A Continuous Time State Task Network model for Short Term Scheduling of Operations in Cascaded Continuous Processing Units

by

Sumit Kumar Bose
sumitb@iimcal.ac.in

Doctoral student, IIM Calcutta, Diamond Harbour Road, Joka P.O., Kolkata 700 104, India

&

Subir Bhattacharya
Professor, IIM Calcutta, Diamond Harbour Road, Joka P.O., Kolkata 700 104, India
A Continuous Time State Task Network model for Short Term Scheduling of Operations in Cascaded Continuous Processing Units

Sumit Kumar Bose*, Subir Bhattacharya  
*Indian Institute of Management Calcutta, Kolkata -700104, India

Abstract
In this paper we propose a novel continuous time model based on state task network for the problem of scheduling \(m\) products in \(n\) continuous processing units with simultaneous arrival of liquid inputs and multiple intermediate upliftment dates. The units can process only one product line at a time and are buffered by fixed capacity storage tanks. An optimization model for the problem tries to balance the spillage cost, upliftment failure penalty and the changeover cost. The present work develops a continuous time MINLP model for the problem based on the global event point representation scheme. The model is more general as we do not pre-fix some of the event point variables on the dates of upliftments. The MINLP model is linearized using standard linearization techniques and the resulting MILP model can output, in reasonable time, optimal schedule for 2 weeks for a 3-unit 3-product scenario as encountered by us in a refinery.

Keywords: Scheduling, Mixed integer optimization, Linearization, Continuous chemical process, State task network, Continuous time.

1 Introduction
The scheduling problem discussed in this paper involves \(n\) cascaded continuous processing units responsible for processing \(m\) product lines. Each product line needs to be processed by all the units in the same sequence, and has dedicated finite capacity storage tanks before and after every processing unit. A unit can process only one product line at a time. Inputs for all the product lines arrive continuously and simultaneously on the input side of the first unit in the sequence. There are multiple intermediate due dates for the final products. An optimal schedule for the units calls for a trade-off among spillage costs, upliftment failure penalties and changeover costs. We encountered the problem while scheduling operations of the lube processing units in a refinery. In this paper, we propose a continuous time mathematical
model for the problem using state task network representation. An MINLP model was developed for the purpose and linearized using standard linearization techniques. The resulting MILP model was tested using GAMS for three units and three product lines. The model could output schedules for a two-week scheduling horizon within reasonable time.

The issues involved in scheduling operations in batch plants are distinctly different from that of scheduling of operations in continuous process plants. In a batch-processing unit, inputs in right amount and proportion are fed into the unit, ‘treated’ for a fixed amount of time in non-preemptive style, and outputs are taken out after the complete batch gets processed. In contrast, in a continuous processing unit, input streams for a product line are fed in continuously at one end and the output streams flow out simultaneously from the other end.

Short term scheduling of batch processing units addresses the issue of determining the optimal sequence in which the product batches should be taken up for processing; each batch of a particular product typically requires the same amount of processing time. On the other hand, in a continuous processing unit one product line can be processed in a number of stretches interleaved with the processing of other streams. The length of each such stretch, called a *run length*, of a product line is a decision variable. Figure 1 is an example of possible run lengths in a unit considering three product lines or streams. Thus, the problem of scheduling a continuous processing unit involves deciding, for each product, the number of stretches and the duration of each stretch, and interleaving these production stretches of the various product lines in an optimal manner.

![Figure 1: Example of run-lengths in a unit](image)

---

Run 1 of stream 1
Run 1 of stream 3
Run 2 of stream 3
Run 3 of stream 3

Stream 1
Stream 2
Stream 3

Time line
Comprehensive review of the literature that relates to scheduling of operations in batch process plants are presented in Shah(1998), Kallrath(2002) and Floudas and Lin(2004). Pinto and Grossmann (1998) provides a detailed summary of the different assignment and sequencing models used for scheduling in process industry. Grossman et al. (1996) discusses various mixed integer mathematical programming methods used for scheduling batch processes. Schilling and Pantelides (1996), Mendez et al. (2000) and Grunow et al. (2002), for example, consider optimal scheduling of batch plants under different sets of criteria and assumptions. In contrast to the attention paid by the researchers to the scheduling problems arising in batch processing industry, interest in scheduling of operations in continuous processing units is rather recent. Jain and Grossmann (1998) build a mathematical model for cyclic scheduling of continuous processing plants with parallel units and decaying performance. Alle et al. (2004) has developed a continuous time mathematical model for cyclic scheduling of multi product multistage continuous processing plants. The short term scheduling of refinery operations and mixed integer models based on continuous time formulations have been discussed in Jia and Ierapetritou (2004).

Discrete time models for the continuous process scheduling problems are not truly optimal since they do not allow changes in the system within a time grain. Thus the quality of the solution depends significantly on the time granularity of the discrete time models – smaller the time interval closer the solution to the optimal for the problem. However, with smaller time intervals the number of constraints increases significantly, and the model runs even slower. Continuous time formulations can take care of this problem, but they are difficult to arrive at due to difficulties in checking resource violations. A number of continuous time formulations for chemical batch process scheduling problems have been suggested in the recent past. For example, Ierapetritou and Floudas (1998), Ierapetritou et al. (1999) propose a continuous time model based on unit specific event points. In the papers, the authors have formulated continuous time mathematical models based on the state task network representation for continuous processes taking into account multiple intermediate due dates for the finished products. Wang and Guignard (2002) propose a continuous time model for the batch process scheduling problem based on the concept of global event points. Such formulations depend on an initial consideration of the number of event points. The locations of these event points on the time axis are not known a-priori and are variables to be
determined during the optimization process. As already noted in the previous paragraph, the models developed by Alle et al. (2004) and Jia and Ierapetritou (2004) are the notable ones amongst the continuous time models for continuous processes. This paper proposes a continuous time formulation for a class of continuous process scheduling problems using state task network representation proposed by Kondili et al. (1993).

2 Problem Statement

The set of units under consideration consists of a sequence of \( n \) processing units, \( U_1, U_2 \ldots U_n \), buffered by fixed capacity storage tanks to hold intermediate products (Figure 2). The units work in a ‘blocked out’ fashion i.e. the units can process only one product line (also referred to as stream) at a time. Each product line has its exclusive set of tanks before and after every processing unit. We consider \( m \) product lines, \( G_1, G_2 \ldots G_m \), each of which requires to be processed by the \( n \) units in the same sequence. The processing capacity (known as the feedrate measured in MtPD, Metric tons Per Day) of a unit for a product line is considered fixed, but varies from product to product. The processing of a unit involves splitting the feed with the help of reagents into intermediate streams according to a yield percentage fixed for a product line for the unit. The intermediate streams that are relevant for the final products under consideration get deposited in the tanks on the output side. What happens to the other output fractions is beyond the purview of this discussion. Thus, each unit processes one stream at a time, taking its input from the corresponding input tank if it has enough stock and depositing the output into corresponding output tank if it has enough room (ullage). The presence of intermediate storage tanks obviates the need for the units to process the same product in tandem.
The finished products coming out of the last unit \(U_n\) also get stored in fixed capacity tanks to be uplifted according to some pre-specified upliftment schedule. A product can be uplifted several times by specified quantities during the scheduling horizon. Penalty is incurred if the required amount of a finished product is not ready by the specified due date. Shortfall in one upliftment of a product cannot be compensated by providing more during the next upliftment of the same product.

The inputs for the product lines arrive at the input tanks of Unit \(U_1\) simultaneously at constant rates. The rates at which these inputs arrive depend on factors that are beyond the control of this block of units. Each of these streams gets stored in a fixed capacity tank if there is ullage in it; otherwise it spills, and is downgraded to lower valued products. Thus, if a product is processed in the unit \(U_1\), the inputs of all the other products flow in simultaneously and have to be either stored in tanks for future processing or they have to be ‘spilled’ to lower valued products if space is not available in the tanks. Note that spillage is not allowed for intermediate or finished products. In essence, spillage can occur only at the input of \(U_1\) because the inputs for all the product lines are arriving simultaneously and also because these units have no control on the rates at which these inputs arrive. Spillage implies opportunity lost, and hence, has a penalty associated with each unit of input stream spilt. Spillage can be reduced by quick changeovers. But a changeover from one product line to another has its associated cost and time. Thus, we have three factors to balance – the per unit spillage penalty for the input stream of each product line, the cost for changeovers, and the per unit penalty for failing to meet the upliftment schedule. Schedules of the units try to balance these factors by processing each product line in number of stretches during the
scheduling horizon interleaved with the processing of other streams. Longer stretches of run are preferred to reduce changeover cost while shorter stretches may bring down the spillage penalty and/or the upliftment failure penalty.

Given the scenario and a scheduling horizon of \( H \) days, the aim is to determine the start time and end time of processing of different streams (product lines) in different units so that total cost/penalty is minimized. It must be noted that the same stream can be processed a number of times in a processing unit within the same scheduling horizon. A unit \( j \) if it takes up processing of a product, say \( g \), at time \( T \), then the unit must process the same product till \( T+b \) without taking up processing of any other product between times \( T \) and \( T+b \). This restriction is imposed to efficiently utilize the minimum amount of reagents that need to be added when changing over to product \( g \). It may be recalled that the continuous stretch of time during which a product is processed (run) in a unit is called the run length. Thus, run length of a product in a unit must be greater than or equal to the minimum run length \( b \). For simplicity, we assume that the streams running in the units at the time period just before the start of the scheduling horizon have completed their minimum run length. That is, the units are free to changeover and can therefore take up streams for processing different from the one that was being processed. Changing over from one product to another requires some set up time. However our model considers the changeover to be immediate. The changeover cost depends on the unit but is independent of the streams involved. Product-wise feedrates and yield percentages for the units are fixed data. The arrival rates of the inputs in front of (on the left of) \( U_1 \) and the upliftment schedule for the scheduling period are known at the time of planning the units’ schedule. The scheduling problem aims to find out an optimal mix of product sequences of varying run lengths so as to minimize the total cost/penalty.

3 Mathematical Model

An ‘event’ happens when something changes in the system – be it assigning a new task to a unit and/or uplifting some finished product. An event point is the point in time when an event happens. More than one event can happen at the same event point, for example two units can start processing new streams at the same point in time. Every event point is distinct in time. In order to solve a scheduling problem in terms of event points, we specify the number of event points to be used and allow the model to optimally distribute these event points on the
continuous time line so that the cost of the system is optimal with that number of event points. Note that, in general, the inputs neither fix the time of the event points nor do we fix the type or number of events that will take place at a given event point. We solve the model repeatedly, each time allowing one more event point than the previous round. We continue the iterations till the objective value does not improve any further or till it satisfies a certain stopping criteria.

The proposed mathematical formulation uses the event point concept in the following manner:

a. **Set of ‘non-overlapping’ global event points.** The set of event points are defined in a manner that is common for all tasks and all units. In essence the event points are global in nature. An event is said to happen whenever there is a changeover in a processing unit and/or an upliftment of a stream. Thus the position of the event points should align with the time of changeover in the units and/or with the time when the off-take of a stream is scheduled. The locations of two consecutive event points cannot coincide.

b. **Set of virtual upliftment tasks.** Upliftment tasks are defined for every upliftment of every stream. These upliftment tasks are virtual tasks as the tasks are assumed to undergo processing in virtual units and have zero processing time. For the scheduling horizon under consideration, the number of these virtual tasks is known a priori since the upliftment schedule is available. The amount that ‘should’ undergo processing of this task may be different from the amount that actually undergoes processing of this task; and any difference between the former and the latter indicates upliftment failure. Amount that ‘should’ undergo processing of a virtual upliftment task is the quantity of the stream demanded in that particular upliftment. Suppose quantity demanded for a stream in an upliftment is \( d \), then \( d \) is the maximum limit imposed on the processing amount that a virtual task defined for the purpose can undergo. Each such task has a processing cost associated with it if and only if the quantity that undergoes processing of the task is less than the quantity demanded, \( d \). This acts as a surrogate for the upliftment failure penalty.

c. **Time points of all event points are variables to be determined.** This is a major departure from the earlier papers that pre-assigns the time point of some of the events
so as to coincide it with the time of the upliftments. That is, from amongst the set of event points, some of the event points are considered fixed a-priori. While the rest are allowed to vary. For example, if there are 8 event points in the set and there is only one upliftment then, say, the location of the 5th event point will be fixed beforehand. Event points numbered 1 to 4 must occur before the upliftment and event points numbered 6 to 8 must happen after the upliftment. This is too restrictive and may not give rise to optimal schedule. However, in our proposed formulation all the event points in the set are free to be decided by the model. The formulation ensures that the upliftments flush with some or other event points.

d. **Virtual spillage task.** For each product line we define virtual task that takes care of the spillage situations. The start time of these tasks may not flush with any of the event points. Each of the virtual spillage tasks has a processing cost associated with it for processing in virtual units. This acts as a surrogate for the spillage cost.

For simplicity we disallow shutdown during the planning horizon.

We will use the following sets, parameters and variables in our formulation.

*Table 1: Notations for the mathematical formulation*

<table>
<thead>
<tr>
<th>Sets</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$G$ : Set of streams</td>
<td></td>
</tr>
<tr>
<td>$I$ : Set of tasks</td>
<td></td>
</tr>
<tr>
<td>$E$ : Set of states</td>
<td></td>
</tr>
<tr>
<td>$J$ : Set of real units</td>
<td>(excludes the virtual units that processes upliftment and spillage tasks)</td>
</tr>
<tr>
<td>$N$ : Set of event points</td>
<td>$\eta \in [\eta_1, \ldots, \eta_{\text{last}}]$</td>
</tr>
<tr>
<td>$I^g$ : Set of tasks</td>
<td>associated with the stream $g$, $I^g \subseteq I$</td>
</tr>
<tr>
<td>$I_j$ : Set of tasks</td>
<td>that can be performed in unit $j$, $I_j \subseteq I$</td>
</tr>
</tbody>
</table>
\( I^p_e \): Set of tasks that produce state \( e, I^p_e \subseteq I \)

\( I^c_e \): Set of tasks that consume state \( e, I^c_e \subseteq I \)

\( I_{\text{spill}} \): Set of spillage tasks, \( I_{\text{spill}} \)

\( I_o \): Set of virtual upliftment tasks, \( I_o \subseteq I \)

\( E^g \): Set of states associated with stream \( g \).

\( E_{\text{inp}} \): Set of input states

\( E_{\text{int}} \): Set of intermediate states

\( E_{\text{fin}} \): Set of final states

**Parameters**

\( r_i \): Per unit processing cost of task \( i, i \in I_{\text{spill}} \)

\( a_e \): Rate of arrival of state \( e \in E_{\text{inp}} \) in front of \( U_i \)

\( C_e \): Capacity of tank corresponding to state \( e \)

\( q_j \): Cost of changeover from one stream to another in unit \( j \)

\( d_i \): Amount required to be processed by the upliftment task \( i, i \in I_o \) (Demand)

\( \text{dued}_i \): Date at which upliftment task \( i, i \in I_o \) is due

\( u_i \): Per unit penalty of the amount by which task \( i, i \in I_o \), fails to meet the target \( d_i \) (upliftment failure cost)

\( p_i \): Percentage yield of task \( i \notin I_{\text{spill}} \) and \( i \notin I_o \)

\( A_j \): Shutdown cost of unit \( j \)

\( f_i \): Feed rate of task \( i, i \notin I_{\text{spill}} \) and \( i \notin I_o \)

\( b \): Minimum runlength

**Variables**
\[ Z_{i\eta} = \begin{cases} 1 & \text{if upliftment task } i, i \in I_o \text{ takes place at event point } \eta \\ 0 & \text{otherwise} \end{cases} \]

\[ S_{i\eta} : \text{Material available in state } e \text{ at the beginning of event point } \eta \]

\[ T_{\eta} : \text{Time at which event point } \eta \text{ occurs} \]

\[ Y_{i\eta} = \begin{cases} 1 & \text{if task } i \text{ is processed between event points } \eta \text{ and } \eta + 1 \\ 0 & \text{otherwise} \end{cases} \]

\[ V_{j\eta} = \begin{cases} 1 & \text{if } Y_{i\eta} = 1 \text{ and } Y_{i,\eta+1} = 0, i \in I_j \\ 0 & \text{otherwise} \end{cases} \]

\[ B_{i\eta}^{\text{spill}} = \text{Amount processed by task } i, i \in I_{\text{spill}}, \text{ between event point } \eta \text{ and } \eta + 1 \]

\[ B_{i\eta}^{\text{uplift}} = \text{Amount by which task } i, i \in I_o, \text{ fails to complete its target } d_i \]

\[ D_{i\eta} = \text{Duration for which task } i \text{ is processed between event points } \eta \text{ and } \eta + 1; \]

\[ D_{i\eta} = Y_{i\eta}(T_{\eta+1} - T_{\eta}) \]

\[ Y_{i\eta} \text{ are the binary decision variables. } Y_{i\eta} \text{ takes the value of 1 if task } i \text{ is processed between event points } \eta \text{ and } \eta + 1, \text{ otherwise it is 0. The variables } V_{j\eta} \text{ represent changeover points.} \]

The value of \( V_{j\eta} \) is 1 if a task that gets processed in unit \( U_j \) at event point \( \eta \) is different from the task that is processed at event point \( \eta + 1 \). \( B_{i\eta}^{\text{spill}} \) is the amount of raw material spilled between event points \( \eta \) and \( \eta + 1 \) by task \( i, i \in I_{\text{spill}} \). \( B_{i\eta}^{\text{uplift}} \), for \( i \in I_o \), is the amount on which upliftment penalty will be charged. \( D_{i\eta} \) is the duration for which task \( i, i \notin I_{\text{spill}} \) and \( i \notin I_o \), is processed between event points \( \eta \) and \( \eta + 1 \).

### 3.1 Constraints

#### 3.1.1 Storage constraints in level 0 tanks

As mentioned earlier inputs for all the product lines flow in simultaneously to level 0 tanks. An incoming stream waits in the designated fixed capacity tank to be processed by \( U_1 \). If the tank gets filled up, the excess amount spills. For feasibility reasons, \( f_i \) is much higher than \( a_e \) for all tasks \( i, i \in I^c_e \) and \( e \in E_{\text{inp}} \). The stock in the tank corresponding to the states \( e \in E_{\text{inp}} \) at the start of the event point \( \eta + 1 \) is equal to the stock at the start of event point \( \eta \).
\( \eta \) adjusted by the amount arriving between event points \( \eta \) and \( \eta +1 \) minus the withdrawal that takes place in case the state is processed by unit \( U_1 \) or is spilled during the period lying between event points \( \eta \) and \( \eta +1 \). The expression for updating level 0 tank stocks are given by:

\[
S_{e,\eta+1} = S_{eq} + a_e(T_{\eta+1} - T_\eta) - \sum_{i \in I^c_i} f_i D_{in} - \sum_{i \in I^p_i} B^{spill}_{in} \quad \forall e \in E_{inp}, \eta
\]  

(1)

Since spillage and processing of a state \( e \in E_{inp} \) cannot take place simultaneously, \( D_{eq} > 0 \) and \( B^{spill}_{in} > 0 \) in the above equation cannot be concurrent. If state \( e \) is processed in unit 1 between event point \( \eta \) and \( \eta +1 \) (\( Y_{in} = 1 \) i.e. \( D_{eq} \) is positive-see duration constraint in equation (11)), then the equation reduces to \( S_{e,\eta+1} = S_{eq} + a_e(T_{\eta+1} - T_\eta) - f_i D_{in} \), where \( i \notin I^p \) and \( i \in I^c \). If state \( e \) spills during the period between event points \( \eta \) and \( \eta +1 \), then the equation reduces to \( S_{e,\eta+1} = S_{eq} + a_e(T_{\eta+1} - T_\eta) - B^{spill}_{in} \), where \( i \in I^p \) and \( i \in I^c \). Since \( S_{eq} \) has to be less than or equal to the corresponding capacity \( C_e \), the variable \( B^{spill}_{in} \) will assume a positive value, and since task \( i \in I^p \) has a positive processing cost, \( S_{e,\eta+1} \) will be forced to assume the value of \( C_e \). If there is no spillage from the tank corresponding to state \( e \) between event points \( \eta \) and \( \eta +1 \), and if state \( e \) is also not processed by the unit in the period \( \eta \) (\( Y_{in} = 0 \) i.e. \( D_{eq} = 0 \)), then equation (1) reduces to \( S_{e,\eta+1} = S_{eq} + a_e(T_{\eta+1} - T_\eta) \) indicating an increase in the stock position of \( e \) in the tank by an amount \( a_e(T_{\eta+1} - T_\eta) \).

3.1.2 Storage constraints in level 1 to level n-1 tanks

The stocks in the intermediate tanks in levels 1 to \( n-1 \) can be updated using the following constraint:

\[
S_{e,\eta+1} = S_{eq} + \sum_{i \in I^c_i} p_i f_i D_{in} - \sum_{i \in I^p_i} f_i D_{in} \quad \forall e \in E_{inp}, \eta
\]  

(2)

The stock at the start of event point \( \eta +1 \), \( S_{e,\eta+1} \), depends on the opening stock at the start of event point \( \eta \), build up of stock by outflow \( p_i f_i D_{in} \) (where \( i \in I^c_i \)) from the preceding unit if any, and withdrawal of stock \( f_i D_{in} \) (where \( i \in I^p_i \)) by the next unit if any.
3.1.3 **Storage constraints in level n tanks**

Stocks at the finished product tanks need to be updated by a separate expression to account for the upliftments and the possible shortfalls.

\[
S_{e,\eta+1} = S_{e\eta} + \sum_{i \in I_e} p_{i,\eta} D_{in} - \sum_{i \in I_e} (d_i - B^{upf}_{i\eta}) Z_{i,\eta+1} \quad \forall e \in E_{fin}, \eta
\]  

(3)

Stock for state \(e, e \in E_{fin}\), at event point \(\eta + 1\) is the stock that was available for that state at event point \(\eta\) plus the production if any of the state between \(\eta\) and \(\eta + 1\) minus any upliftment that takes place at \(\eta + 1\). In the last term at most one \(Z_{i,\eta+1}\) can be one (see equation (12)). \(B^{upf}_{i\eta}\) is the shortfall, if any, of the targeted amount \(d_i\) of state \(e\).

3.1.4 **Capacity constraints**

The stock in any tank at any point of time cannot exceed its capacity and hence

\[
0 \leq S_{e\eta} \leq C_e \quad \forall e, \eta
\]  

(4)

3.1.5 **Changeover constraints**

At any point of time, a unit can changeover to a maximum of one other stream. Thus,

\[
V_{j\eta} \leq 1 \quad \forall j, \eta
\]  

(5)

Definition of \(V_{j\eta}\) requires defining a mathematical constraint of the following form:

\[
V_{j,\eta+1} = 1 - \sum_{i \in J_j} Y_{i\eta} Y_{i,\eta+1} \quad \forall \eta, j
\]  

(6)

These constraints force \(V_{j\eta}\) to 1, when \(Y_{i\eta}\) is 1 and \(Y_{i,\eta+1}\) is 0 for \(i \in J_j\). For all other combinations of \(Y_{i\eta}\) and \(Y_{i,\eta+1}\), the binary variable \(V_{j\eta}\) assumes a value of 0.

3.1.6 **Event point constraints**

\[
\sum_{i \in I_e} Z_{i\eta} + \sum_j V_{j\eta} \geq 1 \quad \forall \eta
\]  

(7)

Constraint (7) mathematically represents the definition of the event point. Accordingly, the above constraint forces either a changeover in at least one unit at event point \(\eta\) or forces the date of the upliftment to coincide with the time of an event point.
3.1.7 Minimum run-length constraints

The minimum run length is given by the following equation:

$$T_{j \eta} V_{j \eta} - T_{\eta} V_{j \eta} \geq b V_{j \eta} - M (1 - V_{j \eta}) V_{j \eta} \quad \forall j, \eta, \eta_i > \eta$$  (8)

Through this constraint we ensure that any two changeovers in unit $U_j$ will have a gap of at least $b$ periods. For every combination of $V_{j \eta} = 1$ and $V_{j \eta_i} = 1, \eta_i > \eta$ the constraint reduces to $T_{j \eta} - T_{\eta} \geq b$. It must be noted that the event points are non-overlapping. To ensure that non-overlapping event points we add the following constraint to the model:

$$T_{\eta+1} - T_{\eta} \geq \delta \quad \forall \eta$$  (9)

Where $\delta$ is any small positive quantity.

3.1.8 Unit allocation constraints

Exactly one task can be assigned to unit $j$ for processing during event point $\eta$. So

$$\sum_{i \in I_j} Y_{i \eta} = 1 \quad \forall \eta$$  (10)

It may be noted that if we allow shutdown then equality constraint has to replaced by the inequality ($\leq$) constraint above.

3.1.9 Duration constraints

$$T_{\eta+1} - T_{\eta} Y_{i \eta} = D_{i \eta} \quad \forall i, \eta$$  (11)

If $Y_{i \eta} = 1$, then the duration for which task $i, i \in I_o$ and $i \notin I_{\text{spill}}$ is processed between event points $\eta$ and $\eta + 1$, $D_{i \eta}$ is equal to $T_{\eta+1} - T_{\eta}$; otherwise $D_{i \eta}$ is equal to 0.

3.1.10 Upliftment date constraints

$Z_{i \eta}$ is 1 if the time $T_{\eta}$ of event point $\eta$ coincides with the due date of upliftment task $i, i \in I_o$, otherwise it is 0. Further the time of no more than one event point should coincide with the due date of task $i \in I_o$. We achieve this by specifying the following set of constraints:

$$T_{\eta} - \text{dued}t_i Z_{i \eta} = 0 \quad \forall i \in I_o, \eta$$

$$\sum_{i \in I_o} Z_{i \eta} = 1 \quad \forall i \in I_o$$  (12)

Please recall that the upliftment tasks belonging to the set $I_o$ do not require any processing time. That is they are assumed to be instantaneous.
3.1.11 Horizon constraints

\[ \sum_{\eta=1}^{n_{\text{last}}-1} (T_{\eta+1} - T_{\eta}) = H - 1 \]  

(13)

In our formulation we stipulate the first event point to occur at the beginning of the scheduling horizon i.e. \( T_{1} = 1 \) and the last event point to be at the end of scheduling horizon i.e. \( T_{\text{last}} = H \). Remaining event points get distributed within the scheduling horizon. Further, we add the following constraint:

\[ T_{\eta} \leq H \quad \forall \eta \]  

(14)

\[ T_{\eta} \geq 1 \quad \forall \eta \]  

(15)

\[ \sum_{\eta, i \in I_{j}} D_{ij} \leq H - 1 \quad \forall j \]  

(16)

3.2 Objective function

The objective is to develop a schedule that would lead to the least overall cost. The objective function to be minimized is given by:

\[ \min \sum_{\eta, i \in I_{\text{pos}}} r_{i}^{\text{spill}} B_{i}^{\text{spill}} + \sum_{\eta} \sum_{j} q_{j} V_{j} + \sum_{i} u_{i} B_{i}^{\text{spill}} \]  

(17)

The first term gives the cost of spillage, the second term represents the cost of changeovers and the third term denotes the penalties for upliftment failures.

4 Model Simplification

The above formulation has nonlinear terms in many of the constraints. In most of the cases non-linearity involve bilinear products of continuous and binary variables. Equation (6), however, involves bilinear product of two binary variables. Employing standard linearization techniques it is possible to remove all the non-linearity from the proposed formulation. Bilinear products of binary and continuous variables are removed from the formulation by employing linearization techniques presented in Appendix A following Floudas(1995) and Glover(1975). New variables \( r_{i}^{1} \) and \( r_{i}^{2} \) are defined to represent the bilinear terms \( T_{i} Y_{i} \) and \( T_{i} Y_{i} \) in equation (11) and an additional set of constraints given by equations (18) to (21), is imposed on the formulation.
Similarly we replace the terms $T_{n}V_{j_{n}}$ and $T_{n}V_{j_{n}}$ in the minimum run length constraints of equation (8) with a single continuous variable $\tau_{j_{n}}^{3}$ and $\tau_{j_{n}}^{4}$ and add the equivalent constraints represented by the following equations:

\[ MV_{j_{n}} \geq \tau_{j_{n}}^{3} \geq T_{n} + MV_{j_{n}} - M \quad \forall j, \eta \quad (22) \]
\[ T_{n} \geq \tau_{j_{n}}^{3} \quad \forall j, \eta \quad (23) \]
\[ MV_{j_{n}} \geq \tau_{j_{n}}^{4} \geq T_{n} + MV_{j_{n}} - M \quad \forall j, \eta \quad (24) \]
\[ T_{n} \geq \tau_{j_{n}}^{4} \quad \forall j, \eta \quad (25) \]

The term $V_{j_{n}}V_{j_{n}}$ in the minimum run length constraint of equation (8) involves two binary variables and makes the formulation nonlinear. This nonlinearity is removed by replacing the term $V_{j_{n}}V_{j_{n}}$ with the continuous variable $\sqrt{V_{j_{n}\eta}}$ and introducing additional constraints given by:

\[ V_{j_{n}} + V_{j_{n}} - \sqrt{V_{j_{n}\eta}} \leq 1 \quad \forall j, \eta, \eta \quad (26) \]
\[ \sqrt{V_{j_{n}\eta}} \leq V_{j_{n}} \quad \forall j, \eta, \eta \quad (27) \]
\[ V_{j_{n}\eta} \leq V_{j_{n}} \quad \forall j, \eta, \eta \quad (28) \]

Similarly the nonlinear term $B_{i}^{uplift}Z_{i,j_{n}+1}$ in equation (3) is replaced by the continuous variable $\hat{B}_{i}^{uplift}$ and the following constraints:

\[ MZ_{i_{q}} \geq \hat{B}_{i}^{uplift} \geq B_{i}^{uplift} + MZ_{i_{q}} - M \quad \forall i \in I_{o}, \eta \quad (29) \]
\[ B_{i}^{uplift} \geq \hat{B}_{i}^{uplift} \quad \forall i \in I_{o} \quad (30) \]

and $T_{n}Z_{i_{q}}$ in equation (12) is replaced by the continuous variable $\tau_{i_{q}}^{'}$ and the following constraints need to be added:

\[ MZ_{i_{q}} \geq \tau_{i_{q}}^{'} \geq T_{n} + MZ_{i_{q}} - M \quad \forall i \in I_{o}, \eta \quad (31) \]
\[ T_{n} \geq \tau_{i_{q}}^{'} \quad \forall i \in I_{o}, \eta \quad (32) \]
The nonlinear term involving the product of two binary terms in equation (6) is removed by replacing the term $Y_{i\eta}Y_{i',\eta+1}$ with the continuous variable $\bar{Y}_{i\eta}$ and introducing additional constraints given by:

\[
\begin{align*}
Y_{i,\eta+1} + Y_{i\eta} - \bar{Y}_{i\eta} &\leq 1 & \forall i \in I, \eta \\
\bar{Y}_{i\eta} &\leq Y_{i\eta} & \forall i \in I, \eta \\
\bar{Y}_{i\eta} &\leq Y_{i,\eta+1} & \forall i \in I, \eta
\end{align*}
\]

(33) \quad (34) \quad (35)

The resulting final formulation is the following mixed integer linear programming problem.

Minimize

\[
\sum_{\eta} \sum_{i \in I} r_{i} B_{i\eta}^{\text{upflf}} + \sum_{\eta} \sum_{j} q_{j} V_{j\eta} + \sum_{i \in I} u_{i} B_{i}^{\text{upflf}}
\]

(36)

subject to:

\[
S_{e,\eta+1} = S_{e\eta} + a_{e}(T_{\eta+1} - T_{\eta}) - \sum_{i \in I} f_{i} D_{i\eta} - \sum_{i \in I} B_{i\eta}^{\text{upflf}} &\quad \forall e \in E_{\text{upf}}, \eta
\]

(37)

\[
S_{e,\eta+1} = S_{e\eta} + \sum_{i \in I} p_{i} f_{i} D_{i\eta} - \sum_{i \in I} f_{i} D_{i\eta} &\quad \forall e \in E_{\text{int}}, \eta
\]

(38)

\[
S_{e,\eta+1} = S_{e\eta} + \sum_{i \in I} p_{i} f_{i} D_{i\eta} - \sum_{i \in \{I,I'\}} d_{i} Z_{i,\eta+1} + \sum_{i \in \{I,I'\}} \hat{B}_{i}^{\text{upflf}} &\quad \forall e \in E_{\text{fix}}, \eta
\]

(39)

\[
MZ_{i\eta} \geq \hat{B}_{i}^{\text{upflf}} \geq B_{i}^{\text{upflf}} + MZ_{i\eta} - M &\quad \forall i \in I, \eta
\]

(40)

\[
B_{i}^{\text{upflf}} \geq \hat{B}_{i}^{\text{upflf}} &\quad \forall i \in I
\]

(41)

\[
0 \leq S_{e\eta} \leq C_{e} &\quad \forall e, \eta
\]

(42)

\[
V_{j\eta} \leq 1 &\quad \forall j, \eta
\]

(43)

\[
V_{j,\eta+1} = 1 - \sum_{i \in I} \bar{Y}_{i\eta} &\quad \forall j, \eta
\]

(44)

\[
Y_{i,\eta+1} + Y_{i\eta} - \bar{Y}_{i\eta} \leq 1 &\quad \forall i \in I, \eta
\]

(45)

\[
\bar{Y}_{i\eta} \leq Y_{i\eta} &\quad \forall i \in I, \eta
\]

(46)

\[
\bar{Y}_{i\eta} \leq Y_{i,\eta+1} &\quad \forall i \in I, \eta
\]

(47)

\[
\sum_{i \in I} Z_{i\eta} + \sum_{j} V_{j\eta} \geq 1 &\quad \forall \eta
\]

(48)

\[
\tau_{\eta}^{3} - \tau_{\eta}^{4} \geq b_{\eta} V_{j\eta} - M(V_{j\eta} - \bar{V}_{j\eta}) &\quad \forall j, \eta, \eta
\]

(49)

\[
MV_{j\eta} \geq \tau_{\eta}^{3} \geq T_{\eta} + MV_{j\eta} - M &\quad \forall j, \eta
\]

(50)

\[
T_{\eta} \geq \tau_{\eta}^{3} &\quad \forall j, \eta
\]

(51)

\[
MV_{j\eta} \geq \tau_{\eta}^{4} \geq T_{\eta} + MV_{j\eta} - M &\quad \forall j, \eta
\]

(52)
\[ T_\eta \geq \tau^4_{\eta_i} \forall j, \eta \]  
\[ V_{j_{\eta_i}} + V_{j_{\eta}} - \bar{V}_{j_{\eta_i}} \leq 1 \forall j, \eta, \eta_i \]  
\[ \bar{V}_{j_{\eta_i}} \leq V_{j_{\eta_i}} \forall j, \eta, \eta_i \]  
\[ \bar{V}_{j_{\eta_i}} \leq V_{j_{\eta_i}} \forall j, \eta, \eta_i \]  
\[ T_{\eta+1} - T_{\eta} \geq 0.05 \forall \eta \]  
\[ \sum_{i \in I_j} Y_{i_{\eta}} = 1 \forall \eta \]  
\[ \tau^1_{i_{\eta}} - \tau^2_{i_{\eta}} = D_{i_{\eta}} \forall i, \eta \]  
\[ M Y_{i_{\eta}} \geq \tau^1_{i_{\eta}} \geq T_{\eta+1} + M Y_{i_{\eta}} - M \forall i \notin I_o, i \notin I_{\text{spill}}, \eta \]  
\[ T_{\eta+1} \geq \tau^1_{i_{\eta}} \forall i \notin I_o, i \notin I_{\text{spill}}, \eta \]  
\[ M Y_{i_{\eta}} \geq \tau^2_{i_{\eta}} \geq T_{\eta} + M Y_{i_{\eta}} - M \forall i \notin I_o, i \notin I_{\text{spill}}, \eta \]  
\[ T_{\eta} \geq \tau^2_{i_{\eta}} \forall i \notin I_o, i \notin I_{\text{spill}}, \eta \]  
\[ \tau^1_{i_{\eta}} - \text{due}dt_{i_{\eta}} Z_{i_{\eta}} = 0 \forall i \in I_o, \eta \]  
\[ \sum_{\eta} Z_{i_{\eta}} = 1 \forall i \in I_o \]  
\[ M Z_{i_{\eta}} \geq \tau^1_{i_{\eta}} \geq T_{\eta} + M Z_{i_{\eta}} - M \forall i \in I_o, \eta \]  
\[ T_{\eta} \geq \tau^1_{i_{\eta}} \forall i \in I_o, \eta \]  
\[ \sum_{\eta} (T_{\eta+1} - T_{\eta}) = H - 1 \forall \eta \]  
\[ T_{\eta} \leq H \forall \eta \]  
\[ T_{\eta} \geq 1 \forall \eta \]  
\[ \sum_{\eta} \sum_{i \in I_j} D_{i_{\eta}} \leq H - 1 \forall j \]  

The above MILP can be solved by any of the standard available packages.

## 5 Computational Experience

The MILP model proposed in the previous section is modeled in GAMS/XA on a 2.4 GHz Pentium 4 workstation with 1 GB RAM. The fixed data shown in Table 2 are real data collected from a refinery. The per unit spillage costs for the spillage tasks \( r_i \) and the per unit upliftment failure penalties for the final states \( u_i \) have been masked. However, their relative importances have been reflected in the values used. The tank capacities are assumed to be 10000 Mt for easy comparison of the input data for the cases considered.
For the fixed refinery data in Table 2, we show how the model performed in a total of 7 cases. For each case, the scheduling horizon was 14 days, day 1 to day 14. The input of a case consists of the upliftment schedule and the initial stock positions at all the four levels at the beginning of day 1. It must be noted that the values of $V_{j_1}$ and $V_{j_{max}}$ are predefined to 1 for all $j$. The model, therefore, constrains the last run of a stream in a scheduling horizon to be greater than the minimum run-length.

<table>
<thead>
<tr>
<th>Stream</th>
<th>Input Rate</th>
<th>Feed rate of stream g in unit j: (Mt/PD)</th>
<th>Yield Percentage of stream g in unit j:</th>
<th>Spill Penalty</th>
<th>Upliftment Penalty</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$(a_e e \in E_{in} &amp; e \in E^o)$</td>
<td>FEU</td>
<td>SDU</td>
<td>HFU</td>
<td>FEU</td>
</tr>
<tr>
<td>1</td>
<td>620</td>
<td>1250</td>
<td>780</td>
<td>546.0</td>
<td>60</td>
</tr>
<tr>
<td>2</td>
<td>255</td>
<td>1030</td>
<td>576</td>
<td>403.2</td>
<td>50</td>
</tr>
<tr>
<td>3</td>
<td>310</td>
<td>1050</td>
<td>521</td>
<td>364.7</td>
<td>65</td>
</tr>
</tbody>
</table>

Table 3: Base Case data

<table>
<thead>
<tr>
<th>Tank Stock Positions</th>
<th>Stock Positions at the start of day 1 (Mt)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Streams</td>
<td>Level 0</td>
</tr>
<tr>
<td>1</td>
<td>6370</td>
</tr>
<tr>
<td>2</td>
<td>9255</td>
</tr>
<tr>
<td>3</td>
<td>9810</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scheduled Upliftments (Mt)</th>
<th>Streams</th>
</tr>
</thead>
<tbody>
<tr>
<td>Day</td>
<td>Streams</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
</tr>
<tr>
<td>1500</td>
<td>2</td>
</tr>
<tr>
<td>5000</td>
<td>3</td>
</tr>
<tr>
<td>3140</td>
<td></td>
</tr>
</tbody>
</table>

To generate these test cases, we first designed a base case as given in Table 3. A close look at the table would suggest that the initial tank stocks at level 0 are such that more than one stream would start spilling within next $b$ (minimum run length) days, if not processed. The Input Rate column (Table 2) along with the Level 0 stocks (Table 3) considered together, show that stocks at level 0 (stream 1, steam 2, stream 3), if not processed in $U_1$, would start spilling on day 6, day 3 and day 1 respectively. It may be noted that, given the available time horizon, the required quantities for the upliftment on 14th cannot be satisfied. Given the stock
positions at level 3 at the start of day 1, Unit 3 would require 13.10 days of processing. However, to meet the requirement of stream 1, Unit 3 has to process it only for one day. But, if scheduled, stream 1 has to be run for 3 days because of the minimum run length constraint. Thus, all the upliftments on 14th cannot be met.

An appropriate choice for the number of event points in a continuous time model is an unresolved issue. However, like in most of the other continuous time models, we also follow an approach wherein we assume a small number of event points to begin with and solve the model for this given number of event points. Thereafter, in each iteration, we increase the number of event points by one and every time we solve the model. This we go on doing till there is no improvement in the objective value or the time limit exceeds 12 hrs. That is we stop when consideration of an additional event point does not result in any further improvement in the objective value or the time limit imposed is exceeded.

For the problem with 3 product lines and 3 processing units with no upliftments there are 12 tasks and 12 states. We started solving the base case initially with 5 event points as given in table 5. It terminated in 15 seconds with a solution cost of 17258.9. Trying it out with 6 event points increased the time to 2 min 16 seconds, but brought down the solution cost to 8340. We experimented again with 7 event points. The time required now was 1 hour 6 minutes 42 seconds. We report this cost as the final solution for the problem. Figure 4 is the Gantt chart of the Units’ schedules as suggested by the model.

It may be observed that the same base case was tried as in the paper by Bose and Bhattacharya (2005a) using STN formulation. The objective function value was 6169.86 there compared to 8340 here. It may be seen that stream 1 has not at all been scheduled in unit 3 in the output given by the continuous time model. This is due to the constraints imposed at the beginning of the scheduling horizon in our continuous time model. At the beginning of horizon, the units are supposed to be changing over to new streams and hence have to complete the minimum run lengths. Similarly we have forced every unit to complete the minimum run lengths at the last event point that flushes with the end of the scheduling horizon. Horizon effects in the formulation of short term scheduling problem is quite common and this one also is not an exception.
Table 4 shows the different test case instances that include the base case as case 1. These cases were generated by varying the input parameters of the base case. Each case was obtained by altering the initial stock positions and/or the streams to be uplifted along with their amounts. Table 5 summarizes the test results for each of these cases. For each case, we report the solution cost, the number of iterations and the time that GAMS/XA took to locate the optimal solution for a given number of event points. For all the cases the time limit was set to 12 hrs.

6 Conclusion

To the best of our knowledge this is the first attempt to develop continuous time model based on global event points for a class of scheduling problems normally occurring in continuous process plants. Our formulation overcomes the problem of pre-fixing some of the event points on the upliftment dates through the introduction of $Z_{\eta}$ variables. So our application of global event point is more general than the way it has been used by the earlier researchers.

The model needs to be further worked upon in two different counts. Firstly, it needs to be mathematically shown that allowing further event points would not improve the solution. It is our conjecture that once the objective value stabilizes with increase of event points, it cannot
be improved further by allowing more event points. Secondly, the horizon effects as discussed in connection with the experimental results have to be dealt with. However, the model allows reasonably good solution within reasonable time for scheduling horizons of two weeks. It may be noted that the complexity of the model grows more with the number of event points than with the scheduling horizon. For scheduling horizon of one month or more, we suggest the heuristic solution procedure proposed earlier in the paper by Bose and Bhattacharya(2005b).

### Table 4: Case Instances

<table>
<thead>
<tr>
<th>Case No.</th>
<th>Levels</th>
<th>Stock Position At start of day 1 (Mt)</th>
<th>Scheduled Upliftings (Mt)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>6370 3750 2000 1000</td>
<td>14 1500 5000 3140</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>9255 4424 5403.2 1000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>9810 5000 4635.3 2361.053</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>6370 3750 2000 1000</td>
<td>8 1500 3000 2200</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>9255 4424 5403.2 1000</td>
<td>14 1000 2000 1000</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>9810 5000 4635.3 2361.053</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>6370 3750 2000 1000</td>
<td>14 5000 3140 1500</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>9255 4424 5403.2 1000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>9810 5000 4635.3 2361.053</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>6370 3750 2000 1000</td>
<td>8 1500 2000 2200</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>9255 4424 5403.2 1000</td>
<td>14 1800 3000 1000</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>9810 5000 4635.3 2361.053</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>6370 3750 2000 1000</td>
<td>14 5900 - 2700</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>9255 4424 5403.2 1000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>9810 5000 4635.3 2361.053</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>6370 8750 2000 1000</td>
<td>14 5900 - 2700</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>9255 7424 2403.2 1000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>9810 8000 1635.3 2361.053</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>6370 3750 1000 1000</td>
<td>14 5900 - 2700</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>9255 8424 1403.2 2000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>9810 9000 635.3 2361.053</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>6370 3750 2000 1000</td>
<td>8 1500 2000 1000</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>6255 4424 5403.2 1000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>6810 5000 4635.3 2361.053</td>
<td></td>
</tr>
</tbody>
</table>

### Table 5: Results of the Case Instances

<table>
<thead>
<tr>
<th>Case</th>
<th>Number of event points</th>
<th>Solution Cost</th>
<th>Iterations</th>
<th>Time (Hr:Min:Sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
<td>17258.9</td>
<td>113,680</td>
<td>00:00:15</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>8,340.0</td>
<td>784,536</td>
<td>00:02:16</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>8,340.0</td>
<td>20,820,734</td>
<td>01:06:42</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>22439.9</td>
<td>23,307</td>
<td>00:00:02</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>14540.0</td>
<td>632,947</td>
<td>00:01:49</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>8,340.0</td>
<td>19,849,176</td>
<td>01:05:41</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>8,340.0</td>
<td>166,289,017</td>
<td>10:26:56</td>
</tr>
</tbody>
</table>
### Reference:


