Sami Evangelista
sami.evangelista@lipn.univ-paris13.fr

Introduction

This Model form is a short description of the Simple load balancing system model that comes, for the Model Checking Contest @ Petri Nets, with: a set of PNML files, a set of properties to be checked (possibly one file per model instance) and an optional set of properties concerning the model (invariants, etc. – possibly one file per model instance). For Coloured Nets, equivalent PNML P/T net files are proposed too.

Simple load balancing system

Presentation

Description: This net models a simple load balancing system composed of a set of clients, two servers, and between these, a load balancer process (called lb process thereafter).

The clients Their role is to send requests to servers (transition client_send), wait for an answer and get it (transition client_receive). Note that requests are not directly sent to servers but to the lb process so that this one routes it to the appropriate server (i.e., tokens modeling requests are put in place client_request before being put in server_request).

The servers Each of the two servers waits for requests (i.e., tokens in place server_request). When such a request arrives it processes it and send a reply to the client (transition server_process). The server then notifies the lb process (transition server_notify) so that this one possibly rebalances requests between servers. Once the lb process has acknowledged this notification, the server can go back to the idle state (transition server_endloop).

The lb process The lb process has the most complex task.

First, in the idle state, it can receive a request from a client (transition lb_receive_client). It then routes this request either to server 1 (transition lb_route_to_1) either to server 2 (transition lb_route_to_2) depending on the loads of these two servers. Place lb_load is a key element of the net as it used by the lb process to guide its actions. This place always has two tokens $(1, l1)$ and $(2, l2)$ where $l1$ (resp. $l2$) is the number of requests assigned to server 1 (resp. 2).

Second, when a server notifies the lb process that it has completed a request, the lb process records this information by modifying the content of place lb_load and then goes to the "balancing" state (transition lb_idle_receive_notification) in order to possibly reassign requests between servers. In this state (place lb_balancing), the lb process can perform four actions. The first one is to go back to the idle state if loads are already balanced (transition lb_no_balance). Transition lb_balance_to_1 (resp. lb_balance_to_2) models the situation where server 2 (resp. server 1) has more requests to handle than server 1 (resp. server 2). The lb process then reroutes one request from one server to the other. In these first three situations (transitions lb_no_balance, lb_balance_to_1 and lb_balance_to_2), the lb process goes back to the idle state. At last, in the "balancing" state, the server must treat server notifications (transition lb_balancing_receive_notification) in order to keep an up-to-date information on the loads of servers and correctly rebalances.

Origin: Model from the helena tool distribution modified (slightly modified).

Scaling parameter

Name	Description	Values
N	Number of clients.	2,3,4,5,6,7,8,9,10,15,20

Information about the Model

Data on the Model

Number of places	Number of transitions	Number of arcs	Scaling parameter value
14	13	57	2, 5, 10, 15, 20

Stated Properties

Sami Evangelista
sami.evangelista@lipn.univ-paris13.fr

safe	✔	**free choice**	✗	**event graph**	✗
deadlock	✗	**state machine**	✗	**reversible**	✗