INTRODUCTION

The aim of this section is to explain in detail the data flow and the stages during the finding of flocks following a distributed approach. In general, we can break down the process as follows:

1) Pair finding.
2) Center computation.
3) Disk finding.

The main purpose will be to partition the space from the very beginning and ensure all the required data is accessible to each partition. We expect to exploit locality at most and minimize the impact of any required shuffling.

Pair finding

We start dividing the input point-set into partitions using a quadtree data structure. Each leaf in the quadtree will be used as a partition which divides the point-set into no overlapping areas. We will call the space covered by an individual leaf as a *cell*.

It is important to note that a point close to the border of a cell could be matched with points laying in contiguous cells. So, it is required to replicate the data around each cell to deal with those cases. Each cell will have an expansion zone which width is equal to ε (the distance required between points to be consider a pair). In Fig. 1a the expansion zone is described by the red dotted lines around each cell. Similarly, Fig. 1b shows how the points in the expansion zone are replicated to the involved cells. It is expected that each cell is evaluated by different nodes safely as each one has the required data.
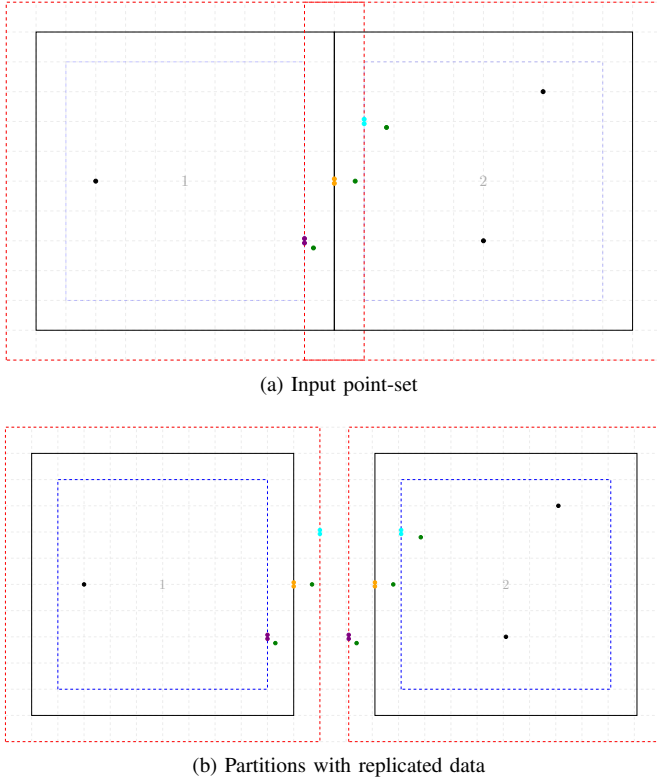

(a) Input point-set

(b) Partitions with replicated data

Fig. 1. Partitioning and replication.

The pair finding takes place at each cell. We perform a partition-based self-distance join using ε as the distance parameter. A simple distance-join will generate duplicate pairs (point A matches point B and point B matches A) so we filter them using the point's id. Pairs were point B's id is greater than point A's id are pruned.

Center computation

From each pair (formed by points A and B), two circles with radius $\frac{\varepsilon}{2}$ will be computed. These two circles have points A and B on their circumference. We will show that if the position of a circle's center lies inside of the current cell they will be safe to operate but those located in the expansion zone or outside of it will require special attention.

Circles with centers in the expansion zone will be repeated in contiguous cells and will increase the computation cost of further stages. In addition, it is possible that pairs of points generates circles with centers outside of the expansion zone. For example, Fig. 2 illustrates the case. Circles a' and b' are generated for pairs of points in cells 1 and 2 respectively, however both are located outside of their expansion zone boundaries. Those circles will be unable to query elements from the original point-set because they were not replicated outside of the expansion zone. We present lemma 1 to show that we can safely remove those kind of circles.

Lemma 1. *Circles with its center lying in the expansion zone or outside of it can be discarded as they will be correctly evaluated by one of the cell in its neighborhood.*

Proof. In order to support our proof we will define some concepts. First, we will divide the area in three zones to clarify our assumptions: we already talked about the expansion zone as the area beyond the border of a cell (between black line and dotted red line in Fig. 2) and a width equal to ε. The border zone is a strip of width equal to ε touching the interior border of a cell. In Fig. 2 it is compromised by the dotted blue and the black lines. The safe zone will be the remaining internal area in the cell which is not covered by the border zone. Second, we will call the contiguous cell with which the current one will replicate a particular circle as the *replicated cell*. In Fig. 2, the cell 2 is the replicated cell of cell 1 and vice versa.

From here, it will be clear that there is a symmetric relation between the circles in a particular zone in the current cell and the circles in its replicated cell. We can certainly said that if circle is located in the border zone of the current cell, it will be located in the expansion zone of its replicated cell. Similarly, any circle with a center lying outside of the expansion zone of the current cell will be located in the safe zone of its replicated cell. Keeping just the circles with centers lying in the current cell (border or safe zone) will be enough to ensure no lost of information (Fig. 3). □

Disk finding

We call *disk* to the circles (and companion center) together to those points which are enclosed by its circumference. After
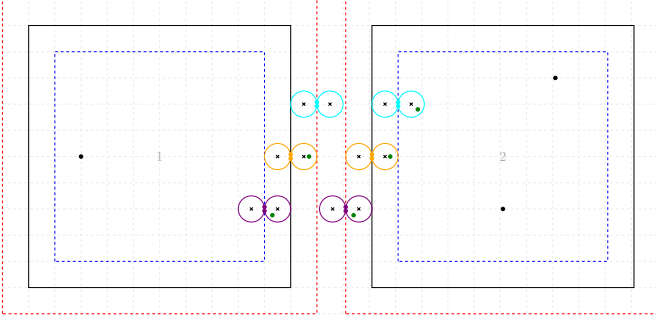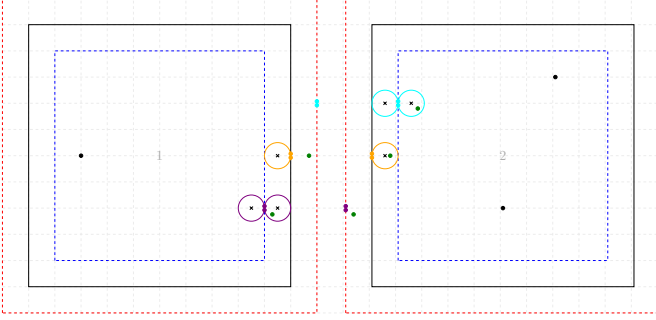
Fig. 2. Center computation.

Fig. 3. Center pruning.

the centers computation, this procedure requires to query the points enclosed by the circles of radius $\frac{\varepsilon}{2}$ recently discovered. We will keep advantage on the current partitioning strategy to perform a partition-based distance join between the set of circles and points. It is valid to note that no all the points from the original input are needed for this operation. Lemma 2 shows that just the points resulting from the pair finding stage are required. It is important because in order to keep the same partitioning just the points involved during the pair finding remain accessible in the current cell.

Lemma 2. *Points which are not part of the pair-set are not involved in the disk finding procedure.*

Proof. Let have a point p_c lying inside of a particular circle. So p_c must be located at most $\frac{\varepsilon}{2}$ distance from the center of that circle and, at most, ε distance to the points p_a and p_b which generated that center. That means that the p_c itself must form pairs between itself and points p_a and p_b and, therefore, be part of the pair-set. On the other hand, if a point lies more than ε distance of any other point, it will not be part of the pair-set and none center will be computed in its surrounding, therefore, it cannot be part of any disk. $\square$