ISR TECHNICAL REPORT 2010-6

Boom-i - A Toolkit for Networked Systems

Kiran Somasundaram, John S. Baras

Boom-i - A Toolkit for Networked Systems - Short Report

Kiran K. Somasundaram
ISR, ECE, UMD
kirans@umd.edu

Boom-i is a tool-kit that offers several graph data structures for networked systems. Unlike many other graph tools that are used for traditional algorithms, Boom-i has specialized data structures for distributed networked iterations. This includes message passing mechanisms, queues, dynamic graph data structures, etc.

All the world is a Graph

Many algorithms/ideas/concepts can be expressed using graphical structures. Boom-i has a framework that incorporates several methods from communication networks, graphical models for inference and distributed computation (synchronous and asynchronous). A special onus is placed on handling graphs with *dynamic relations.*

Boom-i is a completely objected oriented toolkit developed in Java. It is built on a software paradigm - "All the world is a graph and all the objects are players." In other words, it treats every object, irrespective of their type/class, as *players.* The different methods of these objects are *probes* and *responses* for the corresponding players.

Boom-i follows a strict hierarchical abstraction. Having different layers of abstraction helps a system designer to design networked systems with the aid of models and interfaces. All players in Boom-i exist in an abstraction called the *world.* Every player has a list of *characteristics*, which can be heterogeneous among the different players. These characteristics can be dynamic. Dynamic characteristics are handled by *dynamic databases* installed in every player. Examples of characteristics include trustworthiness, delay, credibility, computational capability. There can also exist several *relations* between different players. There are two forms of relations in Boom-i:

- *Arc Relations* - These are unidirectional relations, e.g., Player A is a parent (arc relation) to Player B.
- *Edge Relations* - These are bi-directional relations, e.g., Player A and B are friends (edge relation).

Every relation in Boom-i is handled by a *relation database* installed in the players. Each relation consists of a list of *relatives* (neighbors), which can, again, be characterized. Examples of relations include genealogical relations (parent-child), opinion credibility, communication bandwidth. In essence, the dynamic databases of characteristics and relations help maintain a dynamic weighted graph of players.

There are two other primary types in Boom-i: *managers* and *processors.* Managers are supporting players that help manage the characteristics and relations in a world. They usually have a global view of a particular characteristic or relation, and consequently, serve to *init*, *modify* and *delete* a characteristic or relation for the entire world. A manager is an abstraction - it could be implemented, again, as a distributed network of players. Another object type is *processor.* A processor gives a player a special capability to carry out task. Each player has some capabilities that are realized using different processors.

Semiring Players

A semiring player is player with a semiring processor installed in it. Any semiring processor must satisfy the interface constraints/methods given by the *SemiringProcessorInterface*:

- *doBigOPlus(s1,s2)* - Does the generalized plus operation of the semiring.
- *doBigOTimes(s1,s2)* - Does the generalized product operation of the semiring.
- *isAnnihilator(s)* - Checks if the element s is the generalized zero of the semiring.
- *isIdentity(s)* - Checks if the element s is the generalized one of the semiring.

As an example, consider the *MinPusSemiringProcessor* that satisfies the interface constraints of the *SemiringProcessorInterface.* It implements the *bigOplus* and the *bigOTimes* methods via the *doMin* and *doPlus* methods respectively.

If any of the semiring processors is installed in a set of semiring players, they can, together, exchange messages and solve the algebraic path problem for that particular semiring. Since several computations in networked systems are shown to be algebraic path problems over a semiring, the semiring player abstraction of Boom-i becomes a powerful tool to handle networked computations.

Package List

The class files of Boom-i are categorized into the following packages:

- *characteristics* - A collection of commonly used characteristics that can be used to mark any of the players, edge and arc relations.
- *graph_elements* - These are the primary graph elements that are instantiated in the world. It include nodes, arcs.
- *managers* - A collection of managers for different characteristics defined in the *characteristic*package.
- *my_exceptions* - A collection of exceptions that is thrown every time a characteristic is incorrectly used.
- *players* - A collection of players with different capabilities (*processors*).
- *processors* - A collection of different processors.

For a more detailed Java documentation of the Boom-i, see `http://www.ece.umd.edu/~kirans/boom_i/doc/`. The jar files available for download from `http://www.ece.umd.edu/~kirans/boom_i/jarfiles/boom_i.jar`.