The Trust Competition Testbed

Some Key Requirements

Munindar P. Singh

singh@ncsu.edu

Department of Computer Science
North Carolina State University

Introduction

- The field has matured enough that a testbed can help evaluate and compare approaches
- But every testbed makes its own presuppositions
- Proposal: proceed in phases
  - Study requirements abstractly
  - Evaluate proposed testbeds
  - Select one that we can live with for the next few years

Ontology

- Must enable task-sensitive trust
- Need an ontology to enable interoperation
  - Tasks performed and requested by agents
  - Qualities of service modeled, delivered, measured and, possibly, declared
  - Used by the testbed to specify what it offers
  - Used by agents to interact with each other
  - Select one that we can live with for the next few years

Single-Agent Perspective

- Set up multiple agents, but
- Measure the outcomes for a given agent
  - Quality of service obtained (best, worst, typical)
  - Likelihood of being selected
  - Profit or loss
  - Risk borne

Distributed Perspective

- Set up multiple agents, but
- Measure the outcomes for the agents in aggregation
  - Clustering metrics
  - Authoritativeness metrics (e.g., PageRank)
  - Subgraphs: communities in the distributed sense

Social Perspective

- Study relationships among agents
  - Reciprocity
  - Local aspects of clustering

Preferences and Policies

- Enable agents to declare their preferences
  - Policies for how trust is evaluated by an agent
  - Policies for how the agents interact
  - Evaluate the success or failure with respect to these policies

Threats

What kinds of threats would the testbed help us study?

- Despite our interest in high-level problems, underlying threats can act “unfair”
  - Death of an agent
  - Sudden sleep of an agent: loss of power or networking
  - Byzantine failures: agents become malicious
- Requirement: A suite of failures and an ability to tune them in experiments

Experimentation Modes

- *Independent:* Run multiple approaches separately for the same set up: simple and easy to use during development
- *Competitive:* Run multiple approaches at the same time but going against the “system” agents: a simple kind of bake off
- *Adversarial:* Run multiple approaches against each other: a more difficult kind of bake off to prepare for

Minimum Requirements

- Task-sensitivity
- Preservation of autonomy and heterogeneity
- Formal definitions for above metrics
- Extensibility with new metrics