Characterizing and Selecting Fresh Data Sources

Theodoros Rekatsinas
University of Maryland
thodrek@cs.umd.edu

Xin Luna Dong
Google Inc.
lunadong@google.com

Divesh Srivastava
AT&T Labs-Research
divesh@research.att.com

ABSTRACT

Data integration is a challenging task due to the large numbers of autonomous *data sources*. This necessitates the development of techniques to reason about the benefits and costs of acquiring and integrating data. Recently the problem of *source selection* (i.e., identifying the subset of sources that maximizes the profit from integration) was introduced as a preprocessing step before the actual integration. The problem was studied for static sources and used the accuracy of data fusion to quantify the integration profit.

In this paper, we study the problem of source selection considering *dynamic* data sources whose content changes over time. We define a set of time-dependent metrics, including *coverage*, *freshness* and *accuracy*, to characterize the quality of integrated data. We show how statistical models for the evolution of sources can be used to estimate these metrics. While source selection is NP-complete, we show that for a large class of practical cases, near-optimal solutions can be found, propose an algorithmic framework with theoretical guarantees for our problem and show its effectiveness with an extensive experimental evaluation on both real-world and synthetic data.

Categories and Subject Descriptors

H.2.m [**Database Management**]: Miscellaneous; G.3 [**Mathematics of Computing**]: [Probability and Statistics]

General Terms

Algorithms, Experimentation, Performance

Keywords

Source Selection; Data Integration; Dynamic Data Sources

1. INTRODUCTION

Integrating data from multiple sources is essential in a growing number of application domains, including large scale enterprises that own many data sources, prediction of societal events, such as disease outbreaks, and targeted data analytics where streams of Web and social media data are heavily used. Analyzing multiple data sources collectively can significantly enhance the value of data; however, acquiring and integrating data comes with a monetary and computational cost. To reason about the *profits* of integration and find the optimal subset of sources to be integrated, Dong et al. [3] introduced the problem of *source selection* focusing on *static sources*, that is, data sources whose content does not change over time. Often, however, data sources are *dynamic* (i.e., their content changes over time) raising several challenges. Next, we use two real-world scenarios to illustrate these challenges.

SIGMOD'14, June 22–27, 2014, Snowbird, UT, USA.
Copyright 2014 ACM 978-1-4503-2376-5/14/06 ...$15.00.
http://dx.doi.org/10.1145/2588555.2610504.

1.1 Challenges

The first scenario is that of *listing aggregation*, such as business, job or rental listings. Typically, aggregators offer a search service to end users by integrating listings from multiple *sources*. Each source provides a set of listings and regular updates as new listings become available, or existing listings get updated or removed. Specifically, we consider aggregating business listings (BL) from 43 data sources providing records for US businesses over 2 years.

A second scenario that is increasingly popular is that of collective analysis of online news media for societal-event monitoring [7, 18]. Here, the analyst integrates events mentioned in a diverse set of news media sources and analyzes them collectively to detect patterns characterizing her domain of interest. In particular, we consider the Global Database of Events, Languages and Tone (GDELT) [10] where news articles from 15,275 sources are aggregated in a single repository over one month for analytic tasks.

The first challenge stems from the fact that sources that update their data more frequently are not always more effective at capturing changes in a timely manner.

EXAMPLE 1. *We focus on BL and the freshness of each source (i.e., the ratio of provided up-to-date listings to the total number of listings in the source) and the update frequency of each source. Figure 1(a) shows the average update frequency and average freshness for each source over the 2 year time window. We observe that there is no clear correspondence between the update frequency and freshness of a source and see that sources with high update frequencies may have low freshness, indicating that sources may add to their content frequently but are ineffective at deleting stale data or capturing value changes of previous data items.* □

Even sources with similar update frequencies exhibit different levels of staleness, as exemplified next in our second domain.

EXAMPLE 2. *We consider the 20 largest sources from GDELT and examine how effective sources are at reporting events in a timely manner. Figure 1(d) shows the average delay with which events are reported and the corresponding fraction of delayed events over the total content of each source over one month. While all sources get updated daily, we see that a significant fraction of events are reported with delays.*□

The second challenge is that the quality of available sources may change over time and often the subset of sources that maximizes the integration quality may also change over time.

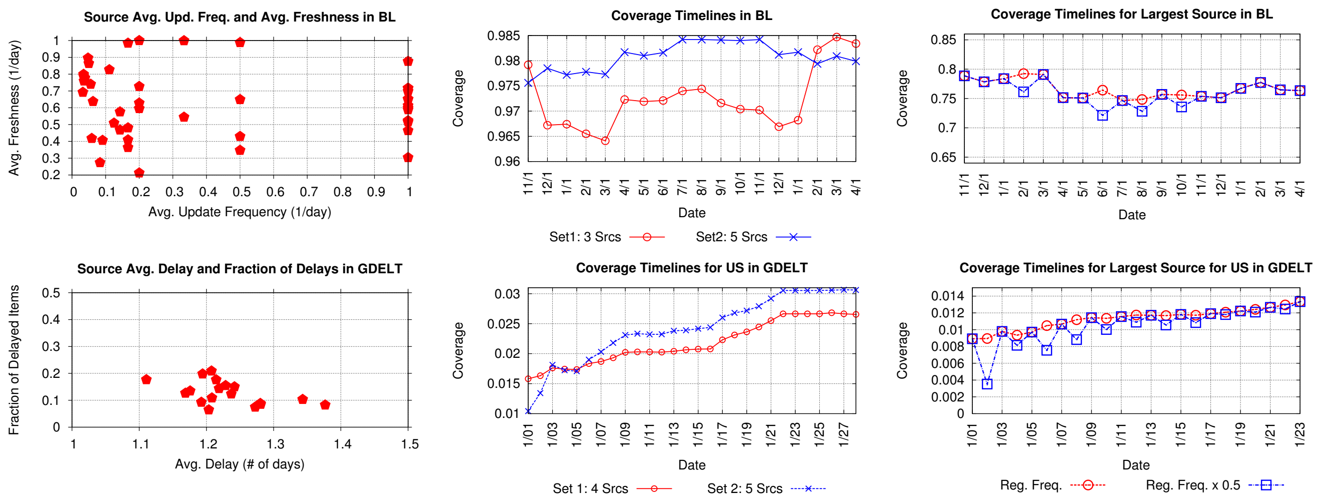


Figure 1: (a) Average update frequencies and average freshness of data sources in BL. (b) Coverage evolution for two sets of sources in BL. (c) Evolution of coverage for the largest source, when incorporating updates with different frequencies for BL. (d) Average delay and fraction of delayed event mentions for the 20 largest sources in GDELT. All sources get updated every day. (e) Coverage evolution for two sets of sources for GDELT corresponding to events in the US. (f) Evolution of coverage for the largest source, when incorporating updates with different frequencies for GDELT.

EXAMPLE 3. *We focus on listings for a single state in BL and consider the coverage (i.e., the ratio of provided instances to total number of instances in the domain) of the integration result for two sets of sources. Both sets contain the two largest sources. Moreover, the first set contains one other source while the second set contains three other sources, with comparable sizes to the source added in the first set. Figure 1(b) shows the evolution of coverage for the two sets. Switching to GDELT, we focus on events for the United States and consider two sets of sources, both containing the two largest sources. The first one contains two extra smaller sources, while the second has three other sources of comparable size. Figure 1(e), shows the coverage evolution for the two sets. In both cases, we observe that the coverage of the two sets differs over time, and the set with the highest quality varies across time.* □

Finally, the third challenge is that choosing to integrate data from a source at a lower frequency than the source update frequency can lead to similar integration quality but reduced cost.

EXAMPLE 4. *For BL, we consider the evolution of coverage for the largest source, when its updates are acquired at half the update frequency. As shown in Figure 1(c), the quality loss is not significant while the cost is reduced significantly since only half of the updates are acquired. We observe the same behavior in GDELT as illustrated in Figure 1(f).*□

1.2 Contributions

Motivated by these examples, we study the problem of *time-aware source selection*, that is, reasoning about the profit of acquiring and integrating dynamic sources to select the optimal subset of sources to be integrated. We build upon the work by Dong et al. [3] that solved this problem for static sources.

We propose a framework that provides the necessary building blocks to derive rigorous time-dependent definitions for data quality metrics, such as coverage and freshness, and statistically models the complex update patterns and data quality changes of different data sources. These models enable us to efficiently and accurately estimate the aforementioned quality metrics at different time points. Our main contributions are as follows:

- We introduce the problem of *time-aware source selection*, where we consider time-dependent data quality metrics to describe the benefit of data integration, and in addition to selecting a subset of sources, we decide the optimal frequency to acquire data from each source (Section 2).
- We introduce a theoretical framework using statistical models to describe the quality and data update patterns of dynamic sources. An overview is presented in Section 2.3 and its components are discussed in detail in Sections 3 and 4.
- While the problem of time-aware source selection is NP-complete [3], we show that many of its instances correspond to well-studied submodular optimization problems for which efficient local-search algorithms with rigorous theoretical guarantees are known (Section 5).
- Finally, we show that our techniques can efficiently find near-optimal solutions on both real-world and synthetic data that contain large numbers of data sources exhibiting various update patterns (Section 6).

2. AN OVERVIEW

In this section we briefly review the source selection problem for static sources. We formally define the problem of *time-aware source selection* and present an overview of our solution.

2.1 Source Selection

Given a set of data sources, we assume two predefined functions that measure the *cost* and *gain* of integration. The cost of integration is a function of the monetary cost to acquire data, and the total resources (including time) needed for integration. The gain quantifies the benefit from the integration and is a function of the integration quality using the same unit as for cost. We define source selection as follows:

DEFINITION 1. (SOURCE SELECTION [3]) *Let $\bar{\mathbf{S}}$ be a set of sources, F be an integration model, $G_F(\cdot)$ be a gain function and $C_F(\cdot)$ a cost function using model F, and β_c be a budget on cost. The Source Selection problem finds a subset $S_I \subseteq \bar{\mathbf{S}}$ that maximizes $G_F(S_I) - C_F(S_I)$ under constraint $C_F(S_I) \leq \beta_c$.*

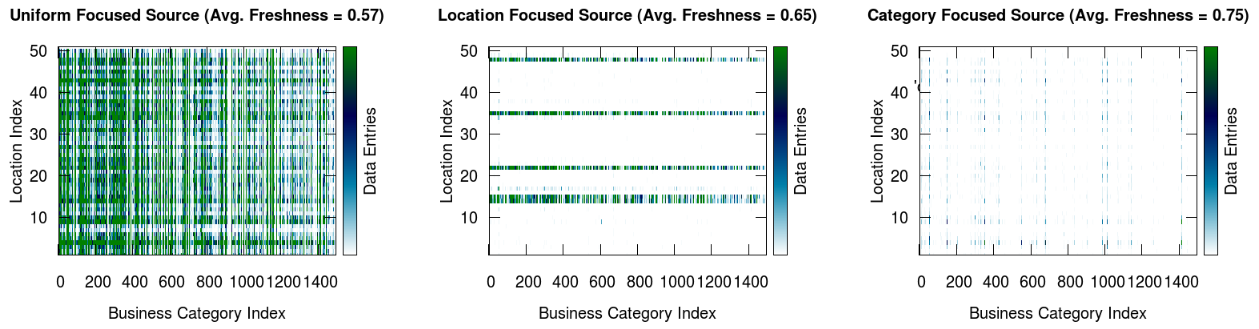


Figure 2: Sources covering different parts of the data domain.

Dong et al. [3] show how $G_F(S_I)$ can be defined in the context of data fusion as the accuracy of the integration result, and use an additive cost function over the costs of the selected sources.

2.2 Selecting Fresh Sources

We consider a data domain Ω where entities change dynamically over time, that is new entities may *appear*, *disappear* or the values of existing entities may change over time. In the remainder of the paper we will refer to this domain as the *world* for convenience. *Dynamic sources* can be viewed as observers of the world that update their content by capturing changes in the world.

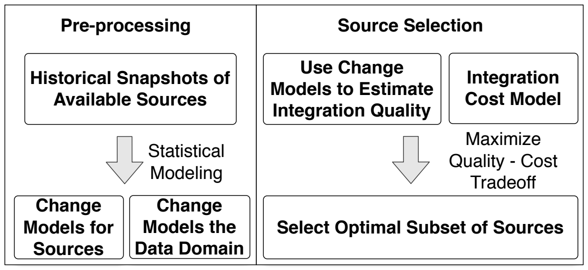
DEFINITION 2. *A data source S is dynamic when it is updating its content by capturing entity appearances, disappearances and value changes from a data domain Ω with a frequency f_S.*

We assume knowledge of the data evolution for a past time window T ending at time t_0. We consider a fixed set of future time points, denoted by T_f, and we wish to maximize the profit of integration for Ω and T_f. Let $G_F(S_I, T_f)$ be the overall gain of integrating S_I using model F for T_f, and $C_F(S_I, T_f)$ be the corresponding integration cost. With $G_F(S_I, t)$ denoting the gain of integrating S_I for a single time point $t \in T_f$, and $A_{t\in T_f}$ denoting an aggregate function (e.g., average or max) over the time points in T_f we define the overall gain as $G_F(S_I, T_f) = \mathcal{A}_{t\in T_f} G_F(S_I, t)$. Similarly to Dong et al. [3] we assume an additive cost model with $C_F(S_I, T_f) = \sum_{S\in S_I} C(S, T_f)$, where $C(S, T_f)$ denotes the cost of source $S \in S_I$ for T_f. We define the problem of time-aware source selection as follows:

DEFINITION 3. (TIME AWARE SOURCE SELECTION) *Let $\bar{\mathbf{S}}$ be a set of sources, F be an integration model, and β_c be a budget on cost. Let T_f be a set of time points of interest. The Time-Aware Source Selection problem finds a subset $S_I \subseteq \bar{\mathbf{S}}$ that maximizes $G_F(S_I, T_f) - C_F(S_I, T_f)$ under the constraint $C_F(S_I, T_f) \le \beta_c$.*

Generalizing the analysis by Dong et al. [3] one can easily show that time-aware source selection is NP-complete. Next, we introduce two variations of the basic time-aware source selection.

Varying update frequencies: Instead of acquiring every source update, we consider acquiring updates at slower frequencies to reduce cost (Example 4). In this version of the time-aware source selection problem, we wish to select both the subset of sources that maximizes the integration profit and their *optimal frequencies* with which updates should be acquired. Given a set of selected sources S_I and their selected frequencies f_{S_I} let $G_F(S_I, f_{S_I}, T_f)$ denote the integration gain of S_I, under model F, with the frequencies specified in f_{S_I} for T_F, and $C_F(S_I, f_{S_I}, T_f)$ denote the corresponding integration cost. We have the following definition:



Pre-processing
Source Selection
Historical Snapshots of Available Sources
Use Change Models to Estimate Integration Quality
Integration Cost Model
Statistical Modeling
Maximize Quality - Cost Tradeoff
Change Models for Sources
Change Models the Data Domain
Select Optimal Subset of Sources

Figure 3: Framework overview.

DEFINITION 4. (VARYING FREQUENCY SOURCE SELECTION) *Let $\bar{\mathbf{S}}$ be a set of sources with variable update frequencies, F be an integration model, and β_c be a budget on cost. Let T_f be a set of time points of interest. The Varying Frequency Source Selection problem finds a subset $S_I \subseteq \bar{\mathbf{S}}$ and their corresponding update frequencies f_{S_I} that maximize $G_F(S_I, f_{S_I}, T_f) - C_F(S_I, f_{S_I}, T_f)$ under the constraint $C_F(S_I, f_{S_I}, T_f) \le \beta_c$.*

Integrating slices of data: Often the data domain Ω can be high-dimensional, characterized by a set of discrete dimensions $\mathbf{D}_\Omega$, and sources may exhibit significant differences in the types of data they cover. Instead of acquiring all the entities from a source, we consider acquiring only a subset (i.e., *a slice*) to reduce the cost.

To illustrate, we consider the business listings and two dimensions: (a) the location of the listing and (b) the category of business (e.g., restaurants in New York). Figure 2 shows three data sources: (1) one providing entities for most location-category pairs, (2) one providing entities for a specific set of locations but across all categories, and (3) one providing entities for a specific set of categories but across all locations. A user focusing on certain locations may consider acquiring the second source and small parts of the first source to increase the overall coverage at a reduced cost.

In such cases, sources can be viewed as aggregates of multiple micro-sources, i.e., elemental sources focusing on certain slices of the data domain. The basic definition of time-aware source selection can be extended to account for this case as follows:

DEFINITION 5. (SLICE TIME AWARE SOURCE SELECTION) *Let $\bar{\mathbf{S}}_m$ be a set of micro-sources corresponding to slices obtained from a set $\bar{\mathbf{S}}$ of data sources, F be an integration model, and β_c be a budget on cost. Let T_f be a set of time points of interest. The Slice Time-Aware Source Selection problem finds a subset $S_I \subseteq \bar{\mathbf{S}}_m$ that maximizes $G_F(S_I, T_f) - C_F(S_I, T_f)$ under the constraint $C_F(S_I, T_f) \le \beta_c$.*

The SLICE TIME AWARE SOURCE SELECTION problem can be easily extended to identify optimal update frequencies as well.

2.3 Overview and Scope of Proposed Solution

The different components of our proposed framework for solving time-aware source selection are shown in Figure 3. Assuming knowledge of data changes over a past time window T, we build a collection of statistical models that describe the update patterns in the world and the sources. We also construct a profile for each source summarizing its content at the end of T, and its effectiveness at capturing data changes (Section 4.1). Then, we estimate the quality of integration for an arbitrary set of sources using the aforementioned models and source profiles (Section 4.2), and propose a set of algorithms for solving the problem of time-aware source selection and its variations (Section 5).

We make the following assumptions in the rest of the paper:

- We assume that the majority of inaccuracies occur due to the ineffectiveness of sources at capturing data changes from the world and not erroneous insertions. We found that in both our domains stale data dominates the mistakes. For BL we found that sources exhibit delays in inserting new listings or deleting listings that disappeared from the world, while for GDELT we found that news sources present varying delays at reporting events.
- We assume an integration scheme across sources that follows the union semantics. For example, consider integrating two sources at a time point t and a restaurant listing that is mentioned in the first one but was never mentioned in the second. In this case, the restaurant entry will be present in the integration result of the two sources. On the other hand if the listing was present in the second source for a time point prior to t but deleted by time t then this entry will not be present in the integration result. This integration scheme is used in many practical applications to form the integration result [6, 17].
- We assume that the sources are independent, that is, each source updates its content independently from others. This assumption is used to prove the theoretical guarantees of the proposed algorithms in Section 5. While this assumption is strict and may not fully hold for the BL and GDELT domains, we were still able to obtain solutions of high quality for the different versions of the time-aware source selection problem (Section 6).
- We assume that the content changes *in the world* follow a Poisson random process and that the lifespan of an entity and the time interval between consecutive updates follow an exponential distribution. We observed that both assumptions hold for the BL and GDELT domains (Section 4.1.1). No such assumptions are made for the content changes of sources, for which we use generic statistical models based on empirical distributions capable of capturing complex update patterns that depend on the update frequency of each source (Section 4.1.2).
- Finally, we assume that in our integration scenarios sufficient historical data are available to learn the statistical models described above. Our approach is well suited for highly dynamic sources since they provide us with more training points and hence more accurate models can be learned.

3. QUALITY OF INTEGRATED DATA

As discussed in Section 2.2, the gain of integration $G_F(S_I, t)$ can be quantified using the quality of the integration result. We introduce time dependent versions of *coverage*, *freshness* and *accuracy* to characterize the quality of a set of dynamic sources.

We characterize the entities in a source or the integration result at a time point t using three categories: (a) *up-to-date*, denoted by Up, corresponding to entities mentioned in the source that also exist in

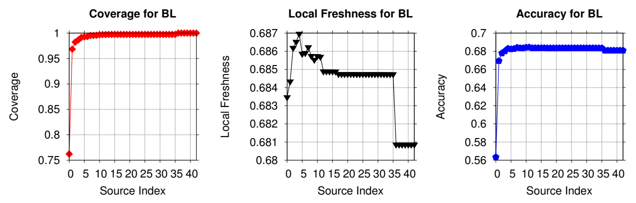


Figure 4: (a) Coverage, (b) freshness and (c) accuracy of integrated data; Sources processed in decreasing order of coverage.

the world and their attribute values in the source are in agreement with the world, (b) *out-of-date*, denoted by Out, corresponding to entities mentioned in the source that are present in the world but latest value changes are not captured by the source, and (c) *non-deleted*, denoted by NDel, corresponding to entities mentioned by the source that have disappeared from the world. We next define the quality of the integration result using these categories.

Let S_I be the selected set of sources to be integrated at time t and $F(S_I)$ the integration result using model F. We define the coverage of $F(S_I)$ at time t, denoted by $\mathsf{Cov(F(S_I), t)}$, as the probability that a random entity from the world Ω at time t belongs to $F(S_I)$. We express this probability as:

$$\mathsf{Cov(F(S_I), t)} = \frac{\mathsf{Up(F(S_I), t) + Out(F(S_I), t)}}{\mathsf{|\Omega|_t}} \quad (1)$$

where $|\Omega|_t$ denotes the total entities in the world at time t.

Next, we define a localized freshness measure for the integrated data at time t as the probability that a randomly selected entry of $F(S_I)$ is up-to-date. We refer to this metric as *local freshness*, denoted by LF, and express it as:

$$\mathsf{LF(F(S_I), t)} = \frac{\mathsf{Up(F(S_I), t)}}{\mathsf{|F(S_I)|_t}} \quad (2)$$

where $|F(S_I)|_t$ denotes the total number of entities in the integration result at time t. The coverage and local freshness are orthogonal, that is, a source with high-freshness does not necessarily exhibit high coverage. Moreover, while the coverage is expected to increase monotonically as more sources are integrated the same does not hold for freshness. We illustrate this using an example from the business listing domain introduced in Section 1.

EXAMPLE 5. *We consider integrating the available sources in decreasing order of coverage. Figure 4(a) shows the coverage of the integration result. The corresponding local freshness is shown in Figure 4(b). While the coverage increases monotonically, we see that the local freshness of the integration result decreases as more source are integrated.*□

In many cases, we wish to reason about coverage and local freshness collectively. Coverage and local freshness are similar to *recall* and *precision* in information retrieval, and hence, can be combined using an F-type measure. Thus, we introduce the *accuracy* of a source or the integration result as the *match rate accuracy* [14]. Before defining accuracy we need to define a global measure of freshness. We define *global freshness*, as the probability that a randomly selected entity from Ω at time t belongs to $F(S_I)$ and its reference is up-to-date. We have:

$$\mathsf{GF(F(S_I), t)} = \frac{\mathsf{Up(F(S_I), t)}}{\mathsf{|\Omega|_t}} \quad (3)$$

We define accuracy as the percentage of correctly matched entities, corresponding to up-to-date entities in $F(S_I)$, to all entiies in $F(S_I)$ together with entities that are present in Ω and not mentioned in $F(S_I)$. We have:

$$\mathsf{Acc}(\mathsf{F}(\mathsf{S_I}),\mathsf{t}) = \frac{\mathsf{Up}(\mathsf{F}(\mathsf{S_I}),\mathsf{t})}{|\ \mathsf{F}(\mathsf{S_I}) \cup \Omega_\mathsf{P}\ |_\mathsf{t}} \quad (4)$$

Using Equations 1, 2 and 3, one can compute accuracy by:

$$\mathsf{Acc}(\mathsf{F}(\mathsf{S_I}),\mathsf{t}) = \frac{\mathsf{GF}(\mathsf{F}(\mathsf{S_I}),\mathsf{t})}{1 - \mathsf{Cov}(\mathsf{F}(\mathsf{S_I}),\mathsf{t}) + \frac{\mathsf{GF}(\mathsf{F}(\mathsf{S_I}),\mathsf{t})}{\mathsf{LF}(\mathsf{F}(\mathsf{S_I}),\mathsf{t})}} \quad (5)$$

Figure 4(c) shows the accuracy corresponding to Example 5.

4. ESTIMATING THE QUALITY METRICS

In this section we first introduce a collection of statistical models that describe the changes in the world and the update patterns in each data source (Section 4.1). Then, we show how these can be used to estimate the quality of integrated data for an arbitrary collection of sources at a future time point (Section 4.2).

4.1 Modeling Data Changes

Estimating the quality of integrated data for future time points requires knowledge of the appearance, disappearance and value update patterns in the world and each source. Extracting the evolution patterns for the world for source-based snapshots requires solving the *history integration* problem [2, 15], that is, unifying the entity streams of the sources into a single stream describing the evolution of the world. The update patterns of sources can then be extracted by comparing the evolution of each source with that of the world.

4.1.1 Modeling Changes in the World

Given a data domain Ω, we assume that (a) entity appearances, disappearances and value changes follow a Poisson random process; and (b) the lifespan of an entity and the time interval for which it does not get updated follow an exponential distribution.

Appearances: We model the number of entity appearances $N_i(\cdot)$ during the time interval $(t, t+\tau]$ as a Poisson distribution with intensity parameter λ_i and have that:

$$\Pr[(N_i(t+\tau) - N_i(t)) = k] = \frac{e^{-\lambda_i \tau}(\lambda_i \tau)^k}{k!} \quad (6)$$

We approximate λ_i as its maximum likelihood estimate (MLE) corresponding to the average rate of data appearances in the world. To compute the latter, we divide the time window T into intervals of fixed length, and calculate the average occurrence rate of entity appearances over these intervals. Finally, we extract the starting point of the Poisson process by calculating the total number of entities in the world by the end of T.

Disappearances: The lifespan of an entity follows an exponential distribution with rate parameter γ_d, that is, the probability that the lifespan of an entity is at most τ is $F_d(\tau) = 1 - e^{-\gamma_d \tau}$. We approximate γ_d by its MLE which is equal to the inverse of the average entity lifespan observed over the time window T. Due to the fixed length of the historical time window we have incomplete observations, that is, there are entities for which we know a lower bound but not their exact lifespan since they did not disappear until the end of T. These observations are called *right censored* and the MLE of γ_d for right censored data is given by:

$$\gamma_d^{-1} = \frac{\text{total lifespan of entities}}{\text{number of disappeared entities}} \quad (7)$$

According to the superposition property of Poisson processes [5], if the appearances of entities occur based on a Poisson process and the lifespan of each entity follows an exponential distribution, the disappearances of entities should also occur based on a Poisson random process with an intensity rate λ_d. Given a time window $(t, t+\tau]$ and with $|\Omega|_x$ denoting the total number of entities in the world at time x we have that $\lambda_d = \frac{1}{\tau}\sum_{x=t}^{t+\tau} \gamma_d \cdot |\Omega|_x$. The intensity parameter λ_d can be estimated using its MLE which corresponds to the average rate of disappearance over the time window T.

Value Updates: We assume that the interval between consecutive value changes of an entity follows an exponential distribution with parameter γ_u. This parameter can be learned similarly to entity disappearance. Moreover, one can easily show that value updates in the world occur based on a Poisson random process with intensity parameter λ_u, following the same process presented above.

Discussion: We presented the aforementioned modeling considering the entire data domain Ω for ease of exposition. However, these techniques are directly generalizable to heterogeneous data domains where different subdomains $\Omega_{<i>} \subseteq \Omega$ exhibit different change patterns, such as the business listing and GDELT domains presented in Section 1. In the case of heterogeneous data domains, we learn a collection of separate models for different homogenous data subdomains. This enables capturing non-uniform update patterns commonly observed in real-world domains. The details are omitted due to space limitations.

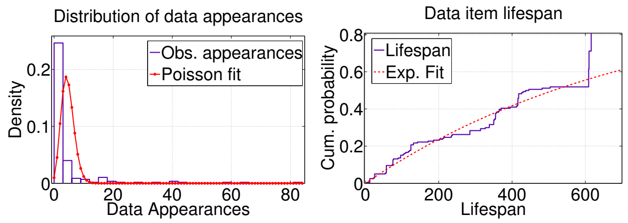


Figure 5: Fitting (a) a poisson distribution to the appearances of data items per time point and (b) an exponential distribution to the lifespan of data items for business listings.

Finally, we show that both BL and GDELT (Section 1) fit these assumptions. We study the distribution of observed appearances per day for various domain points in BL and GDELT and observe that indeed the number of updates per day follows a Poisson distribution. Figures 5(a) and 6 show the fitted and exact distribution for a domain point in BL and GDELT respectively. We next focus on BL, and plot the observed lifespan of data entries for the same domain point as before. Figure 5(b) shows that indeed the lifespan of entities follows an exponential distribution. The observed cumulative distribution for the lifespan presents a peak after 600 days which corresponds to censored data. Similar results were observed for all points in both domains.

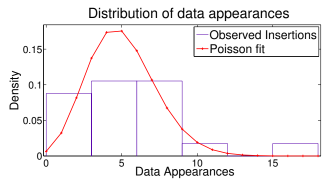


Figure 6: Fitting a poisson distribution to the appearances of data items per time point for GDELT.

4.1.2 Modeling Updates in Data Sources

The update patterns of a data source depend on its effectiveness in capturing changes from the world. We define the effectiveness of a source S in capturing an entity appearance as the probability $G_i(\tau)$ that S will incorporate this entity appearance in its content in a maximum of τ time units. Similarly, we define the probabilities G_d and G_u for entity disappearances and value changes.

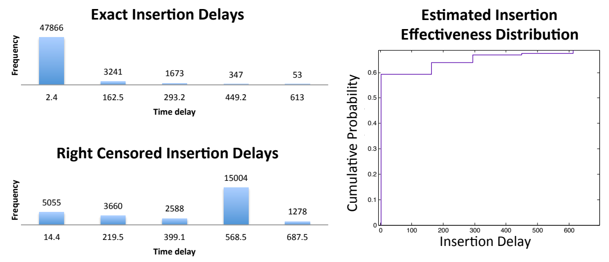


Figure 7: Exact and right censored insertion delay histograms with the effectiveness distribution G_i for a source in BL

Next, we describe how to learn these distributions. For ease of exposition we focus on G_i. The derivations of G_d and G_u are similar. We approximate G_i as the Kaplan-Meier empirical distribution [8] corresponding to the delay between the appearance of an entity in the world and its insertion in a source S. Given the evolution of source S and the world over the time-window T, one can extract two delay histograms characterizing the insertions in S: (a) one corresponding to exact observations, that is, insertions of items that appeared in the world and were also inserted in S before the end of the observed time window T, and (b) one corresponding to right-censored observation, that is, insertions of items that appeared in the world during T but were not inserted in S until the end of T. These two histograms are then combined to extract the empirical distribution G_i. To illustrate this, we consider the business listing domain. Figure 7 shows the two delay histograms corresponding to exact and right-censored observations for a source in BL, together with the learned effectiveness distribution G_i of the source.

The effectiveness distribution G_i assumes as input the duration of the time interval $t - t_c$ between a time point t and the actual occurrence of an entity appearance t_c. However, data sources get updated with a fixed frequency, and hence, the time point t may not be aligned with the latest update point of the source. We extend the effectiveness distributions G_i, G_u and G_d to account for the common case of fixed update frequencies of the sources. Again, we focus on G_i for ease of exposition. Given a source S that gets updated with a frequency f_S, we define $T_S(t)$ to be a function that returns the latest update time point of S until time t inclusive. We define $T_S(t)$ as $T_S(t) = \frac{\lfloor (t-t_0^S) f_S \rfloor}{f_S} + t_0^S$, where t_0^S denotes the last time S was updated during the historical time window T. Using this we update the definition of G_i to be:

$$G_i(t, t_c) = \begin{cases} G_i(T_S(t) - t_c) & \text{if } t \geq T_S(t) \geq t_c \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

Finally, we compute the update frequency of a source S as follows: We consider that $f_S = \frac{1}{u_S}$, where u_S denotes the average update interval of S. Let $M_S = \{t_1, t_2, \ldots, t_m\}$ be the timestamps of different content updates in S ordered by time, and let $I_S = \{t_2 - t_1, t_3 - t_2, \ldots, t_m - t_{m-1}\}$ be the set of observed time intervals for S. We compute u_S by taking the average over the elements of I_S.

Discussion: Similarly to world changes, the aforementioned techniques are directly generalizable to sources that exhibit varying effectiveness on capturing updates for different data subdomains. In this case, we learn a collection of separate models for the different homogenous subdomains to capture the complex update patterns commonly exhibited by real-world sources.

4.2 Quality Estimation

We can now estimate the coverage, freshness and accuracy for integrating a set of data sources S_I at a future time point t. The quality of the integration result for future time points relies on the content changes in the integrated data. We first describe how to estimate the content changes in $F(S_I)$ and we then present how the different quality metrics can be estimated.

4.2.1 *Content Changes Under Union Semantics*

Given a set of sources S_I we want to estimate the content of the integrated data $F(S_I)$ at a future time point t. For this, we first characterize the content of $F(S_I)$, in terms of up-to-date, out-of-date and non-deleted entities, at the end t_0 of the available historical time window T and then we examine how the content of $F(S_I)$ changes by estimating how effectively the entity appearances, disappearances and value changes occurring in the world up to time $t \succ t_0$ are captured in $F(S_I)$.

To determine the content of $F(S_I)$ for a set S_I at time t_0, we consider the up-to-date, out-of-date and non-deleted entities in each source $S \in S_I$ extracted by comparing the content of S with the actual entities in the world. The set of up-to-date entities in $F(S_I)$ is computed by taking the union of up-to-date entities across all sources in S_I. The set of out-of-date and non-deleted entities are extracted in a similar fashion. Conflicts between entities that are up-to-date in one source and out-of-date in another are resolved by considering only the reference with the most recent time-stamp.

Procedurally, we store three different signatures (bit arrays) for each source $S \in \bar{\mathbf{S}}$: (a) a signature B_S^{up} for the up-to-date items, (b) a signature B_S^{cov} for the up-to-date and out-of-date (i.e., the covered) items, and (c) a signature B_S for all the items in the source. All similar bit arrays have the same size across different sources. Using these signatures the number of entities mentioned in $F(S_I)$ is $|\bigvee_{S \in S_I} B_S|$, the number of up-to-date entities is $|\bigvee_{S \in S_I} B_S^{up}|$, and the number of covered entities is $|\bigvee_{S \in S_I} B_S^{cov}|$.

To estimate the content changes in $F(S_I)$ at time t, we need to estimate the effectiveness of S_I at capturing changes in the world. We focus on insertions of new entities. Let $\Pr(\mathsf{Ins}(F(S_I), t, \tau))$ be the probability that a data appearance at time τ was captured in $F(S_I)$ by time t. Assuming that the sources in S_i are independent, this probability corresponds to the probability of at least one source in S_I capturing the appearance of the new data item and have that:

$$\Pr(\mathsf{Ins}(F(S_I), t, \tau)) = \left(1 - \prod_{S \in S_I} \left(1 - G_i^S(T_S(t), \tau)\right)\right) \quad (9)$$

We continue with deletions in $F(S_I)$ corresponding to disappeared items from the world. Let $\Pr(\mathsf{Del}(F(S_I), t, \tau))$ be the probability that an entity disappearance at time τ was captured by $F(S_I)$ until time t. This probability corresponds to the probability that at least one of the sources in S_I that mentioned this entity at time τ captured the disappearance event until time t. According to Equation (1), the probability of an entity being mentioned in a source at a particular time point is equal to its coverage. We have:

$$\Pr(\mathsf{Del}(F(S_I), t, \tau)) = 1 - \prod_{S \in S_I} (1 - \mathsf{Cov}(\mathsf{S}, \tau) \mathsf{G}_\mathsf{d}^\mathsf{S}(\mathsf{T}_\mathsf{S}(\mathsf{t}), \tau)) \quad (10)$$

Following a similar process, we have that the probability of a value update being captured in $F(S_I)$ is given by:

$$\Pr(\mathsf{Upd}(F(S_I), t, \tau)) = 1 - \prod_{S \in S_I} (1 - \mathsf{Cov}(\mathsf{S}, \tau) \mathsf{G}_\mathsf{u}^\mathsf{S}(\mathsf{T}_\mathsf{S}(\mathsf{t}), \tau)) \quad (11)$$

4.2.2 *Quality Estimation at Future Time Points*

We now describe how we estimate the coverage and freshness of the integrated data for a future time point $t \succ t_0$. The accuracy can be derived using Equation (5).

Coverage: Let $\mathbb{E}[|\Omega|_t]$ be the expected number of entities in the world at time t, $\mathbb{E}[\mathsf{Ins}(F(S_I),t)]$ be the expected number of entities of newly appeared entities in the world that have not been deleted until time t and were also insured in $F(S_I)$ up to time t, and $\mathbb{E}[\mathsf{OldCov}(F(S_I),t)]$ the expected number of entities that were already covered by $F(S_I)$ at time t_0 and have not disappeared from the world until time t. We can compute the coverage as follows:

$$\mathsf{Cov}^*(\mathsf{F}(\mathsf{S_I}),\mathsf{t}) = \frac{\mathbb{E}[\mathsf{OldCov}(\mathsf{F}(\mathsf{S_I}),\mathsf{t})] + \mathbb{E}[\mathsf{Ins}(\mathsf{F}(\mathsf{S_I}),\mathsf{t})]}{\mathbb{E}[|\Omega|_\mathsf{t}]} \tag{12}$$

To compute $\mathbb{E}[\mathsf{OldCov}(F(S_I),t)]$ we consider the number of covered entities in $F(S_I)$ at t_0, that is, the sum of up-to-date and out-of-date entities, and multiply that with the probability of an entity not disappearing until time t. Using the memoryless property of the Poisson process for data disappearances we have that:

$$\mathbb{E}[\mathsf{OldCov}(F(S_I),t)] = \mathsf{Cov}(\mathsf{F}(\mathsf{S_I}),\mathsf{t_0})\cdot|\Omega|_{\mathsf{t_0}}\cdot \mathsf{e}^{-\gamma_d(\mathsf{t}-\mathsf{t_0})} \tag{13}$$

where $\mathsf{Cov}(\mathsf{F}(\mathsf{S_I}),\mathsf{t_0})$ and $|\Omega|_{t_0}$ can be computed by the signatures and extracted statistics described above. Since data appearances and disappearances occur based on a Poisson process we have for $\mathbb{E}[|\Omega_P|_t]$ that:

$$\mathbb{E}[|\Omega|_t] = |\Omega|_{t_0} + \sum_{\tau=t_0}^{t}[\lambda_i - \lambda_d] \tag{14}$$

Finally, we compute $\mathbb{E}[\mathsf{Ins}(F(S_I),t)]$ using Equation (9) and the fact that data appearances follow a Poisson random process and the entity lifespan is exponentially distributed:

$$\mathbb{E}[\mathsf{Ins}(F(S_I),t)] = \sum_{\tau=t_0}^{t}\lambda_i\cdot e^{-\gamma_d(t-\tau)}\cdot \Pr(\mathsf{Ins}(F(S_I),t,\tau)) \tag{15}$$

The coverage estimator corresponds to a non-decreasing submodular function. A set function $G : 2^V \to \mathbb{R}$ mapping subsets $A \subseteq V$ into the real numbers is *submodular* [4] if for all $A \subseteq B \subseteq V$, and $v' \in V \setminus B$, it holds that $G(A\cup\{v'\})-G(A) \geq G(B\cup\{v'\})-G(B)$ (i.e., adding v' to a set A increases G no less than adding v' to a superset B of A). Function G is *nondecreasing*, if for every $A \subseteq B \subseteq V$, it holds that $G(A) \leq G(B)$.

THEOREM 1 (SUBMODULAR COVERAGE). *The coverage estimate* $\mathsf{Cov}^*(\cdot)$ *for any set* S_I *and time t is a non-decreasing submodular function.*

PROOF SKETCH. The coverage estimator (Equation (12)) is a non-decreasing submodular function as it is a non-negative linear combination of two monotonic submodular functions. The first function referring to the coverage of $F(S_I)$ at time t_0, can be shown to be non-decreasing submodular as it is derived by the set union function. The second function, corresponding to future time points, is also non-decreasing submodular as it is derived from the probability inclusion exclusion formula for independent events. □

Freshness: Let $\mathbb{E}[\mathsf{Up}(\mathsf{F}(\mathsf{S_I}),\mathsf{t})]$ be the expected number of up-to-date entities in the integration $F(S_I)$ of S_I at time t, $\mathbb{E}[|F(S_I)|_t]$ be the expected number of all entities in $F(S_I)$, and $\mathbb{E}[|\Omega|_t]$ the expected number of entities in the world at time t. We estimate the local and global freshness for S_I as:

$$\mathsf{LF}^*(\mathsf{F}(\mathsf{S_I}),\mathsf{t}) = \frac{\mathbb{E}[\mathsf{Up}(\mathsf{F}(\mathsf{S_I}),\mathsf{t})]}{\mathbb{E}[|F(S_I)|_t]} \tag{16}$$

$$\mathsf{GF}^*(\mathsf{F}(\mathsf{S_I}),\mathsf{t}) = \frac{\mathbb{E}[\mathsf{Up}(\mathsf{F}(\mathsf{S_I}),\mathsf{t})]}{\mathbb{E}[|\Omega_P|_t]} \tag{17}$$

We first show how to compute $\mathbb{E}[|F(S_I)|_t]$. We need to estimate the number of newly inserted and newly deleted entities in $F(S_I)$ until time t. Let $|F(S_I)|_{t_0}$ be the number of entities in $F(S_I)$ at t_0 computed by the signatures in Section 4.2.1. We have:

$$\mathbb{E}[|F(S_I)|_t] = |F(S_I)|_{t_0} + \mathbb{E}[\mathsf{Ins}(F(S_I),t)] - \mathbb{E}[\mathsf{Del}(F(S_I),t)] \tag{18}$$

where $\mathbb{E}[\mathsf{Ins}(F(S_I),t)]$ is as in Equation (15) and $\mathbb{E}[\mathsf{Del}(F(S_I),t)]$ denotes the expected number of deleted items from $F(S_I)$. To compute the expected number of deleted items we multiply the average number of data disappearances per time unit λ_d given by the Poisson occurrence of data disappearances with the probability that an entity disappearance was captured by the sources in S_I. The latter corresponds to $\Pr(\mathsf{Del}(F(S_I),t,\tau))$ computed in Equation (10). We have:

$$\mathbb{E}[\mathsf{Del}(F(S_I),t)] = \sum_{\tau=t_0}^{t}\lambda_d\cdot\Pr(\mathsf{Del},F(S_I),t,\tau)) \tag{19}$$

Next, we show how to compute $\mathbb{E}[\mathsf{Up}(\mathsf{F}(\mathsf{S_I}),\mathsf{t})]$. The latter can be expressed as the summation of three quantities:

- $\mathbb{E}[\text{OldUp}]$: the expected up-to-date entities already present in $F(S_I)$ that did not change in the world until time t.
- $\mathbb{E}[\text{InsUp}]$: the expected newly inserted entities in $F(S_I)$ that appeared in the world during $[t_0,t]$ and their values were not updated until t.
- $\mathbb{E}[\text{ExUp}]$: the expected entities that were present in both $F(S_I)$ and the world, their latest update was captured in $F(S_I)$, and have not disappeared from the world by t.

We have $\mathbb{E}[\mathsf{Up}(\mathsf{F}(\mathsf{S_I}),\mathsf{t})] = \mathbb{E}[\text{OldUp}] + \mathbb{E}[\text{InsUp}] + \mathbb{E}[\text{ExUp}]$.

To compute the three aforementioned quantities we first need to compute the probability of an entity not disappearing until t, denoted by $\Pr(\text{In }\Omega\text{ at }t)$ and the probability of none of its values getting updated during $[t_0,t]$, denoted by $\Pr(\text{Not Upd.},t)$. Since the lifespan and update intervals follow exponential distributions, we have that $\Pr(\text{In }\Omega\text{ at }t) = e^{-\gamma_d(t-t_0)}$ and $\Pr(\text{Not Upd.},t) = e^{-\gamma_u(t-t_0)}$. Finally, recall that the up-to-date entities in $F(S_I)$ are $\mathsf{Up}(\mathsf{F}(\mathsf{S_I}),\mathsf{t_0}) = |\bigvee_{\mathsf{S}\in\mathsf{S_I}}\mathsf{B}^{\mathsf{up}}_{\mathsf{S}}|$. According to the Poisson arrival of changes, we have:

$$\mathbb{E}[\mathsf{OldUp}] = \mathsf{Up}(\mathsf{F}(\mathsf{S_I}),\mathsf{t_0})\Pr(\text{In }world\text{ at t})\Pr(\text{Not upd., t})$$

$$\mathbb{E}[\mathsf{InsUp}] = \sum_{\tau=t_0}^{t}\lambda_i\Pr(\text{In }\Omega\text{ at }t)\Pr(\text{Not upd.},t)\Pr(\mathsf{Ins}(F(S_I),t,\tau))$$

$$\mathbb{E}[\mathsf{ExUp}] = \sum_{\tau=t_0}^{t}\lambda_u\Pr(\text{In }\Omega\text{ at }t)\Pr(\text{Not upd.},t)\Pr(\mathsf{Upd}(F(S_I),t,\tau))$$

where $\Pr(\mathsf{Ins}(F(S_I),t,\tau))$ and $\Pr(\mathsf{Upd}(F(S_I),t,\tau))$ are as in Equation (9) and Equation (11).

We can prove that the global freshness estimate is also a non-decreasing submodular function. However, the same does not hold for local freshness.

THEOREM 2 (SUBMODULAR GLOBAL FRESHNESS). *The global freshness estimate* $\mathsf{GF}^*(\cdot)$ *for any set* S_I *and time t is a non-decreasing submodular function.*

PROOF SKETCH. Similar to the previous proof sketch. □

Estimator Complexity: Given a set of time points of interest T_f, we need to estimate the quality for each $t \in T_f$ (Section 2.2). The run time complexity is $\mathcal{O}(\sum_{t\in T_f}(t-t_0)\cdot|S_I|)$, since evaluating the estimators presented above requires $\mathcal{O}((t-t_0)\cdot|S_I|)$ operations for each $t \in T_f$.

5. SELECTING FRESH SOURCES

Dong et al. [3] proved that not only source selection in the context of data fusion is NP-complete but also estimating the integration quality is #P-hard. In contrast to static source selection, the quality estimators for time-dependent metrics can be approximated efficiently under an integration model using the union semantics (Section 4.2). Moreover, we identified that the coverage and global freshness estimates are non-decreasing submodular functions. Exploiting submodularity, we next present a set of local-search algorithms for solving the different versions of time-aware source selection (Section 2.2) that come with theoretical guarantees on the quality of the solution.

Next, we focus on integration profit functions that satisfy the submodularity property. We discuss how time-aware source selection can be solved for arbitrary profit functions at the end of this section. Given a set of time points of interest T_f, the necessary conditions for a profit function to be submodular are:

- The integration gain $G_F(S_I, t)$ for each time point $t \in T_f$ has to be a non-negative linear function of either estimated coverage or global freshness of $F(S_I)$.
- The aggregate function A to compute $G_F(S_I, T_f)$ (Section 2.2) should be an average (or non-negative weighted average) since the class of submodular functions is closed under non-negative linear combinations.
- The cost function $C_F(S_I, T_f)$ has to be an additive function, so that the profit function $G_F(S_I, T_f) - C_F(S_I, T_f)$ is also submodular since the difference of a submodular and an additive function is still submodular.

Time-Aware Source Selection: We consider the basic version of time-aware source selection (Definition 3). For simplicity, we consider that no constraint is set on the budget β_c. This version corresponds to the problem of maximizing a monotone submodular function, and can be solved by a local-search algorithm introduced by Feige et al. [4] (Algorithm 1).

Algorithm 1 Submodular Maximization

```
1: Input: $\bar{\mathbf{S}}$: set of sources available; $f$: value oracle access to submodular function; $n$: cardinality of $\bar{\mathbf{S}}$;
2: Output: $S_I$: a set of selected sources;
3: Set $v \leftarrow \arg\max\{f(u) | u \in \bar{\mathbf{S}}\}$ and $S_I \leftarrow \{u\}$
4: while one of the following local operations applies do
5:     /* Addition operation on $S_I$. */
6:     if $e \in \bar{\mathbf{S}} \setminus S_I$ such that $f(S_I \cup \{e\}) > (1 + \frac{\epsilon}{n^2} f(S_I))$ then
7:         $S_I \leftarrow S_I \cup \{e\}$
8:     /* Deletion operation on $S_I$. */
9:     if $e \in S_I$ such that $f(S_I \setminus \{e\}) > (1 + \frac{\epsilon}{n^2} f(S_I))$ then
10:        $S_I \leftarrow S_I \setminus \{e\}$
11: return $\arg\max_{\bar{S} \in \{S_I, \bar{\mathbf{S}} \setminus S_I\}} (f(\bar{S}))$
```

This algorithm starts by selecting a single source that maximizes the profit (Ln.3) and then tries to increase the value of the running solution S_I either by *including* a new element in S_I or by *discarding* one of the elements of S_I until a local optimum is reached (Ln. 4 - 10). Once a local optimum is reached, the algorithm checks if the complement of the running selection improves the solution and returns the selected sources (Ln.11). The algorithm is proven to yield a constant-factor approximation of $(1 + \frac{\epsilon}{n^2})$ and is shown to use $\mathcal{O}(\frac{1}{\epsilon} n^3 logn)$ oracle calls [4].

Varying Frequency Source Selection: Selecting the optimal set of sources and their corresponding frequencies can be expressed as an optimization problem with a unified objective function. Let $\bar{\mathbf{S}}$ be the set of available sources. For each source $S_i \in \bar{\mathbf{S}}$ we can select a variable update frequency $f'_{S_i} = \frac{f_{S_i}}{l_i}, l_i \in \{1, 2, \ldots, m_i\}, m_i \in \mathbb{Z}^+$ lower than the original frequency f_{S_i} of the source. We define the *augmented set of available sources* defined as $\mathbf{S_{aug}} = \{S_1^1, S_1^2, \ldots, S_1^{m_1}, \ldots, S_i^1, \ldots, S_i^{m_i}, \ldots S_k^{m_k}\}$ where S_i^j denotes a version of source S_i with an update frequency of $\frac{f_{S_i}}{j}$. We can now select sources from $\mathbf{S_{aug}}$ instead of $\bar{\mathbf{S}}$ - each entry of $\mathbf{S_{aug}}$ can be considered as a different source - under the constraint that only one of the $[l_i]$ versions of an actual source S_i will be selected for integration.

The submodular objective is now defined over the ground set $\mathbf{S_{aug}}$ and the frequency constraints can be expressed as a *uniform matroid constraint*. A uniform matroid U_n^r is defined over a set of n elements, and a subset of the elements is independent if and only if it contains at most r elements. Thus, each of the k constraints corresponds to a uniform matroid constraint of rank 1. Every uniform matroid is also a *partition matroid*. The varying frequency time-aware source selection corresponds to the problem of maximizing a monotone submodular function under a fixed number of partition matroid constraints, and can be solved by an algorithm that yields a constant-factor approximation of $\frac{1}{k+\epsilon}$ [9].

Algorithm 2 Submodular Maximization with Matroid Constraints

```
1: Input: $\mathbf{S_{aug}}$: ground set of sources; $k$: number of matroid constraints;
2: Output: $S_{opt}$: a set of selected sources;
3: Set $V_1 = \mathbf{S_{aug}}$
4: for $i = 1, \cdots, k+1$ do
5:     Apply the approximate local search procedure $A$ on a ground set $V_i$ to obtain a solution $S_i \subseteq V_i$ corresponding to the problem:
           $$max\{f(S) : S \in \cap_{j=1}^k \mathcal{I}_j, S \subseteq V_i\}$$
6:     Set $V_{i+1} = V_i \setminus S_i$
7: return $S_{opt} \leftarrow max\{f(S_1), \cdots, f(S_{k+1})\}$
```

The algorithm is shown in Algorithm 2. The independent sets defined by the matroid constraints divide the ground set of sources in multiple partitions, each corresponding to the intersection of a combination of independent sets from all constraints. The algorithm identifies $k+1$ disjoint partitions for which the optimization objective is locally maximized and returns the partition with the highest objective value. In fact, the algorithm performs $k+1$ iterations (Ln. 4) and at each iteration i uses a local-search procedure (Ln. 5) similar to the one used in the basic version (Algorithm 3) to select an approximately optimal set of sources over a subset V_i of the available sources. Each of the sets V_i corresponds to the union of a subset of the aforementioned partitions. After each iteration the set of available sources for the next iteration is restricted to sources that were not previously selected (Ln. 6). Finally, the algorithm returns the partition, i.e., a subset of the data sources, with the highest objective value (Ln. 7).

The local search procedure is given a set of available data sources and greedily selects a set of available sources that maximizes the optimization objective under the given constraints. The algorithm detects a single source that yields the highest objective value (Ln. 4) and proceeds by searching the neighborhood of the running solution for solutions that improve the objective. The local neighborhood of the running solution is constructed either by removing a source from the solution (Ln. 5-7) or by exchanging a set of selected sources with a new source such that all the constraints are satisfied (Ln. 8-10). The local search procedure iterates until a local optimum is retrieved. The running time of Algorithm 3 is $\frac{1}{\epsilon} n^{\mathcal{O}(k)}$ with $n = |\mathbf{S_{aug}}|$ and k is the number of matroid constraints, and thus the running time of Algorithm 2 is $\mathcal{O}((k+1)\frac{1}{\epsilon} n^{\mathcal{O}(k)})$ [9].

Algorithm 3 Local Search Procedure

```
1: Input: $X$: ground set of sources; $f$: value oracle access to submodular function; $n$: cardinality of $\mathbf{S_{aug}}$;
2: Output: $S_I$: a set of selected sources;
3: Set $v \leftarrow \arg\max\{f(u) | u \in X\}$ and $S_I \leftarrow \{u\}$
4: while one of the following local operations applies do
5:     /* Delete operation on $S_I$. */
6:     if $e \in S_I$ such that $f(S_I \setminus \{e\}) > (1 + \frac{\epsilon}{n^4} f(S_I))$ then
7:         $S_I \leftarrow S_I \setminus \{e\}$
8:     /* Exchange operation on $S_I$. */
9:     if $d \in X \setminus S_I$ and $e_i \in S_I \cup \{\emptyset\}$ (for $1 \leq i \leq k$) are such that $(S_I \setminus \{e_i\}) \cup \{d\} \in \mathcal{I}_i$ for all $i \in [k]$ and $f((S_I \setminus \{e_1, \cdots e_k\}) \cup \{d\}) \geq (1 + \frac{\epsilon}{n^4}) f(S_I)$ then
10:        $S_I \leftarrow (S_I \setminus \{e_1, \cdots, e_k\}) \cup \{d\}$
11: return $S_I$
```

Slice Time-Aware Source Selection: The basic submodular optimization problem of time-aware source selection can be trivially extended to account for this case by including all the micro-sources in $\bar{\mathbf{S}}$. The set of available sources can also be replaced by its corresponding augmented set to account for variable update frequencies of the micro-sources.

Generic Profit Functions: When the profit function used to quantify the integration profit is not submodular (e.g., when the gain is quantified using the accuracy or local freshness of $F(S_I)$), we use the GRASP heuristic introduced by Dong et al. [3] to solve time-aware source selection. We point out that GRASP needs to be extended only when solving the varying frequency source selection problem to account for the frequency constraints presented above. We refer the reader to Dong et al. [3] for details on the algorithm.

6. EXPERIMENTS

We present an empirical evaluation of the proposed framework. The main questions we seek to address are: (1) how accurately can the proposed models predict the data changes in the world, and how effective they are at estimating the quality of data sources at future time points, (2) how the different source selection algorithms perform under different families of gain and cost functions, and (3) how well do the proposed algorithms scale. We empirically study these questions on both real-world and synthetic datasets.

6.1 Experimental Setup

Data: The first dataset we consider is the business listing (BL) dataset introduced in Section 1 containing daily snapshots from 43 data sources providing business listings over a period of 23 months. Each data entry includes the source-id, a description of the business (i.e., phone, address, category) and the timestamp of the last insertion or update operation performed on it. We assign a deletion timestamp to an entry by considering the timestamp of the latest snapshot mentioning it. If that timestamp corrsponds to the end of the observed time window, we assume the entry was not deleted.

To extract the evolution of the world we first detected duplicates across the source snapshots using standard canonicalization and format standardization techniques together with an exact matching algorithm, and then applied an integration scheme following the union semantics describe earlier. The output was verified against a gold standard provided with BL containing a subset of businesses. The sources provide 84,791,789 listings for 28,094,382 distinct businesses over 51 locations (i.e., states including Washington, DC) corresponding to 1496 business types. Figure 8(a) shows the different types of sources contained in the dataset.

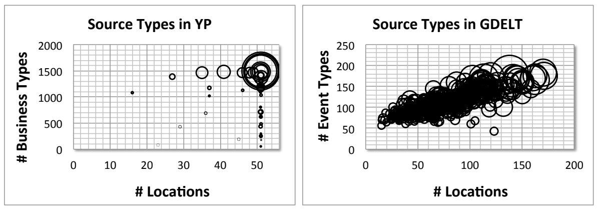


Figure 8: Different source types (a) in BL and (b) GDELT. The radius of each circle is proportional to the size of the source.

The second dataset we use is GDELT (Section 1). GDELT contains daily snapshots of events extracted from articles published in 15,275 news sources over a period of 22 days. All entries contain information about the source reporting the event, and characteristics such as the actors associated with the event, the location and the type of the event. We extract the evolution of the world using similar techniques as for BL. In total the sources provide 2,833,755 entries for 2,219,704 distinct events corresponding to 242 different locations and 236 different event types. Figure 8(b) shows the different types of the 500 largest sources in the dataset.

Finally, we use a collection of synthetically generated datasets BL+ , using BL as a seed, to evaluate the scalability of the proposed algorithms. We decompose the sources in BL into multiple overlapping micro-sources, where each micro-source covers a randomly selected subset of the initial source. If $|L|$ denotes the locations in a source S, we construct each micro-source to contain all the entities from S belonging to a randomly selected subset of locations from the original source. The number of locations in each micro-source is chosen uniformly at random from a uniform distribution $\mathcal{U}(0.2 \cdot |L|, 0.5 \cdot |L|)$. We vary the total number of micro-sources to be in $\{0, 1, 2, 5, 10, 20, 50, 100, 200\}$ obtaining 9 different datasets ranging from 43 to 8643 sources.

Preprocessing: We train the statistical models that describe the changes in BL using the data corresponding to the first 10 months, and evaluate our techniques on the next 13 months. For GDELT, we train the change models using the first 15 days and use the next 7 days for evaluation.

Algorithms: We evaluate the following algorithms:

- Greedy: The same as the greedy algorithm used by Dong et al. [3]. Starting from an empty selection set the algorithm iteratively selects the source that maximizes the integration profit until it reaches a local optimum.
- MaxSub: Depending on the version of time-aware source selection MaxSub corresponds to the submodular optimization algorithms in Section 5.
- GRASP: The GRASP algorithm proposed by Dong et al. [3] for different configurations of (κ, r).

All algorithms are implemented in Java and the evaluation is performed on an Intel(R) Core(TM) i5 2.3 GHz/64bit/8GB machine.

Gain-Cost Models: We consider two families of gain models, i.e., (1) quality driven, and (2) data driven models. For the first, let Q be the quality (i.e., coverage, freshness or accuracy) of the integrated data. We consider: LINEARGAIN assumes that the gain grows linearly with a certain data quality metric Q and sets $G(Q) = 100Q$; QUADGAIN assumes that the gain grows quadratically with Q and sets $G(Q) = 100Q^2$; STEPGAIN assumes that reaching a milestone of quality would significantly increase the gain and sets

$$G(Q) = \begin{cases} 100Q & \text{if } 0 \leq Q < 0.2 \\ 100 + 100(Q - 0.2) & \text{if } 0.2 \leq Q < 0.5 \\ 150 + 100(Q - 0.5) & \text{if } 0.5 \leq Q < 0.7 \\ 200 + 100(Q - 0.7) & \text{if } 0.7 \leq Q < 0.95 \\ 300 + 100(Q - 0.95) & \text{if } 0.95 \leq Q \leq 1.0 \end{cases}$$

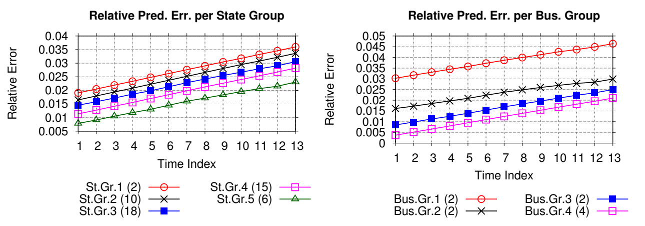


Figure 9: The relative error for predicting the total listings in BL for (a) five state groups and (b) four business category groups over 13 future time points.

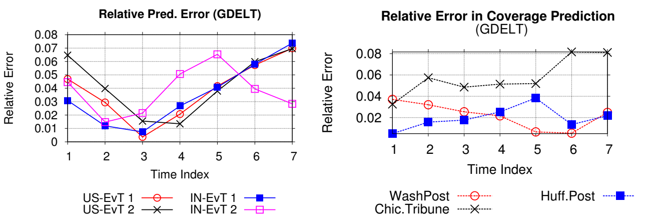


Figure 10: The relative error for predicting (a) the total events for four different event-location pairs in GDELT and (b) the quality of three large US news sources from GDELT for 7 consecutive future time points.

For the second category, denoted by DATAGAIN, we assume a gain of \$10 for each covered item in $F(S_I)$ and for a particular time point t we have $G(F(S_I), t) = 10 \cdot \mathsf{Cov}^*(\mathsf{F}(\mathsf{S_I}), \mathsf{t})|\Omega|_\mathsf{t}$.

Finally, we assume an additive cost function. Similarly to DATAGAIN, we consider that each entity has a basic cost of \$10 and an actual cost of $c = \frac{\$10}{(\text{\#sources mentioning the item})}$. The cost of a source c is the total cost of items contained in it. When considering varying frequencies for sources we assign the source cost to be $c' = c/(1 + m/10)$, where m is the frequency divisor. Finally, we rescale the gain and cost to take values in $[0, 1]$.

6.2 Verifying Model Predictions

We examine how effective our change models are at predicting the world and source changes both in BL and GDELT.

For BL, we predict the number of businesses for the 51 locations, and the ten largest business categories. We compute the relative error between the actual and predicted values. We found that states can be divided in five groups with respect to the absolute value of their prediction error. Figure 9(a) shows the relative error for the representative state of each group. The size of each group is mentioned in the legend. Similarly, the ten largest business categories can be divided in four error groups. Figure 9(b) shows the relative error for the representative business category of each group. Similar behavior was observed for the rest of the business categories. We observe that our models can accurately predict the number of listings as the average relative error is around 2%. Moreover, the increase rate of the error is 0.001.

For GDELT, we predict the number of events for four event-location pairs over 7 days in the future and show the result in Figure 10(a). We observe that the prediction error is relatively small, considering that the amount of training data used in GDELT spanned over a time period of only 15 days. The variation observed is due to the dynamic nature of this domain.

Next, we evaluate our models on predicting the source quality over time. Figure 11 shows the relative error for predicting the coverage, accuracy and local freshness of the two largest sources in BL for 13 months in the future. The maximum relative error is less than 1.5% for the largest source and less than 2.5% for the other source. Figure 10(b) shows the relative error corresponding to the

Table 1: Various algorithms for source selection in BL on the percentage of outputting the best selection and average and worst (reported in parenthesis) profit difference from the best selection. Notation (κ, r) denotes the best performing GRASP algorithm. We consider sources with a fixed update frequency.

Avg. Selection Quality					
Gain	Metric	Msr.	Greedy	Maxsub	Grasp
Linear	cov.	best	16.7%	50%	100% (5, 20)
		diff.	.005 (.01)%	.001 (.007)%	-
	acc.	best	0%	33.3%	83.3% (2, 100)
		diff.	9.5 (53.7)%	.39 (2.31)%	8.9 (53.7)%
Quad.	cov.	best	33.3%	66.7%	100% (10, 100)
		diff.	.017 (.06)%	.012 (.06)%	-
	acc.	best	100%	100%	100% (1,1)
		diff.	-	-	-
Step	cov.	best	50.0%	66.7%	83.3% (10, 100)
		diff.	7.45 (27.8)%	1.76 (10.6)%	.7 (4.2)%
	acc.	best	50%	66.7%	83.3% (5,100)
		diff.	6 (23.98)%	.8 (4.7)%	3.99 (23.98)%
Data	-	best	16.7%	50%	83.3% (5, 20)
		diff.	.004 (.01)%	.001 (.003)%	.002 (.007)%

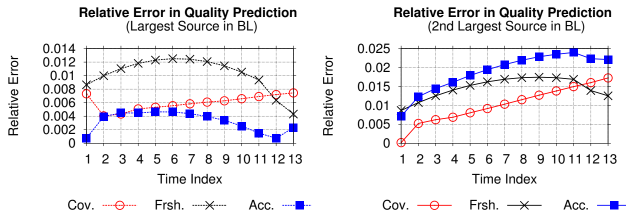


Figure 11: The relative error for predicting the quality of the two largest sources in BL for 13 consecutive future time points.

coverage of the four largest US data sources in GDELT for 7 days in the future. Again, we observe that the relative error is small.

6.3 Source Selection

Next, we evaluate the performance of the proposed source selection algorithms. We focus on two scenarios.

Fixed update frequencies. We assume a fixed update frequency for each data source and consider the basic time-aware source selection problem with a user being interested in ten future time points for six data domain points. We compute the overall gain by taking the average gain across time points. For BL, we select the six largest domain points corresponding to four business types in the states of California and New York. For GDELT, we consider six domain points corresponding to events in the United States. We compare the different algorithms for DATAGAIN, and LINEARGAIN, QUADGAIN, STEPGAIN considering the gain is quantified using coverage and accuracy for BL and coverage for GDELT.

We apply Greedy, MaxSub and GRASP with $\kappa \in \{1, 2, 5, 10\}$ and $r \in \{1, 10, 20, 100\}$ in solving the source selection problem for both datasets. GRASP with $(\kappa = 1, r = 1)$ corresponds to a hill-climbing algorithm. For each gain function, we compare the selections by the various algorithms and choose the one with the highest profit as the best. We report the percentage of times the best selection is returned by each algorithm and for sub-optimal selections we report the average and maximum, reported in parenthesis, profit difference from the best selection. For GRASP we report the (κ, r) configuration that obtained the best selection.

The results for BL are shown in Table 1. MaxSub and GRASP outperform Greedy returning solutions that result in up to 9.5% higher objective values on average and up to 53.7% in the worst case. While GRASP returns the best solution most of the times, the solutions returned by MaxSub are on average comparable to the ones obtained by GRASP with a low average profit difference. This behavior is expected since MaxSub unlike Greedy comes with rigorous theoretical guarantees, and GRASP applies a similar local search procedure to the one used by MaxSub. Nevertheless, we observe that randomization and multiple iterations help GRASP to obtain marginally better solutions. However, if we consider the run time of the algorithms (shown in Table 2), we see that MaxSub is one to two orders of magnitude faster than GRASP. Eventually, depending on the gain function, MaxSub can be a viable alternative compared to GRASP. Although, if the profit requirements are strict one should use GRASP.

Table 2: Average run times of the source selection algorithms for BL. Notation (κ, r) denotes the parameters of GRASP.

Avg. Run Time (sec)							
Gain	Metric	Greedy	Maxsub	(1,1)	(2,10)	(5,20)	(10,100)
Linear	cov.	0.05	0.16	0.16	2.23	4.35	20.13
	acc.	0.42	1.6	1.5	14.9	39.9	144.2
Quad	cov.	0.03	0.11	0.14	1.6	3.3	17.6
	acc.	0.14	0.35	0.43	5.5	11.9	57.8
Step	cov.	0.03	0.11	0.13	1.6	2.8	15.25
	acc.	0.14	0.46	0.5	6.4	14.16	74.02
Data	-	0.04	0.18	0.17	2.4	4.6	25.7

We observed similar results for GDELT. In Table 3 we report the performance and runtime of the various algorithms. We see that for LINEARGAIN and DATAGAIN both MaxSub and GRASP outperform Greedy. While GRASP never fails to detect the best solution, the profit difference between MaxSub and GRASP is very small and more importantly MaxSub is again one to two orders of magnitude faster. We omit the results for QUADGAIN and STEPGAIN since all algorithms were able to retrieve the same solution.

Table 3: Performance and runtime comparison of the source selection algorithms for GDELT, showing the percentage of outputting the best selection and average and worst (reported in parenthesis) profit difference from the best selection. Notation (κ, r) denotes the best performing GRASP algorithm.

Avg. Selection Quality				
Gain	Msr.	Greedy	Maxsub	Grasp
Linear Cov.	best	16.7%	50%	100% (10, 100)
	diff.	4.01 (13.7)%	0.5 (2)%	-
	runtime (sec)	8.58 (37)	74.12 (326)	1231.05 (4363)
Data	best	3.3%	0%	100% (10, 100)
	diff.	5.64 (14.9)%	.91 (3)%	-
	runtime (sec)	1.01 (5.03)	8.96 (44)	868.87 (4322)

Next, we report the average quality of the retrieved solution and the average number of sources selected for BL and GDELT. The results for BL are shown in Table 4. As shown, all algorithms tend to choose fewer sources when the gain is measured with respect to accuracy. We observe that all algorithms tend to select fewer large uniform sources and prefer more specialized smaller sources. Figure 12 shows the various source types selected from GRASP when the LINEARGAIN function with coverage and accuracy is used to specify the gain. A similar behavior was observed for all algorithms. Finally, Table 5 shows the results for GDELT. We see that GRASP and MaxSub were able to select significantly more sources and increase the coverage of the retrieved solution by 5% and 8%.

Table 4: Characteristics of the selected sources for various algorithms on BL for fixed source update frequencies.

Alg	Coverage		Accuracy	
	Avg. Qual.	Avg. #Srcs	Avg. Qual.	Avg. #Srcs.
Greedy	0.52	10	0.49	8
MaxSub	0.56	11	0.57	7.6
GRASP	0.56	11	0.57	8.6

Variable update frequencies. Next, we consider different versions for each source corresponding to different update frequencies. We focus on BL and take seven different versions $S_i^1 \cdots, S_i^7$ for each original source. As before, we assume that the user is interested in the same ten future time points and the same six data domains.

Table 5: Characteristics of the selected sources for various algorithms on GDELT for fixed source update frequencies.

Alg	Greedy	MaxSub	GRASP
Avg. Coverage	0.57	0.62	0.65
Avg. # Srcs	154	163	167

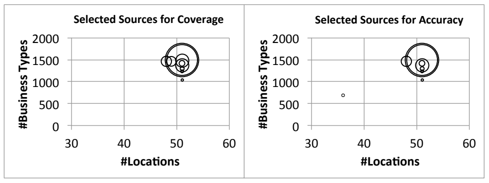


Figure 12: Selected sources when the gain is defined using coverage and accuracy. For accuracy smaller and more specialized sources are preferred.

We observe similar performance for the algorithms, as the one presented above. Namely, GRASP outperforms both Greedy and MaxSub. However, the difference in profit between the retrieved solutions is significantly smaller (less than 0.5% in average) compared to the previous case.

Focusing on the quality of the solutions returned by the various algorithms, we observe that allowing sources to have variable frequencies significantly improves the quality of the retrieved solutions. We observe that the average coverage and accuracy rise to 0.976 and 0.958 respectively, compared to 0.56 and 0.57 for the case of fixed frequencies. The reason is that by reducing the update frequency of a source the corresponding cost is reduced, and hence, the algorithms choose to integrate more sources. We report the corresponding results in Table 6. Finally, we observe that all algorithms preferred selecting large sources with a significantly reduced update frequency. However, for small specialized sources they either select the original update frequency or a small divisor of that. We report the average frequency divisors for uniform and specialized sources in Table 7.

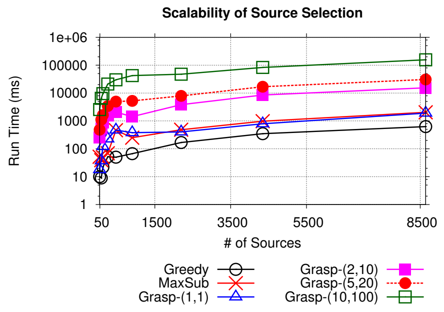
Table 6: Characteristics of the selected sources for various algorithms on BL for sources with variable update frequencies.

Alg	Coverage		Accuracy	
	Avg. Qual.	Avg. #Srcs	Avg. Qual.	Avg. #Srcs.
Greedy	0.96	15.6	0.948	14.6
MaxSub	0.976	15.6	0.958	15
GRASP	0.976	16	0.958	16

6.4 Scalability

First, we evaluate the scalability of the various algorithms as the number of available sources increases. We set the gain function to LINEARGAIN with coverage, and use the synthetically generated datasets BL+ , considering source selection for a single data point for 10 future time points. The corresponding run times are shown in Figure 13(a), where the x-axis corresponds to the number of available data sources and the y-axis (shown in log-scale) to the run time measured in milliseconds. As shown, MaxSub is one to two orders of magnitude faster compared to the best performing alternatives of GRASP, and scales better as the number of sources increases.

Finally, we use BL to examine the scalability of the various algorithms with respect to the size of the input data domain, where the size of the input domain is the number of location-business type pairs specified in a certain user query. We evaluate the performance of Greedy, MaxSub and GRASP with $(\kappa = 5, r = 20)$ for LINEARGAIN with coverage and accuracy. The corresponding run times are shown in Figure 13(b), where the x-axis corresponds to the size of the input domain and the y-axis (shown in log scale) to the run time measured in milliseconds. Again, we observe that MaxSub is an order of magnitude faster than GRASP-(5,20).



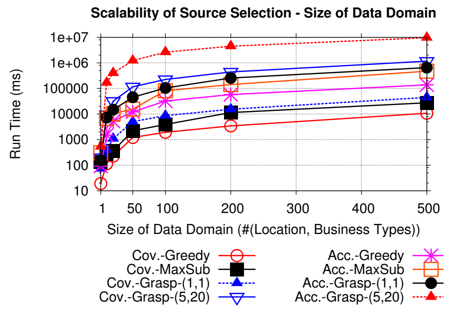


Figure 13: Run time of the various algorithms as (a) the number of sources and (b) the the size of the input domain increases.

Table 7: Average frequency divisor for uniform and specialized sources in the solution of the various algorithms.

Alg	Greedy	MaxSub	GRASP
Uniform Srcs.	4.9	5.2	4.9
Specialized Srcs.	2.6	2.9	3.2

6.5 Discussion

From our experiments we see that the proposed models can accurately estimate the evolution of the world and a source's quality. Also, the quality of the solution returned by MaxSub is comparable to GRASP. However, for large instances of source selection MaxSub is a more appealing option since it is one to two orders of magnitude faster than GRASP and presents better scalability. Finally, selecting the optimal update frequency for the selected set of sources can significantly increase the quality of integrated data.

7. RELATED WORK

Recently, Dong et. al [3] introduced the problem of source selection, and showed how one can maximize the profit of integration by optimizing the gain and cost of integrating sources jointly. However, this work focuses on static sources and is not applicable to sources whose content changes dynamically. Moreover, the proposed algorithms, while effective in practice, do not come with rigorous theoretical guarantees on their performance. In this paper, we considered the problem of source selection for dynamic sources and showed how to select a nearly-optimal set of sources and determine their optimal update frequencies by providing a set of algorithms with rigorous theoretical guarantees.

There has also been a lot of work on source relevance identification [12], but the focus is on finding sources relevant to a given query or domain and not on the quality of sources. A different line of work has considered the problem of online data integration [2, 15], however the cost of acquiring data is not considered. Moreover, a fair amount of work has considered the problem of determining the quality of multiple data sources and leveraging this information during data integration to improve the quality of the outcome [11, 13, 16, 19]. However, this work does not consider dynamic sources. Finally, Cho et al. [1], considered the problem of finding the optimal data extraction frequency from web-pages, but the authors do not reason about their quality and the cost as it is not of high importance in the web-page crawling scenario.

8. CONCLUSIONS AND FUTURE WORK

We studied the problem of source selection for dynamic sources whose content changes over time. We introduced statistical models describing the changes in the data domain and each source, and showed how to estimate the quality of integrated data at future time points. We defined the problem of time-aware source selection where we jointly select the optimal subset of sources to be integrated and their optimal update frequencies. We proposed an efficient local-search algorithm with rigorous theoretical guarantees on the quality of the retrieved solution for a large family of objective functions. Finally, experimental results show the effectiveness and scalability of our proposed framework.

One interesting research direction is to examine scenarios where new sources appear over time. It is of particular interest how sampling or online data integration techniques can be employed to derive meaningful estimates for the quality of newly appeared sources. Finally, we are actively working on relaxing some of the assumptions in the proposed framework such as source independence.

9. REFERENCES

[1] J. Cho and H. Garcia-Molina. Effective page refresh policies for web crawlers. *ACM Trans. Database Syst.*, 28(4), 2003.
[2] X. L. Dong, L. Berti-Equille, and D. Srivastava. Truth discovery and copying detection in a dynamic world. *PVLDB*, 2(1), 2009.
[3] X. L. Dong, B. Saha, and D. Srivastava. Less is more: Selecting sources wisely for integration. *PVLDB*, 6(2), 2012.
[4] U. Feige and V. S. Mirrokni. Maximizing non-monotone submodular functions. In *FOCS*, 2007.
[5] R. G. Gallager. *Discrete Stochastic Processes*. Kluwer Academic Publishers, Boston, 1996.
[6] T. Herzog, F. Scheuren, and W. Winkler. Record linkage. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2010.
[7] T. Hua, C.-T. Lu, N. Ramakrishnan, F. Chen, J. Arredondo, D. Mares, and K. Summers. Analyzing civil unrest through social media. *Computer*, 46(12):80–84, 2013.
[8] E. L. Kaplan and P. Meier. Nonparametric Estimation from Incomplete Observations. *JASA*, 53:457–481, 1958.
[9] J. Lee, V. S. Mirrokni, V. Nagarajan, and M. Sviridenko. Non-monotone submodular maximization under matroid and knapsack constraints. STOC, 2009.
[10] K. Leetaru and P. Schrodt. Gdelt: Global data on events, language, and tone, 1979-2012. *Inter. Studies Association Annual Conf.*, 2013.
[11] X. Li, X. L. Dong, K. Lyons, W. Meng, and D. Srivastava. Truth finding on the deep web: is the problem solved? *PVLDB*, 6(2), 2012.
[12] W. Meng and C. T. Yu. *Advanced Metasearch Engine Technology*. Morgan & Claypool Publishers, 2010.
[13] G. A. Mihaila, L. Raschid, and M.-E. Vidal. Using quality of data metadata for source selection and ranking. In *WebDB*, 2000.
[14] A. C. Morris, V. Maier, and P. Green. From WER and RIL to MER and WIL: improved evaluation measures for connected speech recognition. In *INTERSPEECH*, 2004.
[15] A. Pal, V. Rastogi, A. Machanavajjhala, and P. Bohannon. Information integration over time in unreliable and uncertain environments. In *WWW*, 2012.
[16] G.-J. Qi, C. C. Aggarwal, J. Han, and T. Huang. Mining collective intelligence in diverse groups. WWW, 2013.
[17] M. Stonebraker, D. Bruckner, I. Ilyas, G. Beskales, M. Cherniack, S. Zdonik, A. Pagan, and S. Xu. Data curation at scale: The data tamer system. In *CIDR'13*, 2013.
[18] K. Wilson and J. S. Brownstein. Early detection of disease outbreaks using the internet. *CMAJ*, 180(8):829–831, 2009.
[19] B. Zhao, B. I. P. Rubinstein, J. Gemmell, and J. Han. A bayesian approach to discovering truth from conflicting sources for data integration. *PVLDB*, 5(6), 2012.