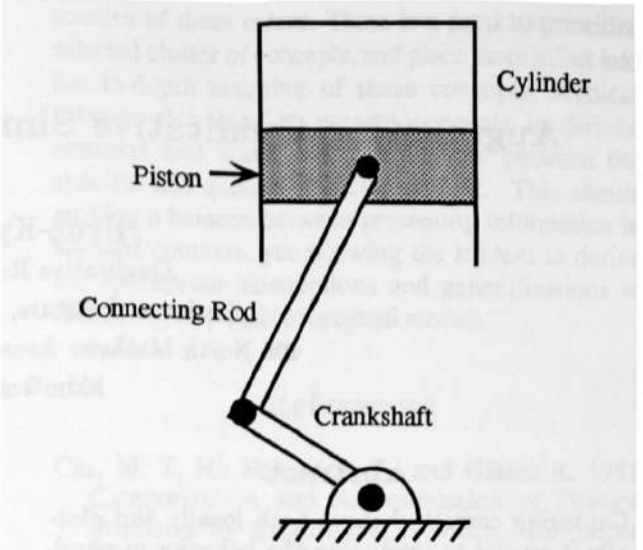

Figure 1: Piston cylinder geometry

on the following assumptions: (1) Working fluid is ideal gas, (2) Fixed mass of working fluid through cycle, and (3) Combustion is modeled as heat addition from external source. In addition to these, friction is considered in motion. With these assumptions, a piston-cylinder repeats the cycle of compression and expansion as the piston moves upward and downward (Figure 2a). When the crankshaft is at *Top dead center (TDC)* and *Bottom dead center (BDC)*, the piston reaches its highest and lowest position, respectively. Each part eventually will stop moving due to friction.

Suppose combustion happens at TDC. The expansion period after combustion (i.e., power stroke) is slightly different from the expansion without combustion: each part will not stop during the former while it might stop during the latter. Once the pressure is increased by combustion, say `(pressure ?cylinder) >= combustion-pres`, the pressure remains high enough to accelerate the crankshaft even though the pressure is decreasing during following expansion period. The geometry of the engine is designed so that once the pressure reaches `combustion-pres` at TDC, the pressure until BDC is greater than `nonstop-pres`. The `nonstop-pres` is a pressure sufficient to overcome the friction of each part.

However, it is impossible to distinguish these different behaviors with current qualitative simulators(Figure 2b) since it requires filtering paths, i.e., sequences of states. In other words, correct analysis of the behavior after combustion requires to prevent the transition from the combustion, `(pressure ?cylinder) >= combustion-pres`, to the path ended in stop state, `(pressure ?cylinder) < nonstop-pres` during the following

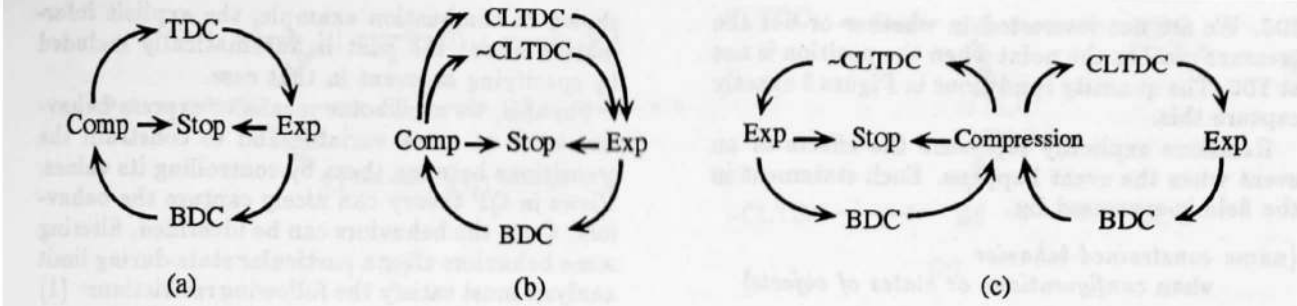

Figure 2: The behavior of a piston-cylinder. These show abstract behaviors rather than actual states in envisionments. While CLTDC represents "combustion lasts at TDC", i.e., (pressure ?cylinder) >= combustion-pres, ~CLTDC represents the state (pressure ?cylinder) < combustion-pres.

expansion, which cannot be done by local filtering. In Figure 2b, the different statuses at TDC are captured while the different expansions are not.

Figure 2c shows the desirable envisionment which captures the accurate behavior of a piston-cylinder. This has more states than Figure 2b and distinguishes the behaviors with and without combustion. Since we do not give any constraint during the compression period after combustion, the behavior during the period are the same as the behavior without combustion.

Modeling Global Constraints

As the example problem shows in the previous section, we need some means to filter next behavior from a particular state. To include this filtering in modeling process, underlying qualitative physics should provide some means to capture behaviors and to give constraints to the behavior. We use *Qualitative Process* (QP) theory (Forbus, 1984) for our basis since it provides the language for conditionalized descriptions: *view* and *process*. View allows us to capture interesting feature, i.e., behavior, by specifying conditions of related individuals. It can also express the constraints in the conditions. On the other hand, process provides the means to control transitions, as described later.

Event

In this section, we introduce a notion of *event* to model the constraints in the behavior after some point. It allows the effect of the state to be explicitly reflected in following paths. Intuitively, it is used to memorize some point, i.e., some special event, and its effects explicitly since they are lost during simulation. This is defined in three parts: *individuals*, *quantity conditions*, and *relations*. Whenever all individuals in *individuals* exist and all conditions in *quantity conditions* are satisfied, the event is active. *relations* contain the constraints about following behavior after the event happens. Figure 3 shows how this is represented

```
Defevent CLTDC ;;; Combustion lasts at TDC
Individuals
 ?pst  :type piston
 ?cyl  :type cylinder
       :conditions (part-of ?cyl ?pst)
 ?crs  :type crankshaft
       :conditions
       (connected ?pst ?crs)
 ?c-g  :type contained-gas
       :form (c-s ?sub GAS ?cyl)
QuantityConditions
 (pressure ?c-g) >=
    (combustion-pres ?pst ?cyl ?crs)
      when (position ?crs) = TDC
Relations
 CLTDC-EXP (pressure ?c-g) >
   (nonstop-pres ?pst ?cyl ?crs)
     when (not ((velocity ?pst) > 0))
```

Figure 3: An example of defevent.

in the example of combustion at TDC.

Quantity conditions consists of a set of statements and each statement is expressed by

(*conditions* when *configurations or states of objects*)

It says the event occurs if *conditions* are true in some *configurations* or *states of objects* (e.g., the pressure in a cylinder is increasing, decreasing, or not changing). Keyword "when" in quantity conditions is also used to guide the negation. Suppose the quantity conditions are written in the way done in QP theory. (We simply use combustion-pres instead of (combustion-pres ?pst ?cyl ?crs) as in the previous section.)

```
QuantityConditions
   (pressure ?c-g) >= combustion-pres
   (position ?crs) = TDC
```

This representation implies CLTDC does not happen if at least one of these conditions is false. However, what we want to express is whether or not CLTDC happens, depending on whether the pressure of a cylinder reaches combustion-pres at

TDC. We are not interested in whether or not the pressure reaches the point when the position is not at TDC. The quantity conditions in Figure 3 exactly capture this.

Relations explicitly represent the effects of an event when the event happens. Each statement in the field is expressed by

(*name constrained-behavior*
when *configurations or states of objects*)

The first part of every statement represents the name of the relation. The remaining parts show the constrained behavior after the event. For instance, in CLTDC, the relation CLTDC-EXP represents the constrained behavior during expansion, i.e., `(not ((velocity ?pst) > 0))`, after combustion. (When the velocity of the piston is non-negative, the gas in the cylinder is expanded.)

If the behavior after the event did not happen (i.e., "no event") is also constrained, this is described by a keyword ":neg". Otherwise, as in CLTDC-EXP, it is not specified as a default. Suppose when the pressure at TDC is less than `combustion-pres` at TDC, the pressure during following expansion cycle is not greater than `nonstop-pres`. (We simply use `nonstop-pres` instead of `(nonstop-pres ?pst ?cyl ?crs)` as in the previous section.) This constraint is then added to relation field as follows:

```
CLTDC-EXP
  (pressure ?c-g) > nonstop-pres
      when (not ((velocity ?pst) > 0))
  :neg (pressure ?c-g) <= nonstop-pres
```

Filtering Behaviors using Event

If we assume underlying simulator finds every correct set of states and local state transitions, what we need for dealing with event is simply to prevent the transitions to the states which lead to the impossible behavior (Figure 4). The remaining part of the envisionment should not be affected by this filtering. For example, in Figure 4, the path from *s1* is a part of the behavior, even though the transition from s0 to s1 is illegal. Thus it must be protected from the filtering.

Limit analysis is a process to compute every state change by derivative relations of variables and continuity. Thus, if the continuity condition in unwanted transition can be automatically violated, current limit analysis can be used for filtering behavior. We do this by introducing an extra variable whose continuity breaks down in the transitions since limit analysis cannot prevent the transitions with existing variables. Intuitively, this variable is used as the tag which informs afterwards whether the event happened or not at some point. Since each state sometimes cannot include sufficient information to predict the next state, as shown in combustion example, the explicit information about the past is automatically included by specifying an event in that case.

For this, we need some means to express behaviors with an extra variable and to constrain the transitions between them by controlling its values. Views in QP theory can nicely capture the behaviors. Once the behaviors can be identified, filtering some behaviors after a particular state during limit analysis must satisfy the following restrictions: (1) If the behavior is constrained, the transitions from the state should be made only to the paths which describe the behavior. The transitions to others should be prevented (Figure 5a). (2) If the behavior is not constrained, the transitions to every possible path should be made (Figure 5b).

Implementation

In this section, we show how global constraints expressed by `defevent` are filtered in the framework of QP theory, using the combustion model. A `defevent` is translated into several views with an extra variable and filtering is done based on the variable.

Generating Views

At first, we need to distinguish whether or not an event happens. This is captured by generating two views for each case with an extra variable. The name of the event and the name prefixed with ~ are used for the names of the two views. For example, two views, i.e., CLTDC and ~CLTDC, are introduced for event CLTDC. The extra variable is set to positive when the event happens and set to non-positive otherwise. The following shows parts of the views CLTDC and ~CLTDC, respectively. The individuals of both views are same as the individuals of the event CLTDC. An extra variable `CLTDC-tag` is introduced with different values.

```
CLTDC:
QuantityConditions
 (position ?crs) = TDC
 (pres ?c-g) >= combustion-pres
Relations
 (CLTDC-tag ?pst ?cyl ?c-g) > 0

~CLTDC:
QuantityConditions
 (position ?crs) = TDC
 (pres ?c-g) < combustion-pres
Relations
 (CLTDC-tag ?pst ?cyl ?c-g) <= 0
```

Two different views are also generated for each relation: one for the subsequent behavior after the event and the other for the behavior after no event. The individuals of both are also same as those of the event. The quantity conditions of both include

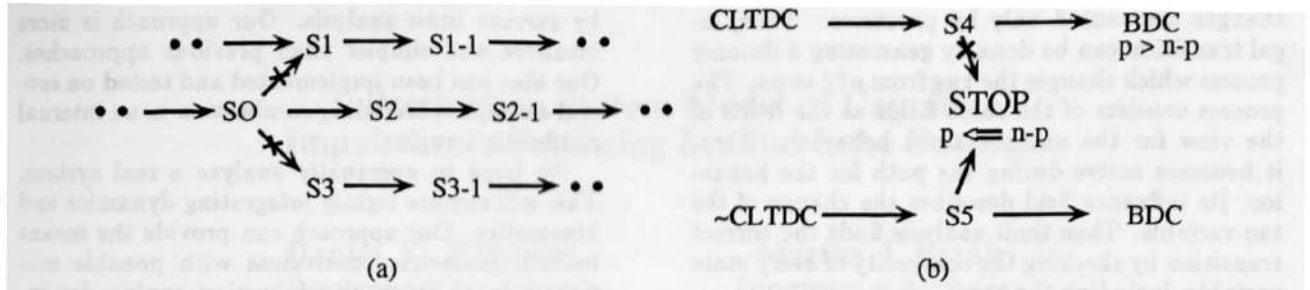

Figure 4: Filtering behaviors. (a) There are three possible paths from s0. Suppose only the path which starts with $s2$ should be selected from $s0$ due to the constraint by a structure description. It requires to prevent the transition to $s1$ and $s3$. (b) The transition from $s4$ to STOP should be prevented. (n-p represents nonstop-pres.)

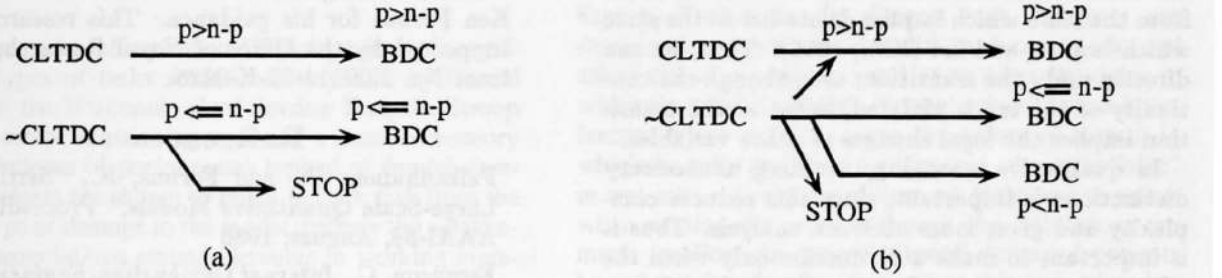

Figure 5: The behavior from TDC to BDC in a piston-cylinder. In (a), expansion after both CLTDC and ~CLTDC are constrained and different from each other. In (b), the expansion after ~CLTDC is not constrained.

the configurations or some states of objects specified after **when**. In addition to these, the former includes the assumption that the tag is positive while the latter includes the assumption that the tag is non-positive. The relations of the views constrain the behavior and are taken from the relations of the event. The name of the relation and the name prefixed with ~ are used for the names of the views. The followings show parts of view CLTDC-EXP and ~CLTDC-EXP, respectively. Since the behavior during expansion after ~CLTDC is not constrained, the relation field in ~CLTDC-EXP is left empty. If it is also constrained, the constrained behavior will be described in the field.

```
CLTDC-EXP:
QuantityConditions
 (not ((vel ?pst) > 0))
 (CLTDC-tag ?pst ?cyl ?c-g) > 0
Relations
 (pres ?c-g) > nonstop-pres

~CLTDC-EXP:
QuantityConditions
 (not ((vel ?pst) > 0))
 (CLTDC-tag ?pst ?cyl ?c-g) <= 0
Relations
 ()
```

Transition

Once these views are generated, the correct behavior for each case are selected by controlling the continuity of the extra variable during limit analysis.

The first requirement—when the behavior is constrained—is easily solved since the views generated by an event are manipulated to set the tag variable with this in mind. The continuity of the tag variable in the illegal path is violated. On the other hand, the continuity of the variable to the legal path is maintained (Figure 5a).

In case of the second requirement—when the behavior is not constrained, it includes the transition between two states which have different tag values. Note that this case happens only to the behavior after no event since an event is introduced to give constraints to the behavior after the event. If the behavior after no event are not constrained, it implies the transition to the constrained path after the event, since the path is one of the possibilities from no event (Figure 5b).

This transition requires to change the tag from **off** to **on**, i.e., from **tag** <= 0 to **tag** > 0. Thus, we need some means to connect the states which have different values of the tag. There are two approaches to handling this:

Make as a continuous change: In QP theory,

changes are caused only by processes. Thus, legal transition can be done by generating a dummy process which changes the **tag** from **off** to **on**. The process consists of the same fields as the fields of the view for the unconstrained behavior. Thus, it becomes active during the path for the behavior. Its influence field describes the change of the tag variable. Then limit analysis finds the correct transition by checking the continuity of every state variable, including the tag.

Make as a discontinuous change: Even though we assume qualitative physics deals with only continuous changes, reasoning about discontinuous change is important to explain many phenomena. Basically, a discrete change is made by specifying the add-list and delete-list. The transition is made from the state which implies delete-list to the state which implies add-list (Kim, 1992). Thus, we can directly make the transition, even though the continuity of the tag is violated, as far as the transition implies the legal changes of other variables.

In qualitative reasoning, avoiding unnecessary distinctions is important, since this reduces complexity and gives more abstract analysis. Thus it is important to make a distinction only when the effects of an event result in qualitatively different behaviors. Unless the effects clearly make differences, such as the compression period in `CLTDC`, we do not consider whether or not the event happens. In other words, the extra variable for tag is not used. `CLTDC-tag`, for instance, is not considered during the compression period. Since the transition between the state with tag and the state without tag does not violate the continuity due to the tag variable, no extra work is done to connect the influenced behavior to the subsequent uninfluenced behavior. For instance, the different paths during expansion are connected to one path for compression in `CLTDC` (Figure 2c) without extra manipulation. Continuity checking between the state variables is enough.

Though multiple events are defined, they can be handled without any difficulty since they are manipulated by independent additional variables. Thus, there is no interference between the events.

Using this approach, `QPE` (an implementation of QP theory) produces the envisionment for several examples including an internal combustion engine (Figure 2c), a spring block, and a neon bulb.

Discussion

In this paper, we have shown how simulation technique combined with qualitative information fails to capture global behaviors through qualitative simulation. A model of an internal combustion engine has been used to illustrate this. We extend current simulation methods by providing the means to constrain subsequent behavior after some event. We describe how the constraints are filtered by current limit analysis. Our approach is more intuitive and simpler than previous approaches. Our idea has been implemented and tested on several examples, including combustion in an internal combustion engine.

We hope to eventually analyze a real system. This will require tightly integrating dynamics and kinematics. Our approach can provide the means to link geometric constraints with possible motions. In an internal combustion engine, for instance, the interaction between the behavior during power stroke, and the geometry of a piston-cylinder could be captured by this technique. Our technique is one step towards that final goal, by providing more accurate prediction of behaviors.

Acknowledgements: I would like to thank Ken Forbus for his guidance. This research was supported by the Office of Naval Research, Contract No. N00014-85-K-0225.

References

Falkenhainer, B., and Forbus, K., "Setting up Large-Scale Qualitative Models," Proceedings of AAAI-88, August, 1988

Ferguson, C., *Internal Combustion Engines*, John Wiley & Sons, 1976

Forbus, K., "Qualitative Process Theory," *Artificial Intelligence*, **24**, 1984

Forbus, K., "Qualitative Physics: Past, Present, and Future" in Shrobe, H. (Eds.), *Exploring Artificial Intelligence*. Morgan Kaufmann Publishers, Inc. 1988

Hopcroft,J. and Ullman, J., *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley Publishing Co., 1979

Kim, H., "Qualitative Reasoning about Discontinuous Changes" submitted to PRICAI-92.

Kuiper, B., "Qualitative Simulation," *Artificial Intelligence*, **29**, 1986

Lee, W. and Kuiper, B., "Non-Intersection of Trajectories in Qualitative Phase Space: A Global Constraint for Qualitative Simulation," Proceedings of AAAI-88, August, 1988

Struss. P., "Global Filters for Qualitative Behaviors," Proceedings of AAAI-88, August, 1988