Declarative Cartography: In-Database Map Generalization of Spatial Datasets

P.K. Kefaloukos, M.V. Salles, M. Zachariasen

Abstract
The Cartographic Visualization Language (CVL) is a declarative domain specific language that allows users to express selection of records for map visualizations. Visual constraints that must hold for the map can be declaratively expressed in CVL, e.g. the minimum proximity between points on the screen. Furthermore, CVL allow users to assign weights to records to express their salience. We wrote a compiler to translate CVL to SQL for computing the selection using standard relational database technology. The underlying problem that is solved is a set cover problem, and the compiler implements two SQL based algorithms for the multiset cover problem.

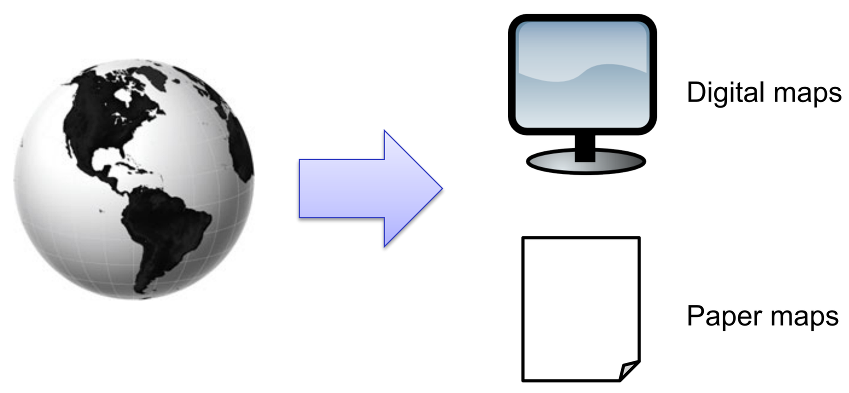
Motivation

Fast and programmable creation of legible, zoomable maps

Massive amounts of raw data — *Bottleneck (human)* — *Legible, zoomable map visualizations*

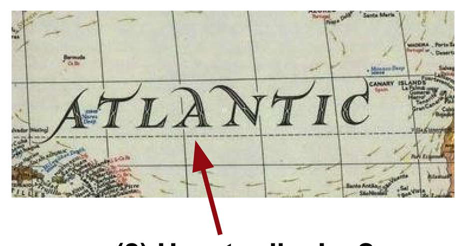
Online data is in explosive and continuous growth, and this is also true for spatial data. There is a high demand for visualizing spatial data on *"zoomable" maps* with many use cases in *social media, journalism, science* and so on. However, selecting the right content for maps at different zoom levels is a fundamentally hard problem, when a legible and useful map is desired. Displaying unfiltered raw data results in clutter, and the task of selecting data based on cartographic principles is time-consuming if done manually. For large datasets and tight deadlines (e.g. in journalism) there is a need for automatic and programmable methods, that allow non-technical experts to express high-level goals of a map visualization. Methods for map content selection must be efficiently implemented using a scalable and programmable data system, e.g. a relational database.

Challenge: generalization



Key decisions:

(1) What to display?



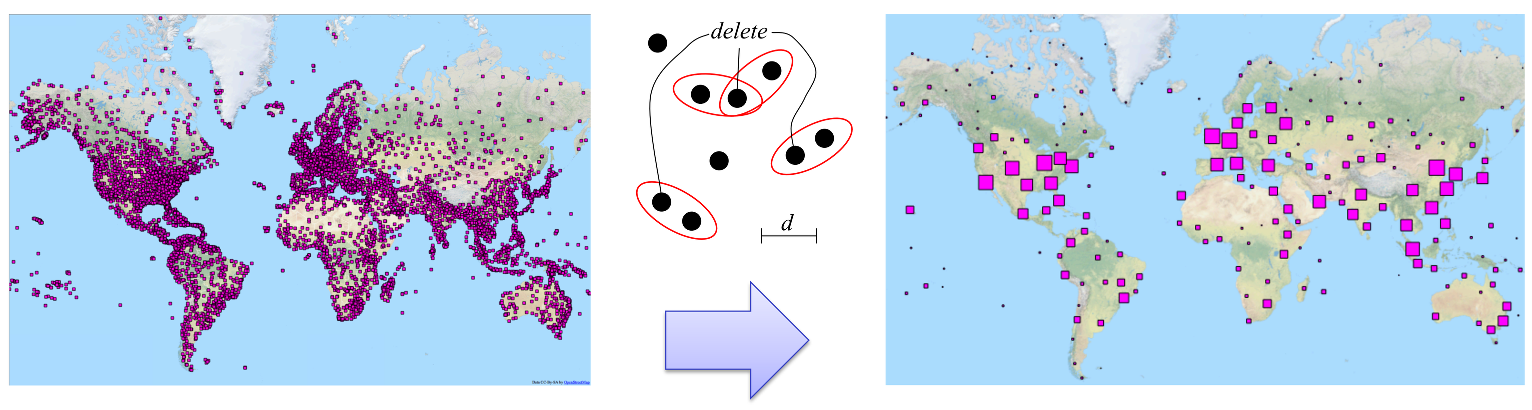
(2) How to display?

Big world VS small screen

Generalization is a subtask of cartography. The goal of generalization is to adapt spatial data for display on scaled down media (e.g. paper maps and computer screens), such that legibility, representation and usefulness of the map is achieved.

Constraints & weights

Before — **After**



Input: raw airports

7411 airport locations in the world from the OpenFlights airports dataset. Each airport is geometrically represented as a point.

Displaying all airports on a world map results in a high degree of clutter.

Output: generalized airports

- **Constraint 1 (proximity)**: minimum proximity between points is 10 px
- **Constraint 2 (visibility)**: maximum number of records shown per grid cell is 16 records
- **Weight**: Prioritize airport records according to number of departures

DSL for generalization

```
GENERALIZE
  airports TO airports2
WITH ID airport_id
WITH GEOMETRY wkb_geometry
AT 18 ZOOM LEVELS
WEIGH BY num_departures
SUBJECT TO
  proximity 10  -- pixels
  visibility 16 -- records
```

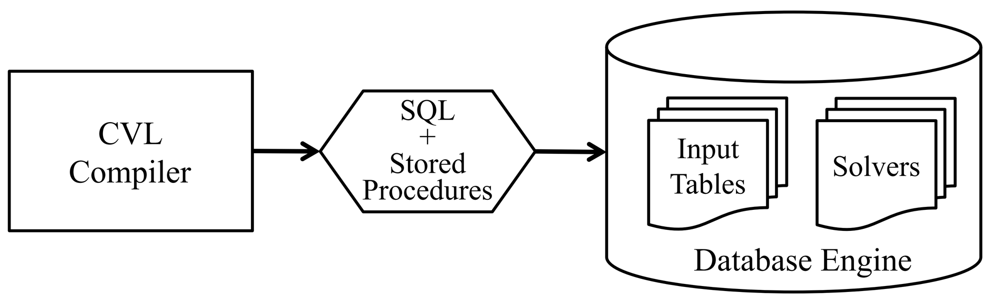
```
CREATE CONSTRAINT proximity
AS NOT EXISTS (
  SELECT cid, rid
  FROM …
)

RESOLVE cid IF DELETE (
  1
)
```

Cartographic Visualization Language (CVL)

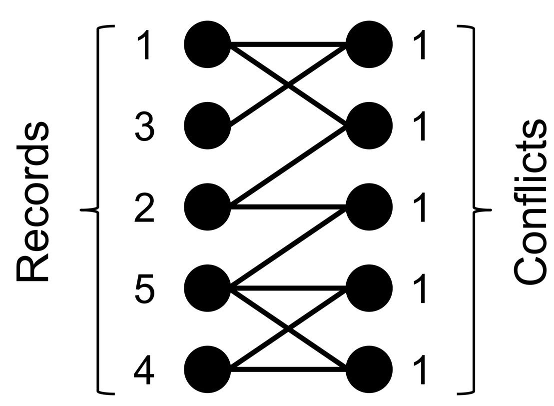
Use CVL to express constraints and weights for zoomable map visualizations . Language is designed to be useable by non-technical experts and to express overall goals for a map (e.g. "no clutter") instead of micro-managing selectiion of individual records.

Compilation to SQL



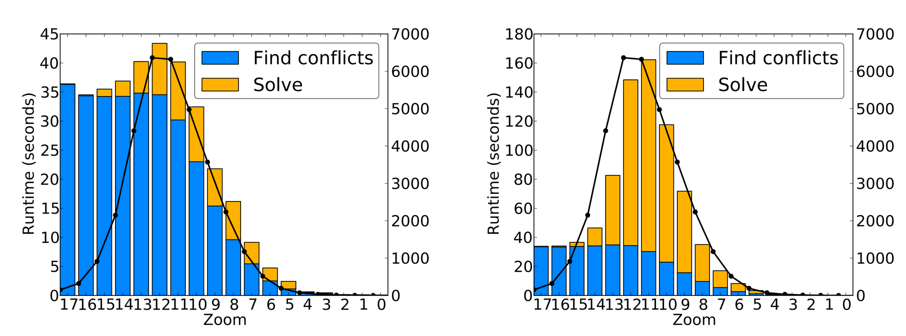
Tasks expressed in SQL

- Computing record weights
- Detecting violated constraints (conflicts)
- Formulating instance of set multicover problem (SMCP)
- Solving instance of SMCP using SGA or LPGA algorithm

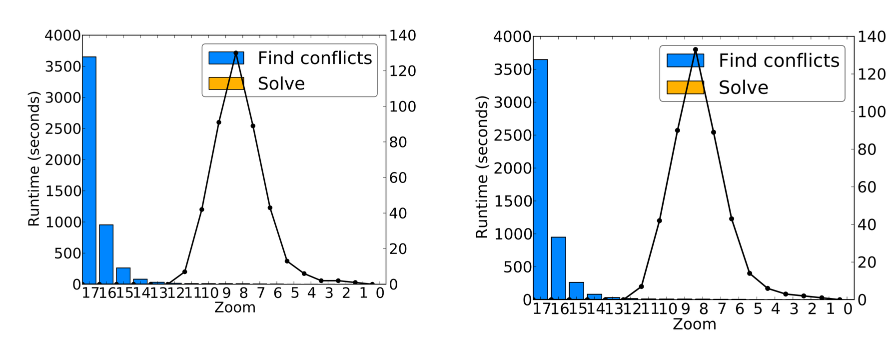


Results

Point data (500K records/500K points)



Complex data (4K records/2M points)



About charts

Comparison of running times for point and complex data. Time breakdown by task:

- (blue) find conflicts
- (yellow) solve conflicts

Black line indicates how many conflicts were found on given zoom level.