Geoinformatica paper extension

University of California, Riverside

February 9, 2024

So far...

- Explore a couple of implementation of k-d tree as new data structure during partitioning:
  - Brown's balanced K-d tree [`https://arxiv.org/abs/1410.5420`]
  - Sedona's KDB tree [https://sedona.apache.org/]

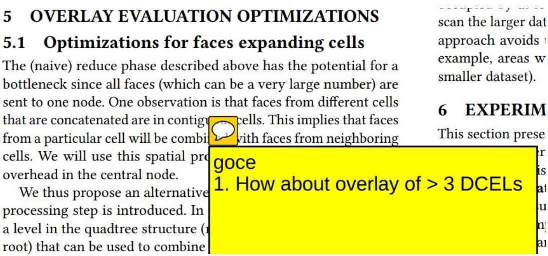
So far...

- We had a serious concern about data availability during k-d construction to complete the optimization...
    - At that stage we deal with a sample of the data.
    - Not certain if we could even detect possible unbalanced cells.
- but...

So far...

- We had seen that given a representative sample detecting unbalanced cells is possible (maybe missing a few).
- During kd-tree creation we could compute a set of initial intervals.
- During edge partitioning we could update those interval accordingly (in progress).
- So, we could expect to save time for sorting and interval finding we perform later in the original approach.

Some questions about other comments...



5 OVERLAY EVALUATION OPTIMIZATIONS

5.1 Optimizations for faces expanding cells

The (naive) reduce phase described above has the potential for a bottleneck since all faces (which can be a very large number) are sent to one node. One observation is that faces from different cells that are concatenated are in contig cells. This implies that faces from a particular cell will be combi ith faces from neighboring cells. We will use this spatial pr overhead in the central node.

We thus propose an alternative processing step is introduced. In a level in the quadtree structure (root) that can be used to combine

scan the larger dat approach avoids example, areas w smaller dataset).

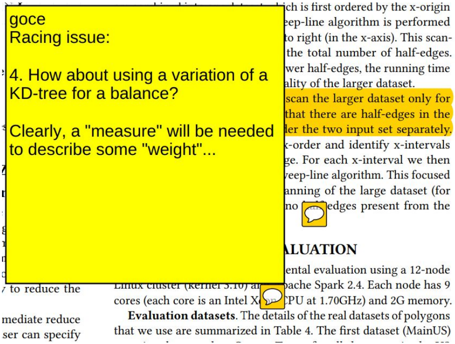
6 EXPERIM

This section prese

goce
1. How about overlay of > 3 DCELs

- Is it more related to compute more than two layers at the same time or something like cascading?

Some questions about other comments...



goce
Racing issue:

4. How about using a variation of a KD-tree for a balance?

Clearly, a "measure" will be needed to describe some "weight"...

ch is first ordered by the x-origin
eep-line algorithm is performed
to right (in the x-axis). This scan-
the total number of half-edges.
wer half-edges, the running time
ality of the larger dataset.
scan the larger dataset only for
that there are half-edges in the
der the two input set separately.
x-order and identify x-intervals
ge. For each x-interval we then
veep-line algorithm. This focused
anning of the large dataset (for
no edges present from the

LUATION

ental evaluation using a 12-node
Linux cluster (kernel 3.10) and pache Spark 2.4. Each node has 9
cores (each core is an Intel X CPU at 1.70GHz) and 2G memory.

Evaluation datasets. The details of the real datasets of polygons that we use are summarized in Table 4. The first dataset (MainUS)

to reduce the

mediate reduce
ser can specify

- With a balanced data structure would we expect a more fair evaluation when we test different levels of the tree?