

Best Response Algorithms for Random Network Games

Stéphane Durand, Federica Garin, Bruno Gaujal

▶ **To cite this version:**

Stéphane Durand, Federica Garin, Bruno Gaujal. Best Response Algorithms for Random Network Games. [Research Report] RR-9066, Inria; Université Grenoble - Alpes; Gipsa-lab; Persival. 2017. hal-01522919

HAL Id: hal-01522919
https://inria.hal.science/hal-01522919

Submitted on 17 May 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Best Response Algorithms for Random Network Games

Stéphane Durand, Federica Garin, Bruno Gaujal

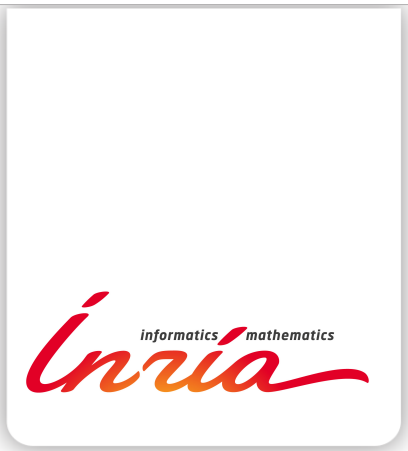
RESEARCH REPORT

N° 9066

June 2016

Project-Team Polaris, Necs

ISSN 0249-6399 ISRN INRIA/RR--9066--FR+ENG



Best Response Algorithms for Random Network Games

Stéphane Durand, Federica Garin, Bruno Gaujal

Project-Team Polaris, Necs

Research Report n° 9066 — June 2016 — 16 pages

Abstract:
In this paper we study distributed algorithms for computing a Nash Equilibrium in potential games. Our algorithms are based on best-response dynamics, with suitable revision sequences (orders of play). First, we show that random iid revision sequences have a convergence time within a constant factor from the optimal revision sequence, on average. Random iid revision sequences can be implemented in a distributed way, allowing us to propose a distributed algorithm, at minimal cost in terms of total execution time. Then we show how to take advantage of the structure of the interactions between players: Convergence time can be improved by letting non-interacting players play simultaneously. In a centralized setup, players can be grouped according to a coloring of the interaction graph so that the complexity scales with the chromatic number instead of the number of players. Finally this technique can also be exploited to design efficient distributed versions of the best response dynamics.

Key-words: Potential Games, Best Response Dynamics, Network Games, Distributed Algorithms

RESEARCH CENTRE
GRENOBLE – RHÔNE-ALPES

Inovallée
655 avenue de l'Europe Montbonnot
38334 Saint Ismier Cedex

Algorithme de meilleure réponse pour jeux aléatoire sur réseaux

Résumé : Dans ce rapport, nous étudions les algorithmes distribués pour calculer un équilibre de Nash dans des jeux de potentiel. Nos algorithmes s'appuient sur la dynamique de meilleure réponse, avec des séquences de révison (ordre de jeu) adéquates. Nous montrons d'abord que des séquences IID ont un temps de convergence moyen à une constante multiplicative de celui d'une séquence optimale. De plus, une séquence aléatoire peut être implémentée de manière distribuée nous permettant de proposer un algorithme distribué avec un coût en temps d'exécution global minimal. Ensuite nous montrons comment utiliser la structure des interactions entre joueurs pour, en permettant aux joueurs non associés d'agir simultanément, améliorer le temps d'exécution. Dans un contexte centralisé, les joueurs peuvent être groupés selon une coloration du graphe d'interaction permettant à la complexité d'être proportionnelle au nombre chromatique de ce graphe au lieu du nombre de joueurs. Enfin, cette structure peut aussi être exploitée dans le contexte distribué pour des algorithmes plus efficaces.

Mots-clés : Jeux de Potentiel, Dynamique de Meilleure Réponse, Algorithmes distribués, jeux sur réseaux

1 Introduction

Potential games have been introduced in [1] and have proven very useful ever since, especially in the context of routing games, first mentioned in [2]. They play a major role in transportation science as well as in computer science [3–5] and in distributed optimization [6].

It is well-known that the Best Response Algorithm (BRA) converges to a pure Nash equilibrium (NE) in potential games [7]. Here, we study the average running time of the algorithm and its dependence on the sequence of play of the players (called the *revision sequence* in the following). When one uses BRA to compute NE in potential games in practice, one is confronted with a mixed feeling.

On one hand, BRA with a round robin revision sequence has been proved optimal among all local search algorithms (converges faster than any local search in the strong stochastic sense), see [**?**].

On the other hand, BRA suffers from two main drawbacks when used in a distributed context. Firstly, the impact of the revision sequence on the performance is still unknown: If one chooses to replace the round robin revision by another order of play, the convergence time will grow since round robin is optimal, but the degradation should be evaluated. This may be critical in cases where round robin is hard to implement, as in a distributed setup. Secondly, the convergence of BRA requires that players play one at a time. Again, this may hamper performance in a distributed context because electing the next active player may require costly coordination between players.

In this paper we provide answers to both drawbacks. Concerning the order of play, we compute the average execution time of BRA under IID revision sequences (that can be seen as the opposite of round robin in terms of imposing an order between players) and we show that it remains within a constant factor (≈ 2) away from the round-robin complexity if we only consider the time to hit a Nash equilibrium instead of the total execution time.

In the second part of the paper, we consider the second weakness of BRA and consider *network games* where the interaction between the players is not total in the following sense: If the payoff function of one player (say player k) does not depend on the actions of another player (say i), we say that players k and i are *indifferent*. We show how to exploit these partial interactions between players to further improve the running time of the BRA algorithm. The basic idea is to let indifferent players play simultaneously since this will not jeopardize the convergence property while accelerating convergence times by acting on several players simultaneously.

We propose several distributed algorithms to implement such revision sequences. These algorithms are set in different contexts: For Algorithms 1 and 4, we assume that a central coordination between the players is possible, while Algorithm 2-3 deals with distributed players with no central coordinator. Algorithms 5 and 6 use a communication graph between the players that corresponds to the interaction graph of the game.

2 Distributed Best Response Algorithm

In this section, we consider the Best Response Algorithm for potential games, and we focus on its distributed implementation. To this aim, we study the performance of this algorithm when the revision sequence (the order of players) is not the optimal round-robin sequence, but a suitable random sequence, based on which we can propose a distributed algorithm.

This work has been partially supported by the LabEx PERSYVAL-Lab.

2.1 Best Response Algorithm and Potential games

We focus on games with a finite number of players and a finite number of strategies per player, also called matrix games.

A game $\mathfrak{G} \stackrel{\text{def}}{=} \mathfrak{G}(\mathcal{N}, \mathcal{A}, u)$ will be a triplet consisting of:

- A finite set of *players* $\mathcal{N} = \{1, \dots, n\}$;
- A finite set $\mathcal{A}_k$ of *actions* (or *pure strategies*) for each player $k \in \mathcal{N}$; The set of *actions profiles* or *states* of the game is $\mathcal{A} \stackrel{\text{def}}{=} \prod_k \mathcal{A}_k$;
- The players' *payoff functions* $u_k : \mathcal{A} \to \mathbb{R}$, for each $k \in \mathcal{N}$.

We define the classical *best response correspondence* $\mathcal{BR}_k(x)$ as the set of all actions that maximizes the payoff for player k under profile $\mathbf{x}$:

$$\mathcal{BR}_k(\mathbf{x}) \stackrel{\text{def}}{=} \left\{ \arg\max_{\alpha \in \mathcal{A}_k} u_k(\alpha; \mathbf{x}_{-k}) \right\}. \tag{1}$$

A *Nash equilibrium* (NE) is a fixed point of this correspondence, i.e., a profile $\mathbf{x}^*$ such that $x_k^* \in \mathcal{BR}_k(\mathbf{x}^*)$ for every player k.

Iteratively using best response may not converge in general, but here we consider potential games only, for which convergence is ensured.

Definition 1 (Potential games). *A game is an (exact)* potential game *[7] if it admits a function (called the potential) $F : \mathcal{A} \to \mathbb{R}$ such that for any player k and any* unilateral *deviation of k from action profile $\mathbf{x}$ to $\mathbf{x}'$*

$$u_k(\mathbf{x}) - u_k(\mathbf{x}') = F(\mathbf{x}) - F(\mathbf{x}'). \tag{2}$$

A game is a generalized ordinal potential game *[7] (or G-potential game for short) if there is $F : \mathcal{A} \to \mathbb{R}$ such that, for any player k and any* strictly profitable unilateral *deviation of k from action profile $\mathbf{x}$ to x', $F(\mathbf{x}') > F(\mathbf{x})$.*

A game is a best-response potential game *[9] (or BR-potential game for short) if there is $F : \mathcal{A} \to \mathbb{R}$ such that for any player k and action profile $\mathbf{x}$*

$$\mathcal{BR}_k(\mathbf{x}) = \left\{ \arg\max_{\alpha \in \mathcal{A}_k} F(\alpha, \mathbf{x}_{-k}) \right\}. \tag{3}$$

As shown in [9], BR-potential games are characterized by the fact that any sequence of profiles generated by unilateral best responses, and containing at least one strict improvement, is not a cycle. In particular, it can be seen that exact potential games are BR-potential games, but there exist G-potential games that are not BR-potential games. Yet, by imposing that the best response correspondence is univalued (our next assumption), it becomes true that any G-potential game is also a BR-potential game.

To avoid ties, we assume in the rest of the paper that the Best Response correspondence is a uniquely defined function: Here, $\mathcal{BR}_k(\mathbf{x})$ will always be reduced to a single action, for any player k:

$$\mathcal{BR}_k(\mathbf{x}) \stackrel{\text{def}}{=} \arg\max_{\beta_k \in \mathcal{A}_k} u_k(\beta_k; \mathbf{x}_{-k}). \tag{4}$$

This assumption holds when all profiles have different potentials and payoffs:

$$F(\mathbf{x}) \neq F(\mathbf{y}) \text{ and } u_k(\mathbf{x}) \neq u_k(\mathbf{y}), \tag{5}$$

whenever $\mathbf{x} \neq \mathbf{y}$, and for all k. Also note that, in general, breaking ties can always be done by ranking actions using an arbitrary fixed order.

When the Best Response is unique, we denote by $\Delta_k(x)$, the configuration obtained after player k has played its best response:

$$\Delta_k(\mathbf{x}) \stackrel{\text{def}}{=} (x_0, \ldots, \mathcal{BR}_k(\mathbf{x}), \ldots, x_{n-1}) = (\mathcal{BR}_k(\mathbf{x}), \mathbf{x}_{-k}). \tag{6}$$

We consider an algorithmic version of the Best Response Dynamics, called a *Best Response Algorithm* (BRA), where the next player is selected according to a *revision sequence* of players $(R_t)_{t\in\mathbb{N}}$. In general, we assume that this sequence of acting players is *fair*: each player appears infinitely often in the sequence.

Classical revision sequences include

- the *Round-Robin* sequence: $R_t = t \mod (n)$
- The random *IID* sequence: $\forall k \in \mathcal{N}, \mathbb{P}(R_t = k) = \rho_k$, for a fixed ρ_k, with $\rho_0 + \cdots + \rho_{n-1} = 1$.

In the latter case, if we assume that the probability of choosing any player k is strictly positive ($\forall k \in \mathcal{N}, \rho_k > 0$), then the revision sequence is fair almost surely.

Algorithm 1: Best Response Algorithm (BRA)

```
Input: Game utilities (u_k(·)); Initial state (x := x(0)); Fair revision sequence R;
Initialize t := 0; List of satisfied players L := ∅;
repeat
    Pick next player k := R_t; t := t + 1;
    if x_k ∉ BR_k(x) then
        Update strategy for player k to any x_k ∈ BR_k(x);
        L := ∅;
    L := L ∪ {k};
until size(L) = n;
```

Theorem 1 (BRA converges to NE [7])**.** *For any potential game $\mathfrak{G}$ and any weakly fair revision sequence R, Algorithm 1 converges in finite time almost surely, to a Nash Equilibrium of $\mathfrak{G}$.*

Proof. This is a well known result, we only provide a sketch of the proof. Along the iterations, the potential increases and thus converges almost surely because it can only take a finite number of values. After convergence of the profile, the algorithm stops as soon as all players have had a chance to play according to R. Using the assumption on R, this occurs after a finite number of iterations, almost surely. the final state α is a NE by definition. □

2.2 Speed of convergence

In this section, we analyze the time complexity of BRA. We will denote by T_{BR} to be the number of steps before convergence of Algorithm BRA.

The quantity T_{BR} depends on the game over which it is run, on the initial state x_0 and on the infinite sequence of revision players R.

It should also be clear that it depends of the game only through the potential F, so we sometimes write $T_{BR}(F, x_0, R)$ or $T_{BR}(\mathfrak{G}, x_0, R)$ the number of steps before convergence of Algorithm BRA for a game $\mathfrak{G}$ with potential F, starting in state x_0, under the condition that the sequence of players is R.

It should be clear that $T_{BR}(\mathfrak{G}, x_0, R)$ is unbounded because the revision sequence R can have a bad behavior (one player might appear too few times for convergence in any bounded time). Even when this is not the case, for example when R is a round-robin sequence, the time for convergence exhibits a large gap when worst case and average cases are compared, as shown in the following proposition from [10].

Proposition 1 ([10]). *In the worse case,* $T_{BR}(F, x_0,$ *round-robin* $) = \Theta(nA^{n-1})$. *The average complexity is* $\mathbb{E}T_{BR}(F, x_0,$ *round-robin* $) = e^{\gamma}n + o(n)$.

In the proposition above, the expectation is taken over all possible potentials F and all initial states x_0. We can also recall the fact that the Round-robin revision is asymptotically optimal among all fair revisions:

Proposition 2 ([10]). *Let R be an arbitrary revision sequence, and let $t \in \mathbb{N}$.*

$$\mathbb{P}(T_{BR}(F, x_0,\ \textit{round-robin}\ ) > t) \leqslant \mathbb{P}(T_{BR}(F, x_0, R) > t).$$

As mentioned in the introduction, using a round robin revision sequence for the players implies the presence of a central coordination of the players (for example the presence of a *token ring*). When such coordination is not available, round robin revision sequences are hard to implement. A more distributed method would be to equip each player with a Poisson clock (with parameter λ_k) and let players play whenever their Poisson clock rings. This would provide with a *IID revision sequence*: The next player to play is player i with probability $p_i \propto \lambda_i$.

In the uniform case ($\forall k \geqslant n, \lambda_k = 1$), the uniform IID revision sequence says that the next player to play is player i with probability $1/n$. In the case of a uniform IID revision sequence, the time of convergence to a NE of BRA can be lower and upper bounded:

Theorem 2. *The average complexity of BRA under a uniform IID revision satisfies:*

$$n\log n + \gamma n + o(n) \leqslant \mathbb{E}T_{BR}(F, x_0,\ \textit{IID}\ ) \leqslant 2e^{\gamma} n\log n + O(n),$$

where γ is the Euler constant ($\gamma \approx 0.58, 2e^{\gamma} \approx 3.56$).

Proof. The proof is inspired from the proof of the round robin case in [10], although computations are more involved here because of the additional randomness due to the revision sequence.

Let us first recall the fact that for any revision sequence R, the number of iteration of BRA under R is asymptotically equal to the number of iteration of another version of BRA (called BRA-IFA in the following). BRA-IFA does not correspond to the behavior of an actual game because the potentials of the configurations change over time: Under BRA-IFA, the potential of one state is re-drawn from a uniform distribution over $[0, 1]$ at every visit of the state (see [10]).

The difference between BRA and BRA-IFA is the fact that due to the re-generation of potentials at every visit, the behavior of BRA-IFA under a uniform IID revision is Markovian with a state space (z, k) where z is the potential of the current state and k is the current number of players satisfied by the current state with potential z.

Let Y_t be the potential at step t ($Y_t \in [0, 1]$) and K_t be the current number of consecutive players whose best response did not change the current potential ($K_t \in \{1, 2, \ldots, n\}$) (number of satisfied players). The couple (Y_t, K_t) is a discrete-time, continuous-space Markov chain whose kernel is:

$$\mathbb{P}\Big((Y_{t+1}, K_{t+1}) = (y, k)\Big|(Y_t, K_t) = (y, k)\Big) = k/n,$$

$$\mathbb{P}\Big((Y_{t+1}, K_{t+1}) = (y, k+1)\Big|(Y_t, K_t) = (y, k)\Big) = y^a(n-k)/n,$$

and, if $u > y$,

$$\mathbb{P}\Big((Y_{t+1}, K_{t+1}) \in ([u,1],1)\Big|(Y_t, K_t) = (y,k)\Big) = (1-u^a)(n-k)/n.$$

All the other transitions have a null probability.

Let $s(y,k)$ be the number of iterations of BRA-IFA before convergence when the current state of the Markov chain is equal to (y,k). With probability k/n, the next player is already satisfied so that nothing happens: $s(y,k) = s(y,k)+1$.
With probability $y^a(n-k)/n$, the next player is new but does not change its choice so that $s(y,k) = s(y,k+1)+1$.
With probability density au^{a-1} the next player is new and finds a new best response with potential u so that $s(y,k) = 1 + s(u,1)$.
Let $S(y,k) = \mathbb{E}[s(y,k)]$. The previous one step analysis of $s(y,k)$ makes $S(y,k)$ satisfy a forward Poisson equation:

$$S(y,k) = 1 + \frac{k}{n}S(y,k) + \frac{n-k}{n}y^a S(y,k+1) + \frac{n-k}{n}\int_y^1 au^{a-1}S(u,1)\mathrm{d}u.$$

By definition, the boundary conditions are: $\forall y, S(y,n) = 0$ (the current state is NE, with potential y when all players agree on this).

By setting $z \stackrel{\text{def}}{=} y^a$, and defining $\sigma(z,k) \stackrel{\text{def}}{=} S(z^{1/a},k)$ and $B(z) \stackrel{\text{def}}{=} \int_z^1 \sigma(v,1)\mathrm{d}v$, The previous recurrence equation becomes

$$\sigma(z,k) = z\sigma(z,k+1) + B(z) + \frac{n}{n-k}.$$

Successive substitution of $\sigma(z,2),\ldots,\sigma(z,n-1)$ in the equality for $k=1$ yields $\sigma(z,1) = B(z)H(z) + V(z)$ where $H(z) \stackrel{\text{def}}{=} 1 + z + \cdots + z^{n-2}$ and $V(z) \stackrel{\text{def}}{=} \sum_{i=0}^{n-2} nz^i/(n-i-1)$.

Since $\sigma(z,1) = -\dot{B}(z)$, this is a differential equation in $B(z)$ of the form $\dot{f} + gf = c$. Using the boundary condition $B(1) = 0$, its generic solution is

$$B(z) = e^{-q(z)}\int_z^1 V(u)e^{q(u)}\mathrm{d}u. \tag{7}$$

where

$$q(y) \stackrel{\text{def}}{=} \int_0^y H(u)\mathrm{d}u. \tag{8}$$

The average number of iterations in the execution of the algorithm starting from an arbitrary profile is $\mathbb{E}[T_{BR}(F,x_0,\ \text{IID}\ )] = \int_0^1 \sigma(z,1)\mathrm{d}z$. Since $\sigma(z,1)$ is decreasing in z, and by using the definition of BRA-IFA,

$$\sigma(0,1) - 1 \leqslant \mathbb{E}[T_{BR}(F,x_0,\ \text{IID}\ )] \leqslant \sigma(0,1).$$

Using $\sigma(0,1) = -\dot{B}(0) = B(0)H(0) + V(0)$, $H(0) = 1$, $V(0) = \frac{n}{n-1}$ and $q(0) = 0$, we get

$$\sigma(0,1) = \frac{n}{n-1} + \int_0^1 V(u)\exp\left(\sum_{i=1}^{n-1}\frac{u^i}{i}\right)\mathrm{d}u\,. \tag{9}$$

This gives the upper bound, as follows (we will use the notation h_n for harmonic numbers):

$$\begin{aligned}
\sigma(0,1) &\leqslant \frac{n}{n-1} + \int_0^1 V(u) \exp\left(\sum_{i=1}^{n-1} \frac{1}{i}\right) \mathrm{d}u \\
&= \frac{n}{n-1} + e^\gamma n \left(\int_0^1 V(u) du\right) (1 + O(1/n)) \\
&= \frac{n}{n-1} + e^\gamma n^2 \sum_{i=0}^{n-2} \frac{1}{(i+1)(n-i-1)} (1 + O(1/n)) \\
&= \frac{n}{n-1} + e^\gamma n^2 \frac{2h_n}{n} (1 + O(1/n)) \\
&= 2e^\gamma n \log(n) + 2\gamma e^\gamma n + O(\log n).
\end{aligned}$$

A numerical computation of (9) (using Maple with a very large precision) yields

$$\sigma(0,1) \approx n\log(n) + \gamma n + C_1 n. \tag{10}$$

with $C_1 \approx 1.22$.

As for the lower bound, it comes from the following simple observation. The time complexity of BRA under a uniform IID revision sequence can be decomposed into two: 1) time to reach a NE and 2) once a NE is reached, the time for the list L of satisfied players to fill up entirely. Under a uniform IID revision sequence, this second part corresponds to the time it takes before each player has played at least once.

This is exactly the coupon collector problem [11]. Therefore, the expected time to get n players to play once is nh_n. In turn, $\sigma(0,1) \geqslant nh_n = n\log(n) + \gamma n + o(n)$.

□

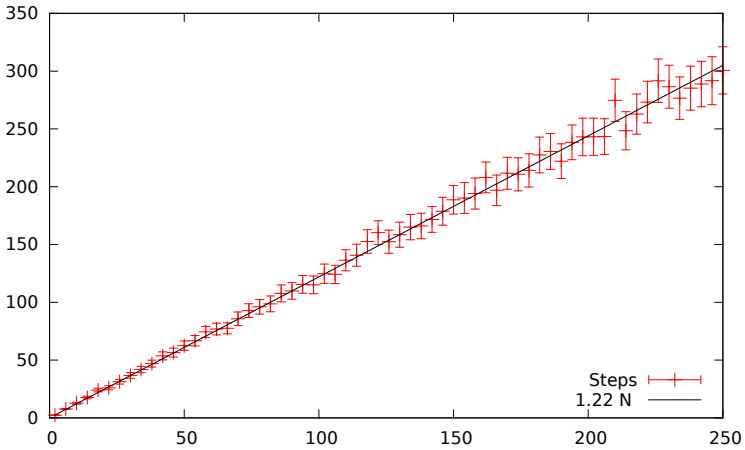


Figure 1: Number of steps of BRA with a uniform IID revision sequences, before reaching a NE in randomly generated game, with 95% confidence intervals.

Remark 1. *Considering the numerically estimated value of* $\sigma(0,1) \approx n\log n + \gamma n + 1.22n$ *implies that the time to reach an equilibrium is around* $\sigma(0,1) - nh_n - \gamma n \approx 1.22n$. *This is*

close to the time to reach an equilibrium using the round robin revision ($\approx 0.78n$ steps, see [10]). While round-robin is a fully centralized revision, the IID revision that can be implemented in a distributed way by letting all players play according to independent Poisson clocks, all with the same rate, as mentioned before. The price to pay to use this distributed revision in terms of convergence time is moderate (a constant factor of order ≈ 1.56).

Executions of BRA under uniform IID revisions, over a large set of potential games with uniformly generated potentials (1,000), show that the convergence time distribution is actually well concentrated around the average value, as displayed in Figure 1.

For games with a range from 2 to 250 players and 10 actions each, the 95% confidence intervals seem to grow linearly with the number of players. This is in accordance with the fact that the second moments grow linearly with n, at least in the RR case (see [**?**]). It should be similar for IID revision sequences, although no explicit computation of the second moment has been done in this case.

2.3 Distributed Algorithm

As mentioned above, a IID revision sequence can be implemented in a distributed way by using Poisson clocks. However, the problem of detecting convergence remains because it involves a centralized computation of the list L of satisfied players.

An alternative is to set up a global communication between the players: Upon reception of a test message (sent by one player, says player 1, according to a sub-process of its Poisson clock) all players test if they are currently playing their best response and send back this information to the message sender who decides to send back a global termination message if all players are currently satisfied.

Here is the algorithm run by player 0 and players $1, \dots, n-1$, respectively:

Algorithm 2: Distributed version of BRA under IID revisions, for the master player 0.

Input: Game utilities $(u_k(\cdot))$; Initial state $(\mathbf{x} := \mathbf{x}(0))$;
Local clock, ticking w.r.t. a Poisson process with rate 1;
repeat
  At each tick of the Poisson clock;
  if $x_0 \notin \mathcal{BR}_0(\mathbf{x})$ **then**
    Update strategy for player 0 to $x_0 \in \mathcal{BR}_0(\mathbf{x})$;
  With probability p
    Interrupt-Clock;
    Send(Termination-Test);
    Collect answers from all players;
    Compute All-Players-Satisfied;
    if All-Players-Satisfied **Send**(End);
    Restart-Clock;
until *All-Players-Satisfied*;

Algorithm 3: Distributed version of BRA under IID revisions, for players k from 1 to $n-1$.

Input: Game utilities $(u_k(\cdot))$; Initial state $(\mathbf{x} := \mathbf{x}(0))$;
Local clock, ticking w.r.t. a Poisson process with rate 1;
repeat
  At each tick of the Poisson clock; **if** $x_k \notin \mathcal{BR}_k(x)$ **then**
    Update strategy for player k to $x_k \in \mathcal{BR}_k(\mathbf{x})$;
  Upon Reception of Termination-Test;
    Interrupt-Clock;
    if $x_k \in \mathcal{BR}_k(x)$ **Send**(Satisfied) **else Send**(Not-Satisfied);
    Restart-Clock;
until *Reception of* End *message*;

The synchronization procedure between player 0 and the other players has a time cost (denoted by $\sigma(n)$ in the following) that depends on the communication structure between players.

Let T_{dist} be the execution time of this distributed algorithm when the Poisson clocks of all players are independent. This time is the same as the time for player 1 to get the variable All-Players-Satisfied true.

This time satisfies $T_{\text{dist}} = Q_1 + \cdots + Q_m + m\sigma(n)$, where Q_i's are the times elapsed between two consecutive messages, the random number m is the number of messages sent before a Nash equilibrium is reached, and $\sigma(n)$ is the time to send and receive one message. By construction of the algorithm, the random variables Q_i's are iid, exponentially distributed with parameter p. Notice that m is independent of the value of Q_i and therefore Wald formula can be used to compute the expectation of T_{dist}:

$$\mathbb{E}[T_{\text{dist}}] = \mathbb{E}[m]\mathbb{E}[Q_1] + \mathbb{E}[m]\sigma(n). \tag{11}$$

On the other hand, the memory-less property of the exponential distribution implies that $T_{\text{dist}} = T_{\text{Nash}} + Q + m\sigma(n)$, where T_{Nash} is the time to reach a Nash equilibrium and Q is exponentially distributed with parameter p. This gives

$$\mathbb{E}[T_{\text{dist}}] = \mathbb{E}[T_{\text{Nash}}] + \mathbb{E}[Q] + \mathbb{E}[m]\sigma(n). \tag{12}$$

By subtracting (12) from (11), and using $\mathbb{E}[Q] = \mathbb{E}[Q_1] = 1/p$, one gets $\mathbb{E}[m] = p\,\mathbb{E}[T_{\text{Nash}}] + 1$ so that

$$\mathbb{E}[T_{\text{dist}}] = \mathbb{E}[T_{\text{Nash}}] + \sigma(n) + p\,\mathbb{E}[T_{\text{Nash}}]\sigma(n) + 1/p.$$

At this point, one can tune the free parameter p. Let $p^* \stackrel{\text{def}}{=} 1/\sqrt{\mathbb{E}[T_{\text{Nash}}]\sigma(n)}$ be the value of p that minimizes the term $p\mathbb{E}[T_{\text{Nash}}]\sigma(n) + 1/p$. One gets

$$\mathbb{E}[T_{dist}] = \mathbb{E}[T_{Nash}] + \sigma(n) + 2\sqrt{\mathbb{E}[T_{\text{Nash}}]\sigma(n)}$$

Using Theorem 2, or more precisely, Equation (10), one can replace $\mathbb{E}[T_{\text{Nash}}]$ by $C_1 n/n = C_1$. Using a classical model for a global synchronization on a distributed algorithm (see [12]), one can also replace $\sigma(n)$ by $C_2 \log(n)$ if a reduction of the answers from all players is possible.

Finally, one gets

$$\mathbb{E}[T_{\text{dist}}] = C_1 + C_2 \log n + 2\sqrt{C_1 C_2}\log^{1/2} n.$$

This complexity says that the distributed algorithm has an asymptotic average speed-up w.r.t. the sequential version given in Algorithm 1 equal to

$$\frac{\mathbb{E}[T_{BR}]}{\mathbb{E}[T_{\text{dist}}]} = \frac{n}{C_2}. \tag{13}$$

Although this suggests that Algorithms 2-3 make a perfect parallelization of the algorithm BRA, this computation is only valid under an ideal PRAM model, *i.e.* when the actions of each player are *atomic*, that is take no time, and are immediately available for all the other players.

3 Network Games

In this section, we consider that players may not all interact with each other, and we want to take advantage of this. Recalling the definition of $\Delta_k(\mathbf{x})$ in (6), we give the following definition of indifferent players.

Definition 2 (Indifferent players, Neighbors, Interaction Graph)**.** *Player i is* indifferent *to player j if for all state x,*

$$\Delta_i(\Delta_j(\mathbf{x})) = \Delta_j(\Delta_i(\mathbf{x})). \tag{14}$$

Otherwise, we say that i and j are neighbors*.*
The interaction graph $\mathcal{W}$ *is the undirected graph linking neighbors.*

Condition (13) is satisfied in particular when the payoff function for player j, $u_j(x)$, does not depend on x_i, which is the definition used in [13] to define network games.

Lemma 1. *In exact potential games, if $u_j(x)$ does not depend on x_i then $u_j(x)$ can be chosen independent of x_j.*

Proof. The proof follows from the definition of potential games. Using the fact that for all x_i, x_i', x_j, x_j', $u_j(\cdots x_j \cdots x_i \cdots) - u_j(\cdots x_j \cdots x_i' \cdots) = 0$ and
$u_j(\cdots x_j' \cdots x_i \cdots) - u_j(\cdots x_j \cdots x_i' \cdots) = 0$, together with the exact potential implies

$$u_i(\cdots x_j \cdots x_i \cdots) - u_i(\cdots x_j' \cdots x_i \cdots) = u_i(\cdots x_j \cdots x_i' \cdots) - u_i(\cdots x_j' \cdots x_i' \cdots).$$

Since the game with payoff u_i is equivalent to the game with payoff $u_i + C$ where C is a constant, the previous equality shows that u_i can be chosen independent of its jth coordinate.

□

A more general sufficient condition for (13) is when for all actions α, β of player i, $\text{sign}(u_i(\alpha, \mathbf{x}_{-i}) - u_i(\alpha, \mathbf{x}_{-i}))$ does not depend on x_j and for all actions γ, μ of player j, $\text{sign}(u_j(\gamma, \mathbf{x}_{-j}) - u_j(\mu, \mathbf{x}_{-j}))$ does not depend on x_i. This condition on the sign generalizes the definition of neighbors given in [13], that only considers the case where $u_j(\mathbf{x})$ does not depend on x_i.

When several players (say i and j) play simultaneously, the corresponding *simultaneous* best response operator is

$$\mathcal{BR}_{\{i,j\}}(\mathbf{x}) \stackrel{\text{def}}{=} \underset{(\alpha_i,\beta_j)\in\mathcal{A}_i\times\mathcal{A}_j}{\arg\max} F(\alpha_i, \beta_j; \mathbf{x}_{-i-j}).$$

The corresponding state is

$$\Delta_{\{i,j\}}(\mathbf{x}) \stackrel{\text{def}}{=} (x_0 \dots \alpha_i \dots \beta_j \dots x_{n-1}),$$

where α_i and β_j are the argmax in the previous equation.

Lemma 2. *For any two indifferent players i and j, $\Delta_i(\Delta_j(\mathbf{x})) = \Delta_j(\Delta_i(\mathbf{x})) = \Delta_{\{i,j\}}(\mathbf{x})$.*

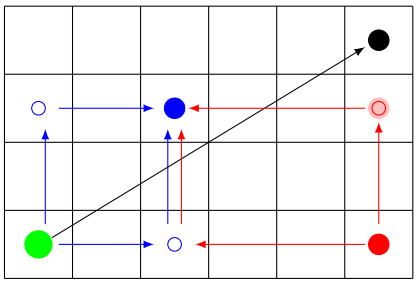


Figure 2: Illustration of the proof of Lemma 2

Proof. Let us consider the potential matrix for two independent players as illustrated by Figure 2. Player 1 acts on the first coordinate (lines), and player 2 chooses the second coordinate (columns). By definition of indifference, starting from any state and letting Player 1 play before Player 2 or Player 2 play before Player 1 leads to the same state.

Let us suppose that the solid blue state in the figure (best-response profile after 1 and 2 have played) does not have the global optimum potential (ie what a two-player best response should be), in solid black in the figure. Now, if the black state and the blue state have distinct second coordinates, let us start the game from a state with first coordinate and second coordinate in common with the blue and black states respectively (solid red state in the figure). If we let player 2 act first, followed by the player 1, we should end again in the blue state. By indifference, the same (blue) state is reached when player 1 plays before the player 2. The intermediary state (pink point) is thus the best response of the player 1. It has a better potential than any state on the same column, including the black state. This implies that the black state cannot be the global optimal.

□

The lemma says that when two players commute, making one play after the other leads to the state with the largest potential among all those currently accessible to those two players, and can be seen as a grouped best response.

Now, let us consider a modified game where a set of two (or more) indifferent players is replaced by a single *super-player* (ij) with A^2 possible actions whose best response function. This concept of super-player is used to design Algorithm 4.

3.1 Fast Parallel Algorithm

Lemma 2 says that when two indifferent players play simultaneously, convergence of BRA is not jeopardized and the game behaves as if both players form a "super-player" with a larger action set. In a game, we can use maximal stable sets in the interaction graph to reduce the time to convergence.
This gives the following version of BRA where indifferent players are grouped as super-players.

Algorithm 4: Maximal BR Algorithm (Maximal Independent Best Response Algorithm (M-BRA))

```
Input:
Game utilities (u_k(·)); Initial state (x := x(0));
Construct a coloring of W;
Generate an infinite sequence R = (R_t)_{t∈N} of colors;
repeat
    Pick next color c in R
    foreach player in c simultaneously do
        chooses the action that maximizes its payoff
until convergence;
```

The cost of each update is shared between the members of each group. Thus each step costs the same time as in the original version of BRA.

The number of steps before finding a Nash equilibrium do not depend on the number of choice per player and is proportional to the number of players, therefore the average time to reach a Nash Equilibrium with the new algorithm is proportional in the number of super-players, equal to χ, the chromatic number of the interaction graph.

Proposition 3. *Let $\mathfrak{G}$ be any potential game whose interaction graph is $\mathcal{W}$. Let $\chi(\mathcal{W})$ be the chromatic number of $\mathcal{W}$, then the average complexity, starting from state x_0, of reaching a NE for Algorithm 4 using a round-robin (resp. a uniform IID) revision sequence is*

$$\mathbb{E}[T_{M-BRA}(\mathfrak{G}, x_0,\ RR\ )] = e^{\gamma}\chi(\mathcal{W})$$

$$\textit{resp. } \mathbb{E}[T_{M-BRA}(\mathfrak{G}, x_0,\ IID\ )] = C_1\chi(\mathcal{W}),$$

where $C_1 \approx 1.22$.

Remark 2. *Actually Algorithm 4 can be further improved if each color is extended into maximal stable sets. In this case a player may have several colors so that it will play more often. This marginally improves the speed of convergence without changing the asymptotiv behavior in $C\chi(\mathcal{W})$.*

3.2 Distributed Version

The previous parallel algorithm is fast, but needs a centralized first step to compute the coloring of the graph and most importantly it requires a coordination between super-players. For games played in distributed contexts such as routing games (see Section 3.3), one needs to design a distributed version of Algorithm 4.

The main assumption used in this section is that players know who are their neighbors and can communicate with them. In other words, the communication network between players coincide

with their interaction graph. This assumption can be justified by the fact that both dependence and communication are a sign of "proximity". In our case study on routing games (see Section 3.3), neighbors can actually communicate through packets using common nodes.

Algorithm 5 is a distributed version of Algorithm 4. It only requires communications between neighbors. The relative rates of the clocks for each player can be tuned according to the interaction graph (see §3.3 for examples) Finding the optimal activation rate for a given interaction graph is an interesting open question that is addressed numericall in a simple case in Section 3.3.

Algorithm 5: Improved Distributed BR Algorithm (ID-BRA)

```
1  Input:
2  Game utilities (u_k(·)); Initial state (x := x(0));
3  Variables :
4  One local Poisson clock per player
5  foreach player k do
6      Boolean table L indexed by the neighbors (initialized by 0).
7      foreach received message m do
8          if(m = lock i) then L[i] ← 1
9           else if(m = unlock i) then L[i] ← 0
10     foreach Poisson tick do
11         if ∀i, L[i] = 0 then
12             Send lock k to all neighbors
13             Choose the action that maximizes payoff: x_k := BR_k(x)
14             Send unlock k to all neighbors
```

3.2.1 Improved Version

Although it is highly distributed, Algorithm 5 has one drawback: the probability to play for one player at any given instant only depends on the interaction graph and is memoryless. Depending on the game, the expected time before the next activation of a specific player can be huge. In Algorithm 6, we increment a counter that counts the number of Poisson clock activations without playing. Here, the rule for choosing who plays within the active players is the one with the highest counter. The probability to act for a given player is lower than in Algorithm 5, especially in a dense graph, but the maximum time between two activations should be lower. This new algorithm prevents players that cannot act easily (those with a high degree) from waiting too long. We believe it performs better than Algorithm 5 for very heterogeneous games (see §3.3).

3.3 Case Study: Routing in a Network with Cross Traffic

One of the main applications of potential games are routing games. Let (V, E) be a communication network over a set V of nodes and a set E of bi-directional communication links, over which we consider the following *multi-commodity flow problem*. A set of *flows* of packets (players) must be routed over the network. Each player is characterized by a source-node, S_k, a destination-node D_k and a nominal arrival rate of packets, λ_k. Also, each flow is assigned a set of paths in the network from its source to its destination. A *configuration* is a choice of one path per flow. For each player, the payoff is the delay on its chosen route. This game is an *atomic non-splittable*

Algorithm 6: Priority BR Algorithm (P-BRA)

Input:
Game utilities $(u_k(\cdot))$; Initial state $(\mathbf{x} := \mathbf{x}(0))$;
Variables :
One ticket $(t_k)_{k\in\mathsf{Players}}$ $(\forall k \in \mathsf{player}(t_k) = 0)$
One local Poisson clock per player
repeat
  foreach *player* k **do**
    foreach *received message* **do**
      send back t_k
    foreach *Poisson tick* **do**
      send **()** to each neighbor
      if t_k *higher than all values received from neighbors* **then**
        choose the action that maximizes payoff: $\mathbf{x}_k := \mathcal{BR}_k(\mathbf{x})$
        $t_k \leftarrow 0$
      else
        $t_k \leftarrow t_k + 1$
until *Convergence*;

routing game. It is a potential game if the delay on each link only depends on the number of players using it.

This problem fits naturally in our distributed framework, where neighbors in the interaction graph are also able to communicate. Indeed, here, two players are indifferent whenever their routes do not share any common node. Since they share a common node, two neighbors can communicate with each other, by using modified packet headers containing messages and post messages for their neighbors in shared nodes.

To test the efficiency of Algorithms 5 and 6, we used the network routing game displayed in Figure 3. This network has $U + 1$ players, each having two paths. Player 0 can be seen as

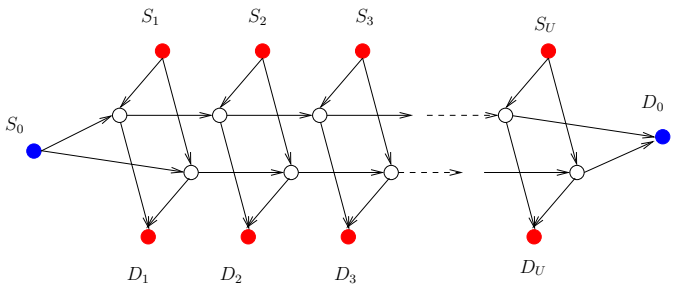


Figure 3: A routing network with cross traffic.

the 'main' flow (blue), from left to right, confronted with cross traffic from all the other players (from 1 to U). Players 1 to U are all indifferent to each other (they do not share any node), while player 0 is a neighbor of them all. In this case, the interaction graph is a star, centered in 0.

We used Algorithms 5 and 6 to compute a NE over a large number of instances of the network

routing problem in Figure 3, by letting the costs (delays) vary randomly as follows. For each player $k \neq 0$, the cost of its first path and its second one are chosen independently, uniformly in $[0, 0.01]$. The additional cost if both player 0 and player k choose a path sharing a node is drawn uniformly in $[0, 1]$. The algorithms have been run varying the parameter λ_0, which is the relative speed of the Poisson clock of player 0 with respect to the ones of all other players.

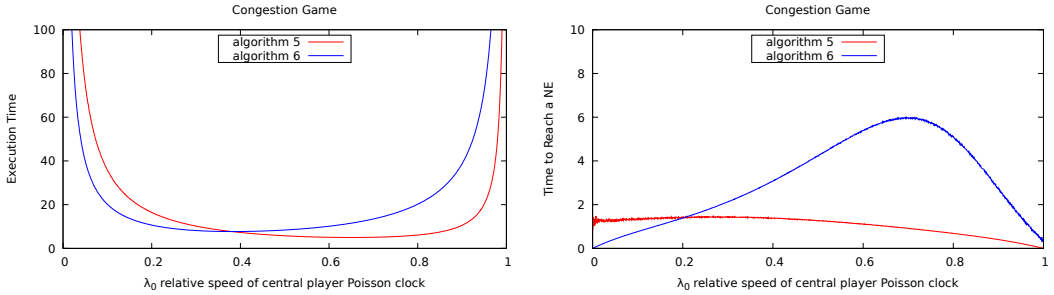


Figure 4: Average total execution time (left) and average time to reach a NE (right) for Algorithms 5 and 6 for the cross traffic routing problem displayed in Figure 3 with 11 players. Each point is the average execution time over 100,000 random games.

The results are reported in Figure 4, that displays the average times for total execution and for reaching a NE. For both algorithms, the time to reach a NE is one order of magnitude smaller than the total time. This means that most of the time is spent to get common knowledge among the players that the NE has been reached. Moreover, for both algorithms, the total execution time is not very sensitive to λ_0, except in the extreme cases where λ_0 is very close to 0 (player 0 rarely plays) or to 1 (all other players rarely play). Also note that the best value for λ_0 is not the same (around 0.8 for Algorithm 5 and around 0.2 for Algorithm 6).

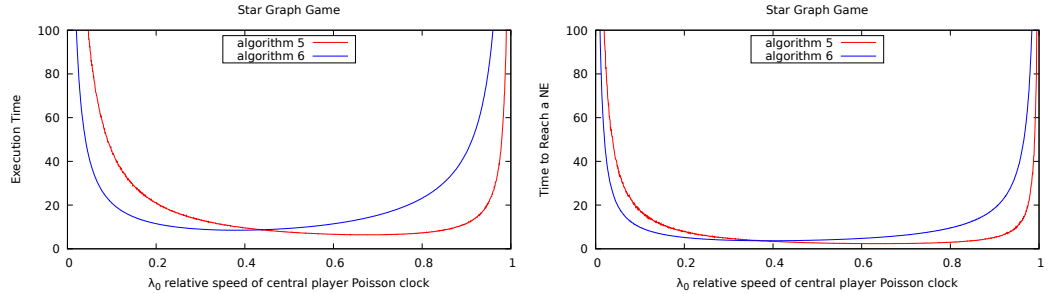


Figure 5: Average total execution time (left) and average time to reach a NE (right) for Algorithms 5 and 6 for a generic potential game with 11 players whose interaction graph is a star. Each point is the average execution time over 100,000 random games.

To give further evidence that parameter tuning in these two algorithms is not easy, we used them to compute a NE over a broader category of games whose interaction graph is the same star graph. The games in Figure 5 all have $U + 1$ players with two actions per player. The payoffs were generated by choosing uniformly in $[0, 1]$ four values for each external player: the cost of choosing either action, and the additional cost of choosing an action also used by the central player. The payoff of the central player is the sum of the payoffs of the other players.

In this case, the behavior of both algorithms w.r.t. λ_0 is displayed in Figure 5 and is rather different from the cross-traffic case. In particular, the time to reach a NE becomes comparable to the total execution time, similarly to classical BRA. The optimal value for λ_0 is close to 0.5 for both algorithms. This shows that the optimal parameter λ_0 depends not only on the interaction graph, but also on the potential function distribution. On the other hand, all these simulations suggest that the performance of the distributed versions of BRA is robust with respect to the choice of the parameters of the Poisson clocks of the users.

References

[1] R. W. Rosenthal, "A class of games possessing pure-strategy nash equilibria," *Int. J. of Game Theory, Springer*, vol. 2, no. 1, pp. 65–67, 1973.

[2] M. Beckman, C. B. McGuire, and C. B.Winsten, *Studies in the Economics of Transportation.* Yale University Press, 1956.

[3] A. Orda, R. Rom, and N. Shimkin, "Competitive routing in multeuser communication networks," *IEEE/ACM Trans. on Networking*, vol. 1, no. 5, pp. 510–521, 1993.

[4] R. G. Gallager, "A minimum delay routing algorithm using distributed computation," *IEEE Transactions on Communications*, vol. 25, no. 1, pp. 73–85, 1977.

[5] J. Wardrop, "Some theoretical aspects of road traffic research. Part ii," *Proc. of the Institute of Civil Engineers*, vol. 1, pp. 325–378, 1954.

[6] T. Roughgarden, *Selfish Routing and the Price of Anarchy.* MIT Press, 2005.

[7] D. Monderer and L. Shapley, "Potential games," *Games and economic behavior, Elsevier*, vol. 14, no. 1, pp. 124–143, 1996.

[8] S. Durand and B. Gaujal, "Complexity and Optimality of the Best Response Algorithm in Random Potential Games," Inria, Research Report RR-8925, Jun. 2016. [Online]. Available: https://hal.inria.fr/hal-01330805

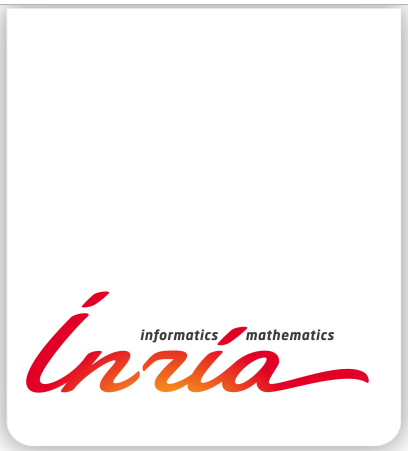
[9] M. Voorneveld, "Best-response potential games," *Economics letters*, vol. 66, no. 3, pp. 289–295, 2000.

[10] S. Durand and B. Gaujal, "Complexity and Optimality of the Best Response Algorithm in Random Potential Games," in *Symposium on Algorithmic Game Theory (SAGT) 2016*, Liverpool, United Kingdom, Sep. 2016, pp. 40–51.

[11] G. Blom, L. Holst, and D. Sandell, *Problems and Snapshots from the World of Probability.* Springer-Verlag, 1994, ch. 7.5 Coupon collecting I.

[12] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms, Third Edition.* MIT Press, 2009.

[13] J. R. Marden and J. S. Shamma, "Revisiting log-linear learning: Asynchrony, completeness and payoff-based implementation," *Games and Economic Behavior*, vol. 75, no. 2, pp. 788 – 808, 2012.



RESEARCH CENTRE
GRENOBLE – RHÔNE-ALPES

Inovallée
655 avenue de l'Europe Montbonnot
38334 Saint Ismier Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399