Algorithm 1: Serial Version of Personalized PageRank with Random Walk

Inputs: a directed graph of $\boldsymbol{N}$ nodes, source node index $\boldsymbol{s}$, teleport probability $\boldsymbol{\alpha}$, maximum iteration time $\boldsymbol{maxIter}$, tolerance $\boldsymbol{\varepsilon}$, convergence checking steps $\boldsymbol{step}$, a function produces random numbers in interval $[0,1)$.

Outputs: an array **PPR** for Personalized PageRank value of each node in the graph.

Procedure:

if $\text{out}(p_s) \neq \emptyset$ **then:**
  $\text{count}[] \coloneqq (0,0,\dots,0)^T$
  $\text{count}_{\text{old}}[] \coloneqq (0,0,\dots,0)^T$
  $i \coloneqq s$
  for $iter \coloneqq 0$ **to** $(maxIter - 1)$**:**
    if $\text{random}() < \alpha$ **or** $\text{out}(p_i) = \emptyset$ **then:**
      $j \coloneqq s$
    else:
      choose a node p_j in $\text{out}(p_i)$ randomly
    end if
    $\text{count}[j] \coloneqq \text{count}[j] + 1$
    $i \coloneqq j$
    if $iter + 1 \equiv 0 (\text{mod } step)$ **then:**
      if $iter + 1 > step$ **then:**
        $r \coloneqq \text{sampleCorrelationCoefficient}(\text{count}_{\text{old}}, \text{count})$
        if $r \geq 1 - \varepsilon$ **then:** // convergence checking
          $iter \coloneqq iter + 1$
          break
        end if
      end if
      $\text{count}_{\text{old}}[] \coloneqq \text{copy}(\text{count}[])$
    end if
  end for
  $\text{PPR}[] \coloneqq \frac{1}{iter} \times \text{count}[]$
else:
  $\text{PPR}[] \coloneqq (0,0,\dots,0)^T$
end if
return PPR[]

Algorithm 2: Parallel Version of Personalized PageRank with Random Walk

Inputs: number of process $\boldsymbol{P}$, a directed graph of $\boldsymbol{N}$ nodes, source node index $\boldsymbol{s}$, teleport probability $\boldsymbol{\alpha}$, maximum iteration time $\boldsymbol{maxIter}$, tolerance $\boldsymbol{\varepsilon}$, convergence checking steps $\boldsymbol{step}$, a function produces random numbers in interval $[0,1)$.

Outputs: an array **PPR** for Personalized PageRank value of each node in the graph.

Procedure:

if $\text{out}(p_s) \neq \emptyset$ **then:**
  for each process do:
    $\text{count}[] \coloneqq (0,0,\dots,0)^T$
    $totalIter \coloneqq 0$
    $\text{totalCount}[] \coloneqq (0,0,\dots,0)^T$
    $\text{totalCount}_{\text{old}}[] \coloneqq (0,0,\dots,0)^T$
    $pMaxIter \coloneqq \lceil maxIter/P \rceil$
    $pStep \coloneqq \lceil pStep/P \rceil$
    $i \coloneqq s$
    for $iter \coloneqq 0$ **to** $(pMaxIter - 1)$**:**
      if $\text{random}() < \alpha$ **or** $\text{out}(p_i) = \emptyset$ **then:**
        $j \coloneqq s$
      else:
        choose a node p_j in $\text{out}(p_i)$ randomly
      end if
      $\text{count}[j] \coloneqq \text{count}[j] + 1$
      $i \coloneqq j$
      if $iter + 1 \equiv 0 (\text{mod } pStep)$ **then:**
        sum reduce count[] from all process to totalCount[] of process 0.
        $converged \coloneqq$ **false**
        if current process is process 0 **then:**
          if $iter + 1 > step$ **then:**
            $r \coloneqq \text{sampleCorrelationCoefficient}(\text{count}_{\text{old}}, \text{count})$
            if $r \geq 1 - \varepsilon$ **then:** // convergence checking
              $converged \coloneqq$ **true**
            end if
          end if
          $\text{totalCount}_{\text{old}}[] \coloneqq \text{copy}(\text{totalCount}[])$
        end if
        broadcast $converged$ to all process from process 0.
        if $converged$ **then:**
          $iter \coloneqq iter + 1$
          break
        end if
      end if
    end for
    sum reduce count[] from all process to totalCount[] of process 0.
    sum reduce $iter$ from all process to $totalIter$ of process 0.
    if current process is process 0 **then:**

PPR[] ≔ $\frac{1}{totalIter}$ × totalCount[]

end if

end for

else:

PPR[] ≔ $(0,0,\ldots,0)^T$

end if

return PPR[]