Scientific Programming 17 (2009) 231–246
DOI 10.3233/SPR-2009-0261
IOS Press

From data to knowledge to discoveries: Artificial intelligence and scientific workflows

Yolanda Gil
Information Sciences Institute, University of Southern California, 4676 Admiralty Way, Marina del Rey, CA 90292, USA
E-mail: gil@isi.edu

Abstract. Scientific computing has entered a new era of scale and sharing with the arrival of cyberinfrastructure facilities for computational experimentation. A key emerging concept is scientific workflows, which provide a declarative representation of complex scientific applications that can be automatically managed and executed in distributed shared resources. In the coming decades, computational experimentation will push the boundaries of current cyberinfrastructure in terms of inter-disciplinary scope and integrative models of scientific phenomena under study. This paper argues that knowledge-rich workflow environments will provide necessary capabilities for that vision by assisting scientists to validate and vet complex analysis processes and by automating important aspects of scientific exploration and discovery.

Keywords: Workflows, computational experimentation, scientific computing, artificial intelligence, automated discovery, knowledge systems

1. Introduction

Computational experimentation is now a ubiquitous technique across science domains. It encompasses all aspects of the scientific experimentation process including data analysis, simulation, hypothesis generation and hypothesis testing. This has driven a tremendous investment in cyberinfrastructure [1] designed to provide shared resources for large-scale computational science. Results from computational experimentation have an ever-increasing impact in scientific practice, producing significant advancements in almost every discipline [35,37,50].

This paper argues that despite the clear impact of current cyberinfrastructure in science, there are severe limitations in terms of the breadth and scope that can be supported. It introduces computational workflows as key artifacts to further computational science. It presents current workflow systems and their capabilities to isolate scientists from execution details in complex distributed environments. Workflow systems have significant benefits and are becoming common elements in cyberinfrastructure. Looking forward, the paper discusses the need to assist scientists at a higher level that requires capturing and exploiting scientific knowledge about the software and data used in computational experimentation. It presents current research in workflow systems that exploit this knowledge to automate complex validations and decision making on behalf of the scientist. Finally, it presents five areas of future research where knowledge-rich workflow systems can provide significant added value to existing cyberinfrastructure capabilities for computational experimentation.

2. Cyberinfrastructure for scientific research

Cyberinfrastructure had its roots in the High Performance Computing community and large-scale scientific computing, where large data repositories and high-end computing facilities needed to reside at specific locations while being effectively accessible by remote users. Cyberinfrastructure broadly construed includes not only data and computing facilities but also instruments, tools, and often the people involved in forming and using all this combined infrastructure [1]. A variety of middleware software enables access and ex-

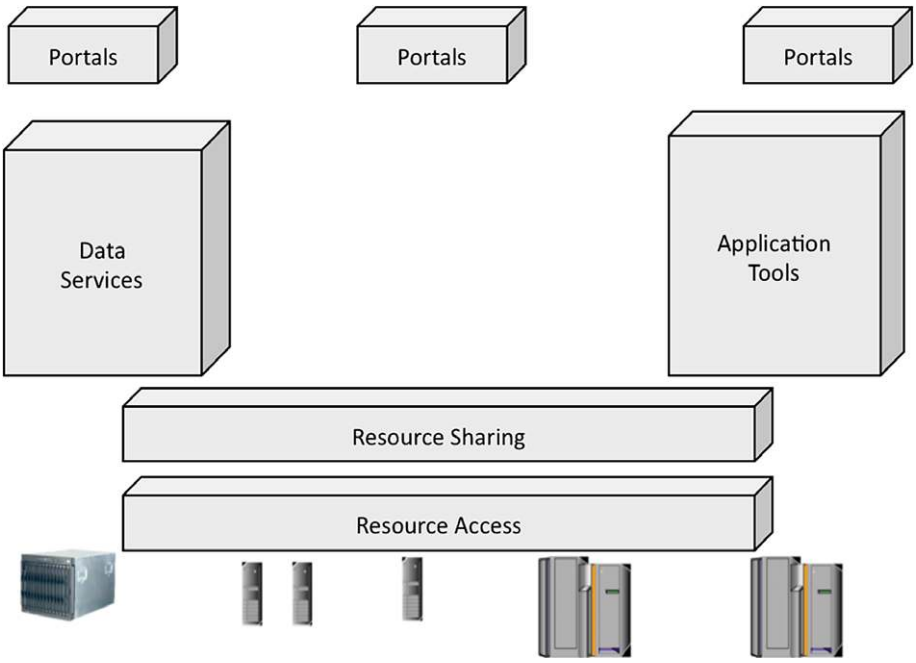

Fig. 1. Common components in current cyberinfrastructure environments.

ploitation of these facilities, including remote access services, interface portals, and data and tool repositories. Figure 1 illustrates at a very high level some of the common components in cyberinfrastructure.

There is no question that cyberinfrastructure is enabling ever-more integrative and transformative science. Today, many scientific collaborations exploit cyberinfrastructure to create sophisticated simulations for earthquakes (www.scec.org), to extract new results from astronomical or particle physics data (www.ivoa.net, milkyway.cs.rpi.edu, www.ligo.caltech.edu), to study ecological and environmental change (www.neoninc.org, www.oceanleadership.org), and to conduct biomedical research (www.birn.org, cabig.nci.nih.gov) among many others. Visionary roadmaps in almost every scientific discipline build on existing cyberinfrastructure to include increasing levels of automation and support for scientific research [35,37,50].

Current cyberinfrastructure has proven effective to tackle two major challenges: scale and distributed sharing. In terms of scale, it enables computations that are beyond terascale and into petascale arena, soon to be in exascale levels. In terms of distributed sharing, the collaborations just mentioned attest to community-wide sharing and access of varied resources including data, instruments, computation and storage.

There are some important questions though that have been recently brought up by the research community in terms of effective exploitation and use of cyberinfrastructure.

The National Science Foundation's Cyberinfrastructure Council released the NSF Cyberinfrastructure Vision for 21st Century Discovery in March 2007 [37], which included the following observation:

> While hardware performance has been growing exponentially – with gate density doubling every 18 months, storage capacity every 12 months, and network capability every 9 months – it has become clear that *increasingly capable hardware is not the only requirement for computation-enabled discovery* (NSF Cyberinfrastructure Vision for 21st Century Discovery, March 2007).

This is indeed a key question. Given the ever-increasing availability of computational resources and their effective integration into grids and collaboratories, there does not seem to be a corresponding acceleration in the pace of scientific advances. The document went on to state five major challenge areas: cyberinfrastructure planning and sustainment, data and visualization, virtual organizations, and learning and workforce development. All five areas are clearly important and deserve major investments. Interestingly, an alternative perspective was put forward by another NSF report [8]:

> A key motivating question posed by domain scientists was: *Given the exponential growth in computing, sensors, data storage, network, and other performance elements, why is the growth of scientific data analysis and understanding not proportional*?
> There was a broad consensus in the group that in the scientific community there is a perceived importance of workflows in accelerating the pace of scientific discoveries. Today, complex scientific analyses increasingly require tremendous amounts of human effort and manual coordination. Data is growing exponentially, but the number of scientists is roughly constant. Thus researchers need exponentially more effective tools to aid in their work, if they are not to be inundated in data and associated tasks. *Workflow environments that support and improve the scientific process at all levels are crucial if we are to sustain the current rapid growth rate in data and processing* (NSF Workshop on Challenges of Scientific Workflows, October 2006).

The argument is that capturing scientific analyses explicitly in declarative data structures known as workflows will enable the development of new aids to scientists for coping with the scale of the new computational environments. Workflows represent complex compositions of software components and the dataflow among them. Workflow systems can then support scientists by automating low-level aspects of the process, providing detailed records of each analysis and its products, and enabling rapid reuse of software compositions. Perhaps a more pressing need in current cyberinfrastructure that was raised in that report results from a perceived threat to the scientific method in research involving complex computations:

> An important requirement is reproducibility of scientific analyses and processes. This requirement is at the core of the scientific method, in that it enables scientists to evaluate the validity of each other's hypothesis and provides the basis for establishing known truths. Reproducibility requires rich provenance information, so that researchers can repeat techniques and analysis methods to obtain scientifically similar results. *Today, reproducibility is virtually impossible for complex scientific applications.* First, because so many scientists are involved, the provenance records are highly fragmented, and in practice they are reflected in a variety of elements including emails, Wiki entries, database queries, journal references, codes (including compiler options), and others. All this information, often stored in a variety of locations and in a variety of forms, needs to be appropriately indexed and made available for referencing. *Without tracking and integrating these crucial bits of information together with the analysis results, reproducibility can be largely impractical, and more likely impossible, for many important discoveries involving complex computations* (NSF Workshop on Challenges of Scientific Workflows, October 2006).

Capturing the scientific analysis process in a declarative manner so that they can be easily reproduced by other groups or replicated on other datasets, also leads us into looking at workflows as an important and missing element in cyberinfrastructure.

The next section introduces workflows, the capabilities that workflow systems are already contributing to scientific computing, and the benefits that result from using workflow environments in science projects.

3. Workflows and workflow systems

Scientific applications can be very complex as software artifacts. They may contain a diverse amalgam of legacy codes, compute-intensive parallel codes, data conversion routines, and remote data extraction and preparation. These individual codes are often stitched together using scripting languages that specify the data and software to be executed, and orchestrate the allocation of computing resources and the movement of data across locations. To manage a particular set of codes, a number of interdependent scripts may be used. Scripted scientific applications are common today in cyberinfrastructure environments.

Although scripted applications provide an approach to specifying and managing computations, there are major drawbacks to their adoption to manage complex scientific software. First, any modifications are costly and error prone. Small routine changes such as adding a new code or a new version of an existing code requires walking through the scripts manually and making changes where appropriate. Adding new requirements could require major changes to a significant portion of the scripts. Second, they require a significant amount of human intervention to specify ad-hoc data and execution management. Although cyberinfrastructure services may be available to determine available

execution and storage resource, they only facilitate the task because the scripts must still manage the resource allocation and data location specifications. Third, any execution failures require manual intervention for recovery and finding a good point to relaunch the script without repeating expensive computations that were successful. Fourth, the scripts must be undergo significant changes to run the computations using a different set of hosts or datasets. Fifth, these scripts typically include a significant amount of code to assemble and record metadata and provenance information about the results of the computations. This code is also error-prone, costly, and hard to evolve. Last but not least, scripting languages are programming languages and as a result are inaccessible to any scientists without computing background. Given that a major aspect of scientific research is the assembly of scientific processes, the fact that scientists cannot assemble or modify the applications themselves results in a significant bottleneck. All these reasons point to the need for better management of complex scientific computations than the commonly used approach of relying on scripting languages.

Workflows have emerged as a useful paradigm to describe, manage, and share complex scientific analyses [8,14,47]. Workflows represent declaratively the components or codes that need to be executed in a complex application, as well as the data dependencies among those components. Workflows have been used for several decades not to express computations but to represent complex processes in human organizations [5,32] that reflect tasks and the flow of information among them, as well as people and resources involved throughout. Here we refer to workflows instead as compositions of computational steps, although one can envision workflows for science applications that combine manual and computational steps.

Some scientific workflows represent compositions of remote services. They specify how to use services provided by third parties to accomplish an overall task. Other scientific workflows combine software components as codes that can be submitted for execution to different remote resources. Some of these codes can be legacy applications, and the workflow expresses how to combine their results into a new end-to-end application.

Workflow systems exploit workflow representations in order to manage the elaboration and execution of workflows in a distributed environment. Several workflow systems have been developed for a variety of applications, including Askalon [51], Cactus [20], Kepler/PtolemyII [31], Pegasus/Wings [6,9], Taverna/myGrid [16,21,38], Triana [48] and Wings [26]. Surveys and overviews of current workflow systems are provided in [47,53]. In this paper we will use Pegasus, Taverna and Wings to illustrate the capabilities and benefits of workflow systems.

Pegasus manages mapping and execution of computational workflows in distributed shared resources that may be highly heterogeneous [6,9]. Mapping involves selecting execution resources for each workflow task. Execution management includes handling new data products and recovery from execution failures. To map workflow tasks, Pegasus uses descriptions of the execution requirements of each of the codes, and finds available hosts in the execution environment that satisfy those requirements. It takes into account queuing times in selecting among suitable resources, and clusters together workflow tasks into a single queued job to improve execution performance. Pegasus also manages new data generated by the workflow, moving it to the location of the next workflow task that will use it and registering results in data catalogs. To manage very large datasets reliably and efficiently, it uses grid services for data transfer and for finding alternative locations of data replicas. Pegasus includes several algorithms for optimizing the selection of execution resources not only based on task performance but also on minimizing queuing delays and data movement times. Another optimization strategy is the reduction of computations by eliminating workflow tasks that generate data that already exists and can be reused, perhaps generated by the prior execution of workflows. Pegasus relies on Condor DAGman and Condor-G [49] to submit jobs in the order specified by the workflow dataflow. Pegasus has also facilities to recover from execution failures that may occur due to bugs in the application codes, memory faults in the execution host, network failures, and other unexpected errors that are commonplace in distributed architectures. When a computation fails, it is retried a few times and then submitted to an alternative resource. If nothing works, DAGman returns a rescue graph that is used by Pegasus to figure out what portions of the computations to resume.

Pegasus is used in several cyberinfrastructure projects. For an application of the Southern California Earthquake Center (www.scec.org/cme) for seismic hazard analysis, Pegasus mapped workflows to heterogeneous shared cyberinfrastructure resources in NSF's TeraGrid (www.teragrid.org), managing more than 260,000 tasks for a total of 1.8 CPU years of computation that generated 20 TB of data in 23 days [7].

Montage is perhaps the most successful deployment of Pegasus, where it is used by a broad community of astronomers [2,25]. Montage is part of the National Virtual Observatory (www.nvo.org) and is used to create science-grade mosaics of the sky from multiple images that may have different characteristics (e.g., different coordinate systems, projection, etc.). Montage includes several application codes for re-projection into common scale and coordinates, modeling background radiation to minimize inter-radiation differences, rectification into common flux scale, and co-addition into a final mosaic. Montage can process data using two alternative approaches: one is a system that parallelizes computations implemented as a message passing interface (MPI) code that can be executed in a cluster, and the other uses Pegasus workflows to parallelize computations and execute them on distributed resources. Detailed comparisons showed that there is no notable difference in the execution performance of these two approaches, and that Pegasus has the additional advantages of fault tolerance and computation management [25]. Pegasus improved runtime by 90% through automatic workflow restructuring and minimizing execution overhead [2].

Taverna [16,21,38] focuses on workflows for bioinformatics applications. In this area, there are thousands of services that are made available over the network for access by a wide community of scientists. The mechanism to access the services varies, some are web services, others are REST services, and others are simply legacy command line applications. Taverna provides a framework to integrate these components and isolates users from this diversity of access mechanisms. Taverna workflows are composed from these services, and are cast in a simple and intuitive workflow language. An important challenge for integrating these services is that they are advertised with simple descriptions that provide no semantics as to what inputs they expect and what outputs they produce. To address this, workflows may include small steps or shims for data format conversion. To execute a workflow, Taverna uses the FreeFluo engine to add the specific invocation details for each of the services. FreeFluo also includes mechanisms for failure recovery, so that when a service fails it looks for an alternative location for the same service and retries the invocation there. Taverna includes more than a thousand diverse services such as the European Molecular Biology Open Source Software Suite (EMBOSS), BioMOBY, the Kyoto Encyclopedia of Genes and Genomes (KEGG) and the National Center for Biotechnology Information (NCBI), totaling more than 3,000 services in 2006 [21]. Taverna workflow applications have executions that range from a few seconds to a few days, and do not require handling large-scale datasets. A recent result obtained with Taverna is the identification of a candidate gene thought to be responsible for resistance to African tripanosomiasis [10]. The workflow looks for correlations between phenotype in microarray data to Quantitative Trait Loci (QTL) genotype data. The paper argues that when this kind of correlation is done manually there is no guarantee of a systematic consideration of hypotheses due to several features: (1) eliminated datasets prematurely to reduce complexity, (2) hypothesis-driven research dominates rather than complements data-drive research, (3) user bias in pursuing hypotheses, (4) reanalysis of data is hard due to changes in software interfaces and data availability, (5) errors due to all the above. The workflow provides a mechanism to systematically and correctly explore variations of parameter settings. In addition, it is possible to re-analyze data since the provenance of any result is made available and the workflows are easily re-executed.

These results illustrate key benefits of workflow systems:

Automation of workflow execution: Data management and execution are automatically handled, including mechanisms for failure recovery and repair. Failures during execution can be handled automatically since the workflow system can figure out what computations remain to be done and where prior computation results reside in the execution environment. Executing the same workflow in a new execution environment becomes trivial, as it is a simple matter of assigning computations to the new resources and this is done automatically.

Managing distributed computations: Whether submitting computing jobs to remote hosts or invoking third party services, workflows manage computations in a distributed environment. Since failures are commonplace and failure recovery can be complex in distributed systems, manual management of distributed applications becomes impractical and is better handled by workflow systems.

Managing parallel computations: Scientific applications often benefit from parallelism, whether to process large datasets efficiently by farming out subsets to different resources or by accessing distributed services concurrently. Work-

flow structures for scientific applications represent parallelism in the dataflow graph and their efficient concurrent execution is automatically managed.

Systematic exploration of the parameter space: Application parameters can be explicitly indicated in the workflow. This enables the systematic assignment of values to explore the space of possible parameter combinations.

Managing the evolution of an application: Workflow applications are modular by design, and as a result the evolution of the application is more manageable. Updating individual components either has little impact in the overall workflow or the impact is localized and amounts to updates to the dataflow structure. With scripts, the overall code has to be changed particularly the metadata propagation, no matter how small the change to an individual component.

Provenance recording: Metadata and provenance information are automatically recorded by the workflow system. When expressing a new application as a workflow, no special code has to be written to record provenance.

Low-cost high-fidelity reproducibility: Workflows provide explicit representations of the computational processes used to derive new data. When a significant result is achieved, there is a detailed provenance trail of what processes were executed and how parameter values were set to obtain those results. Every detail of the provenance of new data products can be recorded and supplied by the workflow system. Workflows can be easily re-executed to reproduce results, and can be easily applied to new datasets to replicate results in alternative settings. When new datasets become available in a shared environment, it is easier to replicate a computational experiment with the new data.

Workflow systems have already demonstrated the benefits of automatic management of computations. If adopted broadly in cyberinfrastructure environments they have the potential to greatly streamline the productivity in computational experimentation processes and accelerate the pace of scientific research, since they can result in significant savings in terms of human time and effort spent in computational experiments. Perhaps more importantly, workflow systems could have a profound impact in reproducibility of computational experiments. Figure 2 shows workflow systems augmenting the common components of cyberinfrastructure.

The next section argues that workflows will be indispensable to support new capabilities envisioned in scientific roadmaps being laid out in many sciences, and that they open the door to new possibilities in cyberinfrastructure to support scientific discovery.

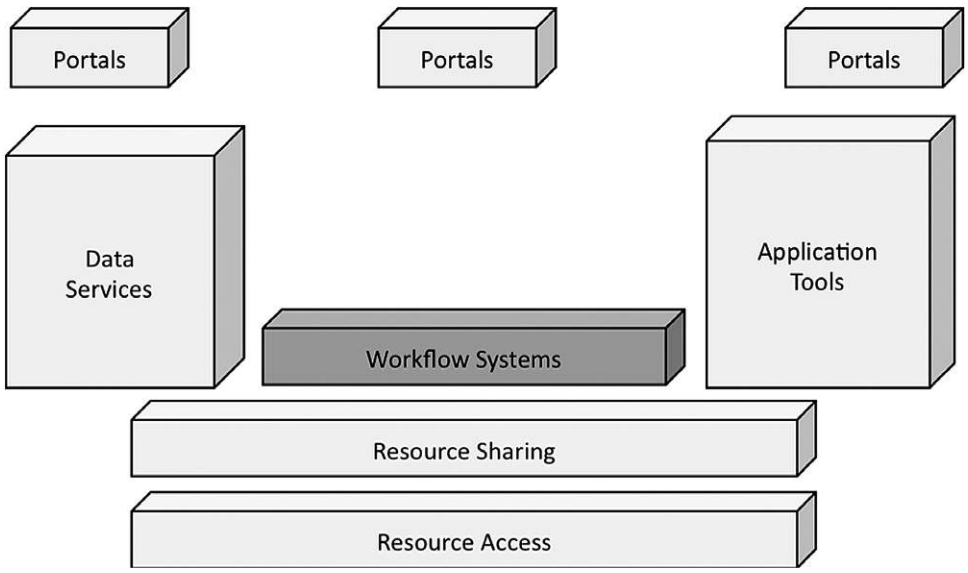

Fig. 2. Workflow systems as components of cyberinfrastructure.

4. Towards large-scope science: The need for a knowledge level view

The coming decades will present great opportunities for scientific discovery on questions encompassing complex natural phenomena that science was not even in a position to pose until now. Of paramount importance to pursuing such questions is breaking the barriers across insular research disciplines to enable increasingly integrative scientific pursuit [50]. From neuroscience to cancer research, any aspect of biomedical research is increasingly viewed as a system of systems science that requires integration of data and models across a variety of disciplines [35]. Physician's observations and data must be integrated with models at the cellular level, the organ level and the system (e.g., circulatory and nervous systems) level. Environmental science is another example of the study of a complex system of systems requiring interdisciplinary integration. The chair of the Science Council of the NSF's 30-year old US Long Term Ecological Research (LTER) network describes the vision for environmental observatories that produce data that can be integrated and analyzed across perspectives and disciplines:

> The importance of self-organizing networks of environmental scientists for identifying and addressing the non-linear and cross-scale phenomena that underlie and, in some cases, define global environmental change today. [...] With the emergence of new complementary networks, such as the National Ecological Observatory Network (NEON), the Global Lake Ecological Observatory Network (GLEON), the Water and Environmental Systems Network (WATERS), and the Oceans Observatory Initiative (OOI), comes the potential for research synergies hardly imaginable even 15 years ago. Equal in importance to collaborations across physical networks are collaborations across disciplinary networks. *If there is one lesson to be learned [...] it is the crucial importance of engaging with other disciplines – and especially with the social and behavioral sciences – to address today's big ecological questions* [41].

This cross-disciplinary view on data analysis may be best described as *large-scope science*:

> Whereas large-scale means increasing the resolution of the solution to a fixed physical model problem, *large-scope means increasing the physical complexity of the model itself. Increasing the scope involves adding more physical realism to the simulation, making the actual code more complex and heterogeneous*, while keeping the resolution more or less constant [42].

This emphasis on large-scope science is in contrast with large-scale science, which has been a major driver to date of cyberinfrastructure research. Large-sale science can be pursued through increasingly more powerful networks and machines, parallel distributed computing techniques, and federated services. But although in fact current cyberinfrastructure manages complexity and heterogeneity, its focus is at the level of hardware resources and services. The complexity and heterogeneity required for large-scope science speaks of a new realm that is not addressed by current cyberinfrastructure. The new challenges that we face are concerned with the diversity of models and the complexity of the methods involved in cross-disciplinary research. There are essentially two levels of concern here: one about how the applications are integrated and another about how the resources are integrated. It is a key division between the behaviors desired and mechanisms used to obtain them. The behaviors depend on the knowledge available in the system to perform a task. The mechanisms are important in that they implement those behaviors, but they are irrelevant to the task of the system in the sense that the choice of mechanism does not affect the system's behavior.

This distinction is crisply expressed in terms of the knowledge level versus the symbol level in artificial intelligence:

> The knowledge level hypothesis: There exists a distinct computer systems level which is characterized by knowledge as the medium and the principle of rationality as the law of behavior [36].

The knowledge level of an intelligent system is concerned with any characterization of that system in terms of its response to requests or goals and what knowledge it uses to solve them. In contrast, a symbol level is concerned with the implementation of the knowledge and the reasoning mechanisms that are used to exploit it. For example, a symbol-level description would characterize a system in terms of whether it uses a genetic algorithm, a neural network, or a rule base. An example of a knowledge-level description would describe an autonomous vehicle in terms of its ability to pursue standing goals of going to a destination, to incorporate opportunistic goals when a lane opens, and

to defend itself from other drivers through fast reactive behaviors.

If we take this distinction to a workflow environment, we can see that the capabilities of workflow systems to map and execute workflows are concerned with the architecture at the symbol level. The scale and sharing are enabled by the symbol-level architecture through infrastructure services and resources. The symbol level is concerned with carrying out the tasks specified in a given workflow. In contrast, the knowledge level of a workflow system would be concerned with the kinds of tasks that it is able to accomplish for a scientist. This suggests a level of workflow descriptions and capabilities that affect what scientific tasks the workflow system can accomplish. This level would be concerned with what scientific tasks it can undertake, what workflows are selected for a task, what workflows are available in the system, and what their coverage is with respect to a set of tasks. The more knowledge, the more kinds of tasks the system can undertake. More knowledge about how to use and integrate workflows will result in improved behavior of the system in terms of solving more tasks and being capable of producing new kinds of results.

Thinking about workflow systems as repositories of scientific knowledge, we can then explore techniques for managing the heterogeneity of that knowledge and the capabilities required to perform complex tasks using that knowledge.

5. Workflows at the knowledge level

What would it mean to describe workflow systems at the knowledge level? What kinds of behaviors should we expect workflow systems to accomplish by using that knowledge? Table 1 shows a set of abstraction layers in the specification of workflows, from more abstract to more specific. A more abstract layer can be implemented by using the information contained on the layer below it. Typically, workflows specify the datasets and the computational steps (services or codes) that are to be used. Those correspond to layers 2 (data) and 1 (computation), which specify the data that will be processed and when it will be processed. Layer 1 workflows are then mapped to execution resources, resulting in layer 0 workflows that are directly executable in the execution environment. Layer 0 workflows correspond to scripted applications, and layers 1 and 2 workflows correspond to the workflows discussed in Section 3.

Going up on layers of abstraction in Table 1, a workflow can be described not by the specific computations but by a sketch the process by specifying abstract classes of computations and by skipping some of the workflow steps to be performed if they are not central to the experiment. For example, a workflow could indicate that an initial dataset is first processed with a normalizing step followed by a discretization step and then a clustering step without specifying which algorithms and implementations are to be used. These workflows are layer 3 workflows that specify how data is to be processed but not specifically when each operation will be carried out relative to others. At a higher layer of abstraction, only a description of the desired results would be specified. For example, the desire to obtain clusters of temporal sightings of bird observation data. At the highest layer of abstraction, only questions are posed and no details are provided about how to find answers to the questions in terms of workflows to be executed or data to be generated. For example, what would be interesting patterns for bird observation data.

Table 2 relates these layers of abstraction to the knowledge level and the symbol level. We should aim to develop systems that can take on workflows and requests at the highest layers of abstraction from users, and then have the systems automate the elaboration of the workflow into the lower layers of abstraction and their corresponding details. The higher the abstraction layer, the closer the workflow representation is to how a scientist may view the process or the request that triggers the process.

Table 1
Layers of abstraction in workflow descriptions

Layer of abstraction	Information specified	What/When/Who/How/Where
(4) Result	Desired data products	*What* result is desired
(3) Method	High-level processes	*How* will data be processed
(2) Data	Input datasets	*Who* (what data) will be processed
(1) Computation	Specific computational steps	*When* will data be processed
(0) Execution	Specify execution resources	*Where* to execute the computations and where to find data

Table 2
The knowledge and symbol levels in workflow descriptions

Level of system description	Abstraction layer	What/When/Who/How/Where
Knowledge level	(5) Question	*What* would be an interesting result
	(4) Result	*What* result is desired
	(3) Method	*How* will data be processed
	(2) Data	*Who* (what data) will be processed
Symbol level	(1) Computation	*When* will data be processed
	(0) Execution	*Where* to execute the computations and where to find data

The highest layers of abstraction are centered around what behaviors the system can exhibit, and the knowledge required to accomplish those behaviors. Knowledge will include constraints that must be satisfied by a workflow in order for it to be valid, strategies to complete or specialize a high-level workflow, effects-centered knowledge to accomplish a given experimental goal, and descriptions of data and their characteristics. Techniques would include constraint reasoning, hierarchical decomposition and abstraction reasoning, automated search, heuristics that focus exploration of possibilities, and ontology-based reasoning of classes of data and computations.

In considering the knowledge level, we leave behind the realm of parallel programming and distributed systems. We enter the realm of artificial intelligence as an enabler of significant new capabilities in workflow systems. Artificial intelligence techniques can play an important role to represent complex scientific knowledge, to automate processes involved in scientific discovery, and to support scientists to manage the complexity of the hypothesis space. The next section illustrates some of these techniques for workflow generation assuming initial descriptions of user requests at the highest levels of abstraction.

6. Reasoning with workflows at the knowledge level

Armed with knowledge of what workflow components do, what the properties of the datasets are, and what experiment design entails, workflow systems can assist scientists by exploiting that knowledge to make automatically domain-relevant decisions.

Wings is a workflow system that starts with high-level user descriptions of desired analyses and uses knowledge about components, data, and workflows to automatically elaborate, validate, and generate workflows to the level of detail that Pegasus needs to map and execute them [12,13,26,27]. Wings assumes that all workflow components, data, and their properties can be organized in hierarchies, and that they can have associated constraints regarding their proper use. It allows the expression of high-level workflow templates that can be reused for different datasets, and represents constraints among datasets and components at the workflow level. Wings represents this knowledge using ontologies and rules, and uses the W3C Web Ontology Language (OWL) (www.w3.org/2004/OWL) and associated reasoners as the basis for workflow representation. Workflows can be expressed at a high level with component classes, and express iterations in a compact manner. Wings uses OWL reasoners to find out whether a component can be used to process a dataset with given properties, to find out whether a component can generate datasets with certain properties, and to check if data can flow between two components based on their respective constraints. Wings issues many such queries as it assists the user to generate workflows. If a scientist is creating a workflow interactively, Wings checks that all the dataflow is consistent with the component constraints. When it is not, it makes suggestions regarding what other components can be substituted to achieve a similar function while respecting the constraints [28]. When input data is selected, it checks that its properties comply with the requirements of the components that will process the data. Once a workflow is specified, Wings elaborates it to generate a complete description of the workflow in a format that can be submitted to Pegasus, which includes command line invocations of each executable code and logical names for all the datasets in the workflow.

In addition to workflow validation, Wings can automatically select components and datasets for the scientist. Wings can select datasets based on the constraints expressed in the workflow, and if several datasets satisfy the constraints then several workflow options will be generated. When a component class is used as a step of the workflow, Wings will choose specific components based on the constraints that apply to that step in the workflow. When several components and

datasets satisfy the constraints in the workflow, it will generate all corresponding workflow candidates and execute the top-k workflows ranked according to user-specified criteria (e.g., faster execution, higher accuracy, etc.). Wings can also automatically generate workflows based on descriptions of what data products are desired by a user. Wings propagates the requirements on data products throughout the workflow using a backward projection [12]. This results on a set of constraints on the input datasets that are used to query data catalogs to find appropriate input data. It then finds the properties of the input data found and propagates them using a forward projection that enables it to generate detailed descriptions of the new data products generated by the workflow [12,27]. This is a very useful capability, as the system can register new datasets obtained through execution and annotate their metadata properties so they can be discovered and reused later to avoid repeating unnecessary and costly computations.

For a seismic hazard analysis application of the Southern California Earthquake Center (http://www.scec.org/cme), Wings was used to expand a workflow template containing a dozen application codes, including MPI codes, into a workflow of more than 8,000 computations [13,27]. Pegasus expanded this workflow to add data movements and registrations for a total of 24,135 jobs. The workflow processed an earthquake forecast model with thousands of possible fault ruptures for a total of 110,000 input files, and run for 1.9 CPU years. Wings generated provenance records for 100,000 new data products.

Taverna also uses knowledge-rich descriptions of components and workflows [17–19]. It matches user requests with available workflows and services. Users can specify the type of service they wish to use, or the type of workflow structure specified as a graph of services and their dataflow. Taverna uses semantic descriptions of services and workflows combined with graph matching algorithms to discovered appropriate workflows for the user.

These results illustrate key additional benefits of adding a knowledge level to workflow systems:

Automation of workflow generation and of repetitive constraint checking tasks: During the generation of even simple workflows, a generation algorithm can formulate hundreds of queries about components and dozens of queries to check constraints about datasets based on their use in the context of the workflow. Scientists should not need to check by hand the myriads of constraints about components and datasets that must be taken into account within an analysis. The system can undertake these kinds of repetitive tasks because it has knowledge about experimental processes (workflows), models (workflow components) and data.

Systematic exploration of the experiment design space: A workflow system can explore in a methodical and exhaustive manner all possible experimental settings for a workflow: all possible combinations of components, all possible relevant data, and all possible parameter settings. Invalid combinations will be automatically (and correctly) ruled out as inconsistent within the context of the workflow. While at a lower layer of abstraction in the symbol level one could explore the space of different parameter settings, the knowledge level is needed to enable this systematic exploration in terms of all workflow configurations that use alternative components and data.

Validation of workflows: Given a user-specified sketch of a workflow, the system can use its knowledge of components and data to ensure that the workflow is valid. Components represent models or operations on data that are designed to work on certain kinds of data, and when composing these models together it is hard for scientists to have fresh in their mind all the constraints on all the models. Even when scientists are intimately familiar with the constraints on those models as published in the literature and code documentation, it is hard to keep track of all of them when the compositions are complex and when they evolve over time.

Automated generation of metadata for new data products: Because the system has knowledge-level descriptions of the kinds of transformations performed on datasets, it can use these descriptions to qualify the properties of new data products. At the symbol level, the description of new data products is limited to what software and input data were used as shown in the provenance records.

Guarantees of data pedigree: The system can include knowledge about well-formed widely accepted workflows that can be directly reused on new datasets. A scientific method that is well tested and widely accepted by a community can be captured in an earmarked workflow that can be referred to as a proof of pedigree of results

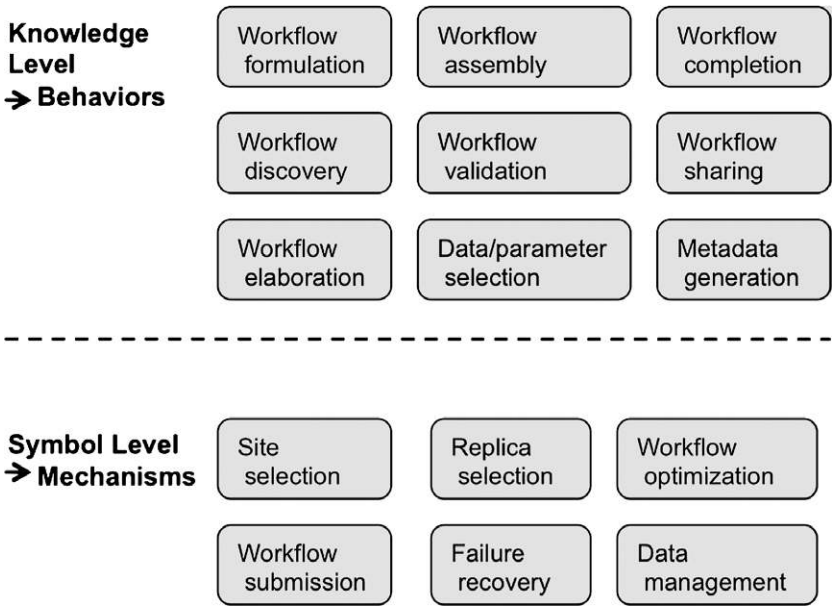

Fig. 3. Functions within a workflow system that form the knowledge level to determine behavior and the symbol level to provide mechanisms.

obtained by it. That is, when the provenance of new data products shows that they were obtained through a highly regarded workflow that serves a guarantee of the high quality of the process used to obtain them, or pedigree, of those new results. This would bypass the current need to check how any surprising results are obtained, either when they look too good or when they look too routine, as the surprise is often times due to errors that lead to incorrect code selection or parameter settings. These workflows provide a guarantee that any results obtained from those workflows comply with vetted methods and their requirements. At the symbol level, the system can offer a provenance trail of how new results were obtained. At the knowledge level, the system can offer a guarantee of trusted provenance or *pedigree* of the new results.

Correct reproducibility and reuse: At the symbol level, workflows can be reused to reproduce results with new datasets. However, at that level only syntax validity can be checked. At the knowledge level, the constraints of the components and data of a workflow can be checked to ensure its correct reuse.

Clearly the knowledge level has benefits that speak directly to large-scope science in terms of managing complexity and heterogeneity through automation of workflow tasks that are closer to the science realm than what was possible in the symbol level. Note that these benefits are in addition to the benefits that we discussed for the symbol level of workflow systems architecture.

There are many possibilities for the knowledge level once it is in place in a workflow system. Here we discussed automatic template-based generation and completion of workflows. Other possibilities to improve automated workflow generation include hierarchical decomposition of tasks in a workflow, selection among software implementations of workflow components based on available execution resources, and dynamic selection of components interleaved with execution based on results obtained from execution of prior steps.

Figure 3 summarizes the functions within a workflow system discussed here contrasting the knowledge level with the symbol level.

7. Looking to the future: From data to knowledge to discoveries

We discussed many additional benefits of having a knowledge level in the architecture of workflow systems. The discussion centered on benefits arising from the ability to automate the generation and validation of workflows from high-level requests. This section argues that the potential for the knowledge level is enormous in terms of significant paradigm shifts in the way computational experimentation is practiced and outlines areas for future investigation. Figure 4 illus-

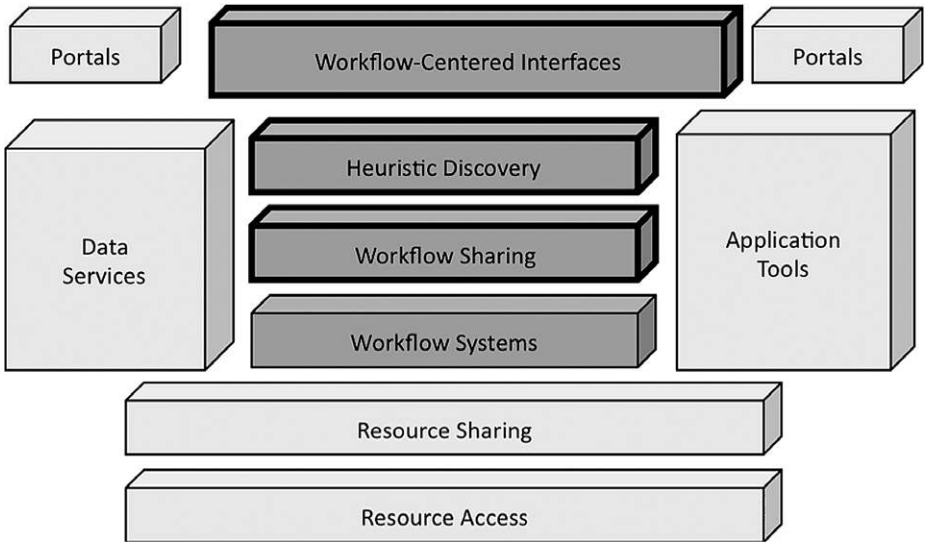

Fig. 4. A view on future cyberinfrastructure components.

trates the potential in terms of significant new cyberinfrastructure capabilities.

7.1. *Workflow as scientific currency*

Workflows are valuable in their own right as scientific research products. Workflows should be the objects of scientific discourse, and their description should be used to capture formally a novel method or analysis process discovered through careful design and testing. Workflow design is a contribution to science in its own right, in fact new methods are publishable in scientific articles and those articles could be accompanied by the formal workflow description as supplementary information to the article that describes the workflow in textual form. For workflows to become scientific currency, workflow descriptions need to become closer to the knowledge level and therefore closer to representing scientific concerns rather than low-level system concerns.

Workflows should become currency of scientific exchanges. Where today we see sharing of data across entire communities, tomorrow we should see workflows being published and exchanged across research groups. Where today we see citations to papers that explain the scientific method used to obtain a result, tomorrow we should see citations of workflows that should be downloadable and inspectable and reproducible at minimal cost. Where today we see common use of portals to access datasets, tomorrow we should see the common use of workflow libraries to access and to contribute workflows.

Workflows could be shared as computational objects much like data is shared today across scientific communities in cyberinfrastructure [15,45]. Unlike data, workflows can evolve over time as new or faster methods are discovered. A workflow may be superseded by a new one, and if so any results obtained with the former would be worth revisiting using the new workflow. A user community could drift from preferring the use of a workflow template to a new workflow that represents an improved or newly created method. It will be important to manage the evolution of workflows as the experimental methods that they represent evolve while being used by a community of scientists.

A related and important area of future research building on the knowledge level is learning from workflow usage in order to improve and adapt the behavior of the workflow system. Rather than expecting users to define reusable templates by hand, a system could learn reusable workflow templates from observing regularities and generalizing traces of workflows executed by a community of users. A workflow system could also learn component and data selection criteria based on what workflows are found most useful by a user community. Workflow patterns that may appear repeatedly in the context of certain types of data analysis could be discovered autonomously by the system by observing usage of workflows over time. One could envision that the learned workflows could ultimately result in new discoveries made by the workflow system, and could be scientific contributions made by the system in its own right.

7.2. Workflows for cross-disciplinary integrative research

Large-scope science requires managing heterogeneity across disciplines. Consider the environmental observatories mentioned in Section 4. There is a clear cross-disciplinary data analysis activity expected of the various scientific communities involved in analyzing the data collected. For example, sensor data regarding weather trends will need to be coupled with ornithology migration data to discover significant correlations between environmental conditions and animal behavior. An ornithologist would very likely not be familiar enough with weather analysis methods to set up valid workflows. However, the system could act as an expert weather analyst and assist the ornithologist by automatically generating workflows from given general questions about general weather patterns known to ornithologists. Although this is an example of chaining workflows together, one could imagine arbitrary interweaving of workflows created by researchers in different disciplines into a complex and heterogeneous cross-disciplinary analysis.

The existence of a knowledge level to reason about behaviors within disciplines would make it possible to envision systems that will reason about behaviors that cross-disciplinary boundaries. These are areas of the research space that will very likely lead to fundamentally new discoveries. These are also areas of research that are precisely the motivation of developing cross-disciplinary research programs such as the environmental observatories in the first place.

7.3. Workflows for education and broadening participation in science

Science must be an ecosystem in order to foster discoveries and innovation across the board. Researchers and educators at all levels are needed to push science forward, from expert Nobel-quality talent to the most inexperienced undergraduate research assistants, from faculty retreats to facilitate inter-disciplinary collaborations to high-school teachers that form future generations of scientists through hands-on involvement in science, from the most prestigious university departments to the humblest of corporate research laboratories and startups. Workflows have enormous potential as a paradigm to facilitate training across this science research ecosystem. Workflows can illustrate methods, data usage, results and processes in a hands-on manner to complement the general or theoretical descriptions found in articles and textbooks. The knowledge-level will enable the presentation of workflows in domain-relevant terms found in those articles and books that is not possible with the symbol level alone.

Just as important as supporting the training of future generations of scientists is the support to train seasoned scientists on new techniques and methods or new areas of research. The need for cross-disciplinary training is already commonplace in any scientific practice. Cross-disciplinary training is already costly to individuals, and there is very little technology for hands-on practice of another science's methods and analyses processes.

One could imagine ultimately opening up science to a much broader population than our current scientist and student pool. Pioneering efforts to volunteer compute cycles have been very successful (setiathome.berkeley.edu, milkyway.cs.rpi.edu). There are already significant projects that rely on citizen scientists to collect data [3,4,33,40] spanning astronomy (www.galaxyzoo.org), ornithology (www.ebird.org), botany (www.windows.ucar.edu/citizen_science/budburst) and weather (wxqa.com). A workflow system could autonomously create combinations of workflows and data that are separately contributed to the system and that may be worth analyzing. Citizen scientists could volunteer their skills to accomplish real scientific analysis tasks by being trained by or assisted by underlying workflow systems.

7.4. Workflows for systematic exploration and discovery

Today's paradigm for computational experimentation is driven by the user's initiative, design choices, and experiential biases. Scientists decide what software to run and with what settings, what data to analyze and with what granularity, and what aspects of the hypothesis space to focus on. Scientists have unquestionable expertise to drive the process, but there are limits to the effort, reliability and coverage of any human-centered task of the complexity required in current and future scientific endeavors. Humans should not be the bottleneck to scientific advances when routine tasks can be automated. We have already seen the benefits of assistance and automation through workflows, but much more can be done.

Workflows can be used to automate processes for heuristic discovery and pattern detection. Through systematic hypothesis generation and elimination, workflows can explore ever more complex phenomena. Scientists today rely on visualizations to understand com-

plex datasets, but there is a limit to what can be visualized for complex phenomena. Pattern detection techniques can search datasets to match patterns (or pattern types) that describe complex relationships across variables. Heuristic-driven search can automatically discover new correlations in datasets. The process would still be driven by the scientist and still be human centered, in that the scientist can provide a battery of potential patterns to seek or heuristics to follow. There are already many examples in the AI literature of scientific discoveries driven by heuristic and pattern search [29, 30,39,43,44]. Workflows today include sometimes visualization steps that plow through very large datasets to highlight specific aspects. Workflows for pattern detection and discovery should be developed to process large data sets and extract phenomena of potential interest. Knowledge level descriptions of the analysis processes are needed to enable the integration of pattern descriptions and heuristics at the appropriate level of the scientific domain.

7.5. Workflows as paradigm for "research cockpits"

Today's scientific environments contain shared distributed resources that are accessible through web portals. Portals are user interfaces that act as a single point of access to data collections, application tools, services, and other resources. Portals are customized to specific purposes or disciplines, and guide users to conduct pre-defined tasks through scripted interfaces. Deviations from the pre-defined system behaviors are not supported.

In coming years, new user interface paradigms will need to be developed for computational experimentation. The underlying system must be able to support flexible behaviors and be configurable by end users. Scientists will conduct long-lasting activities, and the interfaces must be designed to track the information flow over time and to accommodate the dynamic evolution of such activities. More importantly, the user interfaces must be designed to support collaboration not only among humans but between humans and the underlying system and the ongoing activities that must be accomplished jointly.

Aircraft cockpits are a great analogy for the kind of user interface that will be required. The organization of the tools and workspace must support collaboration among several humans (e.g., the pilot, the first officer and the second officer) and the aircraft navigation system, showing how the humans and the aircraft manage the flow of information among them to jointly accomplish the mission [22–24]. Cockpits organize information in a task-centered manner, enable several humans and the system to work as one cognitive unit, and facilitate steering of the mission by all participants. The system can be asked to continue the course on automatic pilot, which is expected to be routine but may require minor adjustments. But when a situation requires careful analysis, all participants collaborate and share information while working towards the joint goal of reaching the destination safely and in compliance with established rules.

Workflows could enable "research cockpits" as a new interaction paradigm for scientists with underlying cyberinfrastructure. Scientific questions will set the overall goals and mission for the system. Along the way, any activities can be represented by workflows that will integrate any of the constraints (rules) to be respected. Workflow systems could automate routine tasks, while collaborating with scientists in novel analyses and to convey key information when outcomes are unusual or unexpected. Knowledge-rich representations of tasks, information, delegation, intention, and scientific goals are needed to support rich interactions for collaboration and automation.

8. Conclusions

Workflows should become first-class citizens in science and cyberinfrastructure. They provide explicit representations of computational analyses and provenance information for new data. Workflow systems today assist scientists by automating non-experiment critical tasks, systematically exploring the hypothesis space, managing parallelism and execution in distributed shared resources, and enabling low-cost reproducibility.

Using semantic representations of workflows will have an empowering effect leveling terms of the scientific processes supported. Today, semantic representations of scientific datasets are becoming more commonly used in cyberinfrastructure architectures to enable integration and reasoning over data. Similarly, knowledge-rich representations of workflows capture scientific principles and constraints that will enable a variety of artificial intelligence techniques to be brought to bear for validation, automation, hypothesis generation, and guarantees of data quality and pedigree. Knowledge-rich workflow systems open the doors to significant new capabilities for automated discovery, ever more integrative research that broadens

the scope of scientific endeavors, education in science at all levels, and novel paradigms for interaction of scientists with cyberinfrastructure to fully exploit its capabilities.

Workflows are a relatively new research area in computer science. More extensive investments in this area stand to greatly benefit scientific computing. Collaborative projects between computer scientists and domain scientists will focus their respective research agenda in relevant directions, clarify priorities, and provide data and experiences that will motivate further research questions. Scientists today speak of a data deluge. Workflows could provide a much needed layer of cyberinfrastructure to move science swiftly through the path from data to knowledge to discoveries.

Acknowledgments

This research was supported by the National Science Foundation under grant CCF-0725332. The author would like to thank Ewa Deelman and Carole Goble for very useful discussions on scientific workflows.

References

[1] D.E. Atkins, K.K. Droegemeier, S.I. Feldman, H. Garcia-Molina, M.L. Klein, D.G. Messerschmitt, P. Messina, J.P. Ostriker and M.H. Wright, Revolutionizing Science and Engineering Through Cyberinfrastructure, National Science Foundation Blue-Ribbon Advisory Panel on Cyberinfrastructure, January 2003; available at: http://www.nsf.gov/publications/pub_summ.jsp?ods_key=cise051203.

[2] G.B. Berriman, A.C. Laity, J.C. Good, J.C. Jacob, D.S. Katz, E. Deelman, G. Singh, M.-H. Su and T.A. Prince, Montage: The architecture and scientific applications of a National Virtual Observatory service for computing astronomical image mosaics, in: *Proceedings of Earth Sciences Technology Conference*, 2006.

[3] Y. Bhattacharjee, Ornithology: Citizen scientists supplement work of Cornell researchers, *Science* **308**(5727) (2005), 1402–1403.

[4] J.P. Cohn, Citizen science: Can volunteers do real research?, *BioScience* **58**(3) (2008), 192–197.

[5] B. Curtis, M.I. Kellner and J. Over, Process modeling, *Communications of the ACM* **35**(9) (1992).

[6] E. Deelman, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, K. Blackburn, A. Lazzarini, A. Arbree, R. Cavanaugh and S. Koranda, Mapping abstract workflows onto grid environments, *Journal of Grid Computing* **1**(1) (2003).

[7] E. Deelman, S. Callaghan, E. Field, H. Francoeur, R. Graves, N. Gupta, V. Gupta, T.H. Jordan, C. Kesselman, P. Maechling, J. Mehringer, G. Mehta, D. Okaya, K. Vahi and L. Zhao, Managing large-scale workflow execution from resource provisioning to provenance tracking: The CyberShake example, in: *IEEE International Conference on e-Science*, Amsterdam, December 4–6, 2006.

[8] E. Deelman and Y. Gil (eds), Final Report of the NSF Workshop on Challenges of Scientific Workflows, National Science Foundation, Arlington, VA, May 1–2, 2006; available at: http://www.isi.edu/nsf-workflows06.

[9] E. Deelman, G. Singh, M. Su, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, G.B. Berriman, J. Good, A. Laity, J.C. Jacob and D.S. Katz, Pegasus: a framework for mapping complex scientific workflows onto distributed systems, *Scientific Programming Journal* **13**(3) (2005).

[10] P. Fisher, C. Hedeler, K. Wolstencroft, H. Hulme, H. Noyes, S. Kemp, R. Stevens and A. Brass, A systematic strategy for large-scale analysis of genotype–phenotype correlations: Identification of candidate genes involved in African Trypanosomiasis, *Nucleic Acids Research* **35**(16) (2007), 5625–5633.

[11] Y. Gil, Workflow composition, in: *Workflows for e-Science*, D. Gannon, E. Deelman, M. Shields and I. Taylor, eds, Springer, 2007.

[12] Y. Gil, P.A. González-Calero, J. Kim, J. Moody and V. Ratnakar, Automatic generation of computational workflows from workflow templates using distributed data and component catalogs, forthcoming (submitted for publication).

[13] Y. Gil, V. Ratnakar, E. Deelman, G. Mehta and J. Kim, Wings for Pegasus: Creating large-scale scientific applications using semantic representations of computational workflows, in: *Proceedings of the 19th Annual Conference on Innovative Applications of Artificial Intelligence (IAAI)*, Vancouver, BC, Canada, July 22–26, 2007.

[14] Y. Gil, E. Deelman, M. Ellisman, T. Fahringer, G. Fox, D. Gannon, C. Goble, M. Livny, L. Moreau and J. Myers, Examining the challenges of scientific workflows, *IEEE Computer* **40**(12) (2007), 24–32.

[15] C. Goble and D. De Roure, myExperiment: Social networking for workflow-using e-scientists, in: *Proceedings of the 2nd Workshop on Workflows in Support of Large-Scale Science*, Monterey, CA, 2007.

[16] C.A. Goble, S. Pettifer, R. Stevens and C. Greenhalgh, *Knowledge Integration: In Silico Experiments in Bioinformatics in The Grid: Blueprint for a New Computing Infrastructure*, 2nd edn, I. Foster and C. Kesselman, eds, Morgan Kaufman, 2003.

[17] A. Goderis, P. Li and C.A. Goble, Workflow discovery: the problem, a case study from e-science and a graph-based solution, in: *ICWS*, IEEE Computer Society, 2006, pp. 312–319.

[18] A. Goderis, U. Sattler and C.A. Goble, Applying description logics for workflow reuse and repurposing, in: *Description Logics*, I. Horrocks, U. Sattler and F. Wolter, eds, CEUR Workshop Proceedings, Vol. 147, CEUR-WS.org, 2005.

[19] A. Goderis, U. Sattler, P. Lord and C. Goble, Seven bottlenecks to workflow reuse and repurposing, in: *Proc. of the 4th Int. Semantic Web Conference*, Galway, Ireland, 2005.

[20] T. Goodale, Expressing workflow in the Cactus framework, in: *Workflows for e-Science*, D. Gannon, E. Deelman, M. Shields and I. Taylor, eds, Springer, 2007.

[21] D. Hull, K. Wolstencroft, R. Stevens, C. Goble, M. Pocock, P. Li and T. Oinn, Taverna: A tool for building and running workflows of services, *Nucleic Acids Research* **34** (2006).

[22] E. Hutchins, *Cognition in the Wild*, MIT Press, 1995.

[23] E. Hutchins, How a cockpit remembers its speeds, *Cognitive Science* **19** (1995).

[24] E. Hutchins and T. Klausen, Distributed cognition in an airline cockpit, in: *Cognition and Communication at Work*, Y. Engeström and D. Middleton, eds, Cambridge University Press, 1996.

[25] D.S. Katz, G.B. Berriman, E. Deelman, J. Good, J.C. Jacob, C. Kesselman, A.C. Laity, T.A. Prince, G. Singh and M.-H. Su, A comparison of two methods for building astronomical image mosaics on a grid, in: *Proceedings of the 7th Workshop on High Performance Scientific and Engineering Computing (HPSEC-05)*, 2005.

[26] J. Kim, E. Deelman, Y. Gil, G. Mehta and V. Ratnakar, Provenance trails in Wings/Pegasus, *Computation and Concurrency: Practice and Experience*, Special issue on the First Provenance Challenge (2008).

[27] J. Kim, Y. Gil and V. Ratnakar, Semantic metadata generation for large scientific workflows, in: *Proceedings of the Fifth International Semantic Web Conference (ISWC-06)*, Athens, GA, November 5–9, 2006.

[28] J. Kim, M. Spraragen and Y. Gil, An intelligent assistant for interactive workflow composition, in: *Proceedings of the 2004 International Conference on Intelligent User Interfaces (IUI)*, Madeira Islands, Portugal, January 2004.

[29] D. Kulkarni and H.A. Simon, The processes of scientific discovery: The strategy of experimentation, *Cognitive Science* **12** (1988), 139–176.

[30] P. Langley, H.A. Simon, G.L. Bradshaw and J.M. Zytkow, *Scientific Discovery: Computational Explorations of the Creative Processes*, MIT Press, Cambridge, MA, 1987.

[31] B. Ludäscher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger-Frank, M. Jones, E. Lee et al., Scientific workflow management and the Kepler system, *Concurrency and Computation: Practice and Experience*, Special Issue on Workflow in Grid Systems **18**(10) (2006).

[32] T.W. Malone, K. Crowston and G.A. Herman, *Organizing Business Knowledge: The MIT Process Handbook*, MIT Press, 2003.

[33] R.E. McCaffrey, Using citizen science in urban bird studies, *Urban Habitats* **3**(1) (2005), 70–86.

[34] S.H. Muggleton, 2020 Computing: Exceeding human limits, *Nature*, Special Issue on 2020 Computing **440** (2006), 413–414.

[35] Nature Editorial, 2020 computing: Milestones in scientific computing, *Nature*, Special Issue on 2020 Computing **440**(7083) (2006).

[36] A. Newell, The knowledge level, *Artificial Intelligence* **18** (1982), 87–127.

[37] Office of Cyberinfrastructure (OCI), National Science Foundation's Cyberinfrastructure Council, Cyberinfrastructure Vision for 21st Century Discovery, March 2007; available at: http://www.nsf.gov/pubs/2007/nsf0728.

[38] T. Oinn, M. Greenwood, M. Addis, M. Nedim Alpdemir, J. Ferris, K. Glover, C. Goble, A. Goderis, D. Hull, D. Marvin, P. Li et al., Taverna: Lessons in creating a workflow environment for the life sciences, *Concurrency and Computation: Practice and Experience*, Special Issue on Workflow in Grid Systems **18**(10) (2006).

[39] T. Okada and H.A. Simon, Collaborative discovery in a scientific domain, in: *Proceedings of the 17th Annual Conference of the Cognitive Science Society*, J.D. Moore and J.F. Lehman, eds, 1995.

[40] J. Raloff, Universities seek armchair astronomers, *Science News* **172**(4) (2007).

[41] G.P. Robertson, Long-term ecological research: Re-inventing network science, *Frontiers in Ecology* **6**(5) (2008).

[42] A. Sameh et al., Report from ACM workshop on strategic directions in computing research, *Computational Science and Engineering* (1996).

[43] H.A. Simon, *The Sciences of the Artificial*, 1st edn, MIT Press, Cambridge, MA, 1969.

[44] H.A. Simon and K. Kotovsky, Human acquisition of concepts for sequential patterns, *Psychological Review* **70** (1963), 534–546.

[45] S. Sonnenburg, M.L. Braun, C.S. Ong, S. Bengio, L. Bottou, G. Holmes, Y. LeCun, K.-R. Müller, F. Pereira, C.E. Rasmussen, G. Rätsch, B. Schölkopf, A. Smola, P. Vincent, J. Weston and R. Williamson, The need for open source software in machine learning, *Journal of Machine Learning Research* **8**(October) (2007), 2443–2466.

[46] A. Szalay and J. Gray, 2020 Computing: Science in an exponential world, *Nature*, Special Issue on 2020 Computing **440** (2006), 413–414.

[47] I. Taylor, E. Deelman, D. Gannon and M. Shields (eds), *Workflows for e-Science*, Springer, 2007.

[48] I. Taylor, M. Shields, I. Wang and A. Harrison, The Triana workflow environment: Architecture and applications, in: *Workflows for e-Science*, I. Taylor, E. Deelman, D. Gannon and M. Shields, eds, Springer, New York, 2007, pp. 320–339.

[49] D. Thain, T. Tannenbaum and M. Livny, Distributed computing in practice: The Condor Experience, *Concurrency and Computation: Practice and Experience* **17**(2–4) (2005), 323–356.

[50] W.M. Washington et al., National Science Board 2020 Vision for the NSF, National Science Board Report, December 2005; available at: www.nsf.gov/publications/pub_summ.jsp?ods_key=nsb05142.

[51] M. Wieczorek, R. Prodan and T. Fahringer, Scheduling of scientific workflows in the ASKALON grid environment, *ACM SIGMOD Record* **34** (2005).

[52] C. Wroe, C. Goble, A. Goderis, P. Lord, S. Miles, J. Papay, P. Alper and L. Moreau, Recycling workflows and services through discovery and reuse, *Concurrency and Computation: Practice and Experience* **19**(2) 2007.

[53] J. Yu and R. Buyya, A taxonomy of scientific workflow systems for grid computing, *ACM SIGMOD Record*, Special Issue on Scientific Workflows **34**(3) (2005).