```
Algorithm: RWR_Cluster

Data: $ListofQuerySets \leftarrow \{Lists\,of\ \ sets\ of\ \ start\,Nodes\}$
      $G \leftarrow Global\ PairWise\ BLAST\ BitScore\ Graph$

RWR Parameters:
 $restart\_prob = probability\ of\ returning\ to\ the\ start\ nodes$
$threshold = repeat\ loop\ till\ L1\ error \leq threshold$
$iter\_cutoff = maximum\ number\ of\ iterations$
$K = Step\ size$
Permutation Statistics parameters:
 $num\_replicates = number\ of\ replications$

Result: $ClusterMap:$
        $Map(Query\_Name \rightarrow Map(Node \rightarrow Scores(Statistics)))$

begin
    $Cluster\_Map \leftarrow Map(Query\_Name \rightarrow Map(Node \rightarrow Scores))$

    Random Walk Section:
    forall the $QuerieSet \in ListofQuerySets$
    do
        $Cluster\_Map \leftarrow RWR\ in\ threads(QuerySet, RWR\_Parameters)$
    end

    Permutation Statistics Section:
    $Population \leftarrow Map(Query\_Name \rightarrow$
    $Random\_Permutation(ListofQuerySets, RWRParameters, num\_replicates)$

    Summarize Statistics Section:
    forall the $QuerySet \in Keys(Cluster\_Map)$
    do
        $Cluster : Map(Node \rightarrow Scores) \leftarrow Cluster\_Map(QuerySet)$
        forall the $c.member:\ members\ of\ ClusterofQuerySet$
        do
            $p.value \leftarrow \frac{|scores\ from\ Population(QuerySet) \geq Score|}{size\ of\ Population}$

            $max.norm.score \leftarrow Score\ normalized\ by\ max\ score\ in\ Cluster$
            $species.norm.score \leftarrow$
            $Score\ normalized\ by\ max\ score\ within\ same\ species\ in\ Cluster$

            Update scores of $Cluster$ with [score,p.value,norm.score,species.norm.score]
        end

        Update $Cluster\_Map$ with $Cluster$
    end
end
```

Algorithm: *$RWR\_in\_threads$*

```
Data: $Queries \rightarrow \{Set\,of\,start\,Nodes\}$
      Global variables:
      $G \leftarrow Global\ PairWise\ BLAST\ BitScore\ Graph$
      $threshold = repeat\ loop\ till\ L1\ error \leq threshold$
      $iter\_cutoff = repeat\ loop\ at\ least\ iteration\ threshold\ times\ K = Step\ size$

Result: $PostProbMap = Map(s \rightarrow probability\,to\,end\,walk\,at\,s)$,
        $s \in Nodes\,in\,G\,with\,non-zero\,post-probabilities$

begin
    $LocalGraph \leftarrow K - Step\ Neighborhood\ of\ Queries\ in\ G$
    $W = Adjacency\ matrix\ from\ LocalGraph$
    $index\_name\_map = Map(index\ of\ W \rightarrow Query\_Name)$
    $name\_index\_map = Map(Query\_Name \rightarrow index\ of\ W)$
    $N = Number\ of\ Start\ Nodes$
    $QIndices \rightarrow \{Set\ of\ indices\ of\ Queries\ in\ W\}$
    $p^0 \leftarrow \{p_i^0\}\ for\ i \in indices\ of\ W$

        $$p_i^0 = \begin{cases} \frac{1}{N}, & \text{if} i \in QIndices \\ 0, & \text{otherwise} \end{cases}$$

    $normalize\ W$
    while $|p^{t+1} - p^t| \leq threshold$ and $iter \leq iter\_cutoff$ do
        $p^{t+1} = (1-r)Wp^t + rp^0$
    end
    $pfinal = p^{t+1}$
    $Create\ a\ post\ probability\ map\ from\ p^{final}$ :
    $PostProbMap = Map(s \rightarrow probability\,to\,end\,walk\,at\,s)$,
    $s \in Nodes\,in\,LocalGraph\,with\,nonzero\,values\ in\ p^{t+1}$

    Update Concurrent HashMap holding Results
end
```

```
**Algorithm:** Permutation_Statistics

**Data**:  $ListofQuerySets \rightarrow \{Lists\,of\ query\,sets\}$
           $Degree\_Node\ Map\ (d \rightarrow \{Set\,of\ Nodes\,with\ degree\ d\})$
           Global variables:
           $G \leftarrow Pairwise\ sequence\ similarity\ network$

**Result**: $Population\_Map:$
           $Map(Query\_Name \rightarrow \{population\ of\ post\ probability\ values\}))$

Get Sample Populations for the QuerySets:
 SampleSpace $\leftarrow Get\_SampleSpace\_for\_QuerySets$ Sets(ListofQuerySets)

 Permutation Statistics Section:
$Population\_Map \leftarrow Post\ rwr\ probability\ values\ from\ permuted\ queries$

**forall the** $queryset \in ListofQuerySets$
 **do**
    $q.size \leftarrow number\ of\ nodes\ in\ queryset\ q.name \leftarrow name\ of\ queryset$ **repeat**
        $Perm\_Query \leftarrow$Randomly sample q.size nodes from SampleSpace[q.name]
        $Update\ Population\_n \leftarrow RWR\ in\ threads(Perm\_Query, RWR\_Parameters)$
        $rep++$
    **until** $rep \leq num\_replicates$;
    $Update:\ Population\_Map[n] \leftarrow Population\_n$
**end**
```

```
Algorithm: $Get\_SampleSpace\_for\_QuerySets$

Data:   $ListofQuerySets \rightarrow \{Lists\,of\;QuerySets\}$
        $Degree\_Map\;(d \rightarrow \{Set\,of\;Nodes\,with\;degree\;d\;for\;G\})$

Result: $SampleSpace\_Map:\;Map(querysetid \rightarrow Set\;of\;Nodes)$

forall the $queryset \in ListofQuerySets$
 do
    $queryset.size \leftarrow number\;of\;nodes\;in\;queryset$
    $max.degree \leftarrow max(degree(nodes\;in\;queryset))$
    $min.degree \leftarrow min(degree(nodes\;in\;queryset))$

    $SampeSpace \leftarrow \{Pooled\;sets\;of\;nodes\;from\;Degree\_Map[max.degree +$
    $range]\;to\;Degree\_Map[min.degree - range]\}$
    $Add\;(querysetid \rightarrow SampleSpace)\;map\;to\;SampleSpace\_Map$
end
```