RC23828 (W0512-038) December 6, 2005
Computer Science

IBM Research Report

An O(*n* log *n*) Algorithm for Constrained Network Nash Equilibrium

D. Towsley
Department of Computer Science
University of Massachusetts
Amherst, MA 01003-4610

L. Wynter
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

An $O(n \log n)$ Algorithm for Constrained Network Nash Equilibria

D. Towsley* and L. Wynter†

April 11, 2005

Abstract

We present an efficient dual algorithm for solving the constrained Nash equilibrium problem over bipartite networks. This problem models selfish routing on networks. In the case of quadratic payoff functions for each agent, we provide an $O(n \log n)$ algorithm, where n is the number of agents in the game. This complexity bound improves known results for this class of problems.

1 Introduction

We focus on the Nash equilibrium paradigm played on a bipartite network structure. This generalizes the setting of [7] (also used in Fotakis et al. [5]), in which the network consisted of a set of parallel links, and all users on a given link experienced the same cost. A bipartite graph structure arises, for example, in dynamic routing, when the dynamics are defined over discretized time intervals. Indeed, routing decisions are, at any time epoch, made from sources with excess load, to destinations, with excess capacity. In other words, at any time period, it is possible to divide the players in the network into those with excess load to be sent and those having excess capacity, and a Nash network game solved over the resulting bipartite graph. Note that, in this framework, in a dynamic context, the structure of the graph changes from one time epoch to the next since the set of sending nodes and the set of receiving nodes changes at each time step.

The complexity of computing Nash equilibria is a related topic which has begun to receive considerable attention. Conitzer and Sandholm [2] prove the NP-hardness of certain Nash equilibrium problems, such as whether there exists a symmetric, 2-player (hence a general) Nash equilibrium in which some or all players have a utility of at least k, or a Pareto-optimal Nash equilibrium. Fabrikant et al. [4] study the complexity of computing pure network Nash equilibria (i.e. only one route is chosen by each agent); they provide a polynomial-time algorithm for the case in which there exists a single origin and destination through the use of Rosenthal's [11] potential function. None of the problems considered in [2] have coupling constraints across the players.

Papadimitriou and Roughgarden [9] present a proof and sketch of a method for finding all Nash equilibria in symmetric games, without coupling constraints, by solving a particular linear system. However, the linear system is of size polynomial in the

*Dept. of Computer Science, University of Massachusetts, Amherst, MA 01003-4610, USA, towsley@cs.umass.edu

†IBM Watson Research Center, P.O. Box 704, Yorktown heights, NY 10598, USA, lwynter@us.ibm.com

size of the compact representation of the game, and solving a linear system itself is of complexity $O(n^k)$ for $k \geq 2.5$ [14, 13]. Hence the complexity of the framework proposed in that reference is significantly higher than that which we propose here.

Fotakis et al. [5] examine mixed Nash network equilibria and provide a sketch of a polynomial-time algorithm for the case in which all users are identical. While their model does include coupling capacity constraints, their simplification of identical traffic means that the strategy spaces of each player are identical.

Blum et al. [1] present a path-following method for solving a particular form of Nash equilibrium games called graphical games. In graphical games, a link is drawn from each node i to every other node j whose actions influence the payoff of i. In our setting, such a representation would result in a complete, rather than bipartite, network structure. Furthermore, their path following algorithm is exponential in the size of the game. Vickrey and Koller [15] present a hill climbing heuristic for a similar type of Nash game.

A paper on a related topic but using a somewhat simplified payoff structure is that of Even-Dar et al [3], who study pure Nash equilibrium in load balancing problems. The corresponding payoff structure in our network game terminology is when all players experience the same payoff function when being routed to a common destination node. This simplification is not assumed in our work. Even-Dar et al. then determine bounds on the number of steps to reach the pure Nash equilibrium, which is related to a polynomial of degree equal to the sum of the weights of the jobs in their system.

In this paper, we consider a mixed Nash network equilibrium problem. The agents' utility functions are quadratic, and are not identical. In particular, the agents' loads are not identical, leading to user-specific flows. Furthermore, the network includes explicit capacity constraints, in our case, on the nodes themselves. These constraints *couple* the individual Nash games across agents, in addition to the usual interaction across players arising from the payoff functions. Capacity constraints of this type complexify the properties of the equilibrium and any algorithmic procedure for solving it. We make use of quadratic utility functions, which allow us to have strong analytical expressions for solutions, given partitions of the Lagrange multipliers. These expressions allow us to define an efficient and implementable algorithm for obtaining a particular Nash network equilibrium, and a proof as to when that equilibrium exists.

The structure of the paper is as follows. In Section 2 we present the basic notation and model. Section 3 provides the expressions for the analytical solutions of the Nash network game, in terms of the Lagrange multiplier values, and properties of the optimal solutions necessary for development of the algorithm. In Section 4, we provide an $O(n^2)$ dual algorithm as well as the $O(n \log n)$ algorithm.

2 Preliminaries

We consider a a strongly connected graph $G = (N, E, C, A)$ where the set of nodes is N, and the set of links is E. Here C denotes the service capacities associated with the nodes in N (node i can process C_i units of work in one time unit) and A denotes the demand processes, also associated with the nodes in N ($\{A_i : i = 0, 1, \ldots n\}$ denotes the amount of work brought into node i). Let $\{Q_i : j = 0, 1, \ldots n\}$ denote the load process at node i. The load process, which may represent random arrivals of work into the nodes, will allow us to define the bipartite graph: depending upon which nodes have excess load and which nodes have excess capacity, they will be represented on the sending or the receiving side of the bipartite network.

The process by which demands may be reallocated amongst agents for processing is defined by a Nash network game, first introduced in [6]. This procedure in general is dynamic; at at each time step, in a dynamic problem, the algorithm is rerun given the particular bipartite network that results from the load process at that time step. The Nash network game is played whenever there exists at least one node i with non-zero residual capacity $\tilde{C}_i := [C_i - Q_i]^+$ and at least one node i with non-zero residual load, $\tilde{A}_i :== [Q_i - C_i]^+$. When these conditions are not met, no reallocation is needed. In this paper, we consider a single time step, and develop an algorithm for a single-period Nash network game.

The game is defined by the payoff functions associated with the nodes. We consider quadratic payoff functions with the following form:

$$U_i(x) = \sum_{k \in N \setminus \{k\}} x_{ik} P_k(x). \tag{1}$$

In other words, the payoff to a particular node, i, is the product of its $k-$destined flow, x_{ik}, times a function of all nodes' flows to node k, summed over all destination nodes, k. We refer to the destination-specific function, P_k, as the receiving agent's *preference* function, and it has the following form:

$$P_k(x) = \beta_k - \sum_{i \in N \setminus \{k\}} a_{ik} x_{ik}, \tag{2}$$

where $\beta_k \geq 0$ and $a_{ik} \geq 0$.

The quadratic form is important for tractability. However, notice that the utility function exhibits reasonable properties; U_i increases for small values of x_{ik}, reaches a maximum and then decreases for large values of x_{ik}, representing reluctance on the part of k to take too heavy a load. Further, β_k represents a willingness-to-accept load on the part of node k, while a_{ik} denotes the relative weight from node k's point of view of receiving load from node i.

As mentioned above, the nodes divide into two classes, *sources* that have excess load beyond their capacity, and *receivers* that have excess capacity. Formally, let $S = \{i \in N : Q_i > C_i\}$ and $R = \{i \in N : Q_i < C_i\}$. Each node $i \in S$ solves the following optimization problem.

$$\max_{x_i} U_i(x) = \sum_{k \in R} x_{ik} \left(\beta_k - \sum_{l \in S} x_{l,k} \right). \tag{3}$$

subject to

$$\sum_{k \in R} x_{ik} \leq \tilde{A}_i, \tag{4}$$

$$\sum_{l \in S} x_{lk} \leq \tilde{C}_k, \quad \forall k \in R, \tag{5}$$

$$x_{ik} \geq 0, k \in R. \tag{6}$$

Hence, this formulation is a capacity-constrained Nash network equilibrium problem, where the constraints (5) *couple* the individual Nash problems of each sending node.

3 Optimal solutions

Existence of a solution to this constrained, quadratic Nash network game is guaranteed under the following assumption.

Assumption 1 *Let* $\sum_{k\in R}\tilde{C}_k \geq \sum_{i\in S}\tilde{A}_i$.

Theorem 1 *Let Assumption 1 hold. Then there exists at least one mixed strategy Nash equilibrium solution to (3)–(6).*

Proof. The preference functions (2) defined for each sending agent i to each receiver k are concave and continuously differnetiable in x_{ik} over the feasible region, and each agent's payoff function, U_i is the the sum of the preference functions. Under 1 the feasible region is not empty. Then, by Theorem 1 in [10] (see also [8]), the game of (3)–(6) is equivalent to a convex game and a Nash equilibrium exists. ∎

Let us now decompose the set of receiving nodes, R, into $R = R_u \cup R_c$ where receivers in R_c use up all of their excess allocation and receivers in R_u do not, and similarly for the set of sending nodes, $S = S_u \cup S_c$ where sources in S_c transfer all of their excess load elsewhere and sources in S_u do not. Further, denote λ_k as the Lagrange multiplier on the capacity constraints, (5), and μ_i the Lagrange multiplier on the demand constraints, (4).

Using this notation, and taking partial derivatives of the Lagrangian of the objective, (3), we obtain that the solutions to the optimization problem satisfy:

$$\beta_k - 2x_{ik} - \sum_{l\neq i;l\in S} x_{lk} + \lambda_k^i + \mu_i = 0, \quad i\in S; k\in R. \tag{7}$$

Note that $\lambda_k^i = 0$ if $k\in R_u$ and $\mu_i = 0$ if $i \in S_u$ and are less than or equal to zero otherwise.

The above equations have the following solutions.

$$x_{ik}^* = \begin{cases} (\beta_k - \sum_{l\in S_c}\mu_l)/(|S|+1), & i\in S_u; k\in R_u \\ (\beta_k + |S|\lambda_k^i - \sum_{l\neq i;l\in S}\lambda_k^l - \sum_{l\in S_c}\mu_l)/(|S|+1), & i\in S_u; k\in R_c, \\ \left(\beta_k + |S|\mu_i - \sum_{l\neq i;l\in S_c}\mu_l\right)/(|S|+1), & i\in S_c; k\in R_u, \\ (|S|+1)^{-1}* & \\ \left(\beta_k + |S|\lambda_k^i - \sum_{l\neq i;l\in S}\lambda_k^l + |S|\mu_i - \sum_{l\neq i;l\in S_c}\mu_l\right), & i\in S_c; k\in R_c \end{cases} \tag{8}$$

where the Lagrange multipliers $\{\lambda_k^i \leq 0 : i\in S_c; k\in R_c\}$ and $\{\mu_i \leq 0 : i \in S_c\}$ are solutions of

$$\sum_{i\in S}\lambda_k^i + \sum_{i\in S_c}\mu_i = (|S|+1)\tilde{C}_k - |S|\beta_k, \quad \forall k\in R_c, \tag{9}$$

$$\sum_{k\in R_c}\left(|S|\lambda_k^i - \sum_{l\in S_c, l\neq i}\lambda_k^l\right) +$$

$$|R|\left(|S|\mu_i - \sum_{\substack{l\in S_c\\ l\neq i}} \mu_l\right) \quad = \quad (|S|+1)\tilde{A}_i - \sum_{k\in R}\beta_k, \quad \forall i \in S_c \tag{10}$$

This Nash system with coupling capacity constraints does not admit a unique solution, as the system is under-determined due to the multiplicity of Lagrange multipliers on the coupling constraints, one for each sender. Indeed, each sender, i, sees the same capacity constraint for each receiver, k, but has its own value of the multiplier, λ_k^i. This amounts to $|S||R_c|$ multipliers, λ_k^i, but only R_c distinct equations.

In a general Nash equilibrium model with coupling constraints, one has no reason to believe that the multipliers on the coupling constraints be equal across senders, for a given receiver.

Lemma 1 *Let either $R_u \neq \emptyset$, that is, let there be at least one unconstrained receiver. Then, at a given Nash equilibrium solution, x^*, there exists a multiplier vector, λ, for the coupling capacity constraints which is equal across all sending nodes; that is, there exists a common vector, λ, such that $\lambda_k^i = \lambda_k^j = \lambda_k$, for all nodes $i, j \in S$, for all receivers, $k \in R$. Furthermore, the multipliers (λ, μ) satisfying that property are given by:*

$$\begin{bmatrix} \lambda \\ \mu \end{bmatrix} = M^{-1} \begin{bmatrix} (|S|+1)\tilde{C}_c - |S|\beta_c \\ (|S|+1)\tilde{A}_c - 1_{|S_c|\times|R|}\beta \end{bmatrix}, \tag{11}$$

where the matrix M is given by

$$M = \begin{bmatrix} |S|\, I_{|R_c|} & 1_{|R_c|\times|S_c|} \\ 1_{|S_c|\times|R_c|} & |R|\left[(|S|+1)I_{|S_c|} - 1_{|S_c|\times|S_c|}\right] \end{bmatrix}, \tag{12}$$

and $1_{m\times n}$ is the matrix of ones of dimension $m \times n$ $(m, n = 1, \ldots)$, I_m $(m = 1, \ldots)$ is the identity matrix of dimension $m \times m$, $\tilde{C}_c$ is the residual capacity of constrained receivers, $k \in R_c$, $C_k - Q_k$, and $\tilde{A}_c = Q_i - C_i$ is the vector of excess loads at constrained sending nodes, $i \in S_c$.

Proof. We make use of the equation system (16)-(17), in which the multipliers satisfy $\lambda_k^i = \lambda_k^j = \lambda_k$, for all nodes $i, j \in S$, for all receivers, $k \in R$. If a solution to this system exists, then the capacity constraint multipliers can be set equal at a solution to that value. Expressing (16)-(17) in terms of matrices and vectors, we obtain

$$\begin{bmatrix} |S|\, I_{|R_c|} & 1_{|R_c|\times|S_c|} \\ 1_{|S_c|\times|R_c|} & |R|\left[(|S|+1)I_{|S_c|} - 1_{|S_c|\times|S_c|}\right] \end{bmatrix} \begin{bmatrix} \lambda \\ \mu \end{bmatrix} = \\ (|S|+1)\begin{bmatrix} \tilde{C}_c \\ \tilde{A}_c \end{bmatrix} - \begin{bmatrix} |S|\beta_c \\ 1_{|S_c|\times|R|}\beta \end{bmatrix}. \tag{13}$$

Note that for ease of presentation, the indices of the vectors λ and μ have been redefined so as to contain no zero entries, and their indices are shifted towards zero as needed.

Then, rearranging terms, we obtain (11)–(12). Inverting the matrix, M, we obtain

$$M^{-1} = \begin{bmatrix} \frac{1}{|S|} I_{|R_c|} + \frac{|S_c|}{|S|\Delta} 1_{|R_c|\times|R_c|} & -\frac{1}{\Delta} 1_{|R_c|\times|S_c|} \\ -\frac{1}{\Delta} 1_{|S_c|\times|R_c|} & \frac{1}{|R|(|S|+1)}\left[I_{|S_c|} + \frac{|R||S|+|R_c|}{\Delta} 1_{|S_c|\times|S_c|}\right] \end{bmatrix} \tag{14}$$

where $\Delta = |R|\,|S|\,(|S_u|+1) - |R_c|\,|S_c|$. From (14), we have the condition that the inverse of M exists when $\Delta \neq 0$, which holds when $|R|\,|S|\,(|S_u|+1) > |R_c|\,|S_c|$. This is true provided that either R_u or S_u is nonempty. When this is the case, then M^{-1} exists, as does a vector of common multipliers, $\lambda_k^i = \lambda_k^j = \lambda_k$, for all nodes $i, j \in S$, for all receivers, $k \in R$, which concludes the proof. ∎

Under Lemma 1, let us suppose that the multipliers $\{\lambda_k^i\}$ can be made equal across the senders, i.e., $\lambda_k^i = \lambda_k$, $i \in S, k \in R_c$. When this is the case, (7) has the following solution

$$x_{ik}^* = \begin{cases} (\beta_k - \sum_{l\in S_c}\mu_l)/(|S|+1), & i \in S_u; k \in R_u \\ (\beta_k + \lambda_k - \sum_{l\in S_c}\mu_l)/(|S|+1), & i \in S_u; k \in R_c, \\ \left(\beta_k + |S|\mu_i - \sum_{l\neq i; l\in S_c}\mu_l\right)/(|S|+1), & i \in S_c; k \in R_u, \\ \left(\beta_k + \lambda_k + |S|\mu_i - \sum_{l\neq i; l\in S_c}\mu_l\right)/(|S|+1), & i \in S_c; k \in R_c \end{cases} \tag{15}$$

where the Lagrange multipliers $\{\lambda_k \leq 0 : k \in R_c\}$ and $\{\mu_i \leq 0 : i \in S_c\}$ are solutions of

$$|S|\lambda_k + \sum_{i\in S_c}\mu_i = (|S|+1)\tilde{C}_k - |S|\beta_k, \quad k \in R_c, \tag{16}$$

$$\sum_{k\in R_c}\lambda_k + |R|\left(|S|\mu_i - \sum_{l\in S_c, l\neq i}\mu_l\right) = (|S|+1)\tilde{A}_i - \sum_{k\in R}\beta_k, \quad i \in S_c \tag{17}$$

If we are only interested in $\mu = \sum_{i\in S_c}\mu_i$ and $\{\lambda_k\}_{k\in R_c}$, then (16) suffices coupled with

$$|S_c|\sum_{k\in R_c}\lambda_k + |R|(|S_u|+1)\mu = (|S|+1)\sum_{i\in S_c}\tilde{A}_i - |S_c|\sum_{k\in R}\beta_k \tag{18}$$

This explicit solution for the equilibrium flows shall be used in the development of the dual algorithm. First, we present some additional results which will be required.

The optimal solutions exhibit the following properties in the case that $\lambda_k^i = \lambda_k$.

Lemma 2 *Let the sending nodes, $i = 1, \ldots |S|$ be numbered such that $\tilde{A}_1 \leq \tilde{A}_2 \leq \cdots \leq \tilde{A}_{|S|}$ and the receiving nodes, $k = 1, \ldots |R|$ be numbered such that $(|S|+1)\tilde{C}_1 - |S|\beta_1 \leq (|S|+1)\tilde{C}_2 - |S|\beta_2 \leq \cdots (|S|+1)\tilde{C}_{|R|} - |S|\beta_{|R|}$, then*

1. *if $i, j \in S$, $i < j$, and $i \in S_u$, then $j \in S_u$,*
2. *if $i, j \in R$, $i < j$, and $i \in R_u$, then $j \in R_u$.*

Proof. Consider two nodes $i, j \in N$. Now consider claim 1. We assume that $j \in R_c$. From (17), we have that

$$\begin{aligned} \tilde{A}_j &= \frac{1}{|S|+1}\left(\sum_{k\in R}\beta_k + \sum_{k\in R_c}\lambda_k + |R|(|S|+1)\mu_j - |R|\mu\right) \\ &> \sum_{k\in R}x_{ik}^* \end{aligned}$$

$$
\begin{aligned}
&= \frac{1}{|S|+1}\left(\sum_{k\in R}\beta_k + \sum_{k\in R_c}\lambda_k - |R|\mu\right) \\
&> \frac{1}{|S|+1}\left(\sum_{k\in R}\beta_k + \sum_{k\in R_c}\lambda_k + |R|(|S|+1)\mu_j - |R|\mu\right) = \tilde{A}_j,
\end{aligned}
$$

which is a contradicttion. Thus, $j \in S_u$.

Consider claim 2. Assume that $j \in R_c$. We have from (15)

$$
\begin{aligned}
\sum_{i\in S} x^*_{ij} &= \frac{|S|(\beta_j+\lambda_j)+\mu}{|S|+1} \\
&= \tilde{C}_j
\end{aligned}
$$

or

$$
(|S|+1)\tilde{C}_j - |S|\beta_j \le \mu
$$

since $\lambda_j \le 0$. On the other hand, since $i \in R_u$, we have

$$
\frac{|S|\beta_i+\mu}{\tilde{C}_i} < |S|+1
$$

or

$$
\mu < (|S|+1)\tilde{C}_i - |S|\beta_i
$$

But this contradicts the fact that $(|S|+1)\tilde{C}_i - |S|\beta_i \le (|S|+1)\tilde{C}_j - |S|\beta_j$. Hence we conclude that $j \in R_u$. ∎

Next we associate an *aggregate flow rate* with each node in S and R as follows. Consider $i \in S$. Suppose that we consider a source load distribution that would be produced by assuming that all sources $j < i$ are constrained, sources $j > i$ are unconstrained and source i is constrained with $\mu_i = 0$. Let $\tilde{S}_j$ denote the flow out of source j in this case. Then

$$
\tilde{S}_j = \begin{cases} \tilde{A}_j, & j \le i, \\ \tilde{A}_i, & i < j \le |S|. \end{cases}
$$

The case $j > i$ follows from summing up x^*_{jk} over all receivers using (15) and then making use of (17) for i to obtain the final result. The total flow rate from sources to receivers is $F^s_i = \sum_{j\le i}\tilde{A}_j + (|S|-i)\tilde{A}_i$. Note that this is also the total flow from sources to receivers for the system where the source load allocation is

$$
\tilde{A}'_j = \tilde{S}_j, \quad j \in S
$$

under the assumption that $S_c = S$ for this allocation. This interpretation will be useful when we present a solution algorithm.

Now consider $j \in R$. Suppose that we consider a receiver load allocation that would be produced by assuming that receivers $j < i$ are at capacity, receivers $j > i$ are unconstrained, and receiver i is constrained with $\lambda_i = 0$. Let $\tilde{R}_j$ denote the flow into receiver j for this case. From the definitions of x_{ik} in (15), $\tilde{R}_j$ is given by

$$
\tilde{R}_j = \begin{cases} \tilde{C}_j, & j \le i, \\ \tilde{C}_i + \frac{|S|(\beta_j-\beta_i)}{|S|+1}, & i < j \le |R|. \end{cases} \tag{19}
$$

The total flow rate into the receivers is then

$$\sum_{j\in R}\tilde{R}_j = \sum_{j\le i}\tilde{C}_j + (|R|-i)\tilde{C}_i + \sum_{j=i+1}^{|R|}\frac{|S|(\beta_j-\beta_i)}{|S|+1}.$$

Based on the above, it is possible to order all of the nodes in a single list, whether they are sources or receivers, based on their aggregate flow rates. Let F_i denote the aggregate flow rate for node i. It is defined as follows,

$$F_i = \begin{cases} (|S|-(i-1))\tilde{A}_i + \sum_{j=1}^{i-1}\tilde{A}_j, & i\in S, \\ \sum_{k\le i}\tilde{C}_k + (|R|-i)\tilde{C}_i + \sum_{k=i+1}^{|R|}\frac{|S|(\beta_k-\beta_i)}{|S|+1}, & i\in R \end{cases} \tag{20}$$

Lemma 3 *We have the following:*

- *if* $i\in S_u$ *then* $\sum_{k\in R}\sum_{j\in S}x^*_{jk}\le F_i$
- *if* $i\in S_c$ *and* $\mu_i<0$ *then* $\sum_{k\in R}\sum_{j\in S}x^*_{jk}>F_i$
- *if* $i\in R_u$ *then* $\sum_{k\in S}\sum_{i\in R}x^*_{kj}\le F_i$
- *if* $i\in R_c$ *and* $\lambda_i<0$ *then* $\sum_{k\in S}\sum_{j\in R}x^*_{kj}>F_i$

Proof. Consider the first claim. According to Lemma 2, there exists an $i_0<i$ such that $j\in S_c$, $j\le i_0$ and $j\in S_u$, $j>i_0$. If $S_u=S$ then take $i_0=0$. It is easily shown that the aggregate flow out of S for the optimum solution, F^*, is given by

$$\begin{aligned} F^* &= \sum_{j=1}^{i_0}\tilde{A}_j + \sum_{j=i_0+1}^{|S|}\sum_{k\in R}x^*_{jk} \\ &= \sum_{j=1}^{i_0}\tilde{A}_j + (|S|-i_0)\sum_{k\in R}x^*_{i_0+1,k} \\ &\le \sum_{j=1}^{i_0}\tilde{A}_j + (|S|-i_0)\tilde{A}_{i_0+1} \\ &\le \sum_{j=1}^{i}\tilde{A}_j + (|S|-i)\tilde{A}_{i+1} \\ &= F_i \end{aligned}$$

The first equality is because all unconstrained senders have the same flow out. The first inequality is a consequence of i_0+1 being an unconstrained sender. The remaining inequality and equality are pretty straightforward. The remaining claims are established in the same manner. ■

We have the following corollary

Corollary 1 *Let the nodes,* $i=1,\dots,|S|+|R|$*, be numbered such that* $F_1\le F_2\le\cdots\le F_{|S|+|R|}$*, and the sets of sending and receiving nodes each numbered according to the ordering of Lemma 2. then*

1. *if* $i,j\in S$*,* $i<j$*, and* $i\in S_u$*, then* $j\in S_u$*,*

2. *if* $i, j \in R$, $i < j$, *and* $i \in R_u$, *then* $j \in R_u$,

3. *if* $i \in S$, $j \in R$, $i < j$, *and* $i \in S_u$, *then* $j \in R_u$,

4. *if* $i \in R$, $j \in S$, $i < j$, *and* $i \in R_u$, *then* $j \in S_u$.

Proof. Consider first claim 1. Let F^* denote the aggregate flow from senders to receivers. Assume that $j \in S_c$, then by Lemma 3, $F^* > F_j$. On the other hand it follows from Lemma 3 that $F_i \geq F^*$. This cannot be since $F_i \leq F_j$. Therefore $j \in S_u$. The other claims follow in a similar manner. ■

The following lemma is useful as well for motivating our algorithm.

Lemma 4 *[Sensitivity analysis] The multiplier values* μ *and* $lambda$ *are such that*

1. μ_i *increases in* $\tilde{A}$;

2. λ_i *decreases in* $\tilde{A}$.

Proof: Suppose $i \in S_c$, then μ_i is given by the following equation (11):

$$\mu_i = -\frac{\sum_{k\in R_c}(|S|+1)\tilde{C}_k - |S|\beta_k}{\Delta} + \frac{\tilde{A}_i}{|R|} - \frac{\sum_{j\in R}\beta_j}{|R|(|S|+1)} + \frac{|R||S|+|R_c|}{|R|\Delta}\sum_{j\in S_c}\tilde{A}_j - \frac{(|R||S|+|R_c|)|S_c|\sum_{k\in R}\beta_k}{|R|(|S|+1)\Delta}. \quad (21)$$

The first claim follows from the fact that the coefficients in front of $\tilde{A}_j$ are positive. Suppose now that $k \in R_c$, then λ_k is

$$\lambda_k = \mathcal{C} - \frac{(|S|+1)}{\Delta}\sum_{j\in S_c}\tilde{A}_j$$

for some constant $\mathcal{C}$. The second claim follows from the fact that the coefficients in front of $\tilde{A}_j$ are negative. ■

We are now able to present the algorithms.

4 An $O(n \log n)$ dual algorithm for the constrained Nash network equilibrium

A consequence of the above properties is that, if $i \in S_c$ for a given $\tilde{A}^{(1)}$, then $i \in S_c$ for any $\tilde{A}^{(2)} \geq \tilde{A}^{(1)}$ componentwise. This is due to the fact that $\mu_i \leq 0$ and that the the multiplier is continuous in $\tilde{A}$. Similarly, if some receiver $k \in R_u$ for a given $\tilde{A}^{(1)}$, then $k \in R_u$ for any $\tilde{A}^{(2)} \geq \tilde{A}^{(1)}$.

Notation. U is used to represent the set of sources that have not been fully loaded and R_u the set of receivers that have not reached their capacity. Initially $U = S$ and $R_u = R$ and at each iteration either U is reduced by one or V is increased by one. We denote q_i to be the load on source $i \in S$. As the algorithm proceeds, q_i is the same for all $i \in U$.

We assume that the nodes are indexed such that $0 < F_1 \leq F_2 \leq \cdots \leq F_{|S|+|R|}$. The index j will be used to refer to the next candidate node to either become fully loaded, if it is a sender, or become constrained if it is a receiver during the course of algorithm.

Algorithm 1.

1. *Initialization.*
   Set $U = S$; $S_c = S$; $R_u = R$; $R_c = \emptyset$; $j = 1$.

2. *Loop.*
   if $j \in S$, then

   (a) Solve for $\{\lambda_k\}, \{\mu_\ell\}$ for all $k \in R_c$ and $\ell \in S_c$ using (16) and (17) with $\tilde{A}_i = \tilde{A}_j$ for all $i \in U$, $\tilde{A}_l$ for all $l \in S \setminus U$, and $\tilde{C}_w$ for all $w \in R_c$;

   (b) if Consistent(j)=1, then $U = U \setminus \{j\}$;
   else go to step 3.

   else

   (a) Set $R_c = R_c \cup \{j\}$, set $\lambda_j = 0$, and solve for $\{\lambda_i : i \in R_c, i \neq j\}$, $\{\mu_\ell\}$, for all $\ell \in S_c$ and q_j using (16) and (17) with $\tilde{A}_i = q_j$ for all $i \in U$, $\tilde{A}_l$ for all $l \in S \setminus U$, $\tilde{C}_w$ for all $w \in R_c \setminus \{j\}$ and q_j in place of $\tilde{C}_j$;

   (b) if Consistent(j)=1, then $R_u = R_u \setminus \{j\}$;

   else reset $R_c = R_c \setminus \{j\}$ and go to step 3.

   $j = j + 1$
   End loop

3. *Termination.*
   Set $S_c = S \setminus U$; $S_u = U$. Solve for optimal $x^*_{i,k}$.

Consistency Test. Consistent(j):

1. If $\lambda_k \leq 0$ and $\mu_i \leq 0$ for all $i = 1 \ldots |S_c|$, $k = 1 \ldots |R_c|$, then go to Step 2. Otherwise, return $Consistent(j) = 0$.

2. Let $\chi_k = \tilde{C}_k$ for all $k \leq j,\ k \in R$, and if $k > j, k \in R$, then $\chi_k = \tilde{C}_\ell + [|S|(\beta_k - \beta_\ell)]/(|S| + 1)$, where $\ell = \max\{k : k = 1, \ldots, j; k \in R\}$, from (19), and let $\chi_i = \tilde{A}_i$ for all $i \leq j,\ i \in U$, and for $i > j$, $\chi_i = \tilde{A}_q$, with $q = \max\{i : i = 1, \ldots j, S; i \in S\}$.

   If the pair of equations are satisfied:

$$\sum_{k=1\ldots|R_c|} \lambda_k(\chi_k - \tilde{C}_k) = 0, \qquad (22)$$

$$\sum_{i=1\ldots|S_c|} \mu_i(\chi_i - \tilde{A}_i) = 0, \qquad (23)$$

   then return Consistent(j)=1.

3. Else, return Consistent(j)=0.

Algorithm 2.
Algorithm 2 makes use of binary search to achieve a complexity of $O(n \log n)$.

1. *Initialization.*
   if node $|S| + |R| \in S$ then $R_c = R$, $S_c = S \setminus \{|S| + |R|\}$
   else $R_c = R \setminus \{|S| + |R|\}$, $S_c = S$;

solve for $\{\lambda_k\}, \{\mu_\ell\}$ for all $k \in R_c$ and $\ell \in S_c$, using (16) and (17) with $\tilde{A}_i$ for all $i \in S_c$, and $\tilde{C}_w$ for all $w \in R_c$;
if Consistent(j) = 1 then goto 3
else set $j_0 = 1$; $j_1 = |S| + |R| - 1$.

2. while $j_1 > j_0 + 1$ do
set $j = \lfloor (j_0 + j_1)/2 \rfloor$
$R_c = \{i : i \leq j, i \in R\}$; $U = \{i : i > j, i \in S\}$;
if $j \in S$ then

   (a) Solve for $\{\lambda_k\}, \{\mu_\ell\}$ for all $k \in R_c$ and $\ell \in S_c$, using (16) and (17) with $\tilde{A}_i = \tilde{A}_j$ for all $i \in U$, $\tilde{A}_l$ for all $l \in S \setminus U$, and $\tilde{C}_w$ for all $w \in R_c$;

   (b) if Consistent(j)=1, then $j_0 = j$;
   else $j_1 = j$.

else

   (a) Set $\lambda_j = 0$, and solve for $\{\lambda_i : i \in R_c, i \neq j\}$, $\{\mu_\ell\}$, for all $\ell \in S_c$ and q_j using (16) and (17) with $\tilde{A}_i = q_j$ for all $i \in U$, $\tilde{A}_l$ for all $l \in S \setminus U$, $\tilde{C}_w$ for all $w \in R_c \setminus \{j\}$ and q_j in place of $\tilde{C}_j$;

   (b) if Consistent(j)=1, then $j_0 = j$;
   else $j_1 = j$.

End while

3. *Termination.*
Set $S_c = S \setminus U$. Solve for optimal $x^*_{i,k}$.

In principle, an algorithm which tests which constraints are active and for every possible combination, tests the KKT condition, is of exponential complexity. Indeed, there are $2^{|S|+|R|}$ constraints, each of which can be active or not. However, thanks to Lemma 2, we are able to solve the problem using a KKT-based active set approach with complexity $O(n^2)$ and $O(n \log n)$, respectively, in Algorithms 1 and 2.

Lemma 5 *The complexity of Algorithm 1 is $O(n^2)$*

Proof. The outer loop executes each time a multiplier must be updated, checking whether it is a receiver's multiplier, a sender's multiplier, or none. Due to the non-state dependent ordering of Lemma 5, this happens once for each node, since the next multiplier to become nonzero is the next in the ordering. Hence, the outer loop executes $O(n)$ times. The inner loop checks for consistency at a particular set of active multipliers, where the number of active multipliers can increase by one at each call to the loop. The formulæ rely only on sums of the q_i. Hence, the two nested loops require in all $O(n^2)$. Finally, obtaining a primal (flow) solution requires in general one computation for each link of the network, i.e., $O(m)$. However, using our algorithm, it requires only four computations for each sending node, i.e., $O(n)$ operations, using the explicit formulæfor the x^*_{ik} as a function of active multiplier values $\{\lambda^*, \mu^*\}$, since there are four possible cases leading to different flow values for each sender. Hence, the algorithm is overall $O(n^2)$. ∎

Lemma 6 *The complexity of Algorithm 2 is $O(n \log n)$*

Proof. The outer loop of Algorithm 2 makes use of a binary search on the single ordered list of nodes satisfying $F_1 \leq F_2 \leq \cdots \leq F_{|S|+|R|}$, instead of using a linear search on the sender and receiver lists individually. Hence, the complexity of the outer loop is reduced to $O(\log n)$ and the remaining steps are unchanged. The overall complexity of Algorithm 2 is thus $O(n \log n)$. ∎

References

[1] B. Blum, C. R. Shelton, and D. Koller, "A Continuation Method for Nash Equilibria in Structured Games", in Proceedings of *IJCAI*, 2003.

[2] V. Conitzer and T. Sandholm, "Complexity results about Nash equilibrium", in Proceedings of *IJCAI*, 2003.

[3] E. Even-Dar, A. Kesselman and Y. Mansour, "Convergence Time to Nash Equilibria," in Proceedings of *ICALP*, 2003.

[4] A. Fabrikant, C. Papadimitriou, and K. Talwar, "The complexity of pure Nash equilibria", in Proceedings of *STOC*, 604–612, 2004..

[5] D. Fotakis, S. Kontogiannis, E. Koutsoupias, M. Mavronicolas, and P. Spirakis, "The Structure and Complexity of Nash Equilibria for a Selfish Routing Game" in Proceedings of *ICALP: International Colloquium on Automata, Languages and Programming*, 123–134, 2002.

[6] Z. Liu, V. Misra, L. Wynter. "Dynamic Offloading in a Multi-Provider Environment: A Behavioral Framework for Use in Influencing Peering" in Proceedings of IEEE/ACM CC Grid 2004: Global and Peer-to-Peer Computing, Chicago, IL, USA, 2004.

[7] E. Koutsoupias and C.H. Papadimitriou, 'Worst-case equilibria", in Proceedings of *STACS*, 404–413, 1999.

[8] A. Orda, R. Rom and N. Shimkin, "Competitive Routing in Multi-User Environments", *IEEE/ACM Transactions on Networking* 1, 510–521, 1993.

[9] C. Papadimitriou and T. Roughgarden, "Computing equilibria in multiplayer games", in Proceedings of *SODA*, 2005.

[10] J. Rosen, "Existence and uniqueness of equilibrium points for concave n-person games", *Econometrica* 33, 520–534, 1965.

[11] R. W. Rosentahl, "A class of games possessing pure-strategy nash equilibria", *International J. of Game Theory*, 2, 65–67, 1973.

[12] A. Tantawi and D. Towsley, "Optimal static load balancing in distributed computer systems", *J. of the ACM* 32-2, 445–465, 1985.

[13] D. Coppersmith and S. Winograd, "Matrix multiplication via arithmetic progressions", *J. Symbolic Comput.* 9, 251–280, 1990.

[14] A. Schonhage, "The fundamental theorem of algebra in terms of computational complexity", Technical report, Math. Institut der. Univ. Tubingen, 26, 1982.

[15] D. Vickrey and D. Koller, "Multi-Agent Algorithms for Solving Graphical Games", in Proceedings of *AAAI*, 2002.