Proceedings of the 2008 Winter Simulation Conference
S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, J. W. Fowler eds.

DISCRETE RATE SIMULATION USING LINEAR PROGRAMMING

Cecile Damiron
Anthony Nastasi

6830 Via Del Oro, Suite 230
Imagine That Inc.
San Jose, CA 95119 USA

ABSTRACT

Discrete Rate Simulation (DRS) is a modeling methodology that uses event based logic to simulate linear continuous processes and hybrid systems. These systems are concerned with the movement and routing of flow.

DRS has multiple advantages. Compared to continuous flow modeling, DRS minimizes the number of simulation calculations and posts events exactly when the model changes state. Compared to discrete event modeling, DRS makes the creation of hybrid models completely transparent. Finally, DRS provides advanced methods for routing flow.

In DRS the challenging part is to accurately calculate the movement of flow. This paper describes how we identified DRS as a global optimization problem and how we used linear programming (LP) algorithms to perform the required calculations. The use of LP provides global oversight and is a major improvement for DRS. The result is an advanced, intuitive, robust and flexible method for simulating the movement of flow.

1 INTRODUCTION

Discrete Rate Simulation (DRS) is a method for simulating continuous, rate-based flow systems and hybrid (combined continuous and discrete event) systems. In the absence of a DRS tool, these types of systems can be particularly challenging to model, since the modeler is forced to compromise how certain elements in the system are characterized. The historical approach was to use continuous or discrete event modeling.

A flow system is characterized by the movement of flow at certain rates between different locations. When a system contains processes whose delay can be described as "units per time units", the system might be considered as a flow system. The main hypothesis is that the rates remain fixed between two events. Continuous modeling has often been used to simulate flow systems. These models often employ a disbursed network of storage sites capable of holding flow with control mechanisms in between directing its movement. Typically, these control mechanisms include valves for modulating flow rates and branching functions for merging and diverging the flow streams.

Hybrid systems, while similar to flow systems in that flow moves continuously at certain rates between locations, have additional elements of discrete event behavior. For example, an oil tanker arriving at a refinery is a discrete event, while piping the tanker's payload to a holding tank is a flow process.

While some systems are best modeled strictly using either continuous or discrete event technologies, linear flow systems and hybrid systems possess behaviors that are better described using DRS. DRS predicts and schedules the exact moment the system has to calculate a new set of flow rates and also determines the appropriate rate of flow for each stream at each event. DRS technologies offer rate-based flow movement capabilities in a discrete event environment.

The first generation DRS technology was introduced by Simulation Dynamics Inc. in 1997 (Siprelle, Phelps 1997). Taking lessons learned from this ground-breaking technology, a second generation DRS technology has now been developed and is incorporated in the Rate library of the ExtendSim AT and Suite packages (Imagine That Inc.). This new DRS technology employs the use of linear programming to determine how flow rates change throughout the system over time. This paper describes the implementation and advantages of this second generation technology. Section 2 of the paper presents the need for a DRS tool; Section 3 explains the DRS approach. The last section presents the integration of linear programming in the DRS approach.

2 NEED FOR A HYBRID SIMULATION TOOL

Historically, the common methods for dynamic modeling were divided between the continuous and discrete event simulation technologies (Monsef 1997).

The continuous simulation method uses a fixed time step for defining the intervals when values in the model should be updated. The size of this time step determines not only the precision of the model but also its computational intensity; smaller time steps result in more frequent calculations. The modeler is faced with balancing the tradeoffs between accuracy and speed of execution. This simulation method is well suited for modeling systems where movement is best characterized continuously; that is, where material moves at a certain rate between different locations. For example, high speed manufacturing systems where some type of product is moving in a continuous fashion can be modeled using the continuous simulation approach.

Discrete event simulation (DES) on the other hand does not advance time at fixed intervals but rather in discrete and uneven blocks of time. In this system, once all the actionable items associated with a particular event have been completed, the discrete event clock jumps forward to the next closest future event time. This method is well suited for modeling systems where events occur at discrete and often random times. For example, queuing systems are ideally modeled using the discrete event approach.

These two simulation methods are typically mutually exclusive. Simulation tools usually require models to be either completely continuous or completely discrete event. As shown in the rest of this section, when the system to simulate is a flow system, neither the continuous nor the discrete event simulation approach provides an appropriate solution (Phelps, Parsons and Siprelle 2002). Primarily, in a continuous model, calculating at each time step is often more calculation than what is required for the simulation. And, since the time step is fixed, it is possible, in fact likely, that a change in state (e.g. tank becomes full) would not be synchronized with the occurrence of the time step calculation. This results in a lack of precision as shown in Figure 2. And finally, it is not uncommon for a system to exhibit mixed behavior where some elements behave continuously while other elements behave discretely.

2.1 Continuous System Example

Assume an intermediate storage tank is filled at a constant rate of 1 ton per minute and is emptied at two different rates, 0.3 tons per minute or 2.1 tons per minute. Also assume:

- The simulation starts with 5 tons in storage and an emptying rate of 0.3 tons per minute.
- The tank capacity of storage is 10 tons.
- When the intermediate storage becomes full, the emptying rate switches to 2.1 tons per minute. When the intermediate storage becomes empty, the emptying rate switches to 0.3 tons per minute.
- The simulation duration is 100 minutes and the fixed time step is 1 minute.

2.1.1 Continuous Model of the System

Figure 1 is a model of the continuous system using the continuous modeling technology of ExtendSim. The blocks called Storage and Sink are holding product, the Filling rate is a constant connected to the Storage block, and the Switch will define which emptying rate (slow or fast) applies to the Storage tank. The Switch is controlled by a Decision block that chooses the emptying rate according to the contents of the Storage tank.

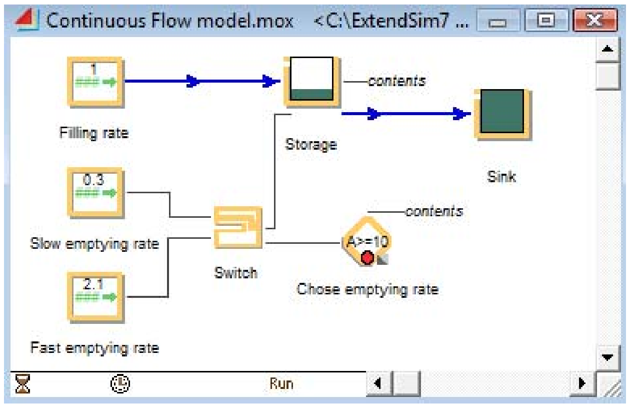

Figure 1: Continuous Flow model

Figure 2 is a graph of the variation of the contents in storage during the simulation. The top horizontal limit is the maximum capacity of storage and the bottom horizontal limit corresponds to empty. As the simulation runs, we can see that the contents of the storage periodically pass the maximum and minimum limits.

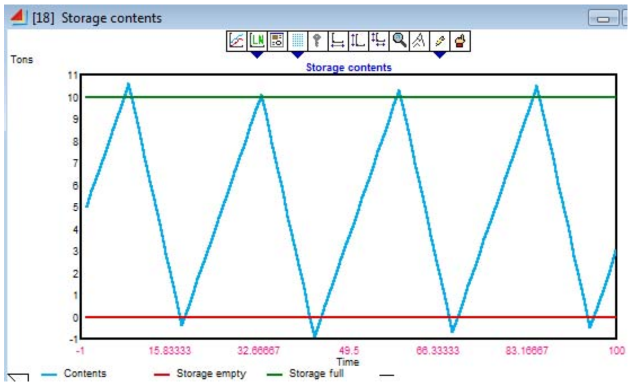

Figure 2: Graph of the storage contents over time

2.1.2 Limitations of the Continuous Approach

Because this is a continuous simulation, the model is reevaluated in fixed, one minute intervals. Consequently, errors will occur whenever an event critical to the model state happens somewhere between the minutes. In this model, critical events occur when the tank becomes full or empty, because this is when the outflow rate is supposed to change. If the tank happens to become full or empty somewhere in between the fixed intervals, the model gets the information after the event happens. This can be seen in Figure 2 where the tank's contents sometimes rise above the full line and sometimes fall below the empty line. This imprecision has an impact on:

- The moment in time when the system switches from one emptying rate to another.
- The quantities entering and exiting the storage area.

Secondly, the model needs only 10 critical threshold crossing time steps: the start of the simulation; the 4 times the storage becomes full; the 4 times the storage becomes empty; and the end of the simulation. However, since the continuous model has 100 time steps, 90% of the calculations are unnecessary.

These observations are not intended to dismiss the value of continuous simulation as a technology. However, they do highlight some limitations in its ability to capture events in linear flow systems where the rates of movement are constant between unevenly occurring events.

2.2 Hybrid System Example

Assume that ships carrying oil arrive at a harbor at different times. Depending on multiple factors, the ships will take a certain amount of time to unload their cargo. Some of the factors taken into account when unloading are:

- The unloading resources available.
- The amount of product to unload and the characteristic of products.
- The availability of storage in the different intermediate storage tanks.
- The refinery's capacity to process product from the intermediate storage areas.

This example exhibits both continuous (unloading process, storage in tank, refinery production) and discrete event (ship arrival, resources) behaviors.

2.2.1 Discrete Event Model of the System

One approach would be to use a discrete event simulation tool and "discretize" the continuous parts of the model. For example, the oil could be represented as discrete entities (items), each one representing a certain amount of liquid. The smaller the quantity represented by an item, the more accurate the simulation will be.

2.2.2 Limitations of the Discrete Event Approach

There are several limitations to using only discrete event modeling to simulate a hybrid system. One is that the execution speed of the model slows down. This occurs because the number events generated in the model increases proportionally with the number of items used to represent the oil flowing through the model.

Furthermore, representing simple continuous elements in the system becomes more complicated than necessary since discrete modeling constructs are being asked to represent continuous phenomena.

In addition, this approach will never be completely accurate in its prediction of how oil levels change throughout the system over time. For example, the moment a tank becomes full will be subject to error since this event might happen when a portion of an item is in the process of loading. In this case, the tank will contain more product than its capacity. This might sometimes be an acceptable simplification (Fiorini and Furia 2007), but not all the time.

Finally, as far as the flow of oil is concerned, the only events of real interest occur when the ships and tanks reach certain levels (such as the empty or full levels). Consequently, using a discrete item to represent the arrival of some relatively small amount of oil results in a large number of redundant events that aren't really useful. Imagine a tank with a capacity of one million gallons and a model where each gallon is represented by one discrete item. In this case one million events would need to pile up in the tank before a critical full event could occur.

2.3 Hybrid Approach

Some simulation tools on the market are capable of mixing the discrete and continuous technologies (Mosterman 2003).With these hybrid types of tools, it is common for the continuous parts of the model to use a fixed time step for triggering calculation times and the discrete parts of the model to use discrete event posting methods to schedule events. It is in fact complicated to get a system like this to work accurately. If we look at the continuous flow example from section 2.1, some component in the model must be able to recognize that a tank's contents could reach a critical state in between two time steps and then predict when it will occur. An event must then be scheduled to intercept and handle this occurrence accordingly. Without that predictive capability, the model will detect the critical state too late.

In addition, the very nature of discrete event scheduling can make this continuous predictive capability problematic. For example, assume the model predicts that a critical state in the tank will occur between time steps and an event is scheduled to handle it. However, also assume a discrete event from some other section of the model also gets scheduled before the tank's critical event arrives. These unforeseen events can alter the rate at which the tank is filling and cause errors in the originally predicted critical event time.

It should be noted that the hybrid method does have value when the continuous aspects of the system are nonlinear. In this case, using hybrid time steps and discrete events is necessary. However, many systems are best described as linear continuous processes. The DRS technology represents a very powerful and easy to use method well suited to modeling the flowing movement of material in those types of hybrid environments.

3 DISCRETE RATE SIMULATION

A second generation Discrete Rate Simulation technology has been implemented in the Rate library of ExtendSim AT and ExtendSim Suite. Designed for modeling linear continuous systems and hybrid systems, this approach moves material through the system at fixed rates until a critical event allows for rates to change.

The building blocks in the Rate library are capable of simulating a wide range of flow systems. They can be categorized as follows:

- Blocks that store and provide flow
- Blocks that control the effective flow rate
- Blocks that route flow

Because this new DRS technology employs an event posting architecture, models can contain a mix of discrete event and discrete rate elements without any difficulty. In ExtendSim that means a combination of blocks from the Rate (DRS) and Item (DES) libraries can be used in the same model.

3.1 Definitions

Definition 1 *In a DRS model, flow is what is stored and moved through the system. Since one unit of flow is not distinguishable from another unit of flow, flow can be almost anything that does not need to be uniquely identified.*

Definition 2 *A flow system is equivalent to a linear continuous system. It is a system where flow moves from one location to another at constant speed between two events. When an event occurs, the system re-evaluates the set of flow rates.*

Definition 3 *While constraints determine the maximum rate that flow can move, the effective rate is the actual rate of movement. Rates are expressed as a quantity of flow per time unit.*

Definition 4 *The state of a discrete rate model changes only when an event occurs. An event might be a tank that becomes empty or full, a maximum rate that changes during the run, a routing block that changes its output proportions, and so forth. DRS models make a calculation to determine, at that precise moment, what the effective rates are in each part of the model.*

3.2 Illustration of DRS with a Continuous Example

As an introduction to DRS, we revisit the continuous flow system that was described in section 2.1. This time the system is modeled using the discrete rate approach. In this new model, the rate of flow between storage tanks is regulated by the valves that have been inserted between them (Figure 3).

Source, Storage, and Sink tanks are used to hold product. Two valve blocks regulate the movement of the flow. As was true for the continuous model, the Switch is controlled by a Decision block that chooses the emptying rate according to the contents of the Storage tank.

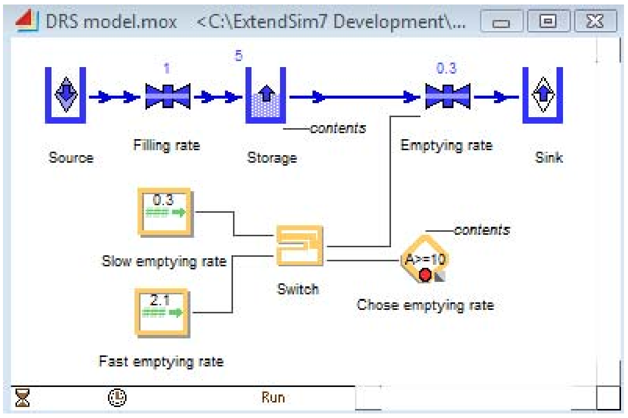

Figure 3: the DRS model

3.2.1 Course of the Simulation

During the simulation, the Storage tank predicts when it will become full or empty and schedules an event accordingly. The timing of these critical events depends upon the tank's current contents and its effective inflow and outflow rates.

When a critical event occurs, the storage tank updates its information (contents of the tank, total amount of inflow and outflow, tank is full…) and informs the model of its new state. This in turn triggers other decisions in the model. In this example, if the Storage tank becomes empty, the maximum rate of the emptying valve switches from 2.1 to 0.3 tons per minute.

If an inflow or outflow rate changes, a new event time is calculated and the future event is scheduled on the event calendar.

3.2.2 Results of the Simulation

The issues that were associated with the continuous approach discussed earlier have been resolved using DRS:

- The total number of events during the simulation has been optimized to 10. The events are listed on Table 1.

- The events occur at the exact time the model changes state.

As shown in Table 1, it is interesting to note that in the Continuous Flow model, the final content of storage after 100 minutes of simulation was 3 tons. In the DRS model it is 9.55 tons. In this case, the continuous model's delays in recognizing a change of state caused significant errors.

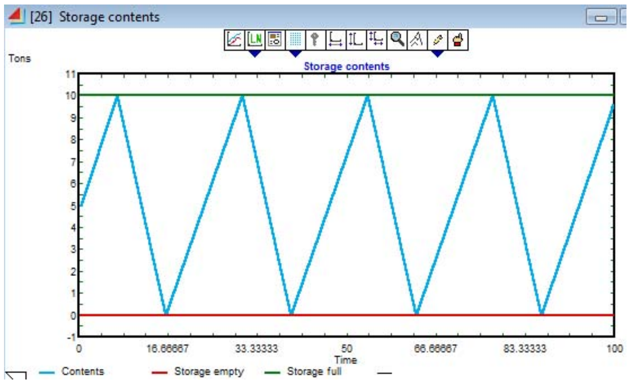

Figure 4: Graph of the storage contents (DRS model)

Table 1: Events in the DRS and continuous models

Event or time step	DRS model		Continuous model	
	Minutes	Tons	Minutes	Tons
Start simulation	0	5	0	5
Storage full	7.142857	10	8	10.6
Storage empty	16.23377	0	18	-0.4
Storage full	30.51948	10	33	10.1
Storage empty	39.61039	0	43	-0.9
Storage full	53.8961	10	59	10.3
Storage empty	62.98701	0	69	-0.7
Storage full	77.27273	10	85	10.5
Storage empty	86.36364	0	95	-0.5
End simulation	100	9.5454	100	3

3.3 Description of the New DRS Approach

This new DRS approach is designed to maximize the rate of flow in the system subject to certain constraints. The complexity of calculating the effective flow rate and the generation of events that dictate new rate calculations are handled by the DRS modeling components. For example, while a valve in a DRS model determines the maximum allowable rate of flow through a particular section of the model, the effective rate can end up being anything from 0 to the maximum. The effective rate will be as close as possible to the maximum rate defined by the valve, but other process limitations in the model might ultimately lower the effective rate.

The amount of time a rate remains constant in a particular section of a model depends on many factors including how the blocks have been configured and changing states in other areas of the model.

3.3.1 Description of the Course of a DRS

The sequence of actions taking place in a DRS model is as follows:

- At the beginning of the simulation, each effective rate in each section of the model is calculated.
- After the effective inflow and outflow rates have been defined for each flow processing block, blocks predict when their next critical events will take place and schedule them accordingly. It is assumed that the effective flow rate through a section will remain constant until the next event has arrived or the rate of inflow or outflow to the model section changes.
- When an event occurs which might change one or more effective rates, each potentially impacted Rate block updates its internal information and the set of effective rates are recalculated. Rate changing events can originate from any part of the model.
- Once the new set of effective rates has been calculated, all affected blocks update their internal information and reschedule their new future events.

3.3.2 Routing Flow in a DRS

The advanced routing capabilities offered by the Rate library provide the flexibility needed to model the kinds of complicated distribution behaviors typically demonstrated by linear continuous processes. The flow distribution logic available in the Merge and Diverge blocks from the Rate library can be divided between fixed rule and non-fixed rule modes (ExtendSim User Guide 2007).

The fixed rules include the Batch/Unbatch, Proportional and Select modes. For these modes, the distribution of flow always follows a predictable path that never deviates from the definite proportions. The Proportional mode, for instance, requires flow to be distributed across branches according to a fixed set of proportions. This proportion must always be strictly respected.

For example, assume a Merge block has two inflow branches and that a proportion of 1:1 has been defined for the top and bottom branches respectively. In addition, assume upstream valves are limiting the maximum flow rates into those inflow branches to 6 and 15 tons per minute respectively, and that a downstream valve is limiting the Merge block's one outflow stream to 16 tons per minute. In this example, the effective rate at each inflow branch will be limited to 6 tons per minute for a combined total of 12 tons per minutes for the merged outflow stream. The lower inflow streams could potentially accept more flow but the 1:1 proportion must be observed when the Merge block is in Proportional mode. See Figure 5.

Non-fixed rules include the Priority, Distributional, and Neutral modes. This means that the block in charge of routing the flow expresses a "preference" on how to distribute the flow between streams, but the main goal here is to maximize throughput irrespective of the preferences. If we take the previous example but use the Priority mode instead of Proportional, the result will be an effective rate of 6 tons per minute for the top inflow stream, 10 tons per minute for the bottom inflow stream and a total of 16 tons per minutes for the merged outflow stream. The non-fixed rules that have been implemented around the DRS architecture allow routing blocks such as the Merge and Diverge to adjust as conditions change over time.

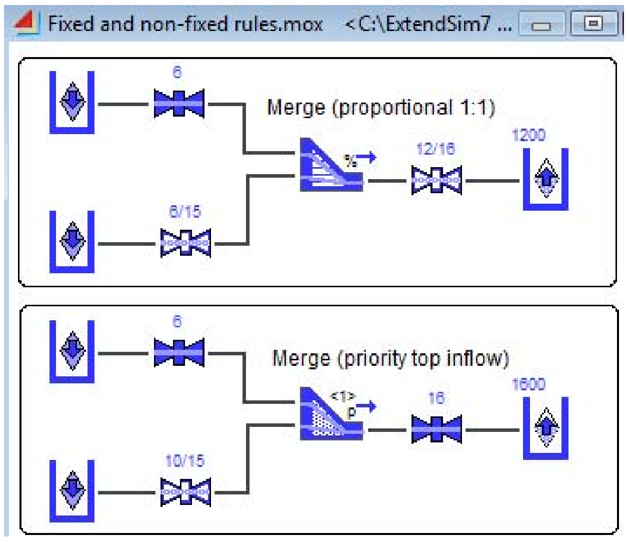
Fixed and non-fixed rules.mox <C:\ExtendSim7 ...
6
Merge (proportional 1:1)
12/16
1200
6/15
6
Merge (priority top inflow)
16
1600
10/15

Figure 5: Fixed and non-fixed rule comparison

3.3.3 Advantages of the DRS

The advantages of this approach are:

- The model schedules the exact moments the model reaches a new state. Consequently, the moment the state should change and the moment the model implements that change are synchronized.
- Since there is no fixed time step, calculations are done only when a scheduled event occurs.
- The modeling components from the Rate library determine the rate at which flow moves through the system given the logic defined by the model's configuration. This makes the work of the modeler more intuitive because some of the decisions are handled by the system.
- The DRS approach offers integrated and advanced logic for controlling the way streams of flow are routed. For example, Priority logic can be used to give one fork in the flow stream a higher priority.
- Mixing flow with discrete entities doesn't cause synchronization problems because each part of the model creates and reacts to scheduled events.

4 HOW DRS CALCULATES EFFECTIVE RATES

The DRS approach relies on two conditions in order to work:

- Each Rate block has to predict its next change-of-state event.
- No matter how complicated the flow system is, the DRS has to accurately determine the set of effective rates in the entire model for each event.

The first condition doesn't cause any problem as long as the model remains in a linear paradigm. The second condition is more complicated to implement.

Learning from the experience of the first generation DRS tool, we came up with the idea of using linear programming to perform the calculations of effective rates. After much testing, the method turns out to be a reliable and powerful approach to perform this type of calculation.

4.1 DRS as a Linear Programming Problem

Being able to define the DRS as a linear programming (LP) problem has the advantage of offering a set of powerful methods for performing the calculations of the effective rates.

Definition 5 Linear programming (LP) *is concerned with the maximization or minimization of a linear objective function in many variables subject to linear equality and inequality constraints* (Dantzig and Thapa 1997).

The field of LP has been an intense area of research for 50 years: "The development of linear programming has been ranked among the most important scientific advances of the mid-20th century" (Hiller and Lieberman 1995).

The general approach taken to define a DRS problem as a LP problem is:

- Each rate section represents a decision variable. See the definition of a rate section below.
- The impact of each block on the calculation of the effective rate is translated into one or more linear constraints in the LP problem. (They can be inequalities or equalities).
- The result of the calculation is a set of variables for the rate sections. This set of values has been maximized through an objective function. (The creation of the objective function will be discussed later and depends on the structure of the flow system.)

Definition 6 *A* rate section *is defined as a network of connected blocks, all possessing the same effective rate. Each rate section can include a succession of blocks and connections. (Imagine That Inc. 2007).*

4.2 Why the LP Method is Well-Suited to Perform DRS Calculations

The fact that the mathematical representation of a flow system can fit the general format of an LP problem allows us to use this as a global approach to perform calculations.

4.2.1 The Need for a Global Approach

The nature of a discrete rate system is such that a change in one part of a model may very well have an impact on multiple effective rates in other parts of the model. A lot of rates are dependent on each other even if they are not directly connected. The LP method combines all the constraints together to perform a global calculation of all the effective rates. This provides a centralized method to solve a global problem.

The first generation DRS method calculated the effective rates using message propagation. With that method, as the system becomes more interconnected, the volume of communication between blocks increases. Furthermore, the blocks decided locally what the effective rate was even if the decision depended on multiple blocks in the system. A local system of decision making was being applied to a globally interconnected system.

The global approach of LP is a better fit for a flow system, ensuring that the results remain reliable no matter how complex the flow system is.

4.2.2 Flow Maximization

A DRS model can be understood as a network of pipes where the flow circulates at a constant pressure. The purpose of the DRS is to move as much flow as possible through the system (i.e. to 'maximize' the flow).

Definition 7 *A* corner-point feasible (CPF) solution *is a solution that lies at a corner of the feasible region* (Hiller and Lieberman 1995).

As the effective rates are the decision variables in the LP, one possible corner-point feasible solution for the LP is to have all the effective rates set to 0. In a DRS, this solution translates to a model with no movement of flow between blocks. This solution is feasible as an LP solution but most likely is not the solution the modeler is looking for. The DRS tries to come up with a solution that is as far from the "no movement" solution as possible. As an analogy, the LP algorithm will maximize the flow by increasing step by step all the effective rates which increase the objective function. Maximizing the flow in a DRS is then equivalent to maximizing an objective function in an LP problem.

4.2.3 Simplex Algorithm Advantages

We chose the simplex method to solve the linear programming problem because it is designed to find the maximum solution for an objective function.

"As long as we have a first feasible solution, the simplex method will find the minimum (or maximum) to the objective function in a limited number of steps" (Gass 1985).

Moreover some other properties of flow systems made the simplex method a good fit:

First, the initial corner-point feasible solution can be found without any calculation. The zero vector is always a feasible solution for the effective rates even if it is usually not an optimal solution. This insures that there will be a solution to the problem unless the problem is unbounded, And the introduction of the equality constraints are easy to implement.

Second, a lot of constraints are upper bound constraints. Upper bounds are constraints that give a maximum constant limitation to a decision variable (effective rate<=maximum rate). By using the "Upper Bound Technique", this type of constraint doesn't add to the complexity of the LP problem. As explained by Hiller and Lieberman (1997) "The most important determinant of computation time for the simplex method is the number of functional constraints, whereas the number of nonnegativity constraints is relatively unimportant.…The upper bound technique avoids this increase in (computational) effort by removing the upper bound constraints from the functional constraints and treating them separately."

Finally, there are only real variables and no integer variables among the decision variables. This makes the simplex algorithm quite simple and fast to execute.

4.3 LP With and Without Non-fixed Rules

In paragraph 3.3.2, we discussed the difference between fixed and non-fixed rules as different methods of routing the flow. This has a very important impact on the LP calculations. Further explanations can be found in the "Biasing Flow" section of the ExtendSim User Guide (2007).

If the flow system doesn't contain any non-fixed rules to route the flow, or if the only non-fixed rules are "neutral", the calculation of the set of effective rates can be performed with only one LP calculation. If the flow contains at least one non-fixed rule (excluding neutral mode), then a cascade of LP calculations has to be done in order to find the set of effective rates.

4.3.1 Overview of the LP Approach

When an event happens that causes the recalculation of effective rates:

- The first phase determines how many effective rates have to be part of the calculation, defining the so-called “LP area”.
- The second phase collects all the constraints from the blocks that are part of the LP area as shown in Table 2.
- The third phase performs the LP calculation.
- The last phase updates the model with the new effective rates.

4.3.2 How the LP Area is Defined

Defining the LP area corresponds to finding the decision variables which are part of the LP calculation. The model determines the following:

- Which part of the model the event initiating the calculation came from.
- All the effective rates that can potentially be affected by the change of state.

The part of the model containing these effective rates is then called the LP area. The perimeter of the LP area will change over time depending on the events and the context of the model.

4.3.3 Example of the Constraints Collected

Listed below are some examples of constraints that blocks can apply to the linear programming problem:

Table 2: List of linear constraints

Block Status	Constraint	Description
Empty tank	$X_{out}<=X_{in}$	The emptying effective rate cannot exceed the filling effective rate.
Full tank	$X_{in}<=X_{out}$	The filling effective rate cannot exceed the emptying effective rate.
Valve	X<=constant constant>=0	The effective rate where a valve is located cannot exceed the maximum constraint.
Diverge flow in proportional mode	$X_{in}=\sum X_{i\ out}$; i=1,…,n.	The sum of inputs has to equal sum of outputs.
	X_{in} = factor * $X_{i\ out}$; i=1,…,n-1.	Each output is a portion of the input. No need to specify the last proportion, it is covered by the previous constraint.
….	…	…

4.3.4 The Objective Functions

When both the LP area and the constraints for the LP problem have been defined, the objective function can be created.

In the case of a LP area with only neutral modes and fixed rules for routing modes, the objective function is very simple - each effective rate has a coefficient of 1. This is because all the information needed to route the flow has been fully transformed into constraints. The only necessary maximization is to optimize each of the effective rates, one after the other.

The LP calculation gets more complicated when the LP area contains routing blocks with non-fixed rules. A non-fixed rule defines a preference for how to route the flow and does not strictly define how the flow has to be routed between streams (unlike, for example, a fixed proportion).

The preference concept gives an additional degree of liberty on how to distribute the flow compared to a model without any non-fixed rules. If we take the example of a priority routing rule (Figure 5), the fixed-rule from the Merge block is that the sum of inflow effective rates equals the outflow effective rate. There is also a preference in the block that the upper inflow should be utilized before the lower inflow. This preference is not translated into the linear constraint; instead it is handled at the level of the objective function in the LP problem.

For each block with a non-fixed rule (with the exception of the neutral mode) there is a separate LP subproblem to solve. Each of these blocks has a bias order and because each block is associated with a LP subproblem, several LP will have to be solved in a cascade to reach the appropriate set of effective rates.

The concept of bias has been introduced in the DRS to determine the order of solution of the different LP subproblems. When preferences are determined in part of the model, other preferences in the same model might potentially be in contradiction with the new ones. This can cause uncertainties on the appropriate set of effective rates to apply to the model. To resolve these uncertainties, each block that defines a preference also has a bias order. The bias order specifies which preference has precedence over the other. The model initially assigns a default bias order to each of these blocks, but each bias order can be customized as needed.

At this point all the blocks are expressing a ranked preference and a specific objective function has been defined for each of them. Through a cascade of calculations, each block’s preference is expressed, in turn, based on its bias order.

As each bias block takes its turn, it calculates the maximum effective rate which could occur at its location and distributes the flow according to its preferences, without taking into consideration the preferences expressed by blocks with a lower bias order. When the effective rates associated with the block have been determined, these rates will be fixed for the subsequent calculations involving blocks with lower rankings.

If we use the example in Figure 5, the objective function of the Merge block in priority mode is: Maximize 2*the upper effective inflow rate + 1*the lower effective inflow rate. The objective function expresses the preference because the function gains more by increasing the upper stream than by increasing the lower stream.

When all the objective functions associated with a block expressing a preference have been determined, and if there are still some effective rates which haven't been calculated, a last LP subproblem is solved with an objective coefficient of 1 for each effective rate. When all the preferences have been resolved, the problem becomes a fixed rule problem.

5 CONCLUSION

The discrete rate simulation (DRS) system presented in this paper has been implemented in the Rate library in the simulation product ExtendSim. This tool has been designed to model flow systems and hybrid systems. The areas of application of this modeling tool can be very diverse:

- It can model the processing of powders, liquids, gases and other fluids in areas like Petrochemical, Manufacturing, Mining, Water treatment, or any other industry that processes commodities in bulk or batches.
- It also can model the processing of "things" that are so numerous that it would be inconvenient or overwhelming to model them individually. This type of modeling could range from food to drugs or even data storage and manipulations, or any other industry that mixes, fills or packages products on high-volume or high-speed lines.

The technical significance of the Rate library is that the DRS model is transformed into a linear programming problem. This allows the accurate calculation of the movement of flow between blocks no matter how complex the flow system is.

The Rate library has been well received by its current users. Development to integrate flow controls to mimic non-linear behavior and the integration of flow attributes is envisioned for the future. Other developments will be determined based on the feedback we receive.

ACKNOWLEDGEMENTS

We would like to thank Antoine Outerleys and Christian Knauber for initiating the idea of using a linear programming algorithm to calculate the effective rates in a flow system. And also Lynn Scheurman for his contribution in the definition of the priority and distributional mode.

REFERENCES

Dantzig, G. B. and M. Thapa. 1997. *Linear Programming 1: Introduction*, ed. P. Glynn. Page 1.

Fiorini, M. M. and J. Furia. 2007. Simulation of continuous behavior using discrete tools: ore conveyor transport. In *Proceedings of the 2007 Winter Simulation Conference*, ed. S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, 1655–1662. Piscataway, NJ: IEEE, Inc.

Gass, S. I.. 1985. *Linear Programming Methods and Applications* Fifth edition. Pages 66 to122.

Hiller, F. S. and G. Lieberman. 1995. *Introduction to operations research* Sixth Edition. Pages 81 to 274.

Imagine That Inc. 2007. *ExtendSim User Guide*, Seventh release.

Imagine That Inc. <www.extendsim.com>.

Monsef, Y. 1997. *Modelling and Simulation of Complex Systems*, Frontiers in Simulation, ed. E.Kerckhoffs, A. Lehmann, H. Pierreval and R. Zobel.

Mosterman, P. J. 2003. Mode Transition Behavior in Hybrid Dynamic Systems. In *Proceedings of the 2002 Winter Simulation Conference*, ed. E. Yücesan, C.-H. Chen, J. L. Snowdon, and J.M. Charnes, 623–631. Piscataway, NJ: IEEE, Inc.

Phelps, R. A., D. Parsons and A. Siprelle. 2002. Non-item based discrete-event simulation tools. In *Proceedings of the 2002 Winter Simulation Conference*, ed. E. Yücesan, C.-H. Chen, J. L. Snowdon, and J.M. Charnes, 182–186. Piscataway, NJ: IEEE, Inc.

Siprelle, A. J. and R. A. Phelps 1997. Simulation of bulk flow and high speed operations. In *Proceedings of the 1997 Winter Simulation Conference*, ed. S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, 706–710. Piscataway, NJ: IEEE, Inc.

AUTHOR BIOGRAPHIES

CECILE DAMIRON is simulation software engineer at Imagine That Inc. She received her Masters of Science in Econometrics and completed postgraduate specialized studies in the use of computer science/engineering in economic analysis & decision-making at the University Lumiere Lyon II at Lyon (France). Before joining Imagine That Inc. in 2004, Ms. Damiron worked for 7 years as a simulation consultant at 1Point2 in France. She can be contacted by email at <ceciled@extendsim.com>

ANTHONY NASTASI is a simulation software engineer at Imagine That Inc. He received a Masters of Science in Regulatory Economics from the University of Wyoming. Before joining Imagine That Inc. in 2000, Mr. Nastasi worked 4 years at Los Alamos National Laboratory modeling textile supply chains. He can be contacted by email at <anthonyn@extendsim.com>