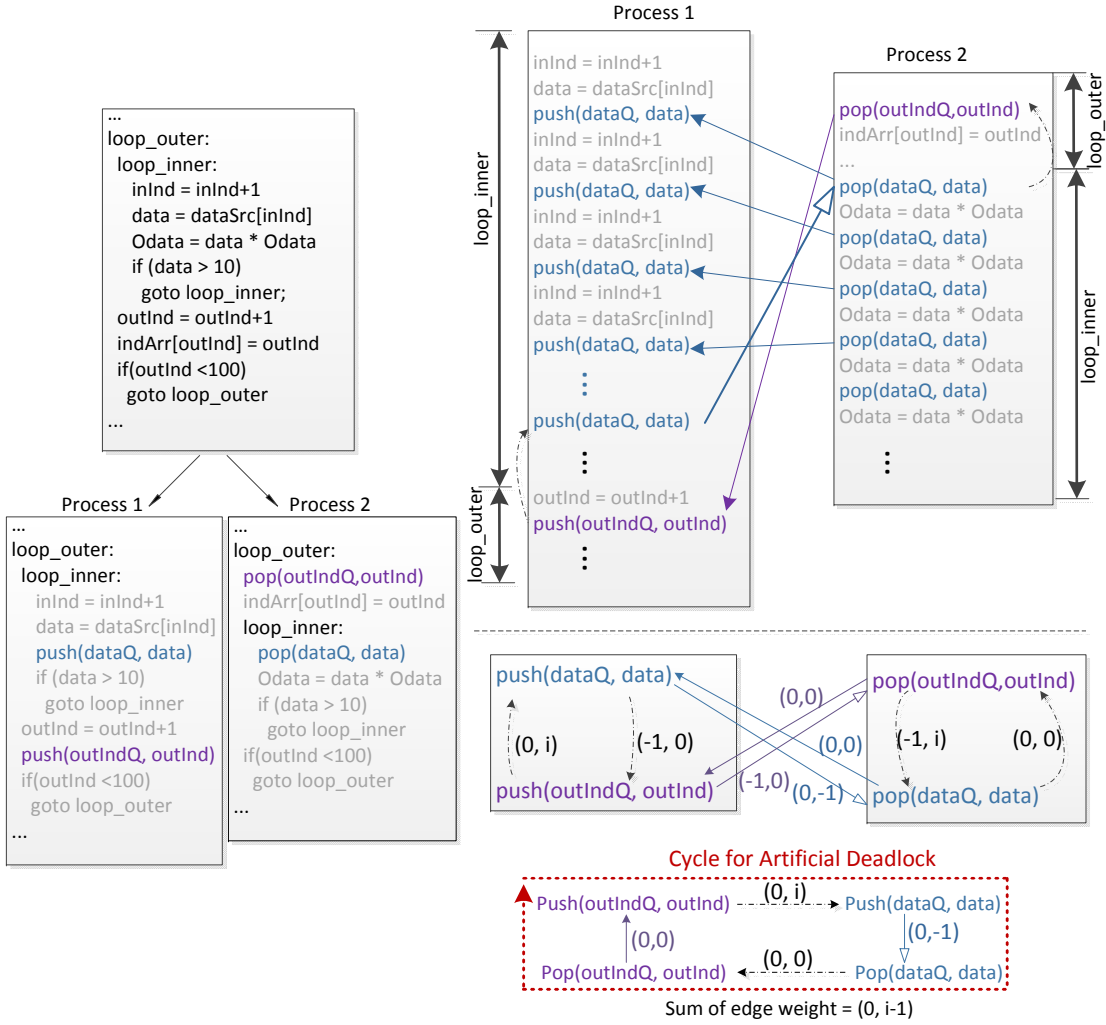
...
loop_outer:
loop_inner:
inInd = inInd+1
data = dataSrc[inInd]
Odata = data * Odata
if (data > 10)
goto loop_inner;
outInd = outInd+1
indArr[outInd] = outInd
if(outInd <100)
goto loop_outer
...
Process 1
...
loop_outer:
loop_inner:
inInd = inInd+1
data = dataSrc[inInd]
push(dataQ, data)
if (data > 10)
goto loop_inner
outInd = outInd+1
push(outIndQ, outInd)
if(outInd <100)
goto loop_outer
...
Process 2
...
loop_outer:
pop(outIndQ,outInd)
indArr[outInd] = outInd
loop_inner:
pop(dataQ, data)
Odata = data * Odata
if (data > 10)
goto loop_inner
if(outInd <100)
goto loop_outer
...
Process 1
loop_inner
loop_outer
inInd = inInd+1
data = dataSrc[inInd]
push(dataQ, data)
push(dataQ, data)
outInd = outInd+1
push(outIndQ, outInd)
Process 2
loop_outer
loop_inner
pop(outIndQ,outInd)
indArr[outInd] = outInd
pop(dataQ, data)
Odata = data * Odata
push(dataQ, data)
(0, i)
(-1, 0)
push(outIndQ, outInd)
(0,0)
(-1,0)
(0,-1)
pop(outIndQ,outInd)
(-1, i)
(0, 0)
pop(dataQ, data)
Cycle for Artificial Deadlock
Push(outIndQ, outInd)
(0, i)
Push(dataQ, data)
(0,-1)
Pop(outIndQ, outInd)
(0,0)
(0, 0)
Pop(dataQ, data)
Sum of edge weight = (0, i-1)