MATLAB And Java Integration For Distributed Signal Image Processing Applications

Elias S. Manolakos, Feras Al-Hawari, Demetris G. Galatopoullos

Communications and Digital Signal Processing Center for Research and Graduate Studies
Electrical and Computer Engineering Department, 442 Dana Research Building, Northeastern University, Boston, MA 02115
(eenas, fhawari, demetris)@ece.neu.edu, (617) 373-7522

The JavaPorts Project

- JavaPorts is a distributed processing framework that facilitates the configuration, development, deployment, and execution of a distributed application on a network of heterogeneous processing nodes.
- A major objective of the project is to enable the user to utilize the substantial computing power on a cluster without an in-depth parallel programming knowledge.
- Another objective for this design is to provide the means for performing what-if performance investigations of distributed configurations before application implementation and to facilitate runtime adaptation.

Significance and Contributions

- Introducing parallel scientific algorithms can take advantage of multiprocessor configurations.
- There is a need for software tools to facilitate the rapid prototyping of distributed applications on PC clusters.
- A JavaPorts parallel application can easily be reconfigured to take advantage of new computing structures, even on a different network, without requiring any code changes.
- The JavaPorts project contributes to component-based parallel computing by providing an intuitive means of describing a distributed application via a high-level Task Graph abstraction.

Novelty and Relevance

The JavaPorts capabilities that are not offered by the standard message passing libraries, such as MPI and PVMD, are:
- Reusable software components (versus SPMD style)
- Anonymous message passing (versus explicit message passing)
- Mixing components written in different languages
- Automatic distributed termination handling of user-defined tasks
- Visual application development capabilities (Java) on top of a common ancestor distributed execution framework.
- Automatic mapping of tasks onto machines (QoS Management System)

Java and Matlab Integration

JavaPorts compliant software components written in Java or Matlab can be integrated into the same application.

Matlab and Java Integration

- Matlab components can utilize Java to interact in a network.
- Matlab can utilize Java’s rich exception set, web-based programming capabilities, IDE and GUI tools.
- Java applications can take advantage of Matlab’s optimized and versatile function toolboxes.

The middleware style nature of both languages enables the transition of serial legacy codes to for better performing parallel implementations.

Related Work

- Matlab and Java based Computing Environments
  - Matlab ports to Distributed Computing
  - Matlab components in Java
  - Matlab and Java in the same application.

Implementing Java and Matlab Components

- JavaPorts provides the ability to integrate Java and Matlab components in the same distributed application.
- Matlab GUI for capturing the task-level configuration.
- GUI for capturing the task-level configuration.
- Matlab/Java components can be automatically added to distributed applications.
- Matlab components can be added to Java applications.
- Matlab components can be added to Java applications.

Future Directions

- Implementing multi-threaded, multi-tasking Matlab components.
- Developing a more user-friendly interface for Matlab components.
- Enhancing performance through parallelization.
- Integrating Matlab components into Java applications.

Current Status

- The latest JavaPorts release provides the developer the ability to:
  - Implement scalable software components using Java and Matlab
  - Rapidly prototype a distributed application by combining new and reusable software components in a visual environment
  - Describe communication between Components via Ports
  - Realize Components from one computing node to another in the network without making any code changes
  - Create hierarchical groupings of Components to increase organization and understanding of the Task Graph
  - Import existing components (functions) from another application or a library

- The new version provides:
  - Graphical interface for creating and executing distributed applications
  - Improved performance through parallelization
  - Enhanced user-friendliness through a more intuitive interface
  - Integration with existing Matlab components

- Future work includes:
  - Enhancing performance through parallelization
  - Creating a Java-based GUI to manage the resource monitoring modules and to run the QoS Management Sessions
  - Creating an intuitive means of describing a distributed application via a high-level Task Graph abstraction
  - Developing a more user-friendly interface for Matlab components

- Future work includes:
  - Enhancing performance through parallelization
  - Creating a Java-based GUI to manage the resource monitoring modules and to run the QoS Management Sessions
  - Creating an intuitive means of describing a distributed application via a high-level Task Graph abstraction
  - Developing a more user-friendly interface for Matlab components

- Future work includes:
  - Enhancing performance through parallelization
  - Creating a Java-based GUI to manage the resource monitoring modules and to run the QoS Management Sessions
  - Creating an intuitive means of describing a distributed application via a high-level Task Graph abstraction
  - Developing a more user-friendly interface for Matlab components

- Future work includes:
  - Enhancing performance through parallelization
  - Creating a Java-based GUI to manage the resource monitoring modules and to run the QoS Management Sessions
  - Creating an intuitive means of describing a distributed application via a high-level Task Graph abstraction
  - Developing a more user-friendly interface for Matlab components