Algorithm 3: A part of Relationship Reduction Algorithm

Input: *CSP# procs of the participants* and *Relationships*: *Init*, *End*, *Next*, *And*, *Xor*

```
// get Activate
for ap in AtomicProcesses do
    // find a composite csp# proc nxtCP with End[] and Next[] (if ap
       ends, then nxtCP is the next composite proc to start. It looks like
       searching up (with the help of End[])in the syntax tree)
    nxtCP=FindNextCompositeProc(ap);
    for child in Init[nxtCP] do
        // find atomic processes in Init[nxtCP]. It looks like searching
           down in the syntax tree
        nxtAP = FindAtomicProcesses(child); // with the help of Init[]
        add nxtAP to Activate[ap];
for xorp in ExclusiveGatewayProcesses do // xorp is a composite csp# proc
 that contains all atomic procs in a EG. Here we assume there are two
 outgoing paths
    // find the first atomic processes in every outgoing path
    // with the help of Init[]
    FirstFAPGroup=FindFisrtAtomicProcesses(Xor[xorp][1]);
    SecondFAPGroup=FindFisrtAtomicProcesses(Xor[xorp][2]);
    for ap in FirstAPGroup do
        add SecondAPGroup to Inactivate[ap];
    for ap in SecondAPGroup do
        add FirstAPGroup to Inactivate[ap];
for andp in ParallelGatewayProcesses do
    // find the last atomic processes in every outgoing path
    // with the help of Init[] and Next[]
    FirstLAPGroup=FindLastAtomicProcesses(And[andp][1]);
    SecondLAPGroup=FindLastAtomicProcesses(And[andp][2]);
    for ap in FirstLAPGroup do
        add SecondLAPGroup to Inactivate[ap];
    for ap in SecondLAPGroup do
        add FirstLAPGroup to Inactivate[ap];
return Activate,Inactivate, Parallel;
```