

Distributed Algorithm Design for Network Optimization Problems with Coupled Objectives

Jianwei Huang

Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong

E-mail: jwhuang@ie.cuhk.edu.hk

Abstract—This paper establishes a framework for designing fast, robust, and distributed algorithms for solving network utility maximization problems with coupled objective functions. We use two case studies in wireless communications to illustrate the key ideas: reverse-engineering the algorithm based on the KKT conditions of the optimization problem, and proving the properties of the algorithms using monotone mapping, contraction mapping, and supermodular game theory.

I. INTRODUCTION

In order to optimize the performance of communication and networking systems, we often model it mathematically as a Network Utility Maximization (NUM) problem [1]. In the NUM formulation, we associate each network entity (user) with a utility function representing its performance/satisfaction/happiness, and thus maximizing the total network utility is equivalent to maximizing the network performance. Due to the distributed and heterogeneous nature of the modern network, it is often challenging to design distributed algorithms that can achieve the global optimal NUM solution.

The difficulty in distributed algorithm design often lies in the coupling nature of the NUM problem. There exist two kinds of coupling: coupled constraints (e.g., limited total network resource) and coupled objective functions (often due to interferences or collisions among wireless nodes). The coupled constraints can be decomposed using the dual or primal decompositions that have been widely studied (see [2], [1] and numerous references therein). The coupled objective functions, however, are more difficult to deal with and thus less studied. One recent innovative approach of dealing with coupled objective functions is to use the “consistency pricing” [3], which is mainly suitable for strictly convex NUM formulations, involves significant amount of message passing among users, and requires updating the variables using small stepsizes which often lead to slow convergence.

This paper establishes a new framework of designing distributed algorithm for NUM with coupled objective functions. The key idea is to “reverse-engineer” the algorithm based on the KKT condition of the NUM problem, which involves “localizing the global objective function” and designing suitable message passing mechanism with proper physical meanings. A key feature of the corresponding algorithm is that it does involve use any small stepsizes, thus typically has much faster and more robust convergence compared with the consistence pricing approach. The algorithm also works for certain cases where the strictly convexity of the NUM problem can not be directly proved using the standard optimization approach.

After introducing the general problem formulation and the distributed algorithm in Section II, we describe the direct and indirect approaches of analyzing the algorithm in Sections III and IV. In Sections V and VI, we introduce two case studies to illustrate the framework based on the recent results in [4] and [5]. We finally conclude in Section VII.

II. NUM PROBLEM AND THE DISTRIBUTED ALGORITHM

A. Problem Formulation

Consider a system consisting of a set $\mathcal{K} = \{1, \dots, K\}$ of users. Each user $k \in \mathcal{K}$ has a *coupled* utility function $U_k(x_k, \mathbf{x}_{-k})$ that depends on both his own local decision variable x_k and other users’ decision variables $\mathbf{x}_{-k} = (x_1, \dots, x_{k-1}, x_{k+1}, \dots, x_K)$. User k can choose x_k from a feasible set $\mathcal{X}_k = [X_k^{\min}, X_k^{\max}]$. Denote $\mathbf{x} = (x_k)_{k=1}^K$ and $\mathcal{X} = \cup_{k \in \mathcal{K}} \mathcal{X}_k$. We are interested in solving the following NUM problem:

$$\max_{\mathbf{x} \in \mathcal{X}} \sum_{k \in \mathcal{K}} U_k(x_k, \mathbf{x}_{-k}). \quad (1)$$

Notice that we intentionally make the constraints (\mathcal{X} ’s) simple here in order to focus the study on the coupled objective function (U_k ’s). The solution framework can be easily combined with various dual and primal decomposition to deal with more complicated and coupled constraint sets (e.g., [4]).

We make the following widely used assumption for the utility functions.

Assumption 1: For each k , $U_k(x_k, \mathbf{x}_{-k})$ is increasing and strictly concave in x_k but not necessarily concave in \mathbf{x} .

Assumption 1 means that Problem (1) is not necessarily a strictly concave maximization problem in variables \mathbf{x} , and thus might have several local/global optimal solutions. Solving such a problem is difficult in general even through centralized computation. We will later show various approaches of solving such a problem in a distributed fashion under certain technical conditions.

B. KKT Conditions

The starting point of our algorithm design is the Karush–Kuhn–Tucker (KKT) conditions of Problem (1), which are the necessary conditions for a global optimal solution.

Definition 1 (KKT Conditions of Problem (1)): Any global optimal solution \mathbf{x}^* of Problem (1) must satisfy the following

conditions for each $k \in \mathcal{K}$,

$$\frac{\partial U_k(x_k^*, \mathbf{x}_{-k}^*)}{\partial x_k^*} + \sum_{j \neq k} \frac{\partial U_j(x_k^*, \mathbf{x}_{-k}^*)}{\partial x_k^*} = \lambda_k^* - \mu_k^*, \quad (2)$$

$$\lambda_k^*(x_k^* - X_k^{\max}) = 0, \quad \mu_k^*(X_k^{\min} - x_k^*) = 0, \quad (3)$$

$$\lambda_k^*, \mu_k^* \geq 0. \quad (4)$$

Definition 2 (KKT Set): The KKT set of Problem (1), Q^{KKT} , contains all solutions that satisfy conditions (2)-(4) for all k .

Our task is to design an distributed algorithm that is guaranteed to converge to the KKT set. If the KKT set is a singleton set, then the corresponding element must be the unique global optimal solution and our algorithm converges to such a solution.

C. Reverse Engineering Local Optimization Objective Function Based on the KKT Conditions

We design distributed algorithm by letting each user solve a local optimization problem that is defined based on *local observation* and *limited message passing among users*. We will first reverse engineer the ‘‘local objective function’’ for each user, such that the KKT conditions can be satisfied if all users maximize their local objective functions properly.

In particular, each user k can chooses a local objective function $Y_k(x_k, \mathbf{x}_{-k})$ that is increasing and strictly concave in x_k and satisfies

$$\frac{\partial Y_k(x_k, \mathbf{x}_{-k})}{\partial x_k} = \frac{\partial U_k(x_k, \mathbf{x}_{-k})}{\partial x_k} + \sum_{j \neq k} \frac{\partial U_j(x_k, \mathbf{x}_{-k})}{\partial x_k}. \quad (5)$$

If user k solves the following local optimization problem for an optimal choice of x_k^* ,

$$\max_{x_k \in \mathcal{X}_k} Y_k(x_k, \mathbf{x}_{-k}^*). \quad (6)$$

the obtained optimal solution x_k^* satisfies (2)-(4).

One seemingly obvious choice of $Y_k(\cdot)$ is the total network utility,

$$Y_k(x_k, \mathbf{x}_{-k}) = \sum_{j=1}^K U_j(x_k, \mathbf{x}_{-k}). \quad (7)$$

This choice, however, is often not practical. This is because it is often too strong to assume that a user knows the exact forms of U_j 's and values of x_j 's for all $j \neq k$. If such information is available, then each user essentially knows the complete network information can simply compute the optimal solution like a centralized controller.

A more practical approach is to construct a local function that only depends on user k 's local observation of the network and some limited information passing among users. We assume that each user k can announce a locally computable message m_k as a function of \mathbf{x} . Define $\mathbf{m}_{-k} = (m_1, \dots, m_{k-1}, m_{k+1}, \dots, m_K)$ and $\mathbf{m} = (m_k)_{k=1}^K$. We will construct a new local objective function $Z_k(x_k, \mathbf{x}_{-k}, \mathbf{m}_{-k})$ for user k such that

$$\frac{\partial Z_k(x_k, \mathbf{x}_{-k}, \mathbf{m}_{-k})}{\partial x_k} = \frac{\partial U_k(x_k, \mathbf{x}_{-k})}{\partial x_k} + \sum_{j \neq k} \frac{\partial U_j(x_k, \mathbf{x}_{-k})}{\partial x_k}. \quad (8)$$

The key requirement is that user k can calculate function $Z_k(\cdot)$ based on its local measurement (which can depend on \mathbf{x}_{-k}) and the messages announced by other users (\mathbf{m}_{-k}).

User k 's local optimization problem is to calculate

$$x_k = q_k(\mathbf{x}_{-k}, \mathbf{m}_{-k}) = \arg \max_{\bar{x}_k \in \mathcal{X}_k} Z_k(\bar{x}_k, \mathbf{x}_{-k}, \mathbf{m}_{-k}). \quad (9)$$

For message m_k , user k need to update it as

$$m_k = f_k(\mathbf{x}). \quad (10)$$

In Sections V and VI, we will illustrate how functions $q_k(\cdot)$ and $f_k(\cdot)$ are derived based on specific problem structures through two case studies in wireless ad hoc networks.

D. Asynchronous and Distributed Algorithm

We are now ready to describe the asynchronous distributed algorithm as in Algorithm 1. For each user k , let $T_{k,x}$ and $T_{k,m}$ be two unbounded sets of positive time instances at which user k updates its local decision variable x_k and message m_k , respectively.

Algorithm 1 Distributed Algorithm to Solve Problem (1)

- 1: Let time $t = 0$.
 - 2: **for all** user k **do**
 - 3: Randomly initialize $x_k(0)$ and $m_k(0)$.
 - 4: **end for**
 - 5: **repeat**
 - 6: $t = t + 1$.
 - 7: **for all** user k **do**
 - 8: **if** $t \in T_{k,x}$ **then**
 - 9: $x_k(t+1) = q_k(\mathbf{x}_{-k}(t), \mathbf{m}_{-k}(t))$.
 - 10: **end if**
 - 11: **if** $t \in T_{k,m}$ **then**
 - 12: Announce $m_k(t+1) = f_k(\mathbf{x}(t))$.
 - 13: **end if**
 - 14: **end for**
 - 15: **until** $(\mathbf{x}(t), \mathbf{m}(t))$ converge
-

A key feature of Algorithm 1 is that there are *no small step-sizes* involved, which is unlike the consistency price approach in [3]. The stepsize-free design often leads to much faster convergence, but makes the analysis (e.g., proving convergence) more difficult. Also, users update the decision variables and messages in a *completely asynchronous* fashion, which makes the algorithm robust to practical implementations involving packet delays, packet loss, lack of clock synchronization, etc.

Now let us look at the connection between Algorithm 1 and the Problem (1).

Definition 3 (Fixed Point Set of Algorithm 1): The set of fixed points of Algorithm 1 is defined as

$$Q^{ALG} \equiv \{(\mathbf{x}, \mathbf{m}) \mid (\mathbf{x}, \mathbf{m}) = (\mathbf{q}(\mathbf{x}, \mathbf{m}), \mathbf{f}(\mathbf{x}))\}, \quad (11)$$

where $\mathbf{q}(\mathbf{x}, \mathbf{m}) = (q_k(\mathbf{x}_{-k}, \mathbf{m}_{-k}))_{k=1}^K$ and $\mathbf{f}(\mathbf{x}) = (f_k(\mathbf{x}))_{k=1}^K$.

We can show that the following connection between the KKT point set Q^{KKT} of Problem (1) and the fixed point set Q^{ALG} of Algorithm 1.

Lemma 1: A decision vector $\mathbf{x}^* \in \mathcal{Q}^{KKT}$ if and only if $(\mathbf{x}^*, \mathbf{m}^*) \in \mathcal{Q}^{ALG}$ for some choice of \mathbf{m}^* .

Corollary 1: If \mathcal{Q}^{ALG} is a singleton set containing the element $(\mathbf{x}^*, \mathbf{m}^*)$, then \mathbf{x}^* must be the unique global optimal solution of Problem (1).

Corollary 2: If Problem (1) has a unique global optimal solution \mathbf{x}^* , then there must exist some \mathbf{m}^* such that $(\mathbf{x}^*, \mathbf{m}^*)$ is the unique element in \mathcal{Q}^{ALG} .

What remains to be shown is the convergence of Algorithm 1 and the uniqueness of the fixed point. There are two main approaches of analyzing Algorithm 1 as explained next.

III. DIRECT ANALYSIS: A MAPPING BASED APPROACH

Assume that we can eliminate the intermedium variables of \mathbf{m} and directly write the update of \mathbf{x} in Algorithm 1 as

$$x_k(t+1) = g_k((x_j(\tau_j^k(t)))_{j=1}^K), \quad \forall k \in \mathcal{K}. \quad (12)$$

where $\tau_j^k(t)$ denotes the time stamp of the most recent version of x_j in the local memory of user k at the beginning of time slot t . The analysis of the mapping $\mathbf{g}(\cdot)$ as in

$$\mathbf{x}(t+1) = \mathbf{g}(\mathbf{x}(\boldsymbol{\tau}(t))) = \left(g_k((x_j(\tau_j^k(t)))_{j=1}^K) \right)_{k=1}^K$$

enables us to prove the convergence of Algorithm 1.

A. Approach 1: Monotone Mapping

Definition 4 (Monotone Mapping): A mapping $\mathbf{g}(\cdot)$ is *monotone increasing* if

$$\tilde{\mathbf{x}} \preceq \hat{\mathbf{x}} \quad \Rightarrow \quad \mathbf{g}(\tilde{\mathbf{x}}) \preceq \mathbf{g}(\hat{\mathbf{x}}), \quad \forall \tilde{\mathbf{x}}, \hat{\mathbf{x}} \in \mathcal{X},$$

and is *monotone decreasing* if

$$\tilde{\mathbf{x}} \preceq \hat{\mathbf{x}} \quad \Rightarrow \quad \mathbf{g}(\tilde{\mathbf{x}}) \succeq \mathbf{g}(\hat{\mathbf{x}}), \quad \forall \tilde{\mathbf{x}}, \hat{\mathbf{x}} \in \mathcal{X},$$

where the inequalities are interpreted as coordinate-wise.

Theorem 1 ([5]): Assume that $\mathbf{g}(\mathbf{x})$ has a unique fixed point \mathbf{x}^* and is a monotone mapping (increasing or decreasing). Starting from any initial point $\mathbf{x} \in \mathcal{X}$, Algorithm 1 converges globally to \mathbf{x}^* .

One weakness of the monotone mapping approach is that we need to prove the existence and uniqueness of the fixed point separately. This motivates us to consider the second approach.

B. Approach 2: Contraction Mapping

Definition 5 (Contraction Mapping): A mapping function $\mathbf{g}(\cdot)$ is a *contraction mapping* if there exists constant $\zeta \in (0, 1)$ such that

$$\|\mathbf{g}(\tilde{\mathbf{x}}) - \mathbf{g}(\hat{\mathbf{x}})\| \leq \zeta \|\tilde{\mathbf{x}} - \hat{\mathbf{x}}\|, \quad \forall \tilde{\mathbf{x}}, \hat{\mathbf{x}} \in \mathcal{X}, \quad (13)$$

where $\|\cdot\|$ is some vector norm.

Theorem 2 ([6]): A contraction mappings has a unique fixed point.

Usually the conditions for proving contraction mapping is more stringent compared with the monotone mapping.

IV. INDIRECT ANALYSIS: A FICTITIOUS GAME APPROACH

A different approach is to map Algorithm 1 as the best response updates of a fictitious game, and then prove its convergence by using the special structure of the game.

To start with, we need to define the key elements of the game: players, strategies, and payoffs. It is natural to consider users in the network as players in the game. This may not always work since the users in the network are cooperatively solving Problem (1) while the players in the game are selfish.

What we can do is to split a user in the network into two fictitious users in the game: player k_x controlling the decision variable x_k and player k_m controlling the message m_k .

- The payoff function of player k_x is $Z_k(x_k, \mathbf{x}_{-k}, \mathbf{m}_{-k})$ that satisfies (8), and Line 9 in Algorithm 1 can be viewed as the best response update of player k_x .
- The payoff function of player k_m should be in the form of $F_k(\mathbf{x})$ such that

$$f_k(\mathbf{x}) = \arg \max_{m_k} F_k(m_k, \mathbf{x}), \quad (14)$$

i.e., Line 12 of Algorithm 1 is the best response update of player k_m .

Next we need to show that the fictitious game has certain nice structures such that its best response updates converge. One possibility is to show that the game is supermodular as we explain next. Other special game structures that are useful include potential game (e.g., [7], [8]) and congestion game (e.g., [9], [10], [11]), which are not discussed here due to space limitations.

A. Supermodular Game Theory

We first introduce some definitions. A real m -dimensional set \mathcal{V} is a *sublattice* of \mathbb{R}^m if for any two elements $a, b \in \mathcal{V}$, the component-wise minimum, $a \wedge b$, and the component-wise maximum, $a \vee b$, are also in \mathcal{V} . In particular, a compact sublattice has a (component-wise) smallest and largest element. A twice differentiable function f has *increasing differences* in variables (x, t) if $\partial^2 f / \partial x \partial t \geq 0$ for any feasible x and t . A function f is *supermodular* in \mathbf{x} if it has increasing differences in (x_k, x_j) for all $k \neq j$.

A game $G = [\mathcal{K}, \{\mathcal{P}_k\}, \{s_k\}]$ is *supermodular* if for each player $k \in \mathcal{K}$, (a) the strategy space \mathcal{P}_k is a nonempty and compact sublattice, and (b) the payoff function s_k is continuous in all players' strategies, is supermodular in player k 's own strategy, and has increasing differences between any component of player k 's strategy and any component of any other player's strategy. The following theorem summarizes several convergence properties of the supermodular game.

Theorem 3 ([4]): In a supermodular game $G = [\mathcal{K}, \{\mathcal{P}_k\}, \{s_k\}]$,

- The set of Nash equilibria is a nonempty and contains a component-wise smallest and largest Nash equilibrium.
- If the players' best responses are single-valued, and each user uses best response updates starting from the smallest (largest) element of its strategy space, then the strategies monotonically and asynchronously converge to the smallest (largest) Nash equilibrium.

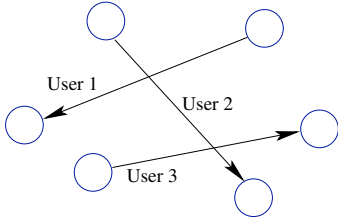


Fig. 1. A single-hop ad-hoc network with $N = 3$ users.

- If each player starts from any feasible strategy and uses best response updates, the strategies will eventually lie in the set bounded component-wise by the smallest and largest Nash equilibrium. If the Nash equilibrium is unique, the best response updates globally converge to that Nash equilibrium from any initial strategies.

V. CASE STUDY I: OPTIMAL RANDOM ACCESS

Consider a single-hop wireless ad-hoc network with a set $\mathcal{K} = \{1, \dots, K\}$ of links. We use the terms “link” and “user” interchangeably. A sample network with three users is shown in Fig. 1. We assume that each user’s receiver node can hear every other user’s transmissions¹. Thus, every user interferes with every other user in the network. Time is divided into equal-length slots. At each slot, user k transmits with probability p_k . A transmission is successful only if it is the only transmission in the current slot. Let r_k denote the average data rate for user k . We have [12]:

$$r_k(\mathbf{p}) = \gamma_k p_k \prod_{j \in \mathcal{K} \setminus \{k\}} (1 - p_j), \quad \forall k \in \mathcal{K}, \quad (15)$$

where $\mathbf{p} = (p_k)_{k=1}^K$ is the vector of users’ transmission probabilities. γ_k denotes the fixed peak data rate for user k , which depends on the channel gains that are assumed to be fixed during the time of interest. We assume that each user $k \in \mathcal{K}$ limits its transmission probability $p_k \in \mathcal{P}_k = [P_k^{\min}, P_k^{\max}] \in (0, 1)$.

We want to solve the following NUM problem:

$$\max_{\mathbf{p} \in \mathcal{P}} \sum_{k \in \mathcal{K}} U_k(r_k(\mathbf{p})), \quad (16)$$

where $\mathcal{P} = \{\mathbf{p} : p_k \in \mathcal{P}_k, \forall k \in \mathcal{K}\}$, and the utility functions are α -fair with $\alpha > 0$ [13]:

$$U_k(x) = \begin{cases} (1 - \alpha)^{-1} x^{1-\alpha}, & \text{if } \alpha \neq 1, \\ \log x, & \text{otherwise.} \end{cases} \quad \forall k \in \mathcal{K}. \quad (17)$$

Using (17), a wide range of well-known efficient and fair allocations can be modeled. In particular, Problem (16) reduces to system throughput maximization with $\alpha \rightarrow 0$, to proportional fair allocation with $\alpha = 1$, to harmonic mean fair allocation with $\alpha = 2$, and to max-min fairness with $\alpha \rightarrow \infty$. We note that although the objective function in (16) is concave in rates $\mathbf{r} = (r_k)_{k=1}^K$, it is not concave in transmission probabilities \mathbf{p} .

¹For more general discussions, see [5].

A. Reverse Engineering of Local Objective Functions

We start to look at the problem where user k wants to maximize the total network utility assuming \mathbf{p}_{-k} is fixed, i.e.,

$$\max_{p_k \in \mathcal{P}_k} \sum_{j \in \mathcal{K}} U_j(r_j(p_k, \mathbf{p}_{-k})). \quad (18)$$

Problem (18) turns out to be a convex optimization problem in p_k and its unique optimal solution is

$$q_k(\mathbf{p}_{-k}) = \left[1 / \left(1 + \sqrt[\alpha]{v_k(\mathbf{p}_{-k})} \right) \right]_{P_k^{\min}}^{P_k^{\max}}, \quad (19)$$

where $[x]_b^a = \max[\min[x, a], b]$,

$$v_k(\mathbf{p}_{-k}) = \gamma_k^{\alpha-1} \sum_{j \in \mathcal{K} \setminus \{k\}} (1/\gamma_j)^{\alpha-1} (1/p_j - 1)^{\alpha-1}. \quad (20)$$

If we choose

$$Y_k(p_k, \mathbf{p}_{-k}) = U_k(p_k) + v_k(\mathbf{p}_{-k}) U_k(1 - p_k),$$

then

$$q_k(\mathbf{p}_{-k}) = \arg \max_{p_k \in \mathcal{P}_k} Y_k(p_k, \mathbf{p}_{-k}). \quad (21)$$

If we further define

$$m_j \equiv f_k(p_j) = (1/\gamma_j)^{\alpha-1} (1/p_j - 1)^{\alpha-1}, \quad \forall j \in \mathcal{K}, \quad (22)$$

then we can write

$$\tilde{v}_k(\mathbf{m}_{-k}) = v_k(\mathbf{p}_{-k}) = \gamma_k^{\alpha-1} \sum_{j \in \mathcal{K} \setminus \{k\}} m_j.$$

Thus, user k can obtain the value of (19) if message m_j is announced by all users $j \neq k$. This enables us to write user k ’s local objective function as

$$Z_k(p_k, \mathbf{m}_{-k}) = U_k(p_k) + \tilde{v}_k(\mathbf{m}_{-k}) U_k(1 - p_k). \quad (23)$$

B. Distributed Random Access (DRA) Algorithm

The distributed algorithm can be obtained from Algorithm 1 by replacing variables \mathbf{x} with \mathbf{p} , decision variable update in Line 9 with (21), and message update in Line 12 with (22), respectively. We call the new algorithm *Distributed Random Access (DRA) Algorithm*.

Let us consider the mapping

$$\mathbf{p}(t+1) = \mathbf{g}(\mathbf{p}(\tau(t))) = \left(g_k((p_j(\tau_j^k(t)))_{j=1}^K) \right)_{k=1}^K,$$

where $\tau_j^k(t)$ denotes the time stamp of the most recent version of m_j in the local memory of user k at the beginning of time slot t . We can show that

Proposition 1: Mapping $\mathbf{g}(\cdot)$ is monotone increasing if $\alpha \geq 1$, and is monotone decreasing if $\alpha \leq 1$.

In order to use Theorem 1 in Section III-A to show the convergence of the DRA algorithm, we need to first prove that it has a unique fixed point. For $\alpha \geq 1$, it has been shown that Problem (16) is equivalent to a strictly convex optimization problem through logarithmic variable transformation [14]. This means that the KKT set of Problem (16) is a singleton set, and by Corollary 2 we know that the DRA algorithm has a unique fixed point. This transformation, however, does not work if $\alpha < 1$. In that case, there is no known method of showing that Problem (16) has a unique global optimal

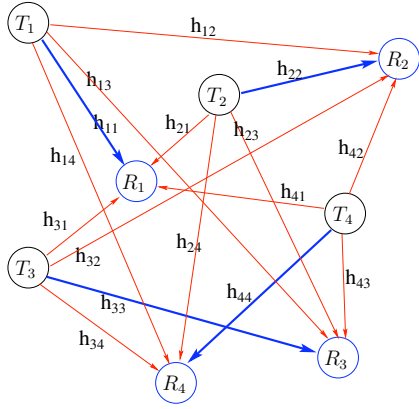


Fig. 2. An example wireless network with four users (pairs of nodes) (T_k and R_k denote the transmitter and receiver of “user” k , respectively).

solution in general. We will use the contraction mapping method in Section III-B to tackle this problem.

Theorem 4: For a given choice of $\alpha \in (0, 1)$, there exists a positive integer \hat{K} such that $\mathbf{g}(\cdot)$ is a contraction mapping if the number of users $K > \hat{K}$.

Theorems 1 and 4 together show that Algorithm 1 globally and asynchronously converges to the unique global optimal solution of Problem (16) when either $0 < \alpha < 1$ and the number of user is large enough, or $\alpha \geq 1$. Thus, the DRA algorithm works properly under delayed or even occasionally lost messages. Interestingly, this robust behavior is accompanied with fast convergence speed as shown in [5].

VI. CASE STUDY II: DISTRIBUTED POWER CONTROL

A. Problem Formulation

Consider a snap-shot of an ad hoc network with a set $\mathcal{K} = \{1, \dots, K\}$ of distinct node pairs. As shown in Fig. 2, each pair consists of one dedicated transmitter and one dedicated receiver. We will again use the terms “pair” and “user” interchangeably in the following. Over the time-period of interest, the channel gains of each pair are fixed. The channel gain between user k ’s transmitter and user j ’s receiver is denoted by h_{kj} . Note that in general $h_{kj} \neq h_{jk}$, since the latter represents the gain between user j ’s transmitter and user k ’s receiver.

Each user k ’s quality of service is characterized by a utility function $U_k(\gamma_k)$, which is an increasing and strictly concave function of the received signal-to-interference plus noise ratio (SINR),

$$\gamma_k(\mathbf{p}) = \frac{p_k h_{kk}}{n_0 + \sum_{j \neq k} p_j h_{jk}}, \quad (24)$$

where $\mathbf{p} = (p_k)_{k=1}^K$ is a vector of the users’ transmission powers and n_0 is the background noise power. The users’ utility functions are coupled due to mutual interference. An example utility function is a *logarithmic utility function* $u_k(\gamma_k) = \theta_k \log(\gamma_k)$, where θ_k is a user dependent priority parameter.²

²In the high SINR regime, logarithmic utility approximates the Shannon capacity $\log(1 + \gamma_k)$ weighted by θ_k . For low SINR, a user’s rate is approximately linear in SINR, and so this utility is proportional to the logarithm of the rate.

The problem we consider is to specify \mathbf{p} to maximize the utility summed over all users, where each user k must also satisfy a transmission power constraint, $p_k \in \mathcal{P}_k = [P_k^{\min}, P_k^{\max}]$, i.e.,

$$\max_{\mathbf{p} \in \mathcal{P}} \sum_{k \in \mathcal{K}} U_k(\gamma_k(\mathbf{p})). \quad (25)$$

We note that although $U_k(\cdot)$ is concave in the SINR, the objective in Problem (25) may not be concave in power \mathbf{p} .

B. Reverse Engineering of Local Objective Functions

The KKT condition (2) for user k can be written as

$$\frac{\partial U_k(\gamma_k(\mathbf{p}^*))}{\partial p_k} - \sum_{j \neq k} m_j(p_j^*, \mathbf{p}_{-j}^*) h_{ij} = \lambda_k^* - \mu_k^*, \quad (26)$$

where

$$\begin{aligned} m_j(\mathbf{p}) &\equiv f_k(\mathbf{p}) = -\frac{\partial U_j(\gamma_j(p_j, \mathbf{p}_{-j}))}{\partial I_j(\mathbf{p}_{-j})} \\ &= \frac{\partial U_j(\gamma_j(\mathbf{p}))}{\partial \gamma_j(\mathbf{p})} \frac{(\gamma_j(\mathbf{p}))^2}{p_j h_{jj}}, \end{aligned} \quad (27)$$

and $I_j(\mathbf{p}_{-j}) = \sum_{k \neq j} p_k h_{kj}$ is the total interference received by user j and is locally measurable. Here, $m_j(p_j, \mathbf{p}_{-j})$ is always nonnegative and represents user j ’s marginal increase in utility per unit decrease in total interference.

Viewing m_j as a *price* charged to other users for generating interference to user k (or Pigovian tax in microeconomics [15]), condition (26) implies that we can write the local objective of user k as

$$Z_k(p_k; \mathbf{p}_{-k}, \mathbf{m}_{-k}) = U_k(\gamma_k(p_k, \mathbf{p}_{-k})) - p_k \sum_{j \neq k} m_j h_{kj}. \quad (28)$$

Notice that the SINR $\gamma_k(p_k, \mathbf{p}_{-k})$ is locally measured by user k (although it depends on the value of \mathbf{p}_{-k}). The value of p_k that maximizes $Z_k(\cdot)$ is

$$\begin{aligned} q_k(\mathbf{p}_{-k}, \mathbf{m}_{-k}) &= \left[\frac{p_k}{\gamma_k(\mathbf{p})} d_k \left(\frac{p_k}{\gamma_k(\mathbf{p})} \left(\sum_{j \neq k} m_j h_{kj} \right) \right) \right]_{P_k^{\min}}^{P_k^{\max}}, \end{aligned} \quad (29)$$

where $\frac{p_k}{\gamma_k(\mathbf{p})}$ is independent of p_k , and

$$d_k(x) = \begin{cases} \infty, & 0 \leq x \leq U_k'(\infty), \\ (U_k')^{-1}(x), & U_k'(\infty) < x < U_k'(0), \\ 0, & U_k'(0) \leq x, \end{cases}$$

with $U_k'(\cdot)$ as the first order derivative of $U_k(\cdot)$.

C. Asynchronous Distributed Pricing (ADP) Algorithm

The distributed algorithm can be obtained from Algorithm 1 by replacing variables \mathbf{x} with \mathbf{p} , variable update in Line 9 with (29), and message update in Line 12 with (27), respectively. We call the new algorithm Asynchronous Distributed Pricing (ADP) algorithm.

To implement the updates, each user k only needs to know its own utility U_k , the channel gain h_{kk} , the current SINR

γ_k , the ‘‘adjacent’’ channel gains h_{kj} for any $j \neq k$, and the price profile \mathbf{m} . By assumption each user knows its own utility. The SINR γ_k and channel gain h_{kk} can be measured at the receiver and fed back to the transmitter. Measuring the adjacent channel gains h_{kj} can be accomplished by having each receiver periodically broadcast a beacon; assuming reciprocity, the transmitters can then measure these channel gains. The adjacent channel gains account for only $1/K$ of the total channel gains in the network; each user does not need to know the other gains. The price information could also be periodically broadcast through this beacon. Since each user announces only a single price, the number of prices scales linearly with the size of the network.

We next characterize the convergence of the ADP algorithm by viewing it in a fictitious game theoretic context. We consider the following game

$$G_{FPP} = [\mathcal{FW} \cup \mathcal{FC}, \{\mathcal{P}_k^{\mathcal{FW}}, \mathcal{P}_k^{\mathcal{FC}}\}, \{s_k^{\mathcal{FW}}, s_k^{\mathcal{FC}}\}],$$

where the players are from the union of the sets \mathcal{FW} and \mathcal{FC} which are both same as \mathcal{K} . \mathcal{FW} is a *fictitious power player set*; each player $k \in \mathcal{FW}$ chooses a power p_k from the strategy set $\mathcal{P}_k^{\mathcal{FW}} = \mathcal{P}_k$ and receives a payoff $Z_k(p_k; \mathbf{p}_{-k}, \mathbf{m}_{-k})$ as in (28). \mathcal{FC} is a *fictitious price player set*; each player $k \in \mathcal{FC}$ chooses a price m_k from the strategy set $\mathcal{P}_k^{\mathcal{FC}} = [0, \infty]$ and receives a payoff

$$F_k^{\mathcal{FC}}(m_k; \mathbf{p}) = -(m_k - f_k(\mathbf{p}))^2, \quad (30)$$

where $f_k(\mathbf{p})$ is given in (27).

We can show that certain instances of G_{FPP} are equivalent to supermodular games through an appropriate strategy space transformation ([4]), and so Theorem 3 applies. We would like to emphasize that even in the case where the KKT set of Problem (25) is not a singleton set, Theorem 3 still enables us to prove conditional convergence (e.g., to the component-wise smallest or largest Nash equilibrium) of the ADP algorithm as long as users choose the initial transmission power appropriately. Finally, we observe that the stepsize-free ADP algorithm can achieve a convergence speed 10 time faster than a gradient-based method using small stepsizes [16].

VII. CONCLUSIONS

In this paper, we establish a framework for designing fast, robust, and distributed algorithm for solving network utility maximization (NUM) problems with coupled objective functions. The key idea is to reverse-engineer the proper local optimization objective functions and message passing mechanisms through examining the KKT conditions of the NUM problem. Since the corresponding algorithm does not involve stepsizes as in the widely used classical optimization-based approaches, we need to rely on various techniques such as monotone mapping, contraction mapping, and supermodular game theory to prove convergence properties. Two case studies are given to illustrate the application of the framework in designing new optimal random access algorithm and optimal power control algorithm in wireless ad hoc networks.

ACKNOWLEDGMENT

This work is supported by the Competitive Earmarked Research Grants (Project Number 412308 and 412509) established under the University Grant Committee of the Hong Kong Special Administrative Region, China, and the National Key Technology R&D Program (Project Number 2007BAH17B04) established by the Ministry of Science and Technology of the People’s Republic of China. The author also wants to thank for great collaborations and helpful discussions with Randall Berry, Mung Chiang, Michael Honig, Vincent Lau, Amir-Hamed Mohsenian-Rad, Bixio Rimoldi, and Vincent Wong.

REFERENCES

- [1] M. Chiang, S. Low, A. Calderbank, and J. Doyle, ‘‘Layering as optimization decomposition: A mathematical theory of network architectures,’’ *Proceedings of the IEEE*, vol. 95, no. 1, pp. 255–312, 2007.
- [2] D. Palomar and M. Chiang, ‘‘A tutorial on decomposition methods for network utility maximization,’’ *IEEE Journal on Selected Areas of Communications*, vol. 24, no. 8, pp. 1439 – 1451, 2006.
- [3] C. W. Tan, D. P. Palomar, and M. Chiang, ‘‘Distributed optimization of coupled systems with applications to network utility maximization,’’ in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Toulouse, France, May 2006.
- [4] J. Huang, R. Berry, and M. L. Honig, ‘‘Distributed interference compensation for wireless networks,’’ *IEEE Journal of Selected Areas in Communications*, vol. 24, pp. 1074–1084, May. 2006.
- [5] A. H. Mohsenian-Rad, J. Huang, M. Chiang, and V. W. S. Wong, ‘‘Utility-optimal random access: Reduced complexity, fast convergence, and robust performance,’’ *IEEE Transactions on Wireless Communications*, vol. 8, no. 2, pp. 898–911, February 2009.
- [6] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. Prentice Hall, 1989.
- [7] D. Monderer and L. Shapley, ‘‘Potential games,’’ *Games and Economic Behavior*, vol. 14, pp. 124–143, 1996.
- [8] G. Scutari, S. Barbarossa, and D. Palomar, ‘‘Potential Games: A Framework for Vector Power Control Problems with Coupled Constraints,’’ *ICASSP 2006*, vol. 4, pp. 241–244, 2006.
- [9] I. Milchtaich, ‘‘Congestion games with player-specific payoff functions,’’ *Games and Economic Behavior*, 1996.
- [10] B. Vocking, ‘‘Congestion games: Optimization in competition,’’ *Proceedings of the 2nd Algorithms and Complexity in Durham Workshop*, Jan 2006.
- [11] M. Liu and Y. Wu, ‘‘Spectrum sharing as congestion games,’’ *Allerton conference*, 2008.
- [12] D. P. Bertsekas and R. Gallager, *Data Communications*, 2nd ed. Prentice Hall, 1992.
- [13] J. Mo and J. Warland, ‘‘Fair end-to-end window-based congestion control,’’ *IEEE/ACM Trans. on Net.*, vol. 8, pp. 556–567, Oct. 2000.
- [14] J. W. Lee, M. Chiang, and R. A. Calderbank, ‘‘Utility-optimal random-access control,’’ *IEEE Transactions on Wireless Communications*, vol. 6, 2007.
- [15] A. Mas-Colell, M. Whinston, and J. Green, *Microeconomic theory*. Oxford university press New York, 1995.
- [16] M. Chiang, ‘‘Balancing transport and physical layers in wireless multihop networks: Jointly optimal congestion control and power control,’’ *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 1, pp. 104–116, Jan 2005.