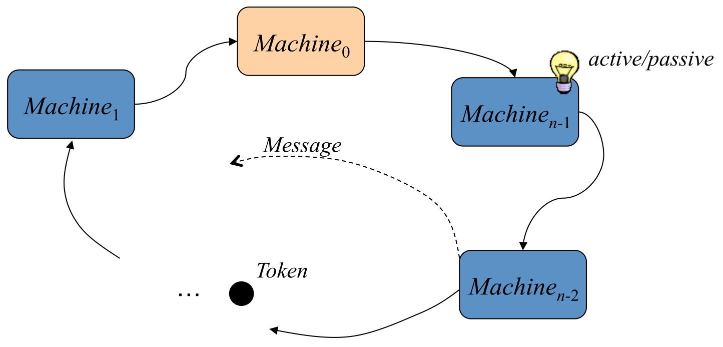
CMPT 475 - Software Engineering II

Due date: 11 April 2011

Assignment # 04

Topic: Distributed Termination Detection Protocol



In this assignment we consider a straightforward generalization of Dijkstra's termination detection protocol for distributed computations [1]. The original distributed algorithm as proposed by Dijkstra, and discussed in class, is meant to detect the stable state of global termination for a single distributed computation running on a network consisting of n machines. In contrast, the version we consider here will allow to detect the stable state of global termination for any number $k \geq 1$ of independent distributed computations running on the same network consisting of n machines. That is, for each of the k computations the termination of this computation will be detected independent of the state of any other computation running on the same network of machines. We may assume here that k is a fixed parameter of the protocol.

Problem Description

Abstractly model the functional requirements defining the extended version of the distributed termination detection protocol in terms of an Abstract State Machine model. In a second step, refine your ASM model of the resulting protocol into an executable CoreASM model. For the CoreASM model you should also provide sample inputs to test the model showing that it runs as expected with at least two concurrently running distributed computations on a network with a reasonable number $n \geq 8$ of machines.

Basic Idea

While there may be more than one way of how to solve the problem in general, the basic idea for extending the original protocol so as to meet the new requirements is relatively simple and straightforward. For the ASM model you may assume a level of abstraction directly comparable to the version of the protocol discussed in class. By doing so, the description of the resulting protocol should only be insignificantly more complex.

Solution

A good solution for this assignment will be a description that, besides correctness and completeness of the formal specification, also bears in mind the following aspects:

- An ASM model of the protocol that complies to the ASM syntax and semantics as presented in class;
- A vocabulary providing a list of declarations for the sets, functions and predicates used in the model, clearly indicating which functions/predicates you consider monitored functions/predicates;
- A proper definition of the initial state insofar as it matters for the operation of the protocol; for instance, for the executable version this also includes the definition of your network in addition to the initial configuration of the machines;
- A concise and clear description of the event mechanisms used in your protocol, specifically, what assumptions do you make how actions trigger events both locally on any of the machines and globally over the distributed network environment;
- Present a clear and conclusive argument why you think you protocol works correctly;
- A brief explanation for how to test the executable version of your protocol, including the test data and test environment (such as the sample network) used so that one can easily repeat your experiments.

Reference

[1] Edsger W. Dijkstra et al. Derivation of a Termination Detection Algorithm for Distributed Computations. *Information Processing Letters 16*: 217–219.