APPLYING BCMP MULTI-CLASS QUEUEING NETWORKS FOR THE PERFORMANCE EVALUATION OF HIERARCHICAL AND MODULAR SOFTWARE SYSTEMS

Simonetta Balsamo  Gian-Luca Dei Rossi  Andrea Marin

Dipartimento di Informatica
Università Ca’ Foscari, Venezia

ESM’2010 Conference, October 25-27 2010
Outline

- Software Modularity and Performance Evaluation
- BCMP Queueing Networks
- An approach to modular and hierarchical software design
- The unfolding algorithm
- Applications and examples
- Conclusion
• Performance analysis of modular and hierarchical systems always an important topic in research (see Smith (1990)).

• Software system: interaction between black-box components, made of other black-box components . . .

An example of a black-box component

Main problem

Defining efficient algorithms to derive performance indexes.
BCMP Networks (Baskett et al. (1975)) are one of the most useful models for performance evaluation.

- A set of queueing centers and a (possibly infinite) set of customers.
- Classes, that determine:
  - routing probabilities;
  - service time distributions.
- Each class belongs to a chain
  - open (external arrivals) or closed.
- Class switching only within the same chain.
- Queueing station of one of the following types:
  1. FCFS Queueing discipline. Exponential and class-independent service time distribution,
  2. PS Queueing discipline,
  3. The station has infinite servers (Delay Station),
  4. LCFSPR service discipline.

With coxian service time distribution
Let us consider a multiple-class and multiple-chain QN, open, closed or mixed, whose queueing stations are of type 1, 2, 3 or 4. If the underlying stochastic process is ergodic, then:

\[ \pi(n) = \frac{1}{G} \prod_{i=1}^{M} g_i(n_i), \]

where

- \( n = (n_1, \ldots, n_M) \) is the state of the network, \( n_i \) is the state of station \( S_i \);
- \( \pi \) is the steady-state distribution of the QN;
- \( g_i(n_i) \) is the steady-state distribution of station \( S_i \) considered in isolation;
- \( G \) is a normalising constant.
• BCMP Networks are widely used. Various solution algorithms.
  • for *open* networks, $G = 1$.
  • for *closed* or *mixed* network, algorithms to compute $G$ or directly derive the average performance indices.

• BCMP Networks are inherently *flat*.
  • no modular and/or hierarchical design.

We propose an algorithm to compute a BCMP from an modular and hierarchical high-level model.
The design framework

A framework for the specification of hardware and software architectures.

- A set of interacting components \( d_1, d_2, \ldots, d_{\ell_1} \). Each component can be
  - A BCMP queueing station
  - A sub-model consisting of components \( d_{(i)1}, \ldots, d_{(i)\ell_2} \).
- Components interact as stations of an open multiple-class and multiple-chain QN.
- Sub-models as black boxes
  - access points with some labels, i.e., input and output classes.

We require that

- the set of input classes and the set of output classes must be equal,
- the model must be well formed.
More on the design framework

- A higher level class could be connected to any lower level class.
- Multiple classes of the higher level submodel could be connected to the same lower level class.
- The same component could be reused in different submodels.

An example of design of a CMS module

Class connections: $A_{d_i,d_j} : \mathcal{R}_i \rightarrow \mathcal{R}_j$

How to keep routing information in case of component or class reuse?
Algorithm UnfoldComponents

Input: routing matrix $P_d$ of component $d$, component counter array $Dc$, functions $A_{i,j}$
Output: unfolded routing matrix $P'_d$ of component $d$

if $d$ is a station then
    foreach class $r$ of $d$ do
        insert in $P'$ rows and a columns for $El_d, r$ and $EO_d, r$ of $P$
    end
else
    foreach $d_i | d \succ d_i$ do
        $Dc_i \leftarrow Dc_i + 1$
        Let $Rc_i$ be a class counter array
        foreach class $r_k$ of $d_i$ as named in $d$ do
            $r_i, j \leftarrow A_{d_i, d_i}(r_k)$
            $Rc_i, j \leftarrow Rc_i, j + 1$
            if $Rc_i, j > \max Rc_i$ then
                $U = \text{UnfoldComponents}(P_{d_i})$
                rename each class $r_{i, j}$ of $d_i$ in $U$ as $r_{i, j}, Dc_i, Rc_i, j$
                insert in $P'$ rows and columns of $U$
            end
        end
        replace column $El_{d_i}, r_{i, j}, Dc_i, Rc_i, j$ in $P'$ with column $d_i, r_k$ of $P$
        replace row $EO_{d_i}, r_{i, j}, Dc_i, Rc_i, j$ in $P'$ with row $d_i, r_k$ of $P$
    end
end
return $P'$
Some notes on the algorithm

• Recursive and top-down.
• It adds classes whenever it is needed in order to not lose customer routing information.
• Base cases: BCMP queueing stations.
• Output: multiple-class, multiple-chain BCMP network.
• Computational complexity $O(rd^n)$ if, on average:
  • every component has the same number of sub-components $d$,
  • every component has the same number of classes $r$,
  • the depth of the model is $n$. 
A small example

Database-indexed file archive for a CMS.

- A class of customer for read operations.
- A class of customers for write operations.

The CMS file archive module and the DB Module

How many classes should station $d_4$ have at the end of the algorithm run?
$d_4$ should have at least 4 classes.

- The algorithm doubles the number of classes for $d_2$.
- The classes of $d_2$ translates directly in classes of $d_4$.
- We cannot know if $d_4$ or $d_2$ are used by other components.
- More complex model topology may lead to a rapid increase of the number of classes.
Conclusion and future works

- A modeling technique for hierarchical systems.
- An algorithm that transforms such models in a multi-class and multi-chain BCMP.
- Main advantages:
  - a modular and hierarchical modelling technique,
  - an efficient and exact analysis method.
- More expressive formalisms, like LQNs in Woodside et al. (1995) may require approximate algorithms.
- Future works:
  - extension of the tractable class of models,
  - integration of the framework with web mining and log analysis.
Any question?

