Università della Svizzera italiana

Faculty of Informatics

USI Technical Report Series in Informatics

Proof

Eduardo[1], Fernando[1]

[1] Faculty of Informatics, Università della Svizzera italiana, Switzerland

Abstract

-

Report Info

Published
December 2012

Number
USI-INF-TR-2010-4

Institution
Faculty of Informatics
Università della Svizzera italiana
Lugano, Switzerland

Online Access
www.inf.usi.ch/techreports

1 ...

Variables are distributed among partitions based on a function $h(i)$ that maps each variable v_i to one partition $P_{h(i)}$. Therefore, no two partitions contain the same variable. The set $\mathscr{C}$ contains all commands allowed by the application. Every command C in $\mathscr{C}$ is associated with a variable set $\mathscr{V}$ and a partition set $\mathscr{P}$. Set $\mathscr{V}$ contains the variables accessed (read/written) by C, while $\mathscr{P}$ is the set of partitions that hold variables in $\mathscr{V}$, i.e., $\mathscr{P} = \{P_{h(i)} : v_i \in \mathscr{V}\}$. C is executed by partition P_x, where x is the lowest subscript in $\mathscr{P}$, i.e., $x = min_{P_i \in \mathscr{P}}(i)$.

Communication between partitions is done with oam-cast, mapping each partition to a multicast group. Such multicast protocol is quasi-genuine, requiring the creating of a *sendersTo* relation. We define it as follows: if the application allows a command C to be associated with a certain partition set $\mathscr{P}$, then every partition in $\mathscr{P}$ can send to any partition in $\mathscr{P}$. More formally, we have:

$$\exists C \in \mathscr{C} \Rightarrow P_i \in sendersTo(P_j) : P_i \in \mathscr{P} \wedge P_j \in \mathscr{P}$$

(I'm not sure the following is necessary, but here it goes)

When a client issues a command C, it simply sends C to some replica in P_x, which will then proceed with oam-cast.

1.1 Proof

Proposition 1. *Let $\mathscr{E}$ be any execution of SSMR. Let C_1 and C_2 be any two commands in $\mathscr{E}$ such that $C_1 \prec C_2$ in real-time. C_2 will not precede C_1 in $\mathscr{E}$.*

Proof. There are x situations in which a command has to precede other in an execution. They are:

i) C_1 anc C_2 access the same variable v_x.

In this case, we have two possibilities:

a) C_1 and C_2 are executed in the same partition.
   Linearizability in this case is obvious, since there is no way to

b) C_1 and C_2 are executed in different partitions.
   In this case, necessarily there is a variable $v_y \neq v_x$, accessed by one of the two commands.

ii) C_1 and C_2 are such that $\mathscr{P}_1 \cap \mathscr{P}_2 = \emptyset$

□