A Novel Congestion Control Scheme for Elastic Flows in Network-on-Chip Based on Sum-Rate Optimization

Mohammad S. Talebi[1], Fahimeh Jafari[1,3], Ahmad Khonsari[2,1], and Mohammad H. Yaghmae[3]

[1] IPM, School of Computer, Tehran, Iran.,
[2] ECE Department, University of Tehran
[3] Ferdowsi University of Mashhad,
mstalebi@ipm.ir, jafari@ipm.ir, ak@ipm.ir, hyaghmae@ferdowsi.um.ac.ir

Abstract. Network-on-Chip (NoC) has been proposed as an attractive alternative to traditional dedicated busses in order to achieve modularity and high performance in the future System-on-Chip (SoC) designs. Recently, end-to-end congestion control has gained popularity in the design process of network-on-chip based SoCs. This paper addresses a congestion control scenario under traffic mixture which is comprised of Best Effort (BE) traffic or elastic flow and Guaranteed Service (GS) traffic or inelastic flow. We model the desired BE source rates as the solution to a rate-sum maximization problem which is constrained with link capacities while preserving GS traffic services requirements at the desired level. We proposed an iterative algorithm as the solution to the maximization problem which has the advantage of low complexity and fast convergence. The proposed algorithm may be implemented by a centralized controller with low computation and communication overhead.

1 Introduction

The *Systems-on-Chip* (SoC) was first designed as a tightly interconnected set of cores, where all components share the same system clock, and the communication between components is via shared-medium busses. With the advance of the semiconductor technology, the enormous number of transistors available on a single chip allows designers to integrate dozens of IP blocks together with large amounts of embedded memory. Such IPs consist of CPU or DSP cores, video stream processors, high-bandwidth I/O, routers, *etc*. As more and more cores are integrated into a single chip, it is becoming increasingly difficult to meet the design constraints while still using the old design methodologies for SoC designs. Shared-medium busses do not scale well, and do not fully utilize potentially available bandwidth. As the features sizes shrink, and the overall chip size relatively increases, interconnects start behaving as lossy transmission lines. Crosstalk, electromagnetic interference, and switching noise cause higher incidence of data errors. Line delays have become very long as compared to gate delays causing synchronization problems between cores. A significant amount of power is dissipated on long interconnects and in clocking network. This trend only

O. Gervasi and M. Gavrilova (Eds.): ICCSA 2007, LNCS 4707, pp. 398 – 409, 2007.

worsens as the clock frequencies increase and the features sizes decrease. Lowering the power supplies and designing smaller logic swing circuits decreases the overall power consumption at the cost of higher data errors.

One solution to these problems is to treat SoCs implemented using *micro-networks*, or *Networks on Chips* (NoCs). Networks have a much higher bandwidth due to multiple concurrent connections. They have regular structure, so the design of global wires can be fully optimized and as a result, their properties are more predictable. Regularity enables design modularity, which in turn provides a standard interface for easier component reuse and better interoperability. Overall performance and scalability increase since the networking resources are shared.

Networking model decouples the communication layers so that design and synthesis of each layer is simpler and can be done separately. In addition, decoupling enables easier management of power consumption and performance at the level of communicating cores. Generally, the concept of NoC which was introduced in [1][2], suggests that different modules would be connected by a simple network of shared links and routers. Examples of NoCs are Æthereal [3], Mango [4] and Xpipes [5]. NoCs provide communication services to IPs. Communication services with guarantees on throughput and latency enable predictable system design. Guarantees are given by reserving communication resources in the NoC (e.g. wires and buffers). Although necessary for hard real-time applications, this results in poor resource utilization for applications that require Variable-Bit Rate (VBR) communication. According to the nature of such kind of traffic, this is also called *Inelastic Flow*. Best Effort service (BE) is another kind of communication services which doesn't need any guarantees on latency and bandwidth. According to the nature of this kind of traffic, this is also called *Elastic Flow*. Elastic Flow or BE service can give high resource utilization by using unreserved or unused resources. However, BE traffic is prone to network congestion. Æthereal [3] and Mango [4] are examples of NoCs that provide both GS and BE services.

Networks with BE services should have a strategy to avoid congestion. The congestion control in NoCs is a novel problem for the resource constrained on-chip designs. During the past two decades, many strategies for congestion control have been proposed for off-chip networks [6, 7, 8]. Congestion control for on-chip networks is still a novel issue, however this problem has been investigated by several researchers [9]-[11]. In [9], a prediction-based flow control strategy for on-chip networks has been proposed where each router predicts future buffer fillings to detect future congestion problems. The buffer filling predictions are based on a router model. Dyad [10] solves congestion problem by switching from deterministic to adaptive routing when the NoC gets congested. In [11] the link utilization has been used as congestion measure and the controller determines the appropriate loads for the BE sources. All of the aforementioned work has dealt with this issue using the predictive control approach to overcome the congestion in the network. As the NoC architecture is similar to a regular data network, in this paper we have used an optimization approach over Best Effort source rates to control the flow.

The main purpose of this paper is to present a congestion control as the solution to a sum-rate maximization problem for choosing the rate of BE sources. We present an algorithm as the solution to the optimization problem and prove its convergence. To evaluate the performance of the proposed approach, we simulate the congestion

control algorithm under a NoC-based scenario. Similar to [10], we have used a controller to implement the proposed algorithm; however our approach is completely different from [10].

This paper is organized as follows. In section 2 we present the system model and formulate the underlying optimization problem for BE flow control. In section 3 we solve the optimization problem using an iterative algorithm and propose the solution as a centralized congestion control algorithm to be implemented as a controller. In section 4 we analyze the convergence behavior of the proposed algorithm and prove the underlying theorem of its convergence. In section 5 we present the simulation results. Finally, the section 6 concludes the paper and states some future work directions.

2 System Model

We consider a NoC with two dimensional mesh topology and wormhole routing. In wormhole networks, each packet is divided into a sequence of *flit*s which are transmitted over physical links one by one in a pipeline fashion. The NoC architecture is assumed to be lossless, and packets traverse the network on a shortest path using a deadlock free XY routing.

We model the congestion control problem in NoC as the solution to an optimization problem. For more convenience, we turn the aforementioned NoC architecture into a mathematical model as in [8]. In this respect, we consider NoC as a network with a set of bidirectional links L and a set of sources S. A source consists of Processing Elements (PEs), routers and Input/Output ports. Each link $l \in L$ is a set of wires, busses and channels that are responsible for connecting different parts of the NoC and has a fixed capacity of c_l packets/sec. We denote the set of sources that share link l by $S(l)$. Similarly, the set of links that source s passes through, is denoted by $L(s)$. By definition, $l \in S(l)$ if and only if $s \in L(s)$ [8].

As previously stated, there are two types of traffic in a NoC: Guaranteed Service traffic (GS) or inelastic flow and Best Effort (BE) traffic or elastic flow. For notational convenience, we divide S into two parts, each one representing sources with the same traffic. In this respect, we denote the set of sources with BE and GS traffic by S_{BE} and S_{GS}, respectively. Each link l is shared between the two aforementioned traffics. GS sources will obtain the required amount of the capacity of links and the remainder should be allocated to BE sources.

Our objective is to choose source rates (PE loads) of BE traffics so that to maximize the sum of rates of all BE traffics. Hence the maximization problem can be formulated as:

$$\max_{x_s} \sum_{s \in S_{BE}} x_s \tag{1}$$

subject to:

$$\sum_{s \in S_{BE}(l)} x_s + \sum_{s \in S_{GS}(l)} x_s \le c_l \qquad \forall l \in L \tag{2}$$

$$x_s > 0 \qquad \forall s \in S_{BE} \tag{3}$$

where source rates, i.e. x_s, $s \in S$, are optimization variables.

The constraint Eq. (2) says the aggregate BE source rates passing thorough link l cannot exceed its free capacity, i.e. the portion of the link capacity which has not been allocated to GS sources. The abovementioned problem is in fact *constrained sum-rate maximization.* Such a problem, in general belongs to the class of *Utility-Maximization Problems* for which the utility function of all sources is considered to be *Identity Function*, i.e. $U_s(x_s) = x_s$. Although the general form of Eq. (1) with a general utility function has been investigated by the authors in another work [12], the approach of this paper to solve problem Eq. (1) is completely different from the previous work. In this paper we focus on the primal problem while in [12] the problem is solved via its dual function. For notational convenience, we define:

$$\hat{c}_l = c_l - \sum_{s \in S_{GS}(l)} x_s \tag{4}$$

Hence, Eq. (2) can be rewritten as:

$$\sum_{s \in S_{BE}(l)} x_s \leq \hat{c}_l \qquad \forall l \in L \tag{5}$$

Although problem Eq. (1) is separable across sources, its constraints will remain coupled across the network. Due to coupled nature of such constrained problems, they have to be solved using centralized methods like interior point methods [13]-[15]. Such computations may pose a great overhead on the system. Instead of such methods, we seek to obtain the solution with simpler operations. One way is to use the subgradient method for constrained optimization problems [16] which will be briefly reviewed in the next section.

For notational convenience in solving the problem, we use matrix notation. In this respect, we define Routing matrix, i.e. $R = [R_{ls}]_{L \times S}$, as following:

$$R_{ls} = \begin{cases} 1 & \text{if } s \in S_{BE}(l) \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

We also define the source rate vector (for BE traffic) and link capacity vector as $x = (x_s,\ s \in S_{BE})$ and $\hat{c} = (\hat{c}_l,\ l \in L)$, respectively. Therefore problem Eq. (1) can be rewritten in the matrix form as follows:

$$\max_x \ \mathbf{1}^T x \tag{7}$$

subject to:

$$Rx \leq \hat{c} \tag{8}$$

$$x_s > 0 \qquad \forall s \in S_{BE} \tag{9}$$

where $\mathbf{1}$ is a vector with all one.

3 Congestion Control Algorithm

In this section, we will solve the sum-rate optimization problem Eq. (7) using subgradient method for constrained optimization problems [13][16] and present a flow control scheme for BE traffic – or elastic flows - in NoC systems to overcome the congestion. Convergence analysis of the algorithm is to be discussed in the next section.

The subgradient method for constrained optimization problems is very similar to *Poljak's Method* [17]. In this method, for a maximization problem like Eq. (1), the optimization variable vector will be adjusted in the direction to the gradient of the objective function. We briefly review this method in lemma 1 as follows.

Lemma 1. *Consider the constrained maximization problem,*

$$\max_{x} f(x) \tag{10}$$

subject to:

$$f_i(x) \geq 0, \quad i = 1..M \tag{11}$$

with the maximal x^* *and the sequence* $x(t)$ *as*

$$x(t+1) = x(t) + \gamma(t)u(x(t)) \tag{12}$$

where

$$u(x(t)) = \begin{cases} \nabla f(x(t)) & \text{if } x(t) \text{ satisfies (11)} \\ \nabla f_j(x(t)) & \exists j \text{ s.t } f_j(x(t)) < 0 \end{cases} \tag{13}$$

where $\gamma(t)$ *is a diminishing step-size rule [13]-[15]. If the* l_2 *-norm of the* $u(x(t))$ *is bounded (Lipschitz Continuity), i.e. there exist* G *such that*

$$\|u\|_2 \leq G \tag{14}$$

and the Euclidian distance of the initial point to the optimal point is bounded, i.e.

$$\left\|x(1) - x^*\right\| \leq D \tag{15}$$

then the sequence $\{x(t)\}_{t=1}^{\infty}$, *as* $t \to \infty$ *will converge to* x^*.

Proof: See [16].

In the sequel, we will solve the optimization problem Eq. (7) using subgradient method for constrained optimization problems as stated in Lemma 1. Regarding Eq. (12), we should calculate $u(x(t))$. According to Eq. (13), if $x(t)$ is feasible, i.e. $Rx \leq \hat{c}$, we have:

$$u = \nabla \mathbf{1}^T x = \nabla x^T \mathbf{1} = \mathbf{1} \tag{16}$$

otherwise at least one of the constraints should be violated. Assuming the corresponding constraint for link $l^{'}$ is violated, i.e. $\sum_{s\in S_{BE}(l^{'})} x_s > \hat{c}_{l^{'}}$. We can represent this constraint in matrix form as:

$$f_{l'}(x) = \mathbf{e}_{l^{'}}^{T}(\hat{c} - Rx) < \mathbf{0} \tag{17}$$

where $\mathbf{e}_{l^{'}}$ is the $l^{'}$th unit vector of $\mathbb{R}^{L}$ space which is zero in all entries except the $l^{'}$th at which it is 1. Therefore, u is given by:

$$u = -\nabla \mathbf{e}_{l^{'}}^{T}(Rx - \hat{c}) = -R^{T}\mathbf{e}_{l^{'}} \tag{18}$$

Using Eq. (16) and Eq. (18), the update equation to solve problem Eq. (7) is given by:

$$x(t+1) = x(t) + \gamma(t)u(x(t)) \tag{19}$$

where $u(x(t))$ is given by:

$$u(t) = \begin{cases} \mathbf{1} & \sum_{s\in S_{BE}(l)} x_s(t) \le \hat{c}_l, \forall l \\ -R^{T}\mathbf{e}_{l^{'}} & \sum_{s\in S_{BE}(l^{'})} x_s(t) > \hat{c}_{l^{'}}, \exists l^{'} \end{cases} \tag{20}$$

Stepsize has an important role on the convergence behavior of the update equation. There are several choices for stepsize, each one belonging to a predefined category and having certain advantages and drawbacks (see [14] and references herein).

In the family of gradient algorithms, for distributed scenarios stepsize is usually chosen to be a small enough constant so that to guarantee the convergence of the algorithm. Constant stepsize is robust in the sense of convergence in time-varying conditions and asynchronous schemes[1]. However, it mainly suffers from slow convergence rate. On the contrary, time-varying stepsizes are defined in such a way to adapt to the error with the desired point, i.e. optimal point of the optimization problem, and hence benefit from much more faster convergence. However, they should be constrained to guarantee that the iterative algorithm will converge.

In our scheme, the algorithm is to be centralized in implementation, and thus we use a time-varying stepsize to take advantage of fast convergence. To this end, we choose $\gamma(t)$ as a time-varying stepsize, to be *square-summable but not summable* [14][16]. In this respect, $\gamma(t)$ satisfies

$$\gamma(t) \ge 0 \qquad \forall t \tag{21}$$

$$\sum_{k=1}^{\infty} \gamma^{2}(t) < \infty \tag{22}$$

[1] Note that Eq. (19) proposes a synchronous scheme, and may diverge in asynchronous ones, e.g. real world conditions with large delays, etc.

$$\sum_{k=1}^{\infty} \gamma(t) = \infty \tag{23}$$

One typical example that satisfies Eq. (21)-(23), is $\gamma(t) = a/(b+t)$, where $a > 0$ and $b \geq 0$, which we have used in this paper.

Eq. (19) and Eq. (20) together propose an iterative algorithm as the solution to problem Eq. (7). In this respect, optimal source rates for BE sources can be found while satisfying capacity constraints and preserving GS traffic requirements. Thus, the aforementioned algorithm can be employed to control the congestion of the BE traffic in the NoC. The above iterative algorithm is decentralized in the nature and can be addressed in distributed scenarios. However, due to well-formed structure of the NoC, we focus on a centralized scheme; a simple controller can be mounted in the NoC to implement this algorithm. The necessary requirement of such a controller is the ability to accommodate simple mathematical operations as in Eq. (19) and Eq. (20) and the allocation of few wires to communicate congestion control information to nodes with a light GS load. Algorithmic realization of the proposed Congestion-Controller for BE traffic is listed as Algorithm 1.

Algorithm 1. Congestion Control for BE Traffics in NoC

Initialization:

1. **Initialize $\hat{c}_l$ of all links.**
2. **Set source rate vector to zero.**

Loop:

Do until $(\max |x_s(t+1) - x_s(t)| < Error)$

1. $\forall s \in S$ **: Compute new source rate:**

$$x(t+1) = x(t) + \gamma(t) u(x(t))$$

where $\gamma(t)$ **can be selected as** $\gamma(t) = a/(b+t)$ and

$$u(t) = \begin{cases} \mathbf{1} & \sum_{s \in S_{BE}(l)} x_s(t) \leq \hat{c}_l, \ \forall l \\ -R^T \mathbf{e}_{l'} & \sum_{s \in S_{BE}(l')} x_s(t) > \hat{c}_{l'} \end{cases}$$

Output:

Communicate BE source rates to the corresponding nodes.

4 Convergence Analysis

In this section, we investigate the convergence analysis of the proposed algorithm using a time-varying stepsize in Eq. (19). As stated in the previous section, in this paper the stepsize is selected to be *square-summable but not summable* [16].

Theorem 1. *The iterative congestion control scheme proposed by Eq. (19) and Eq. (20) with a time-varying stepsize which satisfies Eq. (21)-(23), will converge to the optimal point of problem Eq. (1).*

***Proof*:** By lemma 1, it is clear that if its assumptions hold, the proof of Theorem is done. First, $u(x(t))$ should admit an upper bound in l_2 -norm. In doing so, it suffices to show that its gradient is upper bounded in l_2 -norm. Considering Eq. (16) and (18), we have

$$\begin{aligned} \|u\|_2 &\le \max\{\|\mathbf{1}\|_2, \left\|-R^T \mathbf{e}_l\right\|_2\} \\ &= S \end{aligned} \tag{24}$$

hence u in l_2 -norm is bounded with at least S .

In the next step, we show that the Euclidian distance of the initial point to the optimal point is bounded at least with D , i.e.

$$\exists D > 0 \qquad \text{s.t.} \qquad \left\|x(1) - x^*\right\|_2 \le D$$

According to Eq. (3), we have $x_s > 0, \ \ \forall s \in S_{BE}$. On the other hand, optimal source rates are bounded at most with maximum value of link capacities, i.e.

$$\max_s x_s^* \le \max_l \hat{c}_l \le \max_l c_l \tag{25}$$

therefore,

$$\begin{aligned} \left\|x(1) - x^*\right\|_2 &\le \left\|\max x^* - \min x(1)\right\|_2 \\ &= \left\|\max_l c_l - 0\right\|_2 \\ &= L c_{l\max} \end{aligned} \tag{26}$$

and hence the initial Euclidian distance is bounded and Eq. (26) with Eq. (24) completes the proof.

5 Simulation Results

In this section we examine the proposed congestion control algorithm, listed above as Algorithm 1, for a typical NoC architecture. We have simulated a NoC with 4×4 Mesh topology which consists of 16 nodes communicating using 24 shared bidirectional links; each one has a fixed capacity of 1 Gbps. In our scenario, packets traverse the network on a shortest path using a deadlock free XY routing. We also assume that each packet consists of 500 flits and each flit is 16 bit long.

In order to simulate our scheme, some nodes are considered to have a Guaranteed Service data (such as Multimedia, etc.) to be sent to a destination while other nodes,

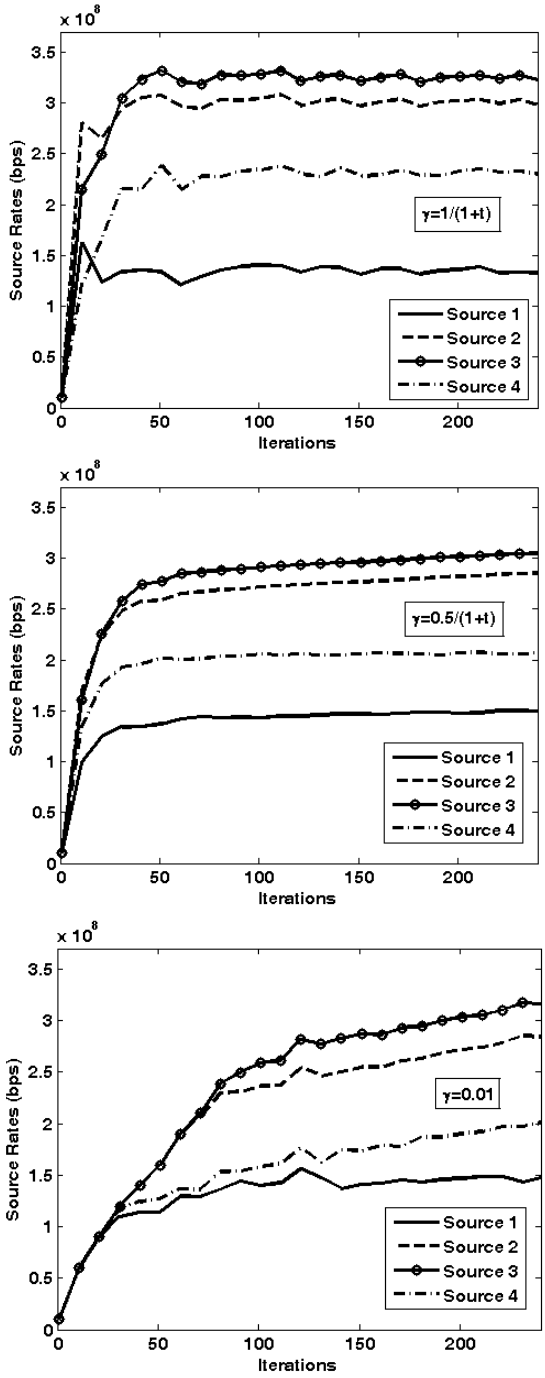

Fig. 1. Source rates for (a) $\gamma = \dfrac{1}{1+t}$, (b) $\gamma = \dfrac{0.5}{1+t}$ and (c) $\gamma = 0.01$

which maybe in the set of nodes with GS traffic, have a Best Effort traffic to be sent. As stated in section 2, GS sources will obtain the required amount of the capacity of links and the remainder should be allocated to BE traffics.

One of the most significant issues of our interest is the convergence behavior of the source rates. We used three different scenarios for step-size; two of them are chosen to be *square-summable but not summable* and the third is set to be constant. For the first two cases, stepsizes are chosen as $\gamma = 1/(1+t)$ and $\gamma = 0.5/(1+t)$ which satisfy Eq. (21)-(23). For the constant case, stepsize is set to be $\gamma = 0.01$. The first and second cases will be comparable with the constant stepsize after about 99 and 49 iterations, respectively.

Variation of source rates for some nodes using aforementioned stepsizes are shown in Fig. 1(a)-(c). Regarding Fig. 1(a), it's apparent that after about 50 iterations, all source rates will be in the vicinity of the steady state point of the algorithm. However, for the second case, Fig. 1(b) reveals that at least 80 iterations needed to have source rates in the vicinity of the optimal point. For the third case, the rate of convergence is even less and at least 150 iterations are needed to fall within the neighborhood of the steady state point of the algorithm. It is clear that compared to the *square-summable but not summable* stepsizes, constant stepsize has much slower rate of convergence. Comparing Fig. 1(a) and 1(b), we realize that the initial value of the stepsize, directly influences the rate of convergence.

In order to have a better insight about the algorithm behavior, the relative error with respect to optimal rates which averaged over all sources, is also shown in Fig. 2. Optimal source rates are obtained using CVX [18] which is MATLAB-based software for solving disciplined convex optimization problems. This figure reveals that *square-summable but not summable* stepsizes can lead to lower relative error in

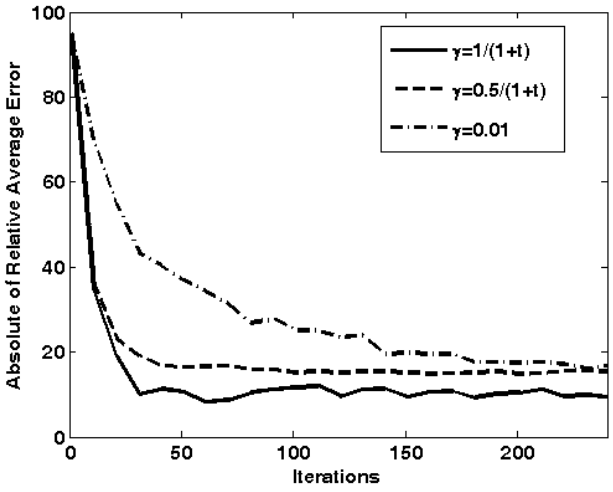

Fig. 2. Average of relative error with respect to optimal solution for the three cases

average with regard to constant stepsize. The faster rate of convergence of the first two cases than the third, can also be verified and it is apparent that the first case slightly acts better from the second in terms of averaged relative error.

6 Conclusion and Future Work

In this paper we addressed the problem of congestion control for BE traffic in NoC systems. Congestion control was modeled as the solution to the sum-rate maximization problem which was solved using subgradient method for constrained optimization problems. This was led to an iterative algorithm which determine optimal BE source rates. We have also studied the realization of the algorithm as a centralized congestion controller and presented a theorem to prove the convergence of the proposed congestion control scheme. Simulation results confirm that the proposed algorithm converges very fast and the computational overhead of the congestion control algorithm is small. Fast convergence of the algorithm also justifies that the delay incurred by the algorithm is very small. Further investigation about the effect of other utility functions on the BE rates and fairness provision is the main direction of our future studies.

References

1. Guerrier, P., Greiner, A.: A Generic Architecture for On-Chip Packet-Switched Interconnections. In: Proc. Design, Automation and Test in Europe Conference and Exhibition (DATE) (2000)
2. Dally, W.J., Towles, B.: Route Packets, Not Wires: On-Chip Interconnection Networks. In: Proc. DAC 2001 (2001)
3. Goossens, K., et al.: The Æthereal network on chip: Concepts, architectures, and implementations. IEEE Design and Test of Computers 22(5) (2005)
4. Bjerregaard, T., et al.: A router architecture for connection oriented service guarantees in the MANGO clockless Network-on-Chip. In: Proc. Design, Automation and Test in Europe Conference and Exhibition (DATE) (2005)
5. Bertozzi, D., et al.: Xpipes: A network-on-chip architecture for gigascale systems-on-chip. IEEE Circuits and Systems Magazine (2004)
6. Kelly, F.P., Maulloo, A., Tan, D.: Rate control for communication networks: Shadow prices, proportional fairness, and stability. J. Oper. Res. Soc. 49(3), 237–252 (1998)
7. Yang, C., et al.: A taxonomy for congestion control algorithms in packet switching networks. IEEE Network 9 (1995)
8. Low, S.H., Lapsley, D.E.: Optimization Flow Control, I: Basic Algorithm and Convergence. IEEE/ACM Transactions on Networking 7(6), 861–874 (1999)
9. Ogras, U., et al.: Prediction-based flow control for network-onchip traffic. In: Proc. DAC (2006)
10. Hu, J., et al.: DyAD - smart routing for networks-on-chip. In: Proc. DAC (2004)
11. van den Brand, J.W., Ciordas, C., Goossens, K., Basten, T.: Congestion- Controlled Best-Effort Communication for Networks-on-Chip. In: Proc. Design, Automation and Test in Europe Conference and Exhibition (DATE) (April 2007)

12. Talebi, M.S., Jafari, F., Khonsari, A.: Utility-Based Congestion Control for Best Effort Traffic in Network-on-Chip Architecture, (Submitted to MASCOTS 2007) (2007)
13. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge Univ. Press, Cambridge, U.K (2004)
14. Bertsekas, D.P.: Nonlinear Programming. Athena Scientific (1999)
15. Bertsekas, D.P., Tsitsiklis, J.N.: Parallel and distributed computation. Prentice-Hall, Englewood Cliffs (1989)
16. Boyd, S.: Convex Optimization II Lecture Notes. Stanford University (2006)
17. Poljak, B.T.: A General Method of Solving Extremum Problems. Soviet Math Doklady 8(3), 593–597 (1967)
18. Grant, M., Boyd, S., Ye, Y.: CVX (Ver. 1.0RC3): Matlab Software for Disciplined Convex Programming, Download available at: http://www.stanford.edu/ boyd/cvx