Journal of Universal Computer Science, vol. 14, no. 5 (2008), 745-765
submitted: 1/11/07, accepted: 28/2/08, appeared: 1/3/08 © J.UCS

Optimal Transit Price Negotiation: The Distributed Learning Perspective

Dominique Barth
(PRiSM Laboratory, Versailles, France
barth@prism.uvsq.fr)

Loubna Echabbi
(INPT, Rabat, Morocco
echabbi@inpt.ac.ma)

Chahinez Hamlaoui
(PRiSM Laboratory, Versailles, France
hamlaoui@prism.uvsq.fr)

Abstract: We present a distributed learning algorithm for optimizing transit prices in the inter-domain routing framework. We present a combined game theoretical and distributed algorithmic analysis, where the notion of Nash equilibrium with the first approach meets the notion of stability in the second. We show that providers can learn how to strategically set their prices according to a Nash equilibrium; even when assuming incomplete information. We validate our theoretical model by simulations confirming the expected outcome. Moreover, we observe that some unilateral deviations from the proposed rule do not seem to affect the dynamic of the system.

Key Words: interdomain prices, games with incomplete information, learning , stability
Category: I.2.6, G.1.7, C.2.4

1 Introduction

The inter-domain market faces two related decisive problems: A price negotiation problem that allows economic contracts, and a routing problem where routing decisions are made according to the concluded business relationships. Some recent works [Afergan 2006], [Barth et all 2007], [Feigenbaum et all 2002], [La and Anantharam 2002] propose to unify these two problems more tightly by enabling a more dynamic interaction between transit price propositions and routing decisions. In order to capture the dynamic aspect of such an interaction, the authors propose the use of a repeated game approach. However, the proposed models require complete information on the underlying game. Such an approach is necessary to analyze the possible outcome of the game. However, it is necessary to go in depth into the problem taking into account some practical constraints. Indeed, providers are not aware about the complete details concerning the topology linking them. They are not even capable of detecting how many providers are actually competing for a specific traffic flow. Accurate information about which providers have actually been chosen to route the traffic and at which price is not necessarily available to all providers at each moment. Hence, an analysis of the underlying game assuming incomplete information is necessary to tackle the problem.

In [Sastry et all. 2004], a decentralized learning algorithm is proposed in order to reach a Nash equilibrium in games with incomplete information. The model considers players as a team of automata, a proof is provided to ensure that convergence can take place into a Nash equilibrium when players follow the learning rule. In [Maillé and Tuffin 2006], a similar algorithm is used to discover how many parallel TCP sessions should be open in a game where TCP connections compete for bandwidth. In [Xing and Chandramouli 2004], learning theory is also used to study the game related to distributed discrete power control in wireless networks. Both in [Sastry et all. 2004] and in [Xing and Chandramouli 2004] an ordinary equation is derived to describe the system and to prove that convergence is tightly related to the Nash equilibrium notion. Indeed the proposed learning algorithm is proved to converge only to a point which is a Nash equilibrium. Moreover pure Nash equilibriums are proved to be asymptotically stable. Thus they act as attractors of the system evolution under the learning algorithm. That means that when initial conditions are close to a point which is a pure Nash equilibrium then the algorithm is ensured to converge to it. Convergence is not ensured otherwise, even with initial conditions close to a mixed Nash equilibrium.

In our problem, we prove that the use of the decentralized learning algorithm ensures convergence only to a pure Nash equilibrium or a specific mixed equilibrium defined as hidden Nash equilibrium. Thus, players are ensured to play their optimal strategies (in the N.E. sense). Pure and hidden Nash equilibriums are the only stable points of the learning algorithm. We also prove the convergence of the algorithm for specific situations. We rely on simulations to state convergence in general cases. The advantage with the use of the learning algorithm is that players are ensured to play their optimal strategy according to a Nash equilibrium, even with lack of information about the underlying topology. We observe via our simulations that the learning process is not disturbed when some providers deviate unilateraly from the initial rule.

The outline of the paper is the following: First, we introduce the considered transit price negotiation model. Then a game theoretical and a distributed algorithmic model are introduced assuming incomplete information. The learning rule that should be applied by each provider to update his strategy is then presented. We conclude our theoretical study by a stability analysis and a discussion about convergence. We evaluate our model via a simulation framework where a communication model is presented to coordinate providers' individual decisions.

Different scenarios are considered to confirm convergence of the learning algorithm and to investigate the behavior of the system when some providers deviate from the initial learning rule.

2 The transit price negotiation model

We start with the model presented in [Shakkottai and Srikant 2006] and generalize the results. In addition to local providers competing for customers, we consider interactions with transit providers that are giving access to the rest of the internet. Figure 1 illustrates the considered model.

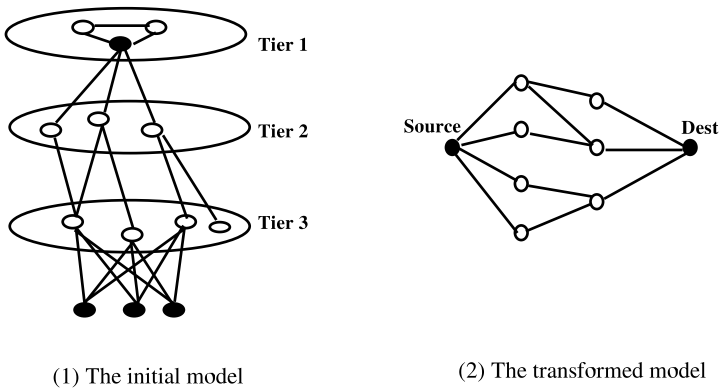

Figure 1: Illustration of the considered model

As in (1), we consider that the traffic destination is the provider at the highest level of hierarchy giving access to the rest of the internet. Customers are supposed to be connected to the same direct providers and that they are price takers: that is they always choose the minimum cost direct provider. Hence, they can be considered as a single source of traffic. The transit negotiation problem can be modelled then as a source and destination connected via a graph of transit providers as in (2). The problem that faces transit providers is how to fix their transit price in order to be chosen to route the traffic. A transit provider does not need only to propose an attractive transit price but also to choose itself the lowest price providers. In the following we describe the generated game model.

2.1 The game with 1 Source and 1 Destination

The game is defined by a source willing to send traffic into a specific destination while using transit providers. Each transit provider is considered as a player which is proposing a transit service against some transit price that he has to set strategically. Each player announces a price to his neighbors with the corresponding route into the destination. The price at which the route is announced is the price at which it was bought plus a transit price determined by the announcing player. In other words, when a provider decides to buy a route from his neighbor, he can announce this route to his own neighbors while proposing an adequate price of the route including his transit price. [1] Thus, the negotiation follows a cascade like model from the destination backward to the source, where each player in the path plays both the customer and the provider role. The objective of each provider is to maximize his own benefit by proposing an attractive transit price but also by choosing the lowest price providers(routes). In case of identical price announces, a player can choose his provider following a pre-order on his providers. We say then that the chosen provider has priority over the others. We call **transit price negotiation game** the game that models how the providers should set their transit prices in order to be chosen on the route from $Source$ to $Dest$.

We consider that the source has an upper bound on price. If the proposed prices are higher than that limit, it does not send the traffic. We will denote it p_{max}. We consider discrete transit prices. Such price discretization depends on the encoding format in control packets used to announce prices. For instance here we take a unit discretization, that is a provider transit price can take values as $1, 2, 3, \ldots$

A provider's action in the transit price negotiation game consists of fixing a transit price at a given stage.

We consider that each player has a distribution over his available actions. Let $S_i = ((s_{i,j})_{j=1,\ldots,p_{max}})$ denote the strategy probability vector employed by player i, where $s_{i,j}$ is the probability that the player i sets a price j. We denote S the strategy profile defined by the players strategy vectors. Note that a pure strategy is a specific mixed strategy with a unitary probability vector.

Definition 1 *Let C be the collection of direct paths (without cycles) between $Source$ and $Dest$. A path $c \in C$ is* **dominant** *if $\forall c' \in C$:*

- *either $|c| < |c'|$*
- *or $|c| = |c'|$ but c is proposed by a priority provider.*

[1] If rational, a provider should propose the route at the price he has bought it plus at least one unit as a transit price in order to have a strictly positive benefit.

where $|c|$ denotes the length of the path c in term of hops.
We denote $c_{dominant}$ the dominant path. Let c_{second} represent the dominant path when $C - \{c_{dominant}\}$ is considered.

The dominant path is unique since the routing choice is deterministic (minimum cost route choice with a rule to break tie).

Proposition 1 *Let S be a pure strategy in the transit price negotiation game with one source-one destination such that the chosen route is $c_{dominant}$ and :*

(a) $$\sum_{i \in c_{dominant}} \sum_{j=1}^{p_{max}} s_{i,j} \cdot j = \begin{cases} |c_{second}| & \text{if } c_{dominant} \text{ has priority} \\ |c_{second}| - 1 & \text{otherwise} \end{cases}$$

(b) $\forall i' \in c_{second} \quad s_{i',1} = 1$
Then S is a pure Nash equilibrium.

It is easy to see that a strategy vector S that satisfies both properties (a) and (b) is a Nash equilibrium. Indeed, The key idea is that a path c is necessarily announced to the source at a price at least equal to $|c|$. No player on the dominant path has an incentive to increase his price since the second route will be cheaper. They do not have incentive to lower their price since they will get less benefit. The other players have no other choice since they will not be chosen anyway.

The question that arises is whether the described profile is the only one possible Nash equilibrium. Let us first consider some notations. We denote c_{winner} the chosen route by the source, which is by the way the lowest price route. Let c_{sec} represent the second lowest price route. We say that a route is **valid** if it is announced to the source at a price at most p_{max}. The following proposition gives a characterization of pure Nash equilibriums in our game.

Proposition 2 *Let S be a pure Nash equilibrium then one of the following assertions is correct:*

1. *There is no valid route: each route c is announced at a price $p(c) > \frac{|c|}{|c|-1} \times p_{max}$.*
2. *There is an only one valid route: The valid route is announced at p_{max} and each other route c is announced at a price $p(c) > \frac{|c|}{|c|-1} \times p_{max}$.*
3. *There is at least two valid routes:*
   - *The winning path is announced at a price p^* equal to $|c_{sec}|$ or $|c_{sec}|$-1 depending on priority,*
   - *All players in c_{sec} are announcing a transit price equal to 1.*

- *If c is a route shorter than c_{win} in terms of hops, then it is necessarily announced at least at $\frac{|c|}{|c|-1} \times p^*$.*

Proof. Since S is an N.E. then each player has no incentive to unilaterally deviate from its chosen strategy.

- In the first case, when no route is valid, this means that no player can make his route valid by lowering alone his price. This means that the price of the route is already greater than p_{max} even if the player announces a price equal to 1. By writing this inequality for each player in a route c, and summing the $|c|$ inequality, we can deduce that $(|c| - 1)p(c) > |c|p^*$.
- In the second case, the only one valid route is necessarily announced at a price p_{max} since this is the maximum that can be announced. Similarly to the first case, since players on other routes do not regret their strategy, we can deduce that $(|c| - 1)p(c) > |c|p^*$.
- In the third case, there are at least two valid routes. Consider p' the price at which the route related to c_{sec} is announced. Then the winning path is announced at p' (or $p' - 1$) otherwise players on the winning path will regret their strategies.
  Since players on the second path do not regret their strategies, then no one of them can lower his price by one to make the route more attractive. Hence, their price must be 1 since this is the only situation where they can no more lower their price. Then $p' = |c_{sec}|$ and thus we have the desired property on the winning path's price.
  If there is some path c shorter than $c_{winning}$ in terms of hops, then similarly to the first and second case, we can deduce the property on the price at which c was announced.

Note that the pure N.E. in the proposition 1 is a special case of the third case in proposition 2. Hence if the winning route is the dominant one with multiple valid routes, the pure N.E. is necessarily the one described by proposition 1.
The following definition introduces a slightly different outcome to our game considering mixed strategies.

Definition 2 *Let S be a Nash equilibrium in the transit price negotiation game with one source-one destination such that:*

(a) $\forall i \in c_{winning}$ S_i is pure and

$$\sum_{i \in c_{winning}} \sum_{j=1}^{p_{max}} s_{i,j} \cdot j = \begin{cases} |c_{sec}| & \textit{if } c_{winning} \textit{ has priority} \\ |c_{sec}| - 1 & \textit{otherwise} \end{cases}$$

(b) $\forall i' \in c_{sec}$ $s_{i',j}$ *is such that the expected utility of players in the dominant path is lower when they move from their strategy in (a).*

Then S *is said to be a* **hidden pure Nash Equilibrium**.

In this definition, S is a particular mixed N.E. from which we can deduce a pure N.E. as in proposition 1.
In reality nodes have only a local view of the game including the topology and thus ignore the length of the possible routes. They are not even aware of the number of players in the game. Each provider has to learn his optimal strategy somehow that providers learn how to set their price in a decentralized way, and thus reach the N.E. where they are ensured to be on the winning route.

2.2 The distributed algorithmic model

As explained before, the distributed context of the model does not enable players to predict their optimal strategy in a one-stage game. They do not even detect if they are on the shortest path from the source to the destination. Hence the game should be analyzed as a repeated game where providers update their strategies at each stage in order to learn their optimal prices. We call the way they choose to update their strategies their *local strategic process*.
We denote $s^t_{i,j}$ the probability that the player i sets a price j at stage t and $S^t_i = ((s^t_{i,j})_{j=1,\ldots,p_{max}})$ the probability vector of player i at stage t. We denote $Proc_i$ the function that represents provider i's local strategic process and $Proc = \{Proc_i$ for each provider $i\}$ the vector of local strategic processes. In this way, let us consider the following algorithm $\mathcal{A}_{Proc}$ which corresponds to the multi-stage game:

1. Each player sets his price according to his chosen strategy.
2. Each player announces his chosen route depending on the received price announcement. Thus the minimum cost routes are computed in a distributed way.
3. Each player deduces his benefit regarding to the outcome of the distributed routing algorithm.
4. Each player changes his strategy using his local chosen strategic process $Proc_i$ [2] which is supposed to learn him his optimal strategy.

Note that to change his strategy, each provider only knows if the traffic crosses him (if he belongs to the chosen route), his own announced price and the price

[2] for example in [Barth et all 2007] we have proposed a tatonment process where a provider decreases his price by one to attract the traffic when he had lost the market at the former stage otherwise he increases his price by one

announced by his provider (if he is not directly linked to the destination). In this sense, we can talk about a distributed algorithm. It can be implemented locally by each provider.

Let us denote by R^t and S^t the source chosen route and the vector of strategies obtained after the t^{th} execution of the algorithm $\mathcal{A}_{Proc}$. Let $Price^t = ((Price_i^t)_{i=1,\dots,n})$ be the vector of prices set by players at stage t and let $random() : S^t \rightarrow Price^t$ be the function that sets prices according to the strategy vector S^t in step 2 of the algorithm. Let $Rout() : Price^t \rightarrow R^t$ be the deterministic function that computes the minimum cost route in step 3 of the algorithm, and let $Benef_i^t$ represent the benefit of player i induced by the computation of R^t, where $Benef_i^t = Price_i^t$ if i belongs to the chosen route R and 0 otherwise.

Given an instance (G, S^0, R^0), where G is the considered interdomain graph and R^0 an empty set, and given a vector of local strategic processes $Proc = \{Proc_i$ for each provider $i\}$, the deterministic polynomial decentralized algorithm $\mathcal{A}_{Proc}$ computes a new vector of strategies $\mathcal{A}_{Proc}(S^t) = S^{t+1}$ leading to a new prices $Price^t = Random(S^t)$ and routes $R^{t+1} = Rout(Price^t)$.

We will focus on the dynamics of the system defined by the distributed model. Note that at each stage, the system is fully described by its *state* (S, R). Let us define two relevant stability notions related to this model:

Definition 3

- *A state (S, R) is* **S-Stable** *by $\mathcal{A}_{Proc}$, if after n executions of the algorithm, the strategies are still stable that is $\forall n > 0 \quad \mathcal{A}_{Proc}^{(n)}(S) = S$, where* $\mathcal{A}_{Proc}^{(n)}(S) = \underbrace{\mathcal{A}_{Proc} o \dots o \mathcal{A}_{Proc}}_{n \quad times}(S, R) = \underbrace{A_{Proc}(A_{Proc}(...(\mathcal{A}_{Proc}}_{n \quad times}(S)) \dots).$
- *A state (S, R) is* **R-Stable** *by $\mathcal{A}_{Proc}$, if after n executions of the algorithm, routes are still stable. That is $\forall n > 0 \quad \mathcal{A}_{Proc}^{(n)}(S) = S^n$, and* $Rout(Random(S^n)) = Rout(Random(S))$.

In this paper, we are interested in a specific class of strategic processes $Proc$ where each player use a local strategic process $Proc_i$ based on learning techniques.

The learning algorithm:

Recall that in the proposed algorithm $\mathcal{A}_{Proc}$, each player updates his strategy according to a local strategic process $Proc_i$. That is a Player i should compute a new strategy probability vector S_i based on $Proc_i$ while considering only local information: $(S_i, Price_i, Benef_i)$. Such a process can follow discrete learning techniques similar to those presented in [Sastry et all. 2004] where the updating rule is given by :

$$Proc_i(s_{i,j}^t) = s_{i,j}^{t+1} = \begin{cases} s_{i,j}^t - b \cdot u_i^t \cdot s_{i,j}^t & \text{if } j \neq Price_i^t \\ s_{i,j}^t + b \cdot u_i^t \cdot \sum\limits_{k \neq Price_i^t} s_{i,k}^t & \text{otherwise} \end{cases}$$

Where:

- u_i^t : is the normalized utility such that: $u_i^t = \frac{Benef_i^t - A_i^t}{B_i^t - A_i^t}$.
  Let $B_i^t = max_{k \leq t} \quad Benef_i^k$ denote the maximal benefit of player i until stage t , and $A_i^t = min_{k \leq t} \quad Benef_i^k$ denote the minimal benefit of player i until stage t. Note that we can set $A_i^t = 0$. The normalization is necessary to ensure values between 0 and 1 and thus keep a valid probability vectors.
- The parameter $b \in [0, 1]$ is the step-size of the updating rule.

Moreover, each player i should start with a vector of strategies S_i^0 such that each component is non-null to give a chance to all prices to be picked.
Note that the learning rule applied by each player uses only local information and does not need players to be aware of their number or the topology linking them.
From Theorem 3.2 in [Sastry et all. 2004], the algorithm $\mathcal{A}_{Proc}$ where $Proc_i$ follows the discrete learning update rule, can only converge to a point that is a Nash equilibrium of the game. This result gives us the link between the distributed algorithmic and the game theoretical facets of the problem.

2.3 Combined convergence and stability analysis

We focus on the analysis of the distributed learning algorithm $\mathcal{A}_{Proc}$ where each local strategic process in $Proc$ follows the learning update rule. We will denote it $\mathcal{A}_{Proc\_learn}$. We will highlight some properties related to $\mathcal{A}_{Proc\_learn}$. The following theorem gives a characterization of stable points of the proposed algorithm.

Theorem 1 *Let (S, R) be a reachable state by $\mathcal{A}_{Proc\_learn}$ at stage t. Then*

$$(S, R) \text{ is S-Stable by } \mathcal{A}_{Proc\_learn}$$
$$\Updownarrow$$
For each player i one of the following propositions is satisfied:

(1) *S_i^t is a pure strategy with:*

$$S_{i,j}^t = 1 \text{ if } j = Price_i^t \text{ and 0 otherwise.}$$

(2) *Whatever the conjunction of prices with the strictly positive probabilities of other players, whatever the price that i can set, his benefit will be null:*

$$\forall j\ s.t\ s^t_{i,j} \neq 0\ then\ \forall t' > t,\ if\ Price^{t'}_i = j\ then\ Benef^{t'}_i = 0,$$

Proof. Suppose that (S, R) is S-Stable , by definition we have $\mathcal{A}_{Proc\_learn}(S) = S$.

That is for each given i, $s^t_{i,j} = \begin{cases} s^t_{i,j} - b \cdot u^t_i \cdot s^t_{i,j} & \text{if } j \neq Price^t_i \\ s^t_{i,j} + b \cdot u^t_i \cdot \sum\limits_{k \neq Price^t_i} s^t_{i,k} & \text{otherwise} \end{cases}$

For $j = Price^t_i$, we have $s^t_{i,j} = s^t_{i,j} + b \cdot u^t_i \cdot \sum\limits_{k \neq Price^t_i} s^t_{i,k}$ which imply that $u^t_i \cdot \sum\limits_{k \neq Price^t_i} s^t_{i,k} = 0$ then :

– Either $\sum\limits_{k \neq Price^t_i} s^t_{i,k} = 0$. Since $S^t_{i,k} \geq 0 \quad \forall k$ Then $S^t_{i,k} = 0 \quad \forall k \neq Price^t_i$. Indeed when a null sum is composed only by positive members this implies that all members are null.
  Moreover $S^t_{i,Price_i} = 1 - \sum\limits_{k \neq Price^t_i} s_{i,k}$ then $S^t_{i,Price_i} = 1$. Hence S_i is a pure strategy and the property (1) is satisfied.

– Otherwise $\sum\limits_{k \neq Price^t_i} s^t_{i,k} \neq 0$ and thus $u^t_i = 0$. Let us prove that $u^t_i = 0 \forall t' \geq t$. More generally, given $t' > t$, we have $S^{t'+1} = S^{t'} = S^t$. Hence by considering the update rule for t' we have:

$$S^{t'+1} = \mathcal{A}_{Proc\_learn}(S^{t'}) \Rightarrow u^{t'}_i \cdot \sum_{k \neq Price^{t'}_i} s^t_{i,k} = 0.$$

  At t', we have:

  • either the picked price is equal to the one at t ($Price^{t'}_i = Price^t_i$) and thus $\sum\limits_{k \neq Price^{t'}_i} s^t_{i,k} = \sum\limits_{k \neq Price^t_i} s^t_{i,k} \neq 0$
  • or it is different and thus in $\sum\limits_{k \neq Price^{t'}_i} s^t_{i,k}$ there is a member $s^t_{i,Price^t_i}$ which is non-null since at t, $Price^t_i$ was picked. The other coefficients are positive (probabilities), then the sum is non-null.

  In both cases, we have in one hand $\sum\limits_{k \neq Price^{t'}_i} s^t_{i,k} \neq 0$, and in the other hand, $u^{t'}_i \cdot \sum\limits_{k \neq Price^{t'}_i} s^t_{i,k} = 0$. We can conclude that $u^{t'}_i = 0$. That is the utility of player i is always null after t.
  Consider j such that $s^t_{i,j} \neq 0$, then there exists at least $t' > t$ such that $Price^{t'}_i = j$. This means that the strategy j will be picked at $t' > t$. Since we have concluded that $u^{t'}_i = 0$ thus $Benef^{t'}_i = 0$. The property (2) is then satisfied.

Let us now suppose that the vector S is such that for each i, S_i satisfies one of the properties (1) or (2) and show that the related state (S, R) is S-Stable. A player i such that S_i satisfies property (2) will always get a zero benefit. Given the updating rule, $u_i^{t'} = 0 \quad \forall t' \geq t$ and thus S_i will not change ($S_i^{t'} = S_i^t$).
A Player i with pure strategy will not change its strategy too since it will always play one action and in the updating rule the other probabilities are null. Hence, S would not change during stage t of $\mathcal{A}_{Proc\_learn}$ and thus (S, R) is S-Stable.

Hence the algorithm will only stop on a point such that there is only one winning route composed by providers that have a pure strategy. Probability vectors of remaining providers do not allow them to pick a winning strategy. This result does not ensure that the winning route is the dominant one. In other words, it does not ensure that the learning process was done correctly by providers in the dominant path. A more powerful result will be presented later.

Corollary 1 *Let (S, R) be a state reachable by $\mathcal{A}_{Proc\_learn}$ at stage t. If (S, R) is S-Stable then it is R-Stable.*

If (S, R) is S-Stable then there are players i such that S_i satisfies property (2) with a null benefit. This means that they are never chosen on a route and thus do not influence the routing choice. The other players have a pure strategy and since the routing choice made by $Rout(.)$ is deterministic, this leads to choose always the same route. Hence, (S, R) is R-Stable.
However, an R-stable state by $\mathcal{A}_{Proc\_learn}$ is not necessarily S-Stable. Indeed, consider the following example depicted in Figure 2.

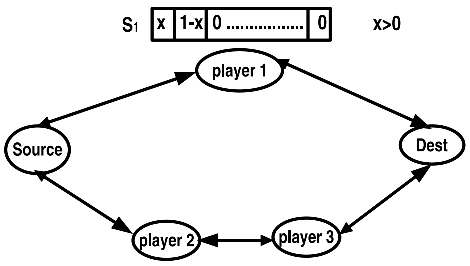

Figure 2: An example of a state which is R-stable but not S-Stable

Suppose that player 1 has the priority in case of identical announces to the source. As depicted in Figure 2 $S_{1,j} \neq 0$ only when $j = 1, 2$. In both cases, the route of player 1 will be chosen. This state is R-stable but not S-Stable since the strategy of player 1 is not pure and he has a non zero benefit. One can expect that in a finite number of stages such state will lead to an S-stable state where

players with pure strategies will choose strategies that corresponds to the maximal price with a non-null probability in the initial state. In our example, player 1 will choose to set price equal to 2.
In Theorem 1 we give a characterization of states that are S-Stable with the learning algorithm $\mathcal{A}_{Proc\_learn}$. We show also that the notion of S-Stable induces the notion of R-Stable for such algorithm. In other words, the algorithm $\mathcal{A}_{Proc\_learn}$, if it converges, will converge to a state which is S-Stable (and thus R-Stable) since then strategies and routes will not change. Such state will satisfies properties (1) and (2) in Theorem 1.
The following theorem concludes the link between the distributed algorithmic and the game theoretical analysis of our problem.

Theorem 2 *The only N.E. that can be discovered by the learning algorithm are pure or hidden pure N.E. Players on the winning path have pure strategies.*

Proof. The algorithm $\mathcal{A}_{Proc\_learn}$ converges only to a state which is S-Stable since then strategies and routes will not change. Given the characterization of S-stable states in Theorem 1, players on the chosen route have pure strategies. Hence, if the algorithm has discovered an N.E. , it is either pure or hidden pure.

If the learning rules are followed, and for b sufficiently small the algorithm will only converge to an N.E [Xing and Chandramouli 2004]. Thus, players on the dominant path are ensured to discover their optimal strategy even with incomplete information assumption. Further with simulations we will see that this outcome can be conserved even if the learning rule is not strictly followed by all players.

2.4 Discussion

The learning algorithm converges only to a point which is a Nash equilibrium, but the convergence is not always ensured. In our problem, we have tried to find simple situations where we can prove the convergence theoretically.

Let us consider the simple case of two players with two possible strategies $\{1, 2\}$ and priority to player 1. Suppose that the algorithm has not converged yet at time t and that the state at this time is $S = (S1, S2)$. Let us denote p_{ij} the probability of player i choosing price j.

Let us prove that we have $S1 = (p_{11}, p_{12}) \neq (0, 1)$. Suppose the contrary, that is player 1 always chooses strategy 2. Hence, player 2 will always win when choosing strategy 1 and always loose when choosing strategy 2. Thus p_{21} is increasing and p_{22} is decreasing leading to a pure strategy $S_2 = (1, 0)$ and thus to a state which is not an N.E. Hence S such that $S_1 = (0, 1)$ cannot be reachable state. We have then two cases:

- $S_1 = (1, 0)$ then player 1 will always win and player 2 can no more change his strategy. The algorithm has reached a stable state.
- $S_1 = (p_{11}, p_{12})$ where p_{11} and p_{12} are non-null. Player 2 never wins with strategy 2, but sometimes wins with strategy 1 as long as $p_{1,1} \neq 1$. Hence p_{22} is decreasing and p_{21} increasing and thus as the system evolves:
    - Either p_{11} reaches his limit first and the situation will be the same as the first case,
    - Or p_{22} and p_{21} reach their limit leading to a state $S_2 = (1, 0)$, enforcing player 1 to choose strategy 1. From this instant p_{11} is increasing and p_{12} decreasing. The algorithm is forced to converge regardless from the random selection.

The idea is that either player 1 will guess his best strategy first, or player 2 eliminates strategies that never let him win enforcing player 1 choice.

The same idea can be used when there are different prices $\{1, \ldots, n\}$, player 2 will eliminate one by one his strategies with the highest prices. When the probability of p_{2n} reaches 0, then p_{1n} becomes decreasing and so on. Either the player 1 discovers his optimal strategy first or he is forced to when player 2 eliminates his useless strategies. The idea is still valid when there are just direct providers between the source and the destination. The dominant provider will take the role of player 1 and the other providers will act as player 2.

For a general topology, we conjecture that the learning algorithm still converges to an N.E. (pure or hidden). Indeed, from simulations we observe that the algorithm always converges. Another issue about convergence is the choice of parameter b which is important to ensure that the obtained situation is an N.E. Indeed, theorem in [Sastry et all. 2004] states that convergence to an N.E. is ensured for a b sufficiently close to 0. A trade-off should be made in order to fix b: a high value of b can lead to a situation which is not an N.E.: players do not take enough time to learn their optimal strategies. A very low value of b may make the convergence slow. In our simulations $b = 0.1$ was sufficient to obtain convergence into N.E. situations.

We have implemented the distributed learning model, and simulate the behavior of the players in order to confirm the theoretical stability results and to observe convergence in the general case. Unilateral deviations from the learning rules are also investigated.

3 Simulation analysis

First, we need to introduce the asynchronous communication model. Indeed, in reality, nodes take their decisions locally and asynchronously. A communication model is needed in order to coordinate player decisions.

3.1 The asynchronous communication model

As the system evolves dynamically, possible events in a node can be: an update of price to customers, a choice of a provider or a switching from a path to another one. Each node informs his customers about relevant events that change his state. This is done using the traffic control messages.

When the source chooses an acceptable route, it sends an update message to inform its provider that his route was chosen. The provider in turn sends an update message to his own provider and so on until the destination. When the source switches on a new received route with a better price, an update message is sent iteratively to the nodes on the old route in order to inform them that their route is no more chosen. At the same time, an update message is sent iteratively to players on the new route to inform them that their route is now chosen. Asynchronously in time, players update their prices according to their strategy vector S which is updated depending in whether they were chosen on a route or not.

3.2 The simulation environment

A simulation of the proposed model was done using the OMNET [Varga 2001] simulator. The considered topology includes one source and one destination connected through several paths of different lengths. Neither queuing, nor scheduling delays were considered in the simulation.
For each player, the game is seen as series of stages where each stage is equal to $70ms$ not necessarily synchronous. At the end of each stage, each node updates his transit price according to the current strategy vector S which is updated by the discrete learning rule.

3.3 Evaluation of the learning algorithm

We consider three different scenarios considering topologies with disjoint and non disjoint paths. We also consider the case where the chosen route is announced by a priority or a non-priority player. We suppose that p_{max} is equal to 6 and the step-size of the updating rule b is equal to 0.1.

3.3.1 Scenario 1

We consider the network topology depicted in Figure 3. Priority is attributed to the player 1. We have then $c_{dominant} = \{Player1, Player2\}$ and $c_{second} = \{Player3, Player4, Player5, Player6\}$. Figure 4 displays the benefit obtained by each player. The curves suggest the convergence to an S-Stable state. Table 1 shows the probability vector obtained by each node (precision 10^{-4}).

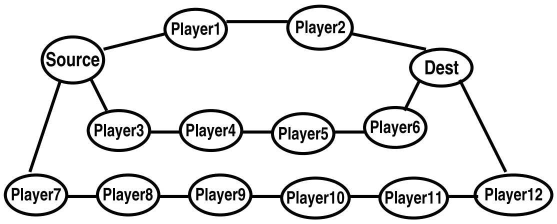

Figure 3: Network topology

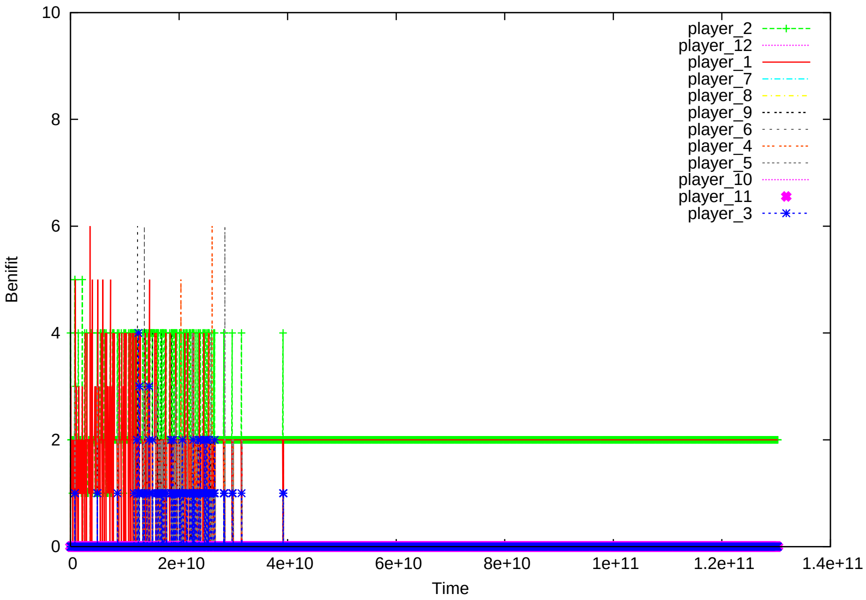

Figure 4: Benefits of players in scenario 1

The probability vectors of nodes in the dominant path converge to pure strategies when those of nodes in other paths converge to mixed strategies. Given Definition 2 the obtained situation is a hidden pure Nash equilibrium. Indeed, strategies of players in the second path (players $1, 2, 3$ and 4) are such that players in the dominant path have no incentive to deviate from their chosen strategies (for example the expected utility from announcing price 3 is 1.90 and for price 4 the expected utility is 0.84). Note that as expected from Theorem 1 since the dominant path is announced by the priority player, the route's price at the obtained N.E. is $|c_{second}| = 4$. From hidden pure N.E. we can deduce a pure N.E. where $S_1 = S_2 = 2$ and $S_3 = S_4 = S_5 = S_6 = 1$.

Prices	S_1	S_2	S_3	S_4	S_5	S_6	$S_7,\ldots,S_{12}$
1	0	0	0.9935	0.8237	0.6349	0.7011	0.40
2	1	1	0.0032	0.1595	0.3563	0.2867	0.30
3	0	0	0.0010	0.0055	0.0029	0.0040	0.10
4	0	0	0.0010	0.0055	0.0029	0.0040	0.10
5	0	0	0.0005	0.0027	0.0014	0.0020	0.05
6	0	0	0.0005	0.0027	0.0014	0.0020	0.05

Table 1: Strategy profile after stabilization in scenario 1.

3.3.2 Scenario 2

In this scenario, we consider the same topology as in scenario 1. However, the priority is attributed to the player 3. The dominant path is still the one composed by $Player1$ and $Player2$. c_{second} is still $\{Player3,\ldots,Player6\}$. Figure 5 shows the benefits of players. We notice that the system still converges but to a slightly

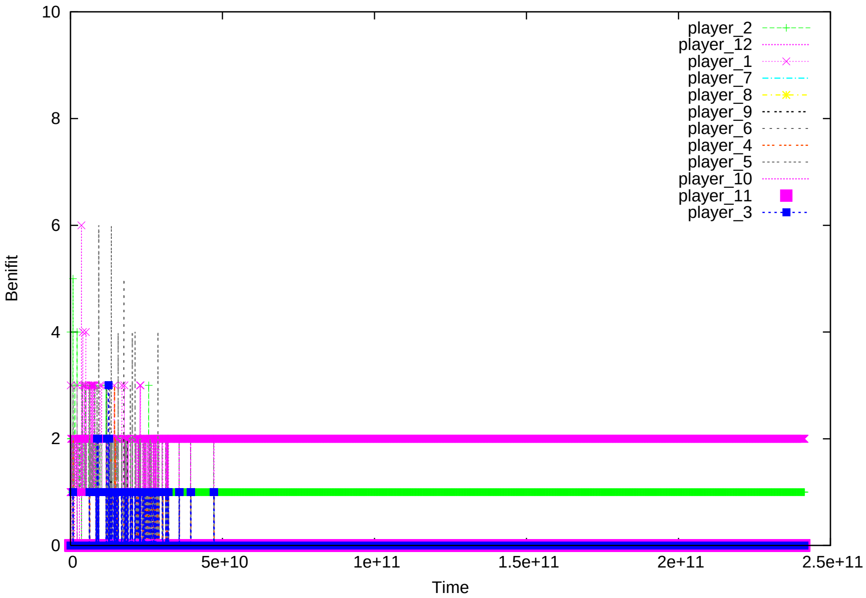

Figure 5: Benefits of players in scenario 2

different state. Table 2 gives the probability vector of each player (precision 10^{-4}).

Prices	S_1	S_2	S_3	S_4	S_5	S_6	$S_7, \ldots, S_{12}$
1	0	1	1	0.9923	1	1	0.40
2	1	0	0	0.0041	0	0	0.30
3	0	0	0	0.0018	0	0	0.10
4	0	0	0	0.0008	0	0	0.10
5	0	0	0	0.0004	0	0	0.05
6	0	0	0	0.0004	0	0	0.05

Table 2: Strategy profile after stabilization in scenario 2.

The obtained strategies form also a hidden N.E. Indeed, in one hand, players on the dominant path have pure strategies. In the other hand, players on second path have strategies such that those on the first one do not have incentive to deviate from their initial strategies (the expected utility from announcing a higher price is quasi null). Note that as expected from theorem 1 since the dominant path is announced by a non priority player, the route's price at the obtained N.E. is $|c_{second}| - 1 = 3$. From this hidden pure N.E. we can deduce a pure N.E. where $S_1 = 2, S_2 = 1$ and $S_3 = S_4 = S_5 = S_6 = 1$.

3.3.3 Scenario 3

We consider the network topology depicted in Figure 6 where *Source* is linked to *Dest* through non-disjoint paths. Priority is attributed to the player 1. The

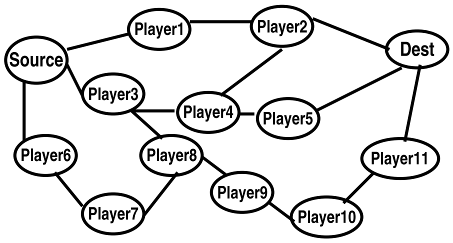

Figure 6: Network topology

algorithm still converges, and the probability vectors are presented by Table 3 (precision 10^{-6}). Figure 7 shows the players benefit.

The algorithm still converges and the obtained strategies form also a hidden N.E. Players on the chosen path have pure strategies and do not have incentive

Prices	S_1	S_2	S_3	S_4	S_5	$S_6, \ldots, S_{11}$
1	1	0	0.999998	0.999990	0.999996	0.40
2	0	1	0.000002	0.000010	0.000004	0.30
3	0	0	0.000000	0.000000	0.000000	0.10
4	0	0	0.000000	0.000000	0.000000	0.10
5	0	0	0.000000	0.000000	0.000000	0.05
6	0	0	0.000000	0.000000	0.000000	0.05

Table 3: Strategy profile after stabilization in scenario 3.

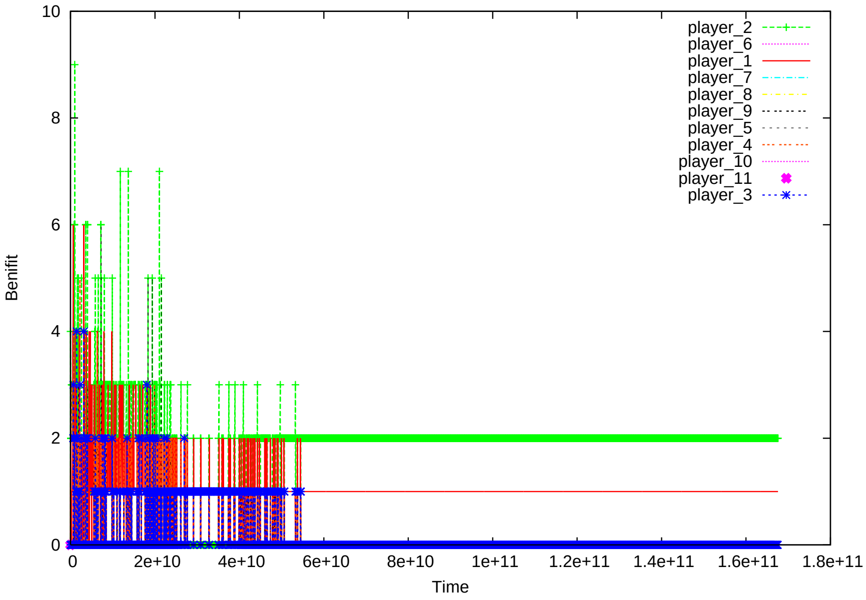

Figure 7: Benefits of players in scenario 3

to deviate from their initial strategies (the expected utility from announcing a higher price is quasi null). Note that as expected from Theorem 1 since the dominant path is announced by a non priority player, the route's price at the obtained N.E. is $|c_{second}| - 1 = 3$. From this hidden pure N.E. we can deduce a pure N.E. where $S_1 = 1, S_2 = 2$ and $S_3 = S_4 = S_5 = S_6 = 1$.

We have noticed that players in the dominant path always succeed to learn how to set their strategies appropriately in order to be chosen. One can argue that with some particular initial conditions the system can stabilize on another N.E. where the winning route is not the dominant one. In that case this means that players in the dominant path have eliminated their optimal strategies inappropriately. This can be resolved if after a certain number of executions, players that observe a null benefit reset their probability vector in order to give a second chance to their eliminated strategies. Hence the dominant path will have a chance to recover its status of privileged path.

3.4 Deviation from the learning rule: a different local strategic process

We investigate what could happen if a player tries unilaterally another local strategic process. We focus on players that initially were not on the chosen route and check whether their behavior affects the learning process of other players and thus the initial outcome. We reconsider the network topology depicted in Figure 3. In both scenarios the priority is attributed to player 1. We simulate the behavior of the system when player 6 decides to not follow the updating rule. He chooses for example a myopic strategy that consists in improving his price by one when his has a strict positive benefit and lowering by one otherwise. Figure 8 depicts benefit of players under the proposed scenario.

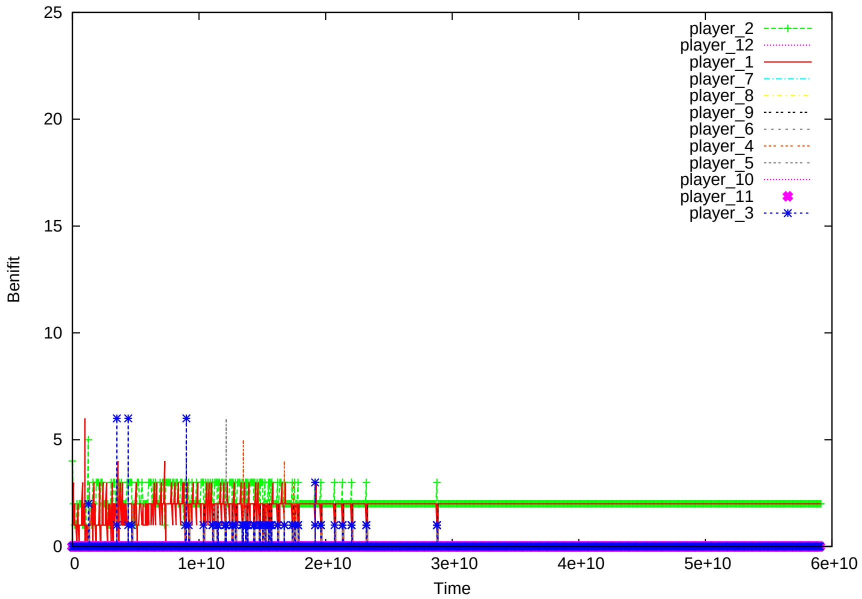

Figure 8: Players benefit when player 6 does not follow the learning rule

The algorithm still converges, and suggests that the learning process of players on the chosen path is not affected. The probability vectors are presented in Table 4 (precision 10^{-6}).

Prices	S_1	S_2	S_3	S_4	S_5	$S_7, \ldots, S_{12}$
1	0	0	0.961234	0.968599	0.968599	0.40
2	1	1	0.019383	0.015700	0.015700	0.30
3	0	0	0.006461	0.005233	0.005233	0.10
4	0	0	0.006461	0.005233	0.005233	0.10
5	0	0	0.003231	0.002617	0.002617	0.05
6	0	0	0.003231	0.002617	0.002617	0.05

Table 4: Strategy profile after stabilization in the scenario with deviation

The probability vectors of nodes in the dominant path converge to pure strategies as in the initial scenario where all players follow the rule. For similar scenarios, we have observed a similar behavior. This suggests that a player that is not supposed to be on the chosen route does not affect the learning process when he decides to not follow rules.

4 Conclusion

We have proposed a complete distributed solution to the transit price negotiation problem with incomplete information. For the one source one destination model, we have shown that the proposed algorithm enables players on the dominant path to fix their transit prices strategically in order to get the market. Simulations confirm that the system converges to the expected outcome. One issue is how to extend the model and the proposed solution to multiple sources. For example this can be the when customers are not linked to the same direct providers.

References

[Afergan 2006] Afergan, M.: "Using Repeated Games to Design Incentive-Based Routing Systems", 25th Conference on Computer Communications, IEEE INFOCOM, 2006.

[Barth et all 2007] Barth, D., Cohen, J., Echabbi, L., Hamlaoui ,C. : "Transit price negotiation: A combined game theoritic and distributed algorithmic approach", NetCOOP proceedings, In Lecture Notes On computer sciences, 4465, 266–275, 2007.

[Feigenbaum et all 2002] Feigenbaum, J., Papadimitriou, C., Sami, R., Shenker, S.: "A BGP-based Mechanism for Lowest-Cost Routing", Proceedings of the 2002 ACM Symposium on Principles of Distributed Computing., 2002.

[La and Anantharam 2002] La, R., Anantharam, V.: "Optimal Routing Control: Repeated Game Approach", IEEE Transactions on Automatic Control, 47,3,437–450, 2002.

[Maillé and Tuffin 2006] Maillé, P., Tuffin, B.: "How many parallel TCP session to open : A pricing perspective", ICQT Proccedings, In Lecture Notes On computer sciences, 4033, 2–12, 2006.

[Sastry et all. 2004] Sastry, P.S, Phansalkar, V., Thathachar, M.A.L: "Decentralized learning of Nash equilibria in multi-person stochastic games with incomplete information", IEEE Transactions on systems, man, and cybernetics, 24, 5, 2004.

[Shakkottai and Srikant 2006] , Shakkottai, S., Srikant,R.: "Economics of Network Pricing with multiples ISPs", IEEE/ACM Transactions on Networking, 14, 6, 2006.

[Varga 2001] Varga, A.: "The OMNeT++ Discrete Event Simulation System", In European Simulation Multiconference, 2001.

[Xing and Chandramouli 2004] Xing, Y., and Chandramouli, R., "Distributed discrete power control in wireless data networks using stochastic learning", Conference on Information sciences and systems CISS, 2004.