Event-Triggered Optimization

Michael Lemmon, University of Notre Dame

October 6, 2009

This section introduces an event-triggered distributed algorithm that solves network utility maximization (NUM) problems in large-scale networked systems [18, 17]. Existing distributed algorithms for the NUM problem are gradient-based schemes whose convergence to the optimal point provided the communication between subsystems is sufficiently frequent. Analytic bounds on the communication interval required to ensure convergence tend to be inversely proportional to certain measures of network complexity such as network diameter and connectivity. As a result, the total message passing complexity in such algorithms can be very great. The event-triggered algorithm presented in this section appears to reduce the message passing complexity by nearly two-orders of magnitude. Moreover, experimental results indicate that this complexity is *scale-free* with regard to network diameter and connectivity.

Related Work: Many problems in networked systems can be formulated as optimization problems. This includes estimation [13] [14] [5], source localization [13], data gathering [2] [1], routing [9], control [16], resource allocation [12] [21] in sensor networks, resource allocation in wireless communication networks [20] [3], congestion control in wired communication networks [6] [8], and optimal power dispatch [7] in electrical power grid. The consensus problem [10] can also be viewed as a distributed optimization problem where the objective function is the total state mismatch between neighboring agents. In all of these problems, we find subsystems communicating with each other in order to collaboratively solve a network optimization problem.

Early distributed algorithms that solve such network optimization problems include the center-free distributed algorithms [4] and distributed asynchronous gradient-based algorithms [15]. These early algorithms suggest that if the communication between adjacent subsystems is sufficiently frequent, then the state of the network will asymptotically converge to the optimal point. Later developments in such distributed algorithms may be found in the networking community. Most of these later algorithms focus on solving the **Network Utility Maximization** (NUM) problem. The NUM problem maximizes a global separable measure of network system performance subject to linear inequality constraints that are directly related to throughput constraints. This problem originates in congestion control for Internet traffic[6] [8]. The NUM problem, however, has a general form and many problems in other areas can be recast as a NUM problem with little or no variation. As a matter of fact, all of the aforementioned problems can be reformulated as NUM problems.

Among the existing algorithms [6] [8] [19] [11] solving the NUM problem, the dual decomposition approach proposed by Low et al. [8] is the most widely used. Low et al. showed that their dual decomposition algorithm was convergent for a step-size that was inversely proportional to two important measures of network size: the maximum path length $\overline{L}$ and the maximum number of neighbors $\overline{S}$.

So as these two measures get large, the step size required to ensure convergence becomes extremely small. Step size, of course, determines the number of computations required for the algorithm's convergence. Under dual decomposition, system agents exchange information at each iteration, so that step size, $|gamma$, also determines the message passing complexity of the algorithm. Therefore if we use the "stabilizing" step size, dual decomposition algorithms will have a message passing complexity that quickly scales to unreasonable levels as we increase the network size, $\overline{L}$, or increase the neighborhoood size $\overline{S}$. In particular, it was shown in [8] that the dual-decomposition is convergent if the step size satisfies

$$0 < \gamma < \gamma^* = \frac{-2\max_{(i,x_i)} \nabla^2 U_i(x_i)}{\overline{L}\overline{S}}$$

where $\overline{L}$ is the maximum number of links any user uses, $\overline{S}$ is the maximum number of users any link has, and $U_i(x_i)$

is the utility user i receives for transmitting at rate x_i. For many networked systems this type of message passing complexity may be unacceptable. This is particularly true for systems communicating over a wireless network. In such networked systems, the energy required for communication can be significantly greater than the energy required to perform computation. As a result, it would be beneficial if we can somehow separate communication and computation in these distributed algorithms. This could reduce the message passing complexity of distributed algorithms such as dual decomposition. This section shows how event-triggering can be used to realize the separation between communication and computation in a primal algorithm solving the NUM problem.

NUM Problem: The NUM problem consists a network of N users and M links. Let $\mathscr{S} = \{1,\ldots,N\}$ denote the set of users and $\mathscr{L} = \{1,\ldots,M\}$ denote the set of links. Each user generates a flow with a specified data rate. Each flow may traverse several links (which together constitute a route) before reaching its destination. The set of links that are used by user $i \in \mathscr{S}$ will be denoted as $\mathscr{L}_i$ and the set of users that are using link $j \in \mathscr{L}$ will be denoted as $\mathscr{S}_j$. The NUM problem takes the form

$$\begin{array}{ll} \text{maximize:} & U(x) = \sum_{i\in\mathscr{S}} U_i(x_i) \\ \text{subject to:} & Ax \le c, \quad x \ge 0 \end{array} \tag{1}$$

where $x = \begin{bmatrix} x_1 & \cdots & x_N \end{bmatrix}^T$ and $x_i \in \Re$ is user i's data rate. $A \in \Re^{M\times N}$ is the routing matrix mapping users onto links and $c \in \Re$ is a vector of link capacities. The jith component, A_{ji}, is 1 if user i's flow traverses link j and is zero otherwise. The jth row of Ax represents the total data rates going through link j. This rate cannot exceed the link's capacity c_j. The cost function $U : \Re^N \to \Re$ is the sum of the user *utility* functions, $U_i : \Re \to \Re$, for $i = 1,\ldots,N$. These utility functions represent the reward, $U_i(x_i)$, (i.e. quality-of-service) that user i receives by transmitting at rate x_i.

A specific example of a NUM problem is shown in figure 1. This figure shows a linear network consisting of $M = 5$ links with $N = 3$ users. User 1 route includes links 1-4, user 2's route includes links 2-3, and user 3's route uses link 3-5. Assuming each link has a capacity limit of 1, the throughput constraint therefore becomes,

$$Ax = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \le \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = c$$

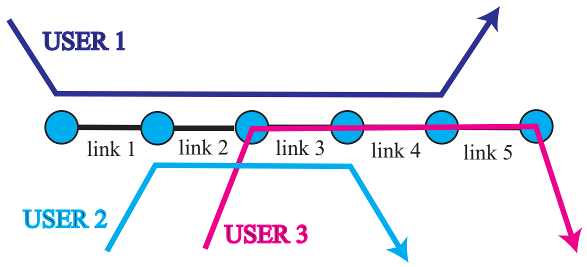

Figure 1: Example of NUM Problem

The solution to the NUM problem maximizes the summed utility seen by all users in the network as a function of the users' transmission rates. These rates must clearly be non-negative. Moreover, if $U_i(x) = \alpha_i \log(x)$ where α_i is a positive constant, then it can be shown that all the user rates in the optimal solution must be positive. In other words, the optimal solution does not result in certain users from being denied access to the network, thereby assuring that all users have "fair" access to network resources.

Augmented Lagrangian Algorithm: While early algorithms used methods based on the dual to the problem in equation (1), this section examines an primal *augmented* Lagrangian method. In particular, we introduce a slack variable $s \in \Re^M$ and replace the link constraints ($c_j - a_j^T x \ge 0$ for all $j \in \mathscr{L}$) by the following equality constraint

$$a_j^T x - c_j + s_j = 0, \quad s_j \ge 0, \quad \text{for all } j \in \mathscr{L}$$

The *augmented cost* then becomes

$$L(x,s;\lambda,w) = -\sum_{i\in\mathscr{S}} U_i(x_i) + \sum_{j\in\mathscr{L}} \lambda_j(a_j^T x - c_j + s_j) + \frac{1}{2}\sum_{j\in\mathscr{L}} \frac{1}{w_j}(a_j^T - c_j + s_j)^2$$

Here a penalty parameter w_j is associated with each link constraint and $w = [w_1,\dots,w_M]$ is the vector of penalty parameters. Suppose λ_j^* is the Lagrange multiplier associated with link j's constraint, $c_j - a_j^T x \geq 0$. Then λ_j is an estimate of λ_j^* with $\lambda = [\lambda_1,\dots,\lambda_M]$. The vector $a_j^T = \left[A_{j1},\cdots,A_{jN}\right]$ is the jth row of the routing matrix A.

$L(x,s;\lambda,w)$ is a continuous function of x and s for fixed λ and w. It was shown that

$$\min_{x\geq 0, s\geq 0} L(x,s;\lambda,w) = \min_{x\geq 0}\min_{s\geq 0} L(x,s;\lambda,w) = \min_{x\geq 0} L_p(x;\lambda,w)$$

where we define the *augmented Lagragian function* associated with the NUM problem as

$$L_p(x;\lambda,w) = -\sum_{i\in\mathscr{S}} U_i(x_i) + \sum_{j\in\mathscr{L}} \psi_j(x;\lambda,w)$$

where

$$\psi_j(x;\lambda,w) = \begin{cases} -\frac{1}{2}w_j\lambda_j^2 & \text{if } c_j - a_j^T x - w_j\lambda_j \geq 0 \\ \lambda_j(a_j^T x - c_j) + \frac{1}{2w_j}(a_j^T x - c_j)^2 & \text{otherwise} \end{cases}$$

The primal algorithm based on the augmented Lagrangian method solves the NUM problem by approximately minimizing $L_p(x;\lambda[k],\overline{w}[k])$ for sequences $\{\overline{w}[k]\}_{k=0}^{\infty}$ and $\{\lambda[k]\}_{k=0}^{\infty}$. Let $x^*[k]$ denote the approximate minimizer for $L_p(x;\lambda[k],\overline{w}[k])$. It has been shown that for appropriately chosen sequences $\{\overline{w}[k]\}_{k=0}^{\infty}$ and $\{\lambda[k]\}_{k=0}^{\infty}$, the sequence of approximate minimizers, $\{x^*k\}_{k=0}^{\infty}$ converges to the optimal point of the NUM problem. The appropriate choices for these sequences is that for all $j\in\mathscr{L}$, that $\{\overline{w}_j[k]\}_{k=0}^{\infty}$ is monotone decreasing to zero and that $\{\lambda_j[k]\}_{k=0}^{\infty}$ is a sequence of Lagrange multiplier estimates where

$$\lambda_j[k+1] = \max\left\{0, \lambda_j[k] + \frac{1}{\overline{w}_j[k]}\left(a_j^T x^*[k] - c_j\right)\right\}$$

A primal algorithm based on the augmented Lagangian method was developed that converges to the exact minimizer of the NUM problem. In many scenarios, however, it may suffice to obtain an approximate minimizer which can be obtained by consider the problem of minimizing $L_p(x;\lambda,w)$ for a fixed λ and w. In particular, if $\lambda_j = 0$ and w_j is sufficiently small, the minimizer of $L_p(x;\lambda,w)$ will be a good approximation to the solution of the original NUM problem. In this regard the **basic primal algorithm** can be stated as follows

1. **Initialization:** Select any initial user rate $x^0 > 0$. Set $\lambda_j = 0$ and select a sufficiently small $w_j > 0$ for all $j\in\mathscr{L}$.
2. **Recursive Loop:** Minimize $L_p(x;\lambda,w)$ by letting

$$x[k+1] = \max\{0, x[k] - \gamma\frac{\partial L_p}{\partial x}(x[k];\lambda,w) \tag{2}$$

for $k = 0,\dots,\infty$.

The smaller w is the more accurate our approximate solution is. The recursion shown in step 2 tries to minimize $L_p(x;\lambda,w)$ using a gradient following method in which γ is a sufficiently small step size. The computations shown above can be easily distributed among users and links.

Event-Triggered NUM Algorithm: Implementing the aforementioned primal algorithm in a distributed manner requires communication between users and links. An event-triggered implementation of the algorithm assumes that the transmission of messages between users and links is triggered by some local error signal crossing a state-dependent threshold. The main problem is to determine a threshold condition that results in message streams ensuring asymptotic convergence to the NUM problem's approximate solution.

We can search for the minimizer of the Lagrangian $L_p(x;\lambda,w)$ using a gradient following algorithm. Assuming that computation is "cheap", we realize the gradient algorithm as a continuous-time system in which

$$\begin{aligned} x_i(t) &= -\int_0^t \left(\frac{\partial L_p}{\partial x_i}(x(\tau);\lambda,w)\right)^+_{x_i(\tau)} d\tau \\ &= \int_0^t \left(\frac{\partial U_i(x_i(\tau))}{\partial x_i} - \sum_{j\in\mathscr{L}_i}\mu_j(\tau)\right)^+_{x_i(\tau)} \tau \end{aligned} \tag{3}$$

for each user $i \in \mathscr{S}$ where

$$\mu_j(t) = \max\left\{0, \lambda_j + \frac{1}{w_j}\left(\sum_{i\in\mathscr{S}_j} x_i(t) - c_j\right)\right\} \tag{4}$$

Here, given a fucntion $f : \Re^+ \to \Re$, its *positive projection* is defined as

$$(f(x))^+_x = \begin{cases} 0 & \text{if } x = 0 \text{ and } f(x) < 0 \\ f(x) & \text{otherwise} \end{cases}$$

The positive projection used above guarantees that the user rate, $x_i(t)$, is always non-negative along the trajectory.

Equation (3) is the continuous-time version of the discrete-time update shown in equation (2). Note that in equation (3), user i computes its rate based only on the information from itself and the information of μ_j from those links that are being used by user i. We can think of μ_j as the jth link's local *state*. From equation (4), link j only needs to be able to measure the total flow that goes through itself. All of this information is locally available so the update of the user rate can be done in a distributed manner.

In the above equation, this link state information is available to the user in a continuous manner. We now consider an *event-triggered* version of equation (3). Here we assume that the user accesses a *sampled* version of the link stsate. In particular, let's associate a sequence of *sampling* instants, $\{T_j^L[\ell]\}_{\ell=0}^{\infty}$ with the jth link. The time $T_j^L[\ell]$ denotes the instant when the jth link samples its link state μ_j for the ℓth time and transmits that state to users $i \in \mathscr{S}_j$. We can see that at any time $t \in \Re$, the sampled link state is a piecewise constant function of time in which

$$\hat{\mu}_j(t) = \mu_j(T_j^L[\ell])$$

for all $\ell = 0,\dots,\infty$ and any $t \in [T_j^L[\ell], T_j^L[\ell+1])$. In this regard, the "event-triggered" version of equation (3) takes the form

$$x_i(t) = \int_0^T \left(\frac{\partial U_i(x_i(\tau))}{\partial x_i} - \sum_{j\in\mathscr{L}_i}\hat{\mu}_j(\tau)\right)^+_{x_i(\tau)} d\tau$$

for all ℓ and any $t \in [T_j^L[\ell], T_j^L[\ell+1])$.

We can now try to establish conditions on the sampling times $\{T_j^L[\ell\}$ that ensure the gradient update shown in equation (2) is convergent. For notational convenience we let the time derivative of the user rate, $x_i(t)$, be denoted as $z_i(t)$. Referring to $z_i(t)$ as the *user state*, we can see that z_i satisfies the equation

$$z_i(t) = \dot{x}_i(t) = \left(\frac{\partial U_i(x_i(t))}{\partial x_i} - \sum_{j\in\mathscr{L}_i}\hat{\mu}_j(t)\right)^+_{x_i(t)}$$

for all $i \in \mathscr{S}$. Now we take $L_p(x;\lambda,w)$ as a candidate Lyapunov function. The directional derivative of L_p is

$$\begin{aligned} \dot{L}_p(x;\lambda,w) &= \sum_{i=1}^{M}\frac{\partial L_p}{\partial x_i}\frac{dx_i}{dt} = -\sum_{i=1}^{N} z_i\left(\frac{\partial U_i(x_i(t))}{\partial x_i} - \sum_{j=1}^{M}\mu_j A_{ji}\right) \\ &\leq -\sum_{i=1}^{N}\left(\frac{1}{2}z_i^2 - \frac{1}{2}\left(\sum_{j=1}^{M}(\mu_j - \hat{\mu}_j)A_{ji}\right)^2\right) \\ &\leq -\frac{1}{2}\sum_{i=1}^{N} z_i^2 + \frac{1}{2}\sum_{j=1}^{M}\overline{LS}(\mu_j - \hat{\mu}_j)^2 \end{aligned}$$

To assure that $\dot{L}_p$ is negative definite, we need to select the sampling times so that

$$\sum_{j=1}^{M} \overline{LS}(\mu_j - \hat{\mu}_j)^2 \leq \sum_{i=1}^{N} z_i^2$$

This almost looks like one of the state-dependent event-triggers we used earlier in section **??**. Unfortunately, this trigger cannot be implemented in a distributed manner. While the left-hand side is separable over the links, the right-hand side is summed over the users. So the preceding analysis does not give rise to an event trigger that can implemented locally.

To develop appropriate "local" event-triggers we consider another sequence of times $\{T_i^S[\ell]\}_{\ell=0}^{\infty}$ for each user $i \in \mathscr{S}$. The time $T_i^S[\ell]$ is the ℓth time when user i transmits its user state to all links $j \in \mathscr{L}_i$. We can therefore see that at any time $t \in \Re$, the sampled user rate is a piecewise constant function of time satisfying

$$\hat{z}_i(t) = z_i(T_i^S[\ell])$$

for all $\ell = 0, \ldots, \infty$ and any $t \in [T_i^S[\ell], T_i^S[\ell+1])$. We can now use this sampled user state in our earlier expression for $\dot{L}_p$ to show that

$$\dot{L}(x;\lambda,w) \leq -\frac{1}{2}\sum_{i=1}^{N}\left[z_i^2 - \rho\hat{z}_i^2\right] - \frac{1}{2}\sum_{j=1}^{M}\left[\rho\sum_{i\in\mathscr{S}_j}\frac{1}{\overline{L}}\hat{z}_i^2 - \overline{LS}(\mu_j - \hat{\mu}_j)^2\right]$$

for some $\rho \in (0,1)$. Now we see that $\dot{L}_p$ is negative definite as long as we can ensure that

$$\begin{array}{rcl} 0 & > & \sum_{i=1}^{N}[z_i^2 - \rho\hat{z}_i^2] \\ 0 & > & \sum_{j=1}^{M}\left[\rho\sum_{i\in\mathscr{S}_j}\frac{1}{\overline{L}}\hat{z}_i^2 - \overline{LS}(\mu_j - \hat{\mu}_j)^2\right] \end{array}$$

In this case, both inequalities are separable. The first one is separable over the users and the second one is separable over the links. We can therefore ensure these conditions are satisfied if

$$z_i^2 - \rho\hat{z}_i^2 > 0 \tag{5}$$

for each $i \in \mathscr{S}$. This condition can be enforced by requiring that the user transmit z_i at those time instants when the inequality is about to be violated. The other condition is satisfies if

$$\overline{LS}(\mu_j - \hat{\mu}_j)^2 < \rho\sum_{i\in\mathscr{S}_j}\frac{1}{\overline{L}}\hat{z}_i^2 \tag{6}$$

for each $j \in \mathscr{L}$. This condition can be enforced by requiring that the link transmit μ_j at those time instants when the inequality is about to be violated. The informal discussion given above therefore establishes that if we generate user/link transmissions using the event triggers in equation (5) and (6), then $L_p(x;\lambda,w)$ indeed becomes a Lyapunov function for this system and we can ensure that this system is convergent to a neighborhood of the optimal solution of the NUM problem.

Figure 2 shows the event-triggered optimization algorithm in graphical from. In this case we see the system "network" that was used to introduce the NUM problem in figure 1. In this case each link in the network has an associated router which monitors the total data flowing through the link ($\sum_{i\in\mathscr{S}_j} x_i(t) - c_j$). Attached to each router is a "price agent" that updates the link state μ_j and checks the event-trigger in equation (6) to determine whether or not it will transmit its local link state. In a dual manner, each user that is pumping data into the network has an associated "rate agent" that updates the user state $z_i(t)$ and checks the trigger in equation (5) to determine when to transmit to the links. We therefore see that our algorithm has both a feedback (link to user) and feedforward path (user to link) in which the information streams are both sporadic in nature.

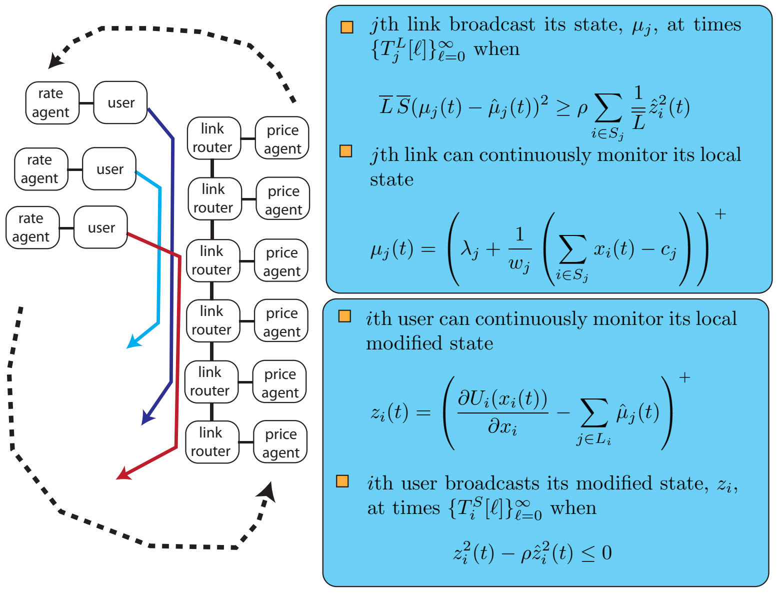
rate agent
user
rate agent
user
rate agent
user
link router
price agent
link router
price agent
link router
price agent
link router
price agent
link router
price agent
link router
price agent
jth link broadcast its state, μ_j, at times $\{T_j^L[\ell]\}_{\ell=0}^{\infty}$ when
$\overline{L}\,\overline{S}(\mu_j(t)-\hat{\mu}_j(t))^2 \geq \rho \sum_{i\in S_j} \frac{1}{\overline{L}} \hat{z}_i^2(t)$
jth link can continuously monitor its local state
$\mu_j(t) = \left(\lambda_j + \frac{1}{w_j}\left(\sum_{i\in S_j} x_i(t) - c_j\right)\right)^+$
ith user can continuously monitor its local modified state
$z_i(t) = \left(\frac{\partial U_i(x_i(t))}{\partial x_i} - \sum_{j\in L_i} \hat{\mu}_j(t)\right)^+$
ith user broadcasts its modified state, z_i, at times $\{T_i^S[\ell]\}_{\ell=0}^{\infty}$ when
$z_i^2(t) - \rho \hat{z}_i^2(t) \leq 0$

Figure 2: Diagram of the event-triggered primal algorithm

Scaling of Event-Triggered Algorithm: We compare the number of message exchanges of our event-triggered algorithms against the dual decomposition algorithm. Simulation results show that event-triggered algorithms reduce the number of message exchanges by up to two orders of magnitude when compared to dual decomposition.

In this simulation, we fix M, N, $\overline{L}$ and vary $\overline{S}$ from 7 to 26. For each $\overline{S}$, all algorithms were run 1500 times, and each time a random network which satisfies the above specification is generated. The mean m_K and standard deviation σ_K of K are computed for each $\overline{S}$. m_k works as our criteria for comparing the scalability of both algorithms. The left hand plot in figure 3 plots the iteration number K (in logarithm scale) as a function of $\overline{S}$ for all algorithms. The circles represent m_K for dual decomposition and the squares correspond to the primal algorithm.

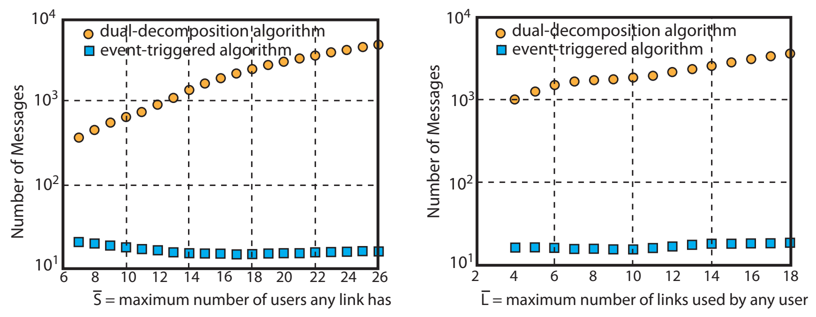

Figure 3: Iteration number K as a function of $\overline{S}$ and $\overline{L}$ for all algorithms.

For the primal algorithm, when $\overline{S}$ increases from 7 to 26, m_K does not show noticeable increase. For the primal algorithm, m_K varies between 15.1 and 21.1. For dual decomposition, m_K increases from 0.3856×10^3 to 5.0692×10^3. Our event-triggered algorithm is up to two orders of magnitude faster than the dual decomposition. We can also see that, unlike the dual decomposition algorithm, which scales superlinearly with respect to $\overline{S}$, the event-triggered algorithm is scale-free.

We also simulated the algorithms as a function of $\overline{L}$. In particular, we varied $\overline{L}$ from 4 to 18. The right hand plot in figure 3 plots K (in logarithm scale) as a function of $\overline{L}$ for all algorithms. For the primal algorithm, when $\overline{L}$ increases from 4 to 18, m_K increases slowly. In particular, m_k increases from 15.0 to 18.2. For dual decomposition, m_K increases from 0.9866×10^3 to 3.5001×10^3. Our event-triggered algorithm again is up to two orders of magnitude faster than the dual decomposition. We also see that our event-triggered algorithms is scale-free.

References

[1] W.P. Chen, J.C. Hou, L. Sha, and M. Caccamo. A distributed, energy-aware, utility-based approach for data transport in wireless sensor networks. *Proceedings of the IEEE Milcom*, 2005.

[2] W.P. Chen and L. Sha. An energy-aware data-centric generic utility based approach in wireless sensor networks. *IPSN*, pages 215–224, 2004.

[3] M. Chiang and J. Bell. Balancing supply and demand of bandwidth in wireless cellular networks: utility maximization over powers and rates. *Proc. IEEE INFOCOM*, 4:2800–2811, 2004.

[4] YC Ho, L. Servi, and R. Suri. A CLASS OF CENTER-FREE RESOURCE ALLOCATION ALGORITHMS'. In *Large Scale Systems Theory and Applications: Proceedings of the IFAC Symposium, Toulouse, France, 24-26 June 1980*, page 475. Franklin Book Co, 1981.

[5] B. Johansson, M. Rabi, and M. Johansson. A simple peer-to-peer algorithm for distributed optimization in sensor networks. In *Proceedings of the 46th IEEE Conference on Decision and Control*, pages 4705–4710, 2007.

[6] F.P. Kelly, A.K. Maulloo, and D.K.H. Tan. Rate control for communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, 49(3):237–252, 1998.

[7] BH Kim and R. Baldick. A comparison of distributed optimal power flow algorithms. *IEEE Transactions on Power Systems*, 15(2):599–604, 2000.

[8] S.H. Low and D.E. Lapsley. Optimization flow control, I: basic algorithm and convergence. *IEEE/ACM Transactions on Networking (TON)*, 7(6):861–874, 1999.

[9] R. Madan and S. Lall. Distributed algorithms for maximum lifetime routing in wireless sensor networks. In *IEEE GLOBECOM'04*, volume 2, 2004.

[10] R. Olfati-Saber, JA Fax, and RM Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

[11] DP Palomar and M. Chiang. Alternative Distributed Algorithms for Network Utility Maximization: Framework and Applications. *IEEE Transactions on Automatic Control*, 52(12):2254–2269, 2007.

[12] Y. Qiu and P. Marbach. Bandwidth allocation in ad hoc networks: A price-based approach. In *Proceedings of IEEE INFOCOM 2003*, volume 2, pages 797–807, 2003.

[13] M. Rabbat and R. Nowak. Distributed optimization in sensor networks. *Proceedings of the third international symposium on Information processing in sensor networks*, pages 20–27, 2004.

[14] A. Speranzon, C. Fischione, and K.H. Johansson. Distributed and Collaborative Estimation over Wireless Sensor Networks. *Proceedings of the IEEE Conference on Decision and Control*, pages 1025–1030, 2006.

[15] J. Tsitsiklis, D. Bertsekas, and M. Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE Transactions on Automatic Control*, 31(9):803–812, 1986.

[16] P. Wan and M. Lemmon. Distributed Flow Control using Embedded Sensor-Actuator Networks for the Reduction of Combined Sewer Overflow (CSO) Events. In *Proceedings of the 46th IEEE Conference on Decision and Control*, pages 1529–1534, 2007.

[17] P. Wan and M.D. Lemmon. Distributed network utility maximization using event-triggered augmented lagrangian methods, June 2009.

[18] P. Wan and M.D. Lemmon. Event-triggered distributed optimization in sensor networks. In *Information Processing in Sensor Networks (IPSN)*, San Francisco, California, USA, April 2009.

[19] JT Wen and M. Arcak. A unifying passivity framework for network flow control. *IEEE Transactions on Automatic Control*, 49(2):162–174, 2004.

[20] L. Xiao, M. Johansson, and SP Boyd. Simultaneous routing and resource allocation via dual decomposition. *IEEE Transactions on Communications*, 52(7):1136–1144, 2004.

[21] Y. Xue, B. Li, and K. Nahrstedt. Optimal resource allocation in wireless ad hoc networks: a price-based approach. *IEEE Transactions on Mobile Computing*, 5(4):347–364, 2006.