Finding Haystacks with Needles: Ranked Search for Data Using Geospatial and Temporal Characteristics

V.M. Megler, David Maier

Computer Science Department, Portland State University
{ vmegler@cs.pdx.edu, maier@cs.pdx.edu }

Abstract. The past decade has seen an explosion in the number and types of environmental sensors deployed, many of which provide a continuous stream of observations. Each individual observation consists of one or more sensor measurements, a geographic location, and a time. With billions of historical observations stored in diverse databases and in thousands of datasets, scientists have difficulty finding relevant observations. We present an approach that creates consistent geospatial-temporal metadata from large repositories of diverse data by blending curated and automated extracts. We describe a novel query method over this metadata that returns ranked search results to a query with geospatial and temporal search criteria. Lastly, we present a prototype that demonstrates the utility of these ideas in the context of an ocean and coastalmargin observatory.

Keywords: spatio-temporal queries, querying scientific data, metadata.

1 Introduction

In the past decade, the number and types of deployed environmental sensors have exploded, with each sensor providing a sequence of observations. Each individual observation has one or more sensor measurements and is associated with a geographic location and a time. Almost a decade ago, this explosion was described as “the Data Deluge” [14], and continued exponential growth in data volumes was predicted [19]. For example, an oceanography observatory and research center with which we collaborate (CMOP, http://www.stccmop.org) now has terabytes of observations spanning more than a decade, reported by a changing set of fixed and mobile sensors. This collection of data provides a rich resource for oceanographic research.

Scientists now research ecosystem-scale and global problems. Marine biologists wish to position their samples within a broader physical context; oceanographers look for comparative times or locations similar to (or dissimilar from) their research target. They want to search these collections of sensor observations for data that matches their research criteria. However, it is getting harder to find the relevant data in the burgeoning volumes of datasets and observations, and the time involved in searching constrains scientist productivity and acts as a limit on discovery. For example, a microbiologist may be looking for “any observations near the Astoria Bridge in June 2009” in order to place a water sample taken there into physical context. Within the observatory, there are many observation types that the microbiologist needs to search. Observations range from a point in space at a point in time, such as a group of water samples, through fixed stations, which have a single point in space but may have a million observations spanning a decade, to mobile sensors. The mobile sensors may collect millions of observations over widely varying geographic and temporal scales: science cruises may cover hundreds of miles in the ocean over several weeks, while gliders and autonomic unmanned vehicles (AUVs) are often deployed for shorter time periods – hours or days – and a few miles, often in a river or estuary. Locating and scanning each potentially relevant dataset of observations is time-consuming and requires understanding each dataset's storage location, access methods and format; the scientist may not even be aware of what relevant datasets exist. Once geospatially located, fixed sensors can easily be filtered based on location but must still be searched on time; identifying whether mobile sensors were close by at the appropriate time may require time-consuming individual analyses of each sensor’s observations.

The scientists have powerful analysis and visualization tools available to them (e.g., [16, 25, 27]); however, these tools must be told the dataset and data ranges to analyze or visualize. While these tools allow the scientist to find needles in a haystack, they do not address the problem of which haystacks are most likely to contain the needles they want. Visualizing a dataset of observations for the desired location in June may confirm there is no match. However, potentially relevant substitutes “close by” in either time or space (say, from late May in the desired place, or from June but a little further away) are not found using current methods, much less ranked by their relevance. Even with a search tool that can find data in a temporal or spatial range, the scientist may not know how far to set those bounds in order to encompass possible substitutes.

We can meet this need by applying concepts from Information Retrieval. The scientists’ problem can be cast as a compound geospatial-temporal query across a collection of datasets containing geospatial and temporal data; the search results should consist of datasets ranked by relevance. The relevance score for each dataset should be an estimate of the dataset content’s geographic and temporal relevance to the query. The desire for real-time response implies that the query be evaluated without scanning each dataset's contents.

This paper describes a method for performing such a ranked search. Our contributions are:

1. An approach, described in Section 2, to scoring and ranking such datasets in response to a geospatial-temporal query. We calculate a single rank across both geospatial and temporal distances from the query terms by formalizing an intuitive distance concept. The approach is scalable and light-weight.
2. An approach, described in Section 3, for creating metadata describing the relevant geospatial and temporal characteristics of a collection of scientific datasets to support the ranking method. The metadata supports hierarchical nesting of datasets, providing scalability and flexibility across multiple collection sizes and spatial and temporal scales.
3. A loosely-coupled, componentized architecture that can be used to implement these approaches (Section 4).

4. A tool that implements these ideas and demonstrates their utility in the setting of an ocean observatory, in Section 5. Figure 5 shows the user interface.

We provide additional notes and implications of our approach in Section 6, describe related work in Section 7 and conclude with future research (Section 8).

In devising the details of our approach, we are biased towards identifying computationally light-weight approaches in order to achieve speed and scalability; as noted in considering the success of Google Maps, “Richness and depth are trumped by speed and ease, just as cheap trumps expensive: not always, but often.” [22] We are also biased towards exploiting well-studied and optimized underlying functions and techniques wherever possible. We assume that after a successful search the scientist (who we also call the user) will access some subset of the identified datasets; we generically refer to a “download”, although it may take other forms.

2 Ranking Space and Time

The scientist identifies a physical area and a time period he or she wishes to explore, which we will refer to as the *query*; we define the query as consisting of both geospatial and temporal *query terms*. The scientists have a qualitative intuition about which observations they consider a complete geospatial or temporal match, a relatively close match, or a non-match for their queries.

The top of Figure 1 shows a temporal query term, denoted T, with a line representing the query time span of “June”. We consider the temporal query to have a center and a radius; here, the center is June 15 and the radius 15 days. Lines A(t), B(t), ..., E(t) represent the time spans of observations stored in datasets A, B, …, E. Span A(t) represents a complete match; all observations in this dataset are from June. Span C(t)'s observations span the month of May and so is “very close”; similarly, Span B(t) is “closer” than Span C(t) but is not a complete match. Span D(t) is further away and Span E(t), with observations in February, is “far” from the June query.

The bottom section of Figure 1 shows a two-dimensional geospatial query term G as drawn on a map, represented by a central point *P* (in our running example, geocoordinate 46.23,-123.88, near the Astoria bridge), and a radius *r* (½ km) within which the desired observations should fall. The marker labeled A(g) represents the geospatial extent of observations in dataset A; here, they are at a single location, for example a fixed station or a set of observations made while anchored during a cruise. Extents B(g), E(g) and F(g) represent single-location datasets further away from the query center. Linear Extents C(g) and D(g) represent transects traveled by a mobile observation station such as a cruise ship, AUV or glider. Polygonal Extents J(g) and K(g) represent the bounding box of a longer, complex cruise track. Point Extent A(g) falls within the radius of the query and so is a complete match to the geographic query term. The qualitative comparison remains consistent across geometry types, with marker B(g) and line C(g) both being considered “very close” and polygon K(g) and marker F(g) being “too far” from the query to be interesting.

Intuitively, these qualitative comparisons can be scaled using a multiple of the search radius. For example, if the scientist searches for “within ½ km of P”, then perhaps a point 5 km away from P is “too far”. However, if the scientist searches for "within 5 km of P", then 5 km away from P is a match but 50 km is too far. In fact, the scientist is applying an implicit scaling model that is specific to his task [24].

The same intuitive scaling can be applied across both the temporal and spatial query terms; temporal observations at F(t) and spatial observations at marker B(g) could be considered equidistant from their search centers. Further, when considering both the temporal and spatial distances simultaneously, the dataset F, with temporal observations F(t) (quite close) at location F(g) (too far), is further from the query than datasets A ("here" in both time and space), B and C ("quite close" in both time and space). These examples illustrate the situation of one dataset *dominating* another: being closer in both time and space. The more interesting case arises in ranking two datasets where neither dominates the other, such as D and F: F is temporally closer, but D is closer in space. To simplify such comparisons, we propose a numeric distance representation that uses the query radii as the weighting method between the temporal and geospatial query terms. For example, had the spatial portion of the query been "within 5 km of P", D(g) and F(g) would both be considered "here" spatially, but D would now be dominated by F since it is temporally dominated by F.

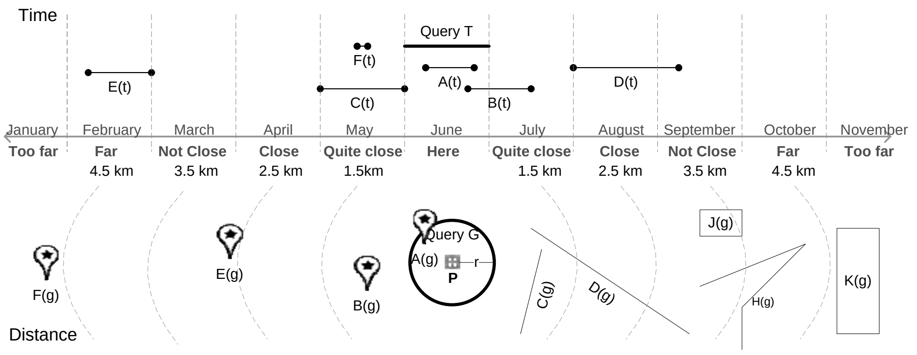


Fig. 1. Example of qualitative geospatial and temporal ranking: the top section shows a temporal query T and the time spans of various observation datasets. Dataset A(t) is a complete match, while datasets B(t), C(t), D(t), E(t) and F(t) are at increasing times from the query. The bottom section shows a geospatial query G, with the geospatial locations and extents of the same observation datasets represented by points (shown by markers), polygons and lines at various distances. In the middle is a qualitative scale that applies to both time and space.

In essence, the observations within a dataset represent a distribution of both temporal and geospatial distances from the query center, with a single point in time or space being the most constrained distribution. Each query term itself represents a distribution of times and locations. In order to rank the datasets, we need a single distance measure to characterize the similarity between the dataset and the query terms. There are many options for representing the proximity of two such entities, with varying computational complexities [23]. A commonly used surrogate for distance between two geographic entities is centroid-to-centroid distance. While it is a poor approximation when the entities are large and close together, it is relatively simple to calculate, at least for simple geometries. However, this measure ignores the radii of the query terms, and does not directly identify overlaps between the geometries. Another well-studied distance measure is minimum (and maximum) distance between two entities. This distance can be estimated by knowing only the bounds of the entities. This latter measure more closely matches our criteria; it can be calculated quickly using information (the bounds) that can be statically extracted from a dataset. This measure can be used to identify key characteristics that will drive our ranking: whether a dataset is within our query bounds and so is a complete match; whether the query and dataset overlap or whether they are disjoint, and if so by how much. This discussion applies equally to the one-dimensional "space" of time. In combining the space and time metrics, we will need to "scale" them by the radii of the respective query terms.

To compute these comparisons across a potentially large number of datasets, we have formulated a numerical similarity value that takes into account query-term radius and dataset distribution and can be cheaply estimated with summary information about temporal or spatial distributions, such as the bounds.

For the temporal term, let Q_{Tmin} and Q_{Tmax} represent the lower and upper bounds of the query time range. Further let d_{Tmin} and d_{Tmax} represent the minimum and maximum times of observations in dataset *d*. For calculation purposes, all times are translated into a monotonically increasing real number, for example "Unix time". Equation 1 below calculates d_{Rmin}, the distance of dataset *d*'s minimum time from the temporal query "center", i.e., the mean of Q_{Tmin} and Q_{Tmax}, then scales the result by the size of the query "radius", i.e., half its range. Similarly Equation 2 calculates d_{Rmax}, the "scaled time-range distance" of the dataset's maximum time. Equation 3 calculates an overall temporal distance d_{Tdist} for the dataset from the query: the first subcase accounts for a dataset completely within the query range, the second through fourth account for a dataset overlapping the query range above, below, and on both sides, and the last subcase accounts for a dataset completely outside of the query range.

Then, we let *s* represent a scaling function that converts the calculated distance from the query center into a relevance score, while allowing a weighting factor to be applied to the distance result; per Montello [24], the implicit scaling factor may change for different users or different tasks. Finally, Equation 4 calculates our overall time score d_{Ts} for this dataset by applying the scaling function to d_{Tdist}. In our current implementation, *s* is $(100 - f * d_{Tdist})$; that is, when the dataset is a complete match it is given a score of 100, whereas if it is *f* "radii" (currently *f* = 10) from the query center it is considered "too far" and given a score of 0 or less.

Similarly, let *C* represent the center location of the geospatial query and *r* the radius. Let the locations of all the observations within a single dataset *d* be represented by a single geometry *g*. By convention this geometry can be a point, line (or polyline) or polygon [12]. Let d_{Gmin} and d_{Gmax} represent the minimum and maximum distances of the geometry from *C*, using some distance measure such as Euclidean distance. Equation 5 calculates the overall distance measure for three subcases: the dataset is completely within the query radius; the dataset overlaps the query circle, or the dataset is completely outside the query circle. Equation 6 gives a geospatial-relevance score d_{Gs} for dataset *d* by again applying the scaling function s to the calculated overall distance measure.

In Equation 7, the geospatial score d_{Gs} and the temporal score d_{Ts} are composed to give an overall score d_{score}. Combining these two distance measures results in a multi-component ranking, which are the norm in web search systems today [7, 17, 18, 20].

We take a simple average of the two distance scores. Note, however, that each of these rankings has been scaled by the radii of the query terms; thus, the user describes the relative importance of time and distance by adjusting the query terms.

$$\begin{aligned} d_{R\min} &= \frac{d_{T\min} - ((Q_{T\max} - Q_{T\min})/2 + Q_{T\min})}{(Q_{T\max} - Q_{T\min})/2} \\ &= (2d_{T\min} - Q_{T\max} - Q_{T\min})/(Q_{T\max} - Q_{T\min}) \end{aligned} \tag{1}$$

$$d_{R\max} = (2d_{T\max} - Q_{T\max} - Q_{T\min})/(Q_{T\max} - Q_{T\min}) \tag{2}$$

$$d_{Tdist} = \begin{cases} 0 & d_{T\min} \geq Q_{T\min}, d_{T\max} \leq Q_{T\max} \\ \dfrac{(|d_{R\max}|-1)^2}{2|d_{R\max} - d_{R\min}|} & d_{T\min} \geq Q_{T\min}, d_{T\max} > Q_{T\max} \\ \dfrac{(|d_{R\min}|-1)^2}{2|d_{R\max} - d_{R\min}|} & d_{T\min} < Q_{T\min}, d_{T\max} \leq Q_{T\max} \\ \dfrac{(|d_{R\max}|-1)^2 + (|d_{R\min}|-1)^2}{2|d_{R\max} - d_{R\min}|} & d_{T\min} < Q_{T\min}, d_{T\max} > Q_{T\max} \\ (|d_{R\min} + d_{R\max}|/2) - 1 & d_{T\min} > Q_{T\max} \vee d_{T\max} < Q_{T\min} \end{cases} \tag{3}$$

$$d_{Ts} = s(d_{Tdist}) \tag{4}$$

$$d_{Gdist} = \begin{cases} 0 & d_{Gmax} \leq r \\ \dfrac{(d_{G\max}/r - 1)^2}{2(d_{G\max} - d_{G\min})/r} & d_{Gmin} \leq r, d_{G\max} \geq r \\ (d_{G\min} + d_{G\max})/r - 1 & d_{Gmin} > r \end{cases} \tag{5}$$

$$d_{Gs} = s(d_{Gdist}) \tag{6}$$

$$d_{score} = (d_{Gs} + d_{Ts})/2 \tag{7}$$

Given a collection of candidate datasets, each dataset's d_{score} can be calculated. Optionally, datasets with $d_{score} \leq 0$ can be discarded. Remaining datasets are sorted in decreasing order of d_{score} into a ranked list and become the results of the query.

We performed a 40-person user study, asking respondents to rank pairs of datasets in response to spatial, temporal, and spatial-temporal queries. The questions included comparisons with different geometries (e.g., polyline to point or polygon). Except for a small number of outlier cases, across all categories, when agreement amongst respondents is greater than 50% our distance measure agrees with the majority opinion. When there is a large disagreement with our distance measure, there is generally large disagreement amongst the respondents. Not surprisingly, these cases are correlated with small differences in distance between the two options.

3 Metadata Representing A Dataset Collection

The scoring and ranking approach described here assumes availability of suitable metadata against which to apply these equations. This section describes creating this metadata from datasets with geospatial and temporal contents, using the collection of observation datasets at our oceanography center as examples. We focus here only on *inherent metadata* [15], that is, information derived from the datasets themselves.

The base metadata requirements of our ranking and scoring approach are simple: the temporal bounds of each dataset, represented as a minimum and maximum time; the spatial footprint of each dataset, represented by a basic geometry type such as a point, line or polygon; and a dataset identifier. The temporal bounds can easily be extracted by scanning the dataset. Similarly, every dataset's observations fall within a geographic footprint. For a single point location such as a fixed sensor, the dataset's metadata record is created by combining the time range information with the fixed geographic location of the sensor.

Mobile sensors store a series of observations along with the geographic location and time for each observation. The overall dataset can be represented by the time range and the maximum geospatial bounds of the points within the dataset, that is, a rectangle (polygon) within which all points occur. The geospatial bounds can be extracted during the scan of the dataset, identifying the lowest and highest x and y coordinates found. For mobile sensors that follow a path or a series of transects during which the observations are collected (as in our case), a more informative alternative is available; the series of points can be translated into a polyline with each pair of successive points representing a line segment. If appropriate, the polyline can be approximated by a smaller number of line segments. The simplified polyline can be compactly stored as a single geometry and quickly assessed during ranking.

To provide for additional expressiveness across the range of possible dataset sizes and scales, we incorporate the idea of hierarchical, nested metadata. Across our collection of observations, we have locations where a single water sample was collected, locations with millions of sensor observations made over many years, and multi-week ocean cruises where millions of observations were collected across several weeks with tracks that crossed hundreds of miles. The hierarchical metadata allows us to capture a simple bounding box for a complex cruise, but also drill down to the individual cruise segments to identify the subset closest to the query terms.

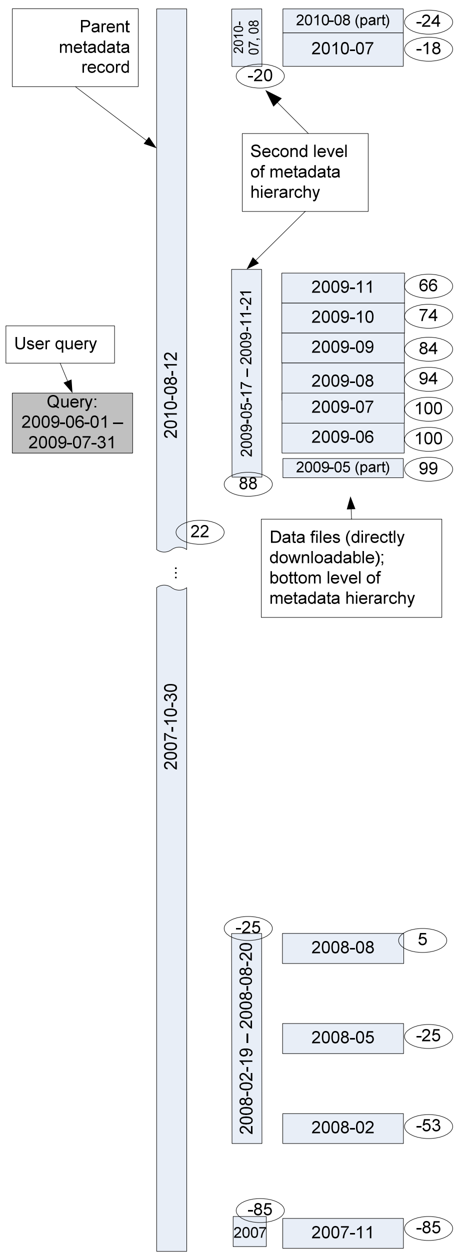


Fig. 2. Scoring example for intermittent data: the right-hand blocks represent downloadable data-sets; the left-hand blocks represent the metadata hierarchy and curation choices (one record per year, plus one for the lifetime). Ovals show the scores given each dataset relative to the query.

Metadata records are classified recursively into parents and children. A record with no parent is a *root* record. A parent record's bounds (both temporal and geospatial) must include the union of the bounds of its children. The children's regions might not cover all of the parent, for example, if there are gaps in a time series. A record with no children is a *leaf* record. A metadata collection is made up a set of root records and their children (recursively). The number of levels within the hierarchy is not limited. For instance, we might decompose a cruise temporally by weeks and days within weeks, then segment each spatially.

The scoring method is applied recursively to the collection of metadata records. We initially retrieve and score root metadata records only. If an entry is deemed interesting, it is added to a list of records whose children will be retrieved on the next pass. An entry is deemed interesting if the minimum geographic and time range distance is not "too far", and the minimum and maximum scaled time or geographic range distances are different from each other. The second criterion implies that if subdivisions of this dataset are available, some of these subdivisions may be more highly relevant than the parent dataset as a whole. We repeat until either the list of records to retrieve is empty or no interesting records have children.

Figure 2 demonstrates these concepts. It shows a fixed sensor station that reports data only during some months. Each light-gray block in the diagram represents a metadata record, showing time duration. In this case, three levels of metadata exist: an overall lifetime record, a medium level for the portion in each year that the station reports data, and a detailed level consisting of a record for each month. Next to each metadata record is shown its score for the given query. It can be seen that there are two individual months that score 100; datasets on either side score in the 90s. The year in which those months occur scores 88, whereas years that do not overlap the query range receive negative relevance scores. The overall lifetime record, which overlaps the query at both ends, receives a score of 22. Parent and child records are returned in the query result, allowing the scientist to choose between accessing only the months of interest or the entire year.

4 Architecture

As shown in Figure 3, our architecture extends existing observatory repositories. In general, observatories contain several major components: a network of sensors; a set of processes that collect observations and normalize them (adjust record formats, apply calibrations, etc.); a repository to store the normalized observations; and a set of analysis programs that access the stored observations. There may also be a web interface that allows the user to view the catalog and download specific subsets of the data. To these existing system components, we add four loosely coupled components: a metadata-creation component, a metadata repository, a scoring-and-ranking component and a user interface.

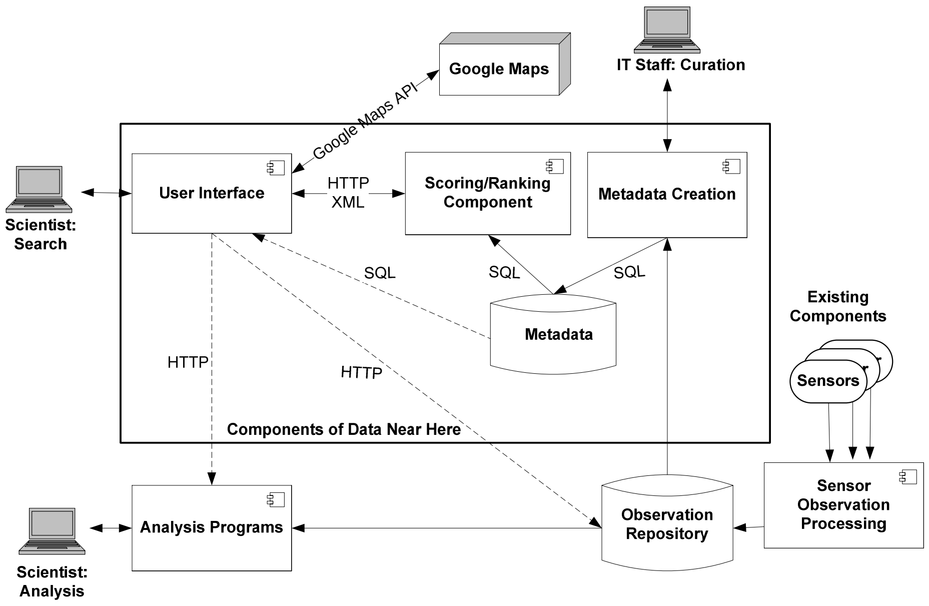


Fig. 3. The combined system and deployment diagram shows existing components and the new components added as part of Data Near Here.

The *metadata-creation component* extracts a minimal set of metadata from the contents of the observation repository to represent the source observations, and stores the extract into its own mini-repository. The goal is to support fast query access by creating a simple abstraction over a far more complex data repository. The IT staff can add new categories of observations (e.g., new types of mobile devices), change the number or grouping of hierarchical levels used to represent data, or change the representation of a category of observations (e.g., treating cruises solely as lines rather than as lines and bounding boxes at different levels of the hierarchy); this activity is a *data curation* process [13]. At present, these changes involve writing or modifying scripts; an informal set of patterns is emerging and could be formalized if desired.

The *scoring-and-ranking component* receives query terms from the user interface and interacts with the metadata. It scores each candidate metadata record, and returns to the user interface a set of ranked records. The scoring and ranking algorithm is loosely coupled with the metadata and is independent of the user interface, allowing different algorithms to be easily tested without modifying the other components.

The *user interface* is responsible for collecting the geospatial and temporal query terms from the user and presenting the search results; it also provides the user with some control over the presentation (e.g., the number of search results to return). The user interface exploits Google Maps [3] for geospatial representation of the query and results. The sole direct interaction between the user interface and the metadata is when the user interface requests metadata information to populate the query interface's selections (for example, the 'Category' entry field in Figure 5). The search results link to the datasets within the repository and optionally to analysis programs.

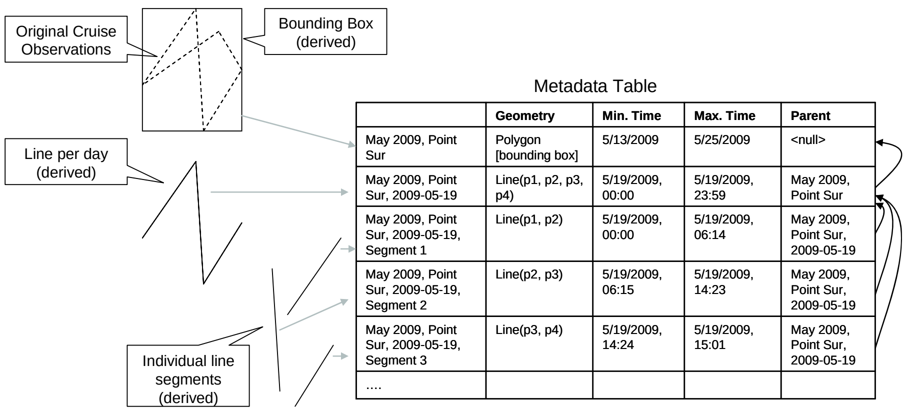
The loosely coupled nature of the components allows maximum flexibility in altering the internal design or methods used by any component without altering the remaining components; the additive nature of the architecture minimizes changes to the existing infrastructure necessary to add this capability.

5 "Data Near Here": An Implementation

The approaches described in this paper have been implemented in an internal prototype at the Center for Coastal Margin Observation and Prediction (CMOP). This center's rich inventory of over 250 million observations is available for public download or direct analysis; additional data can be accessed internally via a variety of tools. The observations and associated metadata are stored in a relational database: most datasets are also stored in NetCDF-formatted downloadable files.

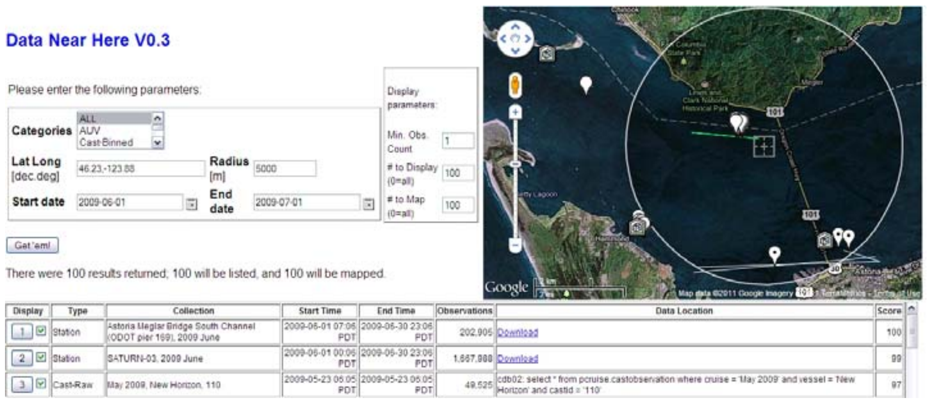
The observational sensors can be loosely grouped by their deployment on fixed or mobile platforms. Mobile sensors are deployed in a series of *missions*, each of which may span hours or days or weeks. Observations may be captured many times a second, either continuously or according to some schedule; there may be a half million or more observations per mission. Hierarchically nested metadata is created at multiple scales; for the Astoria Bridge query, a fixed station that is far distant can be recognized and ignored by looking at a single lifetime entry for the station.

A fixed sensor has a single geographic location over time; its dataset can be geospatially characterized as a single point. Its continuous observations are, for convenience, stored in multiple datasets, each containing a single time range such as a month or (for sparser observations) a year. In addition to dataset leaf records, for each year's worth of observations we create a parent record that summarizes that year's data, plus a lifetime record for the overall time duration of the station.


Original Cruise Observations
Bounding Box (derived)
Line per day (derived)
Individual line segments (derived)
Metadata Table

	Geometry	Min. Time	Max. Time	Parent
May 2009, Point Sur	Polygon [bounding box]	5/13/2009	5/25/2009	<null>
May 2009, Point Sur, 2009-05-19	Line(p1, p2, p3, p4)	5/19/2009, 00:00	5/19/2009, 23:59	May 2009, Point Sur
May 2009, Point Sur, 2009-05-19, Segment 1	Line(p1, p2)	5/19/2009, 00:00	5/19/2009, 06:14	May 2009, Point Sur, 2009-05-19
May 2009, Point Sur, 2009-05-19, Segment 2	Line(p2, p3)	5/19/2009, 06:15	5/19/2009, 14:23	May 2009, Point Sur, 2009-05-19
May 2009, Point Sur, 2009-05-19, Segment 3	Line(p3, p4)	5/19/2009, 14:24	5/19/2009, 15:01	May 2009, Point Sur, 2009-05-19
....				

Fig. 4. Space metadata records for mobile stations (here, a multi-week cruise) are created by creating a line from point observations and simplifying it (middle hierarchy level, on line 2 of the table), then splitting the line into detailed line segments for the leaf records and extracting a bounding box for the parent record.


Data Near Here V0.3

Please enter the following parameters:

Categories	ALL / AUV / Cast-Binned		
Lat Long [dec.deg]	46.23,-123.88	Radius [m]	5000
Start date	2009-06-01	End date	2009-07-01

Display parameters:

Min. Obs. Count	1
# to Display (0=all)	100
# to Map (0=all)	100

Get 'em!

There were 100 results returned, 100 will be listed, and 100 will be mapped.

Display	Type	Collection	Start Time	End Time	Observations	Data Location	Score
1 ✓	Station	Astoria Meglar Bridge South Channel (ODOT pier 169), 2009 June	2009-06-01 07:06 PDT	2009-06-30 23:06 PDT	202,905	Download	100
2 ✓	Station	SATURN-03, 2009 June	2009-06-01 00:06 PDT	2009-06-30 23:06 PDT	1,667,988	Download	99
3 ✓	Cast-Raw	May 2009, New Horizon, 110	2009-05-23 06:05 PDT	2009-05-23 06:05 PDT	49,525	cdb02: select * from pcruise.castobservation where cruise = 'May 2009' and vessel = 'New Horizon' and castid = '110'	97

Fig. 5. Map display of Data Near Here search results for the example query in this paper. The map shows a section of the Columbia River near its mouth that includes Highway 101 crossing the Astoria Bridge between Oregon and Washington. The search center and radius are shown along with a set of markers and lines locating the highest-ranked datasets found for the search. The list below the map shows the four highest-ranked results, the first of which is a complete match; the next three are close either in time or space, but are not complete matches.

As is shown in Figure 4, the track for a mobile-sensor mission can be a represented by a polyline. In order to extract the polyline from the observations, we use the PostGIS *makeline* function to convert each day's worth of observations into a polyline, then apply the PostGIS implementation of the Douglas-Peucker algorithm, *simplify*, to create a simplified polyline. The simplified polyline, along with the day's start and end time, is stored as a metadata record. We create an additional metadata record for the lifetime of the mission; this record is simply the bounding box of the polylines with the begin and end times of the overall mission. We then programmatically extract each line segment from the simplified polyline, match the vertices to the time the mission was at that location, and store each line segment with its time range as a leaf metadata record. This three-level hierarchy for mobile sensors can be created quickly, and provides multiple scales of metadata.

At the end of these processes, we have a consistent metadata format for both fixed and mobile sensor observations. We also have the option of storing multiple sets of metadata representing the same (or similar) underlying data, if, for example, alternative groupings of the data are more appropriate for specific user groups (for example, partitioned by day or by tide). A varying number of levels can be used for a subset of the collection or even a subset of sensors within a specific category; we may wish to, for example, add a daily metadata record for specific fixed sensors. In other cases, such as water-sample data, we chose to only have one level in the hierarchy.

Keeping the metadata up-to-date involves adding new metadata records as new missions occur or new datasets are created. For each category of data, this update can occur automatically via a set of scripts and triggers that check for new datasets and execute the predefined steps. The moment a new metadata record is created, it is available to be searched. Setting up a new category of data requires deciding the number of hierarchical levels to be defined and the download granularities to support, and then setting up the appropriate scripts.

Figure 5 shows the tool's user interface. The user interface combines three interacting elements: a set of text query entry fields, a Google map that can be used to locate the geospatial query and on which the geospatial locations of highly ranked results are drawn, and the query results: a table of highly scoring datasets ordered by score. All available categories of observational data can be searched, or the scientist can limit the search to a subset. Scientists can provide both time and location parameters; they can also search for all times in which observations were taken at a specific location by leaving the time fields blank, and vice versa. The top-ranked results will be displayed on the map – the scientist can select how many results to return and to display. Clicking on a displayed dataset pops up a summary.

A "data location" field provides access to the data. Where the data can be directly downloaded, this field contains a download link. This link is built when the metadata is created and can contain parameters that subset the complete dataset to the relevant portion if the download mechanism allows. In cases where direct download is not currently possible, this field provides the scientist with the dataset location and an extract command for the dataset's access tool; for example, where the data is held only in a relational database, this field can contain a SQL Select statement to extract the relevant data. A future version will allow scientists to directly open a selected dataset in a visualization and analysis tool.

The technologies used to implement the shown architecture were selected based on existing technologies in use in the infrastructure, to allow for easy integration, extension and support. Metadata creation is performed in a combination of SQL and scripts. The repository is a PostGIS/Postgres database and is accessed via dynamic SQL; the footprint data is stored in a PostGIS geometry column. The scoring and ranking component is written in PHP. Geometric functions are performed by PostGIS during data retrieval from the repository, with final scoring and ranking performed in the PHP module. The user interface is implemented using Javascript, JQuery and the Google Maps API. Current experience leads us to believe these technologies will scale to support the observatory's repository for some time. For a much larger repository, other technology choices would provide greater speed. The architecture allows us to easily make these choices per component as needed.

6 Discussion

Here we discuss the tradeoff between user performance and the design of the metadata hierarchy. The response time seen by the user is driven by several main factors: data transfer times between the components (scoring component to user interface, metadata repository to scoring component); the number of hierarchical levels of metadata; the total number of metadata records to be scored; and the complexity of the scoring algorithm.

The intent of the metadata hierarchy is to bridge the gap between the dataset granularity and the footprint of the dataset's content, within the context of efficient real-time user search. The more hierarchical levels, the more queries must be issued to process the children of interesting metadata records; however, the hierarchical design should allow fewer metadata records to be scored overall. An alternative is to score all metadata records in a single query; however, as many of the roots will have an increasing number of descendents over time (e.g., stations that continue to collect data month after month), we expect that ruling out descendents by examining only the parent record will balance the overhead of multiple queries and allow for greater scalability. We expect the user, after a successful search, to download or analyze selected datasets from the results presented. Thus, there is an assumed alignment between a single metadata record and a single accessible or downloadable unit (such as a single dataset). However, in many cases the capability exists to group multiple datasets into a single accessible unit (e.g., by appending them), or alternatively to access subsets of a dataset (e.g., by encoding parameters to limit the sections of the dataset to access). The data curation process should consider the typical footprint and the likely utility to the scientist of different aggregations of that data.

From a query-performance perspective, the number of leaf metadata records is optimal when each dataset is described by a single metadata record and thus there is only one record per dataset to score and rank. Where a single dataset is geospatially and temporally relatively homogenous, this arrangement may be a practical choice. Where a dataset is geospatially or temporally very diverse or is too large to conveniently download, users are best served if a leaf metadata record exists for each subcomponent or segment they may wish to download. The hierarchy provides a mechanism for mediating this mismatch; a single metadata record can be created for a larger dataset with children for the subcomponents. The scoring component may be able to eliminate the dataset and its children from further consideration based on the parent, and only score the children when the parent appears interesting.

To provide a tangible example of this tradeoff, Table 1 shows summary counts for our currently existing metadata records, representing a subset of CMOP's repository. The breakdown by category in Table 2 highlights the different curation choices made for different observation categories. At one extreme, the 22 fixed stations have an average of 8.2 million observations each, and here a three-level hierarchy has been created. At the other extreme is the water-sample collection, with two observations taken per location and time. The same “cast” data is represented in two forms: one is the unprocessed, or “raw”, collection of observations; the same data has also been binned to specific depths and averaged into a much smaller collection of measurements. Variation in geometric representation is also shown; in cruises, for example, the most detailed level is most commonly represented by line segments representing specific cruise transects, but is sometimes represented by points when the cruise vessel was anchored in a single location for a longer period of time. These different representations are easily discerned programmatically from the data but are difficult for a user to identify from the source data without significant effort.

Table 1. Characterization of Data Near Here Metadata. This table summarizes characteristics of the metadata records representing the 225 million observations currently searchable**.**

Metadata records	15,516
Number of observation categories	7
Records at each hierarchy level	
Roots without children	6,564
Roots with children	60
Children with children	800
Children with no children	8,092
Observations represented	225,627,211
Average observations per metadata record	14,541

Table 2. Characterization of Existing Metadata Records by Category.

Category	**Hierarchy Level**	**Geometry**	**Number of Records**	**Number with Children**	**Total Observations Represented**	**Average Observations per Record**
AUV	1	Polygon, Line	22	11	225,757	10,261
	2	Line	29	0	134,841	4,649
Cast-Binned	1	Point	3,066	0	370,967	120
Cast-Raw	1	Point	2,908	0	33,908,614	11,660
Cruise	1	Polygon	20	20	8,064,259	403,212
	2	Line	607	607	8,064,259	13,285
	3	Line, Point	7,125	0	7,615,222	1,068
Glider	1	Polygon	7	7	2,237,628	319,661
	2	Line	128	128	2,237,628	17,481
	3	Line	357	0	1,670,470	4,679
Fixed Stations	1	Point	22	22	180,818,279	8,219,012
	2	Point	65	65	171,903,806	2,644,673
	3	Point	581	0	180,818,239	311,219
Water Samples	1	Point	579	0	1,707	2

The spatial scoring equations were designed to provide a reasonable approximation of distance for the three primary cases – polygon, polyline and point – while minimizing the number and complexity of spatial calculations needed; the current approach uses a total of two spatial calculations (maximum distance and minimum distance between two geometries) for each metadata record scored. Spatial functions can be slow, so minimizing the number and complexity of geometries handled is beneficial. A more complex spatial scoring system can easily be devised; what is less clear is whether, given the uncertainties in people's views of distance [24], the additional complexity provides a better distance score as perceived by the user. What is clear is that the additional complexity will add to the computation time.

7 Related Work

Adapting a definition from Information Retrieval (IR) [20], a dataset is relevant if the scientist perceives that it contains data relevant to his or her information need. In IR systems, the user provides query terms, usually a list of words, to be searched for against an index representing a library of items (where each item may be, for example, a web page). Each item is summarized as index entries of the words found in the document, created prior to receiving the user's query. In almost all cases, the searches are performed against metadata, which itself varies in source and form. In ranked retrieval, each item is given a score representing an estimate of the item's relevance to the query. The list of items is then ranked by ordering items from highest to lowest score. There is much research (e.g., [4, 20, 21]) into ranked relevance of unstructured text documents against text queries. We adapt these ideas to searching contents of scientific datasets with a query consisting of geospatial-temporal search terms which are themselves ranges. The metadata we extract from the datasets performs the role of the index.

Hill et al. [15] present a system for describing and searching a library's digital collection of geographic items. They apply widely accepted collection concepts from paper-based archives that are based on a textual description of a map series (publisher, title, number in series, etc.) to digital map collections. A single collection may contain a set of maps where each map has a different geographic coverage; however, the specific map's geographic coverage is an access or index key to that map. The challenge is how to represent these collections by searchable metadata. They differentiate *contextual metadata*, which is externally provided (e.g., publisher), from *inherent metadata*, derived from automated analysis of the data (e.g., count of items included in a collection). This automatic data analysis adds to the metadata but does not allow the content itself to be searched. They do not provide hierarchical metadata, nor do they discuss methods for ranked search results.

Grossner et al. [11] provide a summary of progress in the last decade in developing a "Digital Earth", and identify gaps in efforts so far. They note that current geographic and temporal search responses provide matches only on one level of metadata; the contents of cataloged digital objects are not exposed and are not searchable. Goodchild [8] notes that most geographic search systems score items based on word matches against metadata without considering the temporal span or geographic content of the items returned, and recognizes [9] the issue of *containment* as an open research question. That is, a map may be cataloged by the extent of its coverage (e.g., “Alaska”) but the search mechanism has no method with which to recognize that this map is a match for an item contained within it, (e.g., a search for “Fairbanks”). Goodchild et al. [10] expand on these concerns in the 2007 review of Geospatial One-Stop (GOS) [1], a state-of-the-art government portal to geographic information. GOS and similar portals such as the Global Change Master Directory’s Map/Date Search [2] now allow searches using both geographic and temporal criteria; three spatial tests are supported (the map view intersects, mostly contains, or completely contains the dataset), and temporal search appears binary – if items do not match the criteria they are not returned. Only one level of metadata is considered; if a relevant item is embedded within a larger item (Fairbanks within Alaska), the relevant item is not returned. In contrast, we explicitly rank returned items based on both the temporal and geographic “distance” of the dataset contents from the query, and address the containment issue with multiple levels of metadata.

One widely-used geospatial search system is Google Maps [22], which searches for a place name or a specified latitude and longitude, and provides nearby points of interest (“restaurants near here”). They do not currently expose a temporal search capability. It is possible for a site to explicitly link a dataset to a specific location using KML, but it is not currently possible to search ranges within linked datasets. Egenhofer [6] describes some desired geographic request semantics but does not propose an implementation.

Addressing a different kind of geographic search problem, Sharifzadeh and Shahabi [26] compare a set of data points with a set of query points where both sets potentially contain geographic attributes, and identify a set of points that are not *dominated* by any other points. They do not specifically address time, but could presumably treat it as another attribute. Their approach develops the database query and algorithm to return the best points, but, unlike our approach, they do not return ranked results nor place the queries within the context of a larger application.

Several researchers [16, 25, 27] have addressed the difficulty scientists have in finding “interesting” data — data relevant to the scientist’s research question — within the exploding quantity of data now being recorded by sensors by focusing on visualization techniques for a specified set of data. The scientist specifies the dataset and range of data within the dataset. The system then presents a visualization of the specified numeric data. The question of how the scientist finds interesting datasets and ranges to visualize is not addressed; that question is the subject of this research.

8 Conclusion

The rapid expansion of deployed observational sensors has led to collection of more observational data than ever before available. The sheer volume of data is creating new problems for scientists trying to identify subsets of data relevant to their research. Techniques to help scientists navigate this sudden plethora of data are a fruitful area for research. This work is one such contribution, focusing on the problem of finding sets of observations “near” an existing location in both time and geospace.

This paper presents a novel approach to providing compound geospatial-temporal queries across a collection of datasets containing geospatial and temporal data; search results consist of datasets ranked by relevance and presented in real time. The approach combines hierarchical metadata extracted from the datasets with a method for comparing distances from a query across geospatial and temporal extents. This approach complements existing visualization techniques by allowing scientists to quickly identify which subset of a large collection of datasets they should review or analyze. The combination of data represented by its geospatial and temporal footprint, using the metadata for search, the metadata hierarchical design and overall loosely-coupled architecture allows for scalability and growth across large, complex data repositories. The prototype described already supports over quarter of a billion observations and more are being added. User response has been very positive.

We plan to extend this work in several directions, including characterizing the observed environmental variables and supporting more expressive queries. The third geospatial dimension, depth, is currently being added. *Contextual metadata* [15] – ownership, terms and conditions, etc. – will be added as the tool gains wider use. The eventual goal is to combine geospatial-temporal search terms with terms such as "with oxygen below 3 mg/liter, where Myrionecta Rubra are present".

Finding relevant data is key to scientific discovery. Helping scientists identify the "haystacks most likely to contain needles" out of the vast quantities of data being collected today is a key component of reducing their time to discovery.

Acknowledgments. This work is supported by NSF award OCE-0424602. We would like to thank the staff of CMOP for their support.

References

1. Geospatial One Stop (GOS), http://gos2.geodata.gov/wps/portal/gos.
2. Global Change Master Directory Web Site, http://gcmd.nasa.gov/.
3. The Google Maps Javascript API V3, http://code.google.com/apis/maps/documentation/javascript/.
4. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. ACM Press New York (1999).
5. Douglas, D.H., Peucker, T.K.: Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. Cartographica. 10, 2, 112–122 (1973).
6. Egenhofer, M.J.: Toward the semantic geospatial web. Proceedings of the 10th ACM international symposium on advances in geographic information systems. pp. 1–4 (2002).
7. Evans, M.P.: Analysing Google rankings through search engine optimization data. Internet Research. 17, 1, 21–37 (2007).
8. Goodchild, M.F., Zhou, J.: Finding geographic information: Collection-level metadata. GeoInformatica. 7, 2, 95–112 (2003).

9. Goodchild, M.F.: The Alexandria Digital Library Project: Review, Assessment, and Prospects, http://www.dlib.org/dlib/may04/goodchild/05goodchild.html, (2004).
10. Goodchild, M.F. et al.: Sharing Geographic Information: An Assessment of the Geospatial One-Stop. Annals of the AAG. 97, 2, 250-266 (2007).
11. Grossner, K.E. et al.: Defining a digital earth system. Transactions in GIS. 12, 1, 145–160 (2008).
12. Herring, J.R. ed: OpenGIS® Implementation Standard for Geographic information - Simple feature access - Part 1: Common architecture, (2010).
13. Hey, T., Trefethen, A.: e-Science and its implications. Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences. 361, 1809, 1809 (2003).
14. Hey, T., Trefethen, A.E.: The Data Deluge: An e-Science Perspective. Grid Computing: Making the Global Infrastructure a Reality (eds F. Berman, G. Fox and T. Hey). pp. 809-824 John Wiley & Sons, Ltd, Chichester, UK (2003).
15. Hill, L.L. et al.: Collection metadata solutions for digital library applications. J. of the American Soc. for Information Science. 50, 13, 1169–1181 (1999).
16. Howe, B. et al.: Scientific Mashups: Runtime-Configurable Data Product Ensembles. Scientific and Statistical Database Management. pp. 19–36 (2009).
17. Kobayashi, M., Takeda, K.: Information retrieval on the web. ACM Comput. Surv. 32, 144–173 (2000).
18. Lewandowski, D.: Web searching, search engines and Information Retrieval. Information Services and Use. 25, 3, 137-147 (2005).
19. Lord, P., Macdonald, A.: e-Science Curation Report, http://www.jisc.ac.uk/uploaded_documents/e-ScienceReportFinal.pdf, (2003).
20. Manning, C.D. et al.: An introduction to information retrieval. Cambridge University Press (2008).
21. Maron, M.E., Kuhns, J.L.: On relevance, probabilistic indexing and information retrieval. Journal of the ACM (JACM). 7, 3, 216–244 (1960).
22. Miller, C.C.: A Beast in the Field: The Google Maps mashup as GIS/2. Cartographica. 41, 3, 187-199 (2006).
23. Miller, H.J., Wentz, E.A.: Representation and Spatial Analysis in Geographic Information Systems. Annals of the AAG. 93, 3, 574-594 (2003).
24. Montello, D.: The geometry of environmental knowledge. Theories and methods of spatio-temporal reasoning in geographic space. 136–152 (1992).
25. Perlman, E. et al.: Data Exploration of Turbulence Simulations Using a Database Cluster. Proceedings of the 2007 ACM/IEEE conference on Supercomputing. pp. 1–11 (2007).
26. Sharifzadeh, M., Shahabi, C.: The spatial skyline queries. Proc. of VLDB. p. 762 (2006).
27. Stolte, E., Alonso, G.: Efficient exploration of large scientific databases. Proc. of VLDB. p. 633 (2002).