On the Stability of Best Effort Flow Control Mechanisms in On-Chip Architectures

Mohammad S. Talebi
School of Computer Science, IPM
mstalebi@ipm.ir

Ahmad Khonsari
ECE Department, University of Tehran
School of Computer Science, IPM
ak@ece.ut.ac.ir, ak@ipm.ir

Abstract

In this paper we present a centralized flow control scheme in NoCs in the presence of both elastic and streaming flow traffic paradigms. We model the desired Best Effort (BE) source rates as the solution to an α-fair utility maximization problem which is constrained with link capacities while preserving Guaranteed Service (GS) traffic requirements at the desired level. We propose an iterative algorithm as the solution to the aforementioned problem which has the benefit of low complexity and fast convergence. We also explore the stability and convergence behavior of the proposed algorithm and prove that it is globally asymptotically stable. Such an algorithm may be implemented by a centralized controller with low computation and communication overhead.

1. Introduction

The high level of system integration characterizing Multi-Processor Systems-on-Chip (MPSoCs) is raising the scalability issue for communication architectures. Towards this direction, traditional system interconnects based on shared busses are evolving both from the protocol and the topology viewpoint. Advanced bus protocols acts in favor of better exploitation of available bandwidth, while more parallel topologies are instead being introduced in order to provide more bandwidth [1]. In the long run, many researchers and SoC designers agree on the fact that this trend approaches the Network-on-Chip (NoC) as a solution to the lack of SoCs' Scalability [2].

Recently, Quality-of-Service (QoS) provisioning in NoC's environment has attracted many researchers and currently it is the focus of many literatures in NoC research community. NoCs are expected to serve as multimedia servers and are required not only to carry Best Effort (BE) traffic, but also Guaranteed Service (GS) traffic which requires tight performance constraints such as necessary bandwidth and maximum delay boundaries.

The Internet Engineering Task Force (IETF), realizing the limitations of the best-effort model, has undertaken serious steps to meet the QoS demand in the Internet infrastructure. Current achievements in integrating more processor cores on a single chip have made it possible to employ these MPSoCs as real time multimedia servers which require intensive computational power. Thus, it is imperative to provide in MPSoCs capabilities such as QoS which has been well available in traditional Internet servers. This implies that the underlying On-Chip communication will be required to provide deterministic bounds on delay and throughput for communication among communicating nodes on a chip. Congestion control as a critical means of providing QoS in traditional data networks is a well known issue and has been widely studied over the past two decades. However, it is still a novel problem in NoCs and to the best of our knowledge only few works has been carried out in this field [3]. Congestion control, or equivalently, end-to-end flow control in NoCs mainly focuses on the resource constrained on-chip designs, with the aim of maximizing network utility while preserving the required Quality-of-Service (QoS).

The rest of the paper is organized as follows. We discuss related work in Section 2. In Section 3 we present the system model and formulate the underlying optimization problem for BE flow control. In Section 4 we solve the underlying optimization problem for a class of utility functions and propose a flow control algorithm. In Section 5 we analyze the convergence behavior of the algorithm and also investigate its stability. Section 6 addresses the implementation aspects of the proposed flow control algorithm in NoCs. Section 7 presents the simulation results. Finally, Section 8 concludes the paper and states some future work directions.

2. Related Works

Flow control mechanisms have been well studied in traditional data networks [4]. A wide variety of flow control mechanisms in data network belongs to the class of end-to-end flow control schemes, like TCP/IP, which is mainly based on the window-based congestion control protocols. In these protocols, routers and intermediate nodes avoid the network from becoming congested by means of packet dropping deterministically (as in DropTail) or randomly (as in RED). Therefore, sent packets are subject to loss and the network must aim to providing an acknowledgment mechanism. On the other hand, On-chip networks pose different challenges. The reliability of on-chip wires allows NoCs to be loss-less. Therefore, there is no need to utilize acknowledgment mechanisms and we face to slightly different concept of flow control.

So far, several works have focused on this issue for NoC systems. In [3] a prediction-based flow-control strategy for on-chip networks is proposed in which each router predicts the buffer occupancy to sense congestion. This scheme controls the packet injection rate and regulates the number of packets in the network. In [5] link utilization is used as a congestion measure and a Model Prediction-based Controller (MPC), determines the source rates.

In this paper, we focus on the flow control for BE traffic as the solution to a utility-based optimization problem. To the best of our knowledge, none of the aforementioned works have dealt with the flow control problem through utility optimization approach. In our seminal work [6], we have modeled desired BE source rates as the solution to a utility-based optimization problem with a general form utility function and aimed at solving the problem using Newton method. In [7], we have also considered this issue via sum-rate optimization problem and used a different approach to solve the problem. In this paper we address the performance analysis of our seminal work [6] with a class of utility functions, known as α-fair functions, which satisfies nice fairness features. We focus on the solution of the flow control problem and investigate stability and convergence behavior of the solution.

3. System Model and Problem Formulation

We consider a NoC architecture which is based on a two dimensional mesh topology and wormhole routing. In wormhole-routed networks, each packet is divided into a sequence of *flits* which are transmitted over physical links one by one in a pipeline fashion. A hop-to-hop credit mechanism guarantees that a flit is transmitted only when the receiving port has free space in its input buffer. We also assume that the NoC architecture is lossless, and packets traverse the network on a shortest path using a deadlock free XY routing [2].

We model the flow control in NoC as the solution to an optimization problem. For the sake of convenience, we turn the aforementioned NoC architecture into a mathematically modeled network, as in [8]. In this respect, we consider NoC as a network with a set of bidirectional links $\mathcal{L} = \{1, 2, \ldots, L\}$ and a set of sources $\mathcal{S} = \{1, 2, \ldots, S\}$. A source consists of Processing Elements (PEs), Routers and Input/Output ports. Each link $l \in \mathcal{L}$ 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 bps. We denote the set of sources that share link l by $\mathcal{S}(l)$. Similarly, the set of links that source s passes through, is denoted by $\mathcal{L}(s)$. It is clear that $s \in \mathcal{S}(l)$ if and only if $l \in \mathcal{L}(s)$.

As discussed in Section 1, there are two types of traffic in a NoC: Guaranteed Service (GS) and Best Effort (BE). For notational convenience, we represent the BE and GS traffic rates by x_s and y_s, respectively. Each link $l \in \mathcal{L}$ is shared between the two traffics. GS traffics will obtain the required amount of link capacity and BE traffics benefit from the remainder.

We assume that source s upon transmitting BE traffic at x_s bps, will acquire a utility of $U_s(x_s)$. Our objective is to choose BE rates so that to maximize the sum of utilities of all BE traffics while satisfying capacity constraints. Hence the maximization problem can be formulated as [8]:

$$\max_{x_s} \sum_{s \in \mathcal{S}} U_s(x_s) \tag{1}$$

subject to:

$$\sum_{s \in \mathcal{S}(l)} x_s + y_s \leq c_l; \qquad \forall l \in \mathcal{L} \tag{2}$$

$$x_s \geq 0; \qquad \forall s \in \mathcal{S} \tag{3}$$

where U_s is a positive, strictly concave and increasing function of BE rate. Optimization variables are BE rates, which in vector form is denoted by $\mathbf{x} = (x_s, s \in \mathcal{S})$ and is in $\Re_+^S$. ($\Re_+^S$ denotes nonnegative real).

The constraint (2) states that the sum of BE rates passing through link l cannot exceed its free capacity, i.e. the portion of c_l which hasn't been allocated to GS traffic. With the above assumptions, problem (1) is a convex optimization problem with linear constraints. Hence it admits a unique maximizer [9] [10]; i.e. there exists an optimal source rate vector, $\mathbf{x}^*$, so that to maximize the sum of utilities in problem (1) while satisfying capacity constraints.

U_s in the economics literature is referred to as utility function, hence problem (1) is called a *utility maximization problem*. There are many choices for utility function with specific properties and behaviors. In this paper we will focus on a class of utility functions which are known to have nice fair properties in terms of economics terminology. These functions are known as α-fair utility functions and defined as [11]:

$$U(x, w_s, \alpha) = \begin{cases} w_s \frac{x_s^{1-\alpha}}{1-\alpha} & \alpha \neq 1 \\ w_s \log x_s & \alpha = 1 \end{cases} \tag{4}$$

where $\alpha > 0$ is a parameter and w_s is a weight factor assigned to source s.

Regarding the definition of α-fair utility functions (4), it can be easily verified that they satisfy abovementioned assumptions and thereby can be selected as the choice of utility function in problem (1). For notational convenience, we define $\hat{c}_l = c_l - \sum_{s \in \mathcal{S}(l)} y_s$. Also, for the sake of simplicity in our derivations throughout this paper, we define the routing matrix as $\mathbf{R} = [R_{ls}]_{L \times S}$, where R_{ls} is defined as

$$R_{ls} = \begin{cases} 1 & \text{if } l \in \mathcal{L}(s) \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

Thereafter, unless the otherwise is stated, we will follow this notation. Regarding this, for the aforementioned class of utility functions, problem (1) can be rewritten as

$$\max_{x_s} \sum_{s \in \mathcal{S}} w_s \frac{x_s^{1-\alpha}}{1-\alpha} \tag{6}$$

subject to:

$$\sum_s R_{ls} x_s \leq \hat{c}_l; \qquad \forall l \in \mathcal{L} \tag{7}$$

$$x_s \geq 0; \qquad \forall s \in \mathcal{S} \tag{8}$$

It is worthmentioning that although in (6), we have excluded the case of $\alpha = 1$; this case can be included as well, via evaluating the limit of $U(w, x, \alpha)$ when α approaches 1. For the sake of convenience, we carry out our analysis for $\alpha \neq 1$, keeping this in mind that if $\alpha = 1$ is the case, similar results would be obtained by evaluating the limit when α approaches 1.

4. Optimal Flow Control Algorithm

In this section, we solve (6) and derive the optimal flow control algorithm.

Although problem (6) can be separated among sources, its constraints will remain coupled across the links over the network. The coupled nature of such constrained problems, necessitates usage of centralized methods like *interior point method* which poses great computational overhead onto the system [9] [10].

One way to reduce the computational complexity is to transform the constrained optimization problem into an unconstrained one, which can be solved efficiently using several methods. According to the *duality theory* [9] [10], each convex maximization (minimization) problem has a *dual problem*. Regarding this terminology, the main problem is retroactively called *primal problem*. Optimal solution of the dual leads to an upper bound (lower bound) to the optima of the primal. With certain conditions (such as strong convexity) such an upper bound (lower bound) is tight and hence solving the dual is equivalent to solving the primal [9]. However, as the dual problem can be defined in such a way to be unconstrained, solving the dual is much simpler than the primal.

In the sequel, we will obtain the dual of problem (6) and solve it using an efficient iterative algorithm.

4.1. Deriving The Dual

We start by writing the Lagrangian of (6). Using the standard optimization methods [9], the Lagrangian of the problem (6) is given by

$$L(\mathbf{x}, \lambda) = \sum_s w_s \frac{x_s^{1-\alpha}}{1-\alpha} - \sum_l \lambda_l \left(\sum_s R_{ls} x_s - \hat{c}_l \right) \tag{9}$$

where λ_l is the positive Lagrange multiplier associated with the corresponding constraint of link l and $\lambda = (\lambda_l, l \in \mathcal{L})$ is the vector of Lagrange Multipliers and belongs to $\Re_+^L$. In economics literature, λ_l is called *shadow price* [8] for the interpretation of its role in solving the primal problem via its dual. We later will discuss about this issue.

Regarding the Lagrangian, the dual function is defined as [9]:

$$g(\lambda) = \max_{x_s} L(\mathbf{x}, \lambda) \tag{10}$$

Duality theory states that the optimal source rate vector, $\mathbf{x}^*$, corresponds to the optimal Lagrange multiplier vector, λ^* [9] [10]. In other words, if $\mathbf{x}$ is a feasible point of the primal problem and $\mathbf{x}$ is primal-optimal, the corresponding λ will be dual-optimal and vice versa. Therefore, at optimality we have

$$\nabla_x L(\mathbf{x}, \lambda)|_{(\mathbf{x}^*, \lambda^*)} = \mathbf{0} \tag{11}$$

where $\mathbf{0}$ is a vector with all zero. From (9), we have

$$\frac{\partial L}{\partial x_s}|_{(\mathbf{x}^*, \lambda^*)} = \frac{d}{dx_s} \left(w_s \frac{x_s^{1-\alpha}}{1-\alpha} \right) |_{x_s^*} - \sum_l R_{ls} \lambda_l^* = 0 \tag{12}$$

Hence, the optimal source rate is given by

$$x_s^* = \left(\frac{w_s}{\sum_l R_{ls} \lambda_l^*} \right)^{\frac{1}{\alpha}} \tag{13}$$

From (13) it's apparent that x_s^* is a decreasing function of λ_l; therefore λ_l can be construed as the price which must be paid for the source rate x_s. As the nature of such a price is hidden to the sources from the primal problem perspective, it is called *shadow price*. Substituting x_s^* into (9) yields

$$g(\lambda) = \sum_s \left(\frac{w_s^{\frac{1-\alpha}{\alpha}}}{1-\alpha} - w_s^{\frac{1}{\alpha}} \right) \left(\sum_l R_{ls} \lambda_l \right)^{\frac{\alpha-1}{\alpha}} + \sum_l \lambda_l \hat{c}_l \tag{14}$$

$$= \sum_l \lambda_l^{\frac{\alpha-1}{\alpha}} \sum_s R_{ls} \left(\frac{w_s^{\frac{1-\alpha}{\alpha}}}{1-\alpha} - w_s^{\frac{1}{\alpha}} \right) + \sum_l \lambda_l \hat{c}_l \tag{15}$$

The dual problem is defined as [9]:

$$\min_{\lambda_l \geq 0} g(\lambda) \tag{16}$$

The dual problem is always convex regardless of convexity or non-convexity of the primal problem. Moreover, the dual problem can be defined to be unconstrained or constrained with simple constraints, as with above. Thus, the primal has been transformed into an unconstrained convex optimization problem.

Convexity of the primal problem (6) guarantees strong duality. Therefore the *duality gap* is zero; i.e. solving the dual leads to the optimal point of the primal [9] [10]. Since dual problem is convex, it admits a unique minimizer, which can be obtained using iterative methods. As the dual problem is unconstrained; solving (16) using iterative methods is much simpler than the primal.

We postpone solving (16) to the next subsection.

4.2. Solving The Dual

In this subsection, we will solve the dual problem using *gradient projection method* [10].

The gradient projection method adjusts shadow prices, i.e. Lagrange multiplier vector, in opposite direction to the gradient of the dual function, i.e. $\nabla g(\lambda)$, as follows:

$$\lambda(k+1) = \Big[\lambda(k) - \gamma(k) \nabla g(\lambda(k))\Big]^+ \tag{17}$$

where $\gamma(k) > 0$ in general is a time-varying stepsize, and $[z]^+ = \max\{z, 0\}$. Since the objective of problem (6) is strictly concave, $g(\lambda)$ is continuously differentiable [10], hence $\nabla g(\lambda)$ exists. Using (15), the l-th element of the gradient vector is given by:

$$\frac{\partial g(\lambda)}{\partial \lambda_l} = \frac{\alpha-1}{\alpha} \lambda_l^{-\frac{1}{\alpha}} \sum_s R_{ls} \left(\frac{w_s^{\frac{1-\alpha}{\alpha}}}{1-\alpha} - w_s^{\frac{1}{\alpha}} \right) + \hat{c}_l$$

$$= \hat{c}_l - \sum_l R_{ls} x_s \tag{18}$$

where the last equation is obtained using (13) and algebraic manipulations. Furthermore, it can be proved that for a general form of U_s, the same result will be obtained. Therefore the update equation is given by:

$$\lambda_l(k+1) = \left[\lambda_l(k) - \gamma(k) \left(\hat{c}_l - \sum_s R_{ls} x_s(\lambda) \right)\right]^+ \tag{19}$$

where $x_s(\lambda(k))$ is the approximate of x_s^* in step k. In the next subsection, we propose a flow control algorithm based on the update equation (19).

4.3. Optimal Algorithm

In this subsection, we present a centralized flow control algorithm for BE traffic in NoC systems which controls the BE source rates in favor of problem (6). Regarding (19) and (13), it is clear that they form an iterative algorithm as the solution to problem (16) and thereby problem (6). In this respect, optimal source rates for BE sources can be found while satisfying capacity constraints and preserving GS traffic requirements. Thus, such an algorithm can be used to control the flow of the BE sources in the NoC. The proposed flow control algorithm is listed below as Algorithm 1.

Algorithm 1 has a decentralized nature and can also be addressed in distributed scenarios. However, due to well-formed structure of the NoC, we focus on a centralized scheme; a controller to be mounted in the NoC to implement this algorithm. The necessary requirements of such a controller is the ability to accomplish simple mathematical operations as in (19) and (13) and the allocation of few dedicated links to communicate congestion control information to nodes with a light GS load. Later, in Sec 6 we will discuss about the implementation aspects of such a controller.

5. Performance Analysis

5.1. Stability Analysis

In this subsection, we explore the stability of Algorithm 1. For the system in order to be robust against perturbations in real dynamic environments, e.g. sharp transitions, the underlying flow control mechanism must be stable. In the sequel, we show that Algorithm 1 is *globally asymptotically stable*; i.e. starting at any initial point and after passing enough iterations, it determines BE source rates which are in the neighborhood of the optimal rates of (6). Furthermore, after passing infinite iterations, Algorithm 1 results in the optimal source rates. In other words, for all initial conditions $(x_s(0), s \in \mathcal{S})$ we have

$$\lim_{k \to \infty} x_s(k) = x_s^*; \quad \forall s \in \mathcal{S} \tag{20}$$

Algorithm 1 Fair BE Flow Control in NoC

Initialization

Initialize the following items:

1. Sets of sources and links including the routing matrix.
2. $\hat{c}_l$ for $l \in L$.

Main Loop

Do until $\max_s |x_s(k+1) - x_s(k)| < \epsilon$

1. $\forall l \in \mathcal{L}$ Compute new link prices:
$\lambda_l(k+1) = \left[\lambda_l(k) - \gamma\left(\hat{c}_l - \sum_l R_{ls} x_s(k)\right)\right]^+$

2. $\forall s \in \mathcal{S}$ Compute new BE source rates as follows:
$x_s(k+1) = \left(\frac{w_s}{\sum_l R_{ls}\lambda_l(k+1)}\right)^{\frac{1}{\alpha}}$

Output

Communicate BE source rates to the corresponding sources.

Algorithm 1. Fair BE Flow Control in NoC

Towards this, we consider a continuous dynamic equation which models the behavior of the shadow price update equation (19). In this respect, under the assumption that the duration of time stamps can be arbitrarily small, we can interpret the $\lambda_l(k+1) - \lambda_l(k)$ as an approximate of the rate at which $\lambda_l(t)$ in the continuous time domain evolves, i.e.

$$\lambda_l(k+1) - \lambda_l(k) \approx \frac{d\lambda_l}{dt}\ |_{t=kT_s} \tag{21}$$

where T_s is the period of the sampling process or can be interpreted as the duration between two consequent iteration steps in the run of Algorithm 1. Based on the above discussion, shadow price update equation (19) in the continuous time model is given by

$$\frac{d\lambda_l}{dt} = -\gamma\left(\hat{c}_l - \sum_s R_{ls} x_s\right) u(\lambda_l(t)) \tag{22}$$

where $u(.)$ is the unit step function which is 1 for positive argument values and 0 otherwise.

In the sequel, we show that the abovementioned continuous-time system, whose discretized version approximates our system model, is globally asymptotically stable; i.e. for all initial conditions $(x_s(0), s \in \mathcal{S})$, we have $\lim_{t\to\infty} x_s(t) = x_s^*$. To show this, we establish a Lyapunov function for the dynamical system defined by (22) and prove that it is globally asymptotically stable. To prove this property, according to the Lyapunov's theorem we must show that the following conditions are satisfied [12]:

C1: The Lyapunov function, $V(\mathbf{x})$ is positive in its domain.

C2: $\frac{dV}{dt} < 0, \qquad \forall \mathbf{x} \neq \mathbf{x}^*.$

C3: $\frac{dV}{dt} = 0 \qquad$ only at $\qquad \mathbf{x} = \mathbf{x}^*.$

The following Theorem states the necessary condition under which we can find a proper Lyapunov function for the dynamical system defined by (22) admitting the conditions C1-C3.

Theorem 1 *Given the system (22), assume $U_s(.), \forall s \in \mathcal{S}$ is strictly concave and $\mathbf{R}$ has full row rank, i.e. given ρ, there exists a unique λ such that $\rho = \mathbf{R}^T\lambda$. Then the unique equilibrium point λ^* of (22), is globally asymptotically stable [13].*

Proof: See Appendix I for proof.

5.2. Convergence Analysis

In this subsection, we investigate the convergence behavior of Algorithm 1. As stepsize has an important role on the convergence behavior of the update equation (19), we mainly focus on the effect of stepsize. The conditions under which Algorithm 1 converges and performance analysis of the algorithm will be obtained with respect to the choice of stepsize.

There are several choices for stepsize, each one belonging to a predefined category and having certain advantages and drawbacks (see [10] and references herein). In the family of gradient algorithm 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. Due to its simplicity and robustness, in this paper we focus on the case of constant stepsize.

The following theorem determines the necessary condition on the stepsize, under which Algorithm 1 converges to the neighborhood of the optima of problem (16) and thereby that of problem (6).

Theorem 2 *The iterative flow control algorithm proposed by (13) and (19) converges to the neighborhood of the optimal point of the primal problem (6) provided that*

$$0 \leq \gamma \leq \frac{2\alpha\underline{w}}{\bar{c}^{\alpha+1}\bar{L}\bar{S}} \tag{23}$$

where $\bar{L}$ is the length of the longest path used by sources, $\bar{S}$ is the number of sources sharing the most congested link, $\underline{w}$ is the minimum weight of sources and $\bar{c}$ is the upper bound on link capacities.

Proof: Proof is omitted due to space limit.

6. Implementation Aspects

In this section we consider the implementation aspects of the proposed BE flow control algorithm.

As stated earlier, Algorithm 1 can be used as a centralized flow control mechanism for BE sources in NoC. In this respect, we consider a simple controller that can be mounted in the NoC, whether as a separate hardware module or a part of its operating system, which is responsible for running the algorithm. From computational complexity point of view, such a controller must have the ability of carrying out simple mathematical and logical operations, as in Algorithm 1. Another issue worth considering is the mechanism with which the controller communicates with BE sources. Since we would like source rate information be communicated without delay and loss, we designate to it several GS links to all sources with light load which can be implemented as a control bus, to communicate the algorithm outputs to BE sources.

Motivated by the end-to-end nature of Algorithm 1, we briefly discuss about the inherent connection of Algorithm 1 with those used for BE data transmission in distributed scenarios such as Internet, etc.

Algorithm 1 is very similar to end-to-end congestion control schemes in data networks, e.g. TCP variants which are widely used to control BE data flow in the Internet. Most of end-to-end schemes use the well-known window-based method, in which each source maintains a window of packets which are transmitted, but not acknowledged. Because packets in data networks may be lost due to dropping at the routers or losing because of link failures, destination should acknowledge the ordered receipt of them in the current window. Each source changes its window size in response to congestion signals, i.e. positive or negative acknowledges or duplicates ones, and thereby avoids the network to face congestion. Roughly Speaking, the source rate in each round trip (i.e. the way from a source to its destination and then back to the source for acknowledgment), is the ratio of window size to the RTT (i.e. duration of the round trip).

Although flow control in TCP is carried out by means of window updates, one can derive the corresponding rate updates, too. Algorithm 1 is very similar to rate update in TCP schemes. Such a close connection stems from the similarity in the underlying flow control problem in both schemes. However, it is worth noting that unlike TCP, in Algorithm 1 we have not considered any window-based transmission and acknowledgment mechanism. This is due to the fact that NoC architecture is lossless, as previously stated in Section 3, and hence all packets will be delivered successfully in the correct order and therefore no acknowledgment mechanism is needed.

7. Simulation Results

In this section we examine the proposed flow control algorithm, listed as Algorithm 1, for a typical NoC architecture. In our scenario, we have used 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 scheme, packets traverse the network on a shortest path using a deadlock free XY routing. We also assume that all sources have logarithmic utility function of the form $\log x_s$. We present our results in the following subsections.

7.1. Convergence Behavior

One of the most significant issues of our interest, is the convergence behavior of the source rates. In this subsection, we have simulated our scheme using 2 different values for step-size, $\gamma = 1.05$ and 0.2, respectively. Weight factor for all sources is assumed to be unity. The convergence behavior of source rates for after about 150 iterations is depicted in Fig. 1 and Fig. 2. Regarding Fig. 1, it's apparent that for $\gamma = 1.05$, after 20 iteration steps the source rates will have very little variations, however, from Fig. 2, i.e. for $\gamma = 0.2$, these threshold of iterations will be at least 85 steps.

In order to have a better insight about the algorithm behavior, the relative error with respect to optimal source rates which is averaged over all active sources, is also shown in Fig. 3. Optimal source rates are obtained using CVX [14] which is a MATLAB-based software for solving disciplined convex optimization problems. This figure reveals the first step size leads to less than $\%10$ error in average just after about 13 iteration steps, and after 20 steps the average error lies below $\%5$. However, the second step size would reach the two aforementioned error margins at the expense of iterating for about 60 and 75 steps, respectively. Although not shown in Fig. 3, with much more iteration steps simulation results verify that the average error curve for the smaller step size lies below that of larger step size.

7.2. Effect of The Weight

Another case of our consideration is the role of weight factor on resource (link capacity) sharing. It is trivial that network shares its resources in favor of sources with larger weight factors. In the next simulation experiment, we set the weight factor of source 2 and 7 to 20. Convergence behavior and steady state source rates are shown in Fig. 4. Comparing Fig. 4 and Fig. 1 reveals that using larger

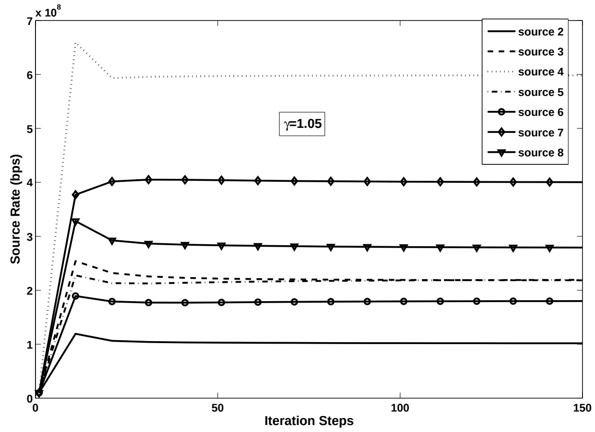

Figure 1. Source rates for $\gamma = 1.05$

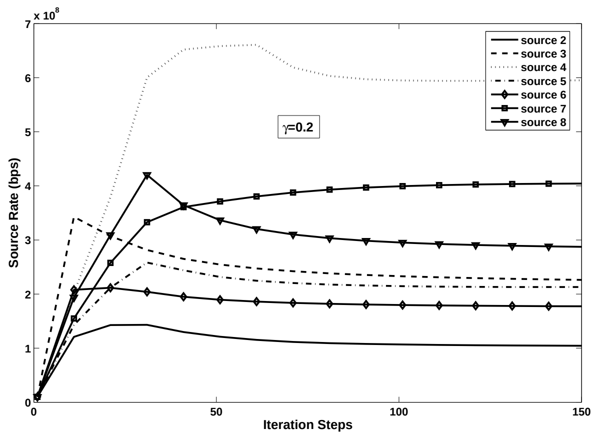

Figure 2. Source rates for $\gamma = 0.2$

weight factors, source 2 and 7 have achieved larger rates; however this is done at the expense of reducing the rate of some other nodes which shared bottleneck links with source 2 and 7. It is also worth mentioning that such an asymmetric case, adversely influences the speed of convergence.

8. Conclusion

In this paper we addressed the problem of α-fair flow control for BE traffic in NoC systems. Flow control was considered as the solution to the utility maximization problem which was solved indirectly through its dual using gradient projection method. This was led to globally asymptotically stable iterative algorithm which can be used to determine optimal BE source rates. The proposed algorithm can be implemented by a centralized controller which admits a light communication and communication overhead to the system. We have also investigated the convergence

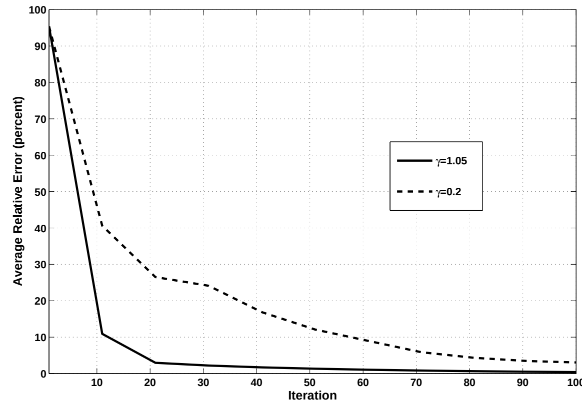

Figure 3. Average Relative Error

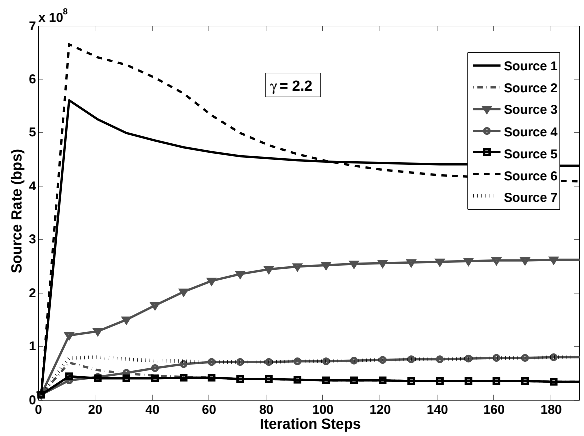

Figure 4. Source rate convergence for asymmetric weight factors

behavior of the algorithm. Further investigations of this research would be the development of a distributed controller and also exploring fairness properties for choosing different values of α.

References

[1] L. Benini and G. DeMicheli, "Networks on Chips: A New SoC Paradigm", *Computer*, vol. 35, no. 1, pp. 70-78, 2002.

[2] W. J. Dally and B. Towles, "Route Packets, Not Wires: On-Chip Interconnection Networks", *Design Automation Conference*, pp. 684-689, 2001.

[3] U. Y. Ogras and R. Marculescu, "Prediction-based flow control for network-on-chip traffic." *Design Automation Conference*, pp. 839- 844, 2006.

[4] C. Yang and A. V. S. Reddy, "A taxonomy for congestion control algorithms in packet switching networks", *IEEE Network*, vol. 9, no. 4, pp. 34-45, 1995.

[5] J. W. van den Brand, C. Ciordas, K. Goossens and T. Basten, "Congestion-Controlled Best-Effort Communication for Networks-on-Chip", *Design, Automation and Test in Europe Conference*, pp. 948-953, 2007.

[6] M. S. Talebi, F. Jafari and A. Khonsari, "A Novel Flow Control Scheme for Best Effort Traffic in NoC Based on Source Rate Utility Maximization", *In proceedings of the Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, pp. 381-386, 2007.

[7] M. S. Talebi, F. Jafari, A. Khonsari and M. H. Yaghmaee, "A Novel Congestion Control Scheme for Elastic Flows in Network-on-Chip Based on Sum-Rate Optimization." *International Conference on Computational Science and its Applications*, pp. 398-409, 2007.

[8] S. H. Low and D. E. Lapsley, "Optimization Flow Control I: Basic Algorithm and Convergence", *IEEE/ACM Transactions on Networking*, vol. 7, pp. 861-875, 1999.

[9] S. Boyd and L. Vandenberghe, Convex Optimization. Cambridge, U.K, Cambridge Univ. Press, 2004.

[10] Dimitri Bertsekas, Nonlinear Programming, Athena Scientific, 1999.

[11] J. Mo and J. Walrand, "Fair end-to-end window based congestion control," *IEEE/ACM Transaction on Networking*, vol. 8, no. 5, pp. 556-567, Oct. 1999.

[12] H. Khalil, Nonlinear Systems. Upper Saddle River, NJ: 2nd edition, Prentice Hall, 1996.

[13] F. Paganini, "On the stability of optimization-based flow control," *American Control Conference*, vol. 6, pp. 4689-4694, 2001.

[14] M. Grant, S. Boyd and Y. Ye, "CVX (Ver. 1.0RC3): Matlab Software for Disciplined Convex Programming". *Download available at: http://www.stanford.edu/ boyd/cvx*.

[15] S. Shakkottai and R. Srikant, Network Optimization and Control, Foundations and Trends in Networking, NOW Publishers, 2007.

Appendix I: Proof of Theorem 1

We adopt the proof from [15]. We consider the following Lyapunov function

$$V(\mathbf{x},\lambda) = \sum_l \left(\hat{c}_l - \sum_s R_{ls} x_s^*\right) \lambda_l$$

$$+ \sum_s \int_{\rho_s^*}^{\rho_s} \left(x_s^* - \left(\frac{w_s}{\sigma}\right)^{\frac{1}{\alpha}} \right) d\sigma \tag{24}$$

where

$$\rho_s = \sum_l R_{ls} x_s \tag{25}$$

To prove the stability of the system, we must show that $V(\mathbf{x},\lambda)$ satisfies conditions C1-C3. The satisfaction of C1 immediately results from the fact that both of $c_l - \sum_s R_{ls} x_s^*$ and λ_l are positive and $(\frac{w_s}{\sigma})^{1/\alpha}$ is decreasing function of σ. Next, we must prove that $\frac{dV}{dt}$ is non-positive. Therefore

$$\begin{aligned}\frac{dV}{dt} &= \sum_l \left(\hat{c}_l - \sum_s R_{ls} x_s^*\right) \frac{d\lambda_l}{dt} \\ &\quad + \sum_s \left(x_s^* - \left(\frac{w_s}{\rho_s}\right)^{\frac{1}{\alpha}} \right) \frac{d\rho_s}{dt} \\ &= (\hat{\mathbf{c}} - \mathbf{R}\mathbf{x}^*)^T \frac{d\lambda}{dt} + (\mathbf{x}^* - \mathbf{x})^T \frac{d\rho}{dt} \\ &= (\hat{\mathbf{c}} - \mathbf{R}\mathbf{x}^*)^T \frac{d\lambda}{dt} + (\mathbf{x}^* - \mathbf{x})^T \mathbf{R}^T \frac{d\lambda}{dt} \\ &= (\hat{\mathbf{c}} - \mathbf{R}\mathbf{x})^T \frac{d\lambda}{dt}\end{aligned}$$

Using (22), the last equality can be rewritten as

$$\begin{aligned}\frac{dV}{dt} &= (\hat{\mathbf{c}} - \mathbf{R}\mathbf{x})^T \frac{d\lambda}{dt} \\ &= -\sum_l \gamma \left(\hat{c}_l - \sum_s R_{ls} x_s\right)\left(\hat{c}_l - \sum_s R_{ls} x_s\right) u(\lambda_l) \\ &= -\sum_l \gamma \left(\hat{c}_l - \sum_s R_{ls} x_s\right)^2 u(\lambda_l) \le 0\end{aligned}$$

Therefore $V(\mathbf{x},\lambda)$ satisfies C2. It's worthmentioning that the last equation guarantees that the algorithm is stable even though source rates during the process of iterative algorithm might fall within intervals in which link capacity constraints can be violated. Moreover, $\frac{dV}{dt} = 0$ only occurs when a link, say link j, is saturated; i.e. for which $\sum_s R_{js} x_s = \hat{c}_j$ or $\sum_s R_{js} x_s < \hat{c}_j$ for which $\lambda_j = 0$. The two cases only occurs at the optimal Lagrange multiplier vector, λ^*, which is unique, as stated earlier. Strong duality guarantees that the dual-optimal, λ^*, leads to the primal-optimal $\mathbf{x}^*$. As a result, $\frac{dV}{dt} = 0$ has the unique root $\mathbf{x} = \mathbf{x}^*$. This fact implies that $V(\mathbf{x},\lambda)$ satisfies C3 and thereby the system described by (22) (and consequently the one described by Algorithm 1) is globally asymptotically stable.