Asynchronous ASMs

Ring Load Balance

Egon Börger

Dipartimento di Informatica, Universita di Pisa
http://www.di.unipi.it/~boerger

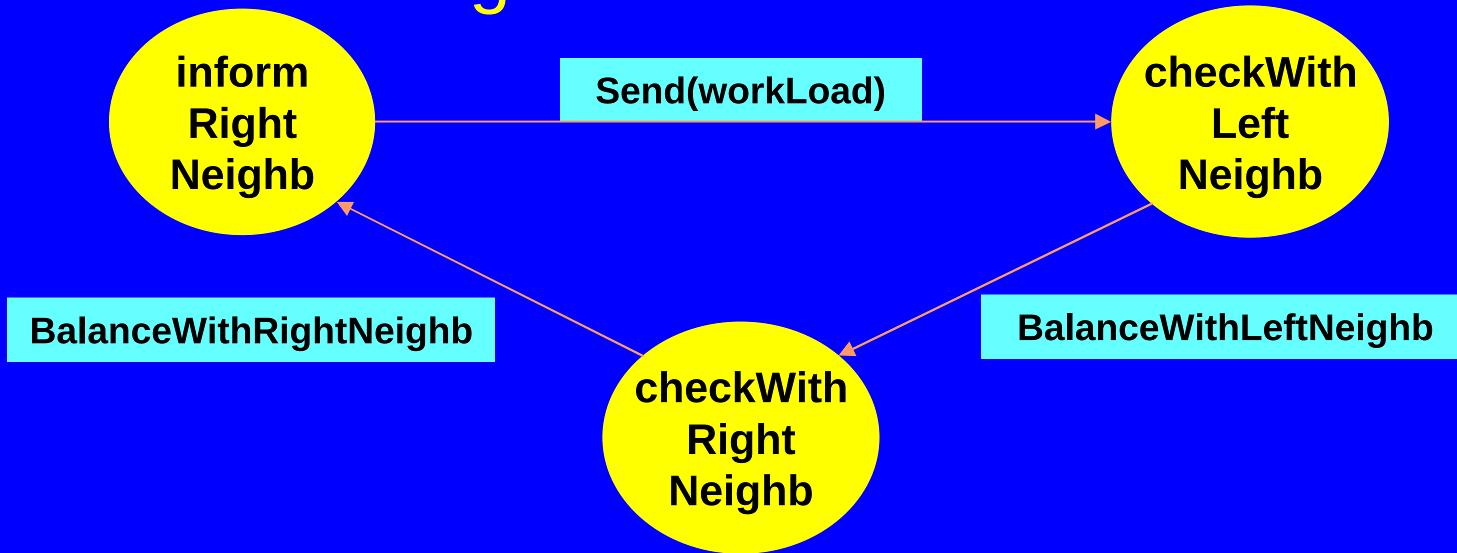
Ring Load Balance: problem statement

- Goal: Design a distributed algorithm for reaching workload balance among the agents of a ring (using only communication between right/left neighbors)
  - keeping message passing mechanism and effective task transfer abstract, assuming fixed number of agents and total workload
- Algorithmic Idea: every agent (ring node)
  - alternately sends
    - a workload info mssg to his right neighbor
    - a task transfer mssg to his left neighbor, transfering a task to balance the workload with the left neighbor
  - updates his workload to balance it with his right neighbor
- so that eventually the difference between the workload of two nodes becomes ≤ 1

Ring Load Balance ASM: Agent Signature

- Agent : a ring of agents equipped with:
  - leftNeighb, rightNeighb: Agent (external functions)
  - workLoad: the current workload
  - neighbLoad: current workload of leftNeighb
  - transferLoad: workload transfered by rightNeighb
  - ctl_state : {informRightNeighb, checkWithLeftNeighb , checkWithRightNeighb }
- Initially
  - ctl_state = informRightNeighb
  - neighbLoad = transferLoad = undef

Ring Load Balance ASM



BalanceWithRightNeighb o
If arrived(transferLoad) then
accept task from rightNeighb

BalanceWithLeftNeighb o
If arrived(neighbLoad) then
transfer task to leftNeighb

accept task from rightNeighb o
workLoad:= workLoad + transferLoad
transferLoad:= undef

arrived(l) o l ≠undef

transfer task to leftNeighb o
let transfer = workLoad > neighbLoad **in**
leftNeighb.transferLoad:= transfer
workLoad:= workLoad – transfer
neighbLoad:= undef

Send(workLoad) o
rightNeighb.neighbLoad
:= workLoad

Ring Load Balance ASM : Correctness property

- Proposition: In every distributed run of agents equipped with the ring load balance ASM, eventually the workload difference between two neighboring nodes becomes ≤ 1.
- Proof (assuming that every enabled agent will eventually make a move): induction on the wieght of run workload differences. Let w = total workLoad of all nodes, a= |Agent|.
  - Case 1: t|a. Then eventually workLoad(n)=w/a for every node n.
  - Case 2: otherwise. Then eventually the workLoad of some neighboring nodes will differ by 1.

Reference

- W.Reisig: Elements of Distributed Algorithms
    Springer-Verlag 1998
  - See Section 37 (in particular Fig. 37.1) and Section 82 for a correctness proof.
- E. Börger, R. Stärk: Abstract State Machines. A Method for High-Level System Design and Analysis Springer-Verlag 2003, see http://www.di.unipi.it/AsmBook
  - See Chapter 6.1