Subsystem Reduction for Qualitative Simulation

Silvia B. González-Brambila, Eduardo F. Morales

UAM-Azcapotzalco *ITESM-Cuernavaca*
sgb@correo.azc.uam.mx *eduardo.morales@.itesm.mx*

Abstract

Qualitative simulation (QS) is an area of artificial intelligence that represent continuos and discrete aspects like space, time and quantity with little information, and makes inferences using symbolic data to represent physical quantities. Tradicionally, QS uses a global state-based representation to represent the behavior of the system. To qualitative simulate a system, some initial values are normally given, along with qualitative differential equations (QDEs). With this information, a qualitative simulation algorithm evaluates all possible combinations of qualitative values and filters out inconsistent states considering qualitatives constraints. This is a combinational process which normally requires exponential time. In this paper, a new algorithm is described which simulates individual components independently and joins their behavioral graphs together until a global behavioral graph is obtained. This algorithm achieves substantial reductions in time and it is polynomial with respect to the number of components. It is shown how with this algorithm is possible to simulate industrial plants of hundreds of variables within a few minutes.

1. Introduction

Communicating knowledge, in verbal of written form, is an important human learning activity. In engineering, explaining how a particular device works is relevant to engineering students, designers and operators of industrial plants. These explanations, however, are normally given from a particular point of view and without considering the user's particular needs. Explanations related to a particular device can be given from different perspectives depending on different needs. An engineer may be interested in knowing the causal dependencies between different state variables. She may be interested in observing how the state variables evolve over time, or what is the main function of a particular device. Her interests may focused on particular state variables and/or particular subsystems.

We have developed a system called AGE (Automatic Generation of Explanations) [3, 13] which automatically produces explanations of engineering devices in natural language considering different perspectives. The type of explanations produced by AGE are causal, behavioral and functional, considering user selected state variables and subsystems.

AGE's architecture is shown in Figure 1. Given a qualitative model of a particular device, AGE generates a global flow sheet that is used for functional explanations, obtains causal dependencies from the qualitative model to produce behavioral explanations, and uses this simulations with functional analysis to produce functional explanations.

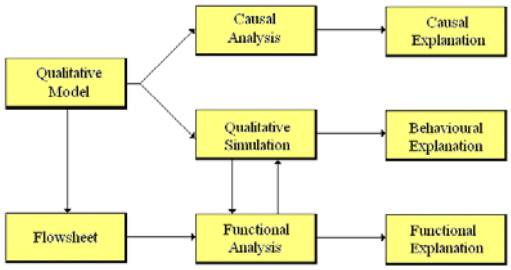


Figure 1 AGE's architecture

In order to create a qualitative model of a chemical process, the user employs a graphical interface to join engineering components, such as pumps, valves, tanks, tubes, stoppers, reactors, etc., taken from a library [8] of components. Each element of this library is associated with a qualitative model. We adopted qualitative models because they allow predictions about the behavior of the system in the absence of exact quantitative information and they tend to express more closely the type of explanations we are interested in (see Figure 2). A complete system is constructed by connecting individual components and producing a general model through a process known as compositional modeling [9].

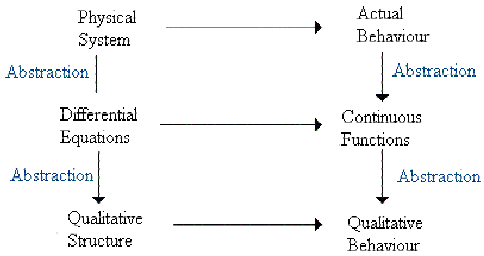


Figure 2 Qualitative models are abstractions of quantitative models

Chemical processes are useful to study problems in existing plants and for the design of new installations, are essential before working in the material balances and for the improvement of the equipment and is a well defined area.

Qualitative simulation is very important in AGE, because functional and behavioral explanations are generated from it. AGE produces a behavioral graph (a graph where each node represent a particular qualitative

state and links represent time sequences) using a re-implementation of QSIM [6], but the algorithm can be very inefficient for large systems. For instance, Catino in [4] simulated a nitric acid plant with 217 variables and 287 constraints in a 224 Mb Sun SparcStation ELC using QSIM. In our re-implementation of QSIM we were not able to simulate a chemical plant with 88 components after 1 day of CPU time (Intel Pentium III 993 MHz, 256 MB). In this paper, we proposed an algorithm which divides each system into smaller subsystems considering design principles of process engineering. Individual components are simulated qualitatively from which their behavioral graph are produced. The algorithm joins this graphs and continues until a complete simulation is obtained.

This paper is organized as follows. Section 2 gives an overview of QSIM and presents some execution times of components and simple systems. In section 3 the subsystem reduction algorithm is described. Section 4 explains how using QSIM for individual components simulation and the subsystem reduction algorithm it is possible to find the behavior for large systems. Section 5 presents some of experimental results using this approach. Related works are presented in Section 6. Finally, conclusions and future research directions are given in Section 7.

2. QSIM

QSIM is an approach to qualitative simulation that uses qualitative differential equations (QDE) to represent a system. QDE are relaxed versions of ordinary differential equations [6]. QSIM predicts the possible behavior set of a QDE.

A QDE model is qualitative in two senses. First, the values of variables are described in terms of their ordinal relations with a finite set of symbolic landmark values. Second, functional relations may be described as monotonic functions [7]. Landmark values are the "natural joints" that break a continuous set of values into qualitatively distinct regions. A landmark value is a symbolic name for a particular real number, whose numerical value may or not be known. It serves as a precise boundary for a qualitative region.

QSIM starts with a QDE and a qualitative description of an initial state. Given a qualitative description of a state, it predicts the possible qualitative state descriptions that can be direct successors of the current state description. Repeating this process produces a graph of qualitative descriptions, in which the paths starting from the root are the possible qualitative behaviors. The resulting behavior graph can still be quite large.

The main step in the QSIM algorithm is generate all the successor states given a state. The successor generation algorithm performs the following steps [6]:

1. Domain restriction
2. Node consistency
3. Arc consistency
4. Exhaustive search
5. Filtering

To guarantee that all possible behaviors are predicted, it is required that all possible qualitative value transitions are predicted, and that the combinations of qualitative values are only deleted when they are inconsistent. The exhaustive nature of the QSIM simulation can produce excessive running times.

When a qualitative model of a component is defined is very important to analyze the possible landmarks of each variable, the initial conditions and the constraints with the corresponding values because the execution time depends on all of these factors.

Table 1 shows the number of variables and constraints of some chemical components. The average execution time considers 10 simulations in a Intel Pentium III 993 MHz, 256 MB.

Component	# variables	# constraints	# landmarks	# nodes behavior graph	Average time (msec)
Tank	4	4	9	10	69.1
Valve	5	6	15	4	1837.7
Separator	19	19	40	137	12087.3
Reactor	9	9	21	32	1672.5
Flash1	14	15	38	2	1978
Flash2	14	14	38	32	201430.7

Table 1 Number of variables and constraints of some components

The execution time not only depends on the number of variables and constraints. For example, the reactor variables and constraints (9) is greater than the valve (5 and 6, respectively), but the average time of the first (1672.5 milliseconds) is shorter than the second (1837.7).

2.1 System composition

In systems composed of several units, it is convenient to use the component-connection approach [6] to construct complete system. The complete specification of a physical component in AGE, requires, besides a qualitative model, the semantic meaning of each state variable and all of its landmark values, as well as its input/output variables (called, *terminalIn* and *terminalOut*) in order to connect it with another component. Each component or unit is also associated with a meaningful name to the user and the name of the substance that it is carrying.

The relation between the number of variables, landmarks and behavior graph nodes is not simple. But in the majority of cases the initial values can significantly reduce the execution time.

For example, in the case of a system composed by a mixer and a reactor, called *MR*, the number of equations is 15 and the variables are 14. If initial values are given for mixer inflow, (q, θ), the average execution time is 44,446 milliseconds. If the mixer outflow is also given fixed values then the average time is reduced to 2003.8 milliseconds In both cases the behavior graph has 18 nodes.

3. Subsystem reduction

For the subsystem reduction process, principles from classical design in process engineering [1], [2] were considered, where the component's system are collocated in accordance of their type. This is, in a new design the first components that are considered are the reactors, then separators, energy transfers units, material management units and lastly, the rest of the equipment.
In our case, units are grouped together using the priority list, showed in Table 2. For example, it is common to mix two or more substances (mixer), heat the product (heater) and finally introduced the product into a reactor. This three units (mixer, heater and reactor) can be merged in one subsystem whose purpose is to react.

Prio	Unit type	Examples
1	Reactor	All types of reactors
2	Separator	Filters, evaporators
3	Energy transfer	Heaters, coolers
4	Material management	Pumps, mixers, compressors
5	Storage and control	Tanks, valves

Table 2 Priority of unit type

Two units, *A* and *B*, can be merged in a subsystem *A*-*B* if *A* is adjacent to *B*, *A* has a priority equal or smaller than *B*, and *A* is topological smaller than *B*.
In the topological sort each node is associated with a vertex and there is a directed edge from node *x* to node *y* if *y* cannot start until *x* is finished.
A large system can go through several grouping processes, so this is an iterative process. After the first unit is selected the system tries to group it with its neighbors. A unit is considered first if it has more external substances, lower priority type and is first in the topological sort of all system. For more details of this algorithm see [3].
Consider the flow sheet of the hydrodealkylation of Toluene shown in Figure 3. Grouping the units result in the systems shown in Figures 4 and 5. Fig. 4 is the first iteration of the algorithm, the reactor systems groups the *compressor*, *pump*, *mixer* and the *reactor*; the *separation system* adjacent to the *reactor system* contains the *flash* and the *separator*, the *separator system 1* groups *valve1* and *separtor1* and *separator system 2* groups the rest of the units. Note that the cycles in the reduced subsystems are conserved.
Algorithm 1 shows the main steps to reduce subsystems, it select the initial node without consider the substances, this is done later. When two or more units are grouped together in one subsystem, they are inserted into a list in order to save this information that is used later.

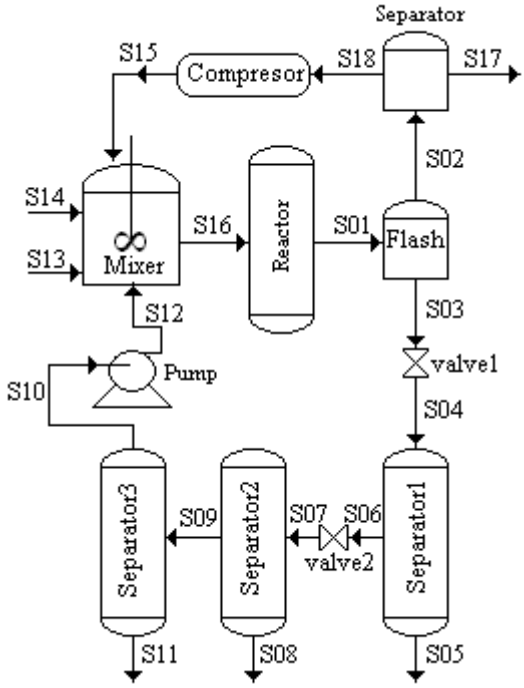


Figure 3. Hydrodealkylation of Toluene

```
FlowDiagram SubsystemReduction ()
{
   initialNode ← select begin unit
   FlowDiagram newDiagram
   insert initialNode in newDiagram
   reduceUnits (initialNode, newDiagram)
      //insert links in accordance to
                    //the  previous flow diagram
   newDiagram.putLinks (this)
      //insert substances in accordance to
      //the previous flow diagram
   newDiagram.createSubstances (this)
}
```

Algorithm 1. Subsystem reduction

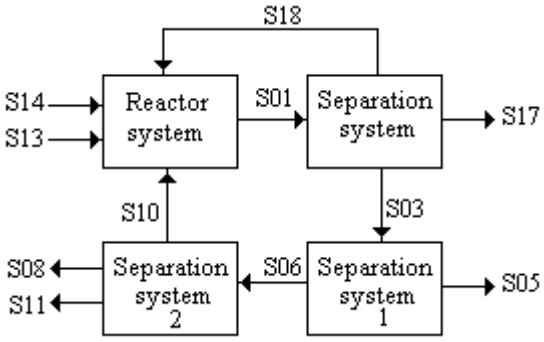


Figure 4. Hydrodealkylation of Toluene first iteration of de subsystems reduction

Figure 5 is obtained from Figure 4; here Reactor system, Separation system and Separation system 2 are grouped in one. So with this reduction there are two subsystems: reactor and separation, and nine substances.

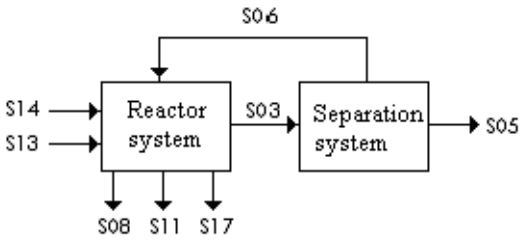


Figure 5. Subsystems of the Hydrodealkylation of Toluene

In Figure 6 all subsystems are grouped into one, where input and products substances are only considered.

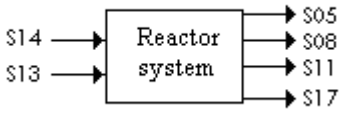


Figure 6. Last iteration of the subsystem reduction of the Hydrodealkylation of Toluene

4. Simulation by components

The proposed algorithm is based on simulating individual behaviour of each component in the system using QSIM. This process produces behavioural graphs for each component. To group two different behaviour nodes, both nodes must correspond in their time tag and their qualitative values of corresponding *terminalOut* and *terminalIn* values must be equal. Then the individual behaviour graphs are grouped in subsystems, using the subsystem reduction algorithm.

So the main idea is to divide the system in subsystems in different abstractions levels, use QSIM to simulate each individual component and obtain their respective behaviour graph considering different abstractions levels. The behaviour graphs are grouped by the connection nodes in the subsystems and only when all their behaviour values are equals. Even though in the component level is possible to generate more states than necessary, they will be eliminated during the union process and significant reductions in execution time are obtained.

To group two different behaviour graph nodes, consider that both must correspond in time and the union qualitative values must be equal (*terminalIn* or *terminalOut*).

For example, suppose we have a unit *A* with a behaviour graph *g1* with an initial node "*a*" with set values {*v_a*}, where {*v_a*} corresponds to all the qualitative variable values of unit *A* at time *t0*. Now suppose we have a unit *B* with behaviour graph *g2* and an initial node "*b*" with set values {*v_b*}. In addition, consider *A* to be before unit *B* in the topological sort of the flow sheet. Since *a* and *b* are initial states they both occur at time *t0*. If the *terminalOut* qualitative values in {v_*a*} are equal to *terminalIn* qualitative values in {v_*b*}, then they can be merge into one state. This new state contains all the values in {v_*a*} and all values in {v_*b*}, except those in the intersection of *terminalOut* in *A* and *terminalOut* in *B*, that are considerer only once. The remaining nodes are merge in a similar form (see Figure 7).

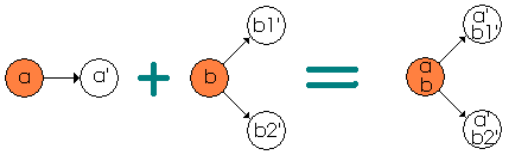


Figure 7 Joint two-behavior graphs

A final node is considered quiescent, if the variable values are the same in the next time until a transition occur, these nodes are called perdurables. In the case of merging, a behaviour graph with only one state (with a value set {*a*}) with another graph with several nodes, the single node needs to be mapped with all the nodes of the other behavioural graph (see Figure 8). So the mapping process is in general *1* to *N*, because one node can be consider more that one.

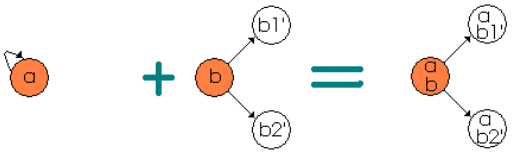


Figure 8 Considering a perdurable node

4.1 Example

Consider a system with a mixer and a reactor, called *MR*. Suppose that the connection variables are only the outflow of the mixer (*M-outflow*) and the inflow of the reactor (*R-Fin*). Suppose that the input flow of the mixer are constant in order to reduce its possible behaviours. The behaviour graph of each component is presented in Tables 3 and 4, respectively. Table 4 shows part of the behavioural graph represented in list form, some of the initial states are *R0*, *R1*, *R2*, *R3*, *R4*, *R5*, *R6*. The QSIM simulation produced 32 states.

State	Adyacents
M0	-

(a) Behaviour graph

Variable	State M0
M-amount	medioLleno,θ
M-outflow	q, θ
M-netflow	0, θ
M-inflow	q, θ
M-Qin1	q, θ
M-Qin2	q, θ

(b) Values of the state

Table 3 Mixer behavior

State	Adjacents
R0	-
R1	-
...	
R5	-
R6	17, 18
...	
R13	19, 20
...	
R17	13, 14, 8, 9, 10
...	
R19	7, 12, 10
...	

(a) Segment of the behaviour graph

Variable	State R0	State R6	State R13	State R17
R-dif	0, ↑	0,↑	dif, θ	<0, dif>,↑
R-Ca	0, ↑	0, ↑	c, θ	<0, c>,↑
R-Fin	q, θ	q, θ	q, θ	q, θ
R-Fout	0, ↑	0, ↑	q, θ	<0, q>,↑
R-Cb	c, θ	c, θ	c, θ	c, θ
R-k	k, θ	k, θ	k, θ	k, θ
R-kCa	0, ↑	0, ↑	kc, θ	<0, kc>,↑
R-MkCb	0, ↓	0, ↓	-kc, θ	<-kc, 0>, ↓
R-D	0, ↑	0, θ	0, θ	0, θ

(b) Values of some states

Table 4 Segment of the reactor behaviour

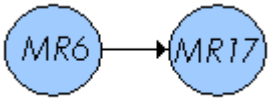
Initially consider the state *M0*, the only mixer initial state, and the reactor initial state *R6*. With this two states we construct a new one (*MR6*) of the behaviour graph of *MR* system. This is possible because *M-outflow* and *R-Fin* has the same value (*q*, θ). First column of Table 5 shows the values of this state.

Variables	MR6	MR17	MR13
M-amount	some,θ	some,θ	some,θ
M-outflow	q, θ	q, θ	q, θ
M-netflow	0, θ	0, θ	0, θ
M-inflow	q, θ	q, θ	q, θ
M-Qin1	q, θ	q, θ	q, θ
M-Qin2	q, θ	q, θ	q, θ
R-dif	0,↑	<0, dif>,↑	dif, θ
R-Ca	0, ↑	<0, c>,↑	c, θ
R-Fin	q, θ	q, θ	q, θ
R-Fout	0, ↑	<0, q>,↑	q, θ
R-Cb	c, θ	c, θ	c, θ
R-k	k, θ	k, θ	k, θ
R-kCa	0, ↑	<0, kc>,↑	kc, θ
R-MkCb	0, ↓	<-kc, 0>, ↓	-kc, θ
R-D	0, θ	0, θ	0, θ

Table 5 Some states of the *MR* system

Next we consider state *R17*, because it is adjacent to *R6*. With *M0* and *R17* another new node of the behaviour graph is constructed. In this case *MR6* and *MR17* must be adjacent, so the behaviour graph of the system is constructed with these nodes linked together (see Figure 9 (a)). In the construction of this new state, *M0* is consider perdurable.

The construction process of the behaviour graph continues with the adjacents of M0 and R17, which are *M0* (perdurable) and *R13*, respectively. This new states are merge and a new state *MR13*, is created, adjacent to *MR17* (see Figure 9 (b)).



(a) First step

(b) Second step

Figure 9 Constructing behaviour graph of *MR* system

This process continues until all nodes are visited.

The algorithm is $O(n^2)$, without considering the QSIM simulation of individual components, because is a depth first in which the nodes of the second graph can be visited more than once.

Although more states per component may be generated at the simulation time, all the inconsistent states are removed by this merging procedure. We have observed in all of our experiments that our merging procedure produces only qualitatively consistent behavioural graphs, and as part of our future work, we are working on a formal proof of this.

By joining individual behavioural graphs of single components, we are able to substantially reduced the computational time required by QSIM.

5. Evaluation

All the experiments in this section was done using Intel Pentium III 993 MHz, 256 MB.

Execution time of the *MRF* system (presented in section 2) using the this algorithm. The time is considerably reduced (it uses 5.6% of the time used by QSIM), and most of this time is consumed in the individual components simulation. Another advantage of our approach, is that we can store behavioural graphs of individual components and re-use them in other systems. This again, can produce significant time reductions.

AGE has been tested on a wide variety of engineering systems ranging from single components to industrial plants. For example, the average execution time of the acyclic process [12] of Figure 10 is 57,893 milliseconds, and the average execution time of simulate the cyclic process [12] of Figure 11 is 76,579 milliseconds.

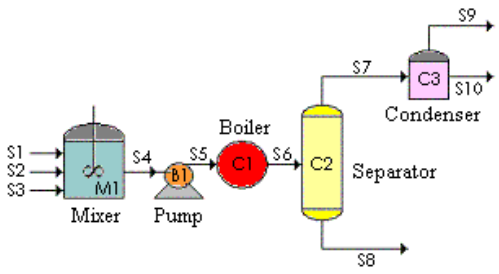


Figure 10 Acyclic process

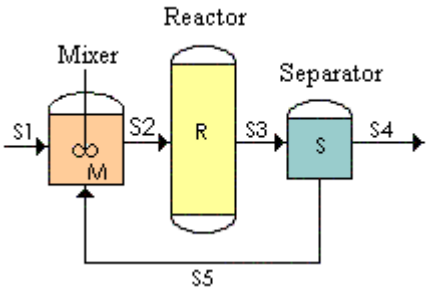


Figure 11 Cyclic process

The average time of the Hydrodealkylation of Toluene of Figure 3 is 60,587 milliseconds.
As another examples, the normal paraffin extraction [5] (see Figure 12) takes 162,688 milliseconds and the production of nitric ammonium (see Figure 13) takes 97,431 milliseconds.

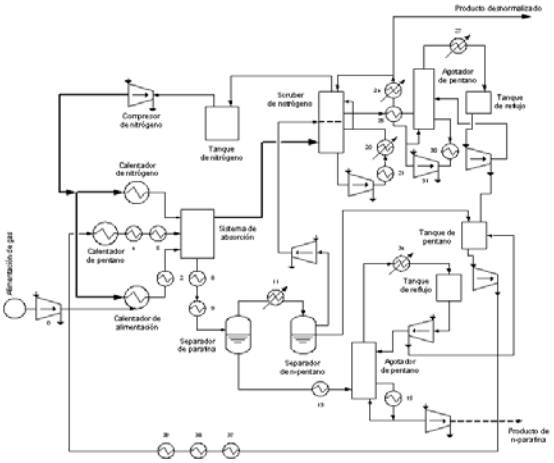


Figure 12 Normal paraffin extraction

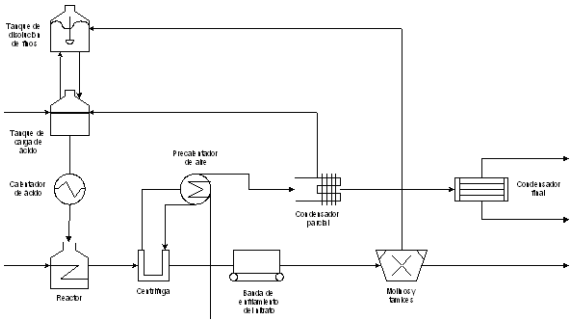


Figure 13 Production of nitric ammonium

We illustrate the reductions achieve by our algorithm and the capabilities of AGE qualitative simulation algorithm with the Empress plant [5]. This plant, showed in Figure 14. It has 132 flows, 88 units and are 638 variables in its qualitative representation. The average execution time is 404343.4 milliseconds (6.74 minutes). The average time for simulate individual components is 129868.7 milliseconds (2.16 minutes). This is a huge reduction in processing time, considering that we were not able to simulate this plant with our re-implementation of QSIM after 1 day of simulation. This time is very reasonable considering the size of the plant.

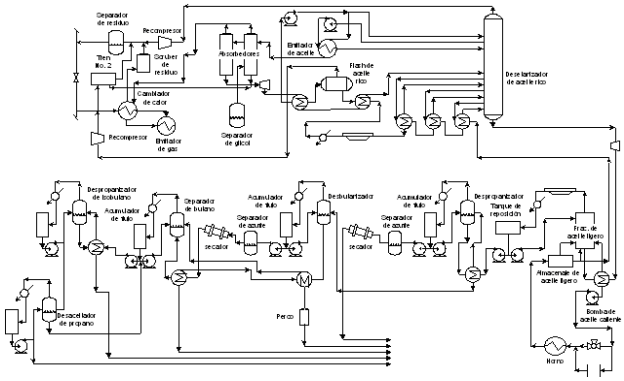


Figure 14 Empress plant

6. Related Works

Several improvements have been suggested on QSIM, however, must of them have been oriented towards more efficient filtering mechanism and extensions to combine it with numerical data [7], [6], and nothing has been done on component decomposition.
They however, do not reduce the combinational process that QSIM follows to produce subsequent qualitative states. So the related works are only a few.
Catino in [4] simulated a nitric acid plant with 217 variables and 287 constraints in a 224 Mb Sun SparcStation ELC using QSIM. The approximated time was 12 hours and some components could not be simulated.
In terms of dividing systems into subsystems, Chong [10] find the system functionality of a chemical processes. The unit representation in based in

Chandrasekarans works [11] and uses a functionality precedence to group immediately neighbors. This works is similar to the subsystem reduction algorithm presented but Chong work is not able to consider cycles, that are very important to chemical engineers.
DecSIM [14] is a model decomposition and simulation algorithm that uses a divide and conquer approach. Variables with the system are partitioned into components so that closely related variables are constrained with the same partition describing the relationships between variables with partition. Each component is viewed as a separate system and is simulate using a state-based representation limited to the variables within the component. Interactions between components are reasoned about as needed to constrain each component. Two types of variables are constrained within each sub-model, within-partition and boundary. DecSIM uses QSIM.

7. Conclusions and future work

QSIM simulation can be very inefficient for large systems due to its combinational components to generate possible qualitative states. In order to scale-up QSIM to larger devices, AGE divides each system into subsystems at different abstraction levels. This algorithm is used to simulate each individual component from which their behavioral graph are obtained. Graphs of contiguous components are joined together using connecting nodes with the same state values and corresponding time stamps. The same process continues (without any further simulation) for contiguous subsystems until a global behavioral graph is constructed.
With this algorithm, it is possible to simulate chemical plants with a large number of states in a reasonable time.
Of course, an important disadvantage of this approach is the strong dependence of the component library, because representation of all components need to by correct, sound, have compatibles *terminalIn* and *terminalOut* variables, and each of this have compatibles landmark spaces. For this reason, components designers needs to have knowledge of processes engineering and is very desirable automatic tools for this task.
The generated behavior graphs were used in a system to generate chemical process explanations in natural language with success.
As part of our future work, we would like to produce a formal proof that our reduction algorithm is able to produce only qualitative consistent behaviors (sound) and that it produces all the qualitative consistent behaviors (complete). Also we plan try in other domains such as electrical and mechanical.

8. References

[1] Beltrán V., M., Delgado N., Ma. de Lourdes, Quintana D., M.B. Guadalupe, *Introducción a la ingeniería química*, Universidad Autónoma Metropolitana, 1997

[2] Douglas, J. M., *Conceptual design of chemical processes*, Mc Graw Hill, 1988

[3] Morales, E., González, S., *Generation of Explanations of Chemical Processes*, Proc. of the 2nd. IASTED International Conference ARTIFICIAL INTELLIGENCE AND APPLICATIONS (AIA2002), pp. 110-115, Benalmádena, Spain, September 9-12, 2002

[4] Catino, A. C., *Automated modeling of chemical plants with application to hazard and operability studies*, Ph. d. thesis, 1993

[5] Himmelblau, D. M., Bischoff, K. B., *Análisis y simulación de procesos*, Reverté, 1992

[6] Kuipers, B., *Qualitative simulation, modeling and simulation with incomplete knowledge* (The MIT Press, Cambridge, Massachusetts, 1994).

[7] B. Kuipers, *Qualitative simulation*, Robert A. Meyers, Ed., Encyclopedia of Physical Science and Technology, 3rd. ed. NY Academic Press, pp. 287-300, 2001.

[8] T. R. Gruber, G. R. Oisen, *An ontology for engineering mathematics*, *Proc. fourth international conference on principles of knowledge representation and reasoning,* San Mateo, CA., 1994, 258-269.

[9] B. Falkenhainer, K. Forbus, *Compositional modelling: finding the right model for the job*. *Artificial intelligence 51*, 1991, 95-143.

[10] Chong, T. T., *Derivation and use of function in the design of chemical processes,* MSc Information Technology, University of Edinburgh, 1995.

[11] Chandrasekaran, B., Josephson, J. R., *Representing function as effect: assigning functions to objects in context and out*, Working notes of the AAAI-96 Workshop on modeling and reasoning with function, August 4, 1996, Portland, OR, 1996

[12] Felder, R. M., Rousseau, R. W., *Elementary principles of chemical process,* John Wiley & Sons, Inc., 3ed., pp. 556, 2000

[13] González-Brambila, S., Morales, E., *Generation of Explanations of Chemical Processes: a demo*, XI Congreso Internacional de Computación, Avances en Ciencias de la Computación e Ingeniería de Cómputo (CIC2002), Vol. II, pp. 333-342, Cd. de México, nov. 25 al 29 de 2002.

[14] Clancy, D. J., Kuipers, B. J., *Model Decomposition and Simulation: A component based qualitative simulation algorithm.* Proceedings from the 14th National Conference on Artificial Intelligence (AAAI-97), August, 1997.

This work is sponsor by grant 400200-5-34812-A of CONACYT for research project "*Structural Recognition in Images*".