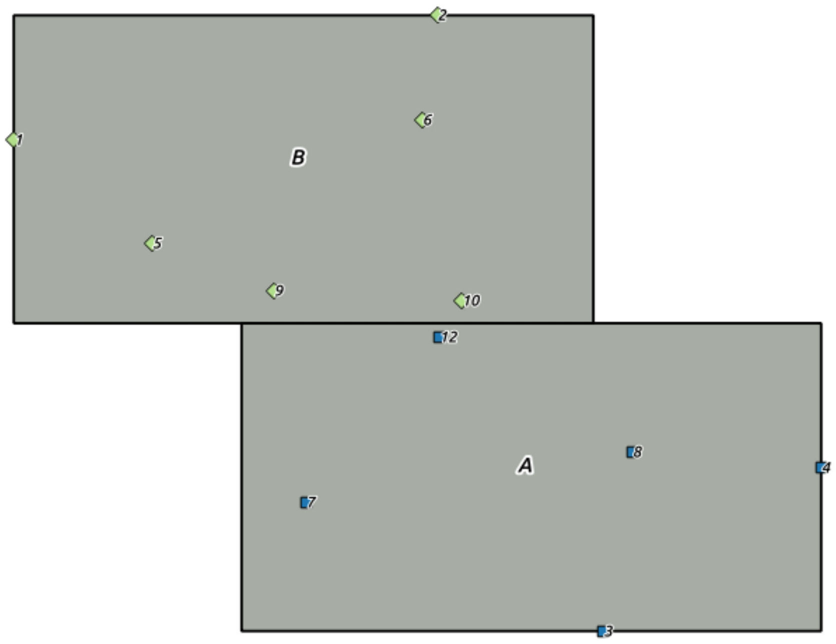

Figure 1: An inital R-Tree over a set of candidates disks represented by their centers.

Proof Sketch

It is assumed that the stages of pairs finding, disks generation and '*less-than-μ*' disk filtering have already been done. We start with a set of candidate disks which are represented by their centers and the set of original points which are enclosed by them. It is expected that they are organized in an R-Tree index as it is shown in figure 1. The following procedure aims to filter out candidate disks when its set of points is a subset of another disk's set of points. Just disks with a superset of points are kept and they are known as maximal disks.

There are two problems during the filtering and report of maximal disks following a parallel approach. First, it can be disks with the same set of points which are reported as maximals in different partitions. We should report only one of them. Second, it can be a disk with a subset of points which appears as a maximal disk in one partition but there is another disk in another partition with a superset of points that contains that set. We should just report the disk containing the superset. We will refer the first problem as *duplicate reporting* and the second one as *subset reporting*.

Duplicate reporting can be easily avoided by using a deterministic way to represent the center of each disk previous to the partitioning. It is, using an unique center for those disks with the same set of points. For example, changing the center of the disks to the MBR's centroid their points will lead to a unique representation of the disks. Note that the change of the center's location does not affect the logic of the procedure due to the set of points of the disks remains the same.

Subset reporting demands more attention. In order to avoid this problem, we have to evaluate disks which potentially can contain the set of points of other disks. Keep

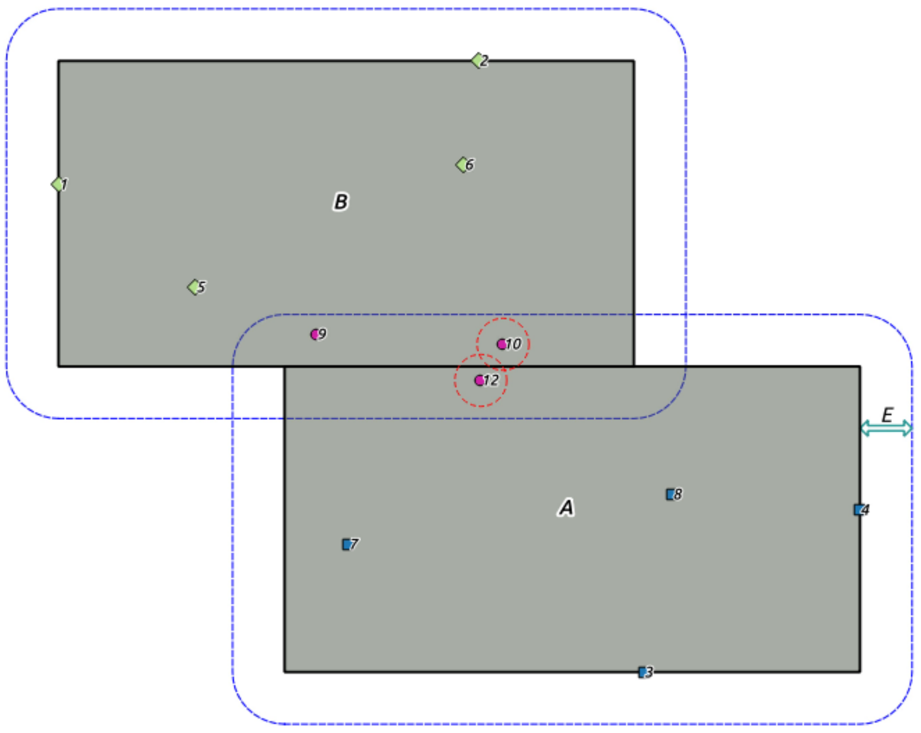

Figure 2: An ϵ expansion over the partitions of the R-Tree. Note that just disks in the expansion areas which intersect each other should be further analyzed.

in mind that just disks which intersect each other should be evaluated as that means that they have points in common. From figure 1, it is clear than disks close to the edge of each partition[1] could intersect other disks in the edge of contiguous partitions. Specifically, disks lying ϵ distance to the edge of their MBRs must be further analyzed but note that those remaining in the internal area are certainly safe.

As we have access to the R-Tree's MBRs, we perform an expansion applying a ϵ buffer around each of them (see figure 2). As the radius of each disk is $\frac{\epsilon}{2}$, it is clear that a disk in any specific partition only can intersect other disks if they lie in the expansion area of its partition.

We re-index the original set of candidate disks using the new expanded MBRs. For those disks which intersect multiple expanded MBRs, a copy of the disk is sent to each of them. If a disk lie in the expansion area of one of the new MBR, it is marked accordingly. A disk can lie in the expansion area of multiple MBRs but it will be outside of the expansion area of its original MBR.

Now we can run a maximal pattern algorithm to find supersets of points and their correspond disks in each partition. Any candidate disk with a subset of points will be obfuscated by its superset (if any) lying in the expansion area avoiding subset reporting. In order to avoid duplicate reporting, caused by copy of disks during expansion, we only

[1]In this context, we refer to partition or MBR indistinctly.

report a maximal disk if it does not lie in the expansion area. As a disk will only be one time outside of an expansion area, we let this disk to be reported by its original partition.

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 $\epsilon$ 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 $\epsilon$
    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 $\rho_i$ which lie $\epsilon$ distance around $c_i$
    if $|\rho_i| \geq \mu$ then
        compute centroid $\varsigma_i$ of the MBR of $\rho_i$
        set $d_i.center$ as $\varsigma_i$
        set $d_i.points$ as $\rho_i$
        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 $\varepsilon_i$ using a buffer of $\epsilon$ distance
    add $\varepsilon_i$ to $E$
end for
for each $d_i$ in $D$ do
    for each $\varepsilon_j$ in $E$ do
        if $d_i.center \cap \varepsilon_j$ then
            add $d_i$ to $\varepsilon_j$
        end if
    end for
end for
$M \leftarrow \emptyset$
for each $\varepsilon_i$ in $E$ do
    $\chi \leftarrow \emptyset$
    for each $d_i$ in $\varepsilon_i$ do
        add $d_i.points$ to $\chi$
    end for
    find the set of maximal patterns $F$ in $\chi$
    for each $f_i$ in $F$ do
        compute centroid $\varsigma_i$ of the items in $f_i$
        if $\varsigma_i$ is not in the expansion area of $\varepsilon_i$ then
            set $m_i.center$ as $\varsigma_i$
            set $m_i.points$ as $f_i$
            add $m_i$ to $M$
        end if
    end for
end for
return M
```