Algorithm 1: getNextCellWithEdges algorithm

```
Input: a quadtree $\mathcal{Q}$ and a list of cells $\mathcal{M}$.
function getNextCellWithEdges($\mathcal{Q}$,$\mathcal{M}$):
    $C \leftarrow$ orphan cells in $\mathcal{M}$
    foreach orphanCell in $C$ do
        initialize cellList with orphanCell
        $nextCellWithEdges \leftarrow nil$
        $referenceCorner \leftarrow nil$
        $done \leftarrow false$
        while $\neg done$ do
            $c \leftarrow$ last cell in cellList
            $cells, corner \leftarrow$ getCellsAtCorner($\mathcal{Q}, c$)
            foreach cell in cells do
                $nedges \leftarrow$ get edge count of cell in $\mathcal{M}$
                if $nedges > 0$ then
                    $nextCellWithEdges \leftarrow cell$
                    $referenceCorner \leftarrow corner$
                    $done \leftarrow true$
                else
                    add cell to cellList
                end
            end
        end
        foreach cell in cellList do
            output(cell,nextCellWithEdges,
             referenceCorner)
            remove cell from $C$
        end
    end
end
```