Kepler Project
Newsletter

News from Kepler/CORE September 2008

Preview of pPOD Kepler Extension Released

The pPOD team at UC Davis recently announced a preview release of a Kepler extension that provides new actors and tools to create and manage phylogenetic analyses. This extension was developed as part of a three-year National Science Foundation grant (http://www.phylodata.org) to address informatics challenges faced by researchers funded by the AToL (Assembling the Tree of Life) initiative. The pPOD extension includes actors that will enable AToL teams to automate phylogenetic analyses as well as reliably record and later reconstruct how results were obtained from primary observations.

The pPOD extension bundles the Kepler beta release with new pPOD actors, a number of phylogenetics applications automated by the actors, and sample workflows. The sample workflows highlight how Kepler can be used to align nucleotide and protein sequences, and to infer phylogenetic trees from these alignments using maximum likelihood, maximum parsimony, and Bayesian methods implemented in widely used tools. The workflows also demonstrate how easily native Java actors and ones that automate external applications or employ remote services can interoperate in a single Kepler workflow. A number of the actors in the package use the Gblocks applica-

(continued on page 2)

Kepler Stakeholder's Meeting

The thirty-four participants at the first Kepler Stakeholder's Meeting, held May 13-14 at UC Davis, discussed and helped set the future direction of Kepler development. Stakeholders presented projects applying Kepler to specific science domains, and spoke about their plans to extend Kepler as well as their needs for new core technologies. Together, the participants worked to identify ways to make it easier to develop the domain-specific system enhancements required by each project, as well as more general development priorities for Kepler in the future.

Over the course of the discussions, the Kepler/CORE team collected and recorded development suggestions, noting a total of one hundred and seventeen specific recommendations. At the end of the session, stakeholders rated each suggestion anonymously using wireless handsets and a five-point assessment scheme (critical, important, useful, useless, not sure). This audience response system helped guarantee that all participants had a voice in determining the importance of each recommendation.

The top development recommendations to emerge from the meeting, based on both the number of "critical" votes and a weighted score analysis, include separating the Kepler execution engine completely from the user interface; creating a cleaner split between workflow authoring and runtime environments; more fully supporting the import and export of actors using the Kepler Archive (KAR) format; providing a mechanism to precisely reference and/or embed component and data dependencies and their versions; and providing a stable, well-defined version of the Kepler kernel and standard extensions. Other top suggestions (as well as ones

(continued on page 4)

pPOD: Continued from Page 1

tion and programs in the Phylip application suite, all of which are bundled in the release. Other actors employ the REST services provided by the CIPRES Portal (http://www.phylo.org/sub_sections/portal/) hosted at the San Diego Supercomputer Center (SDSC), to transparently run the PAUP*, RAxML, MrBayes, and CLUSTAL applications on SDSC computer systems.

The pPOD actors and workflows are based on the Collection-Oriented Modeling and Design (COMAD) paradigm, a new computational model featuring built-in support for processing nested data collections in an assembly-line manner, and a fine-grained method for capturing and representing data provenance. Because of the often nested structure of biomolecular data sets, COMAD is well-suited for automating phylogenetics and other bioinformatics workflows. COMAD also facilitates the development of custom data types for particular domains. The pPOD package includes a custom data model for phylogenetics that allows actors that wrap applications and services employing different data formats to be strung together without intervening format-conversion actors or shims. Data is transparently provided to, and retrieved from, underlying software applications using this data model, allowing workflows to focus on the scientific rather than the data-manipulation aspects of computations. Consequently, the scientific intent of these analyses can be read directly from the Workflow canvas (see Figure 1a).

pPOD workflows also exploit the provenance capabilities provided by COMAD. Each run of the sample pPOD workflows produces an execution trace file. A trace is an XML representation of the data and collections input to and created by a workflow run, the parameter values for the actors, and the detailed provenance of all data created during the run. These traces are listed in a new panel within the Kepler GUI and can be viewed in an interactive provenance browser application bundled with the pPOD preview release (see Figure 1b). All source code developed by the pPOD project for Kepler, including the provenance browser application, are available via the Kepler source code repository. The Kepler/pPOD preview distribution for OS X, along with additional information about the project, can be found at http://daks.ucdavis.edu/kepler-ppod.

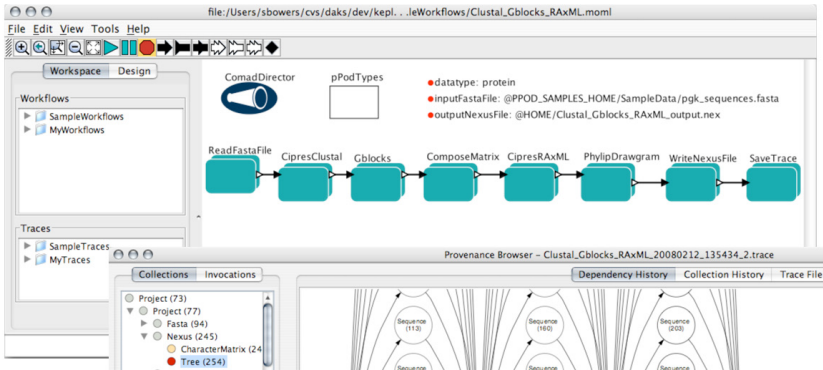


Figure 1a

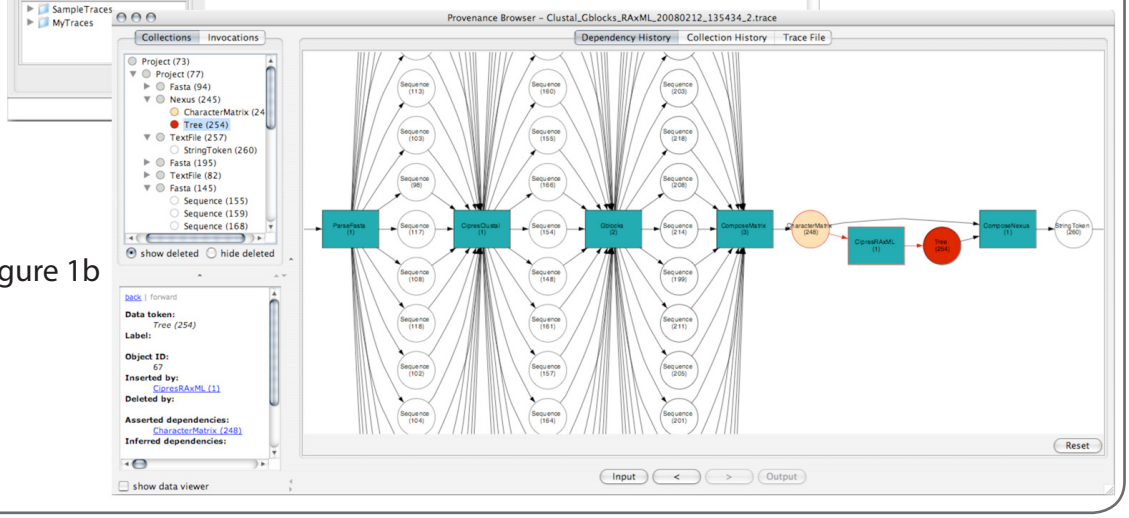
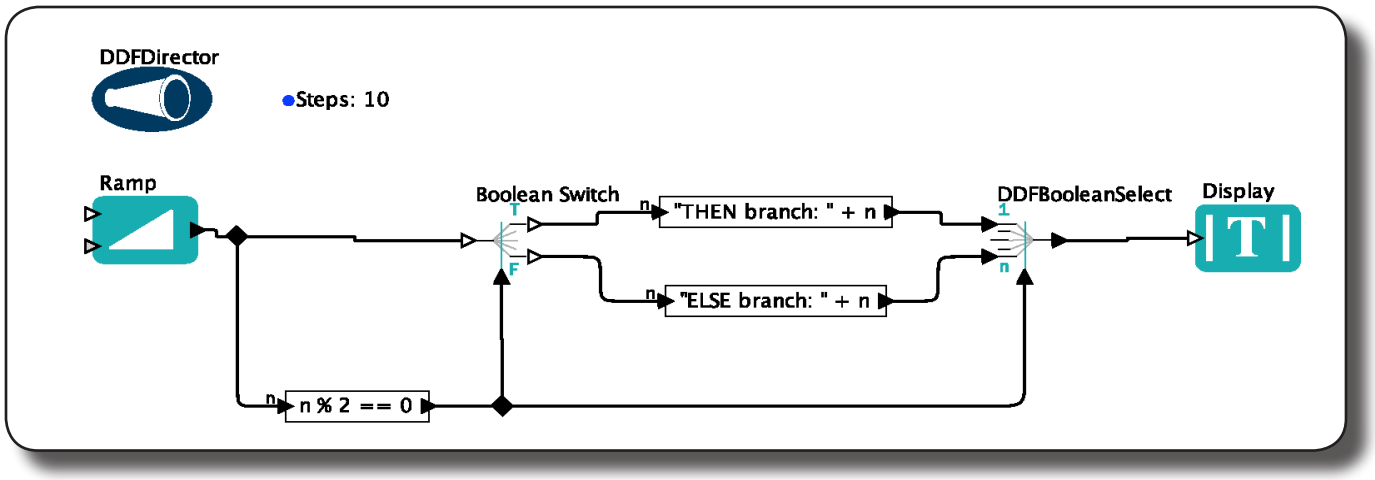


Figure 1b

Figure 1a: Example of a Kepler workflow that uses the COMAD computational model.

Figure 1b: Workflow trace as displayed by the new provenance browser bundled with the pPOD preview release.

Tech Tip: Conditional Routing and Execution in Kepler



A DDF workflow with conditional routing/execution. Note that you must connect the conditional expression to both the BooleanSwitch and the DDFBooleanSelect actor. Consider the Switch (a distributor) and Select (a merger) actors as "brackets" around the alternate execution branches. Usually, the number of tokens sent into the Switch equals the number of tokens emitted from the Select (mimicking a structured programming approach).

When executing parallel branches in a Kepler workflow, both branches typically receive the same (i.e. "cloned") tokens, so both execute in parallel. However, sometimes we would like to conditionally execute either branch, and the decision of which way a token should go may depend on the value of the token. In this case, the alternative branches (e.g., "then" and "else") merge downstream in the workflow (see DDF workflow, above). In Kepler, there are several different ways to implement such data-dependent routing and conditional execution.

1. To calculate a number or a Boolean value based on an incoming token, use <condition> ? <true-expr> : <false-expr> within an Expression actor. Note that the type of token created by the true and false expressions can be different, provided that the downstream actor can handle both types. In this construction we don't actually route the token into one of two distinct branches. Rather we have a single branch, where downstream actors will behave differently, depending on the token they receive.

2. To create an if-then-else branching structure, use the BooleanSwitch actor to pass an incoming token to either a T(rue) or a F(alse) output port, depending on the value of a Boolean token received on a control port. Note that the control input is not required; once set, the control will remain in effect until a new control token arrives. The BooleanSwitch actor cannot be used with an SDF director; use either DDF (for sequential workflows, using the DDFBooleanSelect actor to join the branches) or PN (for parallel processing; use BooleanSelect, DDFBooleanSelect, or NondeterministicMerge actors to join the branches). The two Select actors maintain the original order of the tokens entering the branch; the NondeterministicMerge actor outputs whichever arrives earlier.

Hints:
> The DDF Director's iteration parameter should usually be left at 0 (the default). In the example above, the Ramp actor sets the number of firings with its step parameter, not the DDF Director.
> To use this type of branching inside of an SDF workflow, nest the DDF sub-workflow (one token input, one token output)..

3. To create a switch structure with several branches, replace the BooleanSwitch actor with the Switch actor. The Switch actor expects an integer value on its control port and can output an arbitrary number of branches through its multi-port output. Branch numbering starts from 0. Use a Select or NondeterministicMerge actor to join the branches. Note that this technique works only under PN; there is no appropriate actor to join the branches under DDF.

(continued on page 4)

TECH TIP: Cont. from Page 3

Hints:
> The Switch actor does not provide a default branch. If the control value is out of the range of the existing channels, it will be discarded.
> The order in which you connect actors to the Switch and Select actors determines their order; you cannot define their order otherwise.

Stakeholder: Cont. from Page 1

deemed less useful) are included in the full report, which is available on the Kepler wiki.

Stakeholders also identified general areas they felt were most important for Kepler development by distributing up to three votes among fourteen development categories such as "monitoring and debugging runs" or "sharing actors and workflows." Using this technique, users identified that packaging and distributing the Kepler build, supporting distributed workflow execution, providing a web-based Kepler interface, and continuing to develop the graphical user interface were top priorities.

Meeting with stakeholders representing communities committed to applying Kepler to specific research domains, as well as with potential future users in additional scientific domains, is key to the Kepler mission. The twenty presented projects, ranging in focus from conservation biology to geosciences to phylogenetics and chemistry, demonstrated both the broad range of scientific fields in which Kepler is currently being used and the most pressing development needs of Kepler users.

Kepler/CORE is a collaborative effort lead by a team spanning several of the key institutions that originated the Kepler project: UC Davis, UC Santa Barbara, and UC San Diego. For a full analysis and summary of the stakeholder recommendations, as well as more information about participating projects, please see the Kepler wiki.

News and Events

September 10-11: "Accessing and Using Sensor Data within the Kepler Scientific Workflow System" and "An Integrated Framework for Hybrid and Adaptive Modeling of Sea Surface Temperature: A Workflow-Based Approach to Comparison". Environmental Information Management 2008 held at University of New Mexico. A presentation of recent work and Kepler use-cases for terrestrial ecology and oceanography. For more information, see https://conference.ecoinformatics.org/index.php/eim/eim2008

October 14-16: "Using Kepler to Visualize, Analyze, Communicate, and Document Conservation Science", Conservation Learning Exchange Conference in Vancouver, BC, Canada. Using case studies from the Kruger National Park in South Africa, this Kepler Tools Workshop introduces workflows as tools for conservation science. The demonstration covers how to obtain Kepler and access the demos that ship with it, as well as how to create and run a workflow that addresses common issues in conservation science. For more information, see http://conexmeeting.org/

November 16: Full-day Kepler tutorial, Supercomputing 2008 conference in Austin, TX. Intended for an audience with a computational science background, this tutorial provides an introduction to scientific workflow construction and management. The hands-on session covers principles and foundations of scientific workflows, Kepler environment installation, workflow construction and execution using existing Kepler components and facilities for process and data monitoring and provenance information, as well as high speed data-movement solutions. For more information, see http://sc08.supercomputing.org/

Kepler/CORE and pPOD are collaborative efforts of the University of California at Davis, Santa Barbara, and San Diego funded by the National Science Foundation under grant numbers 0722079 and NSF/IIS-0630033, respectively.