Algorithm 1 Finding maximal disks following a parallel approach.

Input: Set of points T, maximum distance ϵ and minimum size μ
Output: Set of maximal disks M

find the set of pairs of points P in T which are ϵ distance each other
$C \leftarrow \emptyset$
for each p_i in P **do**
  compute disks c_i^1 and c_i^2 of p_i using ϵ
  add c_i^1 and c_i^2 to C
end for
$D \leftarrow \emptyset$
for each c_i in C **do**
  find the set of points ρ_i which lie ϵ distance around c_i
  if $|\rho_i| \geq \mu$ **then**
    compute centroid ς_i of the MBR of ρ_i
    set $d_i.center$ as ς_i
    set $d_i.points$ as ρ_i
    if d_i not in D **then**
      add d_i to D {Prunning duplicate candidates...}
    end if
  end if
end for
build an R-Tree $disksRT$ using centers in D
$E \leftarrow \emptyset$
for each MBR in $disksRT$ **do**
  expand MBR to create an expanded MBR ε_i using a buffer of ϵ distance
  add ε_i to E
end for
for each d_i in D **do**
  for each ε_j in E **do**
    if $d_i.center \cap \varepsilon_j$ **then**
      add d_i to ε_j
    end if
  end for
end for
$M \leftarrow \emptyset$
for each ε_i in E **do**
  $\chi \leftarrow \emptyset$
  for each d_i in ε_i **do**
    add $d_i.points$ to χ
  end for
  find the set of maximal patterns F in χ
  for each f_i in F **do**
    compute centroid ς_i of the items in f_i
    if ς_i is not in the expansion area of ε_i **then**
      set $m_i.center$ as ς_i
      set $m_i.points$ as f_i
      add m_i to M
    end if
  end for
end for
return M