SIB: Managing the Integration of Heterogeneous Biomedical Data

Research Plan

This application addresses Broad Challenge Area 06: and Specific Challenge Topic 06-RR-102*: Infrastructure for biomedical knowledge discovery.

1 The Challenge and Potential Impact

New discoveries in biology rely increasingly on the integration of data from disparate sources. For example, the Cancer Genome Atlas generates a wealth of data about several cancers, including sequence information, copy number and epigenomic changes, expression and proteomic profiling, and detailed clinical and pathological information. Integrating all of the data helps researchers gain a better understanding of the biological principles underlying the disease. The research process underlying such discovery is inherently collaborative and iterative. Yet, computer systems do not support sharing, refinement, and repetition adequately enough to keep pace with rapid increases in data volume and diversity. We propose to design, build, and integrate into ongoing biological research a computing infrastructure that provides first-class primitives for managing the integration of heterogeneous, scientific data. At the core of this infrastructure is the concept of *sibling data*: a collection comprising related data items potentially obtained from disparate sources.

Our proposed system is called SIB. SIB allows users to group data, share “live” collections with collaborators, and publish snapshots of the collection for others. Currently, these tasks are independent and cumbersome: a researcher’s work takes place on a local system or on a computing resource, collaborators share via explicit email or wiki’s, and publishers provide project web pages to make polished data and scripts available for others. These generic collaboration techniques are not sufficient for rigorous-but-fast scientific collaboration. Emailed documents can lose consistency if updated, shared file access can lead to errors and typically lack accountability, and archives rarely identify the precise script and data that together produced results.

Updates to data sets also lead to challenges and unique opportunities. Reproducing the analyses of others is inhibited by ever-changing online databases: different results could represent a failure in the software or represent a real change in outcome. Differentiating the two possibilities is a frustrating and time-consuming task—assuming the correct analysis tools are available to begin with. At the same time, growing on-line datasets can help old analyses produce new results. Perhaps those who collect new basic data can observe how new observations alter the conclusions of prior analyses; such an activity could further motivate fundamental study. We believe such analyses are unnecessarily hampered by a dearth of computing system support for repeatability and integration.

Each sibdata instance is uniquely and permanently named, and it can be stored into remotely-accessible easily-shared small-scale file systems. Sibdata instances include references to source data (which may be cached results of previous queries or a version-specific query to a compliant database), analysis scripts that operate over this source data, intermediate analysis results, detailed metadata including creation and modification times, and data provenance attributes that explicitly and securely identify how and where the data originates. The “back end” remembers how to access constituent data (or “c-data” for short) in remote databases or caches the results of queries; the “front end” provides human-readable names and access mechanisms for standard applications. In between, a sibdata server maintains the “wiring” to support live updates and snapshots, dynamic queries and translation between formats. Users may share their sibdata collections with collaborators and eventually publish them as archival.

We propose to validate the sibdata approach by integrating it with ongoing biological research. Sibdata are accessible through a filesystem interface: our prototype design incorporates a user space file system. This interface allows existing applications and scripts to be integrated into a sibdata instance and shared with others. Traditional scripts can operate over both “open” sibdata in which c-data are refreshed automatically from source databases, or over ”closed” sibdata with c-data capturing the state of a database at a particular time.

The central research of this proposal is twofold: (1) designing and building a SIB infrastructure that will allow efficient access and aggregation of sibling biomedical data, and (2) integrating SIB into the workflow of biology research, thereby leading to new discoveries. The first goal requires SIB provide location-independent access

to sibdata and its source data, an expressive and powerful security architecture that protects confidential data in insecure environments, and provenance data needed to reproducing findings and evaluate the reliability of source data. The second goal requires SIB support sibdata integration tools, multi-dimensional indexing of disparate, heterogeneous biomedical data for answering analytical and statistical queries, and tools for extracting entities and relationships from these data sources. This goal will be closely integrated with co-PI Pop research as part of the Human Microbiome Project.

This proposal is organized as follows. The remainder of this section motivates the use of sibdata in biological research (Section 1.1), describes the sibdata concept in detail (Section 1.2) and the impact of the research and the SIB infrastructure (Section 1.3). The following section describes our approach, focusing on our specific aims in order: the construction of SIB (Section 2.1), the use and development of high-level tools operating on sibdata (Section 2.2), and the integration of SIB in biological research (Section 2.3). We conclude this proposal with our timeline and a description of our team in Section 3.

1.1 Sibdata in Biological Research: Present and Future

Modern biological research is continuously transformed by technological advances that have resulted in the ability to perform, in a high-throughput fashion, an increasingly large array of experimental assays. The integration of different types of data can enable the discovery of novel biological insights that could not be otherwise be obtained. As an example, one of the first analyses of data generated by the Cancer Genome Atlas has revealed an unexpected correlation between methylation patterns and the increased mutation rates and mutation patterns in glioblastoma samples subjected to chemo-therapy[8]. This observation only became apparent once data from high-throughput genotyping and methylation profiling experiments were combined. The need for data integration is not restricted to cancer research. It is central to modern biomedical research, and yet, the computational infrastructure for managing biological data has been slow to adapt to this paradigm. *Our proposal addresses the pressing need for a flexible and robust data management system that is specifically targeted at the heterogeneous data being generated in biomedical research*.

We illustrate how sibdata can directly benefit researchers. We start by outlining the current workflow of a typical research project, then describe how this workflow will be enhanced by the tools that will be developed in our work. We focus on a metagenomic project: an area of research in which Drs. Pop and Salzberg are actively involved as part of the Human Microbiome Project. The general concepts apply to any other area of research.

The scenario described below is inspired by a recent study of gut bacteria in lean and obese twins [14]. The primary goal of the study was to better understand the correlation between gut microbes and obesity, however additional insights were obtained regarding the structure and function of the human gut biome in general. The data generated for this study comprise directed 16S rRNA sequencing generated with two different technologies (454 and Sanger), as well as whole-metagenome random sequencing data (454). In addition, the scientists obtained additional information about the patients, including family associations (data was obtained from both twins and sometimes their mother), clinical data (BMI, medical history, etc.), and geographic information (place of birth and current address).

The Bioinformatic Discovery Process: current data management workflow. While some information is stored in relational databases (e.g., sequencing meta-data in a LIMS system), most of the data are stored and managed as a collection of files on desktop computers or on a shared storage device. Even if some of the data are available in a database, many analysis tasks require the researchers to first save the data to a local file. The following outline describes several analyses that researchers might perform as part of the study described above: (1) The sequence data are extracted from the LIMS database and analyzed using a variety of public software and resources (including web services) as well as in-house scripts. The results of the analyses are recorded as flat files and possibly aggregated into a custom file format (often tab-delimited spreadsheet) using in-house software. (2) The different types of data are integrated using existing statistical tools (Unifrac, Metastats) or through custom scripts (often written in R or Matlab). These analyses usually involve sub-setting the data in different ways and comparing specific subsets. Several comparisons might involve: lean vs. obese patients,

monzygotic vs. dizygotic twin pairs, mothers vs. children, etc. (3) The conclusions of the study are written up in a paper and some of the supporting data and methods are made available in public databases and through publications. There are several problems with this approach:

- It is difficult for researchers at different institutions to collaborate. Significant time and effort is spent converting files to conform with the specific requirements of the in-house setup at each institution. It is also often impractical to share the entire data, leaving each researcher with access to a restricted snapshot.
- Some (often, much) of the data and tools generated during the analysis process remain inaccessible to the wider research community even when the results of a study have been published. As a result, it is difficult (often, impossible) to reproduce the results of a study, or to apply a similar methodology to a new dataset.
- The public data underlying a study's results (the collection of public databases that were used in the analysis) constantly evolve, further restricting the ability of researchers to reproduce the results of a study. As an example, we recently sought to re-establish the results of a comparison of molecular subsystems between bacterial and viral metagenomes. Unfortunately, we found that the data deposited in the MG-RAST database no longer corresponded to the data reported in the original paper.
- It is difficult to ensure the security of the underlying data, which is critical in a clinical setting. Much of the analysis is conducted through flat-files, and it is possible, for instance, to inadvertently reveal personally identifiable information about patients to collaborators who should not have access to these data.

The Bioinformatic Discovery Process: the Sibdata workflow. A researcher working on the project described above would start by aggregating all the data necessary for the study by including them in a new sibdata instance. Sibdata are lightweight: they do not contain any source data. For example, the sequencing data will simply be referred to via a link to the LIMS system rather than be duplicated in the sibdata. As different analyses are performed on the data, snapshots of the resulting information, as well as provenance data (the parameters of the analysis, version information from the remote database being accessed), will be archived by defining new sibdata instances. When working with collaborators it will be sufficient to provide them with a pointer to, or a copy of, a sibdata instance, together with the chits (Section 2.1) that encapsulate relevant access rights. Similarly, a new sibdata could be created and disseminated once the study concludes. The infrastructure we describe in our proposal will ensure *durability* and *consistency* of the data aggregated within the sibdata. *Durability* ensures that the relevant c-data are available, on demand, irrespective of where it is stored. *Consistency* ensures that the same analyses over the same sibdata will yield consistent outcome: the same versions will be available. The *security* and *privacy* of the data will also be ensured. Irrespective of who receives the sibdata, only authorized users will be able to access restricted components of the sibdata.

Proposed User Interface. To enable the same type of interface that biologists are accustomed to, our system will provide users with a file-level representation of a sibdata. Specifically, a user will be able to "mount" a sibdata on a local computer and access it the same way they would access any file system. The underlying data will appear as a collection of directories and flat files within the mounted volume. The user will be able to perform similar types of analyses as they would if the data were stored in flat files, even though individual files might represent links into a remote database requiring a query to be run to retrieve the underlying data.

Throughout the remainder of this proposal we provide the details of the infrastructure we propose to build. Our work builds upon a significant computational infrastructure already available at the University of Maryland, and relies on our extensive expertise in designing and building distributed systems.

1.2 The Sibdata Concept

A sibdata allows users to treat conceptually related data (regardless of source) as a unit. For example, a researcher could construct a single aggregate that holds all of the data supporting the conclusions of a study, the results of computational analyses of the data, and potentially any customized tools necessary to conduct specialized data analysis tasks. The SIB infrastructure provides the tools for creating these units and for managing the integration of data from the underlying sources. The sibdata primitives differ from recent research on federated

databases and data warehouses, and on the development of biomedical ontologies to facilitate cross-database and cross-datatype research. These data "integration-in-the-large" efforts are focused on the entirety of the data generated within a community, and leads to a substantial time lag between integration and discovery.

The focus of our work on sibdata is on local "integration-in-the-small": the specific datasets used by an individual researcher or a small group of collaborators. SIB differs from the various tools used by scientists to simultaneously explore heterogeneous data-sets, e.g., as done through custom tracks in the UCSC Genome Browser. These tools are restricted to pre-established data types (sequences and ranges along a sequence in the case of the Genome Browser). Likewise, the grouping operator of relational databases requires data schemas to be aligned. Sibdata will allow the flexible aggregation of any type of data, including unanticipated combinations of data, even when no explicit relationships have been modeled in the data of the source databases.

Data named by sibdata may represent local text files, images, scripts, or even the output of queries against remote databases. Note that such c-data is only a name used by the sibdata, the actual data still resides at the source databases. Sibdata encapsulation also captures and preserves a variety of both system- and user-defined metadata (queries needed to regenerate the data, auditing/accounting information, provenance information, etc.) along with the c-data. Such metadata adds substantial value to the aggregated data both in terms of improving data management, and by providing rich information to downstream analyses.

The text so far has focused on "closed" sibdata: an aggregate that essentially defines a snapshot across heterogeneous data. However, an "open" sibdata can continuously aggregate data either via user access to data servers, or through proactive data triggers on data servers. Sibdata updates can either replace or append to old data, and open sibdata may be created either from scratch or by "opening" pre-existing closed sibdata.

Furthermore, a user may create a "logical" sib data item which is the output of a specific query on a particular database. Obviously, one (easy) way to support such an item is for the user to execute the query and include the output as a sib data item. However, in SIB, we allow users to embed the query (and its associated metadata that is required to re-execute the query) within the sibdata. This approach is much more efficient (since it does not require data movement) and also allows for sibdata to refer to c-data that can be refreshed as the source databases change.

Sibdata Properties Sibdata are lightweight, location-independent, isolated, durable, and secure. Sibdata are lightweight in that they name data rather than containing it, and are location-independent because such names consist of global-unique IDs (GUIDs). Sibdata are isolated in that they merely contain descriptions of data; they do not modify the data and therefore can not affect each other. When a sibdata is deleted, the sources where the data was obtained are unaffected.

Sibdata are durable in that both the sibdata descriptions, as well as any data named by the sibdata, are permanent to a user-defined degree. This may be guaranteed by the data sources themselves or by the Sib System through caching. Sibdata are secure in that all remote access is through system-defined mediators that enforce security guarantees defined by data creators. Access rights may be granted or delegated to users, groups, or holders of specific public keys. Remote access to c-data sources is therefore only possible to entities explicitly given access. Secure provenance information can allow aggregation of public data of varying reliability along with group or privately owned data, without compromising either the privacy of the data nor the value added by such aggregations.

1.3 Impact

High-throughput experimental techniques such as DNA sequencing, microarrays, mass-spectrometry, and confocal microscopy are central to modern biological research, and data generated by such technologies are rapidly accumulating in biological databases. This information provides new opportunities for scientific discovery through the integration of data from multiple heterogeneous sources. It is, therefore, not surprising that considerable research efforts have been devoted to data integration. These efforts have primarily focused on two aspects of the integration process: (i) low-level logistical issues: controlled-vocabulary biomedical ontologies, federated databases and data warehouses, distributed access protocols and APIs (e.g., CAGrid); and (ii) high-level analysis

tools: statistical methods for data mining from collections of heterogeneous data-sets. As a result, scientists have easy access to a wealth of data stored in biomedical databases as well as to an increasingly large collection of analysis tools. What is missing, however, is a way to manage the specific data-sets (often just sub-sets of larger collections stored in remote databases) used in day-to-day research activities. While computation plays a critical role in data collection, storage, and analysis, the management of data is still largely performed "by hand" through lab notebooks and README files. **The main contribution of our proposal is a computational framework for managing the collections of data-sets and analysis tools used during the biomedical discovery process.**

The infrastructure we proposed is designed to be easily integrated in current analysis pipelines, while providing researchers with several important benefits. First, the data aggregated within a sibdata container can be easily tracked and shared among researchers (within a lab and across institutions). The resultant research process will be more efficient: scientists will no longer need to spend their effort tracking all the data needed for an analysis, ensuring its integrity, or worrying about data transfers. Second, our system enables reproducibility across time and space: a closed sibdata provides a snapshot of the underlying data even if the databases where these data reside are constantly being updated. This seemingly simple primitive will ensure reproduceable results within the same lab and when results are shared across institutions. Third, our infrastructure provides a consistent and robust mechanism for managing data security (imperative when handling clinical data). The security system we provide can ensure that certain data are only accessible to authorized parties, even when the sibdata object containing the protected data is no longer under the direct control of its creator. This guarantee cannot be achieved in the current manual approach for data management.

We expect that the SIB system will have a significant impact on future data-intensive biological research. Our system will allow heterogeneous collections of data and tools to be seamlessly shared among scientists thereby providing new opportunities for collaborative research, potentially accelerating the discovery process. SIB also provides the mechanism for the efficient and broad dissemination of data and scientific results thereby increasing the ability of scientists to build upon prior research. In the short term, our research will directly impact current research activities conducted under the Human Microbiome Project by co-investigators Pop and Salzberg.

2 The Approach

2.1 Building SIB (Specific Aim 1)

We propose to build an entire SIB infrastructure and demonstrate its use in biological problems. In this section, we describe the underlying SIB architecture, followed by a description of individual SIB components.

2.1.1 Key Concepts

The SIB architecture consists of three components:

SIB servers: SIB servers provide the mapping between a sibdata (an aggregate) and its constituent "siblings" (individual data items). When a new sibdata is created, a SIB server assigns it a GUID (a globally unique ID), which can later be used to access the sibdata. Any user with the knowledge of this GUID and with the proper access rights may access this sibdata without the overhead of setting up user accounts. Access to a sibdata only grants access to the mapping between the sibdata and its c-data. Further credentials may be necessary to access the individual c-data.

Security Architecture: Sibdata are protected using "chits". Chits are distributed capabilities that allow the bearer to access a named sibdata. SIB servers provide sibdata creators with a chit with unlimited access to the newly created sibdata. The creator can then locally "derive" weaker chits —e.g., ones that allow to subsets of the sibdata— and share these with others.

Chits (may) embed a public key, which the server uses to create a challenge when the chit is used. The SIB server allows the bearer of a chit access to the sibdata if she can answer the challenge (which means they hold the corresponding private key). The level of access is limited to what is specified in the chit.

SIB data stores: The underlying data referred to by a sibdata is stored in a SIB data store. This data may be completely public (e.g., a public genome database) or may require specific chits for access (e.g., a clinical

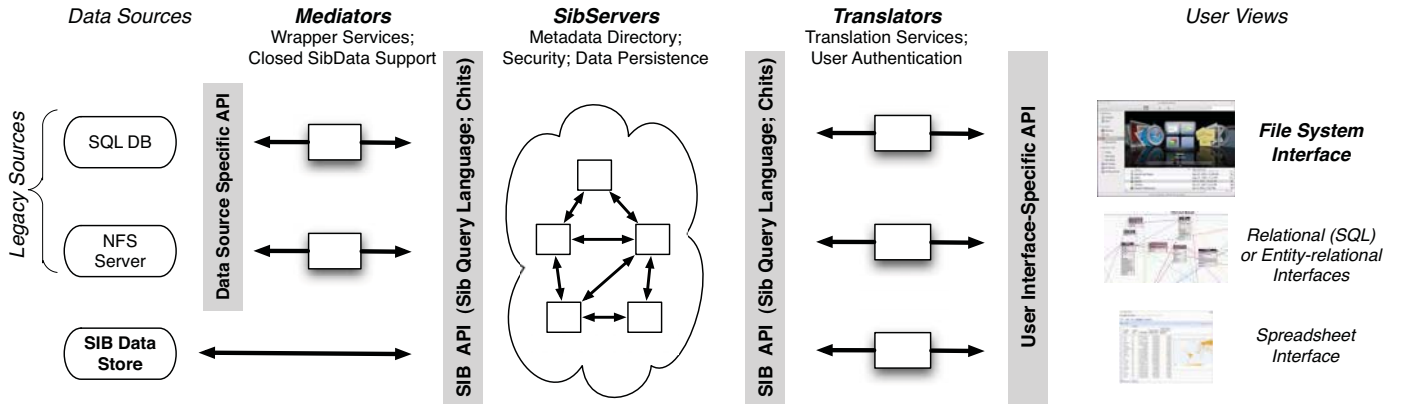

Figure 1: Sib System Architecture

database).

We further make a distinction between *legacy* data sources, and *sib-compliant* data sources. Sib-compliant data sources support the chit-based access mechanism natively, and further agree to maintain copies of data named in sibdata until notified otherwise by the data's owner. However, data sources in current use do not natively support chit based access, and indeed, are likely to implement different access methods. To support such legacy sources, we use source-specific *mediators* that provide a chit-based unified interface, and also provide external data storage to support closed sibdata.

In the rest of this section, we describe these architectural components in more detail. We end with a specific user-interface that we propose to implement. A SIB server merely enables access to a sibdata, but does not specify how scientists can analyze or otherwise interact with this data. We describe a filesystem that we propose to build that will present file-based access to remote data. This is the interface that scientists currently use, and will allow them to continue using existing analysis tools and packages, while still benefit from the aggregation and sharing abstractions provided by SIB. Other interfaces, including web-based interfaces, can also be within SIB.

2.1.2 The SIB Infrastructure

The SIB architecture is shown in Figure 1. Data comes into the system from a variety of data sources such as SQL databases, NFS servers etc. A source may be encapsulated by a source- or type-specific *mediator*, which serves as the interface between the data source and a SibServer (there may be many). The SibServers manage sibdata and serve it to user-defined *translators*, which massage the named data into a form suitable for the chosen *view*, such as the default file system view.

SibServers The main function of SibServers is to serve as a *directory*, storing all metadata about sibdata, including creation time, creator, provenance data, and access rights to it. Users can only see the subset of the directory to which they have rights, as defined by the set of *chits* held by them.

SibServers will also support a high level querying API, called Sib Query Language (SibQL), for searching and updating the directory. SibQL supports keyword search, and a simplified subset of SQL for querying the metadata contained with the sibdata. SibQL also supports commands for chit-based access control, i.e., for granting or revoking access to sibdata.

The SibServer manages individual *sibdata* instances, each of which potentially names an aggregate of data from many disparate sources. Each of the constituent data, or c-data, is named by a globally unique ID (GUID). The sibdata not only defines a set of c-data, but it may also codify more elaborate relationships between the c-data, such as hierarchies. Finally, note that sibdata c-data may be pointers to single instances of data, or they may name other sibdata.

Mediators Mediators provide *wrapping services* to handle legacy sources by (1) marshaling the data generated by a data source, and (2) by translating SibServer requests into commands that the data source can understand. Mediators are typically source type-specific. For example, a mediator for a relational database may convert the results of SQL queries (issued by a SibServer) over the database into XML, which is then passed back to the SibServer. A different mediator on a relational database, on the other hand, may pass the result table back in a tabular format. The user interaction interface will naturally determine the appropriate mediator to be used for a data source.

The mediator is also able to cache *snapshots* of the source data (the "deep content" named by the sibdata) if the underlying data source is not able to guarantee that the current data will be accessible in the future (e.g., it might get modified or overwritten). This functionality is necessary to support *closed sibdata* (see below).

Translators Data from the SibServer is then passed through user-defined *translators* to different types of *views*, such as file system views, SQL views, etc. In the case of a file system view, genome researchers might expect to see the data from each of the disparate sources represented by a hierarchy of flat files.

A translator is a user-defined program that maps data from a source representation to one consistent with the end view. For example, a translator might consist of a short Perl script that interprets each individual c-data as a distinct file, and the data read from each c-data as the contents of the corresponding files. Sub-directories might be created as implied by other relationship information in the sibdata structure.

Translators might also customize their presentations according to system- and user-provided provenance metadata.

Views Users may choose any of several different types of views with which to access the data named by a sibdata. However, the *file system view* will be the default, as it provides an extremely general interface that can be accessed by any program that can read files, including most legacy code [4]. As a simple example of the use of a file-system view, mounting a sibdata that names two web pages and the output of a specific database query might result in a directory containing three files, one for each c-data. Individual research groups also often have custom tools with which to process local data. File system views enable the use of these local tools with arbitrary remote data sources.

File system translators might be generic enough to work for any datasource, but it is more likely that such translators will be specific to individual types of data, or even to specific projects. Translators have the ability to perform arbitrary transformation over both the data and the meta-data. For example, a translator might be specialized so as to create multiple files representing a single source c-data, each representing a distinct aspect of that data. Such transformations might be guided by the metadata accompanying the datastream.

Mounted sibdata are read-only, but the file system view could also be used to create new sibdata aggregations. For example, assume a directory `foo/`, which contains `bar.c` and `etc/`, with `etc/` being the mount of sibdata A. A user might wish to create a new closed sibdata corresponding to an instantaneous snapshot of `/foo`, calling it B. B will consist of GUIDs for each of the constituent c-data, in this case `bar.c` and `etc/`. The GUID for the latter is just A's GUID, but the local file `bar.c` does not have a GUID. The snapshot mechanism would either negotiate a new GUID with the local file server, if the server understands SIB protocols, or the current data would be copied to (*instantiated* at) a mediator that would encapsulate the local server. The mediator would then assign the c-data a new GUID. Finally, the snapshot command would return a GUID that references the new sibdata on a local SibServer.

Dealing With Open Sibdata While "closed" sibdata refer to static snapshots, "open" sibdata are dynamic. In a file system view, files mounted from an open sibdata appear static to a user between `open()` and `close()` calls, but `read()`'s after subsequent `open()`'s might see different versions of the data.

Individual translators might interpret continuous datastreams as either "replacements" or "appends". In the former case, a c-data might represent a datastream generated from a query run periodically against a database. Each time the query is run, new data is pushed through the system and "overwrites" the old version of the file

corresponding to the c-data.

In the case of an append, each new pulse of data is merely appended to the end of the corresponding file, much like normal file system logs. Such a file would behave and perform much like a uni-directional socket.

Provenance Data provenance consists of the data's antecedents, i.e., which user and/or data sources were used to create it. A given c-data provenance could also include the set of other data and rules used in whatever computation created the c-data.

Sibdata are annotated by metadata consisting of $\{\texttt{tag},\texttt{value}\}$ tuples. The system requires a small subset of tuples (creator's public key, various timestamps, etc.), but applications may define other tuples with arbitrary tags.

Translators will make this provenance data available through views. In the file system view, for example, such data can be expressed either through distinguished files or subdirectories in the sibdata mount, or though special `ioctl()` calls. Translators could also run application-specific provenance computations, similar to decentralized trust computations [7, 6].

2.1.3 Security Architecture

SIB controls access to sibdata and to constituent data sources by using *chits*, which encapsulate naming and access rights in an unforgeable package. Each user holds a chit *bag*, which transparently provides chits to SibServers, and from there to data sources, as needed. Though much chit manipulation is transparent to users, users will manipulate rights to data directly through a variety of interfaces, graphical and otherwise.

We describe operations on chits using two hypothetical scientists: Watson and Crick. Aside from revocation, all chit operations can be performed locally, perhaps by Watson as he prepares a chit to be given to Crick. The alterations to the chit that we describe are not destructive, i.e., the previous chit remains and a new chit with more restricted properties is created. The new chit includes a modified chit secret which ensures that the new chit cannot be used to gain the level of access of its parent.

- *Label:* Watson can embed indelible "labels" that will be logged by the server when a chit is used. These log entries can be read to audit accesses, establish provenance, and, if necessary, to identify chits to be revoked.
- *Narrow:* Watson can embed restrictions on the rights permitted by a chit; these restrictions may limit the resources accessible or the operations permitted.
- *Expire:* Watson may set the chit to become invalid at an earlier time.
- *Delegate:* Watson can embed a transfer of rights to Crick, so that the server will authenticate that it is Crick, not Watson, who presents the chit for access. Not all chits require authentication, but to transfer those that do requires embedding the consent of the chit holder.

These four operations can be combined: Watson may *narrow* the rights of a chit to permit only read-only access to an object he owns, *label* the chit as "Crick's" to discourage abuse, mark the chit to *expire* in one week, and *delegate* to allow the chit to be used by the holder of $Crick_{pub}$ instead $Watson_{pub}$. The rights to perform these operations may themselves be limited: Watson may disallow further delegation.

Sib data sources (guarded by mediators) only grant access to data when presented with appropriate chits. This guarantee, together with the expressiveness discussed above, allow myriad users and organizations to combine their confidential data without losing local control.

2.2 Tools for Sibdata Summarization and Correlation (Specific 2)

The SIB infrastructure provides the data scratchpad for discovery. Tools will be then be used to mine and correlate the data. Some of these tools are already used by biologists and they will be supported by SIB. As part of the proposed work, we plan to enrich SIB with multi-dimensional indexing and with tools for extracting concepts and relationships from the c-data of a sibdata.

2.2.1 Multi-dimensional Indices for Sibdata

Biological data produced by gene or protein chips, or by mass spectrometry detecting hundreds of thousands of gene products, proteins, and small organic molecules, is typically multi-dimensional with possibly several thousands of dimensions. Data from these observations are used to compute biological markers that are compared to data generated in clinical studies designed to answer specific clinical research or engineering questions such as drug effectiveness, and method validation. Analyses of this multi-dimensional data involves the creation of a very large number of data groupings, the application of statistical functions on each of these groupings, and comparisons between these computed statistics. These data analyses are very similar to On-Line Analytical Processing (OLAP) that has been very successfully employed in the business world. There, all possible groupings of the *dimension* attributes are formed and aggregate functions are computed on some other (*measure*) attributes. These are all stored in "data cubes" that can be retrieved at any level by "drill down" or "roll-up" operations which avoid the slow real time computation. Attempts to apply this technology on large multi-dimensional biology data have been advocated in recent years [2, 5].

We illustrate the key concepts with an example of a multi-dimensional cube built in the iProClass database [9] which contains links to over 90 biological databases, including databases for protein families, functions and pathways, interactions, structures and structural classifications, genes and genomes, ontologies, literature, and taxonomy. iProClass combines both data warehouse and hypertext navigation methods for integrating data, providing a comprehensive picture of protein properties that may lead to novel prediction and functional inference for previously uncharacterized "hypothetical" proteins and protein groups. Table 1 shows a subset of the protein sequences under the GO classification, PIR subfamilies, motifs, protein domains, etc. All these are useful dimensions for a cube and some are intrinsically hierarchical such as the GO protein function and the taxonomy. The sequences are grouped along these two dimensions and the aggregated measure attributes in the figure are "Sequence Similarity" and "Sequence Count". Using these grouping and aggregated measure values the user can decide which of these to further drill down to view the underlying data. These groupings are expensive to compute on the fly but we can pre-compute them as a cube, providing subsecond access to the group aggregate values as well as drill down queries.

However, until a few years ago, OLAP technology was limited in scaling up with the number of dimensions. OLAP data cubes would not extend beyond twenty flat dimensions and even less in presence of multi-level (hierarchical) dimensions. The number of dimensions and measurements in typical clinical studies is extremely large and can easily reach several thousands, making it prohibitive to use OLAP. However, a new technology called Dwarf Data cubes [12, 11, 13], developed at the University of Maryland, has lifted the above limitations. Dwarf cubes can scale to the requirements of biology data and beyond. The method discovers and eliminates redundancies in the groupings caused by the sparsity of the high-dimensional space. It supports arbitrary number of hierarchical dimensions in a very compressed store. The patented technology (US Patent 7,133,876) was applied to high-dimensionality scientific data including weather forecast models, astrophysics heterogeneous catalogue data, and complex event processing sets. Dwarfs offer two significant benefits. First, they index the underlying multi-dimensional data so that it can be accessed efficiently using multi-dimensional search windows. Second, they compute and capture within their structure any number of sub-groupings along with the statistical values computed on measurements for each sub-group. A Dwarf cube, for example, can compute and store aggregate measurements of several groups of people and compare these measurements for statistical significance. Another typical group comparison arises in comparing diagnostic markers for different drugs tested on carefully selected groups of people. The grouping and computation of such markers is not only efficiently computed but also persistently stored in and accessed from Dwarfs.

Dwarfs cubes are "light-weight" and can be attached to a sibdata without affecting the sibdata's transportability. Furthermore, since they capture the dimension coordinates and the computed aggregates, they in many cases carry all the information the users need for comparison, thus avoiding remote access to the data sources.

Dimensions		Measures	
1: GO	2: Taxonomy	1: SequenceSimilarity	2: SequenceCount
All Terms	All Species	23	44,411
	Viruses	35	1,634
	Cellular organisms	14	42,770
	null	108	7
obsolete molecular function	All Species	31	1,263
	Viruses	9	3
	Cellular organisms	6	1,260
	null		

Table 1: A subset of the PIR Multidimensional Data

2.2.2 Extraction of Entities and Relationships Within Sibdata

It has become increasingly more critical to develop techniques and tools for combining relevant data from heterogeneous sources and presenting them in a form comprehensible to the users. Such techniques involve identifying entities that are the same or very similar, extracting relationships amongst data objects in different structured or unstructured databases, and in general discovering knowledge from distributed, semantically heterogeneous data sources. This process can be seen as a form of *conceptual modeling*, a continuous process that may be performed concurrently with data access. As an example, in the biological domain, for discovery, the researcher has to make hypotheses for relationships that are not explicit in any of the databases but yet can be extracted using his knowledge and private experience and some extraction tools. When a discovery has been made, the knowledge obtained from it needs no further support for re-discovering it. Instead, the discovery needs to be documented and published.

Since the source databases are constantly changing, correlating, modeling, and mapping these data objects have to seamlessly be integrated and be part of biological research. The sibdata is the **data scratchpad that facilitates research and discovery of such knowledge** and the tools that we will develop in this project are aimed at **expediting the task of integrating data from the biological domain**. As part of this project, we propose to incorporate several entity and relationship extraction techniques into the sibdata framework, as well as develop new techniques specific for the biological domain. In addition, we plan to provide support for extracting and mining sibdata, so that users can create new groups and patterns, and that will hopefully allow them to better understand their sibdata, and also extract and retrieve additional information.

We will begin by incorporating support for entity extraction. Entity extraction provides means for using text analysis tools to extract from the data common entities such as people, genes, diseases, and so on. We will use existing name-entity extractors for this purpose, and also develop new techniques that exploit existing usage statistics and ontology information to extract likely entities. Once entities are extracted, users can use the same sibdata mechanism to group entities into collections. The users will have the flexibility to group entities however they like. Because the extraction process will be inherently noisy, the users can also explicitly add, extract, or remove entities. In some cases, users may wish to give semantics to the groups of entities that they construct. To enable that, we will support as primitives the notions of *same-entity* which states that a collection of entity references all refer to the same underlying entity, *same-type* which states that a collection are all of the same object type, and *composite* which states that a collection of items are all part of a larger composite entity.

Next, we will provide support for relationship extraction (also called *link prediction*). Once users have created bags of entities, they can look for relationships among the entities. Using techniques from semantic role extraction, we will use text, statistical, and ontological information to look for candidate relationships among the entities. We will develop algorithms for ranking the relationships, based on likelihood and relevance and other available features, and the user will be able to select the appropriate relationships and instantiations. A novel aspect of our proposed work in this area is that we will allow feedback between the entity extraction process and the relationship extraction process. The output of the entity and relationship extraction process will be represented as sibdata. Thus we can support an interactive and incremental refinement of the sibdata, using the above grouping mecha-

nisms. In addition, we will support an ability to generalize from one sibdata collection to another, so that if we find useful extraction patterns in one setting, we can transfer them to another collection. Our proposed techniques are based on relational clustering [1], and will support vertical and horizontal groupings in a flexible way. In addition to extraction of entities and relationships, performed at the data level, we plan to investigate extraction of metadata mappings and alignments. In some cases, the sibdata will be annotated with schema information, and we can make use of techniques from schema matching and mapping algorithms to propose potential matches among for example columns, based on column names, column types, data matches, and data distribution matches.

2.3 Integration of SIB in the biological research workflow (Specific Aim 3)

Co-investigators Pop and Salzberg are funded under the Human Microbiome Project (grant R01 HG004885) to develop software tools for metagenomic assembly and gene finding. The SIB infrastructure will benefit this work as described below.

First, the data provided to our software comprises two types of information: sequence and quality data; and meta-information about the DNA samples and/or sequencing experiment. Depending on the specific experiment each sample might comprise multiple sequencing experiments (potentially performed with different technologies) or, conversely, multiple samples might be multiplexed within a same sequencing experiment. During assembly it is often advantageous to combine multiple samples in order to improve sequence coverage for shared organisms, then separate the individual assemblies into sample-specific data-sets. In order to keep track of these various combinations of data, we will construct individual sibdata objects for each sample, combining the corresponding sequences and meta-data, then augment this object with the resulting assembly. Further, the results of the gene finding algorithms will also be added to a sample's sibdata object. Thus, we will be able to easily track the data corresponding to each individual sample throughout the analysis process. Note that having full assembly information throughout the analysis process can enhance the analysis of the data — e.g. comparative analyses can take advantage of depth-of-coverage information [15]; and gene finders can interrogate the assembly to determine whether a frame-shift could be caused by sequencing/assembly errors.

In addition, co-investigators Pop and Salzberg are involved in several collaborations with biologists conducting metagenomics projects and plan to rely on SIB to enhance the analysis of the data generated by these projects. In particular, co-PI Pop is involved in a project to study the causes of diarrhea in children from third-world countries, together with colleagues from the UM School of Medicine. This project will sample approx. 1000 sick children and matched healthy controls and will generate multiple types of information: 16S rRNA sequencing, phylogenetic microarray data, proteomic assays, as well as detection assays targeting known pathogens. The SIB infrastructure will provide a natural way for aggregating all these data throughout the analysis process.

3 Timeline and Milestones

The developments described in this proposal build upon several completed and ongoing projects at the University of Maryland. Specifically, we will take advantage of the Dwarf OLAP Multi-dimensional Indexing project, the Chit architecture for security and privacy, and D-DUP — a system for performing entity resolution and duplicate elimination. Our work will be integrated within metagenomic research efforts within the Center for Bioinformatics and Computational Biology.

Months 1-24: Build and refine the SIB infrastructure (Specific Aim 1)
Months 6-18: Develop tools for summarization and correlation and integrate with ongoing development of SIB infrastructure (Specific Aim 2)
Months 12-24: Integrate SIB into ongoing metagenomics projects (Specific Aim 3)

3.1 Team

The PI team, led by Profs. Roussopoulos and Pop, are uniquely qualified to undertake this research and build the SIB infrastructure. The proposed work integrates core biological research with advances in databases and computer systems. The deliverable is a system that can be used by biologists while being of research interest

to database and computer systems researchers. The team has a long and distinguished track record of collaboration and system building, and their accomplishments have, over the last two decades, been recognized by the computer science community. The team brings together an ACM fellow (Roussopoulos), a AAAS fellow (Salzberg) and *all* six of the (relatively) junior members of the team have received a NSF CAREER award. In addition, Dr. Bhattacharjee has received a fellowship from the Alfred P. Sloan Foundation. The team has a history of co-advising students, and of collaborative research.

A post-doctoral researcher will interface between the computational biology and systems groups, with a focus on understanding and mapping the *details* of the biology requirements to concepts and artifacts in the sibdata implementation. We aim to produce a complete working implementation of the SIB system within two years in a form that can be used outside of our own laboratory. The systems programmer will be invaluable towards this end, both in terms of producing code that is usable across platforms and groups and in terms of documenting the intricacies of system for others to use.

SIB is designed to be broadly useful in biology research. Our team is grateful for the letter of support for SIB from Dr. Owen White, Director of Bioinformatics Institute for Genome Sciences at the Department of Epidemiology and Preventive Medicine, University of Maryland School of Medicine. We expect groups such as the Bioinformatics Institute for Genome Sciences to deploy a SIB prototype shortly after its public release.

In the rest of this section, we describe the decomposition of the proposed work between the computational biology, database, and systems PIs.

Computational Biology Professors Pop and Salzberg are currently funded under the Human Microbiome Project (HMP) to develop bioinformatics analysis tools for metagenomic data. As part of the current proposal they will evaluate the application of the SIB infrastructure to metagenomic research, specifically to manage the heterogeneous data (16S rRNA and high-throughput sequencing, microarray, clinical information, etc.) generated by HMP projects. Their research will help refine the specification of the system and determine development priorities in order to best respond to current needs of scientists. Prof. Pop and Salzberg are also involved in several collaborations with biologists generating primary data about the human commensal microbiota. Through these collaborations they will ensure that the SIB system will have an immediate impact on current research projects.

Databases The Database Group is led by Professor Nick Roussopoulos, (database integration and OLAP processing) and includes Professor Lise Getoor (Statistical learning and discovery data links) and Professor Amol Deshpande (probabilistic query models and sensor management). Professor Roussopoulos' research will focus on refining the fundamental sibdata abstraction, develop the semantics for concurrent use and update of sibdata, and develop techniques for on-line analysis of very large, multi-dimensional biological datasets. Dr. Deshpande will investigate and develop high- level query languages and declarative interfaces appropriate for manipulating and querying sibdata and develop techniques for efficient query processing over wide-area, distributed sib data stores. Dr. Getoor will work on incorporating support for extraction of entities, relationships, and links within the overall sibdata framework. She will also investigate the extraction of metadata mapping and alignments to enable finding potential matches at higher levels of data abstraction for the purpose of automated data integration.

Computer Systems The Systems group consists of Dr. Peter J. Keleher (distributed systems, file systems, and databases), Dr. Bobby Bhattacharjee (network protocols and security) and Dr. Neil T. Spring (network protocols and system building). The Systems PIs will be responsible for building the SIB infrastructure as described in this proposal. The research components include the tool support and kernel interfaces for sib data filesystem management, cryptography for chit-based sibdata access, translator and view architectures for presenting and allowing modifications to sibdata, and core algorithms and protocols for distributed access to sibdata. The systems PIs have extensive experience and track record in building large distributed systems, in kernel and filesystem development, in networking protocols and security architectures. Each of the systems PIs have worked together, and have co-advised (and graduated) students. SIB provides an unique forum for the integration of these skills and for collaboration with researchers in both databases and computational biology.