Operational transform
Set theoretic operations

30 Aug 2005

David Barrett-Lennard

Abstract
This paper provides a solution to the operational transform of set theoretic operations.

Introduction
The document state is assumed to be a set over some element type T. T is a value type, and is assumed to support an equality comparison operator. In this paper we study the operational transform of the following set theoretic operations

- ins(v) - insert value v into the set (if not present)
- del(v) - delete value v from the set (if present)

Each of these operations are idempotent.

We need convergence at quiescence even though operations are executed in different orders at different sites. Specifically we require properties TP1 and TP2 [1].

Properties required to achieve convergence

TP1
$\forall\ O_1, O_2 \quad S + [O_1\ IT(O_2, O_1)] = S + [\ O_2\ IT(O_1, O_2)\ ]$

TP2
$\forall\ O_1, O_2, O_3, \quad IT(\ IT(\ O_3, O_1),\ IT(O_2, O_1)) = IT(\ IT(\ O_3, O_2),\ IT(O_1, O_2))$

Discussion
In the following cases, let O_1, O_2 be concurrent operations that are performed on the same document state S.

Let operation O_1 insert value v_1, and O_2 inserts value v_2. The set theoretic union operation is associative and commutative so therefore $(S \cup \{v_1\}) \cup \{v_2\} = (S \cup \{v_2\}) \cup \{v_1\}$. Hence operations O_1, O_2 commute and therefore nothing needs to be done under IT.

Let operation O_1 delete value v_1, and O_2 delete value v_2. According to set theory $(S \setminus \{v_1\}) \setminus \{v_2\} = (S \setminus \{v_2\}) \setminus \{v_1\}$. Hence operations O_1, O_2 commute and therefore nothing needs to be done under IT.

Let operation O_1 insert value v_1, and O_2 delete value v_2. Assuming $v_1 \neq v_2$, $(S \cup \{v_1\}) \setminus \{v_2\} = (S \setminus \{v_2\}) \cup \{v_1\}$. Hence operations O_1, O_2 commute and therefore nothing needs to be done under IT.

The only difficult case is where O_1 inserts value v, and O_2 deletes value v. It is proposed that we disable a delete operation when it IT's past an insert operation of the same value, so at quiescence all sites agree that the value was inserted This is implemented using a disable count, so that a delete operation may be disabled multiple times as it IT's past a number of insert operations. Under ET, the delete operation will only be re-enabled when it has ET'd backward past all the insert operations that caused it to be disabled.

Solution

Let an operation contain the following fields

Field	Description
ins	Boolean flag to indicate whether this is an insert or delete operation
dc	Disable count, only relevant to a delete operation. Initialised to zero when the operation is first generated.
v	Value to be inserted or deleted

The following C++ code shows the implementation of IT/ET

```
void IT(SetOp& O1, const SetOp& O2)
{
    if (O2.ins && !O1.ins && O1.v == O2.v) ++O1.dc;
}

void ET(SetOp& O1, const SetOp& O2)
{
    if (O2.ins && !O1.ins && O1.v == O2.v) --O1.dc;
}
```

Future work

1. In order to support undo it is necessary to flag whether an operation actually inserted or deleted a value. This flag must undergo transformation under IT/ET
2. For performance this solution should be extended to support operations that insert or delete multiple elements at once. This also allows for operations to be closed under merging. Furthermore it will be possible to compress merged operations by eliminating elements that are inserted then deleted by the operation. Compression can also discard disabled delete operations.

References

[1]	David Barrett-Lennard. Operational transform - Control algorithm. Aug 2005