Technische Universität München
Lehrstuhl für Informatik V
Univ.-Prof. Dr. M. Bader
M.Sc. D. Jarema, M.Sc. K. Röhner

SS 2015
Worksheet 9
17.6.2015

Algorithms of Scientific Computing

Refinement Trees and Parallelization with Space-Filling Curves

Exercise 1: Hilbert-Order Encoding of a Quadtree

Given a domain with an obstacle as shown in Figure 1, we want to run a certain simulation on this domain. In order to obtain accurate results it is enough to have a fine mesh only near the boundary of the obstacle. Everywhere else the grid can be as coarse as possible.

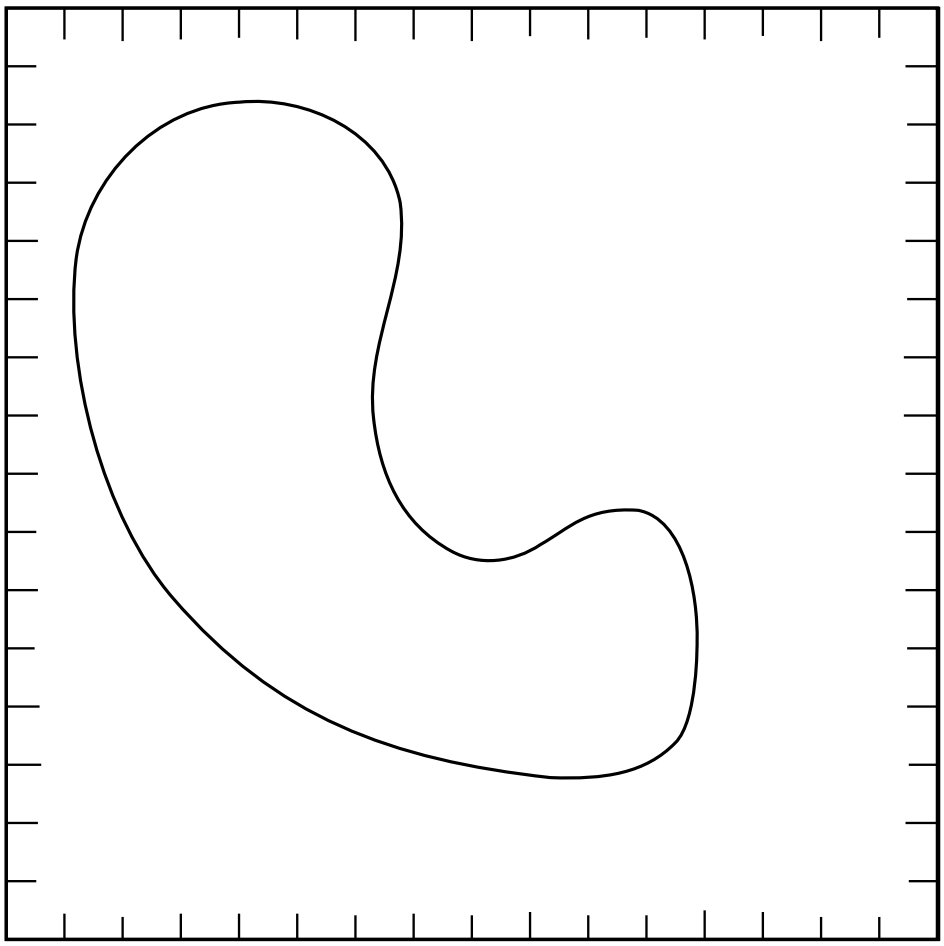


Figure 1: Obstacle in a square domain.

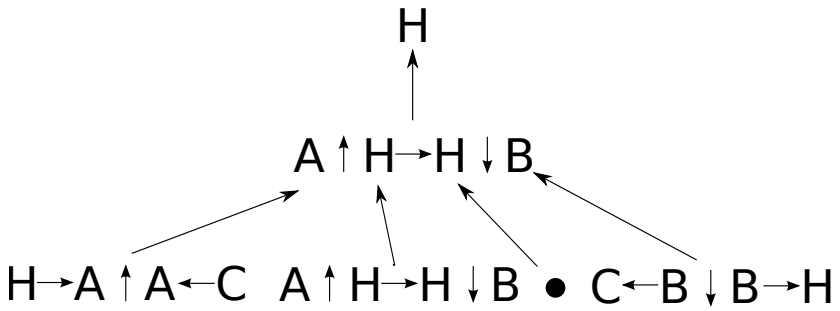
a) Use the following strategy to refine the given square domain: If a square cell contains a boundary of the obstacle, subdivide this square into four equal subsquares. Perform at most four subdivisions, in other words, the maximum depth of the corresponding quadtree is four. Sketch the resulting grid on Figure 1.

b) Draw a corresponding quadtree of the refined domain.

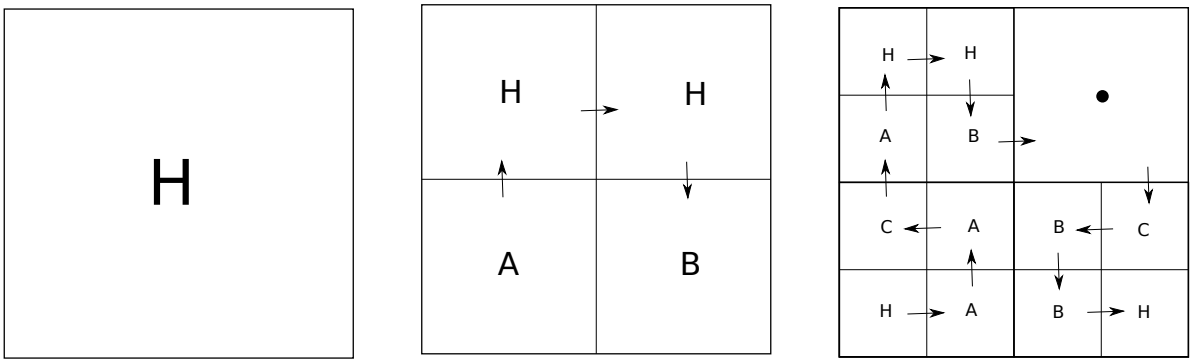
c) Sequentialize the spacetree grid by using a Hilbert space-filling curve. Start with the non-terminal H. Use the following grammar rules for Hilbert orders on quadtrees:

$$H \longleftarrow (A \uparrow H \rightarrow H \downarrow B) \qquad H \longleftarrow \bullet$$
$$A \longleftarrow (H \rightarrow A \uparrow A \leftarrow C) \qquad A \longleftarrow \bullet$$
$$B \longleftarrow (C \leftarrow B \downarrow B \rightarrow H) \qquad B \longleftarrow \bullet$$
$$C \longleftarrow (B \downarrow C \leftarrow C \uparrow A) \qquad C \longleftarrow \bullet$$

Hint: You might want to use the hierarchical tree representation of the refined domain or iteratively map the curve directly into the domain, as shown in Figure 2 a) and b) for the two levels of refinement. Draw the resulting Hilbert curve on Figure 1.



(a) Refinement tree for an adaptive Hilbert curve.



(b) Direct mapping of a Hilbert curve to an adaptive grid.

Figure 2: Two first levels of the derivation of the traversal order for the Hilbert grammar.

d) Implement a vertex-labeling algorithm to plot the iterations of the adaptive Hilbert curve. The algorithm must accept a stream of numbers, which is equivalent to the bitstream encoding of the Hilbert curve, but the single refinement bit is replaced by the number of

nodes in the entire subtree. A skeleton of the program can be found in `Worksheet_9-Template.ipynb`. Check the results of the exercise by encoding the Hilbert curve that traverses through the domain with the obstacle by a stream of numbers and plotting the curve with your program.

Exercise 2: Parallelization with Space-Filling Curves

We want to traverse an adaptive grid in the Hilbert curve order in a parallel setting. Consider the following two situations and complete the tasks:

a) Assume that we use a shared-memory environment and several threads can process the unknowns on an adaptive grid. We can keep only one copy of the refinement structure for all threads in a vertex labeled stream of numbers format, but we would like the threads to be able to traverse only a certain part of the quadtree and skip all other parts. Extend the program from the previous exercise by adding a method that allows to traverse only a part of the tree specified by given start and end node indices.

b) In a distributed-memory setting usually only a separate local grid is stored by each process. In this case, we can keep the grid traversal simple by retaining the global grid representation, but coarsening the non-local partitions as far as possible. Partition the domain represented in Figure 3 for four processes by dividing the Hilbert space-filling curve into equal parts. Sketch all four grid representations for each process with corresponding space-filling curves and local quadtree structures.

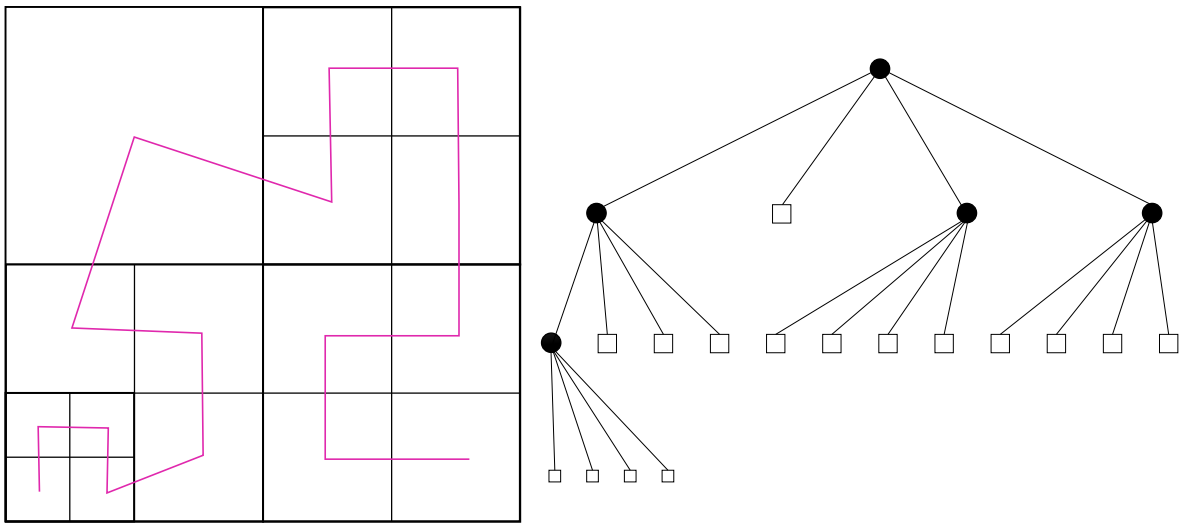


Figure 3: Domain with adaptive grid and corresponding space-filling curve.