Hindawi Publishing Corporation
EURASIP Journal on Wireless Communications and Networking
Volume 2010, Article ID 861472, 13 pages
doi:10.1155/2010/861472

Research Article

Using Model Checking for Analyzing Distributed Power Control Problems

Thomas Brihaye,[1] Marc Jungers,[2, 3] Samson Lasaulce,[4] Nicolas Markey,[5] and Ghassan Oreiby[6]

[1] *Institut de Mathématique, Université de Mons, 7000 Mons, Belgium*
[2] *Centre de Recherche en Automatique de Nancy (CRAN), Nancy Université, CNRS, 2 Avenue de la Forêt de Haye, 54516 Vandoeuvre-les-Nancy, France*
[3] *Systèmes et Applications des Technologies de l'Information et de l'Energie (SATIE) CNRS, UMR8029-École Normale Supérieure de Cachan, ENS Cachan, France*
[4] *Laboratoire des Signaux et Systèmes, CNRS, Supelec, Paris 11, 91190 Gif-sur-Yvette, France*
[5] *Laboratoire Spécification & Vérification, ENS Cachan, 94235 Cachan, France*
[6] *Department of Computer Science, Aalborg University, 9220 Aalborg, Denmark*

Correspondence should be addressed to Samson Lasaulce, lasaulce@lss.supelec.fr

Received 14 September 2009; Revised 13 January 2010; Accepted 25 March 2010

Academic Editor: Ping Wang

Model checking (MC) is a formal verification technique which has been known and still knows a resounding success in the computer science community. Realizing that the distributed power control (PC) problem can be modeled by a timed game between a given transmitter and its environment, the authors wanted to know whether this approach can be applied to distributed PC. It turns out that it can be applied successfully and allows one to analyze realistic scenarios including the case of discrete transmit powers and games with incomplete information. The proposed methodology is as follows. We state some objectives a transmitter-receiver pair would like to reach. The network is modeled by a game where transmitters are considered as timed automata interacting with each other. The objectives are then translated into timed alternating-time temporal logic formulae and MC is exploited to know whether the desired properties are verified and determine a winning strategy.

1. Introduction

Power control (PC) is recognized as an important technical problem in communication networks, especially in wireless networks. With the advent of new communications concepts, such as cognitive radio [1] and open spectrum access in unlicensed bands (see, e.g., [2]), being capable of designing PC algorithms in a distributed way has become particularly important. When inspecting the literature related to distributed PC, a dominant methodology arises from the vast majority of existing works. The PC problem is generally modeled by a game where the players are the transmitters, the sets of strategies are the ranges for the transmit powers and the utility/payoff functions of the players can be for example, the transmission rate (see, e.g., [3–5]) or energy-efficiency (see, e.g., [6–8]). As the terminals are assumed to be free, selfish and rational decision-makers, one fundamental issue is to know whether there is a solution to this conflict of interests. In the literature of PC games the most used solution concept is the Nash equilibrium (NE). An NE corresponds to a game outcome/state effectively observed when every player does the best for itself (rationality assumption), knows the others do so (rationality is common knowledge), and has a complete knowledge of the game played (complete information assumption). One of the properties of such a state is that it is stable to a single deviation, that is, if one player deviates from the equilibrium unilaterally, it looses in terms of utility. Therefore, a generic technical issue that is treated in works on distributed PC is the existence issue for an NE, which amounts to proving (mathematically) the existence of a fixed point (see, e.g., [9] for the standard approach of distributed power control).

In this paper, a different point of view is adopted. The main two differences between the existing game-theoretic works and the work presented here is as follows. First, we make a different behavioral assumption for the players. Each transmitter does not assume that the others do the best for themselves but rather that they can form a coalition which can do the worst for the considered transmitter; a strategy allowing a player to reach its objective under these conditions is called a winning strategy. For instance, this *worst-case assumption* can be very relevant to engineers who want to design transmitters that have to reach a certain quality of service independently of the design/strategy of the other transmitters/manufacturers/operators. Second, we do not want to prove mathematically the existence of an NE or a network state having certain properties. Rather, we formulate the PC problem as a timed game [10] between the considered pair of nodes and its environment, and prove the existence or nonexistence of such a (winning) strategy by translating the objective into a temporal logic formula [11] and exploiting the powerful technique of *model checking* (MC; see, e.g., [12]). Although MC is not very well-known from the communications community, it is a concept of paramount importance in the computer science community, as witnessed by its inventors winning the ACM Turing Award in 2007. It aims at checking, automatically exhaustively and formally (with a computer) if a system (possibly involving several agents) verifies some properties expressed in a given logical language. In our case, MC provides the answer to whether a certain property can be satisfied. Additionally, in the case where it is satisfied, the model checker is able to compute a winning strategy (a PC policy in our case). Our motivation for using model checking (computational approach) instead of proving a theorem (analytical approach) is essentially twofold: we do not consider Nash equilibria (and therefore fixed-point solutions); our study aims at considering the practical case where transmit powers are discrete. Concerning the latter point, note that, as mentioned recently [13], the vast majority of works on distributed PC assume continuous transmit powers while in many existing communications systems it can only be discrete. All results based on compactness and convexity of the strategy sets of the players are not valid anymore. In particular, these results include existence, uniqueness, convergence results for pure Nash equilibria in noncooperative PC games. Therefore, the case of discrete powers can become difficult and even impossible to solve analytically.

This paper is structured as follows. First, we describe the network model under investigation and justify the assumptions made (Section 2). Second, we state some properties a given transmitter-receiver pair would like to be verified and present the strategic form of the distributed PC game viewed from a given transmitter (Section 3). In Section 4, we reformulate this game into a timed game by using the notion of timed automaton and propose a way of modeling the timed game which is compatible with the model checker used, namely, Uppaal-TiGA [14]. In Section 5, we translate the desired properties into formulae expressed in a given temporal logic and use Uppaal-TiGA to know whether the network allows theses properties to be verified. At last, in Section 6 concluding remarks and possible extensions are provided.

2. Problem Statement

Here we present the assumed signal model but it is important to keep in mind that the proposed approach is not inherent to the assumed communication model and network topology, and can be directly applied to other models just by changing the expressions of the different signal-to-interference plus noise ratios (SINRs). In this paper, we assume an interference channel [15] with K transmitter-receiver pairs $(\mathrm{Tx}_i, \mathrm{Rx}_i)$, $i \in \{1, \dots, K\}$, as described in Figure 1. The baseband signal received by Rx_i, can be expressed by:

$$y_i(t) = \sum_{j=1}^{K} h_{ji} x_j(t) + z_i(t), \tag{1}$$

where $\forall (i, j) \in \{1, \dots, K\}^2$, h_{ji} represents the channel gain of the link between nodes j and i; $x_j(t)$ represents the signal transmitted by Tx_j and is assumed to belong to a finite alphabet (an element of this alphabet is called a symbol, e.g., a quadrature amplitude modulation symbol), t being a time index associated with a certain rate at which the medium is used (symbol index, channel use index); $z_i(t)$ is the additive white complex noise at receiver i, z_i is assumed to be distributed according to a zero-mean Gaussian random variable with variance σ_i^2. As we analyze a PC problem, we make the same assumptions as in the related works ([6, 7], etc.), that is, the channel gains are assumed to be constant over the whole duration of the transmission but can be updated on a block-by-block basis; note that a block is defined as a sequence of N consecutive symbols which comprises a training sequence that is, a certain number of consecutive symbols used to estimate the channel, the SINR, and so forth. Usually, the PC problem consists in updating the power of a given transmitter every block duration. Here, with more generality, we assume that it can be updated within a block duration. We will therefore assume a block can be seen as N/n consecutive subblocks of n symbols each and use a time index τ to indicate when, in a given block, the transmit power of a given transmitter is updated. Using this time or sub-block index, one can define within a given block the instantaneous SINR for receiver i as follows:

$$\forall \tau \in \left\{1, \dots, \frac{N}{n}\right\}, \quad \mathrm{SINR}_i(\tau) = \frac{|h_{ii}|^2 p_i(\tau)}{\sigma_i^2 + \sum_{j \neq i} \left| h_{ji} \right|^2 p_j(\tau)} \tag{2}$$

with

$$p_i(\tau) = \frac{1}{n} \sum_{t'=1}^{n} \left| x(t') \right|^2, \tag{3}$$

which means that the instantaneous power results from averaging the power over n consecutive symbols. The vector $(p_1(\tau), \dots, p_K(\tau))$ will be called the network state for sub-block τ. We will call the time elapsed during n symbol

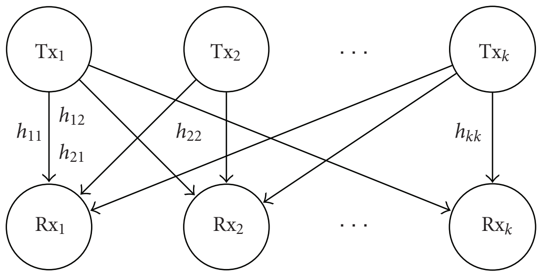

FIGURE 1: Assumed communication model: a K-user interference channel.

durations a time unit (TU). In addition to the assumptions made so far, we will suppose the following

Assumption 1. The transmit power of each transmitter $i \in \{1,\ldots,K\}$ is discrete, $p_i(\tau)[\mathrm{dBm}] \in \mathcal{P}_i$, $\mathcal{P}_i = \{P_0, P_0 + \Delta P, \ldots, P_0 + (M-1)\Delta P\}$, and can be increased or decreased by at most one power increment ΔP every TU; w.l.o.g. M is assumed to be an odd integer.

Assumption 2. Every arriving user starts transmitting at the power $p_i(t=0)[\mathrm{dBm}] = P_{\mathrm{wu}} = P_0 + ((M-1)/2)\Delta P$ ("wu" stands for *wake up*).

Assumption 3. Each transmitter can start/stop transmitting at any time but has to remain at the ON or OFF state during a minimum amount of time t_{ON} or t_{OFF}.

Assumption 4. Each transmitter (say i) knows its SINR and $(h_{ji})_j$ perfectly.

Assumption 5. There is a maximum SINR, denoted by $\gamma_{\max}$, above which no transmitter is allowed to operate.

Assumption 6. Each receiver Rx_i knows the channel gain h_{ii} only.

Assumption 7. The number of transmitters, that is, K can possibly vary at any time, which means in particular that they can be asynchronous (to the best to the authors' knowledge the models currently available in the literature of distributed PC problems cannot account for this feature).

Let us give some motivations for these assumptions. First, concerning Assumption 1, note that the transmit power cannot vary arbitrarily over time, it can only be increased or decreased by one power step every TU. This might translate some physical limitations due to the transmitter (e.g., because of the finite power amplifier slew rate). In fact, for the model checker we exploit in Section 5, this assumption implies a dramatic reduction of complexity for the verification procedure. To conclude with Assumption 1, note that the discrete power assumption ensures the existence of a mixed NE in strategic-form non-cooperative PC games with complete information and rational users. In this paper, we do not want to exploit this result for at least three reasons: it is demanding in terms of information assumptions; we do not focus on the case of network states having the NE property (namely, they are stable to a single deviation) but look at states which can have different properties; mixed strategies are not always relevant in communication networks (e.g., for short transmissions, sporadic traffic with nonstationary network parameters). Note that [13] also addressed the case of discrete powers but the goal in [13] is to study the existence and convergence issues towards a pure or mixed NE by exploiting stochastic learning algorithm. Assumption 2 is arbitrary but reasonable if the environment is unknown before starting the transmission. Assumption 3 accounts for possible constraints from the network or used technology. Assumption 4 is realistic in terms of knowledge since in many communication systems, there is a feedback mechanism allowing the transmitter to be informed with the SINR. This information assumption is also not very strong in comparison to the usual assumptions generally needed to analyze Nash equilibria in PC games, especially games with complete information. Assumption 5 indicates the existence of a constraint on the SINR for every transmitter. For instance, this constraint can be imposed by a regulator or follow from an agreement between operators/manufacturers. Note that this constraint is different from the one on the maximum transmit power. It can be stronger in fact. Indeed, if a transmitter, say i, is very close to its access point, it can happen that transmitting at full power, that is, $p_i[\mathrm{dBm}] = P_{\max}[\mathrm{dBm}] = P_0 + (M-1)\Delta P$ does not meet the condition $\mathrm{SINR}_i \le \mathrm{SINR}_{\max}$. Assumption 6 is a standard assumption, which is necessary to ensure coherent communications. Assumption 7 indicates that the proposed approach applies to networks where transmitters are asynchronous and can enter/leave the network at any time. This is a consequence of our information assumptions: the individual SINR can be known without knowing the instantaneous number of active transmitters and the worst case assumption only requires the knowledge of the maximum transmitters with which a transmitter can interact (this number is necessarily limited in wireless networks because of path loss effects and limited user density). At last, the fact that the arrival/departure instant of a transmitter is not a multiple of the sub-block duration is known to be a negligible effect in real wireless networks, which is what is assumed here.

3. Proposed Network Properties and Power Control Game Definition

Our ultimate objective is to know if, in the network described in the preceding section, every transmitter is satisfied. A transmitter will be said to be satisfied if it can exchange a certain volume of data and at the same time reach a transmission quality target. In fact, we assume that every transmitter wants to reach this objective independently of what the other transmitters do. As a consequence of this, the network can be seen from the point of view of a given transmitter, say i, as a game between itself and its environment. The environment includes the other transmitters and the channel. The channel consists of the matrix $\mathbf{H}$ whose entries are the channel gains h_{ij}. It is fixed for a given block of data. Therefore, transmitter

i will be satisfied if there exists a winning strategy in the game between itself and its environment, that is, the desired properties will be verified whatever the environment does. The existence of such a winning strategy will be proved in Section 5 by using the Uppaal-TiGA model checker. We consider the following properties. Let $\Delta\tau$ and t_{out} be two durations expressed in TUs and $\gamma_{\min}$, γ_- be two SINR thresholds with $\gamma_{\min} \geq \gamma_-$.

Property 1. The network satisfies Property 1 for player i if the following three relations are verified:

$$\exists \overline{\tau}_i, \quad \forall \tau \in \{\overline{\tau}_i, \overline{\tau}_i + 1, \dots, \overline{\tau}_i + \Delta\tau\}, \quad \forall p_{-i}(\tau) \in \widetilde{\mathcal{P}}_{-i}, \mathrm{SINR}_i(\tau) \geq \gamma_{\min}; \tag{4}$$

$$[\exists(\tau_0, \tau_-), \forall \tau \in \{\tau_0, \tau_0 + 1, \dots, \tau_0 + \tau_-\}, \quad \gamma_- \leq \mathrm{SINR}_i(\tau) \leq \gamma_{\min}] \Longrightarrow \tau_- \leq t_{\text{out}}; \tag{5}$$

$$\forall \tau \in \left\{1, \dots, \frac{N}{n}\right\}, \quad \mathrm{SINR}_i(\tau) \geq \gamma_-. \tag{6}$$

Property 2. The network satisfies Property 2 for player i if Property 1 is satisfied in the special case $\gamma_{\min} = \gamma_-$ or equivalently if $t_{\text{out}} = 0$.

Property 3. The network satisfies Property 3 for player i if Property 2 is satisfied in the special case $\Delta\tau = 0$, that is, the target SINR $\gamma_{\min}$ can be reached during at least one sub-block index (namely $\overline{\tau}_i$).

In (4), $p_{-i}(\tau)$ is the $(K-1)$-uplet $(p_1(\tau), \dots, p_{i-1}(\tau), p_{i+1}(\tau), \dots, p_K(\tau))$. This $(K-1)$-dimensional vector lies in general in the product space $\mathcal{P}_1 \times \cdots \times \mathcal{P}_{i-1} \times \mathcal{P}_{i+1} \cdots \times \mathcal{P}_K$. But here, we have used the notation $\widetilde{\mathcal{P}}_{-i}$ to clearly indicate that we assume that the PC policy implemented by every transmitter cannot be arbitrary but has to verify Assumption 5

$$\widetilde{\mathcal{P}}_i = \{p_i \in \mathcal{P}_i, \mathrm{SINR}_i \leq \gamma_{\max}\}. \tag{7}$$

We therefore see that, for a fixed channel, transmitter i assumes the worst case in terms of environment, that is, in terms of behavior for the other transmitters. Said otherwise, a given transmitter is free to do what is best for itself but assumes that the other transmitters can form a coalition aiming at decreasing its chances to reach its objective. The drawback for this assumption is that the obtained results will be pessimistic and the desired property will be met more often than expected. But, we have to keep in mind that one of our goals was to relax the rationality assumption (every transmitter does the best for itself and this is common knowledge to every transmitter). We can see that in the proposed framework it is effectively not needed. This framework can be very useful in some scenarios. For example, it can happen that, over a certain period of time, Tx_2 wants to maximize its transmission rate (high-speed uploading) while later on, for a second phase, it wants to save its energy. From the point of view of Tx_1, assuming that it always wants to save its energy, Tx_2 will not behave rationally over the first phase. In brief, the rationality assumption can be very questionable, especially in heterogeneous networks (with different manufacturers, operators, services, technologies, etc.). Therefore, if Property $q \in \{1, 2, 3\}$ is verified for transmitter i, it means that the latter can reach its objective in the worst environment.

Now, let us comment on the proposed properties separately. Property 3 allows a transmitter to be sure that for a fixed channel state, there will be a moment at which the packet it wants to send can be received reliably, independently of the power dynamics of the other transmitters. This occurs when a short message has to be transmitted. Property 2 allows, this time, the transmitter to be ensured to transmit a certain volume of data ($D \times \Delta\tau$ bits where D is the data rate in bit/s) reliably within a time interval equal to the channel coherence time, that is, the interval over which the channel state can be considered to be fixed. As $t_{\text{out}} = 0$, the property also imposes that connectivity is never lost. Property 1, like Property 2, guarantees that a certain volume of data can be exchanged reliably but is more tolerant than Property 2. Indeed, it allows the receiver to operate below the minimum required SINR for ensuring a certain transmission quality target for a certain duration (t_{out} TUs). For example, this allows one to account for interesting scenarios such as the case where a new transmitter enters the network. When a new user arrives, it can happen that the minimum SINR target of player i, that is, $\gamma_{\min}$ is not met for a short interval only and operating at a lower SINR γ_- can suffice to maintain connectivity. We will comment more on this scenario below. Another scenario where considering this property is relevant is an application or a service where two levels of quality are admitted for the transmission. To conclude on Property 1 we make some comments on the physical feasibility of such a property: (4) translates the existence of a time windows over which the medium is sufficiently good, which happens, for example, when the link h_{ii} is good enough some transmitters stop emitting or the other transmitters are far enough from transmitter i; (5) translates the nonexistence of a too wide windows over which the channel conditions are bad in the sense that the system is not robust to a network change over this time windows; (6) ensures that over the whole block or packet duration connectivity is not lost.

Here, we justify in a more detailed manner why we use two SINR thresholds (namely, $\gamma_{\min}$ and γ_-) instead of one as it is usually the case for the standard formulation of the PC problem (see, e.g., [16]). Indeed, in [16] and related papers the goal is to minimize the transmit power under the constraint $\mathrm{SINR}_i \geq \gamma_-$ with $t_{\text{out}} \to +\infty$; this policy allows one to reach the minimum transmission quality target and save energy. Here, by allowing the SINR target to be different from the minimum SINR target we ensure a certain degree of robustness against networks changes like the arrival of a new user. Therefore, the value of γ_- has to be chosen in accordance with desired robustness degree. To be more concrete, let us consider an example. Assume a network comprising 2 pairs of nodes at time τ and 3 pairs of nodes at time $\tau + 1$ and the existence of a steady state for the network with two users, that is, $p_i(\tau) = \overline{p}_i$. Without loss of generality, consider transmitter 1. Under typical assumptions, the probability that SINR_1 at time $\tau + 1$ is less than γ_- given that it equals $\gamma_{\min}$ at time τ can be checked to be (see Appendix A)

$$\Pr[\mathrm{SINR}_1(\tau+1) \leq \gamma_- \mid \mathrm{SINR}_1(\tau) = \gamma_{\min}] = \Pr\left[|h_{31}|^2 \geq \alpha\left(\frac{\gamma_{\min}}{\gamma_-} - 1\right)\right], \tag{8}$$

where $\alpha = (\sigma_1^2 + |h_{21}|^2\overline{p}_2)/P_{\mathrm{wu}}$. Additionally, if $|h_{31}|^2$ is exponentially distributed (case of Rayleigh fading) we have that the probability of loosing connectivity (outage probability), which we call P_{out}, is given by

$$P_{\mathrm{out}} = \exp\left[-\frac{\sigma_1^2 + |h_{21}|^2\overline{p}_2}{P_{\mathrm{wu}}}\left(\frac{\gamma_{\min}}{\gamma_-} - 1\right)\right]. \tag{9}$$

Therefore, by fixing a desired target for P_{out} we find the appropriate value for $\gamma_{\min}$ to be used in the PC algorithm. As a consequence, for a given channel state, Properties 3–1 will be verified or not but $\gamma_{\min}$ allows one to tune the probability of verifying the latter. The price to be paid for a certain robustness against the arrival of a new user is a higher energy consumption w.r.t. the standard assumption $\gamma_{\min} = \gamma_-$.

At this point we have all the elements to define the PC game under investigation properly. As already mentioned, the game has to be defined from a given transmitter's point of view. Consider transmitter i.

Players. There are two players in the game. Transmitter i is the first player (protagonist) and the rest of the transmitters $(-i)$ constitutes the second player (antagonist).

Strategies. A pure strategy for transmitter i consists of a sequence of causal functions $(p_i^\tau)_{\tau\in\{1,\ldots,N/n\}}$ with

$$p_i^\tau : \left| \begin{array}{ccc} \mathcal{H}_i^{(\tau)} & \longrightarrow & \widetilde{\mathcal{P}}_i, \\ (\mathrm{SINR}_i(1),\ldots,\mathrm{SINR}_i(\tau-1)) & \longmapsto & p_i(\tau), \end{array} \right. \tag{10}$$

where $\mathcal{H}_i^{(\tau)}$ is the set of private histories for the SINR of transmitter i, each possible history verifying Assumptions 1 to 7. Note that the choice of these functions does not depend on a particular action (vector of transmit powers) of the opponent (called a move in game theory) but only depends on the knowledge of the considered transmitter on the game. In particular, it depends on what it believes about its opponent and its objective.

Utilities. If a given Property is satisfied (resp., not satisfied) for transmitter i, transmitter i gets $+1$ (resp., -1) and the environment $-i$ (i.e., the antagonist) gets -1 (resp., $+1$). If a player has several winning strategies (providing him with $+1$), he chooses the one minimizing the consumed energy or equivalently the quantity $\sum_{t=0}^{N} |x_i(t)|^2$.

Proving the existence of a winning strategy for player i in the stated zero-sum game appears to be nontrivial. We note that one of the problems here is the dynamic aspect of the game, which is not present in the conventional approach of the distributed PC problem (e.g., [16]). Indeed, the other transmitters can start/stop emitting whenever they want to and also choose any feasible sequence of actions $(p_i(\tau))_\tau$. Additionally, the game is with incomplete information since a transmitter (say i) does not know all the game; for instance the channel gains of the others are not known $(h_{jk}, k \neq i)$. A possible way of tackling the considered problem is to reformulate the game as a timed game, for which MC can be applied. This is the purpose of the next section.

4. Translating the Distributed PC Problem into a Timed Game

We have presented the assumed network model in Section 2 and the properties we would like it to verify in Section 3. In order to know whether these properties are verified or not we want to exploit the concept of MC and more specifically the Uppaal-TiGA model checker, which is the purpose of Section 5. To this end, we need to translate the network model into a timed automaton model, and express the desired properties in a formal language named TATL (for *timed alternating-time temporal logic*). This is done in Section 4.2. In order to make this paper sufficiently self-contained and the corresponding methodology applicable to other communication scenarios, we first review, in Section 4.1, some important notions on timed automata [17] and timed games [10]. More details are provided in Appendices B and C.

4.1. Review of Basic Notions

4.1.1. Timed Automata. Roughly speaking, a timed automaton is a finite automaton enriched with clocks, which are real-valued variables used to measure the time elapsed between different events of the automaton. All clocks evolve at the same rate, but their values can be reset when firing transitions. The values of the clocks can be used to enable or disable some of the transitions. A *state* (or *configuration*) of a timed automaton is a pair (ℓ, v) where ℓ is a location of the timed automaton and v is a valuation assigning to each clock of the automaton its nonnegative real value. There are two types of moves in timed automata.

Delay Transition. They consist in staying at the same location and letting time elapse. The automaton then goes from some configuration (ℓ, v) to another configuration $(\ell, v+t)$, where $t \in \mathbb{R}^+$ and $v + t : x \mapsto v(x) + t$ (all the clocks evolve at the same rate).

Discrete Transitions. They consist in firing a transition of the timed automaton. A transition $\ell \xrightarrow{g;a;r} \ell'$ of the timed automaton comprises the following elements:

(i) the source and target locations ℓ and ℓ';

(ii) a *guard* g, which is a constraint on clocks (for instance, "$x \geq 1$", requiring that clock x must have a value greater than or equal to 1 for being allowed to fire this transition);

(iii) a label a on the transition, representing the corresponding event;

(iv) the set r of clocks to be reset (e.g., "$x := 0$").

Firing this transition from a state (ℓ, v) is only allowed if the guard is satisfied. We then switch to state (ℓ', v'), with $v'(x) = 0$ if $x \in r$, and $v'(x) = v(x)$ otherwise.

Timed automata are thus a finite representation of an infinite-state automaton. Still, the underlying infinite-state automaton is quite "regular", and this regularity allows one to have several problems decidable (for instance, "is there an execution of the automaton reaching a given state?" (reachability) or "do all executions avoid a given bad state?" (safety)). The algorithms generally rely on a finite-state automaton, called the *region automaton* [17], which roughly exhibits the same behaviors as the infinite-state automaton it abstracts. More precise definitions for timed automata are provided in Appendix B.

4.1.2. Timed Games. Timed games [10] are timed automata in which some transitions can be uncontrollable, in the sense that a player cannot neither prevent them from occurring nor force them to occur. In this setting, a given state is said to be *reachable* if there exists a *strategy* which, if consistently applied, and whichever uncontrollable transitions are applied, ensures that the goal state will be reached. In order to illustrate these notions let us consider a simple example of timed game. Consider a transmission game with two players (the transmitters). Assume that each transmitter has two possible actions: either transmits at full power (called High mode) or at low power (called Low mode). The transmission constraints are as follows. The transmitter has to stay for a minimum amount of time in each mode, and cannot stay there forever: precisely, transmitter i has to stay between m_{L_i} and M_{L_i} (resp., m_{H_i} and M_{H_i}) TUs in the Low (resp., High) mode. We also assume the existence of a state for which the system (or network) is blocked if both transmitters stay at the same time in the High mode for too long, say T TUs; the clock associated with the overall system or network will be denoted by y. Does a given user have a way of controlling its power in order to ensure that the network will not reach the blocking state, whatever the other transmitter will do? This situation can be modeled by a timed game (see Figure 2 where some choices for the values for the guards and invariants have been made. Locations ℓ_0 to ℓ_3 model the different modes of the two transmitters and location ℓ_4 is the blocking location. We have considered the point of view of transmitter 1. We see that user 1 acts as a controller (playing with solid transitions) and the second user acts as the environment (controlling dashed transitions). One can show that the controller has no strategy to avoid reaching the blocking location ℓ_4. Even in this simple example, the non-existence problem of a winning strategy is not trivial. The more general problem of PC is more complex than this problem, which is one of the reasons why automated techniques (such as MC) are used to answer this type of questions. For the interested reader, more precise definitions for timed games are provided in Appendix C.

4.2. Timed Power Control Game Modeling. In this section, we propose a model for the general PC game presented in Section 2, taking into account the features of the model checker Uppaal-TiGA. We proceed in three steps. With each of the first two steps a figure representing the timed automaton effectively implemented is associated. In the first and second steps, for clarity, we restrict our attention to the case of two transmitter-receiver pairs.

Step 1. Figure 3 represents our model of a transmitter. Each transmitter has its own two clocks, denoted by x and y, and can be either ON or OFF. Clock y is used to force the transmitter to stay at least t_{ON} (resp., t_{OFF}) TUs in the ON (resp., OFF) state. In the figure, which corresponds to the exact model implemented in Uppaal-TiGA, the quantities t_{ON} and t_{OFF} are renamed on delay and off delay, and switching between ON and OFF is represented by the three-state loop in the upper part of Figure 3. Let us comment on the other three loops. The one on the left represents the transmitter increasing its power. This can only happen if enough time has elapsed since the last change (i.e., if $x \geq$ `delay` where `delay` is some positive amount of time) and if the floor SINR target is not reached yet (this last condition is encoded by the constraint $A_0 p_1 < m_1(B_0 + C_0 \cdot p_2)$, where A_0, B_0, C_0 and m_1 are chosen, so that this equation represents the required condition on the SINR). The loop on the right is the opposite case, which corresponds to the situation where the transmitter decreases its power. The loop in the middle, represented by a dashed transition, is uncontrollable for the transmitter: the transmitter cannot prevent this transition from occurring (as soon as the guards are true). As a consequence, any winning strategy for the transmitter must in particular be able to handle the case where this transition is fired. This is a "coding trick" we use to model the fact that the transmitter has to decrease its power when it is too high, in the sense that it forces the transmitter to take this case into account: the transmitter has a winning strategy only if the latter is able to handle the case where this transition is fired as soon as it is enabled. In a real communication system, the constraint $\text{SINR}_i \leq \gamma_{\max}$ could be imposed to the transmitter (even if it is a free decision maker), for example, by a regulator or the receiver itself (if the latter breaks the communication). Finally, let us stress on the fact that the transmitter depicted on Figure 3 will "play" against a similar opponent that is, having similar constraints for increasing or decreasing its power but implementing any feasible PC policies.

Step 2. An important feature we want the network to have is that the communication is broken if the SINR goes below a certain threshold. The objective of transmitter 1 will be to establish and maintain a communication on this network. To easily represent this objective, we add an observer automaton. (*Observer* here means that it plays no role in the evolution of the system, but changes states according to the values of the SINR. There is only one copy of this automaton in our system, which is used to keep track of whether transmitter 1 manages to establish and maintain

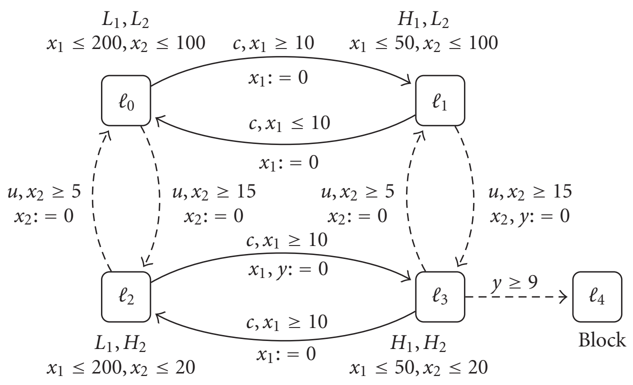

Figure 2: Timed game example: each x_i is the clock for player i; $(m_{L_1}, M_{L_1}, m_{H_1}, M_{H_1}) = (10, 200, 10, 50)$ and $(m_{L_2}, M_{L_2}, m_{H_2}, M_{H_2}) = (15, 100, 5, 20)$; y is the clock for the network; $T = 9$.

a communication.) This is what Figure 4 represents. This automaton has three states. State Emit represents the target state of transmitter 1, where it can emit with an SINR larger than $\gamma_{\min}$. Thus, player 1 can go to this state if the SINR is high enough, encoded by the condition $A_0 p_1 \geq H(B_0 + C_0 p_2)$. Then, state Emit2 is a degraded operating mode, in which the SINR can be below a given threshold (γ_-). Again, we make the transition to Emit2 uncontrollable to "force" the system to go to this state when the guard holds true. Two cases can occur in Emit2: either the transmitter manages to have its SINR back to a reasonable value, in which case it will be allowed to go back to the Emit state; or it does not manage to do so within `timeout` TUs (corresponding to t_{out}), in which case the system will be taken to the Stop state, representing the fact that the connection is broken.

Step 3. In order to model a system with more than two transmitter-receiver pairs, the only things to be changed in Steps 1 and 2 are the expressions of the SINRs and the corresponding constraints. For example, the condition from Emit to Emit2 in the observer becomes $A_0 p_1 < m_1(B_0 + \sum_{i \neq 1} C_0^i p_i)$.

5. Model Checking and Uppaal-TiGA

Now we have translated the PC problem under investigation into a timed game. We will check if the corresponding game satisfies the required properties, which will be expressed as temporal logic formulas. In order to make this paper self-contained, we first briefly review the concept of MC. Then, as the last step of our methodology, we express the desired properties in TATL and use Uppaal-TiGA to check these properties are verified by the timed game modeling the network of interest.

5.1. Model Checking. Formal verification is a field of computer science where the aim is to check that the behavior of an automated system satisfies some given properties. It relies on a mathematical basis, involving logical reasoning. *Model checking* is one of the existing formal-verification techniques: in this setting, the considered automated system is modeled by a finite-state automaton (or by exploiting a related formalism, depending on the properties under consideration). Model checking aims at automatically and exhaustively verifying that all the executions of this automaton satisfies the properties we want to check for the original system. Those properties can be expressed in various ways: basic properties such as reachability or safety can be checked, but richer properties can be handled thanks to *temporal logics*. Those logics extend classical propositional logics with *temporal modalities*. For instance, $\Diamond$`Goal` expresses that "eventually, property `Goal` will hold true", and $\Box\neg$`Error` expresses that "always (in the future), `Error` will not hold" (i.e., an error will never occur). Combining them, we can write $\Box$(`Request` $\Rightarrow$ $\Diamond$`Grant`) for expressing that any request is eventually granted. Original model-checking algorithms have been developed to handle timed automata and *quantitative* properties, such as "every request is granted within 10 time units". Temporal logics have been extended accordingly, so that the above property can be written $\Box$(`Request` $\Rightarrow$ $\Diamond_{\leq 10}$`Grant`). They have also been extended to deal with open systems (games and timed games). In this setting, MC aims at verifying *controllability* properties: "*does there exist a strategy for restricting the behavior of the system in order to enforce the given property*". Temporal logics have been extended in order to express this kind of properties.

In the following, we use the Uppaal-TiGA model-checker, which is a model-checker for timed controllability properties. The three properties stated in Section 3 are then encoded by the following three temporal logic formulae (we use the syntax of Uppaal-TiGA here):

```
control:A < > (Observer.Emit)
control:A < > (Observer.Emit &&
                  Observer.y > Δτ)
control:A[]((Observer.Emit||
                Observer.Emit2)||
                Observer.y ≤ Δτ)
```

The first sentence means that there is a strategy (`control`) under which all executions (`A`) eventually reach (< >) location `Observer.Emit`. The second sentence expresses that there is a strategy to make the system reach location `Observer.Emit` and stay there for at least $\Delta\tau$ time units there (because clock `Observer.y` is reset when we enter this location, so that being in `Observer.Emit` with `Observer.Emit`$> \Delta\tau$ is equivalent to staying in that location for $\Delta\tau$ time units). Finally, the third statement requires the existence of a strategy whose outcomes satisfy the following property: it is always ([]) true that the observer is either in Emit of Emit2, except possibly at the very beginning, when y is small enough. This precisely encodes Property 1.

5.2. Verification Results Obtained by Using Uppaal-TiGA. We now provide some results we have obtained when running the Uppaal-TiGA model-checker for our models and the properties we are interested in. All the quantities used (reals) by Uppaal-TiGA have been approximated by rationals having

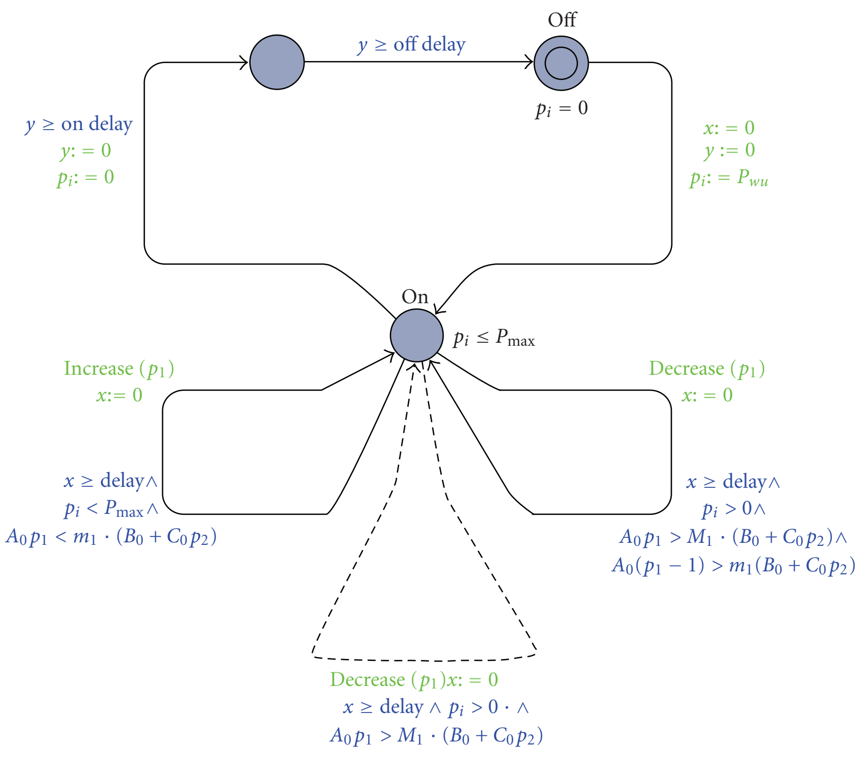

FIGURE 3: Our model of a transmitter. Equations in blue are guards of the closest transition, while updates are in green.

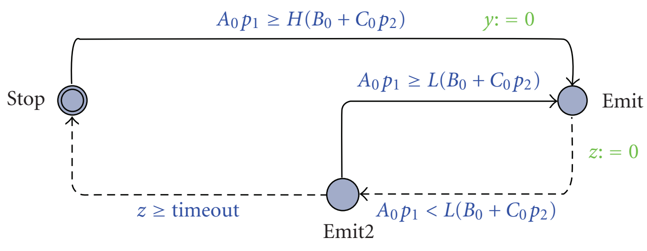
$A_0 p_1 \geq H(B_0 + C_0 p_2)$
$y := 0$
Stop
$A_0 p_1 \geq L(B_0 + C_0 p_2)$
Emit
$z := 0$
$z \geq$ timeout
Emit2
$A_0 p_1 < L(B_0 + C_0 p_2)$

FIGURE 4: The observer implementing whether or not the connection can be established or maintained.

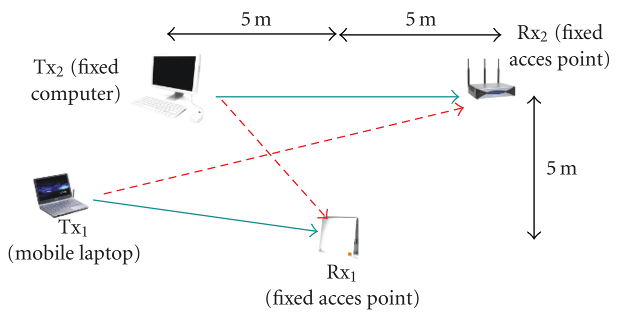
5 m
5 m
Rx_2 (fixed acces point)
Tx_2 (fixed computer)
5 m
Tx_1 (mobile laptop)
Rx_1 (fixed acces point)

FIGURE 5: Assumed scenario. The network comprises one mobile laptop (Tx_1), one fixed computer (Tx_2), and two fixed access points (Rx_1) and (Rx_2).

8 significant numbers. We have assumed a scenario with two transmitter-receiver pairs. Figure 5 represents the topology of the network considered. We assume that three terminals have fixed positions and one terminal can move. Depending on the location of the mobile terminal (e.g., a laptop), the channel gains h_{ij} have different values and the question is precisely to know if the three properties stated in Section 3 are verified in a given point. The channel gains are assumed to follow the following path loss model:

$$\left|h_{ij}\right|^2 = \beta\left(\frac{d_0}{\sqrt{d_{ij}^2 + h^2}}\right)^{\lambda}, \tag{11}$$

where $\beta = 0.01$, d_0 is a reference distance taken to be equal to 0.1 m, d_{ij} the distance between nodes i and j and λ the path loss exponent, taken to be equal to 3. The distance $h = 0.5$ m is used to avoid the divergence of the path loss in $d_{ij} = 0$. The transmit power of Tx_i is such that $P_i[\text{dBm}] \in \{11, 14, 17, 20, 23, 26, 29\}$. The noise power σ_i^2 is -90 dBm. Also $\gamma_-[\text{dB}] = 5$, $\gamma_{\min}[\text{dB}] = 10$ and $\gamma_{\max}[\text{dB}] = 13$.

As mentioned just above only Tx_1 can move. Its coordinates (in meters) are denoted by (a, b) with $a = i * \Delta a$, $b = j * \Delta b$, so that $(a, b) = (5, 10)$ if Tx_1 and Tx_2 are colocated, $(a, b) = (10, 5)$ if Tx_1 and Rx_1 are co-located, and so forth. With these conventions, the distance between Tx_1 and Rx_1 is expressed as $d_{11}^2 = (10 - a)^2 + (5 - b)^2$. With the set of all possible positions for Tx_1 one can associate an image which is labeled by $I_{(i,q)}$ and defined as follows: a white pixel indicates that Property $q \in \{1, 2, 3\}$ is satisfied for transmitter $i \in \{1, 2\}$ at the considered point, a gray pixel indicates that Property q is not satisfied at the

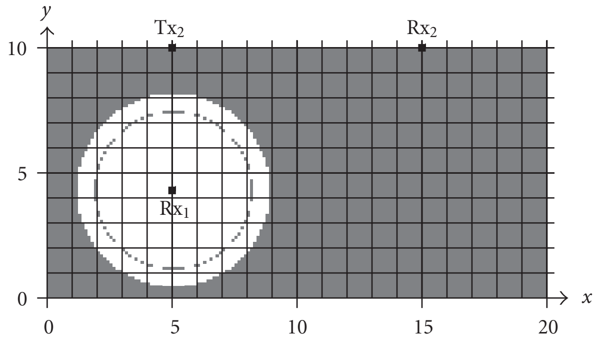

FIGURE 6: $I_{(1,3)}$: verification of Property 1 for transmitter 1 with $(\Delta a, \Delta b) = (0.125\,\text{m}, 0.125\,\text{m})$.

considered point and a black pixel indicates the positions of the fixed terminals. Here are some results we obtained by using Uppaal-TiGA.

5.2.1. Image $I_{(1,3)}$. Here we consider a simple scenario, which allows us to validate the model implemented with Uppaal-TiGA. Figure 6 shows the area in which Property 1 is satisfied for transmitter 1 for $\Delta t = 1\,\text{TUs}$, $t_{\text{out}} = 3\,\text{TUs}$, $t_{\text{ON}} = 5\,\text{TUs}$, $t_{\text{OFF}} = 5\,\text{TUs}$. This area is a disk centered in the position of receiver 1, which can be explained by a simple calculation on the path loss effect. What is more surprising is the presence of the gray ring, which shows that connectivity can be lost as d_{11} increases and then recovered while d_{11} still increases. This non-trivial effect has been observed in many other scenarios. Our interpretation is that this effect is due to the discrete nature of the transmit power. Uppaal-TiGA allowed us to partly confirm this interpretation. Indeed, Uppaal-TiGA allows us to synthesize the corresponding strategies and *play* against it: we could not explore all possible plays, but the existence of the ring appearing in the figure seems to be linked to the fact that the transmitter has to decrease its power one more time by moving from the closest inner adjacent white circle to the gray ring. Figure 7 shows a zoom on a part of this area when the power increment is 1 dBm (instead of 3 dBm in the case of Figure 6). The results provided here are useful not only because they can help assessing the coverage area of a receiver (important for an operator or network owner) but also because they show the importance of having a transmit power quantized more accurately for the highest values. Indeed, using different quantization steps (nonuniform quantization) for the transmit power (always in dB) should help removing the intermediate dead zones appearing in Figure 7.

5.2.2. Image $I_{(2,1)}$. This time we consider the point of view of transmitter 2 vis-a-vis the position of transmitter 1, where the noise power σ_i^2 is -92.2 dBm. Not surprisingly Figure 8 shows that when transmitter 1 is too close to receiver 2 Property 3 cannot be verified for transmitter 2. This explains the gray area and the white area on the left. Here again, an a priori non-trivial effect appears. There is a small white

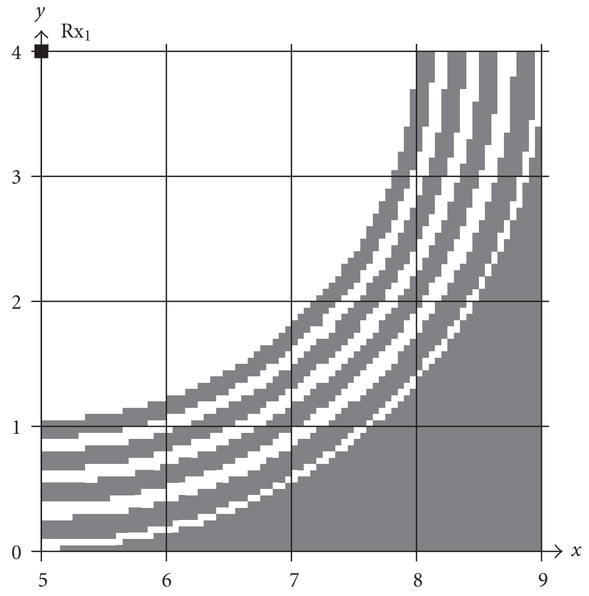

FIGURE 7: $I_{(1,3)}$: showing the influence of the discrete nature of transmit power on connectivity between Tx$_1$ and Rx$_1$ in the sense of Property 1; $(\Delta a, \Delta b) = (0.0625\,\text{m}, 0.0625\,\text{m})$.

area mixed with the gray area in which Property 3 is true. Our interpretation is as follows. When Tx$_1$ is close to Rx$_1$, Tx$_1$ becomes less aggressive in terms of transmit power since its objective can be easily reached, which means that Rx$_2$ receives less interference. But only half of the "disk" is white because in the other half, Tx$_1$ seems to be too close to Rx$_2$ to allow Tx$_2$ and Rx$_2$ to be connected. The existence of such an area is interesting since it shows that designing sensing algorithms based on the distance only can have undesirable effects. Indeed, if Tx$_2$ implements such an algorithm based on the distance between itself and Tx$_1$, it appears that the influence of Tx$_1$ is stronger when it is close to Rx$_1$ than what it is when it is close to Tx$_2$. This shows the type of implications of our results in terms of design. Note that this type of algorithms is implicitly assumed in the wireless game theory literature using best-response or learning-type algorithms.

Finally, Figure 9 shows how players interact with each other by choosing their transmit power over time. In this example, transmitter Tx$_1$ tries to establish and maintain a communication with its receiver Rx$_1$, while Tx$_2$ initiates short communications. In this example, Tx$_2$ is rather far from its receiver, so that it is allowed to turn its power to its maximal value. In order to maintain the communication, Tx$_1$ also has to increase its power. This goes smoothly for the first communication initiated by Tx$_2$, but communication of Tx$_1$ is broken at the second time, because Tx$_1$, that is close to its receiver, had set its power to its minimal value and could not "react" in time to set its SINR back to a reasonable value. This both illustrates the sequence of actions associated with the winning strategies of the transmitters and shows that the proposed setup allows one to accommodate asynchronous transmitters.

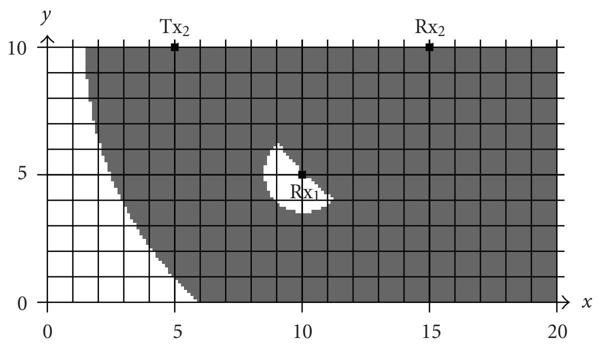

FIGURE 8: $I_{(2,1)}$: Influence of the position of Tx$_1$ on connectivity between Tx$_2$ and Rx$_2$ in the sense of Property 3.

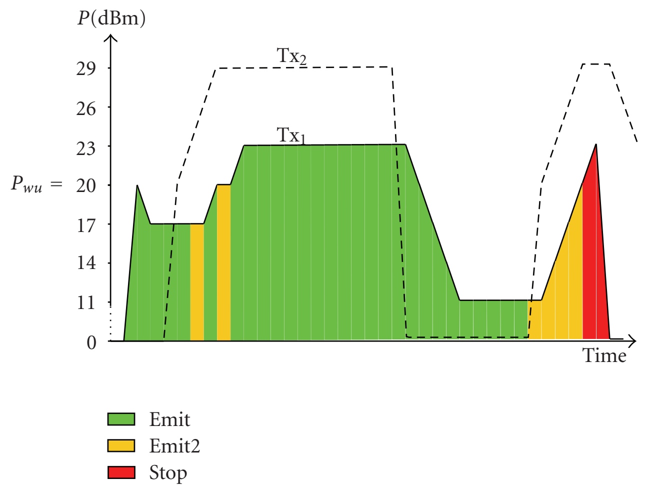

FIGURE 9: Transmit power versus time for the two users.

5.3. Towards an Implementable Distributed Power Control Algorithm. A natural and relevant issue is to assess the gap between the power control procedure and an algorithm implementable in a real terminal (a laptop, a mobile phone, a robot, a sensor, etc.). Here we do not pretend that the proposed procedure is ready to be implemented in a real terminal but we still want to provide some elements towards reaching this objective. Consider transmitter i. There are two steps. The first step is performed off-line by a computation center (which can be different from the transmitter but not necessarily). The second one is always performed online at the transmitter.

First Step. Define a off-line function for which the output is

(i) either a positive answer (yes or 1) associated with a nonzero sequence of functions (as defined in (10)),

(ii) or a negative answer (no or 0) associated with the zero sequence (connection failure),

and the inputs are the (quantized) channel gains $(|h_{ij}|)_{j\neq i}$), the (quantized) noise level σ_i^2, the maximum number of transmitters $K_{\max}$ (already discrete), the sets of transmit powers $\mathcal{P}_j$, $j \in \{1,\ldots,K_{\max}$ (already discrete), and $\mathcal{F}_q$ the logic formulae (written with the syntax of Uppaal-TiGA), where $q \in \{1,2,\ldots,Q\}$ is the formula index ($Q = 3$ in our case), translating the properties to be verified. For every possible 5-tuple $((|h_{ij}|)_j, \sigma_i^2, K_{\max}, (\mathcal{P}_j)_j, (\mathcal{F}_q)_q)$ the model checker is used to provide the value of the output of the function. The corresponding lookup table is stored in the transmitter memory.

Second Step. Define an on-line function for which the output is a sequence of functions and the inputs are the channel gains $(|h_{ij}|)_{j\neq i}$. Every block duration, this function is run (processing) and the look-up table (memory) gives the terminal the sequence of power control functions to be used (see (10)). Within a block, the transmitter plays its actions (transmit power levels) according to this sequence of functions.

6. Summarizing and Concluding Remarks

In comparison with existing works on distributed power control, we have made two key choices, leading to a different methodology to analyze the problem. Instead of assuming that every transmitter does what is best for itself we assume that, from a given transmitter point of view, the other transmitter can form a coalition aiming at choosing the worst environment for the considered transmitter. This worst-case assumption allows one to cope with some assumptions on the behavior of the other transmitters, which can be invalid in certain scenarios like in heterogeneous networks or when transmitters does not act rationally. Although pessimistic, it also allows the engineer to design a transmitter-receiver pair able to operate successfully under any feasible network configuration. The second important choice made in this paper is that the existence of the desired network state (namely verifying some properties for each transmitter) is not proved mathematically as it usually the case for Nash equilibria; one of the motivations for this is that this problem was non-trivial especially because we have considered the practical assumption of discrete transmit power and considered the dynamic aspect of the problem. Rather, we propose another approach. This approach consists in modeling the problem as a timed game between a given transmitter and its environment and ask a model checker for a rigorous answer to whether or not there exists a winning strategy (in the sense of certain desired properties for this player). Note that model checkers not only provide the answer to this question but also a winning strategy when it exists. In a concrete manner, this shows that it is possible to implement a certain mapping between a set of possible channel states and a set of winning strategies. At last, we have illustrated our approach by exploiting the Uppaal-TiGA model checker and shown how non-trivial verification results could be obtained.

In view of the obtained results, the proposed approach is quite general and seems to be promising. By general, we mean that it does not only apply to interference channels but also to any multiuser channel. Of course, the authors made some assumptions which can be discussed and which opens the possibility of extending the present work. For example, in addition to the extensions proposed previously: (a) one could try to remove the assumption of the worst behavior assumption for the environment. Indeed the price to be paid for removing the rationality assumption is that the influence of the other transmitters is overestimated. In a non-cooperative network, the transmitters will not always be able to form a coalition to fight against a given transmitter. In particular, this means that if the Nash equilibrium is effectively the most appropriate solution concept, the proposed approach can be adapted to prove the existence and uniqueness of an NE when proving a fixed-point theorem leads to a failure. The extension of our work to the Nash equilibrium solution concept would be an interesting and non-trivial extension since the protagonist-antagonist approach used in Uppaal-TiGA does not hold anymore; (b) the framework of timed games allows one to deal with real-time applications while we have focused here on the case of delay-tolerant applications; (c) since one can treat the case of dynamic games, it would be very interesting to compare the utilities obtained by Uppaal-TiGA and compare with individually rational levels allowed by equilibria in repeated games; (d) considering a constraint on the average transmit power and a finite energy constraint would also be a very practical feature to be included. The energy constraint should change the transmitters' behaviors, which is *a priori* not that easy to be predicted.

Appendices

A. Derivation of the Outage Probability

The SINR of user 1 at time τ is given by

$$\mathrm{SINR}_1(\tau) = \gamma_1(\tau) = \frac{|h_{11}|^2\overline{p}_1}{\sigma_1^2 + |h_{21}|^2\overline{p}_2}. \tag{A.1}$$

At time $\tau+1$ one new transmitter arrives, transmitter 1 observes its new SINR but is only able to react at time $\tau+2$. We precisely want that at least at time $\tau+1$, while transmitter 1 is "surprised", its communication is not broken that is, its SINR does not go below a certain threshold, say γ_-. At time $\tau+1$ the SINR of user 1 is

$$\mathrm{SINR}_1(\tau+1) = \gamma_1(\tau+1) = \frac{|h_{11}|^2\overline{p}_1}{\sigma_1^2 + |h_{21}|^2\overline{p}_2 + |h_{31}|^2 p_3(\tau+1)}. \tag{A.2}$$

The probability that SINR of player 1 falls, at time $\tau+1$, below γ_- given that it was equal to $\gamma_{\min}$ at time τ is expressed by

$$\begin{aligned}
&\Pr[\mathrm{SINR}_1(\tau+1) \le \gamma_- \mid \mathrm{SINR}_1(\tau) = \gamma_{\min}] \\
&\quad = \Pr\left[\frac{|h_{11}|^2\overline{p}_1}{\sigma_1^2 + |h_{21}|^2\overline{p}_2 + |h_{31}|^2 P_{\mathrm{wu}}} \le \gamma_- \mid \gamma_1(\tau) = \gamma_{\min}\right] \\
&\quad = \Pr\left[\gamma_1(\tau)\frac{1}{1+|h_{31}|^2 P_{\mathrm{wu}}/\left(\sigma_1^2 + |h_{21}|^2\overline{p}_2\right)}\right. \\
&\qquad\qquad \left. \le \gamma_- \mid \gamma_1(\tau) = \gamma_{\min}\right] \\
&\quad = \Pr\left[\gamma_{\min}\frac{1}{1+|h_{31}|^2 P_{\mathrm{wu}}/\left(\sigma_1^2 + |h_{21}|^2\overline{p}_2\right)} \le \gamma_-\right] \\
&\quad = \Pr\left[|h_{31}|^2 \ge \alpha\left(\frac{\gamma_{\min}}{\gamma_-} - 1\right)\right],
\end{aligned} \tag{A.3}$$

where $\alpha = (\sigma_1^2 + |h_{21}|^2\overline{p}_2)/P_{\mathrm{wu}}$. Here we implicitly assumed that randomness comes from the channel gain between transmitter 3 and the corresponding receiver.

B. Timed Automata

Definition 1. A *timed automaton* $\mathcal{A} = (L, X, \Sigma, E, \mathcal{I}, \mathcal{L})$ has the following components: (i) L is a finite set of *locations*, (ii) X is a finite set of *clocks*, (iii) Σ is a finite set of *actions*, (iv) $E \subseteq L \times \Sigma \times \mathcal{G} \times 2^X \times L$ is a finite set of *edges*, (v) $\mathcal{I} : L \to \mathcal{G}$ assigns an *invariant* to each location, and (vi) $\mathcal{L} : L \to 2^{\mathrm{AP}}$ is the *labeling* function.

The semantics of a timed automaton $\mathcal{A}$ is given by a labeled transition system $T_{\mathcal{A}}$. A *state* of $\mathcal{A}$ is a pair $q = (l, \nu)$ such that $l \in L$ and $\nu \vDash \mathcal{I}(l)$. We let Q denote the set of all states.

We distinguish two kinds of transitions: *time transitions* and *switch-transitions*:

(i) Given $q = (l, \nu)$ and $q' = (l', \nu')$ two states of $\mathcal{A}$, there is a *time-transition* in $\mathcal{A}$ between q and q' if there exists $\tau \in \mathbb{R}^+$ such that $l = l'$, $\nu' = \nu + \tau$ and $\nu + \tau' \vDash \mathcal{I}(l)$ for any τ', $0 \le \tau' \le \tau$. We denote this transition by $q \xrightarrow{\tau} q'$.

(ii) Given $q = (l, \nu)$ and $q' = (l', \nu')$ two states of $\mathcal{A}$, there is a *switch transition* in $\mathcal{A}$ between q and q' if there exists $e = (l, a, g, Y, l') \in E$ such that $\nu \vDash g$ and ν' is given by

$$\nu_i' = \begin{cases} 0, & \text{if } x_i \in Y, \\ \nu_i, & \text{if } x_i \notin Y. \end{cases} \tag{B.1}$$

We denote this switch-transition by $q \xrightarrow{e} q'$. To emphasize on the action a, we also use notation $q \xrightarrow{a} q'$ in this case, we use the notation Action$(e) = a$.

We now define the (labeled) transition system $T_{\mathcal{A}}$.

Definition 2. Given a timed automaton $\mathcal{A}$, the *(labeled) transition system associated with* $\mathcal{A}$ is given by $T_{\mathcal{A}} = (Q, \Sigma \cup^{+}, \rightarrow)$ where the transition relation is given by

$$\longrightarrow \; = \bigcup_{\tau \in \mathbb{R}^{+}} \xrightarrow{\tau} \cup \bigcup_{e \in E} \xrightarrow{e}. \tag{B.2}$$

Let $\mathcal{A} = (L, X, \Sigma, E, \mathcal{I}, \mathcal{L})$ be a timed automaton, q_1, q_2 and q_3 be three states of $\mathcal{A}$. If $q_1 \xrightarrow{\tau} q_2$, for some $\tau \in \mathbb{R}^{+}$, and $q_2 \xrightarrow{e} q_3$, for some $e \in E$, we shortly denote $q_1 \xrightarrow{\tau \cdot e}$ this sequence of two transitions. A finite or infinite run ρ is sequence of alternating transitions of the form

$$\rho = q_1 \xrightarrow{\tau_1 \cdot e_1} q_2 \xrightarrow{\tau_2 \cdot e_2} \cdots \xrightarrow{\tau_k \cdot e_k} q_{k+1} \cdots. \tag{B.3}$$

We denote by $\mathrm{Run}_{\mathcal{A}}$ (resp., $\mathrm{Run}^{f}_{\mathcal{A}}$) the set of runs (resp., finite runs) of $\mathcal{A}$.

C. Timed Games

Definition 1. Let $\mathcal{A} = (L, X, \Sigma, E, \mathcal{I})$ be a timed automaton. We say that $\mathcal{A}$ is a *timed game*, if the set of action Σ contains a particular action denoted u.

In this context, the transitions labeled with u are called the *uncontrollable transitions*. They represent the set of actions available to the environment. The other ones are called the *controlled transitions*. We denote by Σ_c the set of actions $\Sigma \setminus \{u\}$.

Before giving the semantics of timed games, let us first explain it intuitively.

Let $\mathcal{A} = (L, X, \Sigma, E, \mathcal{I})$ be a timed game. The game is played by two players, *Player* 1 (the *controller*) and *Player* 2 (the *environment*). At any state q, Player 1 picks a time τ and an action $a \in \Sigma_c$ such that there is a transition $q \xrightarrow{\tau \cdot e} q'$ with $\mathrm{Action}(e) = a$. Player 2 has two choices:

(i) either he can decide to wait for time τ and execute a transition $q \xrightarrow{\tau \cdot e} q'$ proposed by Player 1,

(ii) or he can wait for time τ', $0 \leq \tau' \leq \tau$, and execute a transition $q \xrightarrow{\tau' \cdot e'} q''$ with $\mathrm{Action}(e') = u$.

The game then evolves to a new state (according to the choice of Player 2) and the two players proceed to play as before.

Notice that, in the definition of a timed game, it is implicitly supposed that Player 1 can always formulate a choice (τ, a) in any reachable state q of the game.

We will now formalize the semantics through the concept of *strategy*.

Definition 2. A *(Player* 1*) strategy* is a function

$$\lambda : \mathrm{Run}^{f}_{\mathcal{A}} \longmapsto \mathbb{R}^{+} \times \Sigma_c. \tag{C.1}$$

Before defining the notion of a winning strategy, we need to define several other notions. We say that a run ρ is *maximal* if it is either infinite or ending in a deadlock. An objective Ω of a timed game is a subset of the runs of $\mathcal{A}$. Let ρ be a run of the form $q_1 \xrightarrow{\tau_1 \cdot e_1} q_2 \xrightarrow{\tau_2 \cdot e_2} \cdots \xrightarrow{\tau_k \cdot e_k} q_{k+1} \cdots$, we denote by ρ_i the prefix of ρ ending in q_i. Given a strategy λ and a run ρ, we say that ρ is *played according to* λ for every i, if $\lambda(\rho_i) = (\tau'_i, a_i)$, then either $\tau_i = \tau'_i$ and $\mathrm{Action}(e_i) = a_i$, or $\tau_i \leq \tau'_i$ and $\mathrm{Action}(e_i) = u$. We denote by $\mathrm{Outcome}(\rho, \lambda)$ the set of *maximal* runs extending ρ and played according to λ. Given a state q, a strategy λ and an objective Ω, we say that the strategy λ *is winning for the objective* Ω *from* q if $\mathrm{Outcome}(q, \lambda) \subseteq \Omega$.

Acknowledgment

The work reported herein is partly supported by "Institut Farman" (ENS Cachan) and by CNRS, in the framework of the TOAST project.

References

[1] J. Mitola III and G. Q. Maguire Jr., "Cognitive radio: making software radios more personal," *IEEE Personal Communications*, vol. 6, no. 4, pp. 13–18, 1999.

[2] Q. Zhao and B. M. Sadler, "A survey of dynamic spectrum access," *IEEE Signal Processing Magazine*, vol. 24, no. 3, pp. 79–89, 2007.

[3] W. Yu, G. Ginis, and J. M. Cioffi, "Distributed multiuser power control for digital subscriber lines," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 5, pp. 1105–1115, 2002.

[4] L. Lai and H. E. El Gamal, "The water-filling game in fading multiple-access channels," *IEEE Transactions on Information Theory*, vol. 54, no. 5, pp. 2110–2122, 2008.

[5] E.-V. Belmega, S. Lasaulce, and M. Debbah, "Power allocation games for mimo multiple access channels with coordination," *IEEE Transactions on Wireless Communications*, vol. 8, no. 6, pp. 3182–3192, 2009.

[6] D. Goodman and N. Mandayam, "Power control for wireless data," *IEEE Personal Communications*, vol. 7, no. 2, pp. 48–54, 2000.

[7] F. Meshkati, M. Chiang, H. V. Poor, and S. C. Schwartz, "A game-theoretic approach to energy-efficient power control in multicarrier CDMA systems," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 6, pp. 1115–1129, 2006.

[8] S. Lasaulce, Y. Hayel, R. El Azouzi, and M. Debbah, "Introducing hierarchy in energy games," *IEEE Transactions on Wireless Communications*, vol. 8, no. 7, pp. 3833–3834, 2009.

[9] S. Lasaulce, M. Debbah, and E. Altman, "Methodologies for analyzing equilibria in wireless games," *IEEE Signal Processing Magazine*, vol. 26, no. 5, pp. 41–52, 2009.

[10] O. Maler, A. Pnueli, and J. Sifakis, "On the synthesis of discrete controllers for timed systems," in *Proceedings of the 12th Annual Symposium on Theoretical Aspects of Computer Science (STACS '95)*, vol. 900, pp. 229–242, Munich, Germany, March 1995.

[11] E. Emerson, "Temporal and modal logic," in *Handbook of Theoretical Computer Science*, pp. 995–1072, MIT Press, Cambridge, Mass, USA, 1990.

[12] E. M. Clarke, O. Grumberg, and D. A. Peled, *Model Checking*, MIT Press, Cambridge, Mass, USA, 1999.

[13] Y. Xing and R. Chandramouli, "Stochastic learning solution for distributed discrete power control game in wireless data networks," *IEEE/ACM Transactions on Networking*, vol. 16, no. 4, pp. 932–944, 2008.

[14] G. Behrmann, A. Cougnard, A. David, E. Fleury, K. G. Larsen, and D. Lime, "UPPAAL-tiga: time for playing games!," in

Proceedings of the 19th International Conference on Computer Aided Verification (CAV '07), vol. 4590, pp. 121–125, Berlin, Germany, 2007.

[15] A. B. Carleial, "Interference channels," *IEEE Transactions on Information Theory*, vol. 24, no. 1, pp. 60–70, 1978.

[16] G. Scutari, S. Barbarossa, and D. P. Palomar, "Potential games: a framework for vector power control problems with coupled constraints," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '06)*, vol. 4, pp. 241–244, Toulouse, France, May 2006.

[17] R. Alur and D. L. Dill, "A theory of timed automata," *Theoretical Computer Science*, vol. 126, no. 2, pp. 183–235, 1994.