

Protocolos utilizando Memoria Compartida

Karina M. Cenci¹ Jorge R. Ardenghi²
Departamento de Ciencias de la Computación
Universidad Nacional del Sur – Bahía Blanca

Introducción

Un sistema de memoria compartida consiste de una colección de procesos en comunicación, como en un sistema de red. Pero en vez de enviar y recibir mensajes sobre un canal de comunicación, los procesos realizan operaciones instantáneas sobre datos compartidos.

Los modelos para un sistema de memoria compartida pueden ser sincrónicos y asincrónicos.

En el modelo sincrónico se asume que los componentes toman/ejecutan los pasos simultáneamente, esto es, la ejecución procede en completo sincronismo. Se utiliza para realizar simulaciones de sistemas distribuidos reales.

El modelo asincrónico asume que los componentes separados toman/ejecutan sus pasos en un orden, y a una velocidad relativamente arbitrarios. Es más difícil de programar que el modelo sincrónico a causa de la indeterminación en el orden de los eventos. El modelo asincrónico permite a los programadores ignorar las consideraciones temporales. Los algoritmos diseñados para estos modelos asincrónicos son generales y portables; garantizando la ejecución correcta sobre redes con garantía arbitraria de tiempo.

El modelo parcialmente sincrónico (basado en el tiempo): asume algunas restricciones en el tiempo relativo de los eventos, pero la ejecución no es completamente paso a paso (lock-step) como en el modelo sincrónico. Estos modelos son los más realistas, pero también los más difíciles de programar.

Un sistema de memoria compartida asincrónica consiste de una colección finita de procesos interactuantes por medio de una colección finita de variables compartidas, las variables son utilizadas solamente para la comunicación entre los procesos en el sistema. Cada proceso tiene un pórtico, con el cual puede conectarse con el mundo exterior utilizando acciones de entrada y de salida.

Para llevar a cabo este proyecto de estudio sobre un sistema de memoria compartida asincrónica se adopta el modelo de Máquinas de Estado Finito donde se supone que los procesos se ejecutan en un orden arbitrario y donde esa ejecución es formalizada como una secuencia alternativa de estados de autómatas alternados con acciones (cada acción corresponde a un proceso en particular), donde sucesivas ternas (estado, acción, estado) satisfacen la relación de transición.

Bibliografía

- [1] Jie Wu. *Distributed System Design*, 1999.
- [2] Nancy A Lynch. *Distributed Algorithms*, 1997.
- [3] Sape Mullender. *Distributed Systems*, 2da. Ed. 1993.
- [4] Abraham Silberschatz y Peter Galvin. *Operating System Concepts*, 5ta. Ed. 1998.
- [5] Nancy A. Lynch y Michael Fischer. On describing the behavior and implementation of distributed systems. *Theoretical Computer Science*, 1981.
- [6] Leslie Lamport. Proving the correctness of multiprocess programs. *IEEE Transactions on Software Engineering*, Marzo 1977.
- [7] Michael Raynal. *Algorithms for Mutual Exclusion*. MIT Press, Cambridge, 1986.
- [8] M. Ben Ari. *Principles of Concurrent Programming*. Prentice Hall, Englewood Cliffs, 1982.
- [9] Clyde P. Krustal, Larry Rudolph y Marc Snir. Efficient synchronization on multiprocessors with shared memory. In *Proceedings of the Fifth Annual Symposium on Principles of Distributed Computing*, Agosto 1986.
- [10] Bowen Alpern y Fred B. Schneider. Defining liveness. *Information Processing Letters*, Octubre 1985.

¹ e-mail: kmc@cs.uns.edu.ar

² e-mail: jra@cs.uns.edu.ar