Discounted Integral Priority Routing For Data Networks

Michael Zargham†, Alejandro Ribeiro†, Ali Jadbabaie†

Abstract—**A Discounted Integral Priority (DIP) packet routing algorithm is presented. The method is derived for the network flow model of packet routing used for the derivation of backpressure type methods. Unlike backpressure type methods, DIP routing is designed to reduce the queue lengths rather than simply stabilize them. Our work leverages time discounted integral control to generate an adaptive packet routing algorithm which significantly outperforms its optimization motivated counterparts. Connections are drawn with stochastic heavy ball methods which allow implementation of a decaying stepsize. Stability proofs are presented for a stochastic heavy ball variant of the Discounted Integral Priority routing algorithm with a decaying step size. Our numerical experiments implement Discounted Integral Priority Routing with a unit step size and demonstrate fast convergence and significantly smaller steady state queue backlogs as compared with Soft Backpressure and Accelerated Backpressure.**

I. Introduction

This paper considers the problem of joint routing and scheduling in packet networks. Packets are accepted from upper layers as they are generated and marked for delivery to intended destinations. To accomplish delivery of information nodes need to determine routes and schedules capable of accommodating the generated traffic. From a node-centric point of view, individual nodes handle packets that are generated locally as well as packets received from neighboring nodes. The goal of each node is to determine suitable next hops for each flow conducive to successful packet delivery.

The study of the joint scheduling and routing problem, has been built up from the Backpressure (BP) algorithm, [1]. In BP, nodes keep track of the number of packets in their local queues for each flow and share this information with neighboring agents. Nodes compute the differences between the number of packets in their queues and the number of packets in neighboring queues for all flows and assign the transmission capacity of the link to the flow with the largest queue differential. An alternative interpretation of BP is as a dual stochastic subgradient descent algorithm [2], [3]. This leads to a model of the joint scheduling and routing problem as an feasibility-type optimization of per-flow routing variables that satisfy link capacity and flow conservation constraints. Considering the Lagrange dual problem generates distributed algorithms to find stable operating points of wired [4]–[6] and wireless communication networks [7]–[9].

The convergence rate for subgradient descent methods is logarithmic in the number of iterations, [10]. This has lead to efforts to applying faster converging dual descent type algorithms. In [11], Soft Backpressure (SBP) includes an objective function in the feasibility problem. For appropriate choice of objective the subgradient becomes a unique gradient, improving convergence. In, [12]. Accelerated Backpressure (ABP) extends this idea of SBP further by choosing a strongly convex objective and implementing Accelerated Dual Descent (ADD), a distributed approximation to Newton's Method in place of gradient descent, [13]. SBP and ABP solve these stabilization problems much faster than BP but these algorithms but are unable to effectively clear backlogged queues even if the capacity to do so is available in the network.

In this paper, we shift into more control oriented thinking and observe that in order to eliminate large queues in steady state an integral control term is necessary. In developing Discounted Integral Priority (DIP) routing, we work within the Soft Backpressure framework but rather than using the queues themselves as routing priorities, we use a time discounted sum of the queue history. These priorities can be computed locally using a simple linear update combining the existing priorities and the newly observed queue lengths. Our method can be connected to a class of heavy ball methods like those introduced in [14]. By recasting the Discounted Integral Priority routing as a stochastic heavy ball update for the dual variables, we are able to implement a decaying step size which is a standard requirement for convergence in stochastic optimization. Our stability proofs leverage the decaying step size to prove queue stability under the stochastic heavy ball variant of the DIP algorithm. One of the main challenges in reducing the queues rather than just stabilizing them is the stochastic nature of the packet arrivals. Some related work includes noise cancelation in [15] and adaptive gradient methods in [16].

II. Preliminaries

Consider a given network $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ where $\mathcal{V}$ is the set of nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of links between nodes. Denote as C_{ij} the capacity of link $(i, j) \in \mathcal{E}$ and define the neighborhood of i as the set $n_i = \{j \in \mathcal{V} | (i, j) \in \mathcal{E}\}$ of nodes j that can communicate directly with i. There is also a set of information flows $\mathcal{K}$ with the destination of flow $k \in \mathcal{K}$ being the node $o_k \in \mathcal{V}$. Let $n = |\mathcal{V}|$ be the number of nodes and $K = |\mathcal{K}|$ be the number of information flows in the network and $E = |\mathcal{E}|$ be the number of edges in the network. Define the $n \times E$ matrix A to be the incidence matrix of the graph $\mathcal{G}$ and the reduced incidence matrix A_k as the $(n-1) \times E$ matrix with the row associated with destination node o_k removed. The block diagonal matrix $\bar{A}$ =diag$[A_k]$ is an $(n-1)K \times E \cdot K$ incidence matrix encoding the interrelation of all information flows and nodes in the network.

At time index t terminal $i \neq o_k$ generates a random number

$$a_i^k(t) = a_i^k + \nu_i^k(t) \tag{1}$$

This research is supported by Army Research Lab MAST Collaborative Technology Alliance, AFOSR complex networks program, ARO P-57920-NS, NSF CAREER CCF-0952867, and NSF CCF-1017454, ONR MURI N000140810747 and NSF-ECS-0347285.

†Michael Zargham, Alejandro Ribeiro and Ali Jadbabaie are with the Department of Electrical and Systems Engineering, University of Pennsylvania.

978-1-4799-3512-3/14/$31.00 ©2014 IEEE

of units of information to be delivered to o_k. The random variables $a_i^k(t) \geq 0$ are generated by $\nu_i^k(t)$ which are independent and identically distributed across time with $\mathbb{E}\left[\nu_i^k(t)\right] = 0$ and finite support. Thus the expected value $\mathbb{E}\left[a_i^k(t)\right] = a_i^k$ and $a_i^k(t)$ has finite support. In time slot $[t, t+1)$, node i routes $r_{ij}^k(t) \geq 0$ units of information through neighboring node $j \in n_i$ and receives $r_{ji}^k(t) \geq 0$ packets from neighbor j. The difference between the total number of received packets $a_i^k(t) + \sum_{j\in n_i} r_{ji}^k(t)$ and the sum of transmitted packets $\sum_{j\in n_i} r_{ij}^k(t)$ is added to the local queue – or subtracted if this quantity is negative. Therefore, the number $q_i^k(t)$ of k-flow packets queued at node i evolves according to

$$q_i^k(t+1) = \left[q_i^k(t) + a_i^k(t) + \sum_{j\in n_i} r_{ji}^k(t) - r_{ij}^k(t)\right]^+, \quad (2)$$

where the projection $[\cdot]^+$ into the nonnegative reals is necessary because the number of packets in queue cannot become negative. We remark that (2) is stated for all nodes $i \neq o_k$ because packets routed to their destinations are removed from the system.

To ensure packet delivery it is sufficient to guarantee that all queues $q_i^k(t+1)$ remain stable. In turn, this can be guaranteed if the average rate at which packets exit queues does not exceed the rate at which packets are loaded into them. To state this formally observe that the time average limit of arrivals satisfies $\lim_{t\to\infty} a_i^k(t) = \mathbb{E}\left[a_i^k(t)\right] := a_i^k$ and define the ergodic limit $r_{ij}^k := \lim_{t\to\infty} r_{ij}^k(t)$. If the processes $r_{ij}^k(t)$ controlling the movement of information through the network are asymptotically stationary, queue stability follows if

$$\sum_{j\in n_i} r_{ij}^k - r_{ji}^k \geq a_i^k + \xi \quad \forall\, k, i \neq o_k \quad (3)$$

for some constant $\xi > 0$ which is introduced because stability is guaranteed if the inequalities $\sum_{j\in n_i} r_{ij}^k - r_{ji}^k > a_i^k$ hold in a strict sense. For future reference define the vector $r := \{r_{ij}^k\}_{k,(i,j)}$ grouping variables r_{ij}^k for all information flows and links. Since at most C_{ij} packets can be transmitted in link (i, j) the routing variables $r_{ij}^k(t)$ always satisfy the capacity constraints on the network,

$$\sum_k r_{ij}^k(t) \leq C_{ij} \quad (4)$$

which defines the the set of possible routings

$$\mathcal{C} = \{r \in \mathbb{R}_+^{K\cdot E} : r_{ij}^k(t) \leq C_{ij} \forall (i,j) \in \mathcal{E}\}. \quad (5)$$

The joint routing and scheduling problem can be now formally stated as the determination of nonnegative variables $r(t) \in \mathcal{C}$ that satisfy (4) for all times t and whose time average limits r_{ij}^k satisfy (3). The BP algorithm solves this problem by assigning all the capacity of the link (i, j) to the flow with the largest queue differential $q_i^k(t) - q_j^k(t)$. Specifically, for each link we determine the flow pressure

$$k_{ij}^* = \underset{k}{\operatorname{argmax}} \left[q_i^k(t) - q_j^k(t)\right]^+. \quad (6)$$

If the maximum pressure $\max_k \left[q_i^k(t) - q_j^k(t)\right]^+ > 0$ is strictly positive we set $r_{ij}^k(t) = C_{ij}$ for $k = k_{ij}^*$. Otherwise the link remains idle during the time frame. The backpressure algorithm works by observing the queue differentials on each link and then assigning the capacity for each link to the data type with the largest positive queue differential, thus driving the time average of the queue differentials to zero– stabilizing the queues. To generalize, we reinterpret BP as a dual stochastic subgradient descent.

A. Dual stochastic subgradient descent

Since the parameters that are important for queue stability are the time averages of the routing variables $r_{ij}^k(t)$ an alternative view of the joint routing and scheduling problem is the determination of variables r_{ij}^k satisfying (3) and $\sum_k r_{ij}^k \leq C_{ij}$. This can be formulated as the solution of an optimization problem. Let $f_{ij}^k(r_{ij}^k)$ be any concave function on $\mathbb{R}_+$ and consider the optimization problem

$$\begin{aligned} r^* := \operatorname{argmax} \quad & \sum_{k, i\neq o_i, j} f_{ij}^k(r_{ij}^k) && (7)\\ \text{s.t.} \quad & \sum_{j\in n_i} r_{ij}^k - r_{ji}^k \geq a_i^k + \xi, \quad \forall\, k, i \neq o_k, \\ & \sum_{k\in\mathcal{K}} r_{ij}^k \leq C_{ij}, \qquad \forall\, (i,j) \in \mathcal{E}. \end{aligned}$$

where the domain is a the convex polyhedron defined in (5). Since only feasibility is important for queue stability, solutions to (7) ensure stable queues irrespectively of the objective functions $f_{ij}^k(r_{ij}^k)$. For notational compactness, define $f(r) = \sum_{k,(i,j)} f_{ij}^k(r_{ij}^k)$.

Since the problem in (7) is concave it can be solved by descending on the dual domain. Start by associating multipliers λ_i^k with the constraint $\sum_{j\in n_i} r_{ij}^k - r_{ji}^k \geq a_i^k$ and keep the constraint $r \in \mathcal{C}$ implicit. The corresponding Lagrangian associated with the optimization problem in (7) is

$$\mathcal{L}(r, \lambda) = \sum_{k, i\neq o_k, j} f_{ij}^k(r_{ij}^k) + \sum_{k, i\neq o_k} \lambda_i^k \left(\sum_{j\in n_i} r_{ij}^k - r_{ji}^k - a_i^k + \xi\right) \quad (8)$$

where we introduced the vector $\lambda := \{\lambda_i^k\}_{k,i\neq o_k}$ grouping variables λ_i^k for all flows and nodes. The corresponding dual function is defined as

$$h(\lambda) := \max_{r\in\mathcal{C}} \mathcal{L}(r, \lambda). \quad (9)$$

To compute a descent direction for $h(\lambda)$, compute the primal Lagrangian maximizers for given λ according to the vector function $R : \mathbb{R}_+^{(n-1)K} \to \mathcal{C}$ defined

$$R(\lambda) := \underset{r\in\mathcal{C}}{\operatorname{argmax}}\, \mathcal{L}(r, \lambda). \quad (10)$$

The individual elements $R_{ij}^k(\lambda)$ are ordered by stacking the E dimensional subvectors for each information flow k. A descent direction for the dual function is available in the form of the dual subgradient[1] are obtained by evaluating the constraint slack associated with the Lagrangian maximizers

$$[\nabla h(\lambda)]_i^k := \sum_{j\in n_i} R_{ij}^k(\lambda) - R_{ji}^k(\lambda) - a_i^k - \xi. \quad (11)$$

[1]For an appropriately chosen cost function $f(r)$, the subgradient is unique which motives the use of gradient notation $\nabla h(\lambda)$, (see Proposition 1).

Since the Lagrangian $\mathcal{L}(r,\lambda)$ in (8) is linear in the dual variables λ_i^k the determination of the maximizers $R_{ij}^k(\lambda) := \operatorname{argmax}_{r\in\mathcal{C}} \mathcal{L}(r,\lambda)$ can be decomposed into the maximization of separate summands. Considering the coupling constraints $\sum_k r_{ij}^k \le C_{ij}$ imposed by the domain $\mathcal{C}$ it suffices to consider variables $\{r_{ij}^k\}_k$ for all flows across a given link. After reordering terms it follows that we can compute routes link wise

$$R_{ij}^k(\lambda) = \underset{\{r_{ij}^k\ge 0\}_k}{\operatorname{argmax}} \sum_k f_{ij}^k(r_{ij}^k) + r_{ij}^k\left(\lambda_i^k - \lambda_j^k\right) \qquad \text{s.t.} \quad \sum_{k\in\mathcal{K}} r_{ij}^k \le C_{ij} \tag{12}$$

for each link $(i,j)\in\mathcal{E}$. Introducing a time index t, subgradients $\nabla h_i^k(\lambda_t)$ could be computed using (11) with Lagrangian maximizers $R_{ij}^k(\lambda_t)$ given by (12). For notational convenience, the dual subgradient may also be specified in vector form

$$\nabla h(\lambda) = \bar{A}R(\lambda) - a - \xi\mathbf{1} \tag{13}$$

where $\mathbf{1}$ is the vector of all ones. A subgradient descent iteration could then be defined to find the variables r^* that solve (7) via a dual method which generates a sequence λ_t; see e.g., [17].

The problem in computing $\nabla h_i^k(\lambda)$ is that we don't know the average arrival rates a_i^k. We do observe, however, the instantaneous rates $a_i^k(t)$ that are known to satisfy $\mathbb{E}\left[a_i^k(t)\right] = a_i^k$. Therefore,

$$[g_t(\lambda)]_i^k := \sum_{j\in n_i} R_{ij}^k(\lambda) - R_{ji}^k(\lambda) - a_i^k(t) - \xi, \tag{14}$$

is a stochastic subgradient of the dual function in the sense that its expected value $\mathbb{E}\left[g_i^k(\lambda)\right] = \nabla h_i^k(\lambda)$ is the subgradient defined in (11). Stated in vector form

$$g_t(\lambda) = \bar{A}R(\lambda) - a - \nu_t - \xi\mathbf{1}, \tag{15}$$

where $a_t = a + \nu_t$ and $\mathbb{E}[\nu_t] = 0$ for all t. We can then minimize the dual function using a stochastic subgradient descent algorithm. At time t we have multipliers λ_t and determine Lagrangian maximizers $r_{ij}^k(t) := R_{ij}^k(\lambda_t)$ as per (12). We then proceed to update multipliers along the stochastic subgradient direction according to

$$\lambda_i^k(t+1) = \left[\lambda_i^k(t) - \epsilon\left(\sum_{j\in n_i} r_{ij}^k(t) - r_{ji}^k(t) - a_i^k(t) - \xi\right)\right]^+, \tag{16}$$

where ϵ is a constant stepsize chosen small enough so as to ensure convergence; see e.g., [11].

Properties of the descent algorithm in (16) vary with the selection of the functions $f_{ij}^k(r_{ij}^k)$. Two cases of interest are when $f_{ij}^k(r_{ij}^k) = 0$ and when $f_{ij}^k(r_{ij}^k)$ are continuously differentiable, strongly convex, and monotone decreasing on $\mathbb{R}_+$ but otherwise arbitrary. The former allows recovers the Backpressure Algorithm while the latter leads to the Soft Backpressure algorithm.

B. Soft backpressure

Assume now that the functions $f_{ij}^k(r_{ij}^k)$ are continuously differentiable, strongly convex, and monotone decreasing on $\mathbb{R}_+$ but otherwise arbitrary. In this case the derivatives $\partial f_{ij}^k(x)/\partial x$ of the functions $f_{ij}^k(x)$ are monotonically increasing and thus have inverse functions that we denote as

$$F_{ij}^k(x) := \left[\partial f_{ij}^k(x)/\partial x\right]^{-1}(x). \tag{17}$$

The Lagrangian maximizers in (12) can be explicitly written in terms of the derivative inverses $F_{ij}^k(x)$. Furthermore, the maximizers are unique for all λ implying that the dual function is differentiable. The details are outlined in Proposition 1 originally published in [12].

Proposition 1. *If the functions $f_{ij}^k(r_{ij}^k)$ in (7) are continuously differentiable, strongly concave, and monotone decreasing on $\mathbb{R}_+$, the dual function $h(\lambda) := \max_{r\in\mathcal{C}} \mathcal{L}(r,\lambda)$ is differentiable for all λ. Furthermore, the gradient component along the λ_i^k direction is $[\nabla h(\lambda)]_i^k$ as defined in (11) with*

$$R_{ij}^k(\lambda) = F_{ij}^k\left(-\left[\lambda_i^k - \lambda_j^k - \mu_{ij}(\lambda)\right]^+\right), \tag{18}$$

where $\mu_{ij}(\lambda)$ is either 0 if $\sum_k F_{ij}^k\left(-\left[\lambda_i^k - \lambda_j^k\right]^+\right) \le C_{ij}$ or chosen as the solution to the equation

$$\sum_k F_{ij}^k\left(-\left[\lambda_i^k - \lambda_j^k - \mu_{ij}(\lambda)\right]^+\right) = C_{ij}. \tag{19}$$

While (19) does not have a closed for solution it can be computed quickly numerically using a binary search because it is a simple single variable root finding problem. Computation time cost remains small compared to communication time cost.

The Soft backpressure can be implemented using node level protocols. At each time instance nodes send their multipliers $\lambda_i^k(t)$ to their neighbors. After receiving multiplier information from its neighbors, each node can compute the multiplier differentials $\lambda_i^k(t) - \lambda_j^k(t)$ for each edge. The nodes then solve for μ_{ij} on each of its outgoing edges by using a rootfinder to solve the local constraint in (19), The capacity of each edge is then allocated to the unique information flows via reverse waterfilling as defined in (18). Once the transmission rates are set each node can observe its net packet gain which is equivalent to the stochastic gradient as defined in (14). Finally, each node updates its multipliers by subtracting ϵ times the stochastic subgradient from its current multipliers. As with BP, choosing the stepsize $\epsilon = 1$ causes the multipliers to coincide with the queue lengths for all time.

C. A General Priority-Based Routing Strategy

The Backpressure and and Soft Backpressure algorithms, as detailed in the preceding sections, are fundamentally priority based routing strategies. When viewed in the optimization framework these priorities are dual variables but the priority-based routing strategy $R(\lambda)$ defined in (10) can be implemented for any priority vector λ. We define a general priority-based routing strategy

$$r_t = R(\lambda_t) \text{ for any sequence } \lambda_t,\ \forall t. \tag{20}$$

A priority based routing strategy based on (20) generates a sequence of queue lengths

$$q_{t+1} = q_t - g_t(\lambda_t) - \xi\mathbf{1} \tag{21}$$

where the stochastic gradient function, $g_t(\lambda)$ is defined in (14). The $\xi\mathbf{1}$ offset guarantees that priorities λ_t satisfying $g_t(\lambda_t) \ge 0$

result in a strict reduction. Substituting the definition of the stochastic gradient

$$q_{t+1} = q_t - \bar{A}R(\lambda_t) + a + \nu_t. \tag{22}$$

We define the queue update

$$\Delta q_t(\lambda_t) = \bar{A}R(\lambda_t) - a - \nu_t. \tag{23}$$

The routing strategy $R(\lambda_t)$ for any sequence $\{\lambda_t\}_{t=0}^{\infty}$ generalizes the notion of Soft Backpressure, [11] to a case where the priorities can be generated by any desirable scheme. Another example of a priority based routing algorithm is Accelerated Backpressure, [12]. Priority sequences $\{\lambda_t\}_{t=0}^{\infty}$ are not all equally effective. Determination of sufficient conditions for a priority sequence $\{\lambda_t\}_{t=0}^{\infty}$ to yield stable queue lengths is an open problem.

III. Discounted Integral Priority Based Routing

Observing that existing protocols for packet forwarding which use optimization motivated priorities tend to stabilize but not reduce the queues, we propose a set of priorities motivated by integral control. In particular, since the queues are state variables which we would like to make small at steady state, we set the priorities to a discounted integral of the queues. Consider the set of routing priorities

$$\lambda_t = \sum_{\tau=0}^{t} \alpha^{t-\tau} q_t \tag{24}$$

where $\alpha \in [0, 1)$ is a discounting factor. Observe that for if $\alpha = 0$, the soft backpressure algorithm $\lambda_t = q_t$ is trivially recovered. Backpressure type algorithms are implemented by observing the stochastic gradients $g_t(\lambda_t)$ in order to update the routing variables. In our method, q_t is observed and the priorities are updated

$$\lambda_t = \alpha\lambda_{t-1} + q_t, \tag{25}$$

which can be computed without information from neighboring nodes: $\lambda_i^k(t) = \alpha\lambda_i^k(t-1) + q_i^k(t)$ for all nodes i, information flows k and times t. Under this scheme we do not force any information exchange, rather we allow information to spread through the effect of the changes in the realized routing. See Algorithm 1 for the distributed implementation of the priority based routing $r_t = R(\lambda_t)$ using the discounted integral priorities $\{\lambda_t\}_{t=0}^{\infty}$ chosen according to equation (24).

A. Connection to Heavy Ball Methods

The discounted integral priority update can be expressed as a variant on the heavy ball method. A variety of closely related momentum based algorithms are considered *heavy Ball Methods*, the primary examples are Nesterov's accelerated method, [14] and Polyak's Method, [18].

Lemma 1. *The discounted Integral Priority update in* (25) *can be rewritten as*

$$\lambda_{t+1} = \lambda_t + \Delta q_t(\lambda_t) + \alpha(\lambda_t - \lambda_{t-1}) \tag{26}$$

where $\Delta q_t(\lambda_t)$ is the change in queues induced by routing according to the priorities λ_t and the discounting parameter α is weighting coefficient on the momentum term.

Algorithm 1: Discounted Integral Priority Based Routing

for $t = 0, 1, 2, \cdots$ **do**
  Observe $\{q_i^k(t)\}_k$,
  Compute $\lambda_k^i(t) = \alpha\lambda_i^k(t-1) + q_i^k(t)$
  for *all neighbors* $j \in n_i$ **do**
    Send priorities $\{\lambda_i^k(t)\}_k$ – Receive priorities $\{\lambda_j^k(t)\}_k$
    Compute μ_{ij} such that

$$\sum_k F\left(-[\lambda_i^k(t) - \lambda_j^k(t) - \mu_{ij}]^+\right) = C_{ij}$$

    Transmit packets at rate

$$r_{ij}^k(t) = F(-[\lambda_i^k(t) - \lambda_j^k(t) - \mu_{ij}]^+)$$

  end
end

Proof : Stating equation (25) at time $t + 1$ yields

$$\lambda_{t+1} = \alpha\lambda_t + q_{t+1}. \tag{27}$$

Subtracting (25) from (27) yields

$$\lambda_{t+1} = \lambda_t + (q_{t+1} - q_t) + \alpha(\lambda_t - \lambda_{t-1}). \tag{28}$$

From the definitions in (22) and (23), the change in the queues $\Delta q_t(\lambda) = q_{t+1} - q_t$ completing the proof. ■

From Lemma 1, we see that the discounted integral priority method is equivalent to applying the heavy ball method to a basic soft backpressure algorithm. Soft back pressure in our notation is $q_{t+1} = q_t - \bar{A}R(q_t) + a + \nu_t$, because the priorities and the queues are equivalent for all t, [11]. Unlike the basic soft backpressure algorithm the queues and the priorities do not remain equivalent. Heavy ball type methods are known to significantly improve convergence rates.

B. Stochastic Heavy Ball Routing

The heavy ball method is analyzed for stochastic optimization in [19]. In order to guarantee convergence in stochastic optimization, it is necessary to introduce a decaying step size ϵ_t satisfying $\sum_{t=1}^{\infty} \epsilon_t = \infty$ and $\sum_{t=1}^{\infty} \epsilon_t^2 < \infty$. Motivated by Lemma 1, we introduce Stochastic Heavy Ball routing

$$\lambda_{t+1} = \lambda_t - \epsilon_t g_t(\lambda_t) + \alpha(\lambda_t - \lambda_{t-1}) \tag{29}$$

which is a variant of the Discounted Integral Priority routing algorithm, which allows implementation of a decaying step size ϵ_t. This minor variation allows the direct application of Theorem 2 from [19] from which we conclude that λ_t converges with probability one, to a solution of the dual problem in (9). In the following section, it is shown that convergence of the priorities is sufficient to stabilize the queues.

IV. Stability Analysis

The heavy ball variant of the discounted integral priority method with update as defined in (29) allows in the inclusion of a decaying step size which is standard in the analysis of stochastic optimization algorithms. Another key property is the boundedness of the stochastic gradient.

Lemma 2. *The stochastic gradient is bounded*

$$\|g_t(\lambda)\| \leq \gamma, \qquad \forall t, \, \forall \lambda. \tag{30}$$

The analysis that follows is related to the stochastic convergence analysis for the Accelerated Backpressure (ABP) algorithm as detailed in [20]. As in ABP, the results are built on a specialized version of the supermartingale convergence theorem found in [21, Theorem E.7.4]. The proofs are noticeably simpler than there ABP counterparts because there is no need to characterize the second derivatives of the dual objective.

Proposition 2. *Consider the Stochastic Heavy Ball Routing implemented with the dual step* (29)*, decaying step size* ϵ_t *satisfying* $\sum_t \epsilon_t = \infty$, $\sum_t \epsilon_t^2 < \infty$ *and the stochastic gradient* $g_t(\lambda_t)$ *as defined in* (14)*, then*

$$\lim_{t\to\infty} ||\nabla h(\lambda_t)|| = 0 \tag{31}$$

almost surely.

Proposition 2 is the first step in our theoretic convergence guarantee. By implementing a decaying step size when updating the dual variables, convergence to the optimal dual variables is achieved. From Theorem 2 of [19], priority vectors λ_t converge with probability one and our routing $R_{ij}^k(\lambda_t)$ becomes a feasible routing in the primal problem, (7). We proceed to leverage this fact to ensure the queues remain stable.

Proposition 3. *Consider a dual variables process* λ_t *such that the dual gradient* $\nabla h(\lambda_t)$ *satisfies*

$$\lim_{t\to\infty} ||\nabla h(\lambda_t)|| = 0, \quad \text{a.s.} \tag{32}$$

Then, all queues empty infinitely often with probability one, i.e.,

$$\liminf_{t\to\infty} q_i^k(t) = 0, \quad \text{a.s.}, \qquad \text{for all } k, i \neq o_k \tag{33}$$

Corollary 1. *Consider the Stochastic Heavy Ball Routing Algorithm, defined by* (29) *with* $g_t(\lambda_t)$ *as defined in* (14) *and primal Lagrangian maximizers defined in* (12)*. With step size sequence is chosen to satisfy* $\sum_t \epsilon_t = \infty$ *and* $\sum_t \epsilon_t^2$*, all queues become empty infinitely often with probability one,*

$$\liminf_{t\to\infty} q_i^k(t) = 0, \quad \text{a.s.}, \qquad \text{for all } k, i \neq o_k. \tag{34}$$

Corollary 1 is a sufficient condition for queue stability. In fact it is a much stronger result because the queues are always eventually cleared. This occurs because our dual variables converge to the optimal priorities which yield a routing $R_{ij}^k(\lambda_k)$ which forces the queue lengths to decrease in expectation whenever they are non-zero.

TABLE I
AVERAGE QUEUE BACKLOG STATISTICS

Algorithm	BP	SBP	ABP	DIP
Mean	74.13	16.22	13.92	4.38
St. Dev.	11.52	0.92	2.37	0.02

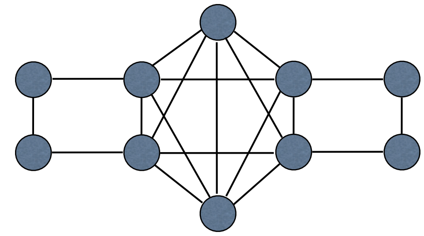

Fig. 1. Several numerical experiments for the Discount Integral Priority (DIP) routing algorithm presented in this section are performed on this simple 10 node network with 5 data types. The destinations are unique for each data type and are chosen randomly.

V. NUMERICAL RESULTS

The main benefit of introducing discounted integral priorities is the ability to drive large queues out of the system. The performance of the DIP routing in its simplest form, as defined in (25) is compared with the performance of the backpressure based queue stabilization methods defined in [11] and [12].

Our experiments take place on the network in Figure 1, with edge capacities chosen uniformly randomly in $(0, 50]$, the strictness inequality parameter $\xi = 0$ and there are $K = 5$ data types with unique destination nodes. Each queue starts with an expected initial backlog of 10 packets. The arrival process has an expectation of 5 packets per queue which results in an expected $5(n-1)K = 45$ packets entering the system at each time. The resulting problem (7) is feasible but non-trivial. Our choice of objective function is

$$f_{ij}^k(r_{ij}^k) = -\frac{1}{2}\left(r_{ij}^k\right)^2 + \beta_{ij}^k r_{ij}^k. \tag{35}$$

The quadratic term captures an increasing cost of routing larger quantities of packets across a link and help to eliminate myopic routing choices that lead to sending packets in cycles. The linear term β is introduced to reward sending packets to their destinations. In our simulations, $\beta_{ij}^k = 10$ for all edges routing to their respective data type destinations $j = o_k$ and all other i, j, k, $\beta_{ij}^k = 0$. Figure 2, is a characteristic example of the relative behaviors of the algorithms being studied. Backpressure (BP) stabilizes the queues but the total number of backlogged packets remains large. Accelerated Backpressure (ABP) stabilizes the queues more quickly that Soft Backpressure (SBP) but neither do any work to remove existing queue build up. Discounting Integral Priority (DIP) routing however drives the total packets queues significantly below their starting levels.

In order to more precisely characterize the relative behavior of these algorithms, we repeat the numerical experiment 100 times and record the *average-average* backlog. Two averages are used, we take a time average over 500 iterations to account for fluctuations and we average over the number of queues. Thus the data being presented is the average number of packets in each queue over the whole trial. Figure 3 is the distribution over all 100 trials. The key observation is that the Discounted Integral Priority (DIP) routing algorithm results in an average queue length of well below the initial average queue length while the other algorithms are at best close to the starting value. In fact the average backlog is close to the average arrival rate. Another important observation is highlighted in Table I; these results are achieved with an extremely tight variance over

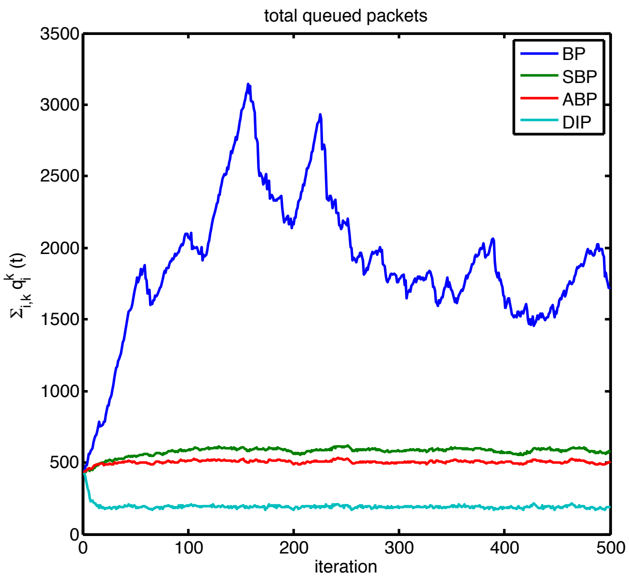

Fig. 2. ABP and SBP stabilize the queues much more effectively than BP, but DIP routing drives the queues down and keeps them stable there.

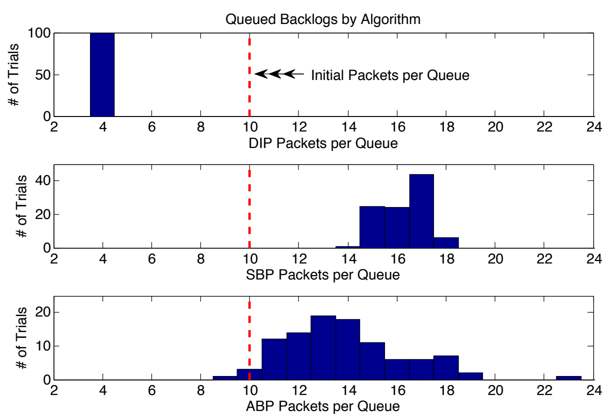

Fig. 3. ABP and SBP stabilize the queues relatively quickly, preventing them from getting too far above the initial queues. ABP is faster than SBP but exhibits more variance. DIP routing however drives the queues well below the initial queue lengths.

many trials. Every realization of the problem has different edge capacities but the same arrival statistics. The implication is that the Discounted Integral Priorities successfully learn information about the arrival statistics and the remaining steady state backlogs result from the fact that the arrivals are stochastic.

VI. CONCLUSION

The Discounted Integral Priority (DIP) routing algorithm is a significant improvement over its optimization motivated backpressure-type counterparts. Most notably, it out performs Accelerated Backpressure with a significantly simpler update scheme. This is achieved by introducing the notion of integral control, which characteristically drives out the steady state error.

The DIP algorithm is shown to be closely related to stochastic heavy ball methods which allows formal guarantees for queue clearing when implementing the variant which includes a decaying step size. The numerical experiments demonstrate that even without including the decaying step size, the DIP algorithm drives down the steady state queues.

Future direction for this work includes expanding the analysis to prove not only queue reduction but also convergence rates. Also, we plan to explore the DIP routing's ability to handle periodic large arrival shocks, something stabilizing algorithms struggle with. Based on our current results, we can expect good performance as long as the shocks are not so frequent that they make the problem infeasible in expectation.

REFERENCES

[1] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, 37:1936–1948, 1992.
[2] L. Georgiadis, M. J. Neely, and L. Tassiulas. Resource allocation and cross-layer control in wireless networks. *Foundations and Trends in Networking*, 1:1–144, 2006.
[3] M. J. Neely, E. Modiano, and C. E. Rohrs. Dynamic power allocation and routing for time varying wireless networks. *IEEE Journal on Selected Areas in Communications, Special Issue on Wireless Ad-Hoc Networks*, 23:89–103, 2005.
[4] F.P. Kelly, A.K. Maulloo, and D.K. Tan. Rate control for communication networks: shadow prices, proportional fairness, and stability. *Journal of the Operational Research Society*, 49:237–252, 1998.
[5] S. Low and D.E. Lapsley. Optimization flow control, I: Basic algorithm and convergence. *IEEE/ACM Transactions on Networking*, 7(6):861–874, 1999.
[6] R. Srikant. *Mathematics of Internet Congestion Control*. Birkhauser, 2004.
[7] M. Chiang, S.H. Low, A.R. Calderbank, and J.C. Doyle. Layering as optimization decomposition: A mathematical theory of network architectures. *Proceedings of the IEEE*, 95(1):255–312, 2007.
[8] M. J. Neely. Energy optimal control for time varying wireless networks. *IEEE Transactions on Information Theory*, 52:2915–2934, 2006.
[9] A. Ribeiro and G. Giannakis. Separation principles in wireless networking. *IEEE Transactions on Information Theory*, 58:4488–4505.
[10] A. Nedić and A. Ozdaglar. Distributed subradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control, forthcoming*, 2008.
[11] A. Ribeiro. Stochastic soft backpressure algorithms for routing and scheduling in wireless ad-hoc networks. In *IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, 2009.
[12] M. Zargham, A. Ribeiro, , and A. Jadbabaie. Accelerated backpressure algorithm. *Proceedings of the IEEE GLOBECOM*, 2013.
[13] M. Zargham, A. Ribeiro, A. Ozdaglar, and A. Jadbabaie. Accelerated dual descent for network optimization. *IEEE Transactions on Automatic Control*, 2014.
[14] Yurii Nesterov. A method of solving a convex programming problem with convergence rate o (1/k2). *Soviet Mathematics Doklady*, 27(2):372–376, 1983.
[15] Amir Beck and Marc Teboulle. Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems. *Image Processing, IEEE Transactions on*, 18(11):2419–2434, 2009.
[16] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12:2121–2159, July 2011.
[17] A. Ribeiro and G. B. Giannakis. Separation theorems of wireless networking. *IEEE Transactions on Information Theory*, 2007.
[18] B. T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.
[19] Sjur Didrik Flam. Optimization under uncertainty using momentum. *Lecture Notes in Economics and Mathematical Systems*, pages 249–256, 2004.
[20] M. Zargham, A. Ribeiro, and A. Jadbabaie. Accelerated backpressure algorithm. *IEEE Transactions on Signal Processing (submitted)*, 2014.
[21] V. Solo and X. Kong. *Adaptive Signal Processing Algorithms: Stability and Performance*. Prentice-Hall, 1995.
[22] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK, 2004.