Detecting Duplicates in Geoinformatics: from Intervals and Fuzzy Numbers to General Multi-D Uncertainty

Scott A. Starks, Luc Longpré
Roberto Araiza, Vladik Kreinovich
University of Texas at El Paso
El Paso, Texas 79968, USA
sstarks@utep.edu, vladik@utep.edu

Hung T. Nguyen
New Mexico State University
Las Cruces, New Mexico 88003, USA
1. Outline

- **Fact:** geospatial databases often contain duplicate records.
- **Problem:** how to detect and delete duplicates.
- **Test case:** measurements of anomalies in the Earth’s gravity field that we have compiled.
- **Previously analyzed case:** closeness of two points \((x_1, y_1)\) and \((x_2, y_2)\) is described as closeness of both coordinates.
- **What was known:** \(O(n \cdot \log(n))\) duplication deletion algorithm for this case.
- **New result:** we extend this algorithm to the case when closeness is described by an arbitrary metric.
2. **Geospatial Databases: General Description**

- **Fact:** researchers and practitioners have collected a large amount of geospatial data.

- **Examples:** at different geographical points \((x, y)\), geophysicists measure values \(d\) of:
  - the gravity fields,
  - the magnetic fields,
  - elevation,
  - reflectivity of electromagnetic energy for a broad range of wavelengths (visible, infrared, and radar).

- **How this data is stored:** corresponding records \((x_i, y_i, d_i)\) are stored in a large geospatial database.

- **How this data is used:** based on these measurements, geophysicists generate maps and images and derive geophysical models that fit these measurements.
3. Gravity Measurements: Case Study

- **Typical geophysical data** (e.g., remote sending images):
  - mainly reflect the conditions of the Earth’s *surface*;
  - cover a reasonably *local* area.

- **Gravity measurements**:
  - gravitation comes from the whole Earth, including *deep* zones;
  - gravity measurements cover *broad* areas.

- **Conclusion**: gravity measurements are one of the most important sources of information about subsurface structure and physical conditions.
4. Duplicates: Where They Come From

• **Fact:** the existing geospatial databases contain many duplicate points.

• **Reason:**
  - databases are rarely formed completely “from scratch”;
  - they are usually built by combining measurements from previous databases;
  - some measurements are represented in several of the combined databases.

• **Conclusion:** after combining databases, we get duplicate records.
5. Why duplicates Are a Problem

- *Main reason:* duplicate values can corrupt the results of statistical data processing and analysis.

- *Example:*
  - when we see several measurement results confirming each other,
  - we may get an erroneous impression that this measurement result is more reliable than it actually is.

- *Conclusion:* detecting and eliminating duplicates is an important part of assuring and improving the quality of geospatial data.
6. Duplicates and Related Uncertainty

- **Ideal case:** measurement results are simply stored in their original form.

- **In this case:** duplicates are identical records, easy to detect and to delete.

- **In reality:** databases use different formats and units.

- **Example:** the latitude can be stored in degrees (as 32.1345) or in degrees, minutes, and seconds.

- **As a result:** when a record \((x_i, y_i, d_i)\) is placed in a database, it is transformed into this database’s format.

- **Fact:** transformations are approximate.

- **Result:** records representing the same measurement in different formats get transformed into values which correspond to close but not identical points”

\[(x_i, y_i) \neq (x_j, y_j).\]
7. Duplicates Corresponding to Interval Uncertainty

Geophysicists produce a threshold \( \varepsilon > 0 \) such that \( \varepsilon \)-closed points \((x_i, y_i)\) and \((x_j, y_j)\) are duplicates.

In other words, if a new point \((x_j, y_j)\) is within a 2D interval \([x_i - \varepsilon, x_i + \varepsilon] \times [y_i - \varepsilon, y_i + \varepsilon]\) centered at one of the existing points \((x_i, y_i)\), then this new point is a duplicate:
8. Duplicates Are Not Easy to Detect and Delete

- **Problem:** detect and delete duplicates.
- **How this is done now:** “by hand”, by a professional geophysicist looking at the raw measurement results (and at the preliminary results of processing these raw data).
- **Limitations:** time-consuming.
- **Natural idea:** use a computer to compare every record with every other record.
- **Analysis:** this idea requires $\frac{n(n-1)}{2} \sim \frac{n^2}{2}$ comparisons.
- **Limitation:** this is impossible for large databases, with $n \approx 10^6$ records.
- **Conclusion:** faster algorithms are needed.
9. From Interval to Fuzzy Uncertainty

- **Typical situation:** geophysicists provide several possible threshold values $\varepsilon_1 < \varepsilon_2 < \ldots < \varepsilon_m$ that correspond to decreasing levels of their certainty:
  - if two measurements are $\varepsilon_1$-close, we are 100% certain that they are duplicates;
  - if two measurements are $\varepsilon_2$-close, then with some degree of certainty, we can claim them to be duplicates, etc.

- **Objectives:**
  - eliminate *certain* duplicates, and
  - mark *possible* duplicates (about which we are not 100% certain) with the corresponding degree of certainty.

- **Reduction to interval case:** we need to solve the interval problem for several different values of $\varepsilon_i$. 
10. What We Did in Our Previous Work

• Previously analyzed case: $\varepsilon$-closeness of two points $(x_i, y_i)$ and $(x_j, y_j)$ is described as $\varepsilon$-closeness of both coordinates.

• Geometric reformulation: the set of all points which are $\varepsilon$-close to a given point is a box.

• Result of the analysis: there exists efficient $O(n \cdot \log(n))$ algorithms for detecting and deleting outliers.

• More general situation: when $\varepsilon$-closeness is described by an arbitrary metric: e.g., Euclidean metric

$$d((x_i, y_i), (x_j, y_j)) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

or $l^p$-metric

$$d((x_i, y_i), (x_j, y_j)) = \sqrt[p]{|x_i - x_j|^p + |y_i - y_j|^p}.$$ 

• What we do now: extend the existing algorithms to this more general metric situation.
11. Formalization of the Problem

- By a *metric*, we mean a triple \((S, c, C)\), where
  - \(S \subseteq \mathbb{R}^m\) is a convex set that contains 0, and
  - \(c > 0\) and \(C > 0\) are numbers such that:
    - \(S\) is *symmetric* (i.e., for every point \(r\), we have \(r \in S\) if and only if \(-r \in S\)) and
    - \([-c, c] \times \ldots \times [-c, c] \subseteq S \subseteq [-C, C] \times \ldots \times [-C, C].\)

- We say that points \(r\) and \(r'\) are \(\varepsilon\)-close if \(\frac{r - r'}{\varepsilon} \in S\).

- *Comment*: the property of \(c\) means that \(S\) contains all points close to 0.

- *Example of interval uncertainty*: \(S\) is a cube:
  \[S = [-1, 1] \times \ldots \times [-1, 1].\]
12. New Algorithm: General Description

- **Stage 1:** for each record, compute the indices
  \[ p_i = \lfloor x_i / (C \cdot \varepsilon) \rfloor, \ldots, q_i = \lfloor y_i / (C \cdot \varepsilon) \rfloor. \]

- **Stage 2:**
  - Sort the records in lexicographic order \( \leq \) by their index vector \( \vec{p}_i = (p_i, \ldots, q_i) \).
  - If several records have the same index vector, check whether some are duplicates of one another, and delete the duplicates.
  - As a result, we get an index-lexicographically ordered list of records: \( r_{(1)} \leq \ldots \leq r_{(n_0)} \) \( (n_0 \leq n) \).

- **Stage 3:** For \( i \) from 1 to \( n_0 \), we compare the record \( r_{(i)} \) with all its \( \leq \)-following “immediate neighbors” \( r_{(j)} \):
  \[ |p_{(i)} - p_{(j)}| \leq 1, \ldots, |q_{(i)} - q_{(j)}| \leq 1. \]
  If \( r_{(j)} \) is a duplicate to \( r_{(i)} \), we delete \( r_{(j)} \).
13. Possibility of Parallelization

- **Problem**: for large \( n \), an \( O(n \cdot \log(n)) \) algorithm still requires too much time.

- **Possible solution**: if we have several processors that can work in parallel, we can speed up computations.

- **Example**: we have \( n^2/2 \) processors.

- **Simple result**: by assigning each pair \((r_i, r_j)\) to a different processor, we can detect and delete all duplicates in one step.

- **Other parallelization results**:
  - If we have at least \( n \) processors, then we can delete duplicates in time \( O(\log(n)) \).
  - If we have \( p < n \) processors, then we can delete duplicates in time \( O \left( \left( \frac{n}{p} + 1 \right) \cdot \log(n) \right) \).
14. Acknowledgments

This work was supported in part:

• by NSF grant EAR-0225670,

• by Texas Department of Transportation grant No. 0-5453, and

• by the Japan Advanced Institute of Science and Technology (JAIST) International Joint Research Grant 2006-08.

We are very thankful to the anonymous referees for the useful comments.