CINET: A CyberInfrastructure for Network Science

Sherif Elmeligy Abdelhamid[1], Richard Alo[2], S. M. Arifuzzaman[1], Pete Beckman[3], Md Hasanuzzaman Bhuiyan[1], Keith Bisset[1], Edward A. Fox[1], Geoffrey C. Fox[4], Kevin Hall[1], S.M.Shamimul Hasan[1], Anurodh Joshi[1], Maleq Khan[1], Chris J. Kuhlman[1], Spencer Lee[1], Jonathan P. Leidig[1], Hemanth Makkapati[1], Madhav V. Marathe[1], Henning S. Mortveit[1], Judy Qiu[4], S.S. Ravi[5], Zalia Shams[1], Ongard Sirisaengtaksin[2], Rajesh Subbiah[1], Samarth Swarup[1], Nick Trebon[3], Anil Vullikanti[1], and Zhao Zhao[1]

[1]Virginia Tech, Blacksburg, VA; [2]University of Houston, Houston, TX; [3]Argonne National Laboratory, Chicago, IL;
[4]Indiana University, Bloomington, IN; [5]University at Albany, SUNY, Albany, NY

***Abstract*—Networks are an effective abstraction for representing real systems. Consequently, network science is increasingly used in academia and industry to solve problems in many fields. Computations that determine structure properties and dynamical behaviors of networks are useful because they give insights into the characteristics of real systems. We introduce a newly built and deployed cyberinfrastructure for network science (CINET) that performs such computations, with the following features: (i) it offers realistic networks from the literature and various random and deterministic network generators; (ii) it provides many algorithmic modules and measures to study and characterize networks; (iii) it is designed for efficient execution of complex algorithms on distributed high performance computers so that they scale to large networks; and (iv) it is hosted with web interfaces so that those without direct access to high performance computing resources and those who are not computing experts can still reap the system benefits. It is a combination of application design and cyberinfrastructure that makes these features possible. To our knowledge, these capabilities collectively make CINET novel. We describe the system and illustrative use cases, with a focus on the CINET user.**

I. Introduction

A. Motivation

Network science research has been expanding at an ever quickening pace since the mid 1990s, as indicated by the numbers of publications related to complex networks [1]. This is not surprising, as the list of application areas employing graph abstractions, theory, and algorithms, or (agent-based) modeling and simulation, includes biology [18], ecology [4], cell immunology [16], social sciences (e.g., collective action, mass movements, revolutions, repression, emotions, technology adoption, drug use, drinking, obesity) [27], [20], [13], [33], health sciences [35], economics [10], computer networks [30], epidemiology [31], statistical physics [6], and language evolution [5]. Network science is useful for understanding system properties and behaviors within these domains. Hence, software and infrastructure that can perform graph-based computations is of great value. In this paper, we describe a free, newly released, web-based cyberinfrastructure for network science (CINET) for performing graph-based computations.

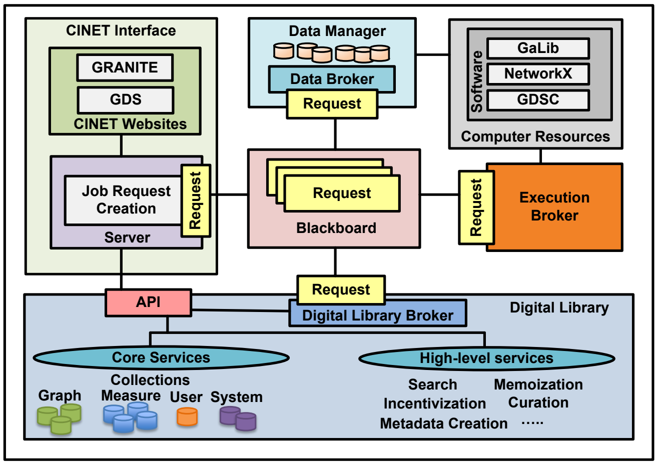


Fig. 1. High level overview of CINET system components and interactions.

B. System Overview

Fig. 1 describes the system at a high level. A web portal supports the complete set of user tasks related to cyberinfrastructure (CI) content and resources. The system has two components: the infrastructure component and the application component. The application component contains two classes of computations. First, the Granite system (with GaLib and NetworkX computation engines) generates graphs, computes a host of graph measures, and finds subgraph motifs. It has over 60 such network analysis operations. Many realistic graphs (e.g., those mined from domain data) from literature are also provided. Second, the graph dynamical systems calculator (GDSC) system (with the GDSC compute engine) computes dynamics on networks. The CI, or infrastructure component, is represented by the rest of Fig. 1 and includes the web-based user interface (UI), a digital library (DL), and a multi-component backend (e.g., job request server, blackboard, data broker, and execution broker) for job and content management and high performance computing (HPC). Both the applications and the CI are described later in more detail. This system is hosted by Virginia Tech and is free for public use.

978-1-4673-4466-1/12/$31.00 ©2012 IEEE

C. Example Use Cases

Two use cases are presented that illustrate the utility of CINET. A user here may be a student, teacher, researcher, or practitioner. In both cases, a user has logged into CINET via the Internet.

Case 1: A user selects a pre-existing graph and a group of graph measures, e.g., distributions of all-pairs shortest paths and of clustering coefficient and finding pentagon subgraphs in the graph. The user clicks a submit button, and CINET provides status updates and notifications as computations progress and complete. Depending on measures selected, graphs may be up to a few billion nodes in size. Output is given in two forms: (i) the raw data (e.g., textual distributions and counts) and (ii) plots (e.g., *.pdf or *.eps files). Data and plots are displayed on-screen and may be downloaded to a user's machine.

Case 2: A user is interested in dynamics on a graph. The user selects a type of graph (e.g., circle or lattice) and number of nodes in the network. She then specifies a local function (such as the *nor* function) that describes how nodes change state based on influence from adjacent (neighboring) nodes. The state of a node can be either 0 or 1. A choice of update sequence is made; all nodes execute their local functions simultaneously or sequentially. The sequential option is chosen and the submit button is clicked. All user inputs (except number of nodes) are specified from drop-down lists. Job status is displayed as GDSC computes all state transitions. Results are details of the long-term graph dynamics, and these data can be viewed on-screen and downloaded as a text file.

D. Contributions

The major contributions of this work are as follows.

1. A free web-based HPC large-graph analysis tool. We envision a user base that includes students, teachers, domain experts, and investigators who do not have the means to build such systems, lack access to HPC resources for large graph analyses, or have no need to concern themselves with computational details. Properties of pre-existing and user-generated graphs can be computed, as can dynamics on selected graphs. Particularly with Granite, large graphs (i.e., those with billions of nodes) can be analyzed. The system is intended to significantly reduce the turn-around time for computing answers to questions and the technical expertise required to use the applications.

2. Extensible applications. For the Granite system, mined or realistic networks can be added by the user community. Software to compute new graph measures and new graph generators can be added (by users). For the GDSC system, new local functions that describe dynamics can be added, as can additional node update schemes, and state spaces. (For GDSC functionality, system administrators must currently add new features, but the system is designed for such enhancements.) These application-based extensions are made with no alterations in the infrastructure.

3. Extensible cyberinfrastructure. The CI is extensible in many dimensions. We separate this extensibility contribution from those for applications to emphasize that the *infrastructure* (i.e., UI, DL, and backend) is designed to handle multiple (distributed) applications. Hence, new applications can be supported by the CI without compromising existing ones. Beyond leveraging this functionality and code base, another practical benefit is that a user need only learn the CINET web-based UI because a new application is, from a user's perspective, a blackbox (just as Granite and GDSC are blackboxes). There are other avenues for extending the CI, e.g., adding computing resources and adding DL features.

4. Promotion of remote interdisciplinary collaboration. Users from different geographic regions can collaborate within and contribute to CINET. Moreover, this tool facilitates investigations by multi-disciplinary teams whose areas of expertise often reside in different organizations. The DL provides collaboration-supporting services.

5. Common repository. As explained above, the network science-related literature spans many fields. As this trend is expected to continue, it becomes increasingly more difficult for researchers to keep abreast of advancing software and methodologies. Tools such as CINET offer a common site where people can go to produce and share graphs and results, as well as contribute software and functionality that benefits the user community.

Organization. The remainder of the paper is organized as follows. Related work is addressed in Section II. The system architecture is explained in Section III, which ties together all subsequent sections. Granite and GDSC applications are described in Section IV. Two of the three main components of CI, the UI and DL, are described in Sections V and VI, respectively. Owing to space limitations, the low-level details of the third CI component, the backend infrastructure, are omitted. However, a high level treatment is provided in the system architecture description. Section VII concludes the paper.

II. Related Work

Several network analysis tools exist, including the Stanford Network Analysis Project (SNAP) [24], which is a general purpose network analysis and graph mining library. SNAP is also available through NodeXL, a graphical front-end that integrates network analysis into Microsoft Office and Excel. Another toolkit, Network Workbench [28], has been used for biomedical, social science, and physics research. Network Workbench provides an online portal for researchers, educators, and practitioners. PEGASUS [11] is a peta-scale distributed graph mining system that runs on clouds. PEGASUS also provides large scale algorithms for important graph mining tasks. Pajek [34] is a tool for the analysis and visualization of networks having thousands or millions of vertices. NetworkX [15] is an open source software package for the generation and study of complex networks. NetworkX contains a large collection of graph algorithms. PEGASUS and

Galib have parallel implementations of some graph algorithms, while others implement sequential algorithms only. However, PEGASUS has very few graph algorithms.

From a dynamics perspective, [36] provides a web-based tool for computing trajectories and phase spaces and is similar to GDSC. Local functions can be combinations of logical and, or, and not, and can also be additive and multiplicative.

Numerous workflow management systems exist to perform functions such as executing an experiment, e.g., Taverna [29] and Pegasus [8]. However, these systems do not provide the full range of functionality and coordination requried in this context.

Existing digital library packages are not capable of supporting large-scale scientific research environments. Scientific digital libraries are an emerging approach related to modeling, managing, analyzing, supporting, and understanding scientific research systems. The eScience and cyberinfrastructure research communities are actively attempting to improve scientific data management practices. Examples of scientific data management projects include earthquake simulation repositories [17], embedded sensor network DLs [3], community earth systems [9], D4Science II [23], mathematical-based retrieval [37], chemistry systems [25], national research data plans [19], and science portals [26]. However, existing systems do not provide the types of services required to support modeling and simulation. These inadequacies motivated the development of our simulation-supporting digital library.

Overall, at the CI level, few systems can match CINET in provided HPC resources, contributor-based content, software environment, and collaboration-supporting services. We are not aware of any system that supports compute-intensive domains such as network science in all of these dimensions. At the application level, we know of no network science application suite that provides high performance distributed algorithms, a wide range of algorithmic modules and measures, and graph dynamics functionality. Furthermore, public availability allows domain analysts to benefit automatically from provided HPC resources. In contrast, to exploit other publicly available software, users currently must have privileges to install new software on contributed HPC machines.

III. System Architecture

CINET is a distributed system that constitutes a set of well-defined processes and services that coordinate to fulfill a given request and perform its associated tasks. CINET embraces a JavaSpaces-based architecture that relies on persistent object exchange for loosely-coupled coordination among services. This supports extensibility and component change-out. Fig. 1 depicts the high-level architecture of the CINET framework with key components that are discussed in this and the following sections in greater detail.

A. *Blackboard*

The blackboard is the central communication and coordination mechanism for CINET. It is currently implemented with a JavaSpace. It provides asynchronous, loose coupling of system components. Components do not need to be aware of the existence of the other components in the system. They simply put requests onto the blackboard and wait for them to be fulfilled.

Requests are Java objects that contain details about how they are to be fulfilled, in the form of an embedded workflow. These are active requests, not simply collections of data. For instance, an analysis request contains not only the parameters to run the analysis but also a runner object that contains the workflow to run the analysis, including pre- and post-processing and validation of the output produced.

B. *Brokers*

A broker is a component that is responsible for providing a service. It does this by monitoring the blackboard for specific requests that it can fulfill. It takes these requests from the space and executes the workflow embedded in the request. In the process of executing the workflow, it may put requests for other services onto the blackboard for other brokers to fulfill. The CINET framework has the following primary brokers.

- Execution Broker: The execution broker is responsible for identifying execution requests and running each on a specific machine. It does this by constructing system-specific job submission scripts and monitoring the progress of the execution. Results generated by execution are typically transferred with the help of the data (management) broker.
- Data (Management) Broker: A data manager is responsible for managing the data resources that reside on a system. A resource may be a path to a file, or a “fully qualified” database (DB) query. The data manager will also request the transfer of non-local datasets, and the creation of non-existent datasets. It also may make decisions about transfer versus regeneration, purging unused datasets, prefetching data, etc.
- Digital Library Broker: The DL broker is responsible for communicating with DLs and fetching appropriate information that is required for the end-to-end execution of an analysis request. Some of the core functions include: add/remove graph(s)/measure(s), add/retrieve/update information about graph(s)/measure(s), and add/retrieve/remove execution results. The DL also performs higher-level services such as searching, memoizing results, incentivizing contributions, metadata indexing, and curating. The DL is described more fully in Section VI.

C. *CINET Interface*

The CINET Interface consists of user interfaces and web applications that enables a user to submit analysis requests, add graph measure software, add graphs, and/or perform administrative tasks. The CINET interface is discussed in Section V.

D. *Compute Resource*

Compute resources are the physical resources on which jobs are executed. Current resources are two HPC Linux clusters at

TABLE I
NUMBER OF GALIB AND NETWORKX MODULES IN CINET

Type	GaLib	NetworkX
Graph Generator	11	11
Centrality	14	11
Shortest path and connectivity	15	8
Subgraph/motif counting	8	2
Others	15	3
Total measures	63	35

Virginia Tech. Potential compute resources include traditional HPC clusters, compute grids (e.g., Open Science Grid) and clouds (e.g., Amazon Web Services), volunteer computing platforms (e.g., BOINC), or dedicated servers. A typical compute resource runs NetworkX and GaLib components (which constitute the Granite compute engine) and the GDSC engine. These components contain the binaries for performing graph analyses. Also, execution and data management brokers run on compute resources to supervise the generation and management of results.

E. Applications

Current applications are in the network science domain, i.e., Galib, NetworkX, and GDSC.

IV. GRAPH ANALYSIS AND MODELING

In this section, we overview the application component capabilities for producing and evaluating structure properties of graphs and for modeling dynamics on graphs. Galib and GDSC were developed by the Network Dynamics and Simulation Science Laboratory at Virginia Tech.

A. Graph Algorithms for Static Analysis of Networks

Static analyses are meant to compute various static measures associated with networks; e.g., the number of triangles, diameter, and breadth-first search (BFS) tree. GaLib and NetworkX are the computation engines that provide CINET with necessary capabilities for analyzing network structures and determining various metrics of interest associated with real-world and artificial networks.

GaLib is a Graph Algorithm Library written in C++. There are other existing graph algorithm libraries such as NetworkX [15], SNAP [24], Pajek [34], Network Workbench [28] and PEGASUS [11]. All of these graph libraries are useful as they contain different sets of graph algorithms (although there are algorithms common to them) and different libraries have different beneficial features. GaLib has about 60 parallel and sequential graph algorithms implemented (see Table I). Currently, CINET includes algorithms from GaLib and NetworkX. We plan to incorporate these other graph libraries into CINET in the future. (Contribution (3) of Section I-D will facilitate these additions.)

GaLib is specifically designed to deal with very large networks, focusing on the challenges arising from working with emerging massive networks. These large networks demand new capabilities and considerations. GaLib's carefully-designed data structures allow us to work with networks containing up to 100 million nodes for sequential algorithms and up to 2 billion nodes for some parallel algorithms. In addition to the carefully-designed data structures and parallel algorithms, GaLib employs other techniques, such as streaming, external memory algorithms, and sampling-based approximation. Some results of analyses generated with GaLib on two networks, Miami [2] and LiveJournal [24], with 2.1 million and 4.8 million nodes, respectively, are shown in Figs. 2 and 3.

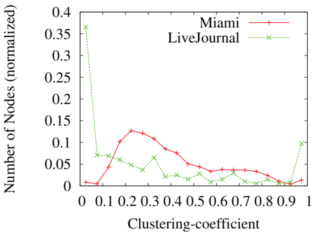


Fig. 2. Clustering coefficient distributions.

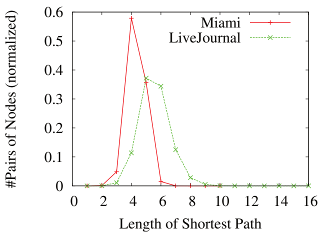


Fig. 3. Shortest path distributions.

TABLE II
PERFORMANCE COMPARISON BETWEEN GALIB AND NETWORKX FOR THREE MEASURES ON AN ERDÖS-RÉNYI RANDOM GRAPH WITH 500K NODES AND AVERAGE DEGREE 20.

Measures	Runtime (sec)		Memory (GB)	
	GaLib	NetworkX	GaLib	NetworkX
Single Source Shortest Path	12	275	0.22	3.6
Check Connectivity	14	360	0.22	3.6
Counting Triangles	20	480	0.22	3.6

NetworkX [15] is an open source software package written in Python, developed at Los Alamos National Laboratory, and was first released in 2005 for public use. NetworkX does not scale well to large networks—a performance comparison of some NetworkX algorithms with those of GaLib is given in Table II. However, NetworkX has some attractive features that makes it very useful. It has several hundreds of graph algorithms implemented (currently 35 of them are included in CINET as shown in Table I). Further, Python is a very powerful language with a large number of built-in data structures supporting rapid prototyping. NetworkX provides various primitive functions for graph algorithms that make it simple and flexible to use for the representation and manipulation of many complex networks, including directed, undirected, and multi-graphs in different data formats. In addition, NetworkX's graph drawing features make it useful for classroom use and studying newly developed algorithms.

B. GDS Calculator

1) Social Sciences As A Case Study: We use social sciences to demonstrate features of the GDSC. Often in the social sciences, human populations are represented as graphs, where vertices (nodes) denote agents and edges correspond to interactions between them. Threshold systems have been used for decades to model collective action and the influence of one (human) agent on another [14], [32], [7]. In its most basic form, in a two-state system, an agent v will change from the non-participating state (e.g., state 0) to the participating state (e.g., state 1) when a threshold θ_v number of its neighbors are in the participating state.

2) Illustrative Results: Selected inputs and outputs are displayed graphically in Fig. 4 for the BITHRESHOLD vertex function and $\{0, 1\}$ vertex state space, where for each vertex v in a population, $\theta_{v,up} = 1$ and $\theta_{v,down} = 3$. Thus, for v in state 0, if at least $\theta_{v,up}$ neighbors are in state 1, then v transitions to state 1; otherwise v's state x_v remains 0. For v in state 1, if less than $\theta_{v,down}$ vertices in its closed neighborhood (i.e., including the state of v) are in state 1, then v will transition to state 0; otherwise it remains in state 1. The graph X, at the left side of Fig. 4, is a 4-vertex square, known formally as Circle-4 (Circ$_4$). The right graphic is the phase space (i.e., all 16 system state transitions) for synchronous vertex (or local) function update; i.e., all vertices update their states simultaneously. There are three 2-cycles in the phase space (identified by the green arrows). Suppose we have a system conforming to these conditions, where a vertex in state 0 means that the agent is not participating in a revolt, and state 1 means that an agent is participating. Then there are many interesting observations to be made about this system, but we give only one. For any initial system state other than $(0, 0, 0, 0)$ or $(1, 1, 1, 1)$, the three 2-cycles indicate that in the long term, two agents will be participating at any one time (i.e., exactly two vertices have states of 1 on the 2-cycles). In GDSC, a user may select from different graph classes and numbers of vertices, vertex functions, and vertex function update schemes.

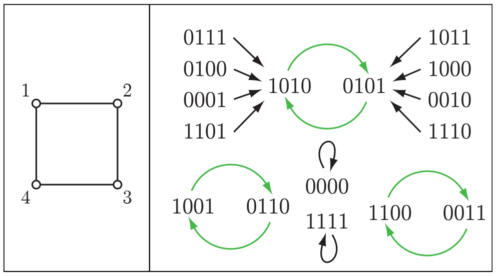


Fig. 4. (Left) graph Circ$_4$; and (right) phase space for the synchronous BITHRESHOLD GDS map with 2-cycles shown in green.

V. USER INTERFACE

Owing to space constraints, we describe here only the Granite UI. The Granite system includes a CI web application that supports the interactive UI. It allows users to work with a wide range of graphs and measures. The objective of this system is to move beyond the traditional database and information management system to integrate content, community, and services. The Granite system interacts with the user through a tab-based interface as shown in Fig. 5. Smart GWT is used for Granite development. The following is a brief description of the Granite UI.

A. Workspace Tab

The workspace tab allows a user to select one or more graphs within CINET. These include instances of random networks (RN), social networks (SN) generated by various groups, and several of the networks from the SNAP repository [24]. Also, the workspace tab provides a list of available measures that can be applied to the selected networks.

B. Results Tab

Once the user clicks on the analyze button, the system will start to process the requested jobs. The results tab shows the progress of each job and allows the user to view output.

C. Add New Graph Tab

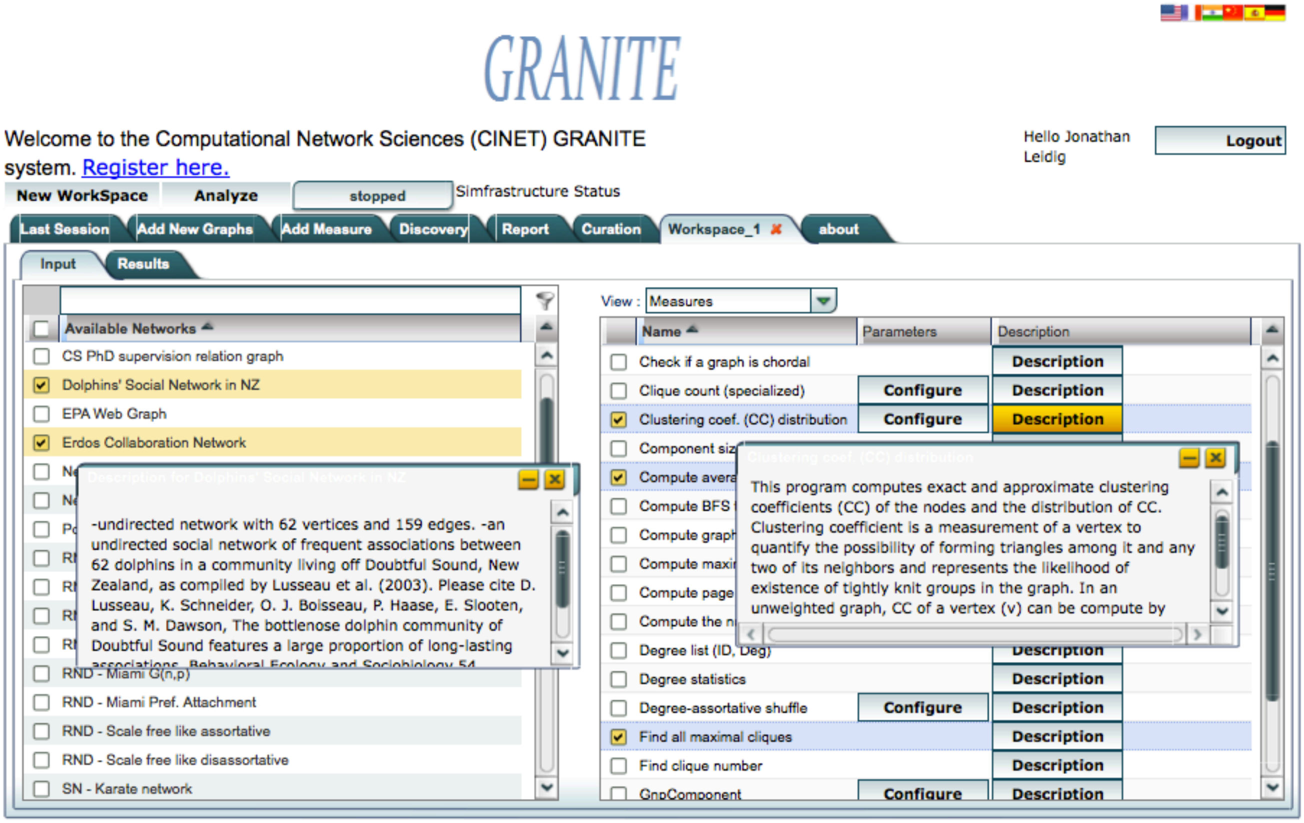
The UI for adding new graphs provides for a two-step process. A user needs to upload a graph and then needs to complete a metadata form for the new graph. After uploading a graph, automated subservices (graph conversion, graph consistency checking, and basic metadata extraction) are sequentially initiated and performed. The new graph is converted to all other available graph formats, and then a GaLib measure is used to check graph consistency. If a graph fails the consistency check, the system rejects the graph and informs the user with error messages returned by the GaLib measure. Otherwise, the system automatically extracts basic properties of the graph, which are normally called degree statistics, such as numbers of nodes and edges, degree distribution, etc., and then transfers the graph file to the production repository to make it available in the list of graphs of the main Granite UI.

D. Add Measure Tab

The Add Measure service provides a UI to add a new measure and associated information to CINET. See Fig. 6. By "adding a new measure," we mean adding an executable to CINET, along with a schema of the required parameters. To add a measure, a user needs to insert information for measure name, description, tool type (e.g., GaLib or NetworkX), output files name(s), executable file name, measure parameter, and runtime information of the measure. Based on the user input, the system dynamically creates different types of widgets to collect information about parameters and output file names. It allows a user to insert information about parameter type, name, and value. Also, several data validation checks are performed by this service. If a measure is successfully added to the system, it will appear in the measure list. Currently, this service is only available for administrative users. This is to ensure that new measure codes are validated and not malicious before being exposed to the user community.

VI. DIGITAL LIBRARY

Digital libraries (DLs) have greatly advanced since the community was formed in the early 1990s. However, efforts to support scientific and simulation-based research have been minimal until now. As a part of CINET, we have developed a simulation-supporting digital library (SimDL) [21], [22]. SimDL is a framework for producing DL instances that



Welcome to the Computational Network Sciences (CINET) GRANITE system. Register here.

Available Networks

- □ CS PhD supervision relation graph
- ☑ Dolphins' Social Network in NZ
- □ EPA Web Graph
- ☑ Erdos Collaboration Network
- □ RND - Miami G(n,p)
- □ RND - Miami Pref. Attachment
- □ RND - Scale free like assortative
- □ RND - Scale free like disassortative
- □ SN - Karate network

Description for Dolphins' Social Network in NZ

-undirected network with 62 vertices and 159 edges. -an undirected social network of frequent associations between 62 dolphins in a community living off Doubtful Sound, New Zealand, as compiled by Lusseau et al. (2003). Please cite D. Lusseau, K. Schneider, O. J. Boisseau, P. Haase, E. Slooten, and S. M. Dawson, The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations, Behavioral Ecology and Sociobiology 54

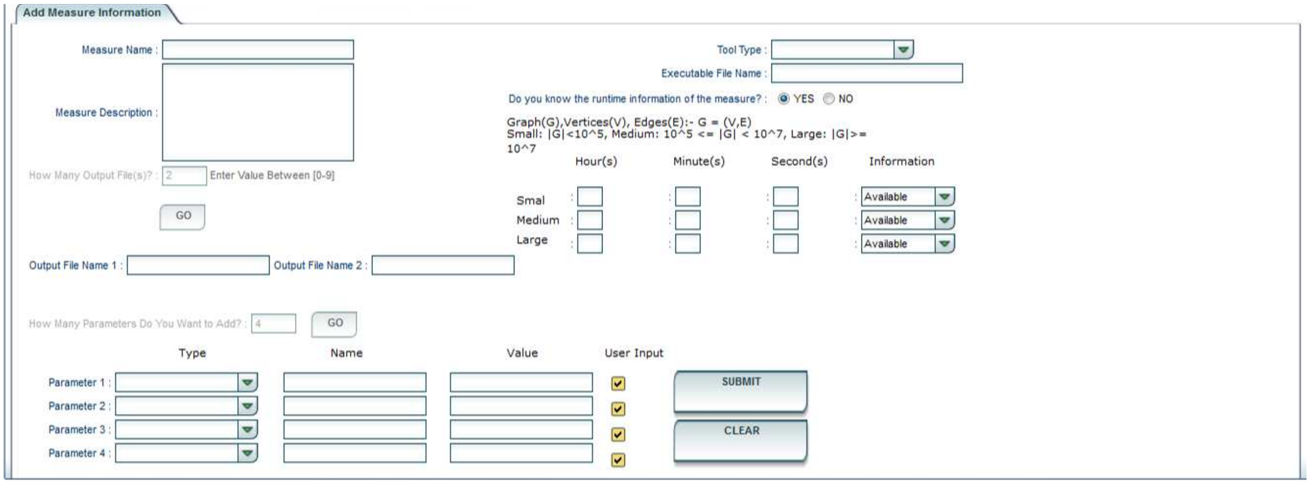
Measures

- □ Check if a graph is chordal
- □ Clique count (specialized)
- ☑ Clustering coef. (CC) distribution
- □ Component siz
- ☑ Compute avera
- □ Compute BFS
- □ Compute graph
- □ Compute maxi
- □ Compute page
- □ Compute the n
- □ Degree list (ID, Deg)
- □ Degree statistics
- □ Degree-assortative shuffle
- ☑ Find all maximal cliques
- □ Find clique number
- □ GnpComponent

Clustering coef. (CC) distribution

This program computes exact and approximate clustering coefficients (CC) of the nodes and the distribution of CC. Clustering coefficient is a measurement of a vertex to quantify the possibility of forming triangles among it and any two of its neighbors and represents the likelihood of existence of tightly knit groups in the graph. In an unweighted graph, CC of a vertex (v) can be compute by

Fig. 5. Granite system user interface showing available large networks and scalable analysis measures.



How Many Output File(s)? : 2

How Many Parameters Do You Want to Add? : 4

Do you know the runtime information of the measure? : ● YES ○ NO

Graph(G),Vertices(V), Edges(E):- G = (V,E)
Small: |G|<10^5, Medium: 10^5 <= |G| < 10^7, Large: |G|>= 10^7

Fig. 6. UI for acquiring new network analysis measures.

support large-scale simulation-based infrastructures. Formal definitions using the 5S formal framework [12] were produced that precisely describe the services required to support this community and identify functionality absent in current open-source DL software. See Fig. 7 for the metamodel produced by these formalizations. From these functional definitions, the minimal set of simulation-supporting services was identified and implemented to produce a software toolkit. The final result is a software package that may be deployed in various simulation research infrastructures.

A. Core Functionality

The SimDL instance utilized in this work includes core information storage and retrieval services to manage graphs, measures, and results, as well as higher-level simulation supporting services to support infrastructure and scientific tasks,

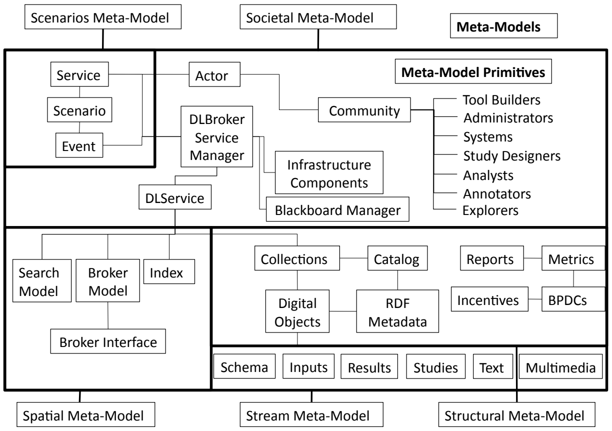


Fig. 7. Metamodel leading to SimDL's architectural design.

TABLE III
PRIMARY METADATA OF EACH MAJOR DATA TYPE.

Graph	Measure	CINET logic	User
Name Description Number of nodes Number of edges File path Network type Graph format Date of addition Owner-right Source	Name Description Parameters Command Category	Network tool supportability Estimated running time based on graph size	ID Real name Group Password

as shown in Fig. 1. CINET uses the metadata provided by the DL to create job requests which are compositions of graphs, measures, and resource information in the form of shell scripts. The system also receives decisions of tool support abilities based on sizes and estimated running times of graphs and measures selected by users.

Table III shows the primary metadata for each major data type used by the CINET system. These are the fundamental data used by the core DL services mentioned above and the simulation supporting services described in the next section.

B. Simulation Supporting Services

Simulation supporting services aim to provide incentives for using the system and efficient data management services for infrastructure components. Each of these services described here was designed and implemented to provide scalable functionality appropriate for highly numeric, data-intensive scientific content. While other DL systems often provide these services in other contexts, e.g., full-text publications, existing implementations do not scale well for automatic indexing and support of large quantities of simulation-produced digital objects. SimDL utilizes the DL broker to communicate with other CI components, e.g., UIs, HPC systems, and simulation models. Through this broker, other components can contribute, register, and query collections of new datasets, simulation studies, metadata records, etc. The broker also is utilized to process requests for the set of simulation-specific DL service implementations. The formal definitions leading to the design of high-level services are overviewed in Fig. 8.

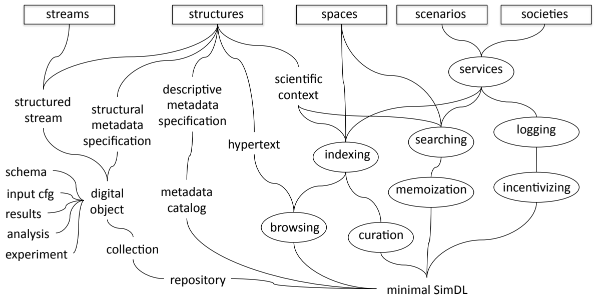


Fig. 8. Formalisation of the minimal simulation-supporting digital library.

To incentivize researchers and educators to contribute to and utilize CINET, components log the submission and usage of graphs, measures, HPC systems, and user behaviors. Thus, we have a logging system that tracks what types of activities and data products are related to users. A DL incentivization service then generates reports detailing statistics for each class of user, e.g., content provider, software provider, student, and HPC system administrator. The memoization service maintains existing simulation results and returns these to the UI if a deterministic simulation study is designed and submitted multiple times. This minimizes unnecessary workloads on contributed systems and provides quicker access time to end-users. The curation service recommends when digital objects should be archived, preserved, migrated, and deleted from the DL based on administrator-defined rules. Additional services execute query searches, filter content lists, support automated metadata indexing, and track provenance through the simulation workflow. This set of minimal SimDL services supports UIs, workflows, HPC systems, simulation models, and user tasks using automated, scalable, and domain-free software implementations.

VII. CONCLUSIONS

We have described CINET, a cyberinfrastructure for network science and analysis of large graphs. The CI provides a middleware platform connecting high performance compute engines on the back end (that run on HPC clusters) with a SimDL instance and data managers that generate, decide the routing of, and control the flow of job submissions made by users through the UI portal. Currently, this CI supports two applications: (i) Granite and (ii) GDSC. The system–both the CI infrastructure and the applications–is extensible in many ways. For example, the CI can support other applications, and the Granite application can incorporate new algorithms provided by the user community. Some algorithms scale to billion-node graphs and some of the technologies to support these have been mentioned. Representative results and performance data on large real networks have been provided. The system is designed to track use of different features so that proper public credit can be given to those contributing data and methods. Free for public use, the system is intended to foster (remote, interdisciplinary) collaboration and provide a common repository for network science research and for users to share data and analysis methods.

Granite and GDSC can be accessed by clicking the "Network Analysis" button on the CINET website (http://ndssl.vbi.vt.edu/cinet). The site provides all information related to the CINET project. The website also acts as an online repository for educational materials related to network science that includes courses and notes, presentations, and survey papers.

Acknowledgment

We thank the reviewers and members of the Network Dynamics and Simulation Science Laboratory (NDSSL) for their suggestions and comments. This work has been partially supported by NSF NetSE Grant CNS-1011769, NSF SDCI Grant OCI-1032677, NSF PetaApps Grant OCI-0904844 and DTRA CNIMS Contract HDTRA1-11-D-0016-0001.

References

[1] National Research Council Committee on Network Science for Future Army Applications. In *Network Science*. The National Academies Press, 2005.

[2] C. L. Barrett, R. J. Beckman, M. Khan, V. S. Anil Kumar, M. V. Marathe, P. E. Stretz, T. Dutta, and B. Lewis. Generation and analysis of large synthetic social contact networks. In *Winter Simulation Conference*, 2009.

[3] C. L. Borgman, J. C. Wallis, M. S. Mayernik, and A. Pepe. Drowning in data: digital library architecture to support scientific use of embedded sensor networks. In *Proc. JCDL 2007*, pages 269–277, 2007.

[4] C. Campbell, S. Yang, R. Albert, and K. Sheab. A network model for plantpollinator community assembly. *Proceedings of the National Academy of Sciences*, 108(1):197–202, 2011.

[5] A. Cangelosi and D. Parisi. Computer Simulation: A New Scientific Approach to the Study of Language Evolution. In A. Cangelosi and D. Parisi, editors, *Simulating the Evolution of Language*. Springer, 2001.

[6] C. Castellano, S. Fortunato, and V. Loreto. Statistical physics of social dynamics. *Rev. Mod. Phys.*, 81(2):591–646, 2009.

[7] D. Centola and M. Macy. Complex Contagions and the Weakness of Long Ties. *American J. Sociology*, 113(3):702–734, 2007.

[8] E. Deelman, J. Blythe, Y. Gil, C. Kesselman, S. Koranda, A. Lazzarini, G. Mehta, M. A. Papa, and K. Vahi. Pegasus and the Pulsar Search: From Metadata to Execution on the Grid. In *Applications Grid Workshop at the Fifth International Conference on Parallel Processing and Applied Mathematics (PPAM)*, pages 821–830, Czestochowa, Poland, 2003.

[9] R. Dunlap, L. Mark, S. Rugaber, V. Balaji, J. Chastang, L. Cinquini, C. DeLuca, D. Middleton, and S. Murphy. Earth system curator: metadata infrastructure for climate modeling. *Earth Science Informatics*, 1:131–149, 2008. 10.1007/s12145-008-0016-1.

[10] D. Easley and J. Kleinberg. *Networks, Crowds and Markets: Reasoning About A Highly Connected World.* Cambridge University Press, New York, NY, 2010.

[11] C. Faloutsos. Project Pegasus. http://www.cs.cmu.edu/~pegasus/, 2009. [Online; accessed 17-July-2012].

[12] M. A. Gonçalves, E. A. Fox, L. T. Watson, and N. A. Kipp. Streams, structures, spaces, scenarios, societies (5S): A formal model for digital libraries. *ACM Trans Inf Syst*, 22(2):270–312, 2004.

[13] S. Gonzalez-Bailon, J. Borge-Holthoefer, A. Rivero, and Y. Moreno. The Dynamics of Protest Recruitment Through an Online Network. *Nature Scientific Reports*, pages 1–7, 2011. DOI: 10.1038/srep00197.

[14] M. Granovetter. Threshold Models of Collective Behavior. *American J. Sociology*, 83(6):1420–1443, 1978.

[15] A. A. Hagberg, D. A. Schult, and P. J. Swart. Exploring network structure, dynamics, and function using NetworkX. In *Proceedings of the 7th Python in Science Conference (SciPy2008)*, pages 11–15, Pasadena, CA, USA, Aug. 2008.

[16] B. Hancioglu, D. Swigon, and G. Clermont. A dynamical model of human immune response to influenza A virus infection. *Journal of Theoretical Biology*, 246(1):70–86, 2007.

[17] T. H. Jordan. SCEC 2009 Annual Report. *Southern California Earthquake Center*, 2009.

[18] U. Karaoz, T. Murali, S. Letovsky, Y. Zheng, C. Ding, C. Cantor, and S. Kasif. Whole-genome annotation by using evidence integration in functional-linkage networks. *Proceedings of the National Academy of Sciences*, 101(9):2888–2893, 2004.

[19] S. Kethers, X. Shen, A. E. Treloar, and R. G. Wilkinson. Discovering Australia's research data. In *Proc. JCDL 2010*, pages 345–348, 2010.

[20] C. Kuhlman, V. Kumar, M. Marathe, S. Ravi, D. Rosenkrantz, S. Swarup, and G. Tuli. Inhibiting the Diffusion of Contagions in Bi-Threshold Systems: Analytical and Experimental Results. In *Proceedings of the AAAI Fall 2011 Symposium on Complex Adaptive Systems (CAS-AAAI 2011)*, pages 91–100, November 2011.

[21] J. Leidig, E. Fox, M. Marathe, and H. Mortveit. Epidemiology experiment and simulation management through schema-based digital libraries. In *Proceedings of the 2nd DL.org Workshop at ECDL*, pages 57–66, 2010.

[22] J. Leidig, E. A. Fox, K. Hall, M. Marathe, and H. Mortveit. SimDL: A Model Ontology Driven Digital Library for Simulation Systems. In *ACM/IEEE Joint Conference on Digital Libraries*, JCDL '11. ACM, 2011.

[23] P. P. Leonardo Candela, Donatella Castelli. D4Science: an e-infrastructure for supporting virtual research. In *Proceedings of IRCDL 2009 - 5th Italian Research Conference on Digital Libraries*, pages 166–169, 2009.

[24] J. Leskovec. Stanford Network Analysis Project. http://snap.stanford.edu/, 2009. [Online; accessed 17-July-2012].

[25] N. Li, L. Zhu, P. Mitra, K. Mueller, E. Poweleit, and C. L. Giles. oreChem ChemXSeer: a semantic digital library for chemistry. In *Proc. JCDL 2010*, pages 245–254, 2010.

[26] R. W. Moore, A. Rajasekar, M. Wan, Y. Katsis, D. Zhou, A. Deutsch, and Y. Papakonstantinou. Constraint-based Knowledge Systems for Grids, Digital Libraries, and Persistent Archives: Yearly Report. In *SDSC TR-2005-5*, 2005.

[27] D. J. Myers and P. E. Oliver. The opposing forces diffusion model: the initiation and repression of collective violence. *Dynamics of Asymmetric Conflict*, 1:164–189, 2008.

[28] NWB Team. Network Workbench Tool. Indiana University, Northeastern University, and University of Michigan. http://nwb.cns.iu.edu, 2006. [Online; accessed 17-July-2012].

[29] T. Oinn, M. Greenwood, M. Addis, N. Alpdemir, J. Ferris, K. Glover, C. Goble, A. Goderis, D. Hull, D. Marvin, P. Li, P. Lord, M. Pocock, M. Senger, R. Stevens, A. Wipat, and C. Wroe. Taverna: lessons in creating a workflow environment for the life sciences. In *Concurrency and Computation: Practice and Experience*, volume 18, pages 1067–1100, 2006.

[30] R. Puzis, M. Tubi, Y. Elovici, C. Glezer, and S. Dolev. A Decision Support System for Placement of Intrusion Detection and Prevention Devices in Large-Scale Networks. *ACM Transactions on Modeling and Computer Simulation*, 22:1–2, 2011.

[31] T. C. Reluga, J. Medlock, and A. S. Perelson. Backward bifurcations and multiple equilibria in epidemic models with structured immunity. *Journal of Theoretical Biology*, 252:155–165, 2008.

[32] T. Schelling. *Micromotives and Macrobehavior*. W. W. Norton and Company, 1978.

[33] J. Tsai, E. Bowring, S. Marsella, and M. Tambe. Empirical evaluation of computational emotional contagion models. In *Proceedings of the 11th International Conference on Intelligent Virtual Agents (IVA 2011)*, 2011.

[34] B. V. and M. A. Pajek - Program for Large Network Analysis. *Connections*, 21(2):47–57, 1998.

[35] T. W. Valente. *Social Networks and Health: Models, Methods, and Applications*. Oxford University Press, 2010.

[36] H. Vastani, N. Eriksson, R. Laubenbacher, A. Jarrah, B. Stigler, and F. Hinkelmann. Discrete Visualizer of Dynamics (DVD) v1.0. http://dvd.vbi.vt.edu/cgi-bin/git/dvd.pl, 2012. [Online; accessed 17-July-2012].

[37] J. Zhao, M.-Y. Kan, and Y. L. Theng. Math information retrieval: user requirements and prototype implementation. In *Proceedings of JCDL '08*, pages 187–196, 2008.