Modeling VP Operation: The Diwali Festival Problem
Rushikesh K. Joshi, Subash Rajaa
Department of Computer Science and Engineering
Indian Institute of Technology Bombay

Abstract
A new higher-level synchronization problem called Diwali Festival Problem (DFP) is presented. The problem captures the requirements for implementing the VP operation for semaphores outside the operating system kernel in presence of P and V operations, and shared variables with atomic reads and writes.

1 VP Operation and DFP
The VP operation [1] for semaphores proposed by Tai and Carver eliminates a subtle error in semaphore based solutions. The error results due to an erroneous assumption that in a sequence V(S1);P(S2), after a process has completed V(S1), it also completes P(S2) before any other process completes V(S2) or P(S2). Further, the VP operation allows us to write neater solutions to certain synchronization problems. Tai and Carver’s definition of VP operation [1] is as shown below.

An execution of VP(S1,S2) by a process T is equivalent to that of V(S1);P(S2) except that when T starts the V(S1) operation, T is guaranteed to be the next process to access S2 (i.e. T executes P(S2) before any other process executes P(S2) or V(S2)).

The Diwali Festival Problem (DFP) presented subsequently captures the requirements for implementing the VP operation outside the operating system kernel. In other words, a solution to DFP is also a solution to VP. We believe that the DFP would aid in deriving elegant implementations for the VP operation outside the OS kernel. DFP can also be posed as a synchronization problem in OS courses.

2 An Implicit Constraint in Definition of VP
The definition of VP by Tai and Carver has an implicit constraint that can be missed easily. Consider the following interleaving from two concurrent processes T1 and T2: (1) Process T1 executes VP(S1,S2) (2) Process T2 executes VP(S3,S2)

The following partial interleaving is seemingly permitted by Tai and Carver’s definition: (1) Process T1 executes the V(S1) of VP(S1,S2) (2) Process T2 executes the V(S3) of VP(S3,S2)

Now according to Tai and Carver’s definition, both T1 and T2 require that they access S2 before each other, which cannot be satisfied. This situation can be avoided only if other VP(*,S2) operations are prohibited from starting between the V(S1) and P(S2) of any given VP(S1,S2) operation. Thus, Tai and Carver’s definition possesses this implicit constraint.

This situation illustrates that the correct way of interpreting the definition of VP, is to look upon the condition for VP as a rule which should hold in any trace of the program.
3 Diwali Festival Problem

The DFP was developed to model the problem of writing a solution to the VP problem as a benchmark problem of synchronization. The synchronization problem is framed in a festive setting of Diwali, a festival of food and lights. The problem is to be solved with P and V operations of general semaphores, and shared variables with atomic reads and writes. However, there is no restriction on the number of semaphores to be used. It can be assumed that the semaphore queues are FCFS (first come first served) type.

The problem statement

Packets of Diwali delicacies are distributed at different counters in a distribution hall. For each kind of delicacy, one exclusive counter is identified. A basket containing packets of the delicacy is kept at every counter.

Three kinds of people are allowed to enter the distribution hall: children, suppliers and hungry suppliers. Their behavior is bound by the below protocol.

Children: After entering the hall, a child joins the queue at one of the counters. When its turn arrives, the child picks up a packet from the basket at the counter, and then leaves the hall. If the basket is found to be empty, the child at the head of the queue waits until a packet is dropped into the basket.

Suppliers: A supplier comes into the hall with one packet of a delicacy and drops it into the basket at the appropriate counter. The supplier then leaves the hall.

Hungry Supplier: A hungry supplier drops a packet into the appropriate basket as in the case of a supplier. In addition, a hungry supplier also gets queued up at any desired delicacy counter. Hungry supplier leaves the hall after obtaining the needed delicacy packet.

The following rule is observed in the hall:

After a hungry supplier interested in eating a delicacy D2 drops a packet into the basket for delicacy D1, and until the hungry supplier gets queued up on the counter for D2:
1. No other child or hungry supplier interested in delicacy D2, gets queued ahead of this hungry supplier; and
2. No other supplier or hungry supplier drops a packet into the basket for D2.

4 Mapping DFP onto VP

Each delicacy can be considered as a semaphore and actions performed by the child, supplier and hungry supplier can be considered as P, V, and VP respectively. The following table summarizes the mapping:

<table>
<thead>
<tr>
<th>In DFP</th>
<th>In VP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Queuing up on the counter for delicacy D</td>
<td>P(D)</td>
</tr>
<tr>
<td>Dropping a packet in delicacy D's basket</td>
<td>V(D)</td>
</tr>
<tr>
<td>Child</td>
<td>Process that executes a P()</td>
</tr>
<tr>
<td>Supplier</td>
<td>Process that executes a V()</td>
</tr>
<tr>
<td>Hungry Supplier</td>
<td>Process that executes a VP()</td>
</tr>
</tbody>
</table>
5 A Relaxation in DFP

We believe that the rule for DFP can be relaxed without affecting the synchronizing effect of the VP operation. The modified rule is stated as follows:

After a hungry supplier interested in eating a delicacy drops a packet into a basket, no other child of hungry supplier gets queued ahead of this hungry supplier at the counter of delicacy, in which the hungry supplier is interested.

The relaxation removes condition 2 from the rule for DFP stated in Section 3. With this modified rule in DFP, the condition for VP operation is modified as:

An execution of VP(S1,S2) by a process T is equivalent to that of V(S1);P(S2) except that when T starts the V(S1) operation, T executes P(S2) before any other process executes P(S2).

This condition allows V(S2)'s to occur between V(S1) and P(S2) of a VP(S1,S2) operation. We are working on a proof using Lipton's reduction method [2] to show that this relaxed condition for VP has the same synchronizing effect as that of the original condition proposed by Tai and Carver.

6 Conclusion

The VP problem was formulated as a higher-level synchronization problem called Diwali Festival Problem. We expect that this will help in thinking about the VP problem in more creative ways, resulting in elegant implementations outside the OS kernel.

References
