Progressive transmission of variable scale vector data over the web

More details on demand

Martijn Meijers, Theo Tijssen and Peter van Oosterom

TU Delft

OTB, GIS Technology group

Introduction

- Part of PhD project
- Hierarchical model for 2D vector data
- 2D vector data and level of detail in one model
- Model forces adoption of area partition
- Investigation on how good the hierarchical structures are suited for progressive transmmission
- Work in progress, not completely finished

Context – MobiMaps

- Test hierarchical model in context of MobiMaps project
- Dutch programme for Geo-Innovation (RGI)
- Partners conducted usability studies – TNO and ITC
- Technological aspects our concern, together with 1Spatial, UK and ESRI, NL
- Structures suited for progressive transmission, giving a better user experience?

Progressive Transmission – Definition

- Progressive transmission – More details on demand, in a client-server setup
- Send less detailed overview first, more details without completely resending all the data again, as time progresses

Progressive Transmission – Issue

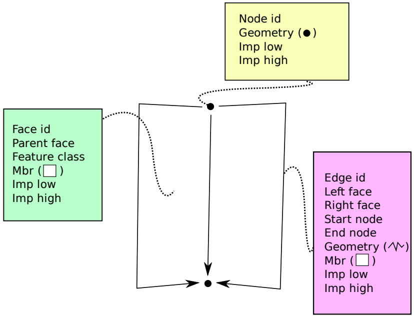
- Research on progressive transmission of 2D vector data focusses mostly on adding vertices to geometry (1 vector map suitable for certain scale, send stream of additions)
- Not so much focus on re-use of already sent data and how to request new data (caching)

Progressive Transmission – Central question

- How to get data out of the hierarchical model, efficiently and in a progressive manner (suitable for re-use at client side)?

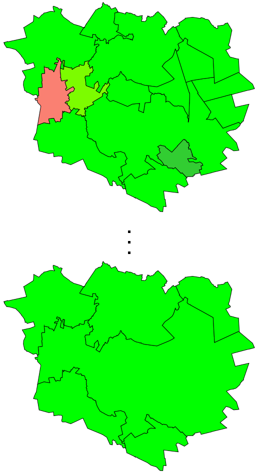
Context – Hierarchical model

- Store information on area objects, e.g. land use
- Use topology, no explicit geometry for area objects, but:
  1. Edge: line that knows which areas are neighbouring, plus bounding box
  2. Face: Area object represented by a bounding box
- Start is a 2D area partition
- How to obtain hierarchical information?



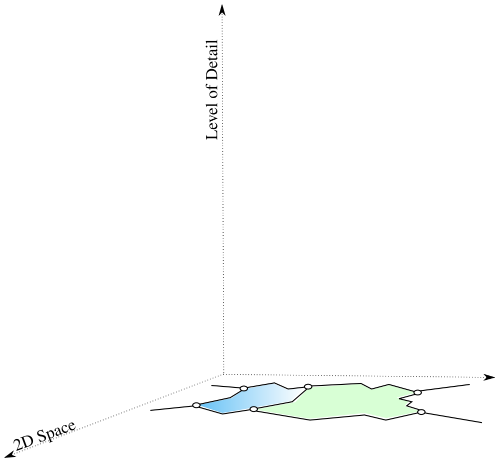
Context – Hierarchical model

- Process of merging objects, until only one object is left
- This process generates hierarchy
- 2 questions:
  1. Which object as candidate for merge?
  2. Which object to merge candidate object to?



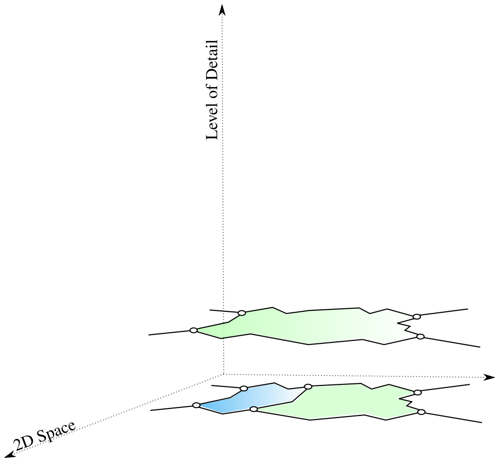
Context – Hierarchical model

- Process creates lifespan information for each primitives (edges *and* faces), in the 'Level of Detail'-dimension
- Map varies with scale, going up in cube (with 'Space' and 'Level of Detail'-dimensions)
- Model is termed tGAP - topological Generalized Area Partitioning



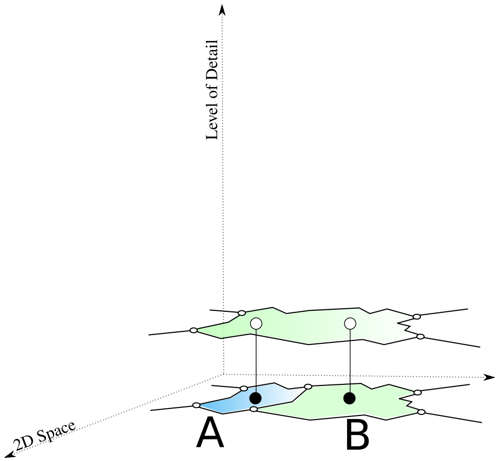
Context – Hierarchical model

- Process creates lifespan information for each primitives (edges *and* faces), in the 'Level of Detail'-dimension
- Map varies with scale, going up in cube (with 'Space' and 'Level of Detail'-dimensions)
- Model is termed tGAP - topological Generalized Area Partitioning



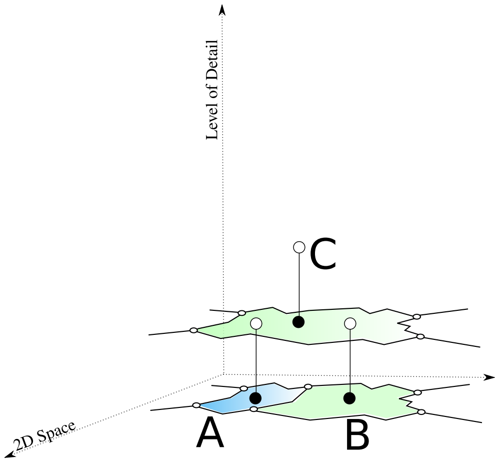
Context – Hierarchical model

- Process creates lifespan information for each primitives (edges *and* faces), in the 'Level of Detail'-dimension
- Map varies with scale, going up in cube (with 'Space' and 'Level of Detail'-dimensions)
- Model is termed tGAP - topological Generalized Area Partitioning



Context – Hierarchical model

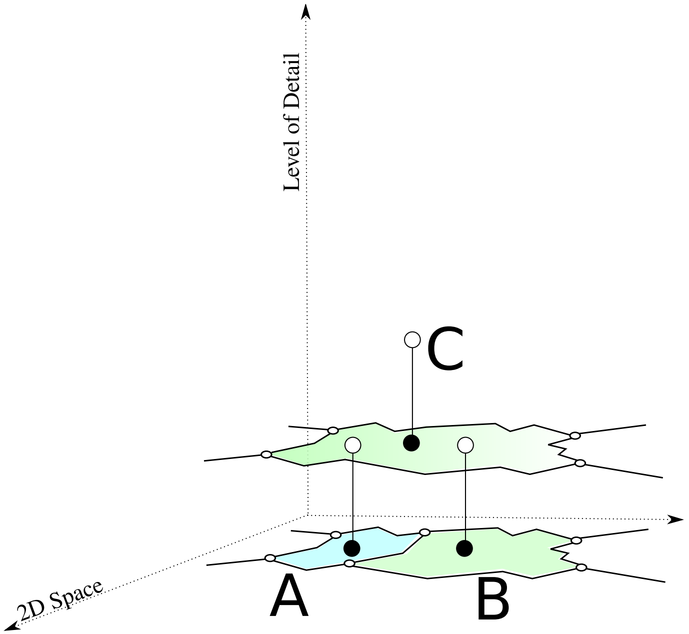
- Process creates lifespan information for each primitives (edges *and* faces), in the 'Level of Detail'-dimension
- Map varies with scale, going up in cube (with 'Space' and 'Level of Detail'-dimensions)
- Model is termed tGAP - topological Generalized Area Partitioning



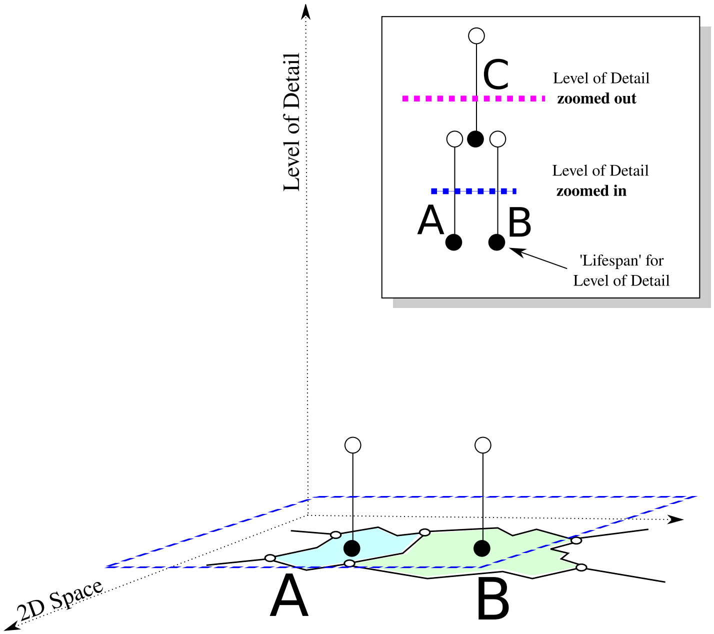
Initial client

- Thin client – Stateless
- Independent requests
- Sends bbox, gets back edges and faces
- Mapping bbox to Level of Detail: done by server
  - Large bbox – low level of detail – zoomed out
  - Small bbox – high level of detail – zoomed in
- Client ‘knows’ nothing about Level of Detail of returned primitives

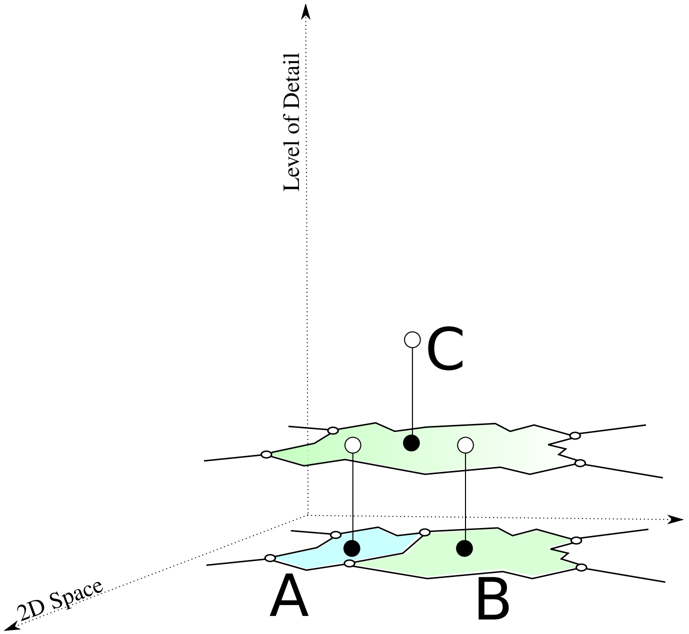
Initial client



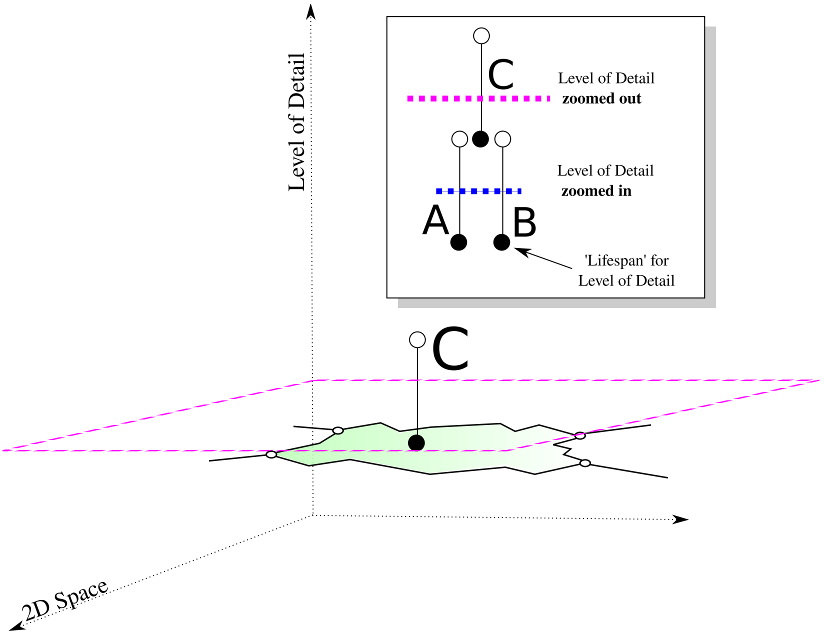
Initial client



Initial client



Initial client



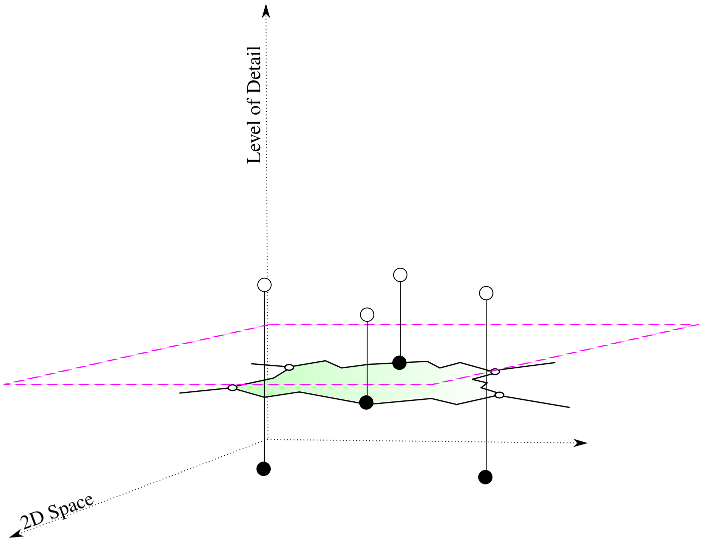
Initial client

- Selection of primitives based on their 3D bounding box (2D spatial extent, 1D Level of Detail) intersecting with 2D viewport at certain Level of Detail
- On retrieval complete:
    1. Clip edges
    2. Reconstruct polygon geometry
    3. Draw polygons

Progressive client

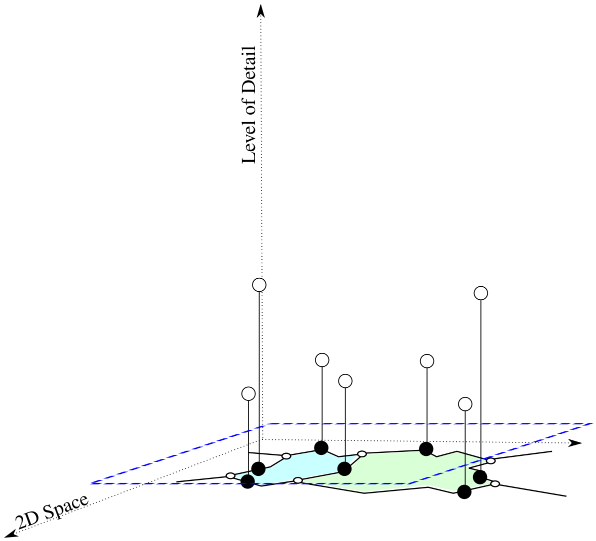
- Benefit from hierarchical data organization and topology
- Client should keep more state – Fat client, with some caching abilities
- Two variants for data retrieval:
  1. Using set difference for the previous and current bounding box
  2. Intersection of primitives with 3D frustum, sending operations as a stream, in order

Progressive client – Variant 1



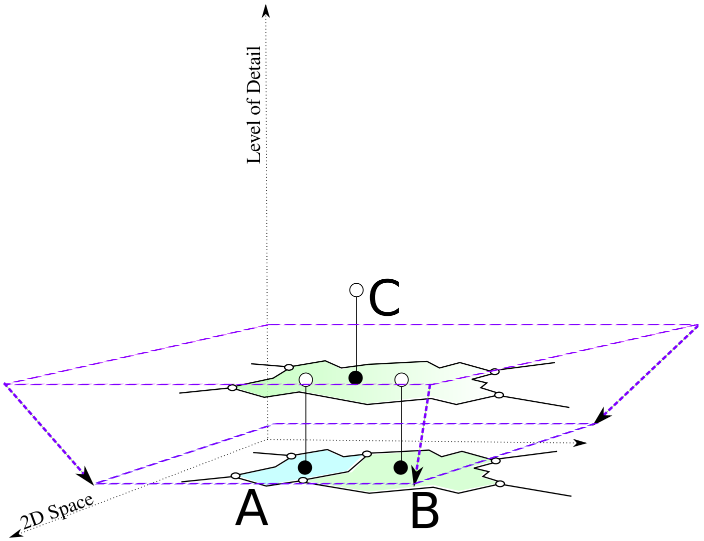
Variant 1 – Set difference

Progressive client – Variant 1



Variant 1 – Set difference

Progressive client – Variant 2

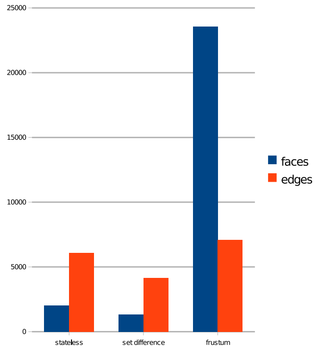


Variant 2 – 3D frustum

Following a users path

- What's the consequence of each alternative for the number of primitives (edges, faces) to be sent?
- Follow user its path – different actions: zooming in and out, panning
- Sum number of primitives retrieved

Following a users path

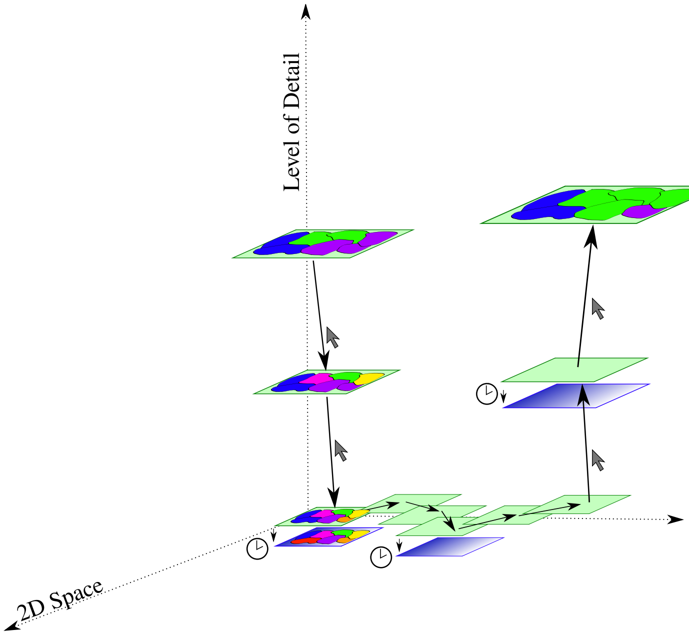


- Interesting observation: 3D intersection can 'pull in' objects being merged (zoom out) or split (zoom in) from an area where user is not zooming in, due to overlapping bounding box
- Similar problems with other indexing structures, e.g. R-Tree is not selective with large linear features

Prototype

- Implemented 2D intersection variant, with using set difference
- HTTP requests and web server
- Client handles requests to server
- After user waits a while, new request is made automatically (same area, more detail)

Prototype



Results

- Can get data out progressively, having different alternatives
- Possbile to do progressive transmission (with data re-use)
- Not including geometry refinement (yet)
- On average: faster response as less data is to be transferred for initial overview

Thank you

Delft University of Technology
OTB, GIS Technology Group

Martijn Meijers
b.m.meijers@tudelft.nl
tel. (+31) 15 27 85 642